MINING ASSOCIATION RULES WITH ADJUSTABLE INTERESTINGNESS

BY

NGUYEN THANH TRUNG

SUPERVISED BY

DR. HA QUANG THUY

A THESIS SUBMITTED THE DEGREE OF BACHELOR OF SCIENCE AT THE FACULTY OF TECHNOLOGY VIETNAM NATIONAL UNIVERSITY, HANOI

JUNE, 2003
ACKNOWLEDGEMENTS

This thesis for bachelor’s degree has been accomplished for three months. During this time, many people have made substantial contributions in one way or another that I would like to mention herein.

First and foremost, I would especially like to thank my research advisor, Dr. Ha Quang Thuy for his invaluable guidance and tremendous motivation that he provided at every step of this work. His enthusiastic support and untiring interest in the subject is deeply appreciated. I have gain immensely from his deep technical insight and thoroughness in problem solving.

Some portions of this thesis have been previously published in the Conference of Junior Scientists 2002 of Vietnam National University, Hanoi, and I owe thanks to Dr. Do Van Thanh, M.Sc. Pham Tho Hoan, B.Sc. Phan Xuan Hieu for their valuable contributions as the co-authors of that paper.

My thanks also go to all of my lecturers at Faculty of Technology of Vietnam National University Hanoi who provided me with indispensable scientific knowledge throughout four school years. Special thanks to the following individuals, and many others who are not mentioned by name, for their teaching: M.Sc. Le Quang Hieu, M.Sc. Nguyen Quang Vinh, M.Sc. Nguyen Dinh Viet, M.Sc. Pham Hong Thai, Dr. Nguyen Tue, M.Sc. Nguyen Nam Hai, M.Sc. Dao Kien Quoc, M.Sc. Le Anh Cuong, Asoc.Prof. Trinh Nhat Tien, Dr. Dinh Manh Tuong, M.Sc. Vu Ba Duy, Asoc.Prof. Nguyen Quoc Toan, M.Sc. Ngo Le Minh, Asoc.Prof. Ngo Quoc Tao. Without the knowledge they equipped me, my thesis would never take shape.

I am particularly grateful to my family for providing me with a source of strength and encouragement, and giving me the best possible education, and imbibing in me a thirst for learning.

Last but not the least my girlfriend Nguyen Thi Thu Thuy who sacrificed time and energy so that this work could be completed. I appreciate it, and hope that the effort has been worthwhile.
ABSTRACT

Over the last several years, the problem of efficiently generating large numbers of association rules has been an active research topic in the data mining community. Many different algorithms have been developed with promising results. There are two current approaches to the association rule mining problem. The first is to mine the frequent itemsets regardless of their coefficients. The second is to assign weights to the items to reflect their importance to the users. However, they both rely on the using of the minimum support which may confuse us. Practically, we may want to mine the best rules to our knowledge instead of those which satisfy a certain threshold, especially if this threshold is an equation. To overcome this problem, we introduce the concept of adjustable interestingness and propose a novel approach in mining association rules based on adjustable interestingness. Our algorithm only works with the most interesting rules, thus reducing significantly search space by skipping many uninteresting itemsets and pruning those that cannot generate interesting itemsets at the earlier stage. Therefore, the total time needed for the mining is substantially decreased.
TABLE OF CONTENTS

Acknowledgements .....................................................................................................i

Abstract...................................................................................................................... ii

Table of contents ...................................................................................................... iii

List of tables and figures ........................................................................................... iv

CHAPTER 1: Introduction .........................................................................................1
  1.1. What is data mining? ........................................................................................1
  1.2. Data mining versus query tools ........................................................................2
  1.3. Mining association rules ...................................................................................3
  1.4. Outline of the thesis ..........................................................................................5

CHAPTER 2: Mining association rules with weighted items....................................6
  2.1. Introduction ......................................................................................................6
  2.2. Problem definition ............................................................................................7

CHAPTER 3: Mining association rules with adjustable interestingness .................10
  3.1. Interestingness and interesting itemsets .........................................................10
  3.2. Interestingness constraints .............................................................................11
  3.3. Motivation behind interesting itemsets and adjustable interestingness .........12

CHAPTER 4: Algorithm for mining association rules with adjustable interestingness (MARAI) .........................................................................................................14
  4.1. Motivation ......................................................................................................14
  4.2. Preliminaries...................................................................................................15
  4.3. Basic properties of itemset-tidset pairs ..........................................................18
  4.4. MARAI: Algorithm design and implementation ...........................................20
  4.5. Experimental Evaluation ................................................................................25

CHAPTER 5: Conclusion.........................................................................................28

References ..................................................................................................................a

Appendix ....................................................................................................................b
LIST OF TABLES AND FIGURES

Table 1. Database of a stationery store ................................................................. 8
Table 2. Transactions of a stationery store .......................................................... 9
Table 3. Itemsets sorted into descending order of their interestingness .......... 11
Table 4. Itemsets sorted into descending order of the interestingness .......... 17
Table 5. All interesting itemsets ........................................................................ 18
Table 6. Database characteristics ..................................................................... 25

Figure 1. Example database and frequent itemsets .......................................... 4
Figure 2. Example database .............................................................................. 15
Figure 3. The MARAI algorithm ....................................................................... 22
Figure 4. Search process using adjustable interestingness ............................... 23
Figure 5. Performance of the MARAI algorithm on Cosmetic ......................... 26
Figure 6. Performance of the MARAI algorithm on Census ............................ 27
CHAPTER 1

INTRODUCTION

In this chapter, we introduce the concept of data mining, and explain why it is regarded as such important developments. As companies is the background of mining association rules.

1.1. What is data mining?

There is confusion about the exact meaning between the terms ‘data mining’ and ‘knowledge discovery in databases (KDD)’. At the first international KDD conference in Montreal in 1995, it was proposed that the term ‘KDD’ be used to describe the whole process of extraction of knowledge from data. An official definition of KDD is: ‘the non-trivial extraction of implicit, previously unknown and potentially useful knowledge from data’ [2]. The knowledge which is discovered must be new, not obvious, and human must be able to use it for a particular purpose. It was also proposed that the term ‘data mining’ should be used exclusively for the discovery stage of the KDD process. The whole KDD steps include selection, preprocessing, transformation, data mining and the interpretation or evaluation. Data mining has been focused on as it is the most significant and most time-consuming among KDD steps.

The sudden rise of interest in data mining can partly be explained by the following factors [2]:

1. In the 1980s, all major organizations built infrastructural databases, containing data about their clients, competitors, and products. These databases form a potential gold-mine; they contain gigabytes of data with much ‘hidden’ information that cannot easily be traced using SQL (Structure Query Language). Data mining algo-
2. As the use of networks continues to grow, it will become increasingly easy to connect databases. Thus, connecting a client’s file to a file with demographic data may lead to unexpected views on the spending patterns of certain population groups.

3. Over the past few years, machine-learning techniques have expanded enormously. Neural networks, genetic algorithms and other simple, generally applicable learning techniques often makes it easier to find interesting connections in databases.

4. The client/sever revolution gives the individual knowledge worker access to central information systems, from a terminal on his or her desk.

1.2. Data mining versus query tools

What is the difference between data mining and a normal query environment? What can a data mining tool do that SQL cannot?

