算法二：多源数据挖掘的注释

多源数据挖掘的加权融合方法是与第一篇论文的方法类似，但是，解决拥有多源数据的用户的多层次应用需求问题具有更大的挑战性。

1998 年兴起的多源数据挖掘致力于关联模式发现的方法和技术，符合最初以研究不同类型数据集合中的关联模式挖掘方法为主要目标的数据挖掘领域发展需求。不过，这种多源数据挖掘必须要解决：处理来自多源的数据要面对数据集成所带来的巨大挑战，数据集成导致一些有用模式的消亡，集团公司的多层次挖掘需求，原始数据的私有性保护等问题。这是极为困难的又无法绕开的艰巨任务。针对这个发现，这一篇论文提出了局部模式分析方法，避免了多源的原始数据直接参与运算，实现挖掘复杂性的控制，发现新的全局有用模式，保护原始数据的私有性利益。

基于局部模式分析的多源数据挖掘的具体模式融合过程如下：在任何节点（公司或者子公司）进行模式挖掘，首先将其子节点（如果有）的局部模式作为融合算法的输入，然后，加上从本节点（如果有）的数据中挖掘出的一些模式作为融合算法的输入，最后，通过融合算法有效地综合各种局部模式后产生的模式就是这一层次节点需求的全局模式。这种局部模式分析方法与人类对多源数据的智能处理机制相吻合，可以同时支持全局和局部决策应用，形成了多源多层次模式发现的新机制。与数据集成挖掘方法相比，局部模式分析方法更易于理解与实现，从根本上改变了候选运算数据，减少了存储和计算量，实现了存储和计算难度的控制，保护了原始数据的私有性利益。特别是，局部模式分析能够控制子节点中数据的异质与异构性的传播，完成多源数据挖掘的主体任务。

另外，这一章设计了基于局部模式分析的多源数据中高票模式融合挖掘算法，可以通过调整权重值来扩展该算法，以便发现局部突出模式、例外模式、或者趋势模式（图 2 中的 A、B 和 C 三类模式的具体表现），解决了数据集成后导致这些有用模式消亡的问题。的确，数据集成会导致一些有用模式消亡，下面一个类似的例子可以形象地说明这一点。例如，设 X 和 Y 是争夺网球比赛决赛（3 局 2 胜制）的两名选手，且三局的比分分别是 6：4；0：6；6：4。按照比赛规则，**X 选手取得 2：1 的胜利**（图 2 中的 A 类模式），是冠军。若将这三局比分看成三个数据库的数据，按照数据集成挖掘方法，这三个库中的数据被集成后挖掘到的结果是：**Y 选手取得 14：12 的胜利**，应该获得冠军，但这与赛制规定相左。这是因为将三个库中的数据集成在一起后就不能体现出 2：1 这个三局两胜的局部模式的分布信息，即，数据集成会导致了这类重要信息的消亡。

另外，图 2 中 B 和 C 两类模式通常在数据集成后统计出的支持度很低，也是无法识别出来的。在实际应用中，上面这些隐藏在各个数据源的局部模式是集团公司及其子公司进行决策的最有用的全局信息之一，可以通过扩展这篇论文中的挖掘算法来发现它们，有兴趣的读者可以下载阅读文献[2-4]。

参考文献

[2]. Xindong Wu, Chengqi Zhang and Shichao Zhang. Database Classification for Multi-Database Mining. Information Systems, 30(2005): 71-88.

[3]. Shichao Zhang, Chengqi Zhang and Jeffrey Xu Yu. An Efficient Strategy for Mining Exceptions in Multi-databases. Information Sciences, 165/1-2 (2004): 1-20.

[4]. Chengqi Zhang, Meiling Liu, Wenlong Nie and Shichao Zhang. Identifying Global Exceptional Patterns in Multi-database Mining. IEEE Computational Intelligence Bulletin, Vol. 3 1(2004): 19-24.

Synthesizing High-Frequency Rules from Different Data Sources

Xindong Wu, *Senior Member*, *IEEE*, and Shichao Zhang

Abstract—Many large organizations have multiple data sources, such as different branches of an interstate company. While putting all data together from different sources might amass a huge database for centralized processing, mining association rules at different data sources and forwarding the rules (rather than the original raw data) to the centralized company headquarter provides a feasible way to deal with multiple data source problems. In the meanwhile, the association rules at each data source may be required for that data source in the first instance, so association analysis at each data source is also important and useful. However, the forwarded rules from different data sources may be too many for the centralized company headquarter to use. This paper presents a weighting model for synthesizing high-frequency association rules from different data sources. There are two reasons to focus on high-frequency rules. First, a centralized company headquarter is interested in high-frequency rules because they are supported by most of its branches for corporate profitability. Second, high-frequency rules have larger chances to become valid rules in the union of all data sources. In order to extract high-frequency rules efficiently, a procedure of rule selection is also constructed to enhance the weighting model by coping with low-frequency rules. Experimental results show that our proposed weighting model is efficient and effective.

Index Terms—Large databases, multiple data sources, association rules, synthesizing, weighting, rule selection.

1 Introduction

With the advances in data gathering, storage, and distribution technologies, all companies, large and small, are facing the inevitable challenges brought about by information technological developments. The challenges create not only risks but also opportunities. To *minimize* risks and *maximize* potential benefits, how to use their information effectively and efficiently has become very important when companies make competitive decisions. *Data mining* as an analysis of information, which can extract useful patterns from large databases, has been widely applied to analyze data for decision makers. Accordingly, much work has recently focused on mining useful information from databases.

However, traditional data mining techniques are insufficient for large organizations that have multiple data sources from different branches. In particular, little work has been reported on *postmining* that gathers, analyzes, and synthesizes the patterns discovered from different data sources. While putting all data together from different sources might amass a huge database for centralized processing, mining association rules at different data sources and forwarding the rules (rather than the original raw data) to the centralized company headquarter provides a feasible way to deal with multiple data source problems. In the meanwhile, the association rules at each data source may be required for that data source in the first instance, so association analysis at each data source is also important and useful. However, the number of the forwarded rules may be so large that browsing the rule set and finding interesting rules can be rather difficult for centralized company headquarters. Therefore, it is hard to identify which of the forwarded rules (including different and the same ones) are really useful at the company level. For these reasons, we present a *weighting model* in this paper for *synthesizing* the rules discovered from different data sources, such as different branches of a large company. The proposed approach is particularly utilizable to interstate companies when their data sources have been mined at their branches. For convenience, our work focuses on synthesizing association rules.

The rest of the paper is organized as follows: We begin with a problem statement and related work in Section 2. In Section 3, a synthesizing model by weighting is presented. A relative synthesizing model for association rules from unknown data sources is described in Section 4. In Section 5, we demonstrate the effectiveness and efficiency of our weighting model in experiments.

2 Problem Statement and Related Work

In this section, we formulate the problem, outline our approach, and review related work.

2.1 Synthesizing Rules from Different Data Sources: The Problem

To mine transaction databases for large organizations that have multiple data sources, there are two possible ways: 1) putting all data together from different sources to amass a centralized database for centralized processing, possibly using parallel and distributed mining techniques, and 2) reusing all promising rules discovered from different

- *X. Wu is with the Department of Computer Science, The University of Vermont, Burlington, VT 05404. E-mail: xindong@computer.org.*
- *S. Zhang is with the School of Computing, Guangxi Normal University, the Faculty of Information Technology, University of Technology, Sydney, NSW 2007, Australia. E-mail: zhangsc@it.uts.edu.au.*

Manuscript received 31 May 2000; revised 19 Mar. 2001; accepted 1 June 2001.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 112211.

data sources to form a large set of rules and then searching for valid rules that are useful at the organization level.

Putting all data together from different sources to amass a centralized database has two significant limitations.

1. It might be unrealistic to collect data from different data sources for centralized processing because of the size of the data. For example, different branches of Wal-Mart collect 20 million transactions per day. This is more than the rate at which data can be feasibly collected and analyzed using today's computing power.
2. Because of data privacy and related issues, it is possible that some data sources of an organization may share their association rules but not their original databases.

If the above limitations do not apply with some organizations, we need efficient techniques, such as sampling and parallel and distributed mining algorithms, to deal with the centralized databases thus collected. However, sampling models heavily depend on that the transactions of a given database are randomly appended into the database in order to hold the binomial distribution. Therefore, mining association rules upon paralleling (MARP) [4], [7], [10], [23], [24], [31], [34], [36], which employs hardware technology such as parallel machines to implement concurrent data mining algorithms, is a popular choice. Existing MARP developments endeavor to scale up data mining algorithms by changing existing sequential techniques into parallel versions. These algorithms are effective and efficient, and have played an important role in mining very large databases. However, in addition to the above two limitations, there are two more limitations with MARP for data mining with different data sources.

3. MARP does not make use of local rules at different data sources, and does not generate these local rules either. In real-world applications, these local rules are useful for the local data sources, and would have to be generated in the first instance.
4. Parallel data mining algorithms require more computing resources (such as massively parallel machines) and additional software to distribute components of parallel algorithms among processors of parallel machines, and most importantly, it is not always possible to apply MARP to existing data mining algorithms. Some data mining algorithms are sequential in nature and cannot make use of parallel hardware.

Reusing all promising rules discovered from different data sources is a better way, because the local rules discovered at each data source are useful for the local branch of a large organization in the first instance. However, to reuse the local rules and select from them, we must develop a method to 1) determine the valid rules for the overall organization that would have been discovered from the amassed database, and 2) reduce the size of the candidate rules from different data sources. The following problems arise: 1) any rule from a data source can make a possible contribution to a valid rule for the overall organization, and 2) the amount of promising rules from different data sources can be very large before we determine which ones are of interest.

