Discovery of Unexpected Sequences: from Sequential Patterns to Implication Rules

Haoyuan Li

▶ **To cite this version:**

Haoyuan Li. Discovery of Unexpected Sequences: from Sequential Patterns to Implication Rules. Human-Computer Interaction [cs.HC]. Université Montpellier II - Sciences et Techniques du Languedoc, 2009. English. NNT : . tel-00431117

HAL Id: tel-00431117
https://theses.hal.science/tel-00431117

Submitted on 10 Nov 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Université Montpellier II
— Sciences et Techniques du Languedoc —

Thèse

pour obtenir le grade de
Docteur de l'Université Montpellier II

Discipline : Informatique
Spécialité Doctorale : *Informatique*
Ecole Doctorale : *Information, Structure, Systèmes*

présentée et soutenue publiquement par

Dong Li
le 10 septembre 2009

Extraction de séquences inattendues : des motifs séquentiels aux règles d'implication

Jury

Giuseppe Attardi, Professeur, Università di Pisa, ... Examinateur
Anne Laurent, Maître de Conférences, Université Montpellier 2, ... Co-directrice de Thèse
Trevor Martin, Professeur, University of Bristol, ... Rapporteur
Stan Matwin, Professeur, University of Ottawa, ... Rapporteur
Jacky Montmain, Professeur, École des Mines d'Alès, ... Président
Pascal Poncelet, Professeur, Université Montpellier 2, ... Directeur de Thèse
Mathieu Roche, Maître de Conférences, Université Montpellier 2, ... Examinateur

Remerciements

Je tiens à exprimer ma profonde gratitude à mes encadrants, Prof. Pascal Poncelet et Dr. Anne Laurent pour leurs encouragements, conseils, confiance et soutien au cours de mes études supérieures à l'Université de Montpellier 2. Comme directeurs de ma thèse, ils m'ont guidé afin que je puisse acquérir les compétences nécessaires dans ma carrière universitaire.

C'est un grand honneur d'avoir Prof. Trevor Martin et Prof. Stan Matwin comme rapporteurs, Prof. Giuseppe Attardi, Prof. Jacky Montmain, et Dr. Mathieu Roche comme membres du jury, je tiens à les remercier pour leurs précieux conseils et suggestions constructives.

Mes plus sincères remerciements à Dr. Maguelonne Teisseire, qui m'a encadré et soutenu lors de mes premiers pas dans le domaine de la recherche et plus particulièrement de la fouille de données. Je suis également très reconnaissant au Prof. Stefano A. Cerri pour ses conseils aimables et utiles sur la recherche et le monde académique.

Je me suis senti honoré d'être membre de l'équipe TATOO du Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier, où j'ai pu bénéficier de discussions enrichissantes avec Sandra Bringay, Céline Fiot, Lisa Di Jorio, Cécile Low-Kam, Yoann Pitarch , Marc Plantevit, Julien Rabatel, Chedy Raïssi, Paola Salle, et Hassan Saneifar.

Je voudrais également exprimer mes remerciements à Françoise Armand, Gérard Dray, et François Trousset du Laboratoire de Génie Informatique et d'Ingénierie de Production de l'Ecole des Mines d'Alès.

Enfin, je suis profondément reconnaissant à mes parents et à ma femme, pour leur amour, leur foi et leur soutien au cours de toutes ces années d'étude. Je leur dédie cette thèse.

Le manuscrit est redigé en anglais.

Ph.D. Thesis

by

Dong (Haoyuan) Li

Discovery of Unexpected Sequences: from Sequential Patterns to Implication Rules

September, 2009

Université Monpellier 2, France

Abstract

The sequential patterns can be viewed as an extension of the notion of association rules with integrating temporal constraints, which are effective for representing statistical frequency based behaviors between the elements contained in sequence data, that is, the discovered patterns are interesting because they are frequent. However, with considering prior domain knowledge of the data, another reason why the discovered patterns are interesting is because they are unexpected. In this thesis, we investigate the problems in the discovery of unexpected sequences in large databases with respect to prior domain expertise knowledge. We first methodically develop the framework MUSE with integrating the approaches to discover the three forms of unexpected sequences. We then extend the framework MUSE by adopting fuzzy set theory for describing sequence occurrence. We also propose a generalized framework SOFTMUSE with respect to the concept hierarchies on the taxonomy of data. We further propose the notions of unexpected sequential patterns and unexpected implication rules, in order to evaluate the discovered unexpected sequences by using a self-validation process. We finally propose the discovery and validation of unexpected sentences in free format text documents. The usefulness and effectiveness of our proposed approaches are shown with the experiments on synthetic data, real Web server access log data, and text document classification.

Keywords : Knowledge discovery in databases, data mining, sequence database, interestingness measure, belief, unexpected sequences, sequential patterns, sequence rules, fuzzy logic, hierarchy, validation, text classification.

To my family.

The Tao that can be described is not the enduring and unchanging Tao.
The name that can be named is not the enduring and unchanging name.
— Lao Tsi

Acknowledgements

I wish to express my deep gratitude to my supervisors, Prof. Pascal Poncelet and Dr. Anne Laurent for their continuous encouragement, guidance, confidence and support during my graduate studies at the University of Montpellier 2. As directors of my thesis, they taught me practices and skills needed in my academic career.

It is my great honor to have Prof. Trevor Martin and Prof. Stan Matwin as reporters, Prof. Giuseppe Attardi, Prof. Jacky Montmain, and Dr. Mathieu Roche as committee members. I would like to thank them for their valuable and constructive suggestions and feedbacks.

My deepest thanks to Dr. Maguelonne Teisseire, who educated me and got me interested in data mining research. I am very grateful to Prof. Stefano A. Cerri for his kind and helpful advices about research and career development.

I felt honored to be a member in the TATOO team of the Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier, where I received many kind helps from Sandra Bringay, Céline Fiot, Lisa Di Jorio, Cécile Low-Kam, Yoann Pitarch, Marc Plantevit, Julien Rabatel, Chedy Raïssi, Paola Salle, and Hassan Saneifar.

I also express my distinguish thanks to Françoise Armand, Gérard Dray, and François Trousset in the Laboratoire de Génie Informatique et d'Ingénierie de Production of the École des Mines d'Alès.

Finally, I am deeply indebted to my parents and my wife for their love, faith and support through all these years of study. I dedicate this dissertation to them.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Knowledge discovery in databases (KDD) is the process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data [FPSS96a], which takes account of two objectives: to discover new patterns that can be interpreted as new knowledge of the data, or to verify the hypothesis of users that can be reacted to the discovery.

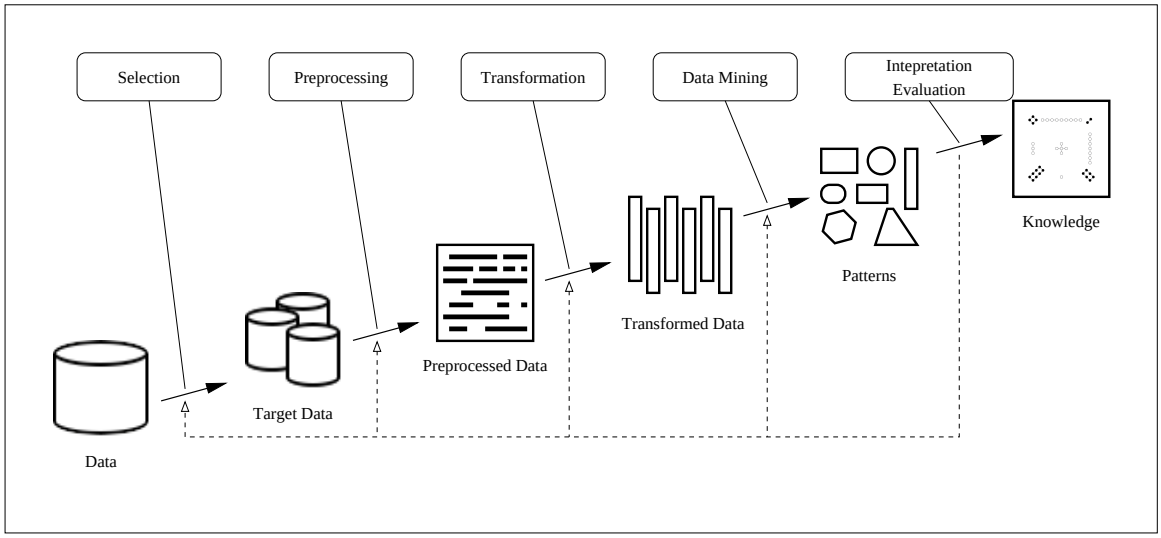


Figure 1.1: A general framework of the knowledge discovery process.

The framework shown in Figure 1.1 illustrates the process of KDD. This process consists of five principal steps: the selection, preprocessing, and transformation of data, the data mining, and the interpretation and evaluation of discovered patterns. In this framework, the *data mining* step plays an essential role, which applies discovery algorithms that produce a particular enumeration of potential interesting patterns in terms of an expression in some language describing a subset of the data or a model applicable to that subset [FPSS96b].

In data mining, the *interestingness* [PSM94, ST95, HH03, McG05] is an important notion that takes an overall measure of pattern value. The measures of interestingness can be categorized into *objective measures* and *subjective measures*, where objective measures rely on the structure of

patterns and the underlying data used in the discovery process, however subjective measures do not depend only on the structure of patterns and the data used in the discovery process, but also on the user who examines the discovered patterns [ST95].

As investigated in many data mining and knowledge discovery literature, one measure is that patterns or rules are interesting because they are *unexpected* to prior user knowledge of the data [PSM94, SS96, LH96, Suz97, PT98, DL98, LHML99, Spi99, PT00, LMY01, WJL03, JS05, PT06]. Most of those existing approaches focus on discovering unexpectedness in the context of association rules [AIS93], however none of the existing approaches deals with unexpected sequences and rules with considering the semantics of data.

Therefore, the discovery of unexpectedness in sequence databases with respect to semantics of data is an under-investigated problem that can be important and interesting for a large number of application domains.

1.1 Motivation

Most real-world applications process the data stored in sequence format, where the elements in data are sequentially ordered with temporal or spatial relation. For examples, in a customer retail database, a sequence can be all purchases of a customer ordered by the time of transaction; in a Web access log file, a sequence can be all of those resources accessed during a user session ordered by the time of request; in a telecommunication network monitoring database, a sequence can be all events during a period ordered by the time of occurrence; in a DNA segment, a sequence is a succession of nucleotide subunits with spatial order, and so on.

A great deal of research work focuses on developing efficient and effective sequential pattern mining algorithms [AS95, SA96b, MCP98, GRS99, PHMAP01, Zak01, AFGY02, YHA03, PHMA$^+$04, WH04, LLT07, PHW07, WHL07, RCP08, Ca09]. With sequential pattern mining, we can extract the sequences reflecting the most frequent behaviors in a database, which can be further interpreted as domain knowledge for variant purposes. However, although mining sequential patterns is essential in most application, the *unexpected sequences* that semantically contradict existing knowledge of data have never less importance when we consider prior knowledge within the data mining process. On the other hand, the term "unexpected" does not mean that such sequences must not be frequent, so that it is very different from non-frequent patterns, such as outliers [KN98, RRS00, JTH01, AP02, BS03, AP05] or rarity [JKA01, Wei04]. In summary, there exist the following two critical problems in finding unexpected sequences in data with frequency based mining methods.

Semantics in Unexpected Sequences. If a sequence is considered as unexpected because it semantically violates a given rule, then frequency based sequence mining approaches are not

applicable to identify such a sequence although it may be extracted.

The redundancy problem is inherent to frequency based data mining methods that they may return an extremely large number of potentially interesting patterns or sequences. Therefore, an unexpected sequence will not appear in the post analysis process except the *minimum support* is no higher than its support value. Further, it might be difficult to seek low frequency unexpected sequences in the post analysis process since the result sequence set may be huge.

Some regular expression based approaches, for instance *SPIRIT* [GRS99] and *MSP-Miner* [dAF05], can find the sequences that respect predefined constraints, however the premise sequence is that the composition of an unexpected sequence must be already known before the extraction and the semantics of unexpectedness cannot be addressed.

Occurrence in Unexpected Sequences. If a sequence is considered as unexpected because it does respect previewed occurrences of sequences (e.g., an *incomplete* or *disordered* subsequence of an expected sequence), then it is impossible to determine such an unexpected sequence with respect to the principle of sequential pattern mining.

In theory, the existence of an unexpected incomplete sequence can be discovered with the *closed sequential pattern* model (*CloSpan* [YHA03]) by computing the difference of support values between closed sequential patterns (as illustrated in the following example), however, unless the structure of an unexpected sequence is known, we have to examine the support values of all the combinations of possible structures of an unexpected sequence to confirm the existence. Nevertheless, even if the existence of unexpected incomplete sequences can be determined, we cannot identify such unexpected sequences for further analysis.

On the other hand, the *gap* (or *distance*) between two subsequences in a sequence is not taken into account in sequential pattern mining, thus an unexpected disordered sequence can never be found by existing approaches.

In this thesis, we investigate the problems of discovering and evaluating unexpected sequences and rules in large sequence databases.

1.2 Contributions

The work presented in this thesis consists of different contributions to the discovery and evaluation of unexpected sequences and unexpected implication rules in sequence databases with respect to the semantics of data.

We investigate the problems including how the unexpectedness can be defined in the context of sequence data mining, how the unexpectedness can be discovered, how to evaluate or validate the discovered unexpected sequences, what implies the unexpectedness, and what the unexpectedness

implies. Moreover, as extensions of the base framework of discovering unexpected sequences, fuzzy set theory [Zad65] and generalizations of data [SA95] are integrated to the unexpected sequence discovery process. We also adapt the discovery and evaluation of unexpected sequence into the context of opinion mining and text classification in terms of exception phrases in free format text documents.

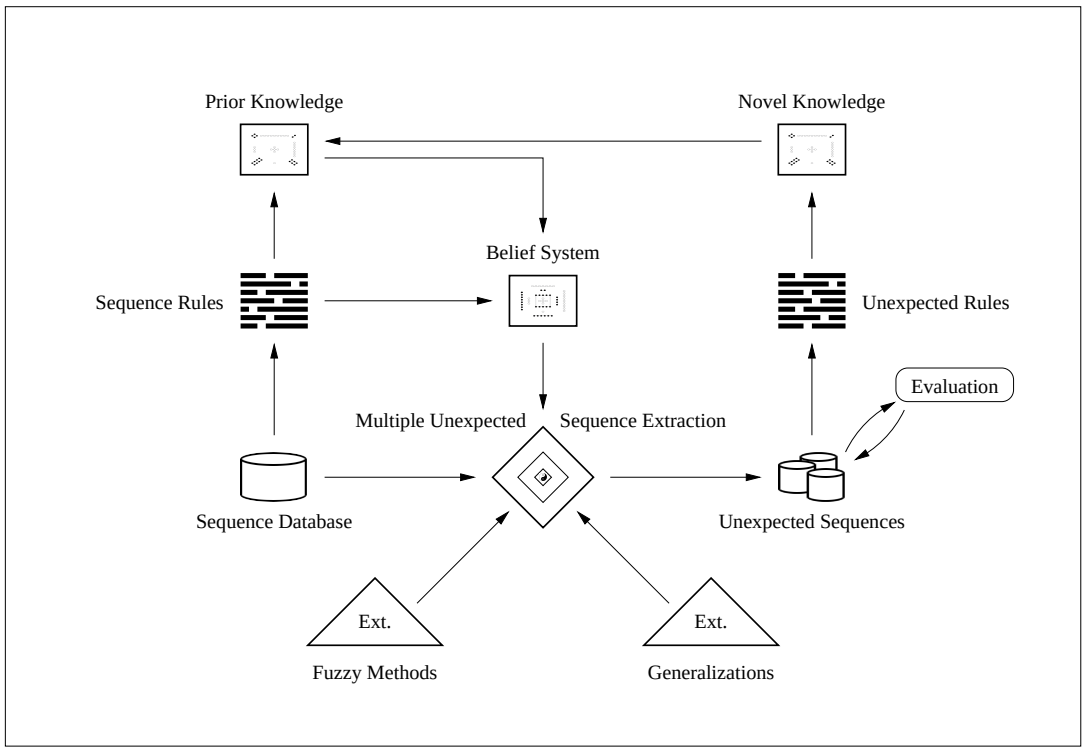


Figure 1.2: Outline of the contributions presented in this thesis.

The work presented in this thesis can be illustrated in Figure 1.2, which include the following contributions.

1. We state the unexpectedness in sequence databases with respect to the belief system constructed from prior knowledge of application domain, where *sequence rules* are essentials. Therefore, in this thesis, we first summarize and formalize two categories of sequence rules, including *sequence association rules* and *predictive sequence implication rules*.

2. We methodically develop a framework, MUSE, for **m**utiple **u**nexpected **s**equence **e**xtraction with respect to a belief system based on sequence rules with integrating *semantic contradictions* of sequence data. The sequence rules can be either discovered by sequence data mining approaches or defined by domain experts. According to different forms of sequence rules, we propose three forms of unexpected sequences with respect to completeness, occurrence, and semantics of sequences.

3. We extend the framework MUSE by adopting fuzzy set theory for describing the unexpectedness on sequence occurrence, which is developed as the approach TAUFU. We also propose

the notion of fuzzy recurrence sequence, with which we further develop the approach UFR to discover unexpected fuzzy recurrences within the framework MUSE.

4. To reduce the complexities in constructing the belief system, we propose a generalization of the sequence rules and semantic contradictions with respect to the concept hierarchies on the taxonomy of data. We also propose the notion of *soft belief* and develop the approach SOFTMUSE to discover *soft unexpected sequences* in hierarchical data, where the belief system consists only of *generalized sequence rules* and a concept hierarchy. The unexpectedness is therefore stated by determining the *relatedness* and *contradiction* with computing the *semantic similarity* between generalized sequence rules with respect to the concept hierarchy.

5. In order to evaluate the discovered unexpected sequences, we propose the notion of *unexpected sequential patterns* for performing a *self-validation* process to the evaluation of unexpected sequences. We also propose three forms of *unexpected implication rules*, including *unexpected class rule*, *unexpected association rule*, and *unexpected occurrence rule*, to study what is associated with the unexpectedness, what implies the unexpectedness, and what the unexpectedness implies.

6. We adapt the notion of unexpected sequence into the context of opinion mining and text classification in terms of unexpected sentences. We propose a *word relatedness* based approach to discover unexpected sentences in free format text documents. We also design a crass-validation based experimental evaluation of unexpected sentences by using text classification methods, which shows that the accuracy of classification can be improved with eliminating unexpected sentences.

1.3 Organization

The rest of this thesis is organized as follows.

- In Chapter 2, we introduce the state-of-the-art of unexpected pattern and rule discovery. We first introduce the interestingness measures for data mining, and then we summarize existing approaches to unexpected pattern and rule discovery.

- In Chapter 3, we formalized two categories of sequence rules. We first introduce existing sequence rule mining approaches, then we propose the form of sequence association rule and propose the notion of predictive sequence implication rule. We also propose the notion of consistent sequence rule set.

- The framework MUSE is proposed in Chapter 4. We first propose a belief system consisting of sequence rules and semantic contradiction between sequences, and then we propose three

forms of unexpected sequences with respect to the different forms of sequence rules. We also outline the framework MUSE with integrating the approaches to discover the three forms of unexpected sequences. The usefulness and effectiveness of the framework MUSE are shown with the experiments on real Web server access records data and synthetic data.

- We propose two fuzzy approaches in Chapter 5 to discover unexpected sequences as extensions of the framework MUSE with fuzzy methods . We first study the fuzzy unexpectedness in sequence occurrence as *tau-fuzzy* unexpected sequences with developing the approach TAUFU. We then propose the notion of unexpected fuzzy recurrence behavior in sequence data with respect to the belief system consists of fuzzy recurrence rules, and the approach UFR is developed to discover unexpected fuzzy recurrences. The approaches TAUFU and UFR are evaluated with the experiments on real Web server access records data.

- To reduce the complexities of constructing the belief system, we generalize the framework MUSE in Chapter 6. We first formalize the hierarchical data model to propose the generalized belief system, which consists of generalized sequence rules and generalized semantic contradiction. We therefore propose the notions of generalized unexpected sequences. As an important improvement of the framework MUSE, in Chapter 6 we also propose the notion of soft belief and soft unexpected sequences in hierarchical data by computing the semantic relatedness and semantic contradiction between generalized sequences, so called the approach SOFTMUSE. Experiments on real Web server access records data shows the performance of discovering soft unexpected sequences.

- We propose the notions of unexpected sequential patterns and unexpected implication rules in Chapter 7. We first propose the notions of unexpected feature and association sequence of unexpected sequences, which we propose the notions of internal and external unexpected sequential patterns with. We can therefore evaluate the quality of discovered unexpected sequences with unexpected sequential patterns by a self-validation process. In this chapter, we also propose the notions of unexpected implication rules, include unexpected class rule, unexpected association rule, and unexpected occurrence rule. Unexpected class rules depict the frequent sequences associated with some unexpectedness; unexpected association rules depict the association relation between the frequent sequences contained in unexpected features and association sequences; unexpected occurrence rules further include antecedent rules and consequent rules, which depict what frequently happens before and after the occurrence of unexpectedness. We evaluate the discovery of unexpected sequential patterns and unexpected implication rules in experiments on discovering unexpected Web usage.

- As a derived approach, in Chapter 8 we propose the discovery and evaluation of unexpected sentences in free format text documents. In this chapter, we first present the part-of-speech

data model of free format text documents, and then we present the discovery of opposite sentiments in the context of opinion mining. We then generalize this approach to general text classification problem, where we propose the notions of unexpected sentences, which semantically contradict the class descriptors extracted from training documents. We also design the extraction and validation of unexpected sentences contained in text documents, where experimental results show that the accuracy of classification can be improved with eliminating unexpected sentences.

- Finally, in Chapter 9, we summarize the work presented in this thesis and propose the perspectives of our future research directions.

Chapter 2

Related Work

In this chapter, we introduce the related work on interestingness measures for data mining and the discovery of unexpected patterns and rules.

2.1 Introduction

In data mining, the *interestingness* [PSM94, ST95, HH99a, HH03, McG05, GH06] is an important notion that takes an overall measure of pattern value with combining validity, novelty, usefulness, and simplicity, where a *pattern* is an expression in some language describing a subset of the data or a model applicable to that subset [FPSS96b]. One reason of patterns or rules being valuable is because they are *unexpected* to prior user knowledge of the data [PSM94, SS96, LH96, BT97, Suz97, PT98, DL98, LHML99, Spi99, HLSL00, PT00, LMY01, WJL03, JS05, PT06].

Before introducing the state-of-the-art of interestingness measures and unexpected pattern and rule discovery, we first formalize the data model considered in this thesis and related work as follows.

Let $R = \{i_1, i_2, \ldots, i_n\}$ be a finite set of n binary-valued attributes, an *item* is an attribute $i_j \in R$. An *itemset* is an unordered collection $I = (i_1 i_2 \ldots i_m)$ of distinct items sorted by lexical order, where $i_j \in R$ is an item. A itemset is also called as a *pattern*. A *transactional database* is a large set $\mathcal{D}$ of *transactions*, where each transaction is an itemset. If a pattern X is a subset of a transaction $\mathcal{I}$, that is, $X \subseteq \mathcal{I}$, then we say that $\mathcal{I}$ *supports* X.

An *association rule* is a rule in the form $X \rightarrow Y$ contained in a transactional database, where $X \cap Y = \emptyset$ are two patterns, which depicts that if the pattern X occurs in a transaction, then the pattern Y also occurs in the same transaction. Association rules are measured by *support* and *confidence*. Given an association rule $X \rightarrow Y$ and a database $\mathcal{D}$, the support of the rule is defined as

$$supp(X \rightarrow Y, \mathcal{D}) = \frac{|\{\mathcal{I} \in \mathcal{D} \mid X \cup Y \subseteq \mathcal{I}\}|}{|\mathcal{D}|}, \tag{2.1}$$

that is, the total number of transactions contained in the database that support the pattern $X \cup Y$ on the total number of transactions contained in the database; the confidence of the rule is defined as

$$conf(X \rightarrow Y, \mathcal{D}) = \frac{|\{\mathcal{I} \in \mathcal{D} \mid X \cup Y \subseteq \mathcal{I}\}|}{|\{\mathcal{I} \in \mathcal{D} \mid X \subseteq \mathcal{I}\}|}, \tag{2.2}$$

that is, the total number of transactions contained in the database that support the pattern $X \cap Y$ on the total number of transactions support the pattern X.

A *sequence* is an ordered list $s = \langle I_1 I_2 \dots I_k \rangle$ of itemsets, where I_j is an itemset. A *sequence database* is a large set of sequences, where each sequence has a unique identification and two different sequences can contain the same ordered list of itemsets. A sequence database can be regarded as a transactional database if we consider each itemset contained in each sequence as a transaction. Therefore we also denoted a sequence database as $\mathcal{D}$, and in the rest of this thesis, the term *database* covers both of the notions of transactional database and sequence database.

Given two sequences $s = \langle I_1 I_2 \dots I_m \rangle$ and $s' = \langle I'_1 I'_2 \dots I'_n \rangle$, if there exist integers $1 \leq i_1 < i_2 < \dots < i_m \leq n$ such that $I_1 \subseteq I'_{i_1}, I_2 \subseteq I'_{i_2}, \dots, I_m \subseteq I'_{i_m}$, then s is a *subsequence* of s', denoted as $s \sqsubseteq s'$, and s' is a *super-sequence* of s; we also say that s *is included in* s', or s' *supports* s. Given a sequence database $\mathcal{D}$, if a sequence $s \in \mathcal{D}$ is not included in any other sequence $s' \in \mathcal{D}$, then we say that the sequence s is a *maximal*. The *support* of a sequence s in a database $\mathcal{D}$, denoted as $supp(s, \mathcal{D})$, is the total number of sequences in $\mathcal{D}$ that support s on the total number of sequences in $\mathcal{D}$, that is,

$$supp(s, \mathcal{D}) = \frac{|\{s' \in \mathcal{D} \mid s \sqsubseteq s'\}|}{|\mathcal{D}|}. \tag{2.3}$$

Denote by $supp_{min}$ a user defined support threshold *minimum support*, a sequence s is *frequent* if $supp(s, \mathcal{D}) \geq supp_{min}$. A *sequential pattern* is a frequent sequence that is maximal.

ID	Sequence
s_1	$\langle (a)(b)(c)(d) \rangle$
s_2	$\langle (ab)(ac)(abc)(ab)(ac) \rangle$
s_3	$\langle (abcde)(be) \rangle$
s_4	$\langle (a)(bc)(d)(be)(a)(ef) \rangle$
s_5	$\langle (a)(b)(c)(d)(e)(f) \rangle$

Table 2.1: A sample sequence database.

Example 1 Table 2.1 shows a sequence database $\mathcal{D} = \{s_1, s_2, s_3, s_5, s_5\}$ that contains 5 sequences. Given a minimum support $supp_{min} = 0.5$, $\langle (b)(c) \rangle$ is a frequent sequence since $supp(s, \mathcal{D}) > supp_{min}$; however, $\langle (b)(c) \rangle$ is not a sequential pattern because with $supp_{min} = 0.5$, we have a maximal sequence $s' = \langle (a)(b)(c) \rangle$ where $supp(s', \mathcal{D}) = 0.6$ and $s \sqsubseteq s'$. □

The rest of this chapter is organized as follows. In Section 2.2, we introduce interestingness measures for data mining, which include objective measures and subjective measures. In Section 2.3, we summarize previous approaches to discover unexpected patterns and rules. Section 2.4 is a discussion on unexpected pattern and rule discovery.

2.2 Interestingness Measure

The discovery of unexpectedness depends on prior knowledge of data that indicates what users expect. Thus, in comparison with the data mining methods based on statistical frequency of data, the methods to discover *unexpectedness* contained in data can be viewed as a process using user-oriented *subjective measures* instead of using data-oriented *objective measures*.

The notions of objective measure and subjective measure for finding potentially interesting patterns (and sequential patterns) or rules are addressed in terms of *interestingness measures* for data mining. McGarry systematically studied the development of interestingness measures in [McG05], where objective measures are considered as using the statistical strength (such as *support*) or structure (such as *confidence*) of discovered patterns or rules to assess their degree of interestingness however subjective measures are considered as incorporating user's subjective knowledge (such as *belief*) into the assessment. For instance, in association rule [AIS93] mining, the *support* of a rule is defined from the statistical frequency of the patterns that constitutes the rule and the *confidence* of a rule is defined from the *premise* $\rightarrow$ *conclusion* structure of rules; however, in unexpected pattern [PT98] mining, the assessment is based on the *beliefs* acquired from prior knowledge of domain.

Not limited to be categorized into objective and subjective, the interestingness measures for data mining are various. For instance, in [HH99a, HH99b, HH01], Hilderman and Hamilton studied heuristic measures of interestingness; in [HLSL00], Hussain et al. discussed a relative interestingness measure; in [JS04], Jaroszewicz and Simovici used Bayesian networks as background knowledge for measuring the interestingness of frequent patterns; in [BGGB05], Blanchard et al. proposed information-theoretic based measures to assess association rule interestingness. Hence, the selection of interestingness is also an important problem in data mining [TKS02].

Piatetsky-Shapiro and Matheus [PSM94] noted that objective measures of interestingness may not handle all factors involved in pattern discovery as a complicated process. The subjective measures of interestingness were studied in the context of the Key Findings Reporter (KEFIR), a system for discovering and explaining "key finding" in large relational databases, applied to the analysis of healthcare information. KEFIR first classifies all possible findings into a predefined set of types, then defines a production rule for each type of findings that specifies the actions to be

taken to indicate how to bring "abnormal" indicators back to their norm. Further, domain experts need to assign a probability of success to the actions in the rule. Finally, the estimated benefit of taking the action for the selected rule is computed as a measure of interestingness. This method provides a good process for defining a subjective measure of interestingness around the correct actions of interest to users.

Silberschatz and Tuzhilin [ST95, ST96] studied subjective measures of interestingness in a domain-independent context. In [ST95], subjective measures of interestingness are categorized into *actionability measure* and *unexpectedness measure*. With actionability measure, a pattern is interesting because "the user can do something about it; that is, the user can react to it to his or her advantage"; however, with unexpected measure, a pattern is interesting because "it is surprising to the user". Unexpectedness is defined by the belief system of data, where two types of beliefs are considered: *soft beliefs* and *hard beliefs*. The soft beliefs are the beliefs associated with a degree that can be changed by the discovered new evidences in data, and the Bayesian approach is adopted for updating the degree of belief in [ST95] by computing the conditional probability (in [ST95], more approaches are discussed in computing the degree of belief, including the Dempster-Shafer approach [Sme88], the frequency approach, the statistical approach, etc.). The hard beliefs are the constraints that cannot be changed with new evidences, and if new evidence in data contradicts such beliefs, then must be some mistakes or errors made in acquiring this new evidence.

2.3 Unexpected Patterns and Rules

In the past years, unexpectedness measure has been widely studied in various approaches to pattern and rule discoveries.

Liu and Hsu studied the unexpected structures of discovered rules in [LH96]. In the proposed approach, the existing rules (denoted as E) from prior knowledge are regarded as fuzzy rules by using fuzzy set theory and the newly discovered rules (denoted as B) are matched against the existing fuzzy rules in the post-analysis process. A rule consists of the *condition* and the *consequent*, so that given two rules B_i and E_j, if the conditional parts of B_i and E_j are similar, but the consequents of the two rules are quite different, then it is considered as *unexpected consequent*; the inverse is considered as *unexpected condition*. The computation of the similarity in the matching is based on the attribute name and value. The same techniques are extended to find unexpected patterns in [LHML99].

Moreover, in [LMY01], Liu et al. investigated the problem of finding unexpected information in the context of Web content mining. The proposed approach aims to discover the Web pages

relevant but unknown to the user (i.e., *competitor Web site*) with respect to existing knowledge of the user (i.e., *user Web site*), where the vector space model with the TF-IDF (*Term Frequency - Inverse Document Frequency*) weight is used in comparing two Web sites: it first computes the corresponding pages between two Web sites by counting the keywords in the pages, then term weights in both documents are compared in order to obtain unexpected terms, and finally unexpected pages and unexpected concepts are extracted by ranking discovered unexpected terms.

Suzuki et al. systematically studied *exception rules* in the context of association rule mining [SS96, Suz96, Suz97, SK98, HLSL00, SZ05, Suz06]. An association rule can be classified into two categories: a *common sense rule*, which is a description of a regularity for numerous objects, and an *exception rule*, which represents, for a relatively small number of objects, a different, regularity from a common sense rule [SS96, Suz96]. In [SS96, Suz96, Suz97, SK98], the exception rules are considered with respect to the common sense rules within the rule pair $r(\mu, \nu)$ defined as follows:

$$r(\mu,\nu) = \begin{cases} A_\mu & \Rightarrow c \\ A_\mu \wedge B_\nu & \Rightarrow c' \end{cases},$$

where A_μ, B_ν are itemsets and c, c' are items. We follow the notions presented in [SK98], $A_\mu \Rightarrow c$, $A_\mu \wedge B_\nu \Rightarrow c'$, and $B_\nu \Rightarrow c'$ are respectively called a common rule, an exception rule, and a reference rule. Such a rule pair can be interpreted as "if A_μ then c, however if A_μ and B_ν then c'". The discovery of rule pairs $r(\mu, \nu)$ is evolutive from [SS96] to [SK98]. In [SS96], an average compressed entropy (ACE) based approach ACEP, where the average compressed entropy of c and A_μ is defined as

$$ACE(c, A_\mu) = p(c, A_\mu) log_2 \frac{p(c|A_\mu)}{p(c)} + p(\overline{c}, A_\mu) log_2 \frac{p(\overline{c}|A_\mu)}{p(\overline{c})}$$

and the interestingness measure of an exception rule is defined by the average compressed entropy product (ACEP) of the rule pair is defined as

$$ACEP(c, A_\mu, c', B_\nu) = ACE(c, A_\mu) \cdot ACE(c', A_\mu \wedge B_\nu).$$

It is not difficult to see that according to the above manner, the an exception rule holds a relatively small number of examples (i.e., low support) in a database. In order to reduce the number of potential interesting exception rules, the notions of *reliable exception rules* and *surprising exception rules* are addressed in [Suz97] and [SK98] based on probabilistic and statistic models. The notion of rule pair is extended to rule triplet in [SZ05, Suz06], where a *negative rule* is regarded as the reference rule. According to the above form of a rule pair, a rule triplet is represented as

$$(A_\mu \Rightarrow c, A_\mu \wedge B_\nu \Rightarrow c', B_\nu \not\Rightarrow c'),$$

where the rule $B_\nu \not\Rightarrow c'$ is the reference rule. In summary, the discovery of exception rules proposed by Suzuki et al. are probabilistic approach based, where the performance is dependent on the

selection of c. The advantage of Suzuki's approaches is that they can discover highly unexpected patterns since it also discovers common sense rules.

In [DL98], Dong and Li proposed neighborhood-based interestingness in association rules, which is based on the distance between rules and the neighborhoods of rules. The neighborhood-based interestingness of a rule is defined in terms of the pattern of the fluctuation of confidences or the density of discovered rules in some of its neighborhoods. The distance between rules is studied in the syntax: given two rules $r_1 = X_1 \Rightarrow Y_1$ and $r_2 = X_2 \Rightarrow Y_2$, the syntax distance between r_1 and r_2 is defined as

$$dist_{iset}(r_1, r_2) = \delta_1 \left|(X_1Y_1) \ominus (X_2Y_2)\right| + \delta_2 \left|X_1 \ominus X_2\right| + \delta_3 \left|Y_1 \ominus Y_2\right|,$$

where $X \ominus Y$ denotes the symmetric difference between two itemsets X and Y (i.e., $(X-Y) \cup (Y-X)$), and $\delta_1, \delta_2, \delta_3$ are non negative real numbers that reflect users' preferences of the contributions of itemsets. The *k-neighborhood* ($k > 0$) of a rule r, denoted as $N(r, k)$, is therefore defined as the set

$$N(r, k) = \{r' \mid dist_{iset}(r, r') \leq k\}.$$

Suppose M is a set of discovered rules and $r \in M$ is a reference rule, the *average confidence* of the k-neighborhood of r is defined as the average confidence $avg(r, k)$ of the rules in the set $M \cap N(r, k) - \{r\}$; the *standard deviation* of the k-neighborhood of r is defined as the standard deviation $std(r, k)$ of the rules in the set $M \cap N(r, k) - \{r\}$. So that if the value $|(|conf(r) - avg(r, k)|) - std(r, k)|$ is larger than a given threshold, then the rule r is said to be interesting with *unexpected confidence* in its k-neighborhood.

Padmanabhan and Tuzhilin proposed a semantics-based belief-driven approach [PT98, PT00, PT02, PT06] to discover unexpected patterns (rules[1]) in the context of association rules. In [PT98], Padmanabhan and Tuzhilin first proposed that a rule $A \Rightarrow B$ is *unexpected* with respect to a belief $X \Rightarrow Y$ in a given database $\mathcal{D}$ if: (1) $B \wedge Y \models FALSE$, which means that the two patterns B and Y logically contradict each other (i.e., $\nexists R$ in $\mathcal{D}$ such that $B \cup Y \subseteq R$); (2) $A \wedge X$ holds on a statistically large subset of tuples in $\mathcal{D}$ (e.g., with respect to a given minimum support, the pattern $A \cup X$ is frequent in the database $\mathcal{D}$); (3) the rule $A \wedge X \Rightarrow B$ holds and the rule $A \wedge X \Rightarrow Y$ hoes not hold (e.g., the support and confidence of $A \wedge X \Rightarrow B$ satisfy given minimum support and minimum confidence but those of $A \wedge X \Rightarrow Y$ do not). An example can be that given a belief `professional` $\Rightarrow$ `weekend` (professionals shopped on weekends), if the rule (`professional, December`) $\Rightarrow$ `weekday` (professionals shopped on weekdays in December) holds but the rule (`professional, December`) $\Rightarrow$ `weekend` (professionals shopped on weekends in December) does not, then the rule `December` $\Rightarrow$ `weekday` is unexpected relative to the belief `professional` $\Rightarrow$

[1] In [PT98, PT00, PT02, PT06], Padmanabhan and Tuzhilin use the terms *pattern* and *rule* interchangeably.

`weekend`. Notice that in this approach, the logically contradiction between patterns is defined by domain experts.

In [PT00, PT06], the *minimal set* of unexpected patterns (rules) is addressed and the refinement of beliefs by discovered unexpected patterns is further proposed in [PT02]. The notion of minimal set of unexpected patterns is defined based on the *monotonicity assumption* $\models_M$ of rules, that is, rule $(A \Rightarrow B) \models_M (C \Rightarrow D)$ if $A \subseteq C$ and $B = D$. Then, given a rule set $\mathcal{R}$, the set $\mathcal{R}'$ is the minimal set of $\mathcal{R}$ if and only if the following conditions hold: (1) $\mathcal{R}' \subseteq \mathcal{R}$; (2) $\forall r \in \mathcal{R}, \exists r' \in \mathcal{R}'$ such that $r' \models_M r$; (3) $\forall r'_1, r'_2 \in \mathcal{R}', r'_1 \not\models_M r'_2$. The computational task is therefore to discover the minimal set of rules in all discovered unexpected patterns (rules).

In [Spi99], Spiliopoulou presented a belief-driven approach to find unexpected sequence rules based on the notion of *generalized sequences*[2]. A generalized sequence (or *g-sequence*) is a sequence in the form $g_1 * g_2 * \ldots * g_n$, where $g_1, g_2, \ldots, g_n$ are elements contained in the sequence (e.g., itemsets) and $*$ is a wild-card (i.e., unknown elements). A *sequence rule* is then built by splitting a given g-sequence into two adjacent parts: *premise* (lhs) and *conclusion* (rhs), denoted as $lhs \hookrightarrow rhs$. Further, a belief over g-sequences is defined as a tuple $\langle lhs, rhs, CL, C \rangle$, where $lhs \hookrightarrow rhs$ is a sequence rule, CL is a conjunction of constraints on the frequency of lhs, and C is a conjunction of constraints on the frequency of elements in lhs and rhs. For example, a belief in the above form can be given as $\langle a * b, c, CL, C \rangle$ with $CL = (support(a*b) \geq 0.4 \wedge confidence(a, b) \geq 0.8)$ and $C = (confidence(a*b, c) \geq 0.9)$. That is, the belief proposed by Spiliopoulou is based on the statistical frequency of the elements contained in a g-sequence with respect to a predefined structure (e.g., $a * b \hookrightarrow c$). Let $\mathcal{B}$ be a collection of predefined beliefs and $r = lhs \hookrightarrow rhs$ be a sequence rule discovered in a given database, then r is *expected* if there exists a belief $b = \langle lhs', rhs, CL, C \rangle \in \mathcal{B}$ such that r and b can be matched by verifying CL and C; otherwise the rule r is *unexpected*.

In [WJL03], Wang et al. studied unexpected association rules with respect to the value of attributes. In this approach, a rule is addressed in the form

$$A_1 = a_1, A_2 = a_2, \ldots, A_k = a_k \Rightarrow C = c,$$

where A_i is a non-target attribute, a_i is a domain value for A_i, C is the target attribute, and c is a domain value for C. $A_1 = a_1, A_2 = a_2, \ldots, A_k = a_k$ is called the *body* and $C = c$ is called the *head* of the rule, respectively denoted $b(r)$ and $h(r)$ of a given rule r. Given a database, a tuple *matches* a rule r if $b(r)$ holds on the tuple; a tuple *satisfies* a rule r if both $b(r)$ and $h(r)$ hold on the tuple. Given a rule r and a data tuple t, the *violation* of r by t, denoted as $v(t, r)$, is defined

[2]Notice that the notion of *generalized sequences* proposed by Spiliopoulou is different from the same term that we will present in Chapter 6: Generalizations in Unexpected Sequence Discovery.

as

$$v(t,r) = \begin{cases} \overline{hm}(t,r) \times bm(t,r) & \text{if } bm(t,r) \geq \sigma \wedge \overline{hm}(t,r) \geq \sigma' \\ 0 & \text{otherwise} \end{cases},$$

where $bm(t,r)$ measures the *body match degree* and $hm(t,r)$ measures the *head match degree* between t and r, $\overline{hm}(t,r) = 1 - hm(t,r)$, and σ, σ' are given thresholds. Further, the user knowledge, denoted as $\mathcal{K}$, is defined from the rules with respect to the preference model, which species the user's knowledge about how to apply knowledge rules to a given scenario or a tuple. Thus, the violation $v_{\mathcal{K}}(t)$ of user knowledge $\mathcal{K}$ by a data tuple t is defined as

$$v_{\mathcal{K}}(t) = agg(\{v(t,r) \mid r \in \mathcal{C}_t\}),$$

where $\mathcal{C}_t$ is the covering knowledge of t (i.e., a set of rules that represent the user knowledge on the data contain t) and agg is a well-behaved aggregate function (i.e., $max(V) \leq agg(V) \leq max(V)$ on a vector V of attribute-value pairs). Therefore, given a database $\mathcal{D}$ and discovered rule r, let S be the set of all tuples that satisfy the rule r, the *unexpectedness support* of the rule r is defined as

$$U_{sup}(r) = \frac{\sum\{v_{\mathcal{K}}(t) \mid t \in S\}}{|S|}.$$

Wang et al. further defined the *unexpectedness confidence* and the *unexpectedness* of a rule r as

$$U_{conf}(r) = \frac{U_{sup}(r)}{|\{t \in \mathcal{D} \mid t \text{ satisfies } b(r)\}|}$$

and

$$U_{nexp}(r) = \frac{U_{sup}(r)}{|\{t \in \mathcal{D} \mid t \text{ satisfies } r\}|}.$$

Hence, the problem of mining unexpected rules is to find all rules with respect to user defined thresholds on unexpectedness support, unexpectedness confidence, and unexpectedness.

In [BT97], Berger and Tuzhilin discussed the notion of unexpected patterns in infinite temporal databases, where the unexpectedness is determined from the occurrences of a pattern. In [JS05], Jaroszewicz and Scheffer proposed a Bayesian network based approach to discover unexpected patterns, that is, to find the patterns with the strongest discrepancies between the network and the database. These two approaches can also be regarded as frequency based, where unexpectedness is defined from whether itemsets in the database are much more, or much less frequent than the background knowledge suggests.

2.4 Discussion

In this chapter we introduced interestingness measures for data mining, and summarized the previous approaches to unexpected pattern and rule discovery.

As listed in Table 2.2, most of the existing approaches to discover unexpected patterns and rules are essentially considered within the context of association rules. To the best of our knowledge, before our work, the approaches proposed in [Spi99] and [BT97] are the only ones that concentrates on sequence data. Indeed, although this work considers the unexpected sequences and rules, it is however very different from our problem in the measures and the notions of unexpectedness contained in sequence data.

Approach	Data Model	Measure	Unexpected Structure
[LH96]	Association rule	Pattern similarity	Association rule
[SS96] – [SZ05]	Association rule	Probabilistic	Association rule
[BT97]	Sequence	Propositional Temporal Logic	Pattern
[DL98]	Association rule	Distance + Frequency	Rule confidence
[PT98] – [PT06]	Association rule	Belief/Semantics	Association rule
[LHML99]	Association rule	Pattern similarity	Pattern
[Spi99]	Sequence	Belief/Frequency	Sequence rule
[LMY01]	Text, Web content	VSM/TF-IDF	Text/Term
[WJL03]	Association rule	Rule match	Association rule
[JS05]	Frequent pattern	Bayesian network/Frequency	Pattern

Table 2.2: A comparison of unexpected pattern and/or rule mining approaches.

In this thesis, the unexpectedness is stated by the semantics of sequence data, instead of the statistical frequency or distance.

We consider the *unexpectedness* within the context of domain knowledge and the aspect *valid* within the context of the classical notions of support and confidence. With summarizing previous approaches, we can find that the detection of unexpectedness is often based on rules, that is, the unexpectedness is considered as the facts that contradict existing rules on data. Therefore, before proposing the notions of unexpected sequences with respect to the belief system on sequence data, in the next chapter, we above all formalize the forms of sequence rules.

Chapter 3

Sequence Rules

In most literatures, unexpectedness is the facts that contradict existing rules on data. Therefore, in order to define the belief system on sequence data for discovering unexpected sequences, in this section, we formalized the notions of sequence rules.

3.1 Introduction

Rule mining is an important topic in data mining research and applications, where the most studied problem is mining association rules [AIS93, AS94, SA95, BMUT97, CA97, DL98, GKM+03, DP06, CG07, HCXY07, CW08, KZC08], which finds frequent association relations between the patterns contained in transactional databases. In this chapter, we formalized two categories of sequence rules, which stand for the fundamentals of the belief system proposed in this thesis. We consider two types of sequences rules in beliefs: *non-predictive sequence rules* without occurrence constraint and *predictive sequence rules* with occurrence constraint, where we propose the forms of *sequence association rules* and *predictive sequence implication rules*.

The sequence rule mining problems and the definitions of sequence rules are very variant in comparison with association rules.

Mannila proposed the notion of *episode rules* [MTV97] that can be regarded as in the form $s_\alpha \rightarrow s_\beta$ of sequence association rules, where the constraint $s_\alpha \sqsubseteq s_\beta$ is applied on episode rules for depicting that the occurrence of the sequence s_α implies the occurrence of the sequence s_β.

In [DLM+98], time and occurrence constraints are applied to sequence rules to analyze time series. Two basic forms [1] on the shapes discretized from time series are proposed: (1) $A \stackrel{t}{\Longrightarrow} B$, which depicts that "if shape A occurs, then shape B occurs within time t"; (2) $A_1 \wedge A_2 \wedge \ldots \wedge A_m \stackrel{v,t}{\Longrightarrow} B$, which depicts that "if shapes A_1, A_2, $\ldots A_m$ occur within v units of times, then shape B occurs within time t". In [HD04], the constraint on time lags, which is similar to the form proposed in

[1] We use the notations $\rightarrow$ or $\Rightarrow$ for denoting a sequence rule as how it was defined the original literature.

[DLM+98], is applied to episode rules [MTV97].

In [CCH02], sequence rules are considered with intermediate elements (e.g., $\langle(ab) * (c)\rangle$ where $*$ denote a sequence of unknown items between (ab) and (c) in two directions: a forward rule $s_\alpha \rightarrow s_\beta$ depicts that the occurrence of $s_\alpha \sqsubseteq s$ implies the occurrence of $s_\alpha \cdot s_\beta \sqsubseteq s$, and a backward rule $s_\alpha \leftarrow s_\beta$ depicts that the occurrence $s_\beta \sqsubseteq s$ implies the occurrence of $s_\alpha \cdot s_\beta \sqsubseteq s$

In [HS05], a sequence rule form $s_\alpha \underset{w}{\overset{t}{\Longrightarrow}} s_\beta$ is proposed in the context of evolutionary computing and genetic programming with a specialized pattern matching hardware from time series, where w is the minimum distance and t is the maximum distance between the sequences s_α and s_β contained in a rule; in [LKL08], the sequence rules in the form $s_\alpha \rightarrow s_\beta$ is studied in terms of recurrent rules.

Therefore, in order to benefit from existing approaches to sequence rule mining, in this chapter, we formalize sequence rules in terms of the notions of *sequence association rules* and *predictive sequence implication rules*.

In this thesis, based on the sequence data model introduced in Section 2.1, we further consider the following supplementary concepts and operations on sequence data.

The *length* of a sequence s is the number of itemsets contained in s, denoted as $|s|$; the *size* of a sequence s is the number of all items contained in s, denoted as $\|s\|$. An *empty sequence* is denoted as $\emptyset$, where $|\emptyset| = \|\emptyset\| = 0$. A sequence of length k is called a *k-length sequence*; a sequence with size n is called a *n-size sequence*, or simply a *n-sequence*.

The *concatenation* of sequences is denoted as $s_1 \cdot s_2$, and the result is the sequence obtained by appending s_2 to the end of s_1, so that we have $|s_1 \cdot s_2| = |s_1| + |s_2|$ and $\|s_1 \cdot s_2\| = \|s_1\| + \|s_2\|$. For example, $\langle(a)(b)\rangle \cdot \langle(b)(c)\rangle = \langle(a)(b)(b)(c)\rangle$.

The rest of this chapter is organized as follows. In Section 3.2, we formalize the form of sequence association rule. In Section 3.3, we propose the notion of predictive sequence implication rule. In Section 3.4 we propose the notion of consistent sequence rule set. Section 3.5 is a discussion on sequence rules.

3.2 Sequence Association Rules

In this section, we formalize the notion of *sequence association rule* with extending the notion of *association rule* to sequence data.

An association rule is therefore a rule in the form $X \rightarrow Y$, where X and Y are two patterns such that $X \cap Y = \emptyset$, which depicts that in a transactional database, if the pattern X occurs in a

transaction $\mathcal{I}$, then the pattern Y occurs in the same transaction, that is,

$$(X \subseteq \mathcal{I}) \Rightarrow (Y \subseteq \mathcal{I}).$$

We can extend the notion of association rules to sequence data as the form $s_\alpha \rightarrow s_\beta$, which depicts that if the sequence s_α occurs as a subsequence in a sequence s, then the sequence s_β occurs as a subsequence in the same sequence s, that is,

$$(s_\alpha \sqsubseteq s) \Rightarrow (s_\beta \sqsubseteq s).$$

Definition 1 (Sequence association rule) *A sequence association rule is a rule in the form $s_\alpha \rightarrow s_\beta$, where s_α, s_β are two sequences.*

As measuring association rules, a sequence association rule $s_\alpha \rightarrow s_\beta$ can be measured by *support* and *confidence* with respect to a database $\mathcal{D}$, denoted as $supp(s_\alpha \rightarrow s_\beta, \mathcal{D})$ and $conf(s_\alpha \rightarrow s_\beta, \mathcal{D})$, which can be defined as

$$supp(s_\alpha \rightarrow s_\beta, \mathcal{D}) = |\{s \in \mathcal{D} \mid s \models (s_\alpha \rightarrow s_\beta)\}|, \tag{3.1}$$

and

$$conf(s_\alpha \rightarrow s_\beta, \mathcal{D}) = \frac{|\{s \in \mathcal{D} \mid s \models (s_\alpha \rightarrow s_\beta)\}|}{|\{s \in \mathcal{D} \mid s_\alpha \sqsubseteq s\}|}. \tag{3.2}$$

In this thesis, a sequence association rule can be either defined by domain experts or discovered from sequence databases. Therefore, different from the constraint $X \cap Y = \emptyset$ on association rules, we do not restrict the intersection subsequences of the sequences s_α and s_β in a sequence association rule $s_\alpha \rightarrow s_\beta$. For instance, in the following sequence sets

$$\mathcal{S}_1 = \left\{ \begin{array}{l} \langle (a)(a)(b)(d) \rangle \\ \langle (a)(a)(b)(d) \rangle \\ \langle (a)(a)(b)(d) \rangle \\ \langle (a) \rangle \\ \langle (a) \rangle \end{array} \right\}, \; \mathcal{S}_2 = \left\{ \begin{array}{l} \langle (a)(b)(c)(a)(b)(d) \rangle \\ \langle (a)(b)(c)(a)(b)(d) \rangle \\ \langle (a)(b)(c)(a)(b)(d) \rangle \\ \langle (a)(b)(c) \rangle \\ \langle (a)(b)(c) \rangle \end{array} \right\}, \text{ and } \mathcal{S}_3 = \left\{ \begin{array}{l} \langle (a)(b)(c)(a) \rangle \\ \langle (a)(b)(c)(a) \rangle \\ \langle (a)(b)(c)(a) \rangle \\ \langle (a)(b)(c) \rangle \\ \langle (a)(b)(c) \rangle \end{array} \right\},$$

the rules $\langle (a) \rangle \rightarrow \langle (a)(b)(d) \rangle$, $\langle (a)(b)(c) \rangle \rightarrow \langle (a)(b)(d) \rangle$, and $\langle (a)(b)(c) \rangle \rightarrow \langle (a) \rangle$ can be obtained without difficulty.

Given a sequence association rule $s_\alpha \rightarrow s_\beta$, the sequence s_α is called the *premise sequence* of the rule and the sequence s_β is called the *conclusion sequence* of the rule. Given a sequence s, if $s_\alpha \sqsubseteq s$ and $s_\beta \sqsubseteq s$, then we say that the sequence s *supports* the rule $s_\alpha \rightarrow s_\beta$, denoted as $s \models (s_\alpha \rightarrow s_\beta)$.

On the other hand, many approaches to sequence classification focus on building sequence classifiers [LZO99, XPDY08], where the inverse can be represented as the form $\ell_\alpha \rightarrow s_\beta$ of sequence

rules, which depicts that given a sequence s, if s can be classified under the class labeled by ℓ_α (denoted as $s \vdash \ell_\alpha$), then $s_\beta \sqsubseteq s$. This form of sequence rules can be also consider as sequence association rules if we consider the class label ℓ_α as an element of a sequence.

Example 2 Assume a Web site that supports anonymous user sessions (labeled as `ANON`) and authorized user sessions (labeled as `AUTH`). The rules

$$\texttt{ANON} \rightarrow \langle(\texttt{index})(\texttt{adv})\rangle \quad \text{and} \quad \texttt{AUTH} \rightarrow \langle(\texttt{login})(\texttt{home})\rangle$$

depicts that anonymous users should access `index` and then `adv` however authorized users should access `login` and then `home`. If we consider the labels `ANON` and `AUTH` as itemsets containing one item in user navigation session sequences, then the above rules can be considered as:

$$r_1 = \langle(\texttt{ANON})\rangle \rightarrow \langle(\texttt{index})(\texttt{adv})\rangle;$$
$$r_2 = \langle(\texttt{AUTH})\rangle \rightarrow \langle(\texttt{login})(\texttt{home})\rangle.$$

The following authorized user navigation session sequence

$$s_1 = \langle(\texttt{AUTH})(\texttt{index})(\texttt{login})(\texttt{home})(\texttt{options})(\texttt{save})(\texttt{logout})\rangle$$

is therefore a sequence that supports the rule r_2. Moreover, the sequence association rule

$$r_3 = \langle(\texttt{login})(\texttt{logout})\rangle \rightarrow \langle(\texttt{options})(\texttt{save})\rangle$$

depicts that the access of `login` and then `logout` implies the access `options` and then `save` within a user navigation session. We have that the sequence s_1 supports the rule r_3. □

Given a sequence rule, if the occurrence position of conclusion sequence can be predicted from the occurrence position of premise sequence, then we say that such a rule is *predictive*; otherwise we say that it is *non-predictive*.

Sequence association rules are non-predictive because they only depict the associations between sequences, and there does not exist any constraints on the occurrence of premise and conclusion sequences. Therefore, given a sequence association rule $s_\alpha \rightarrow s_\beta$, we cannot predict the occurrence position of s_β according to the occurrence position of s_α.

3.3 Predictive Sequence Implication Rules

In this section, we formalize the notion of *predictive sequence implication rules* with considering an *occurrence constraint* between the premise and conclusion sequences of a rule.

In comparison with the form $s_\alpha \rightarrow s_\beta$ of sequence association rules, many sequence rule mining approaches take account of various constraints on the occurrence of the sequences s_α and s_β.

Not difficult to see, all these approaches can be categorized into predictive sequence rules since the occurrence position of conclusion sequence can be explicitly determined from the occurrence of premise sequence. In this section, we propose the form of predictive sequence implication rules by taking account of a distance range onto the form $s_\alpha \rightarrow s_\beta$ of sequence association rules, that is, the occurrence position of the sequence s_β is constrained with respect to the occurrence of the sequence s_α. A predictive sequence implication rules is similar in the form to the sequence rules addressed in [HD04], however our formalization is focused on the context of sequence databases and we also consider such a rule in more general cases.

We consider an *occurrence constraint* on sequence association rules $s_\alpha \rightarrow s_\beta$, which is a constraint on the range of the number of itemsets (also called the *distance* or the *gap*) between the sequences s_α and s_β. The notion of predictive sequence implication rules is formally defined as follows.

Definition 2 (Predictive sequence implication rule) *A predictive sequence implication rule is a rule in the form $s_\alpha \rightarrow^\tau s_\beta$, where s_α, s_β are two sequences and $\tau = [min..max]$ is an occurrence constraint such that $min, max \in \mathbb{N}$ and $min \leq max$.*

Given a predictive sequence implication rule $s_\alpha \rightarrow^\tau s_\beta$, the sequence s_α is called the *premise sequence* of the rule and the sequence s_β is called the *conclusion sequence* of the rule. A predictive sequence implication rule $s_\alpha \rightarrow^\tau s_\beta$ ($\tau = [min..max]$) represents that given a sequence s, if the subsequence $s_\alpha \sqsubseteq s$ occurs, then the subsequence $s_\beta \sqsubseteq s$ occurs within a gap range constrained by τ. This relation can be formally represented as $s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s$, where s' is a sequence such that $min \leq |s'| \leq max$ (denoted as $|s'| \models \tau$), that is,

$$(s_\alpha \sqsubseteq s) \Rightarrow (s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s) \wedge (|s'| \models \tau).$$

Hence, given a sequence s, if there exists a sequence s' such that $|s'| \models \tau$ and $s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s$, then we say that the sequence s *supports* the rule $s_\alpha \rightarrow^\tau s_\beta$, denoted as $s \models (s_\alpha \rightarrow^\tau s_\beta)$.

In a predictive sequence implication rule $s_\alpha \rightarrow^\tau s_\beta$ ($\tau = [min..max]$), the integer min is called the *lower bound* of the constraint τ and the integer max is call the *upper bound* of the constraint τ. Moreover, if the upper bound max of τ is not specified, then we note $\tau = [min..*]$ and we write the rule as $s_\alpha \rightarrow^{[min..*]} s_\beta$; if $min = max = 0$, then we note $\tau = 0$ and we write the rule as $s_\alpha \rightarrow^0 s_\beta$; if $min = 0$ and $max = *$, then we note $\tau = *$ and we write the rule as $s_\alpha \rightarrow^* s_\beta$.

