Technical Report, 1999
School of Computing
National University of Singapore

# Summarizing the Discovered Associations Using Direction Setting Rules

Bing Liu, Wynne Hsu, and Yiming Ma
School of Computing,
National University of Singapore
Lower Kent Ridge Road
Singapore 119260

{liub, whsu, maym}@comp.nus.edu.sg

**Abstract**

Association rules are a fundamental class of patterns that exist in data. The key strength of association rule mining is its completeness. It finds all associations in the data that satisfy the user specified minimum support and minimum confidence constraints. This strength, however, comes with a major drawback. It often produces a huge number of associations. This is particularly true for data sets whose attributes are highly correlated. The huge number of associations makes it very difficult, if not impossible, for a human user to analyze in order to identify those interesting/useful ones. In this paper, we propose a novel technique to overcome this problem. The technique first prunes the discovered associations to remove those insignificant ones, and then finds a special subset of the unpruned associations to form a summary of the discovered associations. We call this subset of associations the *direction setting* (DS) *rules* as they set the directions that are followed by the rest of the associations. Using this summary, the user can interactively focus on the essential aspects (or relationships) of the domain and selectively view the relevant details. The approach is effective because experiment results show that the set of DS rules is typically very small. They can be analyzed manually by a human user. The proposed technique has also been applied successfully to a number of real-life applications.

# Summarizing the Discovered Associations Using Direction Setting Rules


**Abstract**

Association rules are a fundamental class of patterns that exist in data. The key strength of association rule mining is its completeness. It finds all associations in the data that satisfy the user specified minimum support and minimum confidence constraints. This strength, however, comes with a major drawback. It often produces a huge number of associations. This is particularly true for data sets whose attributes are highly correlated. The huge number of associations makes it very difficult, if not impossible, for a human user to analyze in order to identify those interesting/useful ones. In this paper, we propose a novel technique to overcome this problem. The technique first prunes the discovered associations to remove those insignificant ones, and then finds a special subset of the unpruned associations to form a summary of the discovered associations. We call this subset of associations the *direction setting* (DS) *rules* as they set the directions that are followed by the rest of the associations. Using this summary, the user can interactively focus on the essential aspects (or relationships) of the domain and selectively view the relevant details. The approach is effective because experiment results show that the set of DS rules is typically very small. They can be analyzed manually by a human user. The proposed technique has also been applied successfully to a number of real-life applications.


## 1. Introduction

Association rules are a class of important regularities in data. Association rule mining is commonly stated as follows [2]: Let $I = \{i_1, \ldots, i_n\}$ be a set of *items*, and $D$ be a set of data cases. Each data case consists of a subset of items in $I$. An *association rule* is an implication of the form $X \rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ holds in $D$ with confidence $c$ if $c$% of data cases in $D$ that support $X$ also support $Y$. The rule has support $s$ in $D$ if $s$% of the data case in $D$ contains $X \cup Y$. The problem of mining association rules is to generate all association rules that have support and confidence greater than the user-specified minimum support and minimum confidence.

A typical association rule mining algorithm works in two steps. The first step finds all *large* itemsets (a set of items) that meet the minimum support threshold. The second step generates rules from all large itemsets that satisfy the minimum confidence constraint.

The key strength of association rule mining is that it can efficiently discover the complete set of associations that exist in data. These associations provide a complete picture of the underlying regularities in the domain. However, this strength comes with a major drawback. The number of discovered associations can be huge, easily in the thousands or tens of thousands (see Section 8). Clearly, such a large number of associations is very difficult, if not impossible, to be analyzed by a human user. This problem is particularly bad with those data sets whose items are highly correlated.

However, it will not be satisfactory to simply give an arbitrary small subset of the rules to the user if there are indeed a large number of them that exist in the data because this small subset can only give a partial picture of the domain. The question then is: "Can we preserve the full power of association rule mining (i.e. its completeness) without overwhelming the user? This paper shows that it is possible. A novel technique is proposed to prune those insignificant rules and to find a special subset of association rules that represent the essential relationships in the data. We call this subset of associations the *direction setting* (DS) *rules*. The DS rules give a summary of the behavior of the discovered associations. They represent the essential relationships or structure (or skeleton) of the domain. The non-DS rules simply give additional details. Using the DS rules as a summary, the user can interactively focus on the essential aspects of the domain and selectively view the relevant details (non-DS rules).

In this work, we focus on association rule mining from a relational table, which consists of a set of records described by a number of attributes. An item is an attribute value pair, i.e., (attribute = value). Association rule mining in such data is typically targeted at a specific attribute because the user normally wants to know how other attributes are related to this *target* attribute (which can have many values) [13, 5]. With a target attribute, we can express an itemset as follows (instead of a set of items):

$$X \rightarrow y$$

where $y$ is an item (or a value) of the target attribute, and $X$ is a set of items from the rest of the attributes. For simplicity, we call this itemset a *rule* hereafter, regardless of whether it is significant or not. We also say a rule is *large* if it meets the minimum support constraint.

Note that we do not use minimum confidence in our framework, although we can also choose to use it (see Section 7). Minimum confidence does not reflect the underlying relationship of the domain represented by the data [5]. Instead we use statistical correlation as the basis for finding rules that represent the fundamental relations of the domain.

Figure 1 shows the conceptual flow of the proposed technique [1], which consists of two steps, pruning and summarization. Below, we briefly introduce the two steps.

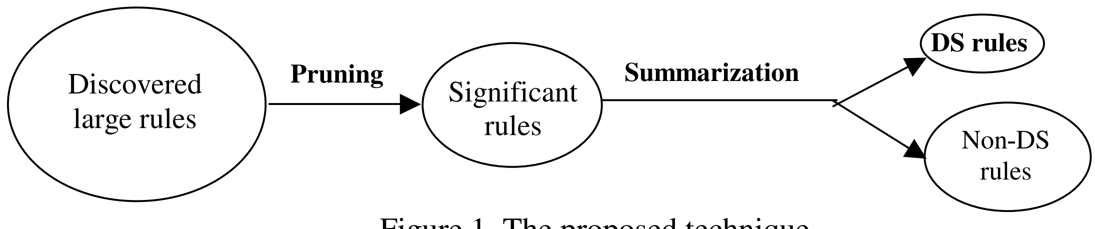


Figure 1. The proposed technique

[1] Here, it is presented as a post-processing method. In implementation, it is combined with rule mining.

### 1.1 Pruning the discovered associations

It is well known that many discovered associations are redundant or minor variations of other associations. Their existence may simply be due to chance rather than true correlation. Thus, those spurious and insignificant rules should be removed. This is similar to pruning of overfitting rules in classification [22]. Rules that are very specific (with many conditions) tend to overfit the data and have little predictive power. Although association rules are not normally used for prediction, rules that only capture the irregularities and idiosyncrasies of the data has no value and should be removed. An example of such rule is shown below.

**Example 1:** We have the following two rules,

R1: Job = yes → Loan = approved [sup = 60%, conf = 90%]

R2: Job = yes, Credit_history = good → Loan = approved [sup = 40%, conf = 91%]

If we know R1, then R2 is of limited value because it gives little extra information. Its slightly higher confidence is more likely due to chance, rather than true correlation. It should thus be pruned. R1 is more general and simple. General and simple rules are preferred.

