Discovering Association Patterns in Large Spatio-temporal Databases

by

Ming-Ho Eric Lee

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF PHILOSOPHY
in
Department of Computing
The Hong Kong Polytechnic University

September 2006

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

______________________________ (Signed)

________Ming-Ho Eric Lee______ (Name of Student)

Abstract

Data mining is concerned with the discovery of hidden patterns in large databases. Among the different types of patterns that can be discovered, “association” patterns are the most important. This is because the discovery of association patterns can lead more easily to the discovery of other patterns for such data mining tasks as classification, clustering or prediction. Given a set of data collected over a certain time period and over a number of different locations, existing data mining approaches do not provide suitable tools to allow association patterns in such a data set to be easily discovered. The objective of this study is therefore to develop new approaches so that patterns that changes from time-period to time-period and from location to location can be discovered. Making use of techniques in meta-mining, probability and statistics, and such techniques as machine learning and fuzzy logic, our objective is to develop data mining techniques capable of discovering such patterns in spatio-temporal databases.

Over the past few years, a considerable number of studies have been made on market basket analysis. Market basket analysis is a useful method for discovering customer purchasing patterns by extracting association from stores’ transaction database. In many business of today, customer transactions can be made in many different geographical locations round the clock, especially after e-business and online shops have become prevalent. The traditional methods that consider only the association rules of an individual location or all locations as a whole are not suitable for such a multi-location environment. Understanding and adapting to changes of customer behavior from time to time and from place to place is an important aspect for a company having transactions collected from multi-locations, for example those running business-to-customer (B2C) business, to survive in continuously changing environment. If applied to B2C business, the methodology developed in this study allow companies to detect changes of customer behavior automatically from customer profiles, in which customers may come from different places over the world, and sales data may be inputted at different time snapshots.

There are three main contributions in the thesis. Firstly, we design a novel and efficient algorithm for mining spatio-temporal association rules which have multi-level time and location granularities, in spatio-temporal databases. From the perspective of business strategists, the discovered rules also must be readily interpreted for easy reading and further usage, in order to be useful. However, different executive personnel will require different interpretation of the rules in different usage scenarios. And under different granularities of time-and-place, the knowledge will be different. The goal of our work is to satisfy such dynamic needs. In this study, we develop an algorithm that can find association rules under different granularities of time-and-place to satisfy the different demands of different decision makers. Unlike Apriori-like approaches, our method scans the database at most twice. By avoiding multiple scans over the target database, our method can reduce the runtime in scanning database.

Secondly, we use membership functions to construct fuzzy calendar-map patterns which represent asynchronous time periods and locations. With the use of fuzzy calendar-map patterns, we can discover fuzzy spatio-temporal association rules which are defined as association rules occur in asynchronous time periods and/or locations.

Thirdly, we propose to mine a set of rules from the discovered collection of spatio-temporal rule sets. These *meta-rules*, rules about rules, represent the kind of knowledge that few existing data mining algorithms have been developed to mine for. In this study, we define problems in discovering the underlying regularities, differences, and changes hidden in spatio-temporal rule sets and propose a new approach, *meta-mining spatio-temporal patterns*, which mines previous spatio-temporal association rule mining results to discover these underlying regularities, differences, and changes.

Experimental results have shown that our methods are more efficient than others, and we can find fuzzy spatio-temporal association rules satisfactorily and so as meta-rules among the set of rules discovered.

Acknowledgements

This work would not have been possible without the help and support of many people; you all have my deepest gratitude. Although those worthy of my thanks are too many to list, I would still want to mention some.

Firstly, I thank my wife and parents. They support me in everything related to my life, studies and work. Their unconditional support and care, as well as the great love and encouragement are deeply appreciated.

My special thanks go to my chief supervisor, Prof. Keith Chan, for continuous support and constructive advice. He read my papers and thesis, and gave me suggestions on my work. His supervision, critical insights and supporting efforts are sincerely appreciated. The direction of spatio-temporal data mining is also inspirited by Prof. Keith Chan. The experience obtained by working with him in the past two years is definitely invaluable.

I also want to express my thanks to the continuous financial support and administration from the Departmental Research Committee, the Innovation and Technology Commission of the Government of HKSAR and the Daka Development Limited. This project cannot be implemented without the funding of the grant.

My great gratitude goes to the Department of Computing, Hong Kong Polytechnic University. The Department has provided me with essential equipment and necessary office enivironment for my research.

Last but no means least, my deeply thanks are expressed to the greatest guidance and the unconditional support from the Lord, God.

Thank you all.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Overview of Data Mining

Data mining is concerned with the nontrivial extraction of implicit, previously unknown, and potentially useful information from data [Frawley, Piatetsky-Shapiro, and Matheus 1991]. It involves the search for patterns of interest in a particular representational form or in a set of such representations (e.g., decision trees, association rules) [Fayyad *et al.* 1996a].

Data mining is also an important step in what is called *knowledge discovery in databases* (KDD) [Fayyad *et al.* 1996a] and, indeed, many researchers use the term data mining to mean KDD (e.g., [Agrawal *et al.* 1996; Han *et al.* 1996; Imielinski, Virmani, and Abdulghani 1996; Silberschatz, Stonebraker, and Ullman 1996]). In this thesis, we use data mining as a synonym for KDD.

To quote from [Matheus, Chan, and Piatetsky-Shapiro 1993], "the grand challenge of data mining is to collectively handle the problems imposed by the nature of real-world databases, which tend to be dynamic, incomplete, redundant, noisy, sparse, and very large." Many interesting studies of data mining have been carried out, drawing upon methods, algorithms, and techniques from fields as diverse as machine learning, pattern recognition, database systems, statistics, artificial intelligence, knowledge acquisition, and data visualization (see, e.g., [Fayyad *et al.* 1996b; Piatetsky-Shapiro and Frawley 1991]).

Data mining techniques can be classified according to the kind of patterns they mine for. Among the different types of patterns that can be mined, association patterns are the most important. This is because the discovery of association patterns, which are presented in IF-THEN rules, can lead more easily to the discovery of other patterns for such data mining tasks as classification, clustering or prediction. The mining of *association rules* aims at discovering interesting relationships or associations among different attribute values [Agrawal, Imielinski, and Swami 1993b; Agrawal and Shafer 1996; Agrawal and Srikant 1994; Cheung *et al.* 1996a; Han and Fu 1995; Houtsma and Swami 1995; Mannila, Toivonen, and Verkamo 1994; Park, Chen, and Yu 1995a, 1995b; Savasere, Omiecinski, and Navathe 1995; Srikant and Agrawal 1995, 1996a]. A *Boolean association rule* involves binary attributes; a *generalized association rule* involves attributes that are hierarchically related; a *quantitative association rule* involves attributes that can take on quantitative or categorical values. An example of an association rule is “90% of transactions that contain bread also contain butter; 3% of all transactions contain both of these items.” The 90% is referred to as the *confidence* and the 3%, the *support*, of the rule. The discovered association rules can be used later for human examination and machine inference, e.g., classification [Liu, Hsu, and Ma 1998].

1.2 Weaknesses of Traditional Approaches

The traditional methods for mining association rules can discover knowledge about inter-relationship among different objects. However, the methods search transactions for knowledge without individually looking at the spatial and temporal domains in the database. In our daily life, people located at different places will plan for their activities based on different seasons, weekly cycle or different festivals of the places where they are. Hence, if the nature of different time intervals at different places is taken into consideration, we should be able to discover more interested association rules. For example, if we look at a database of transactions in an international supermarket, say WalMart, we may find that turkey and pumpkin pie are seldom sold together. However, if we only look at the transactions in the week before Thanksgiving in the United States, we may discover that most transactions contain turkey and pumpkin pie, i.e., the association rule “turkey ⇒ pumpkin pie” has high support and confidence in the transactions that happen in the United States in the week before Thanksgiving. This suggests that we may discover different association rules if different time intervals and locations are considered individually. Some association rules may hold during some time intervals and locations but not the others. Discovering time intervals, locations as well as the association rules that hold during the time intervals and locations may lead to useful information.

1.3 Multiple Levels of Time and Place

Different decision makers at different corporal levels may have different demands for rules interpretation from different level of time and place. That is, different granularities of time-and-place will have different knowledge for different executives. For example, assume that the base unit time is one day and the base unit place is a single location. From the perspective of a strategist, he or she may not probably care about what kind of rules would happen in a single location on a daily basis. This kind of rules is too fragmented to interpret. For example, a CEO of a global company may want to know what kind of rules would be held in a whole year in every branch, whereas a regional manager concerns about only the rules that are hold in every season in a particular country. Different levels of decision makers would have different demands for rules interpretation and different granularity of the pair of time-and-place will have different knowledge for different strategists. In order to meet the different demands of those involved in strategic and tactical decision making, we need to find the rules in every possible combination of time and place, that is, we need to have the ability to find the rules not only in (March, L.A.) but also in (Spring, America). In the cause of finding the rules in a general context (the context here means the pair of time-and place), we use the concept hierarchy in this study to allow the rules or data to be handled at varying levels of abstractions. We study the way to establish the content of time hierarchy and place hierarchy. Hence, the method we proposed in this thesis not only can discover association rules correctly in a multi-location environment, but also can represent the rules in different granularities of time-and-place, which can satisfy different decision making with different demands by using the concept hierarchy.

1.4 Uncertainty in Spatio-temporal Requirements

Mined spatio-temporal patterns from spatio-temporal databases are subject to the location and time requirements specified by users. The spatio-temporal requirements are often vague.

Temporal requirements specified by human beings tend to be ill-defined or uncertain. For example, people in the United States usually buy turkey and pumpkin together at time close to Thanksgiving. The term "close to" is ill-defined and uncertain. To deal with this kind of uncertainty, we borrow the fuzzy set theory [Mitra, Pal, and Mitra 2002] and propose fuzzy calendar algebra to allow users to describe desired temporal requirements easily and naturally in term of fuzzy calendars. Operations that reflect the way in which people reason about temporal specifications in daily life are provided. Users can define complicated calendars with multiple time granularities and different preferences. Different time intervals can have different weights corresponding to their matching degrees to the specified fuzzy calendar. This can be of great help for users to discover knowledge in the time intervals that are of interest to them.

Likewise, there exists some vagueness in spatial requirements. There are a few reasons for the vagueness. First, the boundary of a place is sometimes not very clearcut. For example, it may be hard to classify Mei Foo as in the New Territories West or in the Sham Shui Po area. In this way, when a sales pattern is discovered in a supermarket in Mei Foo, it is difficult for us to make a strong conclusion that the sales pattern should or should not count for a sales pattern in the New Territories West area. Second, a phenomenon or a pattern appears in a location will have an impact on the locations nearby, e.g. the fashion of clothing and dressing in Taiwan will influence that in Hong Kong. Hence, at the very end of the occurrence of the phenomenon, we may hardly define the exact area that the phenomenon involves. In most of the cases, we can only say the phenomenon was discovered close to some locations. Third, some frequent patterns discovered are related to human behaviour, e.g. those found in mobile computing. However, humans have been continuously moving. Therefore, these patterns involving human behaviour do not occur exactly at a single location.

1.5 Rules about Rules

This thesis contributes to the problem definitions of mining the underlying regularities, differences, and changes hidden in rule sets and the introduction of a new approach to dealing with the problems.

Given a collection of rule sets discovered by existing data mining techniques (e.g., [Agrawal *et al.* 1992; Agrawal, Imielinski, and Swami 1993a, 1993b; Agrawal and Shafer 1996; Agrawal and Srikant 1994; Cheung *et al.* 1996a; Han and Fu 1995; Houtsma and Swami 1995; Lu, Setiono, and Liu 1995; Mannila, Toivonen, and Verkamo 1994; Mehta, Agrawal, and Rissanen 1996; Park, Chen, and Yu 1995a, 1995b; Savasere, Omiecinski, and Navathe 1995; Shafer, Agrawal, and Mehta 1996; Srikant and Agrawal 1995, 1996a]), we propose a *meta-mining* approach to discovering a set of rules in the rule sets. These rules are called *meta-rules* because they are rules about rules.

1.5.1 Mining Regularities in Rule Sets

Meta-mining is able to discover the underlying regularities hidden in rule sets. Let us take as an example an interstate or international company. It consists of a number of offices at different geographical locations and each office (or group of offices) maintains its own database [Bright, Hurson, and Pakzad 1992]. In general, local decisions are made at the branches of the international company, whereas global decisions are made at the head office and the branches contribute to these decisions in various ways. To facilitate effective decision making in such an environment, many international companies need to mine multiple data sets throughout their branches [Zhang, Wu, and Zhang 2003; Zhang, Zhang, and Wu 2004]. To do so, one can extract relevant data from multiple data sets to amass a single data set and apply existing data mining techniques (e.g., [Agrawal *et al.* 1992; Agrawal, Imielinski, and Swami 1993a, 1993b; Agrawal and Shafer 1996; Agrawal and Srikant 1994; Bradley, Fayyad, and Reina 1998; Cheeseman and Stutz 1996; Cheung *et al.* 1996a; Ganti *et al.* 1999b; Han and Fu 1995; Houtsma and Swami 1995; Lu, Setiono, and Liu 1995; Mannila, Toivonen, and Verkamo 1994; Mehta, Agrawal, and Rissanen 1996; Park, Chen, and Yu 1995a, 1995b; Savasere, Omiecinski, and Navathe 1995; Shafer, Agrawal, and Mehta 1996; Srikant and Agrawal 1995, 1996a; Zhang, Ramakrishnan, and Livny 1996]) to the single data set [Liu, Lu, and Yao 1998; Ribeiro, Kaufman, and Kerschberg 1995; Wrobel 1997; Yao and Liu 1997; Zhong, Yao, and Ohsuga 1999].

However, this approach is unable to distinguish the relationships supported by a number of tuples in many data sets from those supported by many tuples in only a few data sets. For example, a data mining algorithm may discover a rule stating that "if a customer is married and middle-aged, then he/she gets a home mortgage." This rule may be supported by many tuples in the data sets in only one or two branches. The decisions made by the head office based on this rule may therefore be good for these one or two branches; but they may not be beneficial or may even be harmful to the company as a whole.

To discover the regularities in common in the branches' data sets, we proposed to use a meta-mining approach. Given the rule sets discovered in the data sets, it mines a set of meta-rules from them. These meta-rules represent the regularities hidden in the rule sets, which in turn reflect the regularities embedded in the data sets. Based on the meta-rules discovered, the head office can better make global decisions that are beneficial to the whole company.

Realistically, the meta-mining of regularities in rule sets is not limited to use in international companies. Any public or private organization that maintains a collection of data sets or a data set with implicit groupings in terms of geographical locations, time periods, etc. can benefit from meta-mining. For example, meta-mining techniques can be applied to the rule sets discovered from the data sets collected in different outlets operated by a supermarket chain, different shops operated by an apparel retailer, or different post offices or public libraries operated by a government.

1.5.2 Mining Differences in Rule Sets

Discovered meta-rules can also represent the differences in rules sets. A meta-rule is differential if it is supported by only a few rule sets, representing a relationship that holds in those few rule sets but not in the others. It therefore distinguishes these rule sets from the others. In other words, the meta-rule represents one of the distinctive characteristics of these rule sets and in turn reflects the distinctive characteristics of the corresponding data sets.

For example, let us consider an apparel retailer operating a number of shops at different geographical locations. To maintain its brand, the retailer has each shop supply a basic range of apparel. The differential meta-rules are useful for the retailer as it allows the retailer to identify the differences in the apparel sold in its shops while each shop, in addition to providing the basic clothing range, caters to the preferences of its own customers.

1.5.3 Mining Changes in Rule Sets

The ability to detect and adapt to changes is critical to the success of many individuals and business organizations as it allows decision makers to take the changes into consideration and even take advantage of the changes when they make decisions. Knowing how circumstances will change enables a business organization to not only provide new products and services to satisfy the changing needs of its customers, but also to design corrective actions to prevent or delay undesirable changes.

Existing data mining techniques (e.g., [Agrawal *et al.* 1992; Agrawal, Imielinski, and Swami 1993a, 1993b; Agrawal and Shafer 1996; Agrawal and Srikant 1994; Bradley, Fayyad, and Reina 1998; Cheeseman and Stutz 1996; Cheung *et al.* 1996a; Ganti *et al.* 1999b; Han and Fu 1995; Houtsma and Swami 1995; Lu, Setiono, and Liu 1995; Mannila, Toivonen, and Verkamo 1994; Mehta, Agrawal, and Rissanen 1996; Park, Chen, and Yu 1995a, 1995b; Savasere, Omiecinski, and Navathe 1995; Shafer, Agrawal, and Mehta 1996; Srikant and Agrawal 1995, 1996a; Zhang, Ramakrishnan, and Livny 1996]) aim at producing accurate models of the real world in an efficient manner. They are very useful for human users to better understand the problem domains and for prediction. However, regardless of how accurately a model predicts, it can only predict based on historical data. An approach to this data that does not take into account the information about change that is hidden in its patterns is not optimal, especially when the discovered models are used for classification.

In this thesis, we also study the problem of mining changes in the context of production rules. Given a rule associated with a sequence of interestingness measures (e.g., the Dempster-Shafer measure [Dempster 1967; Shafer 1976], support and confidence [Agrawal, Imielinski, and Swami 1993b], *conviction* [Brin *et al.* 1997b], the chi-squared measure [Brin, *et al.* 1997a], the *J*-measure [Smyth and Goodman 1992], the *adjusted residual* and *weight of evidence* [Chan and Wong 1990, 1991], etc.) in different time periods, we propose to mine a set of meta-rules to represent the regularities governing how a rule changes over time. The change in the rule, in turn, reflects the change in the underlying characteristics hidden in the data. Human users can use the discovered meta-rules to examine the rule and to predict how the rule will change.

1.6 Technical Challenges

The discovery of association rules in spatio-temporal databases can be a very complicated problem. To deal with it effectively, one has to face a number of technological challenges. To quote from [Matheus, Chan, and Piatetsky-Shapiro 1993], "the grand challenge of data mining is to collectively handle the problems imposed by the nature of real-world databases, which tend to be dynamic, incomplete, redundant, noisy, sparse, and very large." Firstly, data collected in many application areas are usually very noisy. There can be many missing values in a database and there can also be many erroneous and inconsistent data values. An effective association discovery tool needs to be able to discover patterns in the midst of very noisy data. In the presence of such data, one also has to find a very good definition for what an "interesting pattern" is. Clearly, such patterns will not appear to be deterministic in a database and if the pattern is not deterministic, a definition for a probabilistic "interestingness" measure is needed. As the data that we deal with are spatio-temporal in nature, there is also a need for a meta-pattern interestingness measure. The reason why this is needed is that, there can be "patterns of patterns" in such a way that patterns can actually be discovered among the patterns that are discovered on different locations within different time periods. As data can be fuzzy, it should be noted that patterns or meta-patterns may also be fuzzy. In addition to a probabilistic interestingness measure, it should be noted that we may also require a fuzzy interestingness measure for the discovery of fuzzy patterns.

Furthermore, dealing with the interaction of space and time is complicated by the fact that they have different semantics. We cannot just treat time as another spatial dimension, or vice versa. For example, time has a natural ordering while space does not. Allied with this, we also need to deal with these spatio-temporal semantics effectively. This includes considering the effects of area and the time interval width not only on the the patterns we mine, but also in the algorithms that find those patterns. Besides, large volume, diverse format, multi-phases, high dimension and multi-scale are well-known complexities of spatio-temporal data. In such data, attribute values of temporally and/or spatially neighboring objects are typically correlated. This severely enlarges the search space in finding frequent patterns. Even worst, operations on spatial data are very expensive as spatial objects are computationally costly. Constructed from lines, polygons, 3D surfaces and to name but a few, operations on spatial objects are very expensive. To manipulate them together with the temporal dimension will further blows up the pattern search space.

Another major difficulty is that many data mining (DM) algorithms do not scale well to huge volumes of data. Spatio-temporal databases are usually relatively large in volume because data are collected from multiple locations continuously. A scalable DM algorithm is characterized by linear increase of its runtime with the linear increase of the number of examples in the data, and within a fixed amount of memory. Most of the DM algorithms are not scalable, but there are several examples of tools that do scale well. They include clustering algorithms [Zhang *et al.* 1996; Bradley *et al.* 1998; Ganti *et al.* 1999a], ML algorithms [Shafer *et al.* 1996; Gehrke *et al.* 1998], and association rule algorithms [Agrawal and Srikant 1994; Agrawal *et al.* 1995; Toivonen 1996]. An overview of scalable DM tools is given in [Ganti *et al.* 1999b]. The most recent approach for dealing with the scalability problem is the Meta Mining (MM) concept. MM generates meta-knowledge from the meta-data generated by DM algorithms [Spiliopoulou and Roddick 2000]. However, to discover meta-rules is not a simple process too because of the two reasons. First, some rules cannot be easily compared due to different rule structures. Second, even with matched rules, it is difficult to know what kind of change and how much change has occurred.

1.7 Potential Application Areas

A data mining engine capable of discovering spatio-temporal association patterns can have many applications in many different areas. Consider a supermarket chain that has different stores in different region. Suppose that transaction data are captured at each store every day, the ability to discover association rules in spatio-temporal data will allow us to understand if there are any differences in the patterns discovered from one location to another and from one time period to another. Such a discovery will allow us to better control inventory, predict sales, etc. A data mining engine capable of discovering how patterns vary across time and space can also be applied to evaluate the effect of public policy measures in education, crime control, health care, and work-force management, etc. It can also be used to evaluate changes in performance, standards of quality, and for customer profiling, etc. We suggest the more specific examples of potential applications are:

First, in computer networking, by considering each IP packet in a computer network as a transaction and the attributes in the IP header as items in the transaction, we can use spatio-temporal association rules to represent normal network activities at different spatial points of network in different time periods of a day; attacks to the network may be flagged when the network behaves differently from its normal behaviors.

Second, effective website personalization is at the heart of many e-commerce applications. To ensure that customers visiting these websites receive useful product recommendations and additional personalized service, website personalization is a critical business strategy. Since the purchase habits of people are often governed by the time and location when and where they are, database techniques, including spatio-temporal data mining and active databases, can be effectively combined to achieve an efficient and scalable personalization framework.

Third, recent advances in communication and information technology, such as the increasing accuracy of GPS technology and the miniaturization of wireless communication devices pave the road for Location-Based Services (LBS). To achieve high quality for such services, data mining techniques are suggested for the analysis of the huge amount of data collected from location-aware mobile devices. Since the two most important attributes of the data collected is time and location, spatio-temporal data mining techniques can be developed to extract interesting knowledge for LBS.

1.8 Organization of the Thesis

This thesis is outlined as follows.

In Chapter 2, we will give a brief review of the previous works in spatio-temporal data mining, especially the mining of spatio-temporal association rules. The idea of meta-mining, and the application of fuzzy logic in data mining will also be introduced.

In Chapter 3, we will formally describe the main problem that will be addressed and the types of spatio-temporal association rules, including fuzzy and precise rules that will be focused in this work. We will define three types of meta-rules. They are added rules, perished rules and change meta-rules. We will also define all the other terminology used in this thesis, including calendar-map patterns, fuzzy match ratio, support and confidence for our defined spatio-temporal association rules and to name but a few.

In Chapter 4, we will explain our proposed solution to the problems defined in Chapter 3. This includes the Spatio-temporal Apriori algorithm (an extension of Apriori, the most well-known association rule mining algorithm), our new algorithm that is believed to be better than Spatio-temporal Apriori in term of runtime. We will depict how the new algorithm can be fuzzified to mine fuzzy spatio-temporal association rules. At the end of the chapter, we will also propose some ways to mine the three types of meta-rules defined in Chapter 3.

In Chapter 5, experiments with the proposed algorithms on the real and synthetic databases will be described. We will put emphasis on comparing our proposed spatio-temporal association rule mining algorithm with Spatio-temporal Apriori. We will review the performance of our proposed meta-mining method on mining meta-rules. We will also show that the proposed fuzzified approach can discover some patterns that non-fuzzy approach cannot discover.

Finally, some possible future work will be discussed in Chapter 6. And we will conclude and discuss all our work in Chapter 7.

Chapter 2
Related Work

The data mining field dates back almost 20 years. However, the field of spatial data mining—where the spatial aspect of the data defines a relationship between every data point (close-to, within, north-of,…), and the field of temporal data mining—where the temporal aspect of the data defines a relationship between every data point (before, during,…) are relatively young and highly demanding. The field of spatio-temporal data mining—where this relationship is both defined by the spatial and temporal aspects of the data, is extremely challenging due to the increased search space for knowledge. Not surprisingly, there have only been a few attempts to extend data mining for spatio-temporal data. Although several of these techniques from the temporal and the spatial data mining field can be adopted for the task of spatio-temporal data mining, with the exception of a few [Tsoukatos and Gunopulos 2001; Hadjieleftheriou *et al.* 2003], there has been little work on the combination of the two. However, two directions can be identified: firstly, the incorporation of time into spatial data mining systems, and secondly, the incorporation of space into temporal data mining systems [Roddick and Spiliopoulou 1999]. It is believed that such a unification of spatial and temporal data mining could be highly beneficial for applications, in which both time and space are important.

2.1 Knowledge Discovery and Data Mining

Due to the automated collection of massive amount of transaction data, data mining (DM) or knowledge discovery (KDD) in databases, defined as the discovery of interesting, implicit, and previously unknown knowledge from data [Koperski, Adhikary, and Han 1996], received significant scientific and commercial interest in recent years.

KDD is a process comprising of many steps, which involves data selection, data pre-processing, data transformation, data mining (search for patterns), and interpretation and evaluation of patterns. The basic steps of the KDD process are presented in Figure 1 (these steps were defined, for example, in [Fayyad *et al.* 1997].

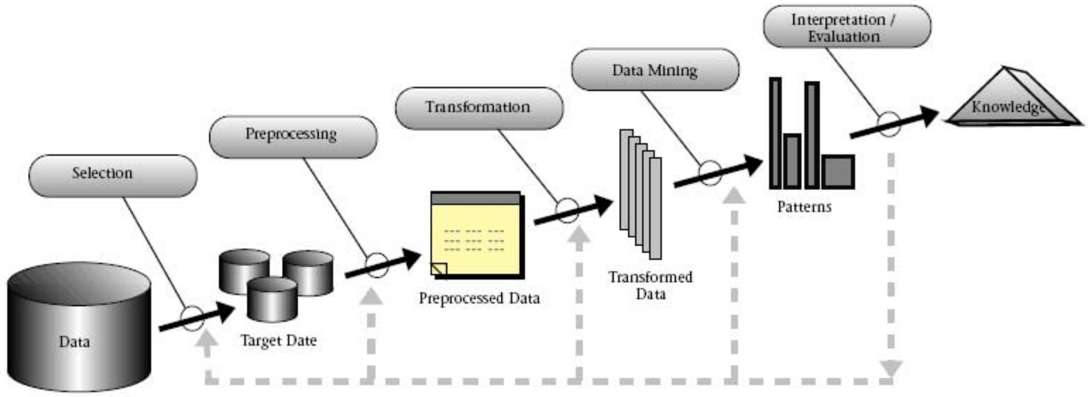


Fig. 1. Basic steps of the KDD process.

The steps depicted start with the raw data and finish with the extracted knowledge, which was acquired as a result of the KDD process. The set of data mining tasks used to extract and verify patterns in data is the core of the process. Data mining (DM) consists of applying data analysis and discovery algorithms for producing a particular enumeration of patterns (or models) over the data [Brunk *et al.* 1997]. Most of current KDD research is dedicated to the DM step. However, this core area typically takes only a small part (estimated at 15%-25%) of the effort of the overall KDD process. The additional steps of the KDD process, such as data preparation, data selection, data cleaning, incorporating appropriate prior knowledge, and proper interpretation of the results of mining, are also essential to derive useful knowledge from data [Gaul and Säuberlich 1999].

Knowledge discovered can be represented in various forms, but one common and intuitively easy to understand form is in terms of rules. A rule is an implication of the form $A \Rightarrow B$, where A and B are sets of attributes. It carries the meaning that if attributes in A take on certain values then with some probability attributes in B take on certain (other) values. Rule mining methods can be categorized into three groups based on the type and relation among the attributes: association rules, rules describing patterns in sequences, cluster characteristic or discriminant rules.

2.1 Rule Mining in Market Basket Analysis

The idea of mining association rules originates from the analysis of market basket data, which can be informally described as the discovery of intra-transaction patterns in large customer transaction databases was first introduced in [Agrawal, Imielinski, and Swami 1993b]. Let $I = \{a_1,\ldots,a_n\}$ be a set of distinct literals, called *items*, and let A, a subset of I, be called a *k-itemset* if $|A| = k$. Let D be a database of transactions T, where each transactions T

is a subset of *I*. A transaction *T* supports an itemset *A* if *A* is a subset of *T*. An association rule is an expression $A \Rightarrow B$, where *A*, *B* are itemsets and $A \cap B = \{\}$ holds. The *support* of an itemset *A* is the fraction of transactions *T* that contain the itemset. An itemset *A* is frequent if its support is above a certain minimum support threshold. The support of a rule $A \Rightarrow B$ is the support of the itemset $A \cup B$. The *confidence* of this rule is the fraction of the support of the rule and the support of *A*, that is the conditional probability P(*B*|*A*). The problem of mining association rules can be defined as finding all associations rules that have a support and confidence greater than a specified minimum support and confidence value. Such rules are often referred to as *strong* rules.

Virtually all association rule mining methods decompose the problem into two phases: first, finding of all frequent itemsets, second, generating all frequent and confident rules from these frequent itemsets. It is easy to see that the search space of all itemsets is $2^{|I|}$ and the search space of all association rules is $3^{|I|}$, although knowing all the itemsets apriori the later search space also reduces to $2^{|I|}$. The traversal of the exponential search space is made possible by the following two properties. Downward closure property of itemset support: All subsets of a frequent itemset must also be frequent. Downward closure property of rule confidence: If the rule $A \Rightarrow B$ is confident then the for any *X* subset of *B* the rule $A \cup X \Rightarrow B \setminus X$ must also be confident.

Although, after the first efficient algorithm [Agrawal and Srikant 1994] there has been little improvement on generating strong association rules from frequent itemset (2nd phase), there has been significant work on finding frequent itemsets (1st phase). One can divide approaches for finding frequent itemsets based on two criteria: a) by their strategy to traverse the search space and b) by their strategy to determine the support values of itemsets. Based on the first criteria today's common approaches are either breadth-first search (BFS) or depth-first search (DFS). BFS approaches generate and test itemsets in levels, starting at level 1 with the trivial 1-itemsets. Candidate itemsets at the *k*th level are generated by intersecting all possible combinations of frequent (*k*-1)-itemsets that actually form a *k*-itemset. State of the art BFS approaches are the **Apriori** algorithm of [Agrawal and Srikant 1994] and the **Partition** algorithm of [Savasere, Omiecinski, and Navathe 1995], DFS approaches, f.ex., recursively descend following the lattice defined by the itemsets. State of the art DFS approaches are the **FP-growth** algorithm of [Han, Pei, Yin 2000] and **Eclat** algorithm of [Zaki *et al.* 1997]. Based on the second criteria approaches can also be divided into two classes: a) approaches that determine the support of an itemset by directly counting its occurrences in the data based, or b) approaches that determine support of an itemset by set intersection. In the first type of support counting approach subset generation and candidate lookup is typically aided by a hash tree or a similar data structure. **Apriori** and **FP-growth** adopt this approach. In the second type of support counting approach support for an itemset is represented in form of a transaction identifier list, *tidlist*. The identifier of a transaction is on the tidlist of an itemset if that itemset is contained in the given transaction. This type of representation of itemsets support is also referred to as vertical database layout in the literature. Having sorted tidlists in memory allows the efficient joining of two itemsets by simply intersecting their tidlists. The actual support of an itemset is the cardinality of its tidlist. **Partition** and **Eclat** determine itemset support based on set intersections. A comparison of the four methods [Hipp, Güntzer, and Nakhaeizadeh 2000] have revealed that the methods all have some types of data for which they perform better than the others, which is why we will consider all four in our project.

The idea of mining sequential patterns also originates from the analysis of market basket data, which can be informally described as the discovery of inter-transaction patterns in large customer transaction databases was first introduced in [Agrawal and Srikant 1995]. In this setting transactions are associated with a unique customer identifier and a transaction time. A *sequence* is a set of temporarily ordered itemsets. A *customer sequence* is an ordered list of transactions that are associated with the same customer. A customer *supports* a sequence *s* if *s* is contained in the customer sequence for this customer. A sequence is *frequent* if its support is greater than a specified minimum support value. A sequence is *maximal* if it is not a subsequence of any other frequent sequence. The problem of mining sequential patterns can be defined as finding all frequent sequences. Since the set of all frequent sequences is a superset of the set of all frequent itemsets, sequential pattern mining algorithms often utilize some of the ideas proposed for the discovery of association rules, i.e.: are an extension of association rule mining. As examples of such extension, the algorithms presented in [Agrawal and Srikant 1995] use ideas presented in [Agrawal and Srikant 1994], and [Zaki 2001], to find maximal sequential patterns, extends the DFS approach for fining association rules [Zaki *et al.* 1997] by decomposing the original problem into independent, smaller sub-problems that can be solved in main memory using efficient lattice search techniques.

2.2 Spatial, Temporal & Spatio-temporal Data Mining

The rule mining tasks described so far, with the exception of transaction time in the case of sequential patterns, were in a sense dimensionless, that is multiple possible items were contained in a transaction; however, most of the data collected in databases describe events or objects in the physical world, which have two special attributes associated with them:

location and time. Despite the abundance of such data, from here on referred to as spatio-temporal data, the number of algorithms that mine such data is few. The main reason for the lack of efficient algorithms is due to the exponential explosion in the search space for knowledge caused by the added spatial and temporal attributes. Depending on which of these attributes the data mining methods take into account, they can be divided into three groups: spatial, temporal, and spatio-temporal [Roddick and Spiliopoulou 1999; Roddick, Hornsby, and Spiliopoulou 2000; Roddick and Lees 2001]. The following subsections attempt to categorize recent, state of the art methods into these three groups.

2.2.1 Association Rules

Since the pioneering work of [Agrawal and Srikant 1994], association rule mining methods were extended to the spatial [Koperski and Han 1995; Han, Koperski, and Stefanovic 1997; Ester, Kriegel, and Sander 1997, Ester *et al.* 1998], and later to the temporal domain [Li, Wang and Jajodia 2000]. With the exception of [Hadjieleftheriou *et al.* 2003], where approximate methods are applied for counting entries in the database, there has been little work done in the spatio-temporal domain. In [Hadjieleftheriou *et al.* 2003] three techniques are proposed for answering density based queries in the spatio-temporal domain for moving objects. Trajectories of objects are modeled as linear functions of time in a three-dimensional space-time grid. Two types of queries are considered: snapshot density and period density queries. The difficulty of the problems lies in the fact that the spatial and/or temporal predicates are not specified by the query, i.e., the solutions should contain all dense regions at a specified future time (snapshot density query) or at any time in the future (period density query). The three techniques that solve the problems are based on: coarse grids—a way of compressing the temporal dimension by merging consecutive space-time cells; lossy counting—an approximate method for counting the number of space-time cell crossings; and finally, dense cell filters—a way of efficiently summarizing dense space-time cells.

2.2.2 Sequential Patterns

The first sequential pattern mining algorithms were first introduced in [Agrawal and Srikant 1995] and further improved in [Srikant and Agrawal 1996b; Zaki 2001; Mannila, Toivonen, and Verkamo 1997]. Spatial extension to these methods, to find spatio-temporal sequential patterns for the task of weather prediction, were added in [Stolorz *et al.* 1995] and later improved in [Tsoukatos and Gunopulos 2001]. In [Tsoukatos and Gunopulos 2001] an efficient DFS algorithm is given to discover spatio-temporal sequential patterns. The algorithm does not enumerate all frequent sequences, but rather aims at discovering only the maximal frequent sequences. This pruning of the search space compensates for the I/O cost incurred from the repeated database scans results in an overall efficient algorithm that can be easily extended to discover sequential patterns at multiple spatial granularities. Even though the method presented efficiently discovers spatio-temporal sequential patterns for weather prediction it is currently unclear whether it is applicable for LBS. While in the former case one seeks relationships between time-varying attributes for fixed locations (temperature, pressure, etc…), in the later case relationships between rather stable attributes (age, sex, income, interest, etc…) for objects with time-varying location is sought for.

