

Upper-Bound Multiple Fuzzy Frequent-Pattern Trees

Tzung-Pei Hong^{1,2}, Chun-Wei Lin¹

¹Department of Computer Science and Information
Engineering

National University of Kaohsiung
Kaohsiung, Taiwan, R.O.C.

²Department of Computer Science and Engineering
National Sun Yat-sen University

Kaohsiung, Taiwan, R.O.C.

tphong@nuk.edu.tw, cwlin@nuk.edu.tw

Tsung-Ching Lin and Shing-Tai Pan

Department of Computer Science and Information
Engineering

National University of Kaohsiung
Kaohsiung, Taiwan, R.O.C.

stpan@nuk.edu.tw, m0985507@mail.nuk.edu.tw

Abstract—In this paper, a novel two-phase fuzzy mining approach based on the designed upper-bound multiple fuzzy frequent-pattern (UBMFFP) tree is proposed to obtain all fuzzy frequent itemsets from a quantitative database. It prunes unpromising itemsets in the first phase, and then finds the actual fuzzy frequent itemsets in the second phase. Experimental results indicate that the proposed approach has better performance than some previous ones.

Keywords—fuzzy data mining; fuzzy set; two-phase mining; FP tree; fuzzy frequent itemsets

I. INTRODUCTION

Depending on the classes of knowledge derived, it may be classified as those finding association rules [1], classification rules [12], clustering rules [16], or sequential patterns [2, 20], among others. Especially, finding association rules in large transaction datasets is most commonly seen in data mining research [4, 18]. In these years, fuzzy sets have been used in many fuzzy data mining approaches for managing quantitative data [5, 8-9]. It used membership functions to transform quantitative values in transactions into linguistic terms through fuzzy theory. Papadimitriou *et al.* proposed a tree structure called the fuzzy frequent pattern tree (FFPT) to mine fuzzy association rules [19]. Lin *et al.* then proposed fuzzy data mining approaches to efficiently mine the fuzzy frequent itemsets from the designed tree structures [13-15]. It only considers the term with the maximum cardinality, thus making the number of fuzzy regions processed equal to the number of original items, which reduces the processing time.

In this paper, an upper-bound multiple fuzzy FP tree (UBMFFP tree) algorithm is proposed, which keeps the fuzzy frequent 1-regions whether they are generated from the same item. In the mining process, the fuzzy regions transformed from the same term (item) cannot be joined into fuzzy frequent itemsets using the combination method. Based on the proposed approach, the mining result are more natural and understandable for human beings.

II. REVIEW OF RELATED WORKS

In this section, the fuzzy association rules mining and the FP-growth algorithm are shortly described.

A. Fuzzy Association Rules Mining

The data mining approach is useful to extract knowledge from real-world databases and represented in a comprehensible form. Linguistic representation produces more understandable knowledge for human beings, which is easily implemented by fuzzy sets. Chan *et al.* proposed the F-APACS algorithm to mine fuzzy association rules [3]. Hong *et al.* proposed a fuzzy aprioritid mining algorithm to mine fuzzy rules from quantitative databases [8]. Papadimitriou *et al.* proposed a fuzzy frequent pattern tree (FFPT) to mine fuzzy association rules [19]. Lin *et al.* then proposed the novel fuzzy data mining approaches to efficiently mine the fuzzy frequent itemsets from the designed tree structures [13-15]. The above algorithms, however, were proposed for handling single region with the maximum cardinality among transformed fuzzy regions of an item, thus generating incomplete fuzzy frequent itemsets.

B. FP-growth Algorithm

Han *et al.* proposed the Frequent-Pattern-tree structure (FP-tree) and the FP-growth algorithm for efficiently mining frequent itemsets without generation of candidate itemsets [7]. Three steps were involved in FP-tree construction. The database to be mined was first scanned to find all the items with their counts. The support of an item was then defined as its count divided by the total number of transactions. The items with their supports equal to or larger than a predefined minimum support were then selected as frequent 1-itemsets (items). Next, the frequent items were sorted in a descending order of their frequencies. At last, the database was scanned again to construct the FP tree according to the sorted order of frequent items. The construction process was executed tuple by tuple, from the first transaction to the last one. After all transactions were processed, the FP tree was completely constructed.

After the FP tree was constructed from a database, the FP-growth mining algorithm was then executed to find all frequent itemsets [7]. Several other algorithms based on the FP-tree structure have also been proposed [6, 10].

III. THE PROPOSED UBMFFP CONSTRUCTION ALGORITHM

A special notation is often used to represent fuzzy sets. Assume that x_1 to x_n are the elements in a fuzzy set A , and μ_1 to

μ_2 are, respectively, their membership grades in A . A is usually represented as follows:

$$A = \frac{\mu_1}{x_1} + \frac{\mu_2}{x_2} + \dots + \frac{\mu_n}{x_n}.$$

A. The UBMFFP- tree construction algorithm

The proposed mining approach integrates the fuzzy-set concepts, the FP-tree-like algorithm and the two-phase processing way to efficiently derive fuzzy frequent itemsets from quantitative transactions. The UBMFFP tree construction algorithm is stated as follows.