It is significant to realize that data mining tools are complementary to query tools. A data mining tool does not replace a query tool but give a lot of additional possibilities [2]. Suppose that we have a large file containing millions of records that describe customers’ purchases in a supermarket. There is a wealth of potentially useful knowledge which can be found by trigger normal queries, such as ‘Who bought butter and bread last week?’ ‘Is the profit of this month more than that of last month?’ and so on. There is, however, knowledge hidden in the databases that is much harder to find using SQL. Examples would be the answers to questions such as ‘What products were often purchased together?’ or ‘What are the subsequent purchases after buying a gas cooker?’. Of course, these questions could be answered using SQL but proceeding in such a way could take days or months to solve the problem, while a data mining algorithm could find the answers automatically in...
a much shorter time, sometimes even in minutes or a couple of hours. It is said that if we know exactly what we are looking for, use SQL; but if we know only vaguely what we are looking for, turn to data mining.

1.3. Mining association rules

There are various kinds of methods to mine the information from the database, such as mining association rules, multi-level data generalization and summarization, classification, and clustering [4]. The most common type is mining association rules.

The problem of mining association rules in databases was first introduced in 1993 by Agrawal [1]. An example of such a rule might be that “90% of customers purchase bread and butter also purchase milk and coffee”. Since its introduction, Association Rules Mining (ARM) [1] has become one of the core data mining tasks. ARM is an undirected or unsupervised data mining technique, which work on massive data, and it produces clear and understandable results. ARM is aimed at finding regularities in data.

The following is a formal statement of the problem [1]: Let $I = \{i_1, i_2, ..., i_m\}$ be a set of literals, called items. A set of items is also called an itemset. An itemset with $k$ items is called a $k$-itemset. Let $D$ be a set of transactions, where each transaction $T$ is a set of items such that $T \subseteq I$. Associated with each transaction is a unique identifier, called its $TID$. We say that a transaction $T$ contains $X$, a set of some items in $I$, if $X \subseteq T$. The support of of an itemset $X$, denoted $\sigma(X,D)$, is the number of examples in $D$ where it occurs as a subset. An itemset is frequent or large if its support is more than a user-specified minimum support ($\text{min}_\text{sup}$) value.

An association rule is an implication of the form $X \Rightarrow Y$ where $X \subseteq I$, $Y \subseteq I$ and $X \cap Y = \phi$. $X$ is called the antecedent of the rule, and $Y$ is called the consequence of the rule. The rule $X \Rightarrow Y$ has support $s$ in the transaction set $D$ if $s\%$ of transactions in $D$ contain both $X$ and $Y$. That is, the support of the association rule
\( X \Rightarrow Y \) is the probability that \( X \cup Y \) occurs in the set of transactions in the database \( D \); it is denote by \( support(X \cup Y) \). The rule \( X \Rightarrow Y \) holds in the transaction set \( D \) with confidence \( c \) if \( c \% \) of transactions in \( D \) that contain \( X \) also contain \( Y \). The confidence of the association rule \( X \Rightarrow Y \) is the probability that a transaction contains \( Y \) given that the transaction contains \( X \), or it may be given mathematically as \( support(X \cup Y) / support(X) \).

**Example 1.1.** Consider a set of itemsets \( I = \{ A, B, C, D, E, F \} \). Let \( D \) be a set of four transactions as following:

<table>
<thead>
<tr>
<th>Transaction identification</th>
<th>Items bought</th>
<th>Frequent pattern</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>A, B, C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>A, C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>A, D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>B, E, F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Figure 1. Example database and frequent itemsets](image)

For rule \( A \Rightarrow C \):

\[
\text{support} = support(\{A\} \cup \{C\}) = 50% \\
\text{confidence} = support(\{A\} \cup \{C\}) / support(\{A\}) = 66%
\]

The problem of discovering all association rules can be decomposed into two subproblems [1]:

1. Find all acts of items (itemsets) that have transaction support above minimum support. The support for an item is the number of transactions that contain the itemset. Recall that an itemset is frequent or large if its support is more than a user-specified minimum support (\texttt{min_sup}) value.
Example 1.2. From the above database, we obtain four frequent itemsets \{A\}, \{B\}, \{C\} and \{A, C\} with supports of 75\%, 50\%, 50\% and 50\% respectively.

2. Use the large itemsets to generate the desired rules. Here is a straightforward algorithm for this task. For every large itemset \(l\), find all non-empty subsets of \(l\). For every such subset \(a\), output a rule of the form \(a \Rightarrow (l - a)\) if the ratio of \(\text{support}(l)\) to \(\text{support}(a)\) is at least \(\text{minconf}\). We need to consider subsets of \(l\) to generate rules with multiple consequents.

Example 1.3. From the frequent itemset \{A, C\} found in example 1.2, we can generate two rules whose confidences are greater than or equal to \(\text{minconf}\) value.

\[
\begin{align*}
\text{Itemset} & \{A, C\} \\
\text{rule} & A \Rightarrow C \\
\text{confidence} & = \frac{\text{support}(\{A\} \cup \{C\})}{\text{support}(\{A\})} = \frac{50\%}{75\%} = 66\% \\
\text{rule} & C \Rightarrow A \\
\text{confidence} & = \frac{\text{support}(\{A\} \cup \{C\})}{\text{support}(\{C\})} = \frac{50\%}{50\%} = 100\%
\end{align*}
\]

As the problem of generating rules from the itemsets in step 2 is straightforward [1], we will not mention it over again in this thesis.

1.4. Outline of the thesis

The remainder of this thesis is as follows. In chapter 2, we state the definition of mining association rules with weighted items. The main aim of this chapter is to provide a background for weight based problems we base our approach on. In chapter 3, we describe the main idea of the thesis. A new term, adjustable interestingness, is also introduced here. After the extensive discussion of mining association rules with adjustable interestingness in chapter 3, we devote chapter 4 to the algorithm for it. Experiments on real databases are also described. Finally, we conclude the thesis with a summary and a discussion of future work.
CHAPTER 2

MINING ASSOCIATION RULES WITH WEIGHTED ITEMS

In the last section, we discussed about mining association rule for unweighted case. In the following, we introduce the conceptual framework of weight and apply it to mining association rules. The concept of weight will be used in the coming chapters.

2.1. Introduction

There have been two approaches to the association rule mining problem. The first one is to mine the frequent itemsets regardless of their coefficients [1, 7]. The second trend is to assign weights to the items to reflect their importance to the users. Some previous works focused on mining frequent itemsets with weighted items [5] and different supports [6].

The association rules, mentioned in previous chapter, are called the ‘unweighted’ association rules [6] as the items are treated uniformly.

*Example 2.1.* The following rule is the unweighted binary association rule from [1]:

\[(Bread = Yes) \Rightarrow (Ham = Yes)\]

with support = 60% & confidence = 80%

The above rule states that the probability of buying bread and ham in a set of transaction is 0.6, and the *confidence* states that probability that buying ham, given that that customer buys bread, is 0.8.
The above rule is an unweighted case. However, it is better for the following cases to consider the importance of the items or attributes.

For example, the rule

\[(\text{Income} = \text{High}) \Rightarrow (\text{School level} = \text{High})\]

is, in human interpretation, probably more interesting than

\[(\text{Price} = \text{High}) \Rightarrow (\text{Sales} = \text{Low})\]

even if the support of the latter rule is much more than that of the former.