2.2.3 Cluster Characteristics or Discriminant Rules

Cluster characteristic or discriminant rules associate objects belonging to a cluster to some attributes with some probability. While clustering has been extensively studied in the past decades, spatial clustering and clustering in the presence of obstacles has only recently received much attention [Indulska and Orlowska 2002; Guo, Peuquet, and Gahegan 2002; Ng and Han 2002; Tung, Hou, and Han 2001; Estivill-Castro and Lee 2000; Zhang, Hsu, and Dayal 2000; Povinelli 2000; Tung *et al.* 2000]. A general approach in spatial data mining is to apply generalization techniques to spatial and non-spatial data to generalize detailed spatial data to higher levels and study the general characteristics and data distributions at these levels [Koperski, and Han 1995]. Generalization has been incorporated into spatial clustering in [Lu, Han, and Ooi 1993; Ng and Han 1994] resulting in two variants: spatial dominant generalization and non-spatial dominant generalization. Spatial dominant generalization focuses on discovering non-spatial characterizations of spatial clusters, while nonspatial dominant generalization focuses on spatial clusters existing in groups of non-spatial objects. It is not clear whether the spatial dominant and/or the non-spatial dominant method can be extended to include the time dimension in a straight forward manner, but the lack of publications seems to indicate that this has never been tried before. Generalization/concept hierarchies have also been used to identify discriminating concepts between groups of spatial objects [Knorr and Ng 1996a-b].

2.3 Supervised Learning for Data Mining

All of the above discussed methods are unsupervised learning methods, which try to extract “interesting” knowledge about the observable attributes. While there are several measures for interestingness [Tan, Kumar, and Srivastava 2002], the measure of “usefulness” can only be defined in terms of an objective function w.r.t some outcome variable or attribute. The task of extracting knowledge from large samples of observable attributes in the presence of an objective function is referred to as supervised learning in the literature [Hastie, Tibshirani, and Friedman 2001]. While there are numerous, sophisticated supervised learning methods that approximate any desired objective function, several simple methods remain popular. The popularity of these methods is due to several reasons. Firstly, they are simple in knowledge representation and hence are easily understood even by non-experts. Secondly, they learn to approximate the objective function in a computationally efficient way. Finally, they are found to be very robust and generalize well for previously unseen examples.

Regression trees are one of the oldest and most used of these simple methods. They try to partition feature space defined by the observable attributes into a set of rectangles, and then fit a simple model to each one. Popular tree building methods are CART, ID3, C4.5 and C5.0 [Hastie, Tibshirani, and Friedman 2001].

Naïve Bayes, another simple but popular method for supervised learning for classification, constructs a probabilistic model to predict the class attribute based on the observable attributes. Since estimating the joint probabilities of all possible combinations of observable attribute outcomes given the class attribute requires an enormous amount of samples, the model makes the naïve assumption that the observable attributes are conditionally independent given the class attribute. Hence, the learning of the model is reduced to estimating the frequency of each observable attribute outcome given the class attribute. The model can be used for classification by turning around the class conditional probability estimates of observable attributes using the Bayes theorem. Bayesian networks try to discover probabilistic causal relationship between observable and class attributes. While the causal relationships are attractive to analysts, their estimation requires large number of observations and is computationally less efficient.

2.4 Fuzzy Logic in Data Mining

If humans describe objects, they effectively use linguistic terms like, for instance, small, old, long, fast. However, classical set theory is hardly suited to define sets of objects that satisfy such linguistic terms. Let us, for examples, assume a person being assigned to the set of tall persons. If a second person is only insignificantly smaller, it should also be assigned to this set, and thus it seems reasonable to formulate a rule like “a person who is less than 1mm smaller than a tall person is also tall” to define our set. However, if we repeatedly apply this rule, obviously persons of any size will be assigned to the set of tall persons. Any threshold for the concept tall will be hardly justifiable. On the other hand, it is easy to find persons that are tall or small, respectively. Modeling the typical cases is not the problem, but the penumbra between the concepts can hardly be appropriately modeled with classical sets.

2.4.1 Conception of Fuzzy Logic

The main principle of fuzzy set theory is to generalize the concept of set membership [Zadeh 1965]. In classical set theory a characteristic function

$$\eta_A : \Omega \to \{0,1\}$$

$$\eta_A(\omega) = \begin{cases} 1, \text{if } \omega \in A \\ 0, \text{else,} \end{cases}$$

defines the memberships of objects $\omega \in \Omega$ to a set $A \subset \Omega$. In fuzzy set theory the characteristic function is replaced by a membership function

$$\mu_M : \Omega \to [0,1]$$

that assigns numbers to objects $\omega \in \Omega$ according to their membership degree to a fuzzy set $M \subset X$. A membership degree of one means that an object fully belongs to the fuzzy set, zero means that it does not belong to the set. Membership degrees between zero and one correspond to partial memberships.

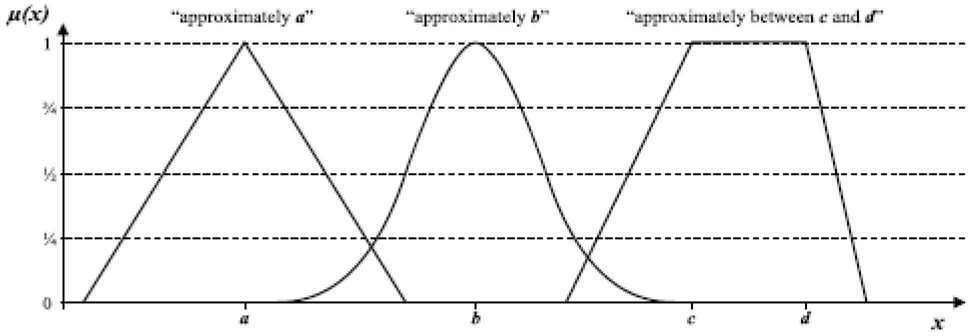


Fig. 2. Examples of typical fuzzy sets.

Membership degrees can be used to represent different kinds of imperfect knowledge, including similarity, preference, and uncertainty. In fuzzy classification rules, fuzzy sets are used to model similarity between attribute values and prototypes, often described by linguistic terms. On the real scale, very common fuzzy sets are so-called fuzzy numbers (or fuzzy intervals) that assume a value of one for a single value $a \in \mathcal{R}$ (or interval $[a, b] \subset \mathcal{R}$), and have monotonously decreasing membership degrees with increasing distance from this core. Fuzzy numbers can be associated with linguistic terms like, for example, “approximately *a*”. In fuzzy rule based systems, typically parameterized membership functions are used, where these are in most cases either triangular, trapezoidal, or Gaussian shaped (cf. Figure 2):

$$\mu_{x_{0,\sigma}}(x) = \exp(-\frac{(x-x_0)^2}{2\sigma^2}).$$

If the complete input range is covered by overlapping fuzzy sets, this is called fuzzy partition. If their number is sufficiently small, the fuzzy sets M are usually associated with linguistic terms, e.g. $A_M \in$ {small, medium, large}. In the following, fuzzy sets M, their corresponding fuzzy membership functions μ_M and the associated linguistic terms A_M will be used interchangeably, where the correspondence is clear.

2.4.2 Application of Fuzzy Logic

In data mining, regardless of how the values of continuous attributes, e.g. height, size, distance, temporal proximity and the like, are discretized, the intervals may not be concise and meaningful enough for human users to easily obtain non-trivial knowledge from the discovered relationships. To better handle continuous data, the use of fuzzy sets for data mining has recently been proposed in the literature [Mitra, Pal, and Mitra 2002]. The resilience to noises and the affinity with the human knowledge representation make fuzzy sets to be used in many data mining systems (e.g., [Au and Chan 1998, 1999, 2001, 2003; Chan and Au 1997b, 2001; Chan, Au, and Choi 2002; Delgado *et al.* 2003; Hirota and Pedrycz 1999; Hüllermeier 2001; Ishibuchi, Yamamoto, and Nakashima 2001; Janikow 1998; Kacprzyk and Zadrozny 2001; Lee and Kim 1997; Maimon, Kandel, and Last 1999; Yager 1991]).

Linguistic summaries introduced in [Yager 1991] express knowledge using a linguistic representation that is natural for human users to comprehend. An example of a linguistic summary is the statement “about half of the people in the database are middle-aged.” However, no algorithm was proposed for generating linguistic summaries in [Yager 1991]. Recently, the use of an algorithm for mining association rules for the purpose of linguistic summaries has been studied in [Kacprzyk and Zadrozny 2001]. This technique extends AprioriTid [Agrawal and Srikant 1994], a well-known algorithm for mining association rules, to handle linguistic terms (fuzzy values). An attribute is replaced by a set of artificial attributes (items) so that a tuple supports a specific item to a certain degree, which is in the range from 0 to 1. Given two user-specified thresholds, *threshold*$_1$ and *threshold*$_2$, an item or an itemset (i.e., a combination of items) is considered interesting if its *fuzzy support* is greater than *threshold*$_1$ and it is also less than *threshold*$_2$. Although this technique is very useful, many users may not be able to set the thresholds appropriately.

In addition to linguistic summaries, an interactive process for the discovery of top-down summaries, which utilizes *fuzzy is-a hierarchies* as domain knowledge, has been described in [Lee and Kim 1997]. This technique aims at discovering a set of *generalized tuples*, such as <technical writer, documentation>. In contrast to association rules, which involve the implications between different attributes, linguistic summaries and generalized tuples only provide the summarization on different attributes. The idea of implication has not been taken into consideration and hence these techniques are not developed for the task of rule discovery.

Furthermore, the applicability of fuzzy modeling techniques to data mining has been discussed in [Hirota and Pedrycz 1999]. Given a relational table, *X*, and a context variable, *A*, the *context-sensitive fuzzy clustering* method reveals the structure in *X* in the context of *A*. Since this method can only manipulate continuous attributes, the values of any discrete attributes are first encoded into numeric values. The context-sensitive fuzzy clustering method is then applied to the encoded data to induce clusters in the context of *A*. Although the encoding technique allows this method to deal with discrete attributes, the distances between the encoded numeric values, which do not possess any meaning in the original discrete attributes, are used to induce the clusters. Therefore, the associations that are concerned with these attributes, which are discovered by the context-sensitive fuzzy clustering method, may be misleading.

2.5 Meta Mining

Meta-mining is concerned with mining previously discovered patterns, which are typically represented in the form of production (if-then) rules [Au and Chan 2002a, 2002b, 2005; Roddick and Spiliopoulou 2002; Spiliopoulou and Roddick 2000; Kurgan and Cios 2004]. It can be used to discover many useful patterns that existing data mining techniques (e.g., [Agrawal *et al.* 1992; Agrawal, Imielinski, and Swami 1993a, 1993b; Agrawal and Shafer 1996; Agrawal and Srikant 1994; Bradley, Fayyad, and Reina 1998; Cheeseman and Stutz 1996; Cheung *et al.* 1996a; Ganti *et al.* 1999b; Han and Fu 1995; Houtsma and Swami 1995; Lu, Setiono, and Liu 1995; Mannila, Toivonen, and Verkamo 1994; Mehta, Agrawal, and Rissanen 1996; Park, Chen, and Yu 1995a, 1995b; Savasere, Omiecinski, and Navathe 1995; Shafer, Agrawal, and Mehta 1996; Srikant and Agrawal 1995, 1996a; Zhang, Ramakrishnan, and Livny 1996]) are not developed to mine for. These patterns are represented in the form of production rules and they are called *meta-rules* because they are rules about rules. The discovered meta-rules are arguably closer to the forms of knowledge that might be considered interesting [Roddick and Spiliopoulou 2002]. For example, the meta-rule “High Income is becoming more associated with Mercedes Benz Ownership” is arguably more interesting than the rule “High Income is associated with Mercedes Benz Ownership.”

Although meta-mining is an important problem, it has received little attention in the literature. To our best knowledge, in addition to our previous work [Au and Chan 2002a, 2002b, 2005], this problem has only been studied in [Spiliopoulou and Roddick 2000; Kurgan and Cios 2004].

A framework for analyzing data mining results, called *higher order mining*, has been proposed in [Spiliopoulou and Roddick 2000]. In this framework, a *first order rule* is a rule discovered in a data set, whereas a *second order rule* is a sequence of first order rules discovered in different data sets. Given a second order rule, the interestingness measures (e.g., the Dempster-Shafer measure [Dempster 1967; Shafer 1976], support and confidence [Agrawal, Imielinski, and Swami 1993b], *conviction* [Brin *et al.* 1997b], the chi-squared measure [Brin *et al.* 1997a], the *J*-measure [Smyth and Goodman 1992], the *adjusted residual* and *weight of evidence* [Chan and Wong 1990, 1991], etc.) of its first order rules can be considered as a time series. One can then apply time series analysis (e.g., ARIMA [Box, Jenkins, and Reinsel 1994]) to analyze the time series. Some of the first order rules of a second order rule may not hold in the corresponding data sets because their interestingness measures may fall below the user-specified thresholds, for example. The time series may therefore contain missing values. However, time series analysis is not developed to deal with missing values. Furthermore, the discovered patterns are embedded in the parameters of the statistical model constructed and hence they are unnatural for human users to comprehend.

This framework has also been used in a meta-mining system proposed in [Kurgan and Cios 2004] to generate data models from already generated data models. The system 1) divides a data set into a number of subsets; 2) generates a set of rule from each data subset using a supervised learning algorithm; and 3) mines a set of (meta-) rules from the rule sets using the same algorithm. The discovered meta-rules can then be used for classification. The experimental results reported in [Kurgan and Cios 2004] show that the performance of the meta-rules discovered from the already discovered rule sets is a little inferior to that of the rules discovered from the data sets in terms of classification rate.

[Spiliopoulou and Roddick 2000] is concerned with revealing changes in rule sets, whereas [Kurgan and Cios 2004] aims at discovering regularities in rule sets. None of them is developed to uncover all of the regularities, differences, and changes.

A related, but not directly applicable, work is *meta-learning* [Prodromidis, Chan, and Stolfo 2000]. Given a collection of data sets or data subsets, it runs a supervised learning algorithm or different learning algorithms on each of them. It then combines the predictions of the learned classifiers to produce a *meta-classifier* by recursively learning *arbiter* and *combiner* models in a bottom-up tree manner [Prodromidis, Chan, and Stolfo 2000]. An arbiter plays the role as a judge whose own prediction is used if the participating classifiers cannot reach a consensus decision. A combiner can further be classified as *class-combiner*, *class-attribute-combiner*, and *binary-class-combiner*. In a class-combiner, the meta-level training instances consist of the correct classification and the predictions; in a class-attribute-combiner, the instances are formed as in a class-combiner with the addition of the attribute vectors; and a binary-class-combiner, the instances are composed in a manner similar to that in a class-combiner except that each prediction has *l* binary predictions where *l* is the number of classes [Prodromidis, Chan, and Stolfo 2000]. An example of the patterns revealed by meta-learning is "given a record, if classifier 1 classifies it into class A and classifier 2 classifies it into class B, then it is classified into class A." Meta-learning indeed is not developed to reveal the underlying patterns hidden in the classifiers.

2.5.1 Mining Regularities in Multiple Data Sets

For an interstate or international company, which comprises a number of offices at different geographical locations and has each office (or group of offices) to maintain its own database, to better make decisions, it needs to mine multiple databases throughout their offices [Zhang, Wu, and Zhang 2003]. However, existing data mining techniques (e.g., [Agrawal *et al.* 1992; Agrawal, Imielinski, and Swami 1993a, 1993b; Agrawal and Shafer 1996; Agrawal and Srikant 1994, 1995; Bradley, Fayyad, and Reina 1998; Cheeseman and Stutz 1996; Cheung *et al.* 1996a; Ganti *et al.* 1999b; Han, Dong, and Yin 1999; Han and Fu 1995; Houtsma and Swami 1995; Lu, Setiono, and Liu 1995; Mannila, Toivonen, and Verkamo 1994, 1995; Mehta, Agrawal, and Rissanen 1996; Park, Chen, and Yu 1995a, 1995b; Savasere, Omiecinski, and Navathe 1995; Shafer, Agrawal, and Mehta 1996; Srikant and Agrawal 1995, 1996a; Zhang, Ramakrishnan, and Livny 1996]) are developed to handle a single database and they are not directly applicable to mining multiple databases.

Recently, several techniques for data mining in multiple databases, including [Liu, Lu, and Yao 1998; Ribeiro, Kaufman, and Kerschberg 1995; Wrobel 1997; Yao and Liu 1997;

Zhong, Yao, and Ohsuga 1999], have been proposed in the literature. These multi-database mining techniques typically involve 1) selecting relevant data from multiple databases; 2) extracting the selected data to amass a single database; and 3) applying existing data mining techniques, such as association rule mining (e.g., [Agrawal, Imielinski, and Swami 1993b; Agrawal and Shafer 1996; Agrawal and Srikant 1994; Cheung *et al.* 1996a; Han and Fu 1995; Houtsma and Swami 1995; Mannila, Toivonen, and Verkamo 1994; Park, Chen, and Yu 1995a, 1995b; Savasere, Omiecinski, and Navathe 1995; Srikant and Agrawal 1995, 1996a]), classification (e.g., [Agrawal *et al.* 1992; Agrawal, Imielinski, and Swami 1993a; Lu, Setiono, and Liu 1995; Mehta, Agrawal, and Rissanen 1996; Shafer, Agrawal, and Mehta 1996]), and clustering (e.g., [Bradley, Fayyad, and Reina 1998; Cheeseman and Stutz 1996; Ganti *et al.* 1999b; Zhang, Ramakrishnan, and Livny 1996]), to the single database.

They can therefore discover only the same kind of patterns as conventional (single-) database mining techniques. They are unable to discover some patterns such as “in general, if a customer is married and middle-aged, then he/she gets a home mortgage.” They also cannot discover such patterns as “in an exceptional manner, if a customer is single and tertiary educated, then he/she has more than one car.” The former represents a regular pattern supported by many branches of an international company, whereas the latter represents a differential pattern supported by only a few branches.

Recently, the mining of *high-vote patterns* in multiple databases has been proposed in [Zhang, Zhang, and Wu 2004]. Given the m databases, D_1, …, D_m in the m branches of a company, a conventional (single-) database mining algorithm is first applied to D_i to discover a set of patterns, R_i, $i = 1, \ldots, m$. Let $R = \{r_j \mid r_j \in R_1 \cup \ldots \cup R_m\}$ and $n = |R|$. The *average voting rate*, *AVR*, is given by:

$$AVR = \frac{1}{n}\sum_{j=1}^{n} voting(r_j)\ , \tag{2.1}$$

where *voting*(r_j) is the *voting rate* of r_j and is calculated by:

$$voting(r_j) = \frac{|\{R_i \mid r_j \in R_i, i = 1, ..., m\}|}{m}\ . \tag{2.2}$$

The interestingness of r_j, *interest*(r_j), is then defined in [Zhang, Zhang, Wu 2004] as:

$$interest(r_j) = \frac{voting(r_j) - AVR}{1 - AVR}. \quad (2.3)$$

A pattern is high-voting if its voting rate is greater than the average voting rate and its interestingness is greater than or equal to a user-specified threshold [Zhang, Zhang, and Wu 2004]. A weakness of this approach is that many users do not have any idea what the threshold should be. Some useful patterns may be missed if it is set too high, whereas many irrelevant patterns may be found if it is set too low.

Instead of concatenating multiple data sources to amass a single data set, a set of association rules can be synthesized from the association rules discovered in the data sources [Wu and Zhang 2003]. The supports of these association rules are estimated in terms of the supports of the underlying association rules and the popularities of the data sources. The experimental results in [Wu and Zhang 2003] show that the synthesized rules are a good approximate of the rules discovered in the concatenated data set. Although this synthesizing technique starts from multiple data sources, it is not developed to discover the regularities in the rule sets.

2.5.2 Mining Differences in Multiple Data Sets

In [Ganti *et al.* 1999a], a framework has been proposed to measure the difference between two data sets by building two models (one from each data set) and measuring the amount of work required to transform one model to the other. It results in a real number to reflect to which degree the two data sets differ from each other. However, it is not developed to explicitly reveal what the differences are.

Recently, the mining of *exceptional patterns* in multiple databases in the context of association rules has been proposed in [Zhang, Zhang, and Wu 2004]. Given the m databases, $D_1, \ldots, D_m$ in the m branches of a company, an association rule mining algorithm is first applied to D_i to discover a set of patterns (i.e., association rules), R_i, $i = 1, \ldots, m$. Let $R = \{r_j \mid r_j \in R_1 \cup \ldots \cup R_m\}$. The interestingness of r_j, *exceptional interest*(r_j), is defined in [Zhang, Zhang, and Wu 2004] as:

$$exceptional\ interest(r_j) = \frac{voting(r_j) - AVR}{-AVR}, \quad (2.4)$$

where $voting(r_j)$ is the voting rate of r_j given by Equation (2.3) and AVR is the average voting rate calculated by Equation (2.2). In addition to this measure, another interestingness measure of r_j with respect to D_i is also defined in [Zhang, Zhang, and Wu 2004] as:

$$exceptional\ interest_i(r_j) = \frac{support_i(r_j) - minsupport_i}{minsupport_i}, \tag{2.5}$$

where $support_i(r_j)$ is the support of r_j in D_i and $minsupport_i$ is the user-specified minimum support for mining patterns in D_i. A pattern r_j is exceptional if 1) its voting rate is greater than the average voting rate and *exceptional interest*(r_j) is greater than or equal to a user-specified threshold; and 2) $exceptional\ interest_i(r_j)$ is greater than or equal to another user-specified threshold for all $i \in \{i \mid r_j \in R_i\}$. Similar to the mining of high-vote patterns, a weakness of this approach is that many users have no idea what the thresholds should be. If they are set too high, some useful patterns may be missed; but if they are set too low, many irrelevant patterns may be found.

2.5.3 Mining Changes in Multiple Data Sets

To deal with the data collected in different time periods, the maintenance of discovered association rules (e.g., FUP [Cheung *et al.* 1996b]) and *active data mining* [Agrawal and Psaila 1995] have been proposed in the literature. Incremental updating techniques (e.g., FUP) can be used to update the discovered association rules if there are additions, deletions, or modifications of any tuples in a database after a set of association rules has been discovered. Active data mining is concerned with representing and querying the shape of the history of parameters for the discovered association rules. Although these techniques can be used to track the variations in supports and confidences of association rules, both of them are not developed to discover and predict rule changes.

Although the mining of rule changes over time is an important problem, it has received little attention. To our best knowledge, in addition to our previous work [Au and Chan 2002a, 2002b, 2005], this problem has only been studied in [Liu *et al.* 2000], [Liu, Hsu, and Ma 2001], and [Spiliopoulou and Roddick 2000]. [Liu *et al.* 2000] is concerned with finding whether a decision tree built in a time period is applicable in other time periods. Given two data sets collected in two different time periods, this method builds a decision tree based on one of the data sets and then builds another based on the other data set such that the latter tree uses the same attribute and chooses the same cut point for the attribute as the former at each step of partitioning. This method can be used to identify three categories of changes in the context of decision tree building: *partition change*, *error rate change*, and *coverage change* [Liu *et al.* 2000]. Compared to [Liu *et al.* 2000], instead of building a decision tree in the next time instance to ensure that it resembles the first, our goal is to discover the changes in rules discovered in different time periods.

Following the idea presented in [Liu *et al.* 2000], a method has been proposed in [Liu, Hsu, and Ma 2001] to find whether a set of association rules discovered in a time period is applicable in other time periods. To do so, it employs chi-square test to determine whether there are any changes in the supports and confidences of the association rules discovered in different time periods. Unlike this method, our goal is to mine (meta-) rules to represent the changes and to predict any changes in the future.

If the underlying data sets are collected in different time periods, the higher order mining framework proposed in [Spiliopoulou and Roddick 2000] can be used to find the changes in the discovered rules. Given a second order rule, the interestingness measures (e.g., the Dempster-Shafer measure [Dempster 1967; Shafer 1976], support and confidence [Agrawal, Imielinski, and Swami 1993b], conviction [Brin *et al.* 1997b], the chi-squared measure [Brin et al. 1997a], the *J*-measure [Smyth and Goodman 1992], the adjusted residual and weight of evidence [Chan and Wong 1990, 1991], etc.) of its first order rules can be considered as a time series, which can be analyzed by time series analysis (e.g., ARIMA [Box, Jenkins, and Reinsel 1994]). The time series may contain missing values because some of the first order rules of a second order rule may not hold in the corresponding data sets as their interestingness measures may fall below the user-specified thresholds, for example. However, time series analysis is not developed to deal with missing values. Furthermore, the discovered patterns are embedded in the parameters of the statistical model constructed. They are therefore not natural for human users to comprehend.

Chapter 3

A Formal Problem Description

We consider a spatio-temporal database D containing transactions records from multiple locations over time periods. The records are ordered first by location identifiers then by timestamps. Our objectives are to extract association rules from database in multi-location environment precisely and to enhance the readability of the rules. For the sake of convenience in presentation, the cardinal of the set, say Σ, is denoted by $| \Sigma |$. Let $I = \{i_1, i_2, ..., i_n\}$ be the set of product items included in D, where i_k is the identifier for the k^{th} item. Let X be a set of items in I. We refer to X as a k-itemset if $| X | = k$. Furthermore, each transaction $Tran_j$ in D is a subset of I and is attached with a timestamp t_v and location identifier p_w to indicate the time and location when and where the transaction occurs.

3.1 Mining Association Rules

Let us suppose that there is a collection of data sets, D_j, $j = 1, ..., n$, partitioned from D by t_v and p_w, i.e. $Y D_j = D$. A set of rules, $R_j = \{r_{j1}, ..., r_{js_j}\}$, is mined from D_j, $j = 1, ..., n$. A rule, $r_{ju} \in R_j$, is an implication of the form $X \Rightarrow Y$, where X and Y are conjunctions of conditions. The antecedent and the consequent of the rule $X \Rightarrow Y$ are denoted as *antecedent*$(X \Rightarrow Y) = X$ and *consequent*$(X \Rightarrow Y) = Y$, respectively.

Example 3.1 An example rule, r, is:

$$Sex = Male \wedge Education = Tertiary \wedge Income = High$$
$$\Rightarrow Mercedes\ Benz\ Ownership = True.$$

The antecedent and the consequent of this rule are:

$$antecedent(r) = (Sex = Male \wedge Education = Tertiary \wedge Income = High)$$

and

$$consequent(r) = (Mercedes\ Benz\ Ownership = True),$$

respectively.

Given a rule, $X \Rightarrow Y$, let *condition*(X) and *condition*(Y) be the sets of all the conditions in its antecedent and consequent, respectively. The set of conditions in the rule $X \Rightarrow Y$ is then given by *condition*($X \Rightarrow Y$) = *condition*(X) ∪ *condition*(Y). Let us further suppose that

$$condition(R_j) = \bigcup_{u=1}^{s_j} condition(r_{ju}).$$

Example 3.2 Let us consider the rule r given in Example 3.1. The set of conditions in its antecedent is:

$$condition(antecedent(r)) = \{Sex = Male, Education = Tertiary, Income = High\},$$

the set of conditions in its consequent is:

$$condition(consequent(r)) = \{Mercedes\ Benz\ Ownership = True\},$$

and the set of conditions in the rule is:

$$condition(r) = \{Sex = Male, Education = Tertiary, Income = High,$$
$$Mercedes\ Benz\ Ownership = True\}.$$

In general, the rule $X \Rightarrow Y$ is associated with one or more interestingness measures (e.g., the Dempster-Shafer measure [Dempster 1967; Shafer 1976], *support* and *confidence* [Agrawal, Imielinski, and Swami 1993b], *conviction* [Brin *et al.* 1997b], the chi-squared measure [Brin *et al.* 1997a], the J-measure [Smyth and Goodman 1992], the *adjusted residual* and *weight of evidence* [Chan and Wong 1990, 1991], etc.). We denote the interestingness measure of the rule $X \Rightarrow Y$ in D_j as *interestingness*$_j$($X \Rightarrow Y$).

Example 3.3 In an association rule mining algorithm, the interestingness of a rule such as that in Example 3.1 is measured in terms of support and confidence. It holds in data set D_j with support,

$$support_j(r) = \frac{|\sigma_{Sex=Male \wedge Education=Tertiary \wedge Income=High \wedge Mercedes\ Benz\ Ownership=True}(D_j)|}{|D_j|},$$

and confidence,

$$confidence_j(r) = \frac{|\sigma_{Sex=Male \wedge Education=Tertiary \wedge Income=High \wedge Mercedes\ Benz\ Ownership=True}(D_j)|}{|\sigma_{Sex=Male \wedge Education=Tertiary \wedge Income=High}(D_j)|},$$

where σ denotes the SELECT operation in *relational algebra* and $|S|$ denotes the cardinality of set S.

3.2 Calendar-based Patterns

Since every transaction in the the database D has a timestamp t_v, we denote timestamps with calendar schema in this thesis in explaining our algorithms. A *calendar schema* is a relational schema $S = (T_a : M_a, T_{a-1} : M_{a-l}, \ldots, T_1 : M_1)$ with a *valid* constraint. Each attribute ***T,*** is a granularity name like year, month and week. Each domain ***M,*** is a finite subset of positive integers. The constraint *valid* is a Boolean function on $M_a \times M_{a-l} \times \ldots \times M_1$, specifying which combinations of the values in $M_a \times M_{a-l} \times \ldots \times M_1$ are "valid". The purpose is to exclude the combinations that we are not interested in or that do not correspond to any time intervals. For example, if we do not want to consider the weekend days and holidays, we can let *valid* evaluate to False for all such days. For brevity, we omit the domains *M,* and/or the constraint *valid* from the calendar schema when no confusion arises.

Given a calendar schema $S = (T_a : M_a, T_{a-1} : M_{a-l}, \ldots, T_1 : M_1)$, a ***simple calendar-based pattern*** (or *calendar pattern* for short) on S is a tuple of the form $<t_a, t_{a-1}, \ldots, t_1>$, where each ***t,*** is in ***M,*** or the wild-card symbol *. The calendar pattern $<t_a, t_{a-1}, \ldots, t_1>$ represents the set-of time intervals intuitively described by "the t_1^{th} T_1 of the t_2^{th} T_2, …, of t_a^{th} T_a." If ***t,*** is the wildcard symbol '*', then the phrase "the t_i^{th}" is replaced by the phrase "every". For example, given the calendar schema (week, day, hour), the calendar pattern (*, 1, 10) means "the 10th hour on the first day (i.e., Monday) of every week". Each calendar pattern intuitively represents the time intervals given by a set of valid tuples in $M_a \times M_{a-l} \times \ldots \times M_1$.

We say a calendar pattern *f covers* another calendar pattern *f'* in the same calendar schema if the set of time intervals of *f'* is a subset of the set of intervals of ***f.*** For example, given the calendar schema (*week*, *day*, *hour*), (l, *, 10) covers (1, 1, 10). It is easy to see that for a given calendar schema $(T_a, T_{a-1}, \ldots, T_1)$, a calendar pattern $<t_a, t_{a-1}, \ldots, t_1>$ covers another calendar pattern $<t_a', t_{a-1}', \ldots, t_1'>$ if and only if for each i, $1 \le i \le a$, either t_i = '*' or

$t_i = t_i^{'}$.

For the sake of presentation, we call a calendar pattern with *k* wild-card symbols a *k-star calendar pattern* (denoted f_k) and a calendar pattern with at least one wild-card symbol a *star calendar pattern.* In addition, we call a calendar pattern with no wild-card symbol (i.e., a 0-star calendar pattern) a *basic time interval* if the combination is “valid”.

3.3 Map-based Patterns

Likewise, we denote location identifier p_w in each transaction with map schema in this thesis. A *map schema* is a relational schema $R = (P_c : D_c, P_{c-1} : D_{c-l}, \ldots, P_1 : D_1)$ with a *valid* constraint. Each attribute *P*, is a granularity name like counry, province/state and city. Each domain *D*, is a finite subset of the strings. The constraint *valid* is a Boolean function on $D_c \times D_{c-l} \times \ldots \times D_1$, specifying which combinations of the values in $D_c \times D_{c-l} \times \ldots \times D_1$ are “valid”. The purpose is to exclude the combinations that we are not interested in or that do not correspond to any locations. For example, if we do not want to consider the cities in Alaska, we can let *valid* evaluate to False for all such places. For brevity, we omit the domains *D*, and/or the constraint *valid* from the map schema when no confusion arises.

Given a map schema $R = (P_c : D_c, P_{c-1} : D_{c-l}, \ldots, P_1 : D_1)$, a *simple map-based pattern* (or *map pattern* for short) on *R* is a tuple of the form $<d_c, d_{c-1}, \ldots, d_1>$, where each *d*, is in *D*, or the wild-card symbol *. The map pattern $<d_c, d_{c-1}, \ldots, d_1>$ represents the set-of locations intuitively described by “the P_1 d_1, in the P_2 d_2, …, in the P_c d_c.” If *d*, is the wildcard symbol ‘*’, then the phrase “the P_i d_i” is replaced by the phrase “every P_i”. For example, given the map schema (counry, province/state, city), the calendar pattern (United States, California, *) means “every city in the state California in the country United States”. Each map pattern intuitively represents the locations given by a set of valid tuples in $D_c \times D_{c-l} \times \ldots \times D_1$.

Unlike calendar patterns, a wildcard symbol * cannot be put in the middle of two *d* in a map pattern. For example, the map pattern <United States, *, Los Angeles> will be evaluated to be false by the Boolean function of the ”valid“ constraint. It is because the meaning of <United States, *, Los Angeles> is the same as that of <United States, California, Los Angeles>, since Los Angeles belongs to one and only one state, i.e. California, that the map pattern <United States, *, Los Angeles> is regarded to be "repeated". Formally speaking, for a valid map pattern, if $D_j = *$, then $D_k = *$, $\forall\ 1 \leq k < j$.

We say a map pattern *e covers* another map pattern *e’* in the same map schema if the set of locations of *e’* is a subset of the set of locations of ***e*.** For example, given the map schema (*country, province/state, city*)**,** (United States, California, *) covers (Unit States, California, Los Angeles). It is easy to see that for a given calendar schema (P_c, P_{c-1}, …, P_1), a valid map pattern <d_c, d_{c-1}, …, d_1> covers another valid map pattern <$d_c{'}$, $d_{c-1}{'}$, …, $d_1{'}$> if and only if for each i, $1 \le i \le c$, either d_i = '*' or d_i=$d_i{'}$.

For the sake of presentation, we call a map pattern with *k* wild-card symbols a *k-star map pattern* (denoted e_k) and a map pattern with at least one wild-card symbol a *star map pattern.* In addition, we call a map pattern with no wild-card symbol (i.e., a 0-star map pattern) a *basic space interval* if the combination is "valid".

3.4 Calendar-map Patterns

Spacetime is a model that combines 3-D or 2-D space and 1-D time into a single construct called the space-time continuum, as shown in Figure 3. According to Euclidean space perception, our universe has three dimensions of space, and one dimension of time. Space and time are the arenas in which all physical events take place — for example, the action of a user's logging in a shopping site may be described in a particular type of space-time, or the transaction of a customer's buying a bottle of coke in a supermarkt may be described in another type of space-time. In any given spacetime, an event is a unique position at a unique time. By combining the two concepts into a single manifold, analysts are able to deal in a unified way with spacetimes which attempt to explain the underlying patterns or repeating cycles of the occurrence of events.

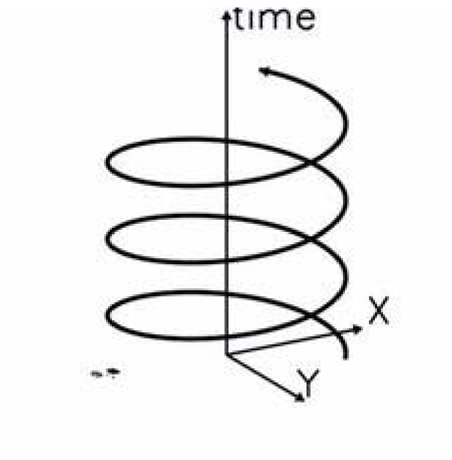


Fig. 3. Space-time continuum.

