Journal of Computer Science and Cybernetics, V.38, N.4 (2022), 293–310
DOI no. 10.15625/1813-9663/38/4/17462

PARALLEL FUZZY FREQUENT ITEMSET MINING USING CELLULAR AUTOMATA

TRINH T.T.TRAN[1], THUAN T.NGUYEN[2], GIANG L.NGUYEN[3], CHAU N.TRUONG[4]

[1]*Duy Tan University, Da Nang, Viet Nam*
[2]*The University of Da Nang, Viet Nam- University of Technology and Education*
[3]*Institute of Information Technology, VAST, Ha Noi, Viet Nam*
[4]*The University of Da Nang, Viet Nam - University of Science and Technology*

Abstract. Finding frequent fuzzy itemsets in operational quantitative databases is a significant challenge for fuzzy association rule mining in the context of data mining. If frequent fuzzy itemsets are detected, the decision-making process and formulating strategies in businesses will be made more precise. Because the characteristic of these data models is a large number of transactions and unlimited and high-speed productions. This leads to limitations in calculating the support for itemsets containing fuzzy attributes. As a result, mining using parallel processing techniques has emerged as a potential solution to the issue of slow availability. This study presents a reinforced technique for mining frequent fuzzy sets based on cellular learning automata (CLA). The results demonstrate that frequent set mining can be accomplished with less running time when the proposed method is compared to iMFFP and NPSFF methods.

Keywords. Frequent fuzzy itemsets; Cellular automata; Parallel mining.

1. INTRODUCTION

Recently, data mining techniques have been designed to obtain useful information from databases [1, 4, 21]. Depending on the variety of knowledge, data mining methods can be divided into categories: association rule [1,4,18,21,27], classification [21,36,39,41], clustering [20, 25, 29], and sequential samples [3, 35]. In particular, association rule mining is crucial to data mining research. [7, 9, 30]. In the first part of the two-stage procedure known as association rule mining, frequent itemsets are taken from a given data collection. From the extracted frequent itemsets, association rules are constructed in the second stage. The main stage of association rule mining is frequent itemset mining since it takes a lot of effort to locate frequent itemsets in a data set. Most research in this area has focused on enhancing frequent itemset mining's efficiency in terms of time and memory. Previous research has mostly concentrated on the binary display of transactional data, that is, solely concerned with the existence or absence of items. However, in practical applications, in addition to attribute values 0 and 1, the database also contains other properties with quantitative values. Due to

*Corresponding author.
E-mail addresses: thuytrinh85.dtu@gmail.com (T.T.T.Tran); thuannt.it.dtu@gmail.com (T.T.Nguyen); nlgiang@ioit.ac.vn (G.L. Nguyen); tnchau73@gmail.com (C.N. Truong).

its ease of use and resemblance to human inference, fuzzy set theory [22, 40] is being used in intelligent systems more frequently [32, 34, 38]. Because linguistic representation makes knowledge simpler for humans to understand, it is widely used. Fuzzy mining algorithms use membership functions that convert quantitative values into a fuzzy set representing linguistic values [19]. In the past, Janikow combined symbolic decision trees on rule-based systems for fuzzy control [23] using fuzzy representation. To extract fuzzy association rules from quantitative data, Hong et al. suggested a fuzzy mining algorithm in the article [15]. Lin et al. then proposed a fuzzy FP tree [28] to improve the mining of fuzzy frequent itemsets from quantitative databases. The multiple fuzzy regular sample tree (MFFP-tree) approaches was later put out by Hong et al. [17] to gather more information than the fuzzy FP-tree algorithm. The algorithms for mining frequent fuzzy itemsets that have been developed are then based on candidate creation, whilst some attempt to make a tree without producing candidates and then locate frequent fuzzy items by scanning the tree. The CFP (Compact Frequent Pattern) tree algorithm, as proposed by K. Suriya Prabha and R. Lawrance [31] is used in the frequent fuzzy itemset mining algorithm. The approach improves the classical FP-tree to locate fuzzy frequent itemsets from quantitative transactions and constructs a fuzzy CFP-tree from the quantitative database. The approach suggests creating a small subtree for each frequent fuzzy item, producing several candidates from the small subtree, and then releasing the current subtrees to include the fuzzy theory concept into the CFP-tree process. The performance is superior to other algorithms in terms of execution time, memory usage, and reduced search space for mining fuzzy frequent items. Time out of the memory area makes room for the next subtree. In the article [5], frequent itemset mining from various datasets with fuzzy data is mentioned. To find fuzzy multi-level association rules in a relational database architecture that can accommodate several tables, the author of the article joins numerous tables using a star schema. To locate frequently occurring itemsets, the algorithm uses joins and entities. The conclusions of the paper's calculations of the support of itemsets connected to other connections with fuzzy-valued attributes, however, still have significant limitations.

Mining using parallel processing techniques has emerged as a viable solution to this issue [2, 8, 16, 26] as a result of the significant growth in computing power that has become available and the concurrent decline in computational cost over the past two decades. In a previous study [26], the author integrated individual databases into a common database. The algorithm then builds each MFFP sub-tree for each sub-database and integrates them into one. The branches in the sub-MFFP tree are efficiently extracted and integrated into the iMFFP tree in sequence. The division of the database into sub-databases and then re-integrated often causes information loss and inconsistent data. Furthermore, constructing and storing sub-trees is often expensive in memory.

In this strategy, the initial quantitative database is transformed into a fuzzy database in the preprocessing step. After extracting 1-item fuzzy frequent itemsets from the data set, the infrequent fuzzy items will be removed. The CA environment will start working after the preprocessing stage and generate CA cells that match each 1-item frequent fuzzy item. Each line of data in the compressed database is read and delivered to the cells concurrently, then they operate in parallel. In this method, L_{RI} mode is used. With this model, there will be no penalty and no rules for the neighborhood between cells. ICLA refers to the cluster of cells generated by data set transactions. The proximity list for neighbors is maintained

by the cells and is updated every time a transaction is received from the environment. As a final step, each cell checks its list of neighborhoods and cuts out the neighborhoods whose support is below the user's minimum threshold. Then each cell collects frequent fuzzy items sorted by proximity list, sends them into the environment, and so on. With the use of the neighborhood list in CLA that helps to limit the requested memory consumption.

The remaining sections of this paper are as follows: Related studies are included in Section 2. The fuzzy frequent itemset mining approach based on cellular learning automata is described in Section 3. Sections 4 and 5 present, respectively, the experimental result and the conclusion.