When $\tau = *$, we also call such a predictive sequence implication rule $s_\alpha \rightarrow^* s_\beta$ a *simple sequence implication rule* $s_\alpha \rightarrow s_\beta$. A simple sequence implication rule $s_\alpha \rightarrow s_\beta$ represents that given a sequence s, if $s_\alpha \sqsubseteq s$, then $s_\alpha \cdot s_\beta \sqsubseteq s$, that is,

$$(s_\alpha \sqsubseteq s) \Rightarrow (s_\alpha \cdot s_\beta \sqsubseteq s).$$

Without loss of generality, we use the term *sequence implication rule* to cover both the notions of predictive sequence implication rule and simple sequence implication rule.

Example 3 Considering again the context described in Example 2, the simple sequence implication rule

$$r_4 = \langle(\texttt{index})\rangle \rightarrow^* \langle(\texttt{logout})\rangle$$

depicts that the access of `index` implies the access of `logout` later. Not difficult to see, the navigation session sequence s_1 addressed in Example 2 also supports the rule r_4. The rule r_4 can be constrained, for example, by $\tau = [1..5]$, that is,

$$r_5 = \langle(\texttt{index})\rangle \rightarrow^{[1..5]} \langle(\texttt{logout})\rangle,$$

which depicts that the gap between the accesses of `index` and `logout` must be in the range [1..5]. Considering the sequence s_1 addressed in Example 2, we have that $s_1 \models r_5$ since within s_1, the length of the subsequence $\langle(\texttt{home})(\texttt{options})(\texttt{save})\rangle$ between $\langle(\texttt{index})\rangle$ and $\langle(\texttt{logout})\rangle$ is 3, which satisfies the constraint $\tau = [1..5]$. However, if we apply the constraint $\tau' = [1..2]$ to the rule r_4, that is,

$$r_6 = \langle(\texttt{index})\rangle \rightarrow^{[1..2]} \langle(\texttt{logout})\rangle,$$

then the sequence s_1 does not support the rule r_6. □

A predictive sequence implication rule can also be measured by the *support* and *confidence*, which have the same definitions to the support and confidence of a sequence association rule defined in Equation (3.1) and Equation (3.2).

3.4 Consistent Sequence Rule Set

In previous sections, we presented the notions of sequence association rules and predictive sequence implication rules. Denote by $\tau = \emptyset$ the occurrence constraint, the sequence association rules can be represented as the form $s_\alpha \rightarrow^\emptyset s_\beta$.

In the rest of this thesis, we use the term *sequence rule* for describing such a unified form $s_\alpha \rightarrow^\tau s_\beta$ of sequence association rules and predictive sequence implication rules. However, in order to simplify the descriptions, we keep the form $s_\alpha \rightarrow s_\beta$ for denoting a sequence association rule where the occurrence constraint is $\emptyset$.

Given a sequence rule $r = s_\alpha \rightarrow^\tau s_\beta$, a *premise function* $\Lambda(r)$, a *conclusion function* $\Delta(r)$, and an *occurrence function* $\tau(r)$ can be defined to return the premise sequence, the conclusion sequence, and the occurrence constraint of the rule r. Based on the premise function, we propose the notion of *consistent sequence rule set* as follows.

Definition 3 (Consistent sequence rule set) *A consistent sequence rule set is a set $\mathcal{R}$ of sequence rules that have the same premise sequence, that is,*

$$\forall r_i, r_j \in \mathcal{R},\ \Lambda(r_i) = \Lambda(r_j).$$

The following examples show the definition of consistent sequence rule set with respect to sequence association rules and predictive sequence implication rules.

Example 4 Given two sets $\mathcal{R}_1, \mathcal{R}_2$ of sequence association rules on sequence class, where

$$\mathcal{R}_1 = \left\{ \begin{array}{l} r_1 : \langle(\texttt{CL1})\rangle \rightarrow \langle(ab)(c)\rangle \\ r_2 : \langle(\texttt{CL1})\rangle \rightarrow \langle(a)(b)(c)(d)\rangle \\ r_3 : \langle(\texttt{CL1})\rangle \rightarrow \langle(abc)\rangle \end{array} \right\} \text{ and } \mathcal{R}_2 = \left\{ \begin{array}{l} r_1 : \langle(\texttt{CL1})\rangle \rightarrow \langle(a)(c)\rangle \\ r_2 : \langle(\texttt{CL2})\rangle \rightarrow \langle(a)(b)(cd)\rangle \\ r_3 : \langle(\texttt{CL1})\rangle \rightarrow \langle(abc)\rangle \end{array} \right\},$$

then the set $\mathcal{R}_1$ is a consistent sequence class rule set however the set $\mathcal{R}_2$ is not consistent since in $\mathcal{R}_2$ we have at least that $\Lambda(r_1) \neq \Lambda(r_2)$. □

Example 5 Given two sets $\mathcal{R}_1, \mathcal{R}_2$ of sequence association rules, where

$$\mathcal{R}_1 = \left\{ \begin{array}{l} r_1 : \langle(e)(f)\rangle \rightarrow \langle(ab)(c)\rangle \\ r_2 : \langle(e)(f)\rangle \rightarrow \langle(a)(b)(c)(d)\rangle \\ r_3 : \langle(e)(f)\rangle \rightarrow \langle(abc)\rangle \end{array} \right\} \text{ and } \mathcal{R}_2 = \left\{ \begin{array}{l} r_1 : \langle(e)(f)\rangle \rightarrow \langle(a)(c)\rangle \\ r_2 : \langle(f)(e)\rangle \rightarrow \langle(a)(b)(cd)\rangle \\ r_3 : \langle(e)(f)\rangle \rightarrow \langle(abc)\rangle \end{array} \right\},$$

then the set $\mathcal{R}_1$ is a consistent sequence association rule set however the set $\mathcal{R}_2$ is not consistent since in $\mathcal{R}_2$ we have at least that $\Lambda(r_1) \neq \Lambda(r_2)$. □

Example 6 Given two sets $\mathcal{R}_1, \mathcal{R}_2$ of sequence rules, where

$$\mathcal{R}_1 = \left\{ \begin{array}{l} r_1 : \langle(e)\rangle \rightarrow^{\emptyset} \langle(ab)(c)\rangle \\ r_2 : \langle(e)\rangle \rightarrow^{*} \langle(a)(b)(c)(d)\rangle \\ r_3 : \langle(e)\rangle \rightarrow^{[2..5]} \langle(abc)\rangle \end{array} \right\} \text{ and } \mathcal{R}_2 = \left\{ \begin{array}{l} r_1 : \langle(e)\rangle \rightarrow^{\emptyset} \langle(a)(c)\rangle \\ r_2 : \langle(f)\rangle \rightarrow^{*} \langle(a)(b)(cd)\rangle \\ r_3 : \langle(e)\rangle \rightarrow^{[2..5]} \langle(abc)\rangle \end{array} \right\},$$

then the set $\mathcal{R}_1$ is a consistent sequence association rule set however the set $\mathcal{R}_2$ is not consistent since in $\mathcal{R}_2$ we have at least that $\Lambda(r_1) \neq \Lambda(r_2)$. □

Given a consistent sequence rule set $\mathcal{R}$, for any rules $r_i, r_j \in \mathcal{R}$, we have that $\Lambda(r_i) = \Lambda(r_j)$, so that we can define the *premise function* $\Lambda(\mathcal{R})$ that returns the *premise sequence* of $\mathcal{R}$, and the *conclusion function* $\Delta(\mathcal{R})$ that returns the *conclusion sequence set* of $\mathcal{R}$, which is defined as

$$\Delta(\mathcal{R}) = \bigcup_{r \in \mathcal{R}} \Delta(r).$$

For instance, the conclusion sequence set of the sequence rule set $\mathcal{R}_1$ in Example 6 is the sequence set $\{\langle(ab)(c)\rangle, \langle(a)(b)(c)(d)\rangle, \langle(abc)\rangle\}$.

3.5 Discussion

In this chapter, we normalized the forms of sequence rules with proposing the forms of sequence association rule and predictive sequence implication rule. We further proposed the notion of consistent sequence rule set, where all sequence rules share the same premise sequence.

The discovery of sequence rules can be handled in different manners, where to reduce the combinations of sequences with the constraints on the rule structure is a core problem. For instance, in mining episode rules [MTV97], which can be represented in the form $s_\alpha \rightarrow s_\beta$ of sequence association rules, the condition $s_\alpha \sqsubseteq s_\beta$ is required.

We are currently working on developing a *pattern-growth* [PHW07] based method for mining predictive sequence implication rules. We do not consider only the *support* ad *confidence* as interestingness measures, but also consider the *gap distribution* between the premise and conclusion sequences in the mining process, which specifies the *predictability* of a rule.

We can also apply additional constraints on the sequences s_α and s_β in a sequence rule $s_\alpha \rightarrow^\tau s_\beta$ for reducing the number of rules, such as $s_\beta \not\sqsubseteq s_\alpha$ or $s_\alpha \wedge s_\beta = \emptyset$, where $s_\alpha \wedge s_\beta$ denotes the *intersection* of the sequences s_α and s_β, which is the set of all maximal subsequences of s_α and s_β. For instance, we have that

$$(\langle (d)(ab)(bc)(ac)(e)\rangle \wedge \langle (ae)(cd)(bf)\rangle) = \left\{ \begin{array}{l} \langle (a)(b)\rangle \\ \langle (a)(c)\rangle \\ \langle (d)\rangle \\ \langle (e)\rangle \end{array} \right\}.$$

Not difficult to see, the intersection of two or more sequences is the set of sequential patterns supported by all of those sequences.

In the next chapter, we will propose the framework MUSE, which discovers multiple unexpected sequences with respect to a belief system on sequence data, where each belief consists of a consistent sequence rule set and semantic contradictions between sequences.

Chapter 4

Multiple Unexpected Sequence Extraction

In previous chapters we have introduced the problems stated in discovering unexpectedness in databases and formalized the forms of sequence rules. In this chapter, we present the belief-driven framework MUSE for discovering unexpected sequences with respect to prior knowledge of data.

A part of the work presented in this chapter has been published in the *Actes des 8ièmes Journées Francophones Extraction et Gestion des Connaissances (EGC 2008)*, in the *8th Industrial Conference on Data Mining (Industrial ICDM 2008)*, and in the book *Rare Association Rule Mining and Knowledge Discovery: Technologies for Infrequent and Critical Event Detection (IGI Publishing, 2009)*; has been accepted to be published in the journal *La Revue des Nouvelles Technologies de l'Information (RNTI)* and in the *International Journal of Business Intelligence and Data Mining (IJBIDM)*.

4.1 Introduction

Unexpectedness towards existing knowledge is applicative to broad applications like the discovery of disregarded customer shopping behaviors, analysis of misbehaviors in Web access logs, detection of credit card frauds, study of variations in DNA segments, and so on.

In Chapter 1, we addressed the problems about the semantics and occurrence in unexpected sequences, which can be detailed in the following examples.

Example 7 Let $\mathcal{D}$ be a customer transaction database and assume that we can find the following sequential pattern with the minimum support $supp_{min} = 0.5$:

$$s = \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})(\texttt{Rock-Music})\rangle\,,$$

where $supp(s, \mathcal{D}) = 0.6$. This sequential pattern can be interpreted as "60% of all the customers purchase a `Sci-Fi novel`, then purchase `action and Sci-Fi movies` later, and then purchase a `rock music CD` later". Assume that in the database $\mathcal{D}$, there exist 6% of customers who purchased

a `Sci-Fi novel` then `action and Sci-Fi movies`, would purchase a `classical music CD` instead of a `rock music CD`, then such a behavior can be considered as unexpected to the behavior interpreted from the sequential pattern s. Notice that the unexpectedness is caused by the contradiction between `rock music CD` and `classical music CD`, but not because the frequency is low. In fact, with sequential pattern mining, we are able to find such an unexpected behavior only if the minimum support is no greater than 0.06. However, with $supp_{min} = 0.06$, the result set of all discovered sequential patterns might be huge and that makes it impossible to identify the unexpected behavior. □

Example 8 Let $\mathcal{D}$ be a Web access log database that with the minimum support $supp_{min} = 0.5$, we can find the sequential pattern

$$s = \langle(\texttt{login})(\texttt{list})(\texttt{read})(\texttt{read})(\texttt{logout})\rangle\,,$$

where $supp(s, \mathcal{D}) = 0.8$. This sequential pattern can then be interpreted as "80% of users visit the login page, then visit the message list page, then read messages, and at last logout". Now let the sequential pattern

$$s_0 = \langle(\texttt{login})(\texttt{list})(\texttt{logout})\rangle$$

be an expected access sequence with respect to the workflow of services, where we do not require the access of the page `read` in the workflow since there can be no new unread messages for a user. Assume that the sequence

$$s'_0 = \langle(\texttt{login})(\texttt{logout})\rangle$$

is unexpected to the work flow s_0 and it is caused by errors in listing all messages of a user. Let

$$s_1 = \langle(\texttt{login})(\texttt{list})(\texttt{read})(\texttt{read})(\texttt{logout})\rangle$$

and

$$s_2 = \langle(\texttt{login})(\texttt{options})(\texttt{save})(\texttt{logout})\rangle$$

be two sequential patterns (in order to simplify this example, s_1 and s_2 are not subsequences of one same sequence), then we have that

$$supp(s'_0, \mathcal{D}) \geq supp(s_0, \mathcal{D}) \geq supp(s_1, \mathcal{D}) \text{ and } supp(s'_0, \mathcal{D}) \geq supp(s_2, \mathcal{D}).$$

Assume that s_1 and s_2 are the only sequential patterns other than s_0 that include s'_0, then we can conclude the existence of the unexpected sequence s'_0 if and only if

$$supp(s'_0, \mathcal{D}) > supp(s_1, \mathcal{D}) + supp(s_2, \mathcal{D}).$$

Nevertheless, if s'_0 is unknown, then we have to examine the support values of all possible combinations of subsequences of s_0, s_1 and s_2 for seeking unexpected sequences, and the computational task of identifying unexpected sequences will become extremely hard. □

In order to investigate the unexpectedness mentioned in above problems, in this chapter, we develop a belief-driven framework MUSE (Multiple Unexpected Sequence Extraction) for finding unexpected sequences with respect to prior knowledge on the *occurrence* and the *semantics* of sequences.

In this chapter, we further consider the following supplementary concepts and operations on sequence data.

Given a sequence s, we denote $s^\top$ the first itemset of s and $s_\perp$ the last itemset of s. For two sequences s and s' such that $s \sqsubseteq s'$: we note $s \sqsubseteq^\top s'$ if we have $s^\top \subseteq s'^\top$; note $s \sqsubseteq_\perp s'$ if we have $s_\perp \subseteq s'_\perp$; and note $s \sqsubseteq^\top_\perp s'$ if we have $s^\top \subseteq s'^\top$ and $s_\perp \subseteq s'_\perp$. We denote $s \sqsubseteq_c s'$ that the sequence s is a *consecutive subsequence* of the sequence s', that is, there exist sequences s_a, s_b, and s_c such that $s' = s_a \cdot s_b \cdot s_c$, $|s| = |s_b|$, and $s \sqsubseteq s_b$. For instance, the sequence $\langle (a)(b)(c) \rangle$ is a consecutive subsequence of the sequence $\langle (e)(a)(bd)(cd)(f) \rangle$.

The *subtraction* of two sequences s_1 and s_2 ($s_2 \sqsubseteq s_1$) is denoted as $s_1 \setminus s_2$, and the result is the sequence obtained by removing the first occurrence of s_2 from s_1; if $s_2 \not\sqsubseteq s_1$, then $s_1 \setminus s_2 = s_1$. We have $|s_1 \setminus s_2| \geq |s_1| - |s_2|$ and $\|s_1 \setminus s_2\| = \|s_1\| - \|s_2\|$. For example, $\langle (ab)(bc)(ac)(e) \rangle \setminus \langle (a)(b)(c) \rangle = \langle (b)(c)(a)(e) \rangle$, however $\langle (ab)(bc)(ac)(e) \rangle \setminus \langle (a)(d) \rangle = \langle (ab)(bc)(ac)(e) \rangle$ since $\langle (a)(d) \rangle \not\sqsubseteq \langle (ab)(bc)(ac)(e) \rangle$. The *complete subtraction* of two sequences s_1 and s_2 is denoted as $s_1 \setminus^* s_2$, that is to remove all occurrences of s_2 from s_1, if $s_2 \sqsubseteq s_1$; otherwise $s_1 \setminus^* s_2 = s_1$. For instance, $\langle (ab)(bc)(ac)(b) \rangle \setminus^* \langle (a)(b) \rangle = \langle (b)(c)(c) \rangle$.

The rest of this chapter is organized as follows. In Section 4.2, we propose a belief system of sequence data, based on which we propose the notions of unexpected sequences in Section 4.3. We present the framework MUSE in Section 4.4 and show the experimental results in Section 4.5. Finally, Section 4.6 is a discussion.

4.2 Belief System

In this section, we present the *belief system* on prior knowledge, which is based on sequence rules with integrating semantic contradiction between sequences.

Hence, in our approach, a *belief* specifies that if a sequence s_α occurs, then a sequence s_β will occur with or without an *occurrence constraint* on the gap between them, however a sequence s_γ should not occur at the occurrence position of the sequence s_β. A sequence s is therefore unexpected if (1) the sequence s_α occurs and the sequence s_β occurs (without respecting the occurrence constraint, if the occurrence constraint is specified); or (2) the sequence s_α and the sequence s_γ occurs (with respect to the occurrence constraint, if the occurrence constraint is specified).

4.2.1 Semantic Contradiction

In this section, we introduce the notion of *semantic contradiction* between sequences. Since a sequence s can also represent an itemset if it contains only one itemset (i.e., $|s| = \|s\|$), or represent an item if it contains only one item (i.e., $\|s\| = 1$), the semantic contradiction can also be applied to itemsets and items. Therefore, we use the term *element* in the following definition in order to generalize the semantic contradiction.

Definition 4 (Semantic contradiction) *Given two elements e_ϕ and e_θ, the semantic contradiction between e_ϕ and e_θ is a boolean value determined by a predicate $o(e_\phi, e_\theta)$: if e_ϕ semantically contradicts e_θ, then $o(e_\phi, e_\theta)$ returns 1; otherwise $o(e_\phi, e_\theta)$ returns 0.*

Given two elements e_ϕ and e_θ, denote by $e_\phi \not\approx_{sem} e_\theta$ when $o(e_\phi, e_\theta) = 1$. The semantic contradiction is symmetric but not transitive. We have that $e_\phi \not\approx_{sem} e_\theta$ is equivalent to $e_\theta \not\approx_{sem} e_\phi$, however $e_\phi \not\approx_{sem} e_\theta$ and $e_\theta \not\approx_{sem} e_\varphi$ do not imply that $e_\phi \not\approx_{sem} e_\varphi$. The predicate $o(e_\phi, e_\theta)$ can be designed to compute the semantic contradiction between the elements e_ϕ and e_θ in various manners. For instance, given a set $\mathcal{E}$ of elements, we can build a projection table T of predefined relations on $\mathcal{E} \times \mathcal{E}$, and then the semantic contradiction between any elements $e_\phi, e_\theta \in \mathcal{E}$ can be returned by $o(e_\phi, e_\theta)$ with searching the table T; the semantic contradiction can also be determined by the semantic relatedness between the *concepts* associated with items contained in elements, which can be computed with examining the semantic similarity between concepts and even with concept hierarchies.

The following example illustrates how semantic contradictions between two items are determined with respect to a concept hierarchy in the context of Web usage analysis.

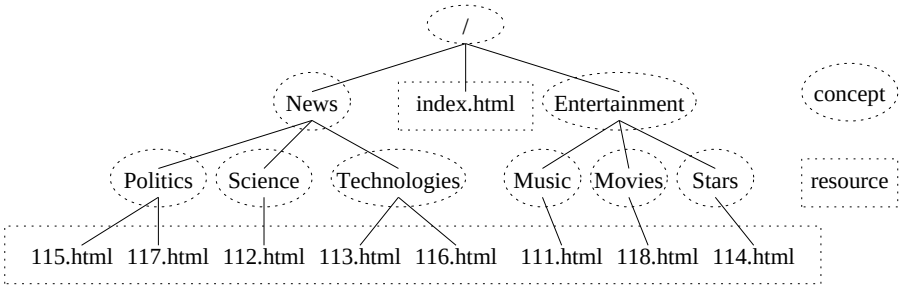


Figure 4.1: A Web site structure hierarchy.

Example 9 In the context of Web usage analysis, the semantic contradictions between resources can be determined from Web site structure. For instance, `login` and `logout` (we ignore the file extension for simplifying the description) can be viewed as semantically contradicting each other, that is, `login` $\not\approx_{sem}$ `logout`. On the other hand, as shown in Figure 4.1, the semantic contradictions can also be determined from Web site structure hierarchies by computing the relatedness between concepts. For instance, `115` and `117` can be viewed as semantically equivalent because they are

both under the concept `Politics`; however `115` and `118` can be viewed as semantically contradicting each other if we consider the path length and semantic relatedness between the concepts `Politics` and `Movies` as relative criteria. □

In [PT98, PT06], the contradiction specified by domain experts between two patterns is a constraint that two patterns logically contradict each other, that is, such two patterns cannot appear in same time (e.g., *weekday* and *weekend*). Our definition of *semantic contradiction* is based on the semantics of two elements, so that logical contradiction can be covered by semantic contradiction. Semantic contradiction can be specified on a broad of notions, from concepts to user classes, and even more complex data types. For example, *login* semantically contradicts *logout* because they are two opposite concepts; a purchase of *Mac* and *Snow Leopard* semantically contradicts a purchase of *PC* and *Windows 7* because they belong to two contradicting user groups; opposite functions contained in DNA sequences can be considered as semantically contradicting each other; acceleration and deceleration processes of a mobile robot can also be considered as semantically contradicting each other; in natural language, two sentences can semantically contradict each other, such as "I like Mac" and "I like PC".

Now we introduce the *consistent semantic contradiction set*, which is similar to the notion of consistent sequence rule set and proposed as follows.

Definition 5 (Consistent semantic contradiction set) *A consistent semantic contradiction set is a set $\mathcal{M}$ of semantic contradictions that for any two relations $(e_{\phi_i} \not\approx_{sem} e_{\theta i}), (e_{\phi_j} \not\approx_{sem} e_{\theta j}) \in \mathcal{M}$, we have that $e_{\phi_i} = e_{\phi_j}$.*

The following example shows the definition of consistent semantic contradiction set, where the relations are considered between sequences.

Example 10 Given two sets $\mathcal{M}_1, \mathcal{M}_2$ of semantic contradictions between sequences, where

$$\mathcal{M}_1 = \left\{ \begin{array}{l} o_1 : \langle (a)(b) \rangle \not\approx_{sem} \langle (b)(a) \rangle \\ o_2 : \langle (a)(b) \rangle \not\approx_{sem} \langle (ab) \rangle \\ o_3 : \langle (a)(b) \rangle \not\approx_{sem} \langle (abc)(ef) \rangle \end{array} \right\} \text{ and } \mathcal{M}_2 = \left\{ \begin{array}{l} o_1 : \langle (a)(b) \rangle \not\approx_{sem} \langle (b)(a) \rangle \\ o_2 : \langle (a)(a) \rangle \not\approx_{sem} \langle (ab) \rangle \\ o_3 : \langle (a)(b) \rangle \not\approx_{sem} \langle (abc)(ef) \rangle \end{array} \right\},$$

then the set $\mathcal{M}_1$ is a consistent semantic contradiction set however the set $\mathcal{M}_2$ is not consistent since in $\mathcal{M}_2$ we have at least that $o_1 = \langle (a)(b) \rangle \not\approx_{sem} \langle (b)(a) \rangle$ and $o_2 = \langle (a)(a) \rangle \not\approx_{sem} \langle (ab) \rangle$. □

In this thesis, the semantic contradiction is one of the primary criteria to measure unexpected sequences, which will be presented in the next chapter.

4.2.2 Sequence Belief

In this section, we propose the notion of sequence beliefs for representing prior knowledge of sequence data with respect to the occurrence and semantics of sequences.

According to our published works [LLP07, LLP08a, LLP08b, LLP08c, LLP09], a *belief* is a sequence implication rule r with respect to a semantic contradiction between $\Delta(r)$ and a predefined sequence, which can be formally denoted as the following form

$$\{s_\alpha \rightarrow^\tau s_\beta\} \wedge \{s_\beta \not\approx_{sem} s_\gamma\}$$

or simply written as $[s_\alpha; s_\beta; s_\gamma; \tau]$. The semantics of such a belief is that within a sequence covered by the knowledge described by the rule $s_\alpha \rightarrow^\tau s_\beta$, if s_α occurs, then s_β occurs with respect to the occurrence constraint τ where s_β cannot be replaced by s_γ. If the semantic contradiction is empty, which is denoted as $\{\emptyset\}$, then only occurrence constraint is examined.

In this manner, let us consider the case that the occurrence of $\langle(a)\rangle$ is expected to be followed the occurrence of $\langle(b)\rangle$, or of $\langle(c)\rangle$, or of $\langle(d)\rangle$, then the following three beliefs must be specified:

$$b_1 = \{\langle(a)\rangle \rightarrow^* \langle(b)\rangle\} \wedge \{\emptyset\},$$
$$b_2 = \{\langle(a)\rangle \rightarrow^* \langle(c)\rangle\} \wedge \{\emptyset\},$$
$$b_3 = \{\langle(a)\rangle \rightarrow^* \langle(d)\rangle\} \wedge \{\emptyset\}.$$

Now given a set of sequences

$$\{s_1 = \langle(a)(b)\rangle, s_2 = \langle(a)(c)\rangle, s_3 = \langle(a)(d)\rangle, s_4 = \langle(a)(e)\rangle\},$$

then we have the following violations:

1. s_1 violates b_2 and b_3 because $\langle(a)\rangle \sqsubseteq s_1$ is not followed by $\langle(c)\rangle \sqsubseteq s_1$ neither $\langle(d)\rangle \sqsubseteq s_1$;
2. s_2 violates b_1 and b_3 because $\langle(a)\rangle \sqsubseteq s_2$ is not followed by $\langle(b)\rangle \sqsubseteq s_1$ neither $\langle(d)\rangle \sqsubseteq s_1$;
3. s_3 violates b_1 and b_2 because $\langle(a)\rangle \sqsubseteq s_3$ is not followed by $\langle(b)\rangle \sqsubseteq s_1$ neither $\langle(c)\rangle \sqsubseteq s_1$;
4. s_4 violates b_1, b_2, and b_3 because $\langle(a)\rangle \sqsubseteq s_4$ is not followed by $\langle(b)\rangle \sqsubseteq s_4$, $\langle(c)\rangle \sqsubseteq s_4$, neither $\langle(d)\rangle \sqsubseteq s_4$.

Obviously, according to the above context, only the violation caused by the sequence s_4 is really interesting and the violations caused by the sequences s_1, s_2, and s_3 are redundant. We addressed this problem in [LLP09] by reorganizing all beliefs consisting of simple sequence implication rules together and then considering the violations of such beliefs integrally. However, even though such consideration is successful in mining unexpected sequences, the semantics of each belief is not clear.

In order to maintain the proper semantics of each belief, we consider a belief as a minimum semantically complete unit by using the notions of consistent sequence rule set and consistent semantic contradiction set. We therefore propose the definition of *sequence belief* as follows.

Definition 6 (Sequence belief) *A sequence belief is a conjunction $\mathcal{R} \wedge \mathcal{M}$, where $\mathcal{R}$ is a non-empty consistent sequence rule set and $\mathcal{M}$ is a consistent semantic contradiction set such that for each relation $(s_{\beta_i} \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M}$, we have that $s_{\beta_i} \in \Delta(\mathcal{R})$, and for any relation $(s_{\beta_i} \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M}$, there does not exist $s_{\beta_j} \in \Delta(R)$ such that $s_{\gamma_i} \sqsubseteq s_{\beta_j}$.*

The semantic constraint imposed on $\mathcal{R}$ by $\mathcal{M}$ requires that for each semantic contradiction $(s_{\beta_i} \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M}$, there exists a sequence rule $r \in \mathcal{R}$ such that $s_{\beta_i} = \Delta(r)$, since a relation $s_{\beta_j} \not\approx_{sem} s_{\gamma_j}$ that does not correspond to any sequence rule in $\mathcal{R}$ is meaningless to the semantics of a belief. A belief can be generated from existing domain knowledge on common behaviors of the data, or from predefined workflows. Following Example 11 and Example 12 illustrate how beliefs are constructed with respect to different contexts.

Example 11 Before considering the customer purchase behaviors addressed in Example 7, we first assume that according to prior knowledge of the retail database, we know the `youths` like to watch `Sci-Fi movies`, thus, the following sequence class rule may be built with respect to youth purchase behaviors:

$$r_1 = \langle(\texttt{Youth})\rangle \rightarrow \langle(\texttt{Sci-Fi-Movie})\rangle.$$

With considering that `Sci-Fi movies` semantically contradict `opera movies` and `drama movies`, then the following belief b_1 can be constructed:

$$b_1 = \left\{ \; r_1 : \langle(\texttt{Youth})\rangle \rightarrow \langle(\texttt{Sci-Fi-Movie})\rangle \; \right\} \wedge \left\{ \begin{array}{l} o_1 : \langle(\texttt{Sci-Fi-Movie})\rangle \not\approx_{sem} \langle(\texttt{Opera-Movie})\rangle \\ o_2 : \langle(\texttt{Sci-Fi-Movie})\rangle \not\approx_{sem} \langle(\texttt{Drama-Movie})\rangle \end{array} \right\}.$$

According to frequent customer purchase behaviors, we can create the sequence implication rule

$$r_2 = \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^* \langle(\texttt{Rock-Music})\rangle,$$

which indicates that the purchase of a `Sci-Fi novel` then `action and Sci-Fi movies` later implies the purchase of a `rock music CD`. If we just expect that a purchase of `rock music CD` should be performed after the precedent purchases, then following belief b_2 can be established for describing this requirement:

$$b_2 = \left\{ \; r_2 : \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^* \langle(\texttt{Rock-Music})\rangle \; \right\} \wedge \left\{ \; \emptyset \; \right\},$$

where the semantic contradiction set is empty without considering semantic contradictions between sequences. Now let the `classical music` be semantically contradicting the `rock music`, then we have the semantic contradiction

$$o_3 = \langle(\texttt{Rock-Music})\rangle \not\approx_{sem} \langle(\texttt{Classical-Music})\rangle,$$

so that the belief b_2 can be rewritten as follows:

$$b_3 = \left\{ r_2 : \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^* \langle(\texttt{Rock-Music})\rangle \right\} \wedge \left\{ o_3 : \langle(\texttt{Rock-Music})\rangle \not\approx_{sem} \langle(\texttt{Classical-Music})\rangle \right\}.$$

Moreover, if customer transaction records show that most of customers purchase a `rock music CD` in a short delay after purchasing a `Sci-Fi novel` then `action and Sci-Fi movies`, for example in the next 3 to 5 purchases, then belief b_3 can be further rewritten as:

$$b_4 = \left\{ r_3 : \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^{[3..5]} \langle(\texttt{Rock-Music})\rangle \right\} \wedge \left\{ o_3 : \langle(\texttt{Rock-Music})\rangle \not\approx_{sem} \langle(\texttt{Classical-Music})\rangle \right\}.$$

□

Example 12 Considering the context described in Example 2 and Example 8, we can construct the following sequence association rule

$$r_1 = \langle(\texttt{home})(\texttt{list})\rangle \rightarrow \langle(\texttt{login})(\texttt{logout})\rangle$$

based on the assumed facts: (1) the authorized users access user home page `home` and then access `list` for verifying new messages; (2) the access of `login` is not obligate for user authorization since the login process can be effected by cookies; (3) the access of `logout` is not obligated to close user session. The following sequence implication rules can be also obtained from the above facts and that the access of `logout` should not be directly after the access of `login`:

$$r_2 = \langle(\texttt{login})(\texttt{home})\rangle \rightarrow^* \langle(\texttt{logout})\rangle,$$
$$r_3 = \langle(\texttt{login})\rangle \rightarrow^{[1..*]} \langle(\texttt{logout})\rangle.$$

Hence, we have the following beliefs without semantic contradictions:

$$b_1 = \left\{ r_2 : \langle(\texttt{login})(\texttt{home})\rangle \rightarrow^* \langle(\texttt{logout})\rangle \right\} \wedge \left\{ \emptyset \right\},$$
$$b_2 = \left\{ r_3 : \langle(\texttt{login})\rangle \rightarrow^{[1..*]} \langle(\texttt{logout})\rangle \right\} \wedge \left\{ \emptyset \right\}.$$

The beliefs b_1 and b_2 can be further combined as belief b_3:

$$b_3 = \left\{ r_4 : \langle(\texttt{login})\rangle \rightarrow^0 \langle(\texttt{home})\rangle \right\} \wedge \left\{ o_1 : \langle(\texttt{home})\rangle \not\approx_{sem} \langle(\texttt{logout})\rangle \right\},$$

where `logout` can be viewed as semantically contradicting `home` in the context of user login process. Other user behaviors can also be represented by beliefs. For instance, the following belief

$$b_4 = \left\{ r_5 : \langle(\texttt{login})(\texttt{list})\rangle \rightarrow^{[0..5]} \langle(\texttt{read})\rangle \right\} \wedge \left\{ o_2 : \langle(\texttt{read})\rangle \not\approx_{sem} \langle(\texttt{logout})\rangle \right\}$$

depicts that we expect that users will not logout to the system too quickly, for example, after reading at least 5 messages. □

Given a belief b, if a sequence s supports at least one rule contained in this belief and no semantic contradiction of any other rules can be found in the sequence s, then we say that the sequence s satisfies the belief b or the sequence s *supports* the belief b, denoted as $s \models b$. The satisfaction of beliefs is specified in the following manners.

1. Let b be a belief that consists of a consistent sequence association rule set $\mathcal{R}$ and a consistent semantic contradiction set $\mathcal{M}$, if there exists a rule $(r = s_\alpha \rightarrow s_\beta) \in \mathcal{R}$ such that $s \models r$, and for any semantic contradiction $(s_{\beta_i} \not\approx_{sem} s_{\gamma_j}) \in \mathcal{M}$ there does not exist a rule $(r' = s_\alpha \rightarrow s_{\beta_i}) \in \mathcal{R}$ such that $s \models r'$, then we have that sequence $s \models b$.

2. Let b be a belief that consists of a consistent sequence implication rule set $\mathcal{R}$ and a consistent semantic contradiction set $\mathcal{M}$, if there exists a rule $(r = s_\alpha \rightarrow^\tau s_\beta) \in \mathcal{R}$ such that $s \models r$, and for any semantic contradiction $(s_{\beta_i} \not\approx_{sem} s_{\gamma_j}) \in \mathcal{M}$ there does not exist a rule $(r' = s_\alpha \rightarrow^{\tau'} s_{\beta_i}) \in \mathcal{R}$ such that $s \models r'$, then we have that sequence $s \models b$.

Nevertheless, given a belief $b = \mathcal{R} \wedge \mathcal{M}$, we say that a sequence s *does not satisfy* the belief b if there does not exist any rule $(r = s_\alpha \rightarrow s_\beta) \in \mathcal{R}$ such that $s \models r$, denoted as $s \not\models b$.

Example 13 Let us consider the belief

$$b = \left\{ \begin{array}{l} r_1 : \langle (a) \rangle \rightarrow^* \langle (b) \rangle \\ r_2 : \langle (a) \rangle \rightarrow^{[2..2]} \langle (bc) \rangle \\ r_3 : \langle (a) \rangle \rightarrow^{[0..2]} \langle (d) \rangle \end{array} \right\} \wedge \left\{ \begin{array}{l} o_1 : \langle (b) \rangle \not\approx_{sem} \langle (cd) \rangle \\ o_1 : \langle (bc) \rangle \not\approx_{sem} \langle (bd)(c) \rangle \\ o_2 : \langle (d) \rangle \not\approx_{sem} \langle (ac) \rangle \end{array} \right\},$$

and the sequences $s_1 = \langle (a)(b)(c)(bc)(d) \rangle$, $s_2 = \langle (a)(b)(ad)(bd) \rangle$, and $s_3 = \langle (a)(d)(c) \rangle$, we have that $s_1 \models b$ because

$$s \models \{r_1, r_2\} \text{ and } s \not\models \left\{ \begin{array}{l} \langle (a) \rangle \rightarrow^* \langle (cd) \rangle, \\ \langle (a) \rangle \rightarrow^{[2..2]} \langle (bd)(c) \rangle \\ \langle (a) \rangle \rightarrow^{[0..2]} \langle (ac) \rangle \end{array} \right\};$$

we also have $s_2 \not\models b$ and $s_3 \not\models b$ because $s_2 \not\models \{r_1, r_2, r_3\}$ and $s_3 \not\models \{r_1, r_2, r_3\}$. □

4.2.3 Belief Tree Representation

In this section, we propose a tree representation of a *belief base* consisting of a set of sequence beliefs. Before constructing the tree representation, we first propose the notions of the *premise sequence* and the *conclusion sequence set* of belief as follows.

Definition 7 (Premise sequence of belief) *Given a belief $b = \mathcal{R} \wedge \mathcal{M}$, the premise sequence of the belief b, denoted as $\Lambda(b)$, is the premise sequence of the consistent rule set $\mathcal{R}$ contained in the belief, that is, $\Lambda(b) = \Delta(\mathcal{R})$.*

Definition 8 (Conclusion sequence set of belief) *Given a belief $b = \mathcal{R} \wedge \mathcal{M}$, the conclusion sequence set of the belief b, denoted as $\Delta(b)$, is the set of all conclusion sequences of the consistent rule set $\mathcal{R}$ contained in the belief, that is, $\Delta(b) = \Delta(\mathcal{R})$.*

Considering the Definition 5, multiple contradictions are allowed to be associated with the same sequence in a consistent semantic contradiction set $\mathcal{M}$, that is, for $(s_{\beta_i} \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M}$ and $(s_{\beta_j} \not\approx_{sem} s_{\gamma_j}) \in \mathcal{M}$, the relation $(s_{\beta_i} = s_{\beta_j}) \wedge (s_{\gamma_i} \neq s_{\gamma_j})$ is the only constraint on the sequences addressed in $\mathcal{M}$. Thus, we further propose the notion of the *contradiction set* of belief $b = \mathcal{R} \wedge \mathcal{M}$ with respect to a conclusion sequence in $\mathcal{R}$, which is defined as follows.

Definition 9 (Contradiction sequence set of belief) *Given a sequence belief $b = \mathcal{R} \wedge \mathcal{M}$, let $s_\beta \in \Delta(b)$ be a conclusion sequence. The contradiction sequence set of the belief b with respect to the sequence s_β, denoted as $\Theta(b, s_\beta)$, is the set of sequences such that for each sequence s_{γ_i} contained in each relation $(s_\beta \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M}$, we have that $s_{\gamma_i} \in \Theta(b, s_\beta)$.*

Therefore, given a belief b can then be regarded as a tree link $\Lambda(b) \longrightarrow \Delta(b) \longrightarrow \Theta(b, s_\beta)$.

A *belief tree*, denoted as T, is a tree representation of a belief. According to the notions defined in above, a belief tree is a tree structure defined as below.

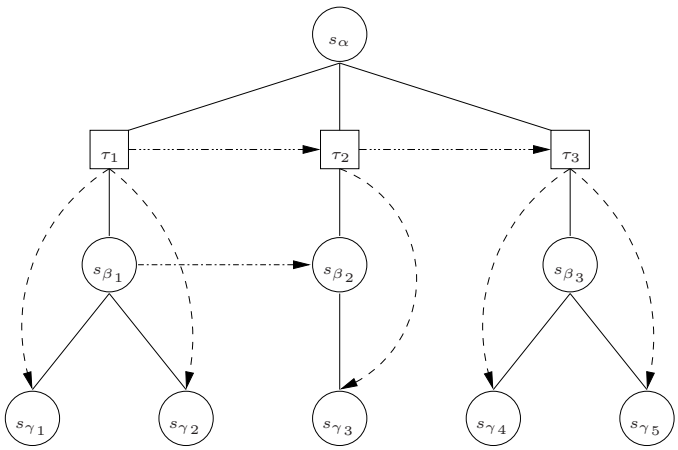


Figure 4.2: A belief tree example.

1. A belief tree T corresponding to a belief b consists of one root node s_α-node for the sequence $s_\alpha = \Lambda(b)$, a set of τ-nodes as the sub-nodes of the root, and a set of sequence subtrees consisting of s-nodes.

2. The τ-node has two field: *min* and *max* corresponding to the occurrence constraint in a sequence implication rule. If the belief consists of sequence association rules, we let $min = -1$.

3. A s-node contains a sequence. In our implementation, a s-node is a reference (e.g., a *pointer* in C/C++, or originally a *reference* in JAVA) to a sequence stored external to the tree structure.

4. Each τ-node possesses a sequence subtree. The sub-root node of a sequence subtree corresponds to a conclusion sequence $s_\beta \in \Delta(b)$ and the sub-nodes correspond to the set of sequences $s_\gamma \in \Theta(b, s_\beta)$. Each τ-node is linked by appending order for optimizing the performance of traversal.

5. A τ-link connects a τ-node and each s-node corresponding to each sequence $s_\gamma \in \Theta(b, s_\beta)$.

6. A s-link connects all $(s_{\beta_i}, s_{\beta_j}) \in \Delta(b)$ such that $s_{\beta_i} = s_{\beta_j}$, with respect to the appending order. For instance, in Figure 4.2, $s_{\beta_1} = s_{\beta_2}$.

Figure 4.2 shows a belief tree example. Based on this definition, we have the following belief tree construction algorithm `BeliefTree` (Algorithm 1). Given an input belief $b = \mathcal{R} \wedge \mathcal{M}$, the algorithm first creates a belief tree T with the root node $s_\alpha = \Lambda(\mathcal{R})$. For each sequence rule $r \in \mathcal{R}$, the algorithm appends the occurrence constraint τ as a τ-node to the root node and appends the conclusion sequence s_β as a s-node to the newly appended τ-node. Then, for each relation $(s_\beta \not\simeq_{sem} s_\gamma) \in \mathcal{M}$, the algorithm finds the location of the s-node of s_β in the tree and appends s_γ as a s-node to s_β. Finally, the algorithm outputs the belief tree T. To construct a belief tree, the algorithm scans the consistent sequence rule set $\mathcal{R}$ and the consistent semantic contradiction set $\mathcal{M}$ once.

Algorithm 1: `BeliefTree` (b) : Belief tree construction.

Input : A belief $b = \mathcal{R} \wedge \mathcal{M}$.
Output : A belief tree T.

```
1  s_α := Λ(R);
2  T := BeliefTree.Create(s_α);
3  foreach r ∈ R do
4      n_τ := T.appendTauNode(r.τ);    /* do not create new τ-node if the same τ exists */
5      n_s := T.appendSeqNode(n, Δ(r));
6      n'_s := T.getLastSeqNode(n_s);    /* find last s-node having the same sequence with n_s */
7      T.linkSeqNode(n'_s, n_s);
8  foreach o ∈ M do
9      n_s := T.getSeqNode(o.s_β);
10     n'_s := T.appendSeqNode(n_s, o.s_γ);
11     T.linkTauNode(n_s.parent, n'_s);
12 return T;
```

A *belief base*, denoted as $\mathcal{B}$, is a set of sequence beliefs. Example 14 shows a tree representation of a belief base with 4 different beliefs.

Example 14 Given a belief base containing the following 4 beliefs:

$$b_1 = \left\{ \langle (l_1) \rangle \rightarrow \langle (a)(ab) \rangle \right\} \wedge \left\{ \langle (a)(ab) \rangle \not\approx_{sem} \langle (c)(d) \rangle \right\};$$
$$b_2 = \left\{ \langle (a)(b) \rangle \rightarrow \langle (c)(d) \rangle \right\} \wedge \left\{ \langle (c)(d) \rangle \not\approx_{sem} \langle (d)(c) \rangle \right\};$$
$$b_3 = \left\{ \langle (a)(d) \rangle \rightarrow^{[2..5]} \langle (b)(c) \rangle, \langle (a)(d) \rangle \rightarrow^{0} \langle (d) \rangle \right\} \wedge \left\{ \langle (b)(c) \rangle \not\approx_{sem} \langle (cd) \rangle \right\};$$
$$b_4 = \left\{ \langle (a)(c) \rangle \rightarrow^{*} \langle (cd) \rangle \right\} \wedge \left\{ \langle (cd) \rangle \not\approx_{sem} \langle (ab) \rangle, \langle (cd) \rangle \not\approx_{sem} \langle (b)(c) \rangle \right\}.$$

The corresponding belief base tree is shown in Figure 4.3. □

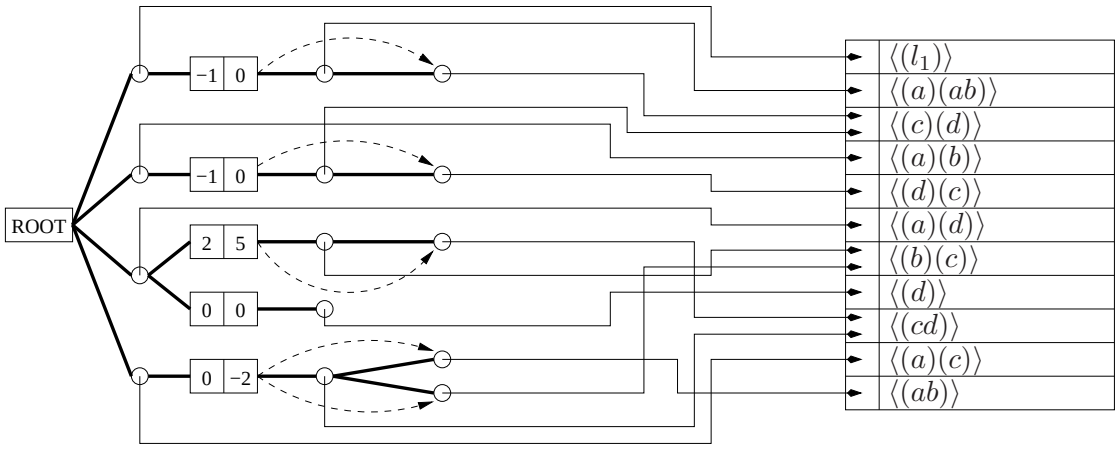


Figure 4.3: An example tree presentation of a belief base.

The tree structure shown in Figure 4.3 is called a *belief base tree*, denoted as $\mathcal{T}$, which consists of a null root node shared for all sub belief trees representing each belief $b \in \mathcal{B}$.

4.3 Unexpected Sequences

In this section, we propose three forms of unexpected sequences with respect to the belief system presented in the previous section.

4.3.1 Completeness Unexpectedness

We first study the unexpected sequences stated by the beliefs with simple sequence implication rules $s_\alpha \rightarrow^* s_\beta$ without considering semantics constraints, where the unexpectedness is caused by incompleteness of sequence.

We call the unexpectedness caused by incompleteness of sequence the *α-unexpectedness*, which is formally called the *completeness-unexpectedness*. We define a unexpected sequence with completeness-unexpectedness as follows.

Definition 10 (Completeness-unexpected sequence) *Given a sequence s and a belief b of consistent simple sequence implication rules, let $s_\alpha = \Lambda(b)$. For each $s_\beta \in \Delta(b)$, if $(s_\alpha \sqsubseteq s) \wedge (s_\alpha \cdot s_\beta \not\sqsubseteq s)$, then the sequence s is a completeness-unexpected sequence with respect to the belief b, denoted as $s \not\Vdash_\alpha b$. We also call such an unexpected sequence an α-unexpected sequence,.*

The simple sequence implication rules contained in a belief b state that at least one sequence in the conclusion set $\Delta(b)$ of the belief b should occur after the occurrence of the premise sequence $\Lambda(b)$ in an expected sequence. Considering that given a simple sequence implication rule $(s_\alpha \rightarrow^* s_\beta)$ and a sequence s, the occurrence constraint $\tau = [0..*]$ is broken if and only if $(s_\alpha \sqsubseteq s) \wedge (s_\alpha \cdot s_\beta \not\sqsubseteq s)$, so that the only factor of this violation is the sequence s_α. We therefore name this form of unexpectedness the α-unexpectedness, and such an unexpected sequence is so called an α-unexpected sequence.

Example 15 Let us consider the belief b_2 listed in Example 11, i.e.,

$$b_2 = \left\{ \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^* \langle(\texttt{Rock-Music})\rangle \right\} \wedge \left\{ \emptyset \right\},$$

which determines α-unexpected sequences. This belief depicts that a purchase of `rock music CD` is expected after the purchases of a `Sci-Fi novel` then `action and Sci-Fi movies` later, otherwise it is unexpected. Therefore, the sequence

$$s_1 = \langle(\texttt{Sci-Fi-Novel})(\texttt{Printer})(\texttt{Action-Movie Sci-Fi-Movie})(\texttt{Play-Station})\rangle,$$

does not support the belief b_2 (i.e., $s_1 \not\models b_2$) and violates the belief b_2 (i.e., $s_1 \not\Vdash_\alpha b_2$); however the sequence

$$s_2 = \langle(\texttt{Sci-Fi-Novel})(\texttt{Printer})(\texttt{Sci-Fi-Movie})(\texttt{Play-Station})\rangle,$$

does not support the belief b_2 (i.e., $s_2 \not\models b_2$) neither violates the belief b_2. □

Let us recall the problem stated in Section 4.2.2 with considering the following independent beliefs

$$b_1 = \left\{ \langle(a)\rangle \rightarrow^* \langle(b)\rangle \right\} \wedge \left\{ \emptyset \right\},$$
$$b_2 = \left\{ \langle(a)\rangle \rightarrow^* \langle(c)\rangle \right\} \wedge \left\{ \emptyset \right\},$$
$$b_3 = \left\{ \langle(a)\rangle \rightarrow^* \langle(d)\rangle \right\} \wedge \left\{ \emptyset \right\}$$

and the sequence set

$$\{s_1 = \langle(a)(b)\rangle, s_2 = \langle(a)(c)\rangle, s_3 = \langle(a)(d)\rangle, s_4 = \langle(a)(e)\rangle\}.$$

According to the definition of α-unexpected sequences, the following unexpectedness can be obtained:

$$\left\{ \begin{array}{l} s_1 \not\Vdash_\alpha b_2 \\ s_1 \not\Vdash_\alpha b_3 \end{array} \right\}, \left\{ \begin{array}{l} s_2 \not\Vdash_\alpha b_1 \\ s_2 \not\Vdash_\alpha b_3 \end{array} \right\}, \left\{ \begin{array}{l} s_3 \not\Vdash_\alpha b_1 \\ s_3 \not\Vdash_\alpha b_2 \end{array} \right\}, \text{ and } \left\{ \begin{array}{l} s_4 \not\Vdash_\alpha b_1 \\ s_4 \not\Vdash_\alpha b_2 \\ s_4 \not\Vdash_\alpha b_3 \end{array} \right\}.$$

Semantically, the beliefs b_1, b_2, and b_3 depict that $\langle(b)\rangle$, $\langle(c)\rangle$, or $\langle(d)\rangle$ should occur after the occurrence of $\langle(a)\rangle$, thus, in this meaning, only the sequence s_4 is unexpected. However, with respect to b_1, b_2, and b_3, all of the 4 sequences are α-unexpected. This ambiguity is avoided by combining the beliefs b_1, b_2, and b_3 into one single belief with consistent rules, that is,

$$b_4 = \left\{ \begin{array}{l} \langle(a)\rangle \rightarrow^* \langle(b)\rangle \\ \langle(a)\rangle \rightarrow^* \langle(c)\rangle \\ \langle(a)\rangle \rightarrow^* \langle(d)\rangle \end{array} \right\} \wedge \left\{ \begin{array}{l} \emptyset \end{array} \right\},$$

with which we have that $s_1 \models b_4$, $s_2 \models b_4$, $s_3 \models b_4$, and $s_4 \not\models_\alpha b_4$.

Given a belief b and a sequence s, the α-unexpectedness can be discovered by verifying the occurrence of the premise sequence $s_\alpha = \Lambda(b)$ and the absence of each conclusion sequence $s_\beta \in \Delta(b)$.

In order to match the occurrence of $s \sqsubseteq s'$ within a specified *range* of the occurrence of the first itemset of the sequence s, we designed three algorithms `SeqMatchFirst` (to find the first occurrence of s in s'), `SeqMatchMax` (to find the maximal occurrence of s in s'), and `SeqMatchMin` (to find the first non-redundant occurrence of s in s'). One of the three algorithms can be selected with respect to different discovery strategies, thus, in the remainder of this thesis, we use `SeqMatch` as the subsequence matching routine, and `SeqMatchAll` as the routine that matches all occurrences of a subsequence.

Example 16 Let us consider the sequences $s = \langle(a)(b)(c)\rangle$ and $s' = \langle(a)(b)(a)(a)(b)(c)(c)\rangle$. Th algorithms `SeqMatchFirst`, `SeqMatchMax`, and `SeqMatchMin` return the beginning (starting from 0) and ending positions of the sequence s in s'. These three algorithms are based on linear matching for subsequence inclusion, which scan the sequence s' once. `SeqMatchFirst` returns $(0, 5)$ corresponding to the first (a) and the first (c); `SeqMatchMax` returns $(0, 6)$ corresponding to the first (a) and the last (c); `SeqMatchMin` returns $(3, 5)$, corresponding to the last (a) and the first (c). □

With the routine `SeqMatch`, the discovery of α-unexpectedness is therefore proposed as listed in Algorithm 2. The algorithm accepts a belief T, a sequence s, and a pair *pos* indicating the occurrence of the sequence s_α contained in the s_α-node of T in the sequence s as inputs (i.e., $s_\alpha \sqsubseteq s$ is already confirmed). If T contains no τ-node corresponding to $\tau = *$ then the algorithm returns $pair(-1, -1)$ to declare the failure; otherwise, for each s-node connected to the τ-node corresponding to $\tau = *$, the algorithm matches the occurrence of the sequence s_β contained in the s-node within the range $[pos.second + 1, |s| - 1]$ (i.e., from the itemset next to the end of the occurrence of $s_\alpha \sqsubseteq s$ till to the end of s). If any s_β is matched, then the algorithm returns a tuple of -1; otherwise, the algorithm returns $tuple(s.id, pos.first, |s| - 1)$, which corresponds the

occurrence of the α-unexpectedness discovered in the sequence s, that is, from the beginning of s_α till to the end of s. In the worst case, $|\Delta(b)|$ matches are performed onto the sequence s.

Algorithm 2: `UxpsMatchAlpha` (T, s, pos) : Matching α-unexpectedness.

Input : A belief tree T, a sequence s, and a pair pos indicating the occurrence of the sequence s_α contained in the s_α-node of T in s.

Output : The occurrence of α-unexpectedness in s with respect to T.

```
1 if n_τ := T.getTauNode(WILD) then     /* find the τ-node corresponding to τ = * */
2     while n_{s_β} := T.nextSubNode(n_τ) do
3         uxp := SeqMatch(n_{s_β}.data, s, pair(pos.second + 1, |s| − 1));
4         if uxp.first ≠ −1 then
5             return pair(−1, −1);
6     return tuple(s.id, pos.first, |s| − 1);
7 else
8     return tuple(−1, −1, −1);
```

The α-unexpectedness depicts the unexpectedness contained in data with the characteristics such as "if the element s_α occurs, then at least one of the elements $s_{\beta_1}, s_{\beta_2}, \ldots, s_{\beta_n}$ should occurs later; otherwise it is unexpected", that is,

$$s_\alpha \rightarrow (s_{\beta_A} \vee s_{\beta_2} \vee \ldots \vee s_{\beta_n}).$$

This model is essential because the model

$$s_\alpha \rightarrow (s_{\beta_1} \wedge s_{\beta_2})$$

can be reduced to be the model

$$s_\alpha \rightarrow comp(s_{\beta_1}, s_{\beta_2}),$$

where $comp(s_{\beta_1}, s_{\beta_2})$ is a composition function of s_{β_1} and s_{β_2}, for example, $(s_{\beta_1} \cdot s_{\beta_2})$, $(s_{\beta_2} \cdot s_{\beta_1})$, etc.

The discovery of α-unexpectedness is applicative in many application domains when the effects of missing elements are critical. For instance, in Web access log analysis, we may find that an incomplete user navigation sequence often implies the errors like server failure or remote intrusion attempts. In the context of bioinformatics, this form of unexpectedness may also be found in DNA segments and such incomplete segments might cause, for example, abnormal behaviors.

4.3.2 Occurrence Unexpectedness

The notion of completeness-unexpectedness (α-unexpectedness) has been proposed and studied in the previous section. In this section, we study the unexpected sequences stated by the beliefs

with predictive sequence implication rules $s_\alpha \rightarrow^{\tau} s_\beta$, where $\tau \neq *$, without considering semantics constraints.

The unexpectedness studied in this section is caused by the occurrence position of sequences, which is called the *β-unexpectedness*, or formally called the *occurrence-unexpectedness*. We define a unexpected sequence with occurrence-unexpectedness as follows.

Definition 11 (Occurrence-unexpected sequence) *Given a sequence s and a belief b of consistent sequence implication rules, let $s_\alpha = \Lambda(b)$. If there exists $s_\beta \in \Delta(b)$ such that for each rule $(s_\alpha \rightarrow^{\tau_i} s_\beta)$ contained in the belief b we have not that $(s_\alpha \sqsubseteq s) \wedge (s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s) \wedge (|s'| \models \tau_i)$, then the sequence s is an occurrence-unexpected sequence with respect to the belief b, denoted as $s \nVdash_\beta b$. We also call such an unexpected sequence a β-unexpected sequence.*

The predictive sequence implication rules contained in a belief b state that at least one sequence in the conclusion set $\Delta(b)$ of the belief b should occur after the occurrence of the premise sequence $\Lambda(b)$ in an expected sequence, with respect to the occurrence constraint τ associated with the rule. Considering that given a predictive sequence implication rule $(s_\alpha \rightarrow^{\tau} s_\beta)$ and a sequence s, the occurrence constraint $\tau \neq *$ is broken if and only if $s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s$ where $|s'| \not\models \tau$, so that the primary factor of this violation is the occurrence position of the sequence s_β. We therefore name this form of unexpectedness the β-unexpectedness, and such an unexpected sequence is so called a β-unexpected sequence.

Example 17 Let us consider the belief b_4 listed in Example 11:

$$b_4 = \left\{ r_3 : \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^{[3..5]} \langle(\texttt{Rock-Music})\rangle \right\} \wedge \left\{ o_3 : \langle(\texttt{Rock-Music})\rangle \not\preceq_{sem} \langle(\texttt{Classical-Music})\rangle \right\}.$$

With this belief, the purchase of a `rock music CD` is expected within the next 3 to 5 purchases after the purchases of a `Sci-Fi novel` then `action and Sci-Fi movies` later, however if the purchase of a `rock music CD` is out of the range [3..5], then the belief is broken. Thus, for example, the customers who purchase a `rock music CD` just in the next purchase, such as the sequence

$$s = \langle(\texttt{Sci-Fi-Novel})(\texttt{Printer})(\texttt{Action-Movie Sci-Fi-Movie})(\texttt{Rock-Music})\rangle,$$

is β-unexpected to the belief b_4 and might be valuable to make new promotion strategies on related products. Notice that $\langle(\texttt{Classical-Music})\rangle$ in this belief is not considered in the context of β-unexpectedness. □

Notice that in a consistent sequence implication rule set, there is no constraints on the value of the occurrence constraint τ associated with each rule. Therefore, given a belief b, there can exist

two rules r_1 and r_2 in this belief such that $r_1 = s_\alpha \rightarrow^{\tau_1} s_\beta$ and $r_2 = s_\alpha \rightarrow^{\tau_2} s_\beta$. In this case, we consider the disjunction of the two τ values for determining β-unexpectedness, that is, $s_\alpha \cdot s' \cdot s_\beta \not\sqsubseteq_c s$ where $|s'| \not\models (\tau_1 \vee \tau_2)$. For instance, considering two rules $s_\alpha \rightarrow^{\tau_1} s_\beta$ and $s_\alpha \rightarrow^{\tau_2} s_\beta$ contained in a consistent sequence implication rule set, if $\tau_1 = [2..6]$ and $\tau_1 = [4..8]$, then $(\tau_1 \vee \tau_2) = [2..8]$, shown as Figure 4.4.

