Int. J. Fuzzy Syst.
DOI 10.1007/s40815-016-0246-1

Efficient Mining of Multiple Fuzzy Frequent Itemsets

Jerry Chun-Wei Lin[1] · Ting Li[1] · Philippe Fournier-Viger[2] · Tzung-Pei Hong[3,4] · Jimmy Ming-Tai Wu[5] · Justin Zhan[5]

Received: 21 December 2015 / Revised: 9 August 2016 / Accepted: 24 August 2016

Abstract Traditional association-rule mining or frequent itemset mining only can handle binary databases, in which each item or attribute is represented as either 0 or 1. Several algorithms were developed extensively to discover fuzzy frequent itemsets by adopting the fuzzy set theory to the quantitative databases. Most of them considered the maximum scalar cardinality to find, at most, one represented item from the transformed linguistic terms. This paper presents an MFFI-Miner algorithm to mine the complete set of multiple fuzzy frequent itemsets (MFFIs) without candidate generation. An efficient fuzzy-list structure was designed to keep the essential information for mining process, which can greatly reduce the computation of a database scan. Two efficient pruning strategies are developed to reduce the search space, thus speeding up the mining process to discover MFFIs directly. Substantial experiments were conducted to compare the performance of the proposed algorithm to the state-of-the-art approaches in terms of execution time, memory usage, and node analysis.

Keywords Fuzzy frequent itemsets · MFFI-Miner · Multiple regions · Fuzzy-list structure

✉ Jerry Chun-Wei Lin
jerrylin@ieee.org

Ting Li
tingli@ikelab.net

Philippe Fournier-Viger
philfv@hitsz.edu.cn

Tzung-Pei Hong
tphong@nuk.edu.tw

Jimmy Ming-Tai Wu
wmt@wmt35.idv.tw

Justin Zhan
justin.zhan@unlv.edu

1 School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China

2 School of Natural Sciences and Humanities, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China

3 Department of Computer Science and Engineering, National University of Kaohsiung, Kaohsiung, Taiwan

4 Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan

5 Department of Computer Science, University of Nevada, Las Vegas, NV, USA

1 Introduction

Depending on the various requirements, mined knowledge generally can be classified as association rules (ARs) [1, 2, 8, 12], sequential patterns (SPs) [3, 16], classification [25], clustering [5], or high-utility itemsets (HUIs) [7, 20], among other designations. Agrawal et al. [2] first presented the Apriori algorithm to mine ARs in the level-wise manner. It requires time-consuming computations to find the FIs level by level, and the numerous candidate itemsets have to be generated. Therefore, Han et al. [10] designed the frequent pattern (FP)-tree structure with a FP-growth mining algorithm to find FIs without candidate generation. Based on the FP-tree structure and the FP-growth mining algorithm, the FIs can be discovered efficiently.

In real-life situations, quantitative databases can be used to provide more information for decision making than that of the traditional binary databases. However, it is difficult to handle quantitative databases based on crisp sets. Fuzzy set theory [26], which was proposed to handle quantitative databases, is based on pre-defined membership functions to transform the quantitative values in the transactions into the representation of linguistic terms. Hong et al. [11] suggested a mining approach using fuzzy data to discover fuzzy frequent itemsets (FFIs) in a level-wise manner. This approach uses the maximum cardinality mechanism to generate-and-test the candidate itemsets level by level. The maximum cardinality mechanism can greatly reduce the computations used to discover FFIs; however, some information may be lost. At this point, Hong et al. [13] designed a gradual data-reduction approach (GDF) for mining multiple fuzzy frequent itemsets (MFFIs). Lin et al. then presented several algorithms [14, 22, 23] for mining MFFIs based on their designed tree structures. Although the tree-based algorithms are better than that of the Apriori-like approach, the amounts of computation are still required for mining MFFIs.

In this paper, a fuzzy-list structure with the designed multiple fuzzy frequent itemset (MFFI)-Miner algorithm is presented to mine MFFIs. The designed MFFI-Miner algorithm efficiently mines MFFIs without candidate generation based on the designed fuzzy-list structure. This approach can be used to reduce the computations of the generate-and-test mechanism in a level-wise manner. Two efficient pruning strategies are also designed to reduce the search space of the enumeration tree based on the fuzzy-list structure. Thus, the computations for mining MFFIs can be greatly reduced. Experiments showed that the proposed approaches have better mining performance than that of the state-of-the-art Apriori-based and pattern-growth algorithms.

2 Related Work

It is efficient to adopt fuzzy set theory [26] for mining fuzzy association rules (FARs) from quantitative databases since the information discovered can be represented in linguistic terms using a natural language. In the past, Chan et al. [6] proposed the F-APACS algorithm to discover FARs by transforming the quantitative values of attributes into linguistic representations. Kuok et al. [15] designed an efficient method to discover the FARs from the numerical databases. Hong et al. [11] developed the FDTA algorithm to mine FARs from quantitative databases. Lin et al. [18] designed a fuzzy frequent pattern (FFP)-tree structure to mine FFIs from quantitative databases. For the FFP-tree structure, the same FFIs in different transactions may be put into different tree paths since the sorted order of the same FFIs in different transactions may not be the same. Thus, the FFP-tree structure generates loose, and huge, tree nodes. To solve these limitations, Lin et al. then developed the compressed fuzzy frequent pattern (CFFP)-tree structure [19] and the upper-bound fuzzy frequent pattern (UBFFP)-tree structure [21] for deriving FFIs. Compared to the FFP-tree structure, the global sorting strategy was adopted in the CFFP-tree structure, thus reducing the number of tree nodes. Although the node numbers in the CFFP-tree structure can be significantly reduced, each node requires an extra array to keep the fuzzy value of the current nodes with any of its prefix nodes in the path. The UBFFP-tree algorithm [21] uses the same sorting strategy as the CFFP-tree structure, but derives the candidates of the FFIs with upper-bound fuzzy values. An additional database scan is required to mine the actual FFIs from the remaining candidates.