In this work, we measure the significance of a rule using chi-square test ($\chi^2$) for correlation from classical statistics.

### 1.2 Summarizing the unpruned rules

Pruning can reduce the number of rules substantially. However, the number of unpruned rules can still be very large (see Section 7). This step finds a subset of the rules, called DS rules, to summarize the unpruned rules.

Essentially, DS rules are significant association rules that set the directions for non-DS rules to follow. The direction of a rule is the type of correlation it has, i.e., *positive correlation* or *negative correlation* or *independence*, which is also computed using $\chi^2$ test. Let us see an example.

**Example 2:** We have the following rules discovered from the data:

R1: Job = yes → Loan = approved [sup = 40%, conf = 70%]

R2: Own_house = yes → Loan = approved [sup = 30%, conf = 75%]

$\chi^2$ analysis shows that having a job is positively correlated to the grant of a loan, and owning a house is also positively correlated to obtaining a loan. Then, the following association is no surprise to us:

R3: Job = yes, Own_house = yes → Loan = approved [sup = 20%, conf = 90%]

because it intuitively follows R1 and R2. We can use R1 and R2 to provide a summary of the three rules. R1 and R2 are DS rules as they set the direction (positive correlation) that is followed by R3. In real-life data sets, a large number of associations are like R3.

From the example, we see that the DS rules give the essential relationships of the domain. The non-DS rule is not surprising if we already know the DS rules. However, this, by no means, says that non-DS rules are not interesting. Non-DS rules can provide further details about the domain. For example, the non-DS rule above (R3) gives a higher confidence. This is analogous to summarization of a text article. From the summarization, we know the essence of the article. If we are interested in the details of a particular aspect, the summary can point us to them in the article. In the same way, the DS rules give the essence of the domain and points the user to those related non-DS rules. Non-DS rules are basically combinations of DS rules. In the above Example 2, R3 is a combination of R1 and R2.

Experiment results, including real-life applications, show that although the number of discovered rules can be huge, the number of DS rules is very small (see Section 8). They can be analyzed manually by the human user to obtain the basic relationships in the data. He/she can then focus his/her attention on those interesting aspects of the relationships, and to see the relevant non-DS rules (in some applications, the user may find that the DS rules are sufficient, see Section 8.2). This process can be easily facilitated by an interactive user interface (see Section 7). The proposed technique makes association rule mining effective and practical for data sets whose items are highly correlated. The user can now obtain a complete picture of the domain without being overwhelmed by a huge number of rules.

## 2. Related Work

The problem of too many rules has been studied by many researchers. [9] proposed an approach to allow the user to specify what he/she wants to see using templates. The system then retrieves those match rules from the set of discovered rules. This method, however, does not prune those insignificant rules and it does not provide a summary of the discovered rules.

In subjective interestingness research in data mining, [12, 21, 23] proposed a number of methods for finding unexpected rules. Instead of asking the user to specify what he/she wants to see as in [9], these approaches ask the user to specify his/her existing knowledge about the domain. The system then finds those unexpected rules by comparing the user’s knowledge with the discovered rules. Again, these methods do not prune and do not attempt to summarize the association rules.

[25] and [19] investigated using item constraints specified by the user in rule mining to generate only the relevant rules. Essentially, the constraints restrict the items or combination of items that are allowed to participate in mined rules. This approach also does not prune

those insignificant rules and does not summarize the unpruned rules.

[26] introduces the concept of association rule cover for pruning association rules. A cover is basically a subset of the discovered associations that can cover the database. The number of rules in a cover can be quite small. A greedy algorithm is proposed to find a good cover and the remaining rules are pruned. The problem with this method is that the advantage of association rules, its completeness, is lost. Clearly, a better approach is to summarize the discovered rules. From this summary, the user can obtain an overall picture of the domain.

[5] proposed a rule pruning technique using *minimum improvement*, which is the difference between the confidence of a rule and the confidence of any proper sub-rule with the same consequent. Those rules that do not meet this minimum improvement in confidence are pruned. Our pruning method is similar. However, we use chi-square test as the basis for pruning (minimum improvement can be easily incorporated in our framework). We will see in Section 7 that even after pruning the number of rules left can still be very large. Summarization is thus important. The method in [5] does not perform summarization.

There are also a number of other methods in classification research for rule pruning such as pessimistic error rate [22] and minimum description length [16] based pruning. In our work, we choose the widely used chi-square test statistics for rule pruning.

[1] introduces a technique to remove two types of redundant rules, i.e., simple and strict redundancy. Essentially, a rule is redundant with respect to another rule if the support and confidence of the redundant rule are always at least as large as the support and confidence of the latter. Simple redundancy tries to remove those rules that are derived from the same itemset. For example, AB => C is redundant with respect to the rule A => BC. This, however, does not happen in our situation because in our case one itemset represents only one rule. Strict redundancy applies to two itemsets and one is a subset of the other, e.g., X => Y is redundant with respect to X => YZ. This situation also does not apply in our situation because we focus on association rules that have only one fixed attribute on the right hand side. Also our proposed technique does not use minimum confidence for rule generation (see the problems with minimum confidence in [5]), but statistical correlation (or significance).

Other related work includes correlation rule mining introduced in [6]. It uses chi-square test for measuring the correlation. A correlation rule is simply a set of items that are correlated. [11] introduces a technique for clustering association rules. However, it mainly deals with generation of numeric association rules, i.e., how to join the adjacent intervals to produce more general rules and fewer rules. Clearly, both works are different from ours.

## 3. Chi-Square Test for Independence and Correlation

Chi-square test statistics ($\chi^2$) is a widely used method for testing independence and/or correlation [18]. In our proposed technique, $\chi^2$ is used in pruning as well as in finding the direction setting rules. Below, we give an introduction to chi-square test.

Essentially, $\chi^2$ test is based on the comparison of observed frequencies with the corresponding expected frequencies. The closer the observed frequencies are to the expected frequencies, the greater is the weight of evidence in favor of independence.

**Example 3**: In a loan application domain, we have 500 people who applied for loan in a bank. Out of the 500 people, 300 had a job and 200 did not have a job. 280 people were granted loan and 220 were not. We also know that 200 people who had a job were granted loan, which can also be expressed as an association rule:

$$\text{Job} = \text{yes} \rightarrow \text{Loan} = \text{approved} \qquad [\text{sup} = 200/500, \text{conf} = 200/300]$$

This information gives us a 2×2 *contingency table* containing four cells (Figure 2). Note that the table has only 1 degree of freedom, which is sufficient for our work. See [18 or any standard statistics text] for more details about degrees of freedom.

| | Loan = approved | Loan = not-approved | Row Total: |
|---|---|---|---|
| Job = yes | **200** | **100** | 300 |
| Job = no | **80** | **120** | 200 |
| Column Total: | 280 | 220 | 500 |

Figure 2. Contingency table for *Job* and *Loan*

Our question is "Is loan approval dependent on whether one has a job or not?" In dealing with this problem we set up the hypothesis that the two attributes are independent. We then compute the expected frequency for each cell as follows: Of the 500 people included in the sample, 300 (60% of the total) had a job, while 200 (40% of the total) had no job. If the two attributes are truly independent, we would expect the 280 approved cases to be divided between job = yes and job = no in the same ratio (60% and 40%); similarly, we would expect the 220 not-approved cases to be divided in the same fashion.

