MINISTRY OF EDUCATION
AND TRAINING

VIETNAM ACADEMY OF
SIENCE AND TECHNOLOGY

GRADUATE UNIVERSITY OF SIENCE AND TECHNOLOGY
…….…..***…………

TRAN THI THUY TRINH

MINING FUZZY FREQUENT ITEMSETS BASED ON TREE STRUCTURE AND PARALLEL PROCESSING TECHNIQUE

Major: Information System
Major code: 9 48 01 04

SUMMARY OF COMPUTER DOCTORAL THESIS

Ha Noi – 2023

The thesis has been completed at: Graduate University of Science and Technology- Vietnam Academy of Science and Technology

Supervisor 1: **Assoc.Prof. Phd. Nguyen Long Giang**
Supervisor 2: **Phd. Truong Ngoc Chau**

Reviewer 1: …
Reviewer 2: …
Reviewer 3: ….

The thesis shall be defended in front of the Thesis Committee at Vietnam
Academy Of Science And Technology - Graduate University Of Science And Technology, at ….… hour……, date…… month….…year 2023

This thesis could be found at:

- The National Library of Vietnam
- The Library of Graduate University of Science and Technology

INTRODUCTION

1. The necessity of the research

Research associated with practical application is an activity that requires a lot of time and effort of scientists. Moreover, in the 4.0 technology era, applications not only support basic business features but also help people make relatively accurate predictions at the present time and in the future. The rapid growth of these intelligent systems increases the demand for practical applications resulting in the generation of large amounts of data daily. Traditional statistical tools and methods are based on application needs, but they are not capable of handling the huge amounts of data that are derived from these applications. The analysis of such data is a top priority task otherwise it will turn into a very complex and unfavorable system. To overcome this problem, data mining [1]–[3] is one of the approaches that benefits by assisting in data analysis and summarizing data into useful information. The concept of data mining is to generate previously unidentified information with great relevance from the database for decision making. Depending on the variety of knowledge, data mining methods can be divided into the following categories: association rules [4]–[8], classification [7], [9]–[11], clustering [12]–[14] and sequential samples [15], [16]. In particular, association rule mining is very important for data mining research [17]–[19]. In common business transactions, association rules are of the form A→B with the purpose of finding the relationship of items in the database. This helps businesses make decisions in planning business and marketing strategies. In the first stage of association rule mining, frequent itemsets are obtained from a given set of data. From the extracted frequent item sets, association rules are built in the second stage. The main stage of association rule mining is frequent item set mining because it takes a lot of effort to locate frequent itemsets in a data set. Most of the research in this field has focused on improving the efficiency of cluster mining in terms of time and memory.

Traditional association rule or frequent item set mining algorithms [20], [21] mostly only represent transaction data in binary value, that is, it deals with the occurrence of items. However, with a clear approach, it is difficult to mine frequent item sets for association rules in databases containing quantitative data. Due to its ease of use and similarity to human inference, fuzzy set theory [22], [23] is being used in intelligent systems more often [24]–[27]. Linguistic representation makes knowledge simpler for humans to understand, so it is widely used. Therefore, in order to mine fuzzy association rules from the quantitative database, the domains of the quantitative attribute will be converted into a fuzzy set represented in the linguistic variables by using the membership function [28], this approach can reduce computations. Several fuzzy mining algorithms have been widely studied and developed using fuzzy set theory to convert the quantitative value of items into linguistic terms based on the same mechanism as regular Apriori [29]. [30], [31] [32].

The author used byte-vector representing the tidlist, the compressed list used contributes to the performance increase. Previously, Janikow combined symbolic decision trees on rule-based systems for fuzzy control [34] using fuzzy representation. Watanabe and Fujioka [35], [36] have defined equivalence redundancy of fuzzy elements and related theorems for fuzzy association rule mining. The goal of the algorithm is to refine the time spent on rule mining and at the same time remove redundant rules in data mining applications. However, most fuzzy association rule mining methods apply Apriori [37] to generate candidates and check their support, thus requiring multiple rescans of the database, since so it causes slow and inefficient process in large database. Furthermore, with the fuzzy representation in the above algorithms, the fuzzy set of the quantitative attributes and their membership functions depends on the subjective opinion of the expert or the availability. This problem causes "sharp" boundaries between fuzzy intervals, so it is difficult to determine the extent of the membership function for elements near the interval boundary. This is the first gap identified in the research problem of the thesis.

Instead of using the conventional approach according to Apriori, Lin et al. implemented fuzzy frequent tree (FFP)-tree method [38], [39] to mine fuzzy frequent itemsets based on pattern growth mechanism. This approach applied both fuzzy set theory and FP tree structure (Frequent pattern) to build FFP tree (Fuzzy Frequent Pattern) which can be used for mining process. The transformed language variables and their membership degree are sorted in ascending order of each transaction, thus preserving the downward closure property for recursive construction of the condition tree and mining fuzzy frequent items. This approach can require a lot of computation time when the transaction size is very large. The tree compression algorithm (CFFP-Compact Fuzzy Frequent Pattern) [40] was then designed to reduce the size of the FFP tree. Thus, an array is attached to each node by preserving the fuzzy values for the currently processed language variable with any of its prefix itemsets in the path. Although the number of tree nodes of the CFFP tree is significantly reduced compared with the FFP tree algorithm, it is necessary to keep an additional array of each node to store the member values of the currently processed node with any language variables in the way. Therefore, it requires a large amount of memory to hold such information, which is not efficient in a sparse database. To overcome this limitation, the upper-bound fuzzy frequent pattern (UBFFPT) algorithm [41] was then designed to keep not only the dense tree structure, but also the ability to fuzzy frequent item sets from memory limit compared to FFP tree and CFFP tree algorithm. The UBFFPT tree algorithm can efficiently mine fuzzy frequent items keeping the same size of tree nodes as the CFFP tree algorithm but memory and computational usage can be greatly reduced. The above algorithms only use a single language term to represent the item being processed in the database, so the information detected may be incomplete. Many algorithms related to multiple fuzzy frequent set mining [42]–[44] have been proposed to help knowledge be mined more fully than traditional methods. Then, Hong et al. [42] created a tree-based structure based on the concept of FP and FFPT trees [38] while maintaining multiple 1-item fuzzy frequent itemsets with an MFFP tree designed for mining necessary information extraction. Not only is a single language variable maintained to represent an item, but every item's fuzzy value is greater than the minimum support threshold. In order to make wise decisions, a more complete set of facts is thus maintained. The similar concept is then implemented for UBMFFP trees [44] and CMFFP trees [43]. Effective techniques for decision-making can consequently be attained with more comprehensive information regarding various derived fuzzy frequent patterns. Nevertheless, because the mining of fuzzy frequent itemsets in these methods is done recursively from the tree structure, a lot of memory is needed to store the temporary trees. The thesis will address this as the second gap.

Frequent itemset mining from many fuzzy datasets is mentioned in the article [45]. In the article, the author merges multiple tables using a star schema to find fuzzy multi-level association rules in a relational database model, capable of handling many tables. The algorithm uses joins and entities to recognize frequent item sets. However, the results of the paper still have many limitations in calculating the support of item sets related to other connections containing fuzzy properties. Another method such as [46] uses the differential evolutionary algorithm (DE) to mine optimized statistically significant fuzzy association rules that have large numbers and significant measurable values with strict control over the risk of speculative rules. In addition, the pattern-based algorithm proposed in [47] aims to find fuzzy association rules from large quantitative data sets. Various studies have been carried out not only to improve the performance but also to improve the search speed of fuzzy association rules with hash tables, schemas or tree data structures [40], [41], 43], [44]. The FFI-Miner frequent fuzzy item set mining algorithm [48] was developed to mine the complete set of FFIs without generating candidates. Algorithms using efficient pruning strategy were also developed to reduce the search space, thus speeding up the mining process to directly detect frequent fuzzy item sets. Frequent patterns are sets of items found in a significant number of transactions. Along with the increase in data size, the data types are heterogeneous and the data variation is extremely dynamic. Therefore, extending efficient fuzzy mining algorithms to the era of big data is an important problem, mining by applying parallel processing techniques has become a possible way to overcome this problem. processing time problem. This is the third gap identified in the thesis.

In Vietnam, association rule mining has been researched by research groups at the Institute of Information Technology under the Vietnam Academy of Science and Technology such as the doctoral thesis of Nguyen Huy Duc [49] introducing the FSM algorithm as an algorithm. quickly mine all the high-stakes item sets in the transaction database and propose the AFSM (Advanced FSM) algorithm based on the steps of the FSM algorithm with a new method of more efficient pruning of the candidate item sets. The doctoral thesis of Nguyen Long Giang [50] presents data mining methods using rough set theory. Author Nguyen Cong Hao's article [51] presents a fuzzy association rule processing method based on Hedge algebra. The research group of Prof. Dr. Vo Dinh Bay and Prof. Dr. Le Hoai Bac proposed a method of mining common item sets in a clear database like [52]–[55], which can be considered as the foundation for the research in the thesis.

This thesis aims to address the three gaps identified above. The research to solve those problems is really necessary not only in terms of theoretical development but also in terms of practical application. That is the motivation for the author of the thesis to conduct a research on the topic "**Fuzzy frequent item set mining based on tree structure and parallel processing techniques**" to come up with new effective methods of item set mining and mining of fuzzy association rules based on fuzzy set theory.

