Basic Association Rules

Guichong Li and Howard J. Hamilton
Department of Computer Science
University of Regina
Regina, SK, Canada, S4S 0A2
{liguicho, hamilton}@cs.uregina.ca

Abstract Previous approaches for mining association rules generate large sets of association rules. Such sets are difficult for users to understand and manage. Here, the concept of a restricted conditional probability distribution is used to explain an association rule. Based on this concept, a new type of association rules, called basic association rules, is defined. We propose the GenBR algorithm to generate the set of classes of basic association rules. Theoretical analysis shows that the search space of the algorithm can be translated to an *n*-cube graph. The set of classes of basic association rules generated by GenBR is easy for users to understand and manage. Our experiments on synthetic and real datasets show that GenBR is either faster than previous approaches or generates fewer rules or both.

Keywords: Association Rules, Probabilistic/Statistical Methods, Data Reduction/Preprocessing, Restricted Conditional Probability Distributions, Inference Systems, and Basic Association Rules.

1 Introduction

Techniques for mining association rules [1,2] were originally devised for application to market basket data, but they have also been applied in many other domains to perform tasks [21,23,26]. *Market basket data* describes the items purchased from retail stores grouped into transactions. A *transaction* typically consists of items bought together at the same point of time, but it may consist of items bought by a customer over a period of time. An *itemset* is a set of items, and a *frequent itemset X* is an itemset whose frequency in transactions, also referred to as its *support*, denoted as supp(X), is greater than a user specified support threshold, *minsup*.

The main task of association rule discovery is to extract frequent itemsets from market basket data and to generate association rules from these frequent itemsets. An association rule r is an implication of the form $X \rightarrow Y$, where X and Y are two disjoint itemsets. The support of the rule is the support of $X \cup Y$, denoted as supp(r), which is given by the observed probability $P(X = 1, Y = 1)$. The confidence of the rule, denoted conf(r), is given by the conditional observed probability $P(X = 1, Y = 1) / P(X = 1)$, which is denoted as $p(xy) / p(x)$ in this paper. If an association rule has support at least as great as *minsup* and confidence at least as great as the confidence threshold called *minconf*, it is referred to as a *valid association rule*. An association rule with confidence 100% is an *exact association rule*; all other association rules are *approximate association rules*.

The Apriori algorithm [2] was proposed to discover all frequent itemsets and to generate all valid association rules corresponding to these itemsets by a fast algorithm, called FastGenRules. Many algorithms have since been proposed that reduce the time and space required to find the frequent itemsets [2,14]. After all frequent itemsets have been found, valid association rules are generated.

A serious problem in association rule discovery is that the set of association rules can grow to be unwieldy as the number of transactions increases, especially if the support and confidence thresholds are small. As the number of frequent itemsets increases, the number of rules presented to the user typically increases proportionately. Many of these rules may be redundant. The definition of "redundancy" for association rules has varied in previous approaches. Toivonen et al. proposed finding a structural rule cover, which describes the same database rows as the original set of association rules [28]. Therefore, those rules that are not in the cover are regarded as redundant. In [11,20,24,30], the definition of redundant rules is based on several inference rules or an inference system. Therefore, all association rules that can be derived from other rules by applying inference rules are regarded as redundant. We adopt the latter type of definition.

To address the problem of rule redundancy, four types of research on mining association rules have been performed. First, rules have been extracted based on user-defined templates or item constraints [3,27]. Secondly, researchers have developed interestingness measures to select only interesting rules [16,18,19]. Thirdly, researchers have proposed inference rules or inference systems to prune redundant rules and thus present smaller, and usually more understandable sets of association rules to the user [5,11,20,24,30]. Finally, new frameworks for mining association rule have been proposed that find association rules with different formats or properties [7,8,9].

The main problems with previous approaches are that they still generate too many rules, and these rules may be

redundant. For example, a valid association rule $X \rightarrow Y$ that is generated by one these approaches may in fact be derived from some simpler rule $X' \rightarrow Y'$ with the same confidence as $X \rightarrow Y$, where $X' \subseteq X$ and $Y' \subseteq Y$. Inference rules proposed by these approaches do not resemble Armstrong axioms on functional dependencies. As well, in some approaches, inference rules cannot infer the confidence of rules without extra information.

In our research, we are creating an inference system on association rules, consisting of a set of inference rules such as augmentation and transitivity, which resembles the Armstrong axioms on functional dependencies and which allows the inference of the confidences of rules.

The remainder of this paper is organized as follows. In Section 2, we present related work. In Section 3, we define the concept of a basic association rule, and propose a new algorithm called GenBR for generating the set *BR* of classes of basic association rules from a set of frequent itemsets. The computational complexity of GenBR is also described. A comparison of our approach and other approaches is presented in Section 4. Our experiments compared the performance of our algorithm with that of previous algorithms on synthetic datasets and real-life datasets, with respect to the number of rules and the elapsed running time. Conclusions and future work are described in Section 5.

2 Previous Work

Previous research showed that relatively small sets of association rules can be presented to users instead of all valid association rules. As well, for some approaches, inference rules were suggested that allowed additional association rules to be derived from such small sets of rules. In this section, we describe three approaches.

First, *representative association rules* (*RR*) are based on a cover operator with which other non-representative association rules can be generated [20]. Suppose we have an association rule $X \rightarrow Y$. A *cover operator* C, denoted $C(X \rightarrow Y)$, is given by

$$C(X \rightarrow Y) = \{X \cup Z \rightarrow V \mid Z, V \subseteq Y \wedge Z \cap V = \varnothing \wedge V \neq \varnothing\}$$

The set of all representative association rules is a minimal set of rules that covers all association rules by means of the cover operator. The FastGenRepresentative algorithm was proposed to efficiently compute a *RR* [20].

Secondly, a kind of non-redundant association rules with minimal antecedents and maximal consequents, called minimal non-redundant association rules, has been identified as particularly useful and relevant [5]. An association rule r: $X \rightarrow Y$ is a *minimal non-redundant association rule* iff there does not exist an association rule r': $X' \rightarrow Y'$ with $\text{supp}(r) = \text{supp}(r')$, $\text{conf}(r) = \text{conf}(r')$, $X' \subseteq X$ and $Y \subseteq Y'$. A small non-redundant generating set for all valid association rules is formed by combining a generic basis *GB* for exact association rules and an informative basis *IB* for approximate association rules. *RI* is defined as a transitive reduction of the informative basis corresponding to *IB*. Given a closure operator c of the Galois connection, a set *FC* of frequent closed itemsets, the set G of their generators, and a partial order $\prec$ (inclusion relation) on the set of itemsets, the definitions of *GB*, *IB* and *RI* are as follows.

$$GB = \{r: g \rightarrow (f \setminus g) \mid f \in FC \wedge g \in G_f \wedge g \neq f\}$$
$$IB = \{r: g \rightarrow (f \setminus g) \mid f \in FC \wedge g \in G \wedge c(g) \subset f\}$$
$$RI = \{r: g \rightarrow (f \setminus g) \mid f \in FC \wedge g \in G \wedge c(g) \prec f \wedge \nexists f'\ c(g) \prec f' \prec f\}$$

Bastide et al. have proven that *GB* and *IB* contain only minimal non-redundant association rules and all exact association rules and approximate association rules can be derived from *GB* and *IB*, respectively [5]. The Gen-GB and Gen-RI algorithms were proposed to generate a generic basis and a transitive reduction of the informative basis, respectively. According to the definition of a minimal non-redundant association rule, the support and confidence of any association rules inferred from the generating set are the same as the support and confidence of the rules from which they were inferred. The authors claim that none of the Armstrong axioms hold in non-redundant association rules. A similar approach has been proposed for discovering a small cover for association rules based on closed itemsets, which adapts the Duquenne-Guigues basis for exact association rules and the Luxenburger results for approximate association rules [24].

Thirdly, informative cover has been proposed together with a new inference rule [11]. Let r, r' be two association rules, denoted $X \rightarrow Y$ and $X' \rightarrow Y'$, such that $X' \cup Y' \subseteq X \cup Y$. If $\text{supp}(X') \leq \text{supp}(X)$, we say that *r covers* r', denoted $r \prec r'$. The goal is to find an *informative cover* that covers all other association rules. The CoverRules algorithm has been proposed to generate an informative cover for association rules [11].

The cover operator in the informative cover approach is similar to the cover operator in the representative association rule approach. The difference between them is that the cover operator of the informative cover approach does not require the antecedent of the resulting association rule to be included in the antecedent of the initial association rule. In addition, the inference procedure is not purely syntactic [11], because it uses

Table 2.1. A Binary Dataset.