$\chi^2$ is used to test the significance of the deviation from the expected values. Let $f_0$ be an observed frequency, and $f$ be an expected frequency. The $\chi^2$ value is defined as follows:

$$\chi^2 = \sum \frac{(f_0 - f)^2}{f}.$$

A $\chi^2$ value of 0 implies the attributes are statistically independent. If it is higher than a certain threshold value (e.g. 3.84 at the 95% significance level [18]), we reject the independence assumption. For our example, we obtain $\chi^2$ = 34.63. This value is much larger than 3.84. The independence assumption is rejected. Thus, we say that the loan approval is correlated to whether one has a job.

However, we still do not know how they correlated, e.g., whether having a job is *positively* or *negatively* correlated to the approval of a loan. Below, we give the definitions in the context of association rule mining.

**Definition 1 (correlated)**: Let $s$ be a minimum support and $c$ be a significance level. $X$ and $y$ of a rule, $X \rightarrow y$, are said to be ($s$, $c$) correlated (*hereafter, merely correlated*) if the following two conditions are met:

1. Its support exceeds $s$.
2. The $\chi^2$ value for the rule with respect to the whole data set exceeds the $\chi^2$ value at the significance level $c$.

**Definition 2 (uncorrelated or independent)**: Let $s$ be a minimum support and $c$ be a significance level. $X$ and $y$ of a rule, $X \rightarrow y$, are said to be ($s$, $c$) uncorrelated (*hereafter, merely uncorrelated or independent*) if the following two conditions are met:

1. Its support exceeds $s$.
2. The $\chi^2$ value for the rule with respect to the whole data set does not exceed the $\chi^2$ value at the significance level $c$.

Similar to those in [6], we define three types of correlation of a rule. For this work, we also call them the directions of a rule.

**Definition 3 (type of correlation or direction)**:

*Positive correlation*: if $X$ and $y$ of a rule $r$, $X \rightarrow y$, are correlated and $f_o / f > 1$, we say that $r$ is a *positive correlation* (or we say $r$ is *significant*). We use 1 to denote a positive correlation. We also say that the *direction* of $r$ is 1.

*Negative correlation:* if $X$ and $y$ of a rule $r$, $X \rightarrow y$, are correlated but $f_o / f < 1$, we say that $r$ is a *negative correlation*. We use -1 to denote a negative correlation. We also say that the *direction* of $r$ is -1.

*Independence*: if $X$ and $y$ of a rule $r$, $X \rightarrow y$, are independent, we say that $r$ shows *independence*. We use 0 to denote independence. The *direction* of $r$ is 0.

In this paper, we are only interested in positively correlated rules. The other two types of rules are not used (see Section 6 if such rules are required). The rule in Example 3 represents a positive correlation because its observed frequency is 200 (i.e., the support count of the rule) and its expected frequency is only 168 (= 300 * 280 / 500).

In general, computing the type of correlation (or the direction) of an association rule $r$, $X$

$\rightarrow y$, is to compare the rule with the whole population or the whole data set. Or more specifically, it is to compare with the rule that has the same conclusion as $r$ but no condition, i.e., $\rightarrow y$. The generic contingency table used is shown in Figure 3.

| | $y$ | $\neg y$ | Row Total |
|---|---|---|---|
| Satisfies $X$ | Support count of $X \rightarrow y$ | Support count of $X \rightarrow \neg y$ | Support count of $X$ in data |
| Does not satisfy $X$ | Support count of $\neg X \rightarrow y$ | Support count of $\neg X \rightarrow \neg y$ | Support count of $\neg X$ in data |
| Column Total: | Support count of $y$ in data | Support count of $\neg y$ in data | Total no. of cases in data |

Figure 3. A generic 2x2 contingency table

Note that in the table, we use the term *support count* rather than frequency to conform to association rule mining. All support counts are available from the two rules (i.e., $X \rightarrow y$ and $\rightarrow y$), after running an association rule mining algorithm. For example, to test the following rule R1 (the rule in Example 3), we compare it with R2:

R1: Job = yes → Loan = approved [sup = 200/500, conf = 200/300]
R2: → Loan = approved [sup = 280/500, conf = 280/500]

Clearly, these two rules specify completely the contingency table in Figure 2 (or Figure 3). R2, which has no condition, gives the *column total* for approved (280) and the total number of cases (500) in the data. R1, which represents the first cell in the table, also has the number 300 for the row total (which is simply the support count of the condition Job = yes). With all this information, the rest of the cells can be computed.

## 4. Direction Setting Rules

Direction setting (DS) rules are positively correlated association rules that set the directions for non-direction setting (non-DS) rules to follow. Let us see some definitions.

**Definition 4 (direction setting rule):** A rule $r$ is a *direction setting* (DS) *rule*, if it satisfies the following conditions:

1. It has the direction of 1 (positive correlation).
2. Its direction is not an element of the set of *expected directions*.

**Definition 5** (**expected directions**): The set of *expected directions* of a rule $r$ is defined as follows:

1. If $r$ is a 1-condition rule, the set of expected direction is {0} (i.e. the condition and the consequent are expected to be independent in the absence of prior knowledge).
2. If $r$ is a $k$-condition rule $r$ ($k > 1$) of the form:

   $r$: $a_1, a_2, \ldots, a_k \rightarrow y$

   the set of expected directions is computed as follows: We can view $r$ as a combination of 2 rules, a 1-condition rule $r_1$ and a ($k$-1)-condition rule $r_{rest}$, with the same consequent $y$ [2]:

$$r_1: \quad a_i \rightarrow y \qquad\qquad r_{rest}: \quad a_1, a_2, \ldots, a_{k\text{-}1} \rightarrow y$$

where $\{a_1, a_2, \ldots, a_{k\text{-}1}\} = \{a_1, a_2, \ldots, a_k\} - \{a_i\}$. The expected direction for this combination, denoted by $E_i$, is defined below ($x.dir$ denotes the direction of rule $x$):

(a). **if** $r_1.dir = r_{rest}.dir = 1$ **then** $E_i = 1$.

(b). **if** ($r_1.dir = 1$ and $r_{rest}.dir = 0$) or ($r_1.dir = 0$ and $r_{rest}.dir = 1$) **then** $E_i = 1$.

(c). **if** $r_1.dir = r_{rest}.dir = 0$ **then** $E_i = 0$.

(d). **otherwise**, $E_i$ = unknown.

Since there are $k$ such combinations, the *set of expected directions* of $r$ is $\{E_i\}$, $i = 1, \ldots, k$.

**Lemma:** All positively correlated 1-condition rules are direction setting rules.

***Proof:*** It follows directly from Definition 4 and 5.

Some notes about the above two definitions:

- In Definition 5, the assumptions (a) to (d) are reasonable because they follow our human intuitions:
  - (a) Two positive correlations generally lead to another positive correlation (Example 1).
  - (b) A positive correlation and an independence generally suggest a positive correlation.
  - (c) Two independence relations lead to another independence relation.
  - (d) For the rest of the situations, we either cannot decide the expected direction (e.g., 1 and -1 combination), or we simply do not care (e.g., -1 and -1 combination).
- The second condition (2) of Definition 4 basically says that if a rule's direction is expected with respect to any one combination, it is not a DS rule. This is reasonable because as long as there is one possible justification for $r$'s direction, we cannot say that $r$ sets a new direction. Thus, $r$ is not a DS rule.