In order to reveal more complete information regarding FFIs, Hong et al. [14] designed an multiple fuzzy frequent pattern (MFFP)-tree algorithm to mine MFFIs from the tree structure. In addition, an MFFP-growth mining approach was proposed to mine fuzzy frequent itemsets from the tree structure. Based on similar structures of the CFFP-tree and UBFFP-tree algorithms, Lin et al., respectively, designed CMFFP-tree [22] and UBMFFP-tree [23] algorithms to speed up the mining process of the MFFIs. Hong et al. [13] presented an efficient GDF algorithm to merge the same fuzzy sets of the transformed transactions into smaller transformed databases, thus speeding up the computations for mining FARs level by level. However, a multiple database scan was required to mine the multiple fuzzy frequent itemsets (MFFIs) in a level-wise manner. Other algorithms of fuzzy data mining are still developed in progress [4, 17, 24].

3 Preliminaries and Problem Definition

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a finite set of m distinct items (attributes) in a quantitative database $D = \{T_1, T_2, \ldots, T_n\}$, in which each transaction $T_q \in D$ and (1) is a subset of I; (2) contains several items with its purchase quantities v_{iq}; (3) has an unique identifier, called *TID*. An itemset X is a set of k distinct items $\{i_1, i_2, \ldots, i_k\}$, where k is the length of an itemset called k-itemset. X is said to be contained in a transaction T_q if $X \subseteq T_q$. A minimum support threshold is defined as δ. The user-specified membership functions are set as μ.

A quantitative database is used as an example shown in the second column of Table 1. The minimum support threshold is set initially as $\delta(= 25\%)$. The membership functions used in this example are shown in Fig. 1.

Definition 1 The linguistic variable R_i is an attribute of a quantitative database whose value is the set of fuzzy linguistic terms represented in natural language as (R_{i1}, R_{i2}, ..., R_{ih}); this variable can be defined in the membership functions μ.

Definition 2 The quantitative value of i denoted as v_{iq}, is the quantitative of the item i in transaction T_q.

Definition 3 The fuzzy set, denoted as f_{iq}, is the set of fuzzy linguistic terms with their membership degrees

Table 1 An example of a quantitative database and transformed results

TID	Items and their quantities	Transformed linguistic terms
1	(C:3), (D:2), (E:1)	$\frac{0.67}{C.M}+\frac{0.5}{C.H}, \frac{0.5}{D.L}+\frac{0.67}{D.M}, \frac{1.0}{E.L}$
2	(B:1), (C:2), (D:1)	$\frac{1.0}{B.L}, \frac{0.5}{C.L}+\frac{0.67}{C.M}, \frac{1.0}{D.L}$
3	(B:3), (C:3), (E:1)	$\frac{0.67}{B.M}+\frac{0.5}{B.H}, \frac{0.67}{C.M}+\frac{0.5}{C.H}, \frac{1.0}{E.L}$
4	(A:3), (C:5), (D:3)	$\frac{0.67}{A.M}+\frac{0.5}{A.H}, \frac{1.0}{C.H}, \frac{0.67}{D.M}+\frac{0.5}{D.H}$
5	(A:1), (B:1), (C:2), (D:1)	$\frac{1.0}{A.L}, \frac{1.0}{B.L}, \frac{0.5}{C.L}+\frac{0.67}{C.M}, \frac{1.0}{D.L}$
6	(B:1), (D:1), (E:2)	$\frac{1.0}{B.L}, \frac{1.0}{D.L}, \frac{0.5}{E.L}+\frac{0.67}{E.M}$
7	(A:4), (B:3), (D:5), (E:3)	$\frac{1.0}{A.H}, \frac{0.67}{B.M}+\frac{0.5}{B.H}, \frac{1.0}{D.H}, \frac{0.67}{E.M}+\frac{0.5}{E.H}$
8	(B:1), (C:2), (D:1)	$\frac{1.0}{B.L}, \frac{0.5}{C.L}+\frac{0.67}{C.M}, \frac{1.0}{D.L}$

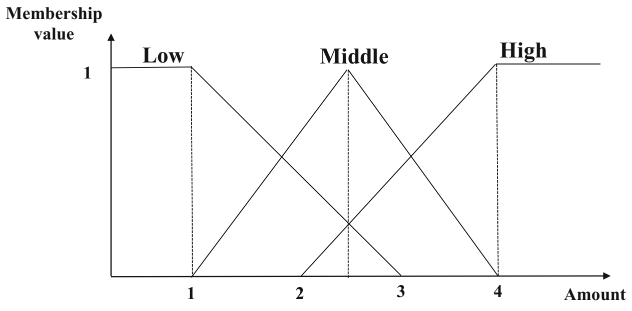

Fig. 1 The used membership functions of fuzzy linguistic 3-terms

(fuzzy values) transformed from the quantitative value v_{iq} of the linguistic variable i by the membership functions μ as:

$$f_{iq}=\mu_i(v_{iq})\left(=\frac{fv_{iq1}}{R_{i1}}+\frac{fv_{iq2}}{R_{i2}}+\cdots+\frac{fv_{iqh}}{R_{ih}}\right), \quad (1)$$

where h is the number of fuzzy linguistic terms of i transformed by μ, R_{il} is the l-th fuzzy linguistic terms of i, fv_{iql} is the membership degree (fuzzy value) of v_{iq} of i in the l-th fuzzy linguistic terms R_{il} and $fv_{iql} \subseteq [0,1]$.