A	*B*	*C*	*D*	*E*
1	0	1	1	1
0	0	1	1	1
1	0	1	1	1
1	1	1	1	0
0	1	0	1	1
1	1	0	1	0

information about the support of the antecedent of the resulting association rule. The inference rules in the two approaches are sound, but neither approach infers the confidence of an association rule.

We found that the rules generated by these approaches may be redundant. If $X \rightarrow Y$ is generated by any of these approaches, it may be possible to derive it from simpler valid association rules.

Example 2.1. Suppose we have the dataset shown in Table 2.1, and that *minsup* is 0.3 and *minconf* is 0.6. The sets of rules generated by the previous approaches are shown in Table 2.2.

Consider the rules in Table 2.2 from the perspective of a user. For *RR*, $CE \rightarrow AD$ can be derived from simpler association rules $CE \rightarrow A$ and $CE \rightarrow D$ by right union as with Armstrong axioms, and $\text{conf}(CE \xrightarrow{0.67} AD) = \text{conf}(CE \xrightarrow{0.67} A) \times \text{conf}(CE \xrightarrow{1.0} D)$. For GB, $AB \rightarrow D$ is unnecessary, because $B \rightarrow D$ is in *GB*. For *C*, $B \rightarrow AD$ can be derived from $B \rightarrow A$ and $B \rightarrow D$ by right union, assuming $B \rightarrow A$ and $B \rightarrow D$ are valid association rules. Transitivity cannot be used between *GB* and *RI*. For example, $E \rightarrow D$ in *GB* and $D \rightarrow A$ in *RI* cannot be used to infer a valid association rule by transitivity. These examples show that some association rules in these generating sets are not in the most desirable form. □

3 Discovery of Basic Association Rules

We propose a new approach to solve the problems mentioned in Section 2.

Table 2.2. The Generated Association Rules

Algorithm	Set	Rule	#Rule
FastGenRules	*AR*	…	36
FastGenRR	*RR*	$\mathbf{CE} \xrightarrow{0.67} \mathbf{AD}$, $C \xrightarrow{0.75} AD$, $D \xrightarrow{0.67} E$, $D \xrightarrow{0.67} C$, $D \xrightarrow{0.67} A$, $B \xrightarrow{0.67} AD$, $AC \xrightarrow{0.75} DE$, $C \xrightarrow{0.75} DE$, $AE \xrightarrow{1.0} CD$, $E \xrightarrow{0.75} CD$, $A \xrightarrow{0.75} CD$	11
Gen-GB and Gen-RI	*GB*	$C \xrightarrow{1.0} D$, $\mathbf{E} \xrightarrow{1.0} \mathbf{D}$, $\mathbf{AB} \xrightarrow{1.0} \mathbf{D}$, $E \xrightarrow{1.0} CD$, $CE \xrightarrow{1.0} D$, $\mathbf{B} \xrightarrow{1.0} \mathbf{D}$, $AC \xrightarrow{1.0} D$, $A \xrightarrow{1.0} D$	8
	RI	$CE \xrightarrow{0.67} AD$, $C \xrightarrow{0.75} AD$, $D \xrightarrow{0.67} E$, $D \xrightarrow{0.67} C$, $\mathbf{D} \xrightarrow{0.67} \mathbf{A}$, $B \xrightarrow{0.67} AD$, $AC \xrightarrow{0.75} DE$, $C \xrightarrow{0.75} DE$, $E \xrightarrow{0.75} CD$, $A \xrightarrow{0.75} CD$	10
CoverRules	*C*	$E \xrightarrow{0.75} CD$, $D \xrightarrow{0.67} E$, $C \xrightarrow{0.75} AD$, $D \xrightarrow{0.67} C$, $\mathbf{B} \xrightarrow{0.67} \mathbf{AD}$, $CE \xrightarrow{0.67} AD$, $D \xrightarrow{0.67} A$	7

3.1 Definitions

Definition 3.1.1. Given a dataset D with I as a set of items and T as a set of transactions, an association rule $X \xrightarrow{p} Y$ over a relation $R \subseteq I \times T$ is said to be in *canonical form* if $|Y| = 1$.

According to this definition, we only consider the case of Y containing a single item. Before we introduce other new notions, let us discuss another concept related to conditional probability.

A *conditional probability distribution* (*CPD*) $P(Y \mid X)$ is defined as $P(X, Y) / P(X)$, where X and Y are random variables [10]. Y is *conditionally independent* of Z given X, denoted as $I(Y, Z \mid X)$, if and only if $P(Y \mid X) = P(Y \mid X, Z)$, where X, Y, and Z are three disjoint sets of random variables. The statement $I(X, Z \mid X)$ is referred to as a *conditional independence statement* (*CIS*) [10].

Four properties that are satisfied by any joint probability distribution (*JPD*) are symmetry, decomposition, weak union, and contraction [10]. For example, for the decomposition property, if $I(X, Y \cup W \mid Z)$, then $I(X, Y \mid Z)$ and $I(X, W \mid Z)$.

We describe a *CIS* in another way in Lemma 3.1.1.

Lemma 3.1.1. Given a subset $X' \subset X$, we say $I(Y, X \setminus X' \mid X')$, i.e., Y is conditionally independent of $X \setminus X'$ given X', if and only if $P(Y \mid X) = P(Y \mid X')$, where X and Y are two disjoint sets of variables.

Proof:

Suppose $Z = X \setminus X'$. Then X', Y, and Z are three disjoint sets of variables. Since $X = X' \cup Z$, $P(Y \mid X) = P(Y \mid X', Z)$. Assuming $P(Y \mid X) = P(Y \mid X')$, we derive $P(Y \mid X', Z) = P(Y \mid X')$. Thus, we have $I(Y, Z \mid X')$, i.e., $I(Y, X \setminus X' \mid X')$ according to the definition of *CIS*.

For the converse, assuming $I(Y, X \setminus X' \mid X')$, then according to the definition of *CIS*, we have $P(Y \mid X', X \setminus X') = P(Y \mid X')$, i.e., $P(Y \mid X) = P(Y \mid X')$. □

Furthermore, we have Lemma 3.1.2.

Lemma 3.1.2. Given two disjoint sets of variables X and Y, and $X' \subset X$, if $I(Y, X \setminus X' \mid X')$, then $\forall Z \subseteq X \setminus X'$, $I(Y, Z \mid X')$, i.e., $\forall Z \subseteq X \setminus X'$, $P(Y \mid X) = P(Y \mid X') = P(Y \mid X', Z)$.

Proof:

If $Z = X \setminus X'$, the proof follows immediately. The decomposition property of conditional independencies states that if $I(X, Y \cup W \mid Z)$, then $I(X, Y \mid Z)$ and $I(X, W \mid Z)$. Because $I(Y, X \setminus X' \mid X')$, $\forall Z \subset X \setminus X'$, we obtain $I(Y, Z \mid X')$ and $I(Y, \{X \setminus X'\} \setminus Z \mid X')$. From $I(Y, X \setminus X' \mid X')$, we obtain $P(Y \mid X) = P(Y \mid X', X \setminus X') = P(Y \mid X')$, and from $I(Y, Z \mid X')$, we obtain $P(Y \mid X') = P(Y \mid X', Z)$. Hence, $P(Y \mid X) = P(Y \mid X') = P(Y \mid X', Z)$. □

To explain this idea more fully, we first give a more general definition as follows.

Definition 3.1.2. Given two disjoint sets of variables X and Y, and a conditional probability distribution $P(Y \mid X)$, a *restricted conditional probability distribution* (*RCPD*),

denoted as $\hat{P}(\hat{Y} \mid \hat{X})$, is a subset of $P(Y \mid X)$ defined by specifying subsets $Domain(\hat{Y}) \subseteq Domain(Y)$ and $Domain(\hat{X}) \subseteq Domain(X)$.

For binary variables $X \in \{0,1\}$ and $Y \in \{0,1\}$, with $\hat{X} \in \{1\}$ and $\hat{Y} \in \{1\}$, the confidence $p(y \mid x)$ of the association rule r: $X \rightarrow Y$ is a *positive conditional probability* of the *RCPD* $\hat{P}(\hat{Y} \mid \hat{X})$.

In the following discussion, $\hat{P}(\hat{Y} \mid \hat{X})$ is simply denoted as $\hat{P}(Y \mid X)$.

Given $X' \subset X$, if we have $\hat{P}(Y \mid X) = \hat{P}(Y \mid X')$, we cannot guarantee that $\forall Z \subseteq X \setminus X'$, $\hat{P}(Y \mid X) = \hat{P}(Y \mid X', Z)$. Hence, the decomposition property cannot be applied in a RCPD.