To further explain the definitions, we list all the possible direction combinations of $r_1$, $r_{rest}$ and $r$ (Figure 4) using the notation:

$$r_1.dir, r_{rest}.dir := r.dir \quad \text{or} \quad r_{rest}.dir, r_1.dir := r.dir$$

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **A** | **(1)** | **1, 1 := 1** | **B** | **(1)** | **1, 0 := 1** | **C** | **(1)** | **0, 0 := 0** |
| | (2) | 1, 1 := -1 | | (2) | 1, 0 := -1 | | (2) | **0, 0 := 1** |
| | (3) | 1, 1 := 0 | | (3) | 1, 0 := 0 | | (3) | 0, 0 := -1 |
| **D** | (1) | -1, -1 := -1 | **E** | (1) | -1, 0 := -1 | **F** | (1) | -1, 1 := 0 |
| | (2) | **-1, -1 := 1** | | (2) | **-1, 0 := 1** | | (2) | **-1, 1 := 1** |
| | (3) | -1, -1 := 0 | | (3) | -1, 0 := 0 | | (3) | -1, 1 := -1 |

Figure 4. All possible direction combinations of $r_1$, $r_{rest}$ and $r$

[2] We can also view the rule $r$ as a combination of more than two shorter rules. The numbers of conditions in the shorter rules do not have to be 1 and $k$-1, but any possible partition. However, these alternatives require much more computation. Conceptually, they do not seem to have any advantage over our 1 and $k$-1 combination.

A(1), B(1) and C(1) represents our expectations (a), (b) and (c) in Definition 5. That is, given the directions of $r_1$ and $r_{rest}$ on the left-hand-side of ":=", if $r$ has the direction on the right-hand-side of ":=", we say $r$'s direction is expected. In fact, D(1) and E(1) can F(1) can also be seen as expected. However, they are not interesting because we are only interested in the positively correlated rules. The interesting situations occur in C(2), D(2), E(2) and F(2) (shaded in Figure 4) because the direction of $r$ is 1, but the expected direction of the $r_1$ and $r_{rest}$ combination is 0 or unknown, which are our cases (c) and (d) in Definition 5. When such situations occur, we say that $r$ sets a new direction. We call $r$ a *potential* DS rule. $r$ will be called a DS rule if for all possible combinations of $r_1$ and $r_{rest}$, the direction of $r$ is different from the expected directions. Since we are only interested in positive correlations, the remaining situations are not relevant (see also Section 6).

**Definition 6** (**non-direction setting rules**): A *non-direction setting* (non-DS) *rule* is a positively correlated rule that is not a DS rule.

## 5. Pruning the Association Rules and Finding Direction Setting Rules

This section presents the basic ideas of pruning and finding direction setting rules. The next section gives the detailed algorithm, which performs both tasks.

### 5.1. Pruning of association rules

Section 3 showed that to test for correlation between the condition and the consequent of a rule $r$, $X \rightarrow y$, we compare it with the rule $R$, $\rightarrow y$, to see whether $r$ is significant with respect to $R$. Those rules that do not show positive correlation are removed. However, we can do much better than that. Consider again Example 1 (which is reproduced here as Example 4).

**Example 4:** We have the following two rules,

$R$: Job = yes $\rightarrow$ Loan = approved [sup = 60%, conf = 90%]

$r$: Job = yes, Credit_history = good $\rightarrow$ Loan = approved [sup = 40%, conf = 91%]

If we know $R$, then $r$ is of limited use because it gives little extra information. Its slightly higher confidence is more likely due to chance, rather than true correlation.

We say $r$ can be pruned with respect to $R$ because within the subset of data cases *covered* by $R$, $r$ is not significant. (A rule *covers* a set of data cases, if the data cases satisfy the conditions of the rule.) The pruning proceeds as that in Section 3. However, instead of using the whole data set, here we test the correlation of $r$ with respect to $R$ as $r$ only covers a subset of the data cases that are covered by $R$. If $r$ does not show a positive correlation with respect to $R$, it should be pruned. In general, additional pruning is done as follows:

- Given a rule $r$, we try to prune $r$ using each *ancestor rule* $R$ (which has the same consequent as $r$ but fewer or 0 conditions). We perform a $\chi^2$ test on $r$ with respect to $R$. If the test shows a positive correlation, it is kept. Otherwise, $r$ is pruned. The reason for the pruning is because within the subset of the data covered by $R$, $r$ is not significant.

## 5.2 Finding DS rules

The process for finding DS rules is as follows: $\chi^2$ test is first used to evaluate each 1-condition rule to determine its direction status, i.e., 1 (positive correlation), -1 (negative correlation), or 0 (independence). Then, it proceeds level-by-level to analyze each rule and decide whether it follows the direction that has already been set by rules at previous levels, or whether it sets a new direction. That is, at level 2 we analyze only 2-condition rules, at level 3, we analyze 3-condition rules and so on. (For easy discussion, from now on we will use the level number and the number of conditions of a rule interchangeably). The analysis proceeds as follows: At level $k$ ($k > 1$), for each $k$-condition rule $r$, we first determine its direction. We then examine each combination of 1-condition rule $r_1$ and ($k$-1)-condition rule $r_{rest}$ of $r$ to determine whether $r$ follows the expected direction set by $r_1$ and $r_{rest}$. If $r$ follows the direction set by at least *one* such combination, we say $r$ is not a DS rule. If $r$ does not follow the direction set by any combination, we say $r$ sets a new direction, and it is a DS rule.

The computation with each combination can be illustrated using the following if-statement (Figure 5) (the corresponding situations in Figure 4 are also indicated):

| | | | |
|---|---|---|---|
| 1 | **A** | **if** $r_1.dir = r_{rest}.dir = 1$ **then** | |
| 2 | A(1) | **if** $r.dir = 1$ **then** | $r$ is not a DS rule |
| 3 | A(2)(3) | **else** $r.dir$ = unknown | |
| 4 | **B** | **elseif** ($r_1.dir = 1$ and $r_{rest}.dir = 0$) or ($r_1.dir = 0$ and $r_{rest}.dir = 1$) **then** | |
| 5 | B(1) | **if** $r.dir = 1$ **then** | $r$ is not a DS rule |
| 6 | B(2)(3) | **else** $r.dir$ = unknown | |
| 7 | **C** | **elseif** $r_1.dir = r_{rest}.dir = 0$ **then** | |
| 8 | C(1) | **if** $r.dir = 0$ **then** | nothing |
| 9 | C(2) | **elseif** $r.dir = 1$ **then** | $r$ is a potential DS rule |
| 10 | C(3) | **else** $r.dir$ = unknown | |
| 11 | **D, E, F** | **else** | |
| 12 | D(2), E(2), F(2) | **if** $r.dir = 1$ **then** | $r$ is a potential DS rule |
| 13 | D(1)(3), E(1)(3), F(1)(3) | **else** $r.dir$ = unknown | |
| 14 | | **endif** | |