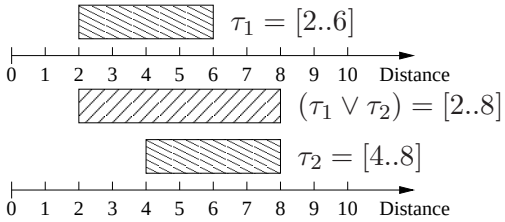


Figure 4.4: Disjunction of occurrence constraints.

Further, if $(\tau_1 \vee \tau_2) = [0..*]$, for example $\tau_1 = [0..5]$ and $\tau_2 = [4..*]$, then in this case, the occurrence-unexpectedness is equivalent to the completeness-unexpectedness.

Given a belief b, the β-unexpectedness can be discovered by determining the occurrence of the premise sequence $s_\alpha = \Lambda(b)$ and the occurrence position of each conclusion sequence $s_\beta \in \Delta(b^\tau)$, where the computational can be performed to a subset of the consistent sequence implication rule set $\mathcal{R}$ of the belief b. In fact, the rule set $\mathcal{R}$ can be considered as a group of subsets $\mathcal{J}$ of the rules $r \in \mathcal{R}$ such that all rules in such a subset $\mathcal{J} \subseteq \mathcal{R}$ have the same conclusion sequence s_β, that is,

$$\forall r_1, r_2 \in \mathcal{R}, \Delta(r_1) = \Delta(r_2) \iff (r_1 \in \mathcal{J}_1) \wedge (r_2 \in \mathcal{J}_2) \wedge (\mathcal{J}_1 = \mathcal{J}_2).$$

Such a maximal subset $\mathcal{J} \subseteq R$ is called a *member set* of the belief b.

Lemma 1 *Given a belief b, if a sequence s violates a member set $\mathcal{J}$ of the belief b, then the sequence s is β-unexpected to the belief b.*

Proof. The proof is immediate. For any two rules $r_1, r_2 \in \mathcal{R}$ such that $r_1 = s_\alpha \rightarrow^{\tau_1} s_{\beta_1}$ and $r_1 = s_\alpha \rightarrow^{\tau_2} s_{\beta_2}$, if $r_1 \in \mathcal{J}_i$ and $s_{\beta_1} = s_{\beta_2}$, then $r_2 \in \mathcal{J}_i$. Thus, if a sequence s violates each rule $r_i \in \mathcal{J}_i$, then s violates each rule $r_j \in \mathcal{R}$ that have the same premise and conclusion sequences. According to Definition 11, s is therefore β-unexpected to this belief. □

The routine of β-unexpectedness discovery is listed in Algorithm 3, which accepts a belief tree T, a sequence s, and a pair *pos* indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in the sequence s as inputs (i.e., $s_\alpha \sqsubseteq s$ is already confirmed).

For each τ-node connected to the root node of T, the algorithm first verifies whether current τ-node corresponds to a predictive sequence implication rule. If not, next τ-node will be selected

till to all τ-nodes are processed. If current τ-node corresponds to a predictive sequence implication rule, from each s-node directly connected to current τ-node, the algorithm uses a recursive routine `SeqNodeMatchBeta` (Algorithm 4) to determine the set of all occurrences of the sequence s_β contained in the s-nodes linked by a s-link. If the set returned by `SeqNodeMatchBeta` is empty, then s is not β-unexpected to the belief corresponding to T; otherwise s is β-unexpected. A global option `FIRST_UXPS_ONLY` can be set to profit from Lemma 1, which returns the β-unexpectedness from the first matched member set of the belief; otherwise, the algorithms returns the set of all occurrences of the matched β-unexpectedness.

Algorithm 3: `UxpsMatchBeta` (T, s, pos) : Matching β-unexpectedness.

Input : A belief tree T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in s.

Output : The set of all occurrences of β-unexpectedness in s with respect to T.

```
1  N := NodeSet.Create();
2  n_τ := T.firstTauNode();
3  while n_τ ≠ null and n_τ ∉ N do
4      if n_τ.data.min = −1 then
5          continue;    /* skip sequence association rules */
6      if n_τ.data.min = 0 or n_τ.data.max = −1 then
7          continue;    /* skip simple sequence implication rules */
8      n_sβ := n_τ.firstSubNode();
9      uxps := SeqNodeMatchBeta(T, N, n_sβ, s, pos);    /* N will be updated */
10     if uxps = ∅ then
11         continue;
12     if options | FIRST_UXPS_ONLY then    /* use the conclusion of Lemma 1 */
13         return uxps;
14     n_τ := T.nextTauNode(n_τ);
15 return uxps;
```

As listed in Algorithm 4, `SeqNodeMatchBeta` accepts a belief tree T, a node set N, a s-node n_s in T containing a sequence s_β, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in the sequence s as inputs, and returns the set of all occurrences of s_β that violates all sequence implication rules represented by $s_\alpha \rightarrow^\tau s_\beta$ via the s-link and the occurrence constraints τ as the parent node of each s-node.

The algorithm first examines the τ value of the τ-node associated with the node n_s, then matches the occurrence of $s_\beta \sqsubseteq s$ with respect to the complement of $\tau \neq *$. If the occurrence of $s_\beta \sqsubseteq s$ is recursively matched by `SeqNodeMatchBeta` in each s_β contained in all s-node followed by the s-link, then a β-unexpectedness is matched and the algorithms returns the set of pairs containing all such occurrences of s_β; otherwise, the algorithm returns an empty tuple set.

Algorithm 4: `SeqNodeMatchBeta` (T, N, n_s, s, pos) : Recursive matching of s-node for β-unexpectedness.

Input : A belief tree T, a node set N, a s-node n_s in T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in s.

Output : The set of all occurrences of the sequence contained in n_s in s.

```
1  N.add(n_s);
2  uxps := TupleSet.Create();
3  n_τ := n_s.parent;
4  if n_τ.data.min = 0 then      /* τ = [0..max] */
5      u := SeqMatch(n_s.data, s, pair(pos.second + n_τ.data.max + 1, |s| − 1));
6      if u.first = −1 then
7          return TupleSet.Create();
8      uxps.add(u);
9  else if n_τ.data.max = −1 then      /* τ = [min..*] */
10     u := SeqMatch(n_s.data, s, pair(pos.second + 1, n_τ.data.min − 1));
11     if u.first = −1 then
12         return TupleSet.Create();
13     uxps.add(u);
14 else      /* τ = [min..max] */
15     u_1 := SeqMatch(n_s.data, s, pair(pos.second + 1, pos.second + n_τ.data.min − 1));
16     u_2 := SeqMatch(n_s.data, s, pair(pos.second + n_τ.data.max + 1, |s| − 1));
17     if u_1.first = −1 and u_2.first = −1 then
18         return TupleSet.Create();
19     if u_1.first ≠ −1 then
20         uxps.add(tuple(s.id, u1.first, u1.second));
21     if u_2.first ≠ −1 then
22         uxps.add(tuple(s.id, u2.first, u2.second));
23 if n := T.nextSeqNode(n_s) then
24     uxps.append(SeqNodeMatchBeta(T, N, n, s, pos));      /* recursion */
25 return uxps;
```

The complement of τ, denoted as $(* \setminus \tau)$, is computed in Algorithm 4 with respect to different values of τ, which is defined as follows.

$$(* \setminus \tau) = \begin{cases} [(max+1)..*] & \text{if } \tau = [0..max] \\ [0..(min-1)] & \text{if } \tau = [min..*] \\ [0..(min-1)] \vee [(max+1)..*] & \text{if } \tau = [min..max] \\ \alpha\text{-unexpectedness} & \text{if } \tau = [0..*] \end{cases} . \tag{4.1}$$

According to the complement of τ, in the worst case, the algorithm `UxpsMatchBeta` matches the input sequence s for $2\,|\Delta(b)|$ times.

The β-unexpectedness focuses on the disordered elements in sequence data, that is, the characteristics like "if the element s_α occurs, then the element s_β should occurs within a range after the occurrence of s_α; if the occurrence of s_β is out of the range, then it is unexpected", which are interesting for many application domains including telecommunication network monitoring, mechanical system exploitation, and so on.

4.3.3 Semantics Unexpectedness

We now study the unexpectedness with considering semantics constraints on sequence rules. This category of unexpected sequences is addressed in the beliefs with non-empty semantic contradiction set, and the unexpectedness is caused by semantic contradiction.

In this case, the occurrence of a sequence is replaced by a sequence where the two sequences semantically contradict each other, so that this form is called the γ-*unexpectedness*, or formally the *semantics-unexpectedness*. We define a unexpected sequence with semantics-unexpectedness as follows.

Definition 12 (Semantics-unexpected sequence) *Given a sequence s and a belief $b = \mathcal{R} \wedge \mathcal{M}$, where $\mathcal{R} \neq \emptyset$ and $\mathcal{M} \neq \emptyset$ are respectively the consistent set of sequence rules and of semantic contradictions, let $s_\alpha = \Lambda(b)$. If $s_\alpha \sqsubseteq s$, and if there exists a sequence rule $r \in \mathcal{R}$ and a semantic contradiction $(s_{\beta_i} \not\approx_{sem} s_{\gamma_j}) \in \mathcal{M}$ such that:*

1. *$s \models (s_\alpha \rightarrow s_{\gamma_j})$, if r is a sequence association rule $s_\alpha \rightarrow s_{\beta_i}$;*
2. *$s \models (s_\alpha \rightarrow^{\tau_i} s_{\gamma_j})$, if r is a sequence implication rule $s_\alpha \rightarrow^{\tau_i} s_{\beta_i}$,*

then the sequence s is a γ-unexpected sequence with respect to the belief b, denoted as $s \nVdash_\gamma b$. We also call such an unexpected sequence a semantics-unexpected sequence.

A belief $b = \mathcal{R} \wedge \mathcal{M}$ with a non-empty semantic contradiction set $\mathcal{M}$ states that the semantic contradictions of a conclusion sequence $s_\beta \in \Delta(b)$ should not occur with the premise sequence $s_\alpha = \Lambda(b)$ with respect to the form of the involved sequence rules. The presence of contradiction sequence violates a sequence rule with respect to the following cases.

1. For a sequence association rule $s_\alpha \rightarrow s_\beta$ and a semantic contradiction $s_\beta \not\approx_{sem} s_\gamma$, if $s_\alpha \sqsubseteq s$ and $s_\gamma \sqsubseteq s$, then the rule is broken.
2. For a sequence implication rule $s_\alpha \rightarrow^\tau s_\beta$ and a semantic contradiction $s_\beta \not\approx_{sem} s_\gamma$, if there exists a sequence s' such that $s' \models \tau$ and $s_\alpha \cdot s' \cdot s_\gamma \sqsubseteq s$, then the rule is broken.

Therefore, the primary factor is the occurrence the sequence s_γ contained in the semantic contradiction, so that we name this form of unexpectedness the γ-unexpectedness, and such an unexpected sequence is so called a γ-unexpected sequence.

Example 18 Let us consider again the belief b_4 studied in Example 17:

$$b_4 = \left\{ \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^{[3..5]} \langle(\texttt{Rock-Music})\rangle \right\} \wedge \left\{ \langle(\texttt{Rock-Music})\rangle \not\approx_{sem} \langle(\texttt{Classical-Music})\rangle \right\}.$$

The `rock music` can be considered as contradicting the `classical music`, that is, the purchase of a `rock music CD` cannot be replaced by a purchase of a `classical music CD`. Thus, since the purchase of a `rock music CD` is expected within the next 3 to 5 purchases after the purchases of a `Sci-Fi novel` then `action and Sci-Fi movies` later, the purchase of a `classical music CD` is not expected within the range of the next 3 to 5 purchases, and the following sequence

$$s = \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})(\texttt{PC})(\texttt{Printer})(\texttt{PC-Book})(\texttt{Classical-Music})\rangle$$

is γ-unexpected to the belief b_4, that is, $s \nVdash_{\gamma} b_4$. □

As mentioned in Definition 6, given a belief $b = \mathcal{R} \wedge \mathcal{M}$, we have the following requirements on the sequence rules in $\mathcal{R}$ and semantic contradictions in $\mathcal{M}$:

(1) $\forall(s_{\beta_i} \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M},\ s_{\beta_i} \in \Delta(\mathcal{R})$;
(2) $\forall(s_{\beta_i} \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M},\ \nexists s_{\beta_j} \in \Delta(R)$ such that $s_{\gamma_i} \sqsubseteq s_{\beta_j}$.

We have discussed the requirement (1) in Section 4.2.2, now let us further discuss the requirement (2) in this section.

We first consider a *specialization/generalization relation* on sequences. For two sequences s_ϕ and s_θ, if $s_\phi \sqsubseteq s_\theta$, then we say that the sequence s_ϕ is more general than the sequence s_θ, denoted as $s_\phi \preceq s_\theta$; we also say that the sequence s_θ is more specific than the sequence s_θ. We write $s_\phi \prec s_\theta$ if $s_\phi \preceq s_\theta$ and not $s_\theta \preceq s_\phi$. For example, we have that $\langle(a)\rangle \preceq \langle(a)(b)\rangle$ since we have that $\langle(a)\rangle \sqsubseteq \langle(a)(b)\rangle$. An analogical example in human cognitions can be that, "apples are more general to appear in market baskets than apples plus oranges appear together". According to this observation, we have the following property on semantics based unexpectedness in this thesis.

Property 1 *The semantics based unexpectedness is no more general than the expectedness in the data.*

Example 19 Let $r = \langle(a)\rangle \rightarrow \langle(b)(c)\rangle$ be a sequence rule, $o_1 = (b)(c) \not\approx_{sem} (b)$, $o_2 = (b)(c) \not\approx_{sem} (b)(cd)$ be two semantic contradictions, and $b_1 = \{r\} \wedge \{o_1\}$, $b_2 = \{r\} \wedge \{o_2\}$ be two beliefs. For the belief b_1, any expected sequence contains $s = \langle(a)(b)(c)\rangle$ and any γ-unexpected sequence is determined by the occurrence of $s' = \langle(a)(b)\rangle$, thus we have that $s' \preceq s$, which means that any expected sequence is unexpected and violates the basis of semantics based unexpectedness.

However, for the belief b_2, any expected sequence contains $s = \langle(a)(b)(c)\rangle$ and any γ-unexpected sequence is determined by the occurrence of $s' = \langle(a)(b)(cd)\rangle$, that is, $s \preceq s'$, which confirms that expectedness is more general than unexpectedness in semantics based context. □

In fact, the γ-unexpectedness stated by the belief b_1 shown in the above example can be replaced by finding α-unexpectedness with a belief containing the rule $\langle(a)(b)\rangle \rightarrow^* \langle(c)\rangle$. Further, if we have a rule $r = \langle(a)\rangle \rightarrow \langle(b)(c)\rangle$ and we want to find the unexpectedness caused by the absence of (b) (i.e., $\langle(a)(c)\rangle$ without (b) between (a) and (c) is unexpected), then the composition of beliefs can be applied to resolve this kind of problems, and which will be discussed in Section 4.6 at the end of this chapter.

The routine of γ-unexpectedness discovery is listed in Algorithm 5, which accepts a belief tree T, a sequence s, and a pair *pos* indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in the sequence s as inputs (i.e., $s_\alpha \sqsubseteq s$ is already confirmed).

Algorithm 5: `UxpsMatchGamma (`T`, `s`, `pos`)` : Matching γ-unexpectedness.

Input : A belief tree T, a sequence s, and a pair *pos* indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in s.

Output : The set of all occurrences of γ-unexpectedness in s with respect to T.

```
1  uxps := TupleSet.Create();
2  range := pair(−1, −1);
3  n_τ := T.firstTauNode();
4  while n_τ ≠ null do
5      if n_τ.data.min = −1 then
6          range.set(0, |s| − 1);    /* sequence association rules */
7      else if n_τ.data.max = −1 then
8          range.set(pos.second + n_τ.data.min + 1, |s| − 1);    /* τ = [min..*], min ≥ 0 */
9      else
10         range.set(pos.second + n_τ.data.min + 1, pos.second + n_τ.data.max);
11     n_{s_γ} := n_τ.firstLinkedNode();
12     while n_{s_γ} ≠ null do
13         u := SeqMatch(n_{s_γ}.data, s, range);
14         if u.first ≠ −1 then
15             uxps.add(tuple(s.id, u.first, u.second));
16             if options | FIRST_UXPS_ONLY then    /* first occurrence of γ-unexpectedness */
17                 return uxps;
18         n_{s_γ} := n_τ.nextLinkedNode(n_{s_γ});
19     n_τ := T.nextTauNode(n_τ);
20 return uxps;
```

For each τ-node connected to the root node of T, the algorithm finds each s-node n_{s_γ} connected

by each τ-link, which contains the sequence $s_\gamma \in \Theta(b, s_\alpha)$, where b is the belief the tree T represents. If an occurrence of s_γ is matched in s with respect to the range specified by the τ value contained in the τ-node, the algorithm adds the occurrence to the set *uxps* of occurrences. If the global option `FIRST_UXPS_ONLY` is set, then the algorithm returns as soon as having added the occurrence of s_γ into *uxps* and returns *uxps*; otherwise, the algorithm returns the set of all occurrences of the matched γ-unexpectedness.

In the worst case, for a belief b, the input sequence s is matched $\sum_{s_\beta \in \Delta(b)} |\Theta(b, s_\beta)|$ times by the algorithm `UxpsMatchGamma`.

The discovery of γ-unexpectedness can be used to find the sequences semantically unexpected to prior knowledge, which is especially interesting for finding the behaviors oriented application domains. For instance, in customer purchase behavior analysis, new product promotion strategies can be addressed from the studies of γ-unexpectedness; in Web usage analysis, the studies of γ-unexpectedness further permit improving site structure, optimizing or personalizing content organization, and so on.

4.4 Approach MUSE

Based on the matching processes of α-unexpected, β-unexpected, and γ-unexpected sequences, we propose the framework MUSE in the sense of *Multiple Unexpected Sequence Extraction*.

The purpose of MUSE is to discover multiple unexpected sequences in a large sequence database with respect to the belief system acquired from prior domain knowledge, which is illustrated as the framework shown in Figure 4.5.

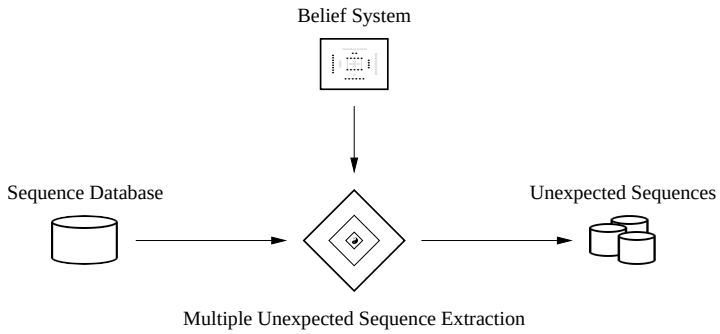


Figure 4.5: The MUSE framework.

The framework accepts a belief base $\mathcal{B}$ and a sequence database $\mathcal{D}$ as inputs, and outputs all unexpected sequences stated by each unexpectedness with respect to each belief $b \in \mathcal{B}$. For each sequence $s \in \mathcal{D}$, the framework first matches whether the premise sequence $s_\alpha = \Lambda(b)$ occurs in s. If we have that $s_\alpha \sqsubseteq s$, then the framework tries to match α-, β- or γ-unexpectedness from the

sequence s by using the algorithms `UxpsMatchAlpha`, `UxpsMatchBeta`, and `UxpsMatchGamma` with respect to the occurrence of s_α in s.

Algorithm 6: MUSE: Multiple Unexpected Sequence Extraction.

Input : A belief base $\mathcal{B}$ and a sequence database $\mathcal{D}$.

Output : Unexpected sequences stated by each unexpectedness with respect to each belief $b \in \mathcal{B}$.

```
1  T := BeliefTree.Create();
2  foreach b ∈ B do
3      T := T.append(BeliefTree(b));
4  foreach s ∈ D do
5      foreach T ∈ T do
6          pos[] := SeqMatchAll(T.n_{s_α}.data, s, pair(0, |s| − 1));
7          while pos[i].first ≠ −1 do      /* s_α ⊑ s */
8              uxp := UxpsMatchAlpha(T, s, pos);
9              if uxp.first ≠ −1 then
10                 output tuple(T.id, ALPHA, s, uxp);
11             uxps := UxpsMatchBeta(T, s, pos);
12             if uxps ≠ ∅ then
13                 output tuple(T.id, BETA, s, pair(pos.first, select(uxps).second));
14             uxps := UxpsMatchGamma(T, s, pos);
15             if uxps ≠ ∅ then
16                 output tuple(T.id, GAMMA, s, pair(pos.first, select(uxps).second));
17             i++;
```

Once one form of unexpectedness is matched, the framework outputs the sequence s as an unexpected sequence with the information of the unexpectedness with the ID of the belief and the form of unexpectedness, and the occurrence of the unexpectedness.

Notice that the algorithms `UxpsMatchBeta` and `UxpsMatchGamma` returns a set of occurrences of the unexpectedness, so that the framework generates a best occurrence of the unexpectedness by using the start position of the premise sequence s_α and the finish position of the conclusion sequence s_β or of the contradiction sequence s_γ. The function *select* takes account of the selection of the occurrence of s_β or s_γ by using a user defined criterion. In this thesis, we select the occurrence corresponding to minimize the length of the unexpectedness.

Not difficult to see, the efficiency of MUSE depends on the sequence match routine `SeqMatch`, which is called in each step of the framework. We have shown that time complexity of `SeqMatch` is linear to the size of input sequence, that is, $\mathcal{O}(n)$ on sequence size n. Given a belief with rule $s_\alpha \rightarrow^\tau s_\beta$ and semantic contradiction $s_\beta \not\approx_{sem} s_\gamma$, the process of mining unexpected sequences is equivalent to the process of mining sequences that support the rules $s_\alpha \rightarrow^{(*\backslash\tau)} s_\beta$ and $s_\alpha \rightarrow^\tau s_\gamma$. In worst case, the time complexity of rule matching is $\mathcal{O}(n^2)$. For example, for rule $\langle(a)\rangle \rightarrow^{[1..*]} \langle(b)\rangle$,

to find β-unexpectedness is equivalent to match the rule $\langle(a)\rangle \rightarrow^{[0..0]} \langle(b)\rangle$ in a sequence: given n-length sequence $\langle(a)(a)\ldots(a)(a)(b)\rangle$, we need call `SeqMatch` $n-1$ times, so totally $n(n-1)$ itemset inclusions are required. However, this case can be optimized to match subsequence $\langle(a)(b)\rangle$ in a sequence and the time complexity can be reduced to $\mathcal{O}(n)$.

4.5 Experiments

In this section we show the experimental results on synthetic data and real Web server access data for evaluating the scalability and effectiveness of the approach MUSE.

Experiments on Synthetic Sequence Data

The scalability of the approach MUSE has been tested first with a fixed belief number of 20 by increasing the size of sequence database from 10,000 sequences to 500,000 sequences, and then with a fixed sequence database size of 100,000 sequences by increasing the number of beliefs from 5 to 25.

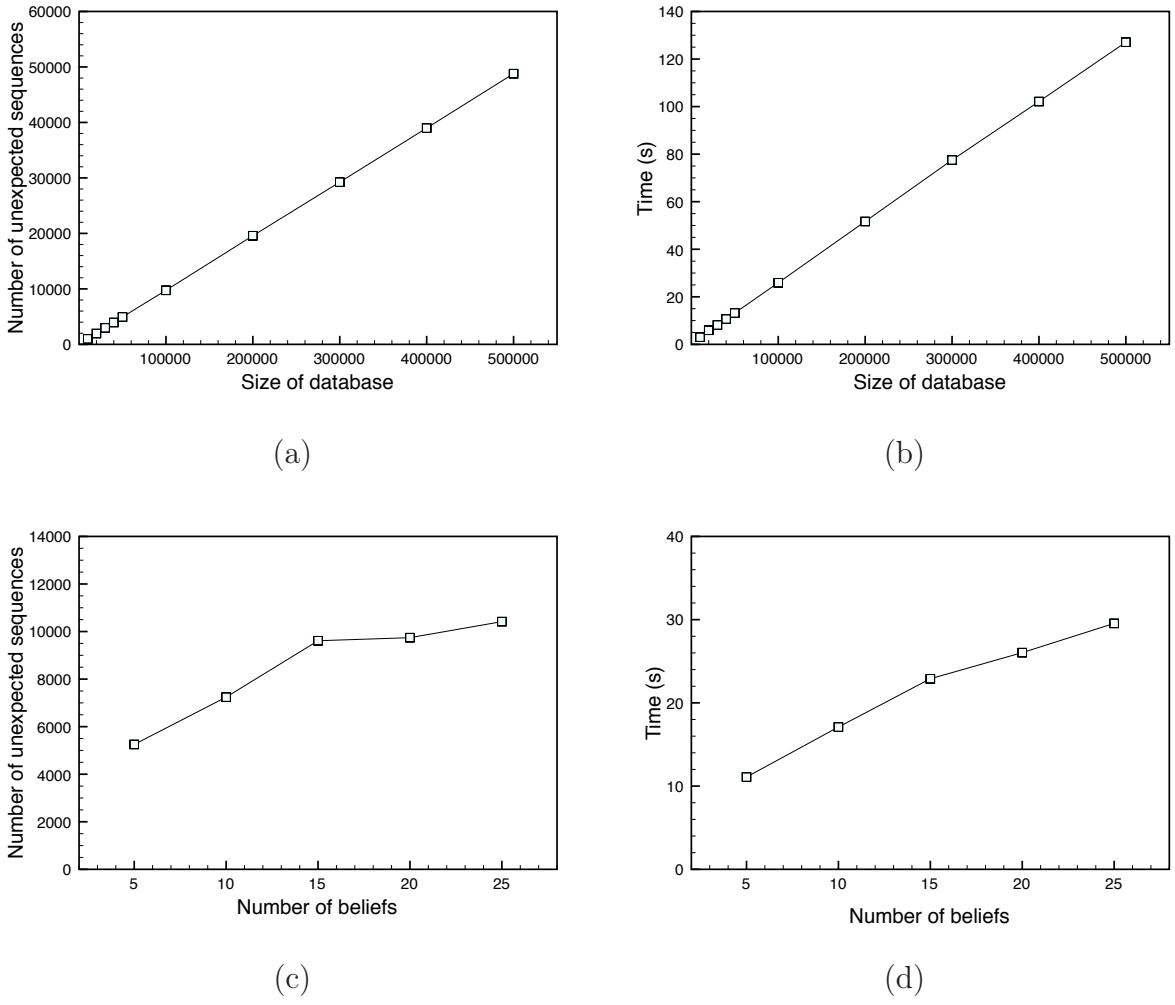


Figure 4.6: Experiments on synthetic data.

Figure 4.6(a) shows that when the belief number is fixed, the number of all unexpected sequences increases linearly with the increasing of the size of sequence database. Because the data sets generated by the IBM Quest Synthetic Data Generator[1] contain repeated blocks, the unexpected sequences with respect to the same 20 beliefs are repeated. Therefore, Figure 4.6(b) shows that, when the belief number is fixed to 20, the run time of the extraction of all unexpected sequences increases linearly with the increasing of the size of sequence database.

Figure 4.6(c) shows that, when the size of sequence database is fixed, the number of all unexpected sequences extracted increases, but not linearly, when the number of beliefs increases. This is a previewed result since the number of unexpected sequences depends on the structure of beliefs. In this test the last 10 beliefs address much less unexpected sequences than others. Figure 4.6(d) shows the increment of run time of the extraction of all unexpected sequences illustrated in Figure 4.6(c), and from which we can find that the increasing rate of extracting time depends on the number of unexpected sequences. In our implementation of the MUSE approach, to predict and process a non-matched sequence is much faster than to predict and process a matched sequence.

Experiments on Web Access Records Data

The effectiveness of the approach MUSE has been tested with Web access data in the framework of Web Usage Mining, which plays an essential role in modem Web applications [BM98, SPF99, MPT00, MDL+00, SCDT00, MDLN02, HKCJ06, MVDA07].

In this experiment, we consider the Web access log in the NCSA Common Logfile Format (CLF, [NCS95]) shown below, which is supported by most mainstream Web servers.

```
remotehost rfc931 authuser [date] "request" status bytes.
```

A Web access log file is generally an ASCII text file, each line contains a CLF log entry that represents a request from a remote client machine to the Web server.

According to the concepts of item, itemset, and sequence, we propose the notion of *session sequence* for representing the user session contained in Web access log entries. Notice that we only consider the `remotehost`, `date`, and `request` fields in our approach for the general-purpose of protecting user privacy.

Definition 13 (Session sequence) *Let $\mathcal{L}$ be an ordered list of Web access log entries and $\ell \in \mathcal{L}$ be a log entry consisting of the properties $\{ip, time, url, query\}$. A session sequence is a sequence*

$$s = \langle (ip, S_0)(\ell_1.url, S_1) \ldots (\ell_n.url, S_n) \rangle,$$

such that:

1. *for any two integers $1 \leq i, j \leq n$ and $i \neq j$, we have $\ell_i.ip = \ell_j.ip$ (denoted as ip);*

[1] *http://www.almaden.ibm.com/cs/projects/iis/hdb/Projects/data_mining/datasets/syndata.html*

2. *for any two integers* $1 \leq i < j \leq n$, *we have* $\ell_i.time < \ell_j.time$;

3. *for any two integers* $1 \leq i < j \leq n$, *we have* $\ell_j.time - \ell_i.time \leq \mu_{max}$, *where* μ_{max} *is the maximum idle time of a session.*

S_0 *is the global parameter set of the session sequence* s. S_i $(1 \leq i \leq n)$ *is the local parameter set of the log entry* ℓ_i.

Given a session sequence s of n $(n > 0)$ log entries, the sequence can be represented as $s = \langle I_0 R_1 R_2 \ldots R_n \rangle$, where $I_0 = (ip, S_0)$ stands for the identification a session and $R_1 = (\ell_1.url, S_1), R_2 = (\ell_2.url, S_2), \ldots, R_n = (\ell_n.url, S_n)$ stand for the requests contained in session. *Notice that in* $R_i = (\ell_i.url, S_i)$, *the index* i *corresponds to the position of the log entry in the user session.* The global parameter set S_0 of the session sequence s can be empty or contain additional information that can be associated with this user session, such as *geographical region*, *time period*, *season* and even *weather*. The local parameter set S_i $(1 \leq i \leq n)$ can also be empty or contain additional information of the log entry ℓ_i, which is mainly considered as the HTTP query of the request.

Example 20 Let us consider the session sequence shown as follows:

$$\langle (\texttt{10.0.0.8}, \texttt{23h}, \texttt{fr})(\texttt{index.php})(\texttt{open.php}, \texttt{p=203}, \texttt{g=5}) \rangle .$$

This sequence represents a user session consisting of two access log entries. The `remotehost` field of this session is `10.0.0.8`, the `date` field is translated to `23h`, and we known the remote host is located in `France`. The page `index.php` without HTTP query was first accessed, i.e., the `request` field is `"index.php"`; the page `open.php` with HTTP query `p=203` and `g=5` was accessed later, which corresponds to the `request` field `"open.php?p=203&g=5"`. □

With the formalization of session sequences, we can apply association rule or sequential pattern mining algorithms for discovering the most general user behaviors of Web sites.

We performed a group of experiments on two large log files containing the access records of two Web servers during a period of 3 months. The first log file, labeled as `LOGBBS`, corresponds to a PHP based discussion forum Web site of an online game provider; the second log file, labeled as `LOGWWW`, corresponds to a Web site that hosts personal home pages of researchers and teaching staffs. We split each log file into three 1-month period files, i.e., `LOGBBS-{1,2,3}` and `LOGWWW-{1,2,3}`. Table 1 details the number of sequences, distinct items, and the average length of the sequences contained in the Web access logs.

The experiments consists of three steps, where each log file corresponds to a belief base, that is, two belief bases are defined, denoted as $\mathcal{B}_{\texttt{BBS}}$ and $\mathcal{B}_{\texttt{WWW}}$. We first define each initial belief base as 5 beliefs constructed from frequent Web usage behaviors and 5 beliefs constructed from

Access Log	Sessions	Distinct Items	Average Length
LOGBBS-1	27,249	38,678	12.8934
LOGBBS-2	47,868	42,052	20.3905
LOGBBS-3	28,146	33,890	8.5762
LOGWWW-1	6,534	8,436	6.3276
LOGWWW-2	11,304	49,242	7.3905
LOGWWW-3	28,400	50,312	9.5762

Table 4.1: Web access logs in experiments.

workflows, denoted as $\mathcal{B}^1_{\texttt{BBS}}$ and $\mathcal{B}^1_{\texttt{WWW}}$, and apply them to discover unexpected sequences in the data sets LOGBBS-1 and LOGWWW-1. Then, we append each belief base with 5 beliefs defined from discovered unexpected sequences, denoted as $\mathcal{B}^2_{\texttt{BBS}}$ and $\mathcal{B}^2_{\texttt{WWW}}$, and apply them to discover unexpected sequences in the data sets LOGBBS-2 and LOGWWW-2. Finally, we append each belief base with 5 beliefs defined from discovered unexpected sequences, denoted as $\mathcal{B}^3_{\texttt{BBS}}$ and $\mathcal{B}^3_{\texttt{WWW}}$, and apply them to discover unexpected sequences in the data sets LOGBBS-3 and LOGWWW-3.

Access Log	Unexpected Sequences
LOGBBS-1 – $\mathcal{B}^1_{\texttt{BBS}}$	1,296
LOGBBS-2 – $\mathcal{B}^2_{\texttt{BBS}}$	11,427*
LOGBBS-3 – $\mathcal{B}^3_{\texttt{BBS}}$	1,512
LOGWWW-1 – $\mathcal{B}^1_{\texttt{WWW}}$	263
LOGWWW-2 – $\mathcal{B}^2_{\texttt{WWW}}$	472
LOGWWW-3 – $\mathcal{B}^3_{\texttt{WWW}}$	1,620

Table 4.2: Number of unexpected sequences.

In the test on data set LOGBBS-2, the number of unexpected sequences is abnormal. After examining the belief base, we find that a new belief defined from the unexpected sequences discovered in the data set LOGBBS-1 is not well defined, which cause a "loop-back" behavior, that is, a sequence unexpected to such a belief corresponds to a frequent behavior. This problem is corrected in the belief base $\mathcal{B}^3_{\texttt{BBS}}$.

The discovery of unexpected sequences can be effective to detect Web frauds or attacks. For instance, in the test on data sets LOGWWW-{1,2,3}, we defined a belief with respect to the workflow of the Web based **MySQL** database management system **phpMyAdmin**[2] as

$$\left\{ \langle(\texttt{sql.php})\rangle \rightarrow \langle(\texttt{index.php})(\texttt{main.php})(\texttt{tbl\_properties\_structure.php})\rangle \right\} \wedge \left\{ \emptyset \right\},$$

which states totally 196 unexpected sequences in all three data sets LOGWWW-{1,2,3}, where all 179

[2] http://www.phpmyadmin.net/

illegal accesses (which have been identified in the whole log file) that tried to access the resource `sql.php` with SQL injection code are detected, and only 17 unexpected sequences are caused by users.

4.6 Discussion

In this chapter, we proposed the framework MUSE for discovering unexpected sequences with respect to the belief system on sequence data. We first defined the belief system, and then we proposed three forms of unexpected sequences with respect to completeness (α-unexpectedness), occurrence (β-unexpectedness), and semantics (γ-unexpectedness) of sequences. We respectively developed three algorithms for discovering α-unexpected sequences (`UxpsMatchAlpha`), β-unexpected sequences (`UxpsMatchBeta`) and γ-unexpected sequences (`UxpsMatchGamma`), and designed the framework MUSE to integrate the discovery process. The scalability and effectiveness are evaluated by synthetic data and real Web server access data.

The proposed belief system consists of sequence rules and semantic contradictions between sequences, where the sequence rules can be discovered in database or defined from domain expertise knowledge, and the semantic contradictions have to manually defined by domain experts. Obviously, the effectiveness of the results strongly relies the specification of the belief system.

The occurrence unexpectedness (β-unexpectedness) stated by the belief system strictly depends on the occurrence constraint on predictive sequence implication rules, however often it might not be precisely observed or defined. Moreover, considering the taxonomy of the items contained in the database, more generalized specification are required to reduce the complexities of constructing the belief system. For instance, given the categories of products, if there exist 10 distinct items for each product category, then even to construct a belief system on three product categories of sequence rules $\texttt{Product}_1 \rightarrow^{\tau} \texttt{Product}_2$ and semantic contradiction $\texttt{Product}_2 \not\approx_{sem} \texttt{Product}_3$, 10^4 sequence rules and 10^2 semantic contradictions must be defined in order to cover all possible combinations of items, and it is obligated to totally generate 10^5 beliefs instead of one belief on the generalization of the taxonomy.

Therefore, in next chapters, we propose the extensions of the framework MUSE with considering fuzzy set theory in sequence occurrence (Chapter 5) and generalizations in discovering unexpected sequences with respect to concept hierarchies of the taxonomy of data (Chapter 6).

On the other hand, a limitation of our current approach is that we do not consider any constraint on the sequences present in a sequence rule. That is, for a sequence rule $s_\alpha \rightarrow^{\tau} s_\beta$, the sequences s_α and s_β do not contain any constraints on their structure. This limitation may effect the discovered unexpected sequences.

Considering a belief consisting of a sequence implication rule $\langle(a)(b)\rangle \rightarrow^* \langle(c)(d)\rangle$ (for simplifying the problem, we do not consider the semantic contradiction in this belief), the following sequences are α-unexpected:

$$\left\{\begin{array}{l} s_1 = \cdots\cdots(a)(b)\cdots\cdots\cdots(c)\cdots\cdots \\ s_2 = \cdots\cdots\cdots(a)(b)\cdots\cdots(c)\cdots\cdots \\ s_3 = \cdots(a)(b)\cdots\cdots\cdots(c)\cdots\cdots\cdots \\ s_4 = \cdots\cdots\cdots\cdots(a)(b)\cdots\cdots(c)\cdots \\ s_5 = (a)\cdots\cdots\cdots\cdots\cdots\cdots\cdots(b) \end{array}\right\}.$$

Not difficult to see, the sequence s_5 has obviously different structure than the other sequences and might be noise data.

Moreover, the sequence rules consider in this thesis do not describe that "if a is *directly* followed by b, then c occurs later", or "if there *does not exist* b between a and c, then d occurs later and *directly* followed by e".

The above problem imposes at least two perspectives on this thesis: (1) to consider complex sequence structure in sequence rules, such as regular expression constrained sequences [GRS99, PHW07]; (2) to refine the discovered unexpected sequences, such as mining outliers [SCA06] in unexpected sequences. These perspectives will be included in our future work.

Chapter 5

Fuzzy Unexpected Sequence Discovery

We proposed the framework MUSE for discovering unexpected sequences in database, with respect to the belief system constructed from prior knowledge. In this chapter, we extend the MUSE framework with two applications of fuzzy set theory: in the extension TAUFU, we measure the occurrence of unexpected conclusion or contradiction sequences with different fuzzy sets; in the extension UFR, we propose a new form of fuzzy sequence rules and discover fuzzy unexpected sequences with respect to the belief system constructed from such rules.

A part of the work presented in this chapter has been published in the *12th International Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2008)*, in the *5th International Conference on Soft Computing as Transdisciplinary Science and Technology (CSTST 2008)*, and in the *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems (IJUFKS)*; has been accepted to be published in the *International Journal of Computational Intelligence Research (IJCIR)*.

5.1 Introduction

In data mining, fuzzy set theory [Zad65] have been many employed to change the domain of the attributes, employing granules defined by fuzzy sets instead of precise values.

For instance, an association rule $X \rightarrow Y$ depicts the relation "if X then Y" between patterns X and Y. With fuzzy sets, there is a very extended way of considering fuzzy association rules as "if X is A then Y is B" in considering various information of attributes (mostly *quantitative attributes* [SA96a]), such as the type "if `beer` is `lot` then `potato chips` is `lot`" or "if `age` is `old` then `salary` is `high`" [CA97, DMSV03, DP06, HLW03, KFW98, hLLk97].

In the same manner, the notion of fuzzy sequential patterns [CTCH01, HCTS03, CH06, FLT07, FMLT08] considers the model sequential patterns like "60% of `young people` purchase a `lot` of `soft drinks`, then purchase `few opera movies` later, then purchase `many PC games`", where the sequence represents "people is `young`, then `soft drinks` is `lot`, then `opera movie` is `few`, and then

PC game is many".

Another application of fuzzy set theory is to discovery *gradual* patterns and rules [Hül02, BCS+07, DJLT08, FMLT08]. In this form of fuzziness in quantitative attributes considers the correlations within the gradual trends of the values of attributes, such as the association rule "if age increases then salary increases", or the sequential pattern "the more visits of search page, the more visits of KB articles later, and at the same time the less visits of question submitting page".

In this chapter, as extensions of the MUSE framework, we consider the binary-valued attributes in databases as other general data mining approaches, however we use fuzzy sets for describing the occurrence and recurrence of sequences.

For instance, if the prior knowledge of customer purchase behaviors indicates that in general the customers purchase a pop music CD within the next 5 purchases after a purchase of an action movie DVD, then a sequence rule can be defined[1] as

$$\langle(\texttt{action movie})\rangle \rightarrow^{[0..5]} \langle(\texttt{pop music})\rangle,$$

for describing that "the intervals between the purchases of action movie and pop music should be no more than 5"; if we further consider that the classical music *semantically contradicts* the pop music, then a semantic contradiction that "a purchase of pop music CD semantically contradicts the purchase of classical music CD" can be applied. We can therefore state the unexpectedness by specifying "after purchasing an action movie DVD, a customer purchases a pop music CD out of the next 5 purchases, or purchases a classical music CD within the next 5 purchases", that is, the following belief:

$$\left\{ \langle(\texttt{action movie})\rangle \rightarrow^{[0..5]} \langle(\texttt{pop music})\rangle \right\} \wedge \left\{ \langle(\texttt{pop music})\rangle \not\approx_{sem} \langle(\texttt{classical music})\rangle \right\}.$$

However, with respect to this belief, if a pop music CD is purchased after 6 other purchases after the purchase of an action movie DVD, then it is difficult to say that it is unexpected because 6 is very close to the upper bound of the range [0..5]; in the same manner, if a classical music CD is purchased after 6 other purchases after the purchase of an action movie DVD, it is also difficult to say that it is expected. Thus, if we consider fuzzy sets in this case, a description like "weak unexpected" could be better then simply concludes "unexpected" or "expected". Therefore, in this chapter, we first extend the MUSE framework with the method TAUFU (Tau-Fuzziness) that takes the fuzzy occurrence of sequences into account.

Other than the unexpectedness on sequence occurrence, the unexpectedness on sequence *recurrence* can be also interesting in the context of sequence data, where elements may occurs repeatedly.

[1] According to our proposition of building beliefs, a sequence rule required by a belief can be either extracted from frequent sequences, or defined by domain experts.

For instance, a custom purchase sequence can be described as "60% of the customers who *often* purchase `action movie DVDs` then `pop music CDs` later, also purchase `PC games` *often*". This kind of correlation between elements can be represented by the sequence rules depicting that "if the sequence s_α repeats in a sequence s, then the sequence s_β repeats in the same sequence s", which reflect the association relation between repeatedly occurred elements in sequence data. With this form of sequence rules, for instance, if we consider that the `classical music` semantically contradicts `PC games`, then the fact "1% customers who *often* purchase `action movie DVDs` then `pop music CDs` later, *often* purchase `classical music CDs`" stands for an unexpected recurrence behavior in a customer purchase database.

Such unexpectedness on sequence recurrence can be interesting for many application domains, including marketing analysis, finance fraud detection, network intrusion detection, Web content personalization, weather prediction, DNA segment analysis, and so on. Therefore, after discussing the fuzzy occurrence of unexpected sequences, in this chapter, we also propose the notion of *fuzzy recurrence rules*, based on the belief system constructed from this form of sequence rules, we further propose an extension UFR (Unexpected Fuzzy Recurrence) for the MUSE framework.

The rest of this chapter is organized as follows. In Section 5.2, we propose the extension *tau-fuzziness* of unexpected sequences that considers the fuzziness on the occurrence constraint τ of sequence implication rules, and develop the method TAUFU for discovering fuzzy unexpected sequences with *tau-fuzziness*. In Section 5.3, we propose a new form of sequence rules so called the *fuzzy recurrence rules* and develop the method UFR for discovering unexpected fuzzy recurrence sequences with respect to beliefs of fuzzy recurrence rules. Finally, we discuss fuzzy unexpectedness in Section 5.4.

5.2 Fuzzy Unexpectedness in Sequence Occurrence

In this section, we first extend the notions of unexpected sequences with the fuzziness on the occurrence constraint τ of sequence implication rules, and then develop the method TAUFU for discovering fuzzy unexpected sequences with the *tau-fuzziness* extension.

5.2.1 Tau-Fuzzy Unexpected Sequences

An *unexpected sequence* is a sequence that violates a belief. In the framework MUSE proposed in the previous chapter, the unexpectedness is stated by the violation of sequence rules or semantic contradiction contained in a belief. We now extend the framework MUSE with the fuzziness on the occurrence constraint τ (so called the *tau-fuzziness*) of sequence implication rules $s_\alpha \rightarrow^\tau s_\beta$. Notice that we do not consider *tau-fuzziness* on sequence association rules $s_\alpha \rightarrow^\emptyset s_\beta$, since we have that $\tau = \emptyset$ for sequence association rules.

According to the structure of sequence implication rules $s_\alpha \rightarrow^\tau s_\beta$ with respect to semantic contradictions $s_\alpha \not\approx_{sem} s_\beta$, three forms of *unexpected sequences* can be mentioned as following: an α-unexpected sequence is unexpected because the occurrence of s_β is missing when $\tau = *$; a β-unexpected sequence is unexpected because the occurrence of s_β violates the constraint τ; a γ-unexpected sequence is unexpected because the occurrence of s_γ with respect to τ violates the semantic contradiction $s_\beta \not\approx_{sem} s_\gamma$.

Not difficult to see, the *tau-fuzziness* is not applicable to α-unexpected sequence stated from a belief on sequence implication rules $s_\alpha \rightarrow^* s_\beta$ and semantic contradictions $s_\alpha \not\approx_{sem} s_\beta$, because there does not exist fuzziness of the occurrence of the sequences s_β or s_γ when the occurrence constraint $\tau = *$, where the existence of s_α or s_β can only be measured by boolean value *true* or $false$.

As defined in Chapter 4, given a belief consisting of a predictive sequence implication rule $s_\alpha \rightarrow^{\tau \neq *} s_\beta$ and a semantic contradiction $s_\beta \not\approx_{sem} s_\gamma$, the discovery of β-unexpectedness or γ-unexpectedness in a sequence s can be determined by examining whether there does exist any sequence s' such that $|s'| \models \tau$ and $s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s$, or whether there exists a sequence s' such that $|s'| \models \tau$ and $s_\alpha \cdot s' \cdot s_\gamma \sqsubseteq_c s$. Therefore, we propose the notion of *tau-fuzziness* on the satisfiability $|s| \models \tau$ between the length of the sequence s and the occurrence constraint τ, denoted as $|s| \models (\tau, d_\tau, \mu_\tau)$, in order to measure the fuzzy unexpectedness of β- and γ-unexpected sequences, where d_τ is a fuzzy degree and μ_τ is a fuzzy membership function.

The *tau-fuzzy satisfaction* $|s| \models (\tau, d_\tau, \mu_\tau)$ of the length of a sequence s can be interpreted as follows. Given a fuzzy membership function $\mu_\tau(|s|, F)$ which returns the fuzzy membership degree of the length of the sequence s in fuzzy set F with respect to the range specified by the occurrence constraint τ. Let $\mathcal{F}$ be a set of predefined fuzzy sets on μ_τ, if there exists a fuzzy set $F \in \mathcal{F}$ such that $\mu_\tau(|s|) \geq d_\tau$, then we say that the length of the sequence s satisfies the occurrence constraint τ with respect to *tau-fuzziness* defined by μ_τ.

Therefore, the *tau-fuzzy* β-unexpected and γ-unexpected sequences can be formally defined as the following definitions.

Definition 14 (*Tau-fuzzy* β-unexpected sequence) *Given a sequence s and a belief b of consistent sequence implication rules, let $s_\alpha = \Lambda(b)$. Let μ_τ be a fuzzy membership function and d_τ be a minimum tau-fuzzy membership degree, if there exists $s_\beta \in \Delta(b)$ such that for each rule $(s_\alpha \rightarrow^{\tau_i} s_\beta)$ contained in the belief b we have not that $(s_\alpha \sqsubseteq s) \wedge (s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s) \wedge (|s'| \models (\tau, d_\tau, \mu_\tau))$, then the sequence s is a tau-fuzzy occurrence-unexpected sequence with respect to the belief b, denoted as $s \nVdash^\tau_\beta b$. We also call such an unexpected sequence a tau-fuzzy β-unexpected sequence.*

Definition 15 (*Tau-fuzzy* γ-unexpected sequence) *Given a sequence s and a belief b of consistent sequence implication rules, let $s_\alpha = \Lambda(b)$. Let μ_τ be a fuzzy membership function and d_τ be a*

minimum tau-fuzzy membership degree, if $s_\alpha \sqsubseteq s$*, and if there exists a sequence rule* $r \in \mathcal{R}$ *and a semantic contradiction* $(s_{\beta_i} \not\approx_{sem} s_{\gamma_j}) \in \mathcal{M}$ *such that* $(s_\alpha \sqsubseteq s) \wedge (s_\alpha \cdot s' \cdot s_\gamma \sqsubseteq_c s) \wedge (|s'| \models (\tau, d_\tau, \mu_\tau))$*, then the sequence* s *is a tau-fuzzy semantics-unexpected sequence with respect to the belief* b*, denoted as* $s \not\Vdash^\tau_\gamma b$*. We also call such an unexpected sequence a tau-fuzzy* γ*-unexpected sequence.*

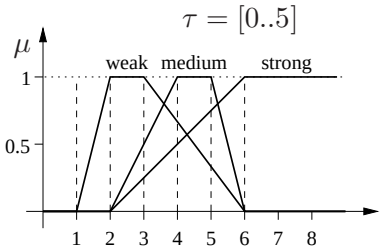


Figure 5.1: Fuzzy sets for β-unexpectedness.

Example 21 We consider a belief on Web site log files, where `home`, `login`, and `logout` stand for the URL resources visited in a user session:

$$b = \left\{ \langle(\texttt{home})\rangle \rightarrow^{[0..5]} \langle(\texttt{login})\rangle \right\} \wedge \left\{ \langle(\texttt{login})\rangle \not\approx_{sem} \langle(\texttt{logout})\rangle \right\}.$$

We consider three fuzzy sets for the each unexpectedness, they are "weak unexpected" (F_w), "medium unexpected" (F_m) and "strong unexpected" (F_s). In a sequence

$$s = \langle(\texttt{home})(\texttt{ad1})(\texttt{ad2})(\texttt{ad3})(\texttt{ad4})(\texttt{login})\rangle\,,$$

we have that $|(\texttt{ad1})(\texttt{ad2})(\texttt{ad3})(\texttt{ad4})| = 4$. Let $\mathcal{F} = \{F_w, F_m, F_s\}$, according to the fuzzy membership functions shown in Figure 5.1, we have that $\mu_\tau(4, F_w) = 0.67$, $\mu_\tau(4, F_m) = 1$ and $\mu_\tau(4, F_s) = 0.5$, so that the best description of the sequence s is "medium unexpected". □

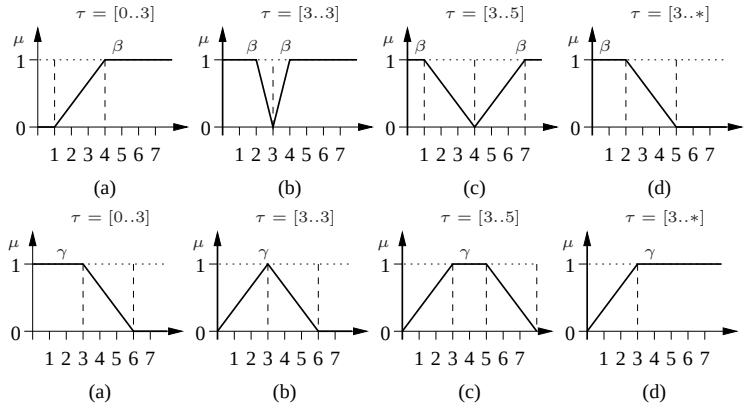


Figure 5.2: Fuzzy sets of the "strong unexpected".

For more details of the fuzziness on the occurrence constraint τ, Figure 5.2 represents "strong unexpected" for β-unexpectedness and γ-unexpectedness with (a) $\tau = [0..3]$, (b) $\tau = [3..3]$, (c) $\tau = [3..5]$ and (d) $\tau = [3..*]$.

5.2.2 Approach TAUFU

In this section, we develop the approach TAUFU (*Tau-Fuzzy*), which include the algorithms `TaufuMatchBeta` and `TaufuMatchGamma` for extending the framework MUSE with the *tau-fuzziness* of unexpected sequences.

The algorithm `TaufuMatchBeta` (Algorithm 7) matches the *tau-fuzzy* β-unexpectedness in a sequence, which can be a replacement of the β-unexpectedness matching routine `UxpsMatchBeta` (Algorithm 3, Section 4.3.2).

Algorithm 7: `TaufuMatchBeta` (T, s, pos) : Matching *tau-fuzzy* β-unexpectedness.

Input : A belief T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in s.

Output : The set of all occurrences of *tau-fuzzy* β-unexpectedness in s with respect to T.

```
1  uxps := TupleSet.Create();
2  n_τ := T.firstTauNode();
3  while n_τ ≠ null and n_τ ∉ N do
4      if n_τ.data.min = −1 then
5          continue;    /* skip sequence association rules */
6      if n_τ.data.min = 0 or n_τ.data.max = −1 then
7          continue;    /* skip simple sequence implication rules */
8      F := FuzzySets(T.id, n_τ.id, BETA);
9      μ_τ := FuzzyMembershipFunction(T.id, n_τ.id, BETA);
10     n_sβ := n_τ.firstSubNode();
11     while n_sβ ≠ null do
12         u := SeqMatchTaufu(n_sβ.data, s, pair(pos.second + 1, |s| − 1));
13         if u.pos.first ≠ −1 then
14             uxps.add(tuple(s.id, u.pos.first, u.pos.second, u.taufu.first, u.taufu.second));
15             if options | FIRST_UXPS_ONLY then    /* use the conclusion of Lemma 1 */
16                 return uxps;
17     n_τ := T.nextTauNode(n_τ);
18 return uxps;
```

The algorithm accepts a belief T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in the sequence s as inputs, and outputs all occurrences of *tau-fuzzy* β-unexpectedness stated in s.

For each occurrence constraint τ, the algorithm first retrieves the fuzzy sets $\mathcal{F}$ and the fuzzy membership function μ_τ associated with the β-unexpectedness stated by the belief represented by $T.id$ by calling $FuzzySets(T.id, n_\tau.id, \texttt{BETA})$ and $FuzzyMembershipFunction(T.id, n_\tau.id, \texttt{BETA})$, where the fuzzy sets and fuzzy membership function associated with τ can be determined by the

belief tree ID $T.id$ and the τ-node ID $n_\tau.id$. The algorithm then matches the best[2] *tau-fuzzy* occurrence of the sequence s_β contained in the s-node n_s in the sequence s. Finally, a set contains all occurrences of *tau-fuzzy* β-unexpectedness stated in s is returned.

The algorithm `TaufuMatchGamma` (Algorithm 8) matches the *tau-fuzzy* γ-unexpectedness in a sequence, which can be a replacement of the γ-unexpectedness matching routine `UxpsMatchGamma` (Algorithm 5, Section 4.3.3).

Algorithm 8: `TaufuMatchGamma` (T, s, pos) : Matching *tau-fuzzy* γ-unexpectedness.

Input : A belief tree T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in s.

Output : The set of all occurrences of *tau-fuzzy* γ-unexpectedness in s with respect to T.

```
1  uxps := TupleSet.Create();
2  range := pair(−1, −1);
3  n_τ := T.firstTauNode();
4  while n_τ ≠ null do
5      F := FuzzySets(T.id, n_τ.id, GAMMA);
6      μ_τ := FuzzyMembershipFunction(T.id, n_τ.id, GAMMA);
7      n_{s_γ} := n_τ.firstLinkedNode();
8      while n_{s_γ} ≠ null do
9          u := SeqMatchTaufu(n_{s_γ}.data, s, pair(pos.second + 1, |s| − 1));
10         if u.first ≠ −1 then
11             uxps.add(tuple(s.id, u.pos.first, u.pos.second, u.taufu.first, u.taufu.second));
12             if options | FIRST_UXPS_ONLY then      /* first occurrence of γ-unexpectedness */
13                 return uxps;
14         n_{s_γ} := n_τ.nextLinkedNode(n_{s_γ});
15     n_τ := T.nextTauNode(n_τ);
16 return uxps;
```

The algorithm accepts a belief T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in the sequence s as inputs, and outputs all occurrences of *tau-fuzzy* γ-unexpectedness stated in s.

`TaufuMatchGamma` follows the principle of the algorithm `UxpsMatchGamma`, however it uses the subroutine `SeqMatchTaufu` instead of the subroutine `SeqMatch`, that is, to find the best *tau-fuzzy* occurrence of the contradiction sequence s_γ contained in the s-node n_{s_γ}.

The subroutine `SeqMatchTaufu` mentioned in `TaufuMatchBeta` and `TaufuMatchGamma` is listed in Algorithm 9. The algorithm accepts a sequence s, a sequence s', and a pair $range$ for bounding

[2]The selection of the *best* occurrence of *tau-fuzzy* unexpectedness by the subroutine `SeqMatchTaufu` listed in Algorithm 9.

the occurrence of s in s' as inputs, and outputs the best *tau-fuzzy* occurrence of s in s'.

Algorithm 9: `SeqMatchTaufu` (s, s', $range$) : Matching best *tau-fuzzy* sequence.

