On Optimal Rule Discovery

Jiuyong Li

Abstract—In machine learning and data mining, heuristic and association rules are two dominant schemes for rule discovery. Heuristic rule discovery usually produces a small set of accurate rules, but fails to find many globally optimal rules. Association rule discovery generates all rules satisfying some constraints, but yields too many rules and is infeasible when the minimum support is small. Here, we present a unified framework for the discovery of a family of optimal rule sets and characterize the relationships with other rule-discovery schemes such as nonredundant association rule discovery. We theoretically and empirically show that optimal rule discovery is significantly more efficient than association rule discovery independent of data structure and implementation. Optimal rule discovery is an efficient alternative to association rule discovery, especially when the minimum support is low.

Index Terms—Data mining, rule discovery, optimal rule set.

1 Introduction

Rules are among the most expressive and human-understandable representations of knowledge; a rule-based method produces self-explanatory results. Therefore, rule discovery has been a major issue in machine learning and data mining.

Heuristic algorithms for rule discovery that were developed in the machine learning community, such as C4.5rules [15], CN2 [6], and RIPPER [7] focus on classification accuracy and usually return small rule sets. However, a heuristic method does not guarantee the discovery of the best-quality rules. A complete or optimal rule set is more desirable whenever it is computationally feasible.

Association rule discovery [1] produces a complete rule set within the minimum support and confidence constraints. It has been widely accepted because of the simplicity of the problem statement and the effectiveness of pruning by support. Association rule discovery is a general-purpose rule-discovery scheme and has wide applications. Classification is one application. CBA [12] makes use of a method that is similar to the C4.5rules pruning method to prune an association rule set and produces more accurate classifiers than C4.5rules. This suggests that some rules in the complete rule set, which are missed by C4.5rules, make CBA classifiers more accurate. However, association rule discovery usually produces too many rules and is inefficient when the minimum support is low.

Nonredundant association rule discovery [19] improves the efficiency of association rule discovery. However, the requirements for redundant rules are strict and the efficiency of nonredundant rule discovery can be further improved. We discuss the relationships between our proposed optimal rule set and the redundant rule set in Section 3.

Optimal rule discovery uncovers rules that maximize an interestingness measure. The search for maxima further prunes the search space and, hence, optimal rule discovery is significantly more efficient than association rule discovery.

One type of optimal rule set is k-largest rule sets, which contain the top k rules measured by an interestingness metric. Webb and Zhang's k-optimal rule set [18] is a typical example. k-optimal rules are measured by a leverage metric. However, the top k rules may come from the same section of data and leave some records in a data set uncovered by rules. This is a drawback for optimal rule sets containing k-largest rules.

The problem of not reasonably covering the data set exists in other optimal rule sets, such as the SC optimality rule set [2] and rule sets defined by all confidence and bond [13]. In an SC optimality rule set, a rule with higher confidence and support excludes another rule with lower confidence and support. When the records covered by the excluded rule are not covered by another rule, these records lose their representative rules in the SC-optimal rule set. In the generation of rule sets defined by all confidence and bond, rules with different targets are compared directly for the exclusion of rules from the rule sets. Rules with different targets have different implications and they should not be used to exclude each other.

The definition of optimal rule sets in this paper is very close to a special constraint rule set with a zero confidence improvement [3], which consists of rules whose confidences are greater than confidences of all their simpler form rules. A rule covers a subset of records covered by one of its simpler form rules and, hence, it is guaranteed that the records covered by the excluded rule are covered by other rules with higher confidence. A PC optimality rule set [2] postprunes the constraint association rule set with a zero confidence improvement [3] and, hence, is in the same category as the optimal rule sets that we discuss.

We achieve the following four developments in this paper: First, we present a general definition for a family of optimal rule sets for a range of interestingness metrics. Second, we prove that the family of optimal rule sets observe the same antimonotonic property. Third, we develop an effective algorithm for mining the family of optimal rule sets without assuming that the target is fixed to one class. Fourth, we characterize the relationships between

• The author is with the Department of Mathematics and Computing, University of Southern Queensland, Toowomba, QLD 4350, Australia. E-mail: jiuyong@usq.edu.au.

Manuscript received 5 Apr. 2005; revised 25 July 2005; accepted 20 Sep. 2005; published online 17 Feb. 2006.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0136-0405.

an optimal rule set and a nonredundant rule set and reveal the relationships among support pruning, closure pruning, and optimality pruning.

The rest of this paper is arranged as follows: Section 2 presents definitions, Section 3 gives properties of the family of optimal rule sets, Section 4 provides a complete algorithm for mining the optimal rule sets, Section 5 illustrates the relationships among support pruning, closure pruning, and optimality pruning, Section 6 presents proof-of-concept experimental results, and Section 7 concludes the paper.

2 Definitions

Consider a relational data set D with n attribute domains. A record of D is a set of attribute-value pairs, denoted by T. A *pattern* is a subset of a record. We say a pattern is a *k-pattern* if it contains k attribute-value pairs. All the records in D are categorized by a set of classes C.

An *implication* is denoted by $P \rightarrow c$, where P is called the *antecedent*, and c is called the *consequence*. The *support* of pattern P is defined to be the ratio of the number of records containing P to the number of all the records in D, denoted by $\mathrm{supp}(P)$. The support of implication $P \rightarrow c$ is defined to be the ratio of the number of records containing both P and c to the number of all the records in D, denoted by $\mathrm{supp}(P \rightarrow c)$. The *confidence* of the implication $P \rightarrow c$ is defined to be the ratio of $\mathrm{supp}(P \rightarrow c)$ to $\mathrm{supp}(P)$, represented by $\mathrm{conf}(P \rightarrow c)$.

An *association rule* is a strong implication whose both support and confidence are not less than given thresholds from a data set.

The cover set of pattern P is the set of IDs of records containing P, represented by $\mathrm{cov}(P)$. The cover set of rule $P \rightarrow c$ is a set of IDs of records containing both P and c, denoted by $\mathrm{cov}(P \rightarrow c)$. Clearly, if $P \subset Q$, then we have $\mathrm{cov}(P) \supseteq \mathrm{cov}(Q)$ and $\mathrm{cov}(P \rightarrow c) \supseteq \mathrm{cov}(Q \rightarrow c)$.

For simplicity, in the rest of this paper, we use uppercase letters, for example, P and Q, to stand for patterns and lowercase letters, for example, $a, b, \ldots$, to stand for attribute-value pairs. We abbreviate $P \cup Q$ as PQ and $P \cup \{a\}$ as Pa.

The association rule definition is understandable, but it has the following major obstacles in real-world applications:

1. the confidence is not suitable for a variety of applications;
2. the number of association rules is too large; and
3. the support pruning is not efficient when the minimum support is low.

To overcome the first obstacle, many interestingness metrics have been proposed to measure interestingness of rules. For example, lift (interest or strength), gain, added-value, Klosgen, conviction, p-s, Laplace, cosine, certainty factor, Jaccard, and many others discussed by Tan et al. [16].

All these interestingness metrics are used monotonically. A rule with a higher value in a metric is more interesting than a rule with a lower one. To generalize, we use *Interest* to stand for an interestingness metric and $\mathrm{Interestingness}(P \rightarrow c)$ for the interestingness of rule $P \rightarrow c$.

Some interestingness metrics are not monotonic with the interestingness, such as odds ratio. A rule with an odds ratio that is significantly greater than or less than 1 is interesting. For example, let c be a disease and A be a symptom or exposure factor. A high odds ratio of rule $A \rightarrow c$ means the disease has higher occurrence probability in the cohort with A than the cohort without A and vice versa. In this paper, we consider an odds ratio that is greater than 1 and, hence, monotonic with the interestingness. When we need rules with odds ratios lower than 1, we switch the consequence. For example, when we divide data into the disease group and the nondisease group. A high odds ratio in the nondisease group means a low odds ratio in the disease group.

To make the rule definition more general, we replace the confidence with the value of an interest metric. Formally, we have the following definition:

Definition 1 (the general rule). *A rule is a strong implication whose both support and interestingness are not less than given thresholds.*

In the rest of this paper, a rule refers to a generalized rule instead of an association rule. To proceed with the discussions of obstacles 2 and 3, we give another definition:

Definition 2 (general and specific relationships). *Given two rules $P \rightarrow c$ and $Q \rightarrow c$, where $P \subset Q$, we say that the latter is more specific than the former and the former is more general than the latter.*

A specific rule covers a subset (at most the equal set) of records covered by one of its more general rules. More formally, $\mathrm{cov}(Q \rightarrow c) \subseteq \mathrm{cov}(P \rightarrow c)$. Therefore, the removal of a specific rule from a rule set does not reduce the total coverage of the rule set.