2. Research scope and subjectives

a. Research objectives

The objective of the thesis is to propose solutions to mining fuzzy frequent itemsets in quantitative databases, to overcome the "sharp boundary" problem when partitioning fuzzy data for quantitative attributes.

Specifically, the thesis focuses on proposing solutions to:

- Identify fuzzy sets for each quantitative attribute in the database through clustering techniques.
- Reduced storage memory during fuzzy frequent item set mining
- Reduce processing time in mining fuzzy frequent itemsets in large databases.

b. Research objects

- Frequent itemset mining algorithms in transactional databases
- Algorithms of fuzzy frequent item set mining, fuzzy association rule mining in quantitative databases.

c. Research scope

- The thesis studies fuzzy association rules, fuzzy frequent itemsets in quantitative databases.

- Synthesize scientific publications related to fuzzy frequent item set mining methods.
- Compare experiments with existing algorithms

3. Research methods

The following research techniques were employed in the thesis:

- Synthesize and assess published results on fuzzy frequent item set mining techniques from various sources of information acquired. On this foundation, suggest new outcomes and assess new results by putting various algorithms to the test. Apply the results to address a real-world issue.
- Comparative method is used to analyze methodologies and algorithms that have been proposed to address relevant research problems. Thereby, generating ideas for novel algorithms to solve research problem.
- Experimental method is used to assess the accuracy and viability of the suggested algorithms, real data sets are used for testing.

4. The main contributions of the thesis

The main contributions of the thesis are to propose and solve the following problems:

- Propose a method to determine fuzzy sets for each quantitative attribute in the database through clustering techniques. More specifically, the thesis presents the EMC clustering technique. The goal of these algorithms is to divide data into meaningful clusters. These clusters are then used to classify each quantitative attribute as a fuzzy set and determine their membership functions. [CT2], [CT4].
- Propose a method to mine fuzzy frequent itemsets in quantitative databases using Node-list data structure. Fuzzy frequent itemset mining based on PP_code or POS_code helps to limit the required memory consumption. [CT1], [CT2], [CT5].
- Propose a parallel processing method to mine fuzzy frequent itemsets using the approach of cellular learning automata (CLA). According to CLA, space is represented as a lattice, with each element being a cell. Line by line, transaction data will be read and simultaneously transferred to cells, which are processed in parallel. Through the use of these autonomous data cells, mining of fuzzy frequent itemsets is performed. This process shortens the execution time of the algorithm. [CT3].

5. The main research contents of the thesis

The thesis consists of Introduction, 03 chapters and conclusion.

- Introduction: Presenting the necessity and motivation of the research topic; research objectives, objects and scope; Research Methods; the main contributions and structure of the thesis.
- Chapter 1: Theoretical foundations
- Chapter 2: Mining fuzzy frequent itemsets based on tree structure.
- Chapter 3: Mining fuzzy frequent itemsets using parallel processing

CHAPTER 1: THEORETICAL FOUNDATIONS

In this chapter, the author presents the basic concepts of association rules, quantitative association rules, fuzzy logic, fuzzy association rules and related studies on fuzzy association rules. From there, identify outstanding issues to be solved in Chapter 2.

1.1 Association rule

1.1.1 Basic concepts of association rules [55]

Definition 1.1: Transaction database

Assume $I = \{i_1, i_2, \dots, i_m\}$ is the set of items. $D = \{T_1, T_2, \dots, T_n\}$ is a set of transactions, called the transaction database, where each transaction *t* in D has the form (tid, X) where each transaction t has identifier tid itemset t-itemset, $t = (tid, t - itemset)$; X is called the itemset if $X \subseteq I$.

Definition 1.2: The support of an itemset

The support of an itemset X in the transaction database D denoted $sup\ (X)$ is the number of transactions containing the item set X, calculated by the following formula:

$$sup(X) = |t|\ X \subseteq t, t \in D| \tag{1.1}$$

In which the symbol |.| is the number of transactions.

Definition 1.3: Frequent itemset

An item set X contained in transaction database D is said to be frequent if its support ($sup(X)$) is greater than or equal to a given minimum support threshold (minsup) defined by the user. Therefore, support is considered as the frequency of simultaneous occurrence of items.

Definition 1.4: Association rules

An association rule is a proposition of the form X →Y, where X and Y are sets of items that satisfy the following conditions: $X \subseteq I$, $Y \subseteq I$ và $X \cap Y = \emptyset$. X is called antecedent while Y is called consequent, the rule means X implies Y.

Definition 1.5: The support of a rule

Given an association rule $r = X \rightarrow Y$, the support of rule r denoted as sup(r) is the ratio of the number of transactions T ⊆ D containing both itemset X and itemset Y to the total number of transactions in D is defined as:

$$sup(r) = \frac{|\{T \in D | T \supset X \cup Y\}|}{|D|} \tag{1.2}$$

Definition 1.6: The confidence of a rule

Given an association rule $r = X \rightarrow Y$, the confidence of the rule r denoted as conf(r) is the ratio of the number of transactions T ⊆ D containing both itemset X and itemset Y to the total number of transactions in D contains the itemset X, defined as follows:

$$conf(r) = \frac{|\{T \in D | T \supset X \cup Y\}|}{|\{T \in D | T \supset X\}|} = \frac{sup(X \cup Y)}{sup(X)} \tag{1.3}$$

Definition 1.7: Strong association rule

Given an association rule $r = X \rightarrow Y$, if the rule r satisfies both the minimum support (minsup) and minimum confidence thresholds (minconf), it is called a strong association rule, that is:

$$sup(r = X \rightarrow Y) = P(X \cup Y) \geq minsup$$

$$conf(r = X \rightarrow Y) = P(X \cup Y) = \frac{sup(X \cup Y)}{sup(X)} \geq minconf$$

Problem statement: The association rule problem is stated as follows [49]:

For a transactional database D, minimum support minsup, minimum confidence minconf. Find all association rules of the form $X \rightarrow Y$ that satisfy $sup(X \cup Y) \geq minsup$ và độ tin cậy $conf(X \rightarrow Y) = \frac{sup(X \cup Y)}{sup(X)} \geq minconf$.

1.1.2 Association rule in binary database

Binary association rules refer to classical rules in the shopping cart analysis problem. Where products may or may not be in the transaction, only Boolean values (represented by 1s and 0s) are produced. Therefore, every item in the transaction can be identified as a binary attribute with the domain {0,1}. The model is defined in [55] as follows:

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of binary attributes, called items. Let T be the transaction database. Each transaction t is represented as a binary vector with $t[k] = 1$ if transaction t contains the entry i_k and $t[k] = 0$ otherwise. Given X is a set of items contained in I, a transaction t satisfies X if every item in X, $i_k \in X, t[k] = 1$.

1.1.3 Association rule in quantitative database

According to this form of binary association rule, items are only interested in whether or not they appear in the transaction database, not how often or how often they occur. In fact, the database contains not only binary attributes, but also quantitative and categorical attributes that cannot be mined by classical techniques. The mining of rules in such data can be called a quantitative association rule problem [29].The quantitative association rule mining strategy is implemented by converting the attributes with quantitative values to binary values. In this method, each quantifiable/categorical value of the form $\langle attribute, value \rangle$ is mapped to a binary value. Then, binary association rule mining techniques are performed to find the rule. In quantitative association rule mining, the attributes can be both quantitative and categorical.

1.2 Overview of Fuzzy logic

1.2.1 Fuzzy set

Given a universe set U with elements denoted by u , $U = \{x\}$. A fuzzy set $\tilde{A}$ over U is a set characterized by a function $\mu_A(u)$ that associates each element $u \in U$ with a real number in the interval [0,1].

$$\tilde{A} = \{(u, \mu_A(u)) \mid u \in U\} \tag{1.4}$$

Where $\mu_A(u)$ is a mapping from U to [0,1] and is called the membership function of the fuzzy set $\tilde{A}$.

1.2.2 Membership function

The membership function $\mu_A(u)$ defined for the set A on the universe set U in the classical set concept has only two values of 1 if $u \in A$ or 0 if $u \notin A$. However, in the concept of a fuzzy set, the membership function value indicates the membership degree of the element u into the fuzzy set A. The specified interval of the function $\mu_A(u)$ is the interval [0, 1], where the value 0 indicates the degree of non-belonging, and the value 1 indicates the degree of complete belonging.

$$\mu(A) : U \rightarrow [0, 1] \tag{1.5}$$

The type of the fuzzy set depends on different types of membership functions. There are many different types of member functions proposed.

1.2.3 Linguistic variables

Linguistic variable is characterized by a quintuple $(X, T(X), U, R, M)$, where X is the variable name, T(X) is the set of linguistic values of the variable X, U is a universe of discourse, each linguistic value is considered a fuzzy variable on U combined with the base variable u, R is a syntactic rule that generates linguistic values of $T(X)$, M is the the semantic rule assigns each linguistic value in $T(X)$ to a fuzzy set on U.

For an example: Given X is a linguistic variable named AGE, the base variable u is taken according to the age of a person whose domain is defined as $U = [0,100]$. The set of linguistic values $T(AGE) = \{very\ young, young, middle, \text{old}, \text{very old}\}$.

1.2.4 Fuzzy Logic operations

Three basic fuzzy logic operations: complement, union and intersection are commonly used in fuzzy set theory, described below [22].