By using the weights, the importance of the attributes or items can be reflected, and we can mine the rules with interestingness. For example, we can add the weights to the sales transactions, where the items are under promotion, or with more profits. The unweighted association rules would be the same if the database did not change, thus it cannot provide a flexible way for the users to adjust the priority of the rules. Therefore, the mining association rules for weighted items was presented in [6] to resolve this problem.

### 2.2. Problem definition

Similar to section 1.3, we consider a database with a set of transaction \(D\), a set of attributes or items \(I\), and each transaction is assigned a transaction identifier \(TID\).

Based on the definitions in section 1.3, the weights and weighted association rules are defined [6]:

**Definition 1.** An item weight, \(w\), where \(0 \leq w \leq 1\), defines the importance of the item. 0 indicates the least important item, and 1 denotes the most important item.

For example, if the weight of the itemset \(X\) is 0.95, it tells us the itemset is important in the set of transaction \(D\). The weight of 0.1 indicates a less important set.
Definition 2. A weighted association rule (or association rule with weighted item) has the form \( X \Rightarrow Y \), where \( X \subseteq I \), \( Y \subseteq I \), \( X \cap Y = \emptyset \), and the items in \( X \) and \( Y \) are given by the weights.

Definition 3. The weighted support of the binary weighted rule \( X \Rightarrow Y \) is the adjusting ratio of the support, or mathematically,

\[
wsupport(X, Y) = \left( \sum_{j \in (X \cup Y)} w_j \right) support(X, Y)
\]

where the weights of the items \( \{i_1, i_2, \ldots, i_m\} \) are \( \{w_1, w_2, \ldots, w_m\} \) respectively.

In order to find the interesting rules, two thresholds, minimum weighted support (\( w_{\text{min sup}} \)) and minimum confidence (\( \text{minconf} \)) must be specified.

Definition 4. An itemset \( X \) is called a large weighted itemset if the weighted support of the itemset \( X \) is greater than or equal to the weighted support threshold, or mathematically,

\[
wsupport(X) \geq w_{\text{min sup}}
\]

Definition 5. A weighted association rules \( X \Rightarrow Y \) is called an interesting rule if the confidence of itemset \( (X \cup Y) \) is greater than or equal to a minimum confidence threshold, and \( (X \cup Y) \) is a large weighted itemset.

<table>
<thead>
<tr>
<th>Product ID</th>
<th>Item</th>
<th>Average Profit</th>
<th>Weight</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Eraser</td>
<td>100</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Ball-pen</td>
<td>200</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Notebook</td>
<td>300</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pencil</td>
<td>500</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Pen</td>
<td>1000</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Database of a stationery store
Example 2.2. Suppose in a stationery store, a database is shown in Table 1. Each item includes information of name, profit and given weight. Table 2 gives the transaction database. For each transaction, there will be a transaction identifier (TID) and the names of items. Suppose there are only 5 items and totally 8 transactions in the transaction database.

Regardless of the weights given, if the value of minsup is set to 0.4, \( \{1, 4\} \) will be a large itemset since its support is 50%. However, \( \{1, 4, 5\} \) is not a large itemset as it appears only two times in the database.

But if we take weights of items into account, and the given value of wminsup is 0.4, \( \{1, 4\} \) will not be a large weighted itemset since

\[
(0.1 + 0.5) \times \frac{4}{8} = 0.3 \leq 0.4
\]

On the contrary, \( \{1, 4, 5\} \) will be a large itemset since

\[
(0.1 + 0.5 + 1) \times \frac{2}{8} = 0.4 \geq 0.4
\]

By the same argument, \( \{5\}, \{1, 2, 5\} \) will be large weighted itemsets.

Although itemset \( \{1, 4\} \) has a greater support than that of \( \{1, 2, 5\} \), it seem to be true that the latter otherwise make a greater profit than the former can do. In this case, we say that itemset \( \{1, 2, 5\} \) is more interesting than itemset \( \{1, 4\} \), or the interestingness of itemset \( \{1, 2, 5\} \) is greater than that of itemset \( \{1, 4\} \).
CHAPTER 3

MINING ASSOCIATION RULES WITH ADJUSTABLE INTERESTINGNESS

In this chapter, we design a new concept, adjustable interestingness. Furthermore, a novel approach in mining association rules based on adjustable interestingness is introduced.

3.1. Interestingness and interesting itemsets

Based on the definitions of weighted itemsets in previous chapter, we extend the definitions of interestingness and interesting itemsets.

**Definition 1.** The interestingness of an itemset \( X \), denoted \( \text{interest}(X) \), is the coefficient correlation between the number of transactions in which it occurs as a subset and the total weight of its items, or mathematically,

\[
\text{interest}(X) = \left( \sum_{j \in X} w_j \right) \text{support}(X)
\]

In order to find the interesting itemsets, the threshold, minimum interestingness \( (\text{min}_\text{int}) \) must be specified.

**Definition 2.** An itemset \( X \) is called an interesting itemset if the interestingness of the itemset \( X \) is greater than or equal to the interestingness threshold, or mathematically,

\[
\text{interest}(X) \geq \text{min}_\text{int}
\]
Example 3.1. From the database in Table 1 and 2, we can calculate the interestingness of itemsets as the following table. The itemsets are sorted into descending order of their interestingness.

<table>
<thead>
<tr>
<th>Itemset</th>
<th>W*</th>
<th>S*</th>
<th>I*</th>
</tr>
</thead>
<tbody>
<tr>
<td>{5}</td>
<td>1</td>
<td>62.5%</td>
<td>0.625</td>
</tr>
<tr>
<td>{2, 5}</td>
<td>1.2</td>
<td>37.5%</td>
<td>0.45</td>
</tr>
<tr>
<td>{1, 4, 5}</td>
<td>1.6</td>
<td>25%</td>
<td>0.4</td>
</tr>
<tr>
<td>{4, 5}</td>
<td>1.5</td>
<td>25%</td>
<td>0.375</td>
</tr>
<tr>
<td>{3, 5}</td>
<td>1.3</td>
<td>25%</td>
<td>0.325</td>
</tr>
<tr>
<td>{1, 4}</td>
<td>0.6</td>
<td>50%</td>
<td>0.3</td>
</tr>
<tr>
<td>{1, 5}</td>
<td>1.1</td>
<td>25%</td>
<td>0.275</td>
</tr>
<tr>
<td>{4}</td>
<td>0.5</td>
<td>50%</td>
<td>0.25</td>
</tr>
<tr>
<td>{2, 4, 5}</td>
<td>1.7</td>
<td>12.5%</td>
<td>0.2125</td>
</tr>
<tr>
<td>{2, 3, 5}</td>
<td>1.5</td>
<td>12.5%</td>
<td>0.1875</td>
</tr>
<tr>
<td>{1, 2, 5}</td>
<td>1.3</td>
<td>12.5%</td>
<td>0.1625</td>
</tr>
</tbody>
</table>

* W, S and I are acronyms for Weight, Support and Interestingness, respectively.

Table 3. Itemsets sorted into descending order of their interestingness

If the value of min_int is 0.3, we obtain six interesting itemsets; these are: {5}, {2, 5}, {1, 4, 5}, {4, 5}, {3, 5}, {1, 4}. Of these six interesting itemsets, five contain item 5 which represents for pens. It proves that the interestingness of an itemset is made up of its weight and support.