Since we want to represent the association rules extracted from multi-location environment in general, we define the combination of the period of time and the position of locations in a hierarchical way. Notice that, the time periods and their structures, i.e. calendar patterns, are defined according to specific needs for the problems, such as a year has four seasons; a season has three months, and so on. Likewise, the locations and their structures, i.e. map patterns, are also defined clearly, such as a country has a lot of provinces; a province has a lot of cities, and so on.

Definition 3.1. T, P are tree structures that T has a root cover all the scope of positions. Let $T=\{T_1 \cup T_2 \cup \ldots \cup T_i \cup \ldots \cup T_a\}$ be the set of all time periods and $P=\{P_1 \cup P_2 \cup \ldots \cup P_j \cup \ldots \cup P_c\}$ be the set of scope of positions, where T_i is the set of time periods at level i and P_j is the set of positions at level j. We use the parameter a to refer to the number of levels in time conceptual tree T and the parameter c to refer to the number of levels in position conceptual tree P. Notice that, P_1 means the leaf level in the tree and P_c means the root level in the tree.

Definition 3.2. Let $T_i=\{T_{(i,1)}, T_{(i,2)}, \ldots, T_{(i,x)}, \ldots, T_{(i,ni)}\}$ and $P_j=\{P_{(j,1)}, P_{(j,2)}, \ldots, P_{(j,y)}, \ldots, P_{(j,nj)}\}$ where $T_{(i,x)}$ means the x^{th} time period at level i and $P_{(j,y)}$ means the y^{th} position at level j. The parameter ni in $T_{(i,ni)}$ can be defined as the number of time periods at level i and the parameter nj in $P_{(j,nj)}$ can be defined as the number of positions at level j. Note that P_1 means the set of all unit stores in P and T_1 means the set of all unit times in T.

Further, the time periods in the same level are mutually disjoint and form a complete partition of T_i. The same can be said about the place structure. We can make it clear what we intend by the following expressions:

(1) If i is between 1 to a then

(a) $T_{(i,j)} \cap T_{(i,k)} = \varnothing$ (if $j \neq k$);

(b) $\bigcup_j T_{(i,j)} = T_i$

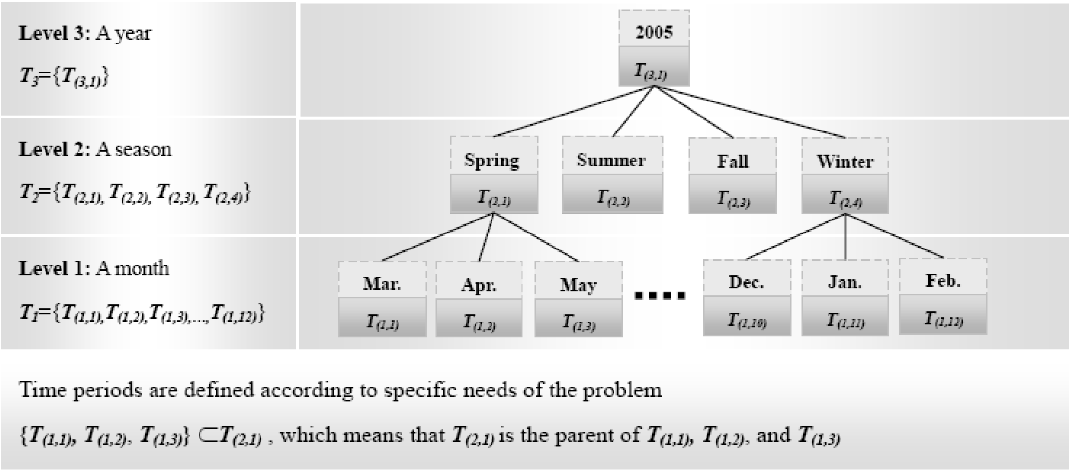
(2) If i is between 1 to c then

(a) $P_{(i,j)} \cap P_{(i,k)} = \varnothing$ (if $j \neq k$);

(b) $\bigcup_j P_{(i,j)} = P_i$

Definition 3.3. Consider two time periods $T_{(i1,x1)}$ and $T_{(i2,x2)}$, $T_{(i2,x2)}$ is the parent of $T_{(i1,x1)}$ if the $T_{(i1,\,x1)} \subset T_{(i2,\,x2)}$ and $i2 = i1 + 1$. The expression $T_{(i1,\,x1)} \subset T_{(i2,\,x2)}$ can be defined as all time periods of $T_{(i1,\,x1)}$ are covered by those of $T_{(i2,\,x2)}$. In the same way, consider two scope of positions $P_{(j1,\,y1)}$ and $P_{(j2,\,y2)}$, $P_{(j2,\,y2)}$ is the parent of $P_{(j1,\,y1)}$ if the $P_{(j1,\,y1)} \subset P_{(j2,\,y2)}$ and $j2 = j1 +1$. The expression $P_{(j1,\,y1)} \subset P_{(j2,\,y2)}$ can be defined as all scope of positions of $P_{(j1,\,y1)}$ are covered by those of $P_{(j2,\,y2)}$.

We use Figure 4 to illustrate an example of time hierarchy tree and the concept will be the same of the place hierarchy tree.



Level 3: A year
$T_3=\{T_{(3,1)}\}$
2005
$T_{(3,1)}$
Level 2: A season
$T_2=\{T_{(2,1)}, T_{(2,2)}, T_{(2,3)}, T_{(2,4)}\}$
Spring
$T_{(2,1)}$
Summer
$T_{(2,2)}$
Fall
$T_{(2,3)}$
Winter
$T_{(2,4)}$
Level 1: A month
$T_1=\{T_{(1,1)},T_{(1,2)},T_{(1,3)},\ldots,T_{(1,12)}\}$
Mar.
$T_{(1,1)}$
Apr.
$T_{(1,2)}$
May
$T_{(1,3)}$
Dec.
$T_{(1,10)}$
Jan.
$T_{(1,11)}$
Feb.
$T_{(1,12)}$
Time periods are defined according to specific needs of the problem
$\{T_{(1,1)}, T_{(1,2)}, T_{(1,3)}\} \subset T_{(2,1)}$, which means that $T_{(2,1)}$ is the parent of $T_{(1,1)}$, $T_{(1,2)}$, and $T_{(1,3)}$

Fig. 4. An example of time hierarchy tree.

Since time and place have their own specific hierarchy trees, in order to show the rules that are represented in different point of view of time and place, we need to use a combine relationship between *T* and *P* to facilate us to achieve this goal. Hence, we use *TP* lattice to state the combined relationsip between *T* and *P* clearly.

Definition 3.4. Given the time hierarchy *T* and place hierarchy *P*, we can form an aggregation lattice, *TP* lattice, from *T* and *P* by the following rules:

(1) Node (T_i, P_j) exists in *TP* for $b \leqq i \leqq a$ and $b \leqq j \leqq c$

(2) Arc $(T_i, P_j) \rightarrow (T_{i+1}, P_j)$ exists in *TP* for $b \leqq i < a$ and $b \leqq j \leqq c$

(3) Arc $(T_i, P_j) \rightarrow (T_i, P_{j+1})$ exists in *TP* for $i = b$ and $b \leqq j < c$

From the perspective of a strategist, sometimes he would probably not concern about what kind of rules would happen in a single location day by day. And therefore, it would be meaningless for us to discover the rules that would be held during a unit time in a unit place. In this case, we can set *b* to be 2, so that we will treat (T_2, P_2) as a starting point of the combine relation in *TP* lattice.

Example 3.4. Given the four level of time hierarchy *T* and place hierarchies *P*, we can form a *TP* lattice with nine different conditions that will be shown in Figure 5.

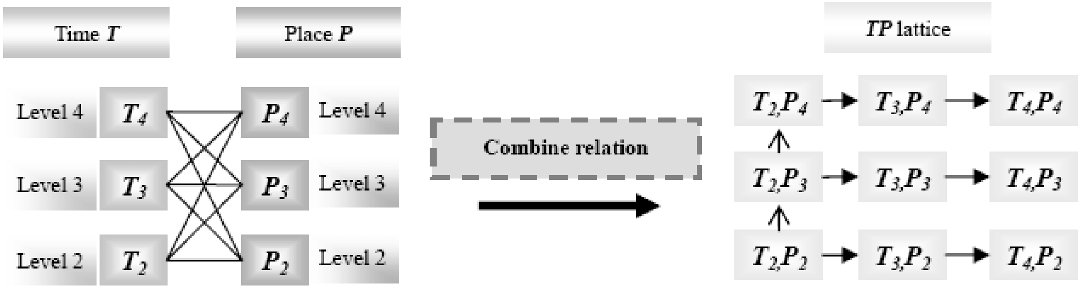


Fig. 5. An example of *TP* lattice.

Through the concept of *TP* lattice, using the method of combining, we can find rule under more general context, i.e. By the concept of *TP* lattice, our algorithm can find the rules not only in (April, San Francisco), but also can find the rule in (Spring, San Francisco) or (April, California), and so on.

Definition 3.5. The single node (T_i, P_j) contains the set of all possible combinations of calendar patterns and map patterns, i.e. $\{\{(T_a, T_{a-1}, ..., T_k, ..., T_1)\} \times \{(P_c, P_{c-1}, ..., P_l, ..., P_1)\}\}$ where '×' is Cartesian product of two sets, satisfying the following rules:

(1) $T_k = *, \ \forall\ k < i$

(2) $T_k \neq *, k = i$

(3) $T_k = *$ or $\{T_{(k,1)}, T_{(k,2)}, ..., T_{(k, nk)}\}, \ \forall\ k > i$

(4) (T_a, T_{a-1}, ..., T_k, ..., T_1) is a valid calendar pattern

(5) $P_l = *, \ \forall\ l < i$

(6) $P_l \neq *, l = i$

(7) $P_l = *$ or $\{P_{(l,1)}, P_{(l,2)}, ..., P_{(l, nl)}\}, \ \forall\ l > i$

(8) (P_c, P_{c-1}, ..., P_l, ..., P_1) is a valid map pattern

Example 3.5. According to Definition (3.5) and given the following conditions and constraints:

1) The calendar schema S = (year, month, date).
2) A valid calendar pattern s in S must have year = {2005, 2006} and month = {1, 2}.
3) The map schema R = (country, state/province, city).
4) A valid map pattern r in R must have state/province = {California, Florida}.

The node (T_2, P_2) in the *TP* lattice should contain the following set:

{{(United States, California, *), (United States, Florida, *)} ×
{(*, 1, *), (2005, 1, *), (2006, 1, *), (*, 2, *), (2005, 2, *), (2006, 2, *)}}
= {(United States, Califoria, *, *, 1, *) → In every January in California
(United States, Califoria, *, 2005, 1, *) → In January of 2005 in California
(United States, Califoria, *, 2006, 1, *) → In January of 2006 in California
(United States, Califoria, *, *, 2, *) → In every February in California
(United States, Califoria, *, 2005, 2, *) → In February of 2005 in California
(United States, Califoria, *, 2006, 2, *) → In February of 2006 in California
(United States, Florida, *, *, 1, *) → In every January in Florida
(United States, Florida, *, 2005, 1, *) → In January of 2005 in Florida
(United States, Florida, *, 2006, 1, *) → In January of 2006 in Florida
(United States, Florida, *, *, 2, *) → In every February in Florida
(United States, Florida, *, 2005, 2, *) → In February of 2005 in Florida
(United States, Florida, *, 2006, 2, *)} → In February of 2006 in Florida

For the sake of presentation, we call a calendar-map pattern with k wild-card symbols a *k-star calendar-map pattern* (denoted ef_k) and a calendar-map pattern with at least one wild-card symbol a *star calendar-map pattern.* In addition, we call a map pattern with no wild-card symbol (i.e., a 0-star map pattern) a *basic space-time interval* if the combination is "valid".

3.5 Mining Spatio-temporal Association Rules

The problem addressed in this thesis is to find, from spatio-temporal databases, map-calendar-based spatio-temporal association rules which holds in the calendar schema and map schema specified by users. Let $\tau = \{i_1, i_2, \ldots, i_l\}$ be a set of items. Let D be a spatio-temporal database of transactions where each transaction t is associated with a pair of identifiers (*TID, SID*) , and a set of items t_c such that $t_c \subseteq \tau$. *TID* is time information indicating the time when the transaction occurred whereas *SID* is location information indicating the place where the transaction occurred. Let D be divided into a sequence of $n \times o$ partitions, $P_{1,1}$, $P_{1,2}$, …, $P_{2,1}$, $P_{2,2}$, …, and $P_{n,o}$, each $P_{i,j}$ containing a set of transactions occurring in the corresponding time interval T_i and location S_j with the temporal and spatial units being those of the smallest granularities, i.e. basic time interval and basic space interval. n is the total number of basic time intervals while o is the total number of basic space intervals. Mining map-calendar-based spatio-temporal association rules in the database D is to discover interesting patterns with calendar-based periodicity and map-based repeatability in D. That is, to discover every association rule, which holds in an enough number of time intervals and locations given by the corresponding calendar pattern and map pattern. An association rule with respect to a time interval, T_i, and a location S_j, is an implication of the form

$$X \overset{T_i, S_j}{\Rightarrow} Y$$

where $X \subseteq \tau$, $Y \subseteq \tau$, and $X \cap Y = \emptyset$. Let $|P_{i,j}(I)|$ be the number of transactions containing itemset I in partition $P_{i,j}$. The association rule $X \overset{T_i, S_j}{\Rightarrow} Y$ is said to have support s% in the partition $P_{i,j}$ if

$$\left|P_{i,j}(X \cup Y)\right| = \left|P_{i,j}\right| \times s\%, \tag{3.1}$$

where $|P_{i,j}|$ denotes the number of transactions in partition $P_{i,j}$. For an association rule $X \overset{T_i, S_j}{\Rightarrow} Y$, let

$$\frac{\left|P_{i,j}(X \cup Y)\right|}{\left|P_{i,j}(X)\right|} = c\%, \tag{3.2}$$

The rule is said to hold in $P_{i,j}$ or (T_i, S_j) with confidence c%. For a given pair of confidence and support thresholds, c% and s%, and a given pair of time interval and location (T_i, S_j), association rules in (T_i, S_j) are those which have confidence and support greater than or equal to c% and s% in $P_{i,j}$, respectively.

Furthermore, a map-calendar-based spatio-temporal association rule with respect to a calendar pattern, v, and a map pattern, u, is an implication of the form

$$X \overset{u,v}{\Rightarrow} Y$$

Assume that $|v|$ time intervals are covered by v and $|u|$ locations are covered by u. If an association rule holds in at least $m \times |u| \times |v|$ combinations of locations covered by u and time intervals covered by v, where m is a user-defined match ratio ($0 < m \leq 1$), it is said to be a map-calendar-based spatio-temporal association rule that holds in u and v.

3.6 Fuzzy Calendar Patterns

Temporal expressions are widely used in our daily life. However, temporal requirements specified by human beings tend to be ill-defined or uncertain. It is hard, or even impossible, for users to provide a crisp description about their desired calendars. To formulate human reasoning into the process of knowledge discovery, fuzzy set theory is adopted for the construction of calendars in this section. Fuzzy concepts and operations are introduced to help users express their desired calendars easily and conveniently.

To construct a calendar, the hierarchy of time granularity, e.g., week, month, and year, has to be determined to handle descriptions of multiple time granularities [Giannella, Han, Pei, Yan, and Yu 2003]. For each time granularity, fuzzy sets which describe the distribution of all the time intervals in the time granularity can be specified. Each fuzzy description of a time granularity, e.g., *in the middle of a year*, *at the very beginning of a month*, or *at the end of a week*, etc., forms a *basic* fuzzy calendar.

Definition 3.6. A basic fuzzy calendar pattern, A, characterizes a fuzzy proposition about the collection of time intervals in a time granularity, described by a membership function where

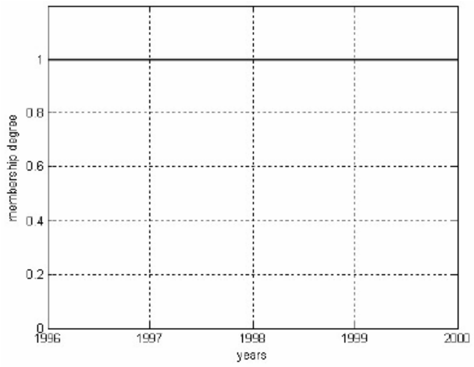
$$\mu_A = U \to [0,1]$$

for every time interval . The function value indicates the matching degree of input time to the fuzzy calendar pattern *A*.

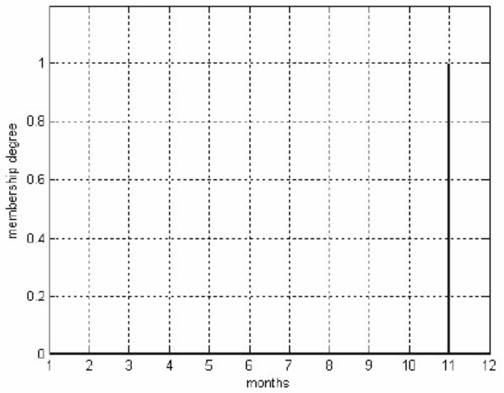
In a calendar schema S=(T_a:M_a, T_{a-1}:M_{a-1}, …, T_1:M_1), we let the fuzzy calendar pattern *FC*, defined by users under the calendar schema, has the membership function F=(F_a, F_{a-1}, …, F_1). F_i is a membership function for the corresponding time unit T_i in the calendar schema *S*. In that way, the degree of membership of a basic time interval (t_a, t_{a-1}, …, t_1), $t_i \in M_i$, where i=1,…, *a*, in the fuzzy calendar pattern *FC* is calculated as:

$$degree_{rc}(t_a, t_{a-1}, ..., t_1) = F_a(t_a) \times F_{a-1}(t_{a-1}) \times ... \times F_1(t_1)\,, \tag{3.3}$$

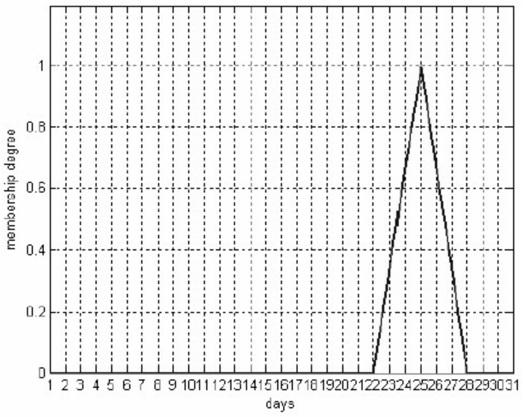
In Figure 6, we use the fuzzy calendar pattern “close to (*, 11, 25)”, i.e. “close to November 25”, as an example. We define the set of membership functions F=(F_{year}, F_{month}, F_{day}) to describe the fuzzy calendar pattern. This is under the assumption that only the dates between two days before and after November 25 of each year can be said to be “close to November 25 of each year”.



Membership function F_{year}



Membership function F_{month}



Membership function F_{day}

Fig. 6. Use a membership function to represent (*, 11, 25).

Some examples of basic fuzzy calendar patterns are shown in Figure 7. Usually, the shapes and the number of fuzzy sets describing a time granularity are arbitrarily specified by the user. Hedges such as *very* and *more or less* can also be used. With such fuzzy descriptions about time, users do not have to know the exact boundaries between interesting and non-interesting time intervals. Furthermore, the time intervals which are more important can have a larger membership degree and will contribute bigger influence, which is intuitively desirable. Fuzzy calendars can also be used to describe crisp time intervals. For example, Wednesday and May can be described by the fuzzy calendar patterns with singletons, as shown in Figure 8.

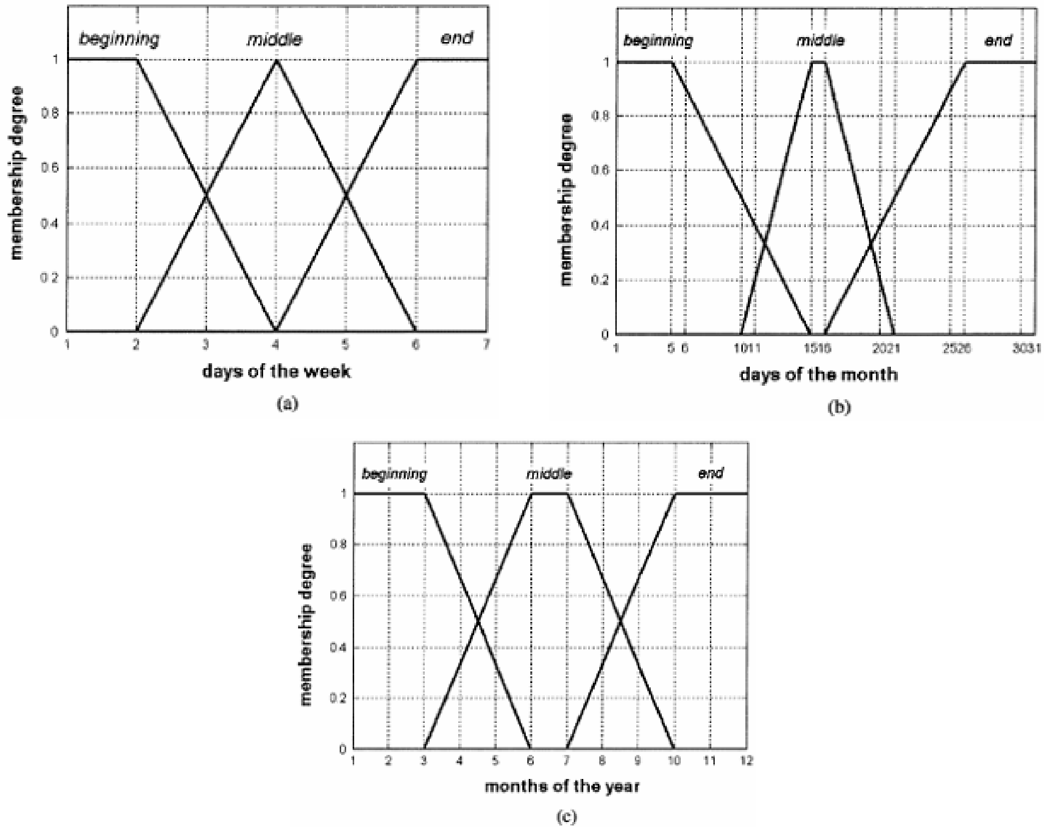


Fig. 7. Basic fuzzy calendar patterns associated with the time granularity of (a) week (b) month (c) year.

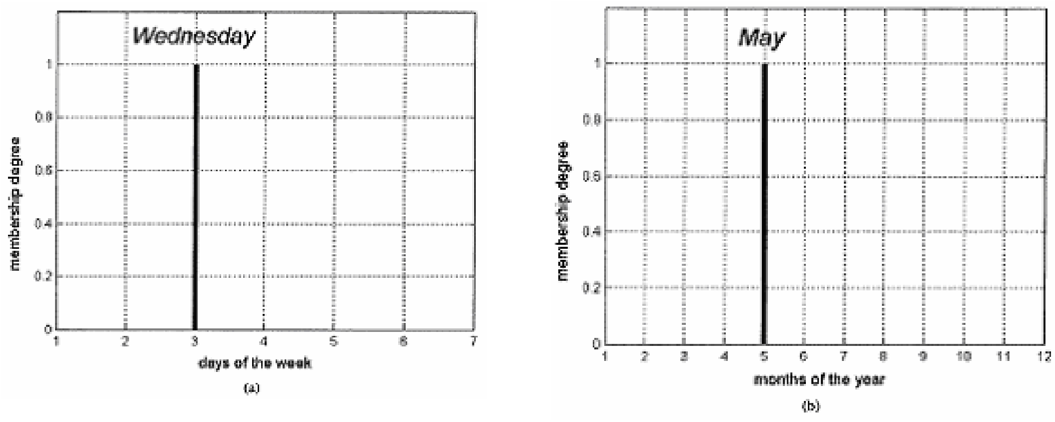


Fig. 8. An Singletons describing crisp time intervals (a) Wednesday, and (b) May, respectively.

3.7 Fuzzy Map Patterns

Most of the linguistic description such as *close* and *prefer* are *fuzzy* in nature. The conventional feature map approach does not capture linguistic and heuristic knowledge in an effective manner. Linguistic terms such as *Close* and *Far* are often used to describe distances. The distance is an important factor governing huumans' decision making and behaviour. For example, the distance from highway is an important factor for many buyers to determine whether a site is suitable as a residential, commercial, or industrial site. Also, people will usually purchase things in the supermarkets the closest to their home. Since the descriptions of many linguistic terms are relative, we need to define the range that the membership functions of the linguistic variable are to cover. The range is termed the *universe of discourse* of the membership function. The membership values of each function are usually normalized between 0 and 1, where 0 indicates non-member and 1 indicates full member of the membership function, respectively. Figure 9 shows the membership functions of three linguistic variables: *Very Close, Moderately Close,* and *Far,* used to describe the *distance from supermarket.* For example, 0 and 9 miles cannot be considered as *moderately close* because they both have 0 membership values in the *moderately close* membership function. 4 miles is *really moderately close* because its has full membership value for being *moderately close.* 2 miles has 0.66 membership value of being *moderately close,* and it is considered as *more or less* (more toward *more) moderately close.* In this thesis, we assume locations are points in calculating distances among places. The whole collection of locations forms a set of points. Definition (3.7) summaries our assumption.

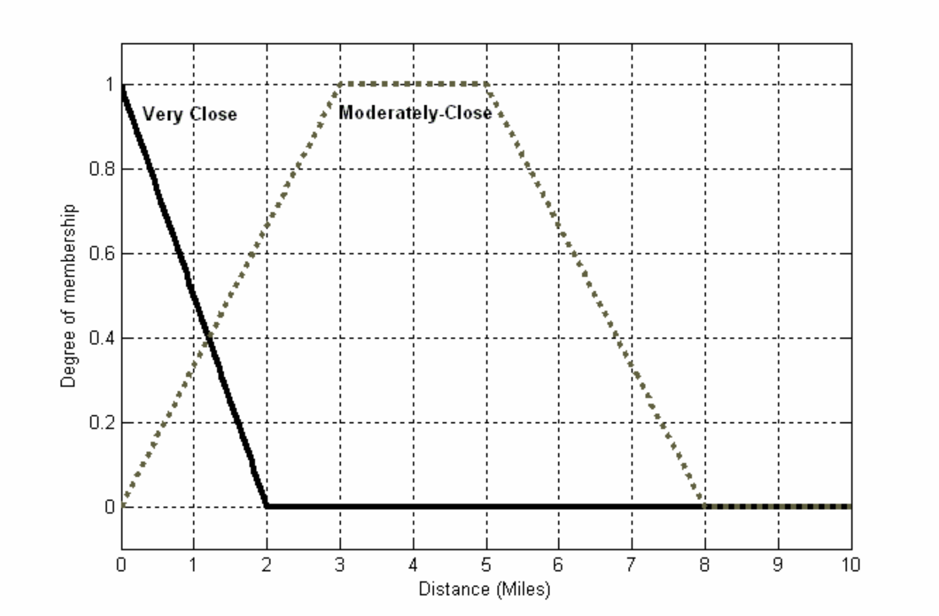


Fig. 9. Examples of distance membership functions.

Definition 3.7. A point-set *S* is said to be a *metric space* if there exists a function, distance, that takes ordered pairs (*s*,*t*) of elements of *S* and returns a real number distance(*s*, *t*) that satisfies the following three conditions:

M1. For each pair *s*, *t* in *S*, distance(*s*, *t*) > 0 if *s* and *t* are distinct points and distance(*s*, *t*) = 0 if, and only if, *s* and *t* are identical.

M2. For each pair *s*, *t* in *S*, the distance from *s* to *t* is equal to the distance from *t* to *s*, distance(*s*, *t*) = distance(*t*, *s*).

M3. (*Triangle inequality)* For each triple *s*, *t*, *u* in *S*, the sum of the distances from *s* to *t* and from *t* to *u* is always at least as large as the distance from *s* to *u*, that is:

$$\text{distance}(s, t) + \text{distance}(t, u) \geq \text{distance}(s, u).$$

The first condition M1 stipulates that the distance between points must be a positive number unless the points are the same, in which case the distance will be zero. The second condition M2 ensures that the distance between two points is independent of which way round it is measured. The third condition M3, the *triangle inequality*, states that it must always be at least as far to travel between two points via a third point rather than to travel directly.

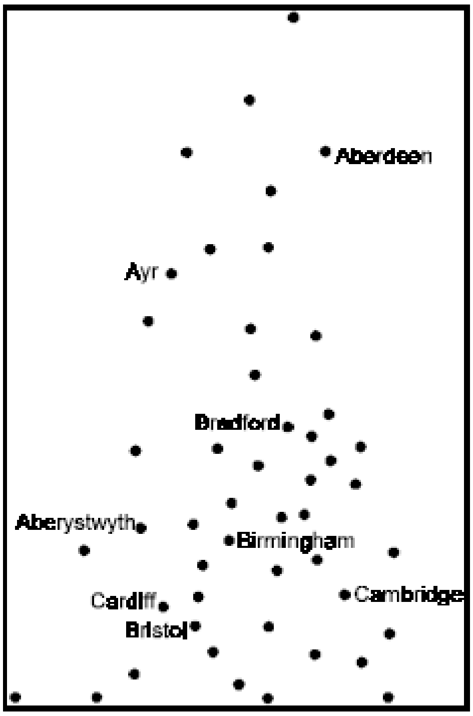
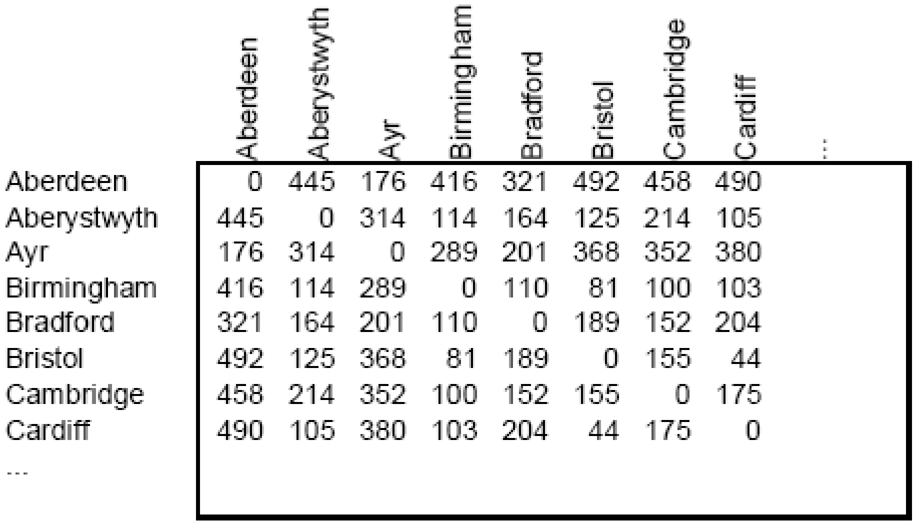


Fig. 10. Forty eight British centres of population.

We can see that a geographic space does not always admit such a metric. It is reasonable to suppose that condition M1 is satisfied by any distance function. However, context and travel-time both provide examples where condition M2 does not hold. Contextual knowledge is a key feature of human apprehension of geographic space. Different contexts provide us with quite distinct models of the surrounding space. For example, a bicyclist will have a different perception of his or her geographic neighbourhood from a driver of a wide load or an airline pilot. Even for the same person and application, distance may be perceived differently depending upon geographic location. Thus an observer in New York might perceive the distance from London to Edinburgh differently from an observer in London. Present computer systems do not generally support context-based representations. Condition M3 does hold for travel-time metrics, but does not hold for context related metrics. We illustrate this by means of an example.

Example 3.6. Figure 10 shows 48 centres in Great Britain. Their distances, measured in miles along major roads, have been calculated [Collins 1995], and some examples are given in Table 1. These distances relate to a global view, and are here termed *objective distances*, since they take no account of users, applications or locations (except that they assume users to be travellers along major roads).

Table 1. Part of the objective distance relationship between 48 British centres of population.



	Aberdeen	Aberystwyth	Ayr	Birmingham	Bradford	Bristol	Cambridge	Cardiff	...
Aberdeen	0	445	176	416	321	492	458	490	
Aberystwyth	445	0	314	114	164	125	214	105	
Ayr	176	314	0	289	201	368	352	380	
Birmingham	416	114	289	0	110	81	100	103	
Bradford	321	164	201	110	0	189	152	204	
Bristol	492	125	368	81	189	0	155	44	
Cambridge	458	214	352	100	152	155	0	175	
Cardiff	490	105	380	103	204	44	175	0	
...									

In this example, context is accounted for in the following manner. For each centre *c*, the mean μ_c of the distances from *c* to all centres is calculated. The relativised distance reldis from centre *c* to centre *d* is then determined by the formula:

$$\text{reldis}(c,d) = \frac{\text{distance}(c,d)}{\mu_c}$$

Some relativised distances are shown in Table 2. Note that the table is asymmetric, since reldis(*c*,*d*) $\neq$ reldis(*d*,*c*). This accords with our intuition regarding context dependent distance. For example, reldis(Aberdeen, Birmingham) = 1.1 and reldis(Birmingham, Aberdeen) = 2.6, reflecting the notion that from the perspective of Birmingham, closely surrounded by several centres, Aberdeen is relatively far away, but from the context of the relatively outlying and isolated Aberdeen, Birmingham is relatively closer.

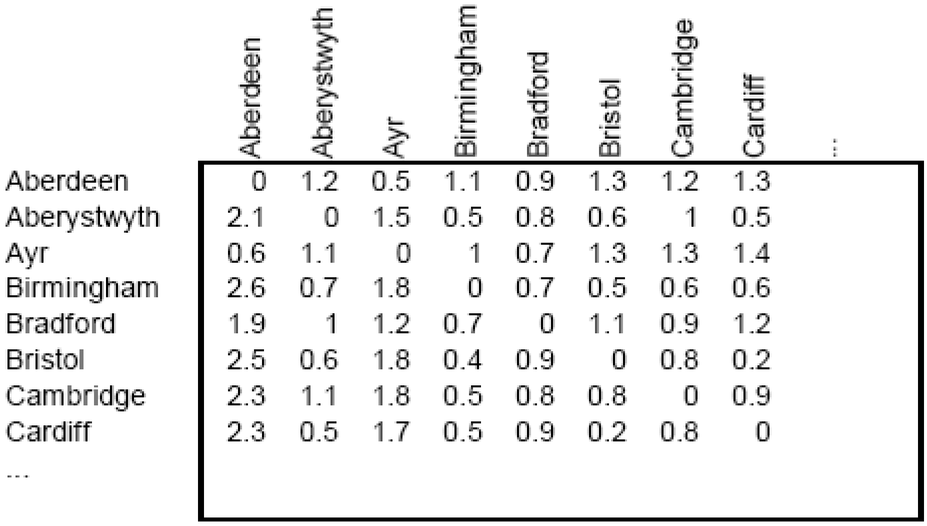
We may also note that the reldis relationship does not obey the triangle inequality. For example:

reldis (Birmingham, Aberdeen) = 2.6
reldis (Birmingham, Ayr) = 1.8
reldis (Ayr, Aberdeen) = 0.6

and so

reldis (Birmingham, Aberdeen) > reldis (Birmingham, Ayr) + reldis (Ayr, Aberdeen)

Table 2. Part of the relativised distance relationship between the 48 centres.