Complement: The complement operation of fuzzy set A is denoted ⌐A. The membership function of ⌐A can be defined as:

$$\mu_{\neg A}(x) = 1 - \mu_A(x), \quad \forall x \in X \tag{1.9}$$

Union: The union of two fuzzy sets A and B is denoted $A \cup B$. The membership function of $A \cup B$ for the normal operation can be defined as follows:

$$\mu_{A\cup B}(x) = max\{\mu_A(x), \mu_B(x)\}, \quad \forall x \in X \tag{1.10}$$

Intersection: The intersection operation of two fuzzy sets A and B is denoted $A \cap B$. The membership function of $A \cap B$ for the normal operation can be defined as follows:

$$\mu_{A\cap B}(x) = min\{\mu_A(x), \mu_B(x)\}, \quad \forall x \in X \tag{1.11}$$

1.3 Fuzzy association rules

1.3.1 Fuzzy transactional database

Let $I = \{I_1, I_2, \dots, I_m\}$ be the set of n attributes, i_u is the uth attribute in I. $D_Q = \{T_1, T_2, \dots, T_n\}$ is a set of transactions with each $T_v \in D_Q$ is a subset of I containing items that have a quantitative value and have a unique identifier TID. A transaction T is said to contain X if $X \subseteq T_q$, in which X is a set containing some items contained in I. Each attribute I_k can be associated with the represented fuzzy set of values $F_{ik} = \{f_{ik}^1, f_{ik}^2, \dots, f_{ik}^h\}$ where f_{ik}^j is the jth fuzzy value in F_{ik}. Using the related membership function to determine the fuzzy set for each attribute, the quantitative database D_Q is transformed into a database containing the fuzzy value D_f.

1.3.2 The support of fuzzy itemset

A fuzzy attribute set in fuzzy association rule is a pair $\langle X, A\rangle$ where A is the set of fuzzy sets corresponding to the attributes in X and $X \subseteq I$.

The support of the itemset $\langle X, A\rangle$ denoted by $fsup(\langle X, A\rangle)$ is determined by the following formula:

$$fsup(\langle X, A\rangle) = \sum_{t\in T} \mu_{x1}(t) \otimes \mu_{x2}(t) \otimes \dots \otimes \mu_{xp}(t) \tag{1.12}$$

where, $\mu_{xp}(t)$ is fuzzy value of attribute x_p in a transaction *t*.

$\otimes$ is T-norm operator. In fuzzy logic theory, it has the same role as the AND operation in classical logic. There are many ways to choose T-norm operation such as:

$$a \otimes b = min(a, b)$$

$$a \otimes b = ab$$

$$a \otimes b = max(0, a + b - 1)$$

$$a \otimes b = \begin{cases} a\ (if\ b = 1) \\ b\ (if\ a = 1) \\ 0\ (if\ a, b < 1) \end{cases}$$

Intersection: $a \otimes b = 1 - min\left[1, ((1-a)^w + (1-b)^w)^{\frac{1}{w}}\right]$ với $(w > 0)$

The minimization operation and algebraic product are the two most suitable operations because it is convenient for calculations and shows the close relationship between the attributes in the frequent sets.

When we choose the minimization operation for T-norm operator, the formula for calculating the support of the itemset $\langle X, A\rangle$ will be:

$$fsup(\langle X, A\rangle) = \sum_{t\in T} min\{\mu_{x1}(t),\ \mu_{x2}(t), \dots,\ \mu_{xp}(t)\} \tag{1.13}$$

When we choose algebraic product for T-norm operator, the formula for calculating the support of the itemset $\langle X, A\rangle$ will be:

$$sup(\langle X, A\rangle) = \sum_{t\in T} \prod_{x_p\in X} \{\ \mu_{xp}(t)\} \tag{1.14}$$

1.3.3 The fuzzy frequent itemset

Definition 1.8: (The fuzzy frequent itemset): [41]

An item set $\langle X, A\rangle$ is said to be frequent if its support is greater than or equal to the user-defined minimum support (*fminsup*) $fsup(\langle X, A\rangle) \geq fminsup$.

Mining frequent fuzzy item sets is the problem of extracting all frequent fuzzy item sets of the form:

$$FFI_k = \{X \mid fsup(X) \geq \delta\ \times |D_f|\} \tag{1.15}$$

1.3.4 Fuzzy association rules

After obtaining the fuzzy intervals and their corresponding membership functions for each fuzzy itemset of quantitative attributes, a transformed database D_F (by fuzzification) is created from the original database. Given a fuzzy database $D_F = \{T_1, T_2, \dots, T_n\}$ with attributes $i_j \in I$ and fuzzy sets F_{ij} corresponding to the attributes in I. A fuzzy association rule has the following form:

$If\ X = \{x_1, x_2 \dots, x_p\}\ is\ A = \{a_1, a_2 \dots, a_p\}\ then\ Y\ = \{y_1, y_2 \dots, y_q\}\ is\ B = \{b_1, b_2 \dots, b_q\}$

where: $a_i \in F(x_i),\ i = 1, \dots, p$ and $b_j \in F(y_j),\ j = 1, \dots, q$. X and Y are ordered subsets of I and distinct, $X \cap Y = \emptyset$. X is called antecedent while Y is called consequent, the rule means X implies Y.

An example of an association rule takes the form: *IF AGE is Young THEN Salary is Low.*

Definition 1.9: (The support of a fuzzy association rule)

The support of a fuzzy association rule $X\ is\ A\ \Rightarrow Y\ is\ B$ is determined by the following formula:

$$fsup(\langle X\ is\ A\ \Longrightarrow Y\ is\ B\rangle) = fsup(\langle X \cup Y, A \cup B\rangle) \tag{1.16}$$

Definition 1.10: (The confidence of a fuzzy association rule)

The confidence of a fuzzy association rule $X\ is\ A\ \Rightarrow Y\ is\ B$ is determined by the following formula:

$$fconf(\langle X\ is\ A\ \Longrightarrow Y\ is\ B\rangle) = \frac{fsup(\langle X\ is\ A\ \Longrightarrow Y\ is\ B\rangle)}{fsup(\langle X, A\rangle)} \tag{1.17}$$

A rule is said to be frequent if its support is greater than or equal to fminsup, that is $fsup(\langle X\ is\ A\ \Longrightarrow Y\ is\ B\rangle) \geq fminsup$.

A rule is said to be reliable if its confidence is greater than or equal to the user-defined minimum confidence fminconf, that is $fconf(\langle X\ is\ A\ \Longrightarrow Y\ is\ B\rangle) \geq fminconf$.

1.4 Related works

1.4.1 Apriori-based approach studies

The studies based on Apriori approach were used to mine fuzzy frequent itemsets, then the remaining fuzzy frequent itemsets can be used to generate fuzzy association rules such as F-APACS [69], [31], [32]. In which, the values of the quantitative attributes are first converted into representations of linguistic terms with their membership values according to predefined membership functions.

1.4.2 Extensive studies from Apriori

Several variant algorithms have been presented to mine fuzzy association rules [70], [71], [72], [73], [74]. Then the author also developed a multi-level fuzzy mining algorithm to mine fuzzy association rules by integrating the concepts of fuzzy set and multi-level classification [28].

1.4.3 Tree-based methods

To solve the problem of computation time, Papadimitriou proposed the algorithm of Frequent Fuzzy Pattern Tree (FFPT) [75]. Lin then presents another framework for fuzzy mining to find fuzzy frequent items based on tree structure. Three algorithms, FP fuzzy frequent tree (FFP)-tree [38], compressed fuzzy frequent tree (CFFP)-tree [39] and upper bound fuzzy frequent pattern tree (UBFFP)-tree [40] have been developed to mine fuzzy frequent itemset from quantitative databases. These algorithms differ mainly in the tree structure.

1.5 Define research problem

In Apriori-based fuzzy data mining methods, quantitative values are converted into fuzzy sets according to predefined membership functions. Then, fuzzy frequent itemsets and fuzzy association rules can be generated based on the Apriori implementation. Since the execution time of Apriori-based methods is time-consuming, tree-based fuzzy data mining methods are described to speed up the mining process. Basically, these methods are modified from the FP-tree to handle fuzzy itemset. The mining of frequent fuzzy item sets is done entirely in the tree, which takes up a lot of memory space. This thesis has proposed a fuzzy frequent itemset mining algorithm based on the Node-list structure to solve the problem of memory space in the works [CT1], [CT2], [CT5].

Furthermore, member functions can be given by experts. However, expert opinions may not always be available. To solve this problem, the thesis has proposed a method to determine fuzzy sets for each quantitative attribute in the database by EMC clustering technique in the works [CT2], [CT2], [CT4].

Along with the increase in the size of data, data mining in large databases has become an important issue. Besides applying cloud computing or other parallel and distributed architectures to speed up the fuzzy mining process is also worth investigating. The thesis proposes a parallel processing method for fuzzy frequent itemset mining using the cellular learning automata approach. [CT3]

Conclusion of chapter 1

In Chapter 1, the thesis presents an overview of issues related to association rules, fuzzy logic and fuzzy association rules, different fuzzy data mining methods, including fuzzy data mining based on Apriori, tree-based fuzzy data mining and then identify the research problems of the thesis.