3.2. Interestingness constraints

By sorting the itemsets into descending order of their interestingness, we have two diverse ways to mine the most interesting itemsets. The first is to set a threshold for minimum interestingness, or min_int. In the example 3.1, when the min_int value is set to 0.3, there are six most interesting itemsets found in the database. That is, there are only six itemsets whose interestingness are greater than or equal to 0.3.
Since the number of itemsets found is unpredictable, it may be cumbersome when \( \text{min\_int} \) is lowered to 0.

In this thesis, we present an alternative way to mine the most interesting itemsets. By this way, the \( \text{min\_int} \) is adjusted throughout the mining process. The term \text{constraint} is defined as the number of itemsets for which we desire to mine and it must be specified. From the example 3.1, if the \text{constraint} value is set to 5, we can mine five most interesting itemsets whose interestingness are 0.325 or over. Therefore, the \( \text{min\_int} \) value is adjusted to 0.325 afterward. Similarly, if the \text{constraint} is 10, the \( \text{min\_int} \) is adjusted to 0.1875 since the interestingness of ten most interesting items are greater or equal to 0.1875. It is clear that the greater the \text{constraint} is, the smaller the \( \text{min\_int} \) is adjusted to.

### 3.3. Motivation behind interesting itemsets and adjustable interestingness

By setting the interestingness of an itemset, we can get a balance between the two measures, weights and supports. If supports are separated from weights, we can only find itemsets having sufficient support. However, this may ignore some interesting knowledge. Special items and special group of items may be specified individually and have higher priority. For example, there are few customers buying pens, but the profit the pens make is much more than that of other products. As a matter of course, the store clerk will want to put the pens under the promotion rather than others. For this reason, the weight which is a measure of the important of an item is applied.

The interestingness of an item can be computed at the multiplication of weight and support. Interestingness, in some case, can be “the potential usefulness of the knowledge” but it seems to be difficult to understand. It is clear that most end-users are not statisticians, they thus have trouble setting the threshold for \( \text{min\_int} \). Putting a query “Show me twenty most interesting itemsets” is definitely more comprehensible than “Please list itemsets whose interestingness are greater or equal to 0.5”.

Furthermore, it is impractical to generate entire set of interesting itemsets. Our purpose is to mine only most interesting ones. Hence, we design a new concept, adjustable interestingness, in this thesis.

Related work

Our past work [5] addressed the problem of mining association rules with different supports, provided that most of proposed algorithms employing the same minimum support, \( \text{minsup} \), to generate itemsets. In some situation, it may not be appropriate. There may be some itemsets with smaller supports than \( \text{minsup} \) value, however, they can generate more useful rules. By setting the minimum support for each item, we generate closed sets using a triple \( \text{minsup-itemset-tidset} \) and then restrict the number of itemsets to be found, thus the search space is fairly reduced.
CHAPTER 4

ALGORITHM FOR MINING ASSOCIATION RULES WITH ADJUSTABLE INTERESTINGNESS (MARAI)

The main idea of this thesis, adjustable interestingness, has been introduced in the previous chapter. In this case, the meaning of support has been changed, and the CHARM algorithm cannot be applied. In this chapter, we propose the MARAI algorithm as solutions. Thorough experimental performance indicates that our algorithm works effectively in large databases.

4.1. Motivation

It may seem that the CHARM algorithm [7] can be adopted in the interestingness constraints case. However, the meaning of the support, called interestingness, has been changed. Therefore, it is not necessarily true that all subsets of a large weighted itemset are large weighted itemsets.

Example 4.1. Take the database and the set of transaction from example 2.2. For all the possible itemsets, there are only three large weighted itemsets, which are \{1, 4, 5\}, \{5\}, \{1, 2, 5\}. However, \{1, 5\} is not a large weighted itemset, even though it is a subset of both itemset \{1, 4, 5\} and itemset \{1, 2, 5\}.

In this situation, the new algorithm, called MARAI algorithm, is proposed to solve above problem. The framework of our proposed algorithm for mining association rules with adjustable interestingness is similar to the CHARM algorithm, but the detailed steps contain some significant differences. To begin with, we also mine only the closed sets [7]. Closed sets are lossless in the sense that they uniquely determine the set of all frequent itemsets and their exact frequency. The set of all
termine the set of all frequent itemsets and their exact frequency. The set of all closed frequent itemsets can be orders of magnitude smaller than the set of all frequent itemsets, especially on dense databases. Before introducing the new algorithm, we will reiterate some concepts represented in previous chapters and describe the problem setting and preliminaries.

4.2. Preliminaries

In this section, we describe the conceptual framework of closed sets [7]. Let $I$ be a sets of items, and $D$ a database of transactions. Each transaction has a unique identifier ($tid$) and contains a set of items. Let $T$ be the sets of all tids. A set $X \in I$ is called an itemset, and a set $Y \in T$ is called a tidset. For convenient, we write an itemset $\{A, C, W\}$ as ACW, and a tidset $\{2, 4, 5\}$ as 245. For an itemset $X$, we denote the set of all tids that contain $X$ as a subset by $t(X)$. For a tidset $Y$, we denote the set of items appearing in all the tids of $Y$ by $i(Y)$. The notion $X \times t(X)$ refers to an itemset-tidset pair, or an IT-pair [7].

<table>
<thead>
<tr>
<th>DISTINCT BOOK ITEMS</th>
<th>DATABASE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Item ID</strong></td>
<td><strong>Weight</strong></td>
</tr>
<tr>
<td>A</td>
<td>0.2</td>
</tr>
<tr>
<td>C</td>
<td>0.2</td>
</tr>
<tr>
<td>D</td>
<td>0.3</td>
</tr>
<tr>
<td>T</td>
<td>0.4</td>
</tr>
<tr>
<td>W</td>
<td>0.1</td>
</tr>
<tr>
<td>6</td>
<td>C D T</td>
</tr>
</tbody>
</table>

**Figure 2.** Example database

Consider the database shown in Figure 2. There are five different items, $I = \{A, C, D, T, W\}$, and six transactions $T = \{1, 2, 3, 4, 5, 6\}$. The table on the left shows the information about the items in a book store. The information includes the identifi-
cation of the items, the author names of such items and the given weight of each item. The table on the right shows the transaction database. For each transaction, there will be a transaction identifier and a set of items in which the transaction contains. Suppose there are only 5 items and totally 6 transactions in the transaction database.

The corresponding tidset of ACW, denoted \( t(ACW) \), is 1345 since there are 4 transactions 1, 3, 4, 5 containing ACW as a subset. The corresponding itemset of 245, denoted \( i(245) \), is CDW as the sets of items \{C, D, W\} is common to all the tids 2, 4 and 5.

It is worth mentioning that \( t(X) = \cap_{x \in X} t(x) \), and \( i(Y) = \cap_{y \in Y} i(Y) \).

**Example 4.2.** \( t(ACW) = t(A) \cap t(C) \cap t(W) = 1345 \cap 123456 \cap 12345 = 1345 \) and \( i(245) = i(2) \cap i(4) \cap i(5) = CDW \cap ACDW \cap ACDTW \).