Thus, the second column of Table 1 is then transformed by the membership functions shown in Fig. 1, and the transformed results are shown in the third column of Table 1.

Definition 4 The transformed fuzzy linguistic term R_{il} is represented and denoted as the fuzzy itemset in the field of fuzzy data mining.

Definition 5 The support of the fuzzy itemset, denoted $supp(R_{il})$, is the sum of its transformed fuzzy values, which can be defined as:

$$\mathrm{supp}(R_{il})=\sum_{R_{il}\subseteq T_q \wedge T_q \in D'} fv_{iql}, \quad (2)$$

where D' is the quantitative database D transformed by membership functions ($=\mu$), and the size of D' is the same as the original D.

Definition 6 The support of fuzzy k-itemset ($k\geq 2$), denoted as $supp(X)$, is the sum of the minimum fuzzy values among k items of X, which can be defined as:

$$\mathrm{supp}(X)=\{X\in R_{il}|\sum_{X\subseteq T_q \wedge T_q\in D'}\min(fv_{aql},fv_{bql}), a,b\in X, a\notin b\} \quad (3)$$

Problem Statement: The problem of multiple fuzzy frequent itemset mining (MFFIM) in this paper is to not only speed up the mining process of the required information but also discover the complete set of MFFIs, in which the support count of each fuzzy itemset X is no less than the pre-defined minimum support count as:

$$\mathrm{MFFIs} \leftarrow \{X|\mathrm{supp}(X)\geq \delta \times |D|\}, \quad (4)$$

where δ is the minimum support threshold, and $|D|$ is the database size.

4 Proposed MFFI-Miner Algorithm

In the past, most algorithms handle the FFIM using the maximum scalar cardinality mechanism to derive the FFIs [11, 18, 19, 21]. However, if only the linguistic term with maximum cardinality is used to represent each item (attribute), some useful FFIs may be missed and the set of discovered FFIs may be incomplete. This paper presents an efficient fuzzy-list structure to compress the databases and keep the necessary information for a later mining process. Two efficient strategies were also developed to prune the search space for discovering the MFFIs. Details are, respectively, described below.

4.1 Transformation Phase

For the quantitative database, the pre-defined membership functions are used to convert the quantitative value of each linguistic variable (item) into several fuzzy linguistic terms (fuzzy itemsets) with their membership degrees (fuzzy values). The fuzzy values of the same fuzzy itemset are summed up together as the support value of the fuzzy itemset. In this phase, if the fuzzy itemset with its support count is no less than the minimum support count, it is considered as the FFI and kept in the transformed databases.

Strategy 1 *For the remaining fuzzy itemsets with their fuzzy values in a transaction T_q, the fuzzy itemsets are sorted in their support-ascending order to perform the intersection operation for a later construction and mining process of fuzzy k-itemset $(k \geq 2)$.*

4.2 Fuzzy-List Construction Phase

After the original database is transformed and revised, the remaining fuzzy itemsets in L_1 are used to build their own fuzzy-list structure for keeping the necessary information. The definitions used in the fuzzy-list structure are given below.

Definition 7 A fuzzy itemset R_{il} in transaction T_q, and $R_{il} \subseteq T_q$. The set of fuzzy itemsets after R_{il} in T_q is denoted as T_q/R_{il}.

Definition 8 The fuzzy value of R_{il} in transaction T_q is defined as the internal fuzzy value and denoted as $if(R_{il}, T_q)$.

Definition 9 The resting fuzzy value except R_{il} in transaction T_q is denoted as $rf(R_{il}, T_q)$. This value is calculated by performing the maximum operation to get the maximum fuzzy value among the resting fuzzy itemsets in the transaction, which can be defined as:

$$rf(R_{il}, T_q) = \max\{if(z, T_q) | z \in (T_q/R_{il})\}. \tag{5}$$

Based on this definition, the upper-bound value after the processed fuzzy itemset R_{il} can be thus obtained. This value can be used to maintain the downward closure property for later combinational procedure, and the **correctness** and **completeness** of the combinations can be maintained. From the given example, the results of the constructed fuzzy-list structure for all fuzzy frequent 1-itemsets in L_1 are shown in Fig. 2.

For example in Fig. 2, the element (1, 0.5, 0.67) in the fuzzy-list of (*C.H*) shows that 1 represents the transaction T_1; 0.5 represents the internal fuzzy value is 0.5; 0.67 represents the resting fuzzy value after (*C.H*) is 0.67. For the fuzzy-list structures of fuzzy k -itemsets $(k \geq 2)$, it is unnecessary to scan the revised database but only perform the intersection operation of the fuzzy-list by the *tids* in k fuzzy-list structures. The fuzzy-list structures are performed in 2-way comparisons to quickly find the combined fuzzy-list structures. The construction algorithm of fuzzy-list structure is shown in Algorithm 1.

```
Algorithm 1: Fuzzy-list Construction
Input: $P_x.FL$, the fuzzy-list of $P_x$; $P_y.FL$, the fuzzy-list of $P_y$.
Output: $P_{xy}.FL$, the fuzzy-list of $x$ and $y$.
if $P_x$ and $P_y$ belong to the same item then
    return null.
$P_{xy}.FL \leftarrow null$;
for each $E_x \in P_x.FL$ do
    if $\exists E_y \in P_y.FL$ and $E_x.tid == E_y.tid$ then
        $E_{xy}.tid \leftarrow E_x.tid$;
        $E_{xy}.if \leftarrow min(E_x.if, E_y.if)$;
        $E_{xy}.rf \leftarrow E_y.rf$;
        $E_{xy} \leftarrow< E_{xy}.tid, E_{xy}.if, E_{xy}.rf >$;
        append $E_{xy}$ to $P_{xy}.FL$.
return $P_{xy}.FL$.
```

In the designed fuzzy-list structure, the *SUM.R_{il}.if* can be represented as the actual fuzzy value of the processed fuzzy itemset (R_{il}) and defined as follows.