2. RELATED WORKS

2.1. Statement problem

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

The following formula is used to calculate an item's fuzzy support

$$f\sup(A_{jk}) = \frac{1}{n}\sum_{i=1}^{n} f_{j,k}^{(i)}. \tag{1}$$

Assume that a set of fuzzy items $A = \{A_1, A_2, \ldots, A_p\}$ has a transaction fuzzy value T_i that is determined by the intersection of the membership values $f_{Ak}^{(i)}$

$$f_A^{(i)} = \bigcap_{k=1}^{p} f_{A_k}^{(i)}. \tag{2}$$

In this study, the minimum function is used as an intersection operator along with τ -norm like $a\tau b$ in a fuzzy set. Consequently, the fuzzy value of A is

$$f_A^{(i)} = \min f_{A_k}^{(i)}. \tag{3}$$

where $1 \leq k \leq p$. The support of fuzzy itemset $A = \{A_1, A_2, \ldots, A_p\}$ is defined by

$$f\sup(A) = \frac{1}{n}\sum_{i=1}^{n} f_A^{(i)} \tag{4}$$

The problem of mining frequent fuzzy itemsets is expressed as: $FFI_p = \{A \mid \sup(A) \geq \delta\}$ where δ is a minimal fuzzy that aid the threshold known as minsup.

2.2. Cellular learning automata

2.2.1. Learning automata

Learning automata (LA) [13] is an abstract model that discovers the best course of action from a limited pool of options through repeated interactions with an unknowable random environment. After assessing the chosen course of action, the environment replies to the LA with an improved signal. The LA modifies its internal state and chooses the subsequent action based on the chosen action and the signal it received.

2.2.2. Cellular learning automata

The Cellular Learning Automata (CLA), which combines CA and LA [6], was developed by Beigy in 2004. Assigning an auto-learning cell to each cell to modify the state transition of the related auto-learning cell is the fundamental concept of CLA. The learning automaton that resides in each cell decides on its action (the cell's state) at each instant in time based on the action probability distribution. An auto-learning cell's immediate surroundings are other auto-learning cells located inside of nearby cells. The local environment is regarded as non-stationary because the neighborhood automatons' action probability distributions fluctuate as CLA evolves. Similar to CA, CLA runs under local rules. Based on local rules and the actions chosen by the surrounding auto-learning cell, the reinforcement signal is sent to the auto-learning cell. The auto-learn then modifies its action probability vector in accordance with the reinforcement signal it has received. This procedure is repeated until each cell is in its ideal state. CLA is applied in business related problems such as mining customer behaviour in shopping activity [10] or for community detection in complex social networks [24].

2.2.3. Irregular cellular learning automation

The expansion of conventional CLA known as irregular cellular learning automation (ICLA) [11] does away with the restriction of rectangular grid configurations. Since many applications, like those involving graphs, wireless sensor networks, and immunological networks, cannot be described using standard meshes, such generalization appears to be important. Each node in ICLA is a cell with an auto-learner, and the node's neighbors make up the cell's local environment. ICLA is thought of as an undirected graph. ICLA is functionally similar to CLA despite having an odd structure. ICLA has been discovered to be effective in a variety of applications [11, 12, 33].

3. PROPOSED METHOD

In this part, we propose an ICLA-based frequent fuzzy itemset mining technique. In this strategy, the quantitative database is initially transformed into a fuzzy database by data preparation. Following the extraction of frequent fuzzy 1-itemsets from the data set, infrequent fuzzy items will be eliminated. The cellular automata environment will begin operating following the preprocessing stage and produce cellular automata cells in accordance with each frequent fuzzy 1-item. The resultant dataset's rows are read and concurrently delivered to the cells, who then cooperate with one another in parallel. In this method, the LRI mode is utilized, which benefits from CLA, where there is no penalty and no regulation for the neighborhood between cells. The ICLA refers to the cluster of cells produced by dataset transactions. The proximity list for neighbors is maintained by the cells and is updated each time a transaction from the environment is received. The last step is for each cell to check its proximity list and prune the neighborhoods whose support falls below the minimal user-defined threshold. Then, each cell collects frequent fuzzy objects based on a proximity list-curated survey, sends them into the environment, and so forth.

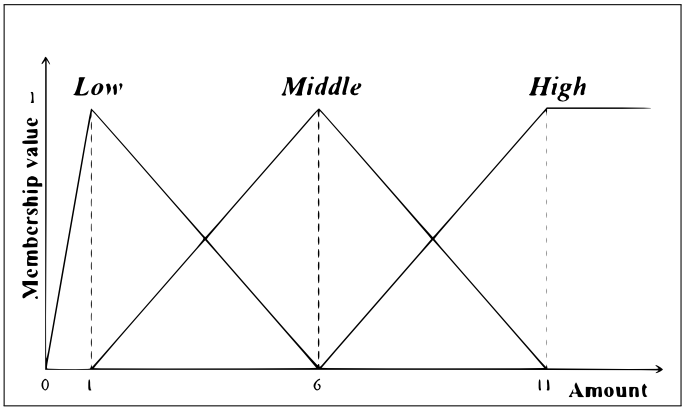

Figure 1: Membership function in the example

3.1. Preprocessing

Assume that there is a quantitative database which has 4 transactions and contains items A, B, C, D, and E as shown in Table 1, the minimum support is set to 30%.

Table 1: Quantitative dataset

Tid	Items
1	$(A:3)$ $(C:9)$ $(D:3)$ $(E:6)$
2	$(A:7)$ $(B:2)$ $(C:8)$
3	$(B:2)$ $(C:9)$
4	$(A:8)$ $(C:10)$ $(D:2)$
5	$(A:5)$ $(B:2)$ $(C:7)$
6	$(A:5)$ $(C:10)$ $(D:3)$ $(E:2)$
7	$(A:3)$ $(B:3)$ $(C:9)$ $(E:6)$
8	$(C:9)$ $(D:3)$

Assume that all of the items in Figure 1 have the same membership function. The items in this example are represented by the three fuzzy regions Low, Middle, and High. The membership function in Figure 1 is used to calculate the membership values for each item.

The database after being converted from a quantitative database to a fuzzy database is shown in Table 2.

3.2. Mining frequent fuzzy 1-itemset

The fuzzy support of each fuzzy item in the transaction is calculated as (Eq.1) and is checked against the minimum support threshold. The support of fuzzy items is shown in Table 3.

Fuzzy items with support less than the threshold are removed from the data set. The remaining items satisfy the minimum support of 30% as shown in Table 4.

In this example, we use 15 fuzzy items, and the minimum support is assumed to be 30%.