The *support* of an itemset \( X \), denoted \( \sigma(X) \), is the number of transactions in \( D \) where it occurs as a subset [1], i.e., \( \sigma(X) = |t(X)| \) [7]. The *weight* of an itemset \( X \), denoted \( \text{weight}(X) \), is the total weight of items in which the itemset \( X \) contains, i.e., \( \text{weight}(X) = \sum_{j \in X} w_j \) [6].

We use the notation \( \omega(X) \) to refer to the interestingness of the itemset \( X \). As described in previous chapter, \( \text{interest}(X) = \left( \sum_{j \in X} w_j \right) \text{support}(X) \). We thus have \( \omega(X) = \sigma(X) \times \text{weight}(X) \).

**Example 4.3.** \( \sigma(ACW) = \frac{4}{6} = 67\% \), \( \text{weight}(ACW) = 0.2 + 0.2 + 0.1 = 0.5 \), and \( \omega(ACW) = 67\% \times 0.5 = 0.33 \).
The table below shows all 31 itemsets which are sorted into descending order of the interestingness.

<table>
<thead>
<tr>
<th>Itemset</th>
<th>W*</th>
<th>S*</th>
<th>I* ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTW</td>
<td>0.9</td>
<td>50%</td>
<td>0.45</td>
</tr>
<tr>
<td>CT</td>
<td>0.6</td>
<td>67%</td>
<td>0.4</td>
</tr>
<tr>
<td>ACT</td>
<td>0.8</td>
<td>50%</td>
<td>0.4</td>
</tr>
<tr>
<td>CTW</td>
<td>0.7</td>
<td>50%</td>
<td>0.35</td>
</tr>
<tr>
<td>ATW</td>
<td>0.7</td>
<td>50%</td>
<td>0.35</td>
</tr>
<tr>
<td>CD</td>
<td>0.5</td>
<td>67%</td>
<td>0.33</td>
</tr>
<tr>
<td>ACW</td>
<td>0.5</td>
<td>67%</td>
<td>0.33</td>
</tr>
<tr>
<td>CDT</td>
<td>0.9</td>
<td>33%</td>
<td>0.3</td>
</tr>
<tr>
<td>AT</td>
<td>0.6</td>
<td>50%</td>
<td>0.3</td>
</tr>
<tr>
<td>CDW</td>
<td>0.6</td>
<td>50%</td>
<td>0.3</td>
</tr>
<tr>
<td>AC</td>
<td>0.4</td>
<td>67%</td>
<td>0.27</td>
</tr>
<tr>
<td>T</td>
<td>0.4</td>
<td>67%</td>
<td>0.27</td>
</tr>
<tr>
<td>ACDW</td>
<td>0.8</td>
<td>33%</td>
<td>0.27</td>
</tr>
<tr>
<td>TW</td>
<td>0.5</td>
<td>50%</td>
<td>0.25</td>
</tr>
<tr>
<td>CW</td>
<td>0.3</td>
<td>83%</td>
<td>0.25</td>
</tr>
<tr>
<td>DT</td>
<td>0.7</td>
<td>33%</td>
<td>0.23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>W*</th>
<th>S*</th>
<th>I* ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACD</td>
<td>0.7</td>
<td>33%</td>
<td>0.23</td>
</tr>
<tr>
<td>ADW</td>
<td>0.6</td>
<td>33%</td>
<td>0.2</td>
</tr>
<tr>
<td>C</td>
<td>0.2</td>
<td>100%</td>
<td>0.2</td>
</tr>
<tr>
<td>ACDTW</td>
<td>1.2</td>
<td>17%</td>
<td>0.2</td>
</tr>
<tr>
<td>AW</td>
<td>0.3</td>
<td>67%</td>
<td>0.2</td>
</tr>
<tr>
<td>D</td>
<td>0.3</td>
<td>67%</td>
<td>0.2</td>
</tr>
<tr>
<td>DW</td>
<td>0.4</td>
<td>50%</td>
<td>0.2</td>
</tr>
<tr>
<td>ACDT</td>
<td>1.1</td>
<td>17%</td>
<td>0.18</td>
</tr>
<tr>
<td>ADTW</td>
<td>1</td>
<td>17%</td>
<td>0.17</td>
</tr>
<tr>
<td>CDTW</td>
<td>1</td>
<td>17%</td>
<td>0.17</td>
</tr>
<tr>
<td>AD</td>
<td>0.5</td>
<td>33%</td>
<td>0.17</td>
</tr>
<tr>
<td>ADT</td>
<td>0.9</td>
<td>17%</td>
<td>0.15</td>
</tr>
<tr>
<td>DTW</td>
<td>0.8</td>
<td>17%</td>
<td>0.13</td>
</tr>
<tr>
<td>A</td>
<td>0.2</td>
<td>67%</td>
<td>0.13</td>
</tr>
<tr>
<td>W</td>
<td>0.1</td>
<td>83%</td>
<td>0.08</td>
</tr>
</tbody>
</table>

*W, S and I are acronyms for Weight, Support and Interestingness, respectively.

Table 4. Itemsets sorted into descending order of the interestingness

An itemset is *interesting* if its interestingness is greater than or equal to a user-specified minimum interestingness (min_int) value, i.e., if \( \omega(X) \geq \text{min_int} \). An interesting itemset is called *closed* if there exists no proper superset \( Y \supset X \) with \( \sigma(X) = \sigma(Y) \). The term closed used in this thesis is similar to the term closed defined in [3, 7]. A set of interesting closed itemsets is a subset of the corresponding set of interesting itemsets. This subset is necessary and sufficient to cover all of the information about the interesting itemsets.
Example 4.4. Given min_int be 2. There are 23 interesting itemsets as follows:

<table>
<thead>
<tr>
<th>Support</th>
<th>Itemsets</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% (6)</td>
<td>C</td>
</tr>
<tr>
<td>83% (5)</td>
<td>CW</td>
</tr>
<tr>
<td>67% (4)</td>
<td>CT, CD, ACW, AC, T, AW, D</td>
</tr>
<tr>
<td>50% (3)</td>
<td>ACTW, ACT, CTW, ATW, AT, CDW, TW, DW</td>
</tr>
<tr>
<td>33% (2)</td>
<td>CDT, ACDW, DT, ACD, ADW</td>
</tr>
<tr>
<td>17% (1)</td>
<td>ACDTW</td>
</tr>
</tbody>
</table>

Table 5. All interesting itemsets

We obtain 10 closed itemsets which are underlined; these are: C, CW, CT, CD, ACW, ACTW, CDW, CDT, ACDW and ACDTW. As the example shows, if \( F \) denotes the sets of interesting itemsets, and \( C \) the set of closed ones, then we have \( C \subseteq F \subseteq I \). Generally, the the set \( C \) can be orders of magnitude smaller than the set \( F \), which itself is orders of magnitude smaller than the set of all itemsets \( I \) (especially for dense database).