This thesis focuses on presenting the solution of problems in the studies [CT1], [CT2], [CT5], CT[4], [CT3]. Specifically, the thesis will focus on researching proposals and solutions to thoroughly solve the following 3 problems:

- Propose a method to determine fuzzy sets for each quantitative attribute in the database through clustering techniques.
- Propose a method to mine fuzzy frequent itemsets in quantitative databases using Node-list data structure.
- Propose a parallel processing method to mine fuzzy frequent itemset using cellular learning automata.

The remaining two chapters of the thesis will present the corresponding solutions to the three research problems above.

CHAPTER 2: FUZZY FREQUENT ITEMSET MINING BASED ON TREE STRUCTURE

In this chapter, the thesis presents the process of performing fuzzy association rule mining. In which, in the first step to convert the quantitative database to the fuzzy database, the author performs the fuzzification of the quantitative values of the items by the EMC clustering method and identifies the fuzzy intervals. The results of these two algorithms are used for the preprocessing step of the data and then applying the membership functions to convert the quantitative value to the fuzzy value. In the second step, the author proposes two popular fuzzy item set mining methods using Node-list structure based on suffix prefix tree (FPPC - Fuzzy Pre-order, Post-order Code) and FPOSC tree (Fuzzy Pre-order) Size Code). The result of the frequent fuzzy item set mining step is the main basis used to perform the search for fuzzy association rules.

2.1 State the fuzzy association rule mining problem

Given a quantitative database $D_Q = \{T_1, T_2, \dots, T_n\}$ and the set of items $I = \{I_1, I_2, \dots, I_m\}$. Given a fuzzy set $A_j = \{A_{j1}, A_{j2}, \dots, A_{jh}\}$ where A_{jk} is defined as the kth element in the fuzzy set A_j of item I_j and $f_{j,k}^{(i)}$ is the fuzzy value (defined by the membership function) of A_{jk} in transaction T_i.

Mining fuzzy association rule (MFAR) is the problem that extracts all rules of form $A \to B$ satisfied $fsp(A \to B) \geq minsup$ and $cfcf(A \to B) \geq minconf$, with $minsup$ and $minconf$ are the threshold predefined by user. The fuzzy association rule mining algorithm is implemented through three main phases:

- Phase one: Convert quantitative database to fuzzy database.
- Phase two: Extract all frequent fuzzy itemset that fuzzy support greater than minimum threshold support $FFI_k = \{A|\ \text{fsp}\ (A) \geq \delta\}$.
- Phase three: Initiate all fuzzy association rules with confidence greater than the minimal confidence threshold from the frequent fuzzy item sets found in Phase two.

2.2 Data clustering algorithm and identification of fuzzy intervals

2.2.1 Basic concepts

2.2.1.1 Data clustering

In data mining, Expectation Maximization (EM) [77]– [79] is a data clustering algorithm used in knowledge discovery tasks. The EM algorithm has the following limitations: First, EM runs fast in the initial loops but slower in the next loops. Second, EM did not always find the optimal parameter for the global, instead the local optimal.

Definition 2.1:

The coefficient of variation C_v is defined as the ratio of the standard deviation σ to the expectation $\bar{x}$ of the cluster i containing the elements $X_i\{x_{i_1}, x_{i_2}, \dots, x_{i_n}\}$:

$$C_v(X_i) = \frac{\sigma}{\bar{x}} \times 100 \qquad (2.1)$$

Definition 2.2: univariant Gaussian distribution [77]–[79]

$$N(X|\bar{x}, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(X-\bar{x})^2}{2\sigma^2}} \qquad (2.2)$$

Definition 2.3: Multivariant Gaussian distribution [77]–[79]

$$N(X|\bar{x}, \Sigma) = \frac{1}{(2\pi|\Sigma|)^{1/2}} exp\left\{-\frac{1}{2}(X-\bar{x})^T \sum^{-1} (X-\bar{x})\right\} \qquad (2.3)$$

The method of parameter estimation (Maximum Likelihood). Calculate the log for the Gaussian distribution [77]–[79].

$$ln\, p(X|\bar{x},\Sigma) = -\frac{1}{2}ln(2\pi) - \frac{1}{2}ln|\Sigma| - \frac{1}{2}(X-\bar{x})^T \sum^{-1}(X-\bar{x}) \quad (2.4)$$

Derivative:

$$\frac{\delta\, ln\, p(X|\bar{x},\Sigma)}{\delta\bar{x}} = 0, \bar{x}_{ML} = \frac{1}{N}\sum_{n=1}^{N} X_n$$

$$\frac{\delta\, ln\, p(X|\bar{x},\Sigma)}{\delta\,\Sigma} = 0, \qquad \Sigma_{ML} = \frac{1}{N}\sum_{n=1}^{N} X_n$$

Where N is the number of samples. Gaussian linear mixed distribution:

$$p(x) = \sum_{k=1}^{K} \pi_k\, \mathcal{N}(X|\bar{x}_k,\Sigma_k) \quad (2.5)$$

Where K is the Number of Gaussians and π_k is the mixing coefficient, with a weight for each Gaussian unit: $0 \le \pi_k \le 1, \sum_{k=1}^{K}\pi_k = 1$. Consider log likelihood:

$$ln\, p(X|\bar{x},\Sigma,\pi) = \sum^{N} ln\, p\,(X_n) = \sum^{N} ln\left\{\sum_{n=1}^{N}\pi_n(X_n|\bar{x}_k,\Sigma_k)\right\} \quad (2.6)$$

2.2.1.2 Determination of fuzzy intervals

When manipulating data in fuzzy databases, the most important problem is how to find a method to handle fuzzy values from which to build matching relationships between them. The values in fuzzy databases are complex, including linguistic values, numeric values, and interval values. There are many different approaches to dealing with fuzzy values that have been studied in recent years, such as: fuzzy set theory [22], possibility theory [22] 80], [81], similarity relationship [82]. The interval values are almost converted to fuzzy numbers in the form of triangles, trapezoids, bells for processing.

2.2.2 Problem

Given a database containing quantitative values D_Q.

The problem is: Determine the set of fuzzy sets of quantitative attributes in D_Q and the corresponding membership functions. Convert quantitative database to fuzzy database.

2.2.3 Data Clustering Algorithm EMC

2.2.3.1 Algorithm idea

EMC algorithm is a dynamically operated iterative optimization technique (Algorithm improved to increase flexibility for clustering while reducing local optimization and increasing global optimization).

1. Step E: For the given parameter values, we can calculate the expected value of the latent variable (Based on the given parameters of the model, calculate the probability for the expected value of the potential variables and label the data points into a group).

2. Step M: The parameters of the model are updated through the latent variables calculated according to the maximum estimation method.

3. Step C: Update the model's parameters based on latent variables calculated by the method of maximum estimator and similarity ratio among objects in a cluster and evaluate the coefficient of variation of the elements in the cluster..

2.2.3.2 EMC Algorithm

EMC algorithm is presented in Algorithm 2.1

The EMC algorithm starts with the parameters for the predictive model. Then execute the 5-process loop shown in Algorithm 2.4.

Algorithm 2.4: EMC (Expectation Maximization Coefficient)

```
Input: Initialize the value of the coefficient of variation $C_{v_{value}}$
Output: Optimal number of clusters
1:   Initialize expectation $\bar{x}_j$, covariance $\Sigma_j$, mixing coefficient $\pi_j$, coefficient of
     variation $C_v$ and evaluate for the initial value of log likelihood
2:   Step E: Based on the model parameters, calculate the probabilities of
     labeling the data points in a group
         $\gamma_j(X) = \frac{\pi_k \mathcal{N}(X|\bar{x}_k, \Sigma_k)}{}$                    (2.7)
3:   Step M: Update model parameters based on groups obtained from step E
         $\bar{x}_j = \frac{\sum_{n=1}^{N} \gamma_j(X_n) X_n}{\sum_{n=1}^{N} \gamma_j(X_n)}$                    (2.8)
                                                                    $\Sigma_j$
         $= \frac{\sum_{n=1}^{N} \gamma_j(X_n)(X_n - \bar{x}_j) X_n - \bar{x}_j{}^T}{\sum_{n=1}^{N} \gamma_j(X_n)}$        (2.9)
                                   N
4:   Evaluate log likelihood.
                                                   $\ln p(X|\bar{x}, \Sigma, \pi) = \sum^{N} \ln$
5:   Step C: Update information about the coefficients of variation of clusters
     and evaluate the variability of the elements for each cluster, specifically,
     we evaluate the coefficient of variation of the ith cluster with $C_{v_i}$ satisfying
     the variable value bias $C_{v_{value}}$ given or not.
         $C \;= \frac{\sum_{n=1}^{N} \gamma_j(X_n) X_n}{}$                    (2.12)
6:   If there is no convergence and the given $C_{v_{value}}$ variation is satisfied, go
     back to step 2. If the likelihood does not change much, the algorithm
```

2.2.3.3 Evaluation of EMC algorithm based on Log Likelihood

To evaluate the effectiveness of the EMC algorithm by the proposed statistical method through the works announced [CT3].

2.2.4 Algorithm for determining fuzzy intervals.

2.2.4.1 Determine the mean

In a fuzzy database, the domain of values of the quantitative attributes of the fuzzy object in which (the attributes can contain clear or fuzzy values) is divided into two or more fuzzy intervals. In fuzzy intervals, an element can belong to more than one interval with different degrees. In this section, it is assumed that each quantitative attribute is divided into three fuzzy intervals using a statistical approach that uses the expectation $\bar{x}$ (mean) and standard deviation (Sd) as illustrated in the below figure.