Definition 10 The SUM.R_{il}.*if* is to sum up the fuzzy values of R_{il} in D', which can be defined as:

$$\text{SUM}.R_{il}.if = \sum_{R_{il} \subseteq T_q \wedge T_q \in D'} if(R_{il}, T_q). \tag{6}$$

In the designed fuzzy-list structure, this value is to estimate the upper-bound value to maintain the downward closure property for later combination based on the level-wise manner and defined as follows.

Definition 11 The SUM.R_{il}.*rf* is to sum up the resting fuzzy values after R_{il} in D', which can be defined as:

$$\text{SUM}.R_{il}.rf = \sum_{R_{il} \subseteq T_q \wedge T_q \in D'} rf(R_{il}, T_q). \tag{7}$$

C.H		
1	0.5	0.67
3	0.5	0.67
4	1	0

E.L		
1	1	0.67
3	1	0.67
6	0.5	1

C.M		
1	0.67	0.5
2	0.67	1
3	0.67	0
5	0.67	1
7	0.67	1

B.L		
2	1	1
5	1	1
6	1	1
7	1	1

D.L		
1	0.5	0
2	1	0
5	1	0
6	1	0
7	1	0

tid *if* *rf*

Fig. 2 The initial constructed fuzzy-list structures

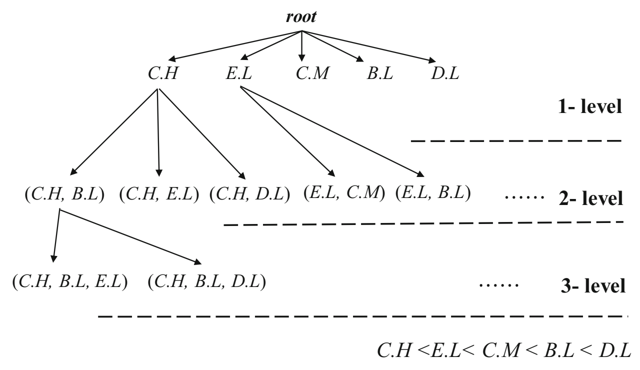

Fig. 3 An enumeration tree of the used example

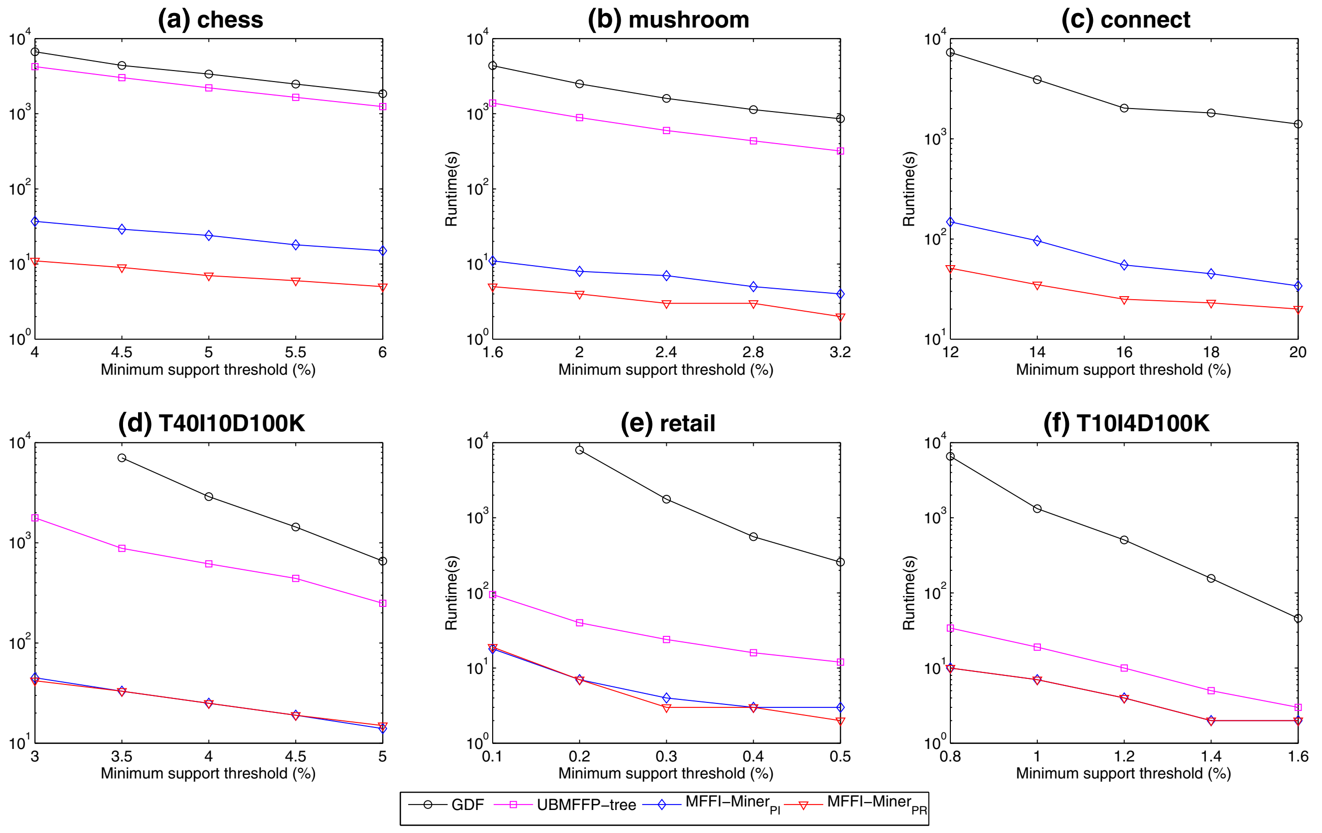

Fig. 4 Runtime with fuzzy linguistic 3-terms

4.3 Search Space of Enumeration Tree

Based on the designed fuzzy-list structure, the search space of the proposed MFFI-Miner algorithm can be represented as an enumeration tree, according to the developed support-ascending order strategy. The depth-first search strategy is used to traverse the enumeration tree and decide whether the child (superset) nodes are required to be generated and explored. The search space of the enumeration tree is shown in Fig. 3.