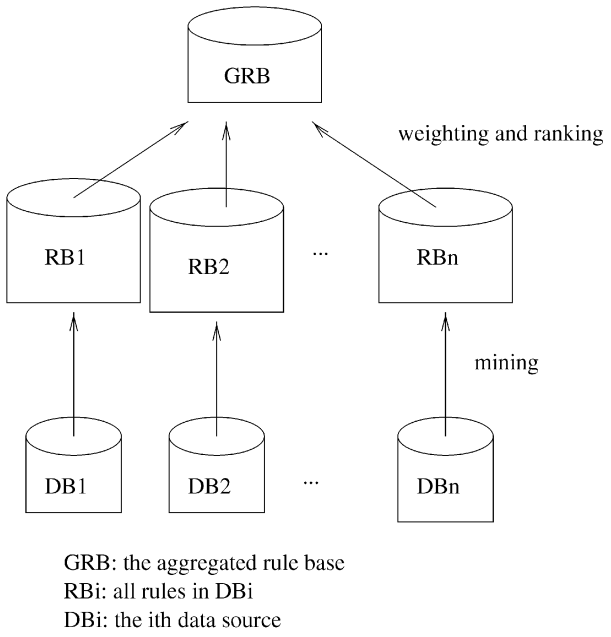

Fig. 1. The synthesizing model.

Based on the above analysis, the problem for our research can be formulated as follows:

Given n data sources from a large organization, we are interested in 1) mining each of these data sources for local rules for each data source, and also 2) synthesizing these local rules to find valid rules for the overall organization that would have been discovered from the union of all these data sources.

There are various existing data mining algorithms that can be used to discover local rules for each data source, including MARP algorithms mentioned above. This paper focuses on synthesizing local rules to find valid rules for the overall organization.

2.2 Our Approach

When association rules are discovered from different data sources of a large company, Fig. 1 illustrates our synthesizing model by weighting, where a database $\mathrm{DB}i(i = 1, \ldots, n)$ is one of the data sources.

Weighting is a common way to gather, analyze, and synthesize information from different sources in scientific research and applications. For example, consider a diagnosis in a hospital. Let A be a patient, and d_1, d_2, d_3, d_4, and d_5 be five medical experts in the hospital with authority weights w_1, w_2, w_3, w_4, and w_5, respectively. After diagnosing, the patient is judged to be with one of the four possible diseases: s_1, s_2, s_3, and s_4. To make a final conclusion, it needs to synthesize the diagnoses by these experts. Assume b_{ij} is the belief of the patient with the jth disease given by expert d_i $(i = 1, 2, 3, 4, 5;$ $j = 1, 2, 3, 4)$. Then, the belief of the patient with disease s_j is synthesized as

$$p_j = \sum_{i=1}^{5} w_i * b_{ij},$$

where $j = 1, 2, 3, 4$. According to the synthesis, we can rank diseases s_1, s_2, s_3, and s_4 by p_1, p_2, p_3, p_4, and take the disease with the highest rank as the final result.[1]

To synthesize association rules from different data sources, the above weighting process is adopted in this paper. In the above diagnosis, the synthesis of experts' opinions is generally straightforward, and the key problem is to allocate a *reasonable weight* to each expert. We will study how to determine proper weights for our tasks in this paper.

Our goal in this paper is to extract *high-frequency rules* from different data sources of a large company. The frequency of a rule is the number of data sources that contain this rule in their respective association rules. A rule is called a high-frequency rule if the rule is supported or voted by most of the data sources. There are two main reasons to focus on high-frequency rules. First, a company headquarter is interested in the rules supported by most of its branches for corporate profitability. Second, high-frequency rules have larger chances to become valid rules in the union of all data sources than low-frequency rules do.

Since putting all association rules together from different data sources of a large company might also amass a huge rule set, we will design an efficient algorithm to improve the proposed weighting model. The high-frequency rules from different data sources will be taken as relevant rules and lower-frequency rules as irrelevant rules. In this way, we can cope with lower-frequency rules before rules from different data sources are synthesized. Consequently, a rule selection procedure will be constructed to enhance the assignment of weights in Section 3.

When some associations come from unknown sources, we will design a relative synthesizing model by clustering in Section 4.

2.3 Related Work

Data mining, also known as knowledge discovery in databases, aims at the discovery of useful information from large collections of data [2], [8], [38]. The discovered knowledge can be rules describing properties of the data, frequently occurring patterns, clusterings of the objects in the database, and so on, which can be used to support various intelligent activities, such as decision-making, planning, and problem-solving. The data mining model adopted in this paper for association rules is the support-confidence framework established by Agrawal et al. [2].

Let $I = \{i_1, i_2, \cdots, i_N\}$ be a set of N distinct literals called *items*, and D a set of transactions over I. Each transaction contains a set of items $i_1, i_2, \cdots, i_k \in I$. A transaction has an associated unique identifier called TID. An *association rule* is an implication of the form $A \rightarrow B$, where $A, B \subset I$, and $A \cap B = \emptyset$. A is called the *antecedent* of the rule, and B is called the *consequent*.

A set of items (such as the antecedent or the consequent of a rule) is called an *itemset*. Each itemset has an associated statistical measure called *support*, denoted as *supp*. For an itemset $A \subset I$, $supp(A) = s$, if the fraction of transactions in D containing A equals to s. A rule $A \rightarrow B$ has a measure of strength called *confidence* (denoted as *conf*) which is defined as the ratio $supp(A \cup B)/supp(A)$.

The problem of mining association rules is to generate all rules $A \rightarrow B$ that have both support and confidence greater than or equal to some user specified thresholds, called minimum support (*minsupp*) and minimum confidence (*minconf*), respectively. For regular associations:

$$supp(A \cup B) \geq minsupp,$$
$$conf(A \rightarrow B) = \frac{supp(A \cup B)}{supp(A)} \geq minconf.$$

Association analysis can be decomposed into the following two subproblems.

1. Generate all itemsets that have support greater than or equal to the user-specified minimum support.
2. Generate all rules that have their minimum confidence in the following naive way: For every large itemset X and any $B \subset X$, let $A = X - B$. If the rule $A \rightarrow B$ has the minimum confidence (or $supp(X)/supp(A) \geq minconf$), then it is a valid rule.

Example 1. Let $T_1 = \{A, B, D\}$, $T_2 = \{A, B, D\}$, $T_3 = \{B, C, D\}$, $T_4 = \{B, C, D\}$, and $T_5 = \{A, B\}$ be five transactions in a database, and the minimum support and minimum confidence be 0.6 and 0.85, respectively. Then, the itemsets with the minimum support are the following: $\{A\}$, $\{B\}$, $\{D\}$, $\{A, B\}$, and $\{B, D\}$, and the valid rules are $A \rightarrow B$ and $D \rightarrow B$.

Association analysis from large databases has received much attention recently [2], [38]. Discovering association rules is to generate all rules in a given database, and efficient algorithms for computing them can be found in [3], [22], [32], [15]. Other measures on uncertainty of association rules include strongly collective itemsets [1], and the chi-squared test model [5], [38]. In [25], Piatetsky-Shapiro proposes that a rule $A \rightarrow B$ is not interesting if $support(A \rightarrow B) \approx support(A) \times support(B)$. In order to implement interesting association analysis, a wide range of problems have been investigated over such diverse topics as models for discovering generalized association rules [1], [34], [38], [40], [35], measurements of interestingness [1], [2], [5], [37], computing large itemsets online [16], optimizing support association rules [30], data mining based on feature selection [17], [20], [33], and parallel data mining for association rules [4], [6], [7], [10], [23], [24], [26], [27], [34].

Recently, an FP-tree-based frequent patterns mining method was developed by Han et al. [15]. This technique leads to three benefits: 1) it can compress a large database into a highly condensed, much smaller data structure, so as to avoid the costly, repeated database scans, 2) FP-tree-based mining adopts a pattern fragment growth method to avoid the costly generation of a large number of candidate sets, and 3) the partitioning-based divide-and-conquer method can reduce the size of the subsequent conditional pattern bases and conditional FP-trees. Their experiments

1. It is not realistic to rediagnose the patient and aim for consistent results at each diagnosis-step by the five experts. This is because some steps such as operations cannot be repeated and a rediagnosis can be very expensive.

show that this technique is more efficient than the Apriori algorithm [3].

Also, an OPUS-based algorithm [41] has been reported by Webb to reduce the searched space by focusing association rules mining with which the searched space consists of all possible items and itemsets in a database. The Apriori algorithm uses a two-step technique to identify association rules, and a search space in Apriori consists of all items and possible itemsets. For example, if we have a market basket database from a grocery store, consisting of n baskets with 1,000 items, then there are $2^{1,000}$ possible itemsets to be counted in the database. To overcome this limitation, Webb proposed to directly (one-step) search for association rules. His experiments have shown that both the searched space and running time can be cut down.

However, existing work [2], [5], [10], [11], [13], [18] has focused on mining frequent itemsets in data sets, and few research efforts have been reported on postmining that gathers, analyzes, and synthesizes association rules from different data sources. Parallel and distributed data mining and metalearning are relevant topics, and are reviewed below, but they are significantly different from the problem we have formulated in Section 2.1.

2.3.1 Parallel and Distributed Data Mining

Due to the size of large databases and the amount of intensive computation involved in association analysis, parallel and distributed data mining has been a crucial mechanism for large-scale data mining applications. Existing research in this area has focused on the study of the degree of parallelism, synchronization, data locality issues, and optimization techniques for global association computation.

For example, Cheung et al. [9] proposed some strategies to leverage the skew of association patterns in a distributed database environment, and some optimizations to efficiently generate global frequent sets. Their main idea was to use local pruning for support count exchange to achieve efficient association analysis.

As discussed in Section 2.1, there are four limitations in applying parallel and distributed data mining techniques to postmining of association results from different data sources. However, parallel and distributed data mining can be combined with our synthesizing model by weighting for very large data mining applications. If each of the data sources is still large, we can apply a mining association rules upon paralleling (MARP) algorithm to discover the local associations from each data source and, then, synthesize these local associations by weighting.

2.3.2 Metalearning

Another related research effort is hierarchical metalearning [6], [26], [27], [28], [21] which has a similar goal of efficiently processing large amounts of data. Metalearning starts with a distributed database or partitions an original database into disjoint subsets, concurrently runs a learning algorithm (or different learning algorithms) on each of the subsets, and combines the predictions from classifiers learned from these subsets by recursively learning "combiner" and "arbiter" models in a bottom-up tree manner. The focus of metalearning is to combine the predictions of learned models from the partitioned data subsets in a parallel and distributed environment. However, unlike our synthesizing model by weighting, metalearning does not produce a global learning model from classifiers from different data subsets.

3 Synthesizing Rules by Weighting

When association rules are forwarded from different, known data sources in the branches of a large company, we present a weighting model to synthesize these rules in this section.