Input : A sequence s, a sequence s', and a pair $range$.
Output : Best *tau-fuzzy* occurrences of s in s' with respect to $range$.

```
1  rank := Rank.Create();
2  while range.first ≠ −1 do
3      uxp := SeqMatchFirst(s, s′, pair(range.first, range.second));
4      while uxp.first ≠ −1 do
5          len := uxp.first − pos.second − 1;
6          foreach F ∈ F do      /* F is accessible in global scope */
7              d_τ := μ_τ(len, F);    /* μ_τ is accessible in global scope */
8              if d_τ ≥ tauf_min then      /* tauf_min is accessible in global scope */
9                  rank.add(F, d_τ, uxp.first, uxp.second);
10         uxp := SeqMatchFirst(s, s′, pair(uxp.first + 1, range.second));    /* next s_β */
11     range := SeqMatchFirst(s.last, s′, pair(range.first, range.second));    /* next s_{α⊥} */
12     if range.first ≠ −1 then
13         range.first := range.first + 1;
14 if rank = ∅ then
15     return pair(−1, −1);
16 best := rank.top();
17 return pair(best.pos, best.taufu);
```

We explain the algorithm `SeqMatchTaufu` with a running example, in which we illustrate the routine of matching a *tau-fuzzy* β-unexpected sequence

$$s = \langle (11)(11)(12)(21)(12)(22)(21)(22)(21)(12) \rangle$$

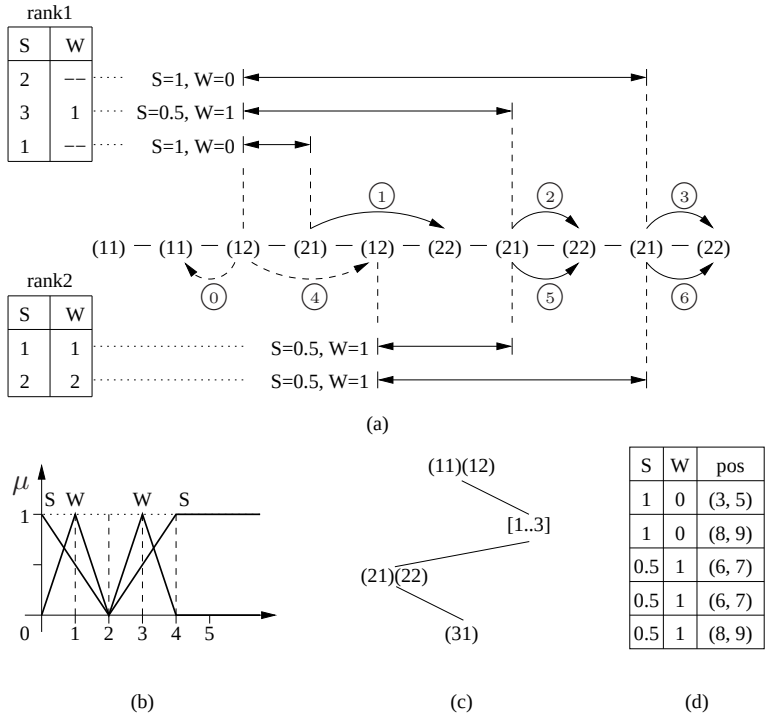
shown in Figure 5.3, where the numbers stand for event IDs. We consider two fuzzy sets "weak unexpected" (labeled as W) and "strong unexpected" (labeled as S) for describing *tau-fuzzy* β-unexpectedness stated by the belief

$$b = \left\{ \langle (11)(12) \rangle \rightarrow^{[1..3]} \langle (21)(22) \rangle \right\} \wedge \left\{ \langle (21)(22) \rangle \not\approx_{sem} \langle (31) \rangle \right\},$$

shown in Figure 5.3(b).

In this example, we illustrate how `SeqMatchTaufu` extracts the *tau-fuzzy* β-unexpectedness from s.

A first minimal premise sequence $s_\alpha = \langle (11)(12) \rangle$ is found by `SeqMatchMin`, that is, the step ⓪, so that the task is to find the best *tau-fuzzy* occurrence of the conclusion sequence $s_\beta = \langle (21)(22) \rangle$. Therefore, the algorithm `SeqMatchTaufu` starts matching *tau-fuzzy* occurrence of $\langle (21)(22) \rangle$ from the position 3 (the first itemset in s is considered as the position 0) till to end of the sequence s, and finds 3 occurrences of $\langle (21)(22) \rangle$ as shown as the steps ①, ②, and ③ with the first loop of the



(a) A *tau-fuzzy* β-unexpected sequence matching routine. (b) Fuzzy sets on the occurrence constraint $\tau = [1..3]$. (c) A branch of the belief tree. (d) Rank table of matched occurrences.

Figure 5.3: Illustration of a *tau-fuzzy* β-unexpected sequence extraction.

while block within the line 2 and the line 13 in Algorithm 9. The fuzzy membership degrees of each occurrence of $\langle(21)(22)\rangle$ are sorted as listed in the table $rank1$ shown in Figure 5.3(a). The occurrence with higher fuzzy membership degree value has better rank; if two occurrences have the same degree, the earlier matched occurrence has better rank. The algorithm continues to find the next position of the last itemset of s_β (i.e., $s_{\alpha\perp}$) as shown as the step ④, and the second loop finds two occurrences of $\langle(21)(22)\rangle$ as shown as the steps ⑤ and ⑥, which are listed in the table $rank2$.

The final order of all matched occurrences of s_β is ranked by 3 criteria: (1) fuzzy membership degree; (2) occurrence position; (3) priority of fuzzy set. In this example, we have that the priority of "strong unexpected" is higher than the priority of "weak unexpected" since we are discovering unexpected sequence, so that the final rank of each occurrence of $\langle(21)(22)\rangle$ is listed in the table shown in Figure 5.3(d) and the algorithm `SeqMatchTaufu` returns the occurrence of $\langle(21)(22)\rangle$ at the position $pos = (3, 5)$ with (stored in the pair $best.pos$) membership degree 1 of "strong unexpected" (stored in the pair $best.taufu$).

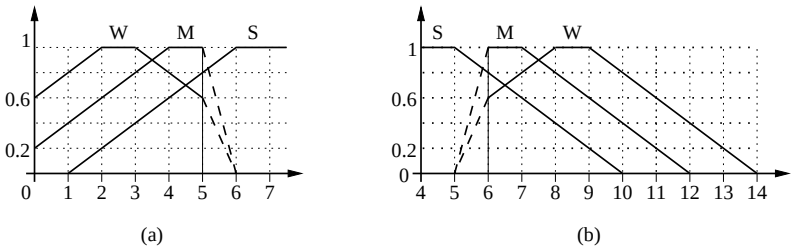
5.2.3 Experiments

To evaluate the effectiveness of the approach TAUFU, we performed a group of experiments to extract unexpected sequences in the access records of a security testing Web server, where a large

number of attacks are logged. The sequence database converted from the access log file contains 67,228 session sequences corresponding to 27,552 distinct items.

Totally 4 groups of 20 beliefs corresponding to 4 categories of occurrence constraints are considered in our experiments: `CAT1` stands for 5 beliefs with $\tau = [0..*]$; `CAT2` stands for 5 beliefs with $\tau = [0..X]$ where $X \geq 0$ is an integer; `CAT3` stands for 5 beliefs with $\tau = [Y..*]$ where $Y > 0$ is an integer; and `CAT4` stands for 5 beliefs with $\tau = [X..Y]$ where $Y \geq X > 0$ are two integers.

To simplify the procedure of our experiments, the ratio of membership function μ is fixed to ± 0.2 for all fuzzy sets "weak unexpected" (W), "medium unexpected" (M), and "strong unexpected" (S). Further, the sets "weak unexpected" and "medium unexpected" do not cover the interval ranges where the membership degree of "strong unexpected" is 1. The interval value of the fuzzy sets "weak unexpected" and "medium unexpected" is fixed to 2 when the membership degree equals 1.



(a) β-unexpected fuzzy sets. (b) γ-unexpected fuzzy sets.

Figure 5.4: Fuzzy sets considered in the expriments

For instance, Figure 5.4 shows the fuzzy sets for a belief in `CAT2`

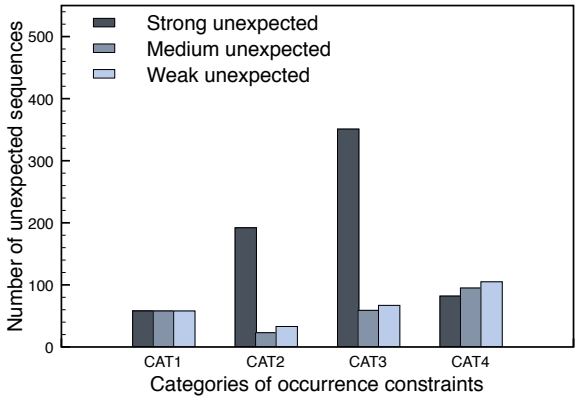
$$\left\{ \langle(\texttt{viewforum})\rangle \rightarrow^{[0..5]} \langle(\texttt{viewtopic})\rangle \right\} \wedge \left\{ \langle(\texttt{viewtopic})\rangle \not\rightarrow_{sem} \langle(\texttt{login})\rangle \right\},$$

the fuzzy partitions are shown in Figure 5.4. The numbers of unexpected sequences (β-unexpected and γ-unexpected) that we find with respect to $tauf_{min} = 1$, $tauf_{min} = 0.7$, and $tauf_{min} = 0.2$ are listed in Table 5.1 with comparing $unexpectedness/tauf_{min}$.

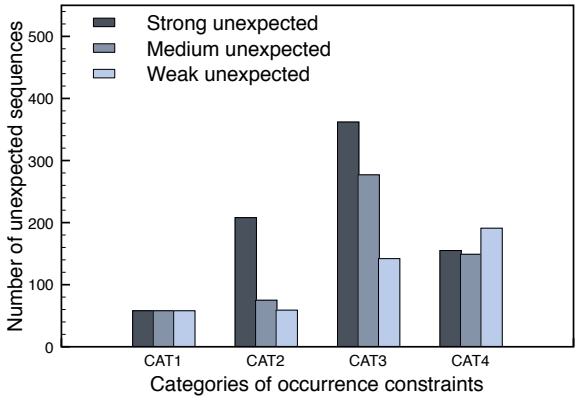
	$\beta/1$	$\gamma/1$	$\beta/0.7$	$\gamma/0.7$	$\beta/0.2$	$\gamma/0.2$
Strong unexpected	47	22	49	23	55	25
Medium unexpected	4	2	7	2	10	6
Weak unexpected	4	1	5	5	6	12

Table 5.1: Number of unexpected sequences stated by a belief in `CAT2`.

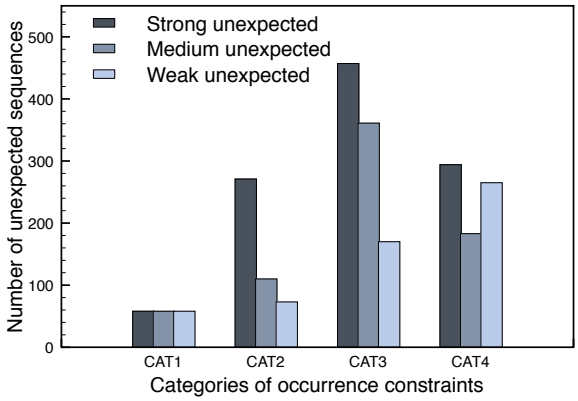
For the fuzzy sets "strong unexpected", "medium unexpected" and "weak unexpected",Figure 5.5(a) shows the total number of tau-fuzzy unexpected sequences with minimum fuzzy degree $tauf_{min} = 1.0$, Figure 5.5(b) shows the total numbers of tau-fuzzy unexpected sequences with minimum fuzzy degree $tauf_{min} = 0.7$, and Figure 5.5(c) shows the total numbers of tau-fuzzy unexpected sequences with minimum fuzzy degree $tauf_{min} = 0.2$.



(a) $tauf_{min} = 1.0$.



(b) $tauf_{min} = 0.7$.



(c) $tauf_{min} = 0.2$.

Figure 5.5: Number of tau-fuzzy unexpected sequences.

In our sequence database of Web intrusion data, the experiments show that the beliefs in `CAT2` and `CAT3` drive a clear view of the unexpectedness, that is, with changing the minimum fuzzy membership degree, the number of strong unexpected sequences does not considerably change, and the number of medium and weak unexpected sequences is lower than the number of strong unexpected sequences, so that they can be considered as noise in the data. However, the number of unexpected sequences stated by the beliefs in `CAT4` show much lower precision in unexpectedness discovery, and the studies in such unexpected sequences have more importance for improving the belief base.

5.3 Unexpected Fuzzy Recurrences in Sequence Data

In this section, we present the problem of discovering unexpected fuzzy recurrence sequences. We first propose the notion of *fuzzy recurrence sequence*, with which we further propose the *fuzzy recurrence rule* as a complement of the forms of sequence rules proposed in Chapter 4. With respect to the beliefs consisting of fuzzy recurrence sequence rules, we therefore propose the discovery of unexpected fuzzy recurrence sequences.

5.3.1 Fuzzy Recurrence Rules

In many applications, *recurrence behaviors* are often present in sequence data. For instance, customers often purchase the products with the same brand; the price of some stocks repeatedly change in the same manners; Web users may repeatedly access the same resources; certain segments repeatedly appear in DNA sequence, and so on.

To study the repeatedly occurred elements in sequences, we first propose the notion of *recurrence sequence* in the form $\langle s, \psi \rangle$, where s is a sequence and ψ is a positive integer. If a sequence s' *supports* a recurrence sequence $\langle s, \psi \rangle$, then the sequence s occurs in s' at least ψ times, denoted as $\langle s, \psi \rangle \sqsubseteq s'$, that is,

$$(\langle s, \psi \rangle \sqsubseteq s') \iff (\underbrace{s \cdots s}_{n} \sqsubseteq s') \wedge (n \geq \psi).$$

A recurrence sequence $\langle s, \psi \rangle$ is also called a *ψ-recurrence sequence*. We use the wildcard "$*$" for denoting the general meaning of the support between sequences, that is,

$$(\langle s, * \rangle \sqsubseteq s') \equiv (s \sqsubseteq s').$$

A *recurrence rule* is a rule on sequences with form $\langle s_\alpha, \psi \rangle \rightarrow \langle s_\beta, \theta \rangle$, where s_α, s_β are two sequences, and ψ, θ are two integers for describing recurrence behaviors in sequence data. A recurrence rule indicates the association relation that given a sequence s, if s_α orderly occurs no

less than ψ times within s, then orderly s_β occurs in s no less than θ times, that is,

$$(\underbrace{s_\alpha \cdots s_\alpha}_{n} \sqsubseteq s) \wedge (n \geq \psi) \Rightarrow (\underbrace{s_\beta \cdots s_\beta}_{k} \sqsubseteq s) \wedge (k \geq \theta).$$

Given a sequence s and a recurrence rule $r = \langle s_\alpha, \psi \rangle \rightarrow \langle s_\beta, \theta \rangle$, if $\langle s_\alpha, \psi \rangle \sqsubseteq s$ and $\langle s_\beta, \theta \rangle \sqsubseteq s$, then we say that s *supports* r, denoted as $s \models r$. For instance, the recurrence rule $r = \langle (a)(b), 3 \rangle \rightarrow \langle (c)(d), * \rangle$ depicts that given a sequence s, if $\langle (a)(b) \rangle$ is contained repeatedly in s no less 3 times, then $\langle (c)(d) \rangle$ is contained in s; in other words, if $\langle (a)(b)(a)(b)(a)(b) \rangle \sqsubseteq s$, then $\langle (c)(d) \rangle \sqsubseteq s$.

Notice that the occurrences of s_α must be ordered, that is, for example, given a rule $r_1 = \langle (a)(b), 2 \rangle \rightarrow \langle (c), * \rangle$, the sequence $s_1 = \langle (a)(a)(c)(b)(b) \rangle$ does not support r_1, but the sequence $s_2 = \langle (a)(b)(c)(a)(b) \rangle$ supports r_1; however, the sequence s_1 supports the rules $r_2 = \langle (a), 2 \rangle \rightarrow \langle (c), * \rangle$ and $r_3 = \langle (b), 2 \rangle \rightarrow \langle (c), * \rangle$.

Considering the integer ψ, a human-friendly interpretation is more flexible and more relevant to described the recurrence in sequence data. For instance, in market basket analysis, to point out that "the customers who often purchase `action movie DVDs` often purchase `pop music CDs`" is more relevant than the conclusion "the customers who purchase at least 7 times of `action movie DVDs` purchase at least 5 times of `pop music CDs`".

We therefore extend the recurrence rule with fuzzy sets, so called the *fuzzy recurrence rule*, in the form $\langle s_\alpha, \zeta_\alpha \rangle \rightarrow \langle s_\beta, \zeta_\beta \rangle$, where ζ_α and ζ_β are two fuzzy sets for describing s_α and s_β, and the sequences $\langle s_\alpha, \zeta_\alpha \rangle$ and $\langle s_\beta, \zeta_\beta \rangle$ are two fuzzy recurrence sequences. Given a sequence s' and a fuzzy recurrence rule $\langle s, \zeta \rangle$, that s' *supports* $\langle s, \zeta \rangle$ is defined as

$$(\langle s, \zeta \rangle \sqsubseteq s') \iff (\underbrace{s \cdots s}_{n} \sqsubseteq s) \wedge (\mu_\zeta(n) \geq recu_{min}), \tag{5.1}$$

where the fuzzy degree measured by the membership function $\mu_\zeta(n)$ must be superior or equal to a threshold $recu_{min}$.

Let us consider the following example.

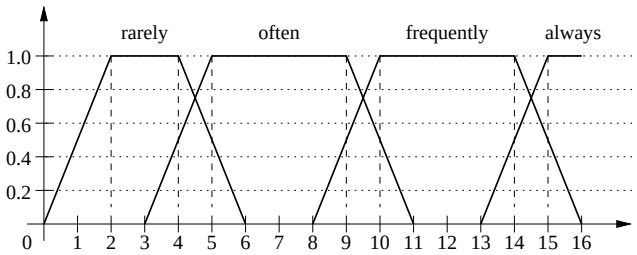


Figure 5.6: Fuzzy sets for describing recurrence rules.

Example 22 Given a set of distinct events $a, b, c, d, \ldots$, an ordered set of events can be represented as the data model of sequence. Assuming that given an event sequence s, if s supports the

recurrence sequence $\langle(a)(b),4\rangle$, then s supports the subsequence $\langle(c)(d)\rangle$; if s supports the recurrence sequence $\langle(a)(b),9\rangle$, then s supports $\langle(c)\rangle$. These behaviors can be described by recurrence rules, such as the rule $r_1 = \langle(a)(b),4\rangle \rightarrow \langle(c)(d),*\rangle$ and the rule $r_2 = \langle(a)(b),9\rangle \rightarrow \langle(c),*\rangle$. Given a sequence s_1 such that $\langle(a)(b),3\rangle \sqsubseteq s_1$ and $\langle(c)(d)\rangle \sqsubseteq s_1$, a sequence s_2 such that $\langle(a)(b),8\rangle \sqsubseteq s_2$ and $\langle(c)\rangle \sqsubseteq s_2$, we have $s_1 \not\models r_1$ and $s_2 \not\models r_2$. However, since the recurrence sequences contained in these sequences and rules are close, the sequences s_1 and s_2 can be still potentially interesting. On the other hand, considering the fuzzy recurrence rules $r_1{}' = \langle(a)(b),rarely\rangle \rightarrow \langle(c)(d),*\rangle$ and $r_2{}' = \langle(a)(b),often\rangle \rightarrow \langle(c),*\rangle$, corresponding to the rules r_1 and r_2 with respect to the fuzzy partitions shown in Figure 5.6, let the threshold $recu_{min} = 0.5$, then we have $s_1 \models r_1{}'$ and $s_2 \models r_2{}'$. We can further define more partitions, such as "always" or "rarely". □

In this thesis, the fuzzy recurrence rules are considered as having been predefined by domain experts, the discovery of fuzzy recurrence rules will be covered in our future research work.

5.3.2 Unexpected Fuzzy Recurrences

We are considering to discover the sequences contained in a database those semantically contradict a given set of fuzzy recurrence rules. In order to find such sequences, we construct a belief base from given fuzzy recurrence rules with semantic contradictions between fuzzy recurrence sequences, so that each sequence not respecting the belief base is unexpected.

The belief system presented in Chapter 4 can be extended to handle fuzzy recurrence rules without any changes.

Let $\langle s_\alpha,\zeta_\alpha\rangle \rightarrow \langle s_\beta,\zeta_\beta\rangle$ be a fuzzy recurrence rule and $\langle s_\beta,\zeta_\beta\rangle \not\approx_{sem} \langle s_\gamma,\zeta_\gamma\rangle$ be a semantic contradiction, where ζ_γ is a fuzzy set for the sequence s_γ. The fuzzy recurrence rule implies an association relation between the fuzzy recurrences $\langle s_\alpha,\zeta_\alpha\rangle$ and $\langle s_\beta,\zeta_\beta\rangle$ that if the recurrence of s_α is ζ_α, then the recurrence of s_β is ζ_β. The semantic contradiction then implies that the recurrence sequences $\langle s_\beta,\zeta_\beta\rangle$ and $\langle s_\gamma,\zeta_\gamma\rangle$ semantically contradict each other.

The notion of consistent sequence rule set can also be applied to fuzzy recurrence rules, that is, a *consistent fuzzy recurrence rule set* is the set of fuzzy recurrence rules where all the rules has the same *premise sequence* $\langle s_\alpha,\zeta_\alpha\rangle$. We also directly use the notions of conclusion sequence set and contradiction sequence set defined in Chapter 4.

Given a belief $b = \mathcal{R} \wedge \mathcal{M}$, let $\langle s_\alpha,\zeta_\alpha\rangle = \Lambda(b)$ be the premise sequence, $\Delta(b)$ be the conclusion sequence set, and $\Theta(b,\langle s_\beta,\zeta_\beta\rangle)$ be the contradiction sequence set, where $\langle s_\beta,\zeta_\beta\rangle \in \Delta(b)$ is a conclusion sequence. Such a belief depicts that given a sequence s, if s supports $\langle s_\alpha,\zeta_\alpha\rangle$, then s supports at least one $\langle s_\beta,\zeta_\beta\rangle \in \Delta(b)$, however s should not support any $\langle s_\gamma,\zeta_\gamma\rangle \in \Theta(b,\langle s_\beta,\zeta_\beta\rangle)$

for each $\langle s_\beta, \zeta_\beta \rangle \in \Delta(b)$, that is,

$$\begin{aligned} (\langle s_\alpha, \zeta_\alpha \rangle \sqsubseteq s) \quad &\wedge \quad (\exists \langle s_\beta, \zeta_\beta \rangle \in \Delta(b), \langle s_\beta, \zeta_\beta \rangle \sqsubseteq s) \\ &\wedge \quad (\forall \langle s_\beta, \zeta_\beta \rangle \in \Delta(b), \nexists \langle s_\gamma, \zeta_\gamma \rangle \in \Theta(b, \langle s_\beta, \zeta_\beta \rangle), \langle s_\gamma, \zeta_\gamma \rangle \sqsubseteq s). \end{aligned} \tag{5.2}$$

Notice that s_β and s_γ are not necessary to be different: we have that $\langle (\texttt{game}), rarely \rangle$ and $\langle (\texttt{game}), always \rangle$ semantically contradict each other.

Example 23 Assume that the customers who purchase `music` and `movies` like to play `games`. If we consider that `games` and `books` semantically contradict each other, where the semantic contradiction can be $\langle (\texttt{game}), often \rangle \not\approx_{sem} \langle (\texttt{book}), often \rangle$, then a belief can be defined as

$$\left\{ \langle (\texttt{music movie}), often \rangle \rightarrow \langle (\texttt{game}), often \rangle \right\} \wedge \left\{ \langle (\texttt{game}), often \rangle \not\approx_{sem} \langle (\texttt{book}), often \rangle \right\}.$$

The fuzzy sets for the purchase of classical music CDs can also be that shown in Figure 5.6. The above belief describes that the customers who *often* purchase `music` and `movies` also purchase `games` *often*, however do not *often* purchase `books`. □

Given a belief b, if a sequence s satisfies Equation (5.2), then we say that the sequence s *supports* the belief b, denoted as $s \models b$. A sequence s *unexpected* to a belief b is denoted as $s \not\models b$.

In Chapter 4 we proposed 3 forms of unexpectedness. Obviously, the α-unexpectedness is not applicable to recurrence rules since there does not exist occurrence constraint in recurrence rules. The occurrence of a conclusion sequence $\langle s_\beta, \zeta_\beta \rangle$ can be violated, if there exists s_β in a sequence however the recurrence of s_β does not satisfies ζ_β. Therefore, we consider two forms of unexpectedness in our approach with respect to the occurrence of a conclusion sequence $\langle s_\beta, \zeta_\beta \rangle$ and a contradiction sequence $\langle s_\gamma, \zeta_\gamma \rangle$ contained in a belief.

Definition 16 (β-unexpected fuzzy recurrence) *Given a sequence s and a belief $b = \mathcal{R} \wedge \mathcal{M}$, where $\mathcal{R}$ is a consistent fuzzy recurrence rule set and $\mathcal{M}$ is a consistent semantic contradiction set on fuzzy recurrence sequences, if s supports $\langle s_\alpha, \zeta_\alpha \rangle$ and there exists a conclusion sequence $\langle s_\beta, \zeta_\beta \rangle \in \Delta(b)$ such that $s_\beta \sqsubseteq s$ and $\langle s_\beta, \zeta_\beta \rangle \not\sqsubseteq s$, then the sequence s is β-unexpected, denoted as $s \not\models_\beta b$.*

The primary factor of the β-unexpectedness in a sequence s is that the recurrence sequence $\langle s_\beta, \zeta_\beta \rangle$ does not occur as expected however at least the sequence s_β occurs in s. Therefore, in comparison with the β-unexpectedness defined in Chapter 4, although the forms of unexpectedness are different, however they have the same semantics.

For instance, considering the belief in Example 23, noted as b, let s be a customer transaction sequence, if we have that $\langle (\texttt{music})(\texttt{movie}), often \rangle \sqsubseteq s$ and $\langle (\texttt{game}), often \rangle \sqsubseteq s$, then s is expected with respect to the fuzzy recurrence rule $\langle (\texttt{music})(\texttt{movie}), often \rangle \rightarrow \langle (game), often \rangle$ (we discuss

the semantic contradiction later); however, if we have $\langle(\texttt{game})\rangle \sqsubseteq s$ but not $\langle(\texttt{game}), often\rangle \sqsubseteq s$, for example, the case $\langle(\texttt{game}), rarely\rangle \sqsubseteq s$, since $\langle(\texttt{game}), rarely\rangle \sqsubseteq s$ implies that $\langle(\texttt{game})\rangle \sqsubseteq s$, then s is a β-unexpected sequence, i.e., $s \nVdash_\beta b$.

Definition 17 (γ-unexpected fuzzy recurrence) *Given a sequence s and a belief $b = \mathcal{R} \wedge \mathcal{M}$, where $\mathcal{R}$ is a consistent fuzzy recurrence rule set and $\mathcal{M}$ is a consistent semantic contradiction set on fuzzy recurrence sequences, if s supports $\langle s_\alpha, \zeta_\alpha\rangle$ and there exists a contradiction sequence $\langle s_\gamma, \zeta_\gamma\rangle \in \Theta(b, \langle s_\beta, \zeta_\beta\rangle)$ where $\langle s_\beta, \zeta_\beta\rangle \in \Delta(b)$, such that $\langle s_\gamma, \zeta_\gamma\rangle \sqsubseteq s$, then the sequence s is γ-unexpected, denoted as $s \nVdash_\gamma b$.*

Respectively, the primary factor of the γ-unexpectedness in a sequence s is that at least one semantic contradiction $\langle s_\beta, \zeta_\beta\rangle \not\approx_{sem} \langle s_\gamma, \zeta_\gamma\rangle$ is broken because the recurrence sequence $\langle s_\gamma, \zeta_\gamma\rangle$ occurs in s. Considering again the belief b in Example 23, let s be a customer transaction sequence, if we have that $\langle(\texttt{music})(\texttt{movie}), often\rangle \sqsubseteq s$ and $\langle(\texttt{book}), often\rangle \not\sqsubseteq s$, then the sequence s is not unexpected with respect to the semantic contradiction $\langle(\texttt{game}), often\rangle \not\approx_{sem} \langle(\texttt{book}), often\rangle$; however, if we have $\langle(\texttt{book}), often\rangle \sqsubseteq s$, then s is a γ-unexpected sequence, i.e., $s \nVdash_\gamma b$. Of course, it is not necessary to forbid $(\texttt{book}) \sqsubseteq s$, for example, according to this belief, the occurrence of $\langle(\texttt{book}), rarely\rangle$ does not imply the γ-unexpectedness.

In Chapter 4 we discussed the coherence in a belief defined in Definition 6. Let $b = \mathcal{R} \wedge \mathcal{M}$ be a belief of sequence rules, we constrain that for any relation $(s_{\beta_i} \not\approx_{sem} s_{\gamma_i}) \in \mathcal{M}$, there does not exist $s_{\beta_j} \in \Delta(R)$ such that $s_{\gamma_i} \sqsubseteq s_{\beta_j}$. The coherence in a belief consists of fuzzy recurrence rules and semantic contradictions on fuzzy recurrence sequences must be considered in sequence inclusions and covers of the fuzzy sets on recurrence.

Given a belief $b = \mathcal{R} \wedge \mathcal{M}$, for any two fuzzy recurrence rules $r, r' \in \mathcal{R}$, let $r = \langle s_\alpha, \zeta_\alpha\rangle \rightarrow \langle s_\beta, \zeta_\beta\rangle$ and $r' = \langle s_\alpha{}', \zeta_\alpha{}'\rangle \rightarrow \langle s_\beta{}', \zeta_\beta{}'\rangle$, where $\langle s_\beta, \zeta_\beta\rangle \not\approx_{sem} \langle s_\gamma, \zeta_\gamma\rangle$ and $\langle s_\beta{}', \zeta_\beta{}'\rangle \not\approx_{sem} \langle s_\gamma{}', \zeta_\gamma{}'\rangle$, the following condition must be satisfied if the belief b is coherent:

$$(s_\beta \not\sqsubseteq s_\gamma{}') \vee (\zeta_\beta \neq \zeta_\gamma{}')$$

For example, let us consider two fuzzy recurrence rules r_1 and r_2. Let $r_1 = \langle(a), often\rangle \rightarrow \langle(c)(d), often\rangle$ and $r_2 = \langle(a), often\rangle \rightarrow \langle(e), often\rangle$ where $\langle(c)(d), often\rangle \not\approx_{sem} \langle(e)(f), often\rangle$ and $\langle(e), often\rangle \not\approx_{sem} \langle(c), often\rangle$. Then r_1 and r_2 are in conflict because $\langle(e)(f), often\rangle$ implies that $\langle(e), often\rangle$.

Given a sequence database $\mathcal{D}$ and a belief base $\mathcal{B}$, the problem of discovering unexpected fuzzy recurrence sequences is therefore to find all sequences $s \in \mathcal{D}$ that contain β-unexpectedness and/or γ-unexpectedness with respect to each belief $b \in \mathcal{B}$ that consist of recurrence rules and semantic contradictions on recurrence sequences.

5.3.3 Approach UFR

In this section we develop the approach UFR (*Unexpected Fuzzy Recurrence*). First, with respect to the framework MUSE, the belief tree construction must take account of the fuzzy sets on the recurrence of sequences in order to address fuzzy recurrence rules, which can be easily handled by adding a field to each s-node in a belief tree. Then, the sequence matching routine `SeqMatch` (including `SeqMatchMax`, `SeqMatchMin`, and `SeqMatchFirst`) must be redesigned in order to find the occurrences of fuzzy recurrence sequences.

The fuzzy recurrence sequence matching routine is therefore the core of the approach UFR, so that we develop the algorithm `SeqMatchUfr` (Algorithm 10), which finds the occurrence of a fuzzy recurrence sequence in a sequence. The algorithm accepts a fuzzy recurrence sequence $\langle s, \zeta \rangle$, a sequence s', and a pair *range* for bounding the occurrence of $\langle s, \zeta \rangle$ in s' as inputs, and outputs the occurrence of $\langle s, \zeta \rangle$ in s', if s' supports $\langle s, \zeta \rangle$ with respect to Equation (5.1).

Algorithm 10: `SeqMatchUfr` ($\langle s, \zeta \rangle$, s', *range*) : Matching fuzzy recurrence sequence.

Input : A fuzzy recurrence sequence $\langle s, \zeta \rangle$, a sequence s', and a pair *range*.
Output : The occurrence of $\langle s, \zeta \rangle$ in s' with respect to *range*.

```
1  μ_ζ := FuzzyMembershipFunction(ζ);
2  pos := pair(0, 0);
3  ran := range;
4  rec := 0;
5  ret := pair(−1, −1);
6  while pos.first ≠ −1 do
7      pos := SeqMatchFirst(s, s′, ran);
8      if pos.first = −1 then
9          break;
10     ran.first := pos.second + 1;
11     rec := rec + 1;
12     if ret.first = −1 then
13         ret.first := pos.first;
14     ret.seconf := pos.second;
15 if μ_ζ(rec) ≥ recu_min then      /* recu_min is globally accessible */
16     return ret;
17 return pair(−1 − 1);
```

Base on the algorithm `SeqMatchUfr`, we develop the β-unexpected fuzzy recurrences as the routine `UfrMatchBeta`, listed in Algorithm 11.

The algorithm accepts a belief T, a sequence s, and a pair *pos* indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in the sequence s as inputs, and outputs all

or the first β-unexpected fuzzy recurrence(s) in s. Notice that the argument *pos* is specified with respect to the form of calling defined in the framework MUSE (Algorithm 6, Section 4.4), which can be extended to handle the recurrence rules with occurrence constraint[3] like $\langle s_\alpha, \zeta_\alpha \rangle \rightarrow^{[1..5]} \langle s_\beta, \zeta_\beta \rangle$.

Algorithm 11: `UfrMatchBeta` (T, s, *pos*) : Matching β-unexpected fuzzy recurrences.

Input : A belief T, a sequence s, and a pair *pos* indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in s.

Output : The set of all β-unexpected fuzzy recurrences in s with respect to T.

```
1  uxps := TupleSet.Create();
2  n_τ := T.firstTauNode();
3  while n_τ ≠ null and n_τ ∉ N do
4      if n_τ.data.min ≠ −1 then
5          continue;   /* recurrence rule is in sequence association rule form */
6      n_sβ := n_τ.firstSubNode();
7      while n_sβ ≠ null do
8          u := SeqMatchFirst(n_sβ.data, s, pair(pos.second + 1, |s| − 1));
9          if u.first ≠ −1 then
10             u := SeqMatchUfr(⟨n_sβ.data, n_sβ.ζ⟩, s, pair(pos.second + 1, |s| − 1));
11             if u.first ≠ −1 then
12                 uxps.add(tuple(s.id, u.first, u.second));
13                 if options | FIRST_UXPS_ONLY then   /* use the conclusion of Lemma 1 */
14                     return uxps;
15     n_τ := T.nextTauNode(n_τ);
16 return uxps;
```

The algorithm first verifies whether the rules are in the form of sequence association rules, that is, $\tau = \emptyset$. Then, for each conclusion sequence $\langle s_\beta, \zeta_\beta \rangle$ contained in the belief of fuzzy recurrence rules, the algorithm verifies whether s_β is contained in s by the subroutine `SeqMatchFirst`. If $s_\beta \sqsubseteq s$, the subroutine `SeqMatchUfr` matches whether $\langle s_\beta, \zeta_\beta \rangle \not\sqsubseteq s$. Thus, finally algorithm returns all β-unexpected fuzzy recurrences $\langle s_\beta, \zeta_\beta \rangle \not\sqsubseteq s$.

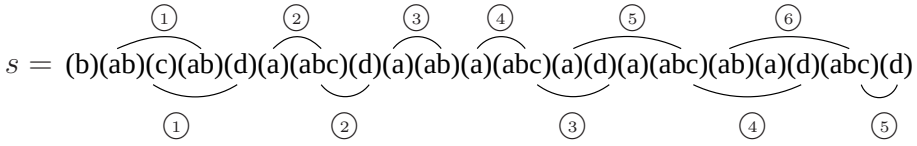


Figure 5.7: Matching β-unexpected fuzzy recurrence.

We illustrate in Figure 5.7 the matching of β-unexpected fuzzy recurrence in a given sequence

[3]We will take account of the recurrence rules with occurrence constraint in our perspectives of future research work.

s with respect to the fuzzy sets shown in Figure 5.6 and the belief

$$\left\{ \langle (a)(ab), often\rangle \rightarrow \langle (c)(d), rarely\rangle \right\} \wedge \left\{ \langle (c)(d), rarely\rangle \not\approx_{sem} \langle (ef)(g), rarely\rangle \right\},$$

where $recu_{min} = 0.6$.

We have that $\langle (a)(ab), often\rangle \sqsubseteq s$ by calling `SeqMatchUfr` before matching β-unexpected fuzzy recurrence (i.e., performed in the main routine of the framework MUSE, where `SeqMatch` is replaced by `SeqMatchUfr`), which is marked as ① to ⑥ above the sequence shown in Figure 5.7 and satisfies the minimum fuzzy membership degree $recu_{min} = 0.6$. Then, $\langle (c)(d), rarely\rangle \sqsubseteq s$ will be verified, where the recurrence of $\langle (c)(d)\rangle$ is marked as ① to ⑤ under the sequence shown in Figure 5.7. According to the fuzzy sets shown in Figure 5.6, we have that $\mu_\zeta(5) = 0.5$ for "rarely", so that we have that $\langle (c)(d), rarely\rangle \not\sqsubseteq s$ and the sequence s is β-unexpected.

With the illustration of matching β-unexpected fuzzy recurrence in a sequence, the matching of γ-unexpected fuzzy recurrences `UfrMatchGamma` is not difficult to understand, which is listed in Algorithm 12.

Algorithm 12: `UfrMatchGamma` (T, s, pos) : Matching γ-unexpected fuzzy recurrences.

Input : A belief T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in s.

Output : The set of all γ-unexpected fuzzy recurrences in s with respect to T.

```
1  uxps := TupleSet.Create();
2  n_τ := T.firstTauNode();
3  while n_τ ≠ null and n_τ ∉ N do
4      if n_τ.data.min ≠ −1 then
5          continue;    /* recurrence rule is in sequence association rule form */
6      n_{s_γ} := n_τ.firstLinkedNode();
7      while n_{s_γ} ≠ null do
8          u := SeqMatchUfr(⟨n_{s_γ}.data, n_{s_γ}.ζ⟩, s, pair(pos.second + 1, |s| − 1));
9          if u.first ≠ −1 then
10             uxps.add(tuple(s.id, u.first, u.second));
11             if options | FIRST_UXPS_ONLY then    /* first occurrence of γ-unexpectedness */
12                 return uxps;
13     n_τ := T.nextTauNode(n_τ);
14 return uxps;
```

The algorithm accepts a belief T, a sequence s, and a pair pos indicating the occurrence of the premise sequence s_α contained in the s_α-node of T in the sequence s as inputs, and outputs all or the first γ-unexpected fuzzy recurrence(s) in s.

5.3.4 Experiments

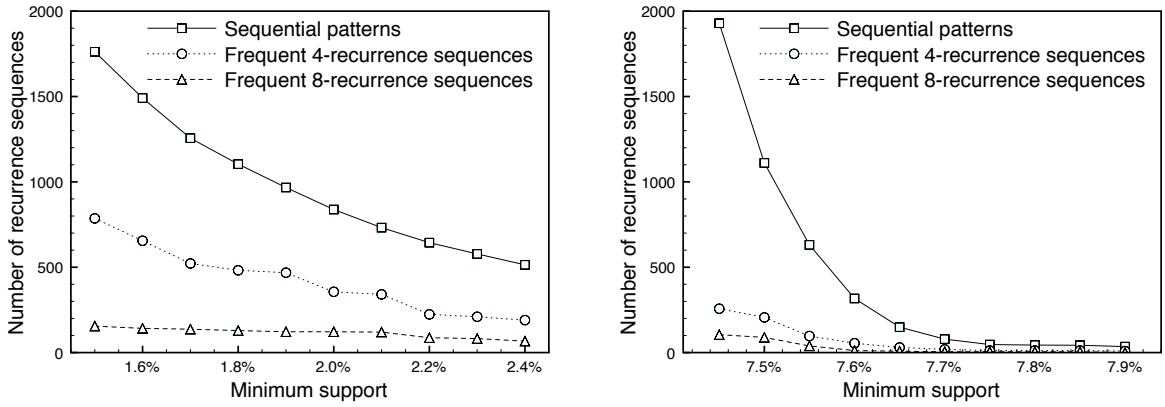
The approach UFR is evaluated with Web access record data. Two types of Web access log are used in our experiments: one is a large access log file of an online forum site (labeled as BBS), and another is a large access log file of a mixed homepage hosting server (labeled as WWW).

Data Set	Size	Distinct Items	Average Length
BBS	135,562	126,383	15.5591
WWW	53,325	85,810	8.3507

Table 5.2: Web access logs used for the evaluation of the approach UFR.

The composition of the two data sets are listed in Table 5.2. We first apply a sequential pattern mining algorithm to discover frequent sequences for studying the general behaviors of the data sets. The frequent 4-recurrence sequences and 8-recurrence sequences are shown in Figure 5.8.

The recurrence sequences in the data sets show that the recurrence behaviors depend on the semantic characteristics of data, for instance, in our experimental data sets, the recurrence behaviors in online forum site are more stronger than those in mixed content Web site.



(a) Data set BBS. (b) Data set WWW.

Figure 5.8: Number of frequent recurrence sequences.

We generate 15 beliefs for each data set after examining the discovered sequential patterns, frequent 4-recurrence and 8-recurrence sequences, which correspond to 3 groups of 5 beliefs: with "rarely", "often" and "frequently", with respect to the fuzzy sets shown in Figure 5.6.

Table 5.3 lists several sample beliefs in our experiments. For instance, the belief

$$\texttt{BBS}_1 = \left\{ \langle(\texttt{f=4}), rarely\rangle \rightarrow \langle(\texttt{f=9}), rarely\rangle \right\} \wedge \left\{ \langle(\texttt{f=9}), rarely\rangle \not\approx_{sem} \langle(\texttt{f=9}), often\rangle \right\}$$

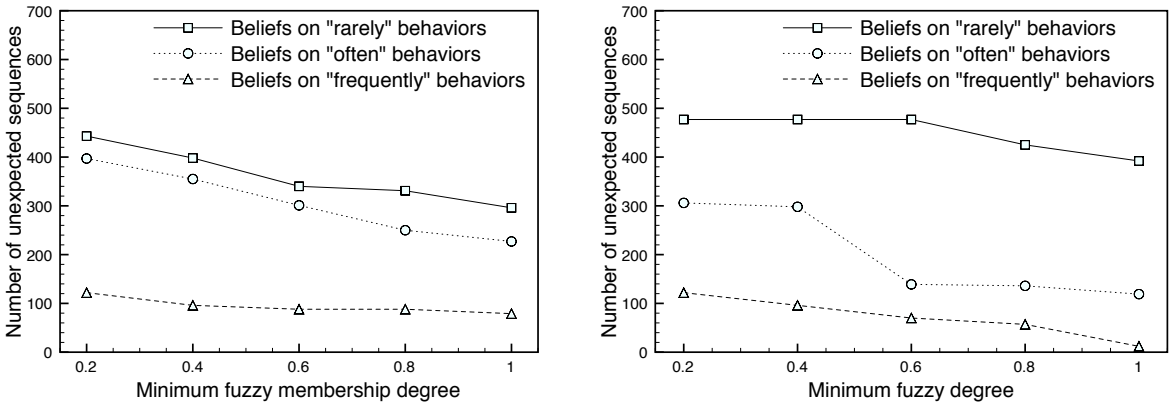
depicts that the forum users who rarely visit the forum No.4 also rarely visit the forum No.9, and that they often visit the forum No.9 is a contradiction; the belief

$$\texttt{WWW}_2 = \left\{ \langle(\texttt{/pub/}), often\rangle \rightarrow \langle(\texttt{/}), rarely\rangle \right\} \wedge \left\{ \langle(\texttt{/}), rarely\rangle \not\approx_{sem} \langle(\texttt{/doc/}), often\rangle \right\}$$

Belief	Premise $\langle s_\alpha, \zeta_\alpha \rangle$	Conclusion $\langle s_\beta, \zeta_\beta \rangle$	Contradiction $\langle s_\gamma, \zeta_\gamma \rangle$
$\texttt{BBS}_1$	$(\texttt{f=4}), rarely$	$(\texttt{f=9}), rarely$	$(\texttt{f=9}), often$
$\texttt{BBS}_2$	$(\texttt{f=0})(\texttt{f=5}), often$	$(\texttt{f=8}), often$	$(\texttt{f=4}), often$
$\texttt{BBS}_3$	$(\texttt{f=5}), frequently$	$(\texttt{f=4}), rarely$	$(\texttt{f=9}), often$
$\texttt{WWW}_1$	$(\texttt{/~li/}), rarely$	$(\texttt{/~li/pub/}), often$	$(\texttt{/~li/pub/}), rarely$
$\texttt{WWW}_2$	$(\texttt{/~li/pub/}), often$	$(\texttt{/~li/}), rarely$	$(\texttt{/~li/doc/}), often$
$\texttt{WWW}_3$	$(\texttt{/~li/}), frequently$	$(\texttt{/~li/doc/}), rarely$	$(\texttt{/~li/doc/}), often$

Table 5.3: Sample beliefs of fuzzy recurrence rules.

(for respecting the thesis layout, we trim the prefix `/~li` of the path) depicts that the homepage visitors who often access the publications located in `/~li/pub/` rarely access the homepage `/~li/`, so that they should not often access the documents located in `/~li/doc/`.



(a) Data set BBS. (b) Data set WWW.

Figure 5.9: Number of sequences with unexpected fuzzy recurrences.

Figure 5.9 shows our experimental results. With the decrease of the minimum fuzzy degree threshold, the number of unexpected sequences increases. In Figure 5.9(a), we find that in the "frequently" fuzzy set, the number of unexpected sequences is much less than those in the other two fuzzy sets, because in the data set the number of long recurrence sequences, such as 8-recurrence sequences, is less. We can also find that the unexpected behaviors focus on the recurrences between "rarely" and "often". In Figure 5.9(b), there is a sharp increase of the number of unexpected sequences in the "often" fuzzy set when the minimum fuzzy membership degree decreases from 0.6 to 0.4, because in the "often" fuzzy set, the fuzzy degree 0.5 corresponds to 4-recurrence sequences, so that a lot of unexpected sequences in the "rarely" fuzzy set are counted as "often".

5.4 Discussion

In this chapter, we first extended the framework MUSE with taking account of the fuzziness in the unexpectedness on sequence occurrence (*tau-fuzzy*) and developed the approach TAUFU. We then proposed the notion of fuzzy recurrence sequence, with which we developed the approach UFR to discover unexpected fuzzy recurrences within the framework MUSE. Experiments on various real Web server access data show the performance of the approaches TAUFU and UFR.

We studied the fuzziness in unexpected sequence occurrence, where the notion of *tau-fuzzy* is based on the gap between premise sequence and conclusion sequences, and the notion of fuzzy recurrence sequence is based on the number of sequence occurrences.

There is a very extended way of considering fuzzy association rules in discovering the unexpectedness in data. It can be a more general model that: from a rule "if X is A, then Y is B", if we consider "A semantically contradicts to C" or "B semantically contradicts to D", then "if X is C, then Y is B" or "if X is A, then Y is D" are unexpected. For instance, if "`age` is `old` → `salary` is `high`" corresponds to prior knowledge, then "`age` is `young` → `salary` is `high`" or "`age` is `old` → `salary` is `low`" can be considered as unexpected.

The same manner can also be extended to gradual rules, that is, if prior knowledge shows that " `age increases` → `salary increases`", then "`age increases` → `salary decreases`" is unexpected, etc.

The fuzzy extensions presented in this chapter improve the flexibility of representing the unexpectedness within the framework MUSE. Our future research work includes the construction and discovery of more general models of unexpected sequences and rules within the framework of fuzzy association rules and fuzzy sequential patterns. On the other hand, in order to improve the flexibility of representing prior knowledge (i.e., the construction of belief system), we study the generalization problem of the framework MUSE in the next chapter.

Chapter 6

Generalizations in Unexpected Sequence Discovery

In the previous chapter, we extended the framework MUSE with fuzzy methods, which improve the interpretability of discovered unexpected sequences. On the other hand, the effectiveness of the framework MUSE, with or without fuzzy extensions, depends on the relevancy of beliefs, where the specification of sequence rules and semantic contradictions with respect to prior knowledge is an essential however complex task. To reduce the complexities in constructing beliefs, we present a generalized approach to discover of unexpected sequences with concept hierarchies.

A part of the work presented in this chapter has been published in the *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems (IJUFKS)*.

6.1 Introduction

The framework MUSE proposed in Chapter 4 discovers unexpected sequences with respect to the beliefs based on prior knowledge, where the effectiveness of MUSE depends on the relevancy of beliefs. However, for constructing beliefs, the specification of sequence rules and semantic contradictions is an essential however complex task.

On the other hand, in real-world database applications, many data have a human-defined taxonomy that is often organized in hierarchies, where the semantics of an item are represented with respect to hierarchical taxonomy of concepts.

Hence, although beliefs can be seriously specified with expertise of application domain, the enumeration of the complete sets of rules and semantic contradiction relations based on items is often a hard work. The following example illustrates this problem.

Example 24 Let us consider the instance addressed in Example 11, where customer transaction records are stored as the items purchased by a customer per transaction. Assume that in each

product category, including `Sci-Fi Novel`, `Action Movie DVD`, `Sci-Fi Movie DVD`, `Rock Music CD`, and `Classical Music CD`, there are 10 different products, that is, 10 distinct items under each end concept with respect to the hierarchical taxonomy shown in Figure 6.1.

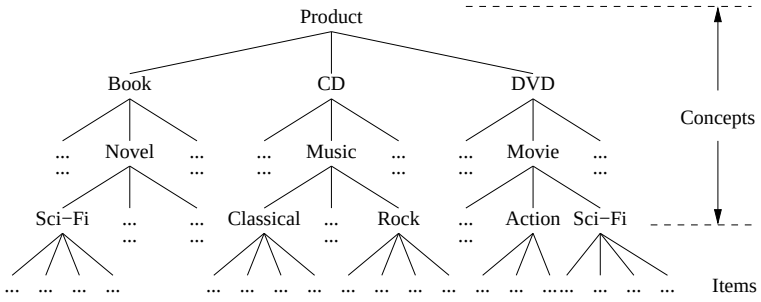


Figure 6.1: Hierarchical taxonomy of products.

Further, we assume that the product relations and customer transaction records are stored in a database like the relations listed in Table 6.1.

Prod.ID	Prod.Category	Prod.Name
...	...	...
12101	Book.Novel.SciFi	...
12102	Book.Novel.SciFi	...
...	...	...
22101	CD.Music.Classical	
22102	CD.Music.Classical	
...	...	...
22301	CD.Music.Rock	...
22302	CD.Music.Rock	...
...	...	...
32201	DVD.Movie.Action	...
32202	DVD.Movie.Action	...
...	...	...
32301	DVD.Movie.SciFi	...
32302	DVD.Movie.SciFi	...
...	...	...

Cust.ID	Trans.ID	Items
...	...	...
C00206	T000586	11105 12108
C00206	T000977	12109
C00206	T001108	32201 32202
C00206	T001210	32205 32307
C00206	T001555	21209
C00206	T001809	22303
C00206	T002112	22507
...	...	...
C01052	T001375	12101
C01052	T001664	22305 32301
C01052	T001792	12108 32308
C01052	T001860	32201 32202 32302
C01052	T002276	31202
C01052	T002279	22101
...	...	...

Table 6.1: Product relations and customer transaction records.

In such a database system, to discover the customer transaction sequences unexpected to the

behaviors described in the belief

$$b_3 = \left\{ \langle(\texttt{Sci-Fi-Novel})(\texttt{Action-Movie Sci-Fi-Movie})\rangle \rightarrow^* \langle(\texttt{Rock-Music})\rangle \right\} \wedge \left\{ \langle(\texttt{Rock-Music})\rangle \not\approx_{sem} \langle(\texttt{Classical-Music})\rangle \right\}$$

of Example 11, each item should be specified according to the `SeqMatch` routine in the approach MUSE, that is, as the form of the following beliefs

$$\cdots\cdots\cdots,$$

$$b_i = \left\{ \langle(12101)(32201\ 32301)\rangle \rightarrow^* \langle(22301)\rangle \right\} \wedge \left\{ \langle(22301)\rangle \not\approx_{sem} \langle(22101)\rangle \right\},$$

$$b_j = \left\{ \langle(12102)(32201\ 32301)\rangle \rightarrow^* \langle(22301)\rangle \right\} \wedge \left\{ \langle(22301)\rangle \not\approx_{sem} \langle(22101)\rangle \right\},$$

$$\cdots\cdots\cdots.$$

Hence, there exist 10^4 sequence rules and 10^2 semantic contradiction relations that cover all possible combinations of items, and it is necessary to totally generate 10^5 beliefs instead of one belief on the generalization of hierarchical taxonomy. □

Indeed, generalizations have been well concentrated in mining association rules [SA95, HF95, HMWG98, TS98, HW02, TL07, KZC08] and sequential patterns [SA96b, TS98, LLW02, dAdSRJ03, MPT04, HY06] during the past decade.

Srikant and Agrawal first studied the generalization problem in association rule mining [SA95], where the taxonomy on items is considered as *is-a* hierarchy. For instance, according to the hierarchy shown in Figure 6.1, we can say that "`Sci-Fi-Novel` *is-a* `Novel` *is-a* `Book`". The proposed approach is therefore to discover the association rules like (`Novel Rock-Music`) → (`Action-Movie`) with considering each concept as an item and pruning itemsets containing an item and its ancestor. This work has been extended to discover generalized sequential patterns in [SA96b], which are maximal frequent sequences like "`Novel` and `Rock-Music` followed by `Action-Movie`, then followed by item `32301`". Many approaches have been developed to improve the efficiency of mining generalized association rules and sequential patterns [HMWG98, HW02, LLW02, dAdSRJ03, MPT04, HY06, TL07, KZC08], which effectively reduce the number of discovered patterns, rules, or sequences in comparison with the results without data generalization.

Therefore, to benefit from high-level knowledge on the taxonomy of data, in this chapter, we propose a generalized approach to discover unexpected sequences with *concept hierarchies* in order to reduce the complexities in belief construction.

The rest of this chapter is organized as follows. We first formalize the definitions of concept hierarchy and generalized sequences in Section 6.2, then propose the notion of generalized beliefs in Section 6.3. In Section 6.4, we discuss the unexpected sequences in hierarchical data with respect

to generalized beliefs, and we further propose a method for determining the semantic contradiction between generalized sequences with respect to concept hierarchies, which proceeds to the discovery of unexpected sequences without specifying semantic contradictions. We show the experiments of discovering unexpected sequences with concept hierarchies in Section 6.6 and Section 6.7 is a discussion.

6.2 Generalized Sequences and Rules

In this section, we first define the notion of concept hierarchies, then we formalize the generalized sequences and generalized sequence rules.

A *concept* is a cognitive unit of knowledge, and a group of semantically related concepts can be represented as a hierarchy, defined as follows.

Definition 18 (Concept hierarchy) *A concept hierarchy* $\mathcal{H} = (\mathcal{C}, \preceq)$ *is a finite set* $\mathcal{C}$ *of concepts and a partial order* $\preceq$ *on* $\mathcal{C}$*.*

In this definition, the partial order $\preceq$ is a *specialization/generalization relation* on the concepts in the set $\mathcal{C}$. For two concepts $c_\phi, c_\theta \in \mathcal{C}$, if $c_\phi \preceq c_\theta$, then we say that the concept c_ϕ is more general than the concept c_θ, and we also say that the concept c_θ is more specific than the concept c_ϕ. We write $c_\phi \prec c_\theta$ if $c_\phi \preceq c_\theta$ and not $c_\theta \preceq c_\phi$. Denote by $level(c_\varphi)$ the *level* of a concept $c_\varphi \in \mathcal{C}$, defined as follows: if for no $c_\phi \in \mathcal{C}$ we have $c_\phi \preceq c_\varphi$, then $level(c_\varphi) = 0$; otherwise $level(c_\varphi) = max(\{level(c_\varphi) \mid c_\varphi \preceq c_\phi\}) + 1$.

Example 25 In Figure 6.1, we have that `Music` $\prec$ `CD`, `Classical` $\prec$ `Music`, and `Classical` $\prec$ `CD`; however `Classical` $\not\preceq$ `Rock` and `Classical` $\not\preceq$ `Sci-Fi`. We also have that $level($`Product`$) = 0$, $level($`Book`$) = level($`CD`$) = level($`DVD`$) = 1$, and so on. □

Given a concept hierarchy $\mathcal{H} = (\mathcal{C}, \preceq)$, denote by $c \in \mathcal{H}$ the concept $c \in \mathcal{C}$. A *generalized pattern* is an unordered collection $C = (c_1 c_2 \ldots c_m)$ of distinct concepts sorted by lexical order, where c_i is a concept and for any $c_i \neq c_j$, $c_i \not\preceq c_j$. A *generalized sequence* is an ordered list $S = \langle C_1 C_2 \ldots C_k \rangle$ of generalized patterns, where C_i is a generalized pattern. Denote $C \in S$ a generalized pattern contained in a generalized sequence S.

The specialization relation $\preceq$ can be applied to generalized patterns and generalized sequences. Given two generalized patterns C and C', if for each concept $c \in C$ there exists a distinct concept $c' \in C'$ such that $c \preceq c'$, then we say that the generalized pattern C is more general than the generalized pattern C' (and C' is more specific than C), denoted as $C \preceq C'$. Given two k-length generalized sequences $S = \langle C_1 C_2 \ldots C_k \rangle$ and $S' = \langle C_1' C_2' \ldots C_k' \rangle$, if for each generalized pattern

C_i and C_i' $(1 \leq i \leq k)$, we have that $C_i \preceq C_i'$, than we say that the generalized sequence S is more general than the generalized sequence S' (and S' is more specific than S), denoted as $S \preceq S'$.

Given a sequence database $\mathcal{D}$ and a concept hierarchy $\mathcal{H}$, each item $i \in \mathcal{D}$ belongs to a concept $c \in \mathcal{H}$, denoted as $i \models c$; if $i \models c_\theta$ and $c_\varphi \preceq c_\theta$, then $i \models c_\varphi$. Let I be an itemset and C be a generalized pattern, if for each $i \in I$ there exist a distinct concept $c \in C$ such that $i \models c$, then we say that the itemset I *supports* the generalized pattern C, denoted as $I \models C$. Let $S = \langle C_1 C_2 \ldots C_m \rangle$ be a generalized sequence on $\mathcal{H}$ and $s = \langle I_1 I_2 \ldots I_n \rangle$ be a sequence in $\mathcal{D}$, if there exist integers $1 \leq i_1 < i_2 < \ldots < i_m \leq n$ such that $I_{i_1} \models C_1, I_{i_2} \models C_2, \ldots, I_{i_m} \models C_m$, then we say that the sequence s *supports* the generalized sequence S, denoted as $s \models S$.

Example 26 Considering Figure 6.1, we have that (`Novel CD`) $\preceq$ (`Sci-Fi-Novel Rock-Music`) however (`Novel DVD`) $\not\preceq$ (`Sci-Fi-Novel Rock-Music`); we also have

$$\langle (\texttt{Book})(\texttt{CD DVD})(\texttt{CD}) \rangle \preceq \langle (\texttt{Sci-Fi-Novel})(\texttt{Rock-Music Action-Movie})(\texttt{Classical-Music}) \rangle .$$

According to Table 6.1, we have that `12101` $\models$ `Sci-Fi-Novel`, (`22301`) $\models$ (`Rock-Music`), and so on. For the customer `C01052`, we have the transaction sequence

$$\langle (\texttt{12101})(\texttt{22305 32301})(\texttt{12108 32308})(\texttt{32201 32202 32302})(\texttt{31202})(\texttt{22101}) \rangle ,$$

which supports the generalized sequence

$$\langle (\texttt{Sci-Fi-Novel})(\texttt{Rock-Music Action-Movie})(\texttt{Classical-Music}) \rangle ,$$

since we have that (`12101`) $\models$ (`Sci-Fi-Novel`), (`22305 32301`) $\models$ (`Rock-Music Action-Movie`), and (`22101`) $\models$ (`Classical-Music`). □

With the notion of generalized sequences, we can further define the notions of generalized sequence rules.

