DATA STREAM MINING IN MEDICAL SENSOR-CLOUD

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

College of Engineering and Science

Victoria University

by

Le Sun

Jan. 2016

ABSTRACT

DATA STREAM MINING IN MEDICAL SENSOR-CLOUD

Le Sun, Ph.D.

Victoria University 2016

Data stream mining has been studied in diverse application domains. In recent years, a population aging is stressing the national and international health care systems. Along with the advent of hundreds and thousands of health monitoring sensors, the traditional wireless sensor networks and anomaly detection techniques cannot handle huge amounts of information. Sensor-cloud makes the processing and storage of big sensor data much easier. Sensor-cloud is an extension of Cloud by connecting the Wireless Sensor Networks (WSNs) and the cloud through sensor and cloud gateways, which consistently collect and process a large amount of data from various sensors located in different areas. In this thesis, I will focus on analysing a large volume of medical sensor data streams collected from Sensor-cloud. To analyse the Medical data streams, I propose a medical data stream mining framework, which is targeted on tackling four main challenges:

Problem 1. Segment data streams. Medical data streams are very long (consistently) and have medical-related features (e.g., pseudo-periodic). In certain situations, it is necessary to segment the long data stream to short sub-sequences and to analyse the data based on these sub-sequences.

Solution 1. I propose a novel concept of a feature vector to capture the features of the key points in a data stream, and introduce a silhouette-value based approach to identify the periodic points that can effectively segment the data

stream into a set of consecutive periods with similar patterns.

Problem 2. Detect abnormal subsequence in a data stream. Anomaly detection in medical data streams can assist doctors or patients in diagnosing disease or analysing physical abnormal signals. It is normally intractable to exactly discover the discords based on the original continuous data streams.

Solution 2. I introduce a classification-based framework for anomaly detection in uncertain pseudo periodic medical data streams. The procedure of pre-processing uncertainties can reduce the noise of anomalies and improve the accuracy of anomaly detection.

Problem 3. Identify variable-length motifs. Finding motifs (i.e., repeated patterns) is still an open problem in continuous stream mining, though it has been studied for decades in the area of mining discrete data streams. Particularly, most researchers assume the repeated patterns having similar lengths in one data stream, which is definitely incompatible with the real-word cases that have multiple types of motifs with different lengths.

Solution 3. I propose an unsupervised Limited-length suffix array based Motif Discovery algorithm (LiSAM) for continuous time series, which is time and space efficient, and supports approximately discovering motifs in different lengths.

Problem 4. Select qualified Cloud service for Sensor-cloud construction. When making a purchasing decision for cloud services, healthcare IT managers should evaluate cloud service providers in terms of user specific requirements, to proactively resolve the precursors of cost leakages or service failures.

Solution 4. I propose a service selection framework to assist healthcare technicians choosing the optimal cloud services.

DOCTOR OF PHILOSOPHY DECLARATION

I, Le Sun, declare that the PhD thesis entitled *Data Stream Mining in Medical Sensor-cloud* is no more than 100,000 words in length including quotes and exclusive of tables, figures, appendices, bibliography, references and footnotes. This thesis contains no material that has been submitted previously, in whole or in part, for the award of any other academic degree or diploma. Except where otherwise indicated, this thesis is my own work.

Signature  Date 22/01/2016

ACKNOWLEDGEMENTS

First and foremost, I am indebted to my principal supervisor, Professor Yanchun Zhang, for fully supporting me throughout the course of my doctoral program at Victoria University and for patiently guiding, pushing and encouraging me to focus on conducting high level research. At the same time, I would like to thank my associate supervisor, Dr. Jiangang Ma, who is also been very supportive. He gracefully gave me encouragement, guidance, and advice that were important for completing this thesis and assisting in each stage of my study.

I would like to thank Professor Hua Wang, who gave me constructive advice on my research. I would like to thank Dr. Rui Zhou, Dr. Hai Dong, PhD candidates Ye Wang, Haixin Jiang, and master student Limeng Zhang, who had valuable discussion with me on data stream mining techniques. I would like to thank all the members and visitors of the Center for Applied Informatics (CAI).

PUBLICATIONS

Refereed Journal articles

1. **Le Sun**, Jiangang Ma, Yanchun Zhang. LiSAM: Fast and Approximate Discovery of Time Series Motifs in Different Lengths. Submitted to *Journal of Time Series Analysis*. 2015.
2. **Le Sun**, Jiangang Ma, Yanchun Zhang, Hai Dong, Farookh Khadeer Hussain. Exploring criteria interdependence in MCDM and its application in Cloud service selection. Submitted to *Computing*. 2015.
3. Jiangang Ma, **Le Sun**, Yanchun Zhang. Supervised Anomaly Detection in Uncertain Pseudo-periodic Data Streams. *ACM Transactions on Internet Technology*. Accepted, 2015.
4. **Le Sun**, Jiangang Ma, Yanchun Zhang. Fuzzy-Ontology-Driven Cloud Service Selection. *Future Generation Computer Systems*. Accepted. 2015.
5. **Le Sun**, Jiangan Ma, Hua Wang, Yanchun Zhang, and Jianming Yong. Cloud Service Description Model: An Extension of USDL for Cloud Services. *IEEE Transactions on Services Computing*. Accepted. 2015. http://doi.ieeecomputersociety.org/10.1109/TSC.2015.2474386.
6. **Le Sun**, Hai Dong, Farookh Khadeer Hussain. Cloud Service Selection: State-of-the-Art and Future Research Directions. *Journal of Network and Computer Applications*, Volume 45, October 2014, pp. 134150.

Refereed Conference articles:

1. **Le Sun**, Hai Dong, Farookh Khadeer Hussain, Omar Khadeer Hussain, Jiangang Ma, and Yanchun Zhang. A Hybrid Fuzzy Framework for Cloud Service Selection. *International Conference on Web Services*, Alaska, USA, June, 2014.

2. **Le Sun**, Hai Dong, Farookh Khadeer Hussain, Omar Khadeer Hussain, Jiangang Ma, and Yanchun Zhang. Multicriteria Decision Making with Fuzziness and Criteria Interdependence in Cloud Service Selection. *IEEE World Congress on Computational Intelligence*, Beijing, China, 2014.

3. **Le Sun**, Hai Dong, and Jamshaid Ashraf. Survey of service description languages and their issues in cloud computing. in *Proceedings of the International Conference on Semantics*, Knowledge and Grids, Beijing, 2012.

4. **Le Sun**, Jaipal Singh, and Omar Khadeer Hussain. Service Level Agreement (SLA) Assurance for Cloud Services: A Survey from a Transactional Risk Perspective. *International Conference on Information Integration and Web-based Applications & Services (iiWAS2012)*, Bali, Indonesia, 2012.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The fast development of the sensor, wireless and cloud computing technologies enable a smart healthcare mechanism that supports the consistent remote monitoring on the physical conditions of patients, elderly people or babies, and the efficient processing of the large sensing data sets. Such a smart healthcare mechanism can enhance the quality of life significantly. However, as investigated by the Cloud Standards Customer Council [43], the healthcare institutions are not keen on building smart healthcare systems based on the IT technologies, especially in developing countries. The under-utilization of IT technologies prevents the wide information sharing and the fast information processing in healthcare industry.

1.1 Background

In this section, I introduce the background knowledge of the healthcare data stream mining. I discuss the background knowledge in terms of three dimensions: the wireless sensor network technologies for healcare, the sensor-cloud technologies for healthcare, and the data stream mining technologies for healthcare.

1.1.1 Wireless Sensor Networks for Healthcare

Making healthcare decisions is a tough task that is influenced by a variety of factors: the lack of medical knowledge, the subjective mistakes of the health-

care givers, the false and incomplete information, and the misunderstanding and misinterpretation. The development of the wireless sensor network technologies improves the accuracy of the collected information. The advancing of the decision making mechanisms that connect the sensor networks and the decision makers guarantee the correct medical decision making.

There have been many researches and applications on building smart healthcare systems based on the wireless sensor network (WSN) technologies. A wireless sensor network is composed of a set of distributed sensor nodes that monitor a specific object, like temperature, blood pressure, heartbeat, and communicate with each other over limited frequencies and bandwidths [10]. The sensor nodes are self-managed and randomly deployed [9], which are capable of sensing, communicating, and processing the monitored symbols [8].

However, along with the advent of hundreds and thousands of health monitoring sensors, the traditional WSNs and anomaly detection techniques cannot handle huge amount of information. They need to overcome difficulties on constantly communicating, saving and processing a large amount of data streams. When the size of the wireless network is very large, the communication power among nodes is constrained by the distance and environment obstructions, which in turn reduces the application performance based on the WSNs [9]. To resolve these challenges, researchers consider to combine cloud computing technologies and WSNs.

1.1.2 Sensor-cloud for Healthcare

Cloud computing paradigms enable a virtual mechanism for IT resource management and usage. Sensor-cloud infrastructure [231] is a technology that integrates cloud techniques into the WSNs. It provides users a virtual platform for utilizing the physical sensors in a transparent and convenient way. Users can control, monitor, create, and check the distributed physical sensors without knowing their physical details, but just using a few of functions. In this section, I summarize the definitions and the features of the sensor-cloud infrastructure based on a previous survey [9].

Cloud computing provides the IT resources for WSNs, which supports the storage and fast processing of a large amount of sensor data streams. The connection between WSNs and clouds is implemented by two types of gateways: sensor gateways and cloud gateways, where sensor gateways collect and compress information, and cloud gateways decompress and process information [112].

Sensor-clouds have been used in various applications, such as the disaster prediction, environment monitoring and healthcare analysis. Sensor-clouds collect data from different applications, and share and process the data based on the cloud computational and storage resources. A user interface is provided to the users to manage and monitor the virtual sensors.

In Sensor-clouds, the sensor modeling language (SML) is utilized to describe the information of the physical sensors, which is processed as the metadata of the sensors. The standard language format enables the collaboration among sensors in different networks and platforms. The XML encoding also provides

a mapping mechanism for transforming commands and information between virtual and physical sensors.

Compared with the traditional sensor networks, Sensor-clouds have the following advantages [9]: (1) the capability of dealing with various data types; (2) scalable resources; (3) user and network collaboration; (4) data visualization; (5) flexible data access and resource usage; (6) low cost; (7) automated resource delivery and data management; (8) less processing and response time.

In healthcare industry, a large amount of medical sensors have been deployed to monitor medical signals. Medical sensor data are gathered and stored for flexible access. Cloud computing enables fast and low-cost accessing to the medical data whenever and wherever needed. Sensor and cloud technologies make it possible to accurately collect and utilize the medical data that diagnosed from the remote patients. Medical sensor monitoring enhances the healthcare system by remotely detecting the early-stage abnormal health signals. Sensors can monitor a variety of body signals such as blood pressure, heart rate, and temperature [42]. The monitored data are transferred to the data analysers or doctors for diagnosing or predicting patient conditions.

1.1.3 Data Stream Mining for Healthcare

The speed of data processing is being improved dramatically, so that data mining processes can be executed automatically and consistently in real-time, which assist decision makers in making instant and accurate decisions. The processed multiple data streams may come from different types of sources, like sensors, mobiles, and cameras [42].

Data mining is a process of identifying useful yet previously hidden knowledge from data sets, containing a series of techniques, such as classification, clustering, and association rule mining [56]. Huge volume healthcare data are being produced from various sources like disease monitoring, patient historical records, specialist diagnosis, and hospital devices. These healthcare data are valuable and useful to assist the care givers in making decision efficiently. Data mining techniques have been applied in healthcare industry in terms of a variety of areas [56], for example, tracking the states of patients who have chronic diseases or physical disabilities, analysing the diagnosis results and recommending the treatment programs, identifying the patterns or rules of the 'fraud and abuse' to detect the fraudulent claims or the wrong referrals.

Data streams are ordered by time, updated continuously, huge volume, and mostly infinite [165]. To resolve data stream mining problems, several challenges should be handled properly, which mainly include: (1) how to instantly process the data streams of infinite volumes without taking massive storage space to store them; (2) how to dynamically deal with the concept drifting and evolution in a data stream. Fig. 1.1 describes a process of data stream mining [165].

1.2 Motivations and Problems

This thesis focuses on analysing large volume of medical sensor data streams based on sensor-cloud architectures. The motivation of this research is analyzed from the following aspects:

1. Segment data streams

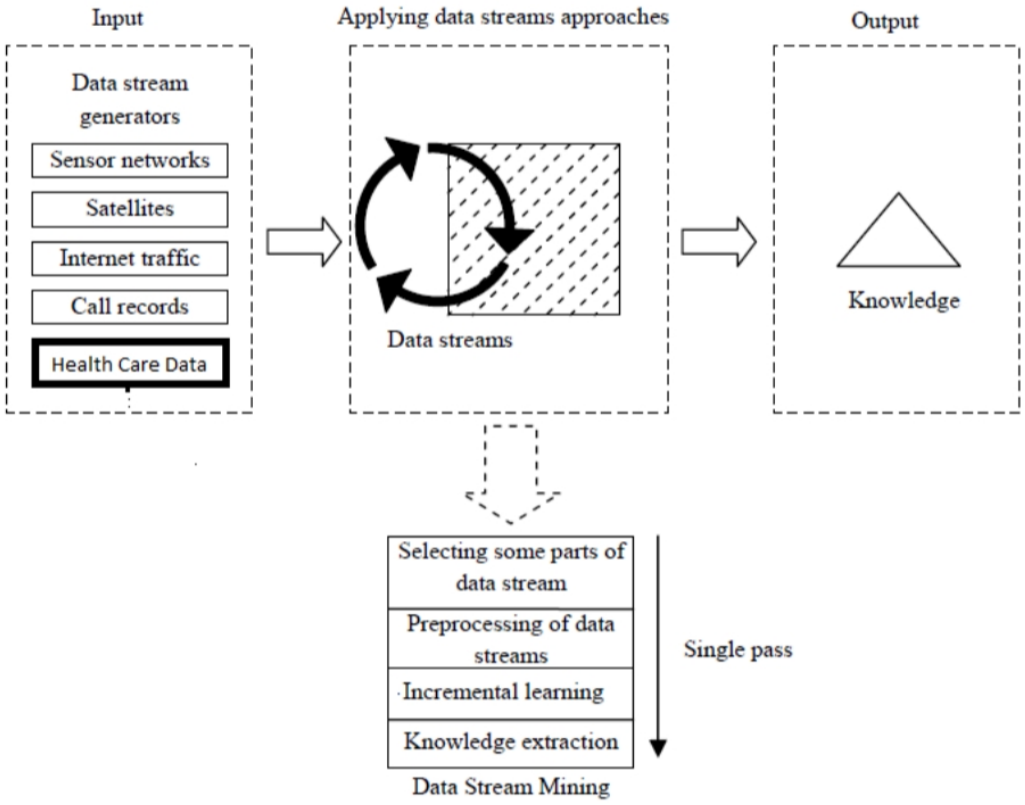


Figure 1.1: Data stream mining [165]

Medical data streams are very long (consistently) and have medical-related features (e.g., pseudo-periodic). In certain situations, it is necessary to segment the long data stream to short sub-sequences and to analyse the data based on these sub-sequences. However, it is difficult to get high performance (e.g., accuracy and computational complexity) by applying the segmentation techniques against general areas to medical data sets. Thus,

Issue 1. Find an efficient way of segmenting medical data streams based on the medical-related features.

2. Detect abnormal subsequences in a data stream

Anomaly detection is a typical example of a data stream mining application. Here, anomalies or outliers or exceptions often refer to the patterns in data streams that deviate expected normal behaviours. Thus, anomaly detection is a dynamic process of finding abnormal behaviours from given data streams.

Anomaly detection in medical data streams can assist doctors or patients in diagnosing diseases or analysing physical abnormal signals. Most existing work detects the abnormal sub-sequences in data streams in an approximate way by first converting the continuous data stream to a discrete data stream. It is normally intractable to exactly discover the discords based on the original continuous data streams.

Issue 2. Find a computation-efficient way to exactly discover the abnormal sub-sequences in uncertain medical data streams

3. Identify variable-length motifs

Finding motifs (i.e., repeated patterns) is still an open problem in continuous stream mining, though it has been studied for decades in the area of mining discrete data streams. It has been proved that the subsequence clustering is meaningless in unsupervised data stream mining area, and the motif grouping in the discrete data stream mining has been applied as a replacement of the subsequence-clustering in the real time series.

In particular, most researchers assume the repeated patterns having similar lengths in one data stream, which is definitely incompatible with the real-word cases that have multiple types of motifs with different lengths. Especially, the occurrences of the same motif may have similar shapes yet variable lengths. So I will also discuss in this thesis the problem of discovering variable-length

motifs in medical sensor data streams, where the discovered motifs can be used to recognize the repeating occurred behaviours or physical conditions.

Issue 3. Discover variable-length motifs in medical sensor data streams with low time and space complexities.

4. Select qualified cloud services for sensor-cloud construction.

When making a purchasing decision for cloud services, healthcare IT managers should evaluate cloud service providers in terms of user specific requirements, to proactively resolve the precursors of cost leakages or service failures. It is valuable to explore the problem of cloud service selection to achieve a best trade-off between the spending and quality of using cloud services for building healthcare sensor-clouds.

In the cloud area, it is more difficult for decision makers to make informed decisions on service usage because of the diversification of service types and the lack of service publication standards.

Issue 4. Find cloud services with the user-expected functions given the fuzzy expressions of user requirements

1.3 Solutions and Contributions

This section introduces the solutions proposed in this thesis against the four problems given in the previous section.

1.3.1 Supervised Anomaly Detection in Uncertain Pseudo-periodic Data Streams

To deal with issue 1 and issue 2, I propose a supervised classification framework for detecting anomalies in uncertain pseudo periodic time series, which comprises four components: an uncertainty identification and correction component (UICC), a time series compression component (TSCC), a period segmentation and summarization component (PSSC), and a classification and anomaly detection component (CADC). First, UICC processes a time series to remove uncertainties from the time series. Then TSCC compresses the processed raw time series to an approximate time series. Afterwards the PSSC identifies the periodic patterns of the time series and extracts the most important features of each period, and finally the CADC detects anomalies based on the selected features.

Contributions

- I present a classification-based framework for anomaly detection in uncertain pseudo periodic time series, together with a novel set of techniques for segmenting and extracting the main features of a time series. The procedure of pre-processing uncertainties can reduce the noise of anomalies and improve the accuracy of anomaly detection. The time series segmentation and feature extraction techniques can improve the performance and the time efficiency of classification.
- I propose the novel concept of a feature vector to capture the features of the turning points in a time series, and introduce a silhouette value based approach to identify the periodic points that can effectively segment the time series into a set of consecutive periods with similar patterns.

1.3.2 LiSAM: Fast and Approximate Discovery of Different-length Time Series Motifs

To deal with issue 3, I propose an unsupervised Limited-length Suffix Array based Motif Discovery algorithm (LiSAM) for continuous time series, which is time and space efficient, and supports approximately discovering motifs in different lengths. I first convert the continuous time series to the discrete time series by using the Symbolic Aggregate approXimation procedure (SAX) [101], and then identify the different-length motifs based on the discrete time series. The illustration of discrete motif discovery is on the basis of an exact substring matching procedure, however, I can easily embed the existing approximate substring matching methods, such as, in LiSAM. That is, I use the exact subsequence grouping of discrete time series to discover the approximate patterns of continuous time series. I can also calculate the exact similarities between the instances of a continuous motif after such an approximate grouping.

Contributions

- *LiSAM* can discover motifs in different lengths (e.g., *maxLength* to *minLength* provided by users), avoid the unexpected trivial-matching by allowing user-defined overlapping degree (represented as α) between the instances of motifs, and support discovering motifs that overlap with each other in a specified degree (β). It can either be an automatic or semi-automatic algorithm by either manually setting all the parameters or by using default parameters (e.g., set $maxLength = \frac{1}{2}|T|$ (T is a time series), $minLength = 2$, $\alpha = 0$ and $\beta = 0$).
- *LiSAM* is both space and time efficient. It has linear space complexity

$O(N)$. Existing approximate solutions [180, 135] applied the suffix tree to model the discrete time series to increase the searching speed of a subsequence, which consumes a large volume of storage space. Instead, I use a limited-length enhanced suffix array with linear space consumption to improve the space efficiency. In addition, in an extreme case that S has maximum LCP intervals, $O(LiSAM) = O(N + n)$, while in the case an interval has maximum child intervals, $O(LiSAM) = O(N + n^2)$, where N is the length of the raw time series T, and n is the length of the discrete time series S. If $N \gg n$, the performance can be improved dramatically.

1.3.3 Decision Making Framework for Cloud Service Selection

To deal with issue 4, I propose the following Cloud service selection approaches.

Cloud-FuSeR: Fuzzy Ontology and MCDM-based Cloud Service Selection

I propose a **F**uzzy **U**ser-oriented **Cloud SeR**vice Selection System (Cloud-FuSeR) that is capable of dealing with fuzzy information and rating cloud services by considering three aspects: (1) the similarities between user-required functions and the service functions provided by cloud providers, (2) the performance of the non-functional properties, and (3) the user preference on different properties.

Contributions

- I build a fuzzy cloud ontology to support the functional similarity calculation, which defines the concept taxonomies of cloud services and the

properties of cloud services, and quantifies the relations among concepts and between concepts and properties;

- I define fuzzy functions to quantify the performance of the non-functional properties and the user preference on different properties, and employ a fuzzy-AHP and fuzzy-TOPSIS techniques to weigh the non-functional properties based on the user preference and to measure the service non-functional performance;

Exploring Criteria Interdependence in MCDM and its Application in Cloud Service Selection

I propose a MCDM framework that helps Decision Makers (DMs) to build the criteria relations and measures the performance of cloud service alternatives.

Contributions

- I describe an I-ISM approach that can help DMs construct different types of criteria relations, which allows the DMs adjust the relations interactively during the construction process until a consistent relation network is established.
- Based on the criteria relation constructed by I-ISM, I use 2-order Choquet Integral approach to aggregate the criterion performance. The 2-order Choquet Integral is a simplified procedure of the Choquet Integral that can aggregate non-additive utilities by considering the utilities of single criterion and the interactive utilities between two criteria.
- I apply the proposed decision making method to the cloud service selection problem, and use a User-oriented sigmoid utility function to get the

intra-utilities of a criterion w.r.t. different alternatives. The user-oriented sigmoid utility function is flexible enough to reflect cloud users' requirements in different contexts.

1.4 Thesis Structure

This thesis has two main parts: Part I (Chapter 2, 3, and 4) introduces the data stream mining technologies for medical anomaly detection and pattern recognition, and Part II (Chapter 5 and Chapter 6) describes the cloud service selection approaches for selecting the sensor-cloud services to support the data stream mining process. Chapter 2 reviews the literature in the data stream mining area, particularly, in anomaly detection and motif identification for data streams. Chapter 3 presents a framework for time series segmentation and anomaly detection in uncertain medical data streams. Chapter 4 introduces an algorithm LiSAM that identifies variable length motifs in medical time series. Chapter 5 and 6 focus on developing decision making methods of selecting sensor-cloud services. Chapter 5 presents a framework Cloud-FuSeR that deals with fuzzy information in decision making process, and Chapter 6 introduces an approach to process the interdependency among decision criteria. Chapter 7 concludes this thesis.

Part I

Part I: Anomaly Detection and Pattern Recognition in Medical Data Streams

CHAPTER 2

RELATED WORK

In this chapter, I introduce the related work in the area of Data Stream Mining. In particularly, I review the anomaly detection and motif discovery for data streams. In Section 2.1, previous survey on anomaly detection is introduced. Then, I review the primary techniques for anomaly detection of time series in recent five years (from 2011 to 2015). In Section 2.3, the related work on motif discovery of data streams is discussed.

2.1 Previous Survey on Anomaly Detection

Chandola et al. [34] did a comprehensive survey on the techniques (developed in the past two decades, until 2010) of anomaly detection in discrete sequences, and discussed how to apply these techniques to the problem of time series anomaly detection. The authors categorised the existing techniques into three general groups: (1) detect abnormal sequences in a sequence database; (2) detect abnormal subsequences in a long sequence; and (3) identify abnormal patterns in a sequence. As the authors stated, the techniques in these three categories can be easily adapted to the domain of anomaly detection in time series, so I present the main techniques analysed in [34] in this section.

2.1.1 Anomalous Sequence Detection in a Sequence Database

Chandola et al. [34] classified techniques in this category as: Similarity-based anomaly detection, Window-based anomaly detection, and Markov-model-

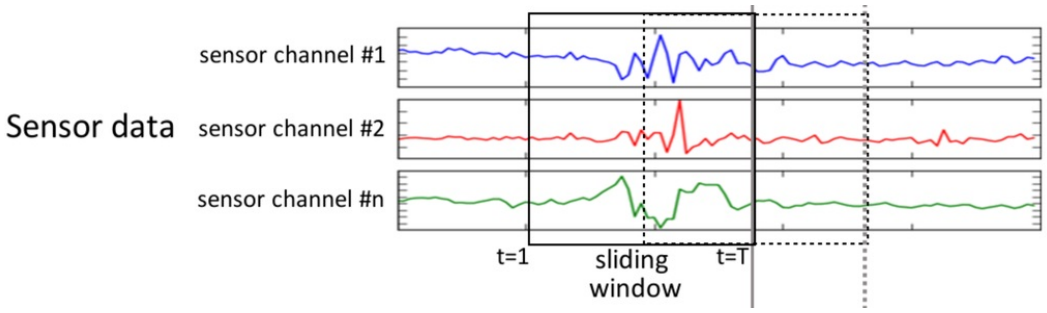


Figure 2.1: An example of sliding-windows on multiple sensor data streams

based anomaly detection.

Similarity-based Techniques

Similarity-based techniques group sequences into a number of clusters based on the similarities (or distances) between these sequences. Different clustering techniques were discussed in [34], such as the k-nearest neighbor(kNN) techniques [171], the k-medoid algorithm [26], the Probabilistic Suffix Trees [224], and Maximum Entropy models [166]. The work of [34] also presented two main distance calculation techniques for discrete sequences: the Simple Matching Coefficient [77], and the length of the longest common subsequence [35].

Sliding-Window-based Techniques

Window-based techniques use the abnormal degrees of the fixed-length windows applied on a sequence to measure the anomalous degree of this sequence. An example of applying sliding-window on multiple sensor data streams is shown in Figure 2.1 [163].

There are various techniques developed for calculating the anomaly scores of windows and of a sequence. The main techniques of anomaly score calculation of windows include the threshold-based sequence time delay embedding (t-STIDE) [217], anomaly score calculation based on the lookahead pairs [68], anomaly score calculation based on the normal [91] or abnormal dictionaries [50, 49], abnormal window identification based on classification techniques [70, 76, 205].

When the anomaly score of each window is obtained, the anomaly score of the entire sequence is calculated by aggregating the anomaly scores of all the windows. Chandola et al. [34] discussed three main aggregation techniques: the average anomaly score [91], the locality frame count [217], and the leaky bucket [73].

Markovian Techniques

Markovian Techniques calculate the probabilities of symbols at certain positions in a test sequence based on the Markov models that are learned from a set of training sequences. This kind of techniques estimate the symbol probabilities based on the symbols in a short historical period (e.g., k) rather then on the previous overall symbols, which is as shown in Formula 2.1.

$$P(t_i|t_1t_2\cdots t_{i-1}) = P(t_i|t_{i-k}t_{i-k+1}\cdots t_{u-1}) \tag{2.1}$$

Different Markovian techniques have been developed. For example, Ye [228] designed a Markovian model that considers one-step backward (i.e., $k = 1$) to estimate the current state probabilities. Michael and Ghosh [152] proposed an

Extended Finite State Automata (EFSA) that stores the frequencies of certain length subsequences to save the memory space of establishing the Markovian model, where the EFSA only keeps the subsequences and the transitions that can happens in the training sequences. Marceau [149] utilized suffix trees to assist the FSA-based state estimation, where a suffix tree maintains the $k + 1$ length subsequences.

The standard Markovian models can only be used to deal with the cases based on the fixed length historical symbols. Variable length Markovian models (e.g., Probabilisitic Suffix Trees (PSTs) [198]) are applied to support a flexible estimation mechanism based on the variable length history.

The sparse Markovian techniques estimate the probability of the current symbols based on a sparse historical period. That is, the symbols in the historical period are not necessarily continuous. Eskin et al. [58] employed the Sparse Markov Transducers (SMTs) to estimate the transition probability distribution of symbols, in which the utilization of wild cards ensures the sparsity of the input sequences. Lee et al. [125] proposed to use a classification approach RIPPER to learn the rules of building the sparse Markovian models of sequences.

Hidden Markov models (HMMs) are also widely used to model sequences. A general process [116, 223] of using HMM is as: (1) learn an HMM for the normal sequences; (2) calculate the probabilities of test sequences; and (3) obtain the anomaly score of each test sequence by using the probabilities' negative log. Florez et al. [64] calculated the best hidden state sequence of an HMM of a normal training sequence by using the Viterbi algorithm [67], and utilized a threshold method to distinguish the normal and abnormal state transitions. The anomalous score of a test sequence is the average anomaly score of the state

transition probabilities.

2.1.2 Anomalous Contiguous Subsequence Detection in a Sequence

Identifying anomalous contiguous subsequences in a long sequence is another important problem in the area of the anomaly detection of sequences. Techniques developed for solving this problem are mainly based on the window-based sequence segmentation, where a sequence is segmented into a set of contiguous subsequences by a window of length k.

Chandola et al. [34] summarized a basic technique [104, 101] for anomaly detection in this category: each window is attached with an anomaly score based on the difference degree between this window and the other windows, and the windows with highest difference degrees are considered as the abnormal subsequences. Keogh et al. [107] proposed a Window Comparison Anomaly Detection method that utilized the Compression-Based Dissimilarity technique to measure the anomaly degree of a window in a time series.

There are researches [101, 218] that focused on reducing the time complexity of the window-based anomaly detection for contiguous sequences. One such technique [74, 134] only scores part of the windows to identify the top n anomalous windows, which prunes anomalous windows in an earlier stage based on the distances of the windows to their m nearest neighbours.

Anomaly detection for contiguous subsequences based on fixed-length (e.g., length k) windows has difficulties on choosing k, especially in the case that there

are different-length abnormal subsequences. Chakrabarti et al. [30] proposed a technique to segment a sequence into different-length subsequences based on Shannons Source Coding Theorem. Gwadera et al. [85] designed a variable Markov chain method for sequence segmentation.

2.1.3 Abnormal Pattern Identification in Sequences

The problem of anomaly detection based on the identification of pattern recognition is to find sequences that are significantly different from the normal sequences in terms of the query patterns. The basic technique summarised by Chandola et al. [34] calculated the anomaly score of a query pattern as formula 2.2.

$$A(\alpha) = |\overline{f_t}(\alpha) - \overline{f_S}(\alpha)| \tag{2.2}$$

where,

$$\overline{f_t}(\alpha) = \frac{\overline{f_t}(\alpha)}{l_t} \tag{2.3}$$

$$\overline{f_S}(\alpha) = \frac{1}{|S|}\Sigma_{s_i \in S}\frac{f_{s_i}(\alpha)}{|l_{s_i}|} \tag{2.4}$$

There are a series of improvements of this technique against different problems. For example, Keogh et al. [106] calculated the anomaly score of a query by counting the occurrence times of this query in a test sequence. Gwadera et al. [86] counted the number of all the windows containing the queried pattern as a subsequence.

2.2 Anomaly Detection in Time Series

In this section, I discuss the techniques of the time series anomaly detection in recent five years. Our discussion focuses on two types of work: online anomaly detection in data streams, and off-line anomaly detection in static time series.

2.2.1 Online Anomaly Detection in Data Streams

Masud et al. [150] focused on developing the data stream classification techniques with the consideration on the concept-drifting and concept-evolution. They designed the ECSMiner algorithm to automatically identify novel classes in a consistent data stream on the basis of the traditional classification techniques. The ECSMiner uses a delayed time period to determine the classification category of a test instance and adopts another time delay and a buffer space to label and store a set of test time points. In addition, to speed up the classification process, the authors defined a 'pseudopoint' that summarizes a cluster of similar time points to reduce both the computation and space complexities.

Tan et al. [202] proposed a one-class anomaly detection algorithm: Streaming Half-Space Trees (HS-Trees). It can dynamically process endless data streams by consuming constant memory volume, and it takes constant time to build and update the tree. The proposed anomaly detection algorithm uses 'Mass' (i.e., the number of data items) to measure the anomaly degree of a streaming data item. The HS-Tree of a data stream can be constructed very fast as the tree construction depends only on the Mass in a data space. The data streams are divided into equal-sized segments (i.e., having similar number of

items), and two consecutive windows are defined: the reference window and the latest window. The initial mass profile of the data stream is first learned based on the data items in the reference window. Then the new arriving data are stored in the latest window, and the anomaly scores of these data are calculated based on the initial mass profile. The reference window is updated consistently as long as the latest window is full, which adapts to the evolution of the data streams.

Das et al. [48] studied the anomaly detection in commercial fleet data streams. The authors developed a multiple kernel learning method to detect safety anomalies in both discrete and continuous data. The proposed method applied the one-class support vector machine (SVM) method to detect anomalous, in which the various types of knowledge are incorporated by using multiple kernels.

At first, the SVM method solves an optimization problem:
minimize

$$Q = \frac{1}{2}\Sigma_{i,j}\alpha_i\alpha_j k(x_i, x_j) \tag{2.5}$$

subject to $0 \leq \alpha_i \leq \frac{1}{lv}$, $\Sigma_i\alpha_i = 1, v \in [0, 1]$;
where α is the Lagrange multiplier, k is a kernel matrix, l is the length of the data sequence, and v is a user-defined parameter for upper-bounding the fraction of the outliers and lower-bounding the fractions of the support vectors.

By solving the optimization problem of 2.5, I can get a set of support vectors $\{x_i | i \in [0, l], \alpha_i > 0\}$. Then the value of a decision function is calculated to estimate the label of a point x_j by using the formula 2.6. If the f_{x_j} is negative, x_j is an abnormal point.

$$f(x_j) = sgn(\Sigma_{i \in I_m} \alpha_i k(x_i, x_j) + \Sigma_{i \in I_{nm}} k(x_i, x_j) - \rho) \tag{2.6}$$

where $I_m = \{i | 0 < \alpha_i < 1\}$, and $I_{nm} = \{i | \alpha_i = 1\}$.

I use a kernel function 2.7 to measure the distance between two discrete points x_i and x_j.

$$K_d(x_i, x_j) = \frac{|LCS(x_i, x_j)|}{\sqrt{l_{x_i}, l_{x_j}}} \tag{2.7}$$

where $LCS(x_i, x_j)$ represents the longest common sequence of x_i and x_j.

The continuous data sequences, the continuous values will be first transformed to discrete variables by using the SAX technique [137], and then the kernel function 2.7 is applied. The kernel for the mixture data sets is defined as an integration of the discrete and continuous data sets by using the weight aggregation function 2.8.

$$k(x_i, x_j) = wK_d(x_i, x_j) + (1 - w)K_c(x_i, x_j) \tag{2.8}$$

Liu et al. [139] investigated the anomaly detection problem in continuous time series using isolation techniques. They pointed out that the time series item classification methods based on the density and distance based measures are too time and space consuming to be applied in large databases. To implement the isolation-based anomaly detection, they utilized two key features (few and different) of the abnormal data items compared with the normal items. They proposed to use an isolation tree (iTree) to isolate the data items: the abnormal items tend to be closer to the root of the iTree, while the normal items are usually at further locations. Then an ensemble of iTrees is constructed to form an

isolation forest (iForest), by sub-sampling the subsets of the time series. The iForest technique can identify the abnormal items by selecting the nodes in the iTrees that are close to the roots in terms of average path lengths. The authors also illustrated how to determine the size of sub-sampling and the number of ensemble iTrees. In particular, a few trees (e.g., 128) and a small subset size for sub-sampling are capable of achieving a high performance on the anomaly detection. The author also indicated that a small sub-sampling size for iTree construction can reduce the swamping and masking degree.

There is various work in anomaly detection in network traffic monitoring. Brauckhoff et al. [25] focused on solving the problem of inaccurate anomaly detection caused by the preprocessing steps for the traffic data streams. The authors analysed the impact of the random packet sampling and temporal aggregation on the signal properties and the correlation structures of the captured traffic streams. They proposed a solution of using a low-pass filter to reduce the aliasing influence. Silveira et al. [189] introduced an traffic anomaly detection approach that is based on a statistical model, namely ASTUTE (A Short-Timescale Uncorrelated-Traffic Equilibrium), to detect strongly correlated flow changes, rather than based on the historical data sets. Khalid et al. [110] developed a motion learning system that models trajectories using DFT-based coefficient feature space representation. They proposed a m-mediods method to represent the class containing n members with m mediods, and provided four anomaly detection algorithms based on the m-mediods method.

Liu et al. [142] explored an approach of automatically identifying outliers in one-class classification problem. The proposed approach performs classification in an unsupervised way to deal with the cases that neither positive nor nega-

tive labels is known in advance. The optimization function for this one-class classification is defined as in formula 2.9.

$$min_{\alpha,\tilde{y}} Q(\alpha, \tilde{y}) = \alpha^T K(I + \gamma_1 L)K\alpha - 2\alpha^T K\tilde{y} \tag{2.9}$$

s.t. $\|\alpha\| = 1$, $\tilde{y} \in \{c^+ + \frac{\gamma_2}{\|\tilde{y}_+\|}, c^-\}^{n\times 1}$, where $\alpha = [\alpha_1, \cdots, \alpha_n]^T \in R^n$ is the coefficient vector; $K = [k(x_i, x_j)]_{1\leq i,j\leq n} \in R^{n\times n}$ is the kernel matrix; $L = D - W$ is the graph Laplacian matrix; $\tilde{y}$ is the new label assignment; $\|\tilde{y}\|_+$ represents the number of positive items in $\tilde{y}$; and $\gamma_1, \gamma_2 > 0$ are two trade-off parameters.

2.3 Motif Identification in Time Series

Fu [41] did a comprehensive survey on the time series mining work conducted before 2010. One section of this work [41] summarises the literature of finding the repeated patterns in time series. In the following, I conduct a further survey on the main techniques of the repeated pattern identification developed in 2010 - 2016.

2.3.1 Single-dimensional Time Series

Ye et al. [227] used a minimum description length (MDL) method to cluster subsequences of a time series, in which the length of each subsequence is measured by bit values. The MDL method is fault-tolerance and parameter-free.

Sart et al. [184] proposed to use hardware-based approaches to speed up the subsequence search of time series, which include Graphics Processing Unit (G-

PU) and Field Programmable Gate Array (FPGA) accelerating techniques that can speed up similarity search in two-order and four-order magnitudes respectively.

Zhao et al. [233] discussed the problem of improving the CRF-based (CRF: conditional random field) classification method by extracting features of time series, and applied the method to the IMU data to recognize the action motifs. The authors first discover motifs of IMU data and the discovered motifs are used as features to learn a Conditional Random Field classifier. In particularly, to discover motifs, the authors transformed the continuous time series to discrete time series by using Piecewise Aggregate Approximation [137] techniques, and then applied a random projection method to identify the repeated subsequences.

Berlin et al. [22] proposed a subsequence classification method for time series based on a dense motif discovery procedure. The authors used a suffix tree structure to accelerate the searching speed of motifs that are used as classification features. A bag-of-words classifier is then applied to recognize activities in the time series.

2.3.2 Multi-dimensional Time Series

Saria et al. [182] focused on finding deformable motifs in continuous time series by proposing a Continuous Shape Template Model (CSTM). They designed a hidden markov model to describe the temporal relations among motifs and random walk samples. Smooth continuous functions are adopted to represent individual motifs, which supports non-linear warp and scale transformations.

The authors used an unsupervised method to identify the motifs with various warps for unsegmented time series.

McGovern et al. [151] concentrated on discovering the ordered temporal motifs in multi-dimensional data streams. The authors proposed a motif identification method that first determines the salient data dimensions, and then identifies the motifs of each dimension and their temporal ordering for event prediction. The proposed method is applied to weather prediction by using a large number of numerically simulated super cell thunderstorms to identify precursor sets represented by the meteorological quantities in a temporal sequence.

Syed et al. [200] discussed the problem of finding predictive elements in physiological datasets based on a motif discovery methodology. The authors used randomized greedy algorithms like TCM and Gibbs sampling to identify motifs of datasets, and they applied a relaxed conservation view to deal with frequent noise.

CHAPTER 3

SUPERVISED ANOMALY DETECTION IN UNCERTAIN PSEUDO-PERIODIC DATA STREAMS

Uncertain data streams have been widely generated in many Web applications. The uncertainty in data streams makes anomaly detection from sensor data streams far more challenging. In this chapter, I present a novel framework that supports anomaly detection in uncertain data streams. The proposed framework adopts the wavelet soft-thresholding method to remove the noises or errors in data streams. Based on the refined data streams, I develop effective period pattern recognition and feature extraction techniques to improve the computational efficiency. I use classification methods for anomaly detection in the corrected data stream. I also empirically show that the proposed approach shows a high accuracy of anomaly detection on a number of real datasets.

3.1 Introduction

Data streams have been widely generated in many Web applications such as monitoring click streams [84], stock tickers [38, 236], sensor data streams and auction bidding patterns [15]. For example, in the applications of Web tracking and personalization, Web log entries or click-streams are typical data streams. Other traditional and emerging applications include wireless sensor networks (WSN) in which data streams collected from sensor networks are being posted directly to the Web. Typical applications comprise environment monitoring (with static sensor nodes) [7] and animal and object behaviour monitoring (with mobile sensor nodes), such as water pollution detection [89] based on water sensor data, agricultural management and cattle moving habits [44], and analysis

of trajectories of animals [83], vehicles [234] and fleets [122].

Anomaly detection is a typical example of a data streams application. Here, anomalies or outliers or exceptions often refer to the patterns in data streams that deviate expected normal behaviours. Thus, anomaly detection is a dynamic process of finding abnormal behaviours from given data streams. For example, in medical monitoring applications, a human electrocardiogram (ECG) (vital signs) and other treatments and measurements are typical data streams that appear in a form of periodic patterns. That is, the data present a repetitive pattern within a certain time interval. Such data streams are called pseudo periodic time series. In such applications, data arrives continuously and anomaly detection must detect suspicious behaviours from the streams such as abnormal ECG values, abnormal shapes or exceptional period changes.

Uncertainty in data streams makes the anomaly detection far more challenging than detecting anomalies from deterministic data. For example, uncertainties may result from missing points from a data stream, missing stream pieces, or measurement errors due to different reasons such as sensor failures and measurement errors from different types of sensor devices. This uncertainty may cause serious problems in data stream mining. For example, in an ECG data stream, if a sensor error is classified as abnormal heart beat signals, it may cause a serious misdiagnosis. Therefore, it is necessary to develop effective methods to distinguish uncertainties and anomalies, remove uncertainties, and finally find accurate anomalies.