Let $D_1, D_2, \cdots, D_m$ be m different data sources from the branches of a large company of similar size[2] (see Fig. 1), and S_i the set of association rules from D_i ($i = 1, 2, \cdots, m$). For a given rule $X \rightarrow Y$, suppose w_1, w_2, $\cdots$, w_m are the weights (see Section 3.2) of $D_1, D_2, \cdots, D_m$, respectively, our synthesizing model is defined as follows:[3]

$$\begin{aligned} supp_w(X \cup Y) &= w_1 * supp_1(X \cup Y) + w_2 * supp_2(X \cup Y) \\ &\quad + \cdots + w_m * supp_m(X \cup Y), \\ conf_w(X \rightarrow Y) &= w_1 * conf_1(X \rightarrow Y) + w_2 * conf_2(X \rightarrow Y) \\ &\quad + \cdots + w_m * conf_m(X \rightarrow Y), \end{aligned}$$

where $supp_w(R)$ is the support of R after synthesizing, $conf_w(R)$ is the confidence of R after synthesizing, $supp_i(R)$ is the support of R in D_i, and $conf_i(R)$ is the confidence of R in D_i, $i = 1, 2, \cdots, m$.

The synthesis of rules in our model is generally straightforward once all weights are reasonably assigned. To assign weights, we first review Good's idea in [14] to determine weights in Section 3.1, and then our weight synthesizing model is designed with an example in Section 3.2. We also construct a procedure of rule selection in Section 3.3 to enhance the weighting model by coping with low-frequency rules.

3.1 Weight of Evidence

To allocate weights, Good defines the weight in favor of a hypothesis, H, provided by evidence, E, as follows [14]:

$$woe(H : E) = log\ \frac{O(H|E)}{O(H)},$$

where $O(x)$ stands for the odds of x. He thinks the $woe(H : E)$ is a concept "almost as important as that of probability itself." Good elucidates simple, natural desiderata for the

2. Throughout this paper, we assume that each database from each data source contains about the same amount of data. If the data sources are of clearly different size, we can preprocess them by splitting the larger ones (and merging the smaller ones if such a merger is possible) and make them of similar size. When merging the smaller data sources is not possible because data sharing is not allowed between different data sources, as one of the anonymous reviewers has suggested, we can either ignore them if their size is below a user specified threshold, or treat them as if they had a similar size as other data sources do.

3. In our approach, we use the same min-support ($minsupp$) and min-confidence ($minconf$) for convenience. If min-supports and min-confidences are different in different data sources, we can still select rules of interest in these data sources under the same min-support and min-confidence before we synthesize them. Therefore, the data sources do not need to be remined. In fact, we can deal with local rules without respect to the min-support and min-confidence because each data source has equal power to recommend its rules.

formalization of the notion of weight of evidence, including an "additive property:"

$$woe(H : E_1 \wedge E_2) = woe(H : E_1) + woe(H : E_2|E_1).$$

This property states that the weight in favor of a hypothesis provided by two pieces of evidence is equal to the weight provided by the first piece of evidence, plus the weight provided by the second piece of evidence, conditioned on our having previously observed the first. Starting from these desiderata, Good is able to show that, up to a constant factor, the weight of evidence must take the form given in the definition of weight. From the definition, it follows directly that:

$$log\ O(H|E) = log\ O(H) + woe(H : E).$$

That is, if we think on a log-odds scale, our final belief in a hypothesis is equal to our initial belief plus the weight of whatever evidence we are presented with. Log-odds are an attractive scale because weights accumulate additively; also because the entire range from $-\infty$ to $+\infty$ is used.

It is obvious that either log-odds or the weight of evidence can be used to synthesize forwarded association rules from different data sources. For simplicity, weights are generally normalized into interval $[0, 1]$ in the following account. Furthermore, Good's idea, for convenience, is emerged as that the weight of each rule is almost as important as that of its frequency in the original data sources and, therefore, we will use the frequency of a rule to evaluate the rule's weight.

3.2 Solving Weights of Data Sources

In order to synthesize association rules from different data sources in the branches of a company, we need to determine the weight for each data source. In our opinion, if all data sources are of similar size, the weight of each data source can be determined by the rules discovered from it. Let $D_1, D_2, \cdots, D_m$ be m different data sources in the branches of a company, S_i the set of association rules from D_i $(i = 1, 2, \cdots, m)$, and $S = \{S_1, S_2, \cdots, S_m\}$. According to Good's definition on weight, we can take *the frequency of a rule* R in S to assign R a weight w_R. In practice, a company headquarter would be interested in the rules that are supported or voted by most of its branches for corporate profitability. High-frequency rules have larger chances to become valid rules in the union of all data sources than low-frequency rules do. Hence, the more the number of data sources that support the same rule, the larger the weight of the rule should be.

In the meanwhile, the intersupport relation between a data source and its rules can be applied to assign the data source a weight. If a data source supports a larger number of high-frequency rules, the weight of the data source should also be higher. We now illustrate the above idea by an example.

Let $minsupp = 0.2$, $minconf = 0.3$, and the following rules be mined from three data sources.

1. S_1 the set of association rules from data source $D1$: $A \wedge B \rightarrow C$ with $supp = 0.4, conf = 0.72$; $A \rightarrow D$ with $supp = 0.3$, $conf = 0.64$; $B \rightarrow E$ with $supp = 0.34$, $conf = 0.7$;
2. S_2 the set of association rules from data source $D2$: $B \rightarrow C$ with $supp = 0.45$, $conf = 0.87$; $A \rightarrow D$ with $supp = 0.36$, $conf = 0.7$; $B \rightarrow E$ with $supp = 0.4$, $conf = 0.6$;
3. S_3 the set of association rules from data source $D3$: $A \wedge B \rightarrow C$ with $supp = 0.5, conf = 0.82$; $A \rightarrow D$ with $supp = 0.25, conf = 0.62$;

Assume $S' = \{S_1, S_2, S_3\}$. Then, there are a total of four rules in S':

- R_1 $A \wedge B \rightarrow C$,
- R_2 $A \rightarrow D$,
- R_3 $B \rightarrow E$, and
- R_4 $B \rightarrow C$.

From the above rules mined from different data sources, there are two data sources that support/vote rule R_1,[4] three data sources support/vote rule R_2, two data sources support/vote rule R_3, and one data source supports/votes rule R_4. Following Good's weight of evidence, we can use the frequency of a rule in S' to assign it a weight. After normalization, the weights are assigned as follows:

$$w_{R1} = \frac{2}{2+3+2+1} = 0.25,$$
$$w_{R2} = \frac{3}{2+3+2+1} = 0.375,$$
$$w_{R3} = \frac{2}{2+3+2+1} = 0.25,$$
$$w_{R4} = \frac{1}{2+3+2+1} = 0.125.$$

We have seen that rule R_2 has the highest frequency and it has the highest weight; rule R_4 has the lowest frequency and it has the lowest weight. Let $S = \{S_1, S_2, \cdots, S_m\}$, and $R_1, R_2, \cdots, R_n$ be all rules in S. Then, the weight of R_i is defined as follows:

$$w_{Ri} = \frac{Num(R_i)}{\sum_{j=1}^{n} Num(R_j)},$$

where $i = 1, 2, \cdots, n$; and $Num(R)$ is the number of data sources that contain rule R, or the frequency of R in S.

In the meanwhile, if a data source supports/votes a larger number of high-frequency rules, the weight of the data source should also be higher. If the rules from a data source are rarely present in other data sources, the data source would be assigned a lower weight. To implement this argument, we can use the sum of the multiplications of rules' weights and their frequency. For the above rule set S', we have

$$w_{D1} = 2 * 0.25 + 3 * 0.375 + 2 * 0.25 = 2.125,$$
$$w_{D2} = 1 * 0.125 + 2 * 0.25 + 3 * 0.375 = 2,$$
$$w_{D3} = 2 * 0.25 + 3 * 0.375 = 1.625.$$

After normalization, the weights of the three data sources are assigned as follows:

4. This support is different from the support defined in Section 2.3, and the support here is the number of data sources that vote the rule.

$$w_{D1} = \frac{2.125}{2.125 + 2 + 1.625} = 0.3695,$$
$$w_{D2} = \frac{2}{2.125 + 2 + 1.625} = 0.348,$$
$$w_{D3} = \frac{1.625}{2.125 + 2 + 1.625} = 0.2825.$$

We have seen that, data source D_1 supports/votes most rules with high weights and it has the highest weight; data source D_3 supports/votes least rules with high weights and it has the lowest weight.

Let $D_1, D_2, \cdots, D_m$ be m different data sources in the branches of a company, S_i the set of association rules from D_i $(i = 1, 2, \cdots, m)$, $S = \{S_1, S_2, \cdots, S_m\}$, and $R_1, R_2, \cdots, R_n$ be all rules in S. Then, the weight of D_i is defined as follows:

$$w_{Di} = \frac{\sum_{R_k \in S_i} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}},$$

where, $i = 1, 2, \cdots, m$.

After all data sources have been assigned weights, we can synthesize the association rules from them. We demonstrate the synthesizing process as follows.