Example 3.1.1. Suppose that we have the transaction dataset in Table 2.1 and that *minsup* is 0.3 and *minconf* is 0.6. Consider two association rules $ACD \rightarrow E$ and $D \rightarrow E$. Although $conf(ACD \rightarrow E) = conf(D \rightarrow E) = 2/3$, $conf(CDE) = 3/4$, i.e., if $X = \{A, C, D\}$ and $Y = \{E\}$. If we choose $X' = \{D\}$, then $Z = X \setminus X' = \{A, C\}$, $p(y \mid x) = p(y \mid x', z) = p(y \mid x') = 2/3$, i.e., $\hat{P}(Y \mid X) = \hat{P}(Y \mid X')$. Let $Z = \{C\} \subset X \setminus X'$, since $p(y \mid x', z) = 3/4$, then $p(y \mid x) \neq p(y \mid x', z)$ and $p(y \mid x') \neq p(y \mid x', z)$, i.e., $\hat{P}(Y \mid X) \neq \hat{P}(Y \mid X', Z)$ and $\hat{P}(Y \mid X') \neq \hat{P}(Y \mid X', Z)$. □

In the context of RCPDs, we hope to find a minimal subset X' of X with respect to Y such that $\hat{P}(Y \mid X) = \hat{P}(Y \mid X')$, and $\forall Z \subseteq X \setminus X'$, $\hat{P}(Y \mid X) = \hat{P}(Y \mid X', Z)$.

Definition 3.1.3. Let X and Y be two disjoint sets of variables. X' is a *minimal conditional subset* (*MCS*) of X with respect to Y if X' is a subset of X that satisfies the following conditions:

(1) $\hat{P}(Y \mid X) = \hat{P}(Y \mid X')$.

(2) $\forall Z \subseteq X \setminus X'$, $\hat{P}(Y \mid X) = \hat{P}(Y \mid X', Z)$.

(3) $\nexists X'' \subset X'$ such that conditions (1) and (2) hold for X''.

If X is a minimal conditional subset of itself with respect to Y, then X is *conditionally minimal* with respect to Y. For example, in Table 2.1, AC is conditionally minimal with respect to E.

We also define the dual.

Definition 3.1.4. Suppose X and Y are disjoint sets of variables. If X' is a MCS of X with respect to Y, then the restricted conditional probability distribution $\hat{P}(\hat{Y} \mid \hat{X}')$ is a *minimal conditional probability distribution* (*MCPD*) of the RCPD $\hat{P}(\hat{Y} \mid \hat{X})$ with respect to $\hat{Y}$. If X is conditionally minimal with respect to Y, then $\hat{P}(\hat{Y} \mid \hat{X})$ is a MCPD of itself.

If $\hat{P}(\hat{Y} \mid \hat{X}')$ is a MCPD of $\hat{P}(\hat{Y} \mid \hat{X})$, then a series of equations follow, i.e., $\forall Z \subseteq X \setminus X'$, $\hat{P}(\hat{Y} \mid \hat{X}) = \hat{P}(\hat{Y} \mid \hat{X}', \hat{Z})$.

Because MCS and MCPD are duals of each other, we use whichever is convenient.

In previous research, MCS and MCPD have not been defined or emphasized by researchers. In the context of mining association rules, we use a RCPD $\hat{P}(Y \mid X)$ for inference on association rules. We do not consider how $\hat{P}(XY)$ and $\hat{P}(X'Y)$ behave.

Example 3.1.2. Suppose we have the transaction dataset shown in Table 2.1. Let $X = \{A, C, D\}$, $Y = \{E\}$, $X' = \{A, C\}$. Because the confidences of both $X \rightarrow Y$ and $X' \rightarrow Y$ are 2/3, i.e., the positive conditional probability $p(e \mid acd) = p(e \mid ac)$, we see that $\hat{P}(Y \mid X) = \hat{P}(Y \mid X')$ and that $\forall Z \subseteq X \setminus X' = \{D\}$, $\hat{P}(Y \mid X) = \hat{P}(Y \mid X', Z)$. Because $p(e \mid acd) \neq p(e \mid a) = 2/4$ and $p(e \mid acd) \neq p(e \mid c) = 3/4$, there is no $X'' \subset X' = \{A, C\}$ such that $\hat{P}(Y \mid X) = \hat{P}(Y \mid X'')$. So X' is a MCS of X with respect to Y. $\hat{P}(Y \mid X')$ is a MCPD of $\hat{P}(Y \mid X)$ with respect to Y.

Similarly, $\{A\}$, $\{C\}$, and $\{E\}$ are three MCSs of $\{A, C, E\}$ with respect to D. □

In the context of mining association rules, a CPD $P(Y \mid X)$ is always referred to as a RCPD $\hat{P}(\hat{Y} \mid \hat{X})$.

We define a new notion of minimal association rules analogously to minimal functional dependencies [25].

Definition 3.1.5. Suppose we have a set I of items and a transaction dataset D. A canonical association rule $X \xrightarrow{p} Y$ over D is a *basic association rule* if X is conditionally minimal with respect to Y, i.e., $\nexists X' \subset X$ such that

(1) $P(Y \mid X) = P(Y \mid X')$ and

(2) $\forall Z \subseteq X \setminus X'$, $P(Y \mid X) = P(Y \mid X', Z)$.

For example, given the transaction dataset shown in Table 2.1, $AC \rightarrow E$ is a basic association rule, and AC is a MCS of ACD with respect to E while $P(E \mid AC)$ is a MCPD of $P(E \mid ACE)$ with respect to E.

3.2 Computing MCPDs

According to the definition of basic association rules, either a MCS X with respect to Y or a MCPD $P(Y \mid X)$ corresponds to a basic association rule $X \rightarrow Y$. The confidence of the rule is a positive conditional probability of $P(Y \mid X)$. Therefore, the crucial task of finding basic association rules is the computation of all MCPDs.

Given a set L of frequent itemsets, $\forall X \in L$, and *minconf*, our approach for computing MCPDs corresponding to X is divided into two steps. We first construct a set of RCPDs in canonical form from X, and then we compute their MCPDs, in which all positive conditional probabilities are at least as great as *minconf*.

Our approach is similar to the approach for discovering the minimal directed *I*-Map of a joint probability distribution (JPD) [10]. Suppose that a permutation (ordering) $Y = \{Y_1, \ldots, Y_n\}$ of a set of variables $X = \{X_1, \ldots, X_n\}$, and $p(x)$ is a JPD of X. This approach computes any minimal set of predecessors $\prod_i$ with respect

to Y_i, and $\prod_i$ satisfies $p(y_i \mid b_i) = p(y_i \mid \pi_i)$, where $\prod_i \subseteq B_i = \{Y_1,\ldots,Y_{i-1}\}$. Hence, a directed minimal *I*-map of $p(x)$ is constructed by designating $\prod_i$ as parents of Y_i. The differences from computing *I*-map from a JPD is that we do not permutate items of a frequent itemset, and the conditions (contexts) in the restricted conditional probabilities contain all items in the frequent itemset except for a single test item.

For example, to find MCPDs corresponding to the frequent itemset $X_1X_2X_3X_4X_5$, first the following RCPDs are constructed:

$P(X_1 \mid X_2X_3X_4X_5)$, $P(X_2 \mid X_1X_3X_4X_5)$, $P(X_3 \mid X_1X_2X_4X_5)$, $P(X_4 \mid X_1X_2X_3X_5)$, and $P(X_5 \mid X_1X_2X_3X_4)$.

For $P(X_1|X_2X_3X_4X_5)$, one can observe all MCSs of $X_2X_3X_4X_5$ with respect to X_1, such as a minimal subset $\prod \subseteq \{X_2X_3X_4X_5\}$, and $\forall Z \subseteq \{X_2X_3X_4X_5\} \setminus \prod$, $p(x_1 \mid x_2x_3x_4x_5) = p(x_1 \mid \pi) = p(x_1 \mid \pi, z)$, where $P(X_1 \mid \prod)$ is a MCPD of $P(X_1 \mid X_2X_3X_4X_5)$. If $p(x_1 \mid x_2x_3x_4x_5)$ is less than *minconf*, then we stop the computation of MCPDs. Otherwise, $\prod$ is a MCS of $X_2X_3X_4X_5$ with respect to X_1. The corresponding basic association rule is $\prod \xrightarrow{p} X_1$, as explained in Section 3.1. Similarly, we compute the MCPDs of $P(X_2 \mid X_1X_3X_4X_5)$, $P(X_3 \mid X_1X_2X_4X_5)$, $P(X_4 \mid X_1X_2X_3X_5)$, and $P(X_5 \mid X_1X_2X_3X_4)$. Finally, a set of MCPDs is obtained. Thus, for each frequent itemset, a set of basic association rules with respect to *X* can be found.