In some cases, we may consider the rule $\emptyset \rightarrow c$ as the most general rule targeting c. Its confidence equals $\mathrm{supp}(c)$. For example, if 80 percent of customers buy bread when they shop in a supermarket, then this is formalized as $\emptyset \rightarrow \mathrm{bread}\ (\mathrm{confidence} = 80\%)$. Such a rule filters many trivial rules that do not surprise a manager, for example, rule $\mathrm{egg} \rightarrow \mathrm{bread}\ (\mathrm{confidence} = 75\%)$.

In other cases, we need to consider one-pattern antecedent rules, such as $a \rightarrow c$ and $b \rightarrow c$, as the most general rules. For example, 99.9 percent of records in a medical data set are not related to a particular disease, but we are still interested in rules with 80 percent confidence in the records since they may reveal some preventative patterns. In this paper, we consider this case.

Obstacles 2 and 3 are closely related. A major reason for many rules being of no interest to users is that they are superfluous. For example, suppose that we have two rules, $(\mathrm{salary} > \$30{,}000) \rightarrow \mathrm{creditCard(approval)}\ (\mathrm{conf} = 85\%)$ and $(\mathrm{salary} > \$30{,}000)$ and $(\mathrm{occupation} = \mathrm{X}) \rightarrow \mathrm{creditCard(approval)}\ (\mathrm{conf} = 84\%)$. The latter rule is superfluous and should be removed.

There are two cases where the latter rule will be interesting: when it has much higher confidence, or when it has much lower confidence than the former rule. The primary goal for rule discovery is to find rules with high interestingness. After identifying a small set of highly interesting rules, their exceptional rules, which have low interestingness, are considered. For example, $(\mathrm{salary} > \$30{,}000)$ and $(\mathrm{occupation} = \mathrm{X}) \rightarrow \mathrm{creditCard(approval)}\ (\mathrm{confidence} = 30\%)$ is an exception to the rule $(\mathrm{salary} > \$30{,}000) \rightarrow \mathrm{creditCard(approval)}\ (\mathrm{confidence} = 85\%)$. After a small set of rules that are of interest to users has been identified, finding their exceptional rules is relatively simple.

TABLE 1
Example of Optimal Rule Set

Rule set	*Optimal rule set*
$a \rightarrow z(2.3)^*$, $b \rightarrow z(2.0), c \rightarrow z(1.8)$	$a \rightarrow z(2.3), b \rightarrow z(2.0), c \rightarrow z(1.8)$
$ab \rightarrow z(2.7), ac \rightarrow z(2.1), bc \rightarrow z(1.9)$	$ab \rightarrow z(2.7)$
$abc \rightarrow z(2.5)$	
**Numbers in parentheses are odds ratios.*	

Therefore, we disregard those superfluous rules in the rule-discovery stage. To achieve this goal, we have the following definition:

Definition 3: (an optimal rule set). *A rule set is optimal with respect to an interestingness metric if it contains all rules except those with no greater interestingness than one of its more general rules.*

Since only more specific rules are removed from an optimal rule set, an optimal rule set covers the same set of records covered by its corresponding complete rule set.

Each interestingness metric defines an optimal rule set and the above definition defines a family of optimal rule sets. We use the following example to elaborate the definition:

Example 1. Let the interestingness metric be the odds ratio (or). We have a rule set and its corresponding optimal rule set as shown in Table 1. Rules $ac \rightarrow z$ $(\mathrm{or} = 2.1)$, $bc \rightarrow z$ $(\mathrm{or} = 1.9)$, and $abc \rightarrow z$ $(\mathrm{or} = 2.5)$ are excluded since their odds ratios are smaller than those of their more general rules.

We provide another example to show the practical implication of an optimal rule set.

Example 2. When Interestingness is measured by an estimated accuracy of a rule, the optimal rule set is an optimal class association rule set [11]. Based on an ordered rule-based classification model, all rules excluded by the optimal class rule set will not be used in building a classifier because they are less accurate and more complex than some rules in the optimal class association rule set covering the same data section. Therefore, a classifier built from the optimal class association rule set is identical to that from a class association rule set. In summary [11]: The optimal class association rule set is the minimum subset of rules with the same predictive power as the complete class association rule set. Further experimental proofs are reported in [10]. Classifiers built on the optimal class association rule sets are at least of the same accuracy as those from CBA [12] and C4.5 rules [15].

Building classifiers from optimal class association rule sets is significantly more efficient than building them from class association rule sets. First, mining optimal class association rule sets is significantly faster than mining class association rule sets. When the minimum support is low, mining class association rule sets may not be feasible. Second, an optimal class association rule set is significantly smaller than a class association rule set and, hence, building classifiers from the optimal class association rule set is more efficient.

In the next section, we discuss properties of the family of the optimal rule sets.

3 Properties of Optimal Rule Sets

In this section, we discuss some properties of the family of optimal rule sets and their relationships with the non-redundant rule set.

We start with some notation. Let X be a pattern where $X \neq \emptyset$ and let c be a class. PX is a proper super pattern of P. PQ is a super pattern of P. $PQ = P$ and $PQX = PX$ when $Q = \emptyset$. PQX is a proper super pattern of P. $\neg c$ stands for a special class occurring in a record where c does not occur, and, therefore, we have $\mathrm{supp}(\neg c) = 1 - \mathrm{supp}(c)$. Similarly for pattern P: $\mathrm{supp}(\neg P) = 1 - \mathrm{supp}(P)$. $P\neg X$ is a pattern with the following support: $\mathrm{supp}(P\neg X) = \mathrm{supp}(P) - \mathrm{supp}(PX)$. Further, we have $\mathrm{supp}(\neg(PX)) = \mathrm{supp}(\neg PX) + \mathrm{supp}(\neg P\neg X) + \mathrm{supp}(P\neg X)$.

Theorem 1 (antimonotonic property). *If* $\mathrm{supp}(PX\neg c) = \mathrm{supp}(P\neg c)$,[1] *then rule* $PX \rightarrow c$ *and all its more-specific rules will not occur in an optimal rule set defined by confidence, odds ratio, lift (interest or strength), gain, added-value, Klosgen, conviction, p-s (or leverage), Laplace, cosine, certainty factor, or Jaccard.*

A proof is provided in the Appendix.

The practical implication of the above theorem is that once $\mathrm{supp}(PX\neg c) = \mathrm{supp}(P\neg c)$ is observed, it is not necessary to search for more specific rules of $PX \rightarrow c$, for example, $PQX \rightarrow c$. Those more specific rules will not be in an optimal rule set. Rule $PX \rightarrow c$ is removed since it is not in an optimal rule set either.

In the following, we consider two special cases of the above theorem:

Corollary 1: (closure property). *If* $\mathrm{supp}(P) = \mathrm{supp}(PX)$, *then rule* $PX \rightarrow c$ *for any* c *and all its more specific rules do not occur in an optimal rule set defined by confidence, odds ratio, lift (interest or strength), gain, added-value, Klosgen, conviction, p-s (or leverage), Laplace, cosine, certainty factor or Jaccard.*

A proof is provided in the Appendix.

The practical implication of the above corollary is that, once $\mathrm{supp}(PX) = \mathrm{supp}(P)$ is observed, it is not necessary to

1. In this paper, we only discuss rules with a single class as the consequence, but this theorem holds for rules with a pattern as the consequence.