Strategy 2 (Pruning by Internal value, PI) *For an itemset X, if SUM.X.if is less than the minimum support count, it is not considered as a FFI and the supersets of X are unnecessary generated and explored in the enumeration tree.*

Theorem 1 *Given the fuzzy-list of a fuzzy itemset R_{il}. If the summed-up value of the R_{il} is less than minimum support count $(\delta \times |D|)$, any extensions of R_{il} are not the FFIs and can be discarded in the search space.*

Strategy 3 (Pruning by Resting value, PR) *For a fuzzy itemset X, if SUM.X.rf of X is less than the minimum support count, the supersets of X are unnecessary generated and explored in the enumeration tree.*

Theorem 2 *Given the fuzzy-list of a fuzzy itemset R_{il}. If the summed-up value of the resting fuzzy values for R_{il} is less than minimum support count $(\delta \times |D|)$, any extensions of R_{il} are not the FFIs (fuzzy frequent itemsets) and can be discarded in the search space.*

Thus, if the summation of the internal fuzzy values or resting fuzzy values of the fuzzy itemset *X* is less than the minimum support count, any extensions of *X* will not be the FFIs and can be directly ignored to avoid the construction phase for its fuzzy-list structure. Thus, the search space can be reduced and pruned in the enumeration tree. Details of the proposed multiple fuzzy frequent itemset (MFFI)-Miner algorithm is described in Algorithm 2.

```
Algorithm 2: MFFI-Miner
   Input: $FLs$, fuzzy-list of 1-itemsets; $\delta$.
   Output: $MFFIs$, the set of multiple fuzzy frequent itemsets.
 1 for each fuzzy-list $X$ in $FLs$ do
 2     if $SUM.X.if \geq \delta \times |D|$ then
 3         $MFFIs \leftarrow X \cup MFFIs$.
 4     if $SUM.X.rf \geq \delta \times |D|$ and $SUM.X.if \geq \delta \times |D|$ then
 5         $exFLs \leftarrow null$;
 6         for each fuzzy-list $Y$ after $X$ in $FLs$ do
 7             $exFL \leftarrow exFLs + Construct(X, Y)$;
 8     $MFFI\text{-}Miner(exFLs, \delta)$;
 9 return $MFFIs$.
```

5 Experimental Evaluation

This section describes how the MFFI-Miner algorithm adopts the PI pruning strategy known as MFFI-Miner$_{PI}$ and how the MFFI-Miner algorithm adopts both PI and PR pruning strategies, known as MFFI-Miner$_{PR}$ in the experiments. The state-of-the-art GDF algorithm [13] and the UBMFFP-tree [23] algorithm are used for comparison against the designed approaches in terms of runtime, memory usage, and node analysis. The compared algorithms in the experiments were implemented in Java. Four real-life chess [9], mushroom [9], connect [9], and retail datasets [9] as well as two synthetic T10I10D100k [9], T10I4D100k datasets [9] were used in the experiments to evaluate the performance of the designed approaches. The quantities of items were randomly assigned in the range of [1, 5] intervals in the datasets. The quantitative datasets first were transformed into several fuzzy linguistic terms based on predefined membership functions shown in Fig. 1. During the experiments for mining MFFIs, the algorithm was terminated if the execution time exceeded 10,000 s. The results are shown and analyzed as follows.

5.1 Runtime

This section compares the runtime of GDF and UBMFFP-tree algorithms to the designed MFFI-Miner$_{PI}$ and MFFI-Miner$_{PR}$ algorithms with respect to variants of minimum support thresholds under fuzzy linguistic 3-terms. Results are shown in Fig. 4.

It can be observed that the proposed approaches outperformed the GDF and UBMFFP-tree algorithms. The proposed MFFI-Miner$_{PI}$ and MFFI-Miner$_{PR}$ algorithms were almost two or three orders of magnitude faster than the GDF or UBMFFP-tree algorithms. When the minimum support threshold was set lower, the runtime of GDF algorithm had no results, as shown in Fig. 4d, e and the UBMFFP-tree algorithm has no results in Fig. 4c. The reason is that when the minimum support threshold is set lower, the GDF algorithm and UBMFFP-tree algorithm require more than 10,000 sec to generate-and-test candidates in a level-wise manner when mining MFFIs. Due to the advantages of the designed fuzzy-list structure, the transformed dataset can be compressed into a dense structure to retain necessary information for mining MFFIs. In addition, two pruning strategies that were developed could be used to greatly reduce the search space for discovering MFFIs. The MFFI-Miner$_{PR}$ always performed better than MFFI-Miner$_{PI}$ in dense datasets, as seen in Fig. 4a–c. Overall, the proposed MFFI-Miner$_{PI}$ and MFFI-Miner$_{PR}$ approaches are acceptable and both of them outperform the state-of-the-art algorithms for mining MFFIs.