4.3. Basic properties of itemset-tidset pairs

A closure of an itemset \( X \), denoted \( c(X) \) is the smallest closed set that contain \( X \) [7]. Recall that \( i(Y) \) is the sets of items common to all the tids in the tidset \( Y \), while \( t(X) \) is the tids common to all the items in \( X \). The closure of an itemset \( X \) can be computed by mapping \( t(X) \) to its image in the itemset space, i.e., \( c(X) = i \circ t(x) = i(t(X)) \) [7]. An itemset \( X \) is closed if and only if \( X = c(X) \). The support of an itemset \( X \) is also equal to the support of its closure, i.e., \( \sigma(X) = \sigma(c(X)) \).
Example 4.5. Since \( c(ACW) = i(t(ACW)) = i(1345) = ACW \), itemset \( ACW \) is closed.

For any two IT-pairs, \( X \times t(X_i) \) and \( X \times t(X_i) \), if \( X \subseteq X_i \) then \( t(X_i) \supseteq t(X_i) \).

Example 4.6. For \( ACW \subseteq ACTW \), \( t(ACW) = 1345 \supseteq 135 = t(ACTW) \).

Let \( X \times t(X_i) \) and \( X \times t(X_i) \) be any two IT-pairs. We have four properties of IT-pairs.

**Property 1.** If \( t(X_i) = t(X_i) \) then \( c(X_i) = c(X_i) = c(X_i \cup X_i) \) [7]. It follows that \( \omega(X_i) \leq \omega(X_i \cup X_i) \geq \omega(X_i) \).

\[ \quad \]

\[ \text{If } t(X_i) = t(X_i) \text{, then obviously } i(t(X_i)) = i(t(X_i)), \text{ i.e., } c(X_i) = c(X_i). \text{ Further, } t(X_i) = t(X_i) \text{ implies that } t(X_i \cup X_i) = t(X_i \cap t(X_i)) = t(X_i). \text{ We thus have } i(t(X_i \cup X_i)) = i(t(X_i)), \text{ giving us } c(X_i \cup X_i) = c(X_i). \]

From \( t(X_i) = t(X_i) = t(X_i \cup X_i) \), we have \( \sigma(X_i) = \sigma(X_i) = \sigma(X_i \cup X_i) \). Further, \( \text{weight}(X_i) \leq \text{weight}(X_i \cup X_i) \geq \text{weight}(X_i) \), then \( \omega(X_i) \leq \omega(X_i \cup X_i) \geq \omega(X_i) \).

Note that \( \omega(X) = \sigma(X) \times \text{weight}(X) \).

Example 4.7. If \( t(AC) = t(AW) = 1345 \), \( c(AC) = c(AW) = c(ACW) = ACW \) then we conclude that \( \omega(AC) = 0.27 \leq \omega(ACW) = 0.33 \geq \omega(AW) = 0.2 \).

This property implies that we can replace every occurrence of \( X_i \) with \( X_i \cup X_i \), and we can remove the element \( X_i \) from further consideration, since its closure is identical to the closure of \( X_i \cup X_i \) but it is not as interesting as \( X_i \cup X_i \).

**Property 2.** If \( t(X_i) \subseteq t(X_i) \), then \( c(X_i) \neq c(X_i) \), but \( c(X_i) = c(X_i \cup X_i) \), thereby \( \omega(X_i) \leq \omega(X_i \cup X_i) \).

\[ \quad \]

\[ \text{If } t(X_i) \subseteq t(X_i) \text{, then } t(X_i \cup X_i) = t(X_i) \cap t(X_i) = t(X_i) \neq t(X_i), \text{ giving us } c(X_i \cup X_i) = c(X_i) \neq c(X_i) \text{ and } \omega(X_i) \leq \omega(X_i \cup X_i) \]
Example 4.8. From the above database, \(45 = t(ACD) \subseteq t(ACW) = 1345\), we have 
\[c(ACD) = c(ACDW) = ACDW\] and \(0.23 = \omega(ACD) \leq \omega(ACDW) = 0.27\).

We will use this observation to replace every occurrence of \(X_i\) with \(X_i \cup X_j\), since they have identical closures and the interestingness of \(X_i\) is less than that of \(X_i \cup X_j\). However, since \(c(X_i) \neq c(X_j)\), we cannot remove itemset \(X_i\) as it may generate itemsets more interesting than itemset \(X_i \cup X_j\).

**Property 3.** If \(t(X_i) \supseteq t(X_j)\), then \(c(X_i) \neq c(X_j)\), but \(c(X_i) = c(X_i \cup X_j)\), giving \(\omega(X_i) \leq \omega(X_i \cup X_j)\).

Similar to property 2 above.

**Property 4.** If \(t(X_i) \neq t(X_j)\), then \(c(X_i) \neq c(X_j) \neq c(X_i \cup X_j)\) [7]. It follows that \(\omega(X_i) \neq \omega(X_i) \neq \omega(X_i \cup X_j)\).

▶ If \(t(X_i) \neq t(X_j)\), then clearly \(t(X_i \cup X_j) = t(X_i) \cap t(X_j) \neq t(X_i) \neq t(X_j)\), giving us \(c(X_i \cup X_j) \neq c(X_i) \neq c(X_j)\), and \(\omega(X_i) \neq \omega(X_i) \neq \omega(X_i \cup X_j)\). ◀

This property means that neither itemset \(X_i\) nor itemset \(X_j\) can be eliminated, both of which lead to different closures, then can generate itemsets with different interestingness.

### 4.4. MARAI: Algorithm design and implementation

In this section, we now present MARAI, an algorithm for mining association rules with adjustable interestingness. The pseudo-code for MARAI appears in Figure 3. The algorithm starts by setting the \(\text{min\_int}\) value to 0 and initializing the prefix class \([P]\) to the single itemsets and their tidsets in Line 1 and 2. The main computation is performed in MARAI-EXTEND which return the set of interesting closed itemset \(C\). Based on the adjustable minimum interestingness, it is possible to generate a summary of the set of interesting closed itemsets. Then there is no need to explicitly count the minimum support, \(\text{min\_sup}\).
MARAI-EXTEND is responsible for considering each combination of IT-pairs appearing in the prefix class \([P]\). For each IT-pair \(X \times t(X)\) (Line 5), it combines it with the other IT-pair \(X_\ell \times t(X_\ell)\), that has support more than its support (Line 7). Each \(X_\ell\) generates a new prefix class \([P_\ell]\) which is initially empty (Line 6). At line 8, the two IT-pairs are combined to produce a new pair \(X \times Y\), where \(X = X_\ell \cup X_i\) and \(Y = t(X_\ell) \cap t(X_i)\). Line 9 tests which of the four IT-pair properties can be applied by calling MARAI-PROPERTY. Once all properties have been processed, we recursively explore the new class \([P_\ell]\) in a depth-first manner (Line 10). \(X\), an extension of \(X_i\), is determined if it is an interesting itemset and if it was already in closed set \(C\) (Line 12). In the case that the support of \(X\) is greater than \(\text{min\_int}\) value, we then insert the itemset \(X\) into the set of closed itemsets \(C\) (Line 13). Nevertheless, we only mine the most interesting itemsets, i.e., the number of itemsets found must not exceed the value of \(\text{constraint}\). Thus, if the number of itemsets in \(C\) is more than \(\text{constraint}\) value (Line 14), then we have to eliminate the least interesting itemset (Line 15) so that the number of itemsets is always equal to \(\text{constraint}\). The minimum interestingness, \(\text{min\_int}\), will be set to the interestingness of the least interesting itemset afterward, i.e., the minimum interestingness has been adjusted (Line 16).