For rule R_1: $A \wedge B \rightarrow C$,

$$\begin{aligned} supp(A \cup B \cup C) &= w_{D1} * supp_1(A \cup B \cup C) \\ &\quad + w_{D3} * supp_3(A \cup B \cup C) \\ &= 0.3695 * 0.4 + 0.2825 * 0.5 = 0.28905, \end{aligned}$$

$$\begin{aligned} conf(A \wedge B \rightarrow C) &= w_{D1} * conf_1(A \wedge B \rightarrow C) \\ &\quad + w_{D3} * conf_3(A \wedge B \rightarrow C) \\ &= 0.3695 * 0.72 + 0.2825 * 0.82 = 0.49769. \end{aligned}$$

For rule R_2: $A \rightarrow D$,

$$\begin{aligned} supp(A \cup D) &= w_{D1} * supp_1(A \cup D) \\ &\quad + w_{D2} * supp_2(A \cup D) + w_{D3} * supp_3(A \cup D) \\ &= 0.3695 * 0.3 + 0.348 * 0.36 \\ &\quad + 0.2825 * 0.25 = 0.306755, \end{aligned}$$

$$\begin{aligned} conf(A \rightarrow D) &= w_{D1} * conf_1(A \rightarrow D) + w_{D2} * conf_2(A \rightarrow D) \\ &\quad + w_{D3} * conf_3(A \rightarrow D) = 0.3695 * 0.64 \\ &\quad + 0.348 * 0.7 + 0.2825 * 0.62 = 0.68043. \end{aligned}$$

For rule R_3: $B \rightarrow E$,

$$\begin{aligned} supp(B \cup E) &= w_{D1} * supp_1(B \cup E) + w_{D2} * supp_2(B \cup E) \\ &= 0.3695 * 0.34 + 0.348 * 0.4 = 0.26483, \end{aligned}$$

$$\begin{aligned} conf(B \rightarrow E) &= w_{D1} * conf_1(B \rightarrow E) + w_{D2} * conf_2(B \rightarrow E) \\ &= 0.3695 * 0.7 + 0.348 * 0.6 = 0.46745. \end{aligned}$$

For rule R_4: $B \rightarrow C$,

$$supp(B \cup C) = w_{D2} * supp_2(B \cup C) = 0.348 * 0.45 = 0.1566,$$
$$conf(B \rightarrow C) = w_{D2} * conf_2(B \rightarrow C) = 0.348 * 0.87 = 0.30276.$$

The ranking of the above rules is R_2, R_1, R_3, and R_4 by their supports. According to this ranking, we can select high-rank rules after the minimum support and minimum confidence.

3.3 Rules Selection

Putting all association rules together from different data sources of a large company might also amass a huge rule set. To overcome this limitation, we now design an efficient algorithm to improve the proposed synthesizing approach in this section.

Using the weighting model in Section 3.2, we can assign a high weight to a data source that supports/votes more high-frequency rules, and a lower weight to a data source that supports/votes less high-frequency rules. However, this model can be enhanced by rule selection. We use two examples below to illustrate rule selection.

Example 2. Let $D_1, D_2, \cdots, D_{11}$ be 11 different data sources of a company, S_i the set of association rules from D_i $(i = 1, 2, \cdots, 11)$, $S_i = \{R_1 : X \rightarrow Y\}$ when $i = 1, 2, \cdots, 10$, and $S_{11} = \{R_2 : X_1 \rightarrow Y_1, \cdots, R_{11} : X_{10} \rightarrow Y_{10}\}$. Then, we have

$$\begin{aligned} w_{R1} &= \frac{Num(R_1)}{\sum_{j=1}^{11} Num(R_j)} \\ &= \frac{10}{10 + \sum_{j=1}^{10} 1} = 0.5, \end{aligned}$$

$$\begin{aligned} w_{Ri} &= \frac{Num(R_i)}{\sum_{j=1}^{11} Num(R_j)} \\ &= \frac{1}{10 + \sum_{j=1}^{10} 1} = 0.05, \end{aligned}$$

where $i = 2, 3, \cdots, 11$.

So,

$$\begin{aligned} w_{Di} &= \frac{\sum_{R_k \in S_i} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}} \\ &= \frac{10 * 0.5}{\sum_{j=1}^{10} 10 * 0.5 + \sum_{j=1}^{10} 1 * 0.05} = 0.099, \end{aligned}$$

where $i = 1, 2, \cdots, 10$.

$$\begin{aligned} w_{D11} &= \frac{\sum_{R_k \in S_{11}} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}} \\ &= \frac{\sum_{j=1}^{10} 1 * 0.05}{\sum_{j=1}^{10} 10 * 0.5 + \sum_{j=1}^{10} 1 * 0.05} = 0.01. \end{aligned}$$

We have seen that, although S_{11} has 10 rules, w_{D11} is still very low due to the fact that S_{11} does not contain high-frequency rules. If $S_{11} = \{R_2 : X_1 \rightarrow Y_1, \cdots, R_{91} : X_{90} \rightarrow Y_{90}\}$, then we have

$$\begin{aligned} w_{R1} &= \frac{Num(R_1)}{\sum_{j=1}^{11} Num(R_j)} \\ &= \frac{10}{10 + \sum_{j=1}^{90} 1} = 0.1, \end{aligned}$$

$$w_{Ri} = \frac{Num(R_i)}{\sum_{j=1}^{11} Num(R_j)} = \frac{1}{10 + \sum_{j=1}^{90} 1} = 0.01,$$

where $i = 2, 3, \cdots, 91$.

So,

$$w_{Di} = \frac{\sum_{R_k \in S_i} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}} = \frac{10 * 0.1}{\sum_{j=1}^{10} 10 * 0.1 + \sum_{j=1}^{90} 1 * 0.01} = 0.09174,$$

where $i = 1, 2, \cdots, 10$.

$$w_{D11} = \frac{\sum_{R_k \in S_{11}} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}} = \frac{\sum_{j=1}^{90} 1 * 0.01}{\sum_{j=1}^{10} 10 * 0.1 + \sum_{j=1}^{90} 1 * 0.01} = 0.0826.$$

In this case, w_{D11} becomes higher. Although w_{D11} cannot cause rules R_i $(2 \le i \le 91)$ to become valid rules in synthesis, the support and confidence of R_1 are slightly weakened by w_{D11}. The larger the number of rules in S_{11}, the more the other rules are weakened.

R_1 is the highest-frequency rule extracted from $S = \{S_1, S_2, \cdots, S_{11}\}$ in Example 2. The rules with lower-frequency (for example, less than 2) can be taken as noise. For efficiency purposes, this noise could be wiped out before the data sources are assigned weights. To eliminate this noise, because our goal is to extract high-frequency rules from different data sources, we can take the high-frequency rules as the relevant rules and, the lower-frequency rules as irrelevant rules. In this way, we can cope with abundance and redundancy of rules before the data sources are assigned weights. For this reason, we construct below, a new algorithm of rule selection from the association rules discovered from different data sources.

Procedure 1 *RuleSelection*(S);
Input: γ - minimum voting degree, S - set of N rules, n - number of data sources;
Output: S - reduced set of rules
for $i = 1$ **to** N
let $Num(R_i) \leftarrow$ the number of data sources that contain rule R_i in S;
if $(Num(R_i)/n < \gamma)$
$S \leftarrow S - \{R_i\}$;
end for;
output S;
end procedure;

The RuleSelection procedure above generates a reduced rule set, S, from the original N rules. If a rule R_i does not have a frequency that meets γ, it is erased from S. γ is the user-specified minimum voting degree by different data sources and $0 \le \gamma \le 1$. For example, if the user is interested in rules that have a voting degree equal to or greater than 80 percent, then 80 percent is the users' minimum interest degree, and γ is 0.8.

After all rules that do not meet γ are deleted, S can be significantly reduced. We now use an example to illustrate the effect on weights by wiping out low-frequency rules.

Example 3. Let $D_1, D_2, \cdots, D_{10}$ be 10 different data sources in a company, S_i the set of association rules from D_i $(i = 1, 2, \cdots, 10)$, $S_i = \{R_1 : X \rightarrow Y\}$ when $i = 1, 2, \cdots, 9$, and $S_{10} = \{R_1 : X \rightarrow Y,\ R_2 : X_1 \rightarrow Y_1,\ \cdots, R_{11} : X_{10} \rightarrow Y_{10}\}$. Then, we have

$$w_{R1} = \frac{Num(R_1)}{\sum_{j=1}^{11} Num(R_j)} = \frac{10}{10 + \sum_{j=1}^{10} 1} = 0.5,$$

$$w_{Ri} = \frac{Num(R_i)}{\sum_{j=1}^{11} Num(R_j)} = \frac{1}{10 + \sum_{j=1}^{10} 1} = 0.05,$$

where $i = 2, 3, \cdots, 11$.

So,

$$w_{Di} = \frac{\sum_{R_k \in S_i} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}} = \frac{10 * 0.5}{\sum_{j=1}^{9} 10 * 0.5 + 10 * 0.5 \sum_{j=1}^{10} 1 * 0.05} = 0.099,$$

where $i = 1, 2, \cdots, 9$.

$$w_{D10} = \frac{\sum_{R_k \in S_{11}} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}} = \frac{10 * 0.5 + \sum_{j=1}^{10} 1 * 0.05}{\sum_{j=1}^{9} 10 * 0.5 + 10 * 0.5 \sum_{j=1}^{10} 1 * 0.05} = 0.109.$$

Because the frequency of R_i is 1 $(2 \le i \le 11)$, rules R_i $(2 \le i \le 11)$ can all be wiped out if $\gamma > 0.1$. After wiping out these rules, we have $w_{Di} = 0.1$, where $i = 1, 2, \cdots, 10$. The errors between the original weights and the changed weights for data sources D_i $(1 \le i \le 9)$ are all $|0.1 - 0.099| = 0.001$, and the error for D_{10} is $|0.1 - 0.109| = 0.009$.

3.4 Algorithm Design

Let $D_1, D_2, \cdots, D_m$ be m data sources, S_i the set of association rules from D_i $(i = 1, 2, \cdots, m)$, $supp_i$ and $conf_i$ the supports and confidences of rules in S_i, and $minsupp$ and $minconf$, the threshold values given by the user. Our synthesizing algorithm for association rules in different data sources is designed as follows:

Algorithm 1 *RuleSynthesizing*
Input: $S_1, S_2, \cdots, S_m$: rule sets; $minsupp$, $minconf$: threshold values;
Output: $X \rightarrow Y$: synthesized association rules;

1. **let** $S \leftarrow \{S_1, S_2, \cdots, S_m\}$;
2. **call** RuleSelection(S);
3. **for** each rule R in S **do**
let $Num(R) \leftarrow$ the number of data sources that contain rule R in S;
let

$$w_R \leftarrow \frac{Num(R)}{\sum_{R' \in S} Num(R')};$$

4. **for** $i = 1$ **to** m **do**
   let
$$w_i \leftarrow \frac{\sum_{R_k \in S_i} Num(R_k) * w_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} Num(R_h) * w_{R_h}};$$
5. **for** each rule $X \rightarrow Y \in S$ **do**
   let $supp_w \leftarrow w_1 * supp_1 + w_2 * supp_2 + \cdots + w_m * supp_m$;
   let $conf_w \leftarrow w_1 * conf_1 + w_2 * conf_2 + \cdots + w_m * conf_m$;
6. **rank** all rules in S by their supports;
7. **output** the high-rank rules in S whose support and confidence are at least $minsupp$ and $minconf$, respectively;
8. **end all**.