Table 2: database after being converted from quantitative database to fuzzy database

Tid	Items
1	$\left(\frac{0.6}{A.Low}, \frac{0.4}{A.Middle}\right)\left(\frac{0.4}{C.Middle}, \frac{0.6}{C.High}\right)\left(\frac{0.6}{D.Low}, \frac{0.4}{D.Middle}\right)\left(\frac{0.2}{E.Low}, \frac{0.8}{E.Middle}\right)$
2	$\left(\frac{0.8}{A.Middle}, \frac{0.2}{A.High}\right)\left(\frac{0.8}{B.Low}, \frac{0.2}{B.Middle}\right)\left(\frac{0.6}{C.Middle}, \frac{0.4}{C.High}\right)$
3	$\left(\frac{0.8}{B.Low}, \frac{0.2}{B.Middle}\right)\left(\frac{0.4}{C.Middle}, \frac{0.6}{C.High}\right)$
4	$\left(\frac{0.6}{A.Middle}, \frac{0.4}{A.High}\right)\left(\frac{0.2}{C.Middle}, \frac{0.8}{C.High}\right)\left(\frac{0.8}{D.Low}, \frac{0.2}{D.Middle}\right)$
5	$\left(\frac{0.2}{A.Low}, \frac{0.8}{A.Middle}\right)\left(\frac{0.8}{B.Low}, \frac{0.2}{B.Middle}\right)\left(\frac{0.8}{C.Middle}, \frac{0.2}{C.High}\right)$
6	$\left(\frac{0.2}{A.Low}, \frac{0.8}{A.Middle}\right)\left(\frac{0.2}{C.Middle}, \frac{0.8}{C.High}\right)\left(\frac{0.6}{D.Low}, \frac{0.4}{D.Middle}\right)\left(\frac{0.8}{E.Low}, \frac{0.2}{E.Middle}\right)$
7	$\left(\frac{0.2}{A.Low}, \frac{0.8}{A.Middle}\right)\left(\frac{0.2}{C.Middle}, \frac{0.8}{C.High}\right)\left(\frac{0.6}{D.Low}, \frac{0.4}{D.Middle}\right)\left(\frac{0.8}{E.Low}, \frac{0.2}{E.Middle}\right)$
8	$\left(\frac{0.8}{B.Low}, \frac{0.2}{B.Middle}\right)\left(\frac{0.4}{C.Middle}, \frac{0.6}{C.High}\right)$

Table 3: The support of fuzzy items

Items	Support
$A.Low$	1.6
$A.Middle$	3.8
$A.High$	0.6
$B.Low$	3
$B.Middle$	1
$B.High$	0
$C.Low$	0
$C.Middle$	3.4
$C.High$	4.6
$D.Low$	2.6
$D.Middle$	1.4
$D.High$	0
$E.Low$	1
$E.Middle$	1.8
$E.High$	0.2

Therefore, only fuzzy itemset with a value greater than 2.4 are kept and other itemset smaller than that are discarded. The result is shown in Table 5.

3.3. Compressing and pruning the dataset

We group related transactions together to reduce the need for repetitive procedures. Before reading the current row and extracting the 1-itemset common fuzzy items and removing the infrequent fuzzy items from the data set, the compressed dataset is first set to Null. When the transaction items have been pruned and there are still more than zero items. The smallest support in the items is added if it is already present in the dataset; otherwise, a new transaction is established with the initial value being the smallest support in the compressed data set. The remaining items are then supported by the smallest support in each transaction. Table 6 shows the compressed dataset as a result.

Table 4: The remaining items satisfy the minimum support of 30%

Items	Support
$C.High$	4.6
$A.Middle$	3.8
$C.Middle$	3.4
$B.Low$	3
$D.Low$	2.6

Table 5: Database after removing infrequent fuzzy items from transactions

Tid	Items	Fsupport
1	$\left(\frac{0.6}{C.High}\right)\left(\frac{0.4}{A.Middle}\right)\left(\frac{0.4}{C.Middle}\right)\left(\frac{0.6}{D.Low}\right)$	0.4
2	$\left(\frac{0.4}{C.High}\right)\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.6}{C.Middle}\right)\left(\frac{0.8}{B.Low}\right)$	0.4
3	$\left(\frac{0.6}{C.High}\right)\left(\frac{0.4}{C.Middle}\right)\left(\frac{0.8}{B.Low}\right)$	0.4
4	$\left(\frac{0.8}{C.High}\right)\left(\frac{0.6}{A.Middle}\right)\left(\frac{0.2}{C.Middle}\right)\left(\frac{0.8}{D.Low}\right)$	0.2
5	$\left(\frac{0.2}{C.High}\right)\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.8}{C.Middle}\right)\left(\frac{0.8}{B.Low}\right)$	0.2
6	$\left(\frac{0.8}{C.High}\right)\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.2}{C.Middle}\right)\left(\frac{0.6}{D.Low}\right)$	0.2
7	$\left(\frac{0.6}{C.High}\right)\left(\frac{0.4}{A.Middle}\right)\left(\frac{0.4}{C.Middle}\right)\left(\frac{0.6}{B.Low}\right)$	0.4
8	$\left(\frac{0.6}{C.High}\right)\left(\frac{0.4}{C.Middle}\right)\left(\frac{0.6}{D.Low}\right)$	0.4

3.4. Mining frequent fuzzy k-itemsets using CLA

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 the compressed dataset and sends the frequent fuzzy 1-itemset to the cells as Figure 2.

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.

For example, the environment reads the first row which contains *C.High, A.Middle, C.Middle, D.Low*, and transfers them to all cells, they process on this fuzzy itemsets. *C.High* will recognize *A.Middle, C.Middle* and *D.Low* as its neighbors, respectively. It is created with a fuzzy support value of 0.8. Similar to *A.Middle, C.Middle,* and *D.Low* items. Because *B.Low* does not exist in the first transaction, it does not have action. The neighbors and proximity list of each cell are shown in Figure 3.