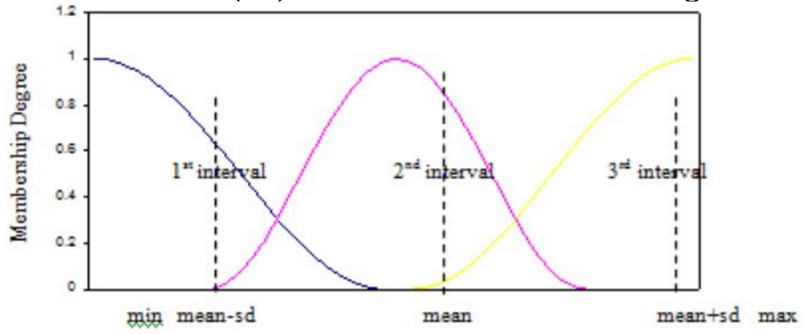


Figure 1.3: Fuzzy intervals of a quantitative attribute

The degree of overlap between the fuzzy data objects belong to two or more cluster is defined as:

$$Overlap = \frac{\sum_{j=1}^{n}|C_j|}{|\bigcup_j^n C_j|} * 100 \tag{2.14}$$

where C_j là *jth* cluster, *j=1, 2,..., n;*

2.2.4.2 Determination the intervals

The first interval: (d^-) is the lower bound of the first interval that is the minimum value in the domain of the age attribute. The upper bound (d^+) is calculated by the standard deviation and the standard deviation (Sd) of the values of the age attribute. The mathematical expressions of (d^-) and (d^+) are presented as follows:

$$\left.\begin{aligned} d^- &= MIN(X_{1Cj}, X_{2Cj}, \dots, X_{NCj}) \\ d^+ &= \bar{x} - \frac{Sd}{2} + \bar{x} \times overlap \end{aligned}\right\} \tag{2.15}$$

The membership function is used by the Z-membership to compute the membership degree for the first interval, which is shown as follows.

$$f(x)_Z = \frac{1}{2} + \frac{1}{2} cos\left(\frac{x - d^-}{d^+ - d^-}\right)\Pi \tag{2.16}$$

The second interval: The lower bound (d^-) and upper bound (d^+) of the second interval is calculated as:

$$\left.\begin{aligned} d^- &= \bar{x} - \frac{Sd}{2} - \bar{x} * overlap \\ d^+ &= \bar{x} + \frac{Sd}{2} + \bar{x} * overlap \end{aligned}\right\} \tag{2.17}$$

Both membership functions S-membership and Z-membership are used to compute the second interval, which is shown as follows:

$$\left.\begin{aligned} f(x)_S &= \frac{1}{2} + \frac{1}{2} cos\left(\frac{\bar{x} - x}{\bar{x} - d^-}\right)\Pi, \text{với } d^- \le x \le \bar{x} \\ f(x)_Z &= \frac{1}{2} + \frac{1}{2} cos\left(\frac{x - \bar{x}}{d^+ - \bar{x}}\right)\Pi, \text{với } \bar{x} \le x \le d^+ \end{aligned}\right\} \tag{2.18}$$

The third interval: The expression below is calculated for the lower bound d^- and the upper bound d^+ of the third interval as follows:

$$\left.\begin{aligned} d^- &= \bar{x} - \frac{Sd}{2} - \bar{x} * overlap \\ d^+ &= MAX(X_{1Cj}, X_{2Cj}, \dots, X_{NCj}) \end{aligned}\right\} \tag{2.19}$$

The third interval use of the membership function (S-Membership) is calculated as follows.

$$f(x)_S = \frac{1}{2} + \frac{1}{2} cos\left(\frac{d^+ - x}{d^+ - d^-}\right)\Pi \tag{2.20}$$

2.2.4.3 Converting Quantitative Database to Fuzzy Database

After determining the fuzzy intervals, the initial quantitative database is converted into a fuzzy database, preparing for fuzzy association rule mining. For each fuzzy set that we previously defined, there is a row in the new database containing the membership degree of the single elements for the particular set.

2.3 Mining fuzzy frequent itemsets

2.3.1 Problem

Given a database containing fuzzy values D_f and minimum support δ

Problem statement: Frequent fuzzy itemsets mining (FFIM) is the problem that extracts all frequent fuzzy itemsets as:

$$FFI_k \coloneqq \{X \mid sup(X) \geq \delta \times |D_f|\}$$

2.3.2 Fuzzy frequent itemset mining using FPPC-tree structure

2.3.2.1 The idea of algorithm

From the database containing the fuzzy value D_f, calculate the support of each fuzzy item A_{il} in the transaction T_q. Check if fuzzy item support A_{il} is greater than minimum support δ then add A_{il} to F_1. Sort fuzzy frequent items in F_1 by decreasing support. Fuzzy items that are not fuzzy frequent items are excluded from D_f. Build the FPPC tree.

After constructing the FPPC tree, by traversing the FPPC tree in pre-order order, we obtain the Node-list of each fuzzy (1-item) frequent item. For each N_i, node, we insert $\langle N_i.pre, N_i.post, N_i.support \rangle$ into the Nodelist of each item represented by N. The FPPC tree is deleted after obtaining the Node-list to reduce memory space.

After obtaining the Node-list of each 1-item frequent item, we perform the Node-list intersection of 1-item frequent items to find the Node-list of the item set (k-itemset). For any candidate (k + 1) P_c, we get the support of P_c by summing the support values of all the FPP_Codes in its Node-list. Based on the support of P_c, we can judge whether P_c is frequent or not. By repeating the above procedure, we find all the frequent fuzzy patterns.

2.3.2.2 Algorithm for building FPPC-tree

The algorithm for building FPPC-tree is presented in Algorithm 2.2

Algorithm 2.1: FPPC_tree_ Construction

```
Input: Database containing the fuzzy value D_f, minimum fuzzy support fminsup $\delta$.
Output:  FPPC-tree (FTr), fuzzy frequent itemset 1-itemset ($F_1$).
(1) Scan database $D_f$ containing fuzzy values to compute the support of each fuzzy
    item $A_{il}$ in the transaction $T_q$ as in the formula:
        $sup(A_{il}) = \sum_{A_{il} \subseteq T_q \wedge T_q \in D_f} f_{il}$
(2) Check if $sup(A_{il}) \geq minsup\ \delta$, put $A_{il}$ in $F_1$. That is $F_1 = \{A_{il} \mid sup(A_{il}) \geq$
    $n \times \delta\}$.
(3) Sort the frequent fuzzy items in $F_1$ in support decreasing order.
(4) if $A_{il}$ $not\ in$ $F_1$, delete $A_{il}$ from all $T_q$ $(q = 1..n)$.
(5) Create root of FPPC-tree and label it as "null"
(6) for each $T_q$ in $D_f${
(7)      Sort the remaining fuzzy items in support decreasing order;
(8)      Insert the fuzzy items into FFPC_tree (this process is similar to MFFP_tree
    [42])
(9)  }
(10)    Traverses FPPC-tree to generate the PP_Code of each node..
```

2.3.2.3 Nodelist construction algorithm of fuzzy frequent items based on FFPC-tree

Construction the Node-list of all frequent fuzzy 1-itemsets is shown in Algorithm 2.3

Algorithm 2.2: Nodelist_Construction

```
Input: FPPC-tree (R) and $L_1$ (the list of frequent fuzzy 1-itemsets)
Output: $NL_1$ (the set of Node list of frequent fuzzy 1-itemsets $L_1$)
1: Create $NL_1$, $NL_1[k]$ is the node list of a kth element in $L_1[k]$
2: **for each** node $N_i$ in R traversed in pre-order **do**
```

```
3:      **if** $N_i.f\_item = L_1[k].f\_item$ **then**
4:          insert <*N.pre*, *N.post*, *N. support*> into NL1[k]
5:      **end if**
6: **end for**
7: **return** $NL_1 = \bigcup_k NL_1[k]$;
```

- ***Nodelist intersection***

Algorithm to perform Nodelist intersection of two fuzzy frequent sets of length k is described in algorithm 2.4.

Algorithm 2.3: FNodelist_Intersection Algorithm

Input: NL_1 and NL_2 t where $NLk1$, $NLk2$ are the node list of two frequent fuzzy k-itemsets.