The RuleSynthesizing algorithm above generates high-rank rules from the association rule sets $S_1, S_2, \cdots, S_m$, where each high-rank rule has a high frequency, support, and confidence. Step 2 calls the procedure of rule selection (see Procedure 1). Low-frequency rules in S are given up in this step. Step 3 assigns a weight to each rule in S according to its frequency. Step 4 assigns a weight to each data source (and therefore its corresponding rule set) by the number of high-frequency rules that the rule set supports. Step 5 synthesizes the support and confidence of each rule in S by the weights of different data sources. According to the weighted supports, we rank the rules of S in Step 6. The output in Step 7 is the high-rank rules selected by the user requirements.

4 Relative Synthesizing by Clustering

In this section, we construct a relative synthesizing model to synthesize association rules from different *unknown data sources*, where "unknown data sources" can be outer data sources collected from the Web, journals, and books. Individuals and organizations can take advantage of the information and knowledge that the Internet provides. Web technologies such as HTTP and HTML have dramatically changed enterprise information management. Information search engines such as Yahoo, Alta Vista, and Excite have offered easier ways to get the information that one needs. Moreover, an intranet based on Internet technology and protocols enables intraorganizational communication and internal information sharing through the corporate internal network. For example, a multinational corporation can benefit from intranets and the Internet to collect, manage, distribute, and share knowledge, inside and outside the corporation. Therefore, the vast amount of information available on the Web has great potential to improve the quality of decisions [19]. This means that companies can collect relevant information on the Web for their own applications, and this leads to huge amounts of information in their databases. Therefore, techniques for identifying truly useful patterns in the data would be very useful. This section synthesizes association rules in such data sources by clustering.

The number of rules may be very large when they are collected from unknown data sources. Consider a rule $X \rightarrow Y$, it may have different supports $supp_1$, $supp_2$, $\cdots$, $supp_m$, and confidences $conf_1$, $conf_2$, $\cdots$, $conf_m$ in the collected association rules. We can use one of the following synthesizing operators to roughly synthesize this rule.

1. Maximum synthesizing operator
$$Max\_Syn(f(R_i)) = Max\{f(R_i) | R_i \text{ is an element of } S_j \text{ for all } S_j \text{ in } S\}$$
2. Minimum synthesizing operator
$$Min\_Syn(f(R_i)) = Min\{f(R_i) | R_i \text{ is an element of } S_j \text{ for all } S_j \text{ in } S\}$$
3. Average synthesizing operator
$$Ave\_Syn(f(R_i)) = \frac{1}{\text{the number of data sources that contain } R_i} \sum_i f(R_i),$$
where S_i is a data source, R_i is a rule, function f is either $supp$ or $conf$, $supp(R_i)$ is the support of R_i in S_i, $conf(R_i)$ is the confidence of R_i, and S is the set of all data sources. The following example illustrates the use of the above methods.

Example 4. Suppose we have the following rules from different unknown data sources.

- $A \wedge B \rightarrow C$ with $supp = 0.4, conf = 0.72$;
- $A \rightarrow D$ with $supp = 0.3, conf = 0.64$;
- $A \rightarrow D$ with $supp = 0.36, conf = 0.7$;
- $A \wedge B \rightarrow C$ with $supp = 0.5, conf = 0.82$;
- $A \rightarrow D$ with $supp = 0.25, conf = 0.62$;

For rule $A \wedge B \rightarrow C$, according to the maximum synthesizing operator, we have

$$Max\_Syn(supp(A \wedge B \rightarrow C)) = Max\{0.4, 0.5\} = 0.5,$$
$$Max\_Syn(conf(A \wedge B \rightarrow C)) = Max\{0.72, 0.82\} = 0.82.$$

According to the minimum synthesizing operator we have

$$Min\_Syn(supp(A \wedge B \rightarrow C)) = Min\{0.4, 0.5\} = 0.4,$$
$$Min\_Syn(conf(A \wedge B \rightarrow C)) = Min\{0.72, 0.82\} = 0.72.$$

According to the average synthesizing operator we have

$$Ave\_Syn(supp(A \wedge B \rightarrow C)) = \frac{1}{2}(0.4 + 0.5) = 0.45,$$
$$Ave\_Syn(supp(A \wedge B \rightarrow C)) = \frac{1}{2}(0.72 + 0.82) = 0.77.$$

Different from these simple operators, we now use clustering to obtain Normal Distribution intervals among the supports and confidences of a collected rule when they exist.

TABLE 1
The Distance Table

	$conf_1$	$conf_2$	$\cdots$	$conf_n$
$conf_1$	$c_{1,1}$	$c_{1,2}$	$\cdots$	$c_{1,n}$
$conf_2$	$c_{2,1}$	$c_{2,2}$	$\cdots$	$c_{2,n}$
$\cdots$	$\cdots$	$\cdots$		$\cdots$
$conf_n$	$c_{n,1}$	$c_{n,2}$	$\cdots$	$c_{n,n}$

TABLE 2
The Distance Relation Table

	$conf_1$	$conf_2$	$conf_3$	$conf_4$	$conf_5$	$conf_6$	$conf_7$	$conf_8$
$conf_1$	*1*	*0.98*	*0.98*	*0.8*	*0.99*	*0.99*	*1*	*0.79*
$conf_2$	*0.98*	*1*	*0.96*	*0.78*	*0.99*	*0.97*	*0.98*	*0.81*
$conf_3$	*0.98*	*0.96*	*1*	*0.82*	*0.97*	*0.99*	*0.98*	*0.77*
$conf_4$	*0.8*	*0.78*	*0.82*	*1*	*0.79*	*0.81*	*0.8*	*0.59*
$conf_5$	*0.99*	*0.99*	*0.97*	*0.79*	*1*	*0.98*	*0.99*	*0.8*
$conf_6$	*0.99*	*0.97*	*0.99*	*0.81*	*0.98*	*1*	*0.99*	*0.78*
$conf_7$	*1*	*0.98*	*0.98*	*0.8*	*0.99*	*0.99*	*1*	*0.79*
$conf_8$	*0.79*	*0.81*	*0.77*	*0.59*	*0.8*	*0.78*	*0.79*	*1*

4.1 Normal Distribution

Suppose a rule $X \rightarrow Y$ has the following supports and confidences in the collected association rules:

$$supp_1, conf_1,$$
$$supp_2, conf_2,$$
$$\cdots$$
$$supp_n, conf_n.$$

If these confidences are irregularly distributed, we can apply one of the above models to synthesize them, but the synthesizing is rather rough. However, if these confidences are in a normal distribution, we can take an interval as the confidence and a corresponding interval as the support. In other words, for $0 \leq a \leq b \leq 1$, let m be the number of confidences belonging to interval $[a, b]$. If $m/n \geq \lambda$, then these confidences are in a normal distribution, where $0 < \lambda \leq 1$ is a threshold given by domain experts. This means that $[a, b]$ can be taken as the confidence of rule $A \rightarrow B$. For the corresponding support, we can estimate an interval as the support of the rule. In other words, suppose we have a random variable $X \sim N(\mu, \sigma^2)$ and we need the probability

$$P\{a \leq X \leq b\} = \frac{1}{\sigma\sqrt{2\pi}} \int_a^b e^{-(x-\mu)^2/2\sigma^2} dx$$

to satisfy $P\{a \leq X \leq b\} \geq \lambda$ and $|b - a| \leq \alpha$, where X is a variable about confidence and is valued from $conf_1, conf_2, \cdots, conf_n$, and α is a threshold given by domain experts.

For $conf_1, conf_2, \cdots, conf_n$, let $c_{i,j} = 1 - |conf_i - conf_j|$ be the *closeness* value between $conf_i$ and $conf_j$, and the closeness value between any two confidences be given in Table 1.

We can use clustering technology to obtain this normal $[a, b]$. To determine the relationship between confidences, a closeness degree measure is required. The measure calculates the closeness degree between two confidences by closeness values. We define a simple closeness degree measure as follows:

$$Close(conf_i, conf_j) = \sum (c_{k,i} * c_{k,j})$$

where "k" is summed across the set of all confidences. In effect, the formula takes the two columns of the two confidences being analyzed, multiplying and accumulating the values in each row. The results can be placed in a resultant "n" by "n" matrix, called a *confidence-confidence matrix*. This simple formula is reflexive so that the generated matrix is symmetric.

For example, let $\lambda = 0.7$, $\alpha = 0.08$, $minconf = 0.65$, a synthesized rule $X \rightarrow Y$ with confidences $conf_1 = 0.7$, $conf_2$ = 0.72, $conf_3$ = 0.68, $conf_4$ = 0.5, $conf_5$ = 0.71, $conf_6$ = 0.69, $conf_7$ = 0.7, and $conf_8$ = 0.91, and the closeness value between any two confidences is given below. (See Table 2.)

Its confidence-confidence matrix is shown as follows (see Table 3).

There are no values on the diagonal since the diagonal represents the auto-correlation of a confidence to itself. Assume 6.9 is the threshold that determines if two confidences are considered close enough to each other to be in the same class. This produces a new binary matrix called the *confidence relationship matrix* as follows (see Table 4).