The cells are read with a support value of 1.0 and the second record of the compressed dataset, which contains the values $C.High$, $A.Middle$, $C.Middle$, and $B.Low$ is sent to them. For cell $C.High$, the neighbors' reviews of $A.Middle$, $C.Middle$, and $B.Low$ are taken into account. Item $B.Low$ was generated with a value of 1.0 because it wasn't already present in the neighborhoods. Existing items gain worth by being added to, increasing their value if they are existent. In this case, $A.Middle$ and $C.Middle$ already exist in the neighborhood of $C.High$, the value of both A.Middle and $C.Middle$ is 1.8. Also, Item $D.Low$ is already

Table 6: Compressed data

Tid	Items	Support
1	$(C.High)\,(A.Middle)\,(C.Middle)\,(D.Low)$	0.8
2	$(C.High)\,(A.Middle)\,(C.Middle)\,(B.Low)$	1
3	$(C.High)\,(C.Middle)\,(B.Low)$	0.4
8	$(C.High)\,(C.Middle)\,(D.Low)$	0.4

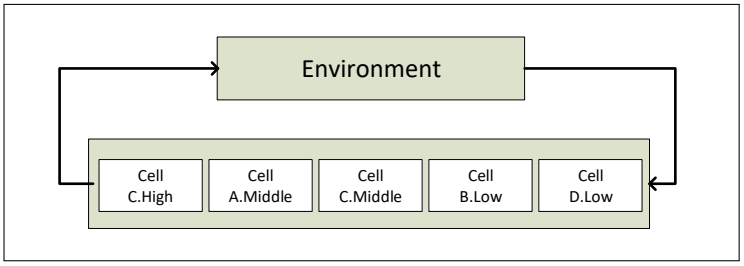

Figure 2: Automata cells according to frequent fuzzy 1-itemset

available in the neighborhood of $C.High$ but not in the second row of the dataset, it remains in the neighborhood of $C.High$ with the same value as 0.8. Additionally, other cells carry out these actions concurrently. After processing the second record of the dataset, the created neighborhoods and the proximity list of cells are displayed in Figure 4.

Similarly, Figures 5, 6 show the generated neighborhoods and the proximity list of cells after running the third row and the fourth row of the dataset.

Each cell automatically discards its neighbors and the proximities whose fuzzy support is lower than the minimum threshold in its proximity list when all the transactions in the compressed dataset are sent to the cells by the environment. A frequent fuzzy 1-itemset is obtained for each cell. All of the frequently used items might be recovered by combining the results of each cell. After pruning, the proximity list of cells is shown in Figure 7.

3.5. Describe the Algorithm CLA-Fuzzy mining

A. Data preprocessing (convert quantitative database to fuzzy database)

- Step 1: Implement the improved clustering algorithm EMC.
- Step 2: Implement fuzzy partitioning algorithm.
- Step 3: Perform database compression.

B. Compressed data and processing algorithm

- Step 4: Through the environment will read each record of compressed transaction data in records 1 and 2.
- Step 5: The environment generates CLA cells by item regularly and performs concurrent processing for those cells (only for records 1 and 2). This step to initializes all

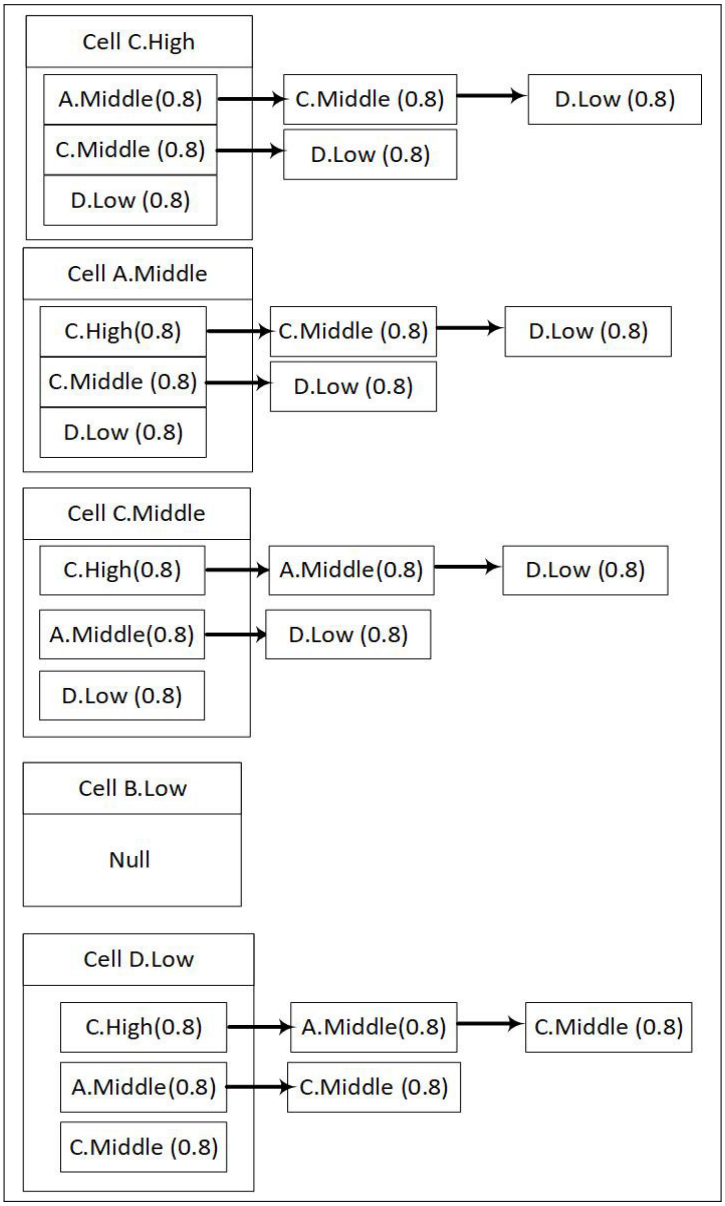

Figure 3: The proximity list of cells after the first row running

items in the transaction and these items will be created corresponding to each cell, these cell values can be null if not exist in the first transaction.

- Step 6: Browse all cells processed from step 5 and update the number of transactions again.

- Step 7: The algorithm ends with the result that the frequent itemsets are stored in the cells and will be mined for different purposes.