Figure 5. Identifying a potential DS rule

An important point about the if-statement in Figure 5 is that *r.dir* in line 3, 6, 10, and 13 is assigned *unknown* instead of taking its original value of 0 or -1 (see Figure 4). This is to overcome the problem caused by direction 0 as the following example illustrates (Figure 6):

1 $r'$: $a_1$, $a_2$, $a_3 \rightarrow y$, $r'.dir = 1$
2 $(r'_1.dir =)1$, 0, -1
3 $0(= r'_{rest}.dir)$
4 $r'$ is not a DS rule due to B(1)

Figure 6. An example situation: the problem of direction 0

In line 1 (Figure 6), "$a_1$, $a_2$, $a_3 \rightarrow y$" is $r'$, and its direction value is 1. Assume the direction of "$a_1 \rightarrow y$" is 1, the direction of "$a_2 \rightarrow y$" is 0 and the direction of "$a_3 \rightarrow y$" is -1 (line 2). The original direction of "$a_2$, $a_3 \rightarrow y$" is 0 (line 3). Thus it is not a direction setting rule. When it is combined with $a_1$, the system will say that $r'$ is not a DS rule according to B(1). This is against our intuition because given the three 1-condition rules' directions, 1, 0, and -1, it is not clear which direction should $r'$ follow. However, if we replace the direction 0 with *unknown* for *r.dir* (line 13 in Figure 5), the above problem will be prevented and $r'$ will be reported as a potential DS rule (because now $r'_{rest}.dir$ = *unknown*). We also replace -1 with *unknown* as this does not cause any problem, and it saves a test in computation. We can now state the key theorem of the proposed technique.

**Theorem:** Using the above procedure to identify DS rules, every non-DS rule, $X \rightarrow y$, is a combination of one or more DS rules and zero or more 0 direction 1-condition rules with the same consequent $y$. That is, with a suitable re-arrangement of its conditions (as the ordering of the conditions are not important) any non-DS rule can be expressed as:

$$dsr+ \; zr^*$$

where *dsr* is any DS rule, and *zr* is any 0 direction 1-condition rule.

***Proof***: We only need to consider line 2 and 5 in the above if-statement in Figure 4, where $r$ is determined to be a non-DS rule. We use induction to prove the theorem. Since non-DS rules only appear at level 2 and above, the base case is a 2-condition non-DS rule, for which the theorem is true due to line 1 and 2, and line 4 and 5. Now, we assume it is true for all ($k$-1)-condition non-DS rules. We then show that it is also true for any $k$-condition non-DS rule.

For the situation in line 2,

$r_1$ is a DS rule (by the lemma in Section 4), and

$r_{rest}$ is either a ($k$-1)-condition DS rule or a ($k$-1)-condition non-DS rule.

Suppose $r_{rest}$ is a ($k$-1)-condition DS rule, then $r$ is a conjunction of two DS rules, i.e.,

$r$ => *dsr dsr* => *dsr+ zr** (where => means "can be expressed as")

Suppose $r_{rest}$ is a ($k$-1) non-DS rule, then by our assumption,

$r_{rest}$ => *dsr+ zr**, hence, $r$ => *dsr dsr+zr** => *dsr+zr**

In both cases, the theorem is true.

For the situation in line 5, we have two cases:

(5a) $r_1.dir = 0$ and $r_{rest}.dir = 1$, and (5b) $r_1.dir = 1$ and $r_{rest}.dir = 0$

Case (5a), once again, $r_{rest}$ is either a ($k$-1)-condition DS rule or a ($k$-1)-condition non-DS rule, while $r_1$ is a 0 direction rule, i.e. $r_1$ is *zr*.

Thus, either *r* => *zr dsr*, or *r* => *zr dsr+zr**.

Re-arranging the terms, we obtain

*r* => *dsr zr* => *dsr+zr**, or *r* => *dsr+zr+* => *dsr+zr**

Hence, the theorem is true for case (5a).

Case (5b), $r_1$ is a DS rule.

Since 0 is only produced in line 8, thus, $r_{rest}$ must be a combination of 0 direction rules, i.e., $r_{rest}$ is *zr+*.

Thus, *r* => *dsr zr+* => *dsr+zr**. Hence, theorem is true for case (5b).

Since the theorem is true when $k = 2$, and if the theorem is true for $k$-1, it is also true for $k$, we have proved the theorem. ■

From the definition of DS rules, and the theorem, we derive three important points:

- Every DS rule *r* can be seen as a *unexpected rule* with respect to $r_1$ and $r_{rest}$ of *r* because *r* does not follow their directions.
- After seeing the DS rules, the non-DS rules are no longer surprising as they are just some combinations of the DS rules and independence rules (the direction of 0). That is, the non-DS rules are expected if the DS rules are known.
- DS rules can guide the user to see the related non-DS rules, if he/she is so interested (see Section 6). Although DS rules give the essential relationships of the domain, non-DS rules can provide additional details.

Note that the above theorem does not say that any possible combination of some DS rules and/or independence rules (0 direction) is a non-DS rule. The reason is that such a combination may not meet the minimum support, or may not show positive correlation.

## 6. The Algorithm

Figure 7 gives the algorithm (called find-DS) for both pruning and finding DS rules. The input parameters are *F* and *T*. *F* is the discovered large rules. *T* is the $\chi^2$ value at a particular significance level. Two points to be noted:

- In the definitions of DS and non-DS rules, we did not mention how pruning is related. Clearly, those pruned rules will not be included in the set of DS or non-DS rules.
- For easy understanding, below the algorithm is presented as a post-processing method of

the discovered rules. However, it can be easily incorporated into a rule miner itself. In such a situation, memory claim can also be performed after a rule is pruned. In fact, we implemented our system this way (see Section 7).

```
Procedure find-DS (F, T)
1   for each r (X → y) ∈ F from level-1 to level-n do
2       r.dir = compDir(r, "→ y", T);
3       for each pair, r1 (Xi→y) and r_rest (X_rest→y), where Xi ∈ X, and X_rest= X − {Xi} do
4           if r is not pruned then evalPrune(r, r_rest) endif;
5           if X_rest ≠ ∅ and r is still a potential DS rule then      /* 1-condition rules do not go */
6               if r1.dir = r_rest.dir = 1 then
7                   if r.dir = 1 then r.justify = ∅           /* r is not a DS rule */
8                   else   r.dir = unknown
9               elseif (r1.dir = 1 and r_rest.dir = 0) or (r1.dir = 0 and r_rest.dir = 1) then
10                  if r.dir = 1 then r.justify = ∅           /* r is not a DS rule */
11                  else   r.dir = unknown
12              elseif r1.dir = r_rest.dir = 0 then
13                  if r.dir = 0 then r.justify = ∅           /* r is not a DS rule */
14                  elseif r.dir = 1 then r.justify = r.justify ∪ {(r1, r_rest)}  /*r is a potential DS rule*/
15                  else  r.dir = unknown
16              else if r.dir = 1 then r.justify = r.justify ∪ {(r1, r_rest)}  /*r is a potential DS rule*/
17                   else  r.dir = unknown
18              endif
19          endif
20      endfor
21      if r.justify ≠ ∅ then
22          if r is pruned then r.dir = unknown   /* Although r sets a new direction, it is pruned */
23          else DSR = DSR ∪ {r}                  /* r is a DS rule */
24  endfor;
25  unprnRules = {r ∈ F | r.prune = 0}.;          /* all the unpruned rules */
26  non-DSR = unprnRules - DSR                    /* all the non-DS rules */
```