According to the notions of sequence rules proposed in Chapter 3, we generalize the notion of sequence rules considered in this thesis, including sequence association rules and predictive sequence implication rules, with respect to concept hierarchies.

Definition 19 (Generalized sequence association rule) *A generalized sequence association rule is a rule in the form $S_\alpha \rightarrow S_\beta$, where S_α, S_β are two generalized sequences.*

For a generalized sequence association rule $r = S_\alpha \rightarrow S_\beta$, the sequence S_α is called the premise sequence and the sequence S_β is called the conclusion sequence. Given a sequence s, if $s \models S_\alpha$ and $s \models S_\beta$, then we say that the sequence s *supports* the rule $S_\alpha \rightarrow S_\beta$, denoted as $s \models (S_\alpha \rightarrow S_\beta)$.

Definition 20 (Generalized predictive sequence implication rule) *A generalized predictive sequence implication rule is a rule in the form $S_\alpha \rightarrow^\tau S_\beta$, where S_α, S_β are two generalized sequences and $\tau = [min..max]$ is a constraint such that $min, max \in \mathbb{N}$ and $min \leq max$.*

Given a sequence s and a generalized predictive sequence implication rule $r = S_\alpha \rightarrow^\tau S_\beta$, if there exists a sequence s' such that $|s'| \models \tau$ and there exist sequences ${s_\alpha}', {s_\beta}' \sqsubseteq s$ such that ${s_\alpha}' \models S_\alpha$, $|{s_\alpha}'| = |S_\alpha|$, ${s_\beta}' \models S_\beta$, $|{s_\beta}'| = |S_\beta|$, and ${s_\alpha}' \cdot s' \cdot {s_\beta}' \sqsubseteq s$, then we say that the sequence s *supports* the rule $S_\alpha \rightarrow^\tau S_\beta$, denoted as $s \models (S_\alpha \rightarrow^\tau S_\beta)$.

As discussed in Section 3.4, we use the term *generalized sequence rule* for describing the unified form $S_\alpha \rightarrow^\tau S_\beta$ of generalized sequence association rules (where $\tau = \emptyset$) and generalized predictive sequence implication rules (where $\tau \neq \emptyset$).

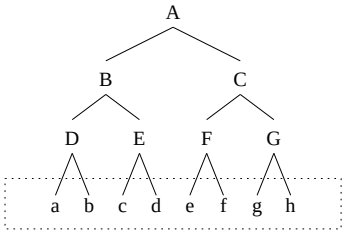


Figure 6.2: A concept hierarchy of items.

Example 27 Figure 6.2 shows a concept hierarchy of concepts and associated items. We have $A \prec B$, $A \prec C$, $B \prec D$, $B \prec E$, $C \prec F$, $C \prec G$, $\{a, b\} \models D$, $\{c, d\} \models E$, $\{e, f\} \models F$, and $\{g, h\} \models G$. With this hierarchy, given a concept occurrence rule $\langle (D) \rangle \rightarrow^* \langle (E)(EF) \rangle$ and a sequence $s = \langle (a)(b)(c)(de) \rangle$, we have $s \models (\langle (D) \rangle \rightarrow^* \langle (E)(EF) \rangle)$ since we have $a \models D$ (or $b \models D$), $c \models E$, and $(de) \models (EF)$. □

6.3 Unexpected Sequences against Generalized Beliefs

The generalization of unexpected sequence discovery is considered in generalizing the belief system introduced in Chapter 4 with respect to hierarchical data.

In this section, we first formalize the generalized belief system, then we propose the unexpected sequences against generalized beliefs.

6.3.1 Generalized Beliefs

With generalized sequence rules and the concept hierarchies, we can therefore generalize the belief system proposed in Section 4.2.

We first discuss the semantic contradiction on generalized sequences with respect to a concept hierarchy of items, which is so called the *generalized semantic contradiction*.

Let us consider a concept hierarchy $\mathcal{H} = (\mathcal{C}, \preceq)$. Given two concepts $c_\phi, c_\theta \in \mathcal{C}$, we define that for $c_\phi \not\approx_{sem} c_\theta$, if an item $i_\phi \models c_\theta$ and $i_\phi \not\models c_\theta$, then $i_\phi \not\approx_{sem} c_\theta$. We also define that for two generalized patterns $C_\phi \not\approx_{sem} C_\theta$, if an itemset $\mathcal{I}_\phi \models C_\phi$ and $\mathcal{I}_\phi \not\models C_\theta$, then $\mathcal{I}_\phi \not\approx_{sem} C_\theta$. In the same manner, we define that for two generalized sequences $S_\phi \not\approx_{sem} S_\theta$, if a sequence $s_\phi \models S_\phi$ and $s_\phi \not\models S_\theta$, then $s_\phi \not\approx_{sem} S_\theta$.

Given a concept hierarchy $\mathcal{H} = (\mathcal{C}, \preceq)$, the relation $\preceq$ is monotone to the semantic contradiction relations: for any $c_\phi, c_\theta, c_\varphi \in \mathcal{C}$, if $c_\phi \not\approx_{sem} c_\theta$ and $c_\phi \preceq c_\varphi$, then $c_\varphi \not\approx_{sem} c_\theta$; given three generalized patterns C_ϕ, C_θ, and C_φ on $\mathcal{H}$, where $|C_\phi| = |C_\theta| = |C_\varphi|$, if $C_\phi \not\approx_{sem} C_\theta$ and $C_\phi \preceq C_\varphi$, then $C_\varphi \not\approx_{sem} C_\theta$; given three generalized sequences S_ϕ, S_θ, and S_φ on $\mathcal{H}$, the semantic contradiction relation on generalized sequences determines that, if $S_\phi \not\approx_{sem} S_\theta$ and $S_\phi \preceq S_\varphi$, then $S_\varphi \not\approx_{sem} S_\theta$.

Example 28 Consider the hierarchy shown in Figure 6.2. If we assume that $B \not\approx_{sem} C$, then we have that $E \not\approx_{sem} F$ and $d \not\approx_{sem} C$; if we assume that $E \not\approx_{sem} C$, then we also have that $E \not\approx_{sem} F$ and $d \not\approx_{sem} F$, but we do not have that $B \not\approx_{sem} C$. If we assume $\langle (D)(F) \rangle \not\approx_{sem} \langle (EG) \rangle$, then we have that $\langle (a)(e) \rangle \not\approx_{sem} \langle (EG) \rangle$, $\langle (b)(e) \rangle \not\approx_{sem} \langle (EG) \rangle$, $\langle (a)(f) \rangle \not\approx_{sem} \langle (EG) \rangle$, and $\langle (b)(f) \rangle \not\approx_{sem} \langle (EG) \rangle$. □

A *generalized sequence belief* is a belief consisting of generalized sequences and generalized semantic contradictions with respect to a concept hierarchy, which is formally defined as follows.

Definition 21 (Generalized sequence belief) *A generalized sequence belief is a conjunction $\mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$, where $\mathcal{R}$ is a non-empty consistent generalized sequence rule set and $\mathcal{M}$ is a consistent generalized semantic contradiction set on the concept hierarchy $\mathcal{H}$, such that for each relation $(S_{\beta_i} \not\approx_{sem} S_{\gamma_i}) \in \mathcal{M}$, we have that $S_{\beta_i} \in \Delta(\mathcal{R})$, and for any relation $(S_{\beta_i} \not\approx_{sem} S_{\gamma_i}) \in \mathcal{M}$, there does not exist $S_{\beta_j} \in \Delta(R)$ such that $S_{\gamma_i} \sqsubseteq S_{\beta_j}$.*

Given a generalized belief B, if a sequence s supports at least one rule contained in this belief and no semantic contradiction of any other rules can be found in the sequence s, then we say that the sequence s satisfies the belief B or the sequence s *supports* the belief B, denoted as $s \models B$. We discuss the satisfaction of a generalized belief B in following cases.

1. Let $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ be a generalized belief that consists of a consistent generalized sequence association rule set $\mathcal{R}$ and a consistent semantic contradiction set $\mathcal{M}$ on the concept hierarchy $\mathcal{H}$. If there exists a generalized rule $(r = S_\alpha \rightarrow S_\beta) \in \mathcal{R}$ such that $s \models r$, and for any generalized semantic contradiction $(S_{\beta_i} \not\approx_{sem} S_{\gamma_j}) \in \mathcal{M}$ there does not exist a rule $(r' = S_\alpha \rightarrow S_{\beta_i}) \in \mathcal{R}$ such that $s \models r'$, then we have that sequence $s \models B$.

2. Let $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ be a generalized belief that consists of a consistent generalized predictive sequence implication rule set $\mathcal{R}$ and a consistent semantic contradiction set $\mathcal{M}$ on the concept hierarchy $\mathcal{H}$. If there exists a rule $(r = S_\alpha \rightarrow^\tau S_\beta) \in \mathcal{R}$ such that $s \models r$, and for any semantic contradiction $(S_{\beta_i} \not\approx_{sem} S_{\gamma_j}) \in \mathcal{M}$ there does not exist a rule $(r' = S_\alpha \rightarrow^{\tau'} S_{\beta_i}) \in \mathcal{R}$ such that $s \models r'$, then we have that sequence $s \models B$.

6.3.2 Generalized Unexpected Sequences

In Section 4.3, we proposed three forms of unexpected sequences stated by sequence beliefs of different form of sequence rules. In this section, we respectively propose the three forms of *generalized unexpected sequences* with respect to generalized beliefs.

The α-unexpected (completeness-unexpected) sequences can be determined by simple sequence implication rules $s_\alpha \rightarrow^* s_\beta$, where s_α and s_β are two sequences. We now propose the notion of *generalized α-unexpected sequences* determined by generalized simple sequence implication rules $S_\alpha \rightarrow^* S_\beta$, where s_α, s_β are two generalized sequences.

Definition 22 (Generalized α-unexpected sequence) *Given a sequence s and a generalized belief $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ where $\mathcal{R}$ and $\mathcal{M}$ are consistent sets of simple generalized sequence implication rules and semantic contradictions on the concept hierarchy $\mathcal{H}$, if $s \models \Lambda(\mathcal{R})$ and for each rule $r \in \mathcal{R}$ we have that $s \not\models r$, then the sequence s is a* generalized α-unexpected sequence *with respect to the belief B, denoted as $s \nVdash_\alpha B$. We also call such an unexpected sequence a* generalized completeness-unexpected *sequence.*

A belief $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ of generalized simple sequence implication rules states that at least one sequence in the conclusion sequence set $\Delta(B)$ of the belief B should occur after the occurrence of the premise sequence $\Lambda(\mathcal{R})$ in an expected sequence. Hence, given a rule $(S_\alpha \rightarrow^* S_\beta) \in \mathcal{R}$ and a sequence s, the occurrence constraint $\tau = [0..*]$ is broken if and only if $s \models S_\alpha$ and $s \not\models S_\alpha \cdot S_\beta$.

The β-unexpected (occurrence-unexpected) sequences can be determined by predictive sequence implication rules $s_\alpha \rightarrow^\tau s_\beta$, where s_α, s_β are two sequences and $\tau \neq *$. Respectively, we can define the form of *generalized β-unexpected sequences* from generalized predictive sequence implication rules $S_\alpha \rightarrow^\tau S_\beta$, where s_α, s_β are two generalized sequences.

Definition 23 (Generalized β-unexpected sequence) *Given a sequence s and a generalized belief $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ where $\mathcal{R}$ and $\mathcal{M}$ are consistent sets of simple generalized sequence implication rules and semantic contradictions on the concept hierarchy $\mathcal{H}$, if $s \models \Lambda(\mathcal{R})$ and for each rule $r \in \mathcal{R}$ we have that $s \not\models r$, then the sequence s is a* generalized β-unexpected sequence *with respect to the belief B, denoted as $s \nVdash_\beta B$. We also call such an unexpected sequence a* generalized occurrence-unexpected *sequence.*

A belief $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ with generalized predictive sequence implication rules states that at least one sequence in the conclusion sequence set $\Delta(B)$ of the belief B should occur after the occurrence of the premise sequence $\Lambda(\mathcal{R})$ in an expected sequence, with respect to the occurrence constraint τ associated with the rule. For instance, given a rule $(S_\alpha \rightarrow^\tau S_\beta) \in \mathcal{R}$ and a sequence s, the occurrence constraint $\tau = [min..max]$ is broken if and only if $s \models S_\alpha$ and there does not exist sequences ${s_\alpha}', {s_\beta}', s' \sqsubseteq s$ such that ${s_\alpha}' \models S_\alpha$, $|{s_\alpha}'| = |S_\alpha|$, ${s_\beta}' \models S_\beta$, $|{s_\beta}'| = |S_\beta|$, and $|s'| \models \tau$.

The γ-unexpected (semantics-unexpected) sequences can be determined by any sequence rules $s_\alpha \rightarrow^\tau s_\beta$ defined in this thesis, where s_α, s_β are two sequences. So that we can define the form of *generalized γ-unexpected sequences* from generalized sequence rules $S_\alpha \rightarrow^\tau S_\beta$, where s_α, s_β are two generalized sequences.

Definition 24 (Generalized γ-unexpected sequence) *Given a sequence s and a generalized belief $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ where $\mathcal{R}$ and $\mathcal{M}$ are consistent sets of simple generalized sequence implication rules and semantic contradictions on the concept hierarchy $\mathcal{H}$, if $s \models \Lambda(\mathcal{R})$ and there exists a rule $r \in \mathcal{R}$ and a semantic contradiction relation $(S_{\beta_i} \not\approx_{sem} S_{\gamma_j}) \in \mathcal{M}$ such that:*

1. *$s \models (S_\alpha \rightarrow S_{\gamma_j})$, if r is a generalized sequence association rule $S_\alpha \rightarrow S_{\beta_i}$;*
2. *$s \models (S_\alpha \rightarrow^{\tau_i} S_{\gamma_j})$, if r is a generalized sequence implication rule $S_\alpha \rightarrow^{\tau_i} S_{\beta_i}$,*

then the sequence s is a generalized γ-unexpected sequence *with respect to the belief B, denoted as $s \nVdash_\gamma B$. We also call such an unexpected sequence a* generalized semantics-unexpected *sequence.*

A belief $B = \mathcal{R} \wedge \mathcal{M} \wedge \mathcal{H}$ with generalized sequence rules and a non-empty generalized semantic contradiction set $\mathcal{M}$ states that the semantic contradictions of the generalized sequence rules contained in $\mathcal{R}$ should not occur with the premise sequence $\lambda(\mathcal{R})$ with respect to the sequence rule form.

Given a sequence s, we examine generalized γ-unexpectedness with two cases: (2) for a generalized sequence association rule $(S_\alpha \rightarrow S_\beta) \in \mathcal{R}$ on $\mathcal{H}$, if there exists $(S_\beta \not\approx_{sem} S_\gamma) \in \mathcal{M}$ such that $s \models S_\alpha$ and $s \models S_\gamma$, then the rule is broken; (2) for a generalized sequence implication rule $(S_\alpha \rightarrow^\tau S_\beta) \in \mathcal{R}$ on $\mathcal{H}$, if there exists $(S_\beta \not\approx_{sem} S_\gamma) \in \mathcal{M}$ and exist sequences ${s_\alpha}', {s_\gamma}', s' \sqsubseteq s$ such that ${s_\alpha}' \models S_\alpha$, $|{s_\alpha}'| = |S_\alpha|$, ${s_\gamma}' \models S_\gamma$, $|{s_\gamma}'| = |S_\gamma|$, and $|s'| \models \tau$, then the rule is broken. In any case that there exists a rule $r \in \mathcal{R}$ broken, then the sequence s is γ-unexpected to the generalized belief B.

6.4 Soft Unexpected Sequences in Hierarchical Data

In this section, we propose an approach to discover *soft unexpected sequences* with respect to generalized rules in hierarchical data without specifying semantic contradictions. We first discuss

the computation of semantic relatedness and contradiction between generalized sequences, and then proposed the notions of soft unexpected sequences.

6.4.1 Semantic Relatedness and Contradiction

Before being able to formally define the notions of soft unexpected sequences, in this section, we first propose the computation of semantic relatedness and contradiction between the generalized sequences on a given concept hierarchy of data taxonomy.

Let us consider again the instance of Web usage analysis, where a generalized occurrence rule can be defined as $\langle(/)\rangle \rightarrow^{[0..5]} \langle(\texttt{Politics})\rangle$ with respect to the concept hierarchy shown in Figure 6.3. For example, to build a belief with "`technology news` semantically contradicts `politics news`", the semantic contradiction $\langle(\texttt{Politics})\rangle \not\approx_{sem} \langle(\texttt{Technology})\rangle$ is necessary. However, depending on user experiences and the taxonomy shown in Figure 6.3, not only the `technology news` contradicts `politics news`.

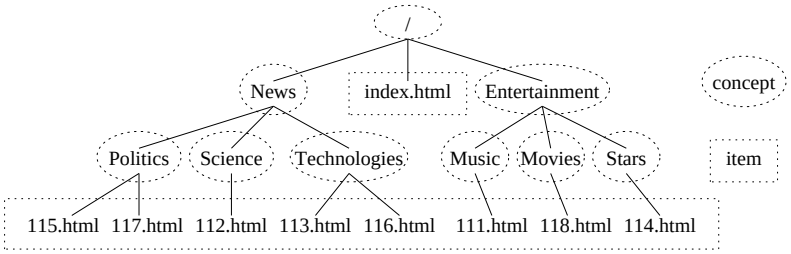


Figure 6.3: A concept hierarchy of Web site structure.

The semantic contradiction of two concepts in a hierarchy is determined by the distance and semantic similarity between the concepts.

Given a concept hierarchy $\mathcal{H}$ and two concepts $c_i, c_j \in \mathcal{H}$, the *semantic distance* between the concepts c_i and c_j in the hierarchy $\mathcal{H}$ is denoted as $\delta(c_i, c_j, \mathcal{H})$; the *semantic similarity* is defined as a score $\lambda(c_i, c_j)$, where $0 \leq \lambda(c_i, c_j) \leq 1$. For two concepts, we have that the more distance the less importance for relatedness, and the less similarity the more contradiction.

Therefore, we propose a simple formula for handling the semantic contradiction degree between concepts, denoted as $\omega_{sem}(c_i, c_j, \mathcal{H})$, as following:

$$\omega_{sem}(c_i, c_j, \mathcal{H}) = \frac{2 - \lambda(c_i, c_j)}{\delta(c_i, c_j, \mathcal{H})}, \tag{6.1}$$

where the semantic distance between the concepts c_i and c_j is defined as the *path-length* (i.e., the number of edges) between the nodes c_i and c_j in the hierarchy $\mathcal{H}$, and if $c_i = c_j$, we define $\delta(c_i, c_j, \mathcal{H}) = 1$.

In Equation (6.1), we have that $0 \leq \lambda(c_i, c_j) \leq 1$ if the semantic similarity between c_i and c_j is defined; otherwise, if the semantic similarity is not defined, we define $\lambda(c_i, c_j) = 1$, so that

$\omega_{sem}(c_i, c_j, \mathcal{H})$ is the reciprocal value of the length-path between c_i and c_j in the hierarchy $\mathcal{H}$. In the case that $c_i = c_j$, we define $\lambda(c_i, c_j) = 2$, so that $\omega_{sem}(c_i, c_j, \mathcal{H}) = 0$.

Notice that we consider the semantic contradiction degree $\omega_{sem}(c_i, c_j, \mathcal{H})$ as a value $0 \leq \omega_{sem} < 1$, that excludes the case that $\delta(c_i, c_j, \mathcal{H}) = 1$ when $\lambda(c_i, c_j)$ is undefined.

	Politics	Science	Technology	Music	Movie	Stars
Politics	1:2	2:0.6857	2:0.7183	4:0.4270	4:0.3388	4:0.2996
Science	2:0.6857	1:2	2:0.6929	4:1	4:1	4:1
Technology	2:0.7183	2:0.9	1:2	4:1	4:1	4:1
Music	4:0.4270	4:1	4:1	1:2	2:0.5159	2:0.4274
Movie	4:0.3388	4:1	4:1	2:0.5159	1:2	2:0.3392
Stars	4:0.2996	4:1	4:1	2:0.4274	2:0.3392	1:2

Table 6.2: Path-length and similarity matrix.

$c_i : c_j$	$\delta(c_i, c_j, \mathcal{H})$	$\lambda(c_i, c_j)$	$\omega_{sem}(c_i, c_j, \mathcal{H})$
Politics : Politics	1	2	0
Politics : Science	2	0.6857	0.65715
Politics : Technology	2	0.7183	0.64085
Politics : Music	4	0.4270	0.39325
Politics : Movies	4	0.3388	0.4153
Politics : Stars	4	0.2996	0.4251
Politics : /	2	1	0.5
Politics : News	1	1	1^*

Table 6.3: Semantic contradiction degrees between concepts.

The semantic similarity between concepts can be determined by various approaches [Res95, NMW97, LCN03, RE03, PS08]. Example 29 shows the computation of semantic contradiction degrees.

Example 29 With the hierarchy shown in Figure 6.3, we have the relations listed in Table 6.2, where the semantic similarity between concepts is determined by the JWSL library [PS08] (in order to compare the the different values, assume that the similarities between concepts `Science`, `Technology` and `Music`, `Movie`, `Stars` are not defined). For instance, the path-length between concepts `Politics` and `Technology` is 2; between `Politics` and `Music` is 4. With the JWSL library we have that the similarity between the concepts `Politics` and `Technology` is 0.7183; between `Politics` and `Music` is 0.4270. Thus, according to Equation (6.1), the semantic contradiction

degrees between `Politics` and other concepts are listed in Table 6.3, where ω_{sem} between `Politics` and `News` is excluded. □

Given a sequence s, a generalized sequence S, and a concept hierarchy $\mathcal{H}$, where for each concept c contained in S, we have that $c \in \mathcal{H}$. We determine the semantic contradiction degree between s and S on $\mathcal{H}$ in the following manner.

We first consider the *compatible-form constraint* on a generalized sequence of concepts and a sequence of items, defined as follows.

Definition 25 (Compatible-form constraint) *Given a generalized sequence S and a sequence s, let $S = \langle C_1 C_2 \dots C_m \rangle$ and $s = \langle I_1 I_2 \dots I_n \rangle$. The compatible-form is a constraint that there exist integers $1 \leq i_1 < i_2 < \dots < i_m \leq n$ such that $|C_1| \leq |I_{i_1}|, |C_2| \leq |I_{i_2}|, \dots, |C_m| \leq |I_{i_m}|$, denoted as $S \lhd s$, and denote by $S \trianglelefteq s$ the case $|C_1| = |I_{i_1}|, |C_2| = |I_{i_2}|, \dots, |C_m| = |I_{i_m}|$.*

In order to determine the semantic contradiction between S and s, we require that $S \lhd s$.

Now we consider the semantic contradiction between a generalized pattern C and an itemset I (where $|C| \leq |I|$) on a hierarchy $\mathcal{H}$, denoted as $\omega_{pat}(C, I, \mathcal{H})$ and defined as follows.

Let $\Omega(c_i, i_j, \mathcal{H}) = max\{\omega_{sem}(c_i, c_j, \mathcal{H}) \mid c_j \in \mathcal{H}, i_j \models c_j\}$ be the maximal semantic contradiction degree between a concept $c_i \in \mathcal{H}$ and an item $i_j \in I$, then the number of the combinations of $\Omega(c_i, i_j, \mathcal{H})$ on the elements in $c_i \in C$ and $i_j \in I$ is the number of permutations of $|C|$ items in I, that is,

$$P(|I|, |C|) = \frac{|I|!}{(|I| - |C|)!}. \tag{6.2}$$

Let $\mathcal{I}$ be the set of such permutations, we denote the semantic contradiction degree between a generalized pattern C and an itemset I as:

$$\omega_{pat}(C, I, \mathcal{H}) = \frac{max\{\sum_{c_i \in C} \Omega(c_i, i_j, \mathcal{H}) \mid i_j \in I', I' \in \mathcal{I}\}}{|C|}. \tag{6.3}$$

Therefore, given a generalized sequence S and a sequence s, for all subsequences $s' \sqsubseteq s$ such that $S \trianglelefteq s'$, the semantic contradiction degree between S and s, denoted as $\omega_{seq}(S, s, \mathcal{H})$, is defined as the average of the sum of $\omega_{pat}(C_i, I_i, \mathcal{H})$ that is maximal, where C_i and I_i are itemsets contained in S and s', that is,

$$\omega_{seq}(S, s, \mathcal{H}) = \frac{max\{\sum_{1 \leq i \leq \|S\|} \omega_{pat}(C_i \in S, I_i \in s', \mathcal{H}) \mid s' \sqsubseteq s, S \trianglelefteq s'\}}{\|S\|}. \tag{6.4}$$

Respectively, we define the *semantic relatedness degree* between concepts, denote by $\eta_{sem}(c_i, c_j, \mathcal{H})$, as following:

$$\psi_{sem}(c_i, c_j, \mathcal{H}) = \frac{\lambda(c_i, c_j)}{\delta(c_i, c_j, \mathcal{H})}, \tag{6.5}$$

and let $\Psi(c_i, i_j, \mathcal{H}) = max\{\psi_{sem}(c_i, c_j, \mathcal{H}) \mid c_j \in \mathcal{H}, i_j \models c_j\}$, in the same manner with the permutation set $\mathcal{I}$ of a given itemset I with respect to a generalized pattern C, we define the semantic relatedness degree between C and I as

$$\psi_{pat}(C, I, \mathcal{H}) = \frac{max\{\sum_{c_i \in C} \Psi(c_i, i_j, \mathcal{H}) \mid i_j \in I', I' \in \mathcal{I}\}}{|C|}. \tag{6.6}$$

Given a generalized sequence S and a sequence s, for all subsequences $s' \sqsubseteq s$ such that $S \trianglelefteq s'$, the semantic relatedness degree between S and s, denoted as $\psi_{seq}(S, s, \mathcal{H})$, is defined as the average of the sum of $\psi_{pat}(C_i, I_i, \mathcal{H})$ that is maximal, where C_i and I_i are itemsets contained in S and s', that is,

$$\psi_{seq}(S, s, \mathcal{H}) = \frac{max\{\sum_{1 \le i \le \|S\|} \psi_{pat}(C_i \in S, I_i \in s', \mathcal{H}) \mid s' \sqsubseteq s, S \trianglelefteq s'\}}{\|S\|}. \tag{6.7}$$

6.4.2 Soft Unexpected Sequences

With the notions of semantic relatedness and contradiction degrees, we formally define the soft unexpectedness of sequences with respect to generalized sequence rules on a concept hierarchy as follows.

Definition 26 (Soft α-unexpected sequence) *Given a sequence s, a generalized sequence rule $r = S_\alpha \rightarrow^* S_\beta$ on a concept hierarchy $\mathcal{H}$, a user defined minimum semantic contradiction degree ω_{min}, and a user defined minimal semantic relatedness degree ψ_{min}, if there exists $s_\alpha \sqsubseteq s$ such that $s_\alpha \models S_\alpha$, and there does not exist $s_\beta \sqsubseteq s$ such that $s_\alpha \cdot s_\beta \sqsubseteq_c s$ and $\psi_{seq}(S_\beta, s_\beta, \mathcal{H}) \ge \psi_{min}$, then s is a soft completeness-unexpected sequence, denoted as $s \nvDash^{\sim}_{\alpha} B$. We also call such an unexpected sequence a soft α-unexpected sequence.*

Definition 27 (Soft β-unexpected sequence) *Given a sequence s, a generalized sequence rule $r = S_\alpha \rightarrow^\tau S_\beta$ ($\tau \neq *$) on a concept hierarchy $\mathcal{H}$, a user defined minimum semantic contradiction degree ω_{min}, and a user defined minimal semantic relatedness degree ψ_{min}, if there exists $s_\alpha \sqsubseteq s$ such that $s_\alpha \models S_\alpha$, and there exist $s', s_\beta, s_\gamma \sqsubseteq s$ such that $|s'| \not\models \tau$, $s_\alpha \cdot s' \cdot s_\beta \sqsubseteq_c s$, and $\psi_{seq}(S_\beta, s_\beta, \mathcal{H}) \ge \psi_{min}$, then s is a soft occurrence-unexpected sequence, denoted as $s \nvDash^{\sim}_{\beta} B$. We also call such an unexpected sequence a soft β-unexpected sequence.*

Definition 28 (Soft γ-unexpected sequence) *Given a sequence s, a generalized sequence rule $r = S_\alpha \rightarrow^\tau S_\beta$ ($\tau \neq *$) on a concept hierarchy $\mathcal{H}$, a user defined minimum semantic contradiction degree ω_{min}, and a user defined minimal semantic relatedness degree ψ_{min}, if there exists $s_\alpha \sqsubseteq s$ such that $s_\alpha \models S_\alpha$:*

1. *if $\tau = \emptyset$ and there exist $s', s_\gamma \sqsubseteq s$ such that $|s'| \models \tau$, $s_\alpha \cdot s' \cdot s_\gamma \sqsubseteq_c s$, and $\omega_{seq}(S_\gamma, s_\gamma, \mathcal{H}) \ge \omega_{min}$; or*

2. if $\tau \neq \emptyset$ *and there exist* $s_\gamma \sqsubseteq s \setminus^* s_\alpha$ *such that* $\omega_{seq}(S_\gamma, s_\gamma, \mathcal{H}) \geq \omega_{min}$,

then s *is a soft semantics-unexpected sequence, denoted as* $s \nvDash^{\sim}_{\gamma} B$. *We also call such an unexpected sequence a soft* γ*-unexpected sequence.*

The soft unexpectedness on semantic relatedness and contradiction can also be described by fuzzy sets, like "*weak relatedness/contradiction*", "*medium relatedness/contradiction*", or "*strong relatedness/contradiction*" with respect to β-unexpected or γ-unexpected sequences, by fuzzy membership functions $\mu_{sem}(\psi_{seq}, \mathcal{F})$ or $\mu_{sem}(\omega_{seq}, \mathcal{F})$, where $\mathcal{F}$ is a set of fuzzy partitions.

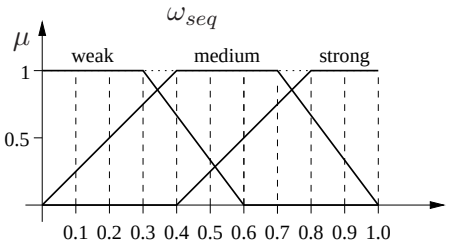


Figure 6.4: Fuzzy sets for semantic contradiction degree.

Example 30 For instance, given a generalized sequence rule $\langle(/)\rangle \rightarrow^* \langle(\texttt{Politics})(\texttt{Movies})\rangle$ on the hierarchy shown in Figure 6.4, the sequence $\langle(\texttt{index})(\texttt{117})(\texttt{118})\rangle$ (we ignore file extensions) is an expected sequence. Given a minimum semantic contradiction degree 0.3, according to Table 6.2 proposed in the previous section, the sequence $\langle(\texttt{index})(\texttt{112})(\texttt{113})\rangle$ is fuzzy γ-unexpected with "medium contradiction" since we have that

$$\omega_{seq}(\langle(\texttt{Politics})(\texttt{Movies})\rangle, \langle(\texttt{112})(\texttt{113})\rangle, \mathcal{H}) = 0.456;$$

the sequence $\langle(\texttt{index})(\texttt{114})(\texttt{113})\rangle$ is fuzzy γ-unexpected with "weak contradiction" since we have that

$$\omega_{seq}(\langle(\texttt{Politics})(\texttt{Movies})\rangle, \langle(\texttt{114})(\texttt{113})\rangle, \mathcal{H}) = 0.338.$$

□

6.5 Approach SoftMuse

In this section, we develop the approach SoftMuse to discover soft unexpected sequences in a sequence database. We first present the main framework of SoftMuse because we cannot directly use the framework Muse for discovering soft unexpected sequences, since the belief system is no longer required. We then present the semantic relatedness/contradiction computation routine `HyMatchSeq` (Hierarchy Matching of Sequences).

To simplify the descriptions, in the algorithms we consider a sequence as an object with the properties ψ_{seq}, ω_{seq}, μ_{sem}, μ_{τ}, etc., which correspond to the notions presented in previous sections. We also assume that the minimum semantic relatedness degree ψ_{min} and the minimum semantic contradiction degree ω_{min} addressed in the definitions of soft unexpected sequences are globally accessible in the algorithms.

The main algorithm of the framework SOFTMUSE is listed in Algorithm 13, which extracts soft unexpected sequences in a sequence database $\mathcal{D}$, with respect to a set $\mathcal{R}$ of generalized sequence rules on a concept hierarchy $\mathcal{H}$, where the minimum semantic relatedness degree ψ_{min} and the minimum semantic contradiction degree ω_{min} are considered as predefined.

Algorithm 13: SOFTMUSE: Soft Multiple Unexpected Sequence Extraction.

Input : A set of generalized sequence rules $\mathcal{R}$ on a concept hierarchy $\mathcal{H}$, and a sequence database $\mathcal{D}$.
Output : All soft unexpected sequences.

```
1  foreach s ∈ D do
2      foreach r ∈ R do
3          pos := HyMatchSeq(r.S_α, s, null, 1, 0);   /* τ = null, ψ_min = 1, ω_min = 0 */
4          if pos.first ≠ −1 then   /* s ⊨ S_α */
5              if r.τ ≠ ∅ then
6                  if r.τ = ∗ then
7                      uxp := HyMatchAlpha(r.S_β, s, pos);
8                      if uxp.first ≠ −1 then
9                          output tuple(r.id, ALPHA, s, uxp);
10                 else
11                     uxp := HyMatchBeta(r.S_β, s, pos, r.τ);
12                     if uxp ≠ ∅ then
13                         output tuple(r.id, BETA, s, uxp);
14             uxp := HyMatchGamma(r.S_γ, s, pos, r.τ);
15             if uxp ≠ ∅ then
16                 output tuple(r.id, GAMMA, s, uxp);
```

For each sequence $s \in \mathcal{D}$ and each generalized sequence rule $(r = S_{\alpha} \rightarrow^{\tau} S_{\beta}) \in \mathcal{R}$, the framework first verifies whether $s \models S_{\alpha}$ by calling the routine `HyMatchSeq` with setting $\psi_{min} = 1$ and $\omega_{min} = 0$. If $s \models S_{\alpha}$, then the framework continues to find soft α-, β-, and γ-unexpectedness in s with respect to the occurrence position of S_{α} and the τ.

The routines `HyMatchAlpha`, `HyMatchBeta`, and `HyMatchGamma` match soft α-, β-, and γ-unexpected sequences with respect to the definitions proposed in Section 6.4.2, hence, we focus on the semantic relatedness/contradiction computation routine `HyMatchSeq` (Algorithm 14).

Given generalized sequence S, a sequence s, the Algorithm `HyMatchSeq` finds the first highest-scored subsequence $s' \sqsubseteq s$ such that $s' \models S$ with respect to an occurrence constraint τ, a minimum semantic relatedness degree ψ_{min} and/or a minimum semantic contradiction degree ω_{min}.

Algorithm 14: `HyMatchSeq` (S, s, τ, ψ_{min}, ω_{min}) : Hierarchy Matching of Sequences.

Input : A generalized sequence S, a sequence s, a occurrence constraint τ, a minimum semantic relatedness degree ψ_{min}, and a minimum semantic contradiction degree ω_{min}.

Output : First highest-scored subsequence $s' \sqsubseteq s$ such that $s' \models S$.

$s' := empty\_sequence$;
$s'.\psi_{seq} := -1$;
$s'.\omega_{seq} := -1$;
if not $S \lhd s$ **then**
    return s';
$\mathcal{X} := \emptyset$;
$\mathcal{S} := seqsat(S, s, \lhd)$;
$\mathcal{S} := \mathcal{S} \setminus \{s'' \mid \mu_\tau(|s''| - |S|, \tau, \mathcal{F}) < \mu_{\tau_{min}}, s'' \in \mathcal{S}\}$;
if $\psi_{min} > 0$ **and** $\omega_{min} = 0$ **then**
    foreach $s'' \in \mathcal{S}$ **do**
        $s''.\psi_{seq} := max\{\psi_{seq}(S, s'', \mathcal{H})\}$; /* semantic relatedness */
        $s''.\omega_{seq} := -1$;
        if $\tau = *$ **then**
            if $s''.\psi_{seq} \not\geq \psi_{min}$ **then**
                $\mathcal{X} := \mathcal{X} \cup s''$;
        else
            $s''.\mu_\tau := \mu_\tau(s''.dist, \tau, \mathcal{F})$;
            if $s''.\psi_{seq} \geq \psi_{min}$ **and** $s''.\mu_\tau \geq \mu_{\tau_{min}}$ **then**
                $\mathcal{X} := \mathcal{X} \cup s''$;
else if $\psi_{min} = 0$ **and** $\omega_{min} > 0$ **then**
    foreach $s'' \in \mathcal{S}$ **do**
        $s''.\omega_{seq} := max\{\omega_{seq}(S, s'', \mathcal{H})\}$; /* semantic contradiction */
        $s''.\psi_{seq} := -1$;
        $s''.\mu_\tau := \mu_\tau(s''.dist, \tau, \mathcal{F})$;
        if $s''.\omega_{seq} \geq \omega_{min}$ **and** $s''.\mu_\tau \geq \mu_{\tau_{min}}$ **then**
            $\mathcal{X} := \mathcal{X} \cup s''$;
if $\mathcal{X} \neq \emptyset$ **then**
    $hs := max\{abs(s''.\mu_\tau * s''.\psi_{min} * s''.\omega_{max}) \mid s'' \in \mathcal{X}\}$; /* highest-score */
    foreach $s'' \in \mathcal{X}$ **do**
        if $abs(s''.\mu_\tau * s''.\psi_{min} * s''.\omega_{max}) = hs$ **then**
            return $s' := s''$;
return s';

A set $\mathcal{F}$ of fuzzy sets is also token into account for handling the fuzzy degrees of μ_τ and μ_{sem}, with respect to a minimum occurrence degree $\mu_{\tau_{min}}$ (to integrate *tau-fuzzy* unexpectedness, see Chapter 5, Section 5.2). Notice that we assume that $\mathcal{F}$, μ_τ, μ_{sem}, and $\mu_{\tau_{min}}$ are predefined and globally accessible.

The algorithm first verifies the compatible form constraint on S and s, if not $S \lhd s$, then returns an empty sequence (line 5); if $S \lhd s$, the function $seqsat(S, s, \lhd)$ returns the set $\mathcal{S}$ of all maximal subsequences (i.e., without splitting itemsets) of $s'' \sqsubseteq s$ such that $S \lhd s''$ and $|s''| = |S|$. All sequences $s'' \in \mathcal{S}$ that cannot satisfy the constraint τ are removed (line 8).

Not difficult to see, the sequence $s'' \in \mathcal{S}$ having the maximal semantic relatedness degree $max\{\psi_{seq}(S, s'', \mathcal{H})\}$ or contradiction degree $max\{\omega_{seq}(S, s'', \mathcal{H})\}$ is also the sequence $s'' \sqsubseteq s$ having the same maximal degree such that $S \trianglelefteq s''$.

The algorithm uses the equations proposed in the previous sections by examining the values of ψ_{min} and ω_{min}: if $\psi_{min} > 0$ and $\omega_{min} = 0$, then compute the semantic relatedness degree of each sequence $s'' \in \mathcal{S}$ for further determining α-unexpected or β-unexpected sequence; if $\psi_{min} = 0$ and $\omega_{min} > 0$, then compute the semantic contradiction degree of each sequence $s'' \in \mathcal{S}$ for further determining γ-unexpected sequence. If the ψ_{seq} or ω_{seq} value of a sequence $s'' \in \mathcal{S}$ satisfies the required condition, and the fuzzy occurrence degree $s''.\mu_\tau \geq \mu_{\tau_{min}}$, then s'' is added to the candidate sequence set $\mathcal{X}$, where $s''.dist$ (line 17 and 14) is the offset of s'' in s, which must correspond to specified occurrence constraint τ.

As shown in Equation (6.2), totally $P(|I|, |C|)$ queries are needed for computing $\omega_{pat}(C, I, \mathcal{H})$ or $\psi_{pat}(C, I, \mathcal{H})$ of a concept pattern C and an itemset I on a hierarchy $\mathcal{H}$. If $|C| = |I|$, then totally $|I|!$ queries must be performed. Therefore, in the worst case, when $|S| = |s| = 1$ and $\|S\| = \|s\|$, totally $\|s\|!$ queries are required.

The proof is immediate since we have that $(m + n)! \geq m! + n!$. In the best case, when $\|S\| = \|s\| = |S| = |s|$, that is, s consists of the itemsets of 1 item, $\|s\|$ queries are required.

Therefore, for a sequence s such that $\|s\| = |s|$ and a generalized sequence S such that $S \lhd s$, the number of queries is the number of the combinations of $|S|$ itemsets in s, that is,

$$_{|s|}C_{|S|} = \begin{pmatrix} |s| \\ |S| \end{pmatrix} = \frac{|s|!}{|S|!(|s| - |S|)!}.$$

For instance, if $|s| = 10$ and $|S| = 5$, then totally $\begin{pmatrix} 10 \\ 5 \end{pmatrix} = 252$ queries are required.

6.6 Experiments

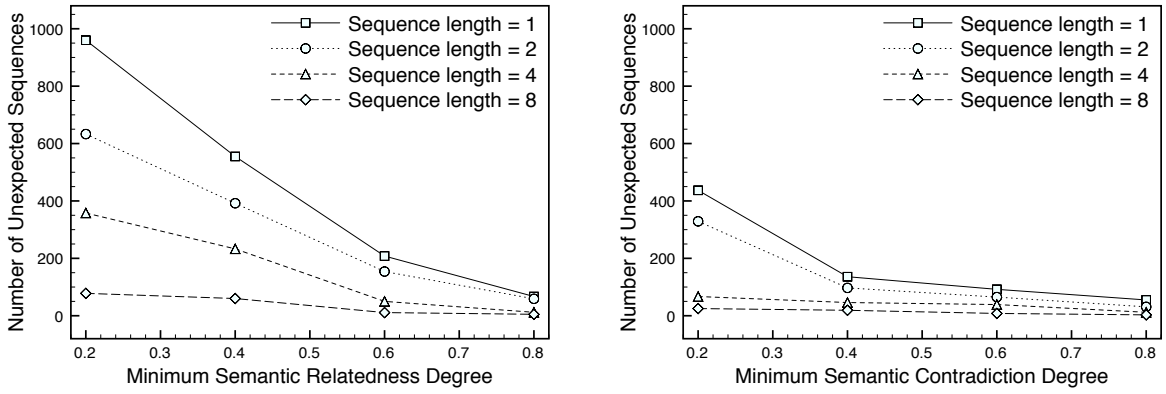
we perform the tests on extracting soft unexpected sequences with 20 soft beliefs, which are manually created from sequential patterns discovered in the data set with examining the concepts

of items.

The hierarchy used in experiments is built from the Web site structure and URI parameters, which contains 35 concepts with maximal path-length of 8, where the similarities between concepts are defined with expertise domain knowledge. An item-index file is used for mapping each item i to an concept c such that $i \models c$, and a concept-index file is used for indexing the path-length and semantic similarity between any two concepts contained in the hierarchy instead of traversing the hierarchy.

Only one category of occurrence constraint τ is considered with soft beliefs: $\tau = [X..Y]$ where $Y \geq X \geq 0$ are two integers. The soft beliefs are classified to 4 groups with respect to the length of S_β (1, 2, 4, 8), each group contains 5 soft beliefs. The length of S_α is no longer than 2. Since the fuzziness on semantic relatedness/contradiction is determined only by degree, we did not specify the fuzzy sets. In order to focus on the performance in considering hierarchies, the range $\tau \pm 2$ is used instead of computing the fuzzy occurrence degree.

Figure 6.5 shows the numbers of unexpected sequences extracted by using soft beliefs with concept hierarchy. The experimental results on soft beliefs show that the effectiveness of the proposed approach highly depends on the size of the sequence S_β in beliefs.



(a) Soft β-unexpected sequences. (b) Soft γ-unexpected sequences.

Figure 6.5: Number of soft unexpected sequences.

For instance, when $|S_\beta| = 1$, the number of β-unexpected sequences extremely increases with decreasing the minimum semantic relatedness degree ψ_{min}. In fact, according to the combinations of items in a sequence, if $|S_\beta|$ is a small value, then there are higher probability to satisfy the semantic relatedness required for matching S_β.

Thus, when $|S_\beta|$ is a small value, the probability to satisfy the semantic relatedness is much lower and much less unexpected sequences are extracted.

The execution time of each test is listed in Table 6.4, which shows that the time for extracting unexpected sequences significantly increases with the increase of $|S_\beta|$. However, the increase of execution time is slower than ${}_{14}C_1 \rightarrow {}_{14}C_2 \rightarrow {}_{14}C_4 \rightarrow {}_{14}C_8$ because with the increase of $|S_\beta|$, the

$\|S_\alpha\|$	$\psi_{min}, \omega_{min} = 0.2$	$\psi_{min}, \omega_{min} = 0.4$	$\psi_{min}, \omega_{min} = 0.6$	$\psi_{min}, \omega_{min} = 0.8$
1	22.1 s	22.1 s	20.4 s	19.2 s
2	93.1 s	90.2 s	93.8 s	90.7 s
4	577.8 s	563.3 s	581.8 s	569.7 s
8	2024.2 s	1998.8 s	1994.3 s	1955.2 s

Table 6.4: Total execution time of each test by using soft beliefs.

satisfaction of τ in the rest of an input sequence (i.e., $s \setminus s_\alpha$ where $s_\alpha \models S_\alpha$) becomes lower, and the step at line 8 in Algorithm 14 avoids matching all combinations of subsequences.

6.7 Discussion

In this chapter, we studied the generalizations of unexpected sequence discovery with respect to concept hierarchies of the taxonomy of data. We formalized the notions of generalized sequences and generalized sequence rules, and then we proposed two new types of unexpected sequences: generalized unexpected sequences and soft unexpected sequences.

Generalized unexpected sequences are determined against generalized beliefs, which consist of generalized sequence rules and semantic contradictions between generalized sequences. The construction of generalized belief system is the same procedure with the construction of a belief system addressed in the original MUSE framework presented in Chapter 4. In fact, the extraction of generalized unexpected sequences follows the same manner of MUSE, except to consider the matching between a sequence and a generalized sequence with respect to a concept hierarchy.

Therefore, we did not further list the related algorithms for discovering generalized unexpected sequences. In contrast, the algorithms and experiments of soft unexpected sequence discovery are carefully studied.

Soft unexpected sequences are determined in hierarchical with respect to generalized sequence rules, instead of explicitly constructing a belief system. The most advantages of the approach SOFTMUSE to discover soft unexpected sequence include:

1. Generalization of data is addressed by using generalized sequence rules;

2. Semantic contradictions are no longer required, where the determination of semantic contradiction is replaced by computing semantic relatedness/contradiction degrees;

3. The *tau-fuzzy* unexpectedness is integrated into SOFTMUSE, and the semantic relatedness/contradiction degree can also be described by using fuzzy sets.

Notice that in soft unexpected sequence discovery, when we consider the semantics, we can determine the semantic contradiction between two single items, for example, between "login" and "logout". However, to define the semantic contradiction for operational conjunction of items with temporal order is hard, which is still an open problem in semantics related data mining tasks.

In the framework MUSE, the belief system consists of sequence rules and semantic contradiction relations, so that the unexpected sequences can be strictly determined within the supervised discovery process. However, because the auto-determination of semantic contradiction within SOFTMUSE is unsupervised, the validation of discovered unexpected sequences is required. Hence, in the next chapter, we will take account of the evaluation of the discovered unexpected sequences in the self-validation schema in terms of the notions of unexpected sequential patterns. We will also present the notions of unexpected implication rules for investigating the structural associations and predictions of unexpectedness in sequence databases.

Chapter 7

Unexpected Sequential Patterns and Implication Rules

In previous chapter, we developed and extended the framework MUSE for discovering various unexpected sequences with fuzzy methods and generalizations in data taxonomy. The followed important task is therefore to evaluate the quality of the discovered unexpected sequences, and then to acquire useful information from such sequences for studying the structure in order to predict the unexpectedness. In this chapter, we propose the notions of unexpected sequential patterns and unexpected implication rules for this purpose.

A part of the work presented in this chapter has been published in the journal *La Revue des Nouvelles Technologies de l'Information (RNTI)* and in the *International Journal of Business Intelligence and Data Mining (IJBIDM)*.

7.1 Introduction

We have discussed the problem relied on unexpected sequence discovery that the number and quality of discovered sequences strongly depend on the belief system, where the correctness of beliefs is ensured by the interpretation of domain expertise knowledge.

On the other hand, the discovered unexpected sequences may contain low frequency noisy data in the database, which cannot be avoided in the discovery process if they violate some beliefs.

Example 31 Let us consider again the example discussed at the end of Chapter 4.

$$\mathcal{S} = \left\{ \begin{array}{l} s_1 = \cdots\cdots(a)(b)\cdots\cdots\cdots(c)\cdots\cdots \\ s_2 = \cdots\cdots\cdots(a)(b)\cdots\cdots(c)\cdots\cdots \\ s_3 = \cdots(a)(b)\cdots\cdots\cdots(c)\cdots\cdots\cdots \\ s_4 = \cdots\cdots\cdots\cdots(a)(b)\cdots\cdots(c)\cdots \\ s_5 = (a)\cdots\cdots\cdots\cdots\cdots\cdots\cdots(b) \end{array} \right\}.$$

Given a belief b consisting of a sequence implication rule $\langle(a)(b)\rangle \rightarrow^* \langle(c)(d)\rangle$, the sequences in the sequence set $\mathcal{S}$ are α-unexpected because for each sequence $s \in \mathcal{S}$ we have that $\langle(a)(b)\rangle \sqsubseteq s$ and $\langle(a)(b)(c)(d)\rangle \not\sqsubseteq s$. However, the sequence s_5 has a completely different structure than other sequences, which can be considered as noisy data that should not be covered by the belief b. Obviously, the approaches proposed in previous chapters cannot filter a sequence like s_5 from the result unexpected sequence set. Moreover, after examining the frequent common structure of the rest unexpected sequences, we can find that a rule $\langle(a)(b)(c)\rangle \rightarrow^* \langle(d)\rangle$ can better state the unexpectedness. □

Therefore, in this chapter, we study the validation of the discovered unexpected sequences for the *evaluation – interpretation – update* process shown in Figure 7.1.

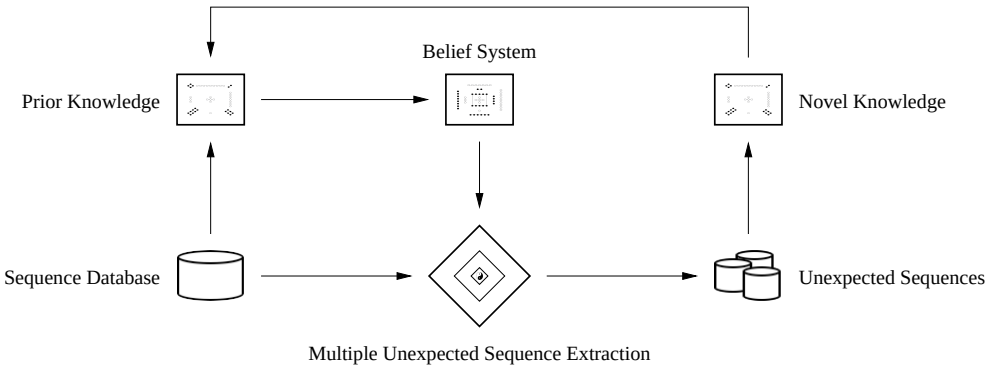


Figure 7.1: The *evaluation – interpretation – update* process.

We propose a *self-validation* process for evaluating unexpected sequences with the notions of *unexpected sequential patterns*. In this process, for a set of unexpected sequences, we first discover unexpected sequential patterns, which include internal and external unexpected sequential patterns for depicting the frequent common structures inside and outside the unexpectedness. Hence, more *contributions* to generated unexpected sequential patterns an unexpected sequence has, more *reliable* the unexpected sequence is.

Further, with mining sequential patterns in different compositions of unexpected sequences, we also propose the notions of *unexpected implication rules*, including *unexpected class rule*, *unexpected association rule*, and *unexpected occurrence rule*, for understanding what happens associated with the unexpectedness, what implies the unexpectedness, and what the unexpectedness implies.

The approaches proposed in this chapter have close connections with sequential pattern mining [AS95].

In the past fifteen years, many approaches have been proposed and developed with focusing on improving the efficiency of execution time and memory usage in sequential pattern mining, such as *Apriori* ([AS95]), *GSP* ([SA96b]), *PSP* ([MCP98]), *PrefixSpan* ([PHMAP01]), *SPADE* ([Zak01]),

SPAM ([AFGY02]), and *DISC* ([Ca09]). There also exist the studies of mining the variances of sequential patterns, such as *SPIRIT* ([GRS99]) for mining sequential patterns with *regular expression* constraints, or *CloSpan* ([YHA03]) and *BIDE* ([WH04]) for mining *closed sequential patterns*.

In the approach *SPIRIT*, the regular expression constraints on the elements in sequences are considered within the sequential pattern mining process, where each regular expression constraint is represented as a deterministic finite automata. A sequence s is accepted by the automata if following the sequence of transitions for the elements of s from the start state results in an accept result, and only the sequential patterns that can be accepted by the automata will be extracted. Different from regular expression constraints, the notion of closed sequential patterns is based on the *closure* property of the support of sequences, which can be considered as a frequency constraint on the mining process. For instance, given three sequences $s_1 = \langle(a)(b)(c)\rangle$, $s_2 = \langle(a)(b)\rangle$, and $s_3 = \langle(a)\rangle$, if in a database $\mathcal{D}$ we have that $freq(s_1, \mathcal{D}) = 0.5$, $freq(s_2, \mathcal{D}) = freq(s_3, \mathcal{D}) = 0.7$, then s_1 and s_2 are two closed sequential patterns.

The sequential pattern mining involved in our proposed approaches can be achieved by any existing algorithms. In this thesis, the approaches *PrefixSpan* and *CloSpan* are used in our implementation of experiments.

The rest of this chapter is organized as follows. In Section 7.2, we first study the composition of an unexpected sequence in formalizing the notions of unexpected feature and host sequence of unexpectedness, with which we then propose the notions of internal and external unexpected sequential patterns, and then we propose a self-validation process for evaluating the discovered unexpected sequences. In Section 7.3, we propose the notions of unexpected implication rules, which include unexpected class rule, unexpected association rule, and unexpected occurrence rule. We show experimental results of discovering unexpected sequential patterns and unexpected implication rules on real Web server access data in Section 7.4. Section 7.5 is a discussion.

7.2 Unexpected Sequential Patterns

In this section, we first study the structure of an unexpected sequence with the notions of unexpected feature and host sequence, and then propose the notions of internal and external unexpected sequential patterns for measuring the discovered unexpected sequences.

7.2.1 Unexpected Feature and Host Sequence

In the previous sections, we proposed three forms of unexpected sequences and developed the discovery algorithms. In order to investigate the unexpected sequences within unified structures,

in this section, we propose the notions of unexpected feature and host sequence for describing the unexpectedness.

The *unexpected feature* of an sequence represents the state of being unexpected, which is defined as follows.

Definition 29 (Unexpected feature) *The unexpected feature of an unexpected sequence s is the consecutive subsequence $s_f \sqsubseteq_c s$ in which the unexpected elements can be strictly bordered.*

The *host sequence* represents the rest of an unexpected sequence without the state of being unexpected, which is defined as follows.

Definition 30 (Host sequence) *The host sequence of an unexpected sequence s is the maximal subsequence $s_a \sqsubseteq_c s$ after eliminating the premise sequence and/or the conclusion sequence, and/or the contradicting sequence.*

Obviously, with respect to the occurrence pair set obtained by matching unexpected sequences, there can exist multiple features and host sequences of an unexpected sequence. Assuming that an unexpected sequence has multiple features, let $\bigcup s_f$ be the set of unexpected features of this sequence. For a feature $s_f \in \bigcup s_f$, if for any other features $s_f{}' \in \bigcup s_f$, we have $|s_f| \leq |s_f{}'|$, then s_f is a *minimal feature*; if for any other features $s_f{}' \in \bigcup s_f$, we have $|s_f| \geq |s_f{}'|$, then s_f is a *maximal feature*.

According to the various forms of unexpectedness with respect to the sequence rule set of a belief, we discuss the unexpected feature and host sequence with the following cases. Without loss of generality, we consider the sequence rule sets of a single rule for simplifying the descriptions.

First, let us consider the unexpected feature and host sequence stated by a belief with a sequence association rule, that is,

$$b = \left\{ \; s_\alpha \rightarrow s_\beta \; \right\} \wedge \left\{ \; s_\beta \not\approx_{sem} s_\gamma \; \right\}. \tag{7.1}$$

As presented in previous sections, only γ-unexpected sequences can be determined by the belief listed in Equation (7.1). Thus, given a sequence s such that $s_\alpha \sqsubseteq s$, the presence of $s_\gamma \sqsubseteq s$ constructs the γ-unexpectedness, which can be written as $s \models (s_\alpha \rightarrow s_\gamma)$. The feature of such an unexpected sequence s is that a consecutive subsequence $s_f \sqsubseteq_c s$ such that $s_\gamma \sqsubseteq_\perp^\top s_f$; the host sequence of s is the sequence $s \setminus s_\gamma$.

Example 32 Given a belief

$$b = \left\{ \; \langle (e)(e) \rangle \rightarrow \langle (a)(b) \rangle \; \right\} \wedge \left\{ \; \langle (a)(b) \rangle \not\approx_{sem} \langle (c)(d) \rangle \; \right\},$$

the sequence

$$s = \langle (a)(c)(e)(c)(e)(d)(c)(d)(e)(e) \rangle$$

is γ-unexpected to the belief b. We have that $\langle (c)(d) \rangle$ is a minimal unexpected feature and $\langle (c)(e)(c)(e)(d)(c)(d) \rangle$ is a maximal unexpected feature of the sequence s. According to s_{f_1}, the host sequence is $s_{a1} = \langle (a)(c)(e)(c)(e)(d)(e)(e) \rangle$, and to s_{f_2}, the host sequence is $s_{a2} = \langle (a)(e)(c)(e)(d)(c)(e)(e) \rangle$. □

We now discuss the unexpected feature and host sequence stated by a belief with a sequence implication rule, that is,

$$b = \left\{ \; s_\alpha \rightarrow^\tau s_\beta \; \right\} \wedge \left\{ \; s_\beta \not\simeq_{sem} s_\gamma \; \right\}. \tag{7.2}$$

With different forms of the occurrence constraint τ, all α-, β-, and γ-unexpected sequences can be determined by the belief listed in Equation (7.2).