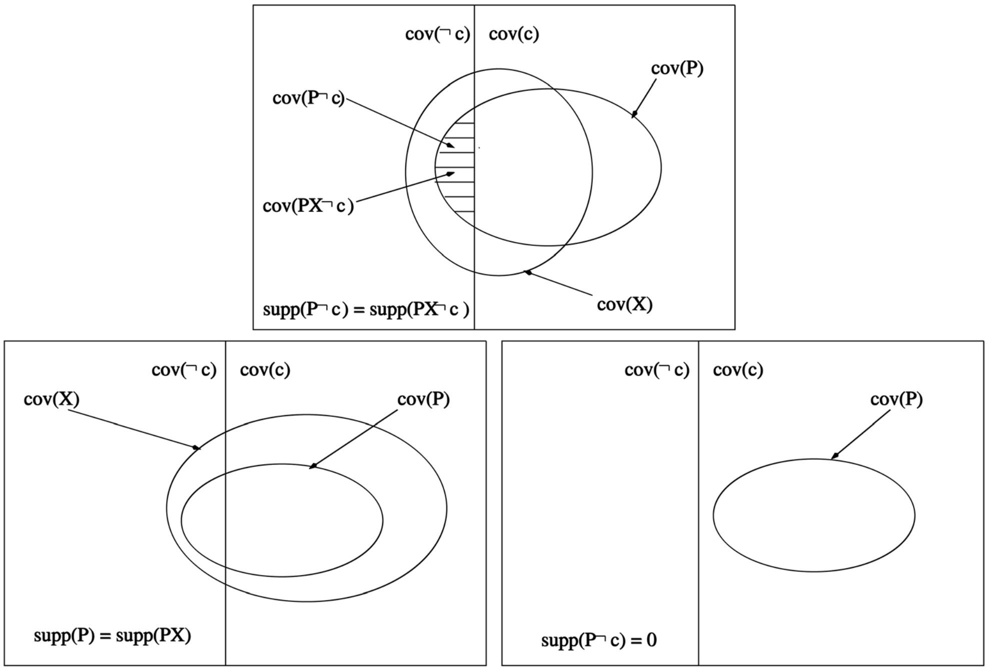

Fig. 1. Depictions of conditions of Theorem 1 and Corollaries 1 and 2. In each diagram, the left-hand rectangle stands for the set of records not containing the class c, and the right-hand rectangle stands for the set of records containing the class c. Ellipses denote the cover sets of patterns P and X.

search for rules with PQX as their antecedent for any Q. Those rules will not be in the optimal rule set.

The reason for naming the above corollary as the closure property is that it is closely related to closed pattern set mining [20].

Pattern P^c is closed if there exists no proper super pattern $X \supset P^c$ such that $\mathrm{cov}(X) = \mathrm{cov}(P^c)$. Consider a chain of patterns $P \subset P' \subset P'' \subset P^c$, which satisfies $\mathrm{cov}(P) = \mathrm{cov}(P') = \mathrm{cov}(P'') = \mathrm{cov}(P^c)$. P^c is the closure of patterns P, P', and P''. A closed pattern is the same as its closure. The support of a pattern is equivalent to that of its closure. In the above example, $\mathrm{supp}(P) = \mathrm{supp}(P') = \mathrm{supp}(P'') = \mathrm{supp}(P^c)$. Further, a pattern Y is a proper generator of Y' if $Y' \supset Y$ and $\mathrm{cov}(Y') = \mathrm{cov}(Y)$ hold. A pattern is a minimal generator if it has no proper generator. For example, P' is a generator of P'' and P^c, and P is a minimal generator of P^c if there exists no $Z \subset P$ such that $\mathrm{cov}(P) = \mathrm{cov}(Z)$.

The closed pattern and minimal generator are two closely related concepts and the mining methods for both are very similar. The condition of Corollary 1 is the fundamental test for mining both patterns. For example, we have $\mathrm{supp}(P) = \mathrm{supp}(PX)$ for any $X \subseteq (P^c \backslash P)$ in the above example. We illustrate this in Section 5. Usually, closed patterns are useful for producing all frequent patterns and minimal generators are useful for generating nonredundant rules.

Zaki [19] gave a general definition of non-redundant association rule sets. Here, we rephrase it in a simple form by constraining the consequence to c. Rule $Y \rightarrow c$ is redundant if there exists $Y \supset X$ such that $\mathrm{cov}(Y) = \mathrm{cov}(X)$. $\mathrm{supp}(Y) = \mathrm{supp}(X)$ is the immediate result of $\mathrm{cov}(Y) = cov(X)$. For example, in the chain $P \subset P' \subset P'' \subset P^c$, all rules, such as $P' \rightarrow c$, $P'' \rightarrow c$, and $P^c \rightarrow c$, are redundant since they have the same support and interestingness as rule $P \rightarrow c$ but are more specific. A rule set is nonredundant if it includes all rules except redundant rules.

Theorem 2: Relationship with the nonredundant rule set. *An optimal rule set is a subset of a nonredundant rule set.*

A proof is provided in the appendix.

We have another special case for Theorem 1.

Corollary 2: Termination property. *If* $\mathrm{supp}(P\neg c) = 0$, *then all more-specific rules of the rule* $P \rightarrow c$ *do not occur in an optimal rule set defined by confidence, odds ratio, lift (interest or strength), gain, added-value, Klosgen, conviction, p-s (or leverage), Laplace, cosine, certainty factor or Jaccard.*

A proof is provided in the appendix.

The practical implication of the above corollary is that once $\mathrm{supp}(P\neg c) = 0$ is observed, it is not necessary to search for more specific rules of $P \rightarrow c$. Those more-specific rules will not be in an optimal rule set. Rule $P \rightarrow c$ is kept since it may be in an optimal rule set.

Let us look at why it is called the termination property. Assume that the interestingness metric is confidence. If $\mathrm{supp}(P\neg c) = 0$, then rule $\mathrm{conf}(P \rightarrow c) = 100\%$. None of its more specific rules improves this confidence and we stop going any further. The same is true for other interestingness metrics.

An illustrative comparison of conditions of Theorem 1 and Corollaries 1 and 2 is given in Fig. 1.

We use the following example to show how the Theorem 1 and its corollaries work.

Example 3. We assume z is a class and do not assume the minimum support requirement in this example.

	b	c		z
a	b	c	d	z
a		c		z
a	b		d	z
	b		d	$\neg z$
a	b	c		$\neg z$

Usually, we have to consider all 15 candidate rules: $a \to z$, $b \to z$, $c \to z$, $d \to z$, $ab \to z$, $ac \to z$, $ad \to z$, $bc \to z$, $bd \to z$, $cd \to z$, $abc \to z$, $abd \to z$, $acd \to z$, $bcd \to z$, and $abcd \to z$.

Since $\mathrm{supp}(ab\neg z) = \mathrm{supp}(a\neg z)$, according to Theorem 1, rule $ab \to z$ and all its more specific rules—$abc \to z$, $abd \to z$, and $abcd \to z$—will not be in an optimal rule set. Similarly, both rule $ac \to z$ and rule $bc \to z$ and their more specific rules will not be in the optimal rule set.

Since $\mathrm{supp}(ad\neg z) = 0$, all more specific rules of $ad \to z$, e.g., $abd \to z$, $acd \to z$, and $abcd \to z$ will not be in the optimal rule set. Similarly, all more specific rules of $cd \to z$ will not be in the optimal rule set.

Since $\mathrm{supp}(d) = \mathrm{supp}(bd)$, $bd \to c$ and all its more specific rules will not be in the optimal rule set.

Therefore, rules that are possible members of the optimal rule set are $a \to z$, $b \to z$, $c \to z$, $d \to z$, $ad \to z$, and $cd \to z$. This set of candidate rules is significantly smaller than the original 15 candidate rules.

4 An Optimal Rule Discovery Algorithm

In this section, we first show how to use Theorem 1 and its corollaries for forward pruning. Then, we discuss a candidate-presentation method for easy pruning. Sections 4.3 and 4.4 present the detailed implementation of forward pruning by Theorem 1 and its corollaries. The complete algorithm is presented in Section 4.5. After that, an illustrative example shows how the algorithm works.

4.1 Basic Ideas and Forward Pruning

General-to-specific searching is very common in rule discovery. For example, C4.5rules, CN2, and Apriori employ this method.

When a heuristic search method is employed, we worry less about combinatorial explosion. However, when we conduct an optimal search, combinatorial explosion is a big problem.

We first look at how association rule discovery alleviates this problem. An association rule discovery algorithm prunes infrequent patterns forwardly. A pattern is frequent if its support is not less than the minimum support. A pattern is potentially frequent only if all its subpatterns are frequent and this antimonotonic property is used to limit the number of patterns to be searched. This is called forward pruning.

Optimal rule discovery makes use of Theorem 1 and its corollaries to forwardly prune rules.