Output: NL_3 the node list of frequent fuzzy (k+1) itemsets.

```
(1)     for ($i = 0; i < NL_1.Size(); i++$) do
(2)         for ($j = 0; i < NL_2.Size(\,); j++$) do
(3)             if ($NL_1[i].fpre\_code < NL_2[j].fpre\_code$) then
(4)                 if ($NL_1[i].fpos\_code > NL_2[j].fpos\_code$) then
(5)                     Insert $NL_2[j]$ into NL3;
(6)                 End if
(7)             else
(8)                 if ($NL_1[i].fpos\_code < NL_2[j].fpos\_code$) then
(9)                     Insert $NL_1[i]$ into NL3;
(10)                End if
(11)            End if
(12)        End for
(13)      return NL3;
(14)    End for
```

2.3.2.4 NFFP algorithm

The NFFP algorithm is described as in Algorithm 2.5

Algorithm 2.4: Fuzzy frequent itemset mining - NFFP

Input: minimum fuzzy support *fminsup* (δ), fuzzy frequent itemset (1-item) (L_1), Nodelist of $L1$ ($NL1$);

Output: Set of fuzzy frequent itemsets (FFIs)

```
(1) For ($k = 2; L_{k-1} \neq \emptyset; k++$) do begin
(2)      For each $p = i_1 i_2 \dots i_{k-2} i_x \in L_{k-1}$ $and$ $q = i_1 i_2 \dots i_{k-2} i_y \in L_{k-1}$ , do
(3)          If $i_x \succ i_y$ then
(4)             $l = i_1 i_2 \dots i_{k-2} i_x i_y$
(5)             If each k-1 subsets $l$ in $L_{k-1}$ then begin
(6)                 l.Node-list = NL_Intersection (p.Node-list, q.Node-list);
(7)                 Calculate $l.support$;
(8)                  If ($l.support \geq n \times \delta$) then begin
(9)                     $L_k = L_k \cup \{l\}$;
(10)                    $NL_k = NL_k \cup \{l.Nodelist\}$;
(11)                end if
(12)            end if
(13)        end if
(14)     end for
(15)     Delete $NL_{k-1}$;
(16) end for
(17)   $FFIs = \bigcup_k L_k$
```

2.3.3 Mining frequent itemsets using the FPOSC-tree structure

2.3.3.1 The idea of algorithm

From the database containing the fuzzy value D_f, calculate the support of each fuzzy item A_{il} in the transaction T_q. Check if fuzzy item support A_{il} is greater than minimum support δ then add A_{il} to F_1. Sort fuzzy frequent items in F_1 by decreasing support. Fuzzy items that are not fuzzy frequent items are excluded from D_f. Build the FPOSC tree.

While building the FPOSC tree it is possible to add several child nodes without having to traverse the tree and the pre-order is calculated at the same time as the Node-list construction of the common fuzzy items. For each node N_i, we insert $\langle N_i.pre, N_i.size, N_i.f\_sup\rangle$ into the Nodelist of each item represented by N. The FPOSC tree is deleted after obtaining the Nodelist to reduce memory space.

After obtaining the Nodelist of each 1-item frequent item, we perform the Nodelist intersection of 1-item frequent items to find the Nodelist of the item set (k-itemset). For any candidate (k + 1) P_c, we get the support of P_c by summing the support values of all the FPP_Codes in its Nodelist. Based on the support of P_c, we can judge whether P_c is frequent or not. By repeating the above procedure, we find all the frequent fuzzy patterns.

2.3.3.2 Algorithm for building FPOSC-tree (Fuzzy Pre-order Size Coding)

The algorithm for constructing the FPOSC tree is determined by adjusting the structure of the FPPC tree [CT1], presented in algorithm 2.6.

Algorithm 2.5: FPOSC-Tree_Construction

Input: Fuzzy Database D_f, fminsup δ

Output: FT_r (FPOSC-tree), F_1 (frequent fuzzy itemsets (length=1)

```
Begin
1:    Traverse $D_f$ to calculate the fuzzy support for each $A_{ik}$ in transaction $T_i$
2:    If $fsup(A_{jk}) \geq \delta$ then
3:        Insert $A_{ik}$ into $F_1$;
4:    End if
5:    If $A_{ik}$ not in $F_1$ then
6:        Delete $A_{ik}$ from all $T_i$ $(i = 1 \dots n)$
7:    End if
8:    Create $FT_r$ NodeRoot=null
9:    Let Flist be the list containing the remaining fuzzy items in each $T_i$
10:   For each $T_i$ in $D_f$
11:       Sort FList in descending order of fsup
12:       $e = FList[0]$ ; e is the first element in Flist
13:       $List_r = List[size - 1]$
14        Insert_tree ($[e|\ List_r], FT_r$ )
15:   End for
```

/* Procedure Insert_Tree is used to recursively invoke the building of the POSC-tree. Where, e is the first element of Flist and Flist is the remaining list */

```
Procedure Insert_tree ($[e|\ List_r], FT_r$ )
1:    Let N is a node corresponds to a branch in the $FT_r$
2     If $e.fitem == N.fitem$ then
3:        Add fuzzy value $f_{j,k}^{(i)}$ of e to fsup of N;
4:    Else
5:        Create a new node N that has fsup as $f_{j,k}^{(i)}$ and add N into the end of
      corresponding branch;
```

```
6:              $N.size = 1$;
7:              If $List_r$ is nonempty then
8:                  Call Insert_Tree ($List_r$,N) recursively
9:              End if
10:         End if
11:         $N.size = N.countChild + 1$
End procedure
```

2.3.3.3 Nodelist construction algorithm of fuzzy frequent items based on FPOSC-tree

The node-list construction of frequent fuzzy item sets (length = 1) (F1 is presented in Algorithm 2.7.

Algor.6: FNode_List_Gen
Input: POSC-tree (FT_r), fuzzy frequent itemset length=1 (F_1)
Output: Node-list of F_1 (NL_1)

```
Begin
1:      for each $N_i$ in $FT_r$ browsed by pre-order do
2:              Let $NL_1[k]$ be Node-list of item kth in $F_1$.
3:              If $N_i.f_{item} == F_1[k].f\_item$ then
4:                      insert $\langle N_i.pre, N_i.size, N_i.fsup \rangle$ into $NL_1[k]$;
5:      Return $NL_1 = \bigcup_k NL_1[k]$
End.
```

The method for construction the intersection of two Node-list is presented in Algorithm 2.8.

Algorithm 2.7: POS_Node-list_Intersect
Input: NL_{k1}, NL_{k2} trong đó NL_{k1}, NL_{k2} are the Node-list of two fuzzy frequent k-itemsets.
Output: NL_{k1+1} – Node-list of fyzzy frequent (k+1) itemsets

```
Begin
1:      for $i = 0; i < NL_{k1}.length; i++$ do
2:          For $j = 0; j < NL_{k2}.length; j++$ do
3:              If $NL_{k1}[i].pre < NL_{k2}[j].pre$ then
4:                  If $NL_{k2}[j].pre < NL_{k1}[i].pre + NL_{k1}[i].size$ then
5:                      Insert $NL_{k2}[j]$ into $NL_{k1+1}$;
6:                  End if
7:              else
8:                  If $NL_{k1}[i].pre < NL_{k2}[j].pre + NL_{k2}[j].size$ then
9:                      Insert $NL_{k1}[i]$ into $NL_{k1+1}$;
10:                 End if
11:             End if
12:         End for
13:     End for
End.
```

2.3.3.4 Algorithm NPSFF

Algorithm 2.8: The algorithm for mining fuzzy frequent itemset NPSFF

```
Input: minimum fuzzy support fminsup ($\delta$), fuzzy frequent itemset (1-item) ($L_1$),
Nodelist of $L_1$ ($NL_1$);
Output: Set of fuzzy frequent itemsets (FFIs)
1:   For ($k = 2; L_{k-1} \neq \emptyset; k++$) do begin
2:       For each $p = i_1 i_2 \ldots i_{k-2} i_x \in L_{k-1}$ and $q = i_1 i_2 \ldots i_{k-2} i_y \in L_{k-1}$ , do
3:           If $i_x \succ i_y$ then
4:               $l = i_1 i_2 \ldots i_{k-2} i_x i_y$
```

```
5:              If each k-1 subsets $l$ in $L_{k-1}$ then begin
6:                  $l$.Node-list = POS_Node-list_Intersect ($p$.Node-list, $q$.Node-list);
7:                  Calculate $l.support$;
8:                  If $(l.support \geq n \times \delta)$ then begin
9:                      $L_k = L_k \cup \{l\}$;
10:                     $NL_k = NL_k \cup \{l.Nodelist\}$;
11:                 End if
12:             End if
13:         End if
14:     End for
15:     Delete $NL_{k-1}$;
16: End for
17: $FFIs = \bigcup_k L_k$ ;
```

2.4 Algorithm for mining fuzzy association rules

All phases in mining fuzzy association rules is performed in Algorithm 2.10 as below:

Algorithm 2.9: MFAR

Input: A quantitative database (D_Q), minimum support threshold δ, minimum confidence minfc

Output: All fuzzy association rules FRs

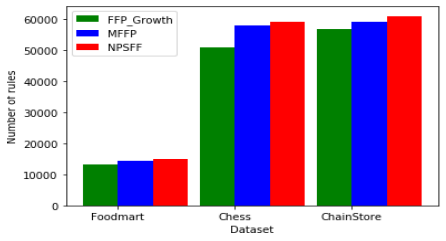
```
Begin
1:  Transform $D_Q$ to $D_f$
2:  Execute FPOSC _Tree_ Construction (Df, $\delta$) to generate FPOSC Tree (FTr),
3:  Execute FNode-list Gen (FTr, $F_1$)
4:  Execute NPSFF $(\delta, L_1, NL_1)$ to find all FFIs
5:  $FRs = \emptyset$;
6:  For each $X \in FFIs$ do
7:      For each $Y \subset X$ && $Y \neq \phi$ do
8:          $fr = X \setminus Y \rightarrow Y$;
9:          $fc(fr) = \frac{sup(XY)}{sup(X)}$;
10:         If $fc(fr) \geq mincf$ then
11:             $FRs = FRs \cup \{fr\}$;
12:         End if
13:     End for
14: End for
15: Return $FRs$
```

2.5 Experiment

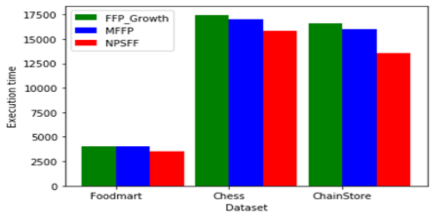
In the experiment, author use the data set obtained from datasets for Frequent Itemsets mining [79] called Foodmart, Chess and Chain store. Each transaction in this data set includes all the items acquired by a customer in one time. The description about data set is shown in Table 2.1. To resolve with the quantitative database, we assigned random numbers for all items in this data set with a distribution in the value range 1–100.