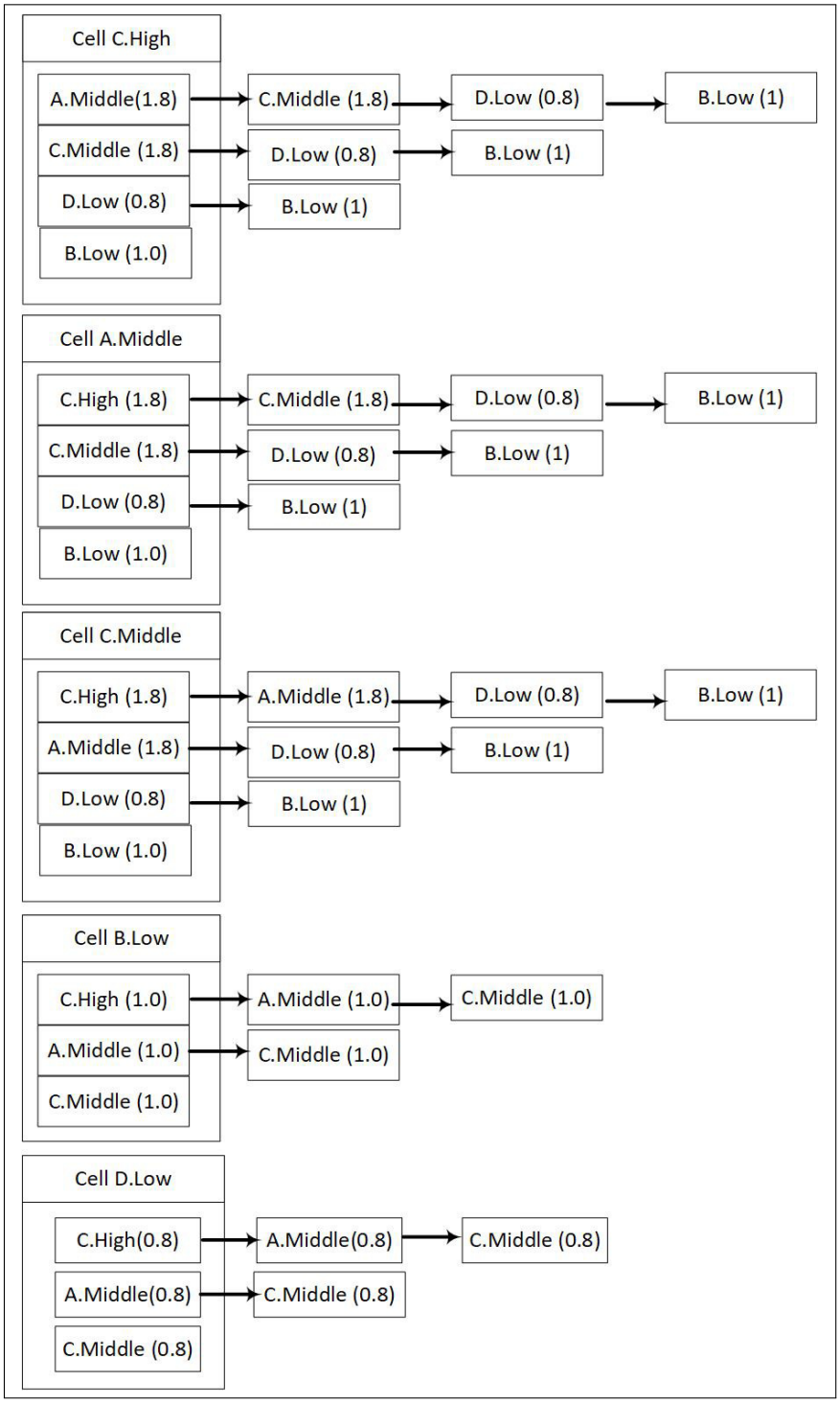

Figure 4: The proximity list of cells after the second row running

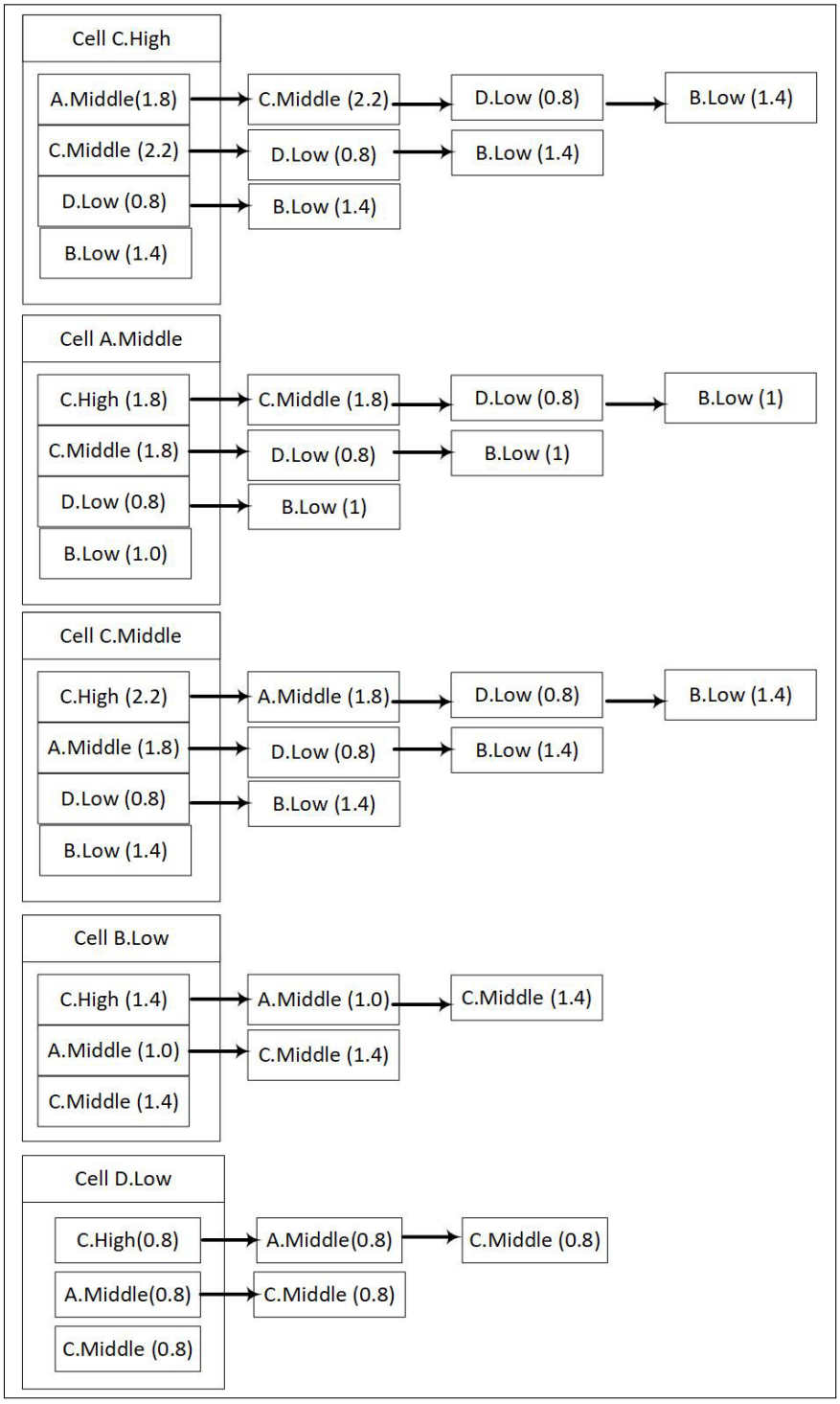

Figure 5: The proximity list of cells after the third row running

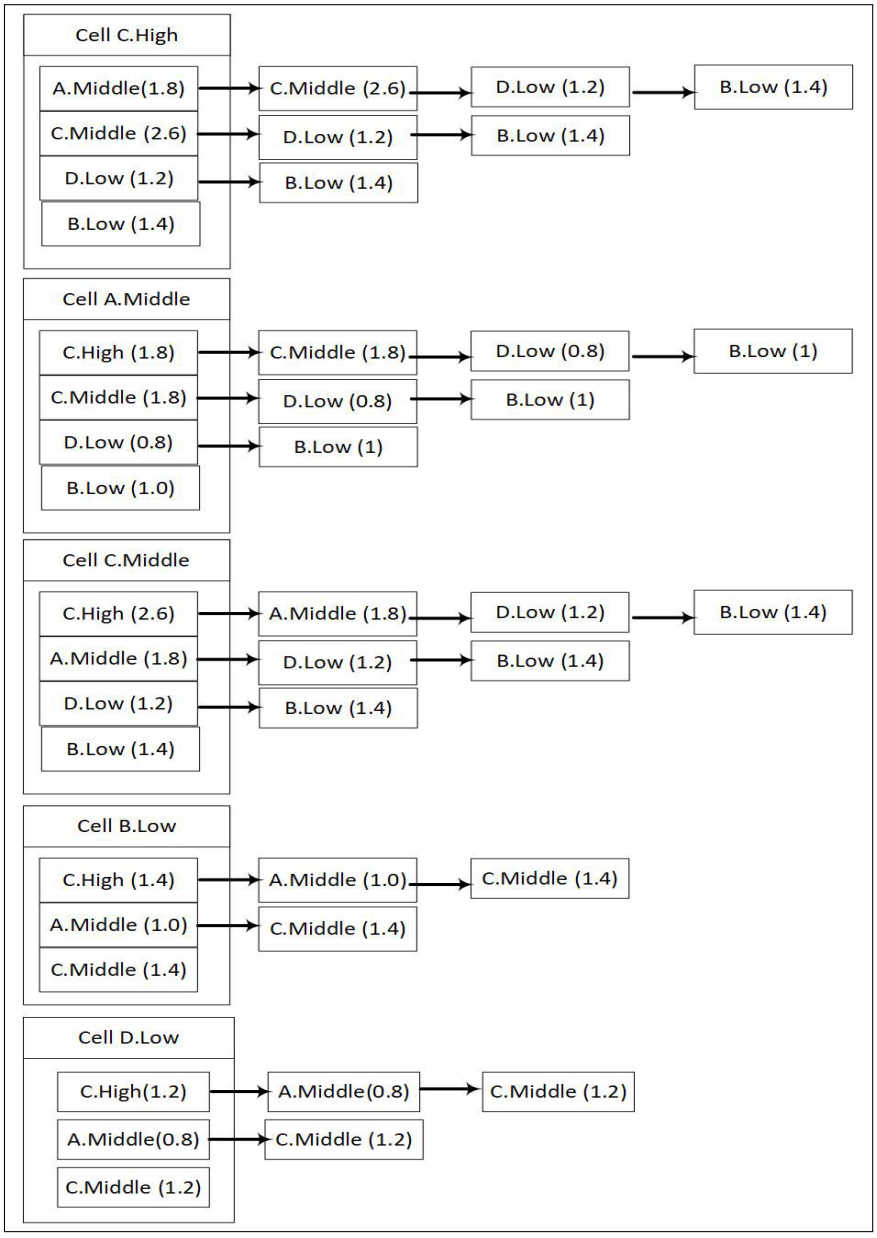

Figure 6: The proximity list of cells after the fourth row running

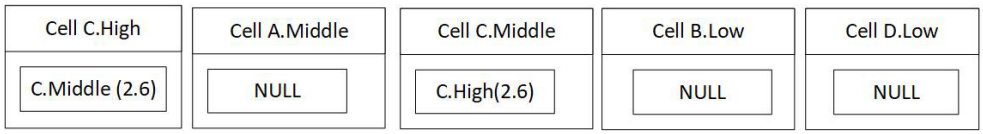

Figure 7: The proximity list after being pruned with minsup =30%

Algorithm 1 : Data_Preprocessing()

Input:
D_Q: Quantitative transaction database;
minsup: Minimum support threshold.
Output:
F_1: A set of frequent fuzzy 1-itemset;
D_f: Fuzzy dataset after removing infrequent;
Execute EMC();
Call FuzzyPartition();
Transform D_Q into D_f;
/* Scan D_f to compute the support of each fuzzy item A_{jk} in T_i as Eq.(1)*/
if fsup(A_{jk})$\geq$ minsup **then**
  Put A_{jk} in F_1
end if
if A_{jk} is not in F **then**
  Remove A_{jk} from all T_i $(i...n)$;
end if
return F_1, D_f

Algorithm 2 : Data_Compression()

Input:
minsup: Minimum support threshold;
F_1: A set of frequent fuzzy 1-itemset;
D_f: Fuzzy dataset after removing infrequent.
Output:
CDS: Compressed data set above table.
for $i = 1$ to D_f **do**