We illustrate how Theorem 1 forwardly prunes rules that are not in an optimal set. Given a pattern $abcd$, assume the target is fixed to z. We usually have to examine candidate rules $a \to z$, $b \to z$, $\ldots$ for one-patterns, $ab \to z$, $ac \to z$, $\ldots$ for two-patterns, $abc \to z$, $abd \to z$, $\ldots$ for three-patterns, and $abcd \to z$ for four-pattern. If we know $\mathrm{supp}(a\neg z) = \mathrm{supp}(ab\neg z)$, then the theorem empowers us to skip examining candidates, $ab \to z$, $abc \to z$, $abd \to z$, and $abcd \to z$ because they are not in the optimal rule set anyway. Corollary 2 works in a similar way.

We show how Corollary 1 forwardly prunes rules that are not in an optimal rule set by using the above example. When the targets are not fixed to z but include x and y too, the number of rule candidates is tripled. We list those including pattern ab in their antecedents as follows: $ab \to x$, $ab \to y$, $ab \to z$, $abc \to x$, $abc \to y$, $abc \to z$, $abd \to x$, $abd \to y$, $abd \to z$, $abcd \to x$, $abcd \to y$, and $abcd \to z$. If we know that $\mathrm{supp}(a) = \mathrm{supp}(ab)$ holds, then all candidates listed above are ignored according to Corollary 1. Corollary 1 prunes more candidates than Theorem 1, but its requirement is stricter.

4.2 Candidate Representation

To facilitate the implementation of forward pruning by Theorem 1 and its corollaries, we define a rule candidate as a pair of (pattern, target set), denoted by (P, C). P is a pattern and C is a set of classes. In a relational data set, we have $P \cap C = \emptyset$. For example, let $P = abc$ and $C = xyz$, and then (P, C) stands for three candidate rules: $abc \to x$, $abc \to y$, and $abc \to z$. To remove a candidate rule, we simply remove a class from the target set. For example, candidate $\{abc, yz\}$ stands for only two candidate rules, namely, $abc \to y$ and $abc \to z$. When the target set is empty, the candidate stands for no rules.

The removal of classes from target set C is determined by Theorem 1 and its corollaries. For example, if we have $\mathrm{supp}(ab\neg x) = \mathrm{supp}(abc\neg x)$, then, according to the Theorem, rule $abc \to x$ and all its more specific rules will not occur in the optimal rule set. x should be removed from the candidate set and the candidate becomes (abc, yz). If we know $\mathrm{supp}(abc\neg y) = 0$, y should be removed from the target set according to Corollary 2, and the candidate becomes (abc, z). Consider another candidate (bcd, xyz). If we have $\mathrm{supp}(bcd) = \mathrm{supp}(cd)$, then, according to Corollary 1, the target set of the candidate should be emptied and the candidate becomes $(bcd, \emptyset)$.

Candidate $(P, \emptyset)$ should be removed since no rules will be generated from it and its supercandidates. (P', C') is called a supercandidate of (P, C) if $P' \supset P$ holds.

The existence of a candidate relies on two conditions: 1) Pattern P is frequent, and 2) target set C is not empty.

4.3 Candidate Generator

For easy comparison, we present Candidate-gen for optimal rule-set discovery in a way similar to Apriori-gen. We call a candidate l-candidate if its pattern is an l-pattern. An l-candidate set includes all l-candidates.

Function: Candidate-gen

```
    // Combining
1) for each pair of candidates $(P_{l-1}s, C_s)$ and $(P_{l-1}t, C_t)$ in an
     $l$-candidate set
2)   insert candidate $(P_{l+1}, C)$ in the $(l+1)$-candidate set
         where $P_{l+1} = P_{l-1}st$ and $C = C_s \cap C_t$
    // Pruning
3)    for all $P_l \subset P_{l+1}$
4)      if candidate $(P_l, C_l)$ does not exist
5)      then remove candidate $(P_{l+1}, C)$ and return
```

```
6)     else $C = C \cap C_l$
7)  if the target set of $(P_{l+1}, C)$ is empty
8)  then remove the candidate
```

First, we explain lines 1 and 2. Suppose that we have two candidates, (abc, xy) and (abd, y). The new candidate is $(abcd, y)$. The intersection of target sets here and in line 6 is to ensure that removed classes from the target set of a candidate never appear in the target set of its super-candidates. The correctness is guaranteed by Theorem 1 and its corollaries since any class removal in the target set is determined by them.

Second, we explain lines 3 to 8. Suppose we have new candidate $(abcd, y)$. It is the combination of (abc, y) and (abd, yz). We need to check if candidates identified by patterns $\{acd\}$ and $\{bcd\}$ exist. Suppose that they do exist and are (acd, y) and (bcd, xz). After considering candidate (acd, y), by line 7, the new candidate remains unchanged. After considering candidate (bcd, xz), by line 7, the target set of the new candidate becomes empty because $C = \{y\} \cap \{x, z\} = \emptyset$. The new candidate is then pruned.

4.4 More Pruning

We have a pruning process in candidate generation and will have another pruning process after counting the support of candidates. This is a key to making use of Theorem 1 and its corollaries for pruning. In the following algorithm, σ is the minimum support.

Function: $\mathrm{Prune}(l+1)$

```
// $l+1$ is the new level where candidates are counted.
1) for each candidate $(P, C)$ in $(l+1)$-candidate set
2)   for each $c \in C$
       // test the frequency individually
3)       if $\mathrm{supp}(Pc)/\mathrm{supp}(c) \leq \sigma$ then remove $c$ from $C$
       // test the satisfaction of Corollary 2
4)       else if $\mathrm{supp}(P\neg c) = 0$ then mark $c$ terminated
5)   if $C$ is empty then remove candidate $(P, C)$ and return
6)   for each $l$-level subset $P' \subset P$
       // test the satisfaction of Corollary 1
7)       if $\mathrm{supp}(P) = \mathrm{supp}(P')$ then empty $C$
       // test the satisfaction of Theorem 1
8)       else if $\mathrm{supp}(P\neg c) = \mathrm{supp}(P'\neg c)$ then
         remove $c$ from $C$
9)   if $C$ is empty then remove candidate $(P, C)$
```

We prune candidates from two aspects, the infrequency of the pattern and the emptiness of the target set.

Here, we consider a local support instead of the global support. Because of the possible skewed distributions of classes, a single global support is not suitable for a variant rule targeting different classes. Many applications have adopted local support in spite of using different names, such as coverage in [3]. The local support of rule $P \rightarrow c$ is defined as $\mathrm{supp(Pc)}/\mathrm{supp}(c)$. The local support is the support in the data subset containing c. It is also called the recall of rule $P \rightarrow c$. We prefer local support since it observes the antimonotonic property of the support. When we make use of local support, infrequent candidate rules are removed one by one, as in line 3.

We are aware of two variants of the forward pruning by Theorem 1 and Corollary 2. One is that rule $PX \rightarrow c$ and all its more-specific rules are not in an optimal rule set as in Theorem 1. In this case, we just remove c from the target set. Another is that all more specific rules of rule $P \rightarrow c$ are not in an optimal rule set except rule $P \rightarrow c$, as in Corollary 2. In this case, we cannot remove c since we may lose rule $P \rightarrow c$. We design a special statue for this situation, namely, termination of target c.

Definition 4. *Target $c \in C$ is terminated in candidate (P, C) if* $\mathrm{supp}(Pc) = 0$.

Terminated c is removed after the rule forming in line 9 of the ORD algorithm.

Line 4 of the Prune function is a direct application of Corollary 2. Target c is marked as terminated and will be removed after a rule is formed. Once c is removed from C, all more-specific rules of $P \rightarrow c$ will be removed in the following rule-candidate generation.

Line 7 of the Prune function is a direct result of Corollary 1. All classes in C are removed and, as a result, candidate (P, C) is removed. No supercandidates of (P, C) will be formed in the following rule-candidate generation.

Line 8 of the Prune function is a direct utilization of Theorem 1. Target c is removed and, therefore, rule $P \rightarrow c$ and all its more-specific rules are removed in the following candidate generation.

All candidates with an empty target set are removed in lines 5 and 9 to ensure their supercandidates are pruned in the following candidate generation.

4.5 ORD Algorithm

Now, we are able to present our algorithm for optimal rule discovery, abbreviated ORD. In this algorithm, any interestingness metric discussed in Section 3 can be used as a rule-selection criterion and the output rule set is an optimal rule set defined by the interestingness metric. It differs from association rule discovery in that it does not form rules from the set of all frequent patterns, but generates rules using partial frequent patterns.