	Aberdeen	Aberystwyth	Ayr	Birmingham	Bradford	Bristol	Cambridge	Cardiff	...
Aberdeen	0	1.2	0.5	1.1	0.9	1.3	1.2	1.3	
Aberystwyth	2.1	0	1.5	0.5	0.8	0.6	1	0.5	
Ayr	0.6	1.1	0	1	0.7	1.3	1.3	1.4	
Birmingham	2.6	0.7	1.8	0	0.7	0.5	0.6	0.6	
Bradford	1.9	1	1.2	0.7	0	1.1	0.9	1.2	
Bristol	2.5	0.6	1.8	0.4	0.9	0	0.8	0.2	
Cambridge	2.3	1.1	1.8	0.5	0.8	0.8	0	0.9	
Cardiff	2.3	0.5	1.7	0.5	0.9	0.2	0.8	0	
...									

It will be useful in what follows to define a proximity or nearness relationship between geospatial entities. For geographic space *G*, define a function nearness, that takes ordered pairs (*s*,*t*) of elements of *G* and returns a real number nearness (*s*, *t*) that satisfies the following conditions:

1. $0 <$ nearness $(s, t) \leq 1$
2. nearness $(s, s) = 1$

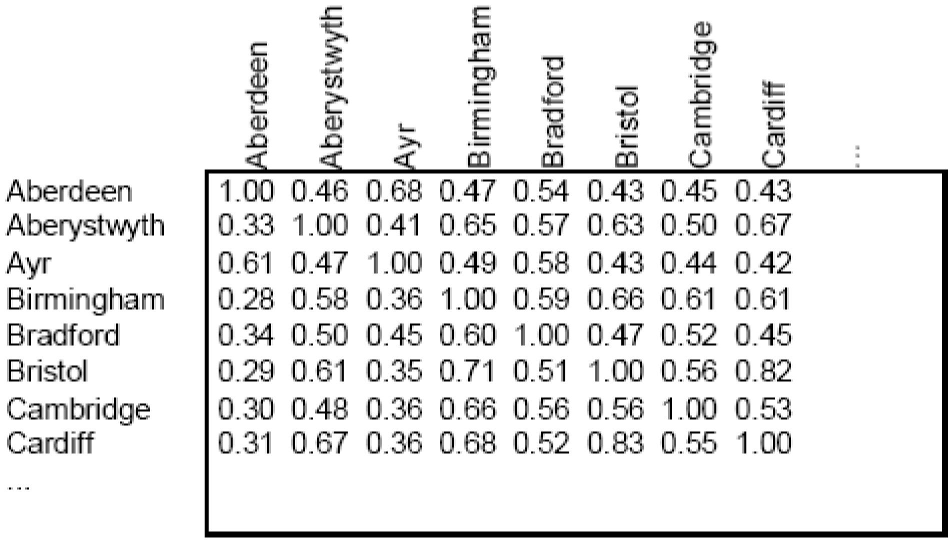
The idea is that if entity *y* is far from entity *x*, then nearness (*x*, *y*) will have a value close to zero, while if entity *y* is near to entity *x*, then nearness (*x*, *y*) will have a value close to 1. Note that, as with distance, nearness is context dependent and asymmetric in general.

For our example of the 48 British centres, we may derive a nearness measure from relative distance by means of the following formula:

$$\text{nearness}\ (x, y) = (\text{reldis}\ (x, y) + 1)^{-1}$$

The return value of the nearness function reflects how much time is needed if someone goes from *x* to *y*, although it is not a value of duration. It is in inverse proportion to the relativised distance from *x* to *y* because the larger the relativised distance from *x* to *y*, the more time it takes to go from *x* to *y*. One is added to the function reldis(*x*, *y*) because if *x* and *y* refer to the same place, the value of the function reldis(*x*, *y*) will be equal to zero. Then, we will be an error of division by zero in the nearness function. Values of the nearness relationship for the some of the 48 centres are shown in Table 3.

Table 3. Part of the nearness relationship between the 48 centres.



	Aberdeen	Aberystwyth	Ayr	Birmingham	Bradford	Bristol	Cambridge	Cardiff	…
Aberdeen	1.00	0.46	0.68	0.47	0.54	0.43	0.45	0.43	
Aberystwyth	0.33	1.00	0.41	0.65	0.57	0.63	0.50	0.67	
Ayr	0.61	0.47	1.00	0.49	0.58	0.43	0.44	0.42	
Birmingham	0.28	0.58	0.36	1.00	0.59	0.66	0.61	0.61	
Bradford	0.34	0.50	0.45	0.60	1.00	0.47	0.52	0.45	
Bristol	0.29	0.61	0.35	0.71	0.51	1.00	0.56	0.82	
Cambridge	0.30	0.48	0.36	0.66	0.56	0.56	1.00	0.53	
Cardiff	0.31	0.67	0.36	0.68	0.52	0.83	0.55	1.00	
…									

Like fuzzy calendar patterns, we can use fuzzy map patterns to represent those map patterns, which are asynchronous in the location dimension. In Section 3.3 discussing map patterns, we use Boolean functions to denote the coverage of map patterns. The Boolean function of a map pattern will return 1 for locations covered the map pattern and otherwise 0. According to this definition, Boolean functions for different levels of location representation can only return a singleton ($d_i \in D_i$) or a constant 1 (d_i = *). In order to find our targeted fuzzy spatio-temporal association rules, we will use membership functions to depict fuzzy map patterns. Values returned from the membership functions will be in the range between 0 and 1. The magnitude of a returned value represents the degree of membership of a particular location to the corresponding fuzzy map pattern of the membership function.

In a map schema $R=(P_c:D_c, P_{c-1}:D_{c-1}, \ldots, P_1:D_1)$, we let the fuzzy map pattern *FC*, defined by users under the map schema, has the membership function $F=(F_c, F_{c-1}, \ldots, F_1)$. F_i is a membership function for the corresponding spatial unit P_c in the map schema *R*. In that way, the degree of membership of a basic space interval $(d_c, d_{c-1}, \ldots, d_1)$, $d_i \in D_i$, i=1,…, *c* in the fuzzy map pattern *FM* is calculated as:

$$degree_{rc}(d_c, d_{c-1}, ..., d_1) = F_c(d_c) \times F_{c-1}(d_{c-1}) \times ... \times F_1(d_1)\,, \tag{3.4}$$

3.8 Fuzzy Calendar-map Patterns

We have explained fuzzy calendar patterns and fuzzy map patterns respectively. In a spatio-temporal database, if a phenomenon or a pattern is discovered, the time and location in which it occurs is an important piece of information. However, the occurrence time and locations of patterns are often uncertain and asynchronous in the sense that patterns seldom appear exactly and sharply at a single time period in a single place. Instead, we can only ensure that the patterns will appear around sometime near somewhere. For example,

> "*If people buy swimming suits, they will usually buy swimming goggle too in places close to Miami around July.*"

The fuzziness can be found in both the temporal and spatial dimensions of patterns. There is a chance that the time when the patterns occur is asynchronous while there is also a chance that the occurrence locations are inexact. Hence, we combine the concept of temporal fuzziness described in Section 3.6 and that of spatial fuzziness described in Section 3.7 and define in this section spatio-temporal fuzziness, which can be used to evaluate and quantize the spatio-temporal uncertainty. We also define fuzzy calendar-map patterns for specifying the fuzzy space-time intervals, in which spatio-temporal association rules are found. To be more precise, given two calendar-map patterns *A* and *C,* we are to formulate whether *C* can be regarded as "close to *A*" and to what extent. In other words, we are to calculate how much *C* is similar to *A*.

First of all, both of the calendar-map patterns *A* and *C* are respectively composed of a calendar pattern and a map pattern. We can denote *A* and *C* as:

$$A = A_t + A_s\,, \tag{3.5}$$

$$C = C_t + C_s \tag{3.6}$$

where A_t and C_t are calendar patterns and A_s and C_s are map patterns.

We assume the meaning of "close to *A*" as fulfilling "close to A_t" temporally and "close to A_s" spatially at the same time. Having the assumption, we can further define that FC_{A_t} represents the fuzzy calendar pattern "close to A_t", that is the set of calendar patterns {C_t | $\mathrm{degree}_{FC_{At}}(C_t) > 0$}. FM_{A_s} represents the fuzzy map pattern "close to A_s", that is the set of map patterns {C_s | $\mathrm{degree}_{FM_{As}}(C_s) > 0$}. The calculation of $\mathrm{degree}_{FC_{At}}(C_t)$ and $\mathrm{degree}_{FM_{As}}(C_s)$ are detailed in Section 3.6 and 3.7. Similarly, we define FCM_A as a set of calendar-map patterns such that {C | $\mathrm{degree}_{FCM_A}(C) > 0$}. $\mathrm{degree}_{FCM_A}(C)$ is the degree of membership quantizing how much the calendar-map pattern *C* is close to *A*. It is calculated as:

$$\begin{aligned}\mathrm{degree}_{FCM_A}(C) &= \mathrm{degree}_{FCM_A}(C_t + C_s) \\ &= \mathrm{degree}_{FC_{At}}(C_t) \times \mathrm{degree}_{FM_{As}}(C_s) \qquad (3.7)\end{aligned}$$

We call the inexact calendar-map pattern "close to *A*" a fuzzy calendar-map pattern. With the multiplication operation between $\mathrm{degree}_{FC_{At}}(C_t)$ and $\mathrm{degree}_{FM_{As}}(C_s)$ in the formula, for any calendar-map pattern *C* be said to be close to *A*, the two components of *C*, i.e. the calendar pattern C_t and map pattern C_s, have to be both close to those of *A* at the same time.

3.9 Mining Fuzzy Rules

Given i) a calendar-map schema $R=(f_n:D_n, f_{n-1}:D_{n-1}, \ldots, f_1:D_1)$ (a combination of a calendar schema and a map schema), where $(f_n:D_n, \ldots, f_i:D_i)$ is a calendar schema, $(f_{i-1}:D_{i-1}, \ldots, f_1:D_1)$ is a map schema and $1 \leq i \leq n$, ii) a group of datasets *D* partitioned by basic space-time intervals in the calendar-map schema *R*, iii) a user-defined threshold value called fuzzy match ratio threshold *fm* ($0 < fm \leq 1$) and iv) a user-defined fuzzy calendar-map pattern, in the calendar-map schema *R*, *FC* = "close to $(d_n, d_{n-1}, \ldots, d_1)$" (where $d_i \in D_i \cup *$, *i*=1, 2, …, *n*, i.e. $(d_n, d_{n-1}, \ldots, d_1)$ is a precise or non-fuzzy calendar-map pattern). A membership function $F=(F_n, F_{n-1}, \ldots, F_1)$, where F_n is the membership function of f_n in the calendar-map schema *R*, is defined. With the membership function, we can calculate the degree of membership of different space-time intervals to the fuzzy calendar-map pattern *FC*, so as to induce if association rules found in the space-time intervals are still valid in the fuzzy calendar-map pattern *FC*.

With a view to finding if an association rule exists in the fuzzy calendar-map pattern FC = "close to (d_n, d_{n-1}, …, d_1)", We perform Fuzzy Rule-based Inference. Firstly, we define a basic space-time interval t_i=(d'_n, d'_{n-1}, …, d'_1) where $d'_j \in D_j$, j=1, 2, …, n. We find the degree of membership of t_i to the fuzzy calendar-map "close to (d_n, d_{n-1}, …, d_1)" where $d_j \in D_j \cup *$, j=1, 2, …, n, based on the following if-then rules:

"If d'_n is close to d_n AND d'_{n-1} is close to d_{n-1} AND … AND d'_1 is close to d_1 THEN t_i is close to (d_n, d_{n-1}, …, d_1)"

With the user-defined membership function F=(F_n, F_{n-1}, …, F_1) of the fuzzy calendar-map pattern FC = "close to (d_n, d_{n-1}, …, d_1)", we can know the degree of membership $F_j(d'_j)$, meaning how d'_j is close to d_j, where j=1, 2, …, n. Using the AND operation, we can calculate the product of the degrees of membership of all temporal and spatial components in t_i. The product will be regarded as the degree of membership of t_i to the fuzzy calendar-map pattern FC.

When the degrees of membership of all basic space-time intervals are found, we can calculate the fuzzy match ratios, to FC, of the association rules discovered in basic space-time intervals. A matched association rule should have a fuzzy match ratio exceeding the user-defined fuzzy match ratio threshold fm. Fuzzy match ratio is a measure for judging if a mined rule is strong enough in a fuzzy time period and in a fuzzy location.

Assume we obtained the degree of membership $degree_{FC}(t_i)$ of each basic space-time interval t_i to FC. FC is the fuzzy calendar-map pattern "close to P", where P is a precise calendar-map pattern (d_n, d_{n-1}, …, d_1), $d_i \in D_i \cup *$, i=1, 2, …, n. Then the fuzzy match ratio FM of an association rule in FC is calculated as:

$$PD = \frac{\sum degree_{FC}(t_i) \times existed}{\sum degree_{FC}(t_i)}, \text{ where } t_i \text{ is covered by } P,, \tag{3.8}$$

$$FD = \frac{\sum degree_{FC}(t_i) \times existed}{\sum degree_{FC}(t_i)}, \text{ where } t_i \text{ is not covered by } P, \tag{3.9}$$

$$FM = PD + \alpha FD, \text{ where } 0 \le \alpha \le 1, \tag{3.10}$$

existed represents if the association rule exists in the space-time interval T_i. If yes, the value of *existed* will be 1 and otherwise 0. *PD* is the ratio that the association rule occurs in the space-time intervals covered by the precise calendar-map pattern *P*. *FD* is the ratio that the association rule occurs in the asynchronous space-time intervals, which are covered by the fuzzy calendar-map pattern *FC* but not the precise calendar-map pattern *P*. α is a parameter adjustable by users in the calculation of *FM*, in order to tune how much the impact of the asynchronous space-time intervals in the fuzzy calendar-map pattern FC will be counted on. The larger the value of α, the larger the impact will be. When if α is set to 0, the asynchronous space-time intervals will be totally ignored. With the formula for calculating the fuzzy match ratio *FM*, we first use *PD* and *FD* to find those association rules, which are valid in space-time intervals represented by precise calendar-map patterns and asynchronous space-time intervals in fuzzy calendar-map patterns respectively, so as to discover our fuzzy association patterns.

If the calculated *FM* is larger than the user-defined fuzzy match ratio threshold *fm*, we can denote the discovered fuzzy spatio-temporal association rule as:

$$x \xrightarrow{FC} y$$

which represents the association rule *x*→*y* is valid in the space-time intervals covered by the fuzzy calendar-map pattern *FC*, i.e. close to (d_n, d_{n-1}, …, d_1).

Example 3.7. Under the calendar-map schema *R*=(year:{1996, …, 2000}, month:{1, …, 12}, day:{1, …, 31}, country:{United States}, state:{California, Florida}, city:{Los Angeles, Orlando}), we can define a set of membership functions F=(F_{year}, F_{month}, F_{day}) of the fuzzy calendar pattern “close to (*, 11, 25)” as shown in Figure 6. We can also define that a set of membership functions F'=($F_{country}$, F_{state}, F_{city}) of the fuzzy map pattern “close to (United States, California, Los Angeles)” in the way that all precise map patterns will have the degrees of membership equal to 0 except the map pattern (United States, California, Los Angeles), which has the degree of membership equal to 1. That is, among all cities, only Los Angeles is regarded as close to itself. We can group *F* and *F’* to form a new set of membership functions F''=(F_{year}, F_{month}, F_{day}, $F_{country}$, F_{state}, F_{city}) for the fuzzy calendar-map pattern “close to (*, 11, 25, United States, California, Los Angeles)” in the calendar-map schema *R*. In the set of membership functions, the definition of “close to (*, 11, 25, United States, California, Los Angeles)” is defined as within the date range ± 2 days from November 25 of each year in Los Angeles. On calculating the degree of membership, only the space-time intervals represented by the 1-star calendar-map patterns (*, 11, 23, United

States, California, Los Angeles), (*, 11, 24, United States, California, Los Angeles), (*, 11, 25, United States, California, Los Angeles), (*, 11, 26, United States, California, Los Angeles), (*, 11, 27, United States, California, Los Angeles) will have non-zero degrees. In other words, except these combinations of dates and cities, there cannot be any other space-time intervals which are defined to be "close to (*, 11, 25, United States, California, Los Angeles)". Therefore, the transactions with timestamp and location identifier not equal to any of these space-time intervals can be skipped in scanning the database to find frequent patterns. For example, the degree of membership of the date (1998, 11, 20) plus the city (United States, Florida, Orlando) to the fuzzy calendar pattern "close to (*, 11, 25, United States, California, Los Angeles)" = $F_{year}(1998) \times F_{month}(11) \times F_{day}(20) \times F_{country}$(United States)$\times F_{state}$(Florida) $\times F_{month}$(Orlando) = $\overbrace{(1\times 1\times 0)}^{\text{temporal part}} \times \overbrace{(1\times 0\times 0)}^{\text{spatial part}} = 0$. The results are zero for both the temporal and spatial parts of the multiplication operation. Hence, we do not need to consider those transactions with this timestamp OR this location.

Assume we need to find association rules in the space-time intervals covered by the fuzzy calendar-map pattern "close to (*, 11, 25, United States, California, Los Angeles)". After scanning the database, we find that the association rule "$turkey \rightarrow pumpkin$" exists in the calendar-map patterns:

1. (1996, 11, 23, United States, California, Los Angeles),
2. (1996, 11, 24, United States, California, Los Angeles),
3. (1996, 11, 26, United States, California, Los Angeles),
4. (1997, 11, 25, United States, California, Los Angeles),
5. (1997, 11, 26, United States, California, Los Angeles),
6. (1998, 11, 24, United States, California, Los Angeles),
7. (1998, 11, 26, United States, California, Los Angeles),
8. (1998, 11, 27, United States, California, Los Angeles),
9. (1999, 11, 23, United States, California, Los Angeles),
10. (1999, 11, 25, United States, California, Los Angeles),
11. (1999, 11, 26, United States, California, Los Angeles),
12. (2000, 11, 25, United States, California, Los Angeles),
13. (2000, 11, 26, United States, California, Los Angeles),
14. (2000, 11, 27, United States, California, Los Angeles),
15. (2000, 11, 26, United States, Florida, Orlando),

16. (2000, 11, 27, United States, Florida, Orlando).

Then, we can calculate the fuzzy match ratio of the association rule in the fuzzy calendar pattern “close to (*, 11, 25, United States, California, Los Angeles)” as:

$PD = \{$

$F_{year}(1997)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1999)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(2000)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})\}$

$\div \{$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1997)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1998)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1999)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(2000)\times F_{month}(11)\times F_{day}(25)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})\}$

$= (1 + 1 + 1) \div 5 = 0.6$

$FD = \{$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(23)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(24)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1997)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1998)\times F_{month}(11)\times F_{day}(24)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1998)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1998)\times F_{month}(11)\times F_{day}(27)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1999)\times F_{month}(11)\times F_{day}(23)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1999)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(2000)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(2000)\times F_{month}(11)\times F_{day}(27)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})\}$

$\div \{$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(23)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(24)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles}) +$

$F_{year}(1996)\times F_{month}(11)\times F_{day}(27)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1997)\times F_{month}(11)\times F_{day}(23)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1997)\times F_{month}(11)\times F_{day}(24)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$F_{year}(1997)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+$

$$
\begin{aligned}
&F_{year}(1997)\times F_{month}(11)\times F_{day}(27)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1998)\times F_{month}(11)\times F_{day}(23)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1998)\times F_{month}(11)\times F_{day}(24)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1998)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1998)\times F_{month}(11)\times F_{day}(27)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1999)\times F_{month}(11)\times F_{day}(23)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1999)\times F_{month}(11)\times F_{day}(24)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1999)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(1999)\times F_{month}(11)\times F_{day}(27)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(2000)\times F_{month}(11)\times F_{day}(23)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(2000)\times F_{month}(11)\times F_{day}(24)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(2000)\times F_{month}(11)\times F_{day}(26)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})+\\
&F_{year}(2000)\times F_{month}(11)\times F_{day}(27)\times F_{country}(\text{United States})\times F_{day}(\text{California})\times F_{day}(\text{Los Angeles})\}
\end{aligned}
$$

$$
\begin{aligned}
&= (0.33 + 0.67 + 0.67 + 0.67 + 0.67 + 0.67 + 0.33 + 0.33 + 0.67 + 0.67 + 0.33) \div\\
&\quad (0.33 + 0.67 + 0.67 + 0.33 + 0.33 + 0.67 + 0.67 + 0.33 + 0.33 + 0.67 + 0.67 + 0.33 +\\
&\quad\ 0.33 + 0.67 + 0.67 + 0.33 + 0.33 + 0.67 + 0.67 + 0.33)
\end{aligned}
$$

$$= 0.601$$

$$FM = PD + \alpha FD$$

Assume $\alpha=0.3$, then $FM = 0.6+0.3\times0.601 = 0.6+0.1803 = 0.7803$. Therefore, if the fuzzy match ratio threshold is set to a value smaller than 0.7803, the association rule "$turkey \rightarrow pumpkin$" is valid in the space-time intervals covered by the fuzzy calendar-map pattern "close to (*, 11, 25, United States, California, Los Angeles)". The association rule can be represented as:

$$\text{turkey} \xrightarrow{\textit{close-to}\ (*,11,\,25,\,\text{United States, California, Los Angeles})} \text{pumpkin}$$

It means in places close to Los Angeles, at around November 25 of each year, there exists a fuzzy spatio-temporal association rule that people likely buy turkey together with pumpkin.

3.10 Mining Meta-rules

The spatio-temporal patterns discovered by solving the association rule mining problems described in previous sections will change as time goes by. In other words, patterns mined in this year are probably different from those mined last year. Understanding and adapting to changes of patterns, e.g. customer behaviour, is an important aspect of surviving in a continuously changing environment. Especially for businesses, knowing what is changing and how it has been changed is of crucial importance because it allows businesses to provide the right products and services to suit the changing market needs [Liu *et al.* 2000]. Data mining is the process of exploration and analysis of large quantities of data in order to discover meaningful patterns and rules. But much of existing data mining research has been focused on devising techniques to build accurate models and to discover rules. Relatively little attention has been paid to mining changes in databases collected over time [Liu *et al.* 2000]. We try to propose a method to solve the problem in this study. We expect the solution should be able to be applied without any changes to discover regularities and differences in patterns between two different calendar-map patterns or even fuzzy calendar-map patterns.

3.10.1 Mining Regularities and Differences

Let us suppose that there is a collection of data sets, $\{D_j, j = 1, \ldots, n\}$, collected at the same time. D_v is a data set identified by the fuzzy calendar-map pattern (or precise calendar-map pattern) FCM_v, where $1 \leq v \leq n$. A set of rules, $R_j = \{r_{j1}, ..., r_{js_j}\}$, is mined from D_j, j = 1, …, n. From R_1, …, R_n, we aim at mining a set of meta-rules to reveal the underlying regularities hidden in the rule sets and the differences between different rule sets.

Definition 3.8. A *meta-rule* mined from rule sets R_1, …, R_n is an implication of the form:

$$X \Rightarrow Y,$$

where X and Y are conjunctions of conditions such that $condition(X) \subseteq \bigcup_{j=1}^{n} condition(R_j)$, $condition(Y) \subseteq \bigcup_{j=1}^{n} condition(R_j)$, and $condition(X) \cap condition(Y) = \varnothing$.

Rather than being supported by data records, a meta-rule is supported by the rules in the rule sets. We say that a rule supports a meta-rule if the set of conditions in the meta-rule is a subset of that in the rule.

Definition 3.9 A meta-rule, $X \Rightarrow Y$, mined from rule sets R_1, …, R_n, is supported by a set of rules:

$$\mathcal{R}(X \Rightarrow Y) = \{r \mid r \in R_1 \cup \ldots \cup R_n, condition(X) \cup condition(Y) \subseteq condition(r)\}. \tag{3.11}$$

A meta-rule represents an association relationship in common in the rule sets if many rules support it. In other words, it represents an underlying regularity hidden in the rule sets.

Definition 3.10. A meta-rule, $X \Rightarrow Y$, mined from rule sets R_1, …, R_n, represents a regularity embedded in them if $|\mathcal{R}(X \Rightarrow Y)|$ is *sufficiently large*. We refer to this meta-rule as a *regular meta-rule*.

On the other hand, a meta-rule represents a distinctive association relationship in the rule sets if only a few rules support it. In other words, it represents a difference between the rule sets.

Definition 3.11. A meta-rule, $X \Rightarrow Y$, mined from rule sets R_1, …, R_n, represents a difference between them if $|\mathcal{R}(X \Rightarrow Y)|$ is *sufficiently small*. We refer to this meta-rule as a *differential meta-rule*.

To reveal regularities and differences in rule sets, we mine regular and differential meta-rules from the rule sets, respectively.

Example 3.8. Let us consider rule sets R_1, …, R_5, each of which contains a set of association rules. They are given in the following:

R_1: $\{i_1, i_2\} \Rightarrow \{i_3\}$
$\{i_4\} \Rightarrow \{i_1\}$

R_2: $\{i_1, i_2\} \Rightarrow \{i_3\}$
$\{i_2, i_3, i_5\} \Rightarrow \{i_4\}$
$\{i_2, i_3\} \Rightarrow \{i_4\}$

R_3: $\{i_2, i_3, i_5\} \Rightarrow \{i_4\}$

R_4: $\{i_1, i_2\} \Rightarrow \{i_3\}$
$\{i_2, i_3, i_5\} \Rightarrow \{i_4\}$

$$R_5\text{:}\ \{i_1, i_2\} \Rightarrow \{i_3\},$$

where i_1, …, i_5 are items.

The meta-rule $\{i_2, i_3\} \Rightarrow \{i_4\}$ is supported by the following rules:

$$R_2\text{:}\ \{i_2, i_3, i_5\} \Rightarrow \{i_4\}$$
$$\{i_2, i_3\} \Rightarrow \{i_4\}$$

$$R_3\text{:}\ \{i_2, i_3, i_5\} \Rightarrow \{i_4\}$$

$$R_4\text{:}\ \{i_2, i_3, i_5\} \Rightarrow \{i_4\},$$

whereas the meta-rule $\{i_4\} \Rightarrow \{i_1\}$ is supported by the following rule:

$$R_1\text{:}\ \{i_4\} \Rightarrow \{i_1\}.$$

The former and the latter meta-rules are supported by 44.4% (= 4 / 9) and 11.1% (= 1 / 9) of all the rules, respectively.

A straightforward approach to determining whether a meta-rule is supported by a sufficiently large or small number of rules is to have a user supply thresholds. For example, if the threshold for determining regular meta-rules is set to 40%, the former meta-rule is found to be regular; and if the threshold for determining differential meta-rules is set to 15%, the latter meta-rule is found to be differential.

3.10.2 Mining Changes

We are also concerned with mining a set of meta-rules to reveal how the rules in the rule sets change over time.

Now, let us further suppose that $\{D_j, i = 1, \ldots n\}$ is set of data identified by a single particular calendar-map pattern or fuzzy calendar-map pattern. D_j is collected in time periods t_j, $j = 1, \ldots, n$, where $t_1, \ldots, t_n$ are consecutive and t_j happens before t_k if $j < k$. Let us consider rules $r_{ju} \in R_j$ and $r_{kv} \in R_k$, $j, k \in \{1, \ldots, n\}$, $j < k$. These represent the same association relationship if, and only if, *antecedent*(r_{ju}) = *antecedent*(r_{kv}) and *consequent*(r_{ju}) = *consequent*(r_{kv}).

Definition 3.12. Given a rule, $r_{ju} \in R_j$, if there exists another rule, $r_{kv} \in R_k$, $j < k$, such that *antecedent*(r_{ju}) = *antecedent*(r_{kv}) and *consequent*(r_{ju}) = *consequent*(r_{kv}), r_{ju} is *equivalent* to r_{kv}, denoted as $r_{ju} \equiv r_{kv}$, because they represent the same association relationship.

It is important to note that although $r_{ju} \equiv r_{kv}$, its interestingness measure in t_j may be different from that in t_k because the rule may change as will be discussed in Definitions 3.13–3.15.

Definition 3.13. Given two rules, $r_{ju} \in R_j$ and $r_{kv} \in R_k$, $j < k$, such that $r_{ju} \equiv r_{kv}$, r_{ju} *changes* during the period from t_j to t_k if $interestingness_j(r_{ju}) \neq interestingness_k(r_{ju})$. We say that r_{ju} is an *emerging pattern* in t_k.

It is possible that rule r_{ju} is found in R_j but not in R_k because it is interesting in t_j but it becomes uninteresting in t_k, $j < k$.

Definition 3.14. Given R_j and R_k, $j < k$, if $r_{ju} \in R_j$ and there does not exist $r_{kv} \in R_k$ such that $r_{ju} \equiv r_{kv}$, we say that r_{ju} is *perished* in t_k and r_{ju} is a *perished rule* in t_k. In this case, the interestingness measure of r_{ju} in t_k is missing.

On the other hand, it is also possible that r_{kv} is not found in R_j but is found in R_k because it is uninteresting in t_j but it becomes interesting in t_k, $j < k$.

Definition 3.15. Given R_j and R_k, $j < k$, if $r_{kv} \in R_k$ and there does not exist any $r_{ju} \in R_j$ such that $r_{ju} \equiv r_{kv}$, we say that r_{kv} is *added* in t_k and r_{kv} is an *added rule* in t_k. In this case, the interestingness measure of r_{kv} in t_j is missing.

An added rule or a perished rule is a special case of an emerging pattern. It is special in that an added rule’s interestingness measure changes from below a threshold to above it, whereas a perished rule’s interestingness measure changes in the reverse direction, from above the threshold to below it. The threshold can be specified by a user or determined by an objective means. Revealing how a rule changed in the past allows one to predict whether it will be added or perished or to what degree it will change in the future.

For each rule in $R_1 \cup \ldots \cup R_n$, we are interested in mining a set of meta-rules to represent the regularities governing how the rule changes during the period from t_1 to t_n. We refer to these meta-rules as *change meta-rules* because they represent how the rule changes over time.

Definition 3.16. For $r \in R_1 \cup \ldots \cup R_n$, a *change meta-rule* is an implication of the form:

$$L^r_{j_1} = l^r_{p_1} \wedge \ldots \wedge L^r_{j_h} = l^r_{p_h} \Rightarrow L^r_{j_q} = l^r_{p_q} \text{ ,}$$

where $L^r_{j_k}$ is an attribute representing $\Delta interestingness_{j_k}(r) = interestingness_{j_k+1}(r) - interestingness_{j_k}(r)$ (i.e., the difference in the interestingness measure of r during the period from t_{j_k} to t_{j_k+1}) and $l^r_{p_k}$ is an attribute value in $dom(L^r_{j_k})$, which denotes the domain of $L^r_{j_k}$, for k = 1, …, h, q and $j_1 < \ldots < j_h < j_q$.

The mining of change meta-rules allows the examination of the regularities governing how a rule changes during a period t_1 to t_n. The discovered meta-rules can also be used to predict how the rule will change in t_{n+1}. The ability to predict how rules will change allows accurate results to be achieved when the discovered rules in the past are used for classification in the future.

Example 3.9. Let us consider the association rules of items i_1, i_2, i_3, and i_4 discovered in three consecutive time periods, t_1, t_2, and t_3. Assume that the association rule discovered in time period t_1 is:

$$r: \quad \{i_1, i_2, i_3\} \Rightarrow \{i_4\}$$

whose support and confidence in t_1 are $support_1(r)$ = 37.8% and $confidence_1(r)$ = 95.0%, respectively.

In time period t_2, the association rule becomes:

$$r': \quad \{i_1, i_2, i_3\} \Rightarrow \{i_4\}$$

whose support and confidence in t_2 are $support_2(r)$ = 34.9% and $confidence_2(r)$ = 94.8%, respectively.

Then in time period t_3, the association rule becomes:

$$r'': \quad \{i_1, i_2, i_3\} \Rightarrow \{i_4\}$$

whose support and confidence in t_3 are $support_3(r)$ = 28.4% and $confidence_3(r)$ = 94.5%, respectively.

The support of the association rule decreases in the period from t_1 to t_2 and in the period from t_2 to t_3. A change meta-rule of support mined from these rules would be:

$$\begin{array}{l} \textit{Change in support in this period} = \textit{Fairly decrease} \\ \quad \Rightarrow \textit{Change in support in next period} = \textit{Highly decrease}. \end{array}$$

This meta-rule of support states that "if the change in support in this period moderately decreases, then the change in support in next period will decrease significantly." The support of the association rule in t_j can then be predicted given the support of this rule in t_{j-1} and that in t_{j-2}.

On the other hand, the confidence of the association rule is more or less the same in the period from t_1 to t_2 and in the period from t_2 to t_3. A change meta-rule of confidence discovered in these rules would be:

$$\begin{array}{l} \textit{Change in confidence in this period} = \textit{More or less the same} \\ \quad \Rightarrow \textit{Change in confidence in next period} = \textit{More or less the same}. \end{array}$$

This states that "if the change in confidence in this period is more or less the same, then the change in confidence in next period will be more or less the same." The confidence of the association rule in t_j can then be predicted given the confidence of this rule in t_{j-1} and that in t_{j-2}.

The other type of change is unexpectedness which is found from many studies about discovering interesting patterns [Liu and Hsu 1996; Liu *et al.* 1997; Padmanabhan and Tuzhilin 1999; Silberschatz and Tuzhilin 1996; Suzuki 1997]. Liu and Hsu (1996) defined unexpected changes as rule similarity and difference aspects. They distinguished unexpected changes to unexpected condition changes and unexpected consequent changes based on a syntactic comparison between a rule and a belief. But we only adapt unexpected consequent changes because most unexpected condition changes usually make no sense. These unexpected consequent changes are the second type of change to detect which has a different rule structure over time. Therefore we redefine the term unexpected changes like the following from the study of Liu and Hsu (1996).

Definition 3.17. Unexpected Changes (or Unexpected Consequent Changes)

Given R_j and R_k, $j < k$, if $r_{ju} \in R_j$ and there exists $r_{kv} \in R_k$ such that the antecedent parts of r_{ju} and r_{kv} are similar but the consequent parts of the two rules are quite different, r_{kv} is unexpected change with respect to r_{ju}.

Example 3.10.

r_{ju} : Income = High, Age = High $\Rightarrow$ Model = Large

r_{kv} : Income = High, Age = High $\Rightarrow$ Model = Medium

In this case, r_{kv} is unexpected consequent change with respect to r_{ju} since the antecedent parts of the two rules are similar, but the consequent parts of the two rules are quite different.

Chapter 4
The Proposed Approaches

In Section 4.1, we propose a way to extend Apriori, the most well-known association rule mining algorithm, to mine spatio-temporal association rules that we have defined in Chapter 3. We call the extended algorithm Spatio-temporal Apriori. In Section 4.2, we propose our new algorithm to mine the kind of spatio-temporal association rules. Compared with Spatio-temporal Apriori, our method is faster and consumes less memory, because it scans the database at most twice. In Section 4.3, we fuzzify our method by using fuzzy calendar-map patterns to find fuzzy rules. Given the sets of rules mined in different combinations of time intervals and locations, we can mine meta-rules, which are regularities, differences and changes among the different sets of rules. Our meta-mining approach will be discussed in Section 4.4.

4.1 Spatio-temporal Apriori

We propose an Apriori-like method, as one of the feasible methods to find frequent itemsets in different calendar-map patterns, which meet the minimum support threshold defined by users. We name the calendar-map patterns, in which the frequent itemsets are found, frequent calendar-map patterns. We call the entire algorithm Spatio-temporal Apriori, which is an extension of Apriori. Spatio-temporal Apriori is simple to implement. It follows the algorithmic flow of Apriori to find frequent itemsets, in which the mined dataset is scanned pass by pass. Frequent itemsets found in one pass are used to create the candidate itemsets in the next pass until no more frequent itemsets can be found. The Apriori algorithm is based on the Apriori theory, which states that “All subsets of frequent itemsets are frequent itemsets”. Hence, in creating candidate itemsets, the algorithm observes a rule, which states ”When an itemset is not frequent, no supersets of it can be made candidate itemsets”. This rule can help reduce the number of candidate itemsets generated.

In the Apriori algorithm, the subroutine, which applies the rule to create candidate itemsets, is called Apriori_gen:

Assume the set of frequent itemsets in the previous level is L_{k-1}. L_{k-1} is a sorted list of itemsets, in which each itemset is unique.