There are a number of related research areas to sensor data stream mining, such as data streams compression, similarity measurement, indexing and querying mechanisms [59]. For example, to clean and remove uncertainty

from data, a method for compressing data streams was presented in [55]. This method uses some critical points in a data stream to represent the original stream. However, this method cannot compress uncertain data streams efficiently because such compression may result in an incorrect data stream approximation and it may remove useful information that can correct the error data.

This chapter focuses on anomaly detection in uncertain pseudo periodic time series. A pseudo periodic time series refers to a time-indexed data stream in which the data present a repetitive pattern within a certain time interval. However, the data may in fact show small changes between different time intervals. Although much work has been devoted to the analysis of pseudo periodic time series [103, 94], few of them focus on the identification and correction of uncertainties in this kind of data stream.

In order to deal with the issue of anomaly detection in uncertain data streams, I propose a supervised classification framework for detecting anomalies in uncertain pseudo periodic time series, which comprises four components: a uncertainty identification and correction component (UICC), a time series compression component (TSCC), a period segmentation and summarization component (PSSC), and a classification and anomaly detection component (CADC). First, UICC processes a time series to remove uncertainties from the time series. Then TSCC compresses the processed raw time series to an approximate time series. Afterwards the PSSC identifies the periodic patterns of the time series and extracts the most important features of each period, and finally the CADC detects anomalies based on the selected features. Our work has made the following distinctive contributions:

- I present a classification-based framework for anomaly detection in uncertain pseudo periodic time series, together with a novel set of techniques for segmenting and extracting the main features of a time series. The procedure of pre-processing uncertainties can reduce the noise of anomalies and improve the accuracy of anomaly detection. The time series segmentation and feature extraction techniques can improve the performance and time efficiency of classification.
- I propose the novel concept of a feature vector to capture the features of the turning points in a time series, and introduce a silhouette value based approach to identify the periodic points that can effectively segment the time series into a set of consecutive periods with similar patterns.
- I conduct an extensive experimental evaluation over a set of real time series data sets. Our experimental results show that the techniques I have developed outperform previous approaches in terms of accuracy of anomaly detection. In the experiment part of this chapter, I evaluate the proposed anomaly detection framework on ECG time series. However, due to the generic nature of features of pseudo periodic time series (e.g. similar shapes and intervals occur in a periodic manner), we believe that the proposed method can be widely applied to periodic time series mining in different areas.

The structure of this chapter is as follows: Section 3.2 introduces the related research work. Section 3.3 presents the problem definition and generally describes the proposed anomaly detection framework. Section 3.4 describes the anomaly detection framework in detail. Section 3.5 presents the experimental design and discusses the results. Finally, Section 3.6 concludes this chapter.

3.2 Related Work

I analyse the related research work from two dimensions: anomaly detection and uncertainty processing.

Anomaly detection in data streams: Anomaly detection in time series has various applications in wide area, such as intrusion detection [204], disease detection in medical sensor streams [148], and biosurveillance [185]. Zhang et al.[138] designed a Bayesian classifier model for identification of cerebral palsy by mining gait sensor data (stride length and cadence). In stock price time series, anomalies exist in a form of change points that reflect the abnormal behaviors in the stock market and often repeating motifs are of interest [220]. Detecting change points has significant implications for conducting intelligent trading [99]. Liu et al. [143] proposed an incremental algorithm that detects changes in streams of stock order numbers, in which a Poisson distribution is adopted to model the stock orders, and a maximum likelihood (ML) method is used to detect the distribution changes.

The segmentation of a time series refers to the approximation of the time series, which aims to reduce the time series dimensions while keeping its representative features [59]. One of the most popular segmentation techniques is the Piecewise Linear Approximation (PLA) based approach [108, 168], which splits a time series into segments and uses polynomial models to represent the segments. Xu et al. [222] improved the traditional PLA based techniques by guaranteeing an error bound on each data point to maximally compact time series. Daniel [126] introduced an adaptive time series summarization method that models each segment with various polynomial degrees. To emphasize the

significance of the newer information in a time series, Palpanas et al. [164] defined user-oriented amnesic functions for decreasing the confidence of older information continuously.

However, the approaches mentioned above are not designed to process and adapt to the area of pseudo periodic data streams. Detecting anomalies from periodic data streams has received considerable attention and several techniques have been proposed recently [65, 81, 128]. The existing techniques for anomaly detection adopt sliding windows [103, 82] to divide a time series into a set of equal-sized sub-sequences. However, this type of method may be vulnerable to tiny difference in time series because it cannot well distinguish the abnormal period and a normal period having small noisy data. In addition, as the length of periods is varying, it is difficult to capture the periodicity by using a fixed-size window [12]. Other examples of segmenting pseudo periods include an peak-point-based clustering method and valley-point-based method [94, 12]. These two methods may have very low accuracy when the processed time series have noisy peak points or have irregularly changed sub-sequences. Our proposed approach falls into the category of classification-based anomaly detection, which is proposed to overcome the challenge of anomaly detection in periodic data streams. In addition, our method is able to identify qualified segmentation and assign annotation to each segment to effectively support the anomaly detection in a pseudo periodic data streams.

Uncertainty processing in data streams: Most data streams coming from real-world sensor monitoring are inherently noisy and uncertain. A lot of work has concentrated on the modelling of uncertain data streams [5, 4, 127]. Dallachiesa et al.[47] surveyed recent similarity measurement techniques of uncer-

tain time series, and categorized these techniques into two groups: probability density function based methods [181] and repeated measurement methods [17]. Tran et al.[210] focused on the problem of relational query processing on uncertain data streams. However, previous work rarely focused on the detection and correction of the missing critical points for a discrete time series.

3.3 Problem Specification and prerequisites

In this section, I first give a formal definition of the problems and then describe the proposed framework of detecting abnormal signals in uncertain time series with pseudo periodic patterns. The symbols frequently used in this chapter are summarized in Table 3.1.

3.3.1 Problem definition

Definition 3.1. *A **time-series** TS is an ordered real sequence: $TS = (v_1, \cdots, v_n)$, where v_i, $i \in [1, n]$, is a point value on the time series at time t_i.*

I use the form $|TS|$ to represent the number of points in time series TS (i.e., $|TS| = n$). Based on the above definition, I define subsequence of a TS as below.

Definition 3.2. *For time series TS, if $SS (\subset TS)$ comprises m consecutive points: $SS = (v_{s_1}, \cdots, v_{s_m})$, I say that SS is a **subsequence** of TS with length m, represented as $SS \sqsubseteq TS$.*

Definition 3.3. *A **pseudo periodic time series** PTS is a time series $PTS = (v_1, v_2, \cdots, v_n)$, $\exists Q = \{v_{p_1}, \cdots, v_{p_k} | v_{p_i} \in PTS, i \in [1, k]\}$, that regularly separates PTS*

Table 3.1: Frequently Used Symbols

Symbols	Meaning
TS	A time series
p_i	The ith point in a TS
SS	A subsequence
PTS	A pseudo periodic time series
Q	A set of period points in a PTS
pd	A period in a PTS
CTS	A compressed PTS
vec_i	A feature vector of point p_i
$sil(p_i)$	Silhouette value of point p_i
$sim(p_i, p_j)$	Euclidean distance based similarity between points p_i and p_j
C	A set of clusters
$msil(C)$	Mean silhouette value of a cluster C
seg_i	A summary of a period
STS	A segmented CTS
A	A set of annotations
Lbs	A set of labels indicating the states
$lb_{(i)}$	The ith label in Lbs

on the condition that

1. $\forall i \in [1, k-2]$, *if* $\triangle_1 = |p_{i+1} - p_i|, \triangle_2 = |p_{i+2} - p_{i+1}|$, *then* $| \triangle_2 - \triangle_1 | \leq \xi_1$; *where* ξ_1 *is a small value.*

2. *let* $s_1 = (v_{p_i}, v_{(p_i)+1}, \cdots, v_{p_{i+1}}) \sqsubseteq PTS$, *and* $s_2 = (v_{p_{i+1}}, v_{(p_{i+1})+1}, \cdots, v_{p_{i+2}}) \sqsubseteq PTS$, *then* $dsim(s1, s2) \leq \xi_2$, *where* $dsim()$ *calculates the dis-similarity between* $s1$ *and*

*s*2, *and* ξ_2 *is a small value.* $dsim()$ *can be any dis-similarity measuring function between time series, e.g., Euclidean distance.*

In particular, $v_{p_{i+1}} \in Q$ *is called a period point.*

An uncertain PTS is a PTS having error detected data or missing points.

Definition 3.4. *If* $pd \sqsubseteq PTS$*, and* $pd = (v_{p_i}, v_{(p_i)+1}, \cdots, v_{p_{i+1}}), \forall v_{p_i} \in Q$*, then pd is called a* ***period*** *of the* PTS*.*

Definition 3.5. *A* ***normal pattern*** M *of a* PTS *is a model that uses a set of rules to describe a behaviour of a subsequence* SS*, where* $m = |SS|$ *and* $|SS| \in [1, |PTS|/2]$*. This behaviour indicates the normal situation of an event.*

Based on the above definitions, I describe types of anomalies that may occur in a PTS. There are two possible types of anomalies in a PTS: local anomalies and global anomalies Given the PTS in Definition 3.5, and a normal pattern $N = (v_1, \cdots, v_m) \sqsubseteq PTS$, a local anomaly ($L$) is defined as:

Definition 3.6. *Assume* $L = (v_{l_1}, \cdots, v_{l_n}) \sqsubseteq PTS$*, L is a local anomaly if either of the two conditions in Definition 3.5 is broken (condition 1 below), and at the same time another two conditions (conditions 2 and 3 below) are satisfied:*

1. $\triangle_N - \triangle_L > \xi_1$ *or* $dsim(N, L) > \xi_2$*;*
2. *frequency of L:* $freq(L) \ll freq(N)$ *and L does not happen in a regular sampling frequency.*
3. $|L| \ll |PTS|$.

Example 3.1. *Fig.3.1 shows two examples of pseudo periodic time series and their local anomalies. Fig.3.1(a) shows a premature ventricular contraction signal in an ECG*

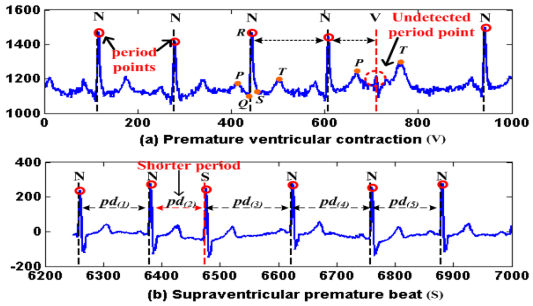


Figure 3.1: Two examples of local anomaly in ECG time series

stream. A premature ventricular contraction (PVC) [128] is perceived as a "skipped beat". It can be easily distinguished from a normal heart beat when detected by the electrocardiogram. From Fig.3.1(a), the QRS and T waves of a PVC (indicated by V) are very different from the normal QRS and T (indicated by N). Fig.3.1(b) presents an example of premature atrial contractions (PACs)[65]. A PAC is a premature heart beat that occurs earlier than the regular beat. If I use the highest peak points as the period points, then a segment between two peak points is a period. From Fig.3.1, the second period (a PAC) is clearly shorter than the other periods.

3.3.2 Wavelet-based error reduction

In the proposed framework, I use wavelet noise reduction method to reduce the white-noise in a time series obtained from the signal collection stage [3]. I briefly introduce this de-noising method in this subsection. The wavelet de-noising process contains the following three steps:

Step 1: wavelet signal decomposition. In this step, a time series is iteratively broken down to finer resolution signals in terms of frequencies. This decomposition process depends on two symmetric filters: low pass filter (LPF) and

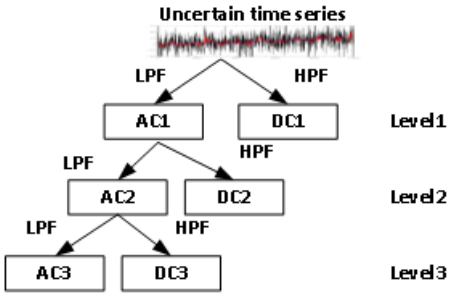


Figure 3.2: Iterative wavelet decomposition

high pass filter (HPF) that are both created from a *mother* wavelet. The LPF filters the low frequency signals (Approximate co-efficient) while the HPF keeps the high frequency signals (Detailed co-efficient). They are applied in a few number of iterative steps, which results in a tree structure with signals decomposed by different banks. The decomposition structure is shown in figure 3.2, where *AC* represents the 'approximate co-efficient' and *DC* means the 'detailed co-efficient'.

Step 2: noise reducing through soft-thresholding. The key step of noise-reducing is to find a noise-threshold that is used to distinguish the normal and noise signals. The soft-thresholding method proposed by Donoho[54] is applied to filter the noises in the high frequency signals (i.e., *DC* in figure 3.2), which is processed as: if the amplitude of a signal point (i.e., wavelet co-efficient) is smaller than a threshold value, the signal point is seen as a noise and is removed (i.e., its co-efficient is set to 0); or else, this point is treated as a normal waveform signal and its value is subtracted by the threshold. In this work, I mainly deal with the white gaussian noise whose threshold value is determined by formula 3.1.

$$t_n = \sigma \sqrt{2 \log n} \tag{3.1}$$

where σ is a noise standard deviation estimated based on the first-level signals with highest frequency (i.e., $DC1$ in figure 3.2), and n is the length of the time series.

Step 3: signal reconstructing. After noise reduction on each level, the remaining signal points are combined together in a bottom-up manner (from level3 to the root in figure 3.2) to obtain a filtered time series.

3.4 Anomaly Detection in Uncertain Periodic Time Series

The proposed framework comprises four main components: a signal noise reduction component (SNRC), a time series compression component (TSCC), a period segmentation and summarization component (PSSC), and a classification and anomaly detection component (CADC). I explain the process of anomaly detection of the proposed framework using an example of the dataset *mitdb*. Fig.3.3 shows the processing progress of *mitdb*. First, the uncertain *mitdb* time series is an input to the SNRC component. The TS1 in Fig.3.3 shows a subsequence of the raw *mitdb*. The SNRC removes the errors in *mitdb*, then the uncertain *mitdb* is transformed into a refined time series (TS2 in Fig.3.3). The TSCC component then further compresses the approximated *mitdb*. The TS3 in Fig.3.3 shows the compressed time series (CTS) that is a compression of the subsequence in TS2. The PSSC component segments the time series and assigns annotations to each segment. TS4 in Fig.3.3 shows the segmented and annotated CTS corresponding to the CTS in TS3. Finally, the CADC component learns

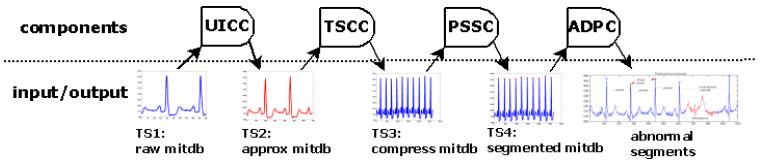


Figure 3.3: Workflow of the *mitdb* processing based on the proposed framework

a classification model based on the segmented CTS to detect abnormal subsequences in similar time series.

In the sequel of this section, I introduce the anomaly detection framework in detail.

3.4.1 Anomaly Detection in refined Time Series

The first step is to remove the noise in the uncertain time series (SNRC). I use the wavelet-based approach introduced in Section 3.3.2 to filter the errors obtained in the signal collection process. The refined time series is then processed for anomaly detection and normal pattern identification, which is based on the unit of *period*. Therefore, I need to identify period points Q that separate PTS into a set of periods. I use a clustering method to categorize the inflexions of a PTS into a number of clusters. Then a cluster quality validation mechanism is applied to validate the quality of each cluster. The cluster with the highest quality will be adopted as the period cluster, that is, the points in the period cluster will be the period points for the time series. The period points are the points that can regularly and consistently separate the PTS better than the points in the other clusters.

The cluster quality validation mechanism is a silhouette-value based

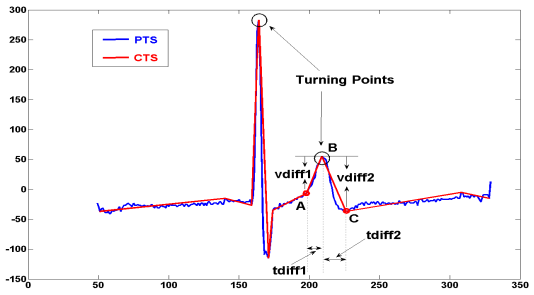


Figure 3.4: A PTS and one of its CTS s

method, in which the cluster that have highest mean silhouette value will be assumed to have the best clustering pattern. To accurately conduct clustering, I introduce a feature vector for each inflexion of PTS, with the optimal intention that each point can be distinguished with others efficiently.

Time Series Compression: TSCC

To save the storage space and improve the calculation efficiency, the raw PTS will first be compressed. In this work, I use the DouglasPeucker (DP) [90] algorithm to compress a PTS, which is defined as: (1) use line segment $\overline{p_1 p_n}$ to simplify the PTS; (2) find the farthest point p_f from $\overline{p_1 p_n}$; (3) if distance $d(p_f, \overline{p_1 p_n}) \leq \lambda$, where λ is a small value, and $\lambda \geq 0$, then the PTS can be simplified by $\overline{p_1 p_n}$, and this procedure is stopped; (4) otherwise, recursively simplify the subsequences $\{p_1, \cdots, p_f\}$ and $\{p_f, \cdots, p_n\}$ using steps $(1-3)$.

Definition 3.7. *Given a* $PTS = (v_1, \cdots, v_n)$, *a* ***compressed time series*** CTS *of* PTS *is represented as* $CTS = (v_{c_1}, \cdots, v_{c_n}) \subseteq PTS$, *where* $\forall p_{c_i} \in CTS$ *is an inflexion, and* $|CTS| \ll |PTS|$.

The feature vector of an inflexion is defined as:

Definition 3.8. *A **feature vector** for a point $p_i \in CTS$ is a four-value vector $vec_i = (vdiff1_i, vdiff2_i, tdiff1_i, tdiff2_i)$, where $vdiff1_i = v_{c_i} - v_{c_{i-1}}$, $vdiff2_i = v_{c_{i+1}} - v_{c_i}$, $tdiff1_i = c_i - c_{i-1}$, and $tdiff2_i = c_{i+1} - c_i$.*

Example 3.2. *Fig.3.4 shows an example of a PTS and one of its compressed time series CTS. The value differences $vdiff1$ and $vdiff2$, and the time differences $wdiff1$ and $wdiff2$ are shown in Fig.3.4.*

Period Segmentation and Summarization: PSSC

PSSC component identifies period points that separate the CTS into a series of periods, which is implemented by three steps: cluster points of CTS, evaluate the quality of clusters based on silhouette value, and Segment and annotate periods. Details of these steps are given below.

Step 1: Cluster Points of CTS Points are clustered into a number of clusters based on their feature vectors.

We use the Euclidean distance of feature vectors of two points to measure the point distance, which is defined as: given points p_1 and p_2, and their feature vectors $vec_1 = (vdiff1_1, vdiff2_1, tdiff1_1, tdiff2_1)$ and $vec_2 = (vdiff1_2, vdiff2_2, tdiff1_2, tdiff2_2)$, the distance between p_1 and p_2 is calculated by Equation 3.2

$$\begin{aligned} dis(p_1, p_2)^2 =& (vdiff1_1 - vdiff1_2)^2 + (vdiff2_1 - vdiff2_2)^2 + \\ & (tdiff1_1 - tdiff1_2)^2 + (tdiff2_1 - tdiff2_2)^2 \end{aligned} \tag{3.2}$$

In this work, I use k-means++ [16] clustering method to cluster points. It has

been validated that based on the proposed feature vector, the k-means++ is more accurate and less time-consumed than other clustering tools (e.g., k-means [88], Gaussian mixture models [174] and spectral clustering [161]). I give an brief introduction of the k-means++ in this section.

k-means++ is an improvement of k-means by first determining the initial clustering centres before conducting the k-means iteration process. k-means is a classical NP-hard clustering method. One of its drawbacks is the low clustering accuracy caused by randomly choosing the k starting points. The arbitrarily chosen initial clusters cannot guarantee a result converging to the global optimum all the time. k-means++ is proposed to solve this problem. K-mean++ chooses its first cluster center randomly, and each of the remaining ones is selected according to the probability of the point's squared distance to its closest centre point being proportional to the squared distances of the other points. The k-means++ algorithm has been proved to have a time complexity of $O(logk)$ and it is of high time efficiency by determining the initial seeding. For more details of k-means++, readers can refer to [16].

Step 2: Evaluate the quality of clusters based on silhouette value. I use the mean Silhouette value[177] of a cluster to evaluate the quality of a cluster. The silhouette value can interpret the overall efficiency of the applied clustering method and the quality of each cluster such as the tightness of a cluster and the similarity of the elements in a cluster. The silhouette value of a point belonging to a cluster is defined as:

Definition 3.9. *Let points in PTS be clustered into k clusters:* $C_{CTS} = \{C_1, \cdots, C_m, \cdots, C_k\}, k \leq |CTS|$. *For any point* $p_i = v_i \in C_m$, *the silhouette value*

of p_i is

$$sil(p_i) = \frac{b(p_i) - a(p_i)}{max\{a(p_i), b(p_i)\}} \tag{3.3}$$

where $a(p_i) = \frac{1}{M-1} \sum_{p_i, p_j \in C_m, i \neq j} sim(p_i, p_j)$, $M = |C_m|$ is the number of elements in cluster m; $b(p_i) = min(\frac{1}{M-1} \sum_{p_i \in C_m, p_j \in C_h, h \neq m} sim(p_i, p_j))$. $sim(p_i, p_j)$ represents the similarity between p_i and p_j.

In the above definition, $sim(p_i, p_j)$ can be calculated by any similarity calculation formula. In this work, I adopt the Euclidean Distance as similarity measure, i.e., $sim(p_i, p_j) = \sqrt{(v_i - v_j)^2 + (t_i - t_j)^2}$, where t_i and t_j are the time indexes of the points p_i and p_j. From the definition, $a(p_i)$ measures the dissimilarity degree between point p_i and the points in the same cluster, while $b(p_i)$ refers to the dissimilarity between p_i and the points in the other clusters. Therefore, a small $a(p_i)$ and a large $b(p_i)$ indicate a good clustering. As $-1 \leq sil(p_i) \leq 1$, a $sil(p_i) \to 1$ means that a point p_i is well clustered, while $sil(p_i) \to_+ 0$ represents the point is close to the boundary between clusters M and H, and $sil(p_i) < 0$ indicates that point p_i is close to the points in the neighbouring clusters rather than the points in cluster M.

The mean value of the silhouette values of points is used to evaluate the quality of the overall clustering result: $msil(C_{CTS}) = \frac{1}{|CTS|} \sum_{p_i \in CTS} sil(p_i)$. Similar to the silhouette value of a point, the $msil \to 1$ represents a better clustering.

After clustering, I need to choose a cluster in which the points will be used as period points for the CTS. The chosen cluster is called **period cluster**. The points in the period cluster are the most stable points that can regularly and consistently separate CTS. I use the mean silhouette value of each cluster to evaluate the efficiency of a single cluster, represented as $msil(C_m) = \sum_{p_i \in C_m} sil(p_i)$, where $-1 \leq msil(C_m) \leq 1$, and $msil(C_m) \to 1$ means the high quality of the cluster

Algorithm 1: Cluster quality validation

1: **procedure** CLUSTERQUALITY($V = \{vec_i | 1 \leq i \leq |CTS|\}, C_{CTS} = \{C_m | 1 \leq m \leq k\}, \eta, \xi$)
2: Calculate $sil(p_i)$
3: **for all do** $p_i \in CTS$
4: Calculate mean silhouette value: $msil(C_{CTS})$
5: **end for**
6: **if** $msil(C_{CTS}) < \eta$ **then**
7: $C_{perid} = NULL$
8: return
9: **end if**
10: $C_{perid} = max(msil(C_m))$
11: $msil(C_m) > \xi$
12: **for all do** $C_m \in C_{CTS}$
13: return C_{perid}
14: **end for**
15: **end procedure**

m. Based on the definition of silhouette values, I give Algorithm 1 of choosing period cluster from a clustering result. Algorithm 1 shows that if the mean silhouette value of the overall clustering result is less than a pre-defined threshold value η, then the clustering result is unqualified. Feature vectors of points need to be re-clustered with adjusted parameters, e.g., change the number of clusters. The previous line indicates that the chosen period cluster is the one with highest mean silhouette values that is higher than a threshold ξ.

Step 3. Segmentation and annotation of periods. As mentioned in the previous section, a CTS can be divided into a series of periods by using the period points. Thus detecting a local anomaly in CTS means to identify an abnormal period or periods. In this section, I introduce a segmenting approach to extract the main and common features of each period. The extracted information will be used as classification features that are used for model learning and anomaly detection. In addition, signal annotations (e.g., *'Normal'* and *'Abnormal'*) are attached to each period based on the original labels of the corresponding PTS. I will first give the concept of a summary of a period.

Definition 3.10. *Given a CTS that has been separated into D periods, a* ***summary*** *of a period* $pd_i = (v_{i_1}, \cdots, v_{i_m}), 1 \leq i \leq D$ *is a vector* $seg_i = (h_i^{min}, t_i^{min}, h_i^{max}, t_i^{max}, h_i^{mea}, p_i^{minmax}, p_i^l)$, *where* h_i^{min} *is the amplitude value of the point having minimum amplitude in period i:* $h_i^{min} = min\{v_{i_k}; 1 \leq k \leq m\}$; t_i^{min} *is the time index of the point with minimum amplitude. If there are two points having the minimum amplitude,* t_i^{min} *is the time index of the first point.* $h_i^{max} = max\{v_{i_k}\}$; t_i^{max} *is the first point with maximum amplitude;* $h_i^{mea} = \frac{1}{m}(\sum v_{i_k})$; $p_i^{minmax} = |t_i^{max} - t_i^{min}|$; $p_i^l = t_{i_m} - t_{i_1}$.

I represent the segmented CTS as $STS = \{seg_1, \cdots, seg_n\}$. Each period corresponds to an annotation *ann* indicating the state of the period. In this chapter, I will only consider two states: *normal* and *abnormal*. Therefore, a STS is always associated with a series of annotations $A_{STS} = \{ann_1, \cdots, ann_n\}$.

For the supervised pattern recognition model, the original PTS has a set of labels to indicate the states of the disjoint sub-sequences of PTS, which are represented as $Lbs = \{lb_{(1)}, \cdots, lb_{(w)}\}$, $\forall lb_{(r)} = \{'N'(Normal), 'Ab'(Abnormal)\}, 1 \leq r \leq w$. However, Lbs cannot be attached to the segmentations of the PTS directly because the periodic separation is independent from the labelling process. To

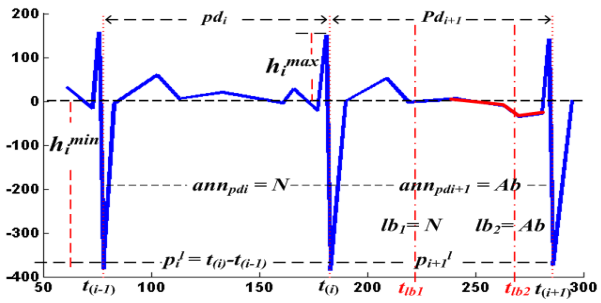


Figure 3.5: Segmentation and annotation of two periods

determine the state of a segmentation, I introduce a logical-multiplying relation of two signals:

Rule 1. *ann* = $\otimes('Ab','N') =' Ab'$ *and* *ann* = $\otimes('N','N') =' N'$.

Assume a period covers a subsequence that is labelled by two signals, if there exists an abnormal behaviour in the subsequence, then based on rule 1, the behaviour of the segmentation of the period is abnormal; otherwise the period is a normal series. This label assignment rule can be extended to multiple labels: given a set of labels $Lbs = \{lb_1, \cdots, lb_r\}$, if $\exists lb_j =' Ab', 1 \leq j \leq r$, the value of Lbs is $'Ab'$, represented as $lbs = \otimes(lb_1, \cdots, lb_r) =' Ab'$; if $\forall lb_j =' N'$, $lbs =' N'$.

According to the above discussion, the annotation of a period pd_i is determined by **Algorithm 2**.

Example 3.3. *I present the segmentation and annotation of a period in Fig.3.5 to explain their processes more clearly. Fig.3.5 shows that* pd_i *does not involve any label and the first label in* pd_{i+1} *is* $lb_1 = N$, *so* $lb_{pd_i} =' N'$. lb_2 *is* $'Ab'$, *hence* pd_{i+1} *is annotated as* $'Ab'$.

Classification and Anomaly detection Component: CADC

From Definition 3.10, each period of a PTS is summarised by seven features of the period: $(h_i^{min}, t_i^{min}, h_i^{max}, t_i^{max}, h_i^{mea}, p_i^{minmax}, p_i^l)$. Using these seven features to abstract a period can significantly reduce the computational complexity in a classification process. In the next section, I validate the proposed anomaly detection framework with various classification methods on the basis of different ECG datasets.

3.5 Experimental Evaluation

Our experiments are conducted in four steps. The first step is to remove the noises and compress the raw ECG time series by utilizing the DP algorithm, and to represent each inflexion in the perceived CTS as a feature vector (see Definition 3.8). Secondly, the K-means++ clustering algorithm is applied to the series of feature vectors of the CTS, and the clustering result is validated by silhouette values. Based on the mean silhouette value of each cluster, a period cluster is chosen and the CTS is periodically separated to a set of consistent segments. Thirdly, each segment is summarised by the seven features (see Definition 3.10). Finally, a normal pattern of the time series is constructed and anomalies are detected by utilizing classification tools on the basis of the seven features.

I validate the proposed framework on the basis of eight ECG datasets [75], which are summarised in Table 3.2 where 'V' represents Premature ventricular contraction, 'A': Atrial premature ventricular, and 'S': Supraventricular prema-

Table 3.2: ECG Datasets used in experiments

Datasets	Abbr.	#ofSamples	Types	#ofAbnor	#ofNor
AHA0001	ahadb	899750	V	115	2162
SA800	svdb	230400	S & V	75	1846
SCDH30	sddb	22099250	V	38	5743
MIT-BIH100	mitdb	650000	A & V	164	2526
MIT-BIH106	mitdb06	650000	A & V	34	2239
MGH/MF001	mgh	403560	S & V	23	776
MIT-BIHLT14046	ltdb	10828800	V	000	000
AFN04	aftdb	7680	NA	NA	NA

ture beat. Apart from the $aftdb$ dataset, each time series is separated into a series of subsequences that are labelled by the dataset provider. I give the number of abnormal subsequences ('#ofAbnor') and the number of normal subsequences ('#ofNor') of each time series in Table 3.2.

Our experiment is conducted on a 32-bit Windows system, with 3.2GHz CPU and 4GB RAM. The ECG datasets are downloaded to a local machine using the WFDB toolbox [188] for 32-bit MATLAB. I use the 10-fold cross validation method to process the datasets.

The metrics used for evaluating the final anomaly classification results include:

(1) Accuracy (acc): $(TP + TN)$ / Number of all classified samples;

(2) Sensitivity (sen): TP / $(TP + FN)$;

(3) Specificity (spe): TN / $(FP + TN)$;

(4) Prevalence (pre): TP / Number of all samples.

(5) Fmeasure (fmea): $2 * \frac{precision*recall}{precision+recall}$, where $recall = sen$, $precision = \frac{TP}{TP+FP}$
TP = true positive, TN = true negative, FP = false positive, and FN = false negative.

Details of the experiments are illustrated in the following sections.

3.5.1 Error Detection and Time Series Compression

At first, I design an experiment for noisy reduction in an uncertain time series. I use the synthetic uncertain data: I plant the additive Gaussian white noise to six time series in Table 3.2: ahadb, aftdb, sddb, svdb, mgh, and mitdb. The performance of the error reduction is evaluated by the mean squared error (MSN) between the six real time series and the synthetic uncertain time series. I use different signal-to-noise ratio ϵ (from 1 to 15) to see the change of the MSE value based on the wavelet de-noising approach. The experiment result is shown in Figure 3.6 (a). We can see that the MSE values of the six time series are decreasing from 0.25 to 0 when the value of the signal-to-noise ratio is increasing from 0 to 15.

The refined time series (whose errors have been reduced) will be compressed by DP algorithm. I use the approach proposed by the work of [176] to assess the stability of the DP compression algorithm under the variations of the change of the scale parameter and the perturbation of data. The former is measured by using a monotonicity index and the latter is quantified by a break-point stability index.

The monotonicity index is used to measure the monotonically decreasing

Table 3.3: Decreasing monotonicity degree of six datasets in terms of the value of λ

	ahadb	svdb	sddb	mitdb	mgh	aftdb
λ	100	100	100	100	100	100

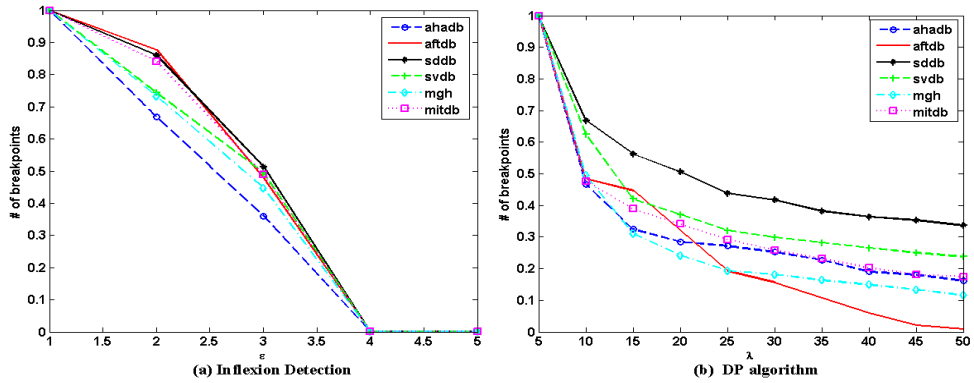


Figure 3.6: MSE of noise reduction and monotonically decreasing number of breakpoints in terms of λ of DP algorithm

or increasing trend of the number of break points when the values of scale parameters of a polygonal approximation algorithm are changed. For the DP algorithm, if the value of the scale parameter λ is increasing, the number of the produced breakpoints of the time series will be decreasing, and vice versa. The decreasing monotonicity index is defined as $M_D = (1 - \frac{T_+}{T_-}) \times 100$, and the increasing monotonicity index is $M_I = (1 - \frac{T_-}{T_+}) \times 100$, where $T_- = -\sum_{\forall \Delta v_i < 0} \Delta v_i / h_i$, $T_+ = \sum_{\forall \Delta v_i > 0} \Delta v_i / h_i$, and $h_i = \frac{v_i + v_{i-1}}{2}$. Both of M_D and M_I are in the range $[0, 100]$, and their perfect scores are 100. I test the decreasing monotonicity degrees for the datasets *ahadb*, *svdb*, *sddb*, *mitdb*, *mgh*, and *aftdb* in terms of different values of λ for DP algorithm. I set $\lambda = 5, 10, 15, 20, 25, 30, 35, 40, 45, 50$ to conduct DP compression. From Table 3.3 and Fig.3.6(b), we can see that the numbers of breakpoints are also 100% decreasing in terms of the increasing λ.

The break-point stability index is defined as the shifting degree of break-

Table 3.4: Endpoint stability of six datasets and pertubations

	ahadb	svdb	sddb	mitdb	mgh	aftdb
Shifting length	10000	10000	10000	10000	10000	100
S	100	99.8988	99.9955	99.9725	99.9348	99.9351

points when deleting increasing amounts from the beginning of a time series. We use the endpoint stability to test the breakpoint stability for fixed parameter settings : $\lambda = 10$ for the DP algorithm. The endpoint stability measurement is defined as $S = (1 - \frac{1}{m}\sum_d \sum_b \frac{s_b^d}{n_d l_d})$, where m is the level number of deletion, d is the dth level, s_b^d is the shifting pixels at breakpoint b, l_d is the length of the remaining time series and n_d is the number of breakpoints after the dth deletion. Table 3.4 shows the deletion length of each running circle and the stability degree of each time series. We iteratively delete 10000 samples from the beginning of the remaining *ahadb* time series, and conduct the DP algorithm based on the new time series. The positions of the identified breakpoints in each running circle are compared with the positions of the breakpoints identified in the whole *ahadb*. From Table 3.4, we can see that each time series is of high stability (i.e. values of S) when conducting the uncertainty detection procedure and the DP algorithm with fixed scale parameters.

3.5.2 Compressed Time Series Representation

Based on Fig.3.6, we set $\lambda = 10$ for time series compression. We then compare three methods of period point representation: (1) inflexions in CTS are represented by feature vectors (FV); (2) inflexions are represented by angles (Angle) of peak points [94]; (3) inflexions are represented by valley points (Valley) [12].

Valley points are points in a PTS, which have values less than an upper bound value (represented as U). U is initially specified by users and will be updated as time evolves. The update procedure is defined as $U_b = \alpha(\sum_{i=1}^{N} V_i)/N$, where N is the number of past valley points and α is an outlier control factor that is determined and adjusted by experts. As stated by Tang et al.[12], the best values of initial upper bound and α in ECG are $50mmHg$ and 1.1. The perceived feature vector sets, angle sets, and valley point sets are passed to the next step in which points are clustered and the period points of the CTS are identified. Each period is then segmented using the proposed segmentation method(see Definition 3.10). Finally, Linear Discriminant Analysis (LDA) and Naive Bayes(NB) classifiers are applied for sample classification and anomaly detection. Fig 3.7 shows the identified period points using the FV-based method for four datasets: *ltdb*, *sddb*, *svdb* and *ahadb*. From Fig 3.7, we can see that for each dataset, the FV-based method successfully identifies a set of periodic points that can separate the CTS in a stable and consistent manner.

Table 3.5 presents the silhouette values of clustering the inflexions in the $CTS\,s$ of seven time series, where column 'mean' refers to the mean silhouette value of a dataset clustering, and the values in columns c(luster)1-6 are the mean silhouette values of each cluster after clustering a dataset. 'NAs' in the sixth column means that the inflexions in the corresponding datasets are clustered into five groups, which present the best clustering performance in this dataset. From Definition 3.9, we know that if the silhouette values in a cluster is close to 1, the cluster includes a set of points having similar patterns. On the other hand, if the silhouette values in a cluster are significantly different from each other or have negative values, the points in the cluster have very different patterns with each other or they are more close to the points in other clusters. Table

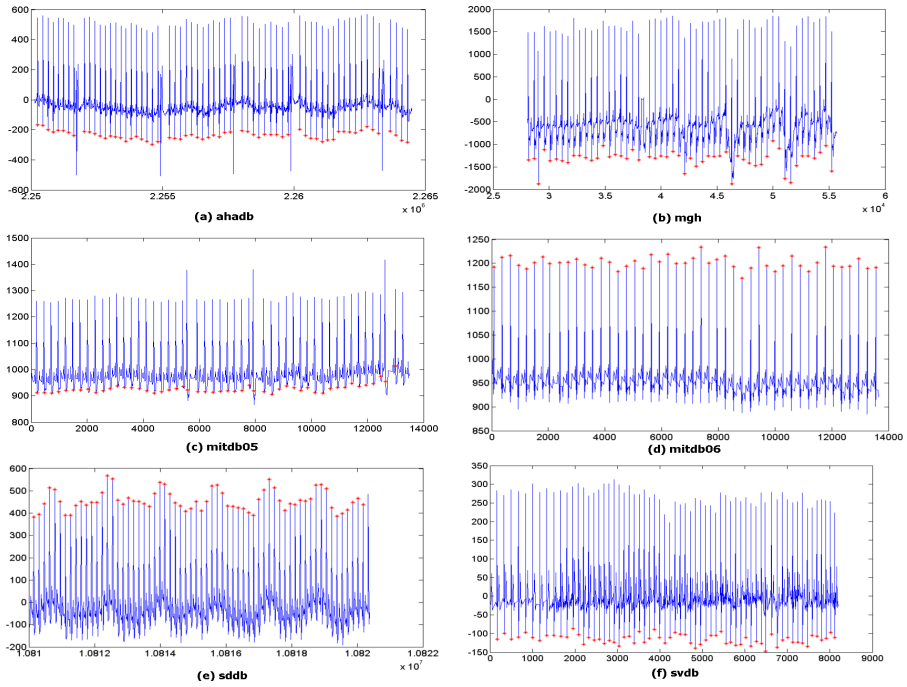


Figure 3.7: Period point identification of four datasets based on feature vectors

3.5 shows that for each of the seven datasets, the mean silhouette values of the overall clustering result and each of the individual clusters are higher than 0.4 ($\eta = 0.4$ in algorithm 1). The best silhouette value of an individual cluster in each dataset is close to or higher than 0.9 ($\xi = 0.8$ in Algorithm 1). In addition, for each dataset, I select the points in the cluster with highest silhouette value as the period points. For example, for dataset *ahadb*, points in cluster 4 are selected as period points.

Fig. 3.8 presents the silhouette values of clustering the inflexions in the CTS *s* of *mitdb* and *ltdb* time series. From this figure, we can see that for both the *mitdb* and *ltdb* datasets, FV-based clustering results in fewer negative silhouette values in all clusters, and he values in each cluster are more similar to each other compared with the angle-based clustering. We also come to a similar conclu-

Table 3.5: Silhouette values of six datasets

Dataset	Silhouette values						
	mean	cluster1 (c1)	c2	c3	c4	c5	c6
ahadb	0.8253	0.4479	0.8502	0.9824	0.9891	0.9381	NA
svdb	0.6941	0.9792	0.6551	0.9703	0.5463	0.5729	0.959
sddb	0.772	0.6888	0.5787	0.965	0.9727	0.6971	0.7529
mitdb	0.9373	0.9877	0.7442	0.9898	0.9711	0.5854	0.3754
mitdb06	0.7339	0.7317	0.8998	0.609	0.8577	0.8669	NA
ltdb	0.9149	0.9164	0.8381	0.9739	0.9079	0.8975	NA
mgh	0.8253	0.4479	0.8502	0.9824	0.9891	0.9381	NA

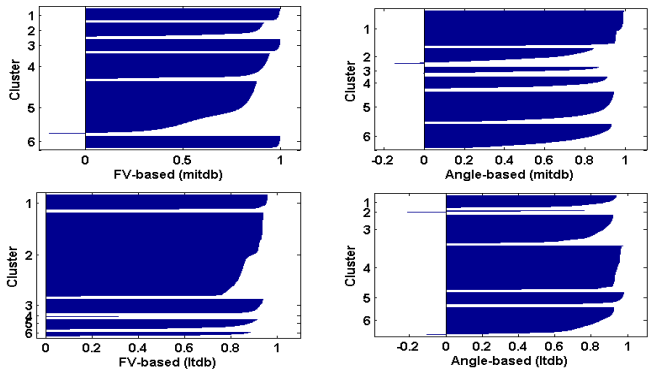


Figure 3.8: Silhouette value comparison between the feature vector based clustering method (FV-based) and the angle-based clustering method for the *mitdb* and *ltdb* datasets

sion by examining their mean silhouette values. The mean silhouette values of FV-based clustering for *mitdb* (corresponding to Fig.3.8(a)) is 0.9373, while the angle-based clustering (Fig.3.8(b)) is 0.7461; and the mean values for *ltdb* are 0.9149 and 0.8155 (Fig.3.8(c) and Fig.3.8(d)) respectively.