Algorithm 1 ORD: Optimal Rule Discovery

Input: a data set D, the minimum local support σ and the minimum interestingness θ by a metric.
Output: an optimal rule set R defined by an interestingness metric.

```
1) Set $R = \emptyset$
2) Count support of one-patterns by arrays
3) Build one-candidate sets
4) Form and add rules to $R$
5) Generate two-candidate set
6) While new candidate set is not empty
7)      Count support of patterns for new candidates
8)      Prune candidates in the new candidate set
9)      Form rules and add optimal rules to $R$
10)     Generate next-level candidate set
11) Return the rule set $R$
```

The above algorithm is self-explanatory and its two main functions have been discussed in the previous sections.

The ORD algorithm is efficient since it does not generate all frequent patterns. It only makes use of a small subset of frequent patterns as shown in experiments. Note that in line 8 in Function Candidate-gen and lines 5 and 9 in Function Prune, all supercandidates of a candidate with the

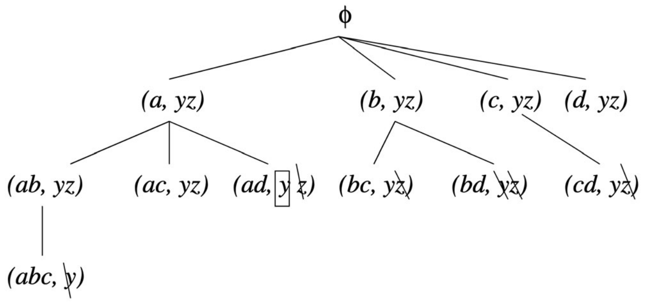

Fig. 2. All candidates searched in Example 4. A class crossed is removed and a class boxed is terminated.

empty target set, are removed on top of those of a candidate with the infrequent pattern.

4.6 An Illustrative Example

We provide an example to show how the ORD algorithm works in this section.

Example 4. In the following data set, y and z are classes. We do not assume the minimum support constraint. All candidates generated by the ORD algorithm are listed in Fig. 2.

	b	c	d	y
a		c		y
a	b	c		y
a	b	c	d	y
a				y
	b		d	z
a		c		z
	b	c	d	z
a				z
a	b	c		z

The first-level candidates are used to prune the second-level candidates. z is removed in candidate (ad, yz) by line 3 in Function Prune due to $\text{supp}(adz) = 0$. y in candidate (ad, yz) is terminated by line 4 in Function Prune because of $\text{supp}(ad\neg y) = 0$. Both y and z are removed in candidate (bd, yz) by line 7 in Function Prune since $\text{supp}(bd) = \text{supp}(d)$ holds. z in candidate (bc, yz) and (cd, yz) is removed by Line 8 in Function Prune because both $\text{supp}(bc\neg z) = \text{supp}(b\neg z)$ and $\text{supp}(cd\neg z) = \text{supp}(d\neg z)$ hold.

After rules have been formed, candidates $(ad, \emptyset)$ and $(bd, \emptyset)$ are removed. As a result, all their supercandidates, such as $(abd, \emptyset)$, $(acd, \emptyset)$, and $(bcd, \emptyset)$, will not be generated according to lines 4 and 5 in Function Candidate-gen.

Candidate (abc, yz) is generated by combining candidates (ab, yz) and (ac, yz) according to lines 1 and 2 in Function Candidate-gen. z is then pruned by line 7 in Function Candidate-gen using its subcandidate (bc, y). y in candidate (abc, y) is removed by line 8 in Function Prune due to $\text{supp}(abc\neg y) = \text{supp}(ab\neg y)$. Subsequently, candidate $(abc, \emptyset)$ is removed.

In the rule-forming procedure, rules are formed by a user-specified interestingness metric. No matter what metric discussed in Section 3 is used, the set of candidates is identical. Only the output rule set differs.

5 Support Pruning, Closure Pruning and Optimality Pruning

In this section, we discuss support pruning, closure pruning, and optimality pruning and then characterize relationships among them. This clarifies the efficiency improvement of optimal rule discovery over association rule discovery and nonredundant rule discovery.

First, look at the support pruning of the following data set:

	b	c	d	e
a	b		d	
a	b	c		
a	b	c	d	
a		c		

Data set A

Fig. 3 shows support pruning using the minimum support of 0.2. We see that the removal of e in Level 1 equals the removal of 15 patterns in subsequent levels, such as ae, be, ..., abe, ace, ..., $abce$, $abde$, ..., and $abcde$.

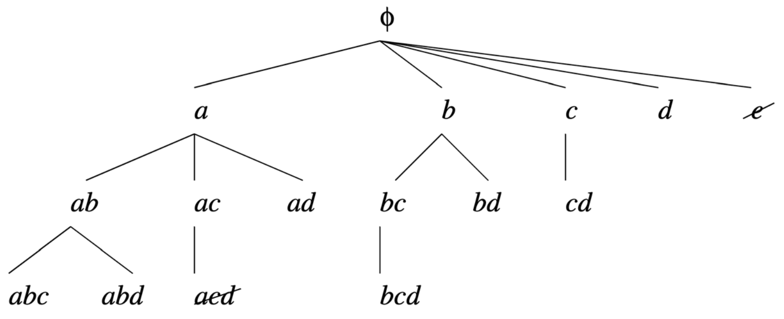

Fig. 3. Support pruning for mining frequent patterns on data set A. Patterns crossed are infrequent.

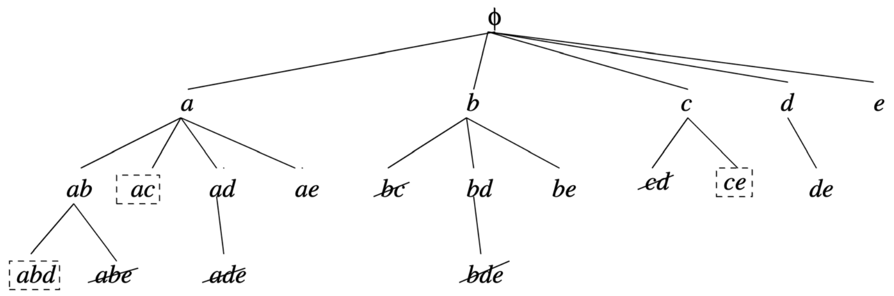

Fig. 4. Closure pruning for mining minimal generators on data set B. Patterns crossed are nonexisting and patterns boxed are terminated

Support pruning works effectively when the underlying data set is sparse or the minimum support is high. However, it does not work well on dense data sets or when the minimum support is low.

Look at the closure pruning by the following data set:

a	b		d	
a		c		e
	b			e
			d	e

Data set B

Fig. 4. shows the closure pruning. There is no minimum support requirement.

As discussed in Section 3, Corollary 1 summarizes the closure pruning. Each pattern in a box is terminated because the support of its superpatterns equals that of itself. If the final goal is to find minimal generators, all candidates are left as they are in Fig. 4. If the the final goal is to find closed patterns, the number of candidates remains unchanged, but patterns in some candidates are extended. For example, ac is terminated due to $\mathrm{supp}(ac) = \mathrm{supp}(c)$ and, as a result, all occurrences of c will be replaced by ac. Similarly, all occurrences of c are further replaced by ce because of the termination of ce by $\mathrm{supp}(ce) = \mathrm{supp}(c)$. Pattern ace is the only closed pattern left out in Fig. 4, and other minimal generators are closed patterns too. Closed patterns can be discovered in the same search tree finding minimal generators. Therefore, both closed-pattern mining and minimal-generator mining search for the same number of candidates and make use of the closure pruning strategy.

Closure pruning works effectively when the underlying data set is dense or the minimum support is low. We use an example to elaborate the first point. Assume that a dense data set contains five identical records $\{a, b, c, d, e\}$. Closed-pattern mining will stop at Level 2 after searching for 15 candidates. In contrast, frequent-pattern mining will continue all the way to Level 5 and search for $2^5 - 1$ candidates. We present the following justification for the second point: $\mathrm{supp}(PX) = \mathrm{supp}(P)$ means $\mathrm{cov}(P) \subseteq \mathrm{cov}(X)$. When $\mathrm{cov}(X)$ remains unchanged, pattern P with a smaller $\mathrm{cov}(P)$ is more probable to satisfy $\mathrm{cov}(P) \subseteq \mathrm{cov}(X)$ than with a larger $\mathrm{cov}(P)$.

The above two pruning strategies are complementary. How does the ORD algorithm use them in an effective way?