At this stage any closed itemset containing \(X_\ell\) has already been generated. We then return to Line 5 to process the next IT-pair in \([P]\).

The result is that we obtain as many as \(\text{constraint}\) most interesting itemsets.
MARAI$(D,\text{constraint})$:
1. $\text{min\_int} = 0$
2. $[P] = \{X_i \times t(X_i) : X_i \in I \land \omega(X_i) \geq \text{min\_int}\}$
3. MARAI – EXTEND$([P], C = \phi)$
4. return $C$ //all interesting closed itemsets

MARAI – EXTEND$([P], C)$:
5. for each $X_i \times t(X_i)$ in $[P]$
6. 
7. for each $X_i \times t(X_i)$ in $[P]$, with $\sigma(X_i) \geq \sigma(X_i)$
8. 
9. MARAI - PROPERTY$([P],[P_i])$
10. if $([P_i] \neq \phi)$ then MARAI - EXTEND$([P_i], C)$
11. delete $[P_i]$
12. if $\omega(X) > \text{min\_int}$ and X is not subsumed then
13. 
14. if $|C| > \text{constraint}$ then
15. 
16. $\text{min\_int} = \min\{\omega(X_i) \mid 1 \leq i \leq \text{constraint}\}$

MARAI – PROPERTY$(([P],[P_i]))$:
17. if $(\omega(X) \geq \text{min\_int})$ then
18. 
19. Remove $X_i$ from $[P]$
20. Replace all $X_i$ with $X$
21. else if $t(X_i) \subseteq t(X_i)$ then //Property 2
22. Replace all $X_i$ with $X$
23. else if $t(X_i) \supseteq t(X_i)$ then //Property 3
24. Remove $X_i$ from $[P]$
25. Add $X \times Y$ to $[P]$
26. else if $t(X_i) \neq t(X_i)$ then //Property 4
27. Add $X \times Y$ to $[P]$

**Figure 3.** The MARAI algorithm
Figure 4. Search process using adjustable interestingness

Example 4.9. Figure 4 shows how MARAI works on our example database. To begin with, let constraint be 5. We use the pseudo-code in Figure 3 to illustrate the computation. We initialize the root class as \([ P ] = \{ A \times 134!, C \times 12345, D \times 245\}, T \times 135\}, W \times 1234!\} \) in line 2. At line 5, we first process the node \( A \times 134! \); it will be combined with the remaining elements in line 7.

\[
X_i = A \times 134!
\]

\[
X_i = C \times 12345 \rightarrow \text{Replace } A \text{ with } AC \text{ //Prop. 2}
\]

\[
X_i = AC \times 134!
\]

\[
X_i = D \times 245!; \text{ Add } AC^! \text{ to } [ P_i ]; \ P_i = \{ AC \} \text{ //Prop. 4}
\]

\[
X_i = T \times 135!; \text{ Add } AC^{-} \text{ to } [ P_i ]; \ P_i = \{ AC, AC^{-} \} \text{ //Prop. 4}
\]

\[
X_i = W \times 1234!; \text{ Replace } AC \text{ with } AC^V
\]

\[
X_i = ACV \times 134!, P_i = \{ ACDV, ACTV \}
\]

We next make a recursive call to MARAI-EXTEND with \( P_i = \{ ACDV, ACTV \} \).

\[
X_i = ACDV \times 4\xi
\]

\[
X_i = ACTV \times 13\xi; \text{ Add } ACDTV \text{ to } [ P_{ii} ]; \ P_{ii} = \{ ACDTV \} \text{ //Prop. 4}
\]
MARAI then makes a recursive call to process class \( P_{11} = \{ \text{ACDT}\} \). Since there is only one element, we jump to line 13, where \( \text{ACDT}\) is added to the interesting closed set \( C \). When we return, the \( \text{ACDW}\) is complete, thus \( \text{ACDW}\) itself is added to \( C \). We next look the element \( \text{ACTV}\) in \( [P_1]\) \( \text{ACTV}\). Since it is the last element, we can move to line 13 and add \( \text{ACTV}\) to \( C \).

The \( A \) (now \( \text{ACV}\)) branch is complete and \( \text{ACV}\) can be inserted to \( C \) likewise.

When we process \( X = \text{CE} \times 12345 \), we find that \( t(C) \supseteq t(D) \), \( t(C) \supseteq t(T) \), and \( t(C) \supseteq t(W) \). Since property 2 applies, we remove \( D \), \( T \), and \( W \) and add \( \text{CE} \), \( \text{CT} \), and \( \text{CW} \) to \( [P_2] \).

We next make a recursive call to MARAI-EXTEND with class \( P_2 = \{ \text{CE}, \text{CT}, \text{CW} \} \).

\[
\begin{align*}
X_i &= \text{CE} \times 245t \\
X_i &= \text{CT} \times 135t: \text{Add CD}^{-} \text{ to } [P_{22}]: P_{22} = \{ \text{CD}^{-} \} \quad \text{//Prop. 4} \\
X_i &= \text{CW} \times 1234: \text{Add CDV to } [P_{22}]: P_{22} = \{ \text{CD}^{-}, \text{CDV} \} \quad \text{//Prop. 4}
\end{align*}
\]

A recursive call of MARAI-EXTEND is taken with class \( P_{22} = \{ \text{CD}^{-}, \text{CDV} \} \).

\[
\begin{align*}
X_i &= \text{CD}^{-} \times 5t \\
X_i &= \text{CDV} \times 24t: \text{Add } \text{CDTV to } [P_{22}]: P_{22} = \{ \text{CDTV} \}
\end{align*}
\]

Since \( \text{CDTV} \times 5 \) is subsumed with \( \text{ACDT} \times 5 \) in \( C \), we discard it. The branch of \( \text{CD}^{-} \) is full done, thus \( \text{CD}^{-} \) is added to \( C \).

We then process element \( \text{CDV} \) in class \( [P_{22}] \). Since it is the last element in \( [P_{22}] \), \( \text{CDV} \) will be added to \( C \). We have initially set the constraint value to 5, i.e., we only desire to mine five most interesting itemsets. Nevertheless, the number of itemsets that \( C \) contains is six; we have to prune the least interesting one. Hence, \( \text{ACDTW} \) will be removed from \( C \) since its interestingness is less than or equal to any other itemset in \( C \). \( C \) now has only 5 itemsets; these are \( \text{ACDV}, \text{ACTV}, \text{ACV}, \text{CD}^{-}, \text{CDV} \) with the interestingness of 0.27, 0.45, 0.33, 0.3 and 0.3, respectively. The minimum interestingness, \( \text{min}_\text{int} \), consequently is increased from 0 to 0.27.
Since it cannot be extended further, $\mathbf{Cl}$ is inserted to $C$. Simultaneously, $\mathbf{ACDW}$ is eliminated from $C$ and $min\_int$ value is set to 0.3.

We next look the element $\mathbf{CT}$ of class $[P_2]$.

\[
X = \mathbf{CT} \times 135
t
\]

\[
X_j = \mathbf{CW} \times 1234: \text{Add } \mathbf{CTV} \text{ to } [P_{23}]: P_{23} = \{\mathbf{CTV}\} //\text{Prop. 4}
\]

$\mathbf{CTV} \times 13\xi$ is subsumed with $\mathbf{ACTV} \times 13\xi$ in $C$, thus it will not be added to $C$.