Table 2.1: Description of the data set for the experiment

Dataset	**Number of transactions**	**Number of items**	**Average number of items per transaction**
Foodmart	4,141	1,559	4.42
Chess	3,196	75	37
ChainStore	111,294	46,086	7.23



Hình 2.1: Number of rules generated from 3 algorithms



Hình 2.2: Execution time of algorithms

2.6 Conclusion of chapter 2

In this chapter, author offers solutions to problems related to "sharp" boundaries between fuzzy intervals by proposing an algorithm for clustering EMC data. The results of this algorithm [CT4] are used in the data preprocessing stage, data partitioning to convert quantitative databases to fuzzy databases. Second, the thesis solves the problem of memory space by providing two fuzzy association rule mining methods based on the Nodelist data structure, namely NFFP [CT1] and NPSFF [CT2]. The author proposes two NFFP algorithms, NPSFF uses FPPC_tree, POSC-tree to store quantitative database with membership values in descending order. Based on the constructed tree, a Nodelist of each common fuzzy item is generated. Then, the NFFP, NPSFF algorithm obtains the Nodelist of the common fuzzy items (k + 1) by intersecting the Nodelist of the frequent k fuzzy items and then extracts the frequent (k + 1) fuzzy files. The advantage of this algorithm is that the FPPC_Tree tree as well as the POSC-tree is used to generate the FPP_Code or POS-code for each node to get the Nodelist of each common fuzzy item and then it will be deleted so that the request can be reduced memory usage requirements.

With the increase in data size, the data types are heterogeneous and dynamic data. Extending efficient fuzzy mining algorithms to the era of big data is an important problem, mining by applying parallel processing techniques has become a viable way to overcome the problem of time. processing time. This issue is discussed in chapter 3.

CHAPTER 3: MINING FUZZY FREQUENT ITEMSETS USING PARALLEL PROCESSING

In this chapter, the author presents a parallel processing method for mining frequent fuzzy itemsets, an important stage in fuzzy association rule mining by using a cellular learning automata (CLA) approach. According to CLA, space is represented as a network, with each element being a cell, line by line, transaction data will be read and simultaneously transferred to cells, they will collaborate with each other in parallel. Without using neighborhood rules, a type of data automation known irregular cellular learning automata (ICLA) is used to generate a neighborhood list for each cell. Through using CLA fuzzy frequent itemset mining is performed. This process shortens the execution time of the algorithm.

3.1 Introduction

In recent years, many algorithms have been developed to study parallel mining problems for association rules, classification, clustering, and other tasks. Agrawal et al. proposed the first parallel association rule mining algorithm [85]–[88], while Wang studied other parallel association rule mining algorithms [89]–[91]. Among parallel architectures, the master-slave architecture is often used. This approach offers significant performance benefits [92]. The main processor allocates tasks to the sub-processors and collects results from them. Some studies use parallel slave-master architecture to perform association rule mining suitable for dense data sets such as [94] [95].

In the field of association rule extraction and PSO, researchers have proposed many parallel computing algorithms [96], [97], [98]. For a large amount of experimental data, a parallel PSO algorithm applied to extract association rules is a possible solution.

In addition, the iMFFP algorithm [99] proposed to integrate different MFFP trees [41] from the branch databases and integrated into the iMFFP tree in sequence. Then, Header_table is created, and frequent item set mining is performed. With this method, the calculation of fuzzy support of fuzzy items will be inaccurate and incomplete because the database is decomposed. Moreover, building each branch of the MFFP tree and gradually integrating it into the complete iMFFP tree will consume memory space.

In this chapter, the author presents a parallel processing method for mining frequent fuzzy item sets, an important stage in fuzzy association rule mining by using a cellular learning automata (CLA) approach. In this strategy, the initial quantitative database is transformed into a fuzzy database in the preprocessing step. After extracting the fuzzy frequent 1-itemset from the data set, the infrequent fuzzy itemset will be removed. The CA environment will start working after the preprocessing phase and generate CA cells that match each fuzzy frequent 1-itemset. Each line of data in the compressed database is read and sent to cells concurrently, then they work in parallel.

3.2 Some concepts about cellular learning automata

3.2.1 Learning Automata (LA)

An LA consists of two parts:

1. A random automata with a limited number of actions and a random environment.
2. Learning algorithm: the algorithm by which the automata will learn the optimal action using that action.

Each action selected by the potential environment is evaluated and an answer is given to an auto-learning data. LA will use this answer and choose its action for the next phase. Figure 3.1 shows the relationship between auto-learning data and the environment [101].

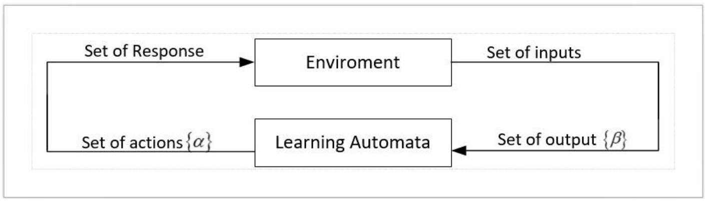


Figure 3.1: Environment, LA and their relationship

3.2.2 Cellular Automata (CA)

A d-dimensional Cellular Automata is a structure of $A = (Z^d, \Phi, N, F)$ [35] where:

- Z^d is a lattice of d-tuples of integer number of which this lattice could consist finite lattice, infinite lattice or semi-finite.
- $\Phi = \{1, \dots, m\}$ is a finite set of states.
- $N = \{x_1, x_2, \dots, x_m\}$ is a finite subset of Z^d called the neighborhood vector ($x_i \epsilon Z^d$).
- F is the local rule of the cellular automata.

3.2.3 Cellular learning automata

Automata is combination of two recent models LA and CA. A d-dimensional cellular Learning Automata is a structure CLA: $A = (Z^d, \Phi, A, N, F)$

- Z^d s a lattice of d-tuples of integer number which this lattice could consist of finite lattice, infinite lattice or semi-finite.
- $\Phi = \{1, \dots, m\}$ is a finite set of states.
- A is collection of learning automat (LA) each of which is assigned to one cell of the CLA. Each cell can have a LA or more than one.
- $N = \{x_1, x_2, \dots, x_m\}$ is a finite subset of Z^d called the neighborhood vector ($x_i \epsilon Z^d$).
- F is the local rule of the cellular automata. This rule can be defined by users.

3.3 Fuzzy frequent itemset mining algorithm using CLA

3.3.1 The idea of algorithm

In the CLA-Fuzzy Mining algorithm [CT3] is performed according to the below procedure:

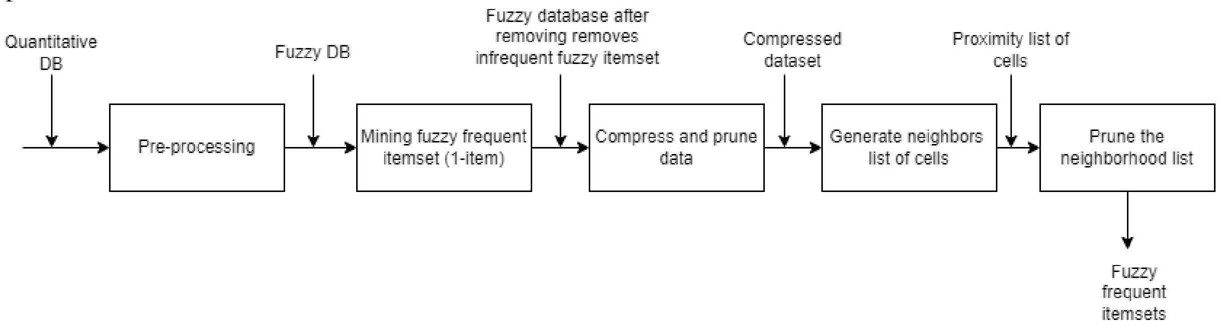

Quantitative DB
Fuzzy DB
Fuzzy database after removing removes infrequent fuzzy itemset
Compressed dataset
Proximity list of cells
Pre-processing
Mining fuzzy frequent itemset (1-item)
Compress and prune data
Generate neighbors list of cells
Prune the neighborhood list
Fuzzy frequent itemsets

Figure 3.2: CLA-Fuzzy Mining algorithm implementation process

3.3.2 Pre-processing

In this step, the database is converted from quantitative database to fuzzy database.

3.3.3 Mining fuzzy frequent 1-item

The fuzzy frequent itemset (1-item) mining is performed like the algorithms in Chapter 2. The fuzzy support of each item in the transaction is calculated by the formula and tested with the minimum support.