Let us look at the following data set concatenating the above two data sets. z is the target for rules. Fig. 5 shows the optimality pruning with the minimum local support of 0.2 in the data subset containing z.

	b	c	d	e	z
a	b		d		z
a	b	c			z
a	b	c	d		z
a		c			z
a	b		d		$\neg z$
a		c		e	$\neg z$
	b			e	$\neg z$
			d	e	$\neg z$

Data set (A + B)

The candidate set searched by optimal rule discovery is the intersection of the candidate set (frequent patterns) for association rule discovery in z data subset and the candidate set (minimal generators) for nonredundant rule discovery in $\neg z$ data subset. The crucial point is that both have to perform simultaneously. Both association rule discovery and nonredundant rule discovery search for more candidates than optimal rule discovery.

Now, we have an insight understanding of optimality pruning as stated in Theorem 1. It makes use of the closure pruning strategy. Comparing Corollary 1 with Theorem 1,

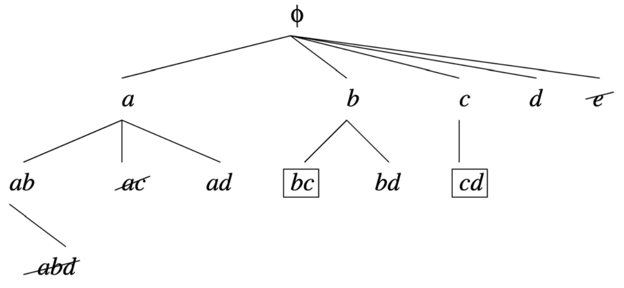

Fig. 5. Optimality pruning for optimal rule discovery targeting z on data set $(\mathrm{A}+\mathrm{B})$. Patterns crossed are removed and patterns boxed are terminated.

TABLE 2
A Brief Description of Data Sets

Name	#Records	#attribute	#classes
Anneal	898	38	5
Hypothyroid	3163	25	2
Mushroom	8124	22	2
Sick	2800	29	2

we find that Theorem 1 states Corollary 1 in the data subset excluding c.

6 Experimental Results

In this section, we empirically evaluate the computational complexity of optimal rule discovery in comparison with association rule discovery and nonredundant association rule discovery on four data sets from UCML repository [4] described in Table 2. The efficiency of optimal rule discovery is its effective optimality pruning. We show that optimality pruning significantly reduces candidates for searching.

The efficiency of an algorithm significantly depends on the data structure and implementation. For example, association rule discovery has various implementations. All are based on support pruning strategy, but their execution times vary. Theoretically, their computational complexities are the same since they all search for all frequent patterns. Their efficiencies vary since they employ different data structures and counting schemes. There are a number of implementations for association rule discovery and we are unable to compare with them individually by execution time.

However, the computational complexity improvement is fundamental and the implementation only accelerates the improvement. An empirical estimation of the complexity for a rule-discovery algorithm is the number of candidates it searches. In this paper, we compare the searched candidates for association rule discovery, for nonredundant association rule discovery, and for optimal rule discovery. An association-discovery algorithm searches for all frequent patterns and a nonredundant rule-discovery algorithm searches for all frequent minimal generators (equivalently, all frequent closed patterns). We compare the number of candidates for the ORD algorithm with the number of frequent patterns and the number of frequent minimal generators. This comparison is independent of the implementation.

In this experiment, we employ the local support as defined in Section 4.4, which is a ratio for an individual class. A pattern is frequent if it is frequent in at least one class. All frequent patterns are stored in the prefix tree.

The experiment was conducted on a 1 GHz CPU computer with 2 Gbytes memory running Linux. We search for rules containing up to eight attribute-value pairs. We do not specify the type of optimal rule set since the same candidate set generates an optimal rule set defined by any interestingness metric discussed in Section 3.

Fig. 6 shows the searched candidates by the ORD algorithm in comparison with the number of frequent patterns and the number of frequent minimal generators. The set of ORD candidates is a very small subset of frequent patterns, and a subset of frequent minimal generators. This trend is more evident when the minimum support is low. This shows that optimal rule discovery has significantly less computational complexity than association rule discovery, and less computational complexity than nonredundant association rule discovery.

In comparison with optimal rule discovery and nonredundant rule discovery, association rule discovery is very inefficient in data sets such as were used in this experiment. The efficiency of association rule discovery deteriorates dramatically when the minimum support is low. Optimal rule discovery is more efficient than nonredundant association rule discovery. Though differences between candidate numbers of nonredundant association rule discovery and optimal rule discovery are squashed in Fig. 6 by the large number of frequent patterns, the discrepancies are still clear in data sets Mushrooms and Sick.

7 Conclusions

In this paper, we discussed a family of optimal rule sets, the properties for their efficient discovery, and their relationships with the nonredundant rule sets. The family of optimal rule sets supports a simple antimonotonic property and an optimal rule set is a subset of a nonredundant rule set. We presented the ORD algorithm for mining optimal rule sets and evaluated its computational complexity on some data sets in comparison with the association rule discovery and nonredundant association rule discovery. The computational complexity of optimal rule discovery is significantly lower than that of association rule discovery and lower than that of nonredundant association rule discovery. We discussed the relationship of optimal pruning with support pruning and closure pruning and we concluded that optimality pruning makes use of both support and closure pruning strategies simultaneously on two disjointed data sets.

Optimal rule discovery is efficient and works well with low or no minimum support constraint. It generates optimal rule sets for a number of interestingness metrics. Therefore, it is a great alternative for association rule discovery.

Appendix

In this appendix, we provide proofs for Theorems 1 and 2, and Corollaries 1 and 2.

Theorem 1 (antimonotonic property). *If* $\mathrm{supp}(PX\neg c) = \mathrm{supp}(P\neg c)$, *then rule* $\mathrm{supp}PX \rightarrow c$ *and all its more specific rules will not occur in an optimal rule set defined by confidence, odds ratio, lift (interest or strength), gain, added-value, Klosgen, conviction, p-s (or leverage), Laplace, cosine, certainty factor, or Jaccard.*

Proof. In the proof, we show

$$\mathrm{Interestingness}(PQX \rightarrow c) \leq \mathrm{Interestingness}(PQ \rightarrow c).$$

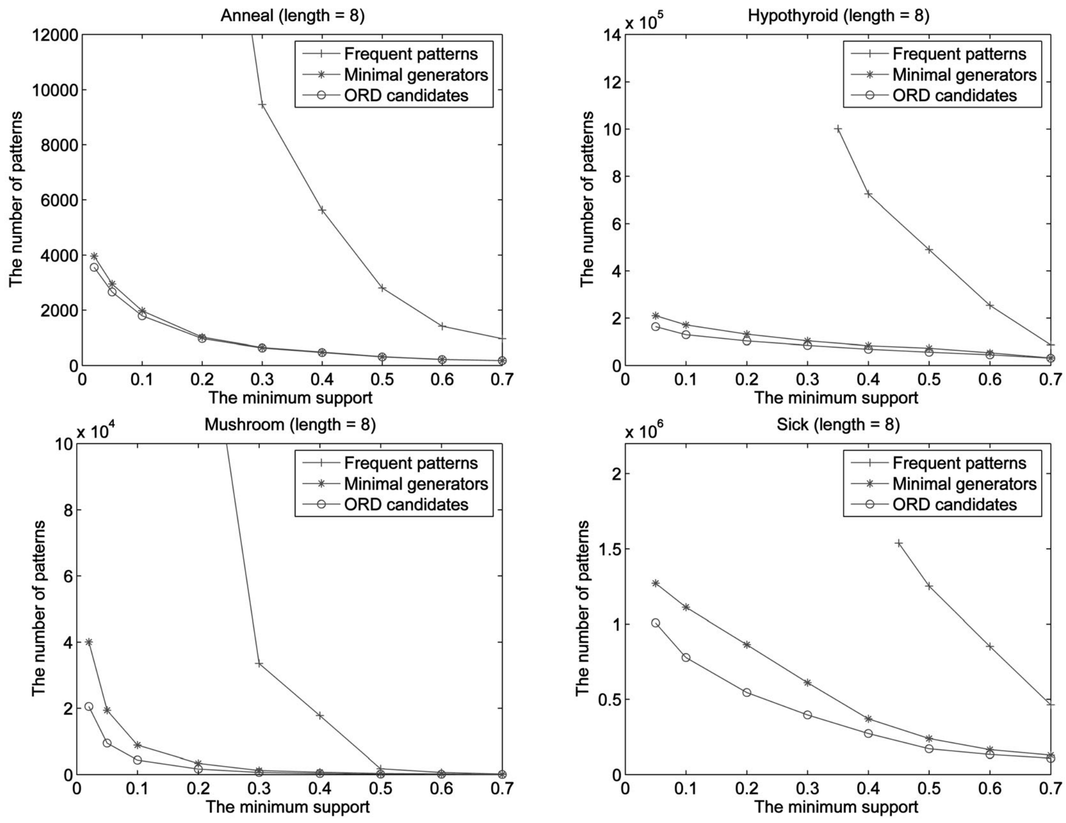

Fig. 6. The number of candidates for optimal rule discovery versus the number of frequent patterns for association rule discovery and the number of frequent minimal generators for nonredundant association rule discovery. The ORD searches a small subset of frequent patterns and a subset of minimal generators.