Fig.3.9 compares the average classification performance on the basis of four datasets using four classifiers: LDA, NB, Decision tree (DT), and AdaBoost (A-

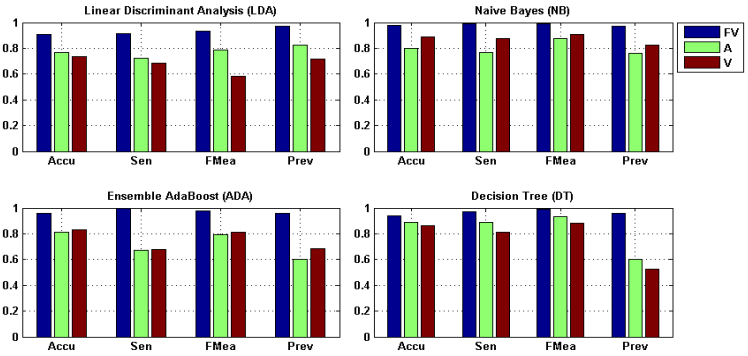


Figure 3.9: Average performance comparison of four classifiers (LDA, NB, ADA, DT) based on feature vector based (FV), angle based (A) and valley point based (V) periodic separating methods

da) with 100 ensemble members. From Fig.3.9, we can see that the classifiers based on the FV periodic separating method have the best performance in terms of the four datasets (i.e., the highest accuracy, sensitivity, f-measure, and prevalence). In the case of LDA and DT, the valley-based periodic separating method has the worst performance while in the cases of NB and Ada, valley-based methods perform better than angle-based methods.

3.5.3 Evaluation of Classification Based on Summarized Features

This section describes the experimental design and the performance evaluation of classification based on the summarized features. This experiment is conducted on seven datasets: *ahadb*, *svdb*, *sddb*, *mitdb*, *mitdb*06, *mgh*, and *ltdb*. From the previous subsections, I know that the seven time series have been compressed and the period segmenting points have been identified (see Table 3.5). The segments of each of the time series are classified by using three classification tools:

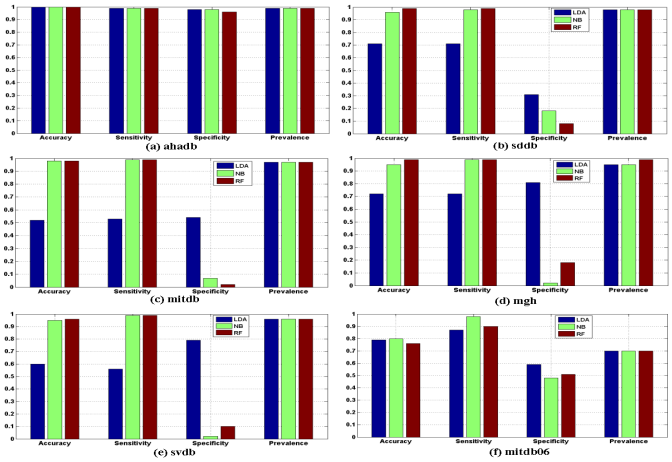


Figure 3.10: Classification performance of six datasets based on the summarized features using classification methods of LDA, RF, and NB

Random Forest with 100 trees (RF), LDA and NB. I use matrices of *acc*, *sen*, *spe*, and *pre* to validate the classification performance.

The classification performance is shown in Fig.3.10, which compares the performance of classification methods LDA, NB and RF, based on datasets (a) *ahadb*, (b) *sddb*, (c) *mitdb*, (d) *mgh*, (e) *svdb*, and (f) *mitdb06*. From the figure, I can see that for all six datasets, the performances of NB and RF are better than the performance of LDA based on the selected features. The accuracy and sensitivity of NB and RF are higher than 80% for each of the datasets. Their prevalence values are over 90% for the first five datasets (a-e). However, I can also see that the feature values of LDA are always higher than the feature values of the other two methods.

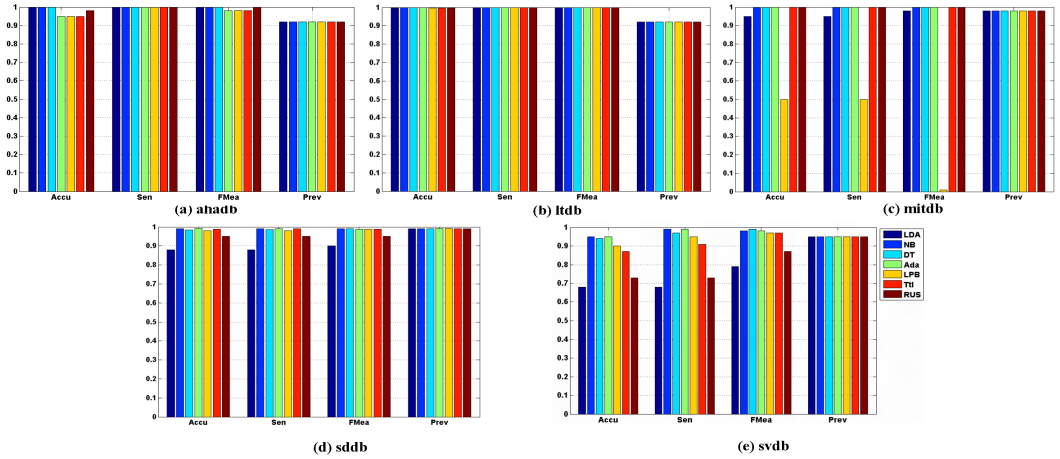


Figure 3.11: Performance of seven classifiers (LDA, NB, DT, Ada, LPB, T-tl, and RUS) based on the proposed period identification and segmentation methods on five datasets ((a) ahadb, (b) ltdb, (c) mitdb, (d) sddb, and (e) svdb)

3.5.4 Performance Evaluation of Other Classification Methods Based on Summarized Features

In this section, I design an experiment to evaluate the performance of the proposed time series segmentation method. Experimental results on the basis of five datasets (i.e.,*mitdb*, *ltdb*, *ahadb*, *sddb* and *svdb*) are presented in this section. I carry out the experiment by the following steps. First, the raw time series are compressed by DP algorithm and periodically separated by feature vector based period identification method. Second, each period is summarized by the proposed period summary method (see Definition 3.10) and is annotated by the annotation process(see Section 3.4.1). The classification methods used in this experiment include LDA, NB, DT, and a set of ensemble methods: AdaBoost (Ada), LPBoost (LPB), TotalBoost (Ttl), and RUSBoost (RUS). The classification performance is validated by five benchmarks: *acc*, *sen*, $fmea$, and *prev*.

Fig.3.11 shows the evaluated results of the classifier performance based on the proposed period identification and segmentation method. From Fig.3.11, I can see that the accuracy values of classification based on the 5 datasets are over 90%, except the cases of LPB with *mitdb*, LDA with *sddb*, LDA with *svdb*, and RUS with *svdb*. Some of them are of more than 98% accuracy. The sensitivity of classification based on the datasets of *ahadb*, *ltdb*, and *mitdb* are closing to 100%. The sensitivity based on the datasets of *sddb* and *svdb* are over 85%. The f-measure rates of classification based on *ahadb*, *ltdb*, *mitdb*, and *sddb* are higher than 95%. The f-measure rates of RUS and LDA based on *mitdb* and *svdb* are less than 80%, but the f-measure of other classifiers based on these two datasets are all higher than 80%, and some of them are closing to 100%. The prevalence rates of classification on the basis of the five datasets are over 90%.

Algorithm 2: Period annotation

procedure ANNOTATION($pd_i = (v_{i1}, \cdots, v_{im})$, $Lbs = (lb_1, \cdots, lb_r)$)
  $t_i^1 = NULL$
  $t_i^{end} = NULL$
  if $\exists lb_j$ that $t_{(i-1)1} \leq t_{j-1} \leq t_{(i-1)m} < t_{i1} \leq t_j \leq t_{im}$ **then**
    $t_i^1 = t_j$
  end if
  if $\exists lb_k$ & $t_{i1} \leq t_k \leq t_{im}$ & $t_{(i+1)1} \leq t_{k+1} \leq t_{(i+1)m}$ **then**
    $t_i^{end} = t_k$
  else if $t_i^1 \neq NULL \parallel t_i^{end} \neq NULL$ **then**
    if $t_i^1 = NULL$ **then**
      $t_i^1 =' N'$
    end if
    if $t_i^{end} = NULL$ **then**
      $t_i^{end} =' N'$
    end if
    $Lbs = Lbs\{t_i^1, \cdots, t_i^{end}\}$
    $lbs = \otimes(Lbs)$
  else
    $lbs = Lbs\{t_{i+1}^1\}$
  end if
  return annotated $pd_i^{'}$
end procedure

3.6 Summary

In this Chapter, I have introduced a framework of detecting anomalies in uncertain pseudo periodic time series. I formally define pseudo periodic time series (PTS) and identified three types of anomalies that may occur in a PTS. I focused on local anomaly detection in PTS by using classification tools. The uncertainties in a PTS are pre-processed by an inflexion detecting procedure. By conducting DP-based time series compression and feature summarization of each segment, the proposed approach significantly improves the time efficiency of time series processing and reduces the storage space of the data streams. One problem of the proposed framework is that the silhouette coefficient based clustering evaluation is a time consuming process. Though the compressed time series contains much fewer data points than the raw time series, it is necessary to develop a more efficient evaluation approach to find the optimal clusters of data stream inflexions. In the future, I am going to find a more time efficient way to recognize the patterns of a PTS. In addition, I will do more testing based on other datasets to further validate the performance of the method. Correcting false-detected inflexions and detecting global anomalies in an uncertain PTS will be the main target of our next research work.

CHAPTER 4

LISAM: FAST AND APPROXIMATE DISCOVERY OF DIFFERENT-LENGTH TIME SERIES MOTIFS

In the previous chapter, I introduced a supervised method for anomaly detection of data streams, and applied the method to ECG data sets. In this chapter, I explore two key problems in time series motif discovery: releasing the constraints of trivial matching between subsequences with different lengths and improving the time and space efficiency. The purpose of avoiding trivial matching is to avoid too much repetition between subsequences in calculating their similarities. I describe a limited-length enhanced suffix array based framework (LiSAM) to resolve the two problems. I convert the consistent time series to the discrete time series using the Symbolic Aggregate approXimation procedure, and introduce two covering relations of the discrete subsequences: α-covering between the instances of LCP (Longest Common Prefix) intervals and β-covering between LCP intervals to support the motif discovery: if an LCP interval is β-uncovered, its instances form a motif. The βUncover algorithm of LiSAM identifies the β-uncovered l-intervals, in which I introduce two LCP tabs: $presuf$ and $nextsuf$ to support the identification of the α-uncovered instances of an l-interval. I prove that in an extreme case that S has maximum LCP intervals, $O(N + n)$, while in the case an interval has maximum child intervals, $O(N + n^2)$, where N is the length of the raw time series T, and n is the length of the symbolized time series S. If $N \gg n$, the performance can be improved dramatically. In addition, it has linear space complexity $O(N)$. Experimental results indicate the accuracy of LiSAM on finding motifs with different lengths.

4.1 Introduction

Motifs of a time series are the frequently-occurred and approximately similar subsequences that can summarize the features of the time series [144]. Motifs have been applied in a variety of areas of time series processing, such as the anomaly detection in moving objects trajectories [131], the semantic analysis for the surgical sensor data streams [6], repeating pattern mining in audio streams [160], and human activity discovery [153]. Especially, it has been applied to the medical signals [102], like Electrocardiography(ECG) [133] and biological signals [145] for normal condition recognition and disease detection. I show examples of motifs in four different ECG time series in Fig. 4.1.

Discovering motifs for time series is an important and tough task. It has been proved that the subsequence clustering is meaningless in unsupervised data stream mining area, and the motif grouping in the discrete data stream mining has been applied as a replacement of the subsequence-clustering in the real time series [105]. In this chapter, I focus on two primary issues in the time series motif discovery: reducing the computational complexity and avoiding unexpected repetitions among different motifs and among instances of one motif. In an unsupervised context with little knowledge about the time series, it might be intractable to find all the motifs with different lengths by using exact and brute-force methods. There has been a series of work focusing on improving the time efficiency. One significant improvement is the method proposed by Minnen et al. [154], which has sub-quadratic time complexity in the time series length.

The subsequence trivial matching [105] and the overlapping among different motifs [155] are two types of motif repetition issues in the literature. To avoid

trivial matching, some methods assumed that the instances of a motif do not overlap with each other at all [154]. We believe that, however, a more flexible and user-manageable mechanism is necessary to control the numbers and styles of the discovered patterns.

I use an example to describe the main problems targeted by our work about different-length motif discovery. Fig. 4.2 shows a piece of time series and its four identified patterns: M1, M2, M3, and M4. From this figure, M1 happens two times, and M2 happens 4 times, where two instances of M2 are covered by the instances of M1, and M1 is a maximum pattern uncovered by any other patterns. M3 and M4 each occurs once, and they overlap with each other in the length oL. We have two problems about this figure:

problem 1 Given a trivial matching threshold, the first two instances of $M2$ trivially match the first two instances of $M1$. However, we can also see that $Freq(M1) < Freq(M2)$, then should M2 be considered as: (1) an independent motif including the two instances covered by M1 ?, (2) a motif without including the two covered instances, or (3) not a motif ?

problem 2 Given that M4 repeats 86 times in the overall time series, while M3 repeats 98 times, and assume that every instance of M4 is covered by an instance of M3, should we discard M3 and M4, and instead determine a maximum-length pattern M5 with less repeated times than M3 as a motif?

Enhancing the time and space complexity, and at the same time, guaranteing an expected accuracy is always one of the top topics in data processing. Some motif discovery researchers used approximate solutions to get an acceptable computational complexity [132]. In this work, I propose an unsupervised Limited-length suffix array based Motif Discovery algorithm (LiSAM) for con-

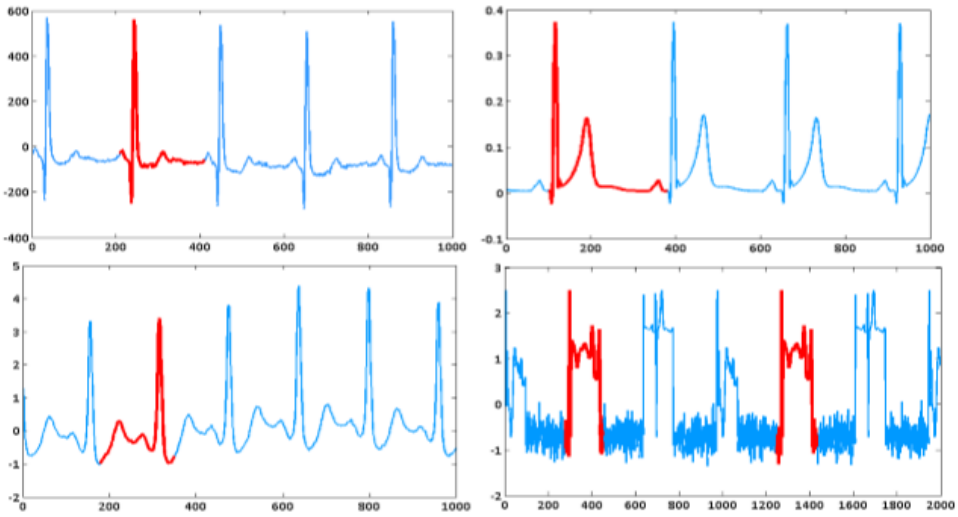


Figure 4.1: Examples of motifs in ECG time series

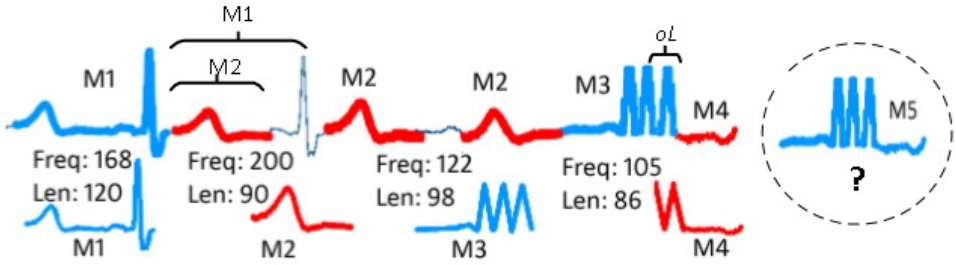


Figure 4.2: Problems in identifying different-length motifs in time series

tinuous time series, which is time and space efficient, and supports approximately discovering motifs in different lengths. I first convert the continuous time series to the discrete time series by using the Symbolic Aggregate approXimation procedure (SAX) [136], and then identify the different-length motifs based on the discrete time series. Our illustration of discrete motif discovery is on the basis of an exact substring matching procedure, however, we can easily embed the existing approximate substring matching methods, such as [95, 111], in LiSAM. That is, we use the exact subsequence grouping of discrete time series to discover the approximate patterns of continuous time series. We can also calculate the exact similarities between the instances of a continuous motif after

such an approximate grouping. The distinctive contribution of LiSAM are as below:

- *LiSAM* can discover motifs in different lengths (e.g., *maxLength* to *minLength* provided by users), avoid the unexpected trivial-matching by allowing user-defined overlapping degree (represented as α) between the instances of motifs, and support discovering motifs that overlap with each other in a specified degree (β). It can either be an automatic or semi-automatic algorithm by either manually setting all the parameters or by using default parameters (e.g., set $maxLength = \frac{1}{2}|T|$ (T is a time series), $minLength = 2$, $\alpha = 0$ and $\beta = 0$).
- *LiSAM* is both space and time efficient. It has linear space complexity $O(N)$. Existing approximate solutions [180, 135] applied the suffix tree to model the discrete time series to increase the searching speed of a subsequence, which consumes a large volume of storage space. Instead, I use a limited-length enhanced suffix array with linear space consumption to improve the space efficiency. In addition, in an extreme case that S has maximum LCP intervals, $O(LiSAM) = O(N + n)$, while in the case an interval has maximum child intervals, $O(LiSAM) = O(N + n^2)$, where N is the length of the raw time series T, and n is the length of the discrete time series S. If $N \gg n$, the performance can be improved dramatically.
- I conduct extensive experiments based on both synthetic time series datasets to evaluate the performance of *LiSAM*. Experimental results show the high accuracy of *LiSAM* and its applicability in the pattern recognition of data streams such as ECG.

4.2 Related Work and Background Knowledge

4.2.1 Related Work

The research on motif discovery in bio-informatics has been conducted for decades, which has derived a large number of popular techniques. For example, the MEME Suite [20] integrates a set of web-based tools for discovering motifs in bio-signals like DNA, RNA, and proteins. Jensen et al. [98] introduced a motif discovery algorithm Gemoda that can be applied to both categorical and continuous sequential data. Gemoda guarantees the discovered motifs having maximal composition and length, and support different similarity calculation metrics. Bandyopadhyay et al. [21] improved the time complexity of discovering (l, d)-motifs, which are to find a repeated string of length l by allowing minimum d mismatch. Grant et al. [80] developed a tool named Find Individual Motif Occurrences (FIMO) that supports motif discovery based on the position-specific scoring matrices.

There also has been a large amount of effort on exploring approximately accurate and fast motif discovery algorithms in continuous time series. Lin et al. [136] first introduced the concept of motif in discrete time series to continuous time series, and proposed the SAX (Symbolic Aggregate approXimation) method to symbolize the continuous time series. The SAX method can lower bound the distance between the original time series based on the symbolized time series. Because of its time efficiency, it supports a streaming time series conversion. Nguyen et al. [191] proposed a disk-efficient approximate motif discovery algorithm that is based on MP_C dimensionality reduction and Skyline Index. The authors stated that the time efficiency of their method is over

that of the random projection. To deal with big time series data, Sahli et al. [180] designed a method ACME that searches sequences in terms of the contiguous searching blocks, which can speed up the searching performance with more than 90% by using thousands of processors. Yankov et al. [226] focused on the problem of finding repeated patterns with variable length under uniform scaling. They also applied the proposed algorithm to a variety of domains such as detecting motifs in brain activities, capturing motions based on motifs, and finding projectile shapes. Floratou et al. [63] concentrated on improving the accuracy of motif discovery in continuous sequential data. They proposed a suffix tree based algorithm FLAME to find different motifs with high accuracy. As motif discovery is an unsupervised process, it is difficult to manually determine the lengths of the motifs in a time series. Against this problem, Yingchareonthawornchai [229] used a compression-based method to discover motifs with variable lengths. The proposed method also supports the motif evaluation and ranking in terms of their importance to the time series.

The exact motif discovery of continuous time series is normally untractable. Mueen et al. [158] designed a tractable exact algorithm to discover motifs based on an intuition of 'early abandoning', which shows high time efficiency compared with an brute-force exact algorithm. The MOEN algorithm [156] is another exact motif discovery method. It enumerates time series motifs at different lengths and is of high time and space efficiencies.

4.2.2 Background Knowledge

I briefly introduce the frequently used symbols and the basic concept of the enhanced suffix array in this section. Readers can refer to [1] for more details. I first introduce and list the symbols and their definitions in this chapter in Table 4.1.

Enhanced suffix array. A suffix array of S is an integer array (suftab) having values $k \in [0, n]$. An enhanced suffix array (ESA) is a suffix array with a number of additional supporting arrays, where two of them (lcptab and bwttab) will be used in this chapter. I use an example of $S_{examp} = aceaceacece$ to describe the ESA that is shown in Table.4.2. The $suftab$ keeps the starting positions of suffixes of S in ascending lexicographic order. The definition of $lcptab$ is in Table 4.1. From Table 4.2, $lcptab[0] = 0$ and $lcptab[n] = 0$.

To group the suffixes that have the longest common prefixes, the concept of LCP interval is proposed. I describe below the definition of an LCP interval from the work of [1].

Definition 4.1. *Given S and its Enhanced suffix array, an interval $[i, j]$ of index (see Table4.2), where $i, j \in [0, n]$ and $i < j$, is a LCP interval with LCP length ℓ if the following conditions are satisfied: (1) $lcptab[i] < \ell$; (2) $lcptab[k] \geq \ell, \forall k \in [i+1, j]$; (3) $lcptab[k] = \ell$ if $\exists k \in [i+1, j]$; (4) $lcptab[j+1] < \ell$. The LCP interval $[i, j]$ with LCP length ℓ can be represented as l_ℓ-$[i, j]$.*

An LCP interval tree indicates the embedding and enclosing relations [1] between LCP intervals. I describe an example of LCP tree of S_{examp} in Fig. 4.3. I can see that the root of the LCP tree covers all the suffixes of S_{examp}. The child intervals are the intervals embedded in their father intervals. The leaf intervals

Table 4.1: Symbols and Definitions

Concepts	Definitions
T	a continuous time series
Σ	a finite ordered alphabet
Σ^* & Σ^+	strings over Σ & $\Sigma^* \backslash null$
S	a discrete time series over Σ with length $\|S\| = n$
$\sim$	$\sim \in \Sigma$, $\sim \notin S$; $\sim > \sigma$, $\forall \sigma \in \Sigma$
$S[i, j]$	substring of S between positions i and j
$suftab$ (suf)	suffix array table of S
$presuf$ (pre)	the suffix index of the previous position of the current suffix in suftab
$nextsuf$ ($next$)	the suffix index of the next position of the current suffix in suftab
$S_{suftab[i]}, i \in [0, n]$	ith suffixes of S $\sim$
$lcptab[i]$	longest common prefix (LCP) of $S_{suf[i-1]}$ and $S_{suf[i]}$, $i \in [1, n]$
$bwttab[i]$ (bwt)	$= S[suftab[i] - 1]$, $\forall suf[i] > 0$; $= null$, if $suf[i] = 0$
l_ℓ-interval, l_ℓ-$[i, j]$	an LCP interval from LCP index i to index j with length ℓ
l-$[l, l]$, $l \in [0, n]$	singleton interval (SI) corresponding to $S_{suf[l]}$
NSI	non-singleton interval
m_ℓ-$[i, j]$	m-interval: instances of l_ℓ interval forming a motif

Table 4.2: Enhanced Suffix Array of S_{examp}

index	suf	lcptab	bwt	$S_{suf[i]}$
0	0	0	null	*aceaceacece* ~
1	3	6	e	*aceacece* ~
2	6	3	e	*acece* ~
3	1	0	a	*ceaceacece* ~
4	4	5	a	*ceacece* ~
5	7	2	a	*cece* ~
6	9	2	e	*ce* ~
7	2	0	c	*eaceacece* ~
8	5	4	c	*eacece* ~
9	8	0	c	*cece* ~
10	10	0	c	*e* ~
11	11	0	e	~

do not enclose any NSI. A fast traversing procedure for LCP trees is defined in [1]. Note that in this chapter I use l_ℓ to represent an l-interval with LCP length ℓ, while use m_ℓ to represent an motif interval (Def. 4.7) with LCP length ℓ. In addition, I refer the normal 'LCP intervals' to non-singleton intervals (NSIs).

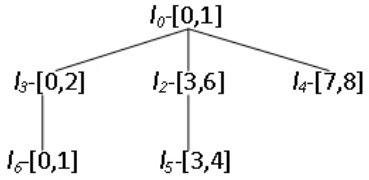


Figure 4.3: LCP tree of S_{examp}

4.3 Problem Definition

In this section, I introduce the basic concepts to be used in LiSAM. A continuous time series T is a sequence of real values that have temporal properties. To identify the motifs of a time series, previous work has given different forms of motif definitions [154, 157]. I summarize these definitions and present a comprehensive motif concept in Definition 4.2.

Definition 4.2. *A **motif** M of a time series T is a set of similar subsequences $SQ = \{sq_0, ..., sq_{n-1}\}$ such that $n \geq 2$, and $\forall i, j \in [0, n-1]$, the length of $|sq_i| \geq 2$, $|sq_i \bigcap sq_j| \leq o$, and $Dis(sq_i, sq_j) \leq d$, where o is an overlapping threshold to constraint the overlapping length in terms of the time period between two subsequences of M, Dis is a distance measure, and $d \geq 0$ is a small value to guarantee a certain similarity among subsequences. We call a subsequence of M as an **instance** of this motif.*

This definition requires that a motif subsequence should occur at least 2 times in a long time series, and each instance should have at least 2 continuous samples. Setting value o can avoid the trivial matching between the instances of a motif [157]. To approximately identify time series motifs, I first convert the real time series T to a discrete time series S by using SAX. I can then group the approximate similar subsequences of S. The discrete motif discovery is like a first-step clustering, which groups together the relatively similar subsequences

of T. In one symbol motif group, the real subsequences are most probably similar with each other depending on the SAX conversion procedure. In the following sections, the discussion on motif discovery is based on S.

For motifs in different lengths, one problem is that if some of the occurrences of a motif m_1 are subsequences of the occurrences of another motif m_2, is m_1 a motif independent with m_2? For example, given a time series *aaafaaa*, if we want a motif with *length* = 3, we can find pattern *aaa*. On the other hands, if we want a motif with uncertain length, e.g., $2 \leq length \leq 4$, we will find patterns *aa* and *aaa*. We can see that every occurrence of *aa* is a part of an occurrence of *aaa*, which means that identifying *aa* is meaningless because whenever *aa* occurs, there is one longer and more regular pattern *aaa* that can better capture the variation of the time series. To formulate a definition that can be applied to the different-length motifs, I introduce here the relation of *cover* among *l*-intervals and among the instances of *l*-intervals.

Definition 4.3. *Given two l-intervals l_{ℓ_1}-$[i_1, j_1]$ and l_{ℓ_2}-$[i_2, j_2]$, $s_{k_1}(k_1 \in [i_1, j_1])$ is an instance of l_{ℓ_1}, $s_{k_2}(k_2 \in [i_2, j_2])$ is an instance of l_{ℓ_2}, $sz_1 = |j_1 - i_1 + 1|$: (1) instance s_{k_1} is α-**covered** by s_{k_2} if $\ell_1 < \ell_2$, s_{k_1} overlaps with s_{k_2} at sub-string s'', where $s'' \subset s_{k_2}$ and $s'' \subset s_{k_1}$, and $|s''| > \alpha$, $|s_{k_1}| \geq \alpha \geq \frac{1}{2} * |s_{k_1}|$. Or else, s_{k_1} is α-**uncovered** by s_{k_2}; (2) Interval l_{ℓ_1} is β-**covered** by l_{ℓ_2}, if β instances of l_{ℓ_1} are α-covered by the instances of l_{ℓ_2}, where $(sz_1 - \beta) < \beta \leq sz_1$, and β is a pre-defined threshold. Or else, l_{ℓ_1} is β-**uncovered** (or uncovered) by l_{ℓ_2}.*

For example, given $\alpha = 2$ and $\beta = 2$, in Table 4.2, we can find *l*-intervals l_6-$[0, 1]$ and l_3-$[0, 2]$, where $s_{k_1} = ace$, $s_{k_2} = aceace$, $sz_1 = 3$. As $|s_{k_1}| > \alpha \geq \frac{1}{2}|s_{k_1}|$, and $s'' = ace$, $|s''| = 3$, then s_{k_1} is *α-covered* by s_{k_2}. As $(sz_1 - \beta) < \beta \leq sz_1$, s_{k_1} is *β-covered* by s_{k_2}.

From Definition 4.1, an l_ℓ-interval is composed of at least two suffixes that have the LCP of length ℓ. Therefore, an l-interval can be seen as a pattern of S, and the LCPs of the l-interval correspond to the occurrences of the pattern. A pattern of S is defined as:

Definition 4.4. *Given an alphabet set Σ and an approximate time series $S \in \Sigma^*$, a **pattern** of S is a time series pt that $1 \leq |pt| < \frac{1}{2}|S|$, $pt \subset S$, and occurs $k (k \geq 2)$ times in S at positions $\{p_1, ..., p_k\}$, $p_1 \neq ... \neq p_k$, where a position is the start point of an occurrence of pt in S.*

For example, the subsequence of l_3-$[0, 2]$, i.e. *ace*, is a pattern of S_{examp} in Table 4.2.

Definition 4.4 shows that a pattern of S has limited length ($|pt| < \frac{1}{2}|S|$). To adapt to the concept of limited-length patterns, I define a limited-length suffix array that will be used later for motif identification.

Definition 4.5. *Given a suffix sf of S, If each suffix in a suffix array has limited length maxL ($maxL < \frac{1}{2}|S|$), this suffix array is called a limited length suffix array.*

For example, Table 4.3 shows a limited length suffix array of S_{examp} with suffix length 6.

In this chapter, our discussion will be based on the limited length suffix array of S. From Definition 4.4, a pattern should occur at least twice in a time series. From the Definition 4.1, an l_ℓ-interval is composed of at least two suffixes that have the LCP of length ℓ. Therefore, an l-interval can be seen as a pattern of S, and the LCPs of the l-interval correspond to the occurrences of the pattern. However, the requirement on the minimum occurrence times of a pattern varies

in different situations. For example, in a very long S (e.g., $\geq$ 10 thousands), the element that repeats a small number of times (e.g., $<$ 10 times) is meaningless for the time series analysis. Therefore, I define a general concept of an approximate motif of discrete time series as below.

Definition 4.6. *Assume $u = S[a, b](a \leq b)$ is an instance of an l-interval l_ℓ-$[i, j]$ of S. Given a lower bound $\beta(\beta \geq 2)$ of the pattern occurrences, if $\varepsilon = j - i + 1 \geq \beta$, and l_ℓ is uncovered by any other l-intervals of S, it is an **approximate motif** of S, represented as $mf = \langle \ell; P = \{p_1, \cdots, p_\varepsilon\}\rangle$, where $\ell = b - a + 1(l \geq 1)$ is the **length** of mf, p_i is the start indexes of the occurrences of u in S, and ε is the **size** of mf.*

In the following, a motif of S refers to an approximate motif of T. The relation between an l-interval and a motif of S is defined as an m-interval.

Definition 4.7. *For an l-interval l_ℓ-$[i, j]$ of S, if the instances of l_ℓ is one-to-one matched to the occurrences of a motif $mf = \langle \ell; suftab[i], \cdots, suftab[j]\rangle$, then l_ℓ is an m-interval, represented as m_ℓ-$[i, j]$.*

Based on Definition 4.7, motifs and m-intervals have the following relation.

Lemma 4.1. *A motif of S corresponds to and only corresponds to one m-interval of S.*

Proof. Given a motif $mf_u = \langle \ell; P_u = \{p_1, \cdots, p_\varepsilon\}\rangle$ of S, as $\varepsilon \geq 2$, then the subsequence u occurs at least twice in S. Based on the definitions of LCP intervals and the suffix array, the suffixes $sf = \{S[p_1, \sim_1], ..., S[p_\varepsilon, \sim_\varepsilon]\}$ are in one LCP interval l_ℓ-$[i, j]$, where $p_1, ..., p_\epsilon \in [i, j]$, $\ell = |u|$ and $\sim_{\epsilon'} = p_{\epsilon'} + maxL - 1, \epsilon' \in [1, \epsilon]$. Assume (1) $\exists k, k \in [i, j]$ that $s_1 = S[suftab[k], suftab[k + \ell - 1]] = u$, but s_1 is not an occurrence of mf_u,i.e., $k \notin P_u$, which is opposite to the given condition that mf_u is a motif of S, because a motif needs to contain all the subsequences fitting

one pattern. Assume (2) $\exists p_x \in P_u$ but $p_x \notin [i, j]$, and $\exists p_y \in P_u$ and $p_y \in [i, j]$, then $(s_1 = S[suftab[p_x], suftab[p_x + \ell - 1]]) = u = (s_2 = S[suftab[p_y], suftab[p_y + \ell - 1]])$, that is, s_1 and s_2 are similar LCP and need to be in one LCP interval (suppose in l'_ℓ-$[i', j']$). As l_ℓ and l'_ℓ have one LCP u, they are the same l-interval, which is opposite to assumption (2). Lemma 4.1 is proved. □

In the following sections, I refer an m-interval to a motif.

4.4 Limited-length Suffix-array-based Motif Discovery

The Limited-length Suffix-Array-based Motif Discovery (LiSAM) Framework identifies motifs of S by determining the α-covering and β-covering degrees between instances of one l-interval and between different l-intervals respectively, which is based on a bottom-up traversing process of identifying LCP intervals of the enhanced suffix array [2]. The LiSAM is composed of two main algorithms: (1) βUncover (Alg. 3) determines whether or not an LCP interval is β-covered by other LCP intervals given a constraint β on the β-covering degree of a motif. From Definition 4.3, the determination of β-covering is based on the α-covering degree. To identify the α-covering relations between instances, part (2) αUncovered (Alg. 6) is described, which determines the nontrivial matching instances of an LCP interval given a constraint on the α-covering degree between motifs. If an l-interval is β-uncovered, the instances of this interval form a motif.

4.4.1 Identify β-uncovered l-intervals for Discrete Time Series

In this section, I first discuss the determination of the β-uncovered intervals with an assumption that $\alpha = 1$ for the α-covering relation between instances. In section 4.4.2, I introduce the α-covered algorithm and illustrate how to interactively perform the β- and α-uncovered algorithms to identify the motifs.

In ESA, identifying LCP intervals is a bottom-up traversing process. When an LCP interval is being processed, its child intervals have been identified, so the child intervals can support the determination of β-covering of the LCP interval. I distinguish the case of an LCP interval having a single character (the singleChar interval) with the case that the interval is comprised of more than one character(the multiChar interval). I give Lemma 4.2 to identify the β-uncovered multiChar intervals.

Lemma 4.2. *Given an multiChar LCP interval l_ℓ-$[i, j]$,its child intervals Θ, and the lower bound of the occurrence times of motifs $\beta \geq 2$, let $\lambda = j - i + 1$, l_ℓ is β-uncovered by other l-intervals if any of the following conditions is satisfied:*

1. $|\Theta| = 0$, $\lambda = \beta$ *and* $bwttab[i, j]$ *are pair-wise different, i.e.,* $bwttab[i'] \neq bwttab[j'], \forall i', j' \in [i, j]$ *and* $i' \neq j'$;
2. $|\Theta| = 0$, *and* $\exists \sigma_1 \neq ... \neq \sigma_\gamma, \sigma_{1,...,\gamma} \in bwttab[i...j], \beta < \gamma \leq \lambda$;
3. $|\Theta| > 0$, $\exists l_{\ell_1}$-$[w_1, z_1]$, $l_{\ell_1} \in \Theta$ *and* $\lambda_\theta = z_1 - w_1 + 1 \geq \beta$, *and* $\exists r_1...r_k \in [w_1, z_1]$ *and* $h_1...h_k \in [i, j]$ *but* $\notin [w_1, z_1]$ *that* $bwttab[r_1] \neq bwttab[h_1], ..., bwttab[r_k] \neq bwttab[h_k], k \geq \beta$.
4. $|\Theta| > 1$, $\exists m_{\ell_1}$-$[w_1, z_1], ..., m_{\ell_k}$-$[w_k, z_k] \in \Theta, k \geq \beta$, *and* $m_{\ell_1}, ..., m_{\ell_k}$ *are β-uncovered.*

Proof. 1. $|\Theta| = 0$, so the characters after the LCP subsequences of l_ℓ are pair-wise different, i.e., $S[suftab[i] + \ell] \neq S[suftab[j] + \ell]$. Meanwhile, $\lambda = \beta$ and $bwttab[i] \neq ... \neq bwttab[j]$. So the instances of l_ℓ are not covered by any longer repeated sequences in S. Hence, l_ℓ is β-uncovered.

2. if $\gamma > \beta$, then at least $\beta + 1$ characters in $bwttab[i, j]$ are different (assume $bwttab[k_1] \neq bwttab[k_2]$); and as $\Theta = 0$, the k_1th and k_2th LCP subsequences are not covered by any longer subsequences of its child intervals. So l_ℓ is β-uncovered.

3. assume l_ℓ have one child interval c_θ, where $\lambda_\theta \geq \beta$, $i \leq w_\theta \leq z_\theta \leq j$ and $\lambda > \beta$. (a)Assume $\lambda = \lambda_\theta$, then $l_\ell = c_\theta$, c_θ is not a child interval of l_ℓ. Assumption (a) is not true. (b) Assume $\lambda - \lambda_\theta < \beta$, then there are $\lambda - \beta$ instances of l_ℓ covered by the instances of c_θ, so interval l_ℓ is covered by interval c_θ, and l_ℓ is not a motif. Assumption (b) is not true. (c) as $\lambda - \lambda_\theta \geq \beta$, then there are at least β instances of l_ℓ that are not covered by the instances of c_θ. In addition, $\exists \sigma_1 \neq ... \neq \sigma_\gamma, \sigma_{1,...,\gamma} \in bwttab[i...j], \beta < \gamma \leq \lambda$, based on the proof of (3), l_ℓ is β-uncovered.

4. if $k = \beta$, as $m_{\ell_1}, ..., m_{\ell_k}$ are k motifs, the subsequences in all of the β intervals are pair-wise different, so the interval l_ℓ, where $\ell < \ell_1, ..., \ell_\beta$, cannot be covered by any of $\{m_{\ell_1}$ (as $\forall |m_{\ell_t}| \geq \beta, t \in [1, k], t \neq 1), ..., m_{\ell_\beta}\}$, that is, the interval l_ℓ cannot be individually covered by any of its k child motifs. So l_ℓ is β-uncovered.

□

For singleChar intervals, the problem of determining their motif property is to avoid finding a shorter singleChar motif β-covered by a longer singleChar motif. Lemma 4.3 shows how to determine if a singleChar interval

is β-uncovered.

Lemma 4.3. *Given a singleChar interval l_ℓ-$[i, j]$ that its LCP subsequence, i.e., $S[suftab[i], suftab[i] + \ell - 1]$, is only comprised of one character (assume σ),*

1. *if l_ℓ does not have child intervals, i.e., $|\Theta| = 0$ and $\exists \sigma_1 \neq ... \neq \sigma_\gamma, \sigma_{1,...,\gamma} \in bwttab[i...j], \beta + 1 \leq \gamma \leq \lambda$, then l_ℓ is β-uncovered;*
2. *if $|\Theta| > 0$ and l_θ-$[w, z] \in \Theta$, that $\exists \sigma_1 \neq ... \neq \sigma_\gamma \neq \sigma$ and $\sigma_{1,...,\gamma} \in bwttab[w...z]$, where $\gamma > 0$; and $\exists [w'...z'] \subset [i..j], z' - w' + 1 \geq \beta$, and $[w'...z']$ is β-uncovered by $[w...z]$, where $\exists \sigma'_1 \neq ... \neq \sigma'_{\gamma'} \neq \sigma$ and $\sigma'_{1,...,\gamma'} \in bwttab[w'...z']$, $\gamma' > 0$.*

Proof. 1. As l_ℓ does not have child intervals, l_ℓ cannot be covered by an interval comprising LCP subsequences of $u' = S[suftab[k_1], ..., suftab[k_1]+\ell'-1]$, where $k_1 \in [i, j], \ell' > \ell$. In addition, as $\exists \sigma_1 \neq ... \neq \sigma_\gamma, \sigma_{1,...,\gamma} \in bwttab[i...j], \beta + 1 \leq \gamma \leq \lambda (\lambda = j - i + 1)$, l_ℓ cannot be covered by an interval comprising LCP subsequences of $u'' = S[suftab[k_2] - 1, ..., suftab[k_2] - 1 + \ell'' - 1]$, where $k_2 \in [i, j], \ell'' > \ell$. So l_ℓ is β-uncovered.

2. Assume $u = S[suftab[i]..suftab[j] + \theta - 1]$ is the prefix of l_ℓ, and $u' = S[suftab[w]..suftab[w] + \theta - 1]$ is the prefix of l_θ, and assume $\exists \sigma_1 \in bwttab[w..z]$ and $\exists \sigma_2 \in bwttab[w', z']$ that $\sigma_1 \neq \sigma$ and $\sigma_2 \neq \sigma$, then (1) any child interval $_\theta$ cannot cover l_ℓ, since $z' - w' + 1 \geq 2$; (2) I prove that under condition 2 in Lemma 4.3, if l_ℓ is a singleChar interval with LCPs like $u = \underbrace{x...x}_{\ell}$, then $\nexists l_\theta$ (the strings of its *singleChar* LCP $u' = \underbrace{x...x}_{\theta}, (\theta > \ell)$) that cover l_ℓ. Assume exist such l_θ, then the strings of the LCP of l_θ include all the stings whose prefixes with length θ are u', i.e., $\exists k (= z - w + 1)$ subsequences $u \subset S$, and there must be $\eta (= k * (\theta - 1))$ *bwttabs* that $bwttab[r_1] = ... = bwttab[r_\eta] = \sigma$, $\eta = z' - w' + 1$ and $k + \eta = j - i + 1$;

which means there must not exist $\sigma'_{1,\ldots,\lambda} \neq \sigma, \lambda > 0$ in $bwttab[w', z']$. This is contradicting with condition 2 of Lemma4.3, so the second statement (2) is correct. Combining statements (1) and (2), the singleChar interval l_ℓ is β-uncovered given condition 2 of Lemma 4.3.