Figure 7. Algorithm find-DS

The algorithm processes the discovered rules level-by-level (line 1) from level-1 to level-$n$ (where $n$ is the highest level). For each rule $r$, $X \rightarrow y$, at a particular level the algorithm works as follows: The procedure *compDir* computes the type of correlation (or direction) of $r$ (in line 2), which is given to *r.dir*. "$\rightarrow y$" is a rule without any condition. It represents the whole data set and also gives the frequency distribution of $y$ in the data. Line 3 starts to process $r$ using all pairs of its ancestors $r_1$ and $r_{rest}$. Evaluation of pruning is done in line 4 using the procedure *evalPrune*. Line 5 checks to see whether $r$ has already been determined to be a non-DS rule (1-condition rules will not go through as $X_{rest} = \emptyset$). If so, there is no need to proceed. From line 6-18, the algorithm analyzes $r$ by considering four cases. This part has been discussed in Section 5.2. *r.justify* is used to record all $r_1$ and $r_{rest}$ combinations that justify $r$ to be a potential DS rule (line 14 and 16). This information is helpful to the user in understanding the DS rules. When we know that $r$ is not a DS rule (line 7, 10 and 13), no recording is needed (we set $r.justify = \emptyset$).

After completing the inner for-loop, if $r.justify \neq \emptyset$, $r$ is justified to be a DS rule. However, if $r$ can be pruned ($r.prune \neq 0$), then it will not be a DS rule (line 22). *r.dir* is set to *unknown* (line 22) so that $r$ can be used in the future to justify a rule for being a DS rule. If $r$ cannot be pruned ($r.prune = 0$), then it is a true DS rule (line 23), and it is included in *DSR* (the set of all DS rules). All the unpruned rules are in *unprnRules* (line 25), and all the non-DS rules are in *non-DSR* (line 26).

Below, we describe the two procedures *compDir* and *evalPrune*. Procedure *compDir* (Figure 8) uses $\chi^2$ test to compute the correlation or direction of $r$. $R$ is an ancestor rule of $r$ ($R$ can be "$\rightarrow y$") and both have the same consequent. In line 1, if $\chi^2(r, R) > T$, we reject the independence assumption, i.e., the conditions and the consequent are correlated. Line 2-5 determine the type of correlation or direction (Definition 3). "$r.cover*(R.sup/R.cover)$" is the expected frequency. *r.sup* is the observed frequency (the number of data cases satisfying both the conditions and the consequent of $r$). *x.cover* is the number of data cases that satisfy the conditions of the rule $x$.

```
Procedure compDir(r, R, T)
1   if χ²(r, R) > T then
2       if r.sup > r.cover * (R.sup / R.cover)  then
3           return(1)
4       else return(-1)
5   else return(0)
```

Figure 8. Computing the direction or correlation of a rule $r$ against a more general rule $R$

The *evalPrune* procedure is shown in Figure 9. In line 1, if $r_{rest}$ has no condition, i.e., it is "$\rightarrow y$", then $r$ is a 1-condition rule. If $r$'s direction is not 1, it should be pruned as it is not a positively correlated rule. Line 2 records that $r$ is pruned because of "$\rightarrow y$" by assigning "$\rightarrow y$" to *r.prune* (which is initialized to 0, indicating $r$ is not pruned). Line 3 starts to prune $r$ using $r_{rest}$. However, $r_{rest}$ itself may have been pruned previously. Thus, the algorithm needs to find the rule that prunes $r_{rest}$. This process is reflected in line 3 and 4. In line 5 and 6, if $r$ does not represent a positive correlation, it is pruned. If $r$ is a positive correlation, then we try to compare it with $r_{rest}$ using chi-square test. If $r$ does not show a positive correlation within the subset of the data covered by $r_{rest}$, it is then pruned by $r_{rest}$.

```
Procedure evalPrune(r, r_rest)
if r_rest has no condition and r.dir ≠ 1 then     /* That is, r_rest is "→ y" */
    r.prune = "→ y";
else  if r_rest.prune ≠ 0 then                     /* If r_rest has been pruned */
          r_rest = r_rest.prune;
      if r.dir ≠ 1 then
          r.prune = r_rest
      elseif compDir(r, r_rest) ≠ 1 then           /* r is not significant compared to r_rest */
          r.prune = r_rest                         /* r can be pruned */
endif
```

Figure 9. Evaluating pruning of a rule $r$ with respect to its ancestor rule $r_{rest}$

**Complexity of the algorithm:** Let $M$ be the number of discovered large rules, and $N$ be the maximum number of conditions of a rule. The time complexity of the algorithm is thus O($MN$) ($\chi^2$ test is very efficient and can be seen as constant). Since $N$ is normally small, less than 10. Thus, the algorithm is linear in the number of large rules.

The above algorithm can be enhanced with many interesting features to make it more powerful. Below, we give two suggestions:

1. As we mentioned earlier, although non-DS rules are not essential, they can provide further details of the domain. A simple user interface can be built to allow the user to view DS rules and let DS rules point to these related non-DS rules. We have such an interface based on a web browser and hyper-links.
2. The proposed technique does not need to use minimum confidence. However, minimum confidence can be incorporated easily into the algorithm as an additional pruning mechanism. In our real-life applications, this has not been as useful as we had expected because the user often does not know what minimum confidence to set. Statistical test tends to set good lower bounds.
3. The algorithm does not report rules with negative correlation (or the direction of -1). Sometime negative correlations can also be useful. For example, it may be interesting to know the situations A(2) and B(2) in Figure 4 (i.e., 1, 1 := -1 and 1, 0 := -1). Such rules can be easily reported by adding additional "elseif" statements in the algorithm.

## 7. Empirical Evaluation

We study the effectiveness of the algorithm for pruning and finding DS rules. We used 30 data sets in our experiments. 25 of them are obtained from the UCI Machine Learning Repository [17], and 5 are our real-life data sets. All these data sets contain huge numbers of associations, which present a major problem for using association rule mining to give a complete picture of the underlying domains. Currently, the 25 public domain data sets are mainly used for classification research. However, classification can only give the user a partial picture of the data, i.e., many interesting/useful rules are not discovered [13]. With our proposed technique, we show that association rule mining can now be effectively and practically applied to these data sets to give a complete picture of the underlying relationships in the data. We also discuss two real-life applications.

### 7.1 Experiment results

Here, we report the experiment results with the 30 data sets. In all the experiments, we use a fixed attribute (target attribute) on the right hand side of association rules. The target attribute is an categorical attribute and can have a number of values (called *target items*). For the 25 UCI data sets, the target attribute in each data set is the *class attribute* used for classification. For our 5 real-life data sets, the target attributes are suggested by our users. For all these data sets, even with a target attribute, the numbers of associations discovered are huge (note that our modified association rule miner is able to use the target attribute). Many data sets cause combinatorial explosion. Due to this reason, we set a hard limit of 80,000 on the total number of large rules processed in memory. Even with such a large limit, mining cannot be completed for many data sets. Using a hard limit is justified because proceeding further only generates rules with many conditions that are hard to understand and difficult to use.