Therefore, rule $(PX \rightarrow c)$ (when $Q = \emptyset$) and all its more specific rules, for example, $PQX \rightarrow c$ (when $Q \neq \emptyset$), will not occur in the optimal rule set.

The only case for the condition $\mathrm{supp}(PX\neg c) = \mathrm{supp}(P\neg c)$ holding is that $\mathrm{cov}(P\neg c) \subseteq \mathrm{cov}(X\neg c)$. We then deduce that $\mathrm{cov}(PQ\neg c) \subseteq \mathrm{cov}(QX\neg c)$ for any Q. Consequently, $\mathrm{supp}(PQX\neg c) = \mathrm{supp}(PQ\neg c)$ holds for any Q.

For the confidence case, consider $f(y) = y/(y + \alpha)$ monotonically increases with y when constant $\alpha > 0$ and $\mathrm{supp}(PQ) \geq \mathrm{supp}(PQX) > 0$:

$$\begin{aligned}\mathrm{conf(PQ \rightarrow c)} &= \frac{\mathrm{supp}(PQc)}{\mathrm{supp}(PQ)}\\ &= \frac{\mathrm{supp}(PQc)}{\mathrm{supp}(PQc) + \mathrm{supp}(PQ\neg c)}\\ &= \frac{\mathrm{supp}(PQc)}{\mathrm{supp}(PQc) + \mathrm{supp}(PQX\neg c)}\\ &\geq \frac{\mathrm{supp(PQXc)}}{\mathrm{supp}(PQXc) + \mathrm{supp}(PQX\neg c)}\\ &= \mathrm{conf}(PQX \rightarrow c).\end{aligned}$$

We then prove the odds ratio case. Odds ratio is a classic statistical metric to measure the association between events. Consider $f(y) = y/(\alpha - y)$ monotonically increases with y when constant $\alpha > 0$ and $\mathrm{supp}(PQ) \geq \mathrm{supp}(PQX) > 0$:

$$\begin{aligned}\mathrm{or}(PQ \rightarrow c) &= \frac{\mathrm{supp}(PQc)\mathrm{supp}(\neg(PQ)\neg c)}{\mathrm{supp}((\neg(PQ)c)\mathrm{supp}(PQ\neg c)}\\ &= \frac{\mathrm{supp}(PQc)(\mathrm{supp}(\neg c) - \mathrm{supp}(PQ\neg c))}{(\mathrm{supp}(c) - \mathrm{supp}(PQc))\mathrm{supp}(PQ\neg c)}\\ &= \frac{\mathrm{supp}(PQc)(\mathrm{supp}(\neg c) - \mathrm{supp}(PQX\neg c))}{(\mathrm{supp}(c) - \mathrm{supp}(PQc))\mathrm{supp}(PQX\neg c)}\\ &= \frac{\mathrm{supp}(PQc)\mathrm{supp}(\neg(PQX)\neg c)}{(\mathrm{supp}(c) - \mathrm{supp}(PQc))\mathrm{supp}(PQX\neg c)}\\ &\geq \frac{\mathrm{supp}(PQXc)\mathrm{supp}(\neg(PQX)\neg c)}{(\mathrm{supp}(c) - \mathrm{supp}(PQXc))\mathrm{supp}(PQX\neg c)}\\ &= \frac{\mathrm{supp}(PQXc)\mathrm{supp}(\neg(PQX)\neg c)}{\mathrm{supp(\neg(PQX)c)supp(PQX\neg c)}}\\ &= \mathrm{or}(PQX \rightarrow c).\end{aligned}$$

Lift, also known as interest [5] or strength [8], is a widely used metric for ranking the interestingness of association rules. It has been used in IBM Intelligent Miner. We make use of the previous results, $\mathrm{conf}(PQ \rightarrow c) \geq \mathrm{conf}(PQX \rightarrow c)$, in the following proof:

$$\begin{aligned}\mathrm{lift}(PQ \rightarrow c) &= \frac{\mathrm{supp}(PQc)}{\mathrm{supp}(PQ)\mathrm{supp}(c)}\\ &= \frac{\mathrm{conf}(PQ \rightarrow c)}{\mathrm{supp}(c)}\\ &\geq \frac{\mathrm{conf}(PQX \rightarrow c)}{\mathrm{supp}(c)}\\ &= \mathrm{lift}(PQX \rightarrow c).\end{aligned}$$

Gain [9] is an alternative for confidence. Fraction θ is a constant in interval $(0, 1)$ and only rules obtaining positive gain are interesting. We use $\mathrm{conf}(PQ \to c) \geq \mathrm{conf}(PQX \to c) > \theta$ in the following proof:

$$\begin{aligned}\mathrm{gain}(PQ \to c) &= \mathrm{supp}(PQc) - \theta \mathrm{supp}(PQ)\\ &= (\mathrm{conf}(PQ \to c) - \theta)\mathrm{supp}(PQ)\\ &\geq (\mathrm{conf}(PQX \to c) - \theta)\mathrm{supp}(PQX)\\ &= \mathrm{gain}(PQX \to c).\end{aligned}$$

The proofs for metrics added-value,

$$\mathrm{addedvalue}(P \to c) = \mathrm{conf}(P \to c) - \mathrm{supp}(c),$$

and Klosgen,

$$\mathrm{Klosgen}(P \to c) = \sqrt{\mathrm{supp}(Pc)}(\mathrm{conf}(P \to c) - \mathrm{supp}(c)),$$

are very straightforward and, hence, we omit them here.

Conviction [5] is used to measure deviations from independence by considering outside negation:

$$\begin{aligned}\mathrm{conviction}(PQ \to c) &= \frac{\mathrm{supp}(PQ)\mathrm{supp}(\neg c)}{\mathrm{supp}(PQ\neg c)}\\ &= \frac{\mathrm{supp}(PQ)(1 - \mathrm{supp}(c))}{\mathrm{supp}(PQ) - \mathrm{supp}(PQc)}\\ &= \frac{1 - \mathrm{supp(c)}}{1 - \mathrm{conf}(PQ \to c)}\\ &\geq \frac{1 - \mathrm{supp}(c)}{1 - \mathrm{conf}(PQX \to c)}\\ &= \mathrm{conviction(PQX \to c)}.\end{aligned}$$

P-s metric (or leverage), $\mathrm{ps}(P \to c) = \mathrm{supp}(Pc) - \mathrm{supp}(P)\mathrm{supp}(c)$, is a classic interestingness metric for rules proposed by Piatesky-Shaprio [14]. The proof for it is very similar to that of gain and hence we omit it.