UBMFFP tree construction algorithm:

INPUT: A quantitative database consisting of n transactions and m items, a set of membership functions, and a predefined minimum support threshold s .

OUTPUT: A constructed UBMFFP tree.

STEP 1: Transform the quantitative value v_{ij} of each item I_j in the i -th transaction into a fuzzy set f_{ij} represented as $(f_{ij1}/R_{j1} + f_{ij2}/R_{j2} + \dots + f_{ijh_j}/R_{jh_j})$ using the given membership functions, where h_j is the number of fuzzy regions for I_j , R_{jl} is the l -th fuzzy region of I_j , $1 \leq l \leq h_j$, and f_{ijl} is v_{ij} 's fuzzy membership value in region R_{jl} .

STEP 2: Calculate the scalar cardinality of each fuzzy region R_{jl} in the transaction data as:

$$count_{jl} = \sum_{i=1}^n f_{ijl}.$$

STEP 3: Check whether the value $count_{jl}$ of the fuzzy region R_{jl} is greater than or equal to the user-specific minimum count $n*s$. If the count of a fuzzy region R_{jl} is equal to or greater than the minimum count, put the fuzzy region in the set of fuzzy frequent regions (L_1). That is:

$$L_1 = \{R_{jl} | count_{jl} \geq n*s, 1 \leq j \leq m\}.$$

STEP 4: While executing the above steps, also find the occurrence number $o(R_{jl})$ of each fuzzy region R_{jl} in the quantitative database. It is later used to decide the item order.

STEP 5: Build the Header_Table by keeping the fuzzy frequent regions in L_1 in the descending order of their occurrence numbers.

STEP 6: Remove the fuzzy regions of the items not in L_1 from the transactions of the transformed database. Sort the remaining fuzzy regions in each transaction according to the order of the fuzzy regions in Header_Table.

STEP 7: Initially set the root node of the UBMFFP tree as **root**.

STEP 8: Insert the updated transactions in the transformed databases into the UBMFFP tree tuple by tuple. The following two cases may exist to concern.

Substep 8-1: If a fuzzy region R_{jl} in the currently processed i -th transaction appears in the corresponding path of the UBMFFP tree, add the membership value of the region R_{jl} in the transaction to the node with R_{jl} ;

Substep 8-2: Otherwise, add a new node with R_{jl} to the end of the corresponding path and set the membership value of the region R_{jl} in the currently processed i -th transaction as the value in the node. Insert a link from the node of R_{jl} in the last branch to the current node. If there is no such branch, insert a link from the entry of R_{jl} in Header_Table to the current node.

After STEP 8, the UBMFFP tree finally constructed.

B. An Example

Assume that the predefined minimum support threshold is set at 30%. The quantitative database is shown in Table I.

TABLE I. QUANTITATIVE DATABASE

TID	Items
1	A:2, C:8, E:10
2	A:2, C:5, E:10
3	B:4, C:8, D:5
4	B:9, D:4
5	A:3, B:5, C:5, D:9
6	A:2, B:7, C:11
7	B:5, C:3, D:3, E:9
8	A:3, B:7, C:9, E:9

Assume that the fuzzy membership functions are the same for all items shown in Figure 1. The UBMFFP tree is constructed as follows.

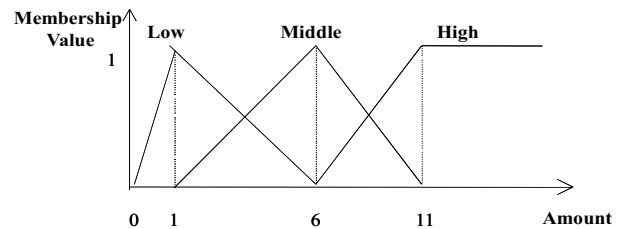


Figure 1. Membership functions used in the example.

The quantitative database in Table I is then transformed into fuzzy sets according to the defined membership function in Figure 1. The results are shown in Table II.