  for $j = 1$ to items **do**
    if items(i, j)==Items$(i + 1, j)$ **then**
      Remove(rows$(i + 1)$);
      Update;
      Support(rows(i)+rows$(i + 1)$);
    end if
  end for
end for
return CDS

Algorithm 3 : CLA_Fuzzy_Mining()

Input:
minsup: Minimum support threshold;
F_1: set of frequent fuzzy 1-itemset;
TDB: Compressed transaction database;
CDS: Compressed data set above table.

Output:
FFIL: Fuzzy Frequent Itemsets List
for $i = 1$ to CDS **do**
  Read transaction from CDS;
  Transfer transactions to parallel cells;
  Through new transactions update the list of neighborhoods and neighborhoods of cells;
end for
Initialize FFIL;
for $i = 1$ to automata cells **do**
  Execute the PruneNeighbors() function for cell[j];
  Execute DFS() function for cells[j];
  for all anItemset on cell[j].FrequentItemset **do**
    if anItemset does not exist in FFIL **then**
      FFIL.add (anItemset);
    else
      nothing;
    end if
  end for
end for
return (FFIL)

4. EXPERIMENTAL RESULTS

We use the Foodmart, Chess, and Chain store data sets from the frequent itemsets mining data sets [14] for this experiment. The dataset's description is displayed in Table 7.