Laplace [6], [17] accuracy is a metric for classification rules. $|D|$ is the number of transactions in D and k is the number of classes. In classification problems, $k \geq 2$ and, usually, $\mathrm{conf}(PQX \to c) \geq 0.5$. Therefore, $k \cdot \mathrm{conf}(PQX \to c) \geq 1$ holds. Function $f(y) = (\alpha|D| + y)/(|D| + ky)$ monotonically decreases with y when $k \cdot \alpha > 1$ and $1/\mathrm{supp}(PQX) \geq 1/\mathrm{supp}(PQ)$.

$$\begin{aligned}\mathrm{Laplace}(PQ \to c) &= \frac{\mathrm{supp}(PQc)|D| + 1}{\mathrm{supp}(PQ)|D| + k}\\ &= \frac{\mathrm{conf}(PQ \to c)|D| + 1/\mathrm{supp(PQ)}}{|D| + k/\mathrm{supp}(PQ)}\\ &\geq \frac{\mathrm{conf}(PQX \to c)|D| + 1/\mathrm{supp}(PQ)}{|D| + k/\mathrm{supp}(PQ)}\\ &\geq \frac{\mathrm{conf}(PQX \to c)|D| + 1/\mathrm{supp}(PQX)}{|D| + k/\mathrm{supp}(PQX)}\\ &= \mathrm{Laplace}(PQX \to c).\end{aligned}$$

The proofs for the following two metrics are straightforward and, hence, we omit them:

$$\mathrm{Cosine}(P \to c) = \mathrm{supp}(Pc)/(\sqrt{\mathrm{supp}(P)\mathrm{supp}(c)})$$

and

$$\begin{aligned}&\mathrm{Certaintyfactor}(P \to c) =\\ &\qquad (\mathrm{conf}(P \to c) - \mathrm{supp}(c))/(1 - \mathrm{supp}(c)).\end{aligned}$$

Finally, we prove the metric of Jaccard.

$$\begin{aligned}&\mathrm{Jaccard}(PQ \to c)\\ &= \frac{\mathrm{supp}(PQc)}{\mathrm{supp}(PQ) + \mathrm{supp}(c) - \mathrm{supp}(PQc)}\\ &= \frac{\mathrm{conf}(PQ \to c)}{1 + \mathrm{supp}(c)/\mathrm{supp}(PQ) - \mathrm{conf}(PQ \to c)}\\ &\geq \frac{\mathrm{conf}(PQ \to c)}{1 + \mathrm{supp}(c)/\mathrm{supp}(PQX) - \mathrm{conf}(PQ \to c)}\\ &\geq \frac{\mathrm{conf}(PQX \to c)}{1 + \mathrm{supp}(c)/\mathrm{supp}(PQX) - \mathrm{conf}(PQX \to c)}\\ &= \mathrm{Jaccard}(PQX \to c).\end{aligned}$$

The theorem has been proved. □

Corollary 1 (closure property). *If* $\mathrm{supp}(P) = \mathrm{supp}(PX)$, *then rule* $PX \to c$ *for any* c *and all its more-specific rules do not occur in an optimal rule set defined by confidence, odds ratio, lift (interest or strength), gain, added-value, Klosgen, conviction, p-s (or leverage), Laplace, cosine, certainty factor, or Jaccard.*

Proof. If $\mathrm{supp}(P) = \mathrm{supp}(PX)$, then $\mathrm{supp}(P\neg c) = \mathrm{supp}(PX\neg c)$ holds for any c. Therefore, this Corollary is proved immediately by Theorem 1. □

Corollary 2 (termination property). *If* $\mathrm{supp}(P\neg c) = 0$, *then all more-specific rules of the rule* $P \to c$ *do not occur in an optimal rule set defined by confidence, odds ratio, lift (interest or strength), gain, added-value, Klosgen, conviction, p-s (or leverage), Laplace, cosine, certainty factor, or Jaccard.*

Proof. If $\mathrm{supp}(P\neg c) = 0$, then $\mathrm{supp}(PX\neg c) = \mathrm{supp}(P\neg c) = 0$ holds for any X. Therefore, this corollary is proved immediately by Theorem 1. □

Theorem 2 (relationship with the nonredundant rule set). *An optimal rule set is a subset of a nonredundant rule set.*

Proof. Suppose that we have $\mathrm{supp}(P) = \mathrm{supp}(PX)$ and there is no $P' \subset P$ such that $\mathrm{supp}(P) = \mathrm{supp}(P')$. The rule $PX \to c$ for any c is redundant. It will not be in an optimal rule set either, according to Corollary 1.

Suppose that $\mathrm{supp}(P) = \mathrm{supp}(PX)$ and there is $P' \subset P$ such that $\mathrm{supp}(P) = \mathrm{supp}(P')$. We always have $\mathrm{supp}(P') = \mathrm{supp}(P'Y)$ for $Y \subseteq (PX \backslash P')$. Rules $P \to c$ and $PX \to c$ are redundant. They will not be in an optimal rule set either, according to Corollary 1.

Further, many nonredundant rules are pruned by Theorem 1.

Therefore, an optimal rule set is a subset of a nonredundant rule set. □

ACKNOWLEDGMENTS

The author thanks Professor Tony Roberts, Professor Ada Fu, and Dr. Xiaodi Huang for their constructive comments. This project is supported by ARC (Australia Research Council) grant DP0559090.

REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami, "Mining Associations Between Sets of Items in Massive Databases," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 207-216, 1993.

[2] R. Bayardo and R. Agrawal, "Mining the Most Interesting Rules," *Proc. Fifth ACM SIGKDD Int'l Conf Knowledge Discovery and Data Mining,* pp. 145-154, 1999.

[3] R. Bayardo, R. Agrawal, and D. Gunopulos, "Constraint-Based Rule Mining in Large, Dense Databases," *Data Mining and Knowledge Discovery J.,* vol. 4, nos. 2/3, pp. 217-240, 2000.

[4] E.K.C. Blake and C.J. Merz, UCI Repository of Machine Learning Databases, http://www.ics.uci . edu/~mlearn/MLRepository. html, 1998.

[5] S. Brin, R. Motwani, J.D. Ullman, and S. Tsur, "Dynamic Itemset Counting and Implication Rules for Market Basket Data," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* vol. 26, no. 2, pp. 255-264, 1997.

[6] P. Clark and R. Boswell, "Rule Induction with CN2: Some Recent Improvements in Machine Learning," *Proc. Fifth European Working Session Learning (EWSL '91),* pp. 151-163, 1991

[7] W.W. Cohen, "Fast, Effective Rule Induction," *Proc. 12th Int'l Conf. Machine Learning (ICML),* pp. 115-123, 1995.

[8] V. Dhar and A. Tuzhilin, "Abstract-Driven Pattern Discovery in Databases," *IEEE Trans. Knowledge and Data Eng.,* vol. 5, no. 6, pp. 926-938, Dec. 1993.

[9] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama, "Data Mining Using Two-Dimensional Optimized Association Rules: Scheme, Algorithms, and Visualization," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 13-23, 1996.

[10] H. Hu and J. Li, "Using Association Rules to Make Rule-Based Classifiers Robust," *Proc. 16th Australasian Database Conf. (ADC),* pp. 47-52, 2005.

[11] J. Li, H. Shen, and R. Topor, "Mining the Optimal Class Association Rule Set," *Knowledge-Based Systems,* vol. 15, no. 7, pp. 399-405, 2002.

[12] B. Liu, W. Hsu, and Y. Ma, "Integrating Classification and Association Rule Mining," *Proc. Fourth Int'l Conf. Knowledge Discovery and Data Mining (KDD '98),* pp. 27-31, 1998.

[13] E. Omiecinski, "Alternative Interest Measures for Mining Associations in Databases," *IEEE Trans. Knowledge and Data Eng.,* vol. 15, no. 1, pp. 57-69, Jan. 2003.

[14] G. Piatetsky-Shapiro, "Discovery, Analysis and Presentation of Strong Rules," *Knowledge Discovery in Databases,* G. Piatetsky-Shapiro, ed., pp. 229-248. Menlo Park, Calif.: AAAI Press/The MIT Press, 1991.

[15] J.R. Quinlan, *C4.5: Programs for Machine Learning.* San Mateo, Calif.: Morgan Kaufmann, 1993.

[16] P. Tan, V. Kumar, and J. Srivastava, "Selecting the Right Objective Measure for Association Analysis," *Information Systems,* vol. 29, no. 4, pp. 293-313, 2004.

[17] G.I. Webb, "OPUS: An Efficient Admissible Algorithm for Unordered Search," *J. Artificial Intelligence Research,* vol. 3, pp. 431-465, 1995.

[18] G.I. Webb and S. Zhang, "K-Optimal Rule Discovery," *Data Mining and Knowledge Discovery J.,* vol. 10, no. 1, pp. 39-79, 2005.

[19] M.J. Zaki, "Mining Non-Redundant Association Rules," *Data Mining and Knowledge Discovery J.* vol. 9, pp. 223-248, 2004.

[20] M.J. Zaki and C.J. Hsiao, "Charm: An Efficient Algorithm for Closed Association Rule Mining," *Proc. SIAM Int'l Conf. Data Mining,* 2002.

Jiuyong Li received the BSc degree in physics and MPhil degree in electronic engineering from Yunna University in China in 1987 and 1998 respectively, and received the PhD degree in computer science from Griffith University in Australia in 2002. He is currently a lecturer at the University of Southern Queensland in Australia. His main research interests are in rule discovery and health informatics.