Because RCPDs do not obey decomposition, to compute a MCPD $P(Y \mid X')$ of $P(Y \mid X)$, we should examine all cases, i.e., $\forall Z \subseteq X \setminus \{Y \cup X'\}$, $P(Y \mid X') = P(Y \mid X', Z)$. This process goes from top to bottom, and is depicted as a semi-lattice [12] in Figure 3.1.

The frequent itemset itself, such as ABCDE, is placed at the first level. At the second level, we construct a set of RCPDs, such as $P(A \mid BCDE)$, $P(B \mid ACDE)$, $P(E \mid ABCD)$, $P(D \mid ABCE)$, and $P(E \mid ABCD)$. All RCPDs at the third level are constructed by setting their contexts as maximal subsets of the contexts of the RCPDs at the second level. At the fourth level, all RCPDs are formed by setting their contexts as the intersections of the contexts of the RCPDs at the third level. The number *k* of itemsets being intersected at level *d* of the semi-lattice is related to the length *l* of the frequent itemset at the first level. We have the formula $k = d - 2$, $4 \le d \le l$. For example, for frequent 4-itemsets, the number of itemsets being intersected at level 4 is 2. For frequent 5-itemsets, the number of itemsets being intersected at level 5 is 3. The depth of the semi-lattice equals the size of the corresponding frequent itemset.

For example, for the computation of the MCPDs of $P(E \mid ABCD)$, we first examine $P(E \mid ABC)$, $P(E \mid ABD)$, $P(E \mid ACD)$, and $P(E \mid BCD)$. If $P(E \mid ABCD) = P(E \mid ABC)$ and $P(E \mid ABC)$ is minimal, then $P(E \mid ABC)$ is already a MCPD of $P(E \mid ABCD)$, and we examine $P(E \mid ABD)$, $P(E \mid ACD)$, and $P(E \mid BCD)$. Similar cases arise

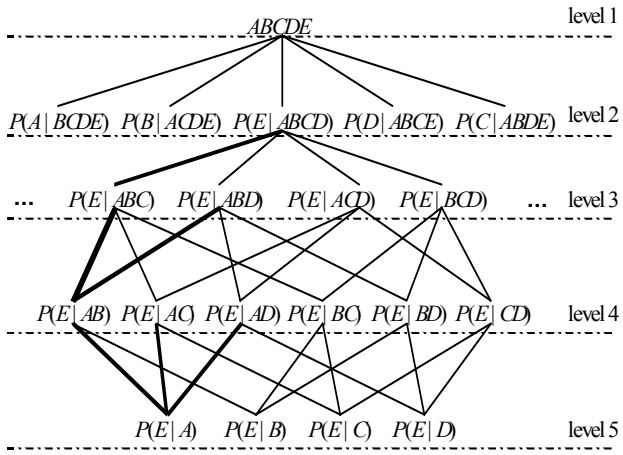

Figure 3.1. A Possible Semi-lattice for the Frequent Itemset ABCDE.

for $P(E \mid ABD)$, $P(E \mid ACD)$, and $P(E \mid BCD)$. If $P(E \mid AB)$ is a MCPD of $P(E \mid ABCD)$, we only require $P(E \mid ABCD) = P(E \mid ABC) = P(E \mid ABD)$. The context of $P(E \mid AB)$ is the intersection of the contexts of $P(E \mid ABC)$ and $P(E \mid ABD)$. If $P(E \mid A)$ is a MCPD of $P(E \mid ABCD)$, we require $P(E \mid AB) = P(E \mid AC) = P(E \mid AD) = P(E \mid ABCD)$, where $P(E \mid AB)$, $P(E \mid AC)$, and $P(E \mid AD)$ have already been obtained. Therefore, according to Definition 3.1.3 and 3.1.4, $P(E \mid A)$ is a MCPD of $P(E \mid ABCD)$.

During extension of the semi-lattice, the RCPDs in new *children* (the nodes at the next level) are also called the *candidate MCPDs* (not unique). We reduce the number of candidate MCPDs of new children by intersecting itemsets in the contexts of their *parents* (nodes at the previous level), and checking whether the candidate MCPDs of the children are equal to those of their parents, and whether the positive conditional probabilities of their MCPDs are at least as great as *minconf*.

To find all basic association rules, we should check all frequent itemsets and compute the corresponding minimal conditional probabilities.

Therefore, we define the following concepts.

Definition 3.2.1. Suppose we have a transaction dataset *D*, *minsup*, *minconf*, and a set *L* of frequent itemsets over *D*. A *class of basic association rules* for $X \in L$ is a set of basic association rules $r: X' \rightarrow A$, denoted as $C_r(X)$, such that

(1) $X' \cup A \subseteq X$
(2) X' is a MCS of $X \setminus A$ with respect to A
(3) $\text{conf}(r) \ge minconf$

Definition 3.2.2. Suppose we have a transaction database *D*, *minsup*, and *minconf*. A *basic association rule system*, denoted as *BR*(*D*, *minsup*, *minconf*), is defined as the set of *k* distinct classes of valid basic association rules, i.e., $BR(D, minsup, minconf) = \{C_r^i \mid C_r^i$ is a class of basic

association rules, $1 \le i \le k$ such that for all j, $1 \le j \le k, j \ne i$, $C_r^i \not\subseteq C_r^j$, $C_r^i \not\supseteq C_r^j$}. $BR(D, minsup, minconf)$ is also simply denoted as BR when D, $minsup$, and $minconf$ are clear from context.

Example 3.2.1. Given the dataset in Table 2.1, $minsup$ = 0.3, and $minconf$ = 0.6, in Figure 3.2, we show how to compute all MCPDs corresponding to the frequent itemset $ACDE$.

Figure 3.2 shows the search space used to find MCSs by computing a set of MCPDs corresponding to the frequent itemset $ACDE$. This search space consists of four semi-lattices, in which the single node at the top level corresponds to the frequent itemset, and other nodes correspond to its subsets, each of which includes a positive conditional probability.

Regardless of the data in the dataset, at the second level in the structure, we always have four positive conditional probabilities for $ACDE$. Each of them corresponds to an item in $ACDE$, such as $p(a \mid cde)$, etc. At the third level of the structure, the itemsets appearing in the contexts of positive conditional probabilities are always maximal subsets of the itemsets appearing in the context of positive conditional probabilities in their parents. For example, along the branch containing $p(a \mid cde)$, de, ce and cd are maximal subsets of cde. If the positive conditional probability of a child is equal to the positive conditional probability of its parent and both positive conditional probabilities are at least as great as $minconf$, then in Figure 3.2, they are connected with a bold arrow; e.g., a bold arrow is shown from the node including $p(a \mid cde)$ to the node including $p(a \mid ce)$, because $p(a \mid cde) = p(a \mid ce)$. If the positive conditional probability of a parent is not equal to the positive conditional probability of its child, but the positive conditional probability of the child is at least as great as $minconf$, we connect the parent node and the child node with a narrow arrow; e.g., a narrow arrow is shown from the node including $p(a \mid cde)$ to the node including $p(a \mid cd)$. If the positive conditional probability of a parent is not equal to the positive conditional probability of its child or a positive conditional probability is less than $minconf$, further computation of minimal conditional probabilities along this path is terminated; e.g., a dotted arrow is shown from the node including $p(a \mid cde)$ to the node including $p(a \mid de)$. Hence, $P(A \mid CE)$ is a MCPD of $P(A \mid CDE)$.

Similarly, we compute all MCPDs of $P(C \mid ADE)$, $P(D \mid ACE)$ and $P(E \mid ACD)$. As a result, a set of MCPDs with respect to the frequent itemset $ACDE$ is obtained, i.e., $P(A \mid CE)$, $P(C \mid AE)$, $P(D \mid E)$, $P(D \mid C)$, $P(D \mid A)$ and $P(E \mid AC)$, where $P(A \mid CE)$ is a MCPD of $P(A \mid CDE)$ with respect to A, and $P(C \mid AE)$ is a MCPD of $P(C \mid ADE)$ with respect to C, etc. □

From the MCPDs corresponding the frequent itemset X, we can readily obtain a class of basic association rules, $C_r(X)$. A class of basic association rules derived from one frequent itemset may be completely included in another class of basic association rules derived from another frequent itemset. Hence, classes of basic association rules that are completely contained in other classes of basic association rules have no more information than the classes containing them. They are called *redundant* classes and are discarded.