Many data sets we use contain numeric attributes. We discretize these attributes into intervals using the target attribute. There are a number of discretization algorithms in machine learning literature for the purpose. We use the entropy-based method given in [7]. The code is taken from MLC++ [10].

The experiments were repeated using different significance levels for $\chi^2$ test (95% and 90%) and different minimum support (minsup) values (2% and 1%). The 95% significance level for $\chi^2$ test is commonly used [18]. A lower level of 90% is used to show the effect of this on the results. We used the minimum supports of 2% and 1% because it is shown in [13] that for these data sets, rules with these support thresholds are sufficiently predictive.

Table 1 gave the results obtained using the significance level of 95% for $\chi^2$ test and minsup = 1%. The other results are summarized here. Below, we explain each column in Table 1. The final row gives the average value for each column.

**Column 1:** It gives the name of the data set (the last 5 are our real-life data sets). The number of records (or cases) in these data sets range from a few hundreds to tens of thousands.

**Column 2:** It shows the number of items in each data set. Recall, an item is an attribute and a value (or an interval) pair, i.e., (<$attribute_i$> = <$value_j$>).

Table 1. Experiment results (minimum support = 1%; significance level = 95%)

| 1 | | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| No. | Data sets | no. of Items | Target items | Large rules | PC. rules | PC rules Aft. pru | 1-cond DS rules | Total no. DS rules | No. of conds in DS rules | Exe. time |
| 1 | Anneal | 121 | 6 | 42893 | 18843 | 347 | 66 | 99 | 1.53 | 6.17 |
| 2 | Austra | 51 | 2 | 80000 | 40668 | 379 | 29 | 94 | 2.05 | 3.98 |
| 3. | Auto | 98 | 7 | 80000 | 43781 | 2175 | 110 | 559 | 2.45 | 2.91 |
| 4 | Breast-w | 31 | 2 | 2757 | 2362 | 97 | 26 | 27 | 1.07 | 0.14 |
| 5 | Chess | 74 | 2 | 80000 | 31939 | 2314 | 50 | 253 | 2.60 | 2.40 |
| 6 | Cleve | 30 | 2 | 30435 | 11707 | 181 | 21 | 22 | 1.22 | 1.75 |
| 7 | Crx | 55 | 2 | 80000 | 40269 | 465 | 31 | 97 | 2.92 | 3.96 |
| 8 | Diabetes | 18 | 2 | 1196 | 565 | 44 | 14 | 14 | 1.00 | 0.05 |
| 9 | German | 66 | 2 | 80000 | 21386 | 496 | 32 | 138 | 2.44 | 3.44 |
| 10 | Glass | 21 | 7 | 1820 | 1036 | 153 | 35 | 42 | 1.18 | 0.07 |
| 11 | Heart | 23 | 2 | 10044 | 3055 | 131 | 18 | 18 | 1.00 | 0.50 |
| 12 | Hepati | 36 | 2 | 80000 | 23097 | 201 | 22 | 41 | 2.44 | 3.81 |
| 13 | Horse | 66 | 2 | 79965 | 24368 | 314 | 41 | 81 | 2.26 | 2.23 |
| 14 | Hypo | 50 | 2 | 80000 | 32543 | 299 | 23 | 49 | 1.96 | 3.92 |
| 15 | Iono | 314 | 2 | 34463 | 13710 | 682 | 238 | 405 | 1.49 | 1.04 |
| 16 | Led7all | 15 | 10 | 1692 | 1389 | 691 | 70 | 72 | 1.08 | 0.06 |
| 17 | Lymph | 51 | 4 | 80000 | 18324 | 511 | 40 | 89 | 1.89 | 3.26 |
| 18 | Pima | 18 | 2 | 1196 | 565 | 44 | 14 | 14 | 1.00 | 0.04 |
| 19 | Sick | 109 | 2 | 80000 | 23538 | 456 | 35 | 73 | 1.65 | 2.69 |
| 20 | Sonar | 82 | 2 | 80000 | 24005 | 1299 | 42 | 54 | 1.65 | 3.09 |
| 21 | Tic-tac | 28 | 2 | 8315 | 2712 | 290 | 14 | 90 | 2.71 | 0.31 |
| 22 | Vehicle | 68 | 4 | 80000 | 49538 | 3259 | 90 | 474 | 2.55 | 3.03 |
| 23 | Waveform | 211 | 3 | 35888 | 30971 | 1835 | 174 | 293 | 1.67 | 0.58 |
| 24 | Wine | 37 | 3 | 27143 | 15013 | 551 | 40 | 68 | 1.54 | 1.53 |
| 25 | Zoo | 37 | 7 | 80000 | 59270 | 1159 | 59 | 204 | 2.05 | 4.56 |
| 26 | Disease | 19 | 2 | 1622 | 462 | 35 | 13 | 13 | 1.00 | 0.07 |
| 27 | Profile | 373 | 5 | 80000 | 29362 | 1229 | 77 | 217 | 2.28 | 4.97 |
| 28 | Quality | 45 | 2 | 67841 | 29776 | 295 | 40 | 89 | 2.88 | 2.34 |
| 29 | Results | 278 | 2 | 54481 | 16426 | 995 | 116 | 151 | 1.40 | 1.43 |
| 30 | Traffic | 79 | 3 | 74610 | 5916 | 231 | 17 | 94 | 2.70 | 4.10 |
| *Average* | | 84 | 3 | 50545 | 20553 | 705 | 53 | 131 | 1.91 | 2.14 |

minimum support = 2%; significance level = 95%:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *Average* | 84 | 3 | 44280 | 23885 | 612 | 50 | 93 | 1.75 | 2.14 |

minimum support = 1%; significance level = 90%:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *Average* | 84 | 3 | 50545 | 23297 | 957 | 57 | 162 | 2.08 | 2.18 |

minimum support = 2%; significance level = 90%:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *Average* | 84 | 3 | 44280 | 25978 | 810 | 53 | 109 | 1.86 | 2.08 |

**Column 3:** It shows the number of items (or values) in the target attribute of each data set.

**Column 4:** It gives the number of large rules generated from each data set. We can see that the number of large rules generated from an association rule miner is huge for each data set. Almost half of the data sets cannot be completed even under the hard limit of 80,000.

**Column 5:** It gives the number of positively correlated (PC) rules found in each data set. The number is much smaller. However, on average, there are still more than 20,000 of them.

**Column 6:** It gives the number of positively correlated (PC) rules after pruning. The number is reduced drastically. More than 96% of the PC rules are pruned. However, on average, the number of unpruned rules is still large, e.g., 705. Such a large number of rules are still hard to be analyzed by a human user.

**Column 7:** It gives the number of 1-condition DS rules.

**Column 8:** It gives the total number of DS rules found in each data set. This number of DS rules is manageable and can be analyzed manually. On average, only 131 rules are actually DS rules. Many of them are 1-condition DS rules.