1. For each $x \in L_{k-1}$, $y \in L_{k-1}$, $x = (x_1, x_2, ..., x_{k-1})$, $y = (y_1, y_2, ..., y_{k-1})$, find all combinations

of x and y, where $x_1 = y_1$, $x_2 = y_2$, ..., $x_{k-2} = y_{k-2}$ but $x_{k-1} < y_{k-1}$.

2. Form a new k-itemset $z = (x_1, x_2, ..., x_{k-1}, y_{k-1})$, such that z is the union of x and y, with all combinations of x and y found in Step 1 found.
3. For each z formed in Step 2, count the number of $x \in L_{k-1}$, such that $x \subset z$. let it be m. If $m = k$, then z is one of the candidates in frequent k-itemsets.
4. The union of all z fulfilling the criteria from Step 1 to Step 3 will form a set of candidates of frequent k-itemsets.

For example, assume L_2 = {*AB*, *AC*, *AD*, *BC*}. Then, with the operation $L_2 \times L_2$, *ABC* can be generated from *AB* and *AC*. *ABD* can be generated from *AB* and *AD*. *ACD* can be generated from *AC* and *AD*. However, due to the fact that *BD* is not in L_2, *ABD* cannot be a candidate itemset. Likewise, *CD* does not exist in L_2, so *ACD* cannot be a candidate too. Hence L_3 will become {*ABC*}.

Spatio-temporal Apriori also uses the Apriori_gen subroutine to find frequent itemsets in each basic space-time interval. Figure 11 shows a flowchart of the Spatio-temporal Apriori algorithm. In the algorithm, a basic space-time interval is the smallest unit to partition the target dataset. Spatio-temporal Apriori follows the algorithmic flow of Apriori, to search for frequent itemsets in frequent calendar-map patterns. Firstly, it finds all frequent 1-itemsets in each basic space-time interval. Then, for each basic space-time interval, it rolls up the counts of the frequent 1-itemsets to that in those calendar-map patterns, which cover the basic space-time intervals.

For example, in the calendar-map schema R = (year: {1996, ..., 2000}, month: {1, ..., 12}, day: {1, ..., 31}, province/state: {California, Florida}, city: {Los Angeles, San Francisco, San Diego, Orlando, Miami, Atlantis}), the items A, B, C are frequent in the basic space-time interval (2000, 10, 20, California, Los Angeles). The n-Star Calender-map patterns that cover the basic space-time intervals are:

1. (2000, 10, *, California, Los Angeles)
2. (2000, *, 20, California, Los Angeles)
3. (*, 10, 20, California, Los Angeles)
4. (2000, *, *, California, Los Angeles)
5. (*, *, 20, California, Los Angeles)
6. (*, 10, *, California, Los Angeles)
7. (2000, 10, *, California, *)

8. (2000, *, 20, California, *)
9. (*, 10, 20, California, *)
10. (2000, *, *, California, *)
11. (*, *, 20, California, *)
12. (*, 10, *, California, *)

Hence, the count of the items (*A*, *B*, *C*) is accumulated to that in the 12 calendar-map patterns when the database partition identified by (2000, 10, 20, California, Los Angeles) is being scanned. At last, after the first scanning of the whole database, the algorithm computes the match ratio of each 1-itemsets in each *n*-star calendar-map patterns, which is equal to the number of (*n*-1)-star calendar-map patterns covered by the *n*-star calendar-map pattern containing the 1-itemsets. The algorithm will keep those 1-itemsets with match ratios larger than or equal to the user-defined match ratio threshold. The kept 1-itemsets in their corresponding *n*-star calendar-map patterns are therefore frequent 1-itemsets in frequent calendar-map patterns.

From the second pass on, we will manipulate each basic space-time interval in 3 steps in every pass: 1) Generate candidate itemsets from the frequent itemsets found in the previous pass. 2) Compute the support of the candidate itemsets in the basic space-time interval. 3) Use the count of frequent itemsets to update that in the *n*-star calendar-map patterns that cover the basic space-time interval. We repeat the 3 steps until no more frequent itemsets can be discovered in any calendar-map patterns.

Spatio-temporal Apriori can search multiple different time intervals and locations with different granularities for frequent patterns with the help of calendar-map schema in one process. Hence, users do not need to find association rules period by period or place by place. However, Spatio-temporal Apriori can have a lot of improvement in term of performance and efficiency. Firstly, as in Apriori, a long database scanning time may be needed, especially for large datasets. The number of database scans needed also depends on the maximum allowed length of frequent itemsets. Secondly, to generate candidate itemsets in each basic space-time interval, the algorithm have to keep track of all the frequent itemsets found in the previous pass. The frequent itemsets can be very large, especially when there are a large number of basic time intervals and locations. Thirdly, in scanning database, the algorithm needs to keep and update the count of itemsets in the whole hierarchy of calendar-map patterns. This can be costly in term of processor and memory utilization. With a view to achieving better performance, we design a new algorithm for mining spatio-temporal association rules, which will be detailed in Section 4.2.

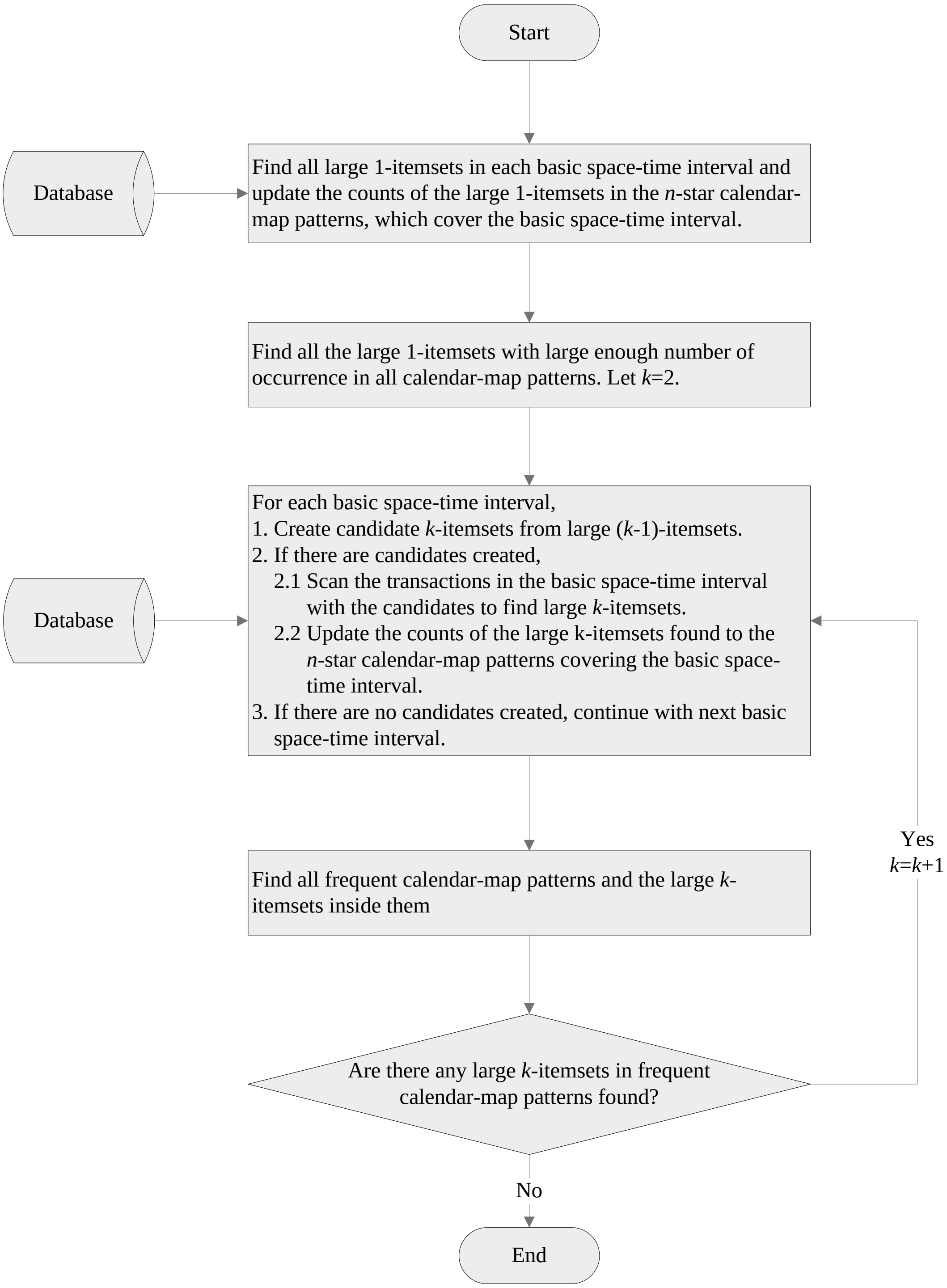

Start
Database
Find all large 1-itemsets in each basic space-time interval and update the counts of the large 1-itemsets in the *n*-star calendar-map patterns, which cover the basic space-time interval.
Find all the large 1-itemsets with large enough number of occurrence in all calendar-map patterns. Let *k*=2.
Database
For each basic space-time interval,
1. Create candidate *k*-itemsets from large (*k*-1)-itemsets.
2. If there are candidates created,
2.1 Scan the transactions in the basic space-time interval with the candidates to find large *k*-itemsets.
2.2 Update the counts of the large k-itemsets found to the *n*-star calendar-map patterns covering the basic space-time interval.
3. If there are no candidates created, continue with next basic space-time interval.
Find all frequent calendar-map patterns and the large *k*-itemsets inside them
Are there any large *k*-itemsets in frequent calendar-map patterns found?
Yes
k=*k*+1
No
End

Fig. 11. The Spatio-temporal Apriori algorithm.

4.2 Our Spatio-temporal Association Rule Mining

In a common situation, an association rule is generated from a frequent itemset with at least two items. In Apriori-like methods, if we compare the number of candidate itemsets with that of frequent itemsets found, we can induce that the largest number of redundant candidates are often candidates of 2-itemset. Moreover, relatively speakingly, the lower the support threshold, the more the number of redundant candidates of 2-itemset. Based on the observation, our algorithm directly starts searching for frequent 2-itemset instead of generating candidates of 2-itemset from frequent 1-itemsets and then looking for frequent 2-itemsets within the candidates. Hence, we design a new algorithm to effectively discover the frequent calendar-map patterns of all frequent itemsets in the target database. Figure 12 is flowchart outlining our algorithm.

Roughly, our method for discovering spatio-temporal association rules can be divided into three phases. They are 1) discovery of frequent 2-itemsets along with their 1-star candidate calendar-map patterns, 2) generation of candidate itemsets along with all their *k*-star candidate calendar-map patterns, and 3) discovery of frequent itemsets along with their frequent calendar-map patterns. In phase 1**,** we are to discover all frequent 2-itemsets L_2 in the dataset D composed of a set of partitions $\{P_i\}$, where a partition P_i is uniquely identified by a basic space-time interval T_i. A frequent 2-itemset $l_2 \in L_2$ discovered will then be stored with a set of 1-star candidate calendar-map patterns $\{l_2.v_i^1\}$, where each $l_2.v_i^1$ covers the basic space-time interval T_i, in which l_2 is discovered. (Note that a basic space-time interval can be covered by more than one 1-star calendar-map pattern.) $l_2.V^1$ is a set of all the 1-star candidate calendar-map patterns of l_2, such that $l_2.v_i^1 \in l_2.V^1 \subset L_2.V^1$.

Note that L_2 and $L_2.V^1$ are empty initially. In the partition P_1, every frequent 2-itemset is found out and inserted into L_2. For each itemset l_2 inserted into L_2, the set of 1-star candidate calendar-map patterns covering T_1 is kept in $l_2.V^1$. The repeating count of l_2 in each $l_2.v_i^1$ is set to 1.

In the rest of the partitions, $P_2, ..., P_n$, discovery of frequent 2-itemsets along with their 1-star candidate calendar-map patterns is iterated partition by partition. Frequent 2-itemsets in a partition P_i, where $2 \le i \le n$, are computed with three different cases. In case 1, a frequent 2-itemset, l_2, is not currently in L_2, and therefore it is inserted into L_2. Also, $\{l_2.v_i^1\}$, a set of 1-star candidate calendar-map patterns covering T_i is kept in $l_2.V^1$. The repeating count of l_2 in each $l_2.v_i^1$ is set to 1. In case 2, the frequent 2-itemset, l_2, is already in L_2, but a new 1-star calendar-map pattern, $l_2.v_i^1$, with respect to the current partition is found. In this case, $l_2.v_i^1$ is inserted into $l_2.V^1$ and l_2*'s* repeating count in $l_2.v_i^1$ is set to 1. In case 3**,** the frequent 2-itemset, l_2, is already in L_2, and all the corresponding 1-star calendar-map patterns, $\{l_2.v_i^1\}$, covering T_i has already been in $l_2.V^1$. In this case, we simply increase l_2*'s* repeating count in all the {$l_2.v_i^1$} by 1.

Property 4.1. The information of a (*k*+1)-star calendar-map pattern, <*, *, *, *, *, …, *, *R*$_{(k+2)}$, …, *R*$_m$>, can be aggregated from the information of all *k*-star calendar-map patterns in <*, *, *, *, *, …, *, *R*$_{(k+1)}$, …, *R*$_m$), where each R_i, for $(k+1) \leq i \leq m$, is indicated by an integer.

Proof Given a time or space granularity $U_{(k+1)}$, assume that there are totally τ time intervals or locations in $U_{(k+1)}$. By definition, all time intervals or locations in a spatio-temporal granularity are indicated with a '*' symbol. For example, if $U_{(k+1)}$ is the time granularity "year", τ is equal to 12, because there are 12 months in a year, and a '*' is used to represent all months in a year in the calendar-map schema. Therefore, the aggregation of all k-star calendar patterns <*, *, *, *, *, …, *, *R*$_{(k+1)}$, …, *R*$_m$> in $U_{(k+1)}$ will be

$$\sum_{j=1}^{\tau} < *,*,*,*,*,...,*, j, R_{(k+2)},...,R_m >$$

$$= < *,*,*,*,*,...,*,*,R_{(k+2)},...,R_m >, , \tag{4.1}$$

which is the (*k*+1)-star calendar pattern, <*, *, *, *, *, …, *, *R*$_{(k+2)}$, …, *R*$_m$>. Thus, if the information of every *k*-star calendar pattern is known, the information of the (*k*+1)-star calendar pattern can also be derived.

In phase 2, Property (4.1) is firstly used to aggregate all other *k*-star candidate calendar-map patterns of itemsets in L_2 from 1-star candidate calendar-map patterns. For an itemset l_2 in L_2, its repeating counts in the 1-star candidate calendar-map patterns, $l_2.v_j^1$'s, have been derived from phase 1, and thus its repeating counts in 2-star candidate calendar-map patterns, $l_2.v_j^2$'s, can be easily obtained. Similarly, its repeating counts in 3-star candidate calendar-map patterns, $l_2.v_j^3$'s, can be aggregated from that in $l_2.v_j^2$*'s,* and so on. Instead of directly generating all *k*-star candidate calendar-map patlerns in phase l, our method generates and scans only 1-star candidate calendar-map patterns in the first scan of the database. Therefore, a smaller number of candidate calendar-map patterns are generated and counted in the process of scanning database. Once all candidate calendar-map patterns of itemsets in L_2 are derived, candidate itemsets C_k, for $k \geq 3$, along with their candidate calendar-map patterns can further be generated. Note that two kinds of candidates are generated in this phase, i.e., the candidates of frequent itemsets (candidate itemsets for short), and the candidates of frequent calendar-map patterns (candidate calendar-map patterns for short). The set of candidate k-itemsets, i.e., C_k of $k \geq 3$, are generated as follows:

$$C_k = \begin{cases} L_2 * L_2 & \text{if } k = 3 \\ C_{k-1} * C_{k-1} & \text{if } k > 3 \end{cases}, \tag{4.2}$$

where * is the JOINT operation given in [Agrawal and Srikant 1994].

Let the set of candidate calendar-map patterns, V_I, where I is a candidate itemset, be

$$V_I = \bigcup_{k \geq 1} \left(\bigcup_{j \geq 1} I.v_j^k \right), \tag{4.3}$$

Intuitively, the candidate calendar-map patterns of an itemset I_k^3 in C_3 can be derived by intersecting the candidate calendar-map patterns of itemsets which are the subsets of I_k^3 in L_2 as below:

$$V_{I_k^3} = \bigcap (\{V_{I_j^2} \mid \forall I_j^2 \subset I_k^3\}), \tag{4.4}$$

However, by utilizing I's repeating counts in V_I, $\forall I \in L_2$, we can find the minimal set of candidate calendar-map patterns for each itemset in C_3. From Equation (4.4), it is clear that the repeating count of a candidate calendar-map pattern of an itemset I_k^3 in C_3, $v_{I_k^3} \cdot count$, will never be larger than any $v_{I_j^2} \cdot count$, $\forall I_j^2 \subset I_k^3$. Therefore, the maximal value of $v_{I_k^3} \cdot count$, $\hat{v}_{I_k^3} \cdot count$, can be obtained by

$$\hat{v}_{I_k^3} \cdot count = \min(\{v_{I_j^2} \cdot count \mid \forall I_j^2 \subset I_k^3\}), \tag{4.5}$$

Furthermore, for each I_k^3, $v_{I_k^3}$ is removed if $\hat{v}_{I_k^3} \cdot count \leq (m \times |v_{I_k^3}|)$. Finally, the candidate calendar-map patterns of C_k, $k \geq 4$, are obtained by

$$V_{I_k^l} = \mathbf{I} \ (\{V_{I_j^{l-1}} \mid \forall I_j^{l-1} \subset I_k^l\}) \text{, for } l \geq 4. \tag{4.6}$$

In the final phase, to discover the frequent patterns from candidates, all candidate itemsets along with their candidate calendar-map patterns are counted in the database in one shot. Note that a candidate itemset is only scanned in the time intervals covered by its candidate calendar-map patterns. As a result, a frequent itemset passes the match ratio of a calendar-map pattern can be found. Spatio-temporal association rules are then obtained.

Our algorithm reduces the database access time by limiting the number of database scanning time to at most twice. It makes use of the concept hierarchy of spatial and temporal domain to reduce the number of candidates that need to be searched in scanning database. It is believed that the execution speed can be improved in this way.

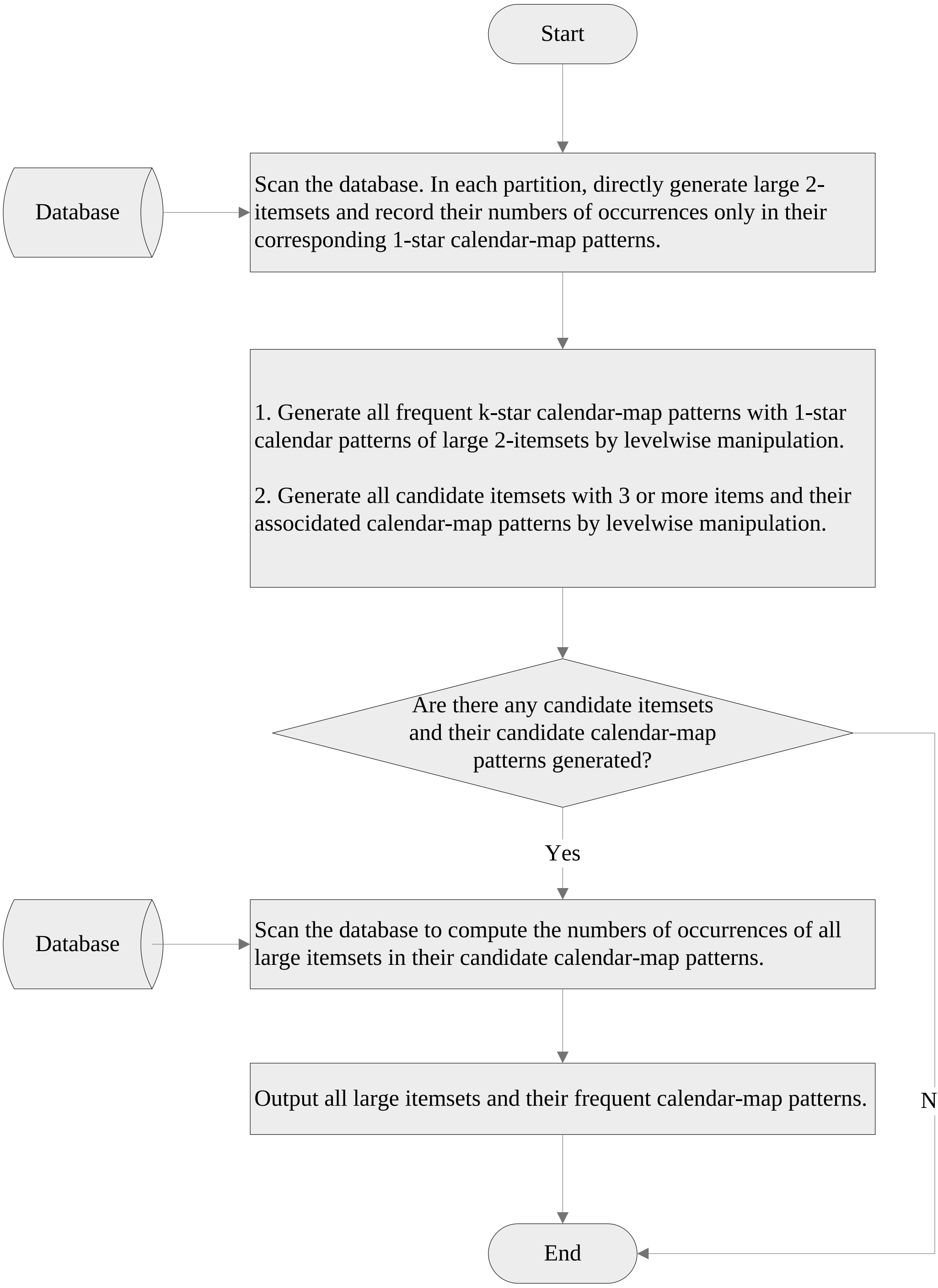

Start
Database
Scan the database. In each partition, directly generate large 2-itemsets and record their numbers of occurrences only in their corresponding 1-star calendar-map patterns.
1. Generate all frequent k-star calendar-map patterns with 1-star calendar patterns of large 2-itemsets by levelwise manipulation.
2. Generate all candidate itemsets with 3 or more items and their associdated calendar-map patterns by levelwise manipulation.
Are there any candidate itemsets and their candidate calendar-map patterns generated?
Yes
Database
Scan the database to compute the numbers of occurrences of all large itemsets in their candidate calendar-map patterns.
Output all large itemsets and their frequent calendar-map patterns.
No
End

Fig. 12. Our spatio-temporal association rule mining algorithm.

4.3 Mining Fuzzy Rules

This thesis provides a method, which allows users to define the membership functions of fuzzy calendar patterns for any interested time periods and fuzzy map patterns for any interested locations. With the membership functions, fuzzy spatio-temporal association rules that we defined can be mined.

First of all, users can design the membership functions of their interested fuzzy calendar patterns and map patterns in the calendar-map schema. For example, consider the calendar-map schema R=(year:{1996, ..., 2000}, month:{1, ..., 12}, day:{1, ..., 31}, province/state: {California, Florida}, city: {Los Angeles, Orlando}), which can be decomposed into the calendar schema R_c=(year:{1996, ..., 2000}, month:{1, ..., 12}, day:{1, ..., 31}) and the map schema R_m=(province: {California, Florida}, city: {Los Angeles, Orlando }). In the calendar schema R_c, the membership function of the fuzzy calendar pattern "close to (*, 11, 25)" is designed as in Figure 6, in which the range of time intervals that have the property "close to (*, 11, 25)" is defined to be two days before and after Novemer 25 of each year. In other words, only the dates defined by the calendar patterns (*, 11, 23), (*, 11, 24), (*, 11, 25), (*, 11, 26), (*, 11, 27) share the property ”close to (*, 11, 25)”. And according to the degree of closure to November 25 of any years, the dates will have different degrees of membership. In the map schema R_m, the membership function of the fuzzy map patterns "close to (California, Los Angeles)" and "Florida, Orlando" are designed as the "very close" line in Figure 7, in which two cities are regarded as being close to each other only if the distance between them is less than or equal to 2km. In other words, for the two cities in the calendar-map schema R, only (Florida, Orlando) is regarded as "close to (Florida, Orlando)" and has a degree of membership equal to 1. Any other map patterns will have a zero degree of membership to the fuzzy map pattern. Likewise, only (California, Los Angeles) is regarded as "close to (California, Los Angeles)" and has a degree of membership equal to 1. Any other map patterns will have a zero degree of membership.

Moving back to the whole calendar-map schema R, where $R = R_c + R_m$, a calendar-map pattern $P \in R$ is composed of a calendar pattern and a map pattern, that is $P = P_c + P_m$ where $P_c \in R_c$ and $P_m \in R_m$. We can define a fuzzy calendar-map pattern F_{cm} as a fuzzy calendar pattern F_c plus a fuzzy map pattern P_m. That is, $F_{cm} = F_c + F_m$. An example of a fuzzy calendar-map pattern in the calendar-map schema R is "close to (*, 11, 25, California, Los Angeles)". The degree of membership of a precise calendar-map pattern P' (P' can be a basic space-time interval or a n-star calendar-map pattern. A precise calendar-map pattern means a non-fuzzy calendar-map pattern.), which is composed of a calendar pattern P_c' and a map pattern P_m', to a fuzzy calendar-map pattern F_{cm} is formulated as:

$$\begin{aligned}\textit{Degree of Membership } F_{cm}\,(P') = \; & \textit{Degree of Membership } F_c(P_c') \times \\ & \textit{Degree of Membership } F_m(P_m'),\end{aligned} \tag{4.7}$$

where *P' = P_c' + P_m'*, *P_c'* is a non-fuzzy calendar pattern and *P_m'* is a non-fuzzy map pattern. For example, according to Figure 6, the degree of membership of the calendar pattern (*, 11, 24) to the fuzzy calendar pattern ”close to (*, 11, 25)” is equal to 0.6. And according to Figure 9, the degree of membership of the map pattern (California, Los Angeles) to the fuzzy map pattern “close to (California, Los Angeles)” is equal to 1. Hence, the degree of membership of the calendar-map pattern (*, 11, 24, California, Los Angeles) to the fuzzy calendar-map pattern "close to (*, 11, 25, California, Los Angeles)" is equal to 0.6 × 1 = 0.6.

We make use of an algorithmic flow similar to that used to find spatio-temporal association rules. The method scans the database at most twice. Before scanning the database, we first find all precise calendar-map patterns with degrees of membership, according to the membership functions that users defined for their interested fuzzy calendar patterns and fuzzy map patterns. Then, when scanning the database, only those precise calendar-map patterns with degrees of membership have to be manipulated.

In each scanning of the database, only those precise calendar-map patterns with non-zero degrees of membership to the interested fuzzy calendar-map patterns have to be considered. In the first scanning of database, our method finds all frequent 2-itemsets in these precise calendar-map patterns. Then we determine if these precise calendar-map patterns are asynchronous calendar-map patterns or not. An asynchronous calendar-map pattern is a calendar-map pattern with a non-zero and non-one degree of membership *m*, i.e. $0 < m < 1$. The degree of membership to the fuzzy calendar-map pattern of these frequent itemsets will then be accumulated to *FD*, for asynchronous calendar-map patterns, or *PD*, for non-asynchronous calendar-map patterns. In other words, given a frequent itemset in a precise calendar-map pattern, its count will be added a value, which is equal to the degree of membership of this calendar-map pattern to the interested fuzzy calendar-map pattern, composed of a fuzzy calendar pattern and a fuzzy map pattern.

After the first scanning of database, we individually divide the *PD* and *FD* of frequent itemsets by the sums of their corresponding degrees of membership. Then, the Equation (3.10) is used to compute the fuzzy match ratio, with which we can filter out frequent 2-itemsets fulfilling the user-defined fuzzy match ratio threshold. In this way, frequent 2-itemsets, which appears at around the same place periodically but asynchronously, can be found.

In the second scanning of database, we apply Equation (4.2), an Apriori’s way to generate candidate itemsets with 3 or more items. Like in the first scanning, we only have to consider those precise calendar-map patterns with non-zero degrees of membership and accumulate the degrees of membership to the *PD* and *FD* of itemsets.

At last, we compute the fuzzy match ratio with the *PD* and *FD* of itemsets by applying Equation (3.10). Again, we can then filter out frequent itemsets with at least the user-defined fuzzy match ratio threshold. These frequent itemset together with the frequent 2-itemsets found before will be frequent itemsets in the fuzzy calendar-map patterns, which users are querying for.

4.4 Mining Meta-Rules in Rule Sets

Following the definitions of regular, differential, and change meta-rules given in Chapter 3, we present how our proposed algorithms can be used to mine such meta-rules from rule sets in this section. Specifically, given a collection of data sets, D_1, ..., D_m, a set of rules, R_j, $j \in \{1, ..., m\}$, are discovered from each data set. Our task here is to mine regular, differential, and change meta-rules from R_1, ..., R_m.

4.4.1 Mining Regularities and Differences

Given R_1, ..., R_m, X is a conjunction of discrete conditions such that $condition(X) \subseteq \bigcup_{j=1}^{n} condition(R_j)$. X is supported by a set of rules:

$$\mathcal{R}(X) = \{r \mid r \in R_1 \cup ... \cup R_m, (X) \in condition(r)\},$$

where $condition(r)$ denotes the set of conditions in r and $condition(R_j) = \bigcup_{r \in R_j} condition(r)$ (defined in Chapter 3).

The support of the conjunction of conditions X is then given by:

$$sup(X) = \frac{|\mathcal{R}(X)|}{|R_1 \cup ... \cup R_m|}, \qquad (4.8)$$

Similarly, an association, $X \rightarrow Y$, where $X, Y \in condition(R_1) \cup ... \cup condition(R_m)$, is supported by a set of rules:

$$\mathcal{R}(X \to Y) = \{r \mid r \in R_1 \cup \ldots \cup R_m, X, Y \in condition(r)\}. \tag{4.9}$$

The support and the confidence of the association $X \to Y$ are then given by:

$$sup(X \to Y) = \frac{|\mathcal{R}(X \to Y)|}{|R_1 \cup \ldots \cup R_m|}, \tag{4.10}$$

and

$$conf(X \to Y) = \frac{sup(X \to Y)}{sup(X)}, \tag{4.11}$$

respectively.

Intuitively, $sup(X)$ and $sup(Y)$ can be considered as being the probability that a rule has the conjunctions of conditions X and Y, respectively. Similarly, $sup(X \to Y)$ can be considered as being the probability that a rule has both X and Y. If X and Y are independent of each other, then $sup(X \to Y) = sup(X) \times sup(Y)$. Hence $sup(X) \times sup(Y) \times | R_1 \cup \ldots \cup R_m|$ yields the expected value of $|\mathcal{R}(X \to Y)|$ (= $sup(X \to Y) \times |R_1 \cup \ldots \cup R_m|$). If $|\mathcal{R}(X \to Y)|$ is *significantly larger* than its expected value, it is sufficiently large. The regular meta-rule $X \Rightarrow Y$ can therefore be formed Definition (3.10). On the other hand, if $|\mathcal{R}(X \to Y)|$ is *significantly smaller* than its expected value, it is sufficiently small. Consequently, the differential meta-rule $X \Rightarrow Y$ can be formed Definition (3.11).

The difference between $sup(X \to Y)$ and $sup(X) \times sup(Y)$ and hence the difference between $|\mathcal{R}(X \to Y)|$ and its expected value can be objectively evaluated in terms of the adjusted residual [Agresti 1990], $d(X \to Y)$, given by the formula:

$$d(X \to Y) = \frac{sup(X \to Y) - sup(X) \times sup(Y)}{\sqrt{sup(X) \times sup(Y) \times [1 - sup(X)] \times [1 - sup(Y)]}}, \tag{4.12}$$

Since the adjusted residual has a normal distribution [Agresti 1990], we can conclude that $sup(X \to Y)$ is significantly larger than $sup(X) \times sup(Y)$ if $d(X \to Y) > 1.96$ (the 95th percentile of the normal distribution). In other words, $|\mathcal{R}(X \to Y)|$ is significantly larger than its expected value and it is therefore sufficiently large. On the other hand, if $d(X \to Y)$

< –1.96, we can conclude that $sup(X \rightarrow Y)$ is significantly smaller than $sup(X) \times sup(Y)$. In other words, $|\mathcal{R}(X \rightarrow Y)|$ is significantly smaller than its expected value and it is therefore sufficiently small.

It is important to note that we need to take care of not only the criterion $d(X \rightarrow Y) > 1.96$, but also $d(X \rightarrow Y) < -1.96$ for meta-mining. The former is to test for the regularities in common in the rule sets (i.e., regular meta-rules), whereas the latter is to test for the distinguishing relationships in only a few rule sets (i.e., differential meta-rules). The adjusted residual can be used as a measure to identify whether the support of an association hidden in the rule sets is sufficiently large or sufficiently small in order to identify regular or differential meta-rules, respectively.

4.4.2 Mining Changes

In this section, we will propose a rule matching method that can detect various types of change meta-rules. The input parameters of the rule matching method are two sets of rules R^t and R^{t+k} discovered at time t and $t+k$ respectively, as well as a user-defined threshold called the Rule Matching Threshold (*RMT*). The meta-mining process is composed of three steps:

Step 1: Calculate the maximum similarity value for each rule discovered at time t and $t+k$.

Step 2: For each rule $r_i^t \in R^t$, calculate the difference measures between r_i^t and $r_j^{t+k} \in R^{t+k}$.

Step 3: Classify the type of changes for the rules using the maximum similarity value and the difference measures.

4.4.2.1 Step 1: Calculation of Maximum Similarity Values

For the explanation of each step, some notations are briefly defined.

δ_{ij} : Difference measure.
Degree of difference between r_i^t and r_j^{t+k} $(-1 \le \delta_{ij} \le 1,\ 0 \le |\delta_{ij}| \le 1)$

s_{ij} : Similarities measure. Degree of similarity between r_i^t and r_j^{t+k} $(0 \le s_{ij} \le 1)$

l_{ij} : Degree of attribute match of the antecedent parts $l_{ij} = |A_{ij}| / \max(|X_i^t|, |X_j^{t+k}|)$

c_{ij} : Degree of attribute match of the consequent parts $c_{ij} = |B_{ij}| / \max(|Y_i^t|, |Y_j^{t+k}|)$

$\left|A_{ij}\right|$: Number of attributes common to both antecedent parts of r_i^t and r_j^{t+k}

$\left|X_i^t\right|$: Number of attributes in the antecedent parts of r_i^t

$\left|X_j^{t+k}\right|$: Number of attributes in the antecedent parts of r_j^{t+k}

$\left|B_{ij}\right|$: Number of attributes common to both consequent parts of r_i^t and r_j^{t+k}

$\left|Y_i^t\right|$: Number of attributes in the consequent parts of r_i^t

$\left|Y_j^{t+k}\right|$: Number of attributes in the consequent parts of r_j^{t+k}

x_{ijk} : Degree of value match of the k^{th} matching attribute in A_{ij}

y_{ij} : Degree of value match of the k^{th} matching attribute in B_{ij}

$$x_{ijk} = \begin{cases} 1, \text{if same value} \\ 0, \text{otherwise} \end{cases}, \quad y_{ij} = \begin{cases} 1, \text{if same value} \\ 0, \text{otherwise} \end{cases}$$

Now we provide similarity measure as follows:

$$s_{ij} = \begin{cases} \dfrac{l_{ij} \times \sum\limits_{k \in A_{ij}} x_{ijk} \times c_{ij} \times \sum\limits_{k \in B_{ij}} y_{ijk}}{\left|A_{ij}\right| \times \left|B_{ij}\right|} & , \text{if } \left|A_{ij}\right| \neq 0 \text{ and} \left|B_{ij}\right| \neq 0 \\ 0 & , \text{if } \left|A_{ij}\right| = 0 \text{ or} \left|B_{ij}\right| = 0 \end{cases}, \tag{4.13}$$

In the formula to compute s_{ij}, $l_{ij} \times \sum\limits_{k \in A_{ij}} x_{ijk} / \left|A_{ij}\right|$ represents a similarity of antecedent part, and $c_{ij} \times \sum\limits_{k \in B_{ij}} y_{ijk} / \left|B_{ij}\right|$ represents a similarity of consequent part between r_i^t and r_j^{t+k}. If the antecedent and consequent parts between r_i^t and r_j^{t+k} and the same, then the degree of similarity becomes 1. The similarity measure can take any value between 0 and 1. To detect added and perished rules, the maximum similarity value is provided as follows:

$$s_i = \max(s_{i1}, s_{i2}, \ldots, s_{i\left|R_{t+k}\right|}) \text{; Maximum Similarity Value of } r_i^t, \tag{4.14}$$

$$s_j = \max(s_{1j}, s_{2j}, \ldots, s_{\left|R_t\right|j}) \text{; Maximum Similarity Value of } r_j^{t+k}, \tag{4.15}$$

The maximum similarity value indicates whether the rule is added or perished. If s_i < RMT, the r_i^t is recognized as a perished rule. If $s_j < RMT$, the the rule r_j^{t+k} becomes an added rule.