This work introduces the experimental results from the algorithms and compares them to the outcomes of the NPSFF algorithm [37] and the iMFFP algorithm [26]. The CLA-Fuzzy Mining algorithm is more effective than the previous two algorithms in terms of processing time and temporary storage memory, according to testing results based on the dataset presented in Table 7. In this work, the suggested algorithm was tested in an integrated development environment, along with the earlier algorithms that were compared (IDE) on Windows 10 x64 machines with an Intel(R) Core(TM) i7-7500U CPU clocked at 2.70GHz, 2.90GHz, or 2.8GHz, and 16GB RAM, JDK8 (the Java Development Kit) and Java Object-Oriented Programming Language are supported.

Table 7: Testing dataset

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 8 shows that the performance of the proposed algorithm CLA-Fuzzy Mining compared with two algorithms iMFFP and NPSFF performed on all data sets is reduced. The application of the method eliminates redundant transactions to compress the data set and

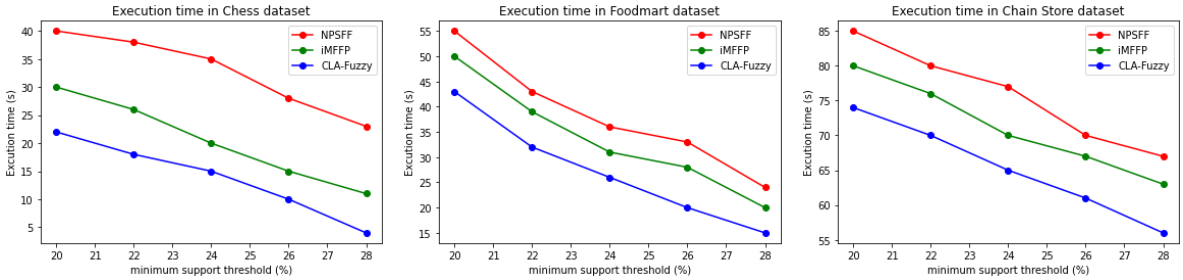

Figure 8: Execution time of algorithms for different data sets

combines with the parallel processing method for cells containing frequent fuzzy itemsets. Compared to other parallel processing methods, the CLA-Fuzzy Mining technique performs substantially better thanks to the environment's automatic updating of information for surrounding cells. On the other hand, the method's advancement is directly proportional to the volume of transactions produced after utilizing minsup to drop the dataset.

As per Figure 9, the memory usage of iMFFP and NPSFF is higher than compared to the CLA-Fuzzy. The procedure of mining fuzzy frequent itemsets based on proximity lists helps to limit the requested memory usage.

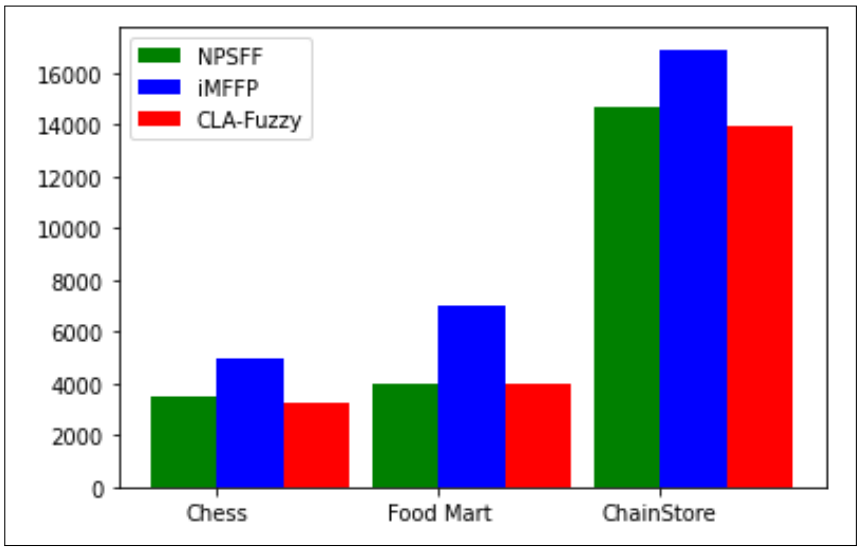

Figure 9: Evaluations of memory usage for different data sets

5. CONCLUSIONS

To mine frequent fuzzy itemsets in transactional databases in the future, many strategies and methodologies are presented in this work. The created neighbor list for all of each cell's neighbors is used in the described CLA-Fuzzy Mining-based frequent fuzzy set mining,

which subsequently employs this list to mine the frequent fuzzy itemsets. The presentation technique was evaluated against two previously created techniques (iMFFP, and NPSFF). According to the experimental findings, our mining method, which is based on CLA-Fuzzy Mining, outperforms iMFFP and NPSFF on various common data sets. To speed up the processing time of the method, many approaches are combined, such as fuzzy partitioning to convert quantitative to fuzzy values and data preprocessing using clustering.

REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, 1993, pp. 207–216.

[2] R. Agrawal and J. C. Shafer, "Parallel mining of association rules," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 962–969, 1996.

[3] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proceedings of the Eleventh International Conference on Data Engineering*. IEEE, 1995, pp. 3–14.

[4] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. very large Data Bases, VLDB*, vol. 1215. Citeseer, 1994, pp. 487–499.

[5] P. Arora, R. Chauhan, and A. Kush, "Frequent itemsets from multiple datasets with fuzzy data," *International Journal of Computer Theory and Engineering*, vol. 3, no. 2, p. 255, 2011.

[6] H. Beigy and M. R. Meybodi, "A mathematical framework for cellular learning automata," *Advances in Complex Systems*, vol. 7, no. 03n04, pp. 295–319, 2004.

[7] F. Berzal, J.-C. Cubero, N. Marın, and J.-M. Serrano, "Tbar: An efficient method for association rule mining in relational databases," *Data & Knowledge Engineering*, vol. 37, no. 1, pp. 47–64, 2001.

[8] J.-S. Chen, F.-G. Chen, and J.-Y. Wang, "Enhance the multi-level fuzzy association rules based on cumulative probability distribution approach," in *2012 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*. IEEE, 2012, pp. 89–94.