□

I use an example to explain Lemma 4.2 and Lemma 4.3. Table 4.3 shows a LiSA of '*aceaceaceceeeccccc*' with $maxL = 6$. If we set $\beta = 2$ and $\beta = 2$, we can get an *m*-interval l_6-$[0, 1]$ since $bwt[0] \neq bwt[1]$ (point 1 of Lemma 4.2); *l*-interval l_3-$[0, 2]$ is not an *m*-interval, as $bwt[1] = bwt[2]$, which means *ace* is β-covered by *eace* (point 2 of Lemma 4.2); *l*-interval l_3-$[3, 4]$ is an *m*-interval (point 1 of Lemma 4.3); and *l*-interval l_2-$[3, 5]$ is not an *m*-interval, as it is covered by l_3-$[3, 4]$ (point 2 of Lemma 4.3).

Based on Lemma 4.2 and 4.3, I design the procedure of determining an LCP interval being β-uncovered in Algorithm 3. The procedure *sinChar*() (line 3) determines whether l_ℓ is a singleChar interval: if it is singleChar, return the character, otherwise, return null. The singleChar status of *l*-intervals can be determined in the construction process of the suffix array. *countUniqChar*() (line 4, Alg.4) calculates the number of different characters in $bwttab[i, j]$. If l_ℓ does not have children (*cd*); it is a singleChar interval; and it has at least β different characters (*uc*) other than the *sc* character, then l_ℓ is a motif (lines 7-8 in Alg. 3, point 1 in Lemma 4.3). If l_ℓ is a multiChar interval ($sc == null$) with more than β unique characters, it is a motif (lines 7-8, point 2 in Lemma 4.2). For a multiChar interval, if it at least has β children that are motif intervals ($l_\ell.mcd.sz$), then it is a motif (lines 10-12, point 4 in Lemma4.2). For each interval, I can use a integer variable to keep the number of motifs of its children, and this

integer value can be determined during the suffix array building process. Lines 13-18 are based on point 3 in Lemma 4.2, where *cu* represents the current child interval; fcd is the first child interval; *nt* and *la* are respectively the next and last child intervals of *cu*; and *lb* and *rb* are the left and right boundaries of an *l*-interval. The procedure $difCharPair()$ (Line 14, Alg.5) compares a child interval (*cu*) of l_ℓ with the other part of l_ℓ (l'), where l' includes the intervals not covered by any child intervals of l_ℓ, and also the other child intervals apart from *cu*. If there are at least two pair of different character pairs in $bwttab[i, j]$, that is, $\exists i_1, i_2, i_3, i_4 \in [i, j]$ that $bwttab[i_1] \neq bwttab[i_2]$ and $bwttab[i_3] \neq bwttab[i_4]$, then $cp \leftarrow true$.

Algorithm 4 calculates the number of different characters in an *l*-interval given that the interval is singleChar or multiChar (based on the value of *sc*). In line 2, if l' does not have children, traverse the index tab (e.g., *i* in Table 4.3) of L-CP table from *w* to *z*, and count the index if $bwttab[x]$ is different with the other characters in $bwttab[i, j]$ and different with the *sc* character (see the procedure $addCnt(c, cx)$ in lines 24-29). The $addCnt()$ indicates that if l' is a multiChar interval ($sc = null$ and $c \neq sc$), or if it is a singleChar interval (without considering the indexes with $bwttab[x] = sc$, i.e., $c \neq sc$), and the current character has not happened in l' ($!l'.has(c)$), then count once and record the character in l'. Lines 7-20 count the pair-wise different characters in each child of l' and in the indexes uncovered by any of its child, where *la* is the child interval of l' before *cu*; *nt* is the child interval after *cu*; and *lb* and *rb* are left and right bounds of an interval. Lines 15 checks which character is in the current child interval *cu*. If *cu* is not a singleChar interval or the character *sc* is not counted($c \neq sc$ in line 15), and the character *c* occurs in *cu* but not in l', then this character *c* is counted (line 16), and is marked as *happened* in the interval l' (line 17).

Algorithm 3: Identify β-uncovered l-intervals

1: **procedure** βUNCOVERED(l_ℓ-[i, j], mi, ma, bwt, β)
2: $sz \leftarrow j - i + 1$
3: $sc \leftarrow sinChar(l_\ell)$
4: $uc \leftarrow countUniqChar(l_\ell, singleChar)$
5: **if** $sz == mt$ & $bwt[i, j]$ are pair-wise different **then**
6: return *true*
7: **else if** $l_\ell.cd == nul$ & $(sc \neq nul$ & $uc \geq \beta || uc > \beta)$ **then**
8: return *true*
9: **else**
10: **if** $l_\ell.mcd.sz \geq mt$ & $sc == null$ **then**
11: return *true*
12: **end if**
13: **for all** $cu \in l_\ell.cd$ **do**
14: $cp \leftarrow cu.difCharPair(cu, l')$
15: **if** cp || $cu == fcd$ & $i < l_\ell.fcd.lb$ & cp || $l_\ell.cu.rb + 1 < l_\ell.nt.lb$ & cp || $cu == la$ & $l_\ell.la.rb + 1 < l_\ell.rb$ & cp **then**
16: return *true*
17: **end if**
18: **end for**
19: **end if**
20: return $false$
21: **end procedure**

Algorithm 4: Count the number of different characters in an l-interval

```
1: procedure COUNTUNIQCHAR(l'-[w, z], sc)
2:     if l_ℓ'.children == null then
3:         for all x ∈ [w, z] do
4:             addCnt(bwt[x], x)
5:         end for
6:     else
7:         for all cu ∈ l_ℓ'.cd do
8:             for all c ∈ bwttab[la.rb + 1...l_ℓ'.rb] do
9:                 addCnt(c, cx)
10:            end for
11:            for all c ∈ bwttab[cu.rb + 1...nt.lb − 1] do
12:                addCnt(c, cx)
13:            end for
14:            for all c ∈ Sigma do
15:                if c ≠ sc & !l_ℓ'.has(c) & cu.has(c) then
16:                    l_ℓ'.cnt + +
17:                    l_ℓ'.has(bwt[cx]) = true
18:                end if
19:            end for
20:        end for
21:    end if
22:    return cnt
23: end procedure

24: procedure ADDCNT(c, cx)
25:    if c ≠ sc & !l_ℓ'.has(c) then
26:        l_ℓ'.cnt + +
27:        l_ℓ'.has(bwt[cx]) = true
28:    end if
29: end procedure
```

Algorithm 5 calculates the different character pairs between one child interval and the other part of l_ℓ. The inputs are two intervals (child intervals or l_ℓ's sub-intervals that are not covered by any child intervals of l_ℓ). l_ℓ is a motif if at least *mt* different character pairs (*cd*) are identified (line 8, point 4 in Lemma4.2); otherwise, count the next pair of characters in $l_{\ell 1}$ and $l_{\ell 2}$. In addition, if the current interval l_ℓ is a singleChar interval ($sc \neq nul$), and its two child intervals $l_{\ell 1}$ and $l_{\ell 2}$ both have at least 1 character that is different with the *sc* character, then l_ℓ is a motif (line 15-16, point 2 in Lemma4.3).

4.4.2 Identify α-uncovered Instances for Discrete Time Series

In section 4.3, I defined the concept of α-covering between instances of one interval. For example, in a time series $s = aceaceace$, if we expect a motif of length 6, we may get a motif with two instances: $\overbrace{ace\underbrace{aceace}_{\text{instance2}}}^{\text{instance1}}$, where $instance2$ 3-covers $instance1$. To control the α-covering degree, I introduce two tabs: $presuf$ and $nextsuf$ that respectively record the indexes of the previous suffix and the next suffix for the current suffix. An example of the two tabs is shown in Table 4.3. The values of *pre* and *next* can be determined during the process of building suffix arrays, so it does not take extra time. The *pre* of the $0th$ suffix is -1 and the *next* of the last suffix is $length(s)$.

Algorithm 6 shows how to identify the α-uncovered instances of an *m*-motif. In Algorithm 6, $\cap$ represents the overlapping part of two suffixes; $s[r..]$ represents the suffix starting from position r. If the index of the suffix (ntS) after the current suffix (suf) is in interval $[a, b]$, and the overlapping length between suffix $s[suftab[p]..]$ and suffix $s[suftab[ntS[p]]..]$ is less than the threshold val-

Algorithm 5: Count different character pairs between two intervals

```
1:  procedure DIFCHARPAIR(l_ℓ1, l_ℓ2)
2:      cd = 0
3:      for all c1 ∈ Sigma do
4:          for all c2 ∈ Sigma & l_ℓ1.has(c1) & c ≠ sc do
5:              if c2 ≠ sc & l_ℓ2.has(c2) & c1 ≠ c2 then
6:                  cd + +
7:              end if
8:              if cd ≥ mt then
9:                  return true
10:             else
11:                 break the inner for-loop
12:             end if
13:         end for
14:     end for
15:     if sc ≠ nul & countUniqChar(l_ℓ1, sc) > 0 & countUniqChar(l_ℓ2, sc) > 0 then
16:         return true
17:     end if
18:     return false
19: end procedure
```

ue α, then the position $suf[p]$ is recorded as a start position of an α-uncovered instance (lines 10-11). If the overlapping length between suffix $s[suf[p]..]$ and suffix $s[suf[ntS[p]]..]$ is over α, then continue checking the suffix after $ntS[p]$, until the checking step is over the *maD* (lines 8 to 15). As the LCP length of the current interval is ℓ, if an instance is *maD* far from the current instance, it is impossible that the two instances can α-cover each other. For each suffix, Algorithm 6 checks its α-covering instances by only iterating the suffixes from start positions afterwards. I temporally create an array ('*visited*' in lines 5, 6, 9) for the *m*-interval to record the visited status of each instance.

Algorithm 6 identifies α-uncovered instances given that the input interval is β-uncovered in terms of $\alpha = 1$. We can also interactively perform the algorithms βUncover and αUncover to determine the β-uncovered motifs in terms of different values of α by using the tab *pre*: in the process of βUncover, for each instance of an *l*-interval, we check both its pre- (i.e. suffixes with prior starting positions) and afterwards-suffixes simultaneously by using the chain-procedure of Algorithm 6(for pre-suffixes, *next* can be simply replaced by *pre*). Specifically, we check each line in $[i, j]$ when $bwttab[i..j]$ is traversed in line 5 of Alg. 3, and determine whether this instance is overlapping with its previous instances *pre*. Remove it if it is overlapped with *pre*. In addition, in Alg. 4, we can check each position in $[i, j]$ in lines 3, 8 and 11, and remove this position if it is overlapped with its previous instance. At last, only the instances that are not overlapping with each other are used to decide if the current *l*-interval is βUncovered.

Algorithm 6: Identify αUncovered l-intervals

```
1:  procedure αUNCOVERED(α, m_ℓ-[a, b])
2:      maD = ℓ − α + 1
3:      for all p ∈ [a, b] do
4:          if m_ℓ.visited then
5:              P.add(p)
6:              m_ℓ.visited = true
7:          end if
8:          while q ≤ maD do
9:              if ntS[p] ∈ [a, b] & |s[suf[p]..] ∩ s[suf[ntS[p]]..]| < α then
10:                 P.add(p)
11:                 m_ℓ[p].visited = true
12:             else if |s[suf[p]..] ∩ s[suf[ntS[p]]..]| ≥ α then
13:                 m_ℓ[p].visited = true
14:             end if
15:         end while
16:     end for
17:     return P
18: end procedure
```

4.4.3 Real Motif Identification for Continuous Time Series

After finding the βUncovered m-intervals for discrete time series, we need to cluster again the instances in one interval given the distance threshold d and based on the Euclidean distance between the real subsequences the instances refer to. As the current instance group is much smaller than the group of all

time series subsequences, and the instances in this group are already very similar to each other after conducting the 'βUncover' grouping, this second-time instance clustering process would be much faster than directly conducting the real subsequence clustering. We can use the classical clustering methods like Hierarchic or k-means. To improve the distance calculation efficiency, a number of pruning procedures can be applied. For example, a simple one is to only cluster the middle points of the instances. The 'early abandoning' [158] is another procedure that stops summing up the squared differences between two instances when the cumulative sum becomes larger than the distance threshold d, which avoids calculating the whole time series to improve the speed.

4.5 Performance Evaluation and Complexity Analysis

In this section, I present the experimental results to show the efficiency of LiSAM. I insert patterns to random time series generated by gaussian white noise, and quantitatively measure the algorithm performance on the simulated data sets, in terms of the overlapping degree between the planted pattern and the discovered pattern of a time series (represented as *old*). In addition, time and space complexities of the proposed algorithm are analyzed. Our experiments are conducted on a windows 64-bit system with 3.2GHz CPU and 4 GB RAM, and is implemented by *Java*.

4.5.1 Accuracy and Inner Quality of Motifs

I extract patterns from six different ECG data streams, repeat each pattern 30 times and insert the repeated patterns to Gaussian white noise data streams separately. The information of the extracted patterns and the parameter settings is shown in Table 4.4. The first three datasets are from the UCR Time Series Classification Archive [40], and the other three are from the Physionet [75]. Particularly, the *nL* is the length of a piece of noise subsequence between two pieces of a pattern. I use the fixed-length intervals (i.e., length of noise subsequences) between two pattern subsequences to make the annotation of the pattern instances easy. Column *sL* sets the parameters of the SAX-based symbol conversion, representing the length of a subsequence that corresponds to a symbol. Columns *maxM* set the upper bounds of the lengths of the discovered patterns. The lower bounds of the lengths of the discovered patterns for all datasets are set as 10.

I use *old* to measure the accuracy of the discovered motifs, which represents the overlapping degree between the inserted pattern (p_i) and the discovered pattern (d_j): $old = \frac{\sum_i \sum_j overlap(p_i,d_j)}{length(plantedPattern)}$. The *old* values for each of the simulated ECG time series are shown in Table 4.4. I can see that the proposed motif discovery algorithm can identify the inserted patterns with very high accuracy (all over 0.9). I compare the shapes of the planted patterns and the discovered motifs in each of the six time series in Fig. 4.4. In addition, I use the average pair-wise distances among instances (represented as *inDis*) of a motif to measure the dissimilarity degree of the instances of one discovered motif (e.g., motif m), which is calculated as: $inDis(m) = \frac{\sum_{i,j} dis(m_i,m_j)}{m.len*m.size}$, where m_i and m_j represent the *ith* and *jth* instances of *m*; and *m.len* is the length of this motif; *m.size* is the number of its instances, and *dis* is the Euclidean distance function. The average *inDis* value

Table 4.3: Pre and Nextsuf

i	pre	next	suf	bwt	sel.
0	-1	6	0	-	aceace
1	11	7	6	e	aceace
2	12	8	3	e	acecee
3	13	4	0	e	cccc~
4	3	5	3	c	ccc~
5	4	10	2	c	cc~
6	0	11	1	a	ceacea
7	1	12	5	a	ceacec
8	2	14	2	a	ceceee
9	14	16	2	e	ceeecc
10	5	17	1	c	c~
11	6	1	0	c	eaceac
12	7	2	4	c	eacece
13	15	3	1	e	ecccc~
14	8	9	2	c	eceeec
15	16	13	1	e	eecccc
16	9	15	2	c	eeeccc
17	10	18	0	c	~

Table 4.4: Parameter Settings of Datasets

Datasets	nL	sL	maxM
ECG200	50	2	100
ECGfivedays	50	2	140
ECGtorse	100	10	1640
ECGtwa01	150	3	300
ECGsvdb800	150	2	170
ECGmitdb100	150	2	150
LTDB14134	-	2	150
SVDB800	-	2	150
AHADB0001	-	2	120
CARTI01	-	2	100

of each time series is shown in Table 4.4, and the distance distribution of each instance pair of the most frequent motif for each dataset is shown in Fig. 4.5. I can see that the instances of one motif for each datasets are very close to each other, all of which have less than 0.1 average instance dis-similarities.

Based on the settings shown in Table 4.4, I validate the performance of OLD, InDis, and shows the identified Number of Motifs (NOM) in terms of a range of β values. The validation results are shown in Table 4.5. I can see that for each Datasets, when the value of β is in a certain range, the highest performance of OLD and InDis, and the number of NOM do not change. For example, for

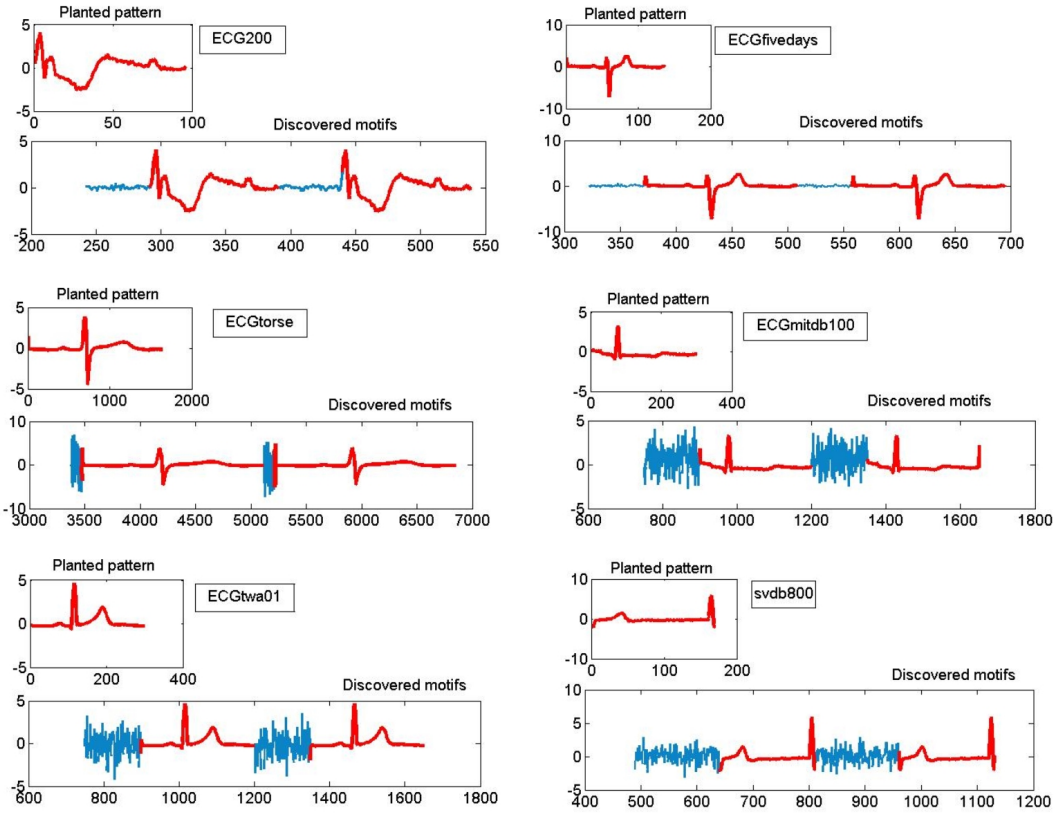


Figure 4.4: Planted patterns and discovered motifs

dataset ECG200, when $\beta \in [2, 100]$, $OLD_{max} = 0.99$, $InDis_{min} = 0.0076$, and 46 motifs are identified; whereas, when $\beta > 100$, the LiSAM cannot find motifs, and the performance of OLD and InDis is worst. I can see from the performance of ECGmitdb100, the NOM changes when the β changes, while the highest performance of OLD and InDis keeps at a certain level, which means the change of β and NOM does not influence the highest performance of the identified motifs.

4.5.2 Pattern Discovery on Real Datasets

I use the proposed SAMOF algorithm to identify the most frequent patterns in four real ECG datasets: the MIT-BIH Long Term Database (LTDB), the

Table 4.5: OLD, InDis, and NOM in terms of β

Datasets	β	OLD	InDis	NOM
ECG200	$2 \sim 100$	0.9892	0.0076	46
	> 100	0	1	0
ECGfivedays	$2 \sim 98$	0.9923	0.0041	39
	$99 \sim 200$	0.9923	0.0041	1
	> 200	0	1	0
ECGtorse	$2 \sim 12$	0.9939	0	4
	> 12	0	1	0
ECGtwa01	$2 \sim 99$	0.9933	0.0086	5
	> 12	0	1	0
ECGsvdb800	2	0.9923	0.0056	2
	$3, 4$	0.9923	0.0134	1
	> 4	0	1	0
ECGmitdb100	2	0.9966	0.0038	3
	3	0.9966	0.007	2
	4	0.9966	0.007	1
	> 4	0	1	0

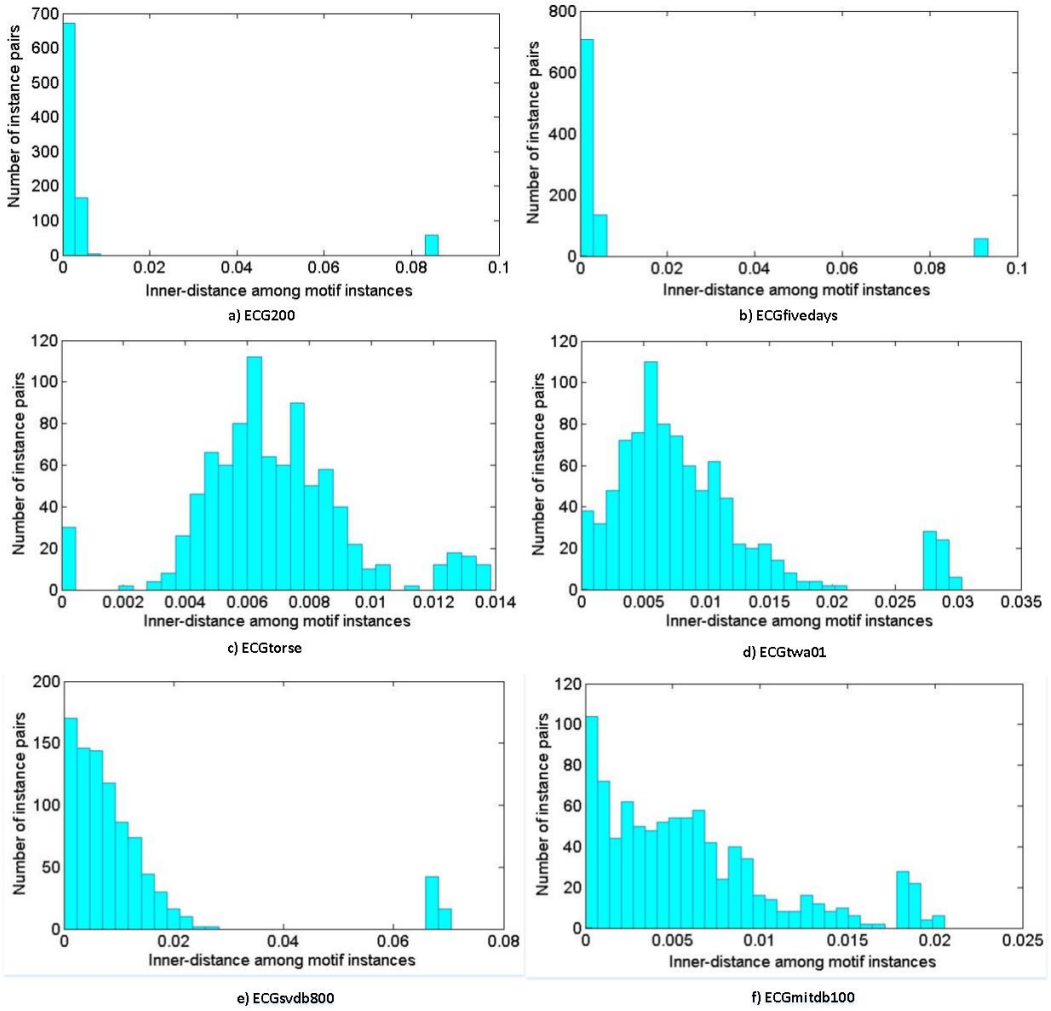


Figure 4.5: Distance distributions of instance pairs of the most frequent motif for six datasets

Supraventricular Arrhythmia Database (SVDB), the American Heart Association Database (AHADB), and the St. Petersburg INCART Arrhythmia Database (CART) [75]. Their information is listed in the bottom part of Table 4.4. For each dataset, I conduct pattern recognition in the first 30000 samples (1:30000). I discover the most frequent motifs for each datasets, and present the motifs in Fig. 4.6.

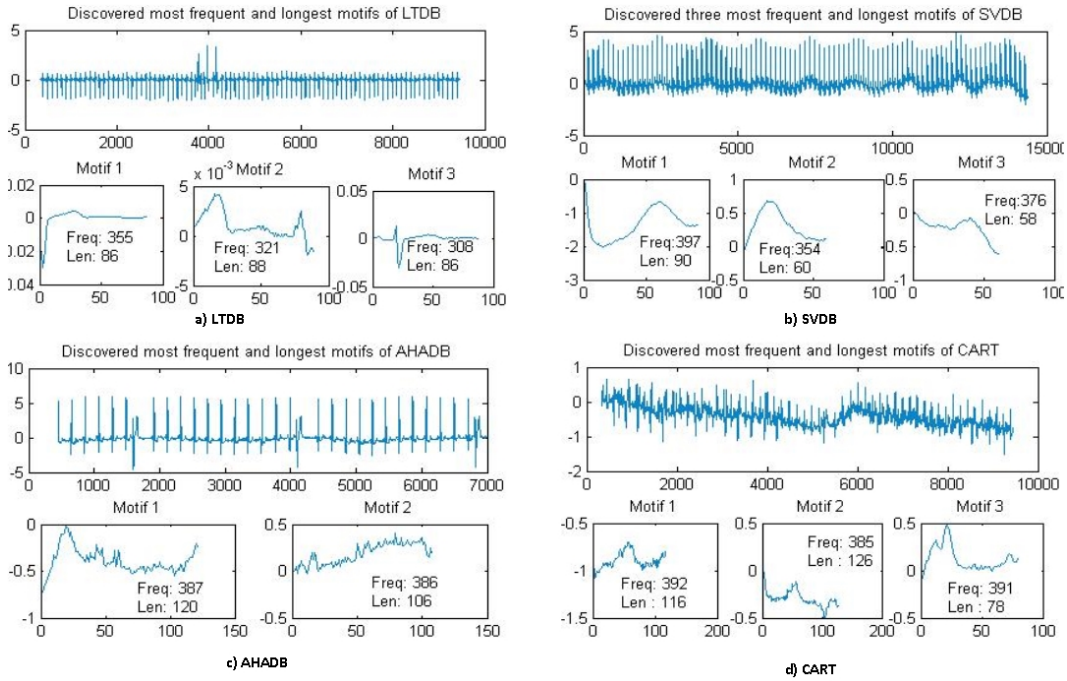


Figure 4.6: Discovered most frequent motifs of four real datasets

4.5.3 Time and Space Complexity Analysis

Time Complexity

The LiSAM mainly contains three steps: 1) discrete the time series based on SAX; 2) establish suffix array for the discrete time series and traverse the suffix array to find the LCP intervals; 3) determine the β-uncovered l-intervals.

If the length of a time series is N, the first step of time series discretion takes ON time. After discretion, if there are n symbols, the maximum time taken to build and traverse the suffix array (step 2) is $n + n = 2n$. The main part of Step 3 is the process of Algorithm 3.

For an LCP interval $l_\ell - [i, j]$, the *hasSingleChar* function can be implemented during the suffix array construction process (line 3 in Alg.3). *countUniqChar*()

function (line 7 in Alg.3, and Alg.4) takes maximum time $z \times r$, where $sz = |w - z|$ is the size of a child interval of l_ℓ, and r is the number of symbols in Σ. The three 'for-loop's in lines 6-8 in Alg.4 are actually a traverse of the index tab in $[w, z]$. As r is a constant normally less than 10, the $O(Alg.4)) = O(sz)$. The time complexity of function *compInterv*() depends on the interval size sz and the complexity of *countUniqChar*(), so it is $O(sz)$. Then the worst time complexity of LiSAM is $O(LiSAM) = O(N + m_0 \times sz_0 + m_1 \times sz_1 + \cdots + m_K \times sz_K)$.

I may intuitively believe that the worst time complexity of LiSAM is $O(N + n^3)$. However, the values of K, m, and sz are interrelated with each other to influence the $O(LiSAM)$. Lemma 4.4 gives their relations. I always exclude singleton intervals SI whenever I mention the *l*-intervals and their child intervals.

Lemma 4.4. *Given a discrete time series S with length n, and an LCP tree LT of S,*

1. *S has maximum $n - 1$ l-intervals, i.e., $max(K) = n - 1$, each LCP interval has at most 2 child non-singleton intervals (abbr. NSI), i.e., $m \leq 2$, and the $max(sz) = n - 1$.*
2. *S has minimum 1 l-interval (i.e., the root interval l-[0..n-1]) that has 0 child NSI. Other than this case, the number of l-intervals of an LCP tree is a decreasing function of the child number of each LCP interval. That is, $K = f(\frac{1}{c_k})$, where c_k is the child number of the kth interval.*
3. *given an l_ℓ-$[i, j]$, the number of its child intervals m is a decreasing function for the sizes of its child intervals: $m = f(\frac{1}{sz})$,*

Proof. I describe the problem of counting the LCP intervals as a problem of picking up elements from a set (see Fig. 4.7). There are n sequential elements in S. Each time I remove any two adjacent elements (e.g., e_i, e_{i+1} in S_n) from S and

combine these two elements as one new elements ($e_{i..i+1}$), and put this new elements back to S. For example, S_{n-1} in Figure 4.7 represents the S after the first time combination, and the number of elements in S_{n-1} is $n-1$. I continue this process until there is only one element in S_1: $e_{1..n}$. In this process, I need to conduct the combination $n-1$ times in total. And each time I can combine both SIs and NSIs. I can see that each combination forms a new NSI, and this NSI has at most two NSI children. For the gth combination, there are $n-g$ elements in the set $S_{n-g}, g = 1, ..., n-1$. A child interval $l_{\ell'}$-$[w, z]$ of any l-intervals in LT has size $n-1$ when it is after the $(n-2)$th combination: $l_{\ell'}$-$[1, n-1]$, which is the maximum size of a child interval.

I then prove that $n-1$ is the maximum number of NSI in LT. If I remove k ($k > 2$) elements from $S_{n-1}, ..., S_{n-w}$, where $w \geq 1$ and $1, ..., w$ are not necessarily adjacent, and I remove 2 elements from $S_{n-v}, \forall v \in [1, n-1]$, and $v \neq 1, ..., w$, then after w times combinations, it remains $n - w \times k$ elements in $S_{n-(w)}$, and requires $n - w \times k - 1$ times combination. So the overall combination times is $t = n - 1 - k(w-1)$, as $k > 2$ and $w > 1$, so $t < n - 1 = max(K)$. I call the behavior of combining more than one elements at one time as multi-combination (statement 1). This proof also indicates that as long as multi-combination happens (once or more than once and at any positions), the total number of LCP intervals will be decreased. Hence, the number of LCP interval is a decreasing function of the child number of each interval (statement 2).

Statement 3 is definite. When $[i, j]$ is fixed, as the child intervals of l_ℓ cannot be overlap with each other, the increase of m will result in the decrease of z. □

Based on Lemma 4.4, it has very low possibility that $O(LiSAM)$ reaches $O(N + n^3)$. I consider two extreme cases:

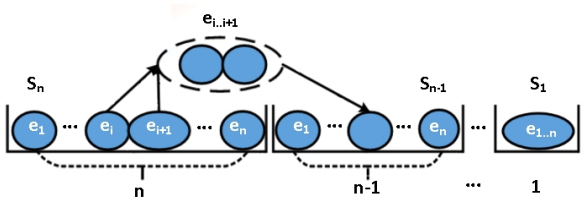


Figure 4.7: Number of LCP intervals

- Assume S has maximum number of LCP intervals $n-1$, the root interval l^{n-1} has $sz_0 = n-1$, and each interval $(l^{n-1}, \cdots, l^1)$ has $max(m) = 2$, then the time complexity of LiSAM is $O(LiSAM) = O(N + (n-1) \times 2 + \cdots + 2 \times 2) = O(N+n)$;
- Assume C is a child interval of an l-interval in LT, has $sz = n-2$, and has $m = \lfloor \frac{n-1}{2} \rfloor$ NSI children, then C is the only child of the root interval $[0, n-1]$, $K = 2+m$, and each child of C has $sz = 2$ and has 0 children, then the time complexity is $O(N + 1 \times m \times (n-1) + m \times 2) = O(N+n^2)$.

If S is highly compressed compared with T (i.e., $N \gg n$), the time complexity of LiSAM can be improved dramatically.

Space Complexity

For the space efficiency, I mainly need to keep three types of information in LiSAM: the raw time series T, the discrete time series S and the LCP table, which consumes $O(N + n + c * n)$ space, where N is the length of T, n is the length of S, and c is a constant that count the number of tabs in the LCP table. Therefore, LiSAM has $O(N+n)$ space complexity, and when $N \gg n$, it becomes to $O(N)$.

4.6 Summary

In this chapter, I proposed an algorithm LiSAM to identify different-length motifs of time series. By converting the raw time series into symbolized time series, I group the symbolized subsequences as approximate motifs. I focused on resolving two important problems: releasing the constraints of trivial matching between subsequences with different lengths and improving the time and space efficiency. I proposed two covering relations: α-covering between instances of l-intervals and β-covering between l-intervals to support the motif discovery. I used the LiSAM algorithm to identify the β-uncovered l-intervals, and I introduced two LCP tabs: $presuf$ and $nextsuf$ to support the identification of the α-uncovered instances of an l-interval. In general, the time and space complexities of LiSAM are $O(N + n^2)$ and $O(N)$ respectively. When $N \gg n$, the time complexities of LiSAM can reach $O(N)$, where N is the length of the raw time series T, and n is the length of the discrete time series S. Experimental results showed the high accuracy of LiSAM on finding different-length motifs.

LiSAM provides an approximate and time&space-efficient way of identifying different-length motifs of static time series, which can be used in rule-learning and feature summarization of time series to support decision making process. In the future, I am going to take further steps on motif identification: identifying motifs in real-time time series, and identifying motifs with similar shapes but with different lengths in terms of the time scale, which are both difficult and important problems in time series mining area.

Part II

Part II: Selecting Cloud Services for Building Medical Sensor-clouds

In Part I, I discussed the data stream mining techniques of both anomaly detection and motif identification in single variate data streams. Respectively, I proposed a supervised anomaly detection method, and an unsupervised motif identification method for Medical time series.

With the advent of a large amount of medical data streams, Cloud computing has been an effective way to improve the computational and storage efficiencies for processing big medical data sets. The first step of moving the existing data stream mining techniques to the Cloud computing paradigm is to choose the appropriate Cloud service providers. In this part, I will introduce two decision making methods that help Cloud users make informed decisions on selecting Cloud services. The first Cloud service selection method *Cloud-FuSeR* focuses on processing the fuzzy information in a Cloud service selection, while the second method mainly deals with the influence of the interdependencies among Cloud service evaluation criteria on decision making of Cloud service selection.

CHAPTER 5

CLOUD-FUSER: FUZZY ONTOLOGY AND MCDM BASED CLOUD SERVICE SELECTION

Selecting accurate cloud services to fulfill the need for ordinary cloud users has become a significant challenge. As Cloud service environments encompass many uncertainties that may hinder users to make sound decisions, it is highly desirable to handle fuzzy information when choosing a suitable service in an uncertain environment. In this chapter, I describe a novel fuzzy decision-making framework that improves the existing Cloud service selection techniques. In particular, I build a fuzzy ontology to model the uncertain relationships between objects in databases for service matching, and present a novel analytic hierarchy process approach to calculate the semantic similarity between concepts. I also present a multi-criteria decision-making technique to rank Cloud services. Furthermore, I conduct extensive experiments to evaluate the performance of the fuzzy ontology-based similarity matching. The experimental results show the efficiency of the proposed method.

5.1 Introduction

With the proliferation of a range of Cloud services over the Internet, efficient and accurate service selection based on user-specific requirements has become a significant challenge for decision makers and Cloud consumers [72]. Various decision-making methods are applied to help service users to find the most appropriate services [196]. However, a service selection environment may encompass many uncertainties, which may hinder the ability of decision makers to make sound decisions.

Consider a Cloud service selection scenario [96] in which a service user is looking for an online Cloud storage service with high security requirements. In the Cloud market, two types of storage services meet the user's requirements: normal storage services without an encryption function and secured storage services with advanced encryption tools. There are three options for the user to choose from: a) a standard storage service without extra data encryption, b) a standard storage service with an encryption service from another service provider (e.g. TrueCrypt [212]), and c) a secured storage service (e.g. Spideroak [192]). Option *a* may be priced the lowest but have the worst security guarantee. Option *b* may provide the highest level of security but be the most costly and the easiest to use (because the user has to configure the security service and encrypt the data himself). Option *c* may support a lower level of security than Option *b*, but can be used in a more convenient way. It is difficult for the user (especially a non-professional) to compare such options in this multi-criteria scenario of vague expressions and un-quantified evaluation factors (e.g. high security and greater convenience). Therefore, an efficient and accurate decision-making approach is highly desirable to guarantee that the chosen services work well in all possible cases, given the uncertainty [27]. I analyze the fundamental problems that need to be resolved in designing such a Cloud service selection system as below:

Problem 1. Why should we study the problems of Cloud service selection?

Web service selection has been explored for over 10 years. Some Web service selection techniques are being applied to the Cloud service area. However, can we say that it is worthless studying Cloud service selection given the mature techniques of Web service selection ?

To select a service, service users are usually concerned with service functions and Quality of Services (QoS). Web services are software services [87]. For a set of Web services with similar functions, service requestors choose services based on their QoS ratings. Compared with Web services, the process of Cloud service selection is more complicated due to the following reasons: a) Cloud services include not only software services (SaaS), but also IaaS and PaaS [109]. When selecting services for a service composition task, Cloud service selection techniques should consider both the composition between services at the same level (e.g., SaaS) and the composition between services at different levels (e.g., SaaS and PaaS); b) A company user may need to choose the right deployment model (i.e., private, public and hybrid) to adapting the size and the usage purpose of the organization. Business objectives may be a deterministic factor for a company user making decisions on using Clouds; c) Compared with Web service selection, balancing benefits and risks is much more difficult for a Cloud user composing services from different providers based on a hybrid deployment model; d) Other than the common QoS (e.g., availability and throughput), Web and Cloud service users focus on different service evaluation dimensions. For example, a Cloud user should be concerned more with the interoperability [193] of a Cloud service to enable flexible data management among heterogeneous hardware; e) As most Cloud users (especially enterprise users) expect a long-term and stable relation with a Cloud service provider, rating factors of Cloud providers should also be defined, such as financial stability, experience and technical expertise of the service provider [193]. □

From the above discussion, the problems faced by a Cloud selection system vary greatly from the problems to be solved by Web service selection techniques.

Problem 2. How to find Cloud services with the user-expected functions given the fuzzy expressions of user requirements?

In the Cloud area, it is more difficult for decision makers to make informed decisions on service usage because of the diversification of service types and the lack of service-publication standards [162]. Defining a common ontology with high extensibility has been widely accepted as a reasonable approach to solve this kind of problems [120]. Ontology is a conceptual framework that models domain knowledge into a format that is both human- and machine-readable [140]. Crisp ontology cannot represent uncertain information or process uncertain reasoning. Several fuzzy ontologies for different domains have been proposed [27, 206, 120]. Nonetheless, little attention has been paid to fuzzy ontology for Cloud service selection. Based on a rigorous survey in this area, I conclude that it is necessary to build a fuzzy ontology to support Cloud users' needs: a) compared to Web services, Cloud services are more heterogeneous, e.g., a variety of terms are used by different providers to describe the same concepts; b) there is no normalization of Cloud service descriptions serving different kinds of users; c) there is usually limited concern for the interdependency of criteria (e.g., compensation and dominance [183]) in the Cloud service selection area. □

Another key issue for service selection is that the candidate services are generally evaluated by multiple criteria [186], e.g., Quality-of-service (QoS), provider reputation, and service price. Multi-criteria decision-making (MCDM) techniques are proposed to handle MCDM problems [93]. The analytic hierarchy process (AHP) [66] and the technique for order performance by similarity to ideal solution (TOPSIS) [37] are two of the most widely applied MCDM ap-

proaches.

Problem 3. How to combine the MCDM techniques with the fuzzification and defuzzification techniques to rank Cloud services based on the fuzzy information of the performance of the service non-functional performance and the fuzzy information of user preference on the non-functional properties?

To handle fuzziness, fuzzy AHP and fuzzy TOPSIS have both been studied and applied to various domains. However, there are still problems in this area that require further study. Three key points are identified here: a) Different defuzzification techniques are usually employed to simplify the inference process, where linguistic variables are defuzzified before complicated operations are conducted (e.g. Eigenvector calculation in AHP). In this way, computational efficiency can be enhanced. Nevertheless, a large amount of information used to capture the uncertainty will be lost and the rationality of defining fuzzy variables is therefore reduced [215]. b) In contrast, there are also approaches [211, 24]that retain the fuzziness of fuzzy variables in the whole inference process, which can provide results with fuzzy rating values. The techniques of ranking fuzzy variables are emphasized in this situation. One of the problems encountered by this kind of approach is that enclosing a wide range of fuzzy information in complicated computational steps (e.g. fuzzy number multiplication) may cause the exaggerated support of a fuzzy number [123], which can result in reduced accuracy in decision-making. Also, Chen et al. [18] have stated that it is highly time-consuming to perform complicated fuzzy number arithmetic operations and linguistic approximations. c) The risk attitude of decision makers can greatly affect the decision-making results. The subjective judgement and preference of decision makers can significantly influence the ranking

results [39]. □

Against the above problems, I propose a **F**uzzy **U**ser-oriented **Cloud S-eR**vice Selection System (Cloud-FuSeR) that is capable of dealing with fuzzy information and rating Cloud services by considering three aspects: (1) the similarities between user-required functions and the service functions provided by Cloud providers, (2) the performance of the non-function properties, and (3) the user preference on different properties.

This work has the following distinctive contributions:

- I build a fuzzy Cloud ontology to support the functional similarity calculation, which defines the concept taxonomies of Cloud services and the properties of Cloud services, and quantifies the relations among concepts and between concepts and properties;
- I define fuzzy functions to quantify the performance of the non-functional properties and the user preference on different properties, and employ a fuzzy-AHP and fuzzy-TOPSIS techniques to weight the non-functional properties based on the user preference and to measure the service non-functional performance;
- I compare the performance of a variety of fuzzification, defuzzification, and distance calculation techniques for solving Cloud service selection problems. I conduct comprehensive experiments to show the influence of the different techniques and the risk attitudes of service users on dealing with the fuzzy service information, and identify the techniques that can achieve a best trade-off between the time efficiency and computational accuracy.

The rest of the chapter is structured as follows: Section 5.2 introduces the related work. Section 5.3 describes the main structure of Cloud-FuSeR.It gives the definition of fuzzy service selection ontology, fuzzy lightweight similarity model, and bipartite-based service function-matching process, and introduces a fuzzy AHP criterion weighting approach and a fuzzy TOPSIS service rating approach. Simulation and experiment results are shown in Section 5.4. In section 5.5, I describe a case study of Cloud storage service selection to show the efficiency of Cloud-FuSeR. Section 5.6 concludes this chapter.