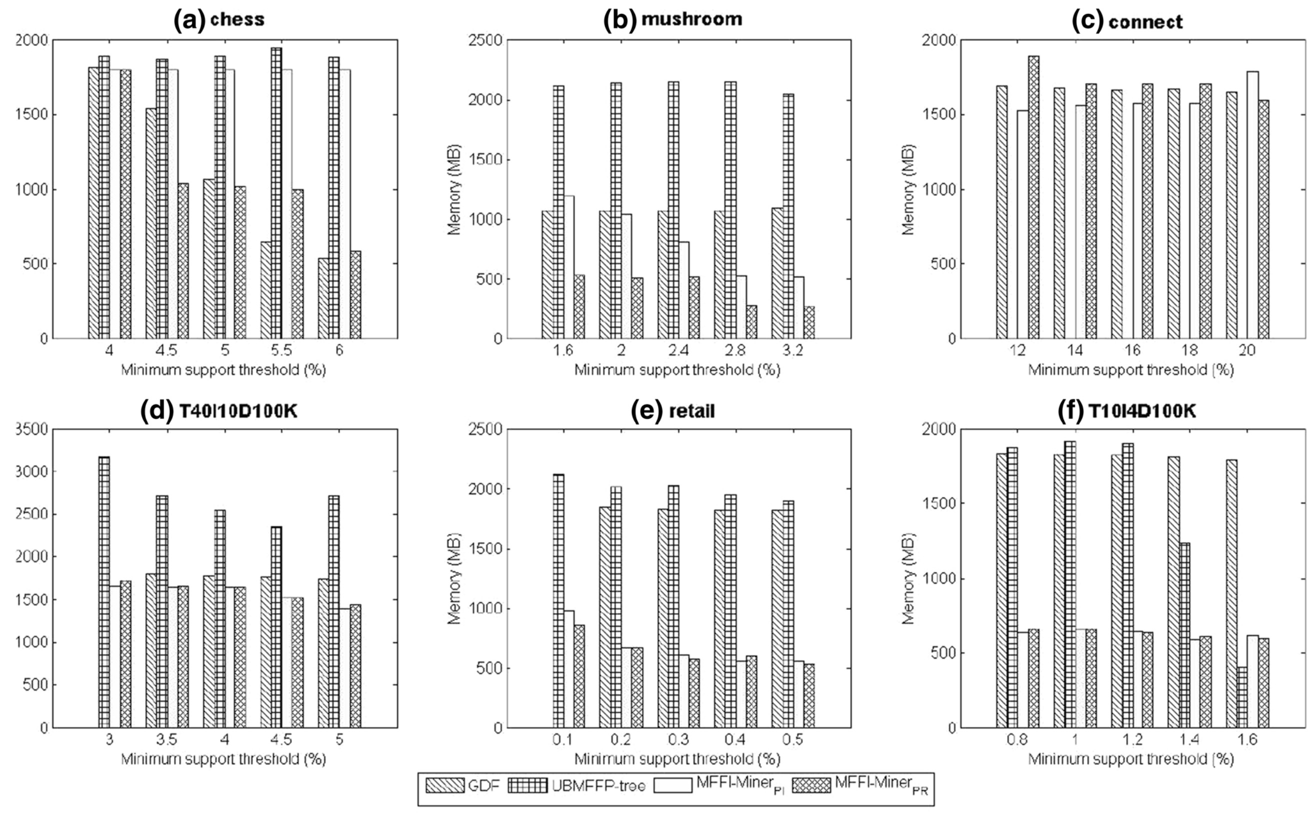

Fig. 5 Memory usage with fuzzy linguistic 3-terms

5.2 Memory Usage

Experiments were carried out to assess the memory usage of the compared algorithms. Results are shown in Fig. 5.

From Fig. 5, it can be observed that in most datasets except for the chess dataset, the proposed MFFI-Miner$_{PI}$ and MFFI-Miner$_{PR}$ algorithms required less memory than the generate-and-test GDF algorithm and always better than the pattern-growth UBMFFP-tree algorithm. The reason is that the designed fuzzy-list structure could be used to compress the dataset as a dense structure. The designed pruning strategies also could be used to reduce the generation of candidates during the mining process, while the UBMFFP-tree algorithm has to store massive candidates and the amounts of memory is required to rescan database for deriving the MFFIs. Even so, the GDF algorithm suffered when the generate-and-test approach was used to mine the MFFIs in a level-wise manner.

When the dataset is very dense, the fuzzy-list structure became larger and more memory usage was required to retain more fuzzy frequent itemsets; this can be observed in Fig. 5a. From the results shown in Fig. 5, only a little more memory was required for the MFFI-Miner$_{PI}$ algorithm. Since the MFFI-Miner$_{PR}$ algorithm adopted a more efficient pruning strategy to reduce the search space in the enumeration tree, nearly the same or less memory was required for the second MFFI-Miner$_{PR}$ algorithm. In summary, the memory usage of the designed approaches had better performance than the GDF algorithm and the UBMFFP-tree algorithm.

5.3 Node Analysis

This section evaluates the performance of the compared algorithms in terms of node numbers of the proposed approaches, which are the numbers of candidate itemsets of the GDF algorithm and the UBMFFP-tree algorithm. The GDF algorithm needed to generate the number of candidate itemsets in a level-wise manner in order to find the actual MFFIs, while the upper-bound strategy used in the UBMFFP-tree algorithm needs to generate more candidates for mining the MFFIs. However, the designed algorithm did not need to generate the numbers of candidates, but rather had to traverse the nodes in the enumeration tree. The results are shown in Fig. 6.