Example 3.2.2. Given the transaction dataset in Table 2.1, $minsup$ = 0.3, and $minconf$ = 0.6, from Example 3.2.1, we obtain $C_r(ACDE)$ = {$CE \rightarrow A$, $AE \rightarrow C$, $E \rightarrow D$, $C \rightarrow D$, $A \rightarrow D$, $AC \rightarrow E$}. Similarly, we also obtain another class of basic association rules corresponding to the frequent itemset ADE, $C_r(ADE)$ = {$A \rightarrow D$, $E \rightarrow D$}. Since $C_r(ADE) \subset C_r(ACDE)$, $C_r(ADE)$ is discarded. □

3.3 The GenBR Algorithm

We propose the GenBR algorithm for generating BR. Its goal is different from that of the second step of the Apriori algorithm [1], which generates all association rules. Our approach consists of two main steps. Given a set of frequent itemsets and $minconf$, GenBR generates all classes of basic association rules. Secondly, the algorithm generates BR by discarding all redundant classes.

The GenBR algorithm, presented in Figure 3.3, generates BR from a set of frequent itemsets L. For each frequent itemset I in L, the GenBC algorithm is called to generate a class of basic association rules corresponding to I. All classes discovered by GenBC are collected into

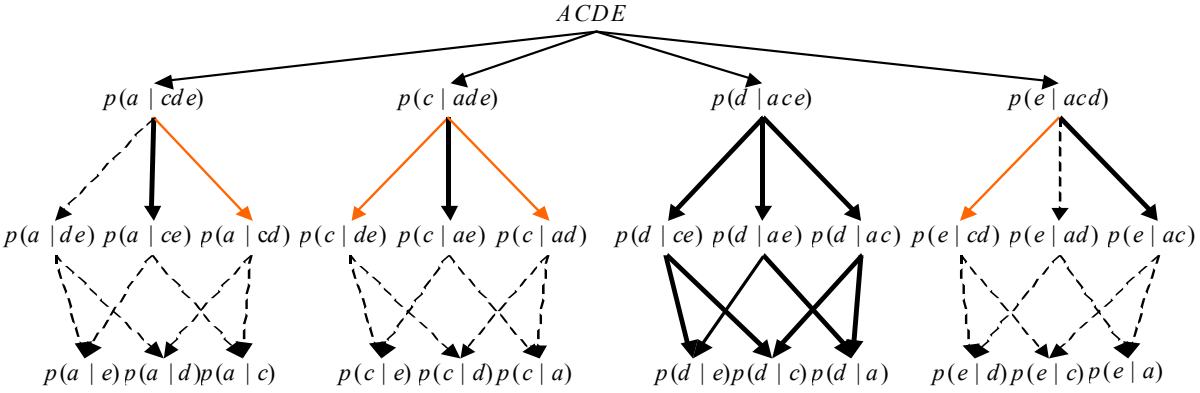

Figure 3.2. The Computation of MCPDs.

Algorithm GenBR(L)
Purpose: generate BR from a set L of frequent itemsets
Input: L, a set of frequent itemsets, where $L^k \subseteq L$ is all itemsets in L containing k items.
Output: BR, a set of classes of basic association rules.
begin
  $BAR = \emptyset$
  foreach $I \in L^k$, $k \geq 2$
  begin
    $BAR = BAR \cup$ GenBC(I)
  end
  BR = **RemoveRedundantClass**(BAR)
  return BR
end

Figure 3.3. The GenBR Algorithm.

Algorithm GenBC(I)
Purpose: generate the class of basic association rules corresponding to I.
Input: I, a frequent itemset.
Output: R, a class of basic association rules corresponding to I.
begin
  $R = \emptyset$
  foreach item $i \in I$
  begin
    $I' = I \setminus \{i\}$
    S = MinimalSubsets(i, I')
    foreach $s \in S$
      $R = R \cup \{s \rightarrow i\}$
  end
  return R
end

Figure 3.4. The GenBC Algorithm.

BAR. The RemoveRedundantClass algorithm removes redundant classes from BAR to give BR.

The GenBC algorithm, presented in Figure 3.4, generates the class of basic association rules corresponding to the frequent itemset I. The main loop of the GenBC algorithm is repeated for each item i in the frequent itemset I. It calls the MinimalSubsets algorithm for computing the MCSs of I' with respect to i. From these MCSs, the algorithm forms a set of basic association rules corresponding to I.

The MinimalSubsets algorithm, presented in Figure 3.5, computes the set of MCSs of I' with respect to i. First, the algorithm determines whether $p = p(i \mid I') \geq minconf$. If so, the algorithm initializes the minimal conditional subset $S = \{I'\}$. The MaximalSubsets function produces a set S' of all maximal subsets of I' as a set of candidate MCSs of I' with respect to i, e.g., S' = MaximalSubsets(BDE) = $\{BD, BE, DE\}$. Because this function is straightforward, we omit it. In the while loop, the algorithm examines the validity of candidate MCSs in S'. For $\forall s \in S$, the algorithm computes the conditional probability $p(i \mid s)$, and then compares $p(i \mid s)$ with p. If $p(i \mid s) = p$, then s is a valid candidate MCS of I' with respect to i, and it is stored in S. If smaller valid candidate

Algorithm MinimalSubsets(i, I')
Purpose: compute a set S of minimal conditional subsets of I' with respect to i.
Input: i, an item such that $I' \cup i$ is a frequent itemset.
  I', an itemset.
Output: S, a set of minimal conditional subsets of I' with respect to i.
begin
  $S = \emptyset$
  $p = \text{supp}(I' \cup \{i\}) / \text{supp}(I')$
  if ($p \geq minconf$) **then**
    $S = \{I'\}$
    S' = MaximalSubsets(I')
    $k = 2$
    while ($S' \neq \emptyset$)
    begin
      foreach $s \in S'$
      begin
        $p_1 = \text{supp}(s \cup \{i\}) / \text{supp}(s)$
        if ($p_1 = p$) **then**
          $S = S \cup \{s\}$
        else
          $S' = S' \setminus \{s\}$
      end
      foreach $s \in S'$
        DelSuperset(s, S) // remove all $s' \in S$ such that $s' \supset s$ from S
      S' = IntersectionSet(S', k)
      $k = k + 1$
    end
  return S
end

Figure 3.5. The MinimalSubsets Algorithm.

MCSs of I' are found, then supersets of them are removed from S. The DelSuperset function (omitted) does this task. The IntersectionSet(S', k) function (omitted) generates all smaller candidate MCSs of I', which are the intersections of itemsets in S' in terms of the depth k of loop. For example, the intersection of the two itemsets AE and AC is equal to A, and it is regarded as a candidate MCS.

Example 3.3.1. Given the dataset in Table 2.1, $minsup$ = 0.3, and $minconf$ = 0.6, we describe the process of generating BR using GenBR. In the while loop of GenBR, we assume $I = \{ACDE\}$ is selected from L. The GenBC algorithm is called to compute the class of basic association rules corresponding to $ACDE$.

The main loop of the GenBC algorithm is repeated for each item in a frequent itemset. Inside the main loop, the MinimalSubsets algorithm is first called to generate all MCSs of CDE with respect to A. $I' = CDE$ and $i = A$. Because the positive conditional probability $p(a \mid cde) \geq minconf$, the MinimalSubsets algorithm begins computing MCPDs of $P(A \mid CDE)$, i.e., a set S of MCSs of CDE with respect to A. Initially, $S = \{I'\}$. MaximalSubsets produces a set S' of all maximal subsets of CDE as candidate MCSs of

CDE with respect to A, $S' = \{DE, CE, CD\}$. In the while loop, itemsets in S' are checked to see if they are candidate MCSs of CDE with respect to A. When CE is examined, $p(a \mid ce) = p(a \mid cde)$, so we obtain $S = \{BDE, CE\}$. For DE and CD, because $p(a \mid de) \neq p(a \mid cde)$ and $p(a \mid cd) \neq p(a \mid cde)$, DE and CD are removed from S', and $S' = \{CE\}$. After DelSuperset, $S = \{CE\}$ and $S' = \{CE\}$. Because S' has only one itemset, $S' = \text{IntersectionSet}(S', k) = \emptyset$. The while loop in the MinimalSubsets algorithm exits, and $S = \{CE\}$ is returned to GenBC. In GenBC, the basic association rule $\{CE \rightarrow A\}$ is placed in R.

Similarly, from the computation of $P(C \mid ADE)$, we have a new basic association rule $AE \rightarrow C$, and $R = \{CE \rightarrow A, AE \rightarrow C\}$. From the computation of $P(D \mid ACE)$, we have new basic association rules $E \rightarrow D$, $C \rightarrow D$ and $A \rightarrow D$, and they are added into R. From the computation of $P(E \mid ACD)$, a new basic association rule $AC \rightarrow E$ is found.