5.2 Literature Review

The issue of Cloud service selection has been the research focus of academia and industry for a few years [196].

There are a number of Ontologies that were designed to support functional querying of Cloud services [11]. Jrad et al. [100] proposed a semantic matching approach to evaluate Cloud resources and select the Cloud services or service compositions with respect to the user requirements and Cloud properties. Two types of ontology model are designed to support the Cloud service matching: a user requirement ontology and a service description ontology, both of which include the concepts of functional and non-functional (e.g., QoS and price) Cloud properties. Di Martino et al. [52] introduced an ontology that classify and categorize the functional concepts of Cloud services and virtual appliances. Though the ontologies in both the above work model Cloud service concepts and properties, they does not support the processing of fuzzy information of Cloud services.

The Cloud Services Measurement Initiative Consortium (CSMIC) [187] developed the Service Measurement Index (SMI), a standard measurement framework to measure the profits and costs of Cloud services. Based on the SMI matrices, Garg et al. [71] designed an AHP-based framework 'SMICloud' that supports the Cloud consumers to compare Cloud services. Liu et al. [141] applied utility theory to determine the service components in a service composition process. Zhang et al. [232] developed an AHP approach that supports the real-time QoS performance evaluation. These work is not capable of handling the fuzzy information (e.g., subjective user expressions or vague service descriptions of service providers) in a Cloud service selection process, which will be solved by our work.

In service area, some researchers applied fuzzy techniques to endeavour a robust service selection system. Techniques to combine the fuzzy set theory with ontologies have been applied in a range of Web service areas [46, 121, 213, 36, 199]. In our work, I will apply the fuzzy set theory to a Cloud service ontology. In addition, there are research work combining fuzzy techniques with multi-criteria decision making (MCDM) techniques for Cloud service selection [60, 172, 169]. Qu et al. [170] introduced a framework of assessing Cloud services based on the subjective feedback of cloud users and the objective evaluation of cloud performance from a trusted third party. Tajvidi et al. [201] designed a comprehensive fuzzy Cloud service selection framework to validate the service configuration information, collect real-time QoS measurement data, and deal with consumers' vague QoS perception. These methods conduct defuzzification at the very beginning (i.e.before conducting complex calculations, such as Eigenvector calculation), which may cause extensive information loss and therefore decrease the rationality of the adoption of fuzzy theories. Some

methods [119, 45] attempt to avoid this problem by retaining the fuzziness in the main processing steps.

5.3 Fuzzy User-oriented Cloud Service Selection System: Cloud-FuSeR

The Cloud-FuSeR is a Cloud service selection system that recommends top-K Cloud services with user-expected functions and with the highest services' non-functional performance given user preference on non-functional properties. I introduce the details of Cloud-FuSeR in this section. An overview of the work flow of Cloud-FuSeR is shown in Figure 5.1. A fuzzy Cloud ontology and a Cloud service repository are built in the pre-processing stage. In step 1, the pair-wise semantic similarities among concepts are calculated through the *Fuzzy lightweight semantic model* and are kept in the *Concept similarity repository*. In step 2, the Cloud services (in the *CloudServiceRepository*) are filtered by the functional matching component: the *Bipatite Model* to find the top-K Cloud services having functions most similar to the user-required functions. Step 3 processes fuzzy user preference on non-function properties by using a *fuzzy AHP* technique to determine the weights of each property. The *fuzzy TOPSIS* component in step 4 accepts three types of input data: the top-K services with optimal function matchings, the performance of non-function properties of the top-K services, and the property weights. It outputs a sequence of ordered top-K services that are ranked based on the three types of input data.

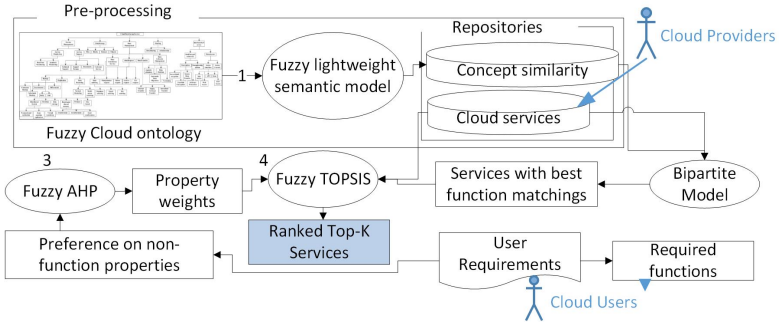


Figure 5.1: Workflow of Cloud-FuSeR

5.3.1 Service Function Matching based on Fuzzy Cloud Ontology

In this section, I introduce a novel fuzzy ontology for service matching. I first define a fuzzy ontology and then give an example of fuzzy cloud storage ontology. After describing similarity calculation between two fuzzy concepts, I present detailed steps for selecting services.

Definition of Fuzzy Cloud Ontology

In this section, I use fuzzy variables to model the relationship between service concepts and the relationship between service concepts and service properties. The proposed ontology model has the following unique features compared with existing fuzzy ontologies: a) it is specific to the domain of Cloud service selection and capable to capture interdependencies between the evaluation criteria of concepts; b) expert-assigned weights are used to establish the fuzzy relations between functional concepts, and between functional concepts and criteria concepts. What is more, the proposed model supports a top-to-bottom hierarchical progress of service selection and forms the basis of service selection framework.

Definition 5.1. *A fuzzy ontology is defined as* $\hat{O} =< \hat{C}, \hat{P}, \hat{I} >$. $\hat{P} = \widehat{DP} \cup \widehat{OP}$ *is a set of fuzzy properties;* $\widehat{DP}$ *refers to fuzzy datatype properties and* $\widehat{OP}$ *represents a set of fuzzy object properties.* $\widehat{DP} = \{f_d\}$, *where d is a datatype property with fuzzy value* f_d. $OP = \{f_Q(c)\}$, *where Q is an object property of concept* $c \in C$ *in the degree of* f_Q. $\hat{I}$*is a set of fuzzy instances;* $\hat{I} = \{f_i(c)\}$, *where i is an instance of the concept c in the degree of* f_i.

Example 5.1. *Assume property serviceResponseTime is an evaluation criterion of Cloud service ServiceMonitoring, and the importance degree of serviceResponseTime for evaluating the ServiceMonitoring service is* 0.75, *then we say that* $d = serviceResponseTime$ *is a datatype property of class ServiceMonitoring with* $f_d = 0.75$.

Definition 5.2. *Assume* $\{A_i, i = 1, \cdots, n\}, F \in C$, *and* A_i *are sub-classes of F, i.e.* $A_i \subset F$, $P_i = \{f_{p_j}(A_i), j = 1 \cdots m\}$ *is a set of m fuzzy properties with respect to* A_i, *then* $P_F = \{\cap(P_i)\} = min\{f_{p_j}(A_1), f_{p_j}(A_2), \cdots f_{p_j}(A_n), j = 1, \cdots, m\}$.

Example 5.2. *Assume class ServiceMonitoring has two subclasses: RealUserMonitoring and SyntheticMonitoring. Property serviceResponseTime is an evaluation criterion for both subclasses. The importance degree of serviceResponseTime for evaluating the RealUserMonitoring service is* 0.75, *and for evaluating the SyntheticMonitoring service is* 0.95. *Then serviceResponseTime is also an evaluation property for class ServiceMonitoring with importance degree* 0.75.

Definition 5.3. *Assume* $A, B \in C$, Q_{AB} *is an object property of A with the range B, and* $f_{Q_{AB}}(A) \in \widehat{OP}$. *If* $f_b(B) \in \hat{I}$, *then* $f_{Q_{Ab}}(A) = min\{f_{Q_{AB}}(A), f_b(B)\}$.

Example 5.3. *Assume hasFunctionalProperty is an object property of the class StorageService. A concept NetworkManagement is a function of a storage service in a degree of* 0.89. *Concept DataTransferManagement is an instance of the concept NetworkManagement in a degree of* 0.95. *Then the concept DataTransferManagement is the value of the object property hasFunctionalProperty in the degree of* 0.89.

Definition 5.4. *If object property q_1 and object property q_2 are the domain and range of an inverseOf property, then $f_{q_1}(q_2) = f_{q_2}(q_1)$.*

Example 5.4. *Assume Is-part-of and Comprise are two object properties between the classes Replication and StorageService, if in* 80% *cases, the objects of Replication is a constitute function of the objects of StorageService, then I can say that Replication is part of StorageService in a degree of* 0.8*, and the degree the StorageService comprises Replication is* 0.8*.*

Definition 5.5. *If an object property q is a transitive property among concepts A, B, and C with in different degrees, i.e., the pair (A, B) is an instance of q in a degree of $f_q(A, B)$, and the pair (B, C) is an instance of q in a degree of $f_q(B, C)$, then the pair (A, C) is an instance of q in a degree of $f_q(A, C) = min(f_q(A, B), f_q(B, C))$.*

Example 5.5. *InternetTransfer is a subclass of DataTransfer in a degree of* 0.99*, and DataTransfer is a subclass of DataManagement in a degree of* 0.9*, then InternetTransfer is a subclass of DataManagement in a degree of* 0.9*.*

Theorem 5.1. *Assume $\{B_i, i = 1, \cdots, n\} \subset A$. If $f_{B_i}(b) \in \hat{I}$, then $f_A(b) \in \hat{I}$ and $f_A(b) = max\{f_{B_i}(b)\}$.*

Proof. Let $f_m = max\{f_{B_i}(b)\}$. If $f_A > f_m$, then $\exists C \subset A$, and $b \in C$ in a degree of $f_C = f_A$ and $f_C > f_m$, hence $f_m \neq max\{f_i\}$. If $f_A < f_m$, then $\exists B_k \subset A$, $f_k = f_m$; let $f_A = 0$, then $f_k > 0$, i.e., $b \in B_k$ and $b \notin A$, then $B_k \not\subset A$. Hence, $f_A = f_m$. □

An Example of Fuzzy Ontology: Fuzzy Cloud Storage Ontology

Figure 5.2 shows an example of fuzzy Cloud service ontology. It contains a taxonomy of concepts of Cloud service functionalities (represented by rectangles),

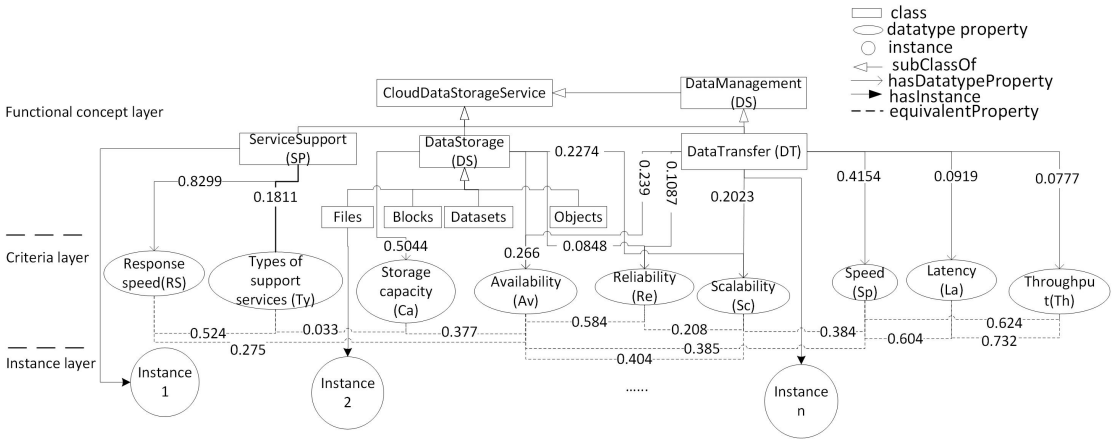


Figure 5.2: A fuzzy Cloud storage ontology

evaluation properties of the concepts (represented by ellipses), and instances of the functionalities. Two types of fuzzy relationship are highlighted in Figure 5.2: the relationship between service functions and properties (namely, the fuzzy evaluation relationship (FER) or fuzzy expert preference (FEP)), and the interdependent relationship between properties (namely, the fuzzy interdependency relationship, FIR). FER implies the importance degree of an evaluation property with respect to the performance of a function. For example, support services are defined to be rated by two properties: the number of types of support service (e.g., online support, phone support, technical support, and document support) and the response speed of the support service. Interesting readers may refer to [195] for the detailed structure of the fuzzy Cloud storage ontology.

To determine *FEP*, i.e., the importance degree of criteria for different functions, the relative weights of criteria are assessed by conducting expert interviews. This method will also be used to identify user preferences on criteria (namely fuzzy user preference, *FUP*) when conducting the QoS-based ranking of Cloud services. The details to determine these two types of fuzzy relationship

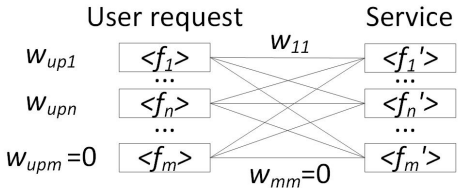


Figure 5.3: Bipartite graph

are given in Section 5.3.1. *FIR* is evaluated automatically using tfidf weighting schema and the cosine correlation similarity measurement method. Figure 5.2 shows the interdependency rate of the criteria with respect to the three functions.

A Fuzzy Lightweight Semantic Model for Service-function-similarity Calculation

This section describes a fuzzy lightweight semantic model (F-lightweight in abbreviation) which is an extension of our previous work: a context-aware lightweight similarity model (C-lightweight in abbreviation) [53]. The C-lightweight model deals with similarity matching with precise information, combining the ontology structure and the context of ontology concepts and relationships. For further details on the lightweight semantic model, readers can refer to [53]. The F-lightweight model improves the C-lightweight model by integrating the fuzzy pseudo-concepts and fuzzy relationships.

The pseudo-concept is extended to a fuzzy pseudo-concept and its representation form is shown in formula 5.1.

$$\zeta_f = \{C_f, [\delta_i, \gamma_{\delta_i}], [o_j, \gamma_{o_j}], [C^x_{f,o_j}, d^x_j], \lambda^y_{o_j}\} \tag{5.1}$$

where C_f is a fuzzy concept in a fuzzy ontology; δ_i is a datatype property of the

concept C_f; γ_{δ_i} is a restriction for δ_i; o_j is an object property and γ_{o_j} is a restriction for o_j. $C^x_{f,o_j}(x = 1, \cdots, k)$ is a concept related by the object property o_j; d^x_j is the degree of the object property o_j being a relationship between concept C_f and fuzzy concept C^x_{f,o_j}, and $\lambda^y_{o_j}(y = 1, \cdots, k-1)$ is a Boolean operation between concepts C^x_{f,o_j}. For example, in the service selection ontology, if a functional concept f_i has an object property criterion C_j, then d^j_x reflects the importance degree of C_j for the evaluation of concept f_i.

The similarity degree between two fuzzy concepts based on the pseudo-concepts and the lightweight structure is represented by the formula 5.2.

$$sim_f(C_{f_1}, C_{f_2}) = (1-\beta) \times sim_{f-cos}(\zeta_{f_1}, \zeta_{f_2}) + \beta \times |sim_{f-Resnik}(\zeta_{f_1}, \zeta_{f_2})| \quad (5.2)$$

, where $sim_{f-cos}(\zeta_{f_1}, \zeta_{f_2})$ is cosine correlation between pseudo-concepts ζ_{f_1} and ζ_{f_2}.

The fuzzy Resnik similarity model for fuzzy pseudo-concepts is defined as in formula 5.3.

$$sim_{f-Resnik}(\zeta_{f_1}, \zeta_{f_2}) = \begin{cases} \frac{max\zeta_f \in s(\zeta_{f_1}, \zeta_{f_2})[-log(P(\zeta_f) * d_{int}(\zeta_f))]}{max\zeta_f \in \theta_f[-log(P(\zeta_f) * d_{int}(\zeta_f))]}, \zeta_{f_1} \neq \zeta_{f_2} \\ 1, \zeta_{f_1} = \zeta_{f_2} \end{cases} \quad (5.3)$$

, where $d_{int}(\zeta_f)$ is the degree of an encountered instance belonging to the concept C_f.

A User-oriented Bipartite Model for Finding Services with User-expected Functions

I now present detailed steps for selecting services. The first step in the service matching procedure is to choose the services with best matched functions. Each

service and each user requirement comprises a set of functions, thus the problem boils down to require the location of all the best matches between the service functions and the requested functions. As different function compositions may have different QoS performance which can cause distinct final ratings for the same service provider, I use a weighted bipartite model for function matching and use a bipartite matching algorithm to find all the possible maximal weighted matches between the user requirements and the service. The steps for finding all the best matches are as follows:

Step 1: Build the bipartite graph. In a service bipartite graph, a node represents a function and an edge between two nodes represents the similarity degree between two functions. If the similarity degree between two functions is higher than a pre-set threshold value, an edge will be added between them. For example, assume service $s_1 = (f_1, \cdots f_m)$, and query $q_1 = (f_1^{'}, \cdots, f_m^{'})$; a bipartite graph is shown in Figure 5.3. The weight values $w_{11}, \cdots, w_{mm}$ are the similarity degree between two functions, and the weight values $w_{up1}, \cdots, w_{upm}$ reflect the importance degree on the queried functions according to the user s preference.

Step 2: Apply the method proposed by Fukuda and Matsui [69] to find all the best matches between the queried functions and the functions contained in a service.

Step 3: Filter out the duplicated matches. If two matches recommend the same set of functions in the same order, the two matches are the same.

Step 4: Choose the top-k services with the highest weighted similarity degree. If $n(n < m)$ queried functions are matched, the weighted similarity degree is calculated by formular 5.4.

$$sim(s_1, q_1) =$$

$$(w_{up1} * w_{11} + \cdots + w_{upn} * w_{mm})/(w_{up1} + \cdots + w_{upn}) \quad (5.4)$$

5.3.2 Service Rating in terms of Non-functional Properties

In this section, I present a multi-criteria decision making technique to rank cloud services. I first introduce an AHP based fuzzy weighting method, and then describe a fuzzy multi-criteria decision making technique to rank services.

Property Weighting based on Fuzzy AHP

The fuzzy pair-wise comparison process is defined as follows:

Step 1. Construct the AHP hierarchy and model the relationships between elements.

Step 2. Determine the local interdependence degree using the pair-wise comparison method. The pair-wise comparison identifies the relative importance between two elements w.r.t. another element. The values used for pair-wise comparison are linguistic variables.

Based on the fuzzy linguistic variable defined in Figure 5.5, as an example, the pair-wise importance weights of each criterion (the criteria abbreviations in reference to the criteria definition in Figure 5.2) for data transfer service (DT), are defined as below: $M_{DT}^{expert} =$

$$
\begin{array}{c}
 \\
Sp \\
Av \\
Re \\
Sc \\
La \\
Th
\end{array}
\begin{array}{c}
\begin{array}{cccccc}
Sp & Av & Re & Sc & La & Th
\end{array} \\
\left(\begin{array}{cccccc}
1 & M & S & M & S & S \\
1/M & 1 & M & EQ & M & M \\
1/S & 1/M & 1 & 1/M & EQ & M \\
1/M & 1/EQ & M & 1 & M & M \\
1/S & 1/M & 1/EQ & 1/M & 1 & EQ \\
1/S & 1/M & 1/EQ & 1/M & 1/EQ & 1
\end{array}\right)
\end{array}
$$

Step 3. Check the consistency of the matrix. For triangular fuzzy numbers, $\beta_{ij} = \gamma_{ij}$, I use the crisp matrix with the middle values $a_{ijm} = (\beta_{ij} + \gamma_{ij})/2$ of all the fuzzy numbers in the corresponding fuzzy matrix as matrix A and check its consistency ratio (CR) by utilizing the method of crisp AHP proposed by Saaty [178]. If $CR > 0.1$, it is assumed that the pair-wise comparison matrix is not consistent and should be reset. For more details of CR calculation, refer to [207]. The consistency rate of M_{DT}^{expert} is 0.012.

Step 4. Find the fuzzy weight vector for each matrix. The method proposed by Csutora et al.[45] closely follows the inference process of crisp AHP by calculating the Eigenvector corresponding to the λ_{max} value of the matrix. For example, the steps to obtain the fuzzy weight vector depending on expert opinions (i.e. fuzzy expert preference) w_{DT}^{expert} are processed as:

a. Set $\alpha = 0$, the left weight vector is:

   $w_{DTl}^{expert} =$

   $(0.4107, 0.1818, 0.1038, 0.149, 0.085, 0.0697)^T$;

   the right weight vector is

   $w_{DTu}^{expert} =$

   $(0.3647, 0.2114, 0.0955, 0.1791, 0.0809, 0.0685)^T$.

b. The middle weight vector is:

$w_{DTm}^{expert} =$

$(0.4231, 0.1861, 0.0683, 0.1861, 0.0683, 0.0683)$.

The k value used to reduce the fuzziness of the weight vector:

$k_{DTl} = 0.6577, k_{DTu} = 1.1602$.

c. The calculation of weight vectors is: $w = (w_i * k_i, w_m, w_u * k_u)$.

The expert weight vector of the data-transfer service when $\alpha = 0$ and $\alpha = 0.5$ is calculated as:

$$w_{DT,\alpha=0.5}^{expert} = \begin{pmatrix} (0.3547, 0.4231, 0.4231) \\ (0.1605, 0.1861, 0.2233) \\ (0.0683, 0.0683, 0.1940) \\ (0.1496, 0.1861, 0.2) \\ (0.0636, 0.0683, 0.0842) \\ (0.0593, 0.0683, 0.0755 \end{pmatrix}$$

I can see that $0 \leq w_{DTl,\alpha=0}^{expert} \leq w_{DTl,\alpha=0.5}^{expert} \leq w_{DTm}^{expert} \leq w_{DTu,\alpha=0.5}^{expert} \leq w_{DTu,\alpha=0}^{expert}$, which satisfies the meaning of the normal α-cut fuzzy numbers: the higher the confidence level, the less support there is for the fuzziness. In this work, the fuzzy weights with $\alpha = 0$ are adopted for later calculation.

Step 5. Defuzzify the fuzzy weight vector depending on the risk attitudes of the decision makers. The risk attitude is determined by the confidence level of the decision maker (i.e., value of α) and the index of optimism μ. For a LR-type fuzzy number $A = (A_L, A_R)$, the defuzzified value based on α and μ can be obtained by conducting a convex combination: $\hat{A}_\alpha = \mu * A_L^\alpha + (1 - \mu)A_R^\alpha$. For example, for a risk-averse decision maker, the value $\alpha = 0, \mu = 0.05$ may be used, and the defuzzified weight of the data transfer service is: $w_{DT,\alpha=0,\mu=0.05}^{expert} =$

$$\begin{matrix} Sp & Av & Re & Sc & La & Th \end{matrix}$$
$$\begin{pmatrix} 0.4154 & 0.239 & 0.1087 & 0.2023 & 0.0919 & 0.0777 \end{pmatrix}$$

Service Rating based on Fuzzy TOPSIS

TOPSIS is a decision technique that is able to rate alternatives by measuring the similarity distance of an alternative to the ideal solution (both positive and negative). The positive ideal solution (*PIS*)(resp. *NIS*) is a reference solution that maximizes (resp. minimizes) the benefit criteria and minimizes (resp. maximizes) the cost criteria[215]. The alternatives with the highest utilities are those that are closest to the positive ideal solution and farthest from the negative ideal solution. The procedure of fuzzy TOPSIS is as follows:

Step 1. Form decision matrix for each sub-service of a service. There are usually two types of properties: quantitative and qualitative. To handle the expression uncertainties, linguistic variables are defined for different properties. Let m be the number of alternatives of service k, which are evaluated by n properties (including i quantitative properties $c_{r_1}, \cdots, c_{r_i}, (r \in (1, \cdots, m))$ and $n - i + 1$ qualitative properties $\tilde{c}_{r,i+1}, \cdots, \tilde{c}_{r,n}$), associated with a set of n membership functions $(f_1, \cdots, f_n)$, then the decision matrix is represented as:

$$D\tilde{M}_k = \begin{pmatrix} f_1^k(c_{11}) & \cdots & f_n^k(c_{1n}) \\ \vdots & \ddots & \vdots \\ f_1^k(c_{m1}) & \cdots & f_n^k(c_{mn}) \end{pmatrix} = \begin{pmatrix} \tilde{u}_1^k(c_{11}) & \cdots & \tilde{u}_n^k(c_{1n}) \\ \vdots & \ddots & \vdots \\ \tilde{u}_1^k(c_{m1}) & \cdots & \tilde{u}_n^k(c_{mn}) \end{pmatrix}$$

Step 2. Determine the weights of the criteria and the weights of the service functions according to the decision maker's preference. The criteria weights and function weights are assigned by decision makers using the fuzzy pair-wise comparison method introduced in Section 5.3.2. The weighting vector of the k^{th}

function is represented as $\tilde{w_k} = (w_{k_1}, \cdots, w_{k_n})^T$.

Step 3. Defuzzify the fuzzy decision matrix and fuzzy weight vectors based on the risk attitude of decision makers, which depends on confidence level (α) and index of optimism (μ). The defuzzification process is introduced in Step 5 of Section 5.3.2. $DM_k = \begin{pmatrix} u^k(c_{11}) & \cdots & u^k(c_{1n}) \\ \vdots & \ddots & \vdots \\ u^k(c_{m1}) & \cdots & u^k(c_{mn}) \end{pmatrix}$,
$w_k = (w_{k_1,\cdots,w_{k_n}})^T$

Step 4. Calculate the aggregated weights (AW): aggregate the defuzzified user weights, expert weights, and function weights:

$$\begin{aligned} AW_k &= w_k^{expert} * w_k^{user} * w_{fun}^k \\ &= (w_{k_1}^{expert} * w_{k_1}^{user} * w_{fun}^k, \cdots, w_{k_n}^{expert} * w_{k_n}^{user} * w_{fun}^k) \\ &= (aw_{k_1}, \cdots, aw_{k_n}) \end{aligned} \tag{5.5}$$

Step 5. Calculate the weighted global decision matrix (WDM). The WDM of the k^{th} function is defined as:

$$\begin{aligned} WDM_k^{\alpha,\mu} &= AW_k * DM_k \\ &= \begin{pmatrix} aw_{k_1} * u_k(c_{11}) & \cdots & aw_{k_n} * u_k(c_{1n}) \\ \vdots & \ddots & \vdots \\ aw_{k_1} * u_k(c_{m1}) & \cdots & aw_{k_n} * u_k(c_{mn}) \end{pmatrix} \end{aligned} \tag{5.6}$$

Step 6. Normalize the weighted decision matrix ($NWDM$). The $NWDM$ of the k^{th} decision matrix is defined as: $NWDM_k = \begin{pmatrix} nwu_k(c_{11}) & \cdots & nwu_k(c_{1n}) \\ \vdots & \ddots & \vdots \\ nwu_k(c_{m1}) & \cdots & nwu_k(c_{mn}) \end{pmatrix}$
where $nwu_{kij} = awu_k(c_{ij})/max(awu_k(c_j)), \forall i = 1, \cdots, m$

Step 7. Define the positive ideal solution (*PIS*) and negative ideal solution (*NIS*). The *PIS* and *NIS* of the k^{th} function is defined by formula 5.7 and 5.8.

$$\begin{aligned} PIS_k = \{&(max(nwu_k(c_{ij}))|c_{ij} \in \Omega_B), \\ &min(nwu_k(c_{ij}))|c_{ij} \in \Omega_C), i = 1, 2, \cdots, m\} \\ &= \{nwu_k(c_1^*), \cdots, nwu_k(c_n^*)\} \end{aligned} \tag{5.7}$$

$$\begin{aligned} NIS_k = \{&(min(nwu_k(c_{ij}))|c_{ij} \in \Omega_B), \\ &(max(nwu_k(c_{ij}))|c_{ij} \in \Omega_C), i = 1, 2, \cdots, m\} \\ &= \{nwu_k(c_1^-), \cdots, nwu_k(c_n^-)\} \end{aligned} \tag{5.8}$$

where Ω_B is a set of benefit criteria, Ω_C is a set of cost criteria.

Step 8. Calculate the distance from *PIS* ($d+$) and *NIS* ($d-$) using formula 5.9 and 5.10.

$$\begin{aligned} d_i^+ &= \Sigma_{k,j} d(nwu_k(c_{ij}), nwu_k(c_j^*)) \\ &= \Sigma_{k,j} |nwu_k(c_{ij}) - nwu_k(c_j^*)| \end{aligned} \tag{5.9}$$

$$\begin{aligned} d_i^- &= \Sigma_{k,j} d(nwu_k(c_{ij}), nwu_k(c_j^-)) \\ &= \Sigma_{k,j} |nwu_k(c_{ij}) - nwu_k(c_j^-)| \end{aligned} \tag{5.10}$$

, where k is the number of functions.

Step 9. Rate and rank the alternatives. The rating value of alternative i is calculated by formula 5.11.

$$\mu_i = 1 - \frac{d_i^+}{d_i^+ + d_i^-} \tag{5.11}$$

Step 10. Rank the alternatives based on both function similarity and criteria rating, calculated by formula 5.12.

$$u_i = \alpha * sim_i + \beta * \mu_i, \alpha, \beta \in [0, 1], \alpha + \beta = 1 \tag{5.12}$$

where α and β are the weights assigned by decision makers based on their preference.

5.4 Experiment and Time Complexity of Cloud-FuSeR

I carry out simulations and evaluations from two steps: (i) service function matching based on the Fuzzy lightweight similarity model (abbr. FL), and (ii) service ranking via the evaluation of QoS parameters using the proposed fuzzy TOPSIS method (namely ALPHA). Step (i) is processed on the basis of a fuzzy Cloud storage service ontology [195], which contains 91 concepts and over 200 instances. Queries of 50 different cases are performed. Each query includes different service functions and various numbers of functions. The first step produces a set of filtered service providers whose provisions are capable of matching the functional requirements of service requesters. Then the services provided by the remained service providers are ranked through the fuzzy TOPSIS approach. Details for the fuzzy TOPSIS are shown in Section 5.3.2.

The simulation is conducted in Windows 7 Enterprise 64-bit OS, with Intel core i5-3470S CPU, 2.90GHz, and 8GB RAM, 1333MHz. Algorithms are implemented in Eclipse JAVA, combined with Matlab and protégé for data processing and ontology management. For each dimension (i.e., semantic similarity and TOPSIS alternative ranking), the proposed framework is compared with the representative models and approaches, and is based on the standard evaluation parameters in each area.

5.4.1 Experiment Design and Result Discussion of Fuzzy-cloud-ontology-based Function Matching

In this experiment, I compare the performance of the proposed FL model with the performance of three typical semantic similarity models (i.e., Resnik, Wu-Palmer, and String-based) in terms of five indicators: *precision*, *recall*, $fallout$, *F-measure*, and *F-measure*($\beta = 2$). The compared methods are as follows:

1) Resnik (abbr. Res). Resnik [173] similarity model is defined as $sim_{Resnik}(C_1, C_2) = \underset{C \in S(C_1,\cdots,C_2)}{max}[-log(P(C))]$: where C_1 and C_2 are two concepts, $S(C_1, C_2)$ is the set of concepts subsuming both C_1 and C_2 , and $P(C)$ is the possibility of encountering an instance of concept C.
2) Wu-Palmer (abbr. WP). Wu-Palmer[221] similarity model is defined as: $sim_{wu\_palmer}(C_1, C_2) = (2 * N)/(N_1 + N_2 + 2 * N_3)$, where C_3 is the least common super concept of C_1 and C_2, then N_1 is the number of nodes on the path from C_1 to C_3. N_2 is the number of nodes on the path from C_2 to C_3. N_3 is the number of nodes on the path from C_3 to root.
3) String-based (abbr. Str). The string-based similarity model [61] is defined as: $\frac{CL}{D1+D2}$, where CL is the length of common longest similar sub-string in description1 and description2, and $D1$ and $D2$ are the length of description1 and the length of description2 respectively.

Assume ML is the number of matched and logically similar concepts, M is the number of matched concepts, L is the number of logically similar concepts, MN is the number of matched and non-logically similar concepts, and NL is the number of non-logically similar concepts, the evaluation indicators are defined as follows:

Table 5.1: Performance Comparison of Four Similarity Models

Model	Thresh	Re	Pr	Fa	$FM_{\beta=1}$	$FM_{\beta=2}$
Str	> 0.5	0.52	0.36	0.21	0.41	0.49
WP	> 0.5	0.68	0.46	0.2	0.53	0.56
Res	> 0.8	0.88	0.46	0.19	0.6	0.69
FL	> 0.85	0.94	0.52	0.15	0.65	0.8

1) Precision (abbr. Pr) [19]. For a single concept, precision is defined as: $Pr(C) = \frac{ML}{M}$. For a collection of n concepts, the average precision is employed, which is defined as: $Ave(Pr) = \Sigma_{i=1}^{n} Pr(C_i)/n$.

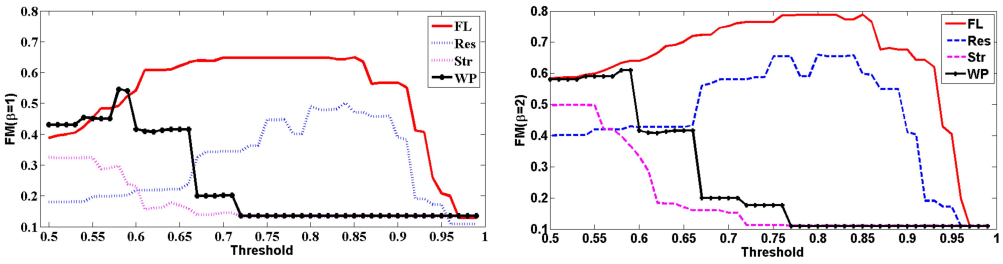
2) Recall (abbr. Re) [19]. For a single concept, recall is defined as: $Re(C) = \frac{ML}{L}$. For a collection of n concepts, total recall is adopted, which is defined as: $Ave(Re) = \Sigma_{i=1}^{n} Re(C_i)/n$.

3) Fallout (abbr. Fa) [225]. For a single concept, fallout is defined as: $Fa(C) = \frac{MN}{NL}$. For a collection of n concepts, the total fallout is defined as:$Ave(Fa) = \Sigma_{i=1}^{n} Fa(C_i)/n$.

4) F-measure (abbr. $FM_{\beta=1}$ or FM). F-measure [175] is defined as the mean of precision and recall: $FM_{\beta=1} = 2{\times}P{\times}R/(P{+}R)$. When FM reaches the highest point, precision and recall achieve the best trade-off. The FM performance comparison of the four models is shown in Figure 5.4a.

5) F-measure ($\beta = 2$) (abbr. $FM_{\beta=2}$). $FM_{\beta=2}$ [92] combines precision and recall based on user preference, which is defined as $FM_{\beta=2} = ((1+\beta^2){\times}P{\times}R)/(\beta^2{\times}P + R)$.

When $\beta = 2$, recall is weighted twice as much as precision, reflecting the concerns of users in practical information retrieval systems [194]. The $FM_{\beta=2}$



(a) FM, β =1 (b) FM, β =2

Figure 5.4: Performance of FM

performance comparison of the four models is shown in Figure 5.4b.

The average value of each indicator is calculated for final performance evaluation. Specifically, $FM_{\beta=1}$ and $FM_{\beta=2}$ are two aggregated metrics which are employed as key benchmarks to find the best threshold value of functional similarity for forming a bipartite matching model. Other indicators will be evaluated based on the identified threshold value. As shown in Figure 5.4, the optimal threshold value (on the x-axis) for each model can be chosen according to the point at which FM reaches its highest value shown on the y-axis. Table 5.1 shows the performance of the candidate models relative to the threshold values (i.e., Thresh) and according to the five indicators.

From this comparison, it can clearly be seen that the FL model performs better than the other three models for each of the indicators.

5.4.2 Experiment Design and Result Discussion of Service Rating in terms of Non-function Properties

I carry out further simulations and experiments to evaluate the performance of ALPHA compared to the traditional TOPSIS method and other fuzzy TOPSIS methods. At first, I define two fuzzy variables that will be used in this experiment. The variables in Figure 5.5(a) represent the rating values of the decision matrix. Figure 5.5(b) defines the linguistic variables for the pair-wise comparison in the AHP process.

Experiment Design

Five indicators are used for evaluation:

1) Mean squared error of ranks (MSER):$MSER_m = E[(r_m - r_0)^2]$, where r_m is the rank produced by method m and r_0 is the rank corresponding to the traditional TOPSIS method, and E refers to the expected value operation.
2) TOP rank matched count (TOP). Counts the number of top ranked alternatives of different methods matching the top ranking given by the traditional TOPSIS.
3) Match count (MATCH). Counts the number of matched positions of alternatives in two rankings, that is, the matching rate between two rankings of the same set of alternatives.
4) Variation between TOP and the expected results of the TOPSIS method (TOPSIS). The definition of the traditional TOPSIS method indicates that the top-ranked alternative should have both the shortest distance to the

PIS and the farthest distance from the *NIS*. In this experiment, I employ the TOPSIS definition as one of the testing indicators. In cases where risk attitude is not considered, I only measure whether the top-ranked alternative is closest to the PIS or farthest from the NIS, while in cases with risk attitude, additional measures are taken. In the case of risk aversion, a check is made to determine whether the distance from the *PIS* is shorter than the distance from the *NIS*, and in the case of risk taking, an additional step taken is to check whether the distance from the *NIS* is shorter than the distance from the *PIS*.

The proposed method attempts to prune the candidate alternatives using the function matching process, where top-k matched services or service compositions are selected to conduct the TOPSIS ranking step. The maximum value of k is set as 20 in this simulation. As for the number of criteria, seven plus or minus two offers the greatest amount of information an observer can provide based on a reasonable judgement [230]. Conducting MCDM analysis with a higher number of criteria may cause problems such as reduced accuracy and increased computation complexity [31]. Based on this theory, I choose a relatively small number of alternatives in this experiment. The performance of the evaluated methods is compared in accordance with a different number of alternatives and criteria. I use the following parameters to perform simulation in this work. 1) Number of criteria: 4,8,6,10,12

2) Number of alternatives: 3,5,7,9,15,20

3) Value of decision matrices: simulate the crisp criterion value of each element fitting log-normal distribution

4) Value of criterion weights: a pair-wise weight between elements is generated from the 10 fuzzy criteria variables defined in Figure 5.5b

5) Number of replications: 60 times simulated decision matrices, each of them combined with 20 times randomly generated criterion weight values.

Based on the indicators and parameters introduced above, I now compared 25 instances of TOPSIS methods, which were differentiated by the risk attitudes of decision makers (i.e., risk-neutral, risk-averse, and risk-taking), by the fuzziness of the ranking process, and by the methods of calculating the distance between numerics. The 25 cases are as follows:

Crisp(case 0). Traditional TOPSIS without involving fuzziness. Crisp decision values are applied to solve traditional TOPSIS. Crisp weights are determined by domain experts based on the nine rating levels (1-9) in traditional AHP. For the other fuzzy cases, the crisp decision values will be first converted to fuzzy numbers based on the fuzzy functions for decision matrices defined in Figure 5.5a; and the fuzzy criteria variables that correspond to the nine rating levels are used to determine the fuzzy criteria weights.

FUZZY-dis-N(1);*FUZZY-dis-A*(2);*FUZZY-dis-S*(3). The fuzzy pair-wise comparison is conducted by using the method proposed by Csutora et al.[45], and the ranking of fuzzy numbers is processed by the method proposed by Liem et al.[209]. Risk attitude is determined by a weighting function that assigns changing weights to the measured fuzzy distance. The functions used are: $f = x \rightarrow$ *risk neutral* (-N); $f = x * x \rightarrow$ *risk averse* (-A); $f = 1 \rightarrow$ *risk seeking* (-S).

FUZZY-mem-N(4);*FUZZY-mem-A*(5);*FUZZY-mem-S*(6). Referencing the work proposed by Chamodrakas et al.[32], in which the degree closest to the PIS and farthest from the NIS is employed under three types of risk attitude. Risk attitude is determined by a connective parameter $p : p = 2 \rightarrow$ *risk neutral*;

$p = 0 \rightarrow$ *risk averse*; $p = \infty \rightarrow$ *risk seeking*.

GMIR(7). GMIR [33] defuzzification technique is utilized to defuzzify the fuzzy decision matrix and the global fuzzy weights obtained from the fuzzy AHP process.

ALPHA-N(8),*ALPHA-A*(9),*ALPHA-S*(10). The proposed method in this work under three types of risk attitude. Risk attitude is determined by the value pair of alpha (α) and the convex parameter (μ), which is defined as: $< \alpha = 0.5, \mu = 0.5 > \rightarrow$ *risk neutral*; $< \alpha = 0.5, \mu = 0.05 > \rightarrow$ *risk averse*; $< \alpha = 0.5, \mu = 0.95 > \rightarrow$ *risk seeking*.

ALPHA2-N(11),*ALPHA2-A*(12),*ALPHA2-S*(13). Referencing the method proposed by Kwong et al.[113], employing alpha-cut and convex defuzzification technique, with respect to three types of risk attitude. Risk attitude is determined in the same way as given above.

Fuzzy-Pdis-S(14). The fuzzy pair-wise comparison is conducted by using the method proposed by Csutora et al.[45]. Fuzzy numbers are used in TOPSIS calculation. P-adic norm with Sum function (see formula5.14) is applied to measure the distance between fuzzy numbers.

Fuzzy-Pdis-M(15). The same setting as the above, except that P-adic norm with Max function (see formula5.14) is utilized for distance measure.

Fuzzy-Pmem-S-N(16),*Fuzzy-Pmem-S-A*(17),
Fuzzy-Pmem-S-S(18). The same setting as the cases of Fuzzy-mem-neutral/averse/seeking, except that P-adic norm Sum function is applied for the distance calculation between fuzzy numbers.

Fuzzy-Pmem-M-N(19),*Fuzzy-Pmem-M-A*(20),
Fuzzy-Pmem-M-S(21). The same setting as the cases of Fuzzy-mem-neutral/averse/seeking, except that P-adic norm Max function is applied for the distance calculation between fuzzy numbers.

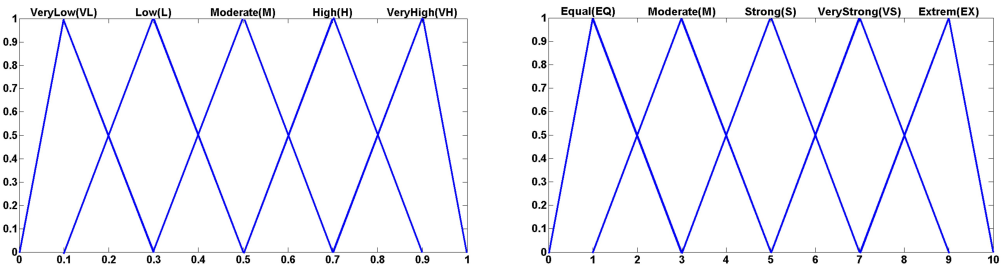
ALPHA-P-N(22),*ALPHA-P-A*(23),*ALPHA-P-S*(24).
The same setting as the cases ALPHA-neutral/averse/ seeking, except that P-adic norm is applied for distance calculation.