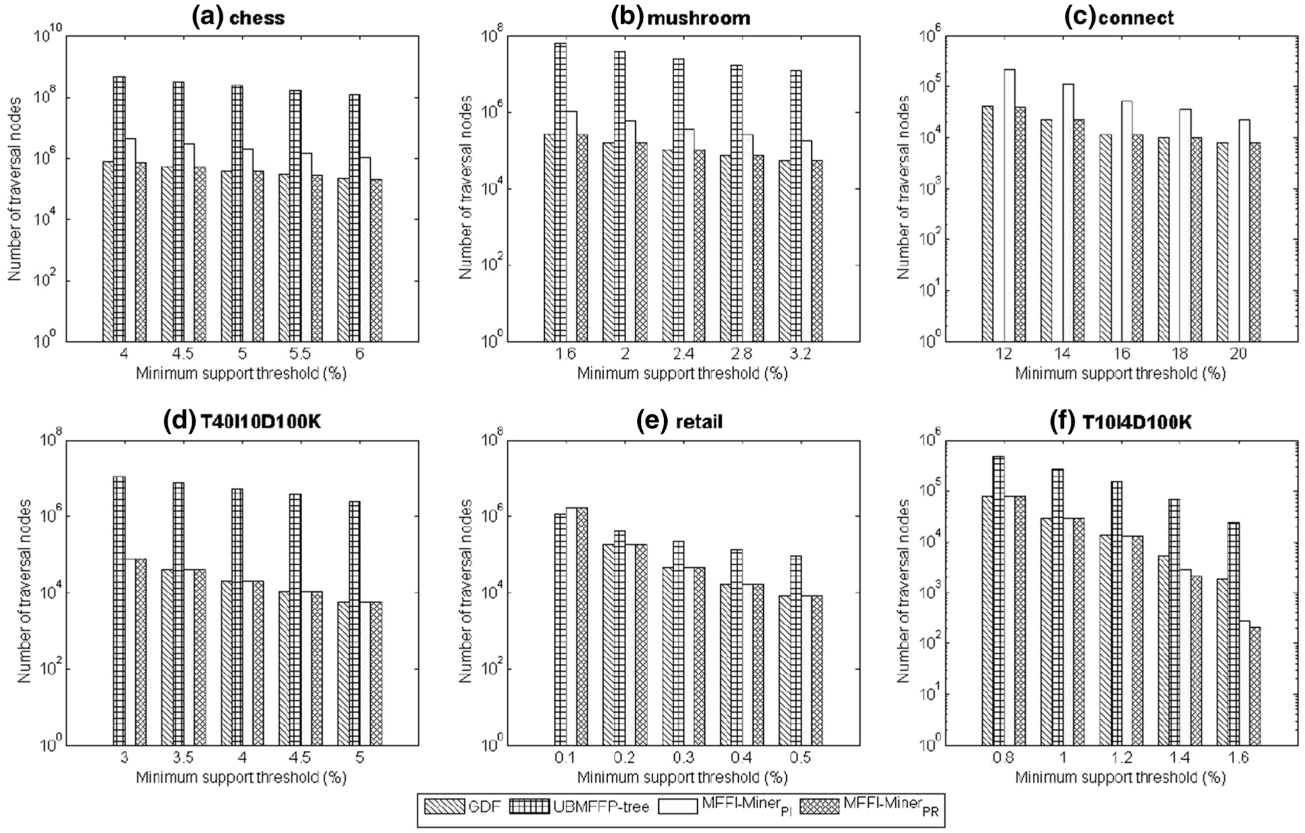

Fig. 6 Node analysis with respect to fuzzy linguistic 3-terms

From Fig. 6, the UBMFFP-tree algorithm had the largest number of determining nodes in most databases. The reason is that UBMFFP-tree algorithm adopted the upper-bound strategy to maintain the downward closure property, which is the over-estimated value for mining MFFIs. Moreover, the MFFI-Miner$_{PI}$ algorithm had the more number of determining nodes than the MFFI-Miner$_{PR}$ algorithm in the dense databases. The reason is that MFFI-Miner$_{PI}$ adopted only the PI strategy to reduce the search space, which was efficient enough to reduce the number of determining nodes in dense databases. In addition, the GDF algorithm and MFFI-Miner$_{PR}$ algorithm had almost the same number of determining nodes in spare datasets for fuzzy linguistic 3-terms; sometimes, the MFFI-Miner$_{PR}$ algorithm outperformed the GDF algorithm, as shown in Fig. 6f. From the observed results, although the MFFI-Miner algorithm required extra memory in order to retain the fuzzy-list structures, the number of candidates could be greatly reduced when the search space was eliminated by using these pruning strategies.

6 Conclusion

In this paper, an efficient fuzzy-list structure was presented to keep the necessary information for mining MFFIs. Two pruning strategies were designed to reduce the search space, thus speeding up computations and reducing memory usage to mine MFFIs. From results of various experiments, it was observed that the proposed approaches outperformed the GDF and UBMFFP-tree algorithms in terms of runtime, memory usage, and number of determining candidates in both real-world and synthetic datasets.