Cliques require all confidences in a cluster to be within the threshold of all other confidences. The methodology to create the clusters using cliques is described in Procedure 2 as follows:

Procedure 2 *Cluster*

Input: $conf_i$: *confidence*, λ: *threshold value*;
Output: $Class$: *class set of closeness confidences*;

1. **let** i=1;
2. **select** $conf_i$ and place it in a new class;
3. $r = k = i + 1$;
4. **validate** if $conf_k$ is within the threshold of all terms within the current class;
5. **if** not, let $k = k + 1$;
6. **if** $k > n$ (number of confidences) **then** $r = r + 1$;

TABLE 3
Confidence-Confidence Matrix

	$conf_1$	$conf_2$	$conf_3$	$conf_4$	$conf_5$	$conf_6$	$conf_7$	$conf_8$
$conf_1$		7.0459	7.0855	6.0181	7.125	7.1252	7.1451	5.9546
$conf_2$	7.0459		7.0247	5.9609	7.0664	7.0646	7.0851	5.9164
$conf_3$	7.0855	7.0247		5.9793	7.0648	7.067	7.0936	5.898
$conf_4$	6.0181	5.9609	5.9793		5.9974	6.0068	6.0181	4.971
$conf_5$	7.125	7.0664	7.0648	5.9974		7.1047	7.125	5.9435
$conf_6$	7.141	7.0646	7.067	6.0068	7.1047		7.1252	5.9341
$conf_7$	7.1451	7.0851	7.0936	6.0181	7.125	7.1252		5.9546
$conf_8$	5.9546	5.9164	5.898	4.971	5.9435	5.9341	5.9546	

TABLE 4
Confidence Closeness Relationship Matrix

	$conf_1$	$conf_2$	$conf_3$	$conf_4$	$conf_5$	$conf_6$	$conf_7$	$conf_8$
$conf_1$		1	1	0	1	1	1	0
$conf_2$	1		1	0	1	1	1	0
$conf_3$	1	1		0	1	1	1	0
$conf_4$	0	0	0		0	0	0	0
$conf_5$	1	1	1	0		1	1	0
$conf_6$	1	1	1	0	1		1	0
$conf_7$	1	1	1	0	1	1		0
$conf_8$	0	0	0	0	0	0	0	

if $r = m$ **then go to** (7) **else** $k = r$; **create** a new class with $conf_i$ in it; **go to** (4);

7. **if** the current class only has $conf_i$ in it and there are other classes with $conf_i$ in them **then**
   delete the current class;
   else $i = i + 1$;
8. **if** $i = n + 1$ **then go to** (9) **else go to** (2);
9. **eliminate** any classes that duplicate or are elements of other classes.

Applying the above procedure to the above example in this section, the following classes are created:

$$\text{Class } 1 : conf_1, conf_2, conf_3, conf_5, conf_6, conf_7$$
$$\text{Class } 2 : conf_4$$
$$\text{Class } 3 : conf_8$$

For Class 1, $a = 0.68$, $b = 0.72$. Hence,

$$|b - a| = |0.72 - 0.68| = 0.04 < \alpha = 0.08,$$
$$P\{a \le X \le b\} = 6/8 = 0.75 > \lambda = 0.7,$$

and

$$b > a > minconf = 0.65.$$

For Class 2, $a = 0.5$, $b = 0.5$. Hence,

$$|b - a| = |0.5 - 0.5| = 0 < \alpha = 0.08,$$
$$P\{a \le X \le b\} = 1/8 = 0.125 < \lambda = 0.7,$$

and

$$b = a < minconf = 0.65.$$

For Class 3, $a = 0.91$, $b = 0.91$. Hence,

$$|b - a| = |0.91 - 0.91| = 0 < \alpha = 0.08,$$
$$P\{a \le X \le b\} = 1/8 = 0.125 < \lambda = 0.7,$$

and

$$b = a > minconf = 0.65.$$

Therefore, $[0.68, 0.72]$ can be taken as the interval of the confidence of rule $A \to B$.

We can also synthesize the corresponding support of a rule into an interval in the same way. For simplicity, we can also take the minimum of supports corresponding to a class as its support.

4.2 Algorithm Design

Let $A \to B$ be a collected rule, $supp_1, conf_1$, $supp_2, conf_2, \cdots, supp_n, conf_n$ the supports and confidences of the rule, $minsupp$ and $minconf$ the threshold values given by the user, and λ and α the threshold values given by domain experts. Our synthesizing algorithm for association rules from different unknown data sources is designed as follows:

Algorithm 2 *RelativeSynthesizing*
Input: $A \to B$: rule;
$supp_1, supp_2, \cdots, supp_n$: *the supports of the rule*;
$conf_1, conf_2, \cdots, conf_n$: *the confidences of the rule*;
$minsupp$, $minconf$, λ, α: *threshold values*;
Output: $A \to B$: *synthesized association rule*;

1. **for** the confidences of $A \to B$ **do**
   call Cluster;
2. **for** each class C **do**
   begin

TABLE 5
Database Characteristics

Database name	$\|R\|$	T	I	$\|r\|$
T5.I2.D100KN1	*968*	*5*	*2*	*19985*
T5.I2.D100KN2	*965*	*5*	*2*	*20046*
T10.I4.D100KN1	*960*	*10*	*4*	*19774*
T10.I4.D100KN2	*967*	*10*	*4*	*19798*
T20.I6.D100K	*968*	*20*	*6*	*19408*

let $a \leftarrow$ the minimum of values in C;
let $b \leftarrow$ the maximum of values in C;
let $d_C \leftarrow |b - a|$;
let $P_C\{a \leq X \leq b\} \leftarrow |C|/n$;
end;

3. **for** all classes **do**
if there is a class C satisfying $d_C \leq \alpha$, $P_C \geq \lambda$ and $a \geq minconf$ **then**
begin
let $supp \leftarrow$ the minimum of supports corresponding to C;
output $A \rightarrow B$ as a valid rule with support $supp$ and confidence interval $[a, b]$;
end;
4. **if** there are no classes satisfying the conditions **then**
begin
let $supp \leftarrow \frac{1}{n}(supp_1 + supp_2 + \cdots + supp_n)$;
let $conf \leftarrow \frac{1}{n}(conf_1 + conf_2 + \cdots + conf_n)$;
if $supp \geq minsupp$ and $conf \geq minconf$ **then**
output $A \rightarrow B$ as a valid rule with support $supp$ and confidence $conf$;
end;

The RelativeSynthesizing algorithm above synthesizes collected association rules from unknown data sources into two kinds of rules: one is that the supports and confidences of each rule are in normal distribution and they are clustered into intervals; and the other is that the supports and confidences of each rule are not in normal distribution and they are roughly synthesized points. Step 1 clusters the confidences of a rule. Step 2 solves the bounded values. Step 3 checks if there is a clustered class that satisfies the given threshold values. The supports and confidences of each rule in normal distribution are evaluated in this step. Otherwise, the rules are synthesized in Step 4.

5 Experiments

Following the problem formulation in Section 2.1, we mainly evaluate the effectiveness of our synthesizing approach by weighting in our experiments. Since clustering is a well-known technique and we have illustrated the use of our synthesizing method by clustering with a detailed example in Section 4, our experiments in this section focus on the synthesizing model by weighting. We compare our proposed approach with the Apriori algorithm [3] to check the ability of our approach in capturing valid association rules (for convenience, we use frequent itemsets instead). As mentioned in Section 2.3.1, one could use a parallel and distributed data mining algorithm instead of a priori to speed up our experiments, but we don't expect such a parallel and distributed data mining algorithm would change our effectiveness results in any way.

In the meanwhile, since the association rules from different data sources are reused in our model, the synthesizing model by weighting (Algorithm 1) is different from existing data mining models. To evaluate the model's efficiency, we compare two implementations of the synthesizing model. The first implements the weighting model in Section 3.2, which synthesizes rules without rule selection, and the second implements Algorithm 1 designed in Section 3.4, which has a rule selection component.

We have performed experiments with different databases, different minimum supports, and different minimum voting degrees, to test the proposed approach. The experiments have provided very similar effectiveness and efficiency results and, therefore, we present below only one set of results for each of effectiveness and efficiency.

5.1 Effectiveness

To evaluate the effectiveness, there are many possible measures one can choose to determine how good an synthesizing approach by weighting is. Our first measure is two types of errors defined as follows:

1. Maximum error (ME):

$$\overset{\text{Max}}{i \in Fset} \{||supp_i^W - supp_i^A||\}$$

2. Average error (AE):

$$\frac{1}{|Fset|} \sum_{i \in Fset} \{||supp_i^W - supp_i^A||\}$$

where $Fset$ is the set of all frequent itemsets synthesized by our approach, $supp_i^W$ is the support of itemset i by our approach, $supp_i^A$ is the support of i by a priori, and $|Fset|$ is the number of elements in $Fset$.

The second measure for the effectiveness is to check whether or not the high-rank frequent itemsets (we take the first 20 in our experiments) in a priori are captured by our approach.

Using the above two measures, we have performed several experiments. We used Oracle 8.0.3 for database management, and implemented our model on Sun Sparc using Java. To demonstrate the performance, we present one of them as follows: The databases we used are five market transaction databases from the Synthetic Classification Data

TABLE 6
Experiment Results

Database name	*cn*	*ln*	*size*
T5.I2.D100KN1	*179034*	*1695*	*4*
T5.I2.D100KN2	*197882*	*1670*	*4*
T10.I4.D100KN1	*313729*	*2470*	*5*
T10.I4.D100KN2	*332455*	*2610*	*5*
T20.I6.D100K	*571843*	*3377*	*5*
WR		*1214*	*4*
D	*1038947*	*2072*	*4*

TABLE 7
The First 20 Large Itemsets

$itemset$	$supp_i^A$	$supp_i^W$	$error$
C9, E15	*0.052870*	*0.052867*	*0.000003*
A11, C9	*0.052440*	*0.052434*	*0.000006*
A27, B247	*0.052430*	*0.052375*	*0.000065*
A27, D103	*0.052290*	*0.052239*	*0.000051*
B247, D103	*0.052290*	*0.052238*	*0.000052*
A27, F12	*0.052280*	*0.052228*	*0.000052*
B247, F12	*0.052280*	*0.052228*	*0.000052*
D103, F12	*0.052280*	*0.052228*	*0.000052*
A27, B247, D103	*0.052280*	*0.052228*	*0.000052*
A27, B247, F12	*0.052280*	*0.052228*	*0.000052*
A27, B103, F12	*0.052280*	*0.052228*	*0.000052*
B247,D103, F12	*0.052280*	*0.052228*	*0.000052*
A27, B247,D103, F12	*0.052280*	*0.052228*	*0.000052*
A27, E17	*0.051910*	*0.051918*	*0.000005*
A27, D31	*0.051200*	*0.051205*	*0.000005*
A27, F7	*0.051180*	*0.051186*	*0.000006*
D31, E17	*0.051180*	*0.051186*	*0.000006*
D31, F7	*0.051180*	*0.051186*	*0.000006*
E17, F7	*0.051180*	*0.051186*	*0.000006*
A27, D31, E17	*0.051180*	*0.051186*	*0.000006*

Sets on the Internet (http://www.kdnuggets.com/). The main properties of the five databases are as follows: There are $|R| = 1,000$ attributes in each database. The average number T of attributes per row is 5, 5, 10, 10, and 20, respectively. The number $|r|$ of transactions is approximately 20,000 in each database. The average size I of maximal frequent sets is 2, 2, 4, 4, and 6, respectively. Table 5 summarizes these parameters.