The measure of distance between numerics is mainly applied to rank the rating values of alternatives in the TOPSIS process. Based on the distance measure methods, the instances compared are classified into two groups: Archimedean-field (AF) and non-Archimedean-field (NAF) (here I focus on p-adic-norm in NAF). One of the key differences between the above two fields is that the former is under the constraints of triangular inequality, i.e., $|x+y| \leq |x|+|y|, x, y \in AF$; while the latter has to satisfy a more restricted ultra-metric triangular inequality, i.e., $|x + y| \leq max(|x|, |y|), x, y \in NAF$. For more details regarding the two norms, readers can refer to the work of [57] and [203]. Below I introduce the p-adic-norm for both crisp and fuzzy numbers [203].

Assume p is a prime, $a \in z$, and $a \neq 0$. I define $ord_p a = m$, where $a \equiv 0(\text{mod } p^m)$. A p-adic norm on the field of Q is defined by formula 5.13.

$$\phi_p(x) = \begin{cases} (\frac{1}{p})^{ord_p x}, x \neq 0 \\ 0, x = 0 \end{cases} \tag{5.13}$$

Then the distance between two crisp numbers $(x, y \in Q)$ is calculated as: $d_\phi(x, y) = \phi_p(x - y)$. To operate fuzzy variables, a metric, depending on ϕ, in



(a) Decision Fuzzy Variable (b) Criteria Rating Fuzzy Variable

Figure 5.5: Two fuzzy variables

Q^3 is defined by formula 5.14.

$$\begin{aligned} \hat{\phi}(x,y,z) &= \phi(x) + \phi(y) + \phi(z), or \\ \hat{\phi}(x,y,z) &= max(\phi(x), \phi(y), \phi(z)), x, y, z \in Q \end{aligned} \tag{5.14}$$

I name the former formula in 5.14 as Sum function, and the latter as Max function. The performance of the Crisp instance(case 0) is utilized as a benchmark of the indicators MSER, TOP, and MATCH.

Experiment Results

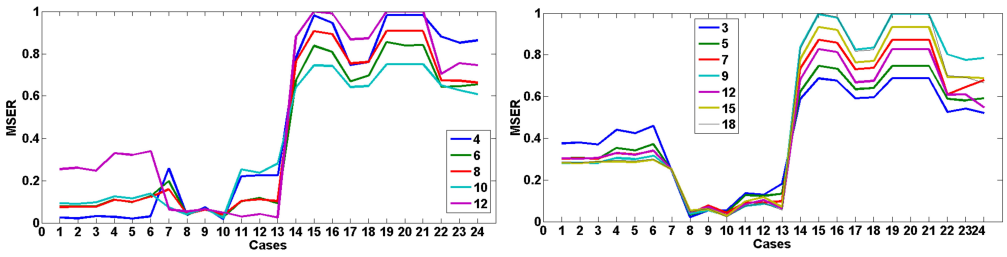
In this section, I compare experimental results. The experiment results of the five indicators are shown in Figures 5.6 to 5.9. On the basis of the MSER definition, the lower the value, the better the performance. Figure 5.6 shows that the proposed method reaches the lowest point in any of the deployments compared to other methods. It also has the best stability when the number of alternatives and criteria are changed, which means it can maintain high performance in ranking cases with different numbers of alternatives and criteria. The MSER performance of the instances of Fuzzy-dis, Fuzzy-mem, GMIR, and ALPHA2 is slightly lower than that of the ALPHA approach (around 0-0.3). However, their

performance is not as stable as ALPHA, and does not show a definite trend when the criterion number is changed. The number of alternatives does not have a substantial influence on the Archimedean-based approaches, while the performance of the p-adic-based approaches shows a downward trend when the alternative number is increased.

Figure 5.7 shows that ALPHA has higher performance than the other methods in terms of the TOP ranked match indicator in accordance with different numbers of criteria and alternatives. The performance of TOP of Fuzzy-dis, Fuzzy-mem, and ALPHA2 instances changes dramatically when the number of criteria changes, and it shows a similar trend to the change of MSER performance under a different number of criteria. The number of alternatives also has a dramatic influence on the TOP performance but does not follow a regular trend.

A comparison of the match count rate is shown in Figure 5.8. Like the TOP ranked match case, the ALPHA has the highest MATCH rate. Its performance is 20% higher than that of GMIR, and is 20% higher than that of fuzzy-dis and fuzzy-mem cases when the number of criteria is 4. When the number of criteria is 12, the performance of fuzzy-dis and fuzzy-mem is much lower than that of ALPHA (around 70% lower). The number of alternatives has a clear influence on the MATCH rate. The MATCH performance of all the cases shows a monotonically decreasing trend when the alternative number is increased.

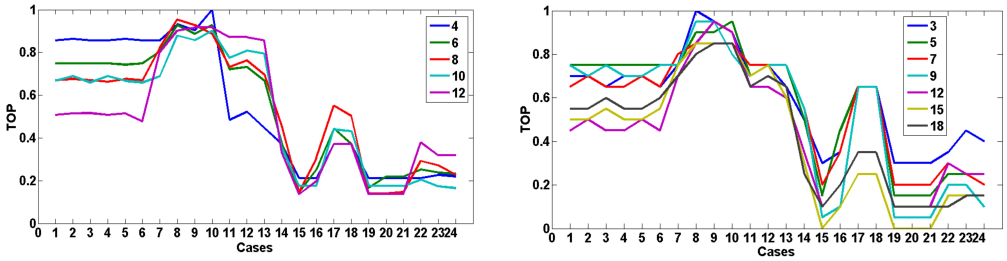
Figure 5.9 shows that most of the cases considered closely follow the standard TOPSIS expectation for the different number of alternatives and criteria. The ALPHA method almost maintains the highest performance under this indicator, while the cases of Crisp, Fuzzy-Pdis-M, and Fuzzy-Pmem-M are on a



(a) MSER under different number of criteria

(b) MSER under different number of alternatives

Figure 5.6: Comparison of rate of MSER

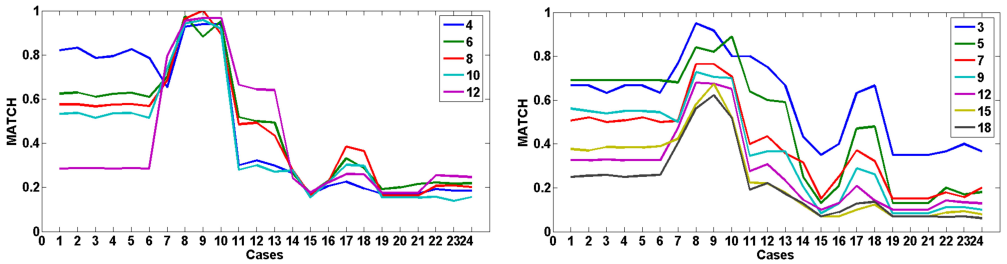


(a) TOP under different numbers of criteria

(b) TOP under different numbers of alternatives

Figure 5.7: Comparison of top matched rate

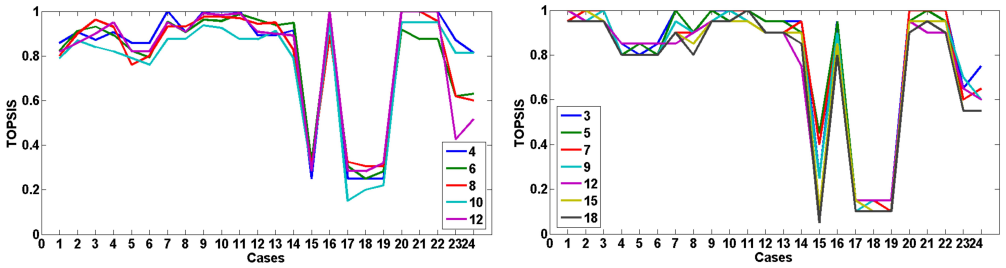
similar TOPSIS rating level as ALPHA. However, the cases of Fuzzy-Pdis-S and Fuzzy-Pmem-S have very low TOPSIS performance. In this case, the p-adic-based distance measure between fuzzy numbers based on the Max function is much better than the p-adic fuzzy distance measure based on the Sum function. Compared to instances of fuzzy-mem and GMIR, the proposed ALPHA has 5% – 10% higher performance.



(a) Matched count under different numbers of criteria

(b) Matched count under different numbers of alternatives

Figure 5.8: Comparison of rate of matched count



(a) TOPSIS under different number of criteria

(b) TOPSIS under different number of alternatives

Figure 5.9: Comparison of rate of aligning TOPSIS definition

Discussion

The proposed fuzzy TOPSIS method (ALPHA) is compared with eight other TOPSIS methods, including Crisp TOPSIS, GMIR, Fuzzy-dis, Fuzzy-mem, ALPHA2, Fuzzy-Pdis, Fuzzy-Pmem, and ALPHA-P) under five indicators: MSER, TOP match count, MATCH count, TOPSISDefinition, and time complexity.

Generally, the performance of MSER, TOP, TOPSISDefinition, and MATCH of the compared instances does not show much change when the number of criteria is changed. However, the number of alternatives usually influences the performance of these indicators negatively, i.e. more alternatives, lower performance. The reason is evident. The involvement of more alternatives will increase the complexity of the problem, raising the rate of deviation from the standard ranking. When the number of alternatives and criteria is changed, the ALPHA method shows better stability than other methods. It provides greater flexibility and maintains higher performance in different decision settings.

The instances that demonstrate the second best performance under the three indicators are the instances of ALPHA2. Compared to ALPHA, they have less stability in general; their performance of MSER indicator, of TOP, and of MATCH in particular is very low when there are fewer criteria. This indicates

that this approach is less accurate, and there is a limitation on setting the number of alternatives and criteria in a decision-making problem. The reason for this is that the approach defuzzifies the fuzzy weights and fuzzy decision matrix before conducting the TOPSIS ranking process, which reduces the support of a fuzzy number to a crisp point, resulting in a loss of uncertainty information. For example, two different fuzzy rating values may be defuzzified to the same crisp value, in spite of the difference in the membership degree between them. Information loss causes instability and reduced accuracy. The number of criteria in particular indicates the volume of information available to a decision maker. A greater amount of information can help decision makers to make more accurate decisions. On the other hand, defuzzification with fewer criteria can result in lower performance.

From the comparison results, the performance of the Fuzzy-dis and Fuzzy-mem approaches on MSER, TOP, and MATCH indicators is easily influenced by the number of criteria and alternatives. The more the criteria and alternatives, in particular, the lower the performance of these two approaches. Both approaches retain uncertain information to the largest extent compared to other Archimedean-based methods, and the exaggeration of fuzzy information causes a larger deviation in their results from the results of the crisp TOPSIS. In addition, the FUZZY-mem and FUZZY-dis are very time-consuming, which makes them computationally much more complex than the other instances under the same condition. However, the alignment of their results to the definition of the TOPSIS method is almost the same as that of the other methods. For the indicators of MSER, TOP, and MATCH, Archimedean-based approaches show higher performance than p-adic-based approaches overall.

Table 5.2: Simulated QoS Performance Values and QoS weights

Service	*inte*						*obs*				*tr*	
	Sp	Av	Re	Sc	La	Th	Ca	Av	Re	Sc	RS	Ty
s1	46	99	98	97	24	33	120	99	96	99	30	5
s2	30	98	99	92	32	10	100	98	97	92	60	5
s3	67	92	96	94	26	8	70	97	99	99	45	4
s4	42	94	96	96	43	15	88	93	98	92	72	3
s5	89	97	98	92	51	32	150	98	96	91	30	4
w^{user}	0.48	0.26	0.12	0.24	0.1	0.08	0.51	0.21	0.09	0.39	0.9	0.11
w^{expert}	0.42	0.24	0.11	0.2	0.09	0.08	0.5	0.27	0.09	0.23	0.83	0.18
w^{expert}_{func}			0.627					0.318			0.132	
$\overline{w}^{user}$	0.38	0.18	0.08	0.22	0.07	0.07	0.39	0.13	0.09	0.38	0.9	0.1
$\overline{w}^{expert}$	0.42	0.19	0.07	0.17	0.07	0.07	0.51	0.19	0.07	0.23	0.83	0.17
$\overline{w}^{user}_{func}$			0.637					0.258			0.105	

Overall, ALPHA performs better than the other Archi-medean based methods in most cases. The number of alternatives influences MSER, TOPSIS definition commitment and TOP ranked match in a negative way. A higher number of criteria and alternatives can cause higher computational complexity. The Archimedean-based approaches perform better than the p-adic-based approaches and the p-adic-based approaches especially are of much higher computational complexity. However, their performance is on a level similar to the Archimedean-based approaches in terms of TOPSIS indicator, which is independent from the benchmark. Deeper research on non-Archimedean-based fuzzy operations needs to be conducted to break through the limitation of the current Archimedean-based approaches.

Table 5.3: Service Ranking based on the Rates of QoS Performance and Functional Similarities

Service instance	s1	s2	s3	s4	s5
f-QoS rate	1	0.814	0.924	0.404	0.737
c-QoS rate	1	0.751	0.79	0.388	0.672
f-function rate	1	1	0.992	0.983	0.962
c-function rate	1	1	0.976	0.92	0.904
f-aggregated	1	0.907	0.958	0.694	0.85
c-aggregated	1	0.876	0.883	0.654	0.788

5.4.3 Time Complexity Discussion of Cloud-FuSeR

In this section, I analyse the time complexity of Cloud service selection by using Cloud-FuSeR, which depends on the following steps: ontology establishment, concept similarity calculation based on the F-lightweight model, function matching based on the bipartite-graph model, non-function property weighting based on the fuzzy AHP, and service rating based on the fuzzy TOPSIS.

The ontology building and similarity calculation are off-line operations. The concept similarities are calculated in advance based on the built ontology and are stored for the users' function querying. So I do not consider the time consumption of these two steps. For the bipartite-graph-based function matching, assume a service or service composition has p functions, and a user is querying q functions, then the time complexity of building a bipartite model is $O(pq)$. As p and q are both the number of functions of a service item, I assume $p = q$, so $O(pq)$ can be written as $O(p^2)$. Let $v = p + q$ is the number of nodes (i.e., functions) of a bipartite graph, and e is the number of edges. Since in a matching

of a bipartite model, a node on the left side can only be matched to a node on the right side, and in the worst case, there may be $v/2$ nodes on each side, and each node on one side is connected to each node on the other side, so the maximum number of edges in a bipartite graph is $e = v/2$. Let L be the number of perfect matchings in the bipartite graph, then $L \leq v$. Hence, the time complexity of finding all the perfect matchings is $O(v(v \log v + e) + Lv(v + e))$. In the worst case, the time complexity reaches $O(v(v \log v + v/2) + Lv(v + v/2)) \sim O(Lv^2)$. Note that v is usually not large (most probably less than 20 functions in a service or service composition), so apparently $v \ll n$, where n is the number of service items in a service repository. To find the top-K best matched services in terms of service functions, I need to check each service item in the service repository, so the overall calculation time is $O(nLv^2)$.

For the service rating based on non-function properties, a fuzzy AHP approach is applied to determine criteria weights. The main operation of an AHP process is to identify the maximum eigenvalue and the corresponding eigenvector, which has the time complexity $O(c^3)$; c is the number of the criteria used to evaluate a service. Sequential operations in the fuzzy TOPSIS mainly include matrix multiplications and distance calculation. The time consumption in this stage depends on the time of the matrix operation. Therefore, for a case with a alternatives (i.e., service candidates) and c criteria, the time complexity is $O(ac)$. Then the time consumption of the non-function rating is $O(c^3) + O(ac)$. Based on the theory of AHP and according to the reality of Cloud services, a service user is usually concerned with a few of specific non-function properties (most probably $5 \leq c \leq 20$). In addition, the previous step of function matching only picks up a few of service candidates (a alternatives, $a \ll n$) that having most similar functions with user requirements. The overall time complexity of func-

tional and non-functional ratings of services is $O(nLv^2 + c^3 + ac)$. Since $L \leq v \ll n$ and $c, a \ll n$, the time complexity of finding and rating the top-K most expected services is approximately as $O(n)$, which is a linear time complexity, depending on the number of services in the service repository.

5.5 A Case Study of Cloud Service Selection through Cloud-FuSeR

I illustrate an example in this section to present the applicability of the proposed framework. To show that the framework is capable of accurately capturing uncertain information in a service selection, I simulate a scenario with precise numerical data, and transfer this crisp scenario to a fuzzy model. Then I compare the ranking results of the precise and the fuzzy models. The inference in the precise model is based on a crisp Cloud storage ontology, and crisp pair-wise comparison and TOPSIS techniques. The scenario is described as: a user requires a Cloud storage service that contains three sub-services: r={troubleshooting service (*tr*), object storage service (*obs*), Internet data transfer service (*inte*)}. For each type of services, as shown in Figure 5.2, there are a range of QoS properties that can be used for describing and evaluating the associated service. In addition to the functional requirements, the user has different preferences on the sub-services and QoS properties.

The first step is to filter service instances according to the functional similarity between user-required services and candidate services. In the fuzzy framework, the similarity calculation is based on the fuzzy Cloud storage ontology [195]; while in the crisp scenario, a crisp ontology is applied, which composed

of the concept taxonomy of the fuzzy ontology, without considering the fuzzy information *FER* and *FIR*. To see the difference between the fuzzy ontology-based and crisp ontology-based similarity calculation, I compare the similarities between two pairs of service functions: *inte-* and *intra-cloud transfer* (*intra*) (represented by *sim1*), and *inte-* and *automatic data migration* (*dm*) (represented by *sim2*). The results are $sim_1^{fuzzy} = 0.9103$, $sim_1^{crisp} = 0.8225$, $sim_2^{fuzzy} = 0.3581$, and $sim_2^{crisp} = 0.4936$, which indicate that the fuzzy similarity model gives higher similarity between services *inte* and *intra*, but less similarity between *inte* and *dm*. From the Cloud storage ontology, *inte* and *intra* have a same set of QoSs, while *inte* and *dm* have different QoSs. The fuzzy model considers the similarity between QoSs, which increase the discrimination between the services with same and different QoSs.

After service filtering, top-5 functional-similar service instances are chosen to be rated by TOPSIS, which are $Simfuzzy(s1, s2, s3, s4, s5) = (1, 1, 0.992, 0.983, 0.962)$ and $Simcrisp(s1, s2, s3, s4, s5) = (1, 1, 0.976, 0.92, 0.904)$. The TOPSIS component evaluates the top-5 service instances based on their QoS values. I simulate the precise numerical QoS values according to real Cloud service data [117], which are standardized using the formula $\frac{x - x_{min} + 0.125(x_{max} - x_{min})}{1.25(x_{max} - x_{min})}$ and are applied in the crisp TOSIS rating procedure. For fuzzy TOPSIS, the standardized values are transferred to fuzzy variables through the fuzzy function in Figure 5.5. The simulated QoS values are shown in Table 5.2.

To conduct the crisp pair-wise comparison, the five rating scales $\{1, 3, 5, 7, 9\}$ proposed by Saaty [178] are applied, corresponding to the fuzzy rating scales $\{EQ, M, S, VS, EX\}$. The calculated importance weights and preference weights

of QoSs and service functions based on the fuzzy procedure are shown in Table 5.2, w^{expert} , w^{user} and $w^{user}_{func'}$ given user's risk-averse attitude. The weights obtained by the crisp pair-wise comparison are represented as $\overline{w}^{user}$, $\overline{w}^{expert}$ and $\overline{w}^{expert}_{func}$ in Table 5.2. The ranking results are shown in Table 5.3, where 'f-' means the ratings based on the fuzzy framework and 'c-' refers to the crisp ratings; *aggregated* is the aggregation of the QoS performance rating and the functional similarity rating. From f-aggregated and c-aggregated, the fuzzy and precise procedures give a similar top-ranking service and a ranking order, which means the proposed fuzzy framework can capture the real service information accurately, so that it can be independently applied to the scenarios having fuzziness and uncertainties.

5.6 Summary

In this chapter, I studied the issues in the Cloud service selection area. At first, I identified the necessity of doing research in Cloud service selection. I then discussed how to select appropriate Cloud services based on the functional and non-functional requirements expressed by fuzzy (i.e., vague or imprecise) knowledge given by users. Against this problem, I proposed a novel fuzzy user-oriented Cloud service selection system (Cloud-FuSeR) to help ordinary services users to select right Cloud services. Cloud-FuSeR includes three primary components: (1) a fuzzy Cloud ontology that supports the similarity calculation of Cloud service concepts and the efficient query of Cloud services or service compositions that most match user-requested functions; (2) a fuzzy AHP approach that calculates the weights of the non-functional properties (i.e., service ranking criteria) in terms of user preference; and (3) a fuzzy TOPSIS ap-

proach that rates the candidate Cloud services based on the weights and the performance of the non-functional properties. I describe a case study of selecting Cloud storage services to present the service selection process, and conducted comprehensive experiments to show the efficiency of the Cloud-FuSeR framework. At last, I theoretically discussed the linear time complexity of this framework.

Our approach is capable of facilitating the automatic transactions in on-line Cloud marketplaces, assisting service users to find Cloud services that meet users' requirements containing fuzzy expressions and unquantifiable decision criteria. The proposed approach and techniques open up some interesting directions for future research. For example, by improving the structure of ontology, I can further simplify the similarity matching. To further improve the performance of Cloud service selection, I can leverage other fuzzy number managing approaches, such as using distance measurement between fuzzy numbers and ranking fuzzy numbers.

CHAPTER 6

EXPLORING CRITERIA INTERDEPENDENCE IN MCDM AND ITS APPLICATION IN CLOUD SERVICE SELECTION

In the previous chapter, I introduced a cloud service selection framework *Cloud-FuSeR* that deals with the fuzzy information to help the service users select appropriate services. However, the Cloud-FuSeR does not consider the interdependencies among decision criteria in the process of cloud service selection. In this chapter, I study such interdependencies of the decision criteria, and explore the influence of the criterion interdependencies on making decisions of the cloud service selection.

6.1 Introduction

With the development of Cloud computing and the proliferation of Cloud services on the Internet, the problem of cloud service selection has become an important and complicated research topic. A number of researchers have developed a series of Cloud service selection or recommendation techniques to help service users make informed investment decisions. Typically, researches in this area focus on solving the following challenges: exact matching between Cloud functional descriptions and functional requirements; QoS performance evaluation of Cloud services; and trust evaluation of Cloud service providers [197]. In this chapter, I discuss the problem of QoS performance evaluation of Cloud services, and explore how to apply the Multi-criteria decision making (MCDM) techniques to Cloud service selection.

MCDM techniques are developed for supporting decision makers making

informed decisions by considering multiple decision criteria [146]. I focus on the MCDM problems with finite decision alternatives and criteria, which can be tackled by MCDM techniques like MAUT [167], ELECTRE [29], and analytic hierarchy process/analytic network process (AHP/ANP) [14]. In other words, the MCDM decision making process is a ranking process of alternatives, where one of the key steps is to identify the relations among criteria and to aggregate the criteria utilities based on their relations to obtain the overall evaluation of alternatives.

The existing MCDM techniques assume that the criteria are independent [23]. However, this assumption does not reflect the fact that there are different types of relations between criteria, and both the individual criterion and the criteria coalitions can influence the ranking results of the alternatives and the final decisions of the MDCM problems. In reality, components in a complex system are interdependent with each other in different forms, and different types of criteria interrelations have different types of influence on the performance of the overall system [114].

Three forms of criteria relations were identified: independent, supportive, and conflicting [118]. Independent criteria do not have relations with each other. Their utilities influence the system performance independently. Supportive criteria are similar with each other in terms of functions, so that they influence each other's utility positively, while their coalition influence the system utility negatively. Conflicting criteria have opposite impacts with each other. The increase of a criterion utility causes the decrease of the utilities of its conflicting criteria. One technique modelling the three types of relations of a decision system is the Fuzzy measure theory [216] that can measure the weights of both the

singleton criteria and the coalitions of criteria. The calculated weights are used to additively aggregate the marginal utilities of criteria to get the overall utilities of alternatives.

Interpretive structural modelling (ISM) [147] is a technique of building relations of elements (e.g., criteria) in a decision system. The constructed element relations represent an organized knowledge that can explain the decision system in a simple way. One disadvantage of ISM is that it can only model the single-type element relations (e.g., i is *preferred* to j; or i supports j), which can cause inaccurate problem solving due to the lack of the complete relation modelling.

From the above discussion, I focus on solving two issues in this chapter: **issue (1)**-identify different types of relations between criteria in a decision system; and **issue (2)**-measure the influence of the individual criterion and the different types of criteria relations on the decision system. I propose a MCDM framework that helps DMs to build the criteria relations and measures the performance of alternatives. The distinctive contributions of this chapter are as below:

- I describe an I-ISM approach that can help decision makers (DMs) construct different types of criteria relations, which allows the DMs adjust the relations interactively during the construction process until a consistent relation network is established. The I-ISM can be used to resolve issue (1).
- Based on the criteria relation constructed by I-ISM, I use 2-order Choquet Integral approach to solve issue (2). The 2-order Choquet Integral [13] is a simplified procedure of the Choquet Integral that can aggregate non-additive utilities by considering the utilities of single criterion and the interactive utilities between two criteria.

- I apply the proposed decision making method to the Cloud service selection problem. I use a User-oriented sigmoid utility function to get the intra-utilities of a criterion w.r.t. different alternatives. The user-oriented sigmoid utility function is flexible enough to reflect Cloud users' requirements in different contexts.
- I conduct extensive experiments based on real QoS performance data of Cloud service providers. Experimental results show the efficiency of our I-ISM.

This chapter is organized as follow: in section 6.2, I discuss the related work in Cloud service selection, and introduce the background knowledge of this work. In section 6.3, I define a user-oriented sigmoid utility function for evaluating Cloud service QoS performance. In section 6.4, I introduce the basic definitions of I-ISM, and explain how the I-ISM can support the non-additive utility aggregation in the 2-order Choquet Integral. I show our experimental results in section 6.5 and conclude this chapter in section 6.6.

6.2 Related Work and Background Knowledge

In this section, I first discuss the related work of Cloud service selection. Then I briefly introduce the background knowledge: Interpretive Structural Modelling (ISM) and 2-order Choquet Integral.

6.2.1 Cloud Service Selection Techniques

I categorize the existing cloud service selection techniques into three main groups [197]: optimization-based, logic-based, and MCDM-based.

Optimization-based. Optimization-based cloud service selection techniques help the DMs find the optimal Cloud services that can minimize or maximize the utilities of criteria under a set of constraints [197]. Wang et al. [214] developed a cloud service selection mechanism DCS that dynamically optimizes the decision making results. Zheng et al. [235] proposed a cloud service ranking prediction framework, named CloudRank in which two greedy algorithms are designed to predict the service performance and rank the Cloud services. Li et al. [129] focused on the cloud service selection problems in hybrid Cloud environment, proposing a two-level optimization procedure of hybrid Cloud service selection.

Logic-based. I call the methods based on the semantic techniques for Cloud service description and matching as logic-based methods. Singh et al. [190] designed an ontology of manufacturing resources in Cloud manufacturing system to support the querying tools based on customers' requirements. Fang et al. [62] designed a cloud service ontology model based on OWL2 (Web Ontology Language), which can be dynamically and collaboratively maintained. The ontology model supports Cloud service querying that can deals with fuzzy information and has high agility. Li et al. [130] introduced a context ontology model to describe SaaS services. The proposed ontology model supports the discovery and recommendation of the SaaS services according to the behavior habits and requesting contexts.

MCDM-based. MCDM methods rank alternatives based on the performance of a variety of criteria. Whaiduzzaman et al. [219] reviewed and compared the multi-criteria decision analysis techniques in Cloud service selection. Rehman et al. [172] introduced a MCDM method for Cloud service selection based on the historical information of Cloud services. The proposed method processes the historical service information in various time periods in an independent and parallel way. Lee et al. [124] introduced a hybrid MCDM model for cloud service selection based on the techniques of fuzzy analytical hierarchy process (FAHP), fuzzy Delphi method (FDM), and balanced scorecard (BSC).

Our work is different with the above discussed techniques by supporting an interactive process of determining different types of criteria relations, and measuring the alternative utilities based on non-additive criteria performance. In the next subsection, I introduce the background knowledge of our work.

6.2.2 Background Knowledge

Interpretive Structural Modelling

The ISM procedure [97] is conducted based on the mapping of binary matrices and digraphs. The basic components of an ISM include a subordinate relation (represented as R) and a set of elements (represented as $E = \{e_i | i = 1, \cdots, n\}$) that need to be partially ordered through the relation, both of which are determined by domain experts or decision makers according to the contexts of the targeted issues. I define that $e_i R e_j$ if e_i is of direct subordinate relation to e_j, or else $e_i \overline{R} e_j$. I can use a directed graph (G) to represent such a partial ordering, in which vertices ($V = \{v_i\}$) and arcs (represented as $A = \{a_{ij}, j = 1, \cdots, n\}$) correspond to

the elements and the subordinate relation respectively. Element e_i is reachable to e_j means that e_i can be connected to e_j via a directed path p, represented as $v_i p v_j$. The length of the path ($l(p)$) is defined as the number of arcs between vertices v_i and v_j in the corresponding digraph. If $l = 1$, v_i is directly connected to v_j; if $l > 1$, v_i is connected to v_j via transitive relations. The process of ISM are as follows:

Step 1. Identify elements E. Identify factors related to a complex system based on expert opinions. These factors have certain relations with each other based on the objectives of the complex system.

Step 2. Identify contextual relation R. As mentioned before, a contextual relation is a subordinate relation between two elements, such as 'influence', 'lead to', and 'prefer to'.

Step 3. Develop structural self-interaction matrix (SSIM). SSIM shows the existence and the direction of a direct relation between two elements by using four symbols: $SSIM = [M_{ij}]$, $m_{ij} = V$, if $e_i R e_j$; $m_{ij} = A$, if $e_j R e_i$; $m_{ij} = X$, if $e_i R e_j$ and $e_j R e_i$; $m_{ij} = O$, if $e_i \overline{R} e_j \& e_j \overline{R} e_i$.

Step 4. Develop reachability matrix (RM). RM is a binary matrix built from SSIM to represent the reachability between elements. First, SSIM is converted to a binary matrix: $BM = [b_{ij}], b_{ij} = 1$; $b_{ij} = 1 \& b_{ji} = 0$, if $m_{ij} = X$; $b_{ji} = 1 \& b_{ij} = 0$, if $m_{ij} = A$; $b_{ij} = 1 \& b_{ji} = 1$, if $m_{ij} = X$; $b_{ij} = 0 \& b_{ji} = 0$, if $m_{ij} = O$, where $b_{ii} = 1$ means that an element is always reachable to itself. Then $RM = BM^k * BM = BM^{k+1} * BM = \cdots$.

Step 5. Classify elements based on Driver power and dependence. Identify the driver power and the dependence of an element, and then classi-

fy the elements into four groups: autonomous, dependent, linkage, and driver/independent. The driver power of an element is the number of elements it can reach; dependence of an element is the number of its antecedents. Each of them can determine the importance degree of an element for a system (e.g., drivers are more important than others).

Step 6. Hierarchically partition elements. Rank the elements according to their driver power and dependence, and identify the reachable set and the antecedent set of each elements. Partition elements hierarchically based on their reachable and antecedent elements.

Step 7. Form ISM digraph. In the ISM digraph, vertices represent elements, and arcs between vertices refer to the relations between elements.

K-order Additive Choquet Integral

A fuzzy integral is a kind of utility aggregation operator that is capable of measuring the influence of the importance of a criterion and the importance of interactions among criteria [208]. A set of importance values needs to be defined to calculate the fuzzy measure that is a set of importance values for all the subsets of a set of elements (e.g., decision criteria and a group of players in a game). The critical step of a fuzzy measure application is to precisely define a fuzzy measure.

Though the fuzzy integral shows more rationality and richness compared with the additive measures (e.g., simple weighted additive, SWA), it has not got enough enthusiasm on application side due to its complexity of determining the hidden fuzzy measures [78]. For example, suppose X is a set of n elements

($|X| = n$). $pow(X)$ represents all the subsets of X. For a $A \in pow(X)$, $\mu(A)$ is the importance of the coalition A for the decision problem. Defining a fuzzy measure on X requires identifying 2^n positive real coefficients, which would become very complex when n is high (SWA only requires n coefficients). Grabisch [78] introduced the k-order additive fuzzy measure to reduce the complexity. In real applications, the weight of singleton criterion and the interaction weight of a pair of criteria are important, and it is very difficult for DMs to determine the importance degree of three or more interactive criteria. Therefore, 2-additive fuzzy measure can be handled easily (only $n(n+1)/2$ parameters need to be identified by using Mobius representation [78]).

According to the monotonicity of a fuzzy measure, two types of set functions are discussed: monotonic fuzzy measure and non-monotonic fuzzy measure. Murofushi [159] argued that the monotonicity is not essential for fuzzy measures. They analysed the non-monotonicity in terms of the offset of disjoint subsets, and proved that it is unnecessary to keep monotonicity for fuzzy measures from a mathematical point of view. In this chapter, I use the 2-additive fuzzy measure with respect to non-monotonic set functions. Basic definitions and properties are introduced below:

Let $A = \{s_1, s_2, \cdots, s_m\}$ be a set of alternatives; $C = \{c_1, c_2, \cdots, c_n\}$ be a set of decision criteria, where $c_i : A \rightarrow R, i \in \{1, \cdots, n\}$; $U(C) = U(c_1, c_2, \cdots, c_n)$ represents the utility of an alternative based on DMs' preference on criteria C, $u_i(c_i(s_k))$ is a marginal utility of criterion i with respect to alternative s_k; and (C, Γ) is a measurable space, where $\Gamma = pow(C)$,

- A non-monotonic fuzzy measure on (X, Γ) is a real-valued set function $\mu : \Gamma \rightarrow R$ satisfying $\mu(\Phi) = 0$ and $\mu(X) < \inf$ [159].

- A Möbius representation of a fuzzy measure is defined as a function a: $\Gamma \rightarrow R$, for each $C \subset X$, $a(C) = \Sigma_{T \subset C}(-1)^{|C-T|}\mu(T)$. If for $k \in R$, $a(T) = 0$ for $|T| > k$, then the fuzzy measure is called k-additive.
- Based on the definition of Möbius representation, for a singleton criteria set, e.g., $\{c_i\}$, $\mu(\{c_i\}) = a(\{c_i\})$; for a couple of criteria (non-ordered), e.g., $\{c_i, c_j\}$, then $\mu(\{c_i, c_j\}) = \mu(c_i) + \mu(c_j) + a(\{c_i, c_j\})$. For a criteria set C with any number of elements, its 2-additive fuzzy measure is $\mu(T) = \Sigma_{c_i \in T} a(\{c_i\}) + \Sigma_{\{c_i, c_j\} \subset T} a(\{c_i, c_j\}), T \subset C$.
- The Möbius representation of the Choquet integral of an alternative in the case of 2-additive is $Ch_\mu(e_k) = \Sigma_{c_i \subset T}(a\{c_i\})u_i(c_i(e_k)) + \Sigma_{\{c_i, c_j\} \subset T} a(\{c_i, c_j\})$ $min\{u_i(c_i(e_k)), u_j(c_j(e_k))\}$.

6.3 A user-oriented Sigmoid Utility Function for Intra-criterion

In this section, I introduce a sigmoid utility function [118] to elastically measure the satisfaction degrees on criteria based on users' requirements and in specific application contexts.

The utility of a criterion for service evaluation reflects the satisfaction degree of the service user to the criterion performance value. Labreuche and Grabisch [115] described a procedure of determining the utility functions of the *ith* criterion (represented as u_i) depending on both the ranking of the elements of $X\rfloor_i$ and the satisfaction-degree difference between element pairs of $X\rfloor_i$. The procedure is defined by four necessary conditions:

$Intra_a$. $\forall x_i^1, x_i^2 \in X_i$, $u_i(x_i^1) \geq u_i(x_i^2) \Leftrightarrow (x_i^1, 0_{-i}) \geq (x_i^2, 0_{-i})$;

$Intra_b$. $\forall x_i^1, x_i^2, x_i^3, x_i^4 \in X_i$, assume $u_i(x_i^1) > u_i(x_i^2)$ and $u_i(x_i^3) > u_i(x_i^4)$, then

$Ra_{x^{1\sim4}} = \frac{u_i(x_i^1)-u_i(x_i^2)}{u_i(x_i^3)-u_i(x_i^4)} = \frac{u(x_i^1,0_{-i})-u(x_i^2,0_{-i})}{u(x_i^3,0_{-i})-u(x_i^4,0_{-i})}$, where $Ra \in \Re^+$;

$Intra_c$. $u_i(0_i) = 0$ and $u_i(1_i) = 1$, where 0_i(resp. 1_i) represents the value of criterion i of an alternative satisfies (resp.cannot satisfy) the user;

$Intra_d$. $\forall x_i^1, x_i^2, x_i^3, x_i^4, x_i^5, x_i^6 \in X_i$ where $u_i(x_i^1) > u_i(x_i^2)$, $u_i(x_i^3) > u_i(x_i^4)$ and $u_i(x_i^5) > u_i(x_i^6)$, this condition ensures the consistency of the satisfaction differences among criteria assigned by the user: $Ra_{x_{1\sim4}} \times Ra_{x_{3\sim6}} = Ra_{x_{1,2,5,6}}$.

Based on $Intra_{a\sim d}$, Labreuche and Grabisch [115] built u_i of the *ith* criterion by setting $x_i^2 = x_i^4 = 0_i$, and $x_i^3 = 1_i$. This method determines the marginal utilities of the alternatives based on the overall performance values of criterion i. However, the question is " **can u_i defined by $Intra_{a\sim d}$ efficiently reflect users' satisfaction degree based on their specific service requirements ?"** .

For example, assume a user requires that a candidate cloud service should have at least 95% availability (abbr. av). If service j has 94% av, it may be irrational to set $u_{av}(j_{av}) = u_{av}(0_{av}) = 0$ (based on $Intra_c$), because the marginal performance of j in terms of criterion av is still quite high (though it is lower than user's expecting lower bound). On the other hand, if services j and g has 96% and 97% av respectively, the DM may not have similar satisfaction differences among the three av values: $u_{av}(96\%) - u_{av}(95\%)(= | \neq)u_{av}(97\%) - u_{av}(96\%)$. In reality, as long as the availability performance is higher than user's expected performance, its improvement may not have too much positive influence on the overall performance of the alternative given the performance of the other criteria, i.e., $u_{av}(96\%) - u_{av}(95\%) \geq u_{av}(97\%) - u_{av}(96\%)$ given $u_i, i \in C, i \neq av$. Therefore, an elastic utility function is needed to measure users satisfiability for different criteria and application contexts according to their specific requirements.

I now introduce a sigmoid utility function. This utility function satisfies the

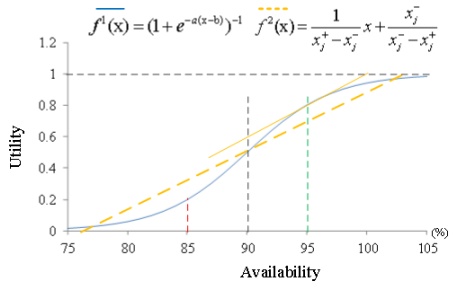


Figure 6.1: A sigmoid utility function of criterion *av* for Cloud services. The user expected lowest *av* is 95%

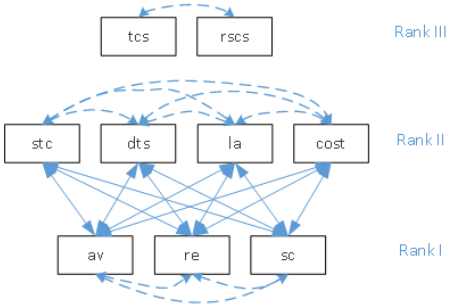


Figure 6.2: A sigmoid utility function of criterion *av* for Cloud services. The user expected lowest *av* is 95%

condition $Intra_a$ and a relaxed condition of $Intra_c$. It also normalizes the value of each criterion to the range (0, 1) in order to ensure a unique and consistent scales for criteria utilities. Based on criterion i, a marginal sigmoid utility function of an alternative is defined as:

$$u_i(x_i) = \begin{cases} (1 + e^{-a(x_i-b)})^{-1}, i \in C_{\mathcal{B}} \\ 1 - (1 + e^{-a(x_i-b)})^{-1}, i \in C_C \end{cases} \tag{6.1}$$

where x_i is the performance value, $C_{\mathcal{B}}$ refers to a set of benefit criteria, and C_C is a set of cost criteria. a controls the steepness (i.e., the changing rate of a criterion utility), and b indicates the median of the function in terms of x-axis. An example of the sigmoid utility function is shown in Fig.6.1. Readers can refer to [118] for more details and examples of the user-oriented sigmoid utility function.

6.4 Criteria Relational Network Construction based on I-ISM

6.4.1 Basic Definitions and an Interactive Structure Construction of I-ISM

To determine relations among decision criteria, decision makers typically pair-wisely compare alternatives or criteria, and directly assign difference thresholds between a pair of elements. Although such kind of method may be easy to apply when there are few alternatives and criteria, usually under 10 [97], users may feel confused in the pair-wise ranking process when the number of alternatives and criteria are more than 10. Therefore, the inconsistency of constraints can cause an optimization problem without any solutions. To help decision makers identify the relations between criteria and between alternatives based on their preferences, I propose an Interactive Interpretive Structural Modelling (I-ISM) method that can determine and modify the element relations by consistently interacting with decision makers in the modelling process.