Example 4.1. Assume the following rules are generated from each dataset D^t and D^{t+k}.

r_1^t : Income = High $\Rightarrow$ Sales = High

r_2^t : Age = High, Preference = Price $\Rightarrow$ Sales = High

r_1^{t+k} : Income = High $\Rightarrow$ Sales = High

r_2^{t+k} : Age = High $\Rightarrow$ Sales = High

r_3^{t+k} : Income = High, Preference = Price $\Rightarrow$ Sales = Low

We can compute the similarity measure between r_2^t, r_2^{t+k} and the maximum similarity value of r_2^t as follows.

$$s_{22} = \frac{\frac{1}{2} \times 1 \times 1 \times 1}{1} = 0.5, \; s_2^t = \max(0, 0.5, 0) = 0.5$$

In the same manner, we can compute the maximum similarity value of each rule.

$s_1^t = \max(1, 0, 0) = 1$ $\quad$ $s_2^t = \max(0, 0.5, 0) = 0.5$

$s_1^{t+k} = \max(1, 0) = 1$ $\quad$ $s_2^{t+k} = \max(0, 0.5) = 0.5$ $\quad$ $s_3^{t+k} = \max(0, 0) = 0$

If we specify *RMT* to be 0.4, then we can conclude that only is an added rule.

4.4.2.2 Calculation of Difference Measures

As we can see from Example (4.1), the maximum similarity value in step 1 is used to discover added or perished rules. The purpose of step 2 is to detect unexpected changes and emerging patterns. To dectect unexpected change, a difference measure is provided as follows:

$$\delta_{ij} = \begin{cases} \dfrac{l_{ij} \times \sum_{k \in A_{ij}} x_{ijk}}{|A_{ij}|} - \dfrac{\sum_{k \in B_{ij}} y_{ijk}}{|B_{ij}|} & , \text{if } |A_{ij}| \neq 0, c_{ij} = 1 \\ -\dfrac{\sum_{k \in B_{ij}} y_{ijk}}{|B_{ij}|} & , \text{if } |A_{ij}| = 0, c_{ij} = 1 \\ \infty & , \text{if } c_{ij} \neq 1 \end{cases}, \tag{4.16}$$

As defined above in the problem definition section, the association rule r_j^{t+k} discovered at time *t*+*k* is an unexpected change with respect to the association rule r_i^t discovered at time *t* if the antecedent parts of the two association rules are similar but their consequent parts are quite different. Based on this definition of unexpected changes, we propose a way to judge whether the rule r_j^{t+k} is an unexpected change with respect to r_i^t with the difference measure δ_{ij}. The conclusions that can be drawn from the value of δ_{ij} can be classified into four cases:

1. If $\delta_{ij} > 0$, then rule r_j^{t+k} is an unexpected consequent change with respect to r_i^t.
2. If $\delta_{ij} < 0$, then rule r_j^{t+k} is an unexpected antecedent change with respect to r_i^t or simply not an unexpected change.
3. If $\delta_{ij} = 0$, then the two rules r_i^t and r_j^{t+k} are either completely the same or completely different. Therefore, some additional measures such as l_{ij}, $\sum_{k \in B_{ij}} y_{ijk} / |B_{ij}|$ and etc. need to be provided to judge between the two cases. For example, If $l_{ij} = 1$ and $\sum_{k \in B_{ij}} y_{ijk} / |B_{ij}| = 1$, then the two rules are same.
4. If $\delta_{ij} = \infty$, this means the attributes of the consequent parts of the two rules are different, i.e. $c_{ij} \neq 1$. In this case, the two rules r_i^t and r_j^{t+k} are completely different.

Hence, r_j^{t+k} is a useful pattern with respect to r_i^t only when $\delta_{ij} \geq 0$

4.4.2.3 Classification of Change Types

In Step 3, we are to classify rules into three types of change. In addition to difference measures, we need some other computation in the classification of change meta-rules.

Although when $\delta_{ij} > 0$, r_j^{t+k} is judged to be an unexpected change with regard to r_i^t, we cannot conclude directly that it is an unexpected change. This is because there are two cases that $\delta_{ij} > 0$ but r_j^{t+k} should not be classified as an unexpected change with regard to r_i^t.

Firstly, if there exists an association rule r_m^t, where $m \neq i$ and $r_m^t \in R^t$, which has the same structure as that of r_j^{t+k}, i.e. $r_j^{t+k} \equiv r_m^t$, then r_j^{t+k} should be classified as an emerging pattern with respect to r_m^t instead if the interestingness of the two rules is different from each other, i.e. *interestingness*(r_i^t) ≠ *interestingness*(r_j^{t+k}).

Secondly, if there exists an association rule r_n^{t+k}, where $n \neq j$ and $r_n^{t+k} \in R^{t+k}$, which has the same structure as that of r_i^t, i.e. $r_i^t \equiv r_n^{t+k}$, then r_j^{t+k} should be regarded as a different rule from r_i^t instead.

As we cannot make a conclusion based on δ_{ij} alone whether r_j^{t+k} is an unexpected change or an emerging pattern, we propose a new measure called Adjusted Difference Measure, which is formulated as:

$$\delta_{ij}^{'} = \left|\delta_{ij}\right| - k_{ij} \text{ where } k_{ij} = \begin{cases} 1, \text{if } \max(s_i^t, s_j^{t+k}) = 1 \\ 0, \text{otherwise} \end{cases}, \tag{4.17}$$

The fact that s_i^t is equal to 1 means that an equivalent rule of r_i^t exists in the ruleset R^{t+k}. And the fact that s_j^{t+k} is equal to 1 means that an equivalent rule of r_j^{t+k} exists in the ruleset R^t. In both cases, the minus k_{ij} prevents r_j^{t+k} from being classified as an unexpected change. If $\delta_{ij}^{'}$ turns out to be greater than the pre-specified *RMT*, then the rule r_j^{t+k} will finally be classified as an unexpected change with respect to r_i^t.

Example 4.2. Consider the five association rules:

r_1^t : Income = High, Preference = Price ⇒ Sales = Low

r_2^t : Age = High, Preference = Price ⇒ Sales = High

r_1^{t+k} : Income = High ⇒ Sales = High

r_2^{t+k} : Age = High ⇒ Sales = High

r_3^{t+k} : Income = High, Preference = Price ⇒ Sales = Low

With the association ruleset, we can compute the difference measure and adjusted difference measure between r_2^t and r_3^{t+k} as follows:

$$\delta_{23} = 0.5;$$
$$\delta'_{23} = 0.5 - 1 = -0.5$$

If we specify that *RMT* is equal to 0.4, we cannot conclude that r_3^{t+k} is an unexpected consequent change with respect to r_2^t because r_3^{t+k} share the same rule structure with r_1^t. Therefore, r_3^{t+k} is an emerging pattern of r_1^t but not an unexpected consequent change with respect to r_2^t. Table 4 summaries how to make use of the values of the measures to classify the type of change meta-rules.

Table 4. Value of measure for each type of change.

Type of Change	**Value of measure to classify**
Emerging Pattern	$\delta_{ij} = 0$ and ($\sum_{k \in A_{uh}} x_{ijk} > 0$ or $\sum_{k \in B_{uh}} y_{ijk} > 0$ > 0 or $l_{ij} > 0$) and $interestingness(r_i^t) \neq interestingness(r_j^{t+k})$
Unexpected Change	$\delta_{ij} > 0$ and $\delta'_{ij} \geq RMT$
Added Rules (Perished Rule)	$s_j < RMT$ ($s_i < RMT$)

4.4.2.4 Evaluation of the Degree of Changes

It will be useful to rank change meta-rules by their degrees of changes because the larger the degrees of changes, the more significant the change meta-rules. Hence, degrees of changes and significance levels of change meta-rules refer to the same thing. We will explain how to evaluate degrees of changes for each class of change rules in this section.

Example 4.3. This example presents the need of additional measures to judge the significance of change meta-rules.

r_i^t : Income = High, Age = High ⇒ Model = Large

r_j^{t+k} : Preference = Price, Age = High ⇒ Model = Small ($\delta_{ij} = \delta'_{ij} = 0.5$)

If *RMT* is equal to 0.4, then the rule r_j^{t+k} will be classified as an unexpected consequent change with respect to r_i^t. However, it is difficult for us to judge whether the change from r_i^t to r_j^{t+k} is significant. The reasons are that 1) the antecedent parts of the two rules are not the same and 2) we do not quantatively know how much r_j^{t+k} has been changed from r_i^t. Therefore, additional measures and judgement are required to decide whether the degree of change from one rule to another is significant.

For this purpose, we adapt the concept of *Unexpectedness* from the study of Padmanabhan and Tuzhilin (1999) to measure the significance levels of unexpected consequent changes. They define unexpectedness using the concept of exception rules [Suzuki 1997].

Definition 4.1. Unexpectedness.

If the association rule $A \Rightarrow B$ is unexpected with respect to the belief $X \Rightarrow Y$, then the following constraint must hold:

1. $B \cap Y = \varnothing$
2. The rule X and $A \Rightarrow B$ holds

To measure the degree of change of the unexpected consequent change r_j^{t+k} with respect to r_i^t, r_i^t is assumed to be a belief or existing knowledge. According to Padmanabhan and Tuzhilin, every unexpected consequent change satisfied Condition (1) of Definition (4.1) is due to Condition (2). Furthermore, the support of the conjunction rule of r_i^t and r_j^{t+k} should be evaluated to check whether Condition (2) of Definition (4.1) holds or not.

Example 4.4. The conjunction rule of the r_i^t and r_j^{t+k} in Example (4.2) is as follows.

$r_{i \cap j}^{t+k}$: Income = High, Age = High, Preference = Price $\Rightarrow$ Model = Small

If the conjunction rule $r_{i \cap j}^{t+k}$ has "large support value" in the dataset at time *t+k*, i.e. D^{t+k}, our conclusion that r_j^{t+k} is an unexpected consequent change with respect to r_i^t is strengthened according to condition 2 of Definition (4.1). Therefore, supports of conjunction rules is a factor to determine degrees of changes for unexpected consequent changes.

However, what is "large support value"? We say that if the support value of the conjunction rule $r_{i \cap j}^{t+k}$ is relatively small compared with that of r_j^{t+k} , then we cannot conclude that r_j^{t+k} is a significant unexpected consequent change with respect to r_i^t . Therefore, a large support value of $r_{i \cap j}^{t+k}$ should be larger than that of r_j^{t+k} . Therefore, the measure of degrees of changes for unexpected consequent changes should take the support values of both r_j^{t+k} and $r_{i \cap j}^{t+k}$ into consideration. To sum up, the measure for degrees of unexpected consequent changes can be formulated as:

$$\alpha_{ij} = \frac{sup^{t+k}(r_{i \cap j})}{sup^{t+k}(r_j)}$$

In the case of emerging patterns, it is a lot simpler to evaluate their significance levels than those of unexpected changes. The increase or decrease rate of support values is used as the measure of significance levels for emerging patterns. To evaluate degrees of changes for added or perished rules, the supports and maximum similarities of the rules are considered. The value of maximum similarity of a rule represents how similar the rule to any rules in another ruleset is. If there is a situation that the support values of two added rules are same, we will intuitively put the rule with a smaller value of maximum similarity in a more important position. The measures of degrees of changes for different classes of change meta-rules are finally summarized as follows. Based on the degrees of changes, we can rank the mined change meta-rules.

$$\alpha_{ij} = \begin{cases} \dfrac{sup^{t+k}(r_i) - sup^{t}(r_i)}{sup^{t}(r_i)} & \text{, for emerging patterns} \\ \dfrac{sup^{t+k}(r_{i \cap j})}{sup^{t+k}(r_j)} & \text{, for unexpected changes} \\ (1 - s_i^{t}) \times sup^{t}(r_i) & \text{, for perished rules} \\ (1 - s_j^{t+k}) \times sup^{t+k}(r_j) & \text{, for added rules} \end{cases} , \tag{4.20}$$

Chapter 5
Experiments

In this section, we perform a simulation study to empirically compare the runtime and number of generated patterns of the proposed methods. Since the application scenarios of the proposed methods are entirely different from those of traditional association rule mining, we will not compare our proposed methods with traditional association rule mining methods in this section. Instead, the main objective of the simulation study is to measure and identify the performance of the proposed methods in finding important spatio-temporal patterns and their runtime in a multi-location environment under different pre-conditions. The proposed algorithms are implemented and tested with the following hardware and software configuration:

(1) Operation system: Windows XP
(2) Hardware: Pentium 3.0G processor, 1024M main memory
(3) Tool: Java 2 Standard Edition Version 1.4.2

5.1 Synthetic Data Sets

In the experiments, we randomly generate synthetic transactional data sets by applying the data generation algorithm proposed by [Chen, Tang, Shen and Hu 2005] with some additional modification. The main process of generating the synthetic transactional data sets will be introduced in this section.

The factors determining the generated data sets are listed in Table 5. In addition, we will generate the time concept hierarchy and place concept hierarchy information from a Poisson distribution with mean B_d and a decided value H_t and H_p, for each generated dataset.

To generate the area sizes of locations, we use two parameters, S_u and S_l, to represent the largest and smallest area sizes, respectively, and the area size of a location i for $1 \leq i \leq q$, denoted by S_i, is generated by a uniform distribution between S_u and S_i. We assume that the total number of transactions and the number of items are dependent on how large the area of a location. In addition, we also allow the locations to have different item replacement (turnover) ratios. This is intuitve because, for example, the list of products sold in a supermarket of WalMart may change over time. In the experiments, these relationships are estimated by generating *m* random numbers for the store *i* from a Poisson distribution with mean S_i. We use the j^{th} number, denoted by W_{ij}, as the weight of location *i* in period *j*. Let D_{ij} denote the number of transactions of location *i* in period *j*. The total number of transactions, *D*, is distributed to the location *i*, and period *j* is determined by:

$$D_{ij} = \frac{D}{\sum_{m=0}^{P}\sum_{n=0}^{T} W_{mn}} \times W_{ij}$$

Furthermore, we assume the number of items in a location is proportional to the square root of its area size. Thus, let $IS_i = \sqrt{S_i}$ for I = 1, 2, …, q. Then, the number of items in location *i*, denoted by N_i, is determined by the following formula:

$$N_i = \frac{r}{Max(IS_i)} \times IS_i$$

Note that the list of items available in a location may change over time, although N_i is kept the same in all periods. Since the parameter I_d is the proportion of items that will be replaced in every period, location *i* replaces $N_i \times I_d$ items in each period. Furthermore, we follow the method used by [Agrawal and Srikant 2004] to generate F_d maximum potentially frequent itemsets with an average length of F_l within each space-time interval.

Finally, we generate all the transactions in the data sets. To generate the transactions for location *i* in period *j*, we generate D_{ij} transactions from a Poisson distribution with average length of transactions *L* and a series of maximum potentially frequent itemsets with the parameters F_d and F_l. If an itemset generated to a transaction from the process has some items not available at store *i* in period *j*, we remove these items, and then repetitively add items into the transaction until we have reached the intended size of the transaction. If the last itemset exceeds the boundary of this transaction, we remove the part that excees the boundary. When adding an itemset to a transaction, we use a corruption level, $c = 0.7$, to simulate the phenomenon that all the items in a frequent itemset do not always appear together. Information on how the corruption level affects the procedure of generating items for a transaction is introduced by [Agrawal and Srikant 2004].

Table 5. Parameters used in simulation.

Parameters	**Description**	**Default**
D	Number of transactions	6M
q	Number of locations	100
m	Number of time periods	60
L	Average length of transactions	10
r	Number of items	1000
F_l	Average length of maximum potentially frequent itemsets	5
F_d	Number of maximum potentially frequent itemsets	80
I_d	Replace rates of items	0.05
S_u, S_l	The Maximum and minimum area sizes of locations	10, 1
H_t	Level of time concept hierarchy	3
H_p	Level of place concept hierarchy	3
B_d	Average branch degree	10

We perform Experiment 1 to 7, for testing our spatio-temporal association rule mining algorithms, on synthetic datasets, changing a different parameter in each dataset. All the parameters other than the controlled variable are set to their default values. A real-world data set has been used in testing meta-rule mining in the last experiment. Why synthetic data sets have been used in most of the experiments is that synthetic data can be generated to have a lot of different properties. Hence, they are used to test the performance of the algorithms with different datasets.

5.2 Experiment 1

We use four different data types, (L=10;F_l=3;D=6M), (L=10;F_l=4;D=6M), (L=10;F_l=5;D=6M), and (L=10;F_l=6;D=6M), to generate around 1000 transactions for each basic space-time intervals. We also apply different support thresholds, 0.06, 0.07, 0.08, 0.09 and 0.1, to investigate how the performance of the two methods, i.e. our proposed new method and Spatio-temporal Apriori, will vary with different support thresholds. The match ratio threshold is defined to be 0.8 in this experiment.

Table 6 contains the numbers of frequent calendar-map patterns and frequent patterns, as well as the maximum length of the frequent itemsets, found in the four sets of data with the two methods and the five different support thresholds. Figure 13 shows a plot of runtime time of the two methods with the 4 datasets against the different support thresholds.

Table 6. Experiment 1: The numbers of candidate and frequent calendar-map patterns as well as the maximum length of generated frequent itemsets by Spatio-temporal Apriori and our method.

			Support threshold				
			0.06	0.07	0.08	0.09	0.1
L=10;F_l=3;D=6M	Candidate Calendar-map patterns of	Spatio-temporal Apriori	65511	40023	28333	22485	15878
		Our Method	72992	46105	29990	24935	17839
	Discovered frequent calendar-map patterns		64844	39596	28071	22206	14727
	Maximal length of frequent itemsets		7	6	6	5	5
L=10;F_l=4;D=6M	Candidate calendar-map patterns of	Spatio-temporal Apriori	205886	133932	73784	48726	29277
		Our Method	212943	174203	89915	55214	34681
	Discovered frequent calendar-map patterns		198454	128495	71432	46618	27756
	Maximal length of frequent itemsets		8	8	8	6	6
L=10;F_l=5;D=6M	Candidate calendar-map patterns of	Spatio-temporal Apriori	56017	45429	34215	18191	6146
		Our Method	55465	46083	36561	25585	9430
	Discovered frequent calendar-map patterns		52942	43431	31732	16540	5412
	Maximal length of frequent itemsets		6	6	6	6	6
L=10;F_l=6;D=6M	Candidate calendar-map patterns of	Spatio-temporal Apriori	180697	122007	48047	9787	1313
		Our Method	192262	138016	83417	21123	2407
	Discovered frequent calendar-map patterns		162640	105666	42128	8968	1068
	Maximal length of frequent itemsets		9	8	8	6	4

In the analysis of Table 6 and Figure 13, we discover that when the maximum length of frequent itemsets becomes larger, if the number of candidate calendar-map patterns increases, then the runtime of Spatio-temporal Apriori will increase tremendously. When the maximum length of frequent itemsets becomes smaller, if the number of candidate calendar-map patterns increases, then the runtime of Spatio-temporal Apriori will increase but the increase tends to be slower. This can be observed by comparing data from (*L*=10;F_l=4;*D*=6M) and (*L*=10;F_l=5;*D*=6M).

Our explanation is that when the maximum length of frequent itemsets becomes larger, Spatio-temporal Apriori has to scan the database for more times. Even worst, when there are more candidate calendar-map patterns that have to be checked in each scanning of database, the adverse influence on execution runtime will become more apparent. Our method, on the other hand, will scan the database at most twice, so the impact of the maximum length of frequent items is less apparent. The impact of an increase in the number of candidate calendar-map patterns is also not as vigorous as that in applying Spatio-temporal Apriori. Hence, our method on these grounds has better performance in term of execution runtime compared with Spatio-temporal Apriori.

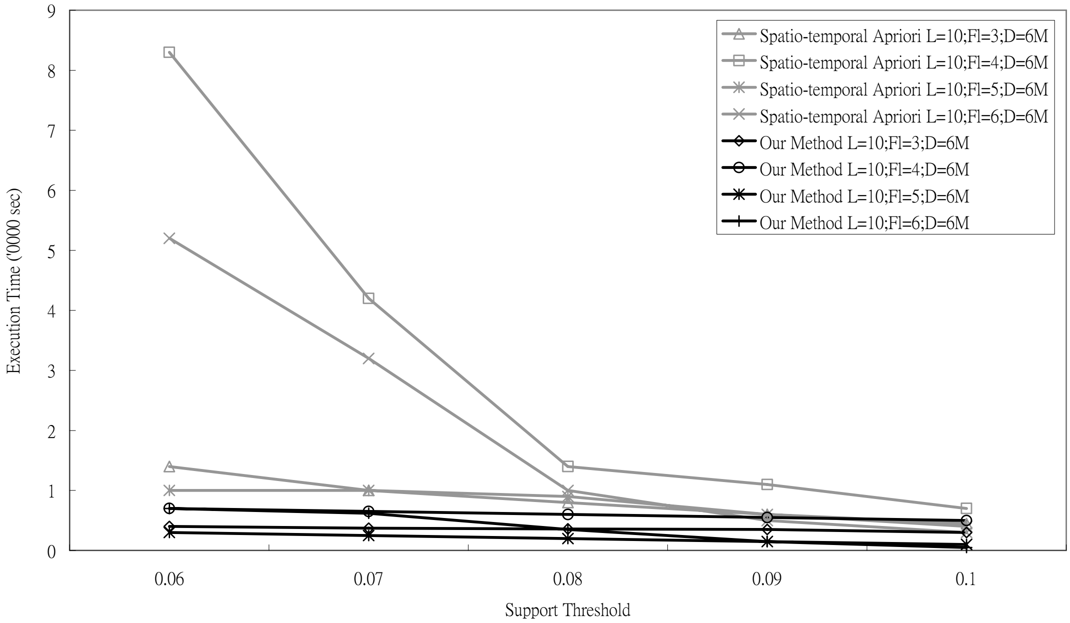


Fig. 13. Experiment 1: Comparison of execution time between Spatio-temporal Apriori and our method.

5.3 Experiment 2

We use four data sets of different sizes in this experiment. They are (*L*=10;*F_l*=4;*D*=4M), (*L*=10;*F_l*=4;*D*=6M), (*L*=10;*F_l*=4;*D*=8M) and (*L*=10;*F_l*=4;*D*=10M). On average, there are 1000 transactions in each basic space-time interval. We apply different support thresholds, 0.06, 0.07, 0.08, 0.09 and 0.1 respectively, to investigate the differences of the two methods in runtime with data sets of different sizes. The match ratio threshold defined in this experiment is 0.8.

Figure 14 is, under the data sets of four different sizes, how the performance of the two methods will change with different support thresholds. The dotted lines are the runtime of Spatio-temporal Apriori whereas the solid line is that of our proposed new method. Figure 15 is, under the different support thresholds, the number of candidate itemsets and the discovered frequent itemsets from the four data sets with the two methods. Figure 16 is, under the different support thresholds, the number of candidate calendar-map patterns and the discovered frequent calendar-map patterns from the four data sets with the two methods. Before the experiment, we know that the number of scanning times in Spatio-temporal Apriori depends on the allowed maximum length of frequent itemsets. From the figures, we can see that when the sizes of data sets become larger, Spatio-temporal Apriori, which relatively needs more number of scanning times, has runtime increasing in phase with the sizes of the data sets. What is more, the phenonemun is more apparent when the support threshold becomes smaller. This is because, with a smaller support threshold, there are more candidates that need to be checked in each scanning. From Figure 14, we can deduce that our method outperforms Spatio-temporal Apriori in term of execution runtime with all the data sets of different sizes.

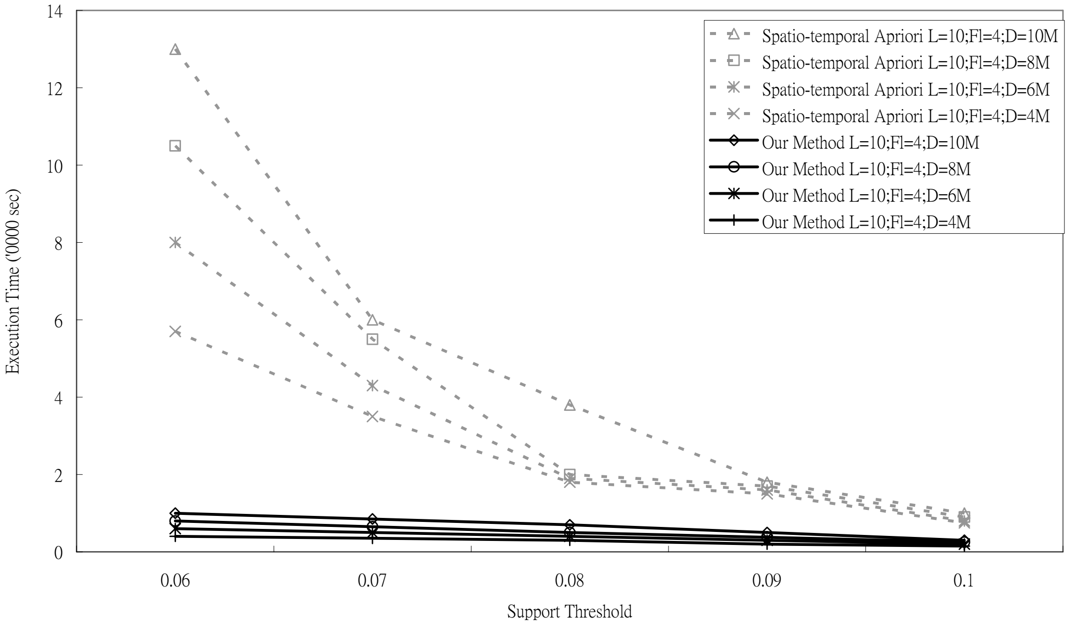


Fig. 14. Experiment 2: Comparison of execution time between Spatio-temporal Apriori and our method.

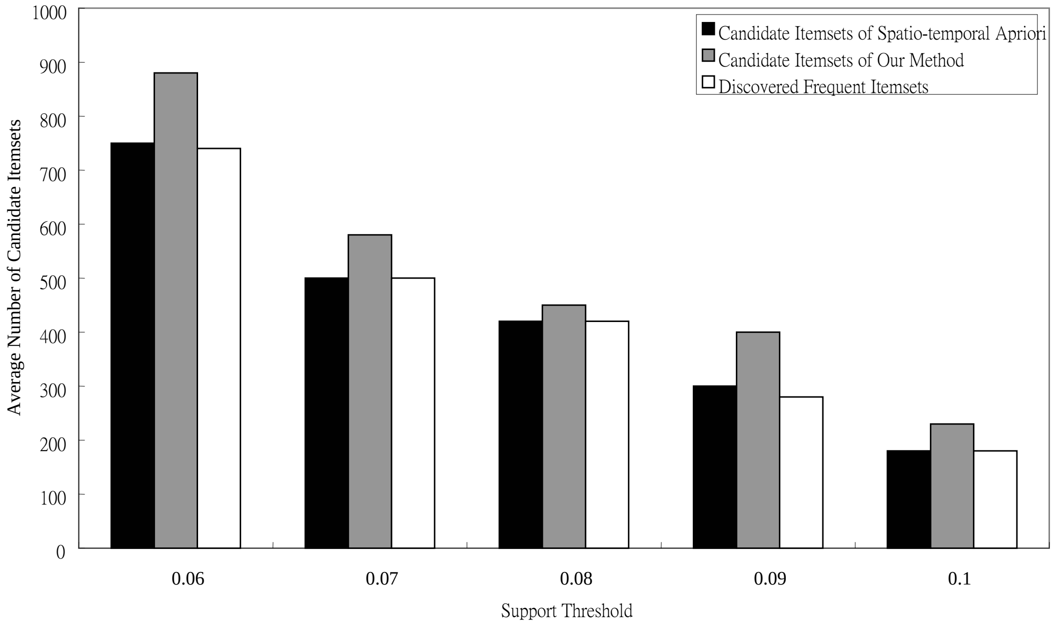


Fig. 15. Experiment 2: Average number of candidate itemsets and frequent itemsets found by Spatio-temporal Apriori and our method.

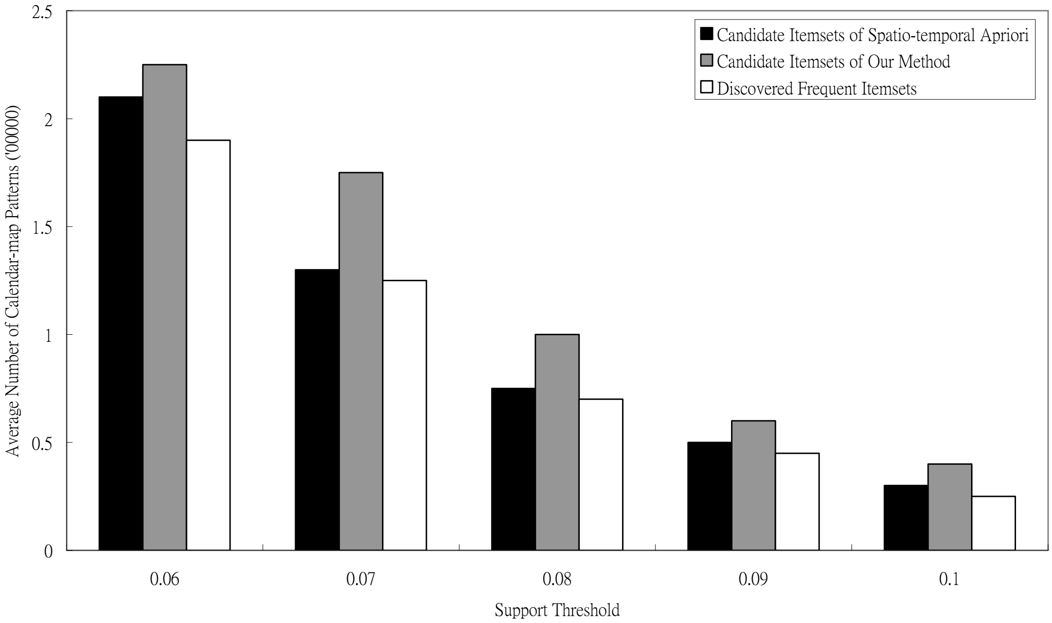


Fig. 16. Experiment 2: Average number of candidate and frequent calendar-map patterns found by Spatio-temporal Apriori and our method.

5.4 Experiment 3

We use four data sets of different sizes, (L=10;F_l=4;D=4M), (L=10;F_l=4;D=6M), (L=10;F_l=4;D=8M) and (L=10;F_l=4;D=10M), partitioned by having around 1000 transactions in each basic space-time interval. By applying different support thresholds, we compare the impact in execution time of keeping only 1-star calendar-map patterns with keeping all calendar patterns in the first scanning of data sets.

Figure 17 represents the average number of calendar-map patterns obtained after the first scanning of the four different data sets with different support thresholds.

We can clearly deduce that since all *k*-star calendar-map patterns include 1-star calendar patterns, the number of 1-star calendar patterns is inevitably less than the number of all k-star calendar patterns for $k \geq 1$. According to the calendar-map schema that we use in this experiment, the number of 1-star calendar-map patterns is roughly equal to 90% of that of all *k*-star calendar-map patterns. Hence, applying our proposed new methods can save about 10% of calendar-map patterns. Moreover, this ratio of the number of *k*-star calendar-map patterns to that of 1-star calendar-map patterns will decrease along with the increase in the complexity of the adopted calendar-map schema, for example when there are more layers in the time/place hierarchy or when there are more members in a particular layer in the hierarchy. From Figure 18, we can see the differences in execution time between keeping only 1-star calendar patterns and keeping all *k*-star calendar patterns. We can clearly see that the method of keeping only 1-star calendar patterns in the first scanning of data sets can help improve the execution efficiency to some extent.

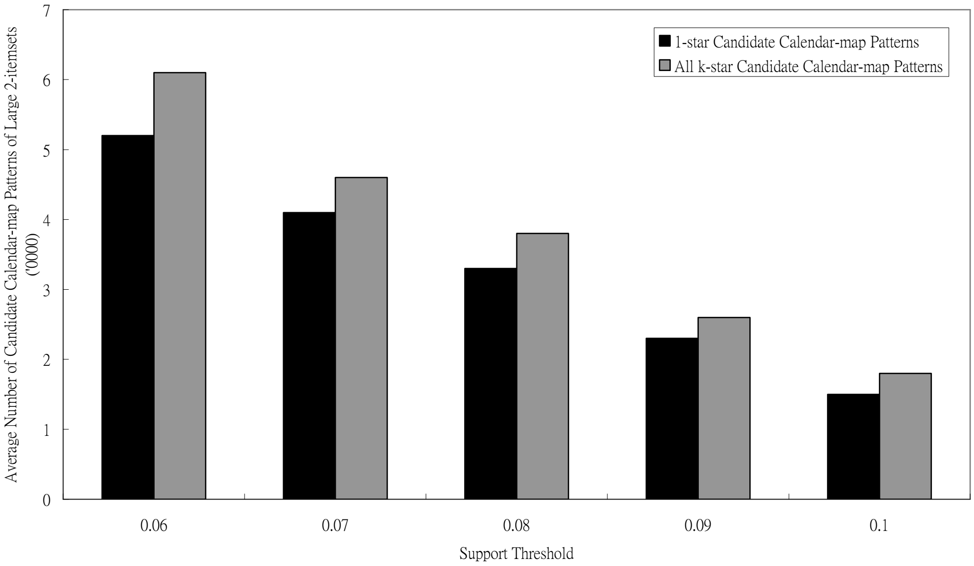


Fig. 17. Experiment 3: Comparison of the average number of calendar-map patterns between keeping only 1-star calendar-map patterns and keeping all *k*-star calendar-map patterns in the first scanning of datasets.

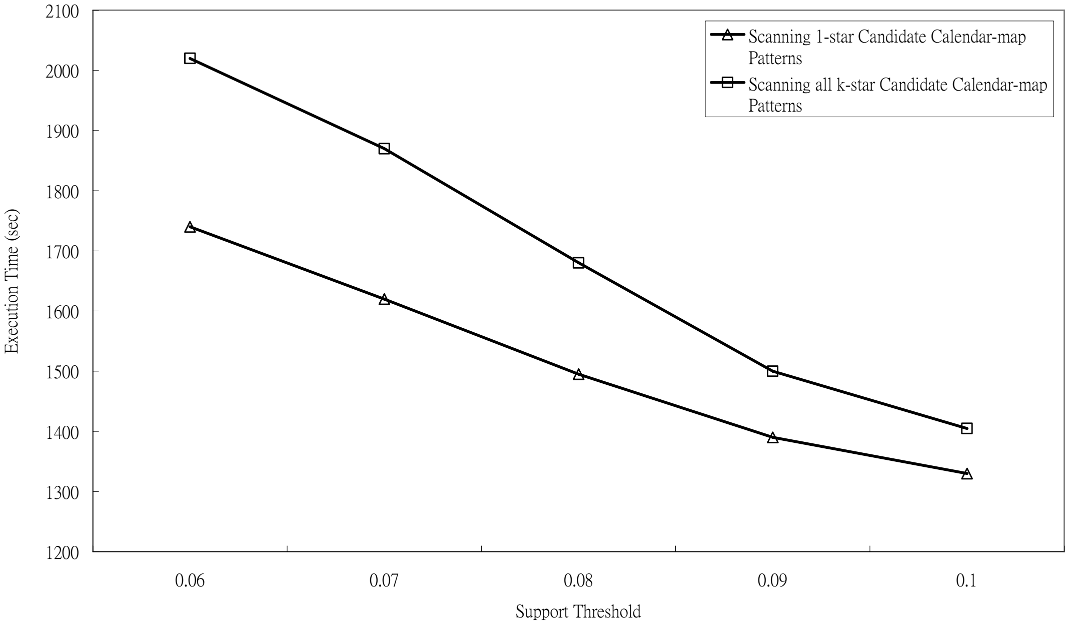


Fig. 18. Experiment 3: Comparison of execution time between keeping only 1-star calendar-map patterns and keeping all *k*-star calendar-map patterns in the first scanning of datasets.