**Column 9:** It gives the average number of conditions in the DS rules. The DS rules are mostly short rules.

**Column 10:** It gives the running time in second (running on 512MB SPARC 2). This time includes rule generation, pruning and finding direction setting rules. We can see that the proposed technique is quite efficient. We could log any time for pruning and finding direction setting rules only due to the efficiency.

From the summary of the other results (below Table 1), we make the following observation:

- When minimum support increases, fewer DS rules are produced. When the significance level is lowered (to 90%), more DS rules are produced. This is because pruning is less stringent. However, the numbers of DS rules are still manageable. The running times are similar because we have the hard limit.

### 7.2 Real-life applications

We have used the proposed technique in two real-life applications: traffic accident application and educational institution application. Both applications are confidential, hence we cannot provide details. Below, we give a brief description of each. The applications use the minimum support of 1% and the significance level of 95% for the $\chi^2$ test. In the traffic accident application, our users are interested in those rules that show what kind of situations are likely to result in serious injuries so that they can educate the drivers and the pedestrians. They are not interested in classifiers that can predict future cases. Thus association rule mining is more appropriate. From the database, 74610 large rules were generated. Out of these, 5916 of them are positive correlations. Our users were shocked and completely overwhelmed. However, using our technique, we identified just 94 DS rules that summarize all the underlying relationships of the data. When we showed the 94 DS rules to our users, they were satisfied because the DS rules are exactly what they needed. They only wanted to understand the key underlying regularities of the data for education purposes and were not so concerned with the full details.

The educational institution application is concerned with helping an educational institution to find out more about a special programme that it offers. We were given the examination results of those students selected to the special programme. The task is to profile the students according to the examination results. The data set (data set 27) has 373 attribute values and 5 target items. From the database, 29362 positively correlated rules were generated. After pruning, we are left with 1229 rules which are still too many to be analyzed manually. However, the number of DS rules is just 217. When we average this over the number of target items, we have 43 rules on average to analyze for each target item. This is certainly manageable. In fact, our users are very happy with the set of DS rules because the DS rules are sufficient to give them a good idea as to what combination of subjects do this special programme students excel in and what combination of subjects do they perform not as well. With this, they are able to focus more on the needs of their students and to provide extra help sessions for the more difficult subjects.

## 8. Conclusion

Data mining is to find patterns or regularities to summarize the data. If it also produces a huge number of patterns, it will be of limited use because a human user does not have the ability to analyze these patterns. However, if such a huge number of patterns do exist in the data, it will not be appropriate to arbitrarily discard any of them or to generate only a small subset of them. It is much more desirable if we can summarize them. This paper proposes such a technique. This technique first prunes off those rules that contain little extra information as compared to their ancestors, and then identifies the direction setting rules to give a global picture of the underlying relationships in the domain. Although the number of discovered associations can be very large, experimental results and real-life applications have shown that the number of direction setting rules is typically very small and with very few conditions. They can be manually inspected by a human user without too much effort.

Finally, we believe that in general any data mining technique that may potentially produce a large number of patterns should provide a technique to summarize the findings or the generated patterns. In this way, the user will be able to obtain an overall picture of the domain without being overwhelmed by a large number of detailed patterns. From this summarized information, he/she can then find some interesting aspects to focus on. The proposed technique represents a major step towards this direction.

## References

[1] Aggarwal, C., and Yu, P. "Online generation of association rules." *ICDE-98*, 402-411.

[2] Agrawal, R., Imielinski, T., Swami, A. “Mining association rules between sets of items in large databases.” *SIGMOD-1993*, 1993, 207-216.

[3] Agrawal, R. and Srikant, R. “Fast algorithms for mining association rules.” *VLDB-94*.
[4] Baralis, E. and Psaila, G. “Designing templates for mining association rules.” *Journal of Intelligent Information Systems*, 9, 1997, 7-32.
[5] Bayardo, R., Agrawal, R, and Gunopulos, D. "Constraint-based rule mining in large, dense databases." To appear in *ICDE-99*, 1999.
[6] Brin, S. Motwani, R. and Silverstein, R. “Beyond market basket: generalizing association rules to correlations.” *SIGMOD-97*, 1997, 265-276.
[7] Fayyad, U. M. and Irani, K. B. 1993. Multi-interval discretization of continuous-valued attributes for classification learning. *IJCAI-93*, 1022-1027.
[8] Han, J. and Fu, Y. “Discovery of multiple-level association rules from large databases.” *VLDB-95*, 1995.
[9] Klemetinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I. “Finding interesting rules from large sets of discovered association rules.” *CIKM-1994.*
[10] Kohavi, R., John, G., Long, R., Manley, D., and Pfleger, K. 1994. MLC++: a machine learning library in C++. *Tools with artificial intelligence*, 740-743.
[11] Lent, B, Swami, A., and Widom, J. “Clustering association rules.” *ICDE-97*, 1997.
[12] Liu, B., Hsu, W., and Chen, S. “Using general impressions to analyze discovered classification rules.” *KDD-97*, 1997, 31-36.
[13] Liu, B., Hsu, W. and Ma, Y. “Integrating classification and association rule mining.” *KDD-98*.
[14] Liu, B., Hsu, W. and Ma, Y. “Pruning and summarizing the discovered associations." KDD-99, 1999.
[15] Mannila, H., Toivonen, H. and Verkamo, A. I. “Discovery of frequent episodes in sequence.” *Data Mining and Knowledge Discovery*, 1(3): 259-289, 1997.
[16] Mahta, M., Agrawal, R. and Rissanen, J. “SLIQ: A fast scalable classifier for data mining.” *Proc. of the fifth Int’l Conference on Extending Database Technology*, 1996.
[17] Merz, C. J, and Murphy, P. UCI repository of machine learning databases [http://www.cs.uci.edu/~mlearn/MLRepository.html ], 1996.
[18] Mills, F. *Statistical Methods*, Pitman, 1955.
[19] Ng. R. T. Lakshmanan, L. Han, J. “ Exploratory mining and pruning optimizations of constrained association rules.” *SIGMOD-98*, 1998.
[20] Piatesky-Shapiro, G., and Matheus, C. “The interestingness of deviations.” *KDD-94*.
[21] Padmanabhan, B., and Tuzhilin, A. “A belief-driven method for discovering unexpected patterns.” *KDD-98*, 1998, 74-100.
[22] Quinlan, R. *C4.5*: *program for machine learning.* Morgan Kaufmann, 1992.
[23] Silberschatz, A., and Tuzhilin, A. “What makes patterns interesting in knowledge discovery systems.” *IEEE Trans. on Know. and Data Eng.* 8(6), 1996, 970-974.
[24] Silverstein, C., Brin, S., Motiwani, R., and Ullman, J. “Scalable techniques for mining causal structures.” *VLDB-98*, 1998.
[25] Srikant, R., Vu, Q. and Agrawal, R. “Mining association rules with item constraints.” *KDD-97*, 1997, 67-73.
[26] Toivonen, H. Klemetinen, M., Ronkainen, P., Hatonen, K., and Mannila, H. “Pruning and grouping discovered association rules.” Mlnet Workshop on Statistics, Machine Learning, and Discovery in Databases, 1995, 47-52.