Finally, the algorithm generates a class of basic association rules corresponding to $ACDE$, $R = \{CE \rightarrow A, AE \rightarrow C, E \rightarrow D, C \rightarrow D, A \rightarrow E, AC \rightarrow E\}$. The algorithm returns R to GenBR. At this point, $BR = \{\{DE \rightarrow A, A \rightarrow B, D \rightarrow B, E \rightarrow B, A \rightarrow E\}\}$.

GenBR continues until all frequent itemsets in L, with size of at least 2, have been processed. Consequently, all basic association rules and their classes are generated and presented, as shown in Table 3.1. □

We define the following terminology over BR:

(1) C_r denotes a class of basic association rules. C_i is a set of items, which appear in C_r.

(2) The class in which a basic association rule $X \rightarrow Y$ resides is denoted as $C_r(X \rightarrow Y)$. For example, $C_r(C \rightarrow D)$ might be referred to as one of $C_r(CDE)$, $C_r(ACD)$, $C_r(ACDE)$ and $C_r(CD)$, as shown in Table 3.1.

(3) $C_i(X \rightarrow Y)$ denotes a set of items, which appear in $C_r(X \rightarrow Y)$. e.g, $E \in C_i(C \rightarrow D) = \{C, D, E\}$.

All redundant classes in Table 3.1 are discovered and discarded by the RemoveRedundantClass algorithm. A redundant class $C_r(I)$ contains no more information than the class $C_r(I')$ containing $C_r(I)$. For example, in Table 3.1, $C_r(AB) \subset C_r(ABD)$. If a $C_r(I_1)$ is redundant and $C_r(I_1) \subset C_r(I_2)$, then $I_1 \subset I_2$.

Table 3.1. A Set of Classes of Basic Association Rules.

No.	Class	Basic association rule	Redundant
1	$C_r(AB)$	$B \rightarrow A$	Yes
2	$C_r(BD)$	$B \rightarrow D$	Yes
3	$C_r(DE)$	$E \rightarrow D$, $D \rightarrow E$	
4	$C_r(AD)$	$D \rightarrow A$, $A \rightarrow D$	Yes
5	$C_r(CE)$	$E \rightarrow C$, $C \rightarrow E$	Yes
6	$C_r(AC)$	$A \rightarrow C$, $C \rightarrow A$	Yes
7	$C_r(CD)$	$D \rightarrow C$, $C \rightarrow D$	
8	$C_r(ABD)$	$B \rightarrow D$, $B \rightarrow A$, $D \rightarrow A$, $A \rightarrow D$	
9	$C_r(ADE)$	$E \rightarrow D$, $A \rightarrow D$	Yes
10	$C_r(ACE)$	$CE \rightarrow A$, $AC \rightarrow E$, $AE \rightarrow C$	Yes
11	$C_r(CDE)$	$E \rightarrow D$, $E \rightarrow C$, $C \rightarrow E$, $C \rightarrow D$	
12	$C_r(ACD)$	$A \rightarrow D$, $A \rightarrow C$, $C \rightarrow D$, $C \rightarrow A$	
13	$C_r(ACDE)$	$E \rightarrow D$, $CE \rightarrow A$, $AC \rightarrow E$, $A \rightarrow D$, $AE \rightarrow C$, $C \rightarrow D$	

All frequent itemsets are arranged in a semi-lattice based on their inclusion relation in order to discover all redundant classes. For example, given the dataset shown in Table 2.1 and minsup = 0.3, all frequent itemsets form the semi-lattice shown in Figure 3.6. To identify the redundant class $C_r(AB)$, RemoveRedundantClass compares $C_r(AB)$ with $C_r(ABD)$ because $AB \subset ABD$. To check whether $C_r(DE)$ is redundant, the algorithm compares $C_r(DE)$ with $C_r(ADE)$ and $C_r(CDE)$. Because $C_r(DE) \not\subset C_r(ADE)$ and $C_r(DE) \not\subset C_r(CDE)$, the algorithm does not need to compare $C_r(DE)$ with $C_r(ACDE)$, and $C_r(DE)$ is non-redundant. That means that to check whether $C_r(I_1)$ is redundant, we only compare $C_r(I_1)$ with all $C_r(I_2)$, where I_1 and I_2 are frequent itemsets, $I_1 \subset I_2$, and $|I_1| = |I_2| - 1$, as justified by Theorem 3.3.1.

Theorem 3.3.1. Given three itemsets I_1, I_2, and I_3, and $I_1 \subset I_2 \subset I_3$, if $C_r(I_1) \not\subset C_r(I_2)$, then $C_r(I_1) \not\subset C_r(I_3)$.

Proof: Suppose the condition is satisfied. Hence, $\exists r$: $X \rightarrow A \in C_r(I_1)$, and a MCP $p(a \mid x)$, but $r \notin C_r(I_2)$, i.e., $\exists Z \subseteq I_2 \setminus \{X, A\}$, $p(a \mid x) \neq p(a \mid x, z)$. Because $I_2 \setminus \{X, A\} \subset I_3 \setminus \{X, A\}$, $\exists Z \subseteq I_3 \setminus \{X, A\}$, $p(a \mid x) \neq p(a \mid x, z)$, i.e., $P(A \mid X)$ is not a MCPD of $P(A \mid I_3 \setminus A)$. Hence, $r \notin C_r(I_3)$. □

In [22], we proposed an inference system for basic association rules. It is called the *C-inference system*, because it permits a rule's confidence to be inferred. We proved that the *C*-inference system holds on BR and that all association rules can be derived from BR by the application of inference rules in the C-inference system. The C-inference system is summarized in Table 3.2, where p and q signify the confidences of association rules. Rules that cannot be derived from other rules by the C-inference system are called *non-redundant association rules*.

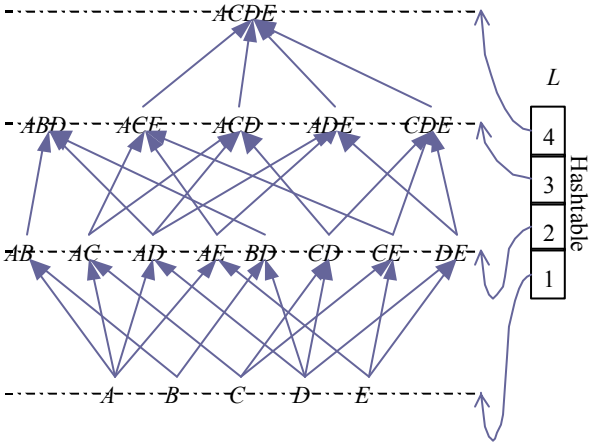

Figure 3.6. A Semi-Lattice for All Frequent Itemsets

Table 3.2. The *C*-inference System on Basic Association Rules

Inference Rule	Premise	Conclusion
Augmentation	$X \xrightarrow{p} Y \in C_r, \forall X' \subseteq C_i \setminus \{X, Y\}$	$XX' \xrightarrow{p} Y$
Pseudo-Transitivity	$X \xrightarrow{p} Y \in C_r, XW \xrightarrow{q} Z \in C_r$	$XW \xrightarrow{q} Z$
Contraction	$X \xrightarrow{p} Y \in AR, XY \xrightarrow{q} Z \in AR$	$X \xrightarrow{pq} YZ$
Additivity	$X \xrightarrow{p} Y \in C_r, W \xrightarrow{q} Z \in C_r$	$XW \xrightarrow{pq} YZ$
Right union	$X \xrightarrow{p} Y \in C_r, X \xrightarrow{q} Z \in C_r$	$X \xrightarrow{pq} YZ$
Left union	$X \xrightarrow{p} Z \in C_r, Y \xrightarrow{p} Z \in C_r$	$XY \xrightarrow{p} Z$

3.4 The Computational Complexity of GenBR

Let us analyze the computational complexity of GenBR. From Figure 3.1, we observed that itemsets contained in the context of RCPDs may be used to construct a specific semi-lattice for computing MCPDs. All nodes in the semi-lattice are subitemsets of an itemset.

This construction corresponds to the problem of generating subsets in mathematics, or creating a hypercube (or a *k*-cube), or a Q_n graph in graph theory [15]. We analyze the computational complexity of our algorithm based on the binomial theorem [13] as follows.