We first consider an α-unexpected sequence. Given a sequence $s \not\models_\alpha b$, which can be considered as $s_\alpha \sqsubseteq s$ and $s_\alpha \cdot s_\beta \sqsubseteq s$, and the unexpectedness can be viewed as the absence of s_β after the occurrence of s_α. Thus, we define the feature s_f of an α-unexpected sequence as follows:

$$s_f = s_\alpha{}' \;\text{ such that }\; s = s_a \cdot s_\alpha{}' \cdot s_c,$$

where $|s_p| \geq 0$ and $s_\alpha \sqsubseteq_c s_\alpha{}'$; with the same context, the host sequence is defined as:

$$s_h = s_a \cdot s_\alpha{}'' \cdot s_c \;\text{ where }\; s_\alpha{}'' = s_\alpha{}' \setminus s_\alpha.$$

We then consider a β-unexpected sequence. Given a sequence $s \not\models_\beta b$, which can be considered as $s \models (s_\alpha \rightarrow^{(*\setminus\tau)} s_\beta)$, where $(* \setminus \tau)$ denotes the complement of τ (e.g., if $\tau = [3..5]$, then $(* \setminus \tau)$ denotes the ranges $[0..2] \vee [6..*]$, see Equation (4.1) in Section 4.3.3). Thus, we define the feature s_f of a β-unexpected sequence as follows:

$$s_f = s_\alpha{}' \cdot s' \cdot s_\beta{}' \;\text{ such that }\; s = s_a \cdot s_\alpha{}' \cdot s' \cdot s_\beta{}' \cdot s_c,$$

where $|s_a|, |s_c| \geq 0$, $|s'| \models (* \setminus \tau)$, $s_\alpha \sqsubseteq_c s_\alpha{}'$, and $s_\beta \sqsubseteq_c s_\beta{}'$; with the same context, the host sequence is defined as:

$$s_h = s_a \cdot s_\alpha{}'' \cdot s' \cdot s_\beta{}'' \cdot s_c \;\text{ where }\; s_\alpha{}'' = s_\alpha{}' \setminus s_\alpha \;\text{ and }\; s_\beta{}'' = s_\beta{}' \setminus s_\beta.$$

Finally, we consider a γ-unexpected sequence. Given a sequence $s \not\models_\gamma b$, which can be considered as $s \models (s_\alpha \rightarrow^\tau s_\gamma)$. Therefore, we define the feature s_f of a γ-unexpected sequence as follows:

$$s_f = s_\alpha{}' \cdot s' \cdot s_\gamma{}' \;\text{ such that }\; s = s_a \cdot s_\alpha{}' \cdot s' \cdot s_\gamma{}' \cdot s_c,$$

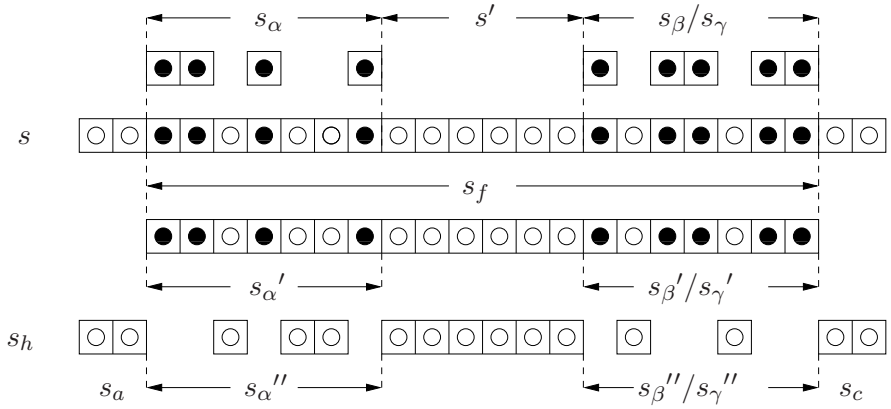


Figure 7.2: A schema of unexpected feature and host sequence.

where $|s_a|, |s_c| \geq 0$, $|s'| \models \tau$, $s_\alpha \sqsubseteq_c s_\alpha{}'$, and $s_\gamma \sqsubseteq_c s_\gamma{}'$; with the same context, the host sequence is defined as:

$$s_h = s_a \cdot s_\alpha{}'' \cdot s' \cdot s_\gamma{}'' \cdot s_c \quad \text{where} \quad s_\alpha{}'' = s_\alpha{}' \setminus s_\alpha \quad \text{and} \quad s_\gamma{}'' = s_\gamma{}' \setminus s_\gamma.$$

Figure 7.2 shows a schema of unexpected feature and host sequence with respect to a belief with predictive sequence implication rules. Not difficult to see, the unexpectedness stated by a belief of sequence implication rules can be represented by $\langle s_\alpha \, [|s'|] \, s_\beta \rangle$ or $\langle s_\alpha \, [|s'|] \, s_\gamma \rangle$, which are called a *signature* of the unexpectedness, where we define $|\langle s_\alpha \, [|s'|] \, s_\beta \rangle| = |s_\alpha| + |s'| + |s_\beta|$. To respect all forms of unexpectedness, we define the *signature* of an α-unexpected sequence as the premise sequence s_α.

Example 33 Considering the belief

$$b = \left\{ \; \langle (a)(b) \rangle \rightarrow^{[2..10]} \langle (c)(d) \rangle \; \right\} \wedge \left\{ \; \langle (c)(d) \rangle \not\approx_{sem} \langle (e)(f) \rangle \; \right\},$$

the following sequences are γ-unexpected to b, where each dot stands for an itemset:

$$\mathcal{S}_U = \left\{ \begin{array}{l} s_1 = \cdots\cdots (a) \cdots (b) \cdots\cdots (e)(f) \cdots\cdots \\ s_2 = \cdots\cdots (a)(b) \cdots\cdots\cdots (e) \cdots (f) \cdots \\ s_3 = \cdots (a) \cdots (b) \cdots\cdots (e) \cdots\cdots (f) \cdots \end{array} \right\}.$$

Two signatures can be found in $\mathcal{S}_U$: $\langle (a)(b) \, [6] \, (e)(f) \rangle$ with respect to s_1 and s_3, and $\langle (a)(b) \, [9] \, (e)(f) \rangle$ with respect to s_2. □

With the notion of signature, given an unexpected sequence s_u stated by a belief with sequence implication rules and let s_t be the signature, the unexpected feature can be formally described as a maximal subsequence $s_f \sqsubseteq_c s$ such that $|s_f| = |s_t|$; the host sequence can be formally described as the sequence $s_h = s \setminus s_t$.

According to the algorithms `UxpsMatchAlpha` (Algorithm 2), `UxpsMatchBeta` (Algorithm 3), and `UxpsMatchGamma` (Algorithm 5), the occurrence position of an unexpected feature can be immediately located by selecting a preferred value (e.g., maximum or minimum) in the returned pair set of all occurrences of unexpectedness with respect to the starting position of the premise sequence.

Notice: *for reducing the redundancy of text, we will directly use the notations* s_α, $s_\alpha{}'$, $s_\alpha{}''$, s_β, $s_\beta{}'$, $s_\beta{}''$, s_γ, $s_\gamma{}'$, $s_\gamma{}''$, *and* ℓ_U *in the rest of this thesis with respect to the above analytical descriptions.*

7.2.2 Internal and External Unexpected Sequential Patterns

In the previous section, we introduced the notion of unexpected features, which represents the structure of the unexpectedness stated in an unexpected sequence. In this section, we propose the notions of internal and external unexpected sequential patterns.

Given a sequence database $\mathcal{D}$ and a belief b, the algorithms `UxpsMatchAlpha`, `UxpsMatchBeta`, and `UxpsMatchGamma` for discovering unexpectedness can be regarded as a bijective function $\pi_f : \mathcal{S}_U \rightarrow \mathcal{S}_F$ that projects an unexpected sequence $s \in \mathcal{S}_U$ to an unexpected feature $s_f \in \mathcal{S}_F$, and a bijective function $\pi_h : \mathcal{S}_U \rightarrow \mathcal{S}_H$ that projects an unexpected sequence $s \in \mathcal{S}_U$ to an host sequence $s_h \in \mathcal{S}_H$. The sets $\mathcal{S}_U$ and $\mathcal{S}_H$ are respectively called as the *unexpected feature set* and the *host sequence set* of the unexpected sequences stated by the belief b in the database $\mathcal{D}$.

Based on the unexpected feature set, we propose the notion of *internal unexpected sequential pattern* as follows.

Definition 31 (Internal unexpected sequential pattern) *Given a sequence database* $\mathcal{D}$ *and a belief* b*, an internal unexpected sequential pattern is a maximal frequent sequence contained in an unexpected feature set* $\mathcal{S}_F$ *determined by the belief* b*, with respect to a user defined minimum support threshold.*

Given a sequence database $\mathcal{D}$ and an unexpected feature set $\mathcal{S}_F$, we consider three criteria for measuring an internal unexpected sequential patterns. The base measure is called the *local support*, denoted as $supp_{local}(s, \mathcal{D})$ and defined as follows:

$$supp_{local}(s, \mathcal{D}) = supp(s, \mathcal{S}_F) = \frac{|\{s' \in \mathcal{S}_F \mid s \sqsubseteq s'\}|}{|\mathcal{S}_F|}. \tag{7.3}$$

As described in Definition 31, we use the local support for determining an internal unexpected sequential pattern. Once we obtain an internal unexpected sequential pattern, dented as u_i, we can further measure the quality by two additional measures: global support and confidence.

The *global support* of an internal unexpected sequential pattern u_i is the support of the sequence u_i in the database $\mathcal{D}$, denoted as $supp_{global}(u_i, \mathcal{D})$, that is,

$$supp_{global}(u_i, \mathcal{D}) = supp(u_i, \mathcal{D}) = \frac{|\{s \in \mathcal{D} \mid u_i \sqsubseteq s\}|}{|\mathcal{D}|}. \tag{7.4}$$

Example 34 Assume that $\mathcal{D}$ is a sequence database of 100 sequences and

$$\mathcal{S}_F = \left\{ \begin{array}{l} \langle(\mathbf{a})(ef)(x_1)(x_2)(c)(x_3)(\mathbf{d})\rangle \\ \langle(\mathbf{a})(x_4)(aef)(x_5)(c)(\mathbf{d})\rangle \\ \langle(\mathbf{a})(ef)(x_6)(x_7)(x_8)(\mathbf{d})\rangle \\ \langle(\mathbf{a})(c)(x_9)(x_{10})(\mathbf{d})\rangle \\ \langle(\mathbf{a})(x_{11})(c)(x_{12})(c)(x_{13})(x_{14})(\mathbf{d})\rangle \end{array} \right\}$$

is an unexpected feature set of 5 sequences, where $x_1 \neq x_2 \neq \ldots \neq x_{14}$ are different items. With the minimum local support threshold 0.5, we can find two internal unexpected sequential patterns $u_{i1} = \langle(a)(ef)(d)\rangle$ with local support $supp_{local}(u_{i1}, \mathcal{D}) = 0.6$ and $u_{i2} = \langle(a)(c)(d)\rangle$ with local support $supp_{local}(u_{i2}, \mathcal{D}) = 0.8$. Assume that in the database $\mathcal{D}$, there are 20 sequences that support the sequence u_{i1} and 50 sequences that support the sequence u_{i2}, then we have that $supp_{global}(u_{i1}, \mathcal{D}) = 0.2$ and $supp_{global}(u_{i2}, \mathcal{D}) = 0.5$. □

Based on host sequence sets, the *external unexpected sequential pattern* is defined as follows.

Definition 32 (External unexpected sequential pattern) *Given a sequence database $\mathcal{D}$ and a belief b, an external unexpected sequential pattern is a maximal frequent sequence contained in the host sequence set $\mathcal{S}_H$ of an unexpected feature set $\mathcal{S}_F$ determined by the belief b, with respect to a user defined minimum support threshold.*

External unexpected sequential patterns can also be measured by local support and global support., and confidence. Given a sequence database $\mathcal{D}$ and an host sequence set $\mathcal{S}_H$, the local support for determining external unexpected sequential patterns, denoted as $supp_{local}(s, \mathcal{D})$, is defined:

$$supp_{local}(s, \mathcal{D}) = supp(s, \mathcal{S}_H) = \frac{|\{s' \in \mathcal{S}_H \mid s \sqsubseteq s'\}|}{|\mathcal{S}_H|}. \tag{7.5}$$

If an external unexpected sequential pattern, denoted as u_e, is found, it can be further measured by the global support, denoted as $supp_{global}(u_e, \mathcal{D})$:

$$supp_{global}(u_e, \mathcal{D}) = supp(u_e, \mathcal{D}) = \frac{|\{s \in \mathcal{D} \mid u_e \sqsubseteq s\}|}{|\mathcal{D}|}. \tag{7.6}$$

Example 35 Assume that $\mathcal{D}$ is a sequence database of 100 sequences and

$$\mathcal{S}_U = \left\{ \begin{array}{l} \langle (e)(c)(y_1)(\mathbf{a})(ef)(x_1)(x_2)(c)(x_3)(\mathbf{d})(y_2)(y_3)(f)(y_4)\rangle \\ \langle (ae)(y_5)(\mathbf{a})(x_4)(aef)(x_5)(c)(\mathbf{d})(cf)(y_6)(y_7)\rangle \\ \langle (y_8)(y_9)(e)(f)(\mathbf{a})(ef)(x_6)(x_7)(x_8)(\mathbf{d})(y_{10})(y_{11})\rangle \\ \langle (c)(\mathbf{a})(c)(x_9)(x_{10})(\mathbf{d})(y_{12})\rangle \\ \langle (f)(y_{13})(\mathbf{a})(x_{11})(c)(x_{12})(c)(x_{13})(x_{14})(\mathbf{d})(y_{14})(c)\rangle \end{array} \right\}$$

is a set of 5 unexpected sequences, where the $(\mathbf{a})\cdots(\mathbf{d})$ parts are unexpected features and $x_1 \neq x_2 \neq \ldots \neq x_{14} \neq y_1 \neq y_2 \neq \ldots \neq y_{14}$ are different items. We can generate the host sequence set

$$\mathcal{S}_H = \left\{ \begin{array}{l} \langle (e)(c)(y_1)(ef)(x_1)(x_2)(c)(x_3)(y_2)(y_3)(f)(y_4)\rangle \\ \langle (ae)(y_5)(x_4)(aef)(x_5)(c)(cf)(y_6)(y_7)\rangle \\ \langle (y_8)(y_9)(e)(f)(ef)(x_6)(x_7)(x_8)(y_{10})(y_{11})\rangle \\ \langle (c)(c)(x_9)(x_{10})(y_{12})\rangle \\ \langle (f)(y_{13})(x_{11})(c)(x_{12})(c)(x_{13})(x_{14})(y_{14})(c)\rangle \end{array} \right\}$$

without any difficulty. With the minimum local support threshold 0.5, we can find two external unexpected sequential patterns $u_{e1} = \langle (e)(ef)\rangle$ with local support $supp_{local}(u_{e1}, \mathcal{D}) = 0.6$ and $u_{e2} = \langle (c)(c)\rangle$ with local support $supp_{local}(u_{e2}, \mathcal{D}) = 0.8$. Assume that in the database $\mathcal{D}$, there are 20 sequences that support the sequence u_{e1} and 50 sequences that support the sequence u_{e2}, then we have that $supp_{global}(u_{e1}, \mathcal{D}) = 0.2$ and $supp_{global}(u_{e2}, \mathcal{D}) = 0.5$. □

The extraction of internal and external unexpected sequential patterns can be performed by using many existing efficient sequential pattern mining approaches in an unexpected feature set $\mathcal{S}_F$ and an host sequence set $\mathcal{S}_H$ with respect to a user specified minimum local support threshold. Once an unexpected sequential pattern is extracted, the computation of global support can be performed by the following simple algorithm `GlobalSupport` (Algorithm 15).

Algorithm 15: `GlobalSupport` $(s, \mathcal{D})$: Computing global support of an unexpected sequential pattern.

Input : An unexpected sequential pattern s, and a sequence database $\mathcal{D}$.
Output : The global support $supp_{global}(s, \mathcal{D})$ of s.

```
1 supp := 0;
2 foreach s' ∈ D do
3     pos := SeqMatchFirst(s, s', pair(0, |s'| − 1));
4     if pos.first ≠ −1 then
5         supp := supp + 1;
6 return supp/|D|;
```

With respect to a belief, internal unexpected sequential patterns depict frequent structures of the unexpected features stated by the unexpectedness; external unexpected sequential patterns

depict frequent correlations between the unexpected sequences that hold the unexpectedness. The local support permits extracting frequent structures of unexpectedness or frequent correlations between unexpected sequences, and the global support further permits measuring the quality of discovered structures correlations. Not difficult to see, an unexpected sequential pattern with high local support but low global support is more interesting than that with low local support but high global support.

7.2.3 Evaluating Unexpected Sequences

Given a sequence database $\mathcal{D}$ and a sequence belief base $\mathcal{B}$, the tasks of the discovery and evaluation of unexpected sequences can be formally described as the following problems.

Problem 1 *Given a belief $b \in \mathcal{B}$, find all unexpected sequences $s \in \mathcal{D}$ such that $s \nVdash b$.*

Problem 2 *Given an extracted unexpected sequence $s \in \mathcal{D}$ such that $s \nVdash b$ and $b \in \mathcal{B}$, evaluate the quality of s.*

In Section 4.3, we studied three forms of unexpected sequences and proposed the algorithms of unexpected sequence discovery. With the notions of the unexpected features and host sequences proposed in Section 7.2.1 and unexpected sequential patterns proposed in Section 7.2.2, in this section, we propose a self-validation approach to the evaluation of discovered unexpected sequences.

Notice that in the rest of this chapter, we discuss the unexpected sequence set $\mathcal{S}$, the unexpected feature set $\mathcal{S}_F$, the host sequence set $\mathcal{S}_H$, the internal unexpected sequential pattern set $\mathcal{U}_I$, and the external unexpected sequential pattern set $\mathcal{U}_E$ within the context of *being determined by the same unexpectedness with respect to the same belief* by using the term *the set*, where all empty sequences are counted into the size of set.

We now present a *self-validation* method for evaluating an unexpected sequence by evaluating its coherence in the internal and unexpected sequential pattern sets $\mathcal{U}_I$ and $\mathcal{U}_E$, with respect to its contribution to the unexpectedness and its correspondence in host sequences. In order to facilitating the descriptions, we represent an unexpected sequence s as a tuple $\langle s, s_f, s_h \rangle$ for the evaluation of unexpected sequences, where s_f is the unexpected feature of s and s_h is the host sequence.

The unexpectedness stated in the unexpected sequence $\langle s, s_f, s_h \rangle$ can be evaluated with ranking the importance contributed by $\langle s, s_f, s_h \rangle$ in generating the internal unexpected sequential pattern set $\mathcal{U}_I$. Hence, we propose the notion of *contribution degree* of an unexpected sequence as follows.

Definition 33 (Contribution degree) *The contribution degree of an unexpected sequence $\langle s, s_f, s_h \rangle$, denoted as $\rho_f \langle s, s_f, s_h \rangle$, is the maximal local support value in all internal unexpected sequential*

patterns $u_i \in \mathcal{U}_I$ *such that* $u_i \sqsubseteq s_f$*, that is,*

$$\rho_f \langle s, s_f, s_h \rangle = max\{supp_{local}(u_i, \mathcal{S}_F) \mid u_i \in \mathcal{U}_I, u_i \sqsubseteq s_f\}.$$

The contribution degree $\rho_f \langle s, s_f, s_h \rangle$ *is also denoted as* $\rho_f(s)$*.*

We also propose the notion of *correspondence degree* to measure the association between an unexpected sequence and all other sequences stating the same unexpectedness, which is defined as follows.

Definition 34 (Correspondence degree) *The correspondence degree of an unexpected sequence* $\langle s, s_f, s_h \rangle$*, denoted as* $\rho_h \langle s, s_f, s_h \rangle$*, is the maximal local support value in all external unexpected sequential patterns* $u_e \in \mathcal{U}_E$ *such that* $u_e \sqsubseteq s_h$*, that is,*

$$\rho_h \langle s, s_f, s_h \rangle = max\{supp_{local}(u_e, \mathcal{S}_H) \mid u_e \in \mathcal{U}_E, u_e \sqsubseteq s_h\}.$$

The correspondence degree $\rho_h \langle s, s_f, s_h \rangle$ *is also denoted as* $\rho_h(s)$*.*

We therefore finally propose the notion of *unexpectedness degree* to measure the quality of an unexpected sequence, which is defined as follows.

Definition 35 (Unexpectedness degree) *The unexpectedness degree of an unexpected sequence* $\langle s, s_f, s_h \rangle$*, denoted as* $\rho \langle s, s_f, s_h \rangle$*, is a non-negative real number value*

$$\rho \langle s, s_f, s_h \rangle = \frac{\theta_f(\rho_f \langle s, s_f, s_h \rangle) + \theta_h(\rho_h \langle s, s_f, s_h \rangle)}{\theta_f + \theta_h},$$

where $\theta_f > 0$ *and* $\theta_h > 0$ *are two integers standing for user preferences of unexpectedness and association. The unexpectedness degree* $\rho \langle s, s_f, s_h \rangle$ *is also denoted as* $\rho(s)$*.*

The unexpectedness degree of a unexpected sequence addresses that the more contribution to the unexpectedness and the more correspondence to other unexpected sequences stating the same unexpectedness, the more importance for this sequence.

Example 36 Let us consider the unexpected sequence set $\mathcal{S}$, the unexpected feature set $\mathcal{S}_F$, and the host sequence set $\mathcal{S}_H$ addressed in Example 34 and Example 35:

$$\mathcal{S} = \left\{ \begin{array}{l} s_1 = \langle (e)(c)(y_1)(\mathbf{a})(ef)(x_1)(x_2)(c)(x_3)(\mathbf{d})(y_2)(y_3)(f)(y_4) \rangle \\ s_2 = \langle (ae)(y_5)(\mathbf{a})(x_4)(aef)(x_5)(c)(\mathbf{d})(cf)(y_6)(y_7) \rangle \\ s_3 = \langle (y_8)(y_9)(e)(f)(\mathbf{a})(ef)(x_6)(x_7)(x_8)(\mathbf{d})(y_{10})(y_{11}) \rangle \\ s_4 = \langle (c)(\mathbf{a})(c)(x_9)(x_{10})(\mathbf{d})(y_{12}) \rangle \\ s_5 = \langle (f)(y_{13})(\mathbf{a})(x_{11})(c)(x_{12})(c)(x_{13})(x_{14})(\mathbf{d})(y_{14})(c) \rangle \end{array} \right\};$$

$$\mathcal{S}_F = \left\{ \begin{array}{l} \langle(\mathbf{a})(ef)(x_1)(x_2)(c)(x_3)(\mathbf{d})\rangle \\ \langle(\mathbf{a})(x_4)(aef)(x_5)(c)(\mathbf{d})\rangle \\ \langle(\mathbf{a})(ef)(x_6)(x_7)(x_8)(\mathbf{d})\rangle \\ \langle(\mathbf{a})(c)(x_9)(x_{10})(\mathbf{d})\rangle \\ \langle(\mathbf{a})(x_{11})(c)(x_{12})(c)(x_{13})(x_{14})(\mathbf{d})\rangle \end{array} \right\};$$

$$\mathcal{S}_H = \left\{ \begin{array}{l} \langle(e)(c)(y_1)(ef)(x_1)(x_2)(c)(x_3)(y_2)(y_3)(f)(y_4)\rangle \\ \langle(ae)(y_5)(x_4)(aef)(x_5)(c)(cf)(y_6)(y_7)\rangle \\ \langle(y_8)(y_9)(e)(f)(ef)(x_6)(x_7)(x_8)(y_{10})(y_{11})\rangle \\ \langle(c)(c)(x_9)(x_{10})(y_{12})\rangle \\ \langle(f)(y_{13})(x_{11})(c)(x_{12})(c)(x_{13})(x_{14})(y_{14})(c)\rangle \end{array} \right\}.$$

With the minimum local and global support threshold 0.5, we can find two internal unexpected sequential patterns $u_{i1} = \langle(a)(ef)(d)\rangle : 0.6$ (we use a shorthand notation $u_{i1} = \langle(a)(ef)(d)\rangle : 0.6$ for u_{i1} having support value 0.6) and $u_{i2} = \langle(a)(c)(d)\rangle : 0.8$, and two external unexpected sequential patterns $u_{e1} = \langle(e)(ef)\rangle : 0.6$, $u_{e2} = \langle(c)(c)\rangle : 0.8$. Let us consider the unexpected sequence $s_3 \in \mathcal{S}$, we have that $u_{i1} \sqsubseteq s_1$ and $u_{i2} \sqsubseteq s_1$, according to the definition, $\rho_f(s_3) = 0.8$; we have also that $u_{e1} \sqsubseteq s_3$, so that $\rho_h(s_3) = 0.6$. For the the unexpected sequence $s_4 \in \mathcal{S}$, we have that $\rho_f(s_4) = 0.6$ and $\rho_h(s_4) = 0.8$ in the same manner. □

	$\rho_f(s)$	$\rho_h(s)$	θ_f:θ_h	$\rho(s)$	θ_f:θ_h	$\rho(s)$	θ_f:θ_h	$\rho(s)$
s_3	0.8	0.6	1:1	**0.7**	3:2	**0.72**	1:4	0.64
s_4	0.6	0.8	1:1	**0.7**	3:2	0.68	1:4	**0.76**

Table 7.1: Unexpectedness degrees with respect to user preferences.

The user preference factors θ_f and θ_h further adjust the weight of the contribution degree and of the correspondence degree. For instance, the following table (Table 7.1) lists the different unexpectedness degrees of the two unexpected sequences s_3 and s_4 listed in Example 36 with respect to different user preferences. In this example, s_3 has higher degree in contribution and s_4 has higher degree in association. If θ_f:θ_h = 1:1, then s_3 and s_4 have the same value of unexpectedness degree, however if we set θ_f:θ_h = 3:2 for considering that the contribution to unexpectedness has more importance, then s_3 is more interesting than s_4; if we set θ_f:θ_h = 1:4 for focusing on studying the associations between unexpected sequences, then s_4 is more interesting than s_3.

With investigating the contribution and association of an unexpected sequence, we can further indicate whether a sequence is unexpected because of incidentals, which possesses less interestingness.

7.3 Unexpected Implication Rules

Rule discovery is an important task in data mining. In this section, we propose three forms of unexpected implication rules including unexpected class rule, unexpected association rule, and unexpected occurrence rule.

7.3.1 Unexpected Class Rules

In this section, we propose the notion of *unexpected class rules* for investigating the most frequent structures[1] associated with the unexpected sequences.

Given the host sequence set $\mathcal{S}_H$ of the unexpected sequence set $\mathcal{S}_U$ determined by a belief with sequence class rules, an unexpected class rule is defined as follows.

Definition 36 (Unexpected class rule) *Given a belief b, let ℓ_U be the label of the unexpectedness stated by b. An unexpected class rule is a rule $\ell_U \rightarrow p_h$, where p_h is a maximal frequent sequence contained in the host sequence set $\mathcal{S}_H$ such that for the premise sequence $s_\alpha = \Lambda(b)$, for each conclusion sequence $s_\beta \in \Delta(b)$, and for each contradiction sequence $s_\gamma \in \Theta(b, s_\beta)$, we have that $s_\alpha \not\sqsubseteq p_h$, $s_\beta \not\sqsubseteq p_h$, and $s_\gamma \not\sqsubseteq p_h$.*

An unexpected class rule $\ell_U \rightarrow p_h$ depicts the implication that if a sequence s is unexpected to a belief on the class labeled by ℓ_U, then s contains the subsequence p_h that is no more specific than all the sequences mentioned in a belief. We measure an unexpected class rule by the *support* and *confidence*. As proposed in the definition, the sequence p_h in an unexpected class rule $\ell_U \rightarrow p_h$ is a sequential pattern discovered in the host sequence set $\mathcal{S}_H$, therefore the support of the rule is the support value of p_h in $\mathcal{S}_H$, denoted as $supp(\ell_U \rightarrow p_h, \mathcal{S}_H, \mathcal{D})$, that is,

$$supp(\ell_U \rightarrow p_h, \mathcal{S}_H, \mathcal{D}) = \frac{|s \in \mathcal{S}_H \mid p_h \sqsubseteq s|}{|\mathcal{S}_H|}.$$

The confidence of an unexpected class rule $\ell_U \rightarrow p_h$ is defined as the fraction of the number of the sequences in $\mathcal{S}_H$ that support p_h on the number of the sequences in $\mathcal{D}$ that support p_h, denoted as $conf(\ell_U \rightarrow p_h, \mathcal{S}_H, \mathcal{D})$, that is,

$$conf(\ell_U \rightarrow p_h, \mathcal{S}_H, \mathcal{D}) = \frac{|s \in \mathcal{S}_H \mid p_h \sqsubseteq s|}{|s \in \mathcal{D} \mid p_h \sqsubseteq s|}.$$

Notice. (1) The sequence database $\mathcal{D}$ is not required for computing the support value of an unexpected class rule, however, in order to keep a consistent form of the formulas, we consider $\mathcal{D}$ as a parameter in representing that such values are addressed in the sequence database $\mathcal{D}$. (2)

[1] We use the term *structure* for describing the *characteristics* of a sequence, which cover the notions of *structure*, *composition*, *behavior*, etc.

The sequence p_h mentioned in an unexpected class rule has the same definition of an external unexpected sequential pattern; however, given a belief b, for each conclusion sequence $s_\beta \in \Delta(b)$ and for each contradiction sequence $s_\gamma \in \Theta(b, s_\beta)$, we have that $s_\beta \not\sqsubseteq p_h$ and $s_\gamma \not\sqsubseteq p_h$, thus we use the notation p_h instead of the notation u_e and do not discuss p_h in the context of external unexpected sequential patterns.

Example 37 In Web navigation pattern analysis, an authenticated user navigation session starts from an access of `login` and then followed by an access of `home`; however the access of `home` without logged in will be redirected back to `login`. A belief can be

$$b = \left\{ \langle(\texttt{AUTH})\rangle \rightarrow \langle(\texttt{login})(\texttt{home})\rangle \right\} \wedge \left\{ \langle(\texttt{login})(\texttt{home})\rangle \not\approx_{sem} \langle(\texttt{home})(\texttt{login})\rangle \right\},$$

which describes that `login` should be followed by `home` but the inverse is not allowed: if a user try to directly access `home`, and the session will be terminated by bring the user to `login`. An unexpected user navigation session may consist of a session identifier (e.g., date and time, remote address, and/or user agent information of the session) and accesses such as $\langle \mathcal{I}_s(\texttt{home})(\texttt{login})\rangle$, where $\mathcal{I}_s$ represents the session identifier. Let $\texttt{U}_b$ label such unexpectedness, then, the rule $\texttt{U}_b \rightarrow \langle \mathcal{I}'_s \rangle$ is an unexpected class rule if the subset $\mathcal{I}'_s$ of session identifier present at such sessions is notably different from other sessions, that is, the rule $\texttt{U}_b \rightarrow \langle \mathcal{I}'_s \rangle$ has a high confidence value. □

Given a sequence database $\mathcal{D}$ and the host sequence set $\mathcal{S}_H$ discovered with respect to a belief b with sequence class rules, we consider the discovery of unexpected class rules as a two-phase process in order to obtain the maximal flexibility.

In the first phase, with a user defined minimum support threshold $supp_{min}$, a set $\mathcal{P}_H$ of sequential patterns can be extracted from the host sequence set $\mathcal{S}_H$ by using many existing efficient sequential pattern mining algorithms.

The second phase is listed in Algorithm 16 (`UnexpClassRules`: Mining Unexpected Class Rules), which accepts an unexpectedness class label ℓ_U, a sequential pattern set $\mathcal{P}_H$, an exclude sequence set $\mathcal{S}_X$, a sequence database $\mathcal{D}$, and a minimum confidence threshold $conf_{min}$ as inputs and outputs all unexpected class rules with respect to $conf_{min}$. The exclude sequence set contains the conclusion sequences and contradiction sequences of a belief b, by which the unexpectedness is determined, that is,

$$\mathcal{S}_X = \Delta(b) \cup \bigcup_{s_\beta \in \Delta(b)} \Theta(b, s_\beta).$$

For each sequence $p_h \in \mathcal{S}_H$, the algorithm first verifies that for each conclusion sequence $s' \in \mathcal{S}_X$, whether $s' \not\sqsubseteq p_h$: if $s' \in \mathcal{S}_X$, the algorithm removes all s' from p_h. Then, the algorithm verifies whether the confidence of p_h in $\mathcal{D}$ satisfies $conf_{min}$. If $conf_{min}$ is satisfied, then the algorithm generates a new unexpected class rules from p_h; finally the algorithm outputs all generated unexpected

class rules.

Algorithm 16: `UnexpClassRules` (ℓ_U, $\mathcal{P}_H$, $\mathcal{S}_X$, $\mathcal{D}$, $conf_{min}$) : Mining unexpected class rules.

Input : A class label ℓ_U, a sequential pattern set $\mathcal{P}_H$, an exclude sequence set $\mathcal{S}_X$, a sequence database $\mathcal{D}$, and a minimum confidence threshold $conf_{min}$.

Output : All unexpected class rules with respect to $conf_{min}$.

```
1  R := ∅;
2  foreach p_h ∈ P_H do
3      foreach s' ∈ S_X do
4          if s' ∈ p_h then
5              p_h := p_h \* s';
6      supp := count(p_h, D);
7      if (conf := p_h.count/supp) ≥ conf_min then
8          r := ClassRule.Create(ℓ_U, p_h, p_h.supp, conf);
9          R := R ∪ r;
10 return R;
```

The model of unexpected class rules also permits studying the frequent structures of all types of unexpected sequences, where we consider the unexpectedness determined by a belief b as a class labeled by ℓ_U. For a sequential pattern $p_h \in \mathcal{P}_H$ extracted from the host sequence set $\mathcal{S}_H$, the algorithm `UnexpClassRules` generates a rule $\ell_U \rightarrow p_h$, where $p_h = p_h \setminus^* (s' \in \mathcal{S}_X)$ and the exclude sequence set $\mathcal{S}_X$ is defined as

$$\mathcal{S}_X = \Lambda(b) \cup \Delta(b) \cup \bigcup_{s_\beta \in \Delta(b)} \Theta(b, s_\beta).$$

Example 38 Let us consider the unexpected sequence set $\mathcal{S}_U$ illustrated in Example 35, where

$$\mathcal{S}_U = \left\{ \begin{array}{l} \langle (e)(c)(y_1)(\mathbf{a})(ef)(x_1)(x_2)(c)(x_3)(\mathbf{d})(y_2)(y_3)(f)(y_4) \rangle \\ \langle (ae)(y_5)(\mathbf{a})(x_4)(aef)(x_5)(c)(\mathbf{d})(cf)(y_6)(y_7) \rangle \\ \langle (y_8)(y_9)(e)(f)(\mathbf{a})(ef)(x_6)(x_7)(x_8)(\mathbf{d})(y_{10})(y_{11}) \rangle \\ \langle (c)(\mathbf{a})(c)(x_9)(x_{10})(\mathbf{d})(y_{12}) \rangle \\ \langle (f)(y_{13})(\mathbf{a})(x_{11})(c)(x_{12})(c)(x_{13})(x_{14})(\mathbf{d})(y_{14})(c) \rangle \end{array} \right\}$$

can be discovered with respect to a belief

$$b_{\texttt{B[ID]}} = \left\{ \langle (a) \rangle \rightarrow^{[2..6]} \langle (c) \rangle \right\} \wedge \left\{ \langle (c) \rangle \not\approx_{sem} \langle (d) \rangle \right\}.$$

We can therefore label this γ-unexpectedness as `B-ID-GA`. According to `UnexpClassRules`, we have the exclude sequence set $\mathcal{S}_X = \{\langle (a) \rangle, \langle (c) \rangle, \langle (d) \rangle\}$. Given $supp_{min} = 0.5$, the sequence $p_h = \langle (e)(a)(ef)(d) \rangle$ is a sequential pattern with support 0.6, and we have that

$$p_h = p_h \setminus^* (s' \in \mathcal{S}_X) = \langle (e)(ef) \rangle.$$

Let $conf_{min} = 0.1$, if the sequence database $\mathcal{D}$ contains 100 sequences and 10 of them support p_h, then we have the rule

$$\texttt{B-ID-GA} \rightarrow \langle (e)(ef) \rangle$$

such that $supp(\texttt{B-ID-GA} \rightarrow \langle (e)(ef) \rangle, \mathcal{S}_H, \mathcal{D}) = 0.6$ and $conf(\texttt{B-ID-GA} \rightarrow \langle (e)(ef) \rangle, \mathcal{S}_H, \mathcal{D}) = 0.3$, where $\mathcal{S}_H$ is the host sequence set, see Example 35. □

7.3.2 Unexpected Association Rules

In this section, we propose the notion of *unexpected association rules*, including *local unexpected association rules* and *global unexpected association rule*, for investigating the most frequent structures associated with the unexpected sequences.

Given the unexpected sequence set $\mathcal{S}_U$, a local unexpected association rule is formally defined as follows.

Definition 37 (Local unexpected association rule) *Given a belief b, let $s_\alpha = \Lambda(b)$ be the premise sequence. A local unexpected association rule is a rule $s_\alpha \rightarrow p_h$, where p_h is a maximal frequent sequence contained in the feature set $\mathcal{S}_F$ such that for each conclusion sequence $s_\beta \in \Delta(b)$ and for each contradiction sequence $s_\gamma \in \Theta(b, s_\beta)$, we have that $s_\alpha \not\sqsubseteq p_h$, $s_\beta \not\sqsubseteq p_h$, and $s_\gamma \not\sqsubseteq p_h$.*

A local unexpected association rule $s_\alpha \rightarrow p_h$ depicts the implication that if the premise sequence s_α occurs together with the sequence p_h, then the conclusion sequence s_β will occur without respect to the occurrence constraint τ, or the contradiction sequence s_γ will occur with respect to the occurrence constraint τ.

We measure a local unexpected association rule by the *support* and *confidence*. We define the support of an unexpected association rule as the support value of p_h in the feature set $\mathcal{S}_F$, denoted as $supp(s_\alpha \rightarrow p_h, \mathcal{S}_F, \mathcal{D})$, that is,

$$supp(s_\alpha \rightarrow p_h, \mathcal{S}_F, \mathcal{D}) = \frac{|s \in \mathcal{S}_F \mid p_h \sqsubseteq s|}{|\mathcal{S}_F|}.$$

Without difficulty, the confidence of a local unexpected association rule $s_\alpha \rightarrow p_h$ is defined as the fraction of the number of the sequences in $\mathcal{S}_F$ that support p_h on the number of the sequences in $\mathcal{D}$ that support p_h, denoted as $conf(s_\alpha \rightarrow p_h, \mathcal{S}_F, \mathcal{D})$, that is,

$$conf(s_\alpha \rightarrow p_h, \mathcal{S}_F, \mathcal{F}) = \frac{|s \in \mathcal{S}_F \mid p_h \sqsubseteq s|}{|s \in \mathcal{D} \mid p_h \sqsubseteq s|}.$$

Given the unexpected sequence set $\mathcal{S}_U$, a global unexpected association rule is formally defined as follows.

Definition 38 (Global unexpected association rule) *Given a belief b, let $s_\alpha = \Lambda(b)$ be the premise sequence. A global unexpected association rule is a rule $s_\alpha \rightarrow p_h$, where p_h is a maximal*

frequent sequence contained in the host sequence set $\mathcal{S}_H$ *such that for each conclusion sequence* $s_\beta \in \Delta(b)$ *and for each contradiction sequence* $s_\gamma \in \Theta(b, s_\beta)$*, we have that* $s_\alpha \not\sqsubseteq p_h$*,* $s_\beta \not\sqsubseteq p_h$*, and* $s_\gamma \not\sqsubseteq p_h$*.*

A global unexpected association rule $s_\alpha \rightarrow p_h$ depicts the implication that if a sequence s is unexpected to a belief with the premise sequence s_α, then s contains the subsequences s_α and p_h, where p_h is no more specific than all the sequences mentioned in a belief.

We also measure an unexpected association rule by the *support* and *confidence* which are defined as follows:

$$supp(s_\alpha \rightarrow p_h, \mathcal{S}_H, \mathcal{D}) = \frac{|s \in \mathcal{S}_H \mid p_h \sqsubseteq s|}{|\mathcal{S}_H|};$$

$$conf(s_\alpha \rightarrow p_h, \mathcal{S}_H, \mathcal{D}) = \frac{|s \in \mathcal{S}_H \mid p_h \sqsubseteq s|}{|s \in \mathcal{D} \mid p_h \sqsubseteq s|}.$$

For instance, in Example 38, we can also find the global unexpected association rule

$$\langle (a) \rangle \rightarrow \langle (e)(ef) \rangle,$$

where $supp(\langle (a) \rangle \rightarrow \langle (e)(ef) \rangle, \mathcal{S}_H, \mathcal{D}) = 0.6$ and $conf(\langle (a) \rangle \rightarrow \langle (e)(ef) \rangle, \mathcal{S}_H, \mathcal{D}) = 0.3$; we also the following two local unexpected association rules

$$\langle (a) \rangle \rightarrow \langle (c) \rangle$$

and

$$\langle (a) \rangle \rightarrow \langle (ef) \rangle,$$

where $supp(\langle (a) \rangle \rightarrow \langle (c) \rangle, \mathcal{S}_F, \mathcal{D}) = 0.8$, $supp(\langle (a) \rangle \rightarrow \langle (ef) \rangle, \mathcal{S}_F, \mathcal{D}) = 0.8$, and the confidence values can be computed with respect to the whole database $\mathcal{D}$.

Example 39 Let us consider again the context of Web navigation pattern analysis, where a belief can be defined as

$$\left\{ \langle (\texttt{options}) \rangle \rightarrow^{[0..2]} \langle (\texttt{save})(\texttt{home}) \rangle \right\} \wedge \left\{ \langle (\texttt{save})(\texttt{home}) \rangle \not\approx_{sem} \langle (\texttt{save})(\texttt{options}) \rangle \right\}$$

for depicting that an authenticated user can change her/his preferences via the page `options` and the modifications can be saved by the action `save`; if the data inputed in `options` are not correct or in cases of invalid Captcha[2] input, user session will be redirected back to `options`. Hence, if user sessions contain $\langle (\texttt{options}) \rangle$ and $\langle (\texttt{save})(\texttt{options}) \rangle$ with a gap in the range [0..2], they are unexpected and can be caused by bad input data. If discovered unexpected association rules show that, for example, the unexpectedness is often associated with a particular type of Web browsers, that is, an unexpected association rule like

$$\langle (\texttt{options}) \rangle \rightarrow \langle (\texttt{user-agent-X}) \rangle$$

[2]The term *Captcha* stands for **C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part, see `http://en.wikipedia.org/wiki/Captcha` for details.

has high confidence value, it is then necessary to improve the site compatibility on user input checking in order to fit more types of browsers. On the other hand, if discovered unexpected association rules show that there exist many accesses of the page `captcha`, such as shown in a local unexpected association rule like

$$\langle(\texttt{options})\rangle \rightarrow \langle(\texttt{captcha})(\texttt{captcha})(\texttt{captcha})(\texttt{captcha})(\texttt{captcha})\rangle,$$

then it is necessary to check the accessibility of the Captcha system. □

With respect to methods proposed in the previous section for mining unexpected class rules, the discovery of unexpected association rules is immediate.

The unexpected association rule mining process is listed in Algorithm 16 (`UnexpAssocRules`: Mining Unexpected Association Rules), which accepts a premise sequence s_α, a sequential pattern set $\mathcal{P}$ generated from the unexpected feature set (for local unexpected association rules) or from the host sequence set (for global unexpected association rule), a sequence database $\mathcal{D}$, and a minimum confidence threshold $conf_{min}$ as inputs and outputs all unexpected association rules with respect to $conf_{min}$. The exclude sequence set contains the conclusion sequences and contradiction sequences of a belief b, by which the unexpectedness is determined, that is,

$$\mathcal{S}_X = \Lambda(b) \cup \Delta(b) \cup \bigcup_{s_\beta \in \Delta(b)} \Theta(b, s_\beta).$$

Algorithm 17: `UnexpAssocRules` (s_α, $\mathcal{P}$, $\mathcal{S}_X$, $\mathcal{D}$, $conf_{min}$) : Mining unexpected association rules.

Input : A premise sequence s_α, a sequential pattern set $\mathcal{P}$, an exclude sequence set $\mathcal{S}_X$, a sequence database $\mathcal{D}$, and a minimum confidence threshold $conf_{min}$.

Output : All unexpected association rules with respect to $conf_{min}$.

```
1  R := ∅;
2  foreach p_h ∈ P do
3      foreach s' ∈ S_X do
4          if s' ⊑ p_h then
5              p_h := p_h \* s';
6      supp := count(p_h, D);
7      if (conf := p_h.count/supp) ≥ conf_min then
8          r := AssociationRule.Create(s_α, p_h, p_h.supp, conf);
9          R := R ∪ r;
10 return R;
```

For each sequence $p_h \in \mathcal{P}$, the algorithm first verifies that for each conclusion sequence $s' \in \mathcal{S}_X$, whether $s' \not\sqsubseteq p_h$: if $s' \in \mathcal{S}_X$, the algorithm removs all s' from p_h. Then, the algorithm verifies

whether the confidence of p_h in $\mathcal{D}$ satisfies $conf_{min}$. If $conf_{min}$ is satisfied, then the algorithm generates a new unexpected association rules from p_h; finally the algorithm outputs all generated unexpected association rules.

7.3.3 Unexpected Occurrence Rules

We finally study the prediction problem of the unexpected sequences. In this section, we propose the *unexpected occurrence rules* in terms of the notions of *unexpected antecedent rule* and *unexpected consequent rule* for predicting the occurrence of the unexpectedness stated by the beliefs with sequence implication rules.

Before we can define the notions of unexpected occurrence rules, let us first examine the composition of a unexpected sequence set $\mathcal{S}_U$ determined by a belief. As discussed in Section 7.2.2, the unexpected sequence discovering process can be regarded as a bijective function $\pi_f : \mathcal{S}_U \rightarrow \mathcal{S}_F$, which projects the unexpected sequence set $\mathcal{S}_U$ to the unexpected feature set $\mathcal{S}_F$. For each unexpected sequence s_u, once the feature s_f is extracted, we can also obtain two subsequences $(s_a, s_c) \sqsubseteq s_u$ such that

$$s_u = s_a \cdot s_f \cdot s_c,$$

where $|s_a| \geq 0$ and $|s_c| \geq 0$. The sequence s_a is called the *antecedent sequence* of the unexpected feature s_f and the sequence s_c is called the *consequent sequence* of the unexpected feature. Hence, two bijective functions $\pi_a : \mathcal{S}_U \rightarrow \mathcal{S}_A$ and $\pi_c : \mathcal{S}_U \rightarrow \mathcal{S}_C$ can be further considered in the unexpected sequence discovering process, where π_a and π_c respectively project the unexpected sequence set $\mathcal{S}_U$ to the antecedent sequence set $\mathcal{S}_A$ and the consequent sequence set $\mathcal{S}_C$.

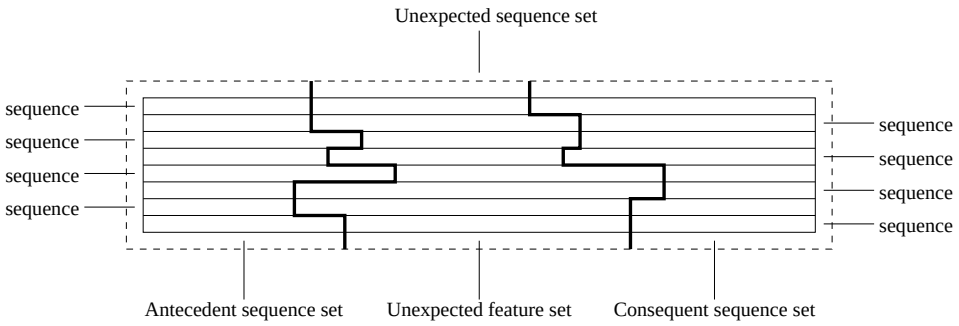


Figure 7.3: Composition of an unexpected sequence set.

An outline of the relations between the unexpected sequence set, the unexpected feature set, the antecedent sequence set, and the consequent sequence set is illustrated in Figure 7.3. An antecedent sequence or a consequent sequence can be empty with respect to the occurrence position of the unexpected feature, however we consider that each empty sequence has its unique sequence ID during the projection, so that the projection is bijective, as shown in Example 40.

Example 40 Let the sequence bordered in (**a**) and (**b**) be the unexpected feature, we have the projections $\pi_a : \mathcal{S}_U \rightarrow \mathcal{S}_A$ and $\pi_c : \mathcal{S}_U \rightarrow \mathcal{S}_C$ as follows:

$$\mathcal{S}_U = \left\{ \begin{array}{l} s_1 = \langle (\mathbf{a})(\mathbf{b})(c) \rangle \\ s_2 = \langle (\mathbf{a})(\mathbf{b})(d) \rangle \\ s_3 = \langle (c)(\mathbf{a})(\mathbf{b}) \rangle \\ s_4 = \langle (d)(\mathbf{a})(\mathbf{b}) \rangle \\ s_5 = \langle (\mathbf{a})(cd)(\mathbf{b}) \rangle \end{array} \right\}; \quad \mathcal{S}_A = \left\{ \begin{array}{l} s_1 = \emptyset \\ s_2 = \emptyset \\ s_3 = \langle (c) \rangle \\ s_4 = \langle (d) \rangle \\ s_5 = \emptyset \end{array} \right\}; \quad \mathcal{S}_C = \left\{ \begin{array}{l} s_1 = \langle (c) \rangle \\ s_2 = \langle (d) \rangle \\ s_3 = \emptyset \\ s_4 = \emptyset \\ s_5 = \emptyset \end{array} \right\}.$$

□

Given a sequence database $\mathcal{D}$ and belief b, let ℓ_U label the unexpectedness (α-, β-, or γ-unexpectedness) stated by the belief b, $\mathcal{S}_U$ be the unexpected sequence set, $\mathcal{S}_F$ be the unexpected feature set, $\mathcal{S}_A$ be the antecedent sequence set, and $\mathcal{S}_C$ be the consequent sequence set, we can therefore define the unexpected occurrence rules as follows.

Definition 39 (Unexpected antecedent rule) *An unexpected antecedent rule is a rule in the form $p_a \rightarrow^t \ell_U$, where p_a is a maximal frequent sequence contained in the antecedent sequence set $\mathcal{S}_A$, ℓ_U is an unexpectedness label, and $t = [min..max]$ is a gap range such that $min, max \in \mathbb{N}$ and $min \leq max$.*

Definition 40 (Unexpected consequent rule) *An unexpected antecedent rule is a rule in the form $\ell_U \rightarrow^t p_c$, where ℓ_U is an unexpectedness label, p_c is a maximal frequent sequence contained in the consequent sequence set $\mathcal{S}_C$, and $t = [min..max]$ is a gap range such that $min, max \in \mathbb{N}$ and $min \leq max$.*

An unexpected antecedent rule $p_a \rightarrow^t \ell_U$ depicts that the occurrence of the sequence p_a implies that the unexpectedness labeled by ℓ_U will occur within the gap range t. If the gap range t cannot be specified, then we write such a rule as $p_a \rightarrow^* \ell_U$, which depicts that the unexpectedness labeled by ℓ_U will occur after the occurrence of the sequence p_a.

We measure the interestingness of an unexpected antecedent rule with three criteria: *support*, *confidence*, and *gap distribution*. Given an unexpected antecedent rule $p_a \rightarrow^t \ell_U$, the sequence p_a is a sequential pattern in the antecedent sequence set $\mathcal{S}_A$, so that the support of an unexpected antecedent rule, denoted as $supp(p_a \rightarrow^t \ell_U)$, is defined as the support value of p_a in $\mathcal{S}_A$, that is,

$$supp(p_a \rightarrow^t \ell_U) = |\{s \in \mathcal{S}_A \mid p_a \sqsubseteq s\}|.$$

The confidence of an unexpected antecedent rule, denoted as $conf(p_a \rightarrow^t \ell_U)$, is defined as the fraction of the support of the sequential pattern p_a on the total number of sequences contained in $\mathcal{D}$ that support the sequence p_a, that is,

$$supp(p_a \rightarrow^t \ell_U) = \frac{|\{s \in \mathcal{S}_A \mid p_a \sqsubseteq s\}|}{|\{s \in \mathcal{D} \mid p_a \sqsubseteq s\}|}.$$

An unexpected consequent rule $\ell_U \rightarrow^t p_a$ depicts that the occurrence of the unexpectedness labeled by ℓ_U implies that the sequence p_a will occur within the gap range t. If the gap range t cannot be specified, then we write such a rule as $\ell_U \rightarrow^* p_a$, which depicts that the sequence p_c will occur after the occurrence of the unexpectedness labeled by ℓ_U.

We measure the interestingness of an unexpected consequent rule with the same criteria: *support*, *confidence*, and *gap distribution*. Given an unexpected consequent rule $\ell_U \rightarrow^t p_c$, the sequence p_c is a sequential pattern in the antecedent sequence set $\mathcal{S}_C$, so that the support of an unexpected consequent rule, denoted as $supp(\ell_U \rightarrow^t p_c)$, is defined as

$$supp(\ell_U \rightarrow^t p_c) = |\{s \in \mathcal{S}_C \mid p_c \sqsubseteq s\}|,$$

that is, the support value of p_c in $\mathcal{S}_C$. The confidence of an unexpected consequent rule, denoted as $conf(\ell_U \rightarrow^t p_c)$, is defined as

$$supp(\ell_U \rightarrow^t p_c) = \frac{|\{s \in \mathcal{S}_C \mid p_c \sqsubseteq s\}|}{|\{s \in \mathcal{D} \mid p_c \sqsubseteq s\}|},$$

that is, the fraction of the support of the sequential pattern p_c on the total number of sequences contained in $\mathcal{D}$ that support the sequence p_c.

Algorithm 18: `UnexpOccurRules` (s_α, $\mathcal{P}_A$, $\mathcal{P}_C$, $\mathcal{D}$, $conf_{min}$) : Mining unexpected occurrence rules.

Input : A class label ℓ_U, a sequential pattern set $\mathcal{P}_A$, a sequential pattern set $\mathcal{P}_C$, a sequence database $\mathcal{D}$, and a minimum confidence threshold $conf_{min}$.

Output : All unexpected occurrence rules with respect to $conf_{min}$.

```
1  R := ∅;
2  foreach p_a ∈ P_A do
3      supp := count(p_a, D);
4      if (conf := p_a.count/supp) ≥ conf_min then
5          gaps := GapDist(p_a, P_A, 0);    /* 0 indicates the gap after p_a */
6          foreach gap ∈ gaps do
7              r := AntecedentRule.Create(ℓ_U, p_a, p_a.supp, conf, gap);
8              R := R ∪ r;
9  foreach p_c ∈ P_C do
10     supp := count(p_c, D);
11     if (conf := p_c.count/supp) ≥ conf_min then
12         gaps := GapDist(p_c, P_C, 1);    /* 1 indicates the gap before p_c */
13         foreach gap ∈ gaps do
14             r := ConsequentRule.Create(ℓ_U, p_c, p_c.supp, conf, gap);
15             R := R ∪ r;
16 return R;
```

The unexpected occurrence rule mining process is listed in Algorithm 18 (`UnexpOccurRules`: Mining Unexpected Occurrence Rules).

The algorithm accepts an unexpectedness class label ℓ_U, a sequential pattern set $\mathcal{P}_A$ generated from the antecedent set $\mathcal{S}_A$ (for unexpected antecedent rules), a sequential pattern set $\mathcal{P}_C$ generated from the consequent set $\mathcal{S}_C$ (for unexpected consequent rules), a sequence database $\mathcal{D}$, and a minimum confidence threshold $conf_{min}$ as inputs and outputs all unexpected occurrence rules with respect to $conf_{min}$.

For each sequence $p_a \in \mathcal{P}_A$ and $p_c \in \mathcal{P}_C$, the algorithm verifies whether the confidence of p_a and p_c in $\mathcal{D}$ satisfies $conf_{min}$. If $conf_{min}$ is satisfied, then the algorithm first computes the distribution of the gap between the last itemset of sequence p_a and the last itemset of each antecedent sequence, and the distribution of the gap between the first itemset of each consequent sequence and the first itemset of the sequence p_c by the routine `GapDist` (listed in Algorithm 19), and then generates a new unexpected occurrence rules from p_a or p_c. Finally the algorithm outputs all generated unexpected occurrence rules with respect to each correspondence of (p_a – gap) or (p_c – gap).

Algorithm 19 shows the routine `GapDist`, which accepts a sequence s, a sequence set $\mathcal{S}$, and a boolean value dir as the inputs, and outputs all gap ranges with respect to each sequence $s' \in \mathcal{S}$ and the direction indicated by dir, where the usage of dir is shown at the lines 5 and 12 in Algorithm 18 that $dir = 0$ serves generating antecedent rules and $dir = 1$ serves generating consequent rules.

Algorithm 19: `GapDist` (s, $\mathcal{S}$, dir) : Computing gap distribution.

Input : A sequence s, a sequence set $\mathcal{S}$, and a boolean value dir.
Output : All gap ranges with respect to each sequence $s' \in \mathcal{S}$ and the direction indicated by dir.

```
1  dist := Array.Create(|max{|s'| | s' ∈ S}| − |s|);
2  foreach s' ∈ S do
3      pos := SeqMatchFirst(s, s', pair(0, |s'| − 1));
4      if dir = 0 then      /* the gap after p */
5          gap := (|s'| − 1) − pos.second;
6      else      /* the gap before p */
7          gap := pos.first;
8      dist[gap] := dist[gap] + 1;
9  ranges := PairSet.Create();
10 ran := FindRange(dist, 0);
11 while ran.first ≠ −1 do
12     ranges.add(ran);
13     ran := FindRange(dist, ran.second + 1);
14 ran := FindBestRange(dist);
15 while ran.first ≠ −1 do
16     ranges.add(ran);
17 return ranges;
```

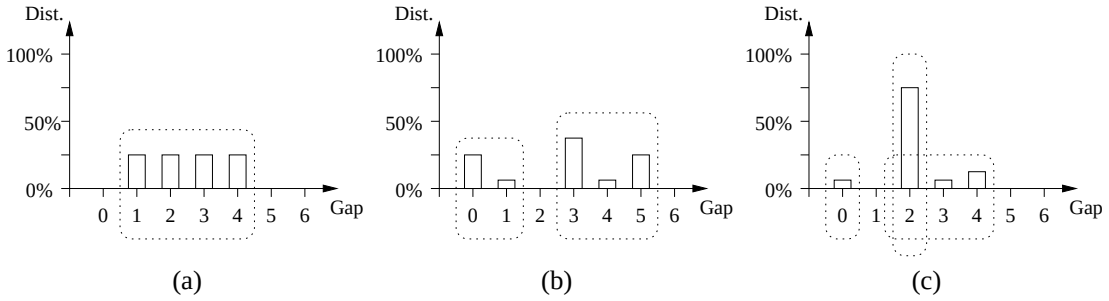


Figure 7.4: Different distributions of gaps.

The algorithm first counts the gap for each sequence, then finds all ranges of gap, and finally finds the best range of gap. For instance, Figure 7.4 shows several different cases in finding gap ranges, where in Figure 7.4(a), 1 range [1..4] can be generated; in Figure 7.4(b), 2 ranges [0..1] and [3..5] can be generated; in Figure 7.4(c), 2 ranges [0..0] and [2..4] can be generated, and in additional, a best range [2..2] is generated with respect to a default difference value 50%, that is, the number of gaps contained in the best range is at least 50% higher than the neighbors, and the number of gaps contained in this range is maximal.