3.3.4 Mining fuzzy frequent n-itemset

➢ **Perform data compression**

Data compression algorithm is shown in Algorithm 3.1

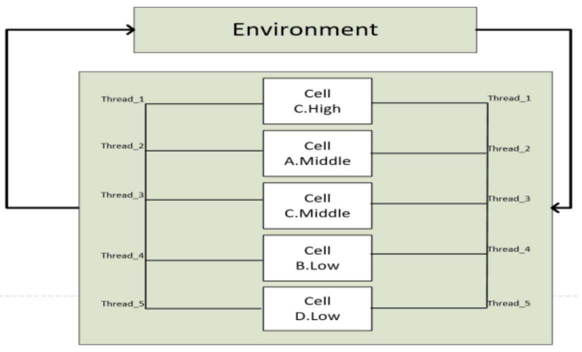
Algorithm 3.1: Data_Compression()

Input: $minsup$: minimum support threshold

```
Output: $CDS$: Compressed dataset
Begin
1:      for $i = 1$ to $D_f$ do
2:          For $j = 1$ to $items$ do
3:              If $items(i,j) == items(i+1,j)$ then
4:                  Remove (rows (i+1));
5:                  Update support (rows(i)+ rows(i+1));
6:              End if
7:          End for
8:      End for
9:      Return CDS
End.
```

- **Determine the neighborhood list**

Each cell learning automata will be created according to the frequent fuzzy 1-itemsets. Because of using ICLA, there is no specific rule in cell's neighborhoods. The cellular automata environment reads line by line each transaction in compressed dataset and sends the frequent fuzzy 1-itemset to the cells. After receiving a row containing fuzzy items from the dataset, cells begin their operation at the same time as others. These cells will update their proximity list depending on the fuzzy items in the received transaction.



Hình 3.3: Automata cells according to frequent fuzzy 1-itemset

- **Prune the neighborhood list**

When all transactions are sent by the environment to cells, each cell deletes neighbors and neighborhoods whose support is less than a user-defined minimum threshold from its list of neighborhoods. The neighborhood list is again used to scan and finally obtain the fuzzy frequent k-Itemset. If these items are already on this list, they will be removed; otherwise, these items will be included in the list of fuzzy frequent itemset.

3.3.5 CLA-FuzzyMining algorithm

The CLA-Fuzzy Mining algorithm is described as in Algorithm 3.2

Algorithm 3.2: CLA_Fuzzy_Mining

```
Input: $minsup$: minimum support threshold
       $F_1$: fuzzy frequent 1-item
       $D_f$: Fuzzy database after removing removes infrequent fuzzy itemset
       CDS: compressed dataset
Output: $FFIL$:  Fuzzy frequent itemsets
Begin
1:      for $i = 1$ to $CDS$ do
3:          CLA_Thread();
4:      End for
5:      Initialize FFIL;
6:      for i=1 to automata cells do
```

```
7:          Execute PruneNeighbors() for cell[i];
8:          Execute DFS() function for cells[i];
9:          for each anItemset on cell[i].FrequentItemset do
10:             if anItemset does not exist in FFIL then
11:                 FFIL.add (anItemset);
12:             else
13:                 Nothing;
14:             End if
15:         End for
16:     End for
17:     Return FFIL;
End.
```

The CLA_Thread() function is described in Algorithm 3.3.

Algorithm 3.3: CLA_Thread()

Input: Recodset (compressed data record), NodeParent[Cell] (representative of cells)

Output: automata cells

```
Begin
1:      Thread theard=new Thread();
2:      thread.Start();
3:      Initialize nodeChil=new Node();
4:      for $i = 1$ to Recodset do
5:          nodeChil.data= Recodset[value];
6:          If(nodeChil in (Recodset)) then
7:              nodeChil.data= Recodset[value]+ nodeChil.data;
8:          else
9:              NodeParent[Cell].next= nodeChil;
10:         End if
11:     End for
12:     Return AutomataCells;
End.
```

3.4 Experiment

In the experimental part the author uses the Foodmart, Chess and ChainStore data sets from the frequent set mining dataset [69] for this test. The description of the data set is shown in Table 3.7. This experiment introduces the experimental results from the algorithms and compares them with the results of the NPSFF algorithm [CT2] and the iMFFP algorithm [33]. CLA- Fuzzy Mining algorithm is more efficient than the previous two algorithms in terms of processing time and temporary storage memory, according to the test results based on the data set presented in figure 3.12 – 3.14.

Table 3.7: Table of experimental data

Dataset name	Transaction#	Items#	Size
Chess	3196	175	0.78 M
Foodmart	4141	1559	12.4 M
ChainStore	111,294	46,086	28.17 M

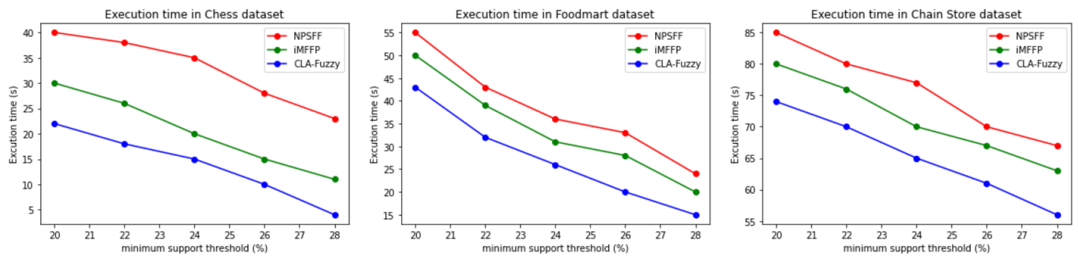


Figure 3.12 – 3.14: Experimental time on data sets

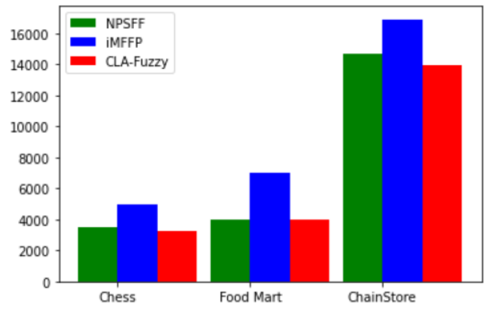


Figure 3.4: Evaluation of memory usage of algorithms on data sets

3.5 Conclusion of chapter 3

To increase efficiency in big data models, records are constantly updated. Chapter 3 focuses on presenting the fuzzy frequent item set mining method according to CLA parallel processing technique. According to CLA, space is represented as a network, with each element being a cell, line by line, transaction data will be read and simultaneously transferred to cells, they will collaborate with each other in parallel. Without using neighborhood rules, a type of data automation known as irregular cellular learning automata (ICLA) is used to generate a neighborhood list for each cell. Through using these automatic data cells, frequent fuzzy set mining is performed. This process shortens the execution time of the algorithm. [CT3].

CONCLUSION

The main purpose of the thesis is to study some fuzzy association rule mining methods. The thesis researches the methods of association rule mining on fuzzy data based on the combination of fuzzy math and the proposed quantitative database. However, these methods are in the process of development, it is necessary to propose new solutions to improve them. Therefore, the thesis proposes an effective approach to the problem of mining fuzzy association rules.

The main results of the thesis are as follows:

(1) Propose a method to determine fuzzy sets for each quantitative attribute in the database through EMC clustering technique. These clusters are then used to classify each quantitative attribute as a fuzzy set and determine their membership functions. The result of this step is to convert the quantitative database to the fuzzy database. [CT2], [CT4].

(2) Propose a method to mine common fuzzy item set based on Nodelist structure, an important step in fuzzy association rule mining. Common fuzzy item set mining based on PP_code or POS_code helps to limit the required memory consumption. [CT1], [CT2]

(3) Propose a parallel processing method for the process of mining frequent fuzzy item sets using CLA mobile auto-learning theory. With this proposal, we aim to reduce processing time for large databases. [CT3

LIST OF PUBLISH

No	ARTICLES
[1]	Tran, T. T., Nguyen, G. L., Truong, C. N., & Nguyen, T. T. “Mining Frequent Fuzzy Itemsets Using Node-List”. Information Systems Design and Intelligent Applications Springer, Singapore, 37-48, 2018
[2]	Tran, T. T., Nguyen, T. N., Nguyen, T. T., Nguyen, G. L., & Truong, C. N., “A Fuzzy Association Rules Mining Algorithm with Fuzzy Partitioning Optimization for Intelligent Decision Systems”. International Journal of Fuzzy Systems, 1-14, 2022 (SCIE – Q2)
[3]	Tran, T. T., Nguyen, T. T., Nguyen, G. L., & Truong, C. N. “Parallel Fuzzy Frequen Itemset Mining Using Cellular Automata”. Journal of Computer Science and Cybernetics, 38(4), 293-310, 2022.
[4]	Tran Thi Thuy Trinh, Nguyen Long Giang, Truong Ngoc Chau, Nguyen Tan Thuan. “Partitioning fuzzy data using statistical methods in fuzzy association rule mining”. Proceedings of the National Conference on Selected Issues of Information & Communications Technologies VNICT – Quy Nhơn, 2017
[5]	Tran Thi Thuy Trinh, Nguyen Tan Thuan, Nguyen Long Giang, Truong Ngoc Chau, Nguyen Quang Huy. “Intelligent learning advisory model applying fuzzy association rules”. Proceedings of the National Conference on Selected Issues of Information & Communications Technologies VNICT – Quảng Ninh, 2020