According to the binomial theorem, the number of subitemsets of a *k*-itemset (nodes of the semi-lattice), denoted *NS*, is given by:

$$NS = C^1_k + \ldots + C^k_k = 2^k - 1 \qquad (3.1)$$

By counting edges from the top of the semi lattice for a *k*-itemset, as shown in Figure 3.7, to its bottom, the number of edges, denoted *NE*, of the semi-lattice is given by:

$$NE = C_k^{k-1} + 2C_k^{k-2} + 3C_k^{k-3} + \ldots + (k-1)C_k^1$$
$$= C_k^{k-1} + 2C_k^{k-2} + 3C_k^{k-3} + \ldots + (k-1)C_k^1 + kC_k^0 - kC_k^0$$
$$= \sum_{i=1}^{k} iC_k^{k-i} - k = \sum_{i=1}^{k} \frac{ik!}{i!(k-i)!} - k$$
$$= k\sum_{i=1}^{k} \frac{(k-1)!}{(i-1)!(k-i)!} - k$$
$$= k\sum_{i=1}^{k} C_{i-1}^{k-1} = k2^{k-1} - k \qquad (3.2)$$

From Equation (3.1), we obtain the number of association rules NR for a *k*-itemset generated by the FastGenRules algorithm [2] as follows:

$$NR = 2^k - 2 \qquad (3.3)$$

According to Figure 3.1, to compute the MCSs of *ABCD* with respect to *E*, in the worst case, the process forms a semi-lattice of *ABCD*. One edge in the semi-lattice corresponds to one comparison. For the whole itemset *ABCDE*, there are five semi-lattices of this kind. Hence, from Equation (3.2), the computational complexity *TC* of GenBR for a *k*-itemset is determined by:

$$TC = k(k-1)(2^{k-2} - 1) \approx k^2 2^{k-2}$$
$$= O(k^2 2^{k-2}) \qquad (3.4)$$

Given that *l* is the length of the longest itemsets in the set of frequent itemsets, the ratio of the complexity of GenBR to that of FastGenRules is:

$$R = O(l^2 2^{l-2})/O(2^{l+1}) = O(l^2 / 2^3)$$
$$= O(l^2) \qquad (3.5)$$

Although the time complexity of GenBR exponentially increases with *l*, we show that GenBR performs very well over actual datasets in the following section.

4 Comparison and Experimental Results

The overall comparison between our approach and four previous approaches is shown in Table 4.1.

Table 4.1. The Overall Evaluation

Algorithm Features	Fast-Gen-Rules	Fast-Gen-RR	Gen-GBRI	Cover-Rules	GenBR
Canonical form					√
Non-redundant					√
Infer a rule's support					
Infer a rule's confidence			√		√
Armstrong axioms-like					√
#Rules	All	Fewer	Most	Fewest	Best
Elapsed time	Fastest	Slower	Slowest	Medium	Fast

First, consider the form of rules generated. Only GenBR generates non-redundant rules in canonical form. We believe that this type of rules is easy for users to understand. Rules generated by GenBR are non-redundant, i.e., these rules and their confidences cannot be derived from simpler rules and their confidences by using the inference rules defined in any of the other approaches. Secondly, consider whether an inference rule permits the support and confidence of rules to be inferred. None of the approaches can infer a rule's support, and only Gen-GBRI and GenBR can infer confidences. Thirdly, consider whether the inference system resembles Armstrong's axioms; only the C-inference system does so. Fourthly, consider the number of rules generated. CoverRules generates the fewest, but GenBR organizes the basic association rules into classes related to the frequent itemsets. GenBR also tends to generate the fewest rules when *minconf* is high. Finally, with respect to the elapsed time, FastGenRules is the fastest on all datasets. GenBR is more efficient than the other approaches, except for a few cases.

As described in the remainder of this section, experiments on several synthetic and real-life datasets were conducted to compare the performance of GenBR and previous algorithms with respect to the number of rules and the elapsed running time.

4.1 Experimental Design

The experimental environment was a PC with a 2.53 GHz Intel CPU, 512MB of RAM, and Microsoft Windows XP. All algorithms were implemented in Microsoft Visual Java++.

The datasets used are listed in Table 4.2. Synthetic datasets, T10I4D100K and T20I6D100K, were generated by running GenData [17], which is a synthetic data generator from IBM. Several well-known benchmark real-world datasets were chosen from [29]. CustInfo-5 was obtained from the CustInfo database [4] by joining five tables. The number of attributes shown for the real-world datasets represents the number after multi-valued attributes were converted to several binary attributes.

Table 4.2. Synthetic and Real-world Datasets

Dataset	# Attributes	#Items	Length of Record or Transactions	# Transactions
Chess	37	75	37	3196
Connect	43	129	43	67557
Mushroom	23	128	23	8124
Molecular	62	483	62	3190
CustInfo-5	14	9168	14	908241
T10I4D100K		1000	10	100000
T20I6D100K		1000	20	100000

To compare the performance of GenBR and previous approaches with respect to the number of rules and the elapsed running time, we implemented the FastGenRules for generating association rules [2] as well as the Gen-GB algorithm for generating generic basis for exact rules, and the Gen-RI algorithm for generating informative basis for approximate association rules [5]. The Gen-GB and the Gen-RI algorithms were combined into the Gen-GBRI algorithm for our experiments. The FastGenRepresentatives (denoted FastGenRR) algorithm generates a set *RR* of representative association rules [20]. CoverRules generates an informative cover of cover rules [11].

The GenBR algorithm consists of two steps to generate basic association rules. The first step generates all classes of basic association rules. The second step removes redundant classes. The GenBR algorithm is more similar to the FastGenRules algorithm than to the other algorithms. To compare the GenBR with FastGenRules with respect to the elapsed running time, we only consider the first step of GenBR, which generates the basic association rules, and we ignore the elapsed time required for the second step.

4.2 Results

We first compared the algorithms with regard to the number of rules generated. GenBR generates fewer rules than FastGenRules, and in some cases fewer rules than FastGenRR, Gen-GBRI, and CoverRules.

Table 4.3. Chess, *minsup* = 80 %

Algorithm / *minconf*		10	30	60	80	90	100
Fast-Gen-Rules	#Rules	552564	552564	552564	552564	349298	4192
	Elapsed time (ms)	39125	38594	39031	39000	**20640**	**969**
Gen-GBRI	#Rules	27791	27791	27791	27791	26852	2228
	Elapsed time	93020	93047	93021	93020	93057	92458
Fast-Gen-RR	#Rules	1678	1678	1678	1678	4378	2228
	Elapsed time	59234	58750	58719	58703	57656	60188
Cover-Rules	#Rules	**226**	**226**	**226**	**226**	**1702**	2228
	Elapsed time	36453	36734	36437	36453	526922	211609
GenBR	#Classes	8112	8112	8112	8112	7710	145
	#Rules	28416	28416	28416	28416	27054	**6**
	Elapsed time	**25391**	**25141**	**25202**	**25593**	25266	16468

For Chess with *minsup* = 80%, the number of rules generated by the algorithms and their elapsed times are shown in Table 4.3.

GenBR generates significantly fewer rules than FastGenRules for all settings of *minconf*. For example, when *minconf* = 10%, FastGenRules generates 552564 rules, while GenBR generates 28416 rules.

Among all approaches, CoverRules generates the fewest rules when *minconf* is 90% or less. For example, in Table 4.3, with *minconf* = 10%, CoverRules generates 226 rules while GenBR generates 28416 rules. However, when *minconf* is 100%, GenBR generates the fewest rules. For example, in Table 4.3, GenBR generates only 6 rules, while the other approaches generate 2228 or more rules.

Table 4.4. Comparison wrt the Number of Rules.

Dataset	Minsup	Fast-Gen-Rules	Fast-Gen-RR	Gen-GBRI	Cover-Rules	GenBR
Chess	80%	4192	2228	2228	2228	**6**
Chess	90%	132	116	116	116	**2**
Connect-4	95%	2270	684	684	684	**10**
Connect-4	97%	245	161	161	161	**4**
Mushroom	30%	8450	557	557	426	**76**
Mushroom	40%	939	169	169	139	**43**
Molecular	5%	2085	1261	1261	1221	**391**
Molecular	10%	57	45	45	43	**22**
CustInfo-5	5%	4913	656	2591	**475**	2591
CustInfo-5	10%	1862	330	1078	**238**	1078
T10I4D100K	0.05%	28695	6075	6075	4805	**2641**
T10I4D100K	0.1%	657	522	522	501	**134**
T20I6D100K	0.2%	67	45	45	36	**26**
T20I6D100K	0.3%	33	11	11	**3**	12

For other datasets and different parameters, the experimental results concerning the number of rules generated are similar to those shown in Table 4.3. We summarize these experimental results in Table 4.4. Because users are generally concerned with association rules with high confidences, we only describe the number of rules with 100% confidence (*minconf* = 100%) for all datasets besides CustInfo-5. For CustInfo-5, since there is no exact association rule when *minsup* = 5% or 10%, we use *minconf* = 90%. Under these conditions, GenBR

generates fewer rules than the other algorithms, except for a few cases when CoverRules generates the fewest.