Typically, four types of 'leads to' relations are modelled by an ISM, distinguished by the direction of a relation between two criteria (e.g., criteria i and j): 1) i influences j; 2) j influences i; 3) mutual influence between i and j; and 4) i independent with j. Specific to the area of multi-criteria decision making, the definition and analysis regarding conflicting and supportive criterion relations in [28] indicate that in a decision making scenario, a relation between a couple of criteria is usually symmetric, that is, if criteria i and j are supportive (or conflicting), then the increase of i's performance can lead to the increase (or decrease) of j's, and vice versa. Therefore, it is rational to assume that the

resulting criteria relational structure is symmetric, i.e., a rational interactive relation model should only have the symmetric influence and the independent relations between criteria. Moreover, I propose to integrate the supportive and conflicting characteristics of relations into the ISM in order to present a clear picture of the types of criteria relations, and to facilitate the identification of fuzzy measures. I will introduce some basic definitions and operations of I-ISM. Let $C = \{c_1, c_2, ..., c_n\}$ be a set of criteria, $F = \{f_1, f_2, ..., fn\}$ be the performance of the criteria, and $SC \subset C$ be a subset of C. For $\forall c_i, c_j \in C$:

Definition 6.1. *An interactive relation from c_i to c_j is defined as a value pair $r_{ij} = (s_{ij}, x_{ij})$, where $s_{ij} \in \{-1, 0, 1\}$ indicates the type of the relation, i.e., how c_i influences c_j; and $x_{ij} \in [0, 1]$ is the interactive degree of the relation, i.e., how important the relation is to the whole system. Details are specified as below:*

- $r_{ij} = (s_{ij} = -1, x_{ij} \neq 0)$*: c_i negatively influences c_j, i.e., $f_i \downarrow (resp. \uparrow) \Rightarrow f_j \uparrow (resp. \downarrow)$, namely c_i* ***conflicts to*** *c_j, represented as r^c_{ij};*
- $r_{ij} = (-1, x_{ij} \neq 0) \& r_{ji} = (-1, x_{ji} \neq 0)$*: c_i and c_j negatively influence each other, i.e., $f_i \downarrow (resp. \uparrow) \Leftrightarrow f_j \uparrow (resp. \downarrow)$, namely c_i is* ***conflicting with*** *c_j, represented as r^{cc}_{ij};*
- $r_{ij} = (s_{ij} = 1, x_{ij} \neq 0)$*: c_i positively influences c_j, i.e., $f_i \uparrow (resp. \downarrow) \Rightarrow f_j \uparrow (resp. \downarrow)$, namely c_i* ***supports*** *c_j, represented as r^s_{ij};*
- $r_{ij} = (1, x_{ij} \neq 0) \& r_{ji} = (1, x_{ji} \neq 0)$*: c_i and c_j positively influence each other, namely c_i* ***is supportive with*** *c_j, represented as r^{ss}_{ij};*
- $r_{ij} = (0, 0)$*: c_i* ***is independent with*** *c_j, represented as r^u_{ij}.*

Property 6.1. $r_{ij} = r_{pq} \Leftrightarrow x_{ij} = x_{pq} \& s_{ij} = s_{pq}$; $r_{ij} = -r_{pq} \Leftrightarrow x_{ij} = x_{pq} \& s_{ij} = -s_{pq}$;

Definition 6.2. *Given a relation r_{ij}, if $r_{ij} = r_{ji}$, then r_{ij} is a symmetric relation, and $changeOf(f_i) \Leftrightarrow changeOf(f_j)$, represented as $c_i \overleftrightarrow{r_{ij}} c_j$.*

The relations *support* and *conflict to* are asymmetric relations; *supportive with*, *conflicting with*, and *independent with* are symmetric relations. In this work, I focus on the symmetric relations. I now illustrate the following definitions and properties based on the assumption of symmetric relations.

Property 6.2. $r_{ii} = r_{ii}^{ss} = (1, 1)$, *i.e., a relation is supportive with itself with the interactive degree of* 1.

Property 6.3. *For* $\forall c$, *if* r_{ij}^{ss}, $f_{ij} < f_i + f_j$; *if* r_{ij}^{cc}, $f_{ij} = f_i + f_j$, *where* f_{ij} *is the importance of the interaction between* c_i *and* c_j *to the system.*

Property 6.4. *For two relations* r_{ij} *and* r_{pq}, *if* $x_{ij} > x_{pq}$, *for* $\forall s_{ij} \& s_{pq}$, *the importance of the interaction between criteria* c_i *and* c_j *for the system is higher than the importance of the interaction between criteria* c_p *and* c_q.

The concept of path and the length of path in I-ISM are similar to the concepts in ISM. I give their formal definitions as below.

Definition 6.3. *A path from* c_i *to* c_j, *written as* p_{ij}, *indicates a relation between them (i.e.,* r_{ij}*), containing a series of linked elements between* c_i *and* c_j. *The length of* r_{ij}, *represented as* l_{ij}, *is the number of arcs between* c_i *and* c_j. *If* $l_{ij} = 1$, r_{ij} *is a direct relation, represented as* $c_i r_\phi c_j$; *if* $l_{ij} > 1 \& l_{ij} < +\infty$, r_{ij} *is a transitive relation, represented as* $c_i r_{\{k, c_1 ... c_k\}} c_j$, *where* $c_1, ..., c_k$ *are elements on the path* p_{ij}, k *is the number of elements; if* $l_{ij} = +\infty$, r_{ij} *is* r_{ij}^u *(represented as* $c_i r_\infty c_j$*).*

In the following, I use $c_i r c_j$ to represent any type of a relation (i.e., $r_\phi, r_{\{...\}}$ or r_∞) between c_i and c_j. In addition, in the case that I only concern about the type and direction of a relation, but not concern about the interaction degree, I also use $c_i s c_j$ to represent $c_i r c_j$, e.g., if $r_{ij}^s = \{1, \forall x_{ij}\}$, then $c_i r_{ij}^s c_j = c_i \{1, \forall x_{ij}\} c_j = c_i \overrightarrow{1} c_j$.

Property 6.5. *Let m be the number of paths between element c_i and c_j, all the paths between c_i and c_j are represented as: $P_{ij} = \{p_{ij}^1 \cup p_{ij}^2 \cup ... \cup p_{ij}^m\}$. If $m = 0$, $r_{ij} = r_{ij}^u$.*

To identify stable relations between criteria, I define two context-aware logical operators *logic addition*($\overline{\vee}$) and *logic multiplication*($\overline{\wedge}$), and relational operators *relation addition*($\vee$) and *relation multiplication*($\wedge$). These logical operators are used to operate the Characteristic matrix of an SSIM to determine the types of relations. $\overline{\vee}$ and $\vee$ are for aggregating two relational paths between two vertices to determine a unique relation between the two criteria. $\overline{\wedge}$ and $\wedge$ are for aggregating a set of relations on a path to find a transitive relation between the two vertices. They are defined as:

Given a set of variables $s_1, s_2, ..., s_n \in \{-1, 0, 1\}$,

Definition 6.4. *Logic addition ($\overline{\vee}$) satisfies:*

- $s_i \overline{\vee} 0 = s_i$;
- $1 \overline{\vee} 1 = 1$, $-1 \overline{\vee} -1 = -1$, $1 \overline{\vee} -1 = \infty$;
- $\overline{\vee}$ *satisfies left associativity:* $s_i \overline{\vee} s_j \overline{\vee} s_k = (s_i \overline{\vee} s_j) \overline{\vee} s_k, i, j, k \in \{1, ..., n\}$.

Definition 6.5. *Relational addition ($\vee$) satisfies:* $r_{ij}^1 \vee r_{ij}^2 \vee ... \vee r_{ij}^m = ((s_{ij}^1 \overline{\vee} s_{ij}^2 \overline{\vee} ... \overline{\vee} s_{ij}^m), max\{x_{ij}^1, x_{ij}^2, ..., x_{ij}^m\})$.

From the definition of $\vee$, if there are m relational paths between vertices c_i and c_j, then their interaction degree $x_{ij} \geq \forall x_{ij}^k, k \in \{1, ..., m\}$.

Definition 6.6. *Logic multiplication ($\overline{\wedge}$) satisfies:*

- $s_i \overline{\wedge} 0 = 0$;

- $1\overline{\wedge}1 = 1, -1\overline{\wedge} - 1 = 1, 1\overline{\wedge} - 1 = -1$;
- $\overline{\wedge}$ *satisfies left associativity:* $s_i\overline{\wedge}s_j\overline{\wedge}s_k = (s_i\overline{\wedge}s_j)\overline{\wedge}s_k, i, j, k \in \{1, ..., n\}$.

Definition 6.7. *Relational multiplication* ($\wedge$) *satisfies: for the kth path between* c_i *and* c_i, $r^k_{i,i+1} \wedge r^k_{i+1,i+2} \wedge ... \wedge r^k_{j-1,j} =$
$((s^k_{i,i+1}\overline{\wedge}s^k_{i+1,i+2}\overline{\wedge}...\overline{\wedge}s^k_{j-1,j}), min\{x^k_{i,i+1}\overline{\wedge}x^k_{i+1,i+2}\})$.

The definition of $\wedge$ indicates that if a relation r_{ij} is a transitive relation, then $x_{ij} \leq x_{pq}$, where $x_{pq} \in \{x_{i,i+1}, x_{i+1,i+2}, ..., x_{j-1,j}\}$.

Definition 6.8. *A SSIM of I-ISM is defined as* $SSIM = (ChM, InD)$; $ChM = \begin{pmatrix} s_{11} & \cdots & s_{1n} \\ \vdots & \ddots & \vdots \\ s_{n1} & \cdots & s_{nn} \end{pmatrix}$ *is the Characteristic Matrix of SSIM; and* $ChM = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$ *is the interactive matrix of SSIM.*

Property 6.6. $SSIM * SSIM = (ChM * ChM, InD * InD)$.

Property 6.7. *A SSIM is consistent. An RM is stable (i.e.,* $RM = RM^k = RM^{k+1}$*), consistent, and symmetric (i.e.,* $\forall c_i, c_j, r_{ij} = r_{ji}$*).*

Property 6.8. *For a set of relations* $R = \{r_{ij}\}$*, if one of the following two cases exists, there is inconsistency in the relations: 1)*$\exists r_{ij} = -r_{ij}$*; 2)if there are m paths between* c_i *and* c_j, $\exists r^a_{ij} \neq r^b_{ij}$, $a, b \in \{m\}$*, then the corresponding SSIM is inconsistent.*

I use four criteria $\{c_1, c_2, c_3, c_4\}$ to explain the usage of the logic operations. Fig.6.3(a) is an initial $SSIM$ given by an decision maker, which shows that:

- $c_1 \overleftrightarrow{1} c_4$: c_1 and c_4 are directly supportive with each other;
- $c_1 \overrightarrow{-1} c_3$: c_1 directly conflict to c_3;

- $c_1 \overleftrightarrow{1} c_4 \overleftrightarrow{1} c_3 \Rightarrow c_1 \overleftrightarrow{1 \overline{\wedge} 1} c_3$: c_1 and c_3 are transitively supportive with each other via c_4;
- Inconsistencies:
  - based on 2) and 3), $c_1 \overrightarrow{-1 \cup 1^2_{c_4}} c_3 \Rightarrow c_1 \overrightarrow{-1 \overline{\vee} 1} c_3 \Rightarrow c_1 \overrightarrow{\infty} c_3$: the relation from c_1 to c_3 cannot be determined due to the conflicting relation between their direct relation $\overrightarrow{-1}$ and their transitive relation via c_4.
  - $c_1 \overrightarrow{-1_{\{1,c_3\}}} c_2$ & $c_2 \overrightarrow{1_{\{2,c_3 c_4\}}} c_1 \Rightarrow c_1(-1 \overline{\vee} 1)c_2 \Rightarrow c_1(\infty)c_2$: the relation between c_1 and c_2 are not symmetric.

Inconsistency is incident to a manually assigned relation matrix. According to the theory proposed by Saaty [179], if the number of considered criteria is over 7, it is quite possible that the decision makers feel confused to decide the pair-wise relations between criteria. However, establishing consistent relations is essential for guaranteeing a consistent and stable reachability matrix. Therefore, it is necessary to develop an interactive way to help the decision makers check and adjust the relation matrix to avoid its inconsistency. Fig. 6.3 shows an example of an interaction process. I illustrate it as below:

Using the logic operations, we raise the power of $SSIM$ to identify the transitive relations, until the matrix reaches its stable state.

- Check the direct inconsistent relations in $SSIM$. If there are direct inconsistencies, return the position of the inconsistent relations, and then decision makers revise the $SSIM$, e.g., if the initial $SSIM$ is:ChM_0 =

$$\begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{pmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \end{array}$$, the revised consistent matrix is $ChM_1 =$

$$\begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{pmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 1 & 0 \\ -1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \end{array}$$, which corresponds to Fig. 6.3a;

- Or else, raise the power of ChM, and record the additive vectors and the median matrix in each power-raised step, i.e., $ChM^k, k = \{2, 3, ...\}$.
- If there are inconsistencies in the vectors or in the median matrix, the raising process will be terminated, and the conflicting elements will be detected and be returned to decision makers as feedbacks, e.g., $ChM^0 =$

$$\begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{pmatrix} 1 & -1 & \infty & \infty \\ -1 & 1 & 1 & 1 \\ \infty & 1 & 1 & \infty \\ \infty & 1 & \infty & 1 \end{pmatrix} \end{array}.$$

It shows that the relations among c_1, c_3, c_4 are inconsistent. Therefore, the additive vectors for the two positions are returned to the decision makers, e.g., the additive vector for position (c_3, c_1) in ChM_2 is calculated as: $c_3?c_1 \Leftarrow (-1, 1, 1, 1)\overline{\wedge}(1, 0, -1, 1)^T = ((-1\overline{\wedge}1)\overline{\vee}(1\overline{\wedge}0)\overline{\vee}(1\overline{\wedge} - 1)\overline{\vee}(1\overline{\wedge}1)) = (-1\overline{\vee}0\overline{\vee} - 1\overline{\vee}1) \Rightarrow c_3 r_\infty c_1$.

The meaning of the additive vector $(-1\overline{\vee}0\overline{\vee} - 1\overline{\vee}1)$ is: $c_3\overrightarrow{-1}c_1$, $c_3\overrightarrow{0_{\{1,c_2\}}c_1}$,

$c_3\overrightarrow{-1}c_1$, $c_3\overrightarrow{1_{\{1,c_4\}}}c_1$. ChM_1 shows that the transitive relation from c_3 to c_1 through c_4 is conflict with the direct relation from c_3 to c_1.

For positions (c_1, c_3), (c_1, c_4), (c_3, c_4), (c_4, c_1), and (c_4, c_3), the additive vectors are $c_1 r_\infty c_3 \Leftarrow (-1\overline{\vee}0\overline{\vee} - 1\overline{\vee}1)$; $c_1 r_\infty c_4 \Leftarrow (1\overline{\vee}0\overline{\vee} - 1\overline{\vee}1)$; $c_3 r_\infty c_4 \Leftarrow (-1\overline{\vee}0\overline{\vee}1\overline{\vee}1)$; $c_4 r_\infty c_1 \Leftarrow (1\overline{\vee}0\overline{\vee} - 1\overline{\vee}1)$; $c_4 r_\infty c_3 \Leftarrow (-1\overline{\vee}0\overline{\vee}1\overline{\vee}1)$.

- The experts adjust the relation matrix based on the returned additive vectors and the powered matrix, e.g., the six additive vectors in last step shows that the inconsistencies are: $c_1\overleftrightarrow{r_{\{1,c_4\}}}c_3$, $c_1\overleftrightarrow{r_{\{1,c_3\}}}c_4$, $c_3\overleftrightarrow{r_{\{1,c_1\}}}c_4$. Thus, the relations among c_1, c_3, and c_4 need to be reviewed and adjusted by decision makers, e.g., suppose after reviewing, the relation is changed to $c_3\overleftrightarrow{-1}c_4$ (see Fig. 6.3b).

- Restart the powering process based on the new matrix ChM^1, until the stable matrix is achieved and there is no feedback in the process. The reachability matrix is obtained, e.g., the final stable and symmetric reachability matrix is $RM * ChM = ChM^3 = ChM^4 = ... = ChM^\infty =$

$$\begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{pmatrix} 1 & & & \\ -1 & 1 & & \\ -1 & 1 & 1 & \\ 1 & -1 & -1 & 1 \end{pmatrix} \end{array}.$$

The reachability diagram is shown in Fig. 6.3c. We can see that the relation $c_4\overleftarrow{-1_{\{1,c_1\ or\ c_3\}}}c_2$ is an identified transitive relation, and $c_1\overleftrightarrow{-1}c_2$ is an identified direct relation.

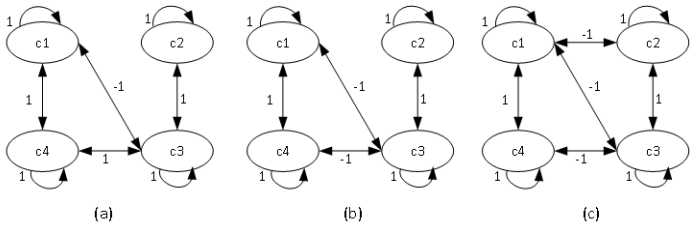


Figure 6.3: Interactive relations among criteria c_1, c_2, c_3, c_4

6.4.2 Supportive and Conflict Power

After establishing the reachability matrix, elements in ISM will be partitioned based on the driver power and the dependence of one sub-ordinate relation. To partition the elements in I-ISM, I additionally consider the r_{ij}^{ss} and r_{ij}^{cc}. Since I focus on the symmetric relations between criteria in this work, the driver power of a criterion equals to its dependence. Therefore, the driver power and the dependence cannot be used to partition the criteria. I define here the **supportive power** and the **conflict power** to measure r_{ij}^{ss} and r_{ij}^{cc}.

Definition 6.9. *Let N be a set of criteria for a decision system, $i, j \in N, i \neq j$. The **supportive power** of i is defined as $S(i) = \Sigma_{j \in N} x_{ij}^{ss}$. The **conflict power** of i is defined as $Cf(i) = \Sigma_{j \in N}(x_{ij}^{cc})$.*

An example of the SSIM of the nine criteria of a cloud service is shown in Table 6.1. We raise the power of SSIM to get a stable Reachability Matrix (RM in Table 6.3). We can see that $S(av) = 3$, and $Cf(re) = 4$.

Table 6.1: SSIM of Nine Criteria of a Cloud Service

SSIM	av(0)	re(1)	sc(2)	stc(3)	dts(4)	la(5)	tcs(6)	rscs(7)	cost(8)
av(0)	1	1	1	−1	0	0	0	0	0
re(1)	1	1	0	0	0	0	0	0	0
sc(2)	0	0	1	−1	0	0	0	0	0
stc(3)	−1	0	−1	1	0	0	0	0	1
dts(4)	0	0	0	0	1	1	0	0	1
la(5)	0	0	0	0	1	1	0	0	0
tcs(6)	0	0	0	0	0	0	1	1	0
rscs(7)	0	0	0	0	0	0	1	1	0
cost(8)	0	0	0	1	1	0	0	0	1

6.4.3 Criteria Partition based on Reachability Matrix

In this section, I introduce a partition procedure for the I-ISM according to the triplet relations, which is an extension the the partition procedure of ISM. To conveniently inspect and construct the relations among criteria, ISM has a partition procedure that can group criteria according to their binary influence relations based on the reachability matrix.

Block Partition: $\Pi_B = (B_1, ..., B_h)$, where $B_l = \{i \in A - B_0 - B_1 - ... - B_{l-1} | r_{ij}^{ss}, \forall j \in B_l$, and $r_{ik}^{cc} || r_{ik}^{u}, \forall k \notin B_l\}$.

For example, the relation partition induced by the reachability matrix RM in Table 6.3 is $\Pi_R = \{\{0, 1, 2\}, \{3, 4, 5, 8\}, \{6, 7\}\}$.

If we assume that the relation value between criteria i and j are $x_{ij}^{cc} = x_{ij}^{ss} = 1$

and $x_{ij}^{u} = 0$. Based on the concepts of conflicting power and supportive power, we can extend the *block partition* to *level partition*. I first introduce a theorem for level partition.

Theorem 6.1. $\forall i, j \in A$, $Cf(i) - S(i) \geq Cf(j) - S(j) \Rightarrow i \geq j$; $Cf(i) + S(i) \geq Cf(j) + S(j) \Rightarrow i \geq j$, *where* $Cf(i) = \{k, r_{ik}^{cc}, \forall k \in A\}$, *and* $S(i) = \{k, r_{ik}^{ss}, \forall k \in A\}$.

Proof. Based on the definitions of *supportive with* and *conflicting with*, the more a criterion being conflicting with the other criteria, the more important the criterion is to the system; while the more a criterion being supportive with the other criteria, the less important the criterion is to the system. Therefore, the higher the value of $Cf - S$, the higher importance degree (or satisfaction degree) the criterion has.

Based on the definitions of *driver power* and *dependence*, the more a criterion being connective with other criteria (i.e., having higher driver power and dependence), the more important the criterion is. Therefore, the higher the value of $Cf + S$, the higher importance degree (or satisfaction degree) the criterion has. □

Given the block partition of a set of criteria, and the assumption that $x_{ij}^{cc||ss} = 1$ and $x_{ij}^{u} = 0$, I define a hierarchic partition to rank the importance degree of criteria for the decision system.

Hierarchic Partition: $\Pi_H = (H_1, ..., H_h)$, where $H_l \in \{i \in A - H_0 - H_1 - \cdots - H_{l-1} | Cf(i) = Cf(j) \& S(i) = S(j), \forall j \in H_l; Cf(i) < Cf(k) \& S(i) > S(k), \forall k \in H_g, g < l\}$.

The values of $Cf - S$ and $Cf + S$ of criteria are used to rank the criteria respectively. Table 6.3 shows the ranking of nine criteria for Cloud service evaluation

based on the values of $Cf-S$ or $Cf+S$. The tab *Rank* indicates that criteria *av*, *re* and *sc* are most important. The criterion ranking hierarchies can be established based on the tab *Rank*, which is shown in Fig. 6.2. In Fig. 6.2, the arcs are all bi-direction, representing the symmetric relation between criteria. The arcs with dotted lines represent the supportive relations between criteria, while the arcs with solid lines represent the conflict relations.

Table 6.2: RM and Ranking of Nine Criteria of a Cloud Service

RM	0	1	2	3	4	5	6	7	8	S	Cf	$Cf-S$	$Cf+S$	Rank
0	1	1	1	-1	-1	-1	0	0	-1	3	4	1	7	I
1	1	1	1	-1	-1	-1	0	0	−1	3	4	1	7	I
2	1	1	1	-1	-1	-1	0	0	-1	3	4	1	7	I
3	-1	-1	-1	1	1	1	0	0	1	4	3	-1	7	II
4	-1	-1	-1	1	1	1	0	0	1	4	3	-1	7	II
5	-1	-1	-1	1	1	1	0	0	1	4	3	-1	7	II
6	0	0	0	0	0	0	1	1	0	2	0	-2	2	III
7	0	0	0	0	0	0	1	1	0	2	0	-2	2	III
8	-1	-1	-1	1	1	1	0	0	1	4	3	-1	7	II

6.4.4 I-ISM in 2-order Additive Choquet Integral

As I introduced in previous section, I-ISM is capable of determining the relations between criteria, which can help the modelling of 2-additive capacity identification in Fuzzy measure theory. Based on the definition of supportive and conflicting relations among criteria, I define property 6.9 that indicates the

influence of different types of criteria relations on the decision system.

Property 6.9. *Given a set of mutually supportive criteria N_{SS} and a set of mutually conflicting criteria N_{CC}, let the interaction index of a criteria coalition $c_1, ..., c_k$ be $I^{\mu}_{1,...,k}$. then* $-1 \leq \{I_\mu(T_2)|\forall T_2 \subset N_{SS}\} \leq 0 \leq \{I_\mu(T_1)|\forall T_1 \subset N_{CC}\} \leq 1$.

6.5 Experimental Evaluation and Application

In this section, I apply the I-ISM to the 2-order additive Choquet Integral and evaluate the efficiency of I-ISM according to the results of alternative ranking by using the Choquet Integral. At first, I obtain the marginal utilities of the criteria of 10 alternatives using the sigmoid utility function proposed in Section 6.3. Then the desired utility value of each alternative are given based on the MACBETH [51] approach, which in this experiment is fixed as $y(x)_{x\in A} = (0.9126, 1, 0.9106, 0.5942, 0.6732, 0.2732, 0.566, 0.3504, 0.5381, 0.49)$, and the order of the service providers is $sp1 > sp2 > sp10 > sp5 > sp6 > sp3 > sp9 > sp4 > sp8 > sp7$. The DM can use the pair-wise comparison of the Analytic Network Process (ANP) [118] to get the importance order of the nine criteria: $av > stc > cost > re > dts > la > tcs > sc > rscs$.

I also give a set of constraint conditions in Table 6.3 to describe DM's intensions on the interactive index between two criteria, where δ_{Sh} is a threshold given by DMs. The value of δ_{Sh} will be set in the specific capacity identification. In addition, I use δ_ϕ represents the importance threshold for singleton criterion. For example, in Fig. 6.2, I have got the block partition of the nine criteria of Cloud services. I assume that the DMs give similar importance of the criteria in the same block, while the criteria in RankI are more important than

the criteria in RankII, and the RankII criteria are more important than RankII criteria. For example, $-\delta_\phi \leq \Phi(stc) - \Phi(sc) \leq \delta_\phi$; $-\delta_\phi \leq \Phi(av) - \Phi(re) \leq \delta_\phi$; $-\delta_\phi \leq \Phi(dts) - \Phi(la) \leq \delta_\phi$; and $\Phi(av) - \Phi(stc) > \delta_\phi$; $\Phi(dts) - \Phi(tcs) > \delta_\phi$. In this experiment, I fix $\delta_\Phi = 0.01$.

Our experiments are conducted on a 64-bit Windows System with Intel i5 3.2 GHz CPU and 4GB RAM. I use 'kappalab' package [79] for identifying capacities and evaluating the alternative ranking results.

In these experiments, I apply four capacity identification procedures to evaluate the quality of ranking of the 10 service providers, which are least squares (ls), linear programming (lp), minimum variance (mv), and minimum distance (md). Readers can refer to [79] for details of these four types of capacity identification procedure. I show the performance of the capacity calculation in terms of two measures:

- entropy (en): the maximum entropy procedure determines the most uncertain capacity: maximize $H_M(\mu) = -\Sigma_{i\in N}\Sigma_{S\subset N\setminus i}\gamma_s(n)[\mu(S \cup i) - \mu(S)]$. The maximum entropy procedure is corresponding to the minimum variance procedure: minimize $\overline{V}(\mu) = \frac{1}{n}\Sigma_{i\in N}\Sigma_{S\subset N\setminus i}\gamma_s(n)(\mu(S \cup i) - \mu(S) - \frac{1}{n})^2$. In the following experiments, I will show both the values of entropy and variance of a capacity to see the capacity quality.
- mean squared error (mean): measure the difference between the expected utilities of alternatives and the induced utilities calculated by the Choquet Integral.

Least Squares Capacity Identification. I evaluate the ranking of the ten Cloud service providers by investigating the changing trend of the *en*, *var*,

Table 6.3: Constraints on the Criteria Preference in terms of DM's Desire

I(ij)	re(1)	sc(2)	stc(3)	dts(4)	la(5)	tcs(6)	rscs(7)	cost(8)
av(0)	[-1, $-\delta_{Sh}$]	[-1, $-\delta_{Sh}$]	[-1,$-\delta_{Sh}$]	[δ_{Sh}, 1]	[δ_{Sh}, 1]	[-1, $-\delta_{Sh}$]	[δ_{Sh}, 1]	[δ_{Sh}, 1]
re(1)		[-1, $-\delta_{Sh}$]	[-1, $-\delta_{Sh}$]	[δ_{Sh}, 1]	[δ_{Sh}, 1]	[-1, $-\delta_{Sh}$]	[δ_{Sh}, 1]	[δ_{Sh}, 1]
sc(2)			[-1, $-\delta_{Sh}$]	[δ_{Sh}, 1]	[δ_{Sh}, 1]	[-1, $-\delta_{Sh}$]	[δ_{Sh}, 1]	[δ_{Sh}, 1]
stc(3)				[δ_{Sh}, 1]	[δ_{Sh}, 1]	[-1, $-\delta_{Sh}$]	[δ_{Sh}, 1]	[δ_{Sh}, 1]
dts(4)					[-1, $-\delta_{Sh}$]	[δ_{Sh}, 1]	[-1, $-\delta_{Sh}$]	[-1, $-\delta_{Sh}$]
la(5)						[δ_{Sh}, 1]	[-1, $-\delta_{Sh}$]	[-1, $-\delta_{Sh}$]
tcs(6)							[-1, $-\delta_{Sh}$]	[-1, $-\delta_{Sh}$]
rscs(7)								[-1, $-\delta_{Sh}$]

and *mean* in terms of different δ_{Sh} and δ_C. I test each combination of $\delta_{Sh} = \{0.1, 0.2, 0.3, 0.4, 0.5\}$ and $\delta_C = \{0.1, 0.2, 0.3, 0.4, 0.5\}$ individually by using the least squares 2-additive (represented as $ls2c$) and 3-additive ($ls3c$) capacity identification. Based on the marginal utilities of the ten service providers, the partial order of the 9 criteria and the values of δ_{Sh} and δ_C, the performance of *en*, *var* and *mean* is shown in Fig. 6.4.

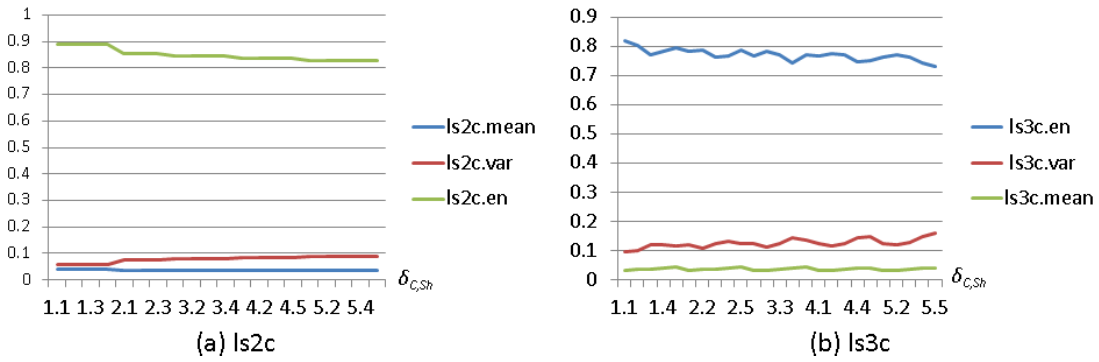


Figure 6.4: The performance of ls2c and ls3c in terms of *en*, *var*, and *mean* w.r.t different δ_{Sh} and δ_C. The x-axis marks '1.1' represent $\delta_{Sh} = 0.1$ and $\delta_C = 0.1$.

From Fig. 6.4, I can see that both the entropy and the variance can be kept in a high performance ($en > 0.7$ and $var < 0.2$) in terms of different thresholds of δ_{Sh} and δ_C by using both ls2c and ls3c capacity identification procedures. The mean squared errors is staying in a very low level ($mean < 0.1$) for both of ls2c and ls3c. The ranking performance results of ls2c and ls3c indicate the efficiency of using the proposed I-ISM model to construct the different types of relations among criteria.

Linear Programming, Minimum Variance, Minimum Distance Capacity Identification. The capacity identification procedures of the linear programming (abbr. *lp*), minimum variance (*mv*), and minimum distance (*md*) only depend on the partial orders of alternatives and criteria. I present the changing trend of the variance and entropy performance in terms of different threshold values δ_C based on 8 capacity identification procedures: 2-additive Linear programming (*lp*2), 2-, 3- and 5- additive minimum variance (*mv*2 and *mv*3), 2-, 3-, and 5-additive minimum distance based on the average quadratic distance between the global scores of the bi-alternatives (*md*2*g*, *md*3*g*, and *md*5*g*), and

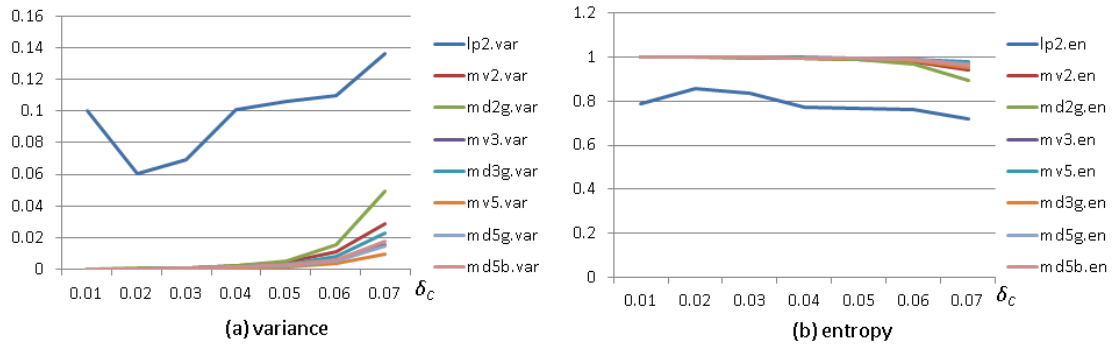


Figure 6.5: The performance of lp, mv and md in terms of *en* and *var* w.r.t different δ_C.

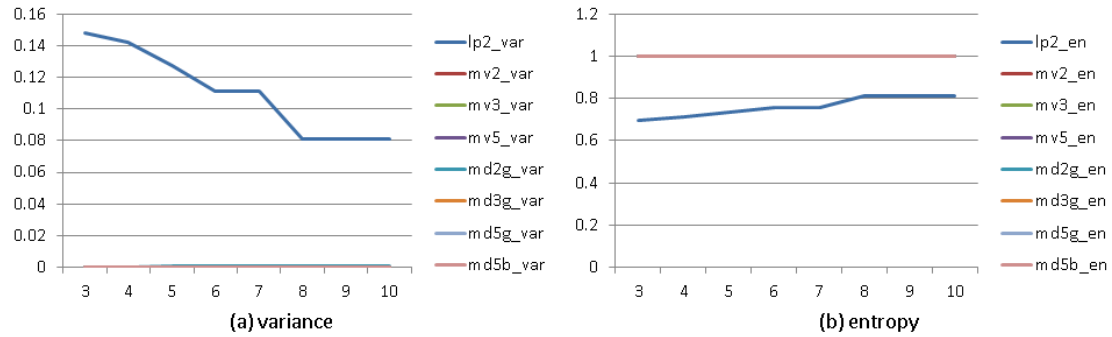


Figure 6.6: The performance of lp, mv and md in terms of *en* and *var* w.r.t different number of alternatives.

5-additive minimum distance with the minimized average distance between bi-alternatives (*md5b*). The performance is shown in Fig. 6.5. The evaluation results show that the performance of the three types of capacity identification procedures based on different $k-additive$ and distance settings is very high ($var < 0.14$ and $en > 0.7$) w.r.t a series of alternative threshold values δ_C.

I again compare the performance of the 8 capacity identification procedures in terms of *en* and *var* w.r.t a series of alternative numbers $3, 4, \cdots, 10$. Based on the performance comparison result in Fig. 6.5, when $\delta_C = 0.02$, the variance

reaches the lowest point and the entropy reaches the highest point with any appropriate $delta_{Sh}$. Therefore, I set $\delta_C = 0.02$ and $delta_{Sh} = 0.01$ in this comparison process, and the performance result is shown in Fig. 6.6. We can see that the performance of all the 8 capacity identification procedures is high ($var < 0.16$ and $en > 0.7$) w.r.t different alternative numbers.

6.6 Summary

In this chapter, I introduced a MCDM approach of Cloud service selection to help Cloud users make informed decisions on choosing most appropriate Cloud services. One distinctive feature of the proposed approach is that it measures the influence of the criteria relations on the decision making performance. This approach comprises two parts: an Interactive Interpretive Structural Modelling (I-ISM) approach to help decision makers establish consistent criteria relations interactively; and a 2-order fuzzy integral procedure to aggregate the utilities of singleton criterion performance and the performance of criteria coalitions. Our Cloud service selection approach improves the existing Cloud service selection techniques by taking into account the interactive utilities of a set of decision criteria. Experimental results show the efficiency of the proposed I-ISM for establishing criteria relations.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

Making healthcare decisions is challenging, which is influenced by a variety of factors: the lack of medical knowledge, the subjective mistakes of the healthcare givers, the false and incomplete information, and the misunderstanding and misinterpretation of the knowledge. The development of the medical sensor and the data mining technologies improve the accuracy of the medical information and the correctness of the medical decision making.

In this thesis, I focused on analysing large volume of medical sensor data streams collected from Sensor-cloud, where the following four problems were discussed:

- Segment data streams. Medical data streams are very long (consistently) and have medical-related features (e.g., pseudo-periodic). In certain situations, it is necessary to segment the long data stream to short subsequences and to analyse the data based on these sub-sequences. However, it is difficult to get high performance (e.g., accuracy and computational complexity) by applying the segmentation techniques against general areas to medical datasets.
- Detect abnormal subsequence in a data stream. Anomaly detection in medical data streams can assist doctors or patients in diagnosing diseases or analysing physical abnormal signals. Most existing work detects data

stream abnormal sub-sequences in an approximate way by first converting the continuous data stream to a discrete data stream. It is normally intractable to exactly discover the discords based on the original continuous data streams.

- Identify variable-length motifs. Finding motifs (i.e., repeated patterns) is still an open problem in continuous stream mining, though it has been studied for decades in the area of mining discrete data streams. Particularly, most researchers assume the repeated patterns having similar lengths in one data stream, which is incompatible with the real-word cases that have multiple types of motifs with different lengths. Especially, the occurrences of the same motif may have similar shapes yet variable lengths. So I discuss in this thesis the problem of discovering variable-length motifs in medical sensor data streams, where the discovered motifs can be used to recognize the repeated occurred behaviours or physical conditions.
- Select qualified cloud services for sensor-cloud construction. When making a purchasing decision for cloud services, healthcare IT managers should evaluate cloud service providers in terms of user specific requirements, to proactively resolve the precursors of cost leakages or service failures. It is valuable to explore the problem of cloud service selection to achieve a best trade-off between the spending and quality of using cloud services for building healthcare sensor-clouds.

I proposed a medical data stream mining framework to analyse the medical data streams (e.g., ECG and EEG), which consists of the following components:

- Supervised classification framework for anomaly detection. I proposed a supervised classification framework for detecting anomalies in uncertain

pseudo periodic time series, which comprises four components: an uncertainty identification and correction component (UICC), a time series compression component (TSCC), a period segmentation and summarization component (PSSC), and a classification and anomaly detection component (CADC). First, UICC processes a time series to remove uncertainties from the time series. Then TSCC compresses the processed raw time series to an approximate time series. Afterwards the PSSC identifies the periodic patterns of the time series and extracts the most important features of each period, and finally the CADC detects anomalies based on the selected features.

- Limited-length suffix array based motif discovery. I proposed an unsupervised Limited-length Suffix Array based Motif Discovery algorithm (LiSAM) for continuous time series, which is time and space efficient, and supports approximately discovering motifs in different lengths. I first converted the continuous time series to the discrete time series by using the Symbolic Aggregate approXimation procedure (SAX) [101], and then identify the different-length motifs based on the discrete time series. The illustration of discrete motif discovery is on the basis of an exact substring matching procedure, however, I can easily embed the existing approximate substring matching methods, such as, in LiSAM. That is, I use the exact subsequence grouping of discrete time series to discover the approximate patterns of continuous time series. I can also calculate the exact similarities between the instances of a continuous motif after such an approximate grouping.
- I proposed a **F**uzzy **U**ser-oriented **Cloud SeR**vice Selection System (Cloud-FuSeR) that is capable of dealing with fuzzy information and

rating cloud services by considering three aspects: (1) the similarities between user-required functions and the service functions provided by cloud providers, (2) the performance of the non-functional properties, and (3) the user preference on different properties.

- MCDM framework considering criteria interdependence. I proposed a MCDM framework that helps DMs to build the criteria relations and measures the performance of cloud service alternatives.

7.2 Future Work

The medical data stream mining work introduced in this thesis is focused on single-variable and non-evolving medical time series. In most cases, however, the medical decision making is based on the simultaneous analysis of multiple data streams, and the information in the data streams are evolving over time. Therefore, in the next stage, I am going to extend the current work from the following aspects:

- I will develop anomaly detection and motif discovery methods for multivariate medical time series, considering the correlations among the time series.
- I will consider the influence of the concept evolution on the medical decisions based on the data stream mining.
- I will develop a ECG-based healthcare application that consistently monitors the patients, and online detect and predict the diseases based on the ECG data streams.

- I will extend the ECG-based healthcare application to various healthcare areas, like EEG data stream mining, body temperature monitoring, and speaking voice analysis.

BIBLIOGRAPHY

[1] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1):53–86, 2004.

[2] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1):53 – 86, 2004. The 9th International Symposium on String Processing and Information Retrieval.

[3] P.M. Agante and J.P.M. de Sa. Ecg noise filtering using wavelets with soft-thresholding methods. In *Computers in Cardiology, 1999*, pages 535–538, 1999.

[4] Charu C. Aggarwal. On high dimensional projected clustering of uncertain data streams. In *IEEE 25th International Conference on Data Engineering*, ICDE'09, pages 1152–1154, Shanghai, China, March 2009. IEEE.

[5] Charu C. Aggarwal and Philip S. Yu. A framework for clustering uncertain data streams. In *IEEE 24th International Conference on Data Engineering*, ICDE'08, pages 150–159, Cancun, Mexico, April 2008. IEEE.

[6] Seyed-Ahmad Ahmadi, Nicolas Padoy, Kateryna Rybachuk, Hubertus Feussner, SM Heinin, and Nassir Navab. Motif discovery in or sensor data with application to surgical workflow analysis and activity detection. In *M2CAI workshop, MICCAI, London*. Citeseer, 2009.

[7] Ian F. Akyildiz, Dario Pompili, and Tommaso Melodia. Underwater acoustic sensor networks: research challenges. *Ad Hoc Netw.*, 3(3):257–279, 2005.

[8] Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. A survey on sensor networks. *Communications magazine, IEEE*, 40(8):102–114, 2002.

[9] Atif Alamri, Wasai Shadab Ansari, Mohammad Mehedi Hassan, M Shamim Hossain, Abdulhameed Alelaiwi, and M Anwar Hossain. A survey on sensor-cloud: architecture, applications, and approaches. *International Journal of Distributed Sensor Networks*, 2013, 2013.

[10] Hande Alemdar and Cem Ersoy. Wireless sensor networks for healthcare: A survey. *Computer Networks*, 54(15):2688–2710, 2010.

[11] Abdullah Alfazi, TalalH. Noor, QuanZ. Sheng, and Yong Xu. Towards ontology-enhanced cloud services discovery. In Xudong Luo, JeffreyXu Yu, and Zhi Li, editors, *Advanced Data Mining and Applications*, volume 8933 of *Lecture Notes in Computer Science*, pages 616–629. Springer International Publishing, 2014.

[12] Lv an Tang, Bin Cui, Hongyan Li, Gaoshan Miao, Dongqing Yang, and Xinbiao Zhou. Effective variation management for pseudo periodical streams. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD'07, pages 257–268, New York, NY, USA, 2007. ACM.

[13] Silvia Angilella, Salvatore Greco, Fabio Lamantia, and Benedetto Matarazzo. Assessing non-additive utility for multicriteria decision aid. *European Journal of Operational Research*, 158(3):734–744, 2004.

[14] Pablo Aragons-Beltrn, Fidel Chaparro-Gonzlez, Juan-Pascual Pastor-Ferrando, and Andrea Pla-Rubio. An {AHP} (analytic hierarchy process)/anp (analytic network process)-based multi-criteria decision approach for the selection of solar-thermal power plant investment projects. *Energy*, 66:222–238, 2014.

[15] Arvind Arasu, Shivnath Babu, and Jennifer Widom. The cql continuous query language: semantic foundations and query execution. Technical Report 2003-67, Stanford InfoLab, 2003.