5.5 Experiment 4

In this experiment, we use the data set *L*=10;F_l=4;*D*=1M, partitioned by having around 1000 transactions in each basic space-time interval. By adjusting the fuzzy match ratio threshold, we compare the number of discovered frequent itemsets with that of ordinary calendar-map patterns. In the simulation, we use two fuzzy calendar-map patterns. Both of them represents those space-time intervals ”close to (*, *, 15, United States, California, Los Angeles)”.

Figure 19 and Figure 20 respectively define the two fuzzy calendar-map patterns FC_1 and FC_2. Both FC_1 and FC_2 defines the range “*k* days before and after 15th of each month, in Los Angeles” as ”close to (*, *, 15, United States, California, Los Angeles)”.The asynchronous nature of FC_1 is that for a space-time intervals in Los Angeles, the closer its date to 15th of any months, the larger its degree of membership to FC_1. Compared with FC_1, in FC_2, the differences among the degrees of memberships of different space-time intervals within the range “*k* days before and after 15th of each month, in Los Angeles” are smaller. Regarding FC_1, if frequent itemsets are found on the space-time intervals having non-zero degrees of membership, the closer their date components to 15th of any months, the larger the chance that FC_1 is found to be a frequent fuzzy calendar-map pattern. On the other hand, for FC_2, only if there are a large enough number of space-time intervals in the range "*k* days before and after 15th of each month, in Los Angeles", FC_2 will be found to be a frequent fuzzy calendar-map pattern.

You may note that when *k* is defined as 0, the experiment will become non-fuzzy. In this experiment, *k* is set to be 3. The support threshold is defined to be 0.07. The weighting value α to control the contribution of asynchronous space-time intervals is defined to be 1 minus fuzzy match ratio threshold, i.e. $\alpha = 1 - fm$. We start from a comparatively large fuzzy match ratio threshold. Then, the threshold will be gradually decremented until patterns meeting user requirements can be discovered. Since α is equal to 1 minus fuzzy match ratio threshold, the larger the fuzzy match ratio threshold, the larger the impact of precise space-time intervals. In this way, the spatio-temporal association rules discovered will occur in some relatively synchronous space-time intervals. However, if there are no interesting patterns found with a comparatively large value of α, this means that not many frequent patterns occur in those relatively synchronous space-time intervals. Hence, we will continue to decrement the fuzzy match ratio threshold and repeat the mining process, in order to consider a larger contribution of asynchronous space-time intervals. The reason is that the smaller the fuzzy match ratio threshold, the larger the value of α and so, the more emphasized the impact of asynchronous space-time intervals.

In this experiment, in order to explain the effects of introducing "fuzziness" into the mining process, we intentionally add some synthetic frequent itemsets near the inquired calendar-map patterns. And then, we compare and see if the frequent itemsets can be found with ordinary calendar-map patterns and also fuzzy calendar-map patterns. This is to prove that using fuzzy calendar-map patterns can in fact find some frequent itemsets, which appear in asynchronous space-time intervals and cannot be discovered by ordinary calendar-map patterns. We first generate a group of frequent itemsets with the allowed maximum length equal to 7. The total number of frequent itemsets generated is *Q*, which is calculated as:

$$Q = 2 \times (C_2^7 + C_3^7 + C_4^7 + C_5^7 + C_6^7 + C_7^7) = 2 \times (21 + 35 + 35 + 21 + 7 + 1) = 240$$

We then randomly add some of the frequent itemsets in the range "5 days before and after 15th of each month, in Los Angeles".

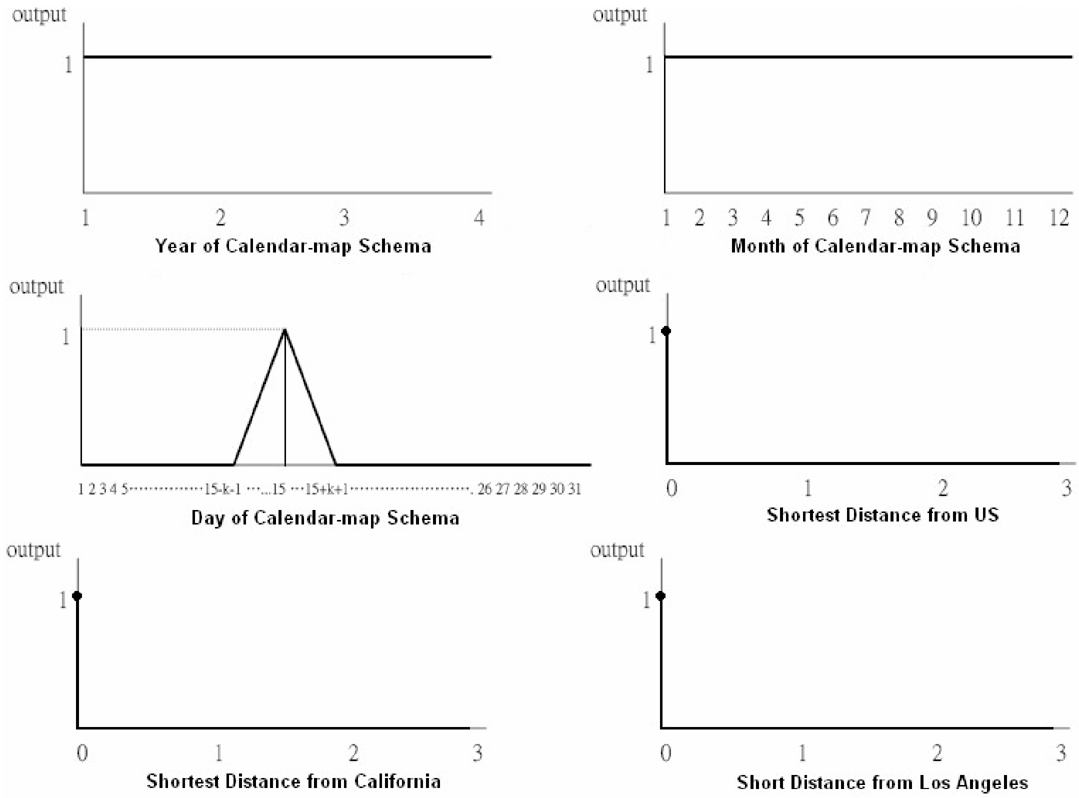


Fig. 19. Experiment 4: The membership functions of the fuzzy calendar-map pattern FC1.

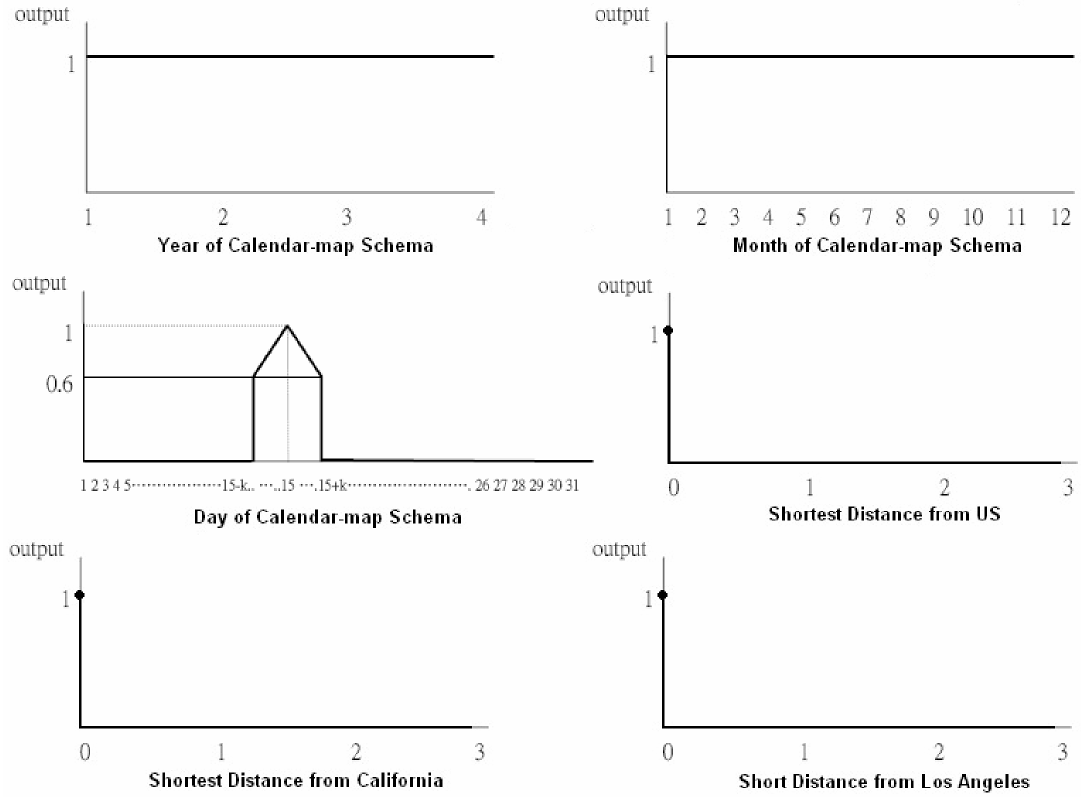


Fig. 20. Experiment 4: The membership functions of the fuzzy calendar-map pattern FC2.

Table 7 shows the quantity of frequent itemsets obtained with the ordinary calendar-map patterns (*, *, 15, United States, California, Los Angeles) (namely the General row) and the fuzzy calendar-map patterns (namely the FC_1 and FC_2 rows), as well as the quantity of the intentionally added frequent itemsets into the calendar-map patterns (namely the Target columns), while the fuzzy match ratio threshold (*Fm*) varies from 0.5 to 0.8.

From Table 7, we can see that some asynchronous spatio-temporal patterns in the data set can be found with the fuzzy calendar-map patterns. With FC_1 and FC_2, we are able to mine different patterns because of their different characteristics. Hence, from Table 7, we discover a phenomenon that if different membership functions of fuzzy calendar-map patterns are being used, different mining results will be achieved. For example, when *Fm* >= 0.7, even with the same fuzzy match ratio threshold, we still can discover some of our added frequent itemsets with FC_2, but not FC_1. Therefore, the conclusion is that users should be aware of the aynchronous nature of their interested calendar-map patterns, in designing the membership functions to fuzzify the calendar-map patterns.

Table 7. Experiment 4: Comparison of the number of frequent itemsets mined with an ordinary calendar-map pattern and fuzzy calendar-map patterns.

	Fm=0.5		*Fm*=0.6		*Fm*=0.7		*Fm*=0.8	
	Total	Target	Total	Target	Total	Target	Total	Target
General	216	0	180	0	143	0	125	0
FC_1	556	240	488	240	434	240	381	240
FC_2	564	240	502	240	209	0	145	0

5.6 Experiment 5

To compare the runtime of our proprosed new spatio-temporal association rule mining algorithm while we change the level of the concept hierarchies or branch degree, we generate five types of data sets shown in Table 8 with some default values such as the replacement rate is 0.01 and the others are just like what mention in Table 5.

Table 8. Experiment 5: Data sets for the simulation.

Data Sets	Branch Degrees	Hierarchies	Number of Transactions	Number of Items
B=2;*H*=5	2	5	10M	1000
B=2;*H*=4	2	4	10M	1000
B=2;*H*=3	2	3	10M	1000
B=3;*H*=3	3	3	10M	1000
B=4;*H*=3	4	3	10M	1000

The simulation results of the runtime are summarized in Figure 21, where the runtime is the length of mining time with each of the data sets in Table 8. The support varies from 0.1% to 1% and the results indicate that the runtime gap is increasing as the minimum support decreases. At high support values, each data set takes about the same time in generating a few rules and most of the time is spent on scanning the data sets, with our proposed new algorithm for spatio-temporal association rule mining.

In Figure 21, the lines (*B*=2;*H*=3), (*B*=3;*H*=3) and (*B*=4;*H*=3) correspond to the results, when we change the branch degree from 2 to 4. The performance deteriorates as the branch degree increases. The reason is that the number of contexts increases as the branch degree increases and we have to spend more time on finding the rules under these different contexts.

The lines (*B*=2;*H*=3), (*B*=2;*H*=4) and (*B*=2;*H*=5) correspond to the results, when we change the number of hierarchies in the concept tree from 3 to 5. The results are reasonable, because the algorithm need to spend more time on combining the relation between time periods and places from different levels. Even worst, since the number of contexts increases as the hierarchies are lengthened, the algorithm also has to spend time on finding rules under these different contexts.

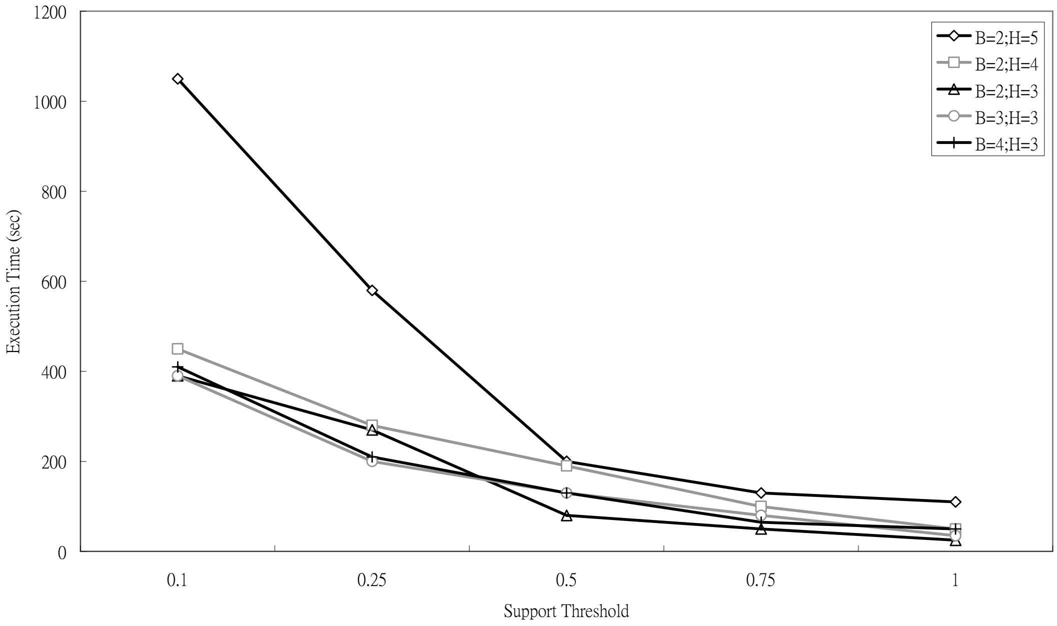


Fig. 21. Experiment 5: Compare the performance for different branch degree and hierarchies.

5.7 Experiment 6

To compare the runtime of our proposed new spatio-temporal association rule mining algorithm while changing the replacement rate, we generate five types of data sets as shown in Table 9 with some parametric settings such as the number of transactions is 10M, the number of levels in the time and place hierarchy is 3 and the branch degree is 4. The others parameters will have their default values as mentioned in Table 5.

Table 9. Experiment 6: Data sets for the simulation.

Data Sets	Branch Degrees	Hierarchies	Replacement Rate	Number of Items
R=0.001;*B*=4;*H*=3	4	3	0.001	1000
R=0.005;*B*=4;*H*=3	4	3	0.005	1000
R=0.01;*B*=4;*H*=3	4	3	0.01	1000
R=0.015;*B*=4;*H*=3	4	3	0.015	1000
R=0.02;*B*=4;*H*=3	4	3	0.02	1000

The simulation results of the runtime are summarized in Figure 22, where the runtime is the length of mining time with each of the data sets in Table 9. The support varies from 0.1% to 1% and the results indicate that the runtime gap is increasing as the minimum support decreases. At high support values, each data set takes about the same time in generating a few rules and most of the time is spent on scanning the data sets with our proposed new algorithm for spatio-temporal association rule mining.

In Figure 22, the lines (*R*=0.001;*B*=4;*H*=3), (*R*=0.005;*B*=4;*H*=3), (*R*=0.01;*B*=4;*H*=3), (*R*=0.015;*B*=4;*H*=3) and (*R*=0.02;*B*=4;*H*=3) correspond to the results, when we change the replacement rate from 0.001 to 0.02. There are no apparent changes in performance, as in the case of increasing the branch degree, when we change the replacement rate. The curves become gentle for support threshold ≥ 0.25%. The results show that the influences of replacement rate on runtime are not obvious.

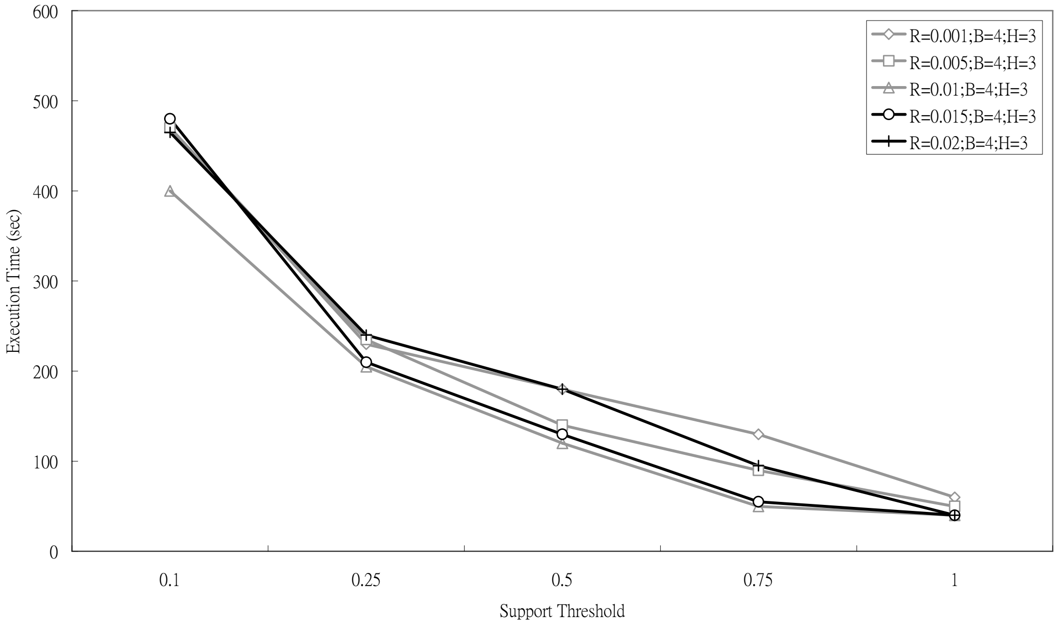


Fig. 22. Experiment 6: Compare the performance for different replacement rates.

5.8 Experiment 7

To compare the number of discovered patterns with different numbers of transactions (*D*) support thresholds (*S*), we generate eight groups of data sets as shown in Table 10.

Table 10. Experiment 7: Data sets for the simulation.

Data Sets	Support Threshold	Number of Transactions	Number of Items
D=5M;*B*=4;*H*=3;*S*=0.25	0.25%	5M	1000
D=10M;*B*=4;*H*=3;*S*=0.25	0.25%	10M	1000

D=15M;*B*=4;*H*=3;*S*=0.25	0.25%	15M	1000
D=20M;*B*=4;*H*=3;*S*=0.25	0.25%	20M	1000
D=5M;*B*=4;*H*=3;*S*=0.5	0.5%	5M	1000
D=10M;*B*=4;*H*=3;*S*=0.5	0.5%	10M	1000
D=15M;*B*=4;*H*=3;*S*=0.5	0.5%	15M	1000
D=20M;*B*=4;*H*=3;*S*=0.5	0.5%	20M	1000

The two lines (*B*=2;*H*=3;*S*=0.25) and (*B*=2;*H*=3;*S*=0.5) correspond to the two sets of results with the support thresholds 0.25% and 0.5% respectively. The data points on the lines are captured when we change the number of transactions from 50000 to 200000. From Figure 23, we can deduce that the number of discovered patterns will increase if the number of transactions increases. Also, more patterns can be discovered with a lower support threshold. This is because a higher minimum support will prune more candidate itemsets, of which the support values do not exceed the threshold. Therefore, with a higher support threshold, fewer numbers of patterns can be discovered.

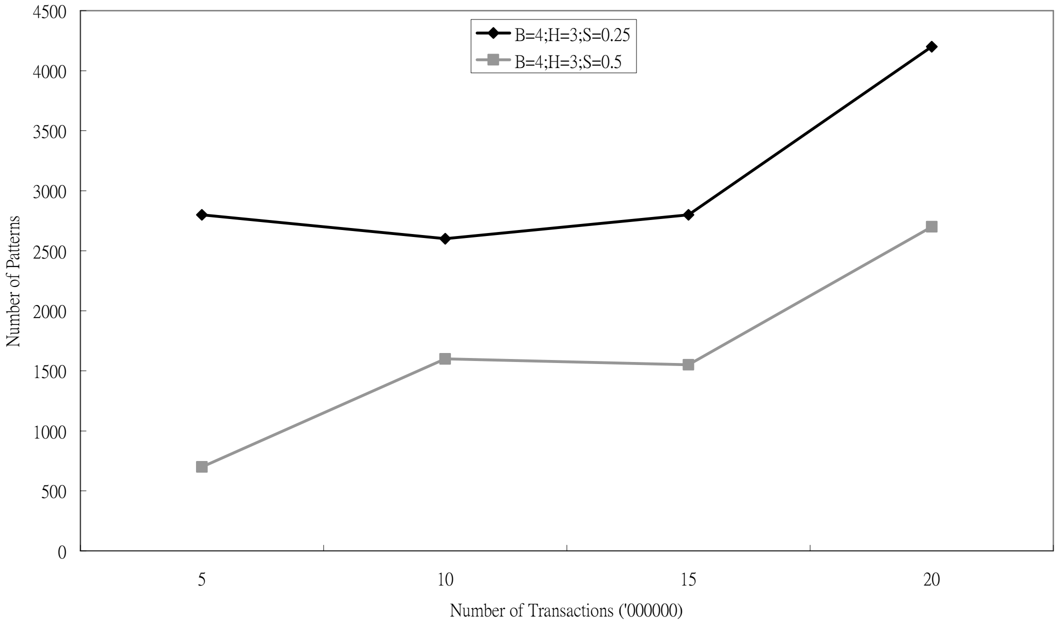


Fig. 23. Experiment 7: The number of patterns in different support thresholds.

5.9 Experiment 8

In this experiment, we test the proposed meta-mining algorithm for effectiveness when it is used to discover the underlying regularities and differences embedded in data sets. We generated six data sets for experimentation. Each tuple in these data sets is characterized by

3 attributes: X, Y, and Z. Each of these attributes can take on two values: T and F. Each data set contains 1,000 tuples. We generated the first five data sets, $D_1, \ldots, D_5$, according to the following association relationships:

$$X = F \wedge Y = F \Rightarrow Z = F$$
$$X = F \wedge Y = T \Rightarrow Z = T$$
$$X = T \wedge Y = F \Rightarrow Z = T$$
$$X = T \wedge Y = T \Rightarrow Z = F.$$

The remaining data set, D_6, was generated according to the following association relationships:

$$X = F \wedge Y = F \Rightarrow Z = F$$
$$X = F \wedge Y = T \Rightarrow Z = F$$
$$X = T \wedge Y = F \Rightarrow Z = F$$
$$X = T \wedge Y = T \Rightarrow Z = T.$$

To further examine the performance of our algorithm in the presence of uncertainty, 5% of noise was added randomly to the data sets by randomly changing the value of Z in 50 tuples (i.e., 5% of all tuples) from F to T or vice versa. We applied our proposed meta-mining algorithm to D_j to discover rules and stored the discovered rules in R_j, $j = 1, \ldots, 6$. The discovered rules together with their adjusted residuals are given in Table 11.

Table 11. Experiment 8: Rules discovered in the data sets.

Rule Set	**Rule**	**Adjusted Residual**
R_1	$X = F \wedge Y = F \Rightarrow Z = F$	16.10
	$X = F \wedge Y = T \Rightarrow Z = T$	16.61
	$X = T \wedge Y = F \Rightarrow Z = T$	16.61
	$X = T \wedge Y = T \Rightarrow Z = F$	17.13
R_2	$X = F \wedge Y = F \Rightarrow Z = F$	15.96
	$X = F \wedge Y = T \Rightarrow Z = T$	16.03
	$X = T \wedge Y = F \Rightarrow Z = T$	16.18
	$X = T \wedge Y = T \Rightarrow Z = F$	16.25
R_3	$X = F \wedge Y = F \Rightarrow Z = F$	15.96
	$X = F \wedge Y = T \Rightarrow Z = T$	15.74
	$X = T \wedge Y = F \Rightarrow Z = T$	16.76
	$X = T \wedge Y = T \Rightarrow Z = F$	16.54
R_4	$X = F \wedge Y = F \Rightarrow Z = F$	16.40
	$X = F \wedge Y = T \Rightarrow Z = T$	16.03

	$X = T \wedge Y = F \Rightarrow Z = T$	16.76
	$X = T \wedge Y = T \Rightarrow Z = F$	16.40
R_5	$X = F \wedge Y = F \Rightarrow Z = F$	17.02
	$X = F \wedge Y = T \Rightarrow Z = T$	16.73
	$X = T \wedge Y = F \Rightarrow Z = T$	16.73
	$X = T \wedge Y = T \Rightarrow Z = F$	16.43
R_6	$X = F \wedge Y = F \Rightarrow Z = F$	7.98
	$X = F \wedge Y = T \Rightarrow Z = F$	9.95
	$X = T \wedge Y = F \Rightarrow Z = F$	9.62
	$X = T \wedge Y = T \Rightarrow Z = T$	27.55

As shown in Table 11, Apriori was first used to uncover all the underlying association relationships embedded in the six data sets. Our proposed meta-mining algorithm was next used to mine meta-rules from the rule sets $R_1, \ldots, R_6$. Table 12 shows the regular meta-rules discovered from the rule sets.

Table 12. Experiment 8: Regular meta-rules discovered in the rule sets.

Regular Meta-Rule	**Adjusted Residual**
$X = F \wedge Y = F \Rightarrow Z = F$	2.60
$X = F \wedge Y = T \Rightarrow Z = T$	2.13
$X = T \wedge Y = F \Rightarrow Z = T$	2.13

The regular meta-rule "$X = F \wedge Y = F \Rightarrow Z = F$" is supported by six rules (one in each rule set), whereas the meta-rules "$X = F \wedge Y = T \Rightarrow Z = T$" and "$X = T \wedge Y = F \Rightarrow Z = T$" are supported by five rules (one in each of $R_1, \ldots, R_5$). All of them represent the regularities in the rule sets, which in turn reflect the characteristics in common in the data sets.

Let us consider the meta-rule "$X = F \wedge Y = T \Rightarrow Z = T$" as an example. It is supported by five rules. Its antecedent "$X = F \wedge Y = T$" is supported by 6 rules, whereas its consequent "$Z = T$" is supported by 11 rules. Assuming that they are independent of each other, the meta-rule is expected to be supported by 2.75 (= 11 × 6 / 24) rules. We next need to decide whether 5 is significantly larger than 2.75. To do so in an objective manner, we propose to use the adjusted residual analysis. The adjusted residual is 2.13, which is greater than 1.96 (the 95th percentile of the normal distribution). We therefore conclude that the meta-rule is supported by a sufficiently large number of rules and hence it represents one of the regularities in the rule sets (i.e., a regular meta-rule).

It is important to note that the meta-rule "$X = T \wedge Y = T \Rightarrow Z = F$" is also supported by five rules (one in each of R_1, …, R_5). Its antecedent "$X = T \wedge Y = T$" and its consequent "$Z = F$" are supported by 6 and 13 rules, respectively. Therefore, we expect that 3.25 (= 13 × 6 / 24) rules would support this meta-rule. To objectively decide whether 5 is significantly larger than 3.25, we make use of the adjusted residual analysis. The adjusted residual is found to be 1.66 (< 1.96). Hence we conclude that the meta-rule is not supported by a sufficiently large number of rules.

In addition to discovering regular meta-rules, our algorithms can also discover differential meta-rules for representing the distinctive relationships in only a few rule sets. Table 13 gives the differential meta-rules discovered from the rule sets.

Table 13. Experiment 8: Differential meta-rules discovered in the rule sets.

Differential Meta-Rule	**Adjusted Residual**
$X = F \wedge Y = T \Rightarrow Z = F$	–2.13
$X = T \wedge Y = F \Rightarrow Z = F$	–2.13

For example, the meta-rule "$X = T \wedge Y = F \Rightarrow Z = F$" is supported by only one rule in R_6. Its antecedent "$X = T \wedge Y = F$" and consequent "$Z = F$" are supported by 6 and 13 rules, respectively. Hence 3.25 (= 13 × 6 / 24) rules are expected to support this meta-rule. We find that 1 is significantly less than 3.25 as the adjusted residual is –2.13 (< –1.96). We conclude that the meta-rule is supported by a sufficiently small number of rules and hence it represents a distinguishing relationship (i.e., a differential meta-rule).

Let us consider the meta-rule "$X = T \wedge Y = T \Rightarrow Z = T$," which is also supported by one rule in R_6. Its antecedent "$X = T \wedge Y = T$" is supported by 6 rules, whereas its consequent "$Z = T$" is supported by 11 rules. We expect it would be supported by 2.75 (= 11 × 6 / 24) rules. The adjusted residual is –1.66 (> –1.96) and hence 1 is not significantly less than 2.75. We therefore conclude that the meta-rule is not supported by a sufficiently small number of rules.

5.10 Experiment 9

To evaluate our proposed methodology for mining change meta-rules, a case study has been conducted to measure how well the methodology performs its intended task of detecting significant changes. Data are prepared from an online e-shop, established by the Daka Development Limited in Hong Kong, which sells various consumer electronic goods. The datasets contain customers' profiles and purchasing history such as age, job, sex, address, registration year, cyber money, number of purchases, total purchase amount, number of visits, and payment method. We constructed a data warehouse which stores all historical data of each individual customer of the e-shop. We then extracted two datasets from the e-shop and loaded into the data warehouse so as to detect significant changes of customer purchasing behavior in it later on. The first dataset contains profiles and purchasing history of all customers from the United States, who had bought more than one product in 2004. The second dataset not only contains the same pieces of information, but also data of customers from the United States, who had made one additional purchase in 2005. After preprocessing the data by cleansing and discretization, the Apriori algorithm [Agrawal, and Srikant 1994] was applied to discover association rules in each of the datasets. We constrainted the consequent parts of the discovered association rules can only have the number of purchases or the total sales amount as output variables. In the condition of minimum support equal to 1%, minimum confidence equal to 80% and maximum allowed length of frequent itemsets equal to 3, we found 127 association rules in the first dataset and 104 association rules in the second dataset. Given the Rule Matching Threshold (*RMT*) equal to 0.4, we found 101 change meta-rules and 24 significant change meta-rules with our meta-mining approach. The number of change meta-rules for each type of change is summarized in Table 14.

Table 14. Experiment 9: Number of change meta-rules for each type of change.

Type of change	**No. of change meta-rules**	**No. of significant change meta-rules**
Emerging Patterns	92	17 (Degree of change > 0.4)
Unexpected Changes	6	4 (Degree of change > 0.3)
Added/Perished Rules	3	3 (Degree of change > 0.01)

Significant emerging patterns, unexpected changes, added/perished rules are summarized in Table 15, 16, and 17 respectively.

Table 15. Experiment 9: Significant emerging patterns (degree of change > 0.4).

r_i^t **(Or** r_j^{t+k} **)**	**Rule Support**		α_{ij} **(> 0.4)**
	$Sup^t(r_i)$	$Sup^{t+k}(r_j)$	
1) Visit=Low, Job=Specialist⇒OrdCnt=Low	0.037	0.078	1.11
2) Visit=Low, ReservedMoney=Low⇒OrdCnt=Low	0.177	0.368	1.08
3) Visit=Low, ReservedMoney=Low⇒Sales=Low	0.177	0.368	1.08
4) Visit=High, Job=Specialist⇒OrdCnt=High	0.021	0.04	0.90
5) Visit=High, Job=Specialist⇒Sales=High	0.021	0.04	0.09
6) Visit=High, Addr=Christiansted⇒OrdCnt=High	0.01	0.017	0.7
7) Visit=High, Addr=Jacksonville⇒OrdCnt=High	0.015	0.025	0.67
8) Visit=High, Addr=Jacksonville⇒Sales=High	0.015	0.025	0.67
9) ReservedMoney=Low, Job=Student⇒Sales=Low	0.011	0.018	0.64
…	…	…	…
17) Visit=Low, Addr=ChungBuk⇒OrdCnt=Low	0.01	0.014	0.40

Table 16. Experiment 9: Significant unexpected changes (degree of change > 0.3).

r_i^t	r_j^{t+k}	δ_{ij}	δ'_{ij}	α_{ij}
1)Sex=F, Addr=Kansas City⇒ OrdCnt=Low (Support: 0.034)	Visit=High,Addr=Kansas City⇒OrdCnt=High (Support: 0.015)	0.5	0.5	0.85
2) Registry=This Year, Addr=Kansas City⇒OrdCnt=Low (Support: 0.032)	Visit=High,Addr=Kansas City⇒OrdCnt=High (Support: 0.015)	0.5	0.5	0.79
3) Payment=Cash, Addr=Kansas City⇒OrdCnt=Low	Visit=High,Addr=Kansas City⇒OrdCnt=High (Support: 0.015)	0.5	0.5	0.58
4) ReservedMoney=Low, Addr=Kansas City⇒OrdCnt=Low	Visit=High,Addr=Kansas City⇒OrdCnt=High (Support: 0.015)	0.5	0.5	0.31

Table 17. Experiment 9: Significant added/perished rules (degree of change > 0.01).

r_i^t	Support	α_{ij}
1) Age=Teen⇒Sales=Low	0.018	0.018
2) Sex=F, Age=Teen⇒Sales=Low	0.015	0.015
3) Age=Thirtith, Addr=Jacksonville⇒Sales=Low	0.012	0.012

From the change meta-rules 4) and 5) in Table 15, we can see a rapid growth (90% growth) in sales for customers, who are both specialists and frequent visitors. Although the support values of the meta-rules are low (0.021, 0.04), these type of customers have a high potential to become loyal customers in the near future due to their high growth rate in sales. Therefore, a marketing campaign, which aims to encourage their revisiting, is believed to be worthy and beneficial to the company. The sales pattern is also supported by the change meta-rule 1) in Table 15.

From the change meta-rules 6), 7) and 8) in Table 15, we can see a rapid growth in sales for customers, who live in Christiansted or Jacksonville city and visit the e-shop frequently. Without the meta-mining approach, the marketing manager may misunderstand that customers who live in Christiansted or Jacksonville city and visit the e-shop frequently are not important because the low support values of the association rules 6), 7) and 8) in Table 15, mined with Apriori, show that they are now not high-sales customers. However, their rapid growth in sales reflects that they are potential customers in the coming future.

With regard to unexpected changes, we identified 4 significant changes. From the change meta-rule 1) in Table 16, we can see that the sales of female customers who live in Kansas City are low from the first dataset. However, in the second dataset, we can deduce that sales for customers, who visit the e-shop frequently and live in Kansas City, are high, even if they are female. It means that the importance of customers who live in Kansas City and visit the e-shop frequently is gradually increasing. Therefore, a change in the existing marketing strategy and plan is required. The change meta-rules 2), 3) and 4) in Table 16 can also be interpreted similarly.