Although the computation complexity of GenBR is exponentially greater than that of FastGenRules (as mentioned in Section 3.4), in our experiments, we observed that the elapsed time for GenBR is significantly less than that for FastGenRules, except for cases where both algorithms have their fastest performance, which correspond to values of *minconf* approaching 100%. For example, in Table 4.3, when *minconf* is 10%, the elapsed time for GenBR is 25391 ms while that of FastGenRules is 39125 ms. When *minconf* is 100%, the elapsed time of GenBR is 16468 ms while the elapsed time of FastGenRules is 969 ms. Across all tested settings of *minconf* from 10% to 100%, GenBR has the lowest maximum elapsed time (25593 ms).

The number of exact association rules greatly affects the elapsed time of GenBR. If there are more exact association rules, there may be more functional dependencies, and consequently, GenBR spends more time.

We summarize the experimental results related to elapsed time by recording the ratios of the elapsed time of GenBR to previous algorithms in Table 4.5. We set *minconf* = 100%. The bold entries identify cases where the corresponding algorithm is faster than GenBR. The results show that GenBR is faster than all algorithms except FastGenRules on most presented datasets.

Table 4.5. Evaluation wrt Elapsed Time

Dataset	Minsup	FastGen-Rules	FastGen-RR	Gen-GBRI	Cover-Rules
Chess	80%	**16.99**	0.27	0.18	0.08
Chess	90%	**1.98**	0.19	0.12	0.09
Connect-4	95%	**11.69**	0.65	**1.12**	0.18
Connect-4	97%	**2.32**	0.35	0.35	0.15
Mushroom	30%	**30.47**	**2.51**	**16.72**	0.66
Mushroom	40%	**10.96**	**1.73**	**6.6**	0.7
Molecular	5%	**1.74**	0.007	0.002	0.004
Molecular	10%	**2.02**	0.08	0.02	0.31
CustInfo-5	5%	**2.73**	0.37	0.1	0.11
CustInfo-5	10%	**2.59**	0.58	0.19	0.18
T10I4D100K	0.05%	**15.86**	0.05	0.01	0.08
T10I4D100K	0.1%	**3.32**	0.06	0.01	0.08
T20I6D100K	0.2%	**2.12**	0.03	0.002	0.07
T20I6D100K	0.3%	**2**	0.31	0.006	**1.3**

5 Conclusions and Future Work

5.1 Conclusions

In this paper, we proposed a new type of association rules, called basic association rules.

First, by referring the relational database theory on functional dependencies, we developed the new concepts of a restricted conditional probability distribution (RCPD), a minimal conditional probability distribution (MCPD), a minimal conditional subset (MCS), and a basic association rule. We established the *C*-inference system on basic association rules, which is similar to Armstrong's axioms on functional dependencies.

Secondly, we proposed the GenBR algorithm for generating a set of classes of basic association rules from a set of frequent itemsets. GenBR efficiently generates basic association rules as compared with the previous approaches for generating small sets of association rules. GenBR also generates fewer rules than previous approaches when *minconf* is high.

Thirdly, arguably, users will find basic association rules to be more manageable and understandable than previously proposed reduced sets of association rules. The rules are concise, the number of rules is small, redundancy among rules in a class of basic association rules has been eliminated, and inference involving confidence values is possible on the rules. This point is argued at greater length in [22].

Fourthly, we showed that the search space of our algorithm to compute basic association rules is a hypercube (*n*-cube) or Q_n graph. This insight aided in our theoretical analysis of the algorithm.

5.2 Future Work

An open problem is to find a more efficient algorithm for discovering basic association rules from frequent itemsets. Finding a heuristic method to discover basic association rules without generating frequent itemsets will also be challenging.

We proposed the idea of a restricted conditional probability distribution as a foundation for mining association rules, and we distinguished it from the traditional conditional probability distribution. It can be regarded as a more general concept than the context-specific conditional probability distribution described in [6]. Further work on restricted conditional probability distributions may be promising.

We established the *C*-inference system on association rules, which may be regarded as an extension of Armstrong's axioms on functional dependencies. This kind of inference system works on exact and approximate rules simultaneously. Further exploration of the properties of the *C*-inference system is needed.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. SIGMOD'93*, pages 207-216, Washington. DC, May 1993.

[2] R. Agrawal and R. Srikant. Fast algorithm for mining association rules. In *Proc. VLDB'94*, Santiago, Chile, 1994.

[3] E. Baralis and G. Psaila. Designing templates for mining association rules. *Journal of Intelligent Information Systems,* 9(1):7-32, July 1997.

[4] B. Barber and H. J. Hamilton. Extracting share frequent itemsets with infrequent subsets. *Data*

Mining and Knowledge Discovery, 7(2):153-185, April 2003.

[5] Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal. Mining minimal non-redundant association rules using frequent closed itemsets. In *Proc. of the First International Conference on Computational Logic*, pages 972-986, 2000.

[6] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proc. 12th Conf. on Uncertainty in Artificial Intelligence* (*UAI-96*), pages 115-123, Portland, Oregon, 1996.

[7] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlation. In *Proc. SIGMOD'97*, pages 265-276, May 1997.

[8] S. Brin, R. Motwani, J. D. Ullman and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proc. SIGMOD'97,* pages 255-264, Montreal, Canada, June 1997.

[9] C. L. Carter, H. J. Hamilton, and N. Cercone. Share based measures for itemsets. In *Proc. PKDD'97*, pages 14-24, Trondheim, Norway, June 1997.

[10] E. Castillo, J. M. Gutierrez, and A. S. Hadi, *Expert Systems and Probabilistic Network Models*, Springer, 1997.

[11] L. Cristofor and D. Simovici. Generating an informative cover for association rules. In *Proc. of the IEEE International Conference on Data Mining,* 2002.

[12] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*, Fourth edition. Cambridge University Press, 1994.

[13] E. G. Goodaire and M. M. Parmenter. *Discrete Mathematics with Graph Theory*, Second Edition. Prentice Hall, Upper Saddle River, NJ, 2002.

[14] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. SIGMOD'00*, pages 1-12, Dallas, TX, May 2000.

[15] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater. *Fundamentals of Domination in Graphs*. Marcel Dekker, 1998.

[16] R. J. Hilderman and H. J. Hamilton. *Knowledge Discovery and Interest Measures*, Kluwer Academic, Boston, 2002.

[17] IBM. GenData, www.almaden.ibm.com/cs/quest/syndata.html

[18] M. Kamber and R. Shinghal. Evaluating the interestingness of characteristic rules. In *Proc. KDD'96,* pages 263-266, Portland, Oregon, 1996.

[19] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivnen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. In *Proc. of the 3rd CIKM Conference,* pages 401-407, November 1994.

[20] M. Kryszkiewicz. Representative association rules and minimum condition maximum consequence association rules. In *Proc. PKDD'98*. pages 361-369, Nantes, France, 1998.

[21] W. Lee, S. J. Stolfo, and K. W. Mokl. A data mining framework for building intrusion detection models. In *Proc. IEEE Symposium on Security and Privacy*, 1999.

[22] G. Li. *Basic Association Rules*, M.Sc. Thesis, Department of Computer Science, University of Regina, 2004.

[23] W. Lin, S. A. Alvarez, and C. Ruiz. Collaborative recommendation via adaptive association rules mining. In *Proc. WEBKDD'2000*, San Francisco, CA, Aug. 2000.

[24] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Closed set based discovery of small covers for association rules. In *Proc. of the 15th Conference on Advanced Databases*, pages 361-381, 1999.

[25] C. M. Ricardo. *Database Systems: Principles, Design, & Implementation*, Macmillan, 1990.

[26] K. Satou, G. Shibayama, T. Ono, Y. Yamamura, E. Furuichi, S. Kuhara, and T. Takagi. Finding association rules on heterogeneous genome data. In *Proc. of the Pacific Symposium on Biocomputing'97* (*PSB'97*), pages 397-408, Hawaii, Jan. 1997.

[27] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *Proc. KDD'97*, pages 67-73.

[28] H. Toivonen, M. Klemettinen, P. Ronkainen, K. Hatonen, and H. Mannila. Pruning and grouping discovered association rules. In *Proc. ECML-95 Workshop on Statistics, Machine Learning, and Knowledge Discovery in Database*, pages 47-52, April 1995.

[29] UCI Machine Learning Repository. ftp://ftp.ics.uci.edu/pub/machine-learning-databases/

[30] M. Zaki, Generating non-redundant association rules. In *Proc. KDD'2000*, pages 34-43. August 2000.