We first mined these databases with the Apriori algorithm [3] and synthesized the results in our weighting model, and then mined the union, D, of T5.I2.D100KN1, T5.I2.D100KN2, T10.I4.D100KN1, T10.I4.D100KN2, and T20.I6.D100K with the Apriori algorithm. 2,072 large itemsets were generated in the union by a priori, and 1,214 large itemsets were generated in our weighting model.

Interestingly, the 1,214 large itemsets generated in our Algorithm 1 by weighting all appear in the large itemsets generated by a priori in the union. These results are summarized in Table 6, where cn is the number of candidate itemsets, ln is the number of large itemsets when $minsupp = 0.025$, $size$ is the length of large itemsets which is the number of items in an itemset, and WR stands for the results of our weighting model.

The results from the two different methods have shown that the first 20 large itemsets in our weighting model WR are consistent with the results by a priori on the union D of these databases when $minsupp = 0.025$. These first 20 large itemsets were ranked by their supports in Table 7.

Here, all listed large itemsets have their length greater than 1. For example, "C9" and "E15" are two items, 0.052870 is the support of itemset $\{C9, E15\}$ in the union, 0.052867 is the support of the itemset generated in our model, and 0.000003 is the difference between the two supports.

The maximum error (ME) and average error (AE) are 0.000065 and 0.00003165 in Table 7, respectively. There are 858 large itemsets that were also generated by a priori, but were not given in our synthesizing model by weighting. This is because the supports of these 858 itemsets are in a very small neighborhood of the $minsupp$. In fact, these itemsets are also kept in the ranked list of frequent itemsets in our weighting model with a lower support. If we are interested in more itemsets, we can still capture them.

5.2 Efficiency

To evaluate the efficiency, we compare two implementations of the synthesizing model with the same conditions. The first implements the weighting model in Section 3.2, called *synthesis by weighting without rule selection* (or $SWNRS$), and the second implements Algorithm 1 designed in Section 3.4, called *synthesis by weighting with rule selection* (or $SWBRS$).

In our experiments, we used four groups of databases from the Synthetic Classification Data Sets on the Internet. Each database in each group is taken as a data source and there are 20, 25, 30, and 35 databases in the four groups,[5] respectively. The main properties of the databases are as follows: There are $|R| = 1,000$ attributes in each database. The average number T of attributes per row is 8. The number $|r|$ of transactions is approximately 10,000 in each database. The average size I of maximal frequent itemsets is 4.

After all databases in a group are mined using the Apriori algorithm, we apply $SWBRS$ and $SWNRS$ to synthesize the mined rules for the group. For $SWBRS$, we

5. The databases in each group were obtained by partitioning a large database. For example, a database with 200,000 transactions is divided into 20 databases for the first group.

TABLE 8
The Running Time (In Seconds)

Data set name	$SWBRS$ 0.2	$SWBRS$ 0.4	$SWNRS$
DS20	32	28	46
DS25	44	37	63
DS30	69	58	94
DS35	91	74	141

(0.2 and 0.4 are two minimum voting degrees for rule selection)

use two values, 0.2 and 0.4, as minimum voting degrees for rule selection. Table 8 shows the running times of $SWBRS$ and $SWNRS$ in generating large itemsets.

Fig. 2 and Fig. 3 show the running times of $SWBRS$ and $SWNRS$ in seconds.

5.3 Analysis

The results with our proposed approach for synthesizing association rules from different sources are encouraging. First, the first 20 frequent itemsets synthesized by our model are also the first 20 frequent itemsets mined by a priori [3]. In particular, the maximum error (ME) and average error (AE) are only 0.000065 and 0.00003165 for the first 20 frequent itemsets. Hence, the proposed approach, which focuses on synthesizing high-frequency rules from different data sources, is able to capture the high-rank frequent itemsets in the amassed database of all data sources.

Also, we have checked the efficiency by comparing $SWNRS$ with $SWBRS$. Because the proposed approach is developed only based on local instances, the identified large itemsets reflect the commonness of data sources. Reusing local instances by weighting is apparently efficient. It is clear that $SWBRS$ is more efficient than $SWNRS$. This is because many lower-frequency itemsets are pruned in our $SWBRS$ implementation. When the minimum voting degree is 0.2, there are 17 percent lower-frequency itemsets to be pruned. When the minimum voting degree is 0.4, there are 42.4 percent lower-frequency itemsets to be pruned. However, the numbers of large itemsets in both cases ($SWBRS0.2$ and $SWBRS0.4$) are approximately equal to the number of large itemsets in $SWNRS$, which are 1,219 and 1,237 large itemsets in $SWBRS0.2$ and $SWBRS0.4$, respectively.

6 Conclusions

Most research efforts on dealing with large databases in data mining have concentrated on scaling up inductive algorithms [29]. These efforts include the design of fast induction algorithms, data partitioning, and using relational representations. Our approach presented in this paper is suitable when data comes from different sources.

To make use of discovered association rules from different data sources, we have proposed a synthesizing model by weighting in this paper. When the data source for

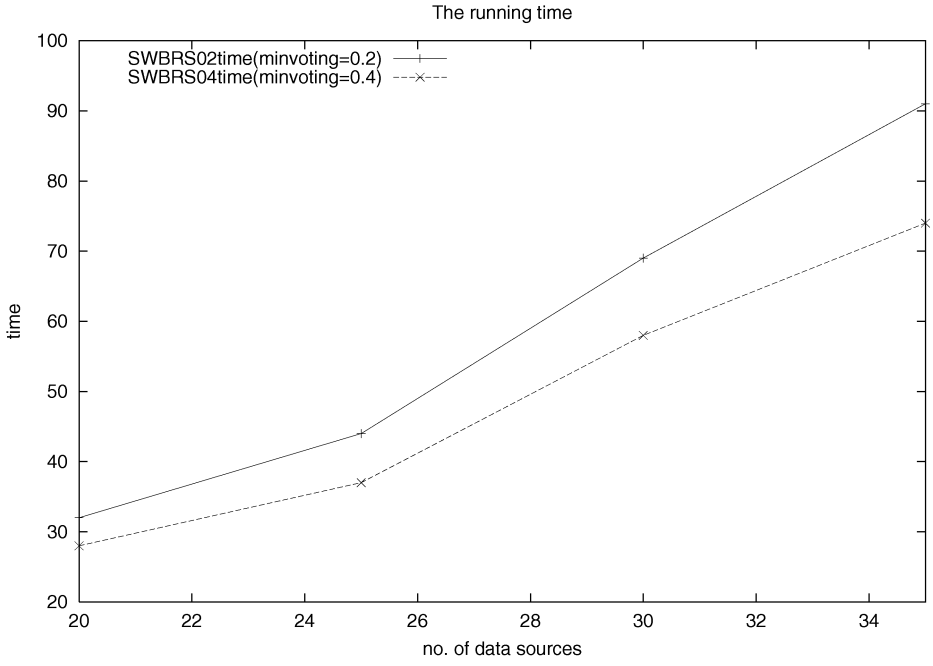

Fig. 2. The efficiency of *SW BRS* when the minimum voting degree has different values.

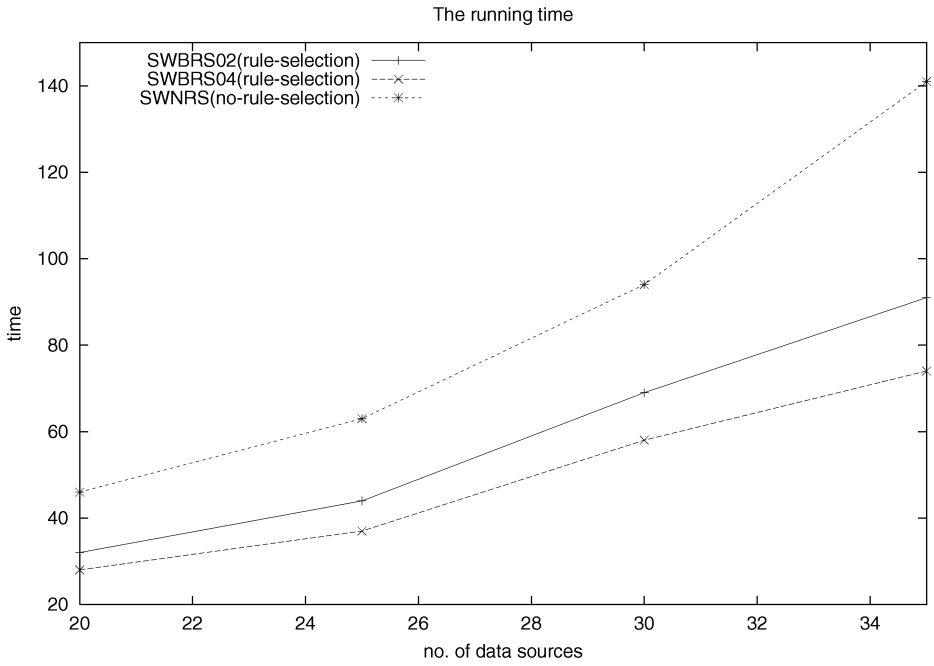

Fig. 3. The comparison of *SW BRS* and *SW NRS*.

each of the mined association rules is clear, we have constructed a weighting model to synthesize these rules. In particular, a rule selection procedure has been constructed to improve the efficiency of the weighting model. When some (or all) of the rules come from unknown data sources, we have also constructed a relative synthesizing model to synthesize these association rules by clustering.

Different from bagging and boosting, our approach can take natural data sources and synthesize the rules from different data sources into a centralized rule set. Our approach differs from incremental batch learning [12] and multilayer induction [43] in that we can mine data sources concurrently, and the sequentiality of subsets in batch learning and multilayer induction is not important in our model. Our synthesizing model is also different from stacked generalization [42], [39] because the original data (from different data sources) are not required in our synthesizing model. The differences of our synthesizing model from parallel and distributed data mining and metalearning have been reviewed in Section 2.3.