[9] M.-S. Chen, J. Han, and P. S. Yu, "Data mining: an overview from a database perspective," *IEEE Transactions on Knowledge and data Engineering*, vol. 8, no. 6, pp. 866–883, 1996.

[10] M. Esmaeilpour, V. Naderifar, and Z. Shukur, "Cellular learning automata for mining customer behaviour in shopping activity," *International Journal of Innovative Computing, Information, & Control*, vol. 8, no. 4, pp. 2491–2511, 2012.

[11] M. Esnaashari and M. Meybodi, "A cellular learning automata based clustering algorithm for wireless sensor networks," *Sensor Letters*, vol. 6, no. 5, pp. 723–735, 2008.

[12] M. Esnaashari and M. R. Meybodi, "Dynamic point coverage in wireless sensor networks: A learning automata approach," in *Computer Society of Iran Computer Conference*. Springer, 2008, pp. 758–762.

[13] M. Esnaashari and M. R. Meybodi, "Irregular cellular learning automata," *IEEE Transactions on Cybernetics*, vol. 45, no. 8, pp. 1622–1632, 2014.

[14] P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C.-W. Wu, and V. S. Tseng, "Spmf: a java open-source pattern mining library," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3389–3393, 2014.

[15] T.-P. Hong, C.-S. Kuo, and S.-L. Wang, "A fuzzy aprioritid mining algorithm with reduced computational time," *Applied Soft Computing*, vol. 5, no. 1, pp. 1–10, 2004.

[16] T.-P. Hong, Y.-C. Lee, and M.-T. Wu, "An effective parallel approach for genetic-fuzzy data mining," *Expert Systems with Applications*, vol. 41, no. 2, pp. 655–662, 2014.

[17] T.-P. Hong, C.-W. Lin, and T.-C. Lin, "The mffp-tree fuzzy mining algorithm to discover complete linguistic frequent itemsets," *Computational Intelligence*, vol. 30, no. 1, pp. 145–166, 2014.

[18] T.-P. Hong, C.-W. Lin, and Y.-L. Wu, "Incrementally fast updated frequent pattern trees," *Expert Systems with Applications*, vol. 34, no. 4, pp. 2424–2435, 2008.

[19] T.-P. Hong, K.-Y. Lin, and B.-C. Chien, "Mining fuzzy multiple-level association rules from quantitative data," *Applied Intelligence*, vol. 18, no. 1, pp. 79–90, 2003.

[20] T.-P. Hong, C.-H. Wu *et al.*, "An improved weighted clustering algorithm for determination of application nodes in heterogeneous sensor networks," 2011.

[21] K. Hu, Y. Lu, L. Zhou, and C. Shi, "Integrating classification and association rule mining: A concept lattice framework," in *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*. Springer, 1999, pp. 443–447.

[22] R. Jain and W. Stallings, "Comments on" fuzzy set theory versus bayesian statistics"," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, no. 4, pp. 332–333, 1978.

[23] C. Z. Janikow, "Fuzzy decision trees: issues and methods," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 1, pp. 1–14, 1998.

[24] M. M. D. Khomami, A. Rezvanian, and M. R. Meybodi, "A new cellular learning automata-based algorithm for community detection in complex social networks," *Journal of computational science*, vol. 24, pp. 413–426, 2018.

[25] B. Lent, A. Swami, and J. Widom, "Clustering association rules," in *Proceedings 13th International Conference on Data Engineering*. IEEE, 1997, pp. 220–231.

[26] C.-W. Lin, T.-P. Hong, Y.-F. Chen, T.-C. Lin, and S.-T. Pan, "An integrated mffp-tree algorithm for mining global fuzzy rules from distributed databases." *J. Univers. Comput. Sci.*, vol. 19, no. 4, pp. 521–538, 2013.

[27] C.-W. Lin, T.-P. Hong, and W.-H. Lu, "The pre-fufp algorithm for incremental mining," *Expert Systems with Applications*, vol. 36, no. 5, pp. 9498–9505, 2009.

[28] C.-W. Lin, T.-P. Hong, and W.-H. Lu, "Linguistic data mining with fuzzy fp-trees," *Expert Systems with Applications*, vol. 37, no. 6, pp. 4560–4567, 2010.

[29] F. Liu, Z. Lu, and S. Lu, "Mining association rules using clustering," *Intelligent Data Analysis*, vol. 5, no. 4, pp. 309–326, 2001.

[30] J. S. Park, M.-S. Chen, and P. S. Yu, "Using a hash-based method with transaction trimming for mining association rules," *IEEE Transactions on Knowledge and Data Engineering*, vol. 9, no. 5, pp. 813–825, 1997.

[31] K. S. Prabha and R. Lawrance, "Mining fuzzy frequent item set using compact frequent pattern (cfp) tree algorithm," *Data Mining and Knowledge Engineering*, vol. 4, no. 7, pp. 365–369, 2012.

[32] P. Pulkkinen and H. Koivisto, "A dynamically constrained multiobjective genetic fuzzy system for regression problems," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 161–177, 2009.

[33] M. Rezapoor Mirsaleh and M. R. Meybodi, "A new memetic algorithm based on cellular learning automata for solving the vertex coloring problem," *Memetic Computing*, vol. 8, no. 3, pp. 211–222, 2016.

[34] R. Senge and E. Hüllermeier, "Top-down induction of fuzzy pattern trees," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 2, pp. 241–252, 2010.

[35] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *International Conference on Extending Database Technology*. Springer, 1996, pp. 1–17.

[36] Y. G. Sucahyo and R. P. Gopalan, "Building a more accurate classifier based on strong frequent patterns," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2004, pp. 1036–1042.

[37] T. T. Tran, T. N. Nguyen, T. T. Nguyen, G. L. Nguyen, and C. N. Truong, "A fuzzy association rules mining algorithm with fuzzy partitioning optimization for intelligent decision systems," *International Journal of Fuzzy Systems*, pp. 1–14, 2022.

[38] X.-Z. Wang, L.-C. Dong, and J.-H. Yan, "Maximum ambiguity-based sample selection in fuzzy decision tree induction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 8, pp. 1491–1505, 2011.

[39] M. Woźniak and B. Krawczyk, "Combined classifier based on feature space partitioning," *International Journal of Applied Mathematics and Computer Science*, vol. 22, no. 4, pp. 855–866, 2012.

[40] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[41] M. F. Zaman and H. Hirose, "Classification performance of bagging and boosting type ensemble methods with small training sets," *New Generation Computing*, vol. 29, no. 3, pp. 277–292, 2011.

Received on August 26, 2022
Revised on December 01, 2022