[16] David Arthur and Sergei Vassilvitskii. K-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[17] Johannes Aßfalg, Hans-Peter Kriegel, Peer Kröger, and Matthias Renz. Probabilistic similarity search for uncertain time series. In Marianne Winslett, editor, *Scientific and Statistical Database Management*, volume 5566 of *Lecture Notes in Computer Science*, pages 435–443. Springer Berlin Heidelberg, New Orleans, LA, USA, 2009.

[18] Zeki Ayağ and Rifat Gürcan Özdemir. Evaluating machine tool alternatives through modified TOPSIS and alpha-cut based fuzzy ANP. *Int. J. Prod. Econ.*, 140(2):630–636, 2012.

[19] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval*, volume 463. Addison-Wesley, Boston, US, 1999.

[20] Timothy L. Bailey, James Johnson, Charles E. Grant, and William S. Noble. The meme suite. *Nucleic Acids Research*, 2015.

[21] Shibdas Bandyopadhyay, Sartaj Sahni, and Sanguthevar Rajasekaran. Pms6: A fast algorithm for motif discovery. *International Journal of Bioinformatics Research and Applications 2*, 10(4-5):369–383, 2014.

[22] Eugen Berlin and Kristof Van Laerhoven. Detecting leisure activities with dense motif discovery. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 250–259, New York, NY, USA, 2012. ACM.

[23] Denis Bouyssou. *Evaluation and decision models: a critical perspective*, volume 32. Springer Science & Business Media, 2000.

[24] Marcello Braglia, Marco Frosolini, and Roberto Montanari. Fuzzy TOPSIS approach for failure mode, effects and criticality analysis. *Qual. Reliab. Eng. Int.*, 19(5):425–443, 2003.

[25] D. Brauckhoff, K. Salamatian, and M. May. A signal processing view on packet sampling and anomaly detection. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9, March 2010.

[26] S. Budalakoti, S. Budalakoti, A.N. Srivastava, M.E. Otey, and M.E. Otey. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 39(1):101–113, Jan 2009.

[27] M.C. Campi and G. Calafiore. Decision making in an uncertain environment: the scenario-based optimization approach. *Multiple Participant Decision Making*, pages 99–111, 2004.

[28] Christer Carlsson and Robert Fullr. Interdependence in fuzzy multiple objective programming. *Fuzzy Sets and Systems*, 65(1):19–29, 1994.

[29] Erkan Celik, Alev Taskin Gumus, and Melike Erdogan. A new extension of the electre method based upon interval type-2 fuzzy sets for green logistic service providers evaluation. *Evaluation*, 44(5):1–15, 2016.

[30] Soumen Chakrabarti, Sunita Sarawagi, and Byron Dom. Mining surprising patterns using temporal description length. In *VLDB*, volume 98, pages 606–617. Citeseer, 1998.

[31] S Chakraborty and P Chatterjee. Selection of materials using multi-criteria decision-making methods with minimum data. *Decis. Sci. Lett.*, 2(3):135–148, 2013.

[32] Ioannis Chamodrakas, I Leftheriotis, and Drakoulis Martakos. In-depth analysis and simulation study of an innovative fuzzy approach for ranking alternatives in multiple attribute decision making problems based on TOPSIS. *Appl. Soft Comput.*, 11(1):900–907, 2011.

[33] Ioannis Chamodrakas and Drakoulis Martakos. A utility-based fuzzy TOPSIS method for energy efficient network selection in heterogeneous wireless networks. *Appl. Soft Comput.*, 12(7):1929–1938, 2012.

[34] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection for discrete sequences: A survey. *Knowledge and Data Engineering, IEEE Transactions on*, 24(5):823–839, May 2012.

[35] Varun Chandola, Varun Mithal, and Vipin Kumar. Comparative evaluation of anomaly detection techniques for sequence data. *IEEE International Conference on Data Mining*, 0:743–748, 2008.

[36] Kuo-Ming Chao, Muhammad Younas, Chi-Chun Lo, and Tao-Hsin Tan. Fuzzy matchmaking for Web services. In *International Conference on Advanced Information Networking and Applications (AINA)*, volume 2, pages 721–726. IEEE Computer Society, 2005.

[37] Chen-Tung Chen. Extensions of the TOPSIS for group decision-making under fuzzy environment. *Fuzzy Set. Syst.*, 114(1):1–9, 2000.

[38] Jianjun Chen, David J. DeWitt, Feng Tian, and Yuan Wang. Niagaracq: a scalable continuous query system for internet databases. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, SIGMOD'00, pages 379–390, New York, NY, USA, 2000. ACM.

[39] Shi-Jay Chen and Shyi-Ming Chen. Fuzzy risk analysis based on similarity measures of generalized fuzzy numbers. *IEEE T. Fuzzy Syst.*, 11(1):45–56, 2003.

[40] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive, July 2015. www.cs.ucr.edu/~eamonn/time_series_data/.

[41] Tak chung Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164 – 181, 2011.

[42] Australian Communications and Media Authority. Sensing and monitoring - recent developments. Technical report, Australian Communications and Media Authority, 9 2011.

[43] Cloud Standards Customer Council. Impact of cloud computing on healthcare. Technical report, Cloud Standards Customer Council, 11 2012.

[44] CSIRO. Sensors and sensor networks 2010-2011 year in review, 2011.

[45] Robert Csutora and James J Buckley. Fuzzy hierarchical analysis: the Lambda-Max method. *Fuzzy Set. Syst.*, 120(2):181–195, 2001.

[46] Parisa D Hossein Zadeh and Marek Z Reformat. Assessment of semantic similarity of concepts defined in ontology. *Inform. Sci.*, 250:21–39, 2013.

[47] Michele Dallachiesa, Besmira Nushi, Katsiaryna Mirylenka, and Themis Palpanas. Uncertain time-series similarity: return to the basics. *Proc. VLDB Endow.*, 5(11):1662–1673, July 2012.

[48] Santanu Das, Bryan L Matthews, Ashok N Srivastava, and Nikunj C Oza. Multiple kernel learning for heterogeneous anomaly detection: algorithm and aviation safety case study. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 47–56. ACM, 2010.

[49] D. Dasgupta and N.S. Majumdar. Anomaly detection in multidimensional data using negative selection algorithm. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 2, pages 1039–1044, 2002.

[50] D. Dasgupta and F. Nino. A comparison of negative and positive selection algorithms in novel pattern detection. In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, volume 1, pages 125–130 vol.1, 2000.

[51] Diala Dhouib. An extension of {MACBETH} method for a fuzzy environment to analyze alternatives in reverse logistics for automobile tire wastes. *Omega*, 42(1):25–32, 2014.

[52] B. Di Martino, G. Cretella, and A. Esposito. Towards a unified owl ontology of cloud vendors' appliances and services at paas and saas level. In *International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*, pages 570–575, July 2014.

[53] Hai Dong, Farookh Khadeer Hussain, and Elizabeth Chang. A context-aware semantic similarity model for ontology environments. *Concurr. Comp. Pract. E.*, 23(5):505–524, 2011.

[54] David Leigh Donoho. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, 41(3):613–627, May 1995.

[55] David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica*, 10(2):112–122, 1973.

[56] M Durairaj and V Ranjani. Data mining applications in healthcare sector a study. *International Journal of Scientific and Technology Research*, 2(10), 2013.

[57] P. Ehrlich. *Real numbers, generalizations of the reals, and theories of continua*. Springer, 1994.

[58] E. Eskin, Wenke Lee, and S.J. Stolfo. Modeling system calls for intrusion detection with dynamic window sizes. In *DARPA Information Survivability Conference amp; Exposition II, 2001. DISCEX '01. Proceedings*, volume 1, pages 165–175 vol.1, 2001.

[59] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Comput. Surv.*, 45(1):12:1–12:34, December 2012.

[60] C. Esposito, M. Ficco, F. Palmieri, and A. Castiglione. Smart cloud storage service selection based on fuzzy logic, theory of evidence and game theory. *IEEE T. Comput.*, PP(99):1–1, 2015.

[61] J. Euzenat and P. Valtchev. An integrative proximity measure for ontology alignment. In *International Workshop Semantic Information Integration*, pages 33–38, 2003.

[62] Daren Fang, Xiaodong Liu, Imed Romdhani, Pooyan Jamshidi, and Claus Pahl. An agility-oriented and fuzziness-embedded semantic model for collaborative cloud service search, retrieval and recommendation. *Future Generation Computer Systems*, 56:11–26, 2016.

[63] A. Floratou, S. Tata, and J.M. Patel. Efficient and accurate discovery of patterns in sequence data sets. *Knowledge and Data Engineering, IEEE Transactions on*, 23(8):1154–1168, Aug 2011.

[64] German Florez, Zhen Liu, Susan M Bridges, Anthony Skjellum, and Rayford B Vaughn. Lightweight monitoring of mpi programs in real time. *Concurrency and Computation: Practice and Experience*, 17(13):1547–1578, 2005.

[65] Victor A. Folarin, Patrick J. Fitzsimmons, and William B. Kruyer. Holter monitor findings in asymptomatic male military aviators without structural heart disease. *Aviat. Space. Envir. MD*, 72(9):836–838, 2001.

[66] Ernest H Forman and Saul I Gass. The analytic hierarchy process-an exposition. *Oper. Res.*, 49(4):469–486, 2001.

[67] Jr. Forney, G.D. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, March 1973.

[68] Stephanie Forrest, Steven A. Hofmeyr, Anil Somayaji, and Thomas A. Longstaff. A sense of self for unix processes. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, SP '96, pages 120–, Washington, DC, USA, 1996. IEEE Computer Society.

[69] Komei Fukuda and Tomomi Matsui. Finding all minimum-cost perfect matchings in Bipartite graphs. *Netw.*, 22(5):461–468, 1992.

[70] Bo Gao, Hui-Ye Ma, and Yu-Hang Yang. Hmms (hidden markov models) based on anomaly intrusion detection method. In *Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on*, volume 1, pages 381–385 vol.1, 2002.

[71] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. A framework for ranking of cloud computing services. *Future Gener. Comp. Sy.*, 29(4):1012–1023, 2013.

[72] Saurabh Kumar Garg, Steven Versteeg, and Rajkumar Buyya. Smicloud:

a framework for comparing and ranking Cloud services. In *IEEE International Conference on Utility and Cloud Computing (UCC)*, pages 210–218. IEEE Computer Society, Dec 2011.

[73] Anup K. Ghosh, Aaron Schwartzbard, and Michael Schatz. Learning program behavior profiles for intrusion detection. In *Proceedings of the 1st Conference on Workshop on Intrusion Detection and Network Monitoring - Volume 1*, ID'99, pages 6–6, Berkeley, CA, USA, 1999. USENIX Association.

[74] Amol Ghoting, Srinivasan Parthasarathy, and MatthewEric Otey. Fast mining of distance-based outliers in high-dimensional datasets. *Data Mining and Knowledge Discovery*, 16(3):349–364, 2008.

[75] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[76] Fabio A. González and Dipankar Dasgupta. Anomaly detection using real-valued negative selection. *Genetic Programming and Evolvable Machines*, 4(4):383–403, December 2003.

[77] John C Gower. A general coefficient of similarity and some of its properties. *Biometrics*, pages 857–871, 1971.

[78] Michel Grabisch. k-order additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems*, 92(2):167–189, 1997. Fuzzy Measures and Integrals.

[79] Michel Grabisch, Ivan Kojadinovic, and Patrick Meyer. A review of methods for capacity identification in choquet integral based multi-attribute utility theory: Applications of the kappalab r package. *European Journal of Operational Research*, 186(2):766–785, 2008.

[80] Charles E. Grant, Timothy L. Bailey, and William Stafford Noble. Fimo: scanning for occurrences of a given motif. *Bioinformatics*, 27(7):1017–1018, 2011.

[81] Aslak Grinsted, John C. Moore, and Svetlana Jevrejeva. Application of the cross wavelet transform and wavelet coherence to geophysical time series. *Nonlinear Proc. Geoph.*, 11(5/6):561–566, 2004.

[82] Yu Gu, Andrew McCallum, and Don Towsley. Detecting anomalies in network traffic using maximum entropy estimation. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, IMC'05, pages 32–32, Berkeley, CA, USA, 2005. USENIX Association.

[83] Joachim Gudmundsson, Marc van Kreveld, and Bettina Speckmann. Efficient detection of patterns in 2d trajectories of moving points. *GeoInformatica*, 11(2):195–215, 2007.

[84] Şule Gündüz and M Tamer Özsu. A web page prediction model based on click-stream tree representation of user behavior. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 535–540. ACM, 2003.

[85] R. Gwadera, A. Gionis, and H. Mannila. Optimal segmentation using tree models. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 244–253, Dec 2006.

[86] Robert Gwadera, Mikhail J. Atallah, and Wojciech Szpankowski. Reliable detection of episodes in event sequences. *Knowledge and Information Systems*, 7(4):415–437, 2005.

[87] Hugo Haas and Allen Brown. Web services glossary. *W3C Working Group Note (11 February 2004)*, 9, 2004.

[88] John A. Hartigan and Manchek A. Wong. Algorithm as 136: a k-means clustering algorithm. *J. Roy. Stat. Soc. C.-APP*, 28(1):100–108, 1979.

[89] Jing He, Yanchun Zhang, and Guangyan Huang. Exceptional object analysis for finding rare environmental events from water quality datasets. *Neurocomputing*, 92(0):69–77, 2012. Data Mining Applications and Case Study.

[90] John Hershberger and Jack Snoeyink. An o(nlogn) implementation of the douglas-peucker algorithm for line simplification. In *Proceedings of the 10th Annual Symposium on Computational Geometry*, SCG'94, pages 383–384, New York, NY, USA, 1994. ACM.

[91] Steven A. Hofmeyr, Stephanie Forrest, and Anil Somayaji. Intrusion detection using sequences of system calls. *J. Comput. Secur.*, 6(3):151–180, August 1998.

[92] George Hripcsak and Adam S Rothschild. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American Medical Informatics Association*, 12(3):296–298, 2005.

[93] Ting-Ya Hsieh, Shih-Tong Lu, and Gwo-Hshiung Tzeng. Fuzzy MCDM approach for planning and design tenders selection in public office buildings. *Int. J. Proj. Manage.*, 22(7):573–584, 2004.

[94] Guangyan Huang, Yanchun Zhang, Jie Cao, Michael Steyn, and Kersi Taraporewalla. Online mining abnormal period patterns from multiple medical sensor data streams. *World Wide Web*, 17(4):569–587, 2014.

[95] Trinh N.D. Huynh, Wing-Kai Hon, Tak-Wah Lam, and Wing-Kin Sung. Approximate string matching using compressed suffix arrays. *Theoretical Computer Science*, 352(13):240 – 249, 2006.

[96] Victoria Ivey. Five tips to keep your data secure on the Cloud, 2013.

[97] FR Janes. Interpretive structural modelling: a methodology for structuring complex issues. *Transactions of the Institute of Measurement and Control*, 10(3):145–154, 1988.

[98] Kyle L. Jensen, Mark P. Styczynski, Isidore Rigoutsos, and Gregory N. Stephanopoulos. A generic motif discovery algorithm for sequential data. *Bioinformatics*, 22(1):21–28, 2006.

[99] Ruoyi Jiang, Hongliang Fei, and Jun Huan. Anomaly localization for network data streams with graph joint sparse pca. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD'11, pages 886–894, New York, NY, USA, 2011. ACM.

[100] Foued Jrad, Jie Tao, Ivona Brandic, and Achim Streit. Sla enactment for large-scale healthcare workflows on multi-cloud. *Future Gener. Comp. Sy.*, 4344:135 – 148, 2015.

[101] E. Keogh, J. Lin, and A. Fu. Hot sax: efficiently finding the most unusual time series subsequence. In *Data Mining, Fifth IEEE International Conference on*, pages 8 pp.–, Nov 2005.

[102] E. Keogh, J. Lin, Ada Waichee Fu, and H. Van Herle. Finding unusual medical time-series subsequences: Algorithms and applications. *IEEE*

Transactions on Information Technology in Biomedicine, 10(3):429–439, July 2006.

[103] Eamonn Keogh, Jessica Lin, and Ada Fu. Hot sax: efficiently finding the most unusual time series subsequence. In *The 5th IEEE International Conference on Data Mining*, ICDM'05, pages 226–233, Houston, Texas, USA, November 2005. IEEE.

[104] Eamonn Keogh, Jessica Lin, Sang-Hee Lee, and HelgaVan Herle. Finding the most unusual time series subsequence: algorithms and applications. *Knowledge and Information Systems*, 11(1):1–27, 2007.

[105] Eamonn Keogh, Jessica Lin, and Wagner Truppel. Clustering of time series subsequences is meaningless: Implications for previous and future research. In *Proceedings of the Third IEEE International Conference on Data Mining*, ICDM '03, pages 115–, Washington, DC, USA, 2003. IEEE Computer Society.

[106] Eamonn Keogh, Stefano Lonardi, and Bill 'Yuan-chi' Chiu. Finding surprising patterns in a time series database in linear time and space. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 550–556, New York, NY, USA, 2002. ACM.

[107] Eamonn Keogh, Stefano Lonardi, and Chotirat Ann Ratanamahatana. Towards parameter-free data mining. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 206–215, New York, NY, USA, 2004. ACM.

[108] Eamonn J. Keogh, Selina Chu, David Hart, and Michael Pazzani. Segmenting time series: a survey and novel approach. In Mark Last, Abraham Kandel, and Horst Bunke, editors, *Data Mining In Time Series Databases*, volume 57 of *Series in Machine Perception and Artificial Intelligence*, chapter 1, pages 1–22. World Scientific Publishing Company, 2004.

[109] Ben Kepes. Understanding the cloud computing stack: Saas, paas, iaas, October 2013.

[110] Shehzad Khalid. Activity classification and anomaly detection using m-mediods based modelling of motion patterns. *Pattern Recognition*, 43(10):3636 – 3647, 2010.

[111] Philipp Koehn and Jean Senellart. *Fast Approximate String Matching with Suffix Arrays and A* Parsing*. Association for Machine Translation in the Americas, AMTA, 2010.

[112] L.P.D. Kumar, S.S. Grace, A. Krishnan, V.M. Manikandan, R. Chinraj, and M.R. Sumalatha. Data filtering in wireless sensor networks using neural networks for storage in cloud. In *Recent Trends In Information Technology (ICRTIT), 2012 International Conference on*, pages 202–205, April 2012.

[113] C.K. Kwong and H. Bai. A fuzzy AHP approach to the determination of importance weights of customer requirements in quality function deployment. *J. Intell. Manuf.*, 13(5):367–377, 2002.

[114] Thomas L Saaty. The analytic network process. *Iranian Journal of Operations Research*, 1(1):1–27, 2008.

[115] Christophe Labreuche and Michel Grabisch. The choquet integral for the aggregation of interval scales in multicriteria decision making. *Fuzzy Sets and Systems*, 137(1):11–26, 2003. Preference Modelling and Applications.

[116] Terran Lane. Hidden markov models for humancomputer interface modeling. In *In Proceedings of IJCAI-99 Workshop on Learning About Users*, pages 35–44, 1999.

[117] Sun Le, Hai Dong, Farookh Khadeer Hussain, Omar Khadeer Hussain, Jiangang Ma, and Yanchun Zhang. Multicriteria decision making with fuzziness and criteria interdependence in Cloud service selection. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1929–1936. IEEE Computer Society, 2014.

[118] Sun Le, Hai Dong, F.K. Hussain, O.K. Hussain, Jiangang Ma, and Yanchun Zhang. Multicriteria decision making with fuzziness and criteria interdependence in cloud service selection. In *2014 IEEE International Conference on Fuzzy Systems*, pages 1929–1936, July 2014.

[119] Amy HI Lee, Wen-Chin Chen, and Ching-Jan Chang. A fuzzy AHP and BSC approach for evaluating performance of IT department in the manufacturing industry in Taiwan. *Expert Syst. Appl.*, 34(1):96–107, 2008.

[120] Chang-Shing Lee, Zhi-Wei Jian, and Lin-Kai Huang. A fuzzy ontology and its application to news summarization. *IEEE Trans. Syst. Man. Cybern. B Cybern.*, 35(5):859–880, 2005.

[121] Chang-Shing Lee, Mei-Hui Wang, and Hani Hagras. A type-2 fuzzy ontology and its application to personal diabetic-diet recommendation. *IEEE T. Fuzzy Syst.*, 18(2):374–395, 2010.

[122] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD'07, pages 593–604, New York, NY, USA, 2007. ACM.

[123] Li-Wei Lee and Shyi-Ming Chen. Fuzzy risk analysis based on fuzzy numbers with different shapes and different deviations. *Expert Syst. Appl.*, 34(4):2763–2771, 2008.

[124] Sangwon Lee and Kwang-Kyu Seo. A hybrid multi-criteria decision-making model for a cloud service selection problem using bsc, fuzzy delphi method and fuzzy ahp. *Wireless Personal Communications*, pages 1–19, 2015.

[125] Wenke Lee, Salvatore J Stolfo, and Philip K Chan. Learning patterns from unix process execution traces for intrusion detection. In *AAAI Workshop on AI Approaches to Fraud Detection and Risk Management*, pages 50–56, 1997.

[126] Daniel Lemire. A better alternative to piecewise linear time series segmentation. In *Proceedings of the 7th SIAM International Conference on Data Mining*, SDM'07, pages 545–550, 2007.

[127] Carson Kai-Sang Leung and Boyu Hao. Mining of frequent item-sets from streams of uncertain data. In *IEEE 25th International Conference on Data Engineering*, ICDE'09, pages 1663–1670, Shanghai, China, March 2009. IEEE.

[128] Matthew N. Levy and Achilles J. Pappano. *Cardiovascular physiology*. Mosby Elsevier, 2007.

[129] Chunlin Li. Hybrid cloud service selection strategy: Model and application of campus. *Computer Applications in Engineering Education*, 2015.

[130] Shaochong Li and Hao-peng Chen. A context-aware framework for saas service dynamic discovery in clouds. In Xian-he Sun, Wenyu Qu, Ivan Stojmenovic, Wanlei Zhou, Zhiyang Li, Hua Guo, Geyong Min, Tingting Yang, Yulei Wu, and Lei Liu, editors, *Algorithms and Architectures for Parallel Processing*, volume 8631 of *Lecture Notes in Computer Science*, pages 671–684. Springer International Publishing, 2014.

[131] Xiaolei Li, Jiawei Han, Sangkyum Kim, and Hector Gonzalez. ROAM: Rule and Motif-Based Anomaly Detection in Massive Moving Object Data Sets. In *SIAM International Conference on Data Mining*, 2007.

[132] Yuan Li, Jessica Lin, and Tim Oates. Visualizing variable-length time series motifs. In *SDM*, pages 895–906. SIAM, 2012.

[133] Zhinan Li, Wenyao Xu, Anpeng Huang, and M. Sarrafzadeh. Dimensionality reduction for anomaly detection in electrocardiography: A manifold approach. In *Wearable and Implantable Body Sensor Networks (BSN), 2012 Ninth International Conference on*, pages 161–165, May 2012.

[134] J. Lin, E. Keogh, A. Fu, and H. Van Herle. Approximations to magic: finding unusual medical time series. In *Computer-Based Medical Systems, 2005. Proceedings. 18th IEEE Symposium on*, pages 329–334, June 2005.

[135] Jessica Lin, Eamonn Keogh, and Stefano Lonardi. Visualizing and discovering non-trivial patterns in large time series databases. *Information Visualization*, 4(2):61–82, July 2005.

[136] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, DMKD '03, pages 2–11, New York, NY, USA, 2003. ACM.

[137] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144, 2007.

[138] Bai ling Zhang, Yanchun Zhang, and Rezaul K. Begg. Gait classification in children with cerebral palsy by bayesian approach. *Pattern Recogn.*, 42(4):581–586, 2009. Pattern Recognition in Computational Life Sciences.

[139] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data*, 6(1):3:1–3:39, March 2012.

[140] Lei Liu, Zhijian Wang, and Jianhong Chen. Fuzzy ontology model and its application in semantic web service description and discovery. In *International Conference on Innovative Computing Information and Control (ICICIC)*, pages 551–551. IEEE Computer Society, 2008.

[141] Min Liu, Mingrui Wang, Weiming Shen, Nan Luo, and Junwei Yan. A quality of service (qos)-aware execution plan selection approach for a service composition process. *Future Gener. Comp. Sy.*, 28(7):1080 – 1089, 2012. Special section: Quality of Service in Grid and Cloud Computing.

[142] Wei Liu, Gang Hua, and J.R. Smith. Unsupervised one-class learning for automatic outlier removal. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3826–3833, June 2014.

[143] Xiaoyan Liu, Xindong Wu, Huaiqing Wang, Rui Zhang, J. Bailey, and K. Ramamohanarao. Mining distribution change in stock order streams. In *IEEE 26th International Conference on Data Engineering*, VLDB'04, pages 105–108, March 2010.

[144] Jessica Lin Eamonn Keogh Stefano Lonardi and Pranav Patel. Finding motifs in time series. In *Proc. of the 2nd Workshop on Temporal Data Mining*, pages 53–68, 2002.

[145] Michael A. Lones and Andy M. Tyrrell. The evolutionary computation approach to motif discovery in biological sequences. In *Proceedings of the 7th Annual Workshop on Genetic and Evolutionary Computation*, GECCO '05, pages 1–11, New York, NY, USA, 2005. ACM.

[146] Mrinmoy Majumder. Multi criteria decision making. In *Impact of Urbanization on Water Shortage in Face of Climatic Aberrations*, SpringerBriefs in Water Science and Technology, pages 35–47. Springer Singapore, 2015.

[147] Anukul Mandal and SG Deshmukh. Vendor selection using interpretive structural modelling (ism). *International Journal of Operations & Production Management*, 14(6):52–59, 1994.

[148] Melanie Manning and Louanne Hudgins. Array-based technology and recommendations for utilization in medical genetics practice for detection of chromosomal abnormalities. *Genet. Med.*, 12(11):742–745, 2010.

[149] Carla Marceau. Characterizing the behavior of a program using multiple-length n-grams. In *Proceedings of the 2000 Workshop on New Security Paradigms*, NSPW '00, pages 101–110, New York, NY, USA, 2000. ACM.

[150] M.M. Masud, Jing Gao, L. Khan, Jiawei Han, and Bhavani Thuraisingham. Classification and novel class detection in concept-drifting data streams under time constraints. *IEEE Transactions on Knowledge and Data Engineering*, 23(6):859–874, June 2011.

[151] Amy McGovern, DerekH. Rosendahl, RodgerA. Brown, and KelvinK. Droegemeier. Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction. *Data Mining and Knowledge Discovery*, 22(1-2):232–258, 2011.

[152] C.C. Michael and A. Ghosh. Two state-based approaches to program-based anomaly detection. In *Computer Security Applications, 2000. ACSAC '00. 16th Annual Conference*, pages 21–30, Dec 2000.

[153] D. Minnen, T. Starner, I. Essa, and C. Isbell. Discovering characteristic actions from on-body sensor data. In *Wearable Computers, 2006 10th IEEE International Symposium on*, pages 11–18, Oct 2006.

[154] David Minnen, Charles L. Isbell, Irfan Essa, and Thad Starner. Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. In *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 1*, AAAI'07, pages 615–620. AAAI Press, 2007.

[155] A. Mueen. Enumeration of time series motifs of all lengths. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 547–556, Dec 2013.

[156] A. Mueen. Enumeration of time series motifs of all lengths. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 547–556, Dec 2013.

[157] Abdullah Mueen, Eamonn Keogh, Qiang Zhu, SydneyS. Cash, M.Brandon Westover, and Nima Bigdely-Shamlo. A disk-aware algorithm for time series motif discovery. *Data Mining and Knowledge Discovery*, 22(1-2):73–105, 2011.

[158] Abdullah Mueen, Eamonn J Keogh, Qiang Zhu, Sydney Cash, and M Brandon Westover. Exact discovery of time series motifs. In *SDM*, pages 473–484. SIAM, 2009.

[159] Toshiaki Murofushi and Michio Sugeno. An interpretation of fuzzy measures and the choquet integral as an integral with respect to a fuzzy measure. *Fuzzy Sets and Systems*, 29(2):201–227, 1989.

[160] Armando Muscariello, Guillaume Gravier, and Frdric Bimbot. Variability tolerant audio motif discovery. In Benoit Huet, Alan Smeaton, Ketan

Mayer-Patel, and Yannis Avrithis, editors, *Advances in Multimedia Modeling*, volume 5371 of *Lecture Notes in Computer Science*, pages 275–286. Springer Berlin Heidelberg, 2009.

[161] Andrew Y. Ng, Michael I. Jordan, Yair Weiss, et al. On spectral clustering: analysis and an algorithm. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, pages 849–856. MIT Press, Cambridge, USA, 2001.

[162] T.H. Noor, Q.Z. Sheng, A. Alfazi, A.H.H. Ngu, and J. Law. Csce: A crawler engine for cloud services discovery on the world wide web. In *IEEE International Conference on Web Services (ICWS)*, pages 443–450, June 2013.

[163] Francisco Javier Ordóñez and Daniel Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.

[164] Themis Palpanas, Michail Vlachos, Eamonn Keogh, and Dimitrios Gunopulos. Streaming time series summarization using user-defined amnesic functions. *IEEE Trans. Knowl. Data Eng.*, 20(7):992–1006, July 2008.

[165] DiptiD. Patil, JyotiG. Mudkanna, Dnyaneshwar Rokade, and VijayM. Wadhai. Concept adapting real-time data stream mining for health care applications. In David C. Wyld, Jan Zizka, and Dhinaharan Nagamalai, editors, *Advances in Computer Science, Engineering and Applications*, volume 166 of *Advances in Intelligent and Soft Computing*, pages 341–351. Springer Berlin Heidelberg, 2012.

[166] D. Pavlov. Sequence modeling with mixtures of conditional maximum entropy distributions. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 251–258, Nov 2003.

[167] Jonathan Pryshlakivsky and Cory Searcy. A heuristic model for establishing trade-offs in corporate sustainability performance measurement systems. *Journal of Business Ethics*, pages 1–20, 2015.

[168] Jianzhong Qi, Rui Zhang, Kotagiri Ramamohanarao, Hongzhi Wang, Zeyi Wen, and Dan Wu. Indexable online time series segmentation with error bound guarantee. *World Wide Web*, 18(2):359–401, 2015.

[169] Lie Qu, Yan Wang, M.A. Orgun, Ling Liu, and A. Bouguettaya. Context-aware cloud service selection based on comparison and aggregation of us-

er subjective assessment and objective performance assessment. In *IEEE International Conference on Web Services (ICWS)*, pages 81–88, June 2014.

[170] Lie Qu, Yan Wang, Mehmet Orgun, et al. Cloud service selection based on the aggregation of user feedback and quantitative performance assessment. In *IEEE International Conference on Services Computing (SCC)*, pages 152–159. IEEE, 2013.

[171] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, pages 427–438, New York, NY, USA, 2000. ACM.

[172] Ziaur Rehman, OmarKhadeer Hussain, and FarookhKhadeer Hussain. Parallel cloud service selection and ranking based on qos history. *Int. J. Parallel. Prog.*, 42(5):820–852, 2014.

[173] Philip Resnik. Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. *J. Artif. Intell. Res.*, 11:95–130, 1999.

[174] Douglas Reynolds. Gaussian mixture models. In StanZ. Li and Anil Jain, editors, *Encyclopedia of Biometrics*, pages 659–663. Springer, USA, 2009.

[175] C.J. Van Rijsbergen. *Information retrieval*. Butterworth-Heinemann, London, 1979.

[176] Paul L. Rosin. Assessing the behaviour of polygonal approximation algorithms. *Pattern Recogn.*, 36(2):505–518, 2003. Biometrics.

[177] Peter J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(0):53–65, 1987.

[178] Thomas L Saaty. How to make a decision: the analytic hierarchy process. *Eur. J. Oper. Res.*, 48(1):9–26, 1990.

[179] Thomas L Saaty. Decision making with the analytic hierarchy process. *International journal of services sciences*, 1(1):83–98, 2008.

[180] Majed Sahli, Essam Mansour, and Panos Kalnis. Parallel motif extraction from very long sequences. In *Proceedings of the 22Nd ACM International*

Conference on Conference on Information & Knowledge Management, CIKM '13, pages 549–558, New York, NY, USA, 2013. ACM.

[181] Smruti R. Sarangi and Karin Murthy. Dust: a generalized notion of similarity between uncertain time series. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD'10, pages 383–392, New York, NY, USA, 2010. ACM.

[182] Suchi Saria, Andrew Duchi, and Daphne Koller. Discovering deformable motifs in continuous time series data. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1465–1471. AAAI Press, 2011.

[183] Prasad Saripalli and Gopal Pingali. Madmac: multiple attribute decision methodology for adoption of Clouds. In *IEEE International Conference on Cloud Computing (CLOUD)*, pages 316–323. IEEE Computer Society, 2011.

[184] D. Sart, A. Mueen, W. Najjar, E. Keogh, and V. Niennattrakul. Accelerating dynamic time warping subsequence search with gpus and fpgas. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 1001–1006, Dec 2010.

[185] Galit Shmueli and Howard Burkom. Statistical challenges facing early outbreak detection in biosurveillance. *Technometrics*, 52(1):39–51, 2010.

[186] Huan-Jyh Shyur and Hsu-Shih Shih. A hybrid MCDM model for strategic vendor selection. *Math. Comput. Model.*, 44(7):749–761, 2006.

[187] Jane Siegel and Jeff Perdue. Cloud services measures for global use: the service measurement index (smi). In *Annual SRII Global Conference (SRII)*, pages 411–415. IEEE, 2012.

[188] Ikaro Silva and George Moody. An open-source toolbox for analysing and processing physionet databases in matlab and octave. *J. Open Res. Softw.*, 2(1), 2014.

[189] Fernando Silveira, Christophe Diot, Nina Taft, and Ramesh Govindan. Astute: Detecting a different class of traffic anomalies. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, pages 267–278, New York, NY, USA, 2010. ACM.

[190] Chirpreet Singh, Qun Shao, Yuqian Lu, Xun Xu, and Xinfeng Ye. Tool s-

election: A cloud-based approach. In James J. (Jong Hyuk) Park, Albert Zomaya, Hwa-Young Jeong, and Mohammad Obaidat, editors, *Frontier and Innovation in Future Computing and Communications*, volume 301 of *Lecture Notes in Electrical Engineering*, pages 237–245. Springer Netherlands, 2014.

[191] Nguyen Thanh Son and Duong Tuan Anh. Discovering approximate time series motif based on mp_c method with the support of skyline index. In *Knowledge and Systems Engineering (KSE), 2012 Fourth International Conference on*, pages 113–120, Aug 2012.

[192] Spideroak. Spideroak, 2014.

[193] Stratecast. Tips for choosing a cloud service provider. Technical Report MSU-CSE-00-2, Department of Computer Science, Michigan State University, march 2011.

[194] Louise T Su. The relevance of recall and precision in user evaluation. *J. Am. Med. Inf. Assoc.*, 45(3):207–217, 1994.

[195] Le Sun. Cloud data storage service ontology, 2013.

[196] Le Sun, Hai Dong, Farookh Khadeer Hussain, Omar Khadeer Hussain, and Elizabeth Chang. Cloud service selection: State-of-the-art and future research directions. *J. Netw. Comput. Appl.*, 45:134 – 150, 2014.

[197] Le Sun, Hai Dong, Farookh Khadeer Hussain, Omar Khadeer Hussain, and Elizabeth Chang. Cloud service selection: State-of-the-art and future research directions. *Journal of Network and Computer Applications*, 45:134–150, 2014.

[198] Pei Sun, Sanjay Chawla, and Bavani Arunasalam. Mining for outliers in sequential databases. In *SDM*, pages 94–105. SIAM, 2006.

[199] Shengtao Sun, Lizhe Wang, Rajiv Ranjan, and Aizhi Wu. Semantic analysis and retrieval of spatial data based on the uncertain ontology model in digital earth. *Int. J. Digit. Earth*, 8(1):3–16, 2015.

[200] Zeeshan Syed, Collin Stultz, Manolis Kellis, Piotr Indyk, and John Guttag. Motif discovery in physiological datasets: A methodology for inferring predictive elements. *ACM Trans. Knowl. Discov. Data*, 4(1):2:1–2:23, January 2010.

[201] M. Tajvidi, R. Ranjan, J. Kolodziej, and Lizhe Wang. Fuzzy cloud service selection framework. In *IEEE International Conference on Cloud Networking (CloudNet)*, pages 443–448, Oct 2014.

[202] Swee Chuan Tan, Kai Ming Ting, and Tony Fei Liu. Fast anomaly detection for streaming data. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1511–1516. AAAI Press, 2011.

[203] J.T. Tate. Fourier analysis in number fields and hecke's zeta-functions. In *Proceedings Instructional Conference*, 1965.

[204] Mahbod Tavallaee, Natalia Stakhanova, and Ali Akbar Ghorbani. Toward credible evaluation of anomaly-based intrusion-detection methods. *IEEE T. Syst. Man. Cy. C.*, 40(5):516–524, September 2010.

[205] Shengfeng Tian, Shaomin Mu, and Chuanhuan Yin. Sequence-similarity kernels for svms to detect anomalies in system calls. *Neurocomput.*, 70(4-6):859–866, January 2007.

[206] Fatma Tiryaki and Beyza Ahlatcioglu. Fuzzy portfolio selection using fuzzy analytic hierarchy process. *Inform. Sci.*, 179(1):53–69, 2009.

[207] Fatemeh Torfi, Reza Zanjirani Farahani, and Shabnam Rezapour. Fuzzy AHP to determine the relative weights of evaluation criteria and Fuzzy TOPSIS to rank the alternatives. *Appl. Soft Comput.*, 10(2):520–528, 2010.

[208] Vicen Torra and Yasuo Narukawa. The interpretation of fuzzy integrals and their application to fuzzy systems. *International Journal of Approximate Reasoning*, 41(1):43–58, 2006. Aggregation Operators and Decision Modeling.

[209] Liem Tran and Lucien Duckstein. Comparison of fuzzy numbers using a fuzzy distance measure. *Fuzzy Set. Syst.*, 130(3):331–341, 2002.

[210] Thanh T. Tran, Liping Peng, Yanlei Diao, Andrew Mcgregor, and Anna Liu. Claro: modelling and processing uncertain data streams. *VLDB J.*, 21(5):651–676, October 2012.

[211] Evangelos Triantaphyllou and Chi-Tun Lin. Development and evaluation of five fuzzy multiattribute decision-making methods. *Int. J. Approx. Reason.*, 14(4):281–310, 1996.

[212] TrueCrypt. TrueCrypt, 2014.

[213] Ping Wang. Qos-aware web services selection with intuitionistic fuzzy set under consumers vague perception. *Expert Syst. Appl.*, 36(3):4460–4466, 2009.

[214] Xiaogang Wang, Jian Cao, and Yang Xiang. Dynamic cloud service selection using an adaptive learning mechanism in multi-cloud computing. *Journal of Systems and Software*, 100:195–210, 2015.

[215] Ying-Ming Wang and Taha Elhag. Fuzzy TOPSIS method based on alpha level sets with an application to bridge risk assessment. *Expert Syst. Appl.*, 31(2):309–319, 2006.

[216] Zhenyuan Wang and George Klir. *Fuzzy measure theory*. Springer Science & Business Media, 2013.

[217] C. Warrender, S. Forrest, and B. Pearlmutter. Detecting intrusions using system calls: alternative data models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, pages 133–145, 1999.

[218] Li Wei, E. Keogh, and Xiaopeng Xi. Saxually explicit images: Finding unusual shapes. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 711–720, Dec 2006.

[219] Md Whaiduzzaman, Abdullah Gani, Nor Badrul Anuar, Muhammad Shiraz, Mohammad Nazmul Haque, and Israat Tanzeena Haque. Cloud service selection using multicriteria decision analysis. *The Scientific World Journal*, 2014, 2014.

[220] William Wilson, Phil Birkin, and Uwe Aickelin. The motif tracking algorithm. *Int. J. Auto. Comput.*, 5(1):32–44, 2008.

[221] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics, 1994.

[222] Zhenghua Xu, Rui Zhang, Ramamohanarao Kotagiri, and Udaya Parampalli. An adaptive algorithm for online time series segmentation with error bound guarantee. In *Proceedings of the 15th International Conference*

on Extending Database Technology, EDBT'12, pages 192–203, New York, NY, USA, 2012. ACM.

[223] Kenji Yamanishi and Yuko Maruyama. Dynamic syslog mining for network failure monitoring. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pages 499–508, New York, NY, USA, 2005. ACM.

[224] Jiong Yang and Wei Wang. Cluseq: efficient and effective sequence clustering. In *Data Engineering, 2003. Proceedings. 19th International Conference on*, pages 101–112, March 2003.

[225] Miin-Shen Yang, Wen-Liang Hung, and Shou-Jen Chang-Chien. On a similarity measure between LR-type fuzzy numbers and its application to database acquisition. *Int. J. Intell. Syst.*, 20(10):1001–1016, 2005.

[226] Dragomir Yankov, Eamonn Keogh, Jose Medina, Bill Chiu, and Victor Zordan. Detecting time series motifs under uniform scaling. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 844–853. ACM, 2007.

[227] Lexiang Ye and Eamonn Keogh. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data Mining and Knowledge Discovery*, 22(1-2):149–182, 2011.

[228] Nong Ye et al. A markov chain model of temporal behavior for anomaly detection. In *Proceedings of the 2000 IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop*, volume 166, page 169. West Point, NY, 2000.

[229] S. Yingchareonthawornchai, H. Sivaraks, T. Rakthanmanon, and C.A. Ratanamahatana. Efficient proper length time series motif discovery. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 1265–1270, Dec 2013.

[230] Mustafa Yurdakul and Yusuf Tansel Ic. Application of correlation test to criteria selection for multi-criteria decision making (MCDM) models. *Int. J. Adv. Manuf. Tech.*, 40(3-4):403–412, 2009.

[231] Madoka Yuriyama and Takayuki Kushida. Sensor-cloud infrastructure - physical sensor management with virtualized sensors on cloud computing. *2013 16th International Conference on Network-Based Information Systems*, 0:1–8, 2010.

[232] M. Zhang, r. Ranjan, M. Menzel, S. Nepal, P. Strazdins, W. Jie, and L. Wang. An infrastructure service recommendation system for cloud applications with real-time qos requirement constraints. *Systems Journal*, PP(99):1–11, 2015.

[233] Liyue Zhao, Xi Wang, G. Sukthankar, and R. Sukthankar. Motif discovery and feature selection for crf-based activity recognition. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3826–3829, Aug 2010.

[234] Yu Zheng, Xing Xie, and Wei-Ying Ma. Geolife: a collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.

[235] Zibin Zheng, Xinmiao Wu, Yilei Zhang, M.R. Lyu, and Jianmin Wang. Qos ranking prediction for cloud services. *IEEE Transactions on Parallel and Distributed Systems*, 24(6):1213–1222, June 2013.

[236] Yunyue Zhu and Dennis Shasha. Statstream: Statistical monitoring of thousands of data streams in real time. In *Proceedings of the 28th International Conference on Very Large Data Bases*, VLDB'02, pages 358–369. VLDB Endowment, 2002.