If a large company is a comprehensive organization that its data sources belong to different types of businesses and have different metadata structures, the data sources would have to be classified before our synthesizing model by weighting can be applied. For example, if an interstate company has 25 branches including five super markets, seven banks, and 13 investment trusts, we cannot directly apply the proposed approach to discover rules from the 25 data sources in the 25 branches. We would need to classify them into three classes according to the types of their businesses, and the data sources in each class can be synthesized by weighting. If the businesses of the branches of a company are mutually different, the proposed approach would not work. Also, if the data sources in a large company are not independent and rules for the overall organization require deep associations between these data sources, our synthesizing model will not work well either. Our future work will attack these problems.

ACKNOWLEDGMENTS

The authors would like to thank the five anonymous reviewers for their detailed constructive comments on the first version of this paper, which have helped improve the paper vastly. This research was supported in part by CASI (the Colorado Advanced Software Institute) under Technology Transfer Grant TTG-01-06 when X. Wu was with the Colorado School of Mines.

REFERENCES

[1] C. Aggarwal and P. Yu, "A New Framework for Itemset Generation," *Proc. ACM PODS*, 1998.
[2] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules Between Sets of Items in Large Databases," *Proc. ACM SIGMOD Conf. Management of Data*, pp. 207-216, 1993.
[3] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. Very Large Database Conf.*, 1994.
[4] R Agrawal and J.C. Shafer, "Parallel Mining of Association Rules," *IEEE Trans. Knowledge and Data Eng.*, vol. 8, no. 6, pp. 962-969, Dec. 1996.
[5] S. Brin, R. Motwani, and C. Silverstein, "Beyond Market Baskets: Generalizing Association Rules to Correlations," *Proc. ACM SIGMOD Int'l Conf. Management of Data*, pp. 265-276, 1997.
[6] P. Chan, "An Extensible Meta-Learning Approach for Scalable and Accurate Inductive Learning," PhD dissertation, Dept. of Computer Science, Columbia Univ., New York, 1996.
[7] J. Chattratichat, "Large Scale Data Mining: Challenges and Responses," *Proc. Third Int'l Conf. Knowledge Discovery and Data Mining*, pp. 143-146, 1997.
[8] M. Chen, J. Han, and P. Yu, "Data Mining: An Overview from a Database Perspective," *IEEE Trans. Knowledge and Data Eng.*, vol. 8, no. 6, pp. 866-881, Dec. 1996.

[9] D. Cheung, V. Ng, A. Fu, and Y. Fu, "Efficient Mining of Association Rules in Distributed Databases," *IEEE Trans. Knowledge and Data Eng.*, vol. 8, no. 6, pp. 911-922, Dec. 1996.

[10] D. Cheung, J. Han, V. Ng, and C. Wong, "Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique," *Proc. 12th Int'l Conf. Data Eng.*, pp. 106-114, 1996.

[11] D. Cheung, S. Lee, and B. Kao, "A General Incremental Technique for Maintaining Discovered Association Rules," *Proc. Fifth Int'l Conf. Database Systems for Advanced Applications,* pp. 185-194, 1997.

[12] S.H. Clearwater, T.P. Cheng, H. Hirsh, H. , and B.G. Buchanan, "Incremental Batch Learning," *Proc. Sixth Int'l Workshop Machine Learning,* Morgan Kaufmann, pp. 366-370, 1989.

[13] R. Godin and R. Missaoui, "An Incremental Concept Formation Approach for Learning from Databases," *Theoretical Computer Science,* pp. 387-419, 1994.

[14] I. Good, *Probability and the Weighting of Evidence.* London: Charles Griffin, 1950.

[15] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns Without Candidate Generation," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 1-12, 2000.

[16] C. Hidber, "Online Association Rules Mining," *Proc. ACM SIGMOD Conf. Management of Data,* 1999.

[17] R. Kohavi and G. John, "Wrappers for Feature Subset Selection," *Artificial Intelligence,* vol. 97, pp. 273-324, 1997.

[18] G. Lee, K.L. Lee, and A.L.P. Chen, "Efficient Graph-Based Algorithms for Discovering and Maintaining Association Rules in Large Databases," *Knowledge and Information Systems,* vol. 3, pp. 338-355, Mar. 2001.

[19] V. Lesser, B. Horling, F. Klassner, A. Raja, T. Wagner, and S. Zhang, "BIG: An Agent for Resource-Bounded Information Gathering and Decision Making," *Artificial Intelligence,* (Special Issue on Internet Information Agents), vol. 118, pp. 197-244, 2000.

[20] H. Liu and R. Setiono, "Incremental Feature Selection," *Applied Intelligence,* pp. 217-230, Sept. 1998.

[21] J. Ortega, M. Koppel, S. Argamon, "Arbitrating Among Competing Classifiers Using Learned Referees," *Knowledge and Information Systems,* vol. 4, pp. 470-490, Mar. 2001.

[22] J.S. Park, M.S. Chen, and P.S. Yu, "An Effective Hash Based Algorithm for Mining Association Rules," *Proc. ACM SIGMOD Conf. Management of Data,* 1995.

[23] J.S. Park, M. Chen, and P.S. Yu, "Efficient Parallel and Data Mining for Association Rules," *Proc. Conf. Information and Knowledge Management,* pp. 31-36, 1995.

[24] S. Parthasarathy, M.J. Zaki, M. Ogihara, and W. Li, "Parallel Data Mining for Association Rules on Shared-Memory Systems," *Knowledge and Information Systems,* vol. 1, pp. 1-29, Mar. 2001.

[25] G. Piatetsky-Shapiro, "Discovery, Analysis, and Presentation of Strong Rules," *Knowledge Discovery in Databases,* G. Piatetsky-Shapiro and W. Frawley, eds., AAAI Press/MIT Press, pp. 229-248, 1991.

[26] A. Prodromidis and S. Stolfo, "Pruning Meta-Classifiers in a Distributed Data Mining System," *Proc. First Nat'l Conf. New Information Technologies,* pp. 151-160, Oct. 1998.

[27] A. Prodromidis, P. Chan, and S. Stolfo, "Meta-Learning in Distributed Data Mining Systems: Issues and Approaches," *Advances in Distributed and Parallel Knowledge Discovery,* H. Kargupta and P. Chan, eds., AAAI/MIT Press, 2000.

[28] A.L. Prodromidis and S.J. Stolfo, "Cost Complexity-Based Pruning of Ensemble Classifiers," *Knowledge and Information Systems,* vol. 4, pp. 449-469, Mar. 2001.

[29] F. Provost and V. Kolluri, "A Survey of Methods for Scaling Up Inductive Algorithms," *Data Mining and Knowledge Discovery,* vol. 2, pp. 131-169, Mar. 1999.

[30] R. Rastogi and K. Shim, "Mining Optimized Support Rules for Numeric Attributes," *Proc. ACM SIGMOD Conf. Management of Data,* 1999.

[31] M. Sayal and P. Scheuermann, "Distributed Web Log Mining Using Maximal Large Itemsets," *Knowledge and Information Systems,* vol. 4, pp. 389-404, Mar. 2001.

[32] A. Savasere, E. Omiecinski, and S. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases," *Proc. Very Large Databases Conf.*, 1995.

[33] V. Seshadri, S. Weiss, and R. Sasisekharan, "Feature Extraction for Massive Data Mining," *Proc. First Int'l Conf. Knowledge Discovery and Data Mining,* pp. 258-262, 1995.

[34] T. Shintani and M. Kitsuregawa, "Parallel Mining Algorithms for Generalized Association Rules with Classification Hierarchy," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 25-36, 1998.

[35] M.-L. Shyu, S.-C. Chen, and R.L. Kashyap, "Generalized Affinity-Based Association Rule Mining for Multimedia Database Queries," *Knowledge and Information Systems,* vol. 3, pp. 319-337, Mar. 2001.

[36] D.B. Skillicorn and Y. Wang, "Parallel and Sequential Algorithms for Data Mining Using Inductive Logic," *Knowledge and Information Systems,* vol. 4, pp. 405-421, Mar. 2001.

[37] R. Srikant and R. Agrawal, "Mining Quantitative Association Rules in Large Relational Tables," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 1-12, 1996.

[38] R. Srikant and R. Agrawal, "Mining Generalized Association Rules," *Future Generation Computer Systems,* vol. 13, pp. 161-180, 1997.

[39] K.M. Ting and I.H. Witten, "Stacked Generalization: When Does It Work?," *Proc. Int'l Joint Conf. Artificial Intelligence '97,* pp. 866-871, 1997.

[40] D. Tsur, J. Ullman, S. Abiteboul, C. Clifton, R. Motwani, S. Nestorov, and A. Rosenthal, "Query Flocks: A Generalization of Association-Rule Mining," *Pro. ACM SIGMOD Int'l Conf. Management of Data,* pp. 1-12, 1998.

[41] G.I. Webb, "Efficient Search for Association Rules," *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 99-107, Aug. 2000.

[42] D.H. Wolpert, "Stacked Generalization," *Neural Networks,* vol. 5, pp. 241-259, 1992.

[43] X. Wu and W. Lo, "Multi-Layer Incremental Induction," *Proc. Fifth Pacific Rim Int'l Conf. Artificial Intelligence,* pp. 24-32, 1998.

Xindong Wu received the PhD degree in artificial intelligence from the University of Edinburgh, Britain. He is currently professor and chair of the Department of Computer Science at the University of Vermont. He is the executive editor of *Knowledge and Information Systems* (a quarterly journal published by Springer-Verlag), and also serves on the editorial boards of the *IEEE Transactions on Knowledge and Data Engineering* and *Data Mining and Knowledge Discovery*. He is chair of the steering committee of the IEEE International Conference on Data Mining (ICDM). Dr. Wu has published more than 90 refereed papers in various conferences and journals (including four different IEEE Transactions). He is the senior member of the IEEE.

Shichao Zhang received the BSc degree from Guangxi Normal University, China, in 1984, the MSc degree from the Computing Division at the China Atomic Energy Institute, and the PhD degree from Deakin University, Australia. He is a professor in the School of Mathematical and Computing Sciences at Guangxi Normal University, and also a research fellow in the Faculty of Information Technology at the University of Technology, Sydney, Australia. He worked as a research fellow at the National University of Singapore for two years from February 1998, and as a visiting scholar at the University of New England, Australia, for eight months from May 1997. His recent research interests include database classification, data analysis, and data mining. He has authored two books and more than 40 refereed papers in international journals and conferences.