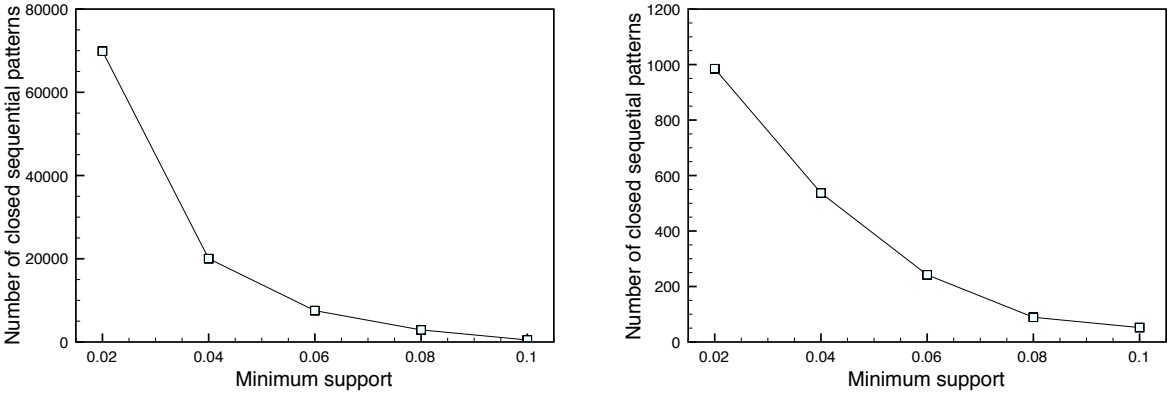
7.4 Experiments

To evaluate of our approach, we performed a serial of experiments on three Web access log files, including a very large log file of a BSD UNIX online discussion forum (labeled as `UNIX`, 11GB) and a large log file of a customer support forum of an online game provider (labeled as `GAME`, 1GB).

All log files are converted to session sequence databases. The global parameter sets of session sequences are fixed to contain hour periods (from 00h to 23h), day periods (from Monday to Sunday). Due to privacy issues, user location information is not included in global parameter sets, other sensible information, such as session ID or login name in HTTP query fields, is also removed.

We first discover closed sequential patterns in data set with the *CloSpan* approach, in order to obtain general characteristics of the data sets, which are used for defining sequence rules contained in beliefs. Figure 7.5 shows the correlations between the number of extracted closed sequential patterns with respect to minimum support value. We generate sequence rules from for each data set for describing Web usage, including 10 sequence association rules and 10 sequence implication rules from the most frequent closed sequential patterns, where one semantic constraint is specified for each rule. From each data set, we create one semantic hierarchy of concepts according to topics, and then we generate 10 generalized sequence association rules and 10 generalized sequence

implication rules from the most frequent sequential patterns with respect to semantic hierarchies. Therefore, for each data set, totally 20 beliefs with/without hierarchies are used for extracting unexpected implication rules on Web usage.

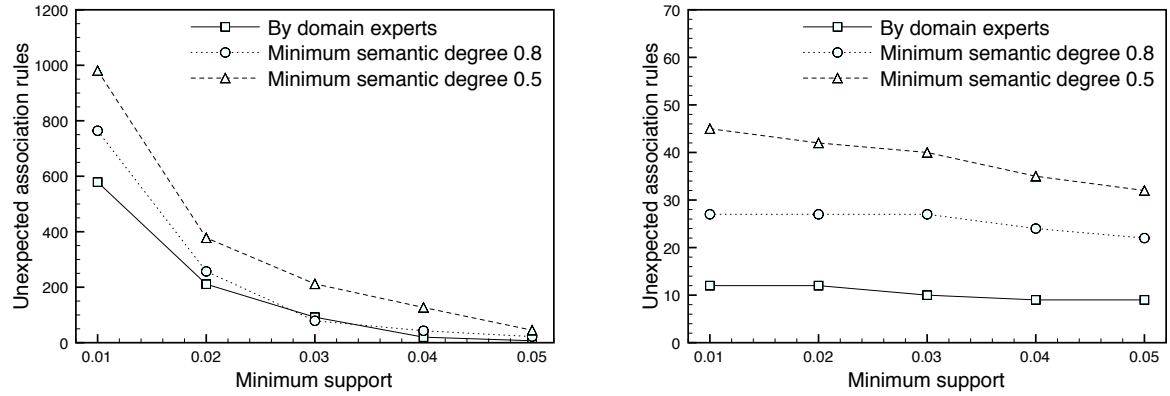


(a) Data set UNIX. (b) Data set GAME.

Figure 7.5: Closed sequential patterns.

For example, the following belief corresponds to an expected browsing order of the GAME data set, where t=2 corresponds to the access of the discussion topic "user terms" and t=5 corresponds to "user manual", such that the site designer wishes that users may read the agreement terms before reading the manual of the forum:

$$\left\{ \langle(/)\rangle \rightarrow^{[0..5]} \langle(\texttt{t=2})(\texttt{t=5})\rangle \right\} \wedge \left\{ \langle(\texttt{t=2})(\texttt{t=5})\rangle \not\simeq_{sem} \langle(\texttt{t=5})(\texttt{t=2})\rangle \right\}.$$

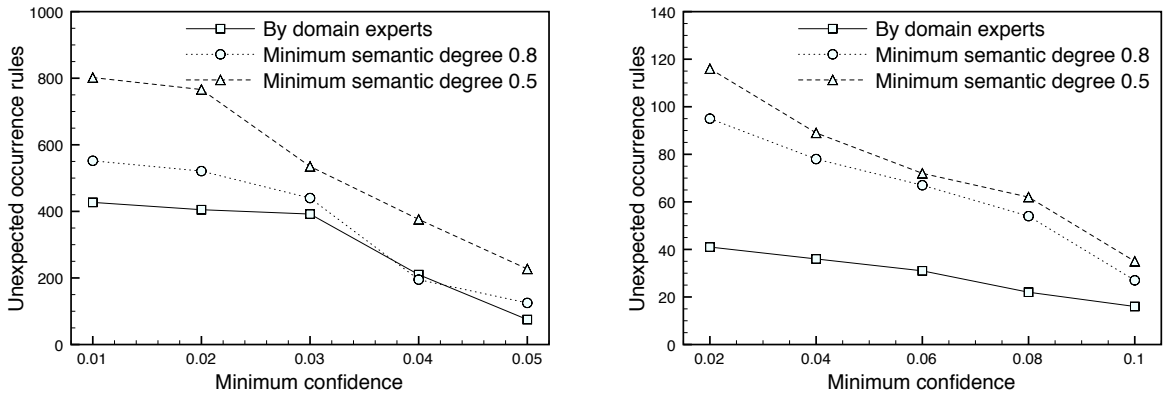


(a) Data set UNIX. (b) Data set GAME.

Figure 7.6: Unexpected association rules.

The number of discovered unexpected association rules in each data set with respect to domain expert defined beliefs and generalized sequence rules is shown in Figure 7.6. In order to not generate too much unexpected rules $\mathcal{I} \rightarrow s_u$, the minimum support for extracting s_u is fixed to 0.5, which produces less sequential patterns. In the figures, the minimum support is used for

extracting $\mathcal{I}$. In the experiments, we compared the number of unexpected rules extracted from domain experts specified beliefs and from hierarchies where minimum semantic degree (include semantic relatedness degree and semantic contradiction degree) is fixed to 0.8 and 0.5.



(a) Data set UNIX. (b) Data set GAME.

Figure 7.7: Unexpected occurrence rules.

The number of discovered unexpected occurrence rules in each data set with respect to domain expert defined beliefs and generalized sequence rules is shown in Figure 7.7, where minimum semantic degree is also fixed to 0.8 and 0.5. In the data set UNIX, the results are similar between domain experts specified beliefs and hierarchies-enabled beliefs with minimum semantic degree 0.8. In the data set GAME, the results are similar between hierarchies-enabled beliefs with minimum semantic degree 0.8 and 0.5.

To illustrated discovered unexpected rules, for example, in the data set GAME, we wish that users of the forum 3 (discussions on the game noted as G3) view several threads in this forum, that is, the sequence implication rule $\langle(\texttt{f=3})\rangle \rightarrow^* \langle(\texttt{f=3})\rangle$, however we know from prior knowledge on playing games that the players of G3 may be not interested in the game discussed in the forum 6 (the game noted as G6), thus the semantic contradiction $\langle(\texttt{f=3})\rangle \not\approx_{sem} \langle(\texttt{f=6})\rangle$ can be defined, that is, the belief

$$b = \left\{ \langle(\texttt{f=3})\rangle \rightarrow^* \langle(\texttt{f=3})\rangle \right\} \wedge \left\{ \langle(\texttt{f=3})\rangle \not\approx_{sem} \langle(\texttt{f=6})\rangle \right\}.$$

With this belief, we discovered the unexpected class rule $\texttt{BETA}_b \rightarrow \langle(\texttt{f=7})\rangle$ where forum 7 discusses a game noted as G7, and the unexpected association rule $\langle(\texttt{f=3})\rangle \rightarrow \langle(\texttt{Sunday})\rangle$, which can be further combined as an sequence association rule $\langle(\texttt{Sunday})\rangle \rightarrow \langle(\texttt{f=7})\rangle$ in the post analysis process. Moreover, from expertise knowledge given by the game provider, we know that the players of G7 seldom play the game noted as G5, then the following belief can be defined:

$$\left\{ \langle(\texttt{Sunday})(\texttt{f=3})\rangle \rightarrow^* \langle(\texttt{f=7})\rangle \right\} \wedge \left\{ \langle(\texttt{f=7})\rangle \not\approx_{sem} \langle(\texttt{f=5})\rangle \right\}.$$

7.5 Discussion

In this chapter, we first studied the problem of self-validation of discovered unexpected sequences with the notions of unexpected sequential patterns with respect to the unexpected feature set and the host sequence set. We then proposed the notions of unexpected implication rules, which include unexpected class rule, unexpected association rules and unexpected occurrence rules, where unexpected association rules further include local and global unexpected association rules, and unexpected occurrence rules include antecedent and consequent rules. Finally we applied the notions of unexpected implication rules to discover unexpected Web usage.

Validation is an important problem in the discovery of unexpected sequences, as well as in machine learning research, where the cross-validation methods are essential. For instance, the notion of cross-validation is addressed in many text classification oriented data mining tasks to examine the effectiveness of text classifiers [Seb02].

However, cross-validation of unexpected sequences is relatively difficult to apply. Indeed, the main issue of performing cross-validation to unexpected sequences is that we cannot measure the distribution of unexpected sequences in a sequence database. When we randomly regroup a sequence database to two subset of sequences, the unexpected sequences stated by some belief may be contained only in one subset, so that in this case it will be impossible to find such unexpected sequences in another subset and the cross-validation will be invalid.

There are many interesting issues related to the validation of unexpected sequences in particular cases. In the next chapter, we will study the validation of unexpected sequences in the context of text classification problems. A approach is derived from MUSE to find unexpected information (including *opposite sentiments* in the context of sentiment classification and *unexpected sentences* in general, which express the information unexpected to what the document expresses) contained in text documents, where we apply the cross-validation methods to evaluate the discovered exception phrases by text classification tools.

Chapter 8

Validation of Unexpected Sentences in Text Documents

In previous chapters, we have developed a general framework for the discovery of unexpected sequences in databases, and proposed a self-validation process for evaluating discovered unexpected sequences. In this chapter, we consider the discovery and cross-validation of unexpected sentences within the context of text classification, where the unexpected sentences are considered in terms of the notions of sentiment classification and general text classification.

A part of the work presented in this chapter has been published in the *19th International Conference on Database and Expert Systems Applications (DEXA 2008)* and in the *Actes des Rencontres Francophones sur la Logique Floue et ses Applications (LFA 2008)*; and has been accepted to be published in the *Intelligent Data Analysis Journal (IDA)*.

8.1 Introduction

Sentiment classification received much attention in analyzing personal opinion orientations contained in user generated contents, such as customer reviews, online forums, discussion groups, blogs, etc., where the orientations are often classified into *positive* or *negative* polarities. Although the sentiment classification of personal opinions is determinative, the sentences expressing the sentiment opposite to the overall orientation expressed by the document can be interesting for many purposes.

For instance, a customer review that has been classified into positive opinions about a product may contain some sentences pointing out the weakness or faults of the product, or a review classified as negative may nevertheless recognizes the good points of the product.

Indeed, sentiment classification can be regarded as a sub-category of *text classification* tasks. The task of text classification is generally performed by the *classifier* that describes how a document is classified (a systematic survey can be found in [Seb02]). The great practical importance of

text classification techniques has been addressed since the last 10 years, which covers the massive volume of user generated content available in the Web, electronic mail, customer reviews, medical records, digital publications, and so on.

On the other hand, many examples can be addressed for illustrating the sentences unexpected to document category as well as the opposite sentiments in the context of sentiment classification. For instance, in an online news group about politics events, discussions on politics are expected to be posted, however the contents on football can be considered as unexpected. One reason to study the unexpected sentences contained in text documents is that according to the principle of classifiers, unexpected sentences may decrease the accuracy of classification results. Further, another reason is that unexpected contents can be interesting because they are unexpected (see Chapter 2).

The task of text classification is performed by the *classifier* that describes how a document is classified, of which a systematic survey can be found in [Seb02]. In recent, the effectiveness of text classification techniques [NMTM00, LSST+02, Seb02] has been addressed in a large range of application domains including categorizing Web pages [YLW04, YHC04, MLK06, SZH+06, SWL06], learning customer reviews [Tur02, DLP03, PE05], and detecting sentiment orientations [PHW02, BCR07]. However, there are many cases where input text documents contain unexpected contents that are opposite to the thematic categories of document, which make the classes of documents difficult to be precisely defined and measured, and even decrease the accuracy of classification results.

Many examples can be addressed in text classification for illustrating the unexpected contents, just like in an online news group about politics events, discussions on politics are expected to be posted, however the contents on football can be considered as unexpected. Another example is that in sentiment classification, which can be viewed as an instance of text classification where the document classes are considered as "positive" and "negative" sentiment orientations instead of topics, and the phrases containing a sentiment opposite to document orientation are unexpected. Therefore, in this context, given a text document under a predefined class, an unexpected sentence is a phrase contained in the document such that it is semantically opposite or unrelated to the class.

We study the unexpected sentences in the context of sentiment classification that classifies documents with respect to the overall sentiment expressed.

Sentiment classification is often used to determine sentiment orientation in user reviews [PLV02, Tur02, DLP03, HL04, PL04, PE05]. The extraction of sentiment orientations is closely connected with Natural Language Processing (NLP) problems, where the positive or negative connotation are annotated by the subjective terms at the document level [Tur02, DLP03, PL04]. In order to obtain precise results, many approaches also consider sentence level sentiment orientation, such

as [DLP03, YH03, HL04, WWH04, WWH05, JL06a, JL06b, WWH06].

In recent literatures, many various methods have been proposed to improve the accuracy and efficiency of sentiment classification, where machine learning based text classification methods are often applied. For instance, Pang et al. [PLV02] studied the sentiment classification problems with Naive Bayes, maximum entropy, and support vector machines; Turney [Tur02] proposed an unsupervised learning algorithm for classifying reviews with sentiment orientations. The effectiveness of text classification techniques has been addressed in a large range of application domains including categorizing Web pages [YHC04, MLK06, SZH+06, SWL06], learning customer reviews [Tur02, DLP03], and detecting sentiment polarities [PLV02, BCR07].

Actually, sentiment classification are performed by considering the adjectives contained in sentences[HM97, Tur01, ES07]. We use WordNet [Cog, Fel98] for determining the antonyms of adjectives required for constructing the belief base, which has been used in many NLP and opinion mining approaches. For instance, in the proposal of [KMMdR04], WordNet is also applied for detecting the semantic orientation of adjectives. In this paper, we extendedly propose a general model of document class descriptors, which considers the adjectives, adverbs, nouns, verbs and negation identifiers.

In this chapter, we propose a general framework for determining unexpected sentences in the context of text classification. In this framework, we use sequential pattern based *class descriptors* for generalizing the characteristics of a document with respect to its class, and *unexpected class patterns* are therefore generated from the *semantic oppositions* of the elements contained in class descriptors. An *unexpected sentence* can be stated in a text document by examining whether it contains any unexpected class patterns. The semantic oppositions of a class descriptor can be determined in various manners. For sentiment classification tasks, the semantic oppositions of sentiment can be directly determined by finding antonyms of adjectives and adverbs. Therefore, in the experiments, we present the extraction of unexpected sentences for sentiment classification within the proposed framework.

Moreover, the effectiveness of subjective approaches to discover unexpected patterns or rules are often judged with respect to domain expertise [PT98, Spi99, LLP07, LLRP08]. In this chapter, we propose a cross-validation process for measuring the overall influence of unexpected sentences by using text classification methods. The experimental evaluation shows that the accuracy of classification are increased without unexpected sentences. Our experiments also show that in the results obtained from the same document sets with randomly-removed sentences, the accuracy are decreased. The comparison between the classification accuracy of the documents containing only randomly-selected sentences and containing only unexpected sentences shows that the latter is significantly lower.

The rest of this chapter is organized as follows. In Section 8.2, we formalize the text documents

in free format to a common sequence data mining model with the part-of-speech tags in order to take the grammar attribute of each word into account. In Section 8.3, we present the notion of contextual opposite sentiments in sentiment classification. We first propose the contextual models for describing sentiment orientation, and then we propose the discovery of contextual opposite sentiments. In Section 8.4, we propose sequential pattern based class descriptors, from which unexpected class patterns can be generated and applied for discovering unexpected sentences. Section 8.5 shows our experimental results on the discovery and evaluation of unexpected sentences. Finally, we discuss in Section 8.6.

8.2 Part-of-Speech Tagged Data Model

We are considering free-format text documents, where each document consists of an ordered list of sentences, and each sentence consists of an ordered list of words.

In this chapter, we treat each word contained in the text as a *lemma* associated with its *part-of-speech* (PoS) tag, including *noun* (*n.*), *verb* (*v.*), *adjective* (*adj.*), *adverb* (*adv.*), etc., denoted as $(lemma|pos)$. For example, the word "are" contained in the text is depicted by $(be|v.)$, where *be* is the lemma of "are" and *verb* is the part-of-speech tag of "be". Without loss of generality, we use the wild-card $*$ and simplified part-of-speech tag for denoting a generalized word. For instance, $(*|adj.)$ denotes an adjective; $(*|adv.)$ denotes an adverb, $(*|n.)$ denotes a noun, $(*|v.)$ denotes a verb, and so on. Further, the *negation identifiers* are denoted as $(*|neg.)$, including *not*, *'nt*, *no* and *never*. We use a generalization relation between two words having the same part-of-speech tag, which is a partial relation $\preceq$ such that: let $w_1 = (lemma_1|pos)$ and $w_2 = (lemma_2|pos)$, we have that $w_1 \preceq w_2$ implies $lemma_1 = lemma_2$ or $lemma_2 = *$. For example, we have that $(be|v.) \preceq (*|v.)$ but $(be|verb) \npreceq (film|n.)$.

A *vocabulary*, denoted as $V = \{w_1, w_2, \ldots, w_n\}$, is a collection of a limited number of distinct words. A *phrase* is an ordered list of words, denoted as $s = w_1 w_2 \ldots w_k$. A phrase can also contain generalized words. For example, $(film|n.)(be|v.)(good|adj.)$ is a phrase; $(film|n.)(*|v.)(good|adj.)$ and $(*|n.)(be|v.)(*|adj.)$ are two phrases with generalized words. The *length* of a phrase s is the number of words (including generalized words) contained in this phrase, denoted as $|s|$. One single word can be viewed as a phrase with length 1. An *empty phrase* is denoted as $\emptyset$, we have that $s = \emptyset \iff |s| = 0$. A phrase with the length k is called a *k-phrase*.

In the context of mining sequence patterns [AS95], a word is an *item* and a phrase is a *sequence*. Given two phrases $s = w_1 w_2 \ldots w_m$ and $s' = w'_1 w'_2 \ldots w'_n$, if there exist integers $1 \leq i_1 < i_2 < \ldots < i_m \leq n$ such that $w_i \preceq w'_{j_i}$ for all w_i, then s is a *sub-phrase* of s', denoted as $s \sqsubseteq s'$. If we have that $s \sqsubseteq s'$, we say that s *is contained in* s', or s' *sup-*

ports s. If a phrase s is not contained in any other phrases, then we say that the phrase s is *maximal*. For example, $(film|n.)(good|adj.)$ is contained in $(film|n.)(be|v.)(good|adj.)$ but not in $(be|v.)(good|adj.)(film|n.)$; $(film|n.)(good|adj.)$ is contained in $(*|n.)(*|adj.)$ but not in $(*|v.)(*|adj.)$. The *concatenation* of phrases is denoted as $s_1 s_2 s_3 \ldots$; the *subtraction* of two phrases s_1 and s_2 is denoted $s_1 \backslash s_2$ if and only if $s_2 \sqsubseteq s_1$. For instance, let $s_1 = w_a w_b w_c w_b w_d$ and $s_2 = w_b w_d$, we have that $s_2 \sqsubseteq s_1$ and $s_1 \setminus s_2 = w_a w_c w_b$: the first occurrence of s_2 in s_1 is removed.

A *sentence* is a *grammatical complete* phrase, denoted as $s^{\#}$. A *document* is a set of sentences, denoted as D. We do not concentrate on the order in the context of sequence data mining though a document is logically an ordered list of sentences. Moreover, in the same context, a document can be generalized to be a set of phrases. In this paper, the determination of sentence is addressed by one of the following symbols "; . ? !" in the text. Given a document D, the *support* or *frequency* of a phrase s, denoted as $supp(s, D)$, is the total number of sentences $s^{\#} \in D$ that support s. Given a user specified threshold of support called *minimum support*, denoted as $supp_{min}$, a phrase is *frequent* if $supp(s, D) \geq supp_{min}$.

Text 1 The actors in this film are all also very good. This is a good film without big budget sets. Very good sound, picture, and seats. □

Example 41 Text 1 contains 3 sentences. If we consider only the nouns, verbs, and adjectives contained in the text, Text 1 corresponds to a document D with 3 phrases:

$$s_1 = (actor|n.)(film|n.)(be|v.)(good|adj.);$$
$$s_2 = (be|v.)(good|adj.)(film|n.)(big|adj.)(budget|n.)(set|n.);$$
$$s_2 = (good|adj.)(sound|n.)(picture|n.)(seat|n.).$$

Given minimum support threshold $supp_{min} = 0.5$, we have maximal frequent phrases $p_1 = (be|v.)(good|adj.)$ and $p_2 = (film|n.)$ where $supp(p_1, D) = 0.667$ and $supp(p_2, D) = 1$. □

The part-of-speech tagged data model is purposed for the ease of data mining tasks. It is not difficult to see that the computational process cannot handle the support of the word "actor" in the sentence "the actors in this film are all also very good" without proper preprocess of the model of text. On the other hand, importing part-of-speech tags into the data model makes it possible to focus only on specified parts of text, such as for building text class descriptors by adjectives and nouns.

8.3 Contextual Opposite Sentiments

In this section, we present a belief-driven approach to discover contextual opposite sentiments in classified free format text reviews.

8.3.1 Contextual Models of Sentiment Orientation

We represent sentiment orientations as rule-format on phrases, that is, $s_\alpha \rightarrow s_\beta$, where s_α and s_β are two phrases; given a phrase s, if we have that $s_\alpha \cdot s_\beta \sqsubseteq s$, then we say that the phrase s *supports* the rule r, denoted as $s \models r$. We therefore propose a belief system for formalizing the opposite sentiments expressed in classified reviews.

A *belief* on phrases, denoted as b, consists of a rule $s_\alpha \rightarrow s_\beta$ and a semantic opposition $s_\beta \not\sim s_\gamma$, where the phrase s_γ is semantically opposite to the phrase s_β. We note such a belief as $b = [s_\alpha; s_\beta; s_\gamma]$, which constrains that if the phrase s_α appears in a phrase s, that is, $s_\alpha \sqsubseteq s$, then the phrase s_β should appear in s after s_β, and the phrase s_γ should not appear in s after s_α, that is,

$$[s_\alpha; s_\beta; s_\gamma] \iff (s_\alpha \sqsubseteq s) \Rightarrow (s_\alpha \cdot s_\beta \sqsubseteq s) \wedge (s_\alpha \cdot s_\gamma \not\sqsubseteq s).$$

A phrase s that supports a belief b is *expected*, denoted as $s \models b$; that violates a belief b is *unexpected*, denoted as $s \not\models b$. Given a belief $b = [s_\alpha; s_\beta; s_\gamma]$ and a phrase s such that $s_\alpha \sqsubseteq s$, the unexpectedness is considered as

$$(s_\alpha \cdot s_\beta \not\sqsubseteq s) \wedge (s_\alpha \cdot s_\gamma \sqsubseteq s) \Rightarrow (s \not\models b),$$

that is, if s_α appears in s, however s_β does not appear in s and s_γ appears in s later, then the phrase s is unexpected. Notice that this definition is more strict than the unexpected sequences defined in Chapter 4.

Example 42 Given a belief $b = [(be|v.); (good|adj.); (bad|adj.)]$ and two phrases

$$s_1 = (be|v.)(a|*)(good|adj.)(film|n.),$$
$$s_2 = (be|v.)(bad|adj.)(actor|n.),$$

we have that $s_1 \models b$ and $s_2 \not\models b$. □

Let M^+ be the *positive sentiment* and M^- be the *negative sentiment*, a sentiment $M \in \{M^+, M^-\}$ can be expressed in documents (denoted as $\mathcal{D} \models M$), sentences (denoted as $S \models M$), phrases (denoted as $s \models M$) or words (denoted as $w \models M$). In addition, we denote the negation of a sentiment M as $\overline{M}$, so that we have that $\overline{M^+} = M^-$ and $\overline{M^-} = M^+$. The negation is taken into account in other text-mining applications (for instance for synonym/antonym extraction process [Tur01]).

Property 2 *Given a sentiment $M \in \{M^+, M^-\}$, if a document $\mathcal{D} \models M$, then there exists at least one sentence $S \in \mathcal{D}$ such that $S \models M$; if a sentence $S \models M$, then there exists at least one word $w \sqsubseteq S$ such that $w \models M$ or at least one phrase $(*|neg.) \cdot w \sqsubseteq S$ (or $w \cdot (*|neg.) \sqsubseteq S$) such that $w \models \overline{M}$.*

Currently we focus on the sentiments expressed by the sentences that contain adjectives and nouns/verbs, such as "*this is a good film*". The sentiment expressed by sentences like "*this film is well produced*" is currently not considered in our approach. Note that we extract basic words relations without the use of syntactic analysis tools [ST93] to avoid the silence in the data (i.e. syntactic relations not extracted by the natural language systems).

With the adoption of rules and beliefs, we can extract the contextual information from reviews by finding the most frequent phrases that consist of at adjectives and nouns/verbs by sequential pattern mining algorithms, where the frequent nouns and verbs reflect topic of reviews, and the sentence-level sentiment orientations are expressed by frequent adjectives.

Contextual Model	Sentiment Rule	Belief Patterns
ADJ.-N. model	$(*\|adj.) \rightarrow (*\|n.)$	$[(\overline{*}\|adj.); \emptyset; (*\|n.)]$ $[(*\|neg.)(*\|adj.); \emptyset; (*\|n.)]$
N.-ADJ. model	$(*\|n.) \rightarrow (*\|adj.)$	$[(*\|n.); (*\|adj.); (\overline{*}\|adj.)]$ $[(*\|n.); (*\|adj.); (*\|neg.)(*\|adj.)]$
V.-ADJ. model	$(*\|v.) \rightarrow (*\|adj.)$	$[(*\|v.); (*\|adj.); (\overline{*}\|adj.)]$ $[(*\|v.); (*\|adj.); (*\|neg.)(*\|adj.)]$ $[(*\|v.)(*\|neg.); (\overline{*}\|adj.); (*\|adj.)]$
ADJ.-V. model	$(*\|adj.) \rightarrow (*\|v.)$	$[(*\|adj.); (*\|v.); (*\|v.)(*\|neg.)]$
NEG.-ADJ.-N. model	$(*\|neg.)(*\|adj.) \rightarrow (*\|n.)$	$[(*\|neg.)(\overline{*}\|adj.); \emptyset; (*\|n.)]$
N.-NEG.-ADJ. model	$(*\|n.)(*\|neg.) \rightarrow (*\|adj.)$	$[(*\|n.)(*\|neg.); (*\|adj.); (\overline{*}\|adj.)]$
V.-NEG.-ADJ. model	$(*\|v.)(*\|neg.) \rightarrow (*\|adj.)$	$[(*\|v.)(*\|neg.); (*\|adj.); (\overline{*}\|adj.)]$
ADJ.-V.-NEG. model	$(*\|adj.) \rightarrow (*\|v.)(*\|neg.)$	$[(\overline{*}\|adj.); \emptyset; (*\|v.)(*\|neg.)]$

Table 8.1: Contextual models of sentiment orientation.

We propose a set of contextual models for constructing the belief base of opinion orientations within the context of review topic, listed in Table 8.1, where the word $(\overline{*}|adj.)$ stands for each antonym of the word $(*|adj.)$. Given a review, each sentence violating a belief generated from one of the belief patterns listed in Table 8.1 stands for an opposite sentiment.

8.3.2 Discovery of Contextual Opposite Sentiments

A training-discovering process is considered in the discovery of contextual opposite sentiments: given a topic context, first a sequential pattern mining approach is applied to find *contextual*

patterns a set of classified training reviews with respect to a set the contextual models listed in Table 8.1, in order to generate the belief patterns; then, from discovered belief patterns, a belief base is constructed to represent the sentiment orientation by using a dictionary of antonyms[1] of the adjectives contained in the contextual models.

Let $\mathcal{V}$ be a set of adjectives expressing the sentiment M, we denote $\overline{\mathcal{V}}$ the set that contains the antonym(s) of each word contained in $\mathcal{V}$. Thus, for each $(*|adj.) \in \mathcal{V}$, we have that $(*|adj.) \models M$ and $(\overline{*}|adj.) \in \overline{\mathcal{V}}$.

Given a *training document* $\mathcal{D}_L$ such that for each sentence $S \in \mathcal{D}_L$, there exist at least one adjective $(*|adj.) \in \mathcal{V}$ or there exist $(*|neg.)$ and at least one adjective $(*|adj.) \in \overline{\mathcal{V}}$. In order to construct the belief base of contextual models, we first apply a sequential pattern mining algorithm for discovering all maximal frequent phrases from $\mathcal{D}_L$ with respect to a minimum support threshold, denoted as $\mathcal{D}_F$. For each phrase $s \in \mathcal{D}_F$, if s supports a contextual model listed in Table 8.1 with the listing-order, then a set of beliefs can be generated from s corresponding to the belief pattern(s) of each contextual model. A belief base $\mathcal{B}_M$ can therefore be constructed with respect to the topic of reviews.

Positive Sentiment Rules	Negative Sentiment Rules
$(be\|V) \rightarrow (good\|J)$	$(bad\|J) \rightarrow (guy\|N)$
$(good\|J) \rightarrow (film\|N)$	$(bad\|J) \rightarrow (be\|V)$
$(good\|J) \rightarrow (be\|V)$	$(bad\|J) \rightarrow (movie\|N)$
$(good\|J) \rightarrow (performance\|N)$	$(bad\|J) \rightarrow (film\|N)$
$(good\|J) \rightarrow (movie\|N)$	$(bad\|J) \rightarrow (thing\|N)$
$(good\|J) \rightarrow (friend\|N)$	$(bad\|J) \rightarrow (year\|N)$
$(great\|J) \rightarrow (film\|N)$	$(bad\|J) \rightarrow (time\|N)$
$(great\|J) \rightarrow (be\|V)$	$(bad\|J) \rightarrow (dialogue\|N)$
$(special\|J) \rightarrow (be\|V)$	$(stupid\|J) \rightarrow (be\|V)$
$(special\|J) \rightarrow (effect\|N)$	$(poor\|J) \rightarrow (be\|V)$

Table 8.2: The top-10 most frequent sentiment rules.

Example 43 Given a phrase $s = (this)(be|v.)(a)(good|adj.)(film|n.)$, where the part-of-speech tags of "this" and "a" are ignored because they are not in the contextual models, we have that s supports the ADJ.-N. and V.-ADJ. models, and the sentiment rules are $(good|adj.) \rightarrow (film|n.)$ and $(be|v.) \rightarrow (good|j.)$. We have that the priority of ADJ.-N. model is higher than V.-ADJ. model (according to the order listed in Table 8.1), so that the rule $(good|adj.) \rightarrow (film|n.)$ is used

[1]The antonym dictionary is based on the WordNet project, which can be found at http://wordnet.princeton.edu/.

for generating beliefs. Let $(bad|adj.)$ be the antonym of $(good|adj.)$, we have two beliefs generated: $[(bad|adj.); \emptyset; (film|n.)]$ and $[(*|neg.)(good|adj.); \emptyset; (film|n.)]$. □

For instance, Table 8.2 lists the top-10 most frequent sentiment rules discovered from the movie review data[2] introduced in [PL04] with respect to the contextual models and belief patterns listed in Table 8.1.

Belief Base of Positive Sentiment	Belief Base of Negative Sentiment
$[(be|v.); (good|adj.); (bad|adj.)]$	$[(not|neg.)(bad|adj.); \emptyset; (guy|n.)]$
$[(be|v.); (good|adj.); (not|neg.)(good|adj.)]$	$[(n't|neg.)(bad|adj.); \emptyset; (guy|n.)]$
$[(be|v.); (good|adj.); (n't|neg.)(good|adj.)]$	$[(bad|adj.); (be|v.); (be|V)(not|neg.)]$
$[(bad|adj.); \emptyset; (film|n.)]$	$[(bad|adj.); (be|v.); (be|V)(n't|neg.)]$
$[(not|neg.)(good|adj.); \emptyset; (film|n.)]$	$[(good|adj.); \emptyset; (film|n.)]$
$[(n't|neg.)(good|adj.); \emptyset; (film|n.)]$	$[(not|neg.)(bad|adj.); \emptyset; (film|n.)]$
.....	.....

Table 8.3: The belief base for discovering opposite sentiments.

A belief base on sentiment orientation can therefore be generated from the discovered sentiment rules, where the antonym dictionaries for constructing the belief bases are given by WordNet. Table 8.3 lists a set of sample beliefs generated from the discovered sentiment rules listed in Table 8.2.

Given a classified review $\mathcal{D}_M$ and a belief base $\mathcal{B}_M$ corresponding to the sentiment orientation M, the procedure of extracting unexpected sentences can be briefly described as follows. For each sentence $S \in \mathcal{D}_M$ and for each belief $b \in \mathcal{B}_M$ such that $b = [s_\alpha; s_\beta; s_\gamma]$, s_α is first matched for improving the performance; if $s_\alpha \sqsubseteq S$, and then if $s_\alpha \cdot s_\beta \not\sqsubseteq S$ and $s_\alpha \cdot s_\gamma \sqsubseteq S$, then S is an unexpected sentence expressing the contextual opposite sentiment $\overline{M}$.

8.4 Unexpected Sentences

In this section, we propose a sequential pattern based class descriptors, from which unexpected class patterns can be generated and applied for discovering unexpected sentences.

8.4.1 Class Descriptors

In this section, we propose a sequential pattern based class descriptors within the context of text classification.

[2] http://www.cs.cornell.edu/People/pabo/movie-review-data/

In [Seb02], Sebastiani generalized the text classification problem as the task of assigning a Boolean value to each pair $\langle D_j, C_i \rangle \in \mathcal{D} \times \mathcal{C}$ where $\mathcal{D}$ is a domain of documents and $\mathcal{C} = \{C_1, C_2, \ldots, C_{|\mathcal{C}|}\}$ is a set of predefined classes. A value $True$ assigned to $\langle D_j, C_i \rangle$ indicates a decision to classify D_j under C_i, while a value of $False$ indicates a decision not to classify D_j under C_i. A *target function* $\Phi : \mathcal{D} \times \mathcal{C} \rightarrow \{True, False\}$ is called the *classifier*. In practical, a *classification status value* (or *categorization status value*) function $\Omega_i : \mathcal{D} \rightarrow [0, 1]$ is considered in the classifier for class $C_i \in \mathcal{C}$. A *threshold* τ_i is therefore defined such that for a document D_j, $\Omega_i(D_j) \geq \tau_i$ is interpreted as $True$ while $\Omega_i(D_j) < \tau_i$ is interpreted as $False$. Most of existing text classifiers can be generalized to this model.

Given a document D and a sentence $s^{\#} \notin D$ such that for a class C_i we have $\Omega_i(D \cup s^{\#}) > \Omega_i(D)$, then there exists a set S of phrases such that for each phrase $s \in S$ we have $s \sqsubseteq s^{\#}$ and $\Omega_i(D \cup s) > \Omega_i(D)$. We say that such a phrase s *supports* the class C_i, denoted as $s \models C_i$, and this phrase s is called a *key phrase* of C_i. Further, given a key phrase s of a class C_i, there exists a set W of words such that for each word $w \in W$ we have $w \subseteq s$ and $\Omega_i(D \cup w) > \Omega_i(D)$. We say that such a word w *supports* the class C_i, denoted as $w \models C_i$, and this word w is called a *key word* of C_i. In additional, we denote $s \not\models C_i$ (respectively for $w \not\models C_i$) that the phrase s *is not* a key phrase of the class C_i, in this meaning, $s \not\models C_i$ does not imply but include the case $\Omega(D \cup s) < \Omega(D)$.

With a threshold τ_i for a class C_i and a document D, let $D \models C_i$ denote that $\Omega_i(D) \geq \tau_i$ is interpreted as $True$ for the classification task, then we have the following property.

Property 3 *Given a class C_i and a document D, if $D \models C_i$, then there exists a subset $D' \subseteq D$ such that for each sentence $s^{\#} \in D'$ we have $s^{\#} \models C_i$, and for each sentence $s^{\#} \in (D \setminus D')$ we have $s^{\#} \not\models C_i$.*

Notice that for Property 3, the set $(D \setminus D')$ can be empty. In this case, each sentence $s^{\#} \in D$ supports the class C_i. According to the definitions of sentence and phrase in Section 8.2, we have the following lemma.

Lemma 2 *Given a class C_i and a document $D \models C_i$, the document D contains a set S of maximal phrases such that if $s \in S$ then $s \models C_i$.*

Considering a document domain $\mathcal{D}$ and a set $\Pi = \{D_1, D_2, \ldots, D_{|\Pi|}\} \in \mathcal{D}$ of documents pre-classified under a class C_i, that is, for each $D_j \in \Pi$ we have $D_j \models C_i$, let $\Gamma = \{s^{\#} \in D \mid D \in \Pi\}$ be the sentences contained in all documents and S_i^+ be the set of all maximal key phrases contained in Γ. For any two phrases $s_m, s_n \in S_i^+$ we have $s_m \not\sqsubseteq s_n$, $s_m, s_n \subseteq \Gamma$ and $s_m, s_n \models C_i$. The set S_i^+ is called the *predictive phrase set* of the class C_i.

Definition 41 (Class descriptor) *Let S_i^+ be the predictive phrase set of a given document class C_i, the class descriptor of the class C_i is a set P_i^+ of phrases such that: (1) each phrase $s \in P_i^+$*

consists only of words with PoS tag in $\{adj., adv., n., v., neg.\}$*; (2) for each phrase* $s \in P_i^+$*, there exists a phrase* $s' \in S_i^+$ *such that* $s \sqsubseteq s'$*; (3) for any two phrases* $s_m, s_n \in P_i^+$*, we have* $s_m \not\sqsubseteq s_n$*. Each phrase* $s \in P_i^+$ *is a class pattern.*

However, given a large set Π of pre-classified documents under the class C_i, it is practically difficult to construct the predictive phrase set S_i^+ containing all predictive phrases in each document. On the other hand, association rules [AIS93] and sequential patterns [AS95] have been used for building text classifiers [LG94, LHM98, LHP01, AZ02, JLT06], where word frequency is a key factor for computing classification status value. In this chapter, we consider the frequent phrases contained in the pre-classified document set as an approximation of the predictive phrase set, so that the class descriptor can further be approximately built from the discovered frequent phrases by filtering the adjectives, adverbs, nouns, verbs, and negation identifiers.

Definition 42 (Approximate class descriptor) *Let* Π *be a set of text document under the class* C_i*, an approximate class descriptor of the document set* Π *for the class* C_i*, denoted as* $\Delta_i(\Pi)$*, is the set of maximal frequent phrases consisting of adjectives, adverbs, nouns, verbs, and negation identifiers in the total text* Γ *of the document set* Π*, with respect to a user defined minimum support threshold.*

In the rest of the chapter, unless explicitly noticed, we consider the *approximate class descriptor* as the *class descriptor*.

A class descriptor consists of a set of maximal frequent phrases where each phrase is a class pattern, which can be modeled by its structure. A class pattern $p = w_1 w_2 \ldots w_n$ is an ordered list of words, which can also be denoted as $p = (lemma_1|pos_1)(lemma_2|pos_2)\ldots(lemma_n|pos_n)$. The structure pos_1-pos_2-...-pos_n is called a *class pattern model*. If a class pattern consists of k words, then we say that it is a k-phrase class pattern, corresponding to a k-phrase class pattern model. For instance, the 2-phrase class pattern $(famous|adj.)(actor|n.)$ corresponds to the class pattern model "ADJ.-N." (we present the PoS tags as upper case in a class pattern model).

Text 2 The other actors deliver good performances as well. □

Example 44 Assume that the sentence listed in Text 2 is contained in one of a large set Π of text documents, which can be represented as

$$s = (other|adj.)(actor|n.)(deliver|v.)(good|adj.)(performance|n.)(well|adv.),$$

where $p_1 = (actor|n.)(good|adj.)$ and $p_2 = (good|adj.)(performance|n.)$ are two 2-phrases, and $p_3 = (actor|n.)(deliver|v.)(good|adj.)$ is a 3-phrase contained in s. Let Γ be the total text of all documents in Π. Given a user specified minimum support threshold min_supp, if we have $\sigma(p_1, \Gamma) \geq min\_supp$, $\sigma(p_2, \Gamma) \geq min\_supp$, and $\sigma(p_3, \Gamma) \geq min\_supp$, then p_1, p_2, and p_3 are

3 class patterns of the class C_i, respectively corresponding to class pattern models "N.-ADJ.", "ADJ.-N.", and "N.-V.-ADJ.". □

8.4.2 Discovery and Cross-Validation of Unexpected Sentences

Given a class pattern p of a text document set Π under a class C_i, we consider the pattern p as a *belief* on the class C_i. Hence, an *unexpected class pattern* is a phrase that semantically contradicts the class pattern p.

We first propose the notion of *ϕ-opposition pattern* of class patterns. For facilitating the following descriptions, let us consider the *semantic opposition relation* $w_1 = \neg w_2$ between two words, which denotes that the word w_1 semantically contradicts the word w_2. We have $w_1 = \neg w_2 \iff w_2 = \neg w_1$. The semantic opposition between words can be determined by finding the antonyms or computing the semantic relatedness of concepts. Currently, the computation of semantic relatedness between concepts have been addressed by various methods [BH06, PS07, GM08, ZMG08].

Definition 43 (ϕ-opposition pattern) *Let $p = w_1 w_2 \dots w_k$ and $p' = w'_1 w'_2 \dots w'_k$ be two k-phrase class pattern. If p' has a sub-phrase $\eta = w_1^\eta w_2^\eta \dots w_\phi^\eta$ and p has a sub-phrase $\varphi = w_1^\varphi w_2^\varphi \dots w_\phi^\varphi$, where $\phi \leq k$, such that $p' \setminus \eta = p \setminus \varphi$ and for any $1 \leq i \leq \phi$ we have $w_i^\eta = \neg w_i^\varphi$, then the phrase p' is a ϕ-opposition pattern of p.*

Given a class pattern p, there exist various ϕ-opposition patterns of p. For example, by detecting the antonyms of words, for a 2-phrase class pattern $(be|v.)(good|adj.)$, $(be|v.)(bad|adj.)$ is one of its 1-opposition pattern since $(good|adj.) = \neg(bad|adj.)$; for a 3-phrase class pattern $(be|v.)(good|adj.)(man|n.)$, according to $(good|adj.) = \neg(bad|adj.)$ and $(man|n.) = \neg(woman|n.)$, two 1-opposition patterns and one 2-opposition pattern can be generated.

Notice that the negation is not token into account with the notion of ϕ-opposition pattern, however it is considered as a general word. For example, $(*|neg.)(bad|adj.)$ is generated as a 1-opposition pattern of the class pattern $(*|neg.)(good|adj.)$.

To take into consideration the negation of sentences, the notion of *ϕ-negation pattern* is proposed as follows.

Definition 44 (ϕ-negation pattern of p) *Let $p = w_1 w_2 \dots w_k$ be a k-phrase class pattern and $p' = w'_1 w'_2 \dots w'_{k'}$ be a k'-phrase class pattern where $p \sqsubseteq p'$ and $k' = k + \phi$ ($\phi > 0$). If $w \in (p' \setminus p)$ implies $w = (*|neg.)$, then the phrase p' is a ϕ-negation pattern of p.*

Not difficult to see, the generation of ϕ-negation patterns depends on the value of ϕ. For example, from the class pattern $(be|v.)(good|adj.)$, a 2-negation pattern $(*|neg.)(be|v.)(*|neg.)(good|adj.)$ can be generated.

Unexpected class patterns can be therefore generated from ϕ-opposition and ϕ-negation patterns of a class pattern. In this paper, we focus on 1-opposition and 1-negation patterns for generating unexpected class patterns.

Given a class descriptor P_i^+ of a text document set Π under a class C_i, let S_i^- be the ensemble of all ϕ-opposition and ϕ-negation patterns of each class pattern $p \in P_i^+$. The set $P_i^- = S_i^- \setminus P_i^+$ is called an *unexpected class descriptor* of the class C_i. Each phrase contained in P_i^- is an *unexpected class pattern*. If a sentence contains an unexpected class pattern, then this sentence is an *unexpected sentence*.

The extraction of unexpected sentences can be performed with respect to the framework of (1) extracting class descriptors from pre-classified documents; (2) building unexpected class descriptors from ϕ-opposition patterns and ϕ-negation patterns of each class descriptor; (3) extracting unexpected sentences that contain unexpected class descriptors.

Not difficult to see, this framework can be performed to extract unexpected sentences with respect to general text classification problems if the unexpected class descriptors can be built.

To evaluate the unexpected sentences extracted from predefined classes of documents, we propose a four-step validation process:

1. The test on the classification of original documents, which shows the accuracy of each class of documents, denoted as $\alpha(D)$;

2. The test on the classification of the documents with randomly-removed n sentences (n is the average number of unexpected sentences per document) in each document, which shows the accuracy of disturbed documents, denoted as $\alpha(D \setminus R)$;

3. The test on the classification of the documents without unexpected sentences, which shows the accuracy of cleaned documents, denoted as $\alpha(D \setminus U)$;

4. The test on the classification of the documents only consists in unexpected sentences, which shows the accuracy of unexpectedness, denoted as $\alpha(U)$.

With comparing to the accuracy of original documents $\alpha(D)$, let the change of accuracy of the documents with randomly-removed sentences be $\delta_R = \alpha(D \setminus R) - \alpha(D)$ and let the change of accuracy of the documents without unexpected sentences be $\delta_U = \alpha(D \setminus U) - \alpha(D)$. According to the principe of text classifiers, we have the following property if the removed unexpected sentences are really unexpected to the document class.

Property 4 *(1)* $\delta_U > 0$*; (2)* $\delta_U \geq \delta_R$*; (3)* $\delta_R \leq 0$ *is expected.*

Therefore, if the results of the cross-validation of document classification shows that the changes of accuracies correspond to the hypothesis on discovered unexpected sentences as proposed in Prop-

erty 4, the we can say that the unexpected sentences contained in discovered unexpected sentences are valid, because the elimination of such sentences increases the accuracy of the classification task.

8.5 Experiments

In this section, we present our experimental evaluation on the unexpected sentences in free format text documents within the context of sentiment classification, where the unexpected class descriptors are built from antonyms of word (determined by WordNet, including adjectives and adverbs) contained in class descriptors.

The data set concerned in our experiments is the movie review data from [PL04], which consists of pre-classified 1,000 positive-sentiment and 1,000 negative-sentiment text reviews. Thus, we consider "positive" and "negative" as two document classes in our experiments, and the goal is to discover unexpected sentences against the two classes and to validate discovered unexpected sentences.

Discovery of Unexpected Sentences.

All documents are initially tagged by the TreeTagger [Ins] toolkit introduced in [Sch94] to identify the PoS tag [San90] of each word. In order to reduce the redundancy in sequence-represented documents, we only consider the words that constitute the class descriptors including the adjectives, adverbs, verbs, nouns, and the negation identifiers. All words associated with concerned tags are converted to PoS tagged sentences with respect to the order appeared in the documents, and all other words are ignored.

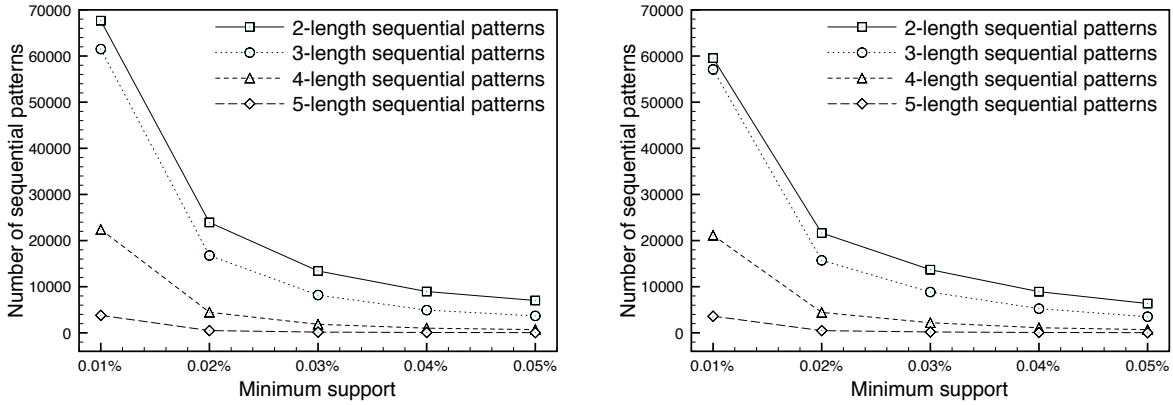
Class	Documents	Sentences	Distinct Words	Average Length
Positive	1,000	37,833	28,777	23.8956
Negative	1,000	36,186	27,224	22.2015

Table 8.4: Total number of sentences and distinct words, with average sentence length.

The total corpus contained in the data set consists of 1,492,681 words corresponding to 7.6 Megabytes. Table 8.4 lists each class of 1,000 documents of the movie review data set in sequence format. A dictionary totally containing 39,655 entries of item:word mapping is built for converting the sequences back into text for next steps.

The discovery of class descriptors is addressed as a training process with the same corpus. For each class, positive or negative in our experiments, all 1,000 sequence-represented documents are combined into one large sequence database, and then we perform *closed sequential pattern* mining algorithm *CloSpan* [YHA03] to find class patterns describing the document class.

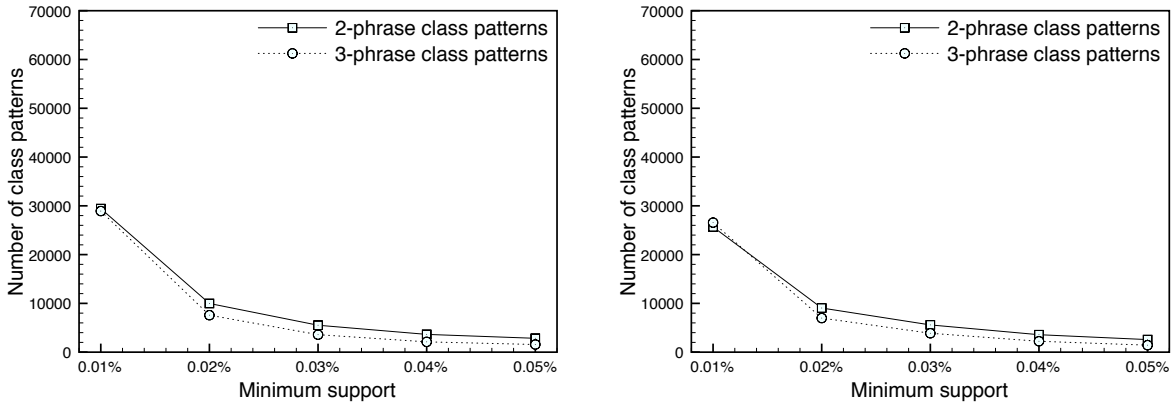
Figure 8.1 shows the number of the discovered sequential patterns with different sequence length. According to the figure, the numbers of 4-length and 5-length sequential patterns strongly deceases when the minimum support value increases, for instance, with $min\_supp = 0.05\%$, the numbers of 2-, 3-, 4-, and 5-length sequential patterns of the class "positive" are respectively 7013, 3677, 705, and 46. Therefore, in order to obtain signifiant results, we find the class patterns limited to 2- and 3-length sequential patterns for next steps of our experiments.



(a) The class "positive reviews". (b) The class "negative reviews".

Figure 8.1: Number of discovered sequential patterns with different sequence length.

We extract the sequential patterns consisting of the adjectives, adverbs, nouns, verbs, and negation identifiers as the class descriptor. Figure 8.2 shows the total numbers of 2-phrase and 3-phrase class patterns that contain at least and at most one adjective or/and adverb, since the adjectives and adverbs are essential in sentiment classification.



(a) The class "positive reviews". (b) The class "negative reviews".

Figure 8.2: Number of 2-phrase and 3-phrase class patterns.

The appearance of discovered 2-phrase class pattern models are listed in Table 8.5, ordered by the alphabet of models and $(*|neg.)$ with respect to different minimum support values. In order to save paper size, we only list the models corresponding to the $conf_{min}$ values 0.01%, 0.03%, and 0.05%.

Class Pattern Models	P-0.01%	N-0.01%	P-0.03%	N-0.03%	P-0.05%	N-0.05%
ADJ.-ADV.	1089	892	134	134	34	32
ADJ.-N.	4049	3109	566	517	257	206
ADJ.-V.	2813	2474	581	558	321	276
ADV.-ADJ.	1654	1314	219	221	83	76
ADV.-N.	3348	3014	452	469	209	169
ADV.-V.	3084	2954	728	781	394	390
N.-ADJ.	2571	2045	292	286	127	100
N.-ADV.	2929	2729	438	478	194	189
V.-ADJ.	3841	3367	940	901	507	448
V.-ADV.	3157	2940	846	931	498	492
NEG-ADJ.	329	314	103	90	60	49
ADJ.-NEG	254	232	70	64	38	34
NEG-ADV.	166	147	79	83	66	62
ADV.-NEG	147	138	71	71	51	52

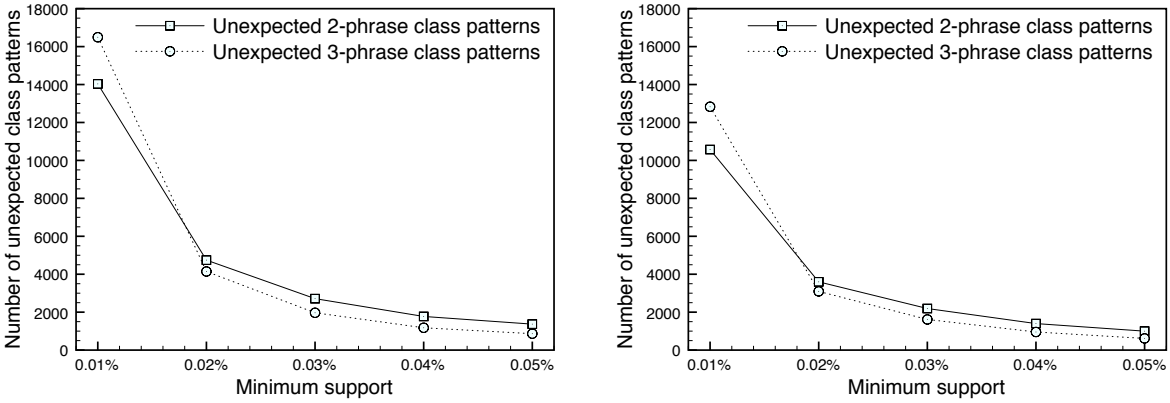
Table 8.5: 2-phrase class pattern models.

Number	Models for class “positive”	Number	Models for class “negative”
2289	V.-V.-ADV.	2343	V.-V.-ADV.
2121	V.-ADV.-V.	2106	V.-ADV.-V.
1801	V.-V.-ADJ.	1689	V.-V.-ADJ.
1691	V.-ADJ.-N.	1616	ADV.-V.-V.
1607	ADV.-V.-V.	1433	V.-ADJ.-N.
1546	V.-ADJ.-V.	1362	V.-ADJ.-V.
1340	V.-ADV.-N.	1212	N.-V.-ADV.
1276	N.-V.-ADV.	1159	V.-ADV.-N.
1045	ADJ.-V.-V.	969	ADJ.-V.-V.
946	N.-V.-ADJ.	861	V.-N.-ADV.

Table 8.6: 10 most frequent 3-phrase class pattern models.

For discovered 3-phrase class pattern models, the top-10 most frequent ones corresponding to $conf_{min} = 0.01\%$ are listed in Table 8.6.

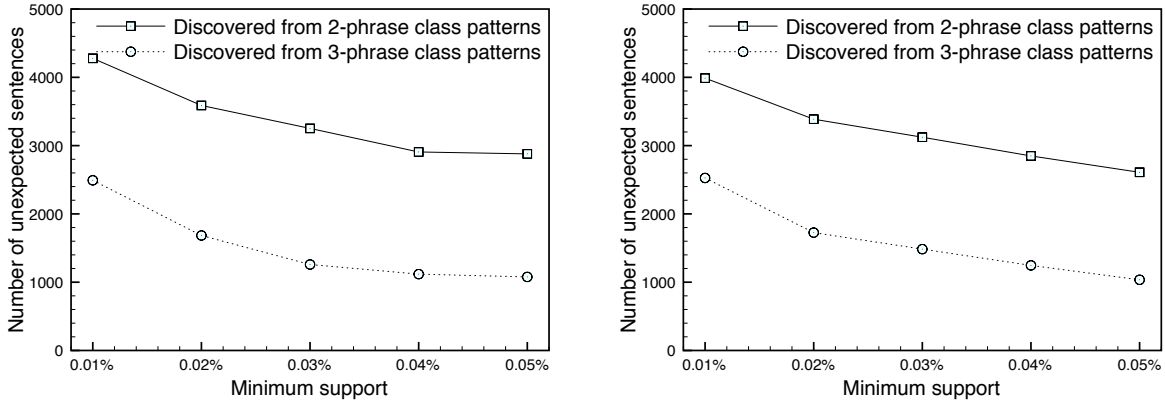
The unexpected class patterns are generated from the semantic oppositions of class patterns. In our experiments, the lexical database WordNet [Cog] is used for determining the antonyms of adjectives and adverbs for constructing semantic oppositions. For a class pattern, if there exist an adjective and an adverb together, then only the antonyms of the adjective will be considered; if the adjective and adverb have no antonym, then this class pattern will be ignored; if there exist more than one antonym, than more than one unexpected class pattern will be generated from all antonyms. The total numbers of unexpected 2-phrase and 3-phrase class patterns are shown in Figure 8.3.



(a) The class "positive reviews". (b) The class "negative reviews".