Since its branch is end, we then add $\mathbf{CT}$ to $C$, deleting $\mathbf{CDW}$ at the same time.

The last element $\mathbf{CW}$ of class $[P_2]$ is next processed. Since the interestingness of $\mathbf{CW}$ is 0.25, less than the value of $min\_int$, it cannot be a generator and is thus pruned. At this step, no new recursion is made and the final interesting closed set $C$ consists of five bold, uncrossed IT-pairs shown in Figure 4.

The example above shows that MARAI reduces significantly search space by skipping many uninteresting itemsets and pruning those cannot generate interesting itemsets at the earlier stage. Therefore, the total time needed for the mining is less than the non-constraint case.

4.5. Experimental Evaluation

A performance study is carried out for the algorithm MARAI. A series of experiments were conducted on a 300MHz Pentium PC with 192MB of memory, running Windows XP. The timing is measured by the CPU time calculated from the built-in timing functions.

<table>
<thead>
<tr>
<th>Database</th>
<th># Items</th>
<th>Avg. Length</th>
<th># Records</th>
<th>Level Searched</th>
</tr>
</thead>
<tbody>
<tr>
<td>cosmetic</td>
<td>67</td>
<td>29</td>
<td>10123</td>
<td>8</td>
</tr>
<tr>
<td>census</td>
<td>98</td>
<td>15</td>
<td>3140</td>
<td>11</td>
</tr>
</tbody>
</table>

Table 6. Database characteristics
Table 6 shows the characteristics of the real databases used in our experiments. It shows the number of items, the average transaction length and the number of records in each database. The table additionally shows the maximum level of search that MARAI performed to discover the most interesting rules when constraint value is set to 100.

In order to initialize the experiment setup, we use a dense database, namely Cosmetic, which we obtained from a cosmetic retailing store. In the generation of the weights, we assume that the weights are equivalent to the price of its product. For the synthetic itemsets, such as product categories, we generate the weights according to our evaluation of their values to the salespeople. We next map the quantitative attributes to the binary type, each of which is mapped to a range of the set of consecutive integers [3]. For example, suppose an attribute is age of customers, we would expect rules related to five ranges of age: under 20, from 20 to 29, from 30 to 39, from 40 to 49 and over 50. After discretized [3], the database consists of 67 items with more than 10,000 transactions.

![Figure 5. Performance of the MARAI algorithm on Cosmetic](image-url)
Figure 5 shows how MARAI works on Cosmetic with an increasing number of constraint value, we kept all other parameters constant. In this figure, the time shown on y-axis is given in seconds. There is a tradeoff between the constraint value presented on x-axis and the running time. As can be seen, with constraint = 5, MARAI takes 19 seconds to show five most interesting itemsets. The more time will be needed for the case of greater constraint. It takes approximately 3 minutes when constraint value is set to 100. However, at the point when many itemsets can satisfy the threshold, the execution time will remain constant. From the figure, the execution time rises with the increasing number of constraint value linearly, implying that the complexity of the algorithm is O(constraint). The interesting rules are listed in more detail in Appendix.

![Figure 6. Performance of the MARAI algorithm on Census](image)

In the following experiment, we use a sparse database, namely Census, which has more items but fewer records than Cosmetic. From figure 6, we observe that a similar amount of time is taken, implying that the execution time increases with the numbers of both items and transactions.
CHAPTER 5

CONCLUSION

In this thesis, we have introduced the concept of adjustable interestingness, making the mining of the association rules possible to be interactive with the users. The users can set the number of interesting itemsets they desire to mine, instead of employing the minimum support value. This is practical and useful since the support value is much incomprehensible when weights are applied. Our ideas base on the fact that the minimum interestingness value may be adjusted during mining process. Finally, we proposed an algorithm to solve our problem.

Future work

We have studied the mining of association rules in binary data. However, a transaction database can be quantitative type. In the experiments mentioned above, we transfer the quantitative database to the binary type by using discretization [3] and it may cause losing much information. In this case, we should utilize fuzzy set [8] to overcome this problem. A fuzzy set is represented by a membership function which assigns to each value of the attribute a value between 0 and 1 to indicate how much this value belongs to the fuzzy set [8].

We applied the MARAI algorithm to real databases in business and census areas, and it seems to be feasible. Clinical databases have accumulated large quantities of information about patients and their medical conditions which could provide new medical knowledge. We are currently in the process of applying our algorithm to real clinical databases of a hospital. We observe that interestingness plays a significant role in mining useful rules in that a certain disease usually have many symptoms, each of which should have diverse importance, thus requiring employing adjustable interestingness.
REFERENCES


APPENDIX

The following rules are the rules mined from the experiments. We set constraint value to 100, thus we only mine a hundred most interesting itemsets. From these itemsets, we extract a number of interesting rules which are sorted into descending order of the interestingness of the itemsets. We assume that the threshold for all rules is 0.75 in Cosmetic, i.e., the confidences of all rules are greater or equal to 75%. The threshold of 0.7 is applied in Census.

Rules extracted from the database Cosmetic

(a) 98.6% customers buying Geo. Nature Anti Trouble Cream product are women.

(b) 98.6% customers buying Geo. Nature Anti Trouble Skin Emulsion product are women.

(c) 99.1% customers buying Geo. UV White Essence product are women.

(d) 98.7% customers buying Geo. Loose Finish Powder product are women.

(e) 97.8% customers buying Geo. Nature Fluid Serum product are women.

(f) 70.0% customers buying Geo. UV White Essence product are from 20 to 29 years of age.

(g) 97.9% customers buying Geo. Nature Firming Eye Cream product are women.

(h) 98.0% customers buying Geo. White Serum product are women.

(i) 100% customers buying Geo. White Serum product and buying for more than 1,000,000 VND are women.

(j) 89% customers in summer are women.

(k) 85.8% customers buying For Men category are men.
(l) 98.5% customers buying Geo UV White category buy Geo. White Serum product.

(m) 86.6% male customers buy for fewer than 500,000 VND.

(n) 76.9% buying Geo. White Water Toner product buy for more than 1,000,000 VND

**Rules extracted from the database **Census**

(a) 87.8% districts: State = Texas $\rightarrow$ Population = High

(b) 89.5% districts: White = Low $\rightarrow$ Population = High

(c) 73.2% districts: State = Texas $\rightarrow$ White = Medium

(d) 74% districts: State = Texas $\rightarrow$ Male = Medium

(e) 70.5% districts: Hispanic = High $\rightarrow$ White = Low

(f) 97.7% districts: Income = High $\rightarrow$ Population = High

(g) 70.4% districts: Income = Medium $\rightarrow$ White = Low

(h) 98.8% districts: College Graduate = High $\rightarrow$ Population = High

(i) 76.7% districts: College Graduate = High $\rightarrow$ Population = High and Hispanic = High

(j) 72.1% districts: White = Low $\rightarrow$ Density = Low

(k) 85.3% districts: State = Illinois $\rightarrow$ White = Low

(l) 93.1% districts: State = Georgia $\rightarrow$ Population = High

(m) 91.9% districts: State = New York $\rightarrow$ Density = Medium

(n) 84.3% districts: State = Illinois $\rightarrow$ High School Graduate = Medium