Acknowledgments This research was partially supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61503092.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Database mining: A performance perspective. IEEE Transactions on Knowledge and Data Engineering **5**(6), 914–925 (1993)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases, *The International Conference on Very Large Data Bases*, pp. 487-499, (1994)
3. Agrawal, R., Srikant, R.: Mining sequential patterns, *The International Conference on Data Engineering*, pp. 3-14, (1995)
4. Antonelli, M., Ducange, P., Marcelloni, F.: A novel associative classification model based on a fuzzy frequent pattern mining algorithm. Expert Systems with Applications **42**(4), 2086–2097 (2015)
5. Berkhin, P.: A survey of clustering data mining techniques, *Grouping Multidimensional Data*, pp. 25-71, (2006)
6. Chan, K. C. C., Au, W. H.: Mining fuzzy association rules, *International Conference on Information and Knowledge Management*, pp. 209-215, (1997)
7. Chan, R., Yang, Q., Shen, Y. D.: Mining high utility itemsets, *IEEE International Conference on Data Mining*, pp. 19-26, (2003)
8. Chen, M.S., Han, J., Yu, P.S.: Data mining: An overview from a database perspective. IEEE Transactions on Knowledge and Data Engineering **8**(6), 866–883 (1996)
9. Frequent Itemset Mining Dataset Repository, http://fimi.ua.ac.be/data/
10. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining and Knowledge Discovery **8**(1), 53–87 (2004)
11. Hong, T.P., Kuo, C.S., Chi, S.C.: Mining association rules from quantitative data. Intelligent Data Analysis **3**(5), 363–376 (1999)
12. Hong, T.P., Lin, C.W., Wu, Y.L.: Incrementally fast updated frequent pattern trees. Expert Systems with Applications **34**(4), 2424–2435 (2008)
13. Hong, T.P., Lan, G.C., Lin, Y.H., Pan, S.T.: An effective gradual data-reduction strategy for fuzzy itemset mining. International Journal of Fuzzy Systems **15**(2), 170–181 (2013)
14. Hong, T.P., Lin, C.W., Lin, T.C.: The MFFP-tree fuzzy mining algorithm to discover complete linguistic frequent itemsets. Computational Intelligence **30**(1), 145–166 (2014)
15. Kuok, C.M., Fu, A., Wong, M.H.: Mining fuzzy association rules in databases. ACM SIGMOD Record **27**(1), 41–46 (1998)
16. Kim, C., Lim, J.H., Ng, R.T., Shim, K.: SQUIRE: Sequential pattern mining with quantities. Journal of Systems and Software **80**(10), 1726–1745 (2007)
17. Lan, G.C., Hong, T.P., Lin, Y.H., Wang, S.L.: Fuzzy utility mining with upper-bound measure. Applied Soft Computing **30**, 767–777 (2015)
18. Lin, C.W., Hong, T.P., Lu, W.H.: Linguistic data mining with fuzzy fp-trees. Expert Systems with Applications **37**(6), 4560–4567 (2010)
19. Lin, C.W., Hong, T.P., Lin, T.C.: An efficient tree-based fuzzy data mining approach. International Journal of Fuzzy Systems **12**(2), 150–157 (2010)
20. Lin, C.W., Hong, T.P., Lu, W.H.: An effective tree structure for mining high utility itemsets. Expert Systems with Applications **38**(6), 7419–7424 (2011)
21. Lin, C.W., Hong, T.P., Lu, W.H.: Mining fuzzy frequent itemsets based on UBFFP trees. Journal of Intelligent & Fuzzy Systems **27**(1), 535–548 (2014)
22. Lin, J.C.W., Hong, T.P., Lin, T.C.: A CMFFP-tree algorithm to mine complete multiple fuzzy frequent itemsets. Applied Soft Computing **28**(C), 431–439 (2015)
23. Lin, J.C.W., Hong, T.P., Lin, T.C., Pan, S.T.: An UBMFFP tree for mining multiple fuzzy frequent itemsets. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems **23**(6), 861–879 (2015)
24. Shitong, W., Chung, K.F.L., Hongbin, S.: Fuzzy taxonomy, quantitative database and mining generalized association rules. Intelligent Data Analysis **9**(2), 207–217 (2005)
25. Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., et al.: Top 10 algorithms in data mining. Knowledge and Information Systems **14**(1), 1–37 (2007)
26. Zadeh, L.A.: Fuzzy sets. Information and Control **8**, 338–353 (1965)

Jerry Chun-Wei Lin is currently working as an Assistant Professor at School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, China. He has published more than 200 research papers in referred journals and international conferences. His research interests include data mining, soft computing, artificial intelligence, social computing, multimedia and image processing, and privacy-preserving and security technologies. He is also one of the project leader of SPMF: An Open-Source Data Mining Library website (http://www.philippe-fournier-viger.com/spmf/), which is a toolkit to collect and implement the library for variants of data mining algorithms. He also serves as the Editor-in-Chief of an international journal of Data Science and Pattern Recognition (DSPR)

Ting Li is s a master student studying at School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, China. His research interests include security and privacy-preserving data mining

Philippe Fournier-Viger is an assistant professor at University of Moncton, Canada. He received a Ph.D. in Cognitive Computer Science at the University of Quebec in Montreal (2010). He has published more than 75 research papers in refereed international conferences and journals. He is the founder of the popular SPMF open-source data mining library

Tzung-Pei Hong received his B.S. degree in chemical engineering from National Taiwan University in 1985 and his Ph.D. degree in computer science and information engineering from National Chiao-Tung University in 1992. He is currently a distinguished professor at the Department of Computer Science and Information Engineering and at the Department of Electrical Engineering. He has published more than 400 research papers in international/national journals and conferences and has planned more than fifty information systems. He is also the board member of more than forty journals and the program committee member of more than three hundred conferences. His current research interests include knowledge engineering, data mining, soft computing, management information systems, and www applications. Dr. Hong is a member of the Association for Computing Machinery, the IEEE, the Chinese Fuzzy Systems Association, the Taiwanese Association for Artificial Intelligence, and the Institute of Information and Computing Machinery

Jimmy Ming-Tai Wu is a Postdoctoral Research Fellow in the Department of Computer Science, University of Nevada, Las Vegas. He was a Postdoctoral Research Fellow in the Department of Computer Science, National University of Kaohsiung, Taiwan. His research interests include data mining, artificial intelligence, bio-information and big data

Justin Zhan is the Director of Big Data Hub and Director of ILAB. He is a faculty member of the Department of Computer Science, University of Nevada, Las Vegas. He has been previously a faculty member at the Carnegie Mellon University, National Center for the Protection of Financial Infrastructure in Dakota State University and North Carolina A&T State University. His research interests include big data, information assurance, social computing, and medical informatics. His research has been sponsored by the National Science Foundation, Department of Defense, and National Institute of Health