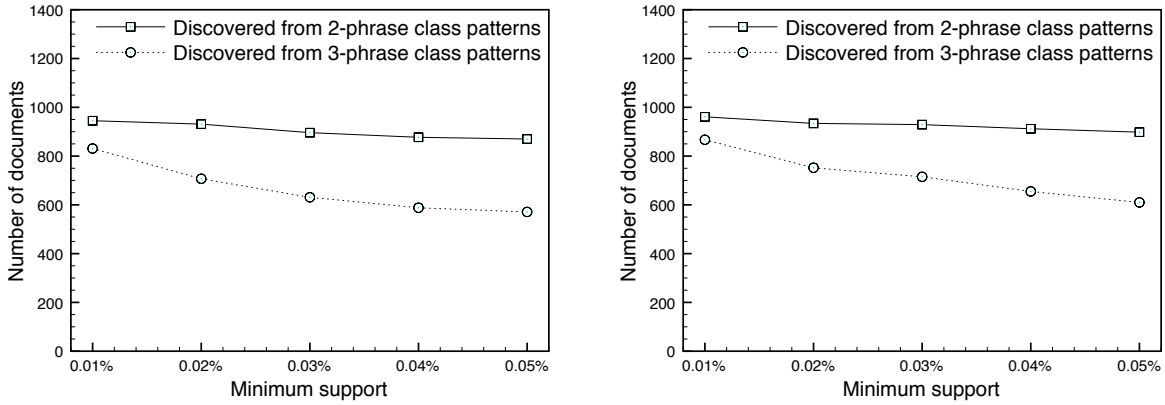
Figure 8.3: Number of 2-phrase and 3-phrase unexpected class patterns.

The total numbers of unexpected sentences determined from unexpected 2-phrase and 3-phrase class patterns are shown in Figure 8.4, and the total numbers of documents that contain unexpected sentences are shown in Figure 8.5.



(a) The class "positive reviews". (b) The class "negative reviews".

Figure 8.4: Number of unexpected sentences discovered from 2-phrase and 3-phrase unexpected class patterns.



(a) The class "positive reviews". (b) The class "negative reviews".

Figure 8.5: Number of documents that contain unexpected sentences discovered from 2-phrase and 3-phrase unexpected class patterns.

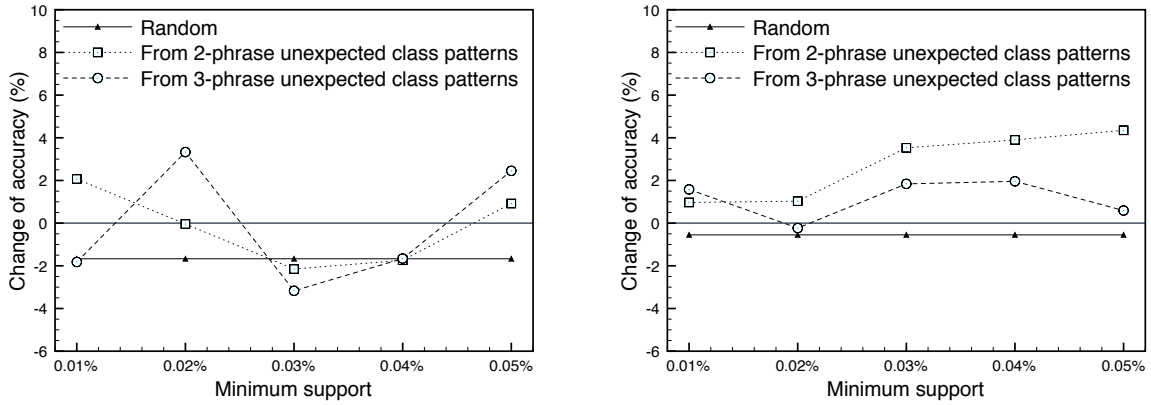
Validation of Unexpected Sentences.

The goal of the evaluation is to use the text classification method to validate the unexpectedness stated in the discovered unexpected sentences with respect to the document class. The unexpectedness is examined by the Bow toolkit [McC96] with comparing the average accuracy of text classification tasks with and without unexpected sentences.

Three methods, *k-Nearest Neighbor* (k-NN), *Naive Bayes*, and *TFIDF* are selected for testing our approach by using classification tasks. The k-NN method [YC94] based classifiers are example-based that for deciding whether a document $D \models C_i$ for a class C_i, it examines whether the k training documents most similar to D also are in C_i. The Naive Bayes based classifiers (see [Lew98]) compute the probability that a document D belongs to a class C_i by an application of Bayes' theorem, which accounts for most of the probabilistic approaches in the text classification. Nevertheless, the TFIDF (term frequency-inverse document frequency) [SB88] based classifiers compute the term frequency for deciding whether a document D belongs a class C_i, however an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the collection and increases the weight of terms that occur rarely. Briefly, in order to learn a model, a prototype vector based on the TFIDF weight of terms is computed for each class, and then the cosine value of a new document between each prototype vector is calculated to assign the relevant class.

In our experiments, two groups of tests are performed, with and without pruning most frequent words common to all documents in the two classes by selecting words with highest average mutual information with the class variable. Each test is performed with 20 trials of a randomized test-train split 40%-60%, and we take into account the final average values of accuracy. All tests are based on the unexpected sentences extracted with 2-phrase and 3-phrase unexpected class patterns obtained by different min_supp values from 0.01% to 0.05%.

The evaluation results on the change of accuracy are shown in Figure 8.6, Figure 8.7, and

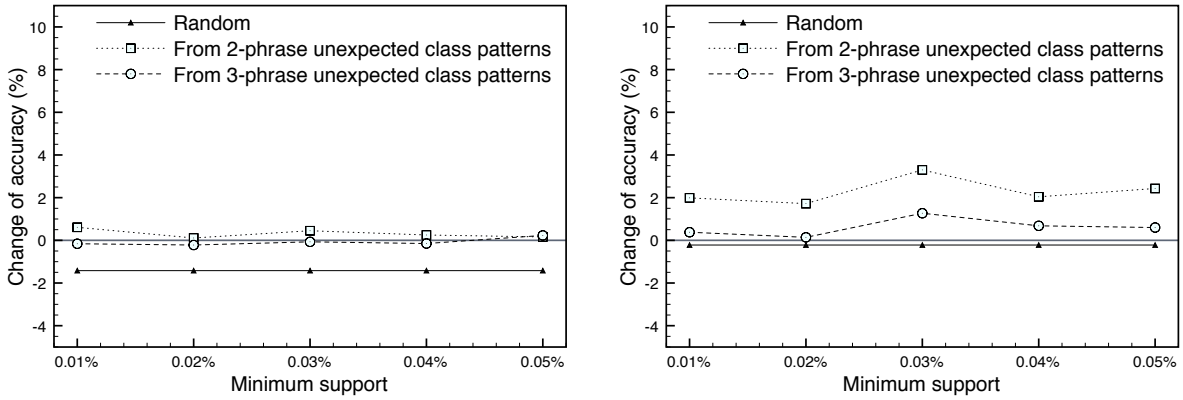


(a) Without frequent word pruning. (b) With frequent word pruning.

Figure 8.6: Change of average accuracy before and after eliminating unexpected sentences by using k-NN method.

Figure 8.8. The results are compared with removing the same number of randomly selected sentences from the documents. In each figure, the average accuracy of the original documents $\alpha(D)$ is considered as the base line "0", and the change of accuracy δ_R of the documents with randomly-removed sentences is considered as a reference line.

In the test results on the k-NN classifier shown in Figure 8.6(a), the change of accuracy is variant with respect to the *min_supp* value for extracting class patterns, however the results shown in Figure 8.6(b) well confirms Property 4. The behavior shown in Figure 8.6(a) also shows that although selecting frequent terms improves the accuracy of classification tasks, the frequent words common to all classes decrease the confidence of the accuracy of classification.

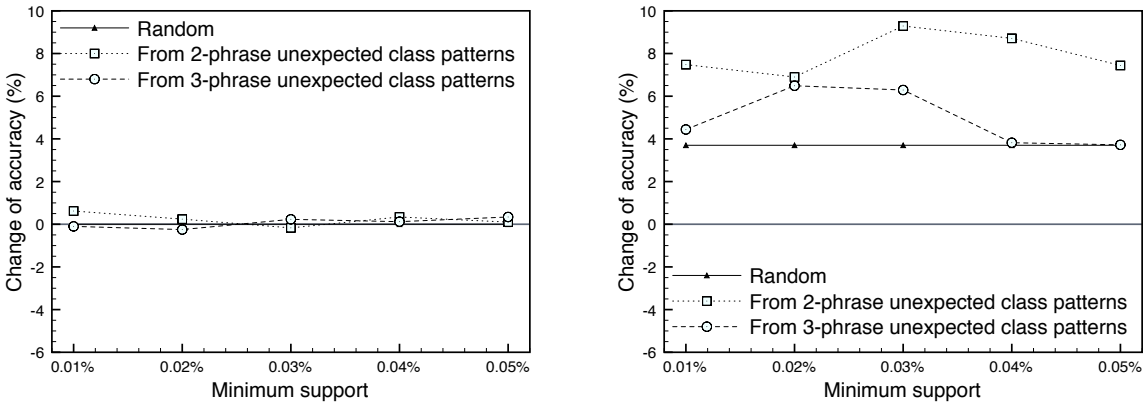


(a) Without frequent word pruning. (b) With frequent word pruning.

Figure 8.7: Change of average accuracy before and after eliminating unexpected sentences by using Naive Bayes method.

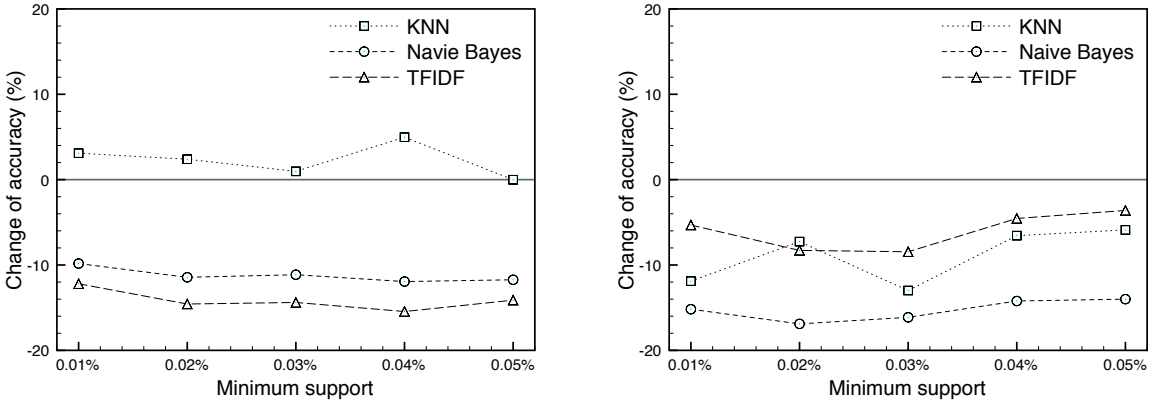
Because Naive Bayes classifiers are probability based, Figure 8.7(a) is reasonable: the unexpected class patterns contained in all eliminated unexpected sentences weakly affect the probability whether a document belongs to a class since the eliminated terms are not frequent, but randomly

selected sentences contains terms important to classify the documents. The prune of the most frequent common words enlarges the effects of unexpected sentences, thus the results shown in Figure 8.7(b) perfectly confirms Property 4.



(a) Without frequent word pruning. (b) With frequent word pruning.

Figure 8.8: Change of average accuracy before and after eliminating unexpected sentences by using TFIDF method.

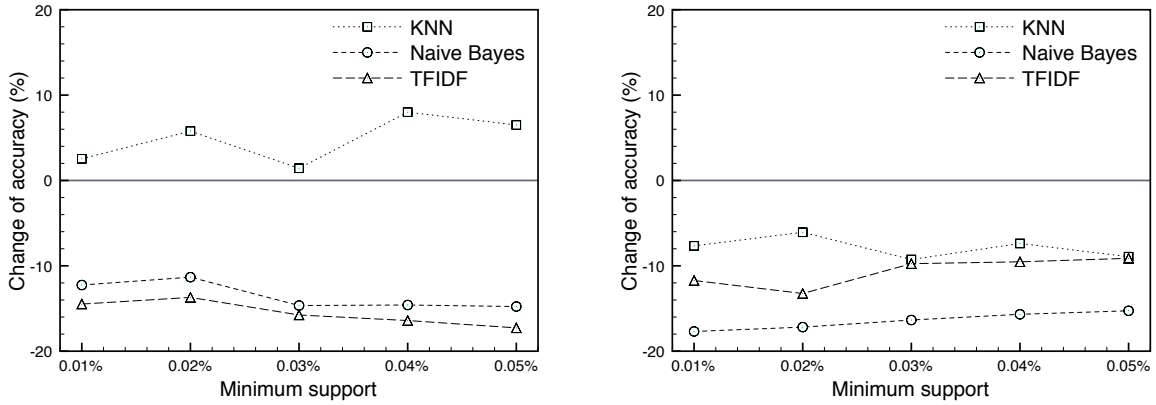


(a) Without frequent word pruning. (b) With frequent word pruning.

Figure 8.9: Change of average accuracy between original documents and the documents consisting of the unexpected sentences discovered from 2-phrase unexpected class patterns.

According to the principle of TFIDF weight, Figure 8.8(a) shows that the effect of comment frequent words in classification tasks is important, so that the elimination of limited number of sentences does not change the overall accuracy. Different from Naive Bayes classifiers, Figure 8.8(b) well confirms Property 4.(1) and Property 4.(2), however Property 4.(3) is not satisfied because the elimination of random selected sentences increases the overall accuracy of the classification.

We also test the accuracy of the classification tasks on the documents consisting of only unexpected sentences, to study the characteristics of unexpected sentences, as shown in Figure 8.9 and Figure 8.10. Not difficult to see, the unexpected sentences are difficult to be classified with



(a) Without frequent word pruning. (b) With frequent word pruning.

Figure 8.10: Change of average accuracy between original documents and the documents consisting of the unexpected sentences discovered from 3-phrase unexpected class patterns.

comparing to original documents. As discussed in previous analysis, the effect of the most frequent common words in k-NN based classifiers is strong.

8.6 Discussion

In this chapter, we studied the effects of unexpected sentences in text document classification. We first formalized text documents with part-of-speech tags, and then proposed the notion of contextual opposite sentiments. We further generalized the formalization of contextual opposite sentiments by proposing the notions of class descriptors and class patterns, from which we further proposed the notion of unexpected class patterns. A phrase containing an unexpected class pattern is therefore an unexpected sentence. In consequence, we evaluated discovered unexpected sentences by text classification, including *k-nearest neighbor* and *naive Bayes* methods. The experimental results show that the discovery of unexpected sentences is effective and the accuracy of classification can be improved by eliminating unexpected sentences in text documents.

The approach proposed in this paper considers 1-opposition and 1-negation unexpected class patterns, which limits the performance of discovering unexpected sentences, although the effectiveness has been already shown. In our future research, we will focus on the construction of complex unexpected class patterns, such as 2-opposition and 2-negation patterns.

Although our proposed approach is theoretically common for discovering any unexpected sentences with respect to the document classes, however, the generation of ϕ-opposition unexpected patterns are currently limited in determining the antonyms of words, which is suitable for adjective and adverb based document classes, for example the positive and negative orientations in sentiment classification. In order to practically porting our approach to more general cases, for

example topic-based document classes, we are interested in adopting semantic similarity based approaches (e.g., [JC97, MCS06]) or semantic hierarchies for generating ϕ-opposition unexpected patterns by determining the relatedness between concepts.

Chapter 9

Conclusions

In this chapter, we first summarize this thesis, and then discuss some perspectives on our future research directions.

9.1 Summary

In this thesis, we investigated the problems in the discovery of unexpected sequences in large databases with respect to prior domain expertise knowledge.

We proposed a belief system consisting of sequence rules and semantic contradiction between sequences, and then we proposed three forms of unexpected sequences with respect to the different forms of sequence rules. We methodically developed the framework MUSE with integrating the approaches to discover the three forms of unexpected sequences. The usefulness and effectiveness of the framework MUSE are shown with the experiments on real Web server access records data and synthetic data.

We developed the approaches TAUFU and UFR to extend the framework MUSE by adopting fuzzy set theory for describing sequence occurrence. We studied the fuzzy unexpectedness in sequence occurrence as *tau-fuzzy* unexpected sequences with developing the approach TAUFU. We then proposed the notion of unexpected fuzzy recurrence behavior in sequence data with respect to the belief system consists of fuzzy recurrence rules, and the approach UFR is developed to discover unexpected fuzzy recurrences. The approaches TAUFU and UFR are evaluated with the experiments on real Web server access records data.

We proposed a generalization of the framework MUSE with respect to the concept hierarchies on the taxonomy of data. To reduce the complexities in constructing the belief system, we propose the notions of generalized unexpected sequences. We also proposed the notion of soft belief and develop the approach SOFTMUSE to discover soft unexpected sequences in hierarchical data, where the belief system consists only of generalized sequence rules and a concept hierarchy. Unexpected sequences are therefore stated by determining the relatedness and contradiction with computing

the semantic similarity between generalized sequence rules on the concept hierarchy.

We proposed the notions of unexpected sequential patterns and unexpected implication rules, in order to evaluate the discovered unexpected sequences by using a *self-validation* process. We also proposed three forms of *unexpected implication rules*, include *unexpected class rule*, *unexpected association rule*, and *unexpected occurrence rule*, to study what is associated with the unexpectedness, what implies the unexpectedness, and what the unexpectedness implies.

As a derived approach, we proposed the discovery and evaluation of unexpected sentences in free format text documents. We presented the part-of-speech data model of free format text documents, and then we presented the discovery of opposite sentiments in the context of opinion mining. We further generalized this approach to general text classification, where we proposed sequential pattern based class descriptors, and then we proposed the notion of unexpected sentences in text documents. The experimental evaluation shows that the accuracy of text classification can be improved with eliminating unexpected sentences.

9.2 Future Work

In this section, we discuss the perspectives on our future research work, which include the following directions.

9.2.1 Mining Predictive Sequence Implication Rules

The framework MUSE developed in Chapter 4 discovers multiple unexpected sequences with respect to a belief system consists of sequence rules and semantic contradictions between sequences. In Chapter 6, we further developed SOFTMUSE that discovers unexpected sequences with respect to sequence rules and concept hierarchies. Therefore, the construction of sequence rules is essential to our proposed approaches.

As discussed in Chapter 3, many existing approaches can be used for mining sequence association rules, so that we are much interested in mining predictive sequence implication rules in the form $s_\alpha \rightarrow^\tau s_\beta$, where τ is a constraint on the range of gaps between the premise and conclusion sequences s_α and s_β. However, the discovery of similar sequence rules is very limited. In [HS05], Hetland and Sætrom proposed a genetic programming [Koz92] based approach to discover sequence rules in time series, where the addressed sequence rules are very close to our notion of predictive sequence implication rules.

We are currently developing a *pattern-growth* [PHW07] based general purposed approach to discover predictive sequence implication rules in sequence databases. In this approach, we consider three interestingness measures in the mining process, including *support*, *confidence*, and *gap distribution*. The *support* of the rule is defined as the number of sequences that support the rule; the *confidence* of the rule is defined as the fraction of the number of sequences that support the

rule on the number of sequences that support the premise sequence s_α. Given a rule $s_\alpha \rightarrow^\tau s_\beta$ and a sequence database $\mathcal{D}$, the *gap distribution* is the distribution of the gaps between the premise and conclusion sequences in the mining process, which specifies the *predictability* of a rule. In Section 7.3.3 of Chapter 7, we have proposed a routine `GapDist` (Algorithm 19) to compute the gap distribution, which can be integrated into the *pattern-growth* framework.

9.2.2 Mining Unexpectedness with Fuzzy Rules

In Chapter 5, we have discussed that there is a very extended way of considering fuzzy association rules and gradual rules in discovering the unexpectedness in data. Hence, we are interested in mining more complex unexpectedness with fuzzy rules, which can be summarized as Table 9.1.

Rule	Semantic Contradiction	Unexpected Rules
if X is A, then Y is B	$A \not\approx_{sem} C$	if X is C, then Y is B
if X is A, then Y is B	$B \not\approx_{sem} D$	if X is A, then Y is D

Table 9.1: Fuzzy unexpectedness.

Unexpectedness can be addressed by considering fuzzy association rules. For instance, if "`age` is `old` → `salary` is `high`" corresponds to prior knowledge, then "`age` is `young` → `salary` is `high`" or "`age` is `old` → `salary` is `low`" can be considered as unexpected, since we have that *old* contradicts *young* and *high* contradicts *low*. The same manner can also be extended to gradual rules. For instance, if prior knowledge shows that " `age increases` → `salary increases`", then "`age increases` → `salary decreases`" is unexpected, since we have that *increase* contradicts *decrease*, etc.

Rule	Semantic Contradiction	Unexpected Rules
if X is A, then Y is B	$A \not\approx_{sem} C$	if X is C and Z is E, then Y is B
if X is A, then Y is B	$A \not\approx_{sem} C$	if X is C, then Y is B and Z is E
if X is A, then Y is B	$B \not\approx_{sem} D$	if X is A and Z is E, then Y is D
if X is A, then Y is B	$B \not\approx_{sem} D$	if X is A, then Y is D and Z is E

Table 9.2: Complex unexpectedness.

Our goal is not discover only unexpected rules, but also the correlations within unexpected rules. Table 9.2 lists more complex cases, where the correlations within unexpected rules can be measured by frequency.

9.2.3 Mining Intermediate Patterns

Let us consider a belief b consisting of a sequence association rule $s_\alpha \rightarrow s_\beta$ and the semantic contradiction $s_\beta \not\approx_{sem} s_\gamma$. From this belief, a sequence supporting the rule $s_\alpha \rightarrow s_\gamma$ is unexpected.

Considering a large sequence database $\mathcal{D}$, a subset $\mathcal{D}_b \subseteq \mathcal{D}$ can be discovered, where each sequence $s \in \mathcal{D}_b$ supports the rule $s_\alpha \rightarrow s_\gamma$. If we can find the rule $s_\alpha \wedge s_\alpha{}' \rightarrow s_\gamma$ in the sequence set $\mathcal{D}_b$ with strong support and confidence value, then we can say that the sequence $s_\alpha{}'$ is a *key sequence*, which may play an important role in the causality of the unexpectedness.

From this observation, we propose the notion of *intermediate patterns* in the context of association rule mining. Given an association rule $X \rightarrow Y$, where X and Y are two patterns (itemsets), the rule depicts that the presence of X implies the presence of Y. Different from this notion, we are interested in the case that the patterns X and Y are not present in same itemsets, however, a pattern Z can occur $X \cup Y$. We call such a rule as a *transition rule*, denoted as $X \xrightarrow{Z} Y$, where X, Y, and Z are three patterns, and the pattern Z is so called a *intermediate pattern*. Such a rule can be represented as follows:

$$(X \xrightarrow{Z} Y) \Rightarrow (X \cup \neg Z \not\rightarrow Y) \wedge (X \cup Z \rightarrow Y).$$

Intermediate patterns can be interesting to many domains. For instance, Swanson found papers that connected terms A and B are also papers that connected B and C. From that, he made connections A to C [Swa86], where one example was a connection between fish oil and migraines. Not difficult to see, in the problem of Swanson's Raynaud-Fish Oil and Migraine-Magnesium discoveries, which is also closely connected to text mining applications [GL96, Sri04, WKdJvdBV01], an intermediate pattern plays the role of the term B.

The notion of *intermediate pattern* can be push back to the context of sequence mining, with the notion of *sequence transition rule*, denoted as $s_\alpha \xrightarrow{s_\gamma} s_\beta$, that is,

$$(s_\alpha \xrightarrow{s_\gamma} s_\beta) \Rightarrow (s_\alpha \cdot \neg s_\gamma \not\rightarrow s_\beta) \wedge (s_\alpha \cdot s_\beta \rightarrow s_\gamma).$$

A sequence transition rule $s_\alpha \xrightarrow{s_\gamma} s_\beta$ depicts that if the sequence s_γ occurs after the occurrence of the sequence s_α, then the sequence s_β will occur later; otherwise, without the sequence s_γ, the sequence s_β does not occur.

To discover (sequence) transition rules and intermediate patterns/sequences can be interesting for finding new trends or new chances for business intelligence. In [Ohs06], a similar business process is introduced in terms of finding *KeyGraph* from events or states for *chance discovery*.

9.2.4 Mining Unexpected Sentences with Dependency Tree

In Chapter 8, we proposed a general framework for discovery unexpected sentences in text documents, where the semantic contradictions are determined from antonyms of words. Obvious, we cannot indicate antonyms for most nouns and verbs, hence, even though we have proposed a general framework, the application is limited to sentiment classification.

We are interested in extending our approach with two methods. On one hand, according to the framework SoftMuse, concept hierarchies can be used for determining semantic contradictions, so

that building concept hierarchies from text (e.g., [CHS05, SC99]) can help to improve our approach to fit the requirement of general text classification problems. On the other hand, dependency parsing of text is well studied in recent (e.g., [Att06, ACC07, AC07, AD09, CA07]), where the dependency tree constructed from text contains semantic information of the text. Therefore, considering the dependency tree constructed from training documents, the unexpected sentences can be extracted with respect to the unexpected tree patterns discovered from dependency tree constructed from test documents.

9.2.5 Applications

Research serves applications. We are also interested in pushing our approaches proposed in the framework of this thesis to real world applications.

In this thesis, we have performed a lot of experiments on Web access log data, which show the effectiveness of our approaches in (but not limited to) the context of Web usage mining. Hence, our perspectives include the development of a complete toolkit WebUser for improving Web sites by analyzing frequent and unexpected Web usage.

Moreover, many data mining approaches consider only binary-valued data model, such as frequent patterns, association rules, sequential patterns, and sequence rules. Hence, our perspectives also include porting our approaches to real world relational database. In this application, we will first construct sequence rules, then generate concept hierarchies from relational data [JHP04], and finally, unexpected behaviors including unexpected sequences, unexpected sequential patterns, and unexpected implication rules can be discovered.

9.3 Final Thoughts

The rule *speed increases* $\rightarrow$ *mass increases* is unexpected to classical laws of physics. Unexpectedness might predicts new knowledge. Knowledge based knowledge discovery is interesting.

Knowledge is like a round, the inside is known and the outside is unknown: the more known, the more unknown.

Figure 9.1: Final thoughts.

Bibliography

[AC07] Giuseppe Attardi and Massimiliano Ciaramita. Tree revision learning for dependency parsing. In *HLT-NAACL*, pages 388–395, 2007.

[ACC07] Giuseppe Attardi, Atanas Chanev, and Massimiliano Ciaramita. Multilingual dependency parsing and domain adaptation using DeSR. In *EMNLP-CoNLL*, pages 1112–1118, 2007.

[AD09] Giuseppe Attardi and Felice Dell'Orletta. Reverse revision and linear tree combination for dependency parsing. In *HLT-NAACL*, pages 261–264, 2009.

[AFGY02] Jay Ayres, Jason Flannick, Johannes Gehrke, and Tomi Yiu. Sequential PAttern Mining using a bitmap representation. In *KDD*, pages 429–435, 2002.

[AIS93] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, pages 207–216, 1993.

[AP02] Fabrizio Angiulli and Clara Pizzuti. Fast outlier detection in high dimensional spaces. In *PKDD*, pages 15–26, 2002.

[AP05] Fabrizio Angiulli and Clara Pizzuti. Outlier mining in large high-dimensional data sets. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):203–215, 2005.

[AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.

[AS95] Rakesh Agrawal and Ramakrishnan Srikant. Mining sequential patterns. In *ICDE*, pages 3–14, 1995.

[Att06] Giuseppe Attardi. Experiments with a multilanguage non-projective dependency parser. In *CoNLL*, pages 166–170, 2006.

[AZ02] Maria-Luiza Antonie and Osmar R. Zaiane. Text document categorization by term association. In *ICDM*, pages 19–26, 2002.

[BCR07] Farah Benamara, Carmine Cesarano, and Diego Reforgiato. Sentiment analysis: Adjectives and adverbs are better than adjectives alone. In *ICWSM*, 2007.

[BCS+07] Fernando Berzal, Juan C. Cubero, Daniel Sánchez, María Amparo Vila Miranda, and José-María Serrano. An alternative approach to discover gradual dependencies. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 15(559-570), 2007.

[BGGB05] Julien Blanchard, Fabrice Guillet, Régis Gras, and Henri Briand. Using information-theoretic measures to assess association rule interestingness. In *ICDM*, pages 66–73, 2005.

[BH06] Alexander Budanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

[BM98] Alex G. Büchner and Maurice D. Mulvenna. Discovering internet marketing intelligence through online analytical Web usage mining. *SIGMOD Record*, 27(4):54–61, 1998.

[BMUT97] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In *SIGMOD*, pages 255–264, 1997.

[BS03] Stephen D. Bay and Mark Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *KDD*, 2003.

[BT97] Gideon Berger and Alexander Tuzhilin. Discovering unexpected patterns in temporal data using temporal logic. In *Temporal Databases: Research and Practice*, pages 281–309. Springer, 1997.

[CA97] Keith C. C. Chan and Wai-Ho Au. Mining fuzzy association rules. In *CIKM*, pages 209–215, 1997.

[CA07] Massimiliano Ciaramita and Giuseppe Attardi. Dependency parsing with second-order feature maps and annotated semantic information. In *IWPT*, pages 133–143, 2007.

[Ca09] Ding-Ying Chiu and Yi-Hung Wu and. Efficient frequent sequence mining by a dynamic strategy switching algorithm. *The VLDB Journal*, 18(1):303–327, 2009.

[CCH02] Yen-Liang Chen, Shih-Sheng Chen, and Ping-Yu Hsu. Mining hybrid sequential patterns and sequential rules. *Information Systems*, 27(5):345–362, 2002.

[CG07] Toon Calders and Bart Goethals. Non-derivable itemset mining. *Data Mining and Knowledge Discovery*, 14(1):171–206, 2007.

[CH06] Yen-Liang Chen and Tony Cheng Kui Huang. A new approach for discovering fuzzy quantitative sequential patterns in sequence databases. *Fuzzy Sets and Systems*, 157(12):1641–1661, 2006.

[CHS05] Philipp Cimiano, Andreas Hotho, and Steffen Staab. Learning concept hierarchies from text corpora using formal concept analysis. *Journal of Artificial Intelligence Research*, 24:305–339, 2005.

[Cog] Cognitive Science Laboratory, Princeton University. WordNet: A lexical database for the english language. http://wordnet.princeton.edu/.

[CTCH01] Ruey-Shun Chen, Gwo-Hshiung Tzeng, C. C. Chen, and Yi-Chung Hu. Discovery of fuzzy sequential patterns for fuzzy partitions in quantitative attributes. In *AICCSA*, pages 144–150, 2001.

[CW08] Yen-Liang Chen and Cheng-Hsiung Weng. Mining association rules from imprecise ordinal data. *Fuzzy Sets and Systems*, 159(4):460–474, 2008.

[dAdSRJ03] Sandra de Amo and Ary dos Santos Rocha Jr. Mining generalized sequential patterns using genetic programming. In *IC-AI*, pages 451–456, 2003.

[dAF05] Sandra de Amo and Daniel A. Furtado. First-order temporal pattern mining with regular expression constraints. In *SBBD*, pages 280–294, 2005.

[DJLT08] Lisa Di-Jorio, Anne Laurent, and Maguelonne Teisseire. Fast extraction of gradual association rules: A heuristic based method. In *CSTST*, pages 205–210, 2008.

[DL98] Guozhu Dong and Jinyan Li. Interestingness of discovered association rules in terms of neighborhood-based unexpectedness. In *PAKDD*, pages 72–86, 1998.

[DLM+98] Gautam Das, King-Ip Lin, Heikki Mannila, Gopal Renganathan, and Padhraic Smyth. Rule discovery from time series. In *KDD*, pages 16–22, 1998.

[DLP03] Kushal Dave, Steve Lawrence, and David M. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *WWW*, pages 519–528, 2003.

[DMSV03] Miguel Delgado, Nicolás Marín, Daniel Sánchez, and María-Amparo Vila. Fuzzy association rules: general model and applications. *IEEE Transactions on Fuzzy Systems*, 11(2):214–225, 2003.

[DP06] Didier Dubois and Eyke Hüllermeier Henri Prade. A systematic approach to the assessment of fuzzy association rules. *Data Mining and Knowledge Discovery*, 13(2):167–192, 2006.

[ES07] Andrea Esuli and Fabrizio Sebastiani. PageRanking WordNet synsets: An application to opinion mining. In *ACL*, pages 424–431, 2007.

[Fel98] Christiane Fellbaum. *WordNet: An electronic lexical database*. MIT Press, 1998.

[FLT07] Céline Fiot, Anne Laurent, and Maguelonne Teisseire. From crispness to fuzziness: Three algorithms for soft sequential pattern mining. *IEEE Transactions on Fuzzy Systems*, 15(6):1263–1277, 2007.

[FMLT08] Céline Fiot, Florent Masseglia, Anne Laurent, and Maguelonne Teisseire. Gradual trends in fuzzy sequential patterns. In *IPMU*, pages 456–463, 2008.

[FPSS96a] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery: An overview. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI/MIT Press, 1996.

[FPSS96b] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. Knowledge discovery and data mining: Towards a unifying framework. In *KDD*, pages 82–88, 1996.

[GH06] Liqiang Geng and Howard J. Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys*, 38(3):9, 2006.

[GKM+03] Dimitrios Gunopulos, Roni Khardon, Heikki Mannila, Sanjeev Saluja, Hannu Toivonen, and Ram Sewak Sharm. Discovering all most specific sentences. *ACM Transactions on Database Systems*, 28(2):140–174, 2003.

[GL96] Michael D. Gordon and Robert K. Lindsay. Toward discovery support systems: A replication, re-examination, and extension of Swanson's work on literature-based discovery of a connection between Raynaud's and fish oil. *Journal of the American Society for Information Science*, 47(2):116–128, 1996.

[GM08] Jorge Gracia and Eduardo Mena. Web-based measure of semantic relatedness. In *WISE*, pages 136–150, 2008.

[GRS99] Minos N. Garofalakis, Rajeev Rastogi, and Kyuseok Shim. SPIRIT: Sequential pattern mining with regular expression constraints. In *VLDB*, pages 223–234, 1999.

[HCTS03] Yi-Chung Hu, Ruey-Shun Chen, Gwo-Hshiung Tzeng, and Jia-Hourng Shieh. A fuzzy data mining algorithm for finding sequential patterns. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 11(2):173–194, 2003.

[HCXY07] Jiawei Han, Hong Cheng, Dong Xin, and Xifeng Yan. Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery*, 15(1):55–86, 2007.

[HD04] Sherri K. Harms and Jitender S. Deogun. Sequential association rule mining with time lags. *Journal of Intelligent Information Systems*, 22(1):7–22, 2004.

[HF95] Jiawei Han and Yongjian Fu. Discovery of multiple-level association rules from large databases. In *VLDB*, pages 420–431, 1995.

[HH99a] Robert J. Hilderman and Howard J. Hamilton. Heuristic measures of interestingness. In *PKDD*, pages 232–241, 1999.

[HH99b] Robert J. Hilderman and Howard J. Hamilton. Heuristics for ranking the interestigness of discovered knowledge. In *PAKDD*, pages 204–209, 1999.

[HH01] Robert J. Hilderman and Howard J. Hamilton. Evaluation of interestingness measures for ranking discovered knowledge. In *PAKDD*, pages 247–259, 2001.

[HH03] Robert J. Hilderman and Howard J. Hamilton. Measuring the interestingness of discovered knowledge: A principled approach. *Intelligent Data Analysis*, 7(4):347–382, 2003.

[HKCJ06] Yueh-Min Huang, Yen-Hung Kuo, Juei-Nan Chen, and Yu-Lin Jeng. NP-miner: A real-time recommendation algorithm by using Web usage mining. *Knowledge Based Systems*, 19(4):272–286, 2006.

[HL04] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *KDD*, pages 168–177, 2004.

[hLLk97] Jee hyong Lee and Hyung Lee-kwang. An extension of association rules using fuzzy sets. In *IFSA*, pages 399–402, 1997.

[HLSL00] Farhad Hussain, Huan Liu, Einoshin Suzuki, and Hongjun Lu. Exception rule mining with a relative interestingness measure. In *PAKDD*, pages 86–97, 2000.

[HLW03] Tzung-Pei Hong, Kuei-Ying Lin, and Shyue-Liang Wang. Fuzzy data mining for interesting generalized association rules. *Fuzzy Sets and Systems*, 138(2):255–269, 2003.

[HM97] Vasileios Hatzivassiloglou and Kathleen McKeown. Predicting the semantic orientation of adjectives. In *ACL*, pages 174–181, 1997.

[HMWG98] Jochen Hipp, Andreas Myka, Rüdiger Wirth, and Ulrich Güntzer. A new algorithm for faster mining of generalized association rules. In *PKDD*, pages 74–82, 1998.

[HS05] Magnus Lie Hetland and Pål Sætrom. Evolutionary rule mining in time series databases. *Machine Learning*, 58(2-3):107–125, 2005.

[Hül02] Eyke Hüllermeier. Association rules for expressing gradual dependencies. In *PKDD*, pages 200–211, 2002.

[HW02] Yin-Fu Huang and Chieh-Ming Wu. Mining generalized association rules using pruning techniques. In *ICDM*, pages 227–234, 2002.

[HY06] Yu Hirate and Hayato Yamana. Generalized sequential pattern mining with item intervals. *Journal of Computers*, 1(3):51–60, 2006.

[Ins] Institute for Computational Linguistics of the University of Stuttgart. TreeTagger: A language independent part-of-speech tagger. http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/.

[JC97] Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *ICCL*, 1997.

[JHP04] Mikael R. Jensen, Thomas Holmgren, and Torben Bach Pedersen. Discovering multidimensional structure in relational data. In *DaWaK*, pages 138–148, 2004.

[JKA01] Mahesh V. Joshi, Vipin Kumar, and Ramesh C. Agarwal. Evaluating boosting algorithms to classify rare classes: comparison and improvements. In *ICDM*, pages 257–264, 2001.

[JL06a] Nitin Jindal and Bing Liu. Identifying comparative sentences in text documents. In *SIGIR*, pages 244–251, 2006.

[JL06b] Nitin Jindal and Bing Liu. Mining comparative sentences and relations. In *AAAI*, pages 1331–1336, 2006.

[JLT06] Simon Jaillet, Anne Laurent, and Maguelonne Teisseire. Sequential patterns for text categorization. *Intelligent Data Analysis*, 10(3):199–214, 2006.

[JS04] Szymon Jaroszewicz and Dan A. Simovici. Interestingness of frequent itemsets using bayesian networks as background knowledge. In *KDD*, pages 178–186, 2004.

[JS05] Szymon Jaroszewicz and Tobias Scheffer. Fast discovery of unexpected patterns in data, relative to a bayesian network. In *KDD*, pages 118–127, 2005.

[JTH01] Wen Jin, Anthony K. H. Tung, and Jiawei Han. Mining top-n local outliers in large databases. In *KDD*, pages 293–298, 2001.

[KFW98] Chan Man Kuok, Ada Wai-Chee Fu, and Man Hon Wong. Mining fuzzy association rules in databases. *SIGMOD Record*, 27(1):41–46, 1998.

[KMMdR04] Jaap Kamps, Robert J. Mokken, Maarten Marx, and Maarten de Rijke. Using WordNet to measure semantic orientation of adjectives. In *LREC*, pages 1115–1118, 2004.

[KN98] Edwin M. Knorr and Raymond T. Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB*, pages 392–403, 1998.

[Koz92] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, 1992.

[KZC08] Daniel Kunkle, Donghui Zhang, and Gene Cooperman. Mining frequent generalized itemsets and generalized association rules without redundancy. *Journal of Computer Science and Technology*, 23(1):77–102, 2008.

[LCN03] Bing Liu, Chee Wee Chin, and Hwee Tou Ng. Mining topic-specific concepts and definitions on the Web. In *WWW*, 2003.

[Lew98] David D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *ECML*, pages 4–15, 1998.

[LG94] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *SIGIR*, pages 3–12, 1994.

[LH96] Bing Liu and Wynne Hsu. Post-analysis of learned rules. In *AAAI/IAAI*, pages 828–834, 1996.

[LHM98] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *KDD*, pages 121–128, 1998.

[LHML99] Bing Liu, Wynne Hsu, Lai-Fun Mun, and Hing-Yan Lee. Finding interesting patterns using user expectations. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):817–832, 1999.

[LHP01] Wenmin Li, Jiawei Han, and Jian Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *ICDM*, pages 369–376, 2001.

[LKL08] David Lo, Siau-Cheng Khoo, and Chao Liu. Efficient mining of recurrent rules from a sequence database. In *DASFAA*, pages 67–83, 2008.

[LLP07] Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Mining unexpected sequential patterns and rules. Technical Report RR-07027 (2007), LIRMM, 2007.

[LLP08a] Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Découverte de motifs séquentiels et de règles inattendus. In *EGC*, pages 535–540, 2008.

[LLP08b] Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Discovering fuzzy unexpected sequences with beliefs. In *IPMU*, pages 1709–1716, 2008.

[LLP08c] Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Mining unexpected Web usage behaviors. In *ICDM*, pages 283–297, 2008.

[LLP09] Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Mining unexpected sequential patterns and implication rules. In Yun Sing Koh and Nathan Rountree, editors, *Rare Association Rule Mining and Knowledge Discovery: Technologies for Infrequent and Critical Event Detection*, Advances in Data Warehousing and Mining Book Series. IGI Publishing, 2009.

[LLRP08] Dong (Haoyuan) Li, Anne Laurent, Mathieu Roche, and Pascal Poncelet. Extraction of opposite sentiments in classified free format text reviews. In *DEXA*, pages 710–717, 2008.

[LLT07] Dong (Haoyuan) Li, Anne Laurent, and Maguelonne Teisseire. On transversal hypergraph enumeration in mining sequential patterns. In *IDEAS*, pages 303–307, 2007.

[LLW02] Ming-Yen Lin, Suh-Yin Lee, and Sheng-Shun Wang. DELISP: Efficient discovery of generalized sequential patterns by delimited pattern-growth technology. In *PAKDD*, pages 198–209, 2002.

[LMY01] Bing Liu, Yiming Ma, and Philip S. Yu. Discovering unexpected information from your competitors' Web sites. In *KDD*, pages 144–153, 2001.

[LSST+02] Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

[LZO99] Neal Lesh, Mohammed J. Zaki, and Mitsunori Ogihara. Mining features for sequence classification. In *KDD*, pages 342–346, 1999.

[McC96] Andrew Kachites McCallum. Bow: A toolkit for statistical language modeling text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow, 1996.

[McG05] Ken McGarry. A survey of interestingness measures for knowledge discovery. *The Knowledge Engineering Review*, 20(1):39–61, 2005.

[MCP98] Florent Masseglia, Fabienne Cathala, and Pascal Poncelet. The PSP approach for mining sequential patterns. In *PKDD*, pages 176–184, 1998.

[MCS06] Rada Mihalcea, Courtney Corley, and Carlo Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, pages 775–780, 2006.

[MDL+00] Bamshad Mobasher, Honghua Dai, Tao Luo, Yuqing Sun, and Jiang Zhu. Integrating Web usage and content mining for more effective personalization. In *EC-Web*, pages 165–176, 2000.

[MDLN02] Bamshad Mobasher, Honghua Dai, Tao Luo, and Miki Nakagawa. Using sequential and non-sequential patterns in predictive Web usage mining tasks. In *ICDM*, pages 669–672, 2002.

[MLK06] Alex Markov, Mark Last, and Abraham Kandel. Fast categorization of Web documents represented by graphs. In *WEBKDD*, pages 56–71, 2006.

[MPT00] Florent Masseglia, Pascal Poncelet, and Maguelonne Teisseire. Web usage mining: How to efficiently manage new transactions and new clients. In *PKDD*, pages 530–535, 2000.

[MPT04] Florent Masseglia, Pascal Poncelet, and Maguelonne Teisseire. Pre-processing time constraints for efficiently mining generalized sequential patterns. In *TIME*, pages 87–95, 2004.

[MTV97] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289, 1997.

[MVDA07] Rokia Missaoui, Petko Valtchev, Chabane Djeraba, and Mehdi Adda. Toward recommendation based on ontology-powered Web-usage mining. *IEEE Internet Computing*, 11(4):45–52, 2007.

[NCS95] NCSA HTTPd Development Team. NCSA HTTPd Online Document: Transfer-Log Directive, 1995.

[NMTM00] Kamal Nigam, Andrew Kachites Mccallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2-3):103–134, 2000.

[NMW97] Craig G. Nevill-Manning and Ian H. Witten. Identifying hierarchical structure in sequences: A linear-time algorithm. *Journal of Artificial Intelligence Research*, 7:67–82, 1997.

[Ohs06] Yukio Ohsawa. Chance discovery: The current states of art. In Yukio Ohsawa and Shusaku Tsumoto, editors, *Chance Discoveries in Real World Decision Making*, Studies in Computational Intelligence, pages 3–20. Springer, 2006.

[PE05] Ana-Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *HLT/EMNLP*, pages 339–346, 2005.

[PHMA+04] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Jianyong Wang, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Meichun Hsu. Mining sequential patterns by pattern-growth: the prefixspan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1424–1440, 2004.

[PHMAP01] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, and Helen Pinto. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *ICDE*, pages 215–224, 2001.

[PHW02] Jian Pei, Jiawei Han, and Wei Wang. Mining sequential patterns with constraints in large databases. In *CIKM*, pages 18–25, 2002.

[PHW07] Jian Pei, Jiawei Han, and Wei Wang. Constraint-based sequential pattern mining: the pattern-growth methods. *Journal of Intelligent Information Systems*, 28(2):133–160, 2007.

[PL04] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, pages 271–278, 2004.

[PLV02] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *EMNLP*, pages 79–86, 2002.

[PS07] Simone Paolo Ponzetto and Michael Strube. Knowledge derived from Wikipedia for computing semantic relatedness. *Journal of Artificial Intelligence Research*, 30:181–212, 2007.

[PS08] Giuseppe Pirrò and Nuno Seco. Design, implementation and evaluation of a new similarity metric combining feature and intrinsic information content. In *ODBASE*, pages 1271–1288, 2008.

[PSM94] Gregory Piatetsky-Shapiro and Christopher J. Matheus. The interingness of deviations. In *KDD Workshop*, pages 25–36, 1994.

[PT98] Balaji Padmanabhan and Alexander Tuzhilin. A belief-driven method for discovering unexpected patterns. In *KDD*, pages 94–100, 1998.

[PT00] Balaji Padmanabhan and Alexander Tuzhilin. Small is beautiful: Discovering the minimal set of unexpected patterns. In *KDD*, pages 54–63, 2000.

[PT02] Balaji Padmanabhan and Alexander Tuzhilin. Knowledge refinement based on the discovery of unexpected patterns in data mining. *Decision Support Systems*, 33(3):309–321, 2002.

[PT06] Balaji Padmanabhan and Alexander Tuzhilin. On characterization and discovery of minimal unexpected patterns in rule discovery. *IEEE Transactions on Knowledge and Data Engineering*, 18(2):202–216, 2006.

[RCP08] Chedy Raïssi, Toon Calders, and Pascal Poncelet. Mining conjunctive sequential patterns. *Data Mining and Knowledge Discovery*, 17(1), 2008.

[RE03] M. Andrea Rodríguez and Max J. Egenhofer. Determining semantic similarity among entity classes from different ontologies. *IEEE Transactions on Knowledge and Data Engineering*, 15(2):442–456, 2003.

[Res95] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453, 1995.

[RRS00] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *SIGMOD*, 2000.

[SA95] Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules. In *VLDB*, pages 407–419, 1995.

[SA96a] Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. In *ACM SIGMOD*, pages 1–12, 1996.

[SA96b] Ramakrishnan Srikant and Rakesh Agrawal. Mining sequential patterns: generalizations and performance improvements. In *EDBT*, pages 3–17, 1996.

[San90] Beatrice Santorini. Part-of-Speech tagging guidelines for the Penn Treebank project. Technical Report MS-CIS-90-47, Department of Computer and Information Science, University of Pennsylvania, 1990.

[SB88] Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[SC99] Mark Sanderson and W. Bruce Croft. Deriving concept hierarchies from text. In *SIGIR*, pages 206–213, 1999.

[SCA06] Pei Sun, Sanjay Chawla, and Bavani Arunasalam. Mining for outliers in sequential databases. In *SDM*, pages 94–105, 2006.

[SCDT00] Jaideep Srivastava, Robert Cooley, Mukund Deshpande, and Pang-Ning Tan. Web usage mining: Discovery and applications of usage patterns from Web data. *SIGKDD Explorations*, 1(2):12–23, 2000.

[Sch94] Helmut Schmid. Probabilistic Part-of-Speech tagging using decision trees. In *NeMLaP*, 1994.

[Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[SK98] Einoshin Suzuki and Yves Kodratoff. Discovery of surprising exception rules based on intensity of implication. In *PKDD*, pages 10–18, 1998.

[Sme88] P. Smets. Belief functions. In P. Smets, A. Mamdani, D. Dubois, and H. Prade, editors, *Non-standard logics for automated reasoning*. Academic Press, 1988.

[SPF99] Myra Spiliopoulou, Carsten Pohle, and Lukas Faulstich. Improving the effectiveness of a Web site with Web usage mining. In *WEBKDD*, pages 142–162, 1999.

[Spi99] Myra Spiliopoulou. Managing interesting rules in sequence mining. In *PKDD*, pages 554–560, 1999.

[Sri04] Padmini Srinivasan. Text mining: Generating hypotheses from MEDLINE. *Journal of the American Society for Information Science and Technology*, 55(5):396–413, 2004.

[SS96] Einoshin Suzuki and Masamichi Shimura. Exceptional knowledge discovery in databases based on information theory. In *KDD*, pages 275–278, 1996.

[ST93] Daniel D. Sleator and Davy Temperley. Parsing English with a link grammar. In *3rd International Workshop on Parsing Technologies*, 1993.

[ST95] Abraham Silberschatz and Alexander Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *KDD*, pages 275–281, 1995.

[ST96] Abraham Silberschatz and Alexander Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):970–974, 1996.

[Suz96] Einoshin Suzuki. Discovering unexpected exceptions: A stochastic approach. In *RSFD*, pages 225–232, 1996.

[Suz97] Einoshin Suzuki. Autonomous discovery of reliable exception rules. In *KDD*, pages 259–262, 1997.

[Suz06] Einoshin Suzuki. Data mining methods for discovering interesting exceptions from an unsupervised table. *The Journal of Universal Computer Science*, 12(16):627–653, 2006.

[Swa86] Don R. Swanson. Fish oil, Raynaud's syndrome, and undiscovered public knowledge. *Perspectives in Biology and Medicine*, 30:7–18, 1986.

[SWL06] Weifeng Su, Jiying Wang, and Frederick H. Lochovsky. Automatic hierarchical classification of structured deep Web databases. In *WISE*, pages 210–221, 2006.

[SZ05] Einoshin Suzuki and Jan M. Zytkow. Unified algorithm for undirected discovery of exception rules. *International Journal of Intelligent Systems*, 20(7):673–691, 2005.

[SZH+06] Yang Song, Ding Zhou, Jian Huang, Isaac G. Councill, Hongyuan Zha, and C. Lee Giles. Boosting the feature space: Text classification for unstructured data on the Web. In *ICDM*, pages 1064–1069, 2006.

[TKS02] Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. Selecting the right interestingness measure for association. In *KDD*, pages 32–41, 2002.

[TL07] Ming-Cheng Tseng and Wen-Yang Lin. Efficient mining of generalized association rules with non-uniform minimum support. *Data & Knowledge Engineering*, 62(1):41–64, 2007.

[TS98] Shiby Thomas and Sunita Sarawagi. Mining generalized association rules and sequential patterns using SQL queries. In *KDD*, pages 344–348, 1998.

[Tur01] Peter D. Turney. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In *ECML*, pages 491–502, 2001.

[Tur02] Peter D. Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *ACL*, pages 417–424, 2002.

[Wei04] Gary M. Weiss. Mining with rarity: a unifying framework. *SIGKDD Explorations*, 6(1):7–19, 2004.

[WH04] Jianyong Wang and Jiawei Han. BIDE: Efficient mining of frequent closed sequences. In *ICDE*, pages 79–90, 2004.

[WHL07] Jianyong Wang, Jiawei Han, and Chun Li. Frequent closed sequence mining without candidate maintenance. *IEEE Transactions on Knowledge and Data Engineering*, 19(8):1042–1056, 2007.

[WJL03] Ke Wang, Yuelong Jiang, and Laks V. S. Lakshmanan. Mining unexpected rules by pushing user dynamics. In *KDD*, pages 246–255, 2003.

[WKdJvdBV01] Marc Weeber, Henny Klein, Lolkje T. W. de Jong-van den Berg, and Rein Vos. Using concepts in literature-based discovery: Simulating Swanson's Raynaud-fish oil and migraine-magnesium discoveries. *Journal of the American Society for Information Science and Technology*, 52(7):548–557, 2001.

[WWH04] Theresa Wilson, Janyce Wiebe, and Rebecca Hwa. Just how mad are you? finding strong and weak opinion clauses. In *AAAI*, pages 761–769, 2004.

[WWH05] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT/EMNLP*, 2005.

[WWH06] Theresa Wilson, Janyce Wiebe, and Rebecca Hwa. Recognizing strong and weak opinion clauses. *Computational Intelligence*, 22(2):73–99, 2006.

[XPDY08] Zhengzheng Xing, Jian Pei, Guozhu Dong, and Philip S. Yu. Mining sequence classifiers for early prediction. In *SDM*, pages 644–655, 2008.

[YC94] Yiming Yang and Christopher G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems*, 13(3):252–277, 1994.

[YH03] Hong Yu and Vasileios Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *EMNLP*, pages 129–136, 2003.

[YHA03] Xifeng Yan, Jiawei Han, and Ramin Afshar. CloSpan: Mining closed sequential patterns in large databases. In *SDM*, pages 166–177, 2003.

[YHC04] Hwanjo Yu, Jiawei Han, and Kevin Chen-Chuan Chang. PEBL: Web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering*, 16(1):70–81, 2004.

[YLW04] Qiang Yang, Ian Tian Yi Li, and Ke Wang. Building association-rule based sequential classifiers for Web-document prediction. *Data Mining and Knowledge Discovery*, 8(3):253–273, 2004.

[Zad65] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[Zak01] Mohammed Javeed Zaki. SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1-2):31–60, 2001.

[ZMG08] Torsten Zesch, Christof Müller, and Iryna Gurevych. Using wiktionary for computing semantic relatedness. In *AAAI*, pages 861–866, 2008.

Publications

Book Chapters

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Mining unexpected sequential patterns and implication rules. In Yun Sing Koh and Nathan Rountree, editors, *Rare Association Rule Mining and Knowledge Discovery: Technologies for Infrequent and Critical Event Detection*, Advances in Data Warehousing and Mining Book Series. IGI Publishing, 2009.

Publications in Refereed International Journals

- Dong (Haoyuan) Li, Anne Laurent, Pascal Poncelet, and Mathieu Roche. Extraction of unexpected sentences: A sentiment classification assessed approach. *Intelligent Data Analysis Journal (IDA)*, 14(1), 2010.

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Discovery of unexpected recurrence behaviors in sequence databases. *International Journal of Computational Intelligence Research (IJCIR)*, accepted 2009.

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. WebUser: mining unexpected Web usage. *International Journal of Business Intelligence and Data Mining (IJBIDM)*, accepted 2009.

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Discovering fuzzy unexpected sequences with concept hierarchies. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems (IJUFKS)*, 17(Supplement-1):113–134, 2009.

Publications in Refereed France National Journals

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Extraction de comportements inattendus dans le cadre du Web Usage Mining. *La Revue des Nouvelles Technologies de l'Information, deuxième numéro spécial sur la : Fouille de données complexes (RNTI)*, 2009.

Publications in Refereed International Conferences

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Recognizing unexpected recurrence behaviors with fuzzy methods in sequence databases. In *Proceedings of the 5th International Conference on Soft Computing as Transdisciplinary Science and Technology (CSTST 2008)*, pages 37–43, Cergy Pontoise, France, October 2008.

- Dong (Haoyuan) Li, Anne Laurent, Mathieu Roche, and Pascal Poncelet. Extraction of opposite sentiments in classified free format text reviews. In *Proceedings of the 19th International Conference on Database and Expert Systems Applications (DEXA 2008)*, pages 710–717, Turin, Italy, September 2008.

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Mining unexpected Web usage behaviors. In *Proceedings of the 8th Industrial Conference on Data Mining (Industrial ICDM 2008)*, pages 283–297, Leipzig, Germany, July 2008. (Best Paper Award)

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Discovering fuzzy unexpected sequences with beliefs. In *Proceedings of the 12th International Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2008)*, pages 1709–1716, Málaga, Spain, Juin 2008.

- Dong (Haoyuan) Li, Anne Laurent, and Maguelonne Teisseire. On transversal hypergraph enumeration in mining sequential patterns. In *Proceedings of the 11th International Database Engineering and Applications Symposium (IDEAS 2007)*, pages 303–307, Banff, Canada, September 2007.

Publications in Refereed France National Conferences

- Dong (Haoyuan) Li, Anne Laurent, Mathieu Roche, and Pascal Poncelet. Recherche de sentiments opposés par une approche floue à partir de textes libres. In *Actes des Rencontres Francophones sur la Logique Floue et ses Applications (LFA 2008)*, pages 26–33, Lens, France, October 2008.

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Découverte de motifs séquentiels et de règles inattendus. In *Actes des 8ièmes Journées Francophones Extraction et Gestion des Connaissances (EGC 2008)*, pages 535–540, Sophia Antipolis, France, January 2008.

Publications in Workshops

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Towards unexpected sequential patterns. In *Atelier Bases de Données Inductives*, Plateforme Afia, Grenoble, France, July 2007.

Technical Reports

- Dong (Haoyuan) Li, Anne Laurent, and Pascal Poncelet. Mining unexpected sequential patterns and rules. *Technical Report RR-07027 (2007)*, Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier, 2007.

Extraction de séquences inattendues: des motifs séquentiels aux règles d'implication

Les motifs séquentiels peuvent être vus comme une extension de la notion d'itemsets fréquents intégrant diverses contraintes temporelles. La recherche de tels motifs consiste ainsi à extraire des enchaînements d'ensembles d'items, couramment associés sur une période de temps bien spécifiée. La construction de règles à partir de ces motifs séquentiels permet d'étendre la notion de règles d'association pour la pris en compte de la temporalité. En fait, cette recherche met en évidence des associations inter-transactions, contrairement à celle des règles d'association qui extrait des combinaisons intra-transactions. Ce problème, posé à l'origine dans un contexte de marketing, intéresse à présent des domaines aussi variés que les télécommunications, la finance, ou encore la médecine et la bioinformatique.

Même s'il existe aujourd'hui de très nombreuses approches efficaces pour extraire des motifs, ces derniers ne sont pas forcément adaptés aux besoins des applications réelles. En fait, les résultats obtenus sont basés sur une mesure statistique et ne tiennent pas compte de la connaissance du domaine. De plus, ces approches sont principalement axées sur la recherche de tendances et ne permettent pas d'extraire des connaissances sur les éléments atypiques ou inattendus.

Dans le cadre de cette thèse, nous nous intéressons donc à la problématique de l'extraction de motifs séquentiels et règles inattendus en intégrant la connaissance du domaine. Le travail présenté dans cette thèse comporte la mise en œuvre d'un cadre MUSE pour l'extraction de séquences inattendues par rapport à un système de croyances, des extensions avec la théorie de logique floue, l'intégration des données hiérarchisées, la définition des motifs séquentiels et règles inattendus et, enfin, l'extraction de phrases inattendues dans des documents textes. Des expérimentations menées sur des données synthétiques et sur des données réelles sont rapportées et montrent l'intérêt de nos propositions.

Mots-clés : Extraction de connaissances, fouille de données, base de données de séquences, mesure d'intérêt, système de croyances, séquences inattendues, motifs séquentiels, règles séquentielles, logique floue, hiérarchie, validation, classification de textes.

Discipline : Informatique
Laboratoire : Laboratoire de Génie Informatique et d'Ingénierie de Production
École des Mines d'Alès
Parc scientifique Georges Besse - 30035 Nîmes cedex 1 - France