Finally, three perished rules are found in Table 17. In 2004, most of their customers were aged around twentith and sales of the other age groups were very low. But in 2005, we can find a trend that the age of customers covers a wider range. Therefore, additional services and products for the elderly and teenagers may be needed.

The results of this case study show that our methodology for mining change meta-rules can discover some hidden, interesting and non-trivial patterns that originary association rule mining approaches cannot find.

Chapter 6
Future Work

The future work can include several directions.

First, we would like to explore other meaningful semantics of spatio-temporal association rules and extend our techniques to solve the corresponding data mining problems.

Second, we would like to consider spatio-temporal patterns in data mining problems other than association rule mining, such as clustering.

Third, mining spatio-temporal patterns involves investigating not only large itemset space and pattern space, but also a large amount of data collected in a long history. It is thus crucial to develop parallel or distributed algorithms for large scale data mining. It would also be interesting to devise online and incremental algorithms for this problem.

Fourth, building the concept hierarchy trees or the calendar-map patterns need background knowledge to determine the granularities of dimension atoms, and control the generalization process. We would like to involve appropriate existing technologies such as cluster or segmentation tools in Customer Relationship Management (CRM) to build the concept hierarchy tree in each dimension for different application domain. There can be more than one way to define hierarchy trees of time and location. Our method assumes one hierarchy of time and location defined by domain experts in this phase. Future work may include handling of multiple hierarchies of time and location as well as automating the hierarchy formation.

Fifth, we would like to design a user interface for representing the rules discovered friendly to help users make use of the discovered patterns.

Sixth, the entire work mentioned in this thesis can be further strengthened with more systematic studies and empirical studies with different discretization methods and multiple date sources of spatio-temporal data.

Last but not the least, more experiments and numerical analysis can be done in the future. Experiments are now conducted in one run. We may make multiple runs and get averages as well as use t-test to see if the difference is significant.

Chapter 7
Conclusions

This work presented a new approach to solve the association rule mining problem handling the spatial and temporal dimension, i.e., the problem of spatio-temporal association rule mining. We have proposed an algorithm to discover calendar-map-based spatio-temporal association rules that appear over any time intervals and locations in a spatio-temporal database. An example of a calendar-map-based spatio-temporal association rule is “turkey and pumpkin are frequently purchased together in the United States in the week before Thank-giving.” It is also an example where existing algorithms fail to mine quite evident spatio- temporal association rules, which justifies the need of a new approach.

From the perspective of a strategist, rules must not only be discovered, but also be suitable for human natural thinking. In other words, rules will not have usability if they do not have readability. For example, executive personnel may not concern about what kind of rules would be held in a single location day by day, but rather to know what kind of the rules would be held in a country as a whole every season. Since different executives will require different interpretation of the discovered rules for different scenarios and these scenarios under different granularities of time-and-place will have different business knowledge, the new method that we developed aims to achieve this goal and meet such dynamic needs by adopting calendar-map schemas.

A user-defined calendar-map schema, e.g., (year, month, day, country, province, city), a combination of the calendar schema (year, month, day) and the map schema (country, province, city), is adopted to specify the interested time intervals and locations as calendar-map patterns. Then, in every space-time interval, the frequent 2-itemsets are discovered along with their 1-star calendar-map patterns. After that, the information of the rest k-star calendar patterns of the frequent 2-itemsets is aggregated from their 1-star calendar-map patterns. Thus, the minimal number of calendar-map patterns are generated and counted in the database. Further, to avoid multiple scans over the database, all candidate itemsets are generated from discovered frequent 2-itemsets and the Apriori downward property is utilized to generate the minimal number of their candidate calendar-map patterns. Finally, all frequent itemsets and their calendar-map patterns are discovered in one shot. Calendar-map-based spatio-temporal association rules are then obtained.

We have also proposed a way to extend Apriori, the most well-known association rule mining algorithm, to mine calendar-map-based spatio-temporal association rules. Experimental results have shown that our proposed new method is more efficient than the Apriori-like approach. An explanation of the experiment results is that the number of database scanning time in Apriori-like approaches increases with the maximum allowed length of frequent itemsets but our method scans the target database at most twice. After the first scanning and before the second scanning, our method generates all k-itemsets and the candidates of their associated calendar-map patterns from frequent 2-itemsets and their frequent calendar-map patterns. Moreover, our method makes use of the nature of calendar-map patterns. It collects itemset counts from 1-star calendar-map patterns to further reduce the database scanning time and therefore increase the overall execution efficiency. One limitation of our proposed algorithm is that the number of *k*-itemsets can be very large to be kept in the memory. It depends on the minimum support threshold and the maximum allowed length of frequent itemsets defined. However, our algorithm requires at most two database scans. Therefore, this is a tradeoff between database and memory usage. Furthermore, our method keeps only 1-star calendar-map patterns and filters out candidate itemsets with match ratio. This can help reduce the number of candidate itemsets and therefore memory usage. Apriori-like algorithms doesn't have this mechanism to reduce the memory usage.

In this thesis, we proposed two classes of spatio-temporal association rules, spatio-temporal association rules with respect to precise match and spatio-temporal association rules with respect to fuzzy match, to represent regular association rules along with their spatio-temporal patterns in terms of calendar-map schemas. We mine spatio-temporal association rules with respect to fuzzy match with the use of fuzzy calendar-map patterns, which can be used to discover patterns often occurring in a particular time interval at a particular location but asynchronously, i.e. not exactly. The reasonale behind is that there is inevitably some time shift or location shift even for some very regular patterns or human behaviour. An immediate advantage is that the-corresponding data mining problem requires less prior knowledge than the prior methods and hence may discover more unexpected rules.

This study proposes to mine a set of rules from the rules sets discovered by a data mining algorithm. These rules are called meta-rules because they are rules about rules. We define the problems of discovering the underlying regularities, differences, and changes hidden in rule sets and propose a new approach to dealing with these problems. We refer to the proposed approach as a meta-mining approach since it mines previous mining results.

Given a collection of rule sets, each of which is discovered in a data set, the meta-mining of regularities is concerned with the discovery of association relationships that are supported by a sufficiently large number of rules in the rule sets. They are in common in different data sets (i.e., the regularities) and hence they are called regular meta-rules. The regular meta-rules are especially useful for an interstate or international company to better make business decisions that are beneficial to the company as a whole.

The meta-mining of differences from the rule sets aims at revealing rules that are supported by a sufficiently small number of rules. They represent the distinguishing characteristics of the few data sets. They are therefore referred to as differential meta-rules. The differential meta-rules are very useful for an international company to better make decisions that are beneficial to specific branches.

In addition to discovering regularities and differences, we also propose to discover the changes in rules over time. The goal in meta-mining changes from rule sets is to uncover the regularities governing how the rules change over time (i.e., the change meta-rules). Change meta-rules reflect change in the underlying characteristics hidden in the data. They can be used for human examination and for predicting how the rules will change in the future. Unless one takes changes into consideration, one can only predict based on historical data and the prediction cannot lead to any change because it will no longer be valid. Knowing the changes in advance allows a business organization not only to provide new products and services to satisfy the changing needs of its customers, but also to design corrective actions to stop or delay undesirable changes.

The experimental results on synthetic data sets for meta-mining also show that our algorithms are effective for discovering the underlying regularities, exceptions, and changes embedded.

In conclusion, our proposed spatio-temporal data mining approach is very effective not only in mining rules from data sets, but also in mining meta-rules from rule sets. The discovered meta-rules effectively represent the underlying regularities, differences, and changes hidden in the rule sets, which in turn reflect the regularities, the differences, and the change of characteristics in the data sets.

To the best of our knowledge, this is the first work, which extends the association patterns (sometimes called association rule discovery problem) to the spatial-temporal databases, where each transaction is associated with a timestamp and location. Hence, in the comparative studies with existing research efforts, we can only compare the performance of our proposed new algorithm with that of the Spatio-temporal Apriori, our extension of the most well-known association rule mining algorithm. Likewise, we did not find a work on meta-mining similar to ours, which aims to discover meta-rules from two or more sets of association rules. We tested our algorithms with both synthetic and real data. Our methods are able to effectively discover those synthetic patterns in tests with synthetic data. In tests with real data, the mined patterns have been verified by the business users of the spatio-temporal database to be non-trivial and interesting. Therefore, the uniqueness and the preciseness in mining both asssocation rules and meta-association rules in spatio-temporal databases are the main contribution of this thesis.

References

Agrawal, R., Ghosh, S., Imielinski, T., Iyer, B., and Swami, A. (1992) "An Interval Classifier for Database Mining Applications," in *Proc. of the 18th Int'l Conf. on Very Large Data Bases*, Vancouver, British Columbia, Canada, pp. 560–573.

Agrawal, R. and Psaila, G. (1995) "Active Data Mining," in *Proc. of the 1st Int'l Conf. on Knowledge Discovery and Data Mining*, Montreal, Canada.

Agrawal, R. and Shafer, J. C. (1996) "Parallel Mining of Association Rules," in *IEEE Trans. on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 962–969.

Agrawal, R. and Srikant, R. (1994) "Fast Algorithms for Mining Association Rules," in *Proc. of the 20th Int'l Conf. on Very Large Data Bases*, Santiago, Chile, pp. 487–499.

Agrawal, R. and Srikant, R. (1995) "Mining Sequential Patterns," in *Proc. of the 11th IEEE Int'l Conf. on Data Engineering*, Taipei, Taiwan, pp. 3–14.

Agrawal, R., Imielinski, T., and Swami, A. (1993a) "Database Mining: A Performance Perspective," in *IEEE Trans. on Knowledge and Data Engineering*, vol. 5, no. 6, pp. 914–925.

Agrawal, R., Imielinski, T., and Swami, A. (1993b) "Mining Association Rules between Sets of Items in Large Databases," in *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, Washington D.C., pp. 207–216.

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., and Verkamo, A. I. (1995) "Fast Discovery of Association Rules," in *Advances in Knowledge Discovery and Data Mining*, Chapter 12, AAAI/MIT Press, 1995.

Agrawal, R., Mehta, M., Shafer, J., Srikant, R., Arning, A., and Bollinger, T. (1996) "The Quest Data Mining System," in *Proc. of the 2nd Int'l Conf. on Data Mining and Knowledge Discovery*, Portland, OR, pp. 244–249.

Au, W. H. and Chan, K. C. C. (1998) "An Effective Algorithm for Discovering Fuzzy Rules in Relational Databases," in *Proc. of the 7th IEEE Int'l Conf. on Fuzzy Systems*, Anchorage, AK, pp. 1314–1319.

Au, W. H. and Chan, K. C. C. (1999) "FARM: A Data Mining System for Discovering Fuzzy Association Rules," in *Proc. of the 8th IEEE Int'l Conf. on Fuzzy Systems*, Seoul, Korea, pp. 1217–1222.

Au, W. H. and Chan, K. C. C. (2001) “Classification with Degree of Membership: A Fuzzy Approach,” in *Proc. of the 1st IEEE Int’l Conf. on Data Mining*, San Jose, CA, pp. 35–42.

Au, W. H. and Chan, K. C. C. (2002a) “An Evolutionary Approach for Discovering Changing Patterns in Historical Data,” in B. V. Dasarathy (Ed.), *Data Mining and Knowledge Discovery: Theory, Tools, and Technology IV*, Proc. of SPIE Vol. 4730, pp. 398–409.

Au, W. H. and Chan, K. C. C. (2002b) “Fuzzy Data Mining for Discovering Changes in Association Rules over Time,” in *Proc. of the 11th IEEE Int’l Conf. on Fuzzy Systems*, Honolulu, HI, pp. 890–895.

Au, W. H. and Chan, K. C. C. (2003) “Mining Fuzzy Association Rules in a Bank-Account Database,” in *IEEE Trans. on Fuzzy Systems*, vol. 11, no. 2, pp. 238–248.

Au, W. H. and Chan, K. C. C. (2005) “Mining Changes in Association Rules: A Fuzzy Approach,” in *Fuzzy Sets and Systems*, vol. 149, no. 1, pp. 87–104.

Box, G. E. P., Jenkins, G. M., and Reinsel, G. C. (1994) *Time Series Analysis: Forecasting and Control*, 3rd Ed., Englewood Cliffs, NJ: Prentice-Hall.

Bradley, P., Fayyad, U., and Reina, C. (1998) “Scaling Clustering Algorithms to Large Databases,” in *Proc. of the 4th Int’l Conf. on Knowledge Discovery and Data Mining*, New York, NY, pp. 9–15.

Bright, M., Hurson, A., and Pakzad, S. (1992) “A Taxonomy and Current Issues in Multidatabase Systems,” in *IEEE Computer*, vol. 25, no. 3, pp. 50–60.

Brin, S., Motwani, R., Silverstein, C. (1997a) “Beyond Market Baskets: Generalizing Association Rules to Correlations,” in *Proc. of the ACM SIGMOD Int’l Conf. on Management of Data*, Tucson, AZ, pp. 265–276.

Brin, S., Motwani, R., Ullman, J. D., and Tsur, S. (1997b) “Dynamic Itemset Counting and Implication Rules for Market Basket Data,” in *Proc. of the ACM SIGMOD Int’l Conf. on Management of Data*, Tucson, AZ, pp. 255–264.

Brunk, C., Kelly, J., and Kohavi, R. (1997) “MineSet: an integrated system for data mining,” in David Heckerman, Heikki Mannila, Daryl Pregibon (Eds.) *Proc. 3rd Int. Conf. On Knowledge Discovery and Data Mining (KDD-97). Newport Beach. California. USA.* AAAI Press, 37-42.

Chan, K. C. C. and Au, W. H. (1997a) “An Effective Algorithm for Mining Interesting Quantitative Association Rules,” in *Proc. of the 12th ACM Symp. on Applied Computing*, San Jose, CA, pp. 88–90.

Chan, K. C. C. and Au, W. H. (1997b) “Mining Fuzzy Association Rules,” in *Proc. of the 6th Int’l Conf. on Information and Knowledge Management*, Las Vegas, NV, pp. 209–215.

Chan, K. C. C. and Au, W. H. (2001) “Mining Fuzzy Association Rules in a Database Containing Relational and Transactional Data,” in A. Kandel, M. Last, and H. Bunke (Eds.), *Data Mining and Computational Intelligence*, New York, NY: Physica-Verlag, pp. 95–114.

Chan, K. C. C. and Wong, A. K. C. (1990) “APACS: A System for the Automatic Analysis and Classification of Conceptual Patterns,” in *Computational Intelligence*, vol. 6, no. 3, pp. 119–131.

Chan, K. C. C. and Wong, A. K. C. (1991) “A Statistical Technique for Extracting Classificatory Knowledge from Databases,” in [Piatetsky-Shapiro and Frawley 1991], pp. 107–123.

Chan, K. C. C., Au, W. H., and Choi, B. (2002) “Mining Fuzzy Rules in a Donor Database for Direct Marketing by a Charitable Organization,” in *Proc. of the 1st IEEE Int’l Conf. on Cognitive Informatics*, Calgary, Alberta, Canada, pp. 239–246.

Cheeseman, P. and Stutz, J. (1996) “Bayesian Classification (AutoClass): Theory and Results,” in [Fayyad *et al.* 1996b], pp. 153–180.

Cheung, D. W., Han, J., Ng, V. T., and Wong, C. Y. (1996b) “Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique,” in *Proc. of the 12th Int’l Conf. on Data Engineering*, New Orleans, LA, pp. 106–114.

Cheung, D. W., Han, J., Ng, V. T., Fu, A. W., and Fu, Y. (1996a) “A Fast Distributed Algorithm for Mining Association Rules,” in *Proc. of the 4th Int’l Conf. on Parallel and Distributed Information Systems*, Miami Beach, FL, pp. 31–42.

Collins. (1995) *Road Atlas Britain*, HarperCollins, London.

Delgado, M., Marín, N., Sánchez, D., and Vila, M. A. (2003) “Fuzzy Association Rules: General Model and Applications,” in *IEEE Trans. on Fuzzy Systems*, vol. 11, no. 2, pp. 214–225.

Dempster, A. P. (1967) “Upper and Lower Probabilities Induced by a Multi-Valued Mapping,” in *Annals of Mathematical Statistics*, vol. 38, pp. 325–339.

Ester, M., Frommelt, A., Kriegel, H. P., and Sander, J. (1998) “Algorithms for Characterization and Trend Detection in Spatial Databases,” in *Proc. of KDD’98*, pp. 44–50.

Ester, M., Kriegel, H. P., and Sander, J. (1997) “Spatial Data Mining: A Database Approach,” in *Proc. of SSD’97*, pp. 47–66.

Estivill-Castro, V. and Lee, I. (2000) “AUTOCLUST+: Automatic Clustering of Point-Data Sets in the Presence of Obstacles,” in *Proc. of TSDM’00*, pp. 133–146.

Fayyad, U. M., Piatetsky-Shapiro, G., and Smyth, P. (1996a) “From Data Mining to Knowledge Discovery: An Overview,” in [Fayyad *et al.* 1996], pp. 1–34.

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, R. (Eds.) (1996b) *Advances in Knowledge Discovery and Data Mining*, Menlo Park, CA; Cambridge, MA: AAAI/MIT Press.

Fayyad, U.M., Piatetsky-Shapiro, G., and Smyth, P. (1997) “From Data Mining to Knowledge Discovery: An Overview,” in U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth & R. Uthurusamy (Eds.) *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, pp. 1–29.

Ganti, V., Gehrke, J., Ramakrishnan, R., and Loh, W. Y. (1999a) “A Framework for Measuring Changes in Data Characteristics,” in *Proc. of the 18th ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems*, Philadelphia, PA, pp. 126–137.

Ganti, V., Ramakrishnan, R., Gehrke, J., Powell, A. L., and French, J. C. (1999b) “Clustering Large Data Sets in Arbitrary Metric Spaces,” in *Proc. of the 15th Int’l Conf. on Data Engineering*, Sydney, Australia, pp. 502–511.

Gaul, W. and Säuberlich, F. (1999) “Classification and positioning of data mining tools,” in W. Gaul & H. Locarek-Junge (Eds.) *Classification in the Information Age.* Berlin: Springer, pp. 143–152.

Gehrke, J., Ramakrishnan, R. and Ganti, V. (1998) “RainForest - a Framework for Fast Decision Tree Construction of Large Data sets,” in *Proceedings of the 24th International Conference on Very Large Data Bases*, San Francisco, pp. 416-427.

Giannella, C., Han, J., Pei, J., Yan, X., and Yu, P. S. (2003), “Mining frequent patterns in data streams at multiple time granularities,” in *Next Generation Data Mining*, H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha, Eds. Menlo Park, CA: AAAI/MIT.

Guo, D., Peuquet, D., and Gahegan, M. (2002) "Opening the black box: interactive hierarchical clustering for multivariate spatial patterns," in *Proc. of ACM GIS'02*, pp. 131–136.

Hadjieleftheriou, M., Kollios, G., Gunopulos, D., and Tsotras, V. J. (2003) "On-Line Discovery of Dense Areas in Spatio-temporal Databases," in *Proc. of SSTD'03*, pp. 306–324.

Han, J. , Dong, G., and Yin, Y. (1999) "Efficient Mining of Partial Periodic Patterns in Time Series Database," in *Proc. of the 15th IEEE Int'l Conf. on Data Engineering*, Sydney, Australia, pp. 106–115.

Han, J. and Fu, Y. (1995) "Discovery of Multiple-Level Association Rules from Large Databases," in *Proc. of the 21st Int'l Conf. on Very Large Data Bases*, Zurich, Switzerland, pp. 420–431.

Han, J. and Fu, Y. (1996) "Exploration of the Power of Attribute-Oriented Induction in Data Mining," in [Fayyad *et al.* 1996b], pp. 399–421.

Han, J., Fu, Y., Wang, W., Chiang, J., Gong, W., Koperski, K., Li, D., Lu, Y., Rajan, A., Stefanovic, N., Xia, B., and Zaiane, O. R. (1996) "DBMiner: A System for Mining Knowledge in Large Relational Databases," in *Proc. of the 2nd Int'l Conf. on Data Mining and Knowledge Discovery*, Portland, OR, pp. 250–255.

Han, J., Koperski, K., Stefanovic, N. (1997) "GeoMiner: A System Prototype for Spatial Data Mining," in *Proc. of SIGMOD'97*, pp. 553–556.

Han, J., Pei, J., and Yin, Y. (2000) "Mining Frequent Patterns without Candidate Generation," in *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, Dallas, TX, 2000, pp. 1–12.

Hastie, T., Tibshirani, R., and Friedman, J. (2001) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer Verlag, ISBN: 0-387-95284-5.

Hipp, J., Güntzer, U., and Nakhaeizadeh, G. (2000) "Algorithms for Association Rule Mining – A General Survey and Comparison," in *Proc. of ACM SIGKDD Explorations*, 2(1), pp. 58–64.

Hirota, K. and Pedrycz, W. (1999) "Fuzzy Computing for Data Mining," in *Proc. of the IEEE*, vol. 87, no. 9, pp. 1575–1600.

Houtsma, M. and Swami, A. (1995) "Set-Oriented Mining for Association Rules in Relational Databases," in *Proc. of the 11th Int'l Conf. on Data Engineering*, Taipei, Taiwan, pp. 25–33.

Hüllermeier, E. (2001) "Implication-Based Fuzzy Association Rules," in *Proc. of the 5th European Conf. on Principles of Data Mining and Knowledge Discovery*, Freiburg, Germany, pp. 241–252.

Imielinski, T., Virmani, A., and Abdulghani, A. (1996) "DataMine: Application Programming Interface and Query Language for Database Mining," in *Proc. of the 2nd Int'l Conf. on Data Mining and Knowledge Discovery*, Portland, OR, pp. 256–262.

Indulska, M. and Orlowska, M. E.. (2002) "Gravity based spatial clustering," in *Proc. of ACM-GIS'02*, pp. 125–130.

Ishibuchi, H., Yamamoto, T., and Nakashima, T. (2001) "Fuzzy Data Mining: Effect of Fuzzy Discretization," in *Proc. of the 1st IEEE Int'l Conf. on Data Mining*, San Jose, CA, pp. 241–248.

Janikow, C. Z. (1998) "Fuzzy Decision Trees: Issues and Methods," in *IEEE Trans. on Systems, Man, and Cybernetics – Part B: Cybernetics*, vol. 28, no. 1, pp. 1–14.

Kacprzyk, J. and Zadrozny, S. (2001) "On Linguistic Approaches in Flexible Querying and Mining of Association Rules," in H. L. Larsen, J. Kacprzyk, S. Zadrozny, T. Andreasen, and H. Christiansen (Eds.), *Flexible Query Answering Systems: Recent Advances*, Proc. of the 4th Int'l Conf. on Flexible Query Answering Systems, Heidelberg, Germany: Physica-Verlag, pp. 475–484.

Knorr, E. M. and Ng, R. T. (1996a) "Finding Aggregate Proximity Relationships and Commonalities in Spatial Data Mining," in *IEEE Trans. Knowl. and Data Eng.*, 8(6), pp. 884–897.

Knorr, E. M. and Ng, R. T. (1996b) "Extraction of Spatial Proximity Patterns by Concept Generalization," in *Proc. of KDD'96*, pp. 347–350.

Koperski, K. and Han, J. (1995) "Discovery of Spatial Association Rules in Geographic Information Databases," in *Proc. of SSD'95*, pp. 47–66.

Koperski, K., Adhikary, J., and Han, J. (1996) "Spatial Data Mining: Progress and Challenges," in *Proc. of ACM SIGMOD'96*.

Kurgan, L. A. and Cios, K. J. (2004) "Meta Mining Architecture for Supervised Learning," in *Proc. of the 7th Int'l Workshop on High Performance and Distributed Mining*, Lake Buena Vista, FL, pp. 18–26.

Lee, D. H. and Kim, M. H. (1997) "Database Summarization Using Fuzzy ISA Hierarchies," in *IEEE Trans. on Systems, Man, and Cybernetics – Part B: Cybernetics*, vol. 27, no. 4, pp. 671–680.

Li, Y., Wang, X. S., and Jajodia, S. (2000) "Discovering Temporal Patterns in Multiple Granularities," in *Proc. of TSDM'00*, pp. 5–19.

Liu, B., and Hsu, W. (1996) "Post-Analysis of Learned Rules," in *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*.

Liu, B., Hsu, W., and Chen, S. (1997) "Using general impressions to analyze discovered classification rules," in *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*.

Liu, B., Hsu, W., and Ma, Y. (1998) "Integrating Classification and Association Rule Mining," in *Proc. of the 4th Int'l Conf. on Knowledge Discovery and Data Mining*, New York, NY, pp. 80–86.

Liu, B., Hsu, W., and Ma, Y. (2001) "Discovering the Set of Fundamental Rule Changes," in *Proc. of the 7th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, San Francisco, CA, pp. 335–340.

Liu, B., Hsu, W., Han, H. S., and Xia, Y. (2000) "Mining Changes for Real-Life Applications," in *Proc. of the 2nd Int'l Conf. on Data Warehousing and Knowledge Discovery,* London Greenwich, U.K.

Liu, H., Lu, H., and Yao, J. (1998) "Identifying Relevant Databases for Multidatabase Mining," in *Proc. of the 2nd Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, Melbourne, Australia, pp. 210–221.

Lu, H., Setiono, R., and Liu, H. (1995) "NeuroRule: A Connectionist Approach to Data Mining," in *Proc. of the 21st Int'l Conf. on Very Large Data Bases*, Zurich, Switzerland, pp. 478–489.

Lu, W., Han, J., and Ooi, B. C. (1993) "Discovery of General Knowledge in Large Spatial Databases," in *Proc. of Far East Workshop on GIS*, pp. 275–289.

Maimon, O., Kandel, A., and Last, M. (1999) "Information-Theoretic Fuzzy Approach to Knowledge Discovery in Databases," in R. Roy, T. Furuhashi, and P. K. Chawdhry (Eds.), *Advances in Soft Computing – Engineering Design and Manufacturing*, London, U.K.: Springer-Verlag, pp. 315–326.

Mannila, H., Toivonen, H., and Verkamo, A. I. (1994) "Efficient Algorithms for Discovering Association Rules," in *Proc. of the AAAI Workshop on Knowledge Discovery in Databases*, Seattle, WA, pp. 181–192.

Mannila, H., Toivonen, H., and Verkamo, A. I. (1995) "Discovering Frequent Episodes in Sequences," in *Proc. of the 1st Int'l Conf. on Knowledge Discovery and Data Mining,* Montreal, Canada, pp. 210–215.

Mannila, H., Toivonen, H., and Verkamo, A. I. (1997) "Discovery of frequent episodes in event sequences," in Report C-1997-15, University of Helsinki, Department of Computer Science.

Matheus, C. J., Chan, P. K., and Piatetsky-Shapiro, G. (1993) "Systems for Knowledge Discovery in Databases," in *IEEE Trans. on Knowledge and Data Engineering*, vol. 5, no. 6, pp. 903–913.

Mehta, M., Agrawal, R., and Rissanen, J. (1996) "SLIQ: A Fast Scalable Classifier for Data Mining," in *Proc. of the 5th Int'l Conf. on Extending Database Technology,* Avignon, France, pp. 18–32.

Mitra, S., Pal, S. K., and Mitra, P. (2002) "Data Mining in Soft Computing Framework: A Survey," in *IEEE Trans. on Neural Networks*, vol. 13, no. 1, pp. 3–14.

Ng, R. and Han, J. (1994) "Efficient and effective clustering method for spatial data mining," in *Proc. of VLDB'94,* pp. 144–155.

Ng, R. T. and Han, J. (2002) "CLARANS. A Method for Clustering Objects for Spatial Data Mining," in *IEEE Trans. Knowl. Data Eng.*, 14(5), pp. 1003–1016.

Padmanabhan, B., and Tuzhilin, A. (1999) "Unexpectedness as a measure of interestingness in knowledge discovery," in *Decision Support Systems* (27), pp. 303–318.

Park, J. S., Chen, M. S., and Yu, P. S. (1995a) "An Effective Hash-Based Algorithm for Mining Association Rules," *in Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, San Jose, CA, pp. 175–186.

Park, J. S., Chen, M. S., and Yu, P. S. (1995b) "Efficient Parallel Data Mining for Association Rules," in *Proc. of the 4th Int'l Conf. on Information and Knowledge Management*, Baltimore, MD, pp. 31–36.

Piatetsky-Shapiro, G. and Frawley, W. J. (Eds.) (1991) *Knowledge Discovery in Databases*, Menlo Park, CA; Cambridge, MA: AAAI/MIT Press.

Povinelli, R. J. (2000) "Identifying Temporal Patterns for Characterization and Prediction of Financial Time Series Events," in *Proc. of TSDM'00,* pp. 46–61.

Prodromidis, A. L., Chan, P. K., and Stolfo, S. J. (2000) "Met-Learning in Distributed Data Mining Systems: Issues and Approaches," in H. Kargupta and P. Chan (Eds.), *Advances in Distributed and Parallel Knowledge Discovery*, Menlo Park, CA; Cambridge, MA: AAAI/MIT Press, pp. 79–112.

Ribeiro, J., Kaufman, K., and Kerschberg, L. (1995) "Knowledge Discovery from Multiple Databases," in *Proc. of the 1st Int'l Conf. on Knowledge Discovery and Data Mining*, Montreal, Canada, pp. 240–245.

Roddick, J. and Lees, B. G. (2001) "Paradigms for spatial and spatio-temporal data mining," in *Geographic Data Mining and Knowledge Discovery*, H. Miller and J. Han (Eds), Taylor & Francis.

Roddick, J. F. and Spiliopoulou, M. (2002) "A Survey of Temporal Knowledge Discovery Paradigms and Methods," in *IEEE Trans. on Knowledge and Data Engineering*, vol. 14, no. 4, pp. 750–767.

Roddick, J. F. and Spiliopoulou, M. (1999) "A Bibliography of Temporal, Spatial and Spatio-Temporal Data Mining Research," in *SIGKDD Explorations* 1(1), pp. 34–38.

Roddick, J. F., Hornsby, K., and Spiliopoulou, M. (2000) "An Updated Bibliography of Temporal, Spatial, and Spatio-temporal Data Mining Research," in *Proc. of TSDM'00*, pp. 147–164.

Savasere, A., Omiecinski, E., and Navathe, S. (1995) "An Efficient Algorithm for Mining Association Rules in Large Databases," in *Proc. of the 21st Int'l Conf. on Very Large Data Bases*, Zurich, Switzerland, pp. 432–444.

Shafer, G. (1976) *Mathematical Theory of Evidence*, Princeton, NJ: Princeton University Press.

Shafer, J., Agrawal, R., and Mehta, M. (1996) "SPRINT: A Scalable Parallel Classifier for Data Mining," in *Proc. of the 22nd Int'l Conf. on Very Large Data Bases*, Mumbai (Bombay), India, pp. 544–555.

Silberschatz, A. and Tuzhilin, A. (1996) "What Makes Patterns Interesting in Knowledge Discovery Systems," in *IEEE Trans. on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 970–974.

Silberschatz, A., Stonebraker, M., and Ullman, J. (1996) "Database Research: Achievements and Opportunities into the 21st Century," in *SIGMOD Record*, vol. 25, no. 1, pp. 52–63.

Smyth, P. and Goodman, R. M. (1992) "An Information Theoretic Approach to Rule Induction from Databases," in *IEEE Trans. on Knowledge and Data Engineering*, vol. 4, no. 4, pp. 301–216.

Spiliopoulou, M. and Roddick, J. F. (2000) "Higher Order Mining: Modelling and Mining the Results of Knowledge Discovery," in N. F. F. Ebecken and C. A. Brebbia (Eds.), *Data Mining II – Proc. of the 2nd Int'l Conf. on Data Mining Methods and Databases for Engineering, Finance, and Other Fields,* Southampton, U.K.: WIT Press, pp. 309–320.

Srikant, R. and Agrawal, R. (1995) "Mining Generalized Association Rules," in *Proc. of the 21st Int'l Conf. on Very Large Data Bases,* Zurich, Switzerland, pp. 407–419.

Srikant, R. and Agrawal, R. (1996a) "Mining Quantitative Association Rules in Large Relational Tables," in *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, Montreal, Canada, pp. 1–12.

Srikant, R. and Agrawal, R. (1996b) "Mining sequential patterns: Generalizations and performance improvements," in *Proc. of EDBT'00*, pp.3–17.

Stolorz, P. E., Nakamura, H., Mesrobian, E., Muntz, R. R., Shek, E. C., Santos, J. R., Yi, J., Ng, K., Chien, S. Y., Mechoso, C. R., and Farrara, J. D. (1995) "Fast Spatio-Temporal Data Mining of Large Geophysical Datasets," in *Proc. of KDD'95*, pp. 300–305.

Suzuki, E. (1997) "Autonomous discovery of reliable exception rules," in *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97).*

Tan, P. N., Kumar, V., and Srivastava, J. (2002) "Selecting the right interestingness measure for association patterns," in *Proc. of ACM SIGKDD'02*, pp. 31–41.

Toivonen, H. (1996) "Sampling Large Databases for Association Rules," in *Proceedings of the 22nd International Conference on Very Large Data Bases*, San Francisco, pp. 134-145, 1996.

Tsoukatos, I. and Gunopulos, D. (2001) "Efficient Mining of Spatiotemporal Patterns," in *Proc. of SSTD'01*, pp.425–442.

Tung, A. K. H., Hou, J., and Han, J. (2001) "Spatial Clustering in the Presence of Obstacles," in *Proc. of ICDE'01*, pp. 359–367.

Tung, A. K. H., Ng, R. T., Lakshmanan, L. V. S., and Han, J. (2000) "Geo-Spatial Clustering with User-Specified Constraints," in *Proc. of MDM/KDD'00*, pp. 1–7.

Wrobel, S. (1997) "An Algorithm for Multi-Relational Discovery of Subgroups," in *Proc. of the 1st European Symp. on Principles of Data Mining and Knowledge Discovery,* Trondheim, Norway, pp. 367–375.

Wu, X. and Zhang, S. (2003) “Synthesizing High-Frequency Rules from Different Data Sources,” in *IEEE Trans. on Knowledge and Data Engineering*, vol. 15, no. 2, pp. 353–367.

Yager, R. R. (1991) “On Linguistic Summaries of Data,” in [Piatetsky-Shapiro and Frawley 1991], pp. 347–363.

Yao, J. and Liu, H. (1997) “Searching Multiple Databases for Interesting Complexes,” in *Proc. of the 1st Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, Singapore, pp. 198–210.

Zadeh, L. A. (1965) *Fuzzy Sets.* Information and Control, 8, pp. 338-353.

Zaki, M., Parthasarathy, S., Ogihara, M., and Li, W. (1997) “New algorithms for fast discovery of association rules,” in *Proc. of KDD’97*.

Zaki, M. (2001) “Efficient enumeration of frequent sequences,” in *Proc. of CIKM’01*, pp. 68–75.

Zhang, B., Hsu, M., and Dayal, U. (2000) “K-Harmonic Means - A Spatial Clustering Algorithm with Boosting,” in *Proc. of TSDM’00*, pp. 31–45.

Zhang, S., Wu, X., and Zhang, C. (2003) “Multi-Database Mining,” in *IEEE Computational Intelligence Bulletin*, vol. 2, no. 1, pp. 5–13.

Zhang, S., Zhang, C., and Wu, X. (2004) *Knowledge Discovery in Multiple Databases*, London, U.K.: Springer-Verlag.

Zhang, T., Ramakrishnan, R., and Livny, M. (1996) “BIRCH: An Efficient Data Clustering Method for Very Large Databases,” in *Proc. of the ACM SIGMOD Int’l Conf. on Management of Data*, Montreal, Canada, pp. 103–114.

Zhong, N., Yao, Y., and Ohsuga, S. (1999) “Peculiarity Oriented Multi-Database Mining,” in *Proc. of the 3rd European Conf. on Principles of Data Mining and Knowledge Discovery*, Prague, Czech Republic, pp. 136–146.