TABLE II. FUZZY SETS TRANSFORMED FROM TABLE I

TID	Items
1	$\frac{0.8}{A.Low} + \frac{0.2}{A.Middle} + \frac{0.6}{C.Middle} + \frac{0.4}{C.High} + \frac{0.2}{E.Middle} + \frac{0.8}{E.High}$
2	$\frac{0.8}{A.Low} + \frac{0.2}{A.Middle} + \frac{0.2}{C.Low} + \frac{0.8}{C.Middle} + \frac{0.2}{E.Middle} + \frac{0.8}{E.High}$
3	$\frac{0.4}{B.Low} + \frac{0.6}{B.Middle} + \frac{0.6}{C.Middle} + \frac{0.4}{C.High} + \frac{0.2}{D.Low} + \frac{0.8}{D.Middle}$
4	$\frac{0.4}{B.Middle} + \frac{0.6}{B.High} + \frac{0.4}{D.Low} + \frac{0.6}{D.Middle}$
5	$\frac{0.6}{A.Low} + \frac{0.4}{A.Middle} + \frac{0.2}{B.Low} + \frac{0.8}{B.Middle} + \frac{0.2}{C.Low} + \frac{0.8}{C.Middle} + \frac{0.4}{D.Middle} + \frac{0.6}{D.High}$
6	$\frac{0.8}{A.Low} + \frac{0.2}{A.Middle} + \frac{0.8}{B.Middle} + \frac{0.2}{B.High} + \frac{1}{C.High}$
7	$\frac{0.2}{B.Low} + \frac{0.8}{B.Middle} + \frac{0.6}{C.Low} + \frac{0.4}{C.Middle} + \frac{0.6}{D.Low} + \frac{0.4}{D.Middle} + \frac{0.4}{E.Middle} + \frac{0.6}{E.High}$
8	$\frac{0.6}{A.Low} + \frac{0.4}{A.Middle} + \frac{0.8}{B.Middle} + \frac{0.2}{B.High} + \frac{0.4}{C.Middle} + \frac{0.4}{C.High} + \frac{0.6}{E.Middle} + \frac{0.6}{E.High}$

The minimum count is then calculated as $(8 * 0.3) (= 2.4)$. The occurrence number of each frequent fuzzy region is calculated and shown in Table III.

TABLE III. COUNTS AND THE OCCURRENCE NUMBERS OF FUZZY REGIONS

Item	Count	Occurrence number
<i>A.Low</i>	3.6	5
<i>B.Middle</i>	4.2	6
<i>C.Middle</i>	3.6	6
<i>C.High</i>	2.4	4
<i>E.High</i>	2.8	4

The fuzzy regions in L_1 are then sorted in descending order of their occurrence numbers and are put in Header_Table. The fuzzy regions not in L_1 are removed from the transactions from Table II. The remaining fuzzy regions in each transaction are then sorted according to the order in the Header_Table. The updated transactions are shown in Table IV.

TABLE IV. THE UPDATED TRANSACTIONS

TID	Fuzzy frequent regions
1	$\frac{0.6}{C.Middle} + \frac{0.8}{A.Low} + \frac{0.4}{C.High} + \frac{0.8}{E.High}$
2	$\frac{0.8}{C.Middle} + \frac{0.8}{A.Low} + \frac{0.8}{E.High}$
3	$\frac{0.6}{B.Middle} + \frac{0.6}{C.Middle} + \frac{0.4}{C.High}$
4	$\frac{0.4}{B.Middle}$
5	$\frac{0.8}{B.Middle} + \frac{0.8}{C.Middle} + \frac{0.6}{A.Low}$
6	$\frac{0.8}{B.Middle} + \frac{0.8}{A.Low} + \frac{1}{C.High}$
7	$\frac{0.8}{B.Middle} + \frac{0.4}{C.Middle} + \frac{0.6}{E.High}$
8	$\frac{0.8}{B.Middle} + \frac{0.4}{C.Middle} + \frac{0.6}{A.Low} + \frac{0.6}{C.High} + \frac{0.6}{E.High}$

The updated transactions in Table IV are used to construct the UBMFFP tree tuple by tuple from the first transaction to the last one. The results are shown in Figure 2.

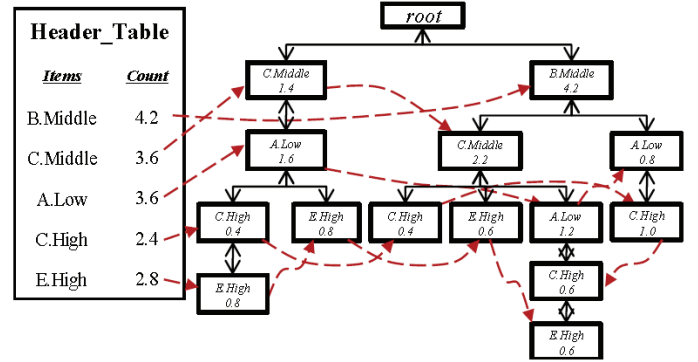


Figure 2. The finally constructed UBMFFP tree.

IV. THE UBMFFP-GROWTH MINING ALGORITHM

After the UBMFFP tree is constructed, the fuzzy frequent itemsets with more than one fuzzy region can then be found by the proposed UBMFFP-growth mining algorithm.

A. Proposed Mining Algorithm

The details of the UBMFFP-growth mining algorithm are as follows.

UBMFFP-growth mining algorithm:

INPUT: The already constructed UBMFFP tree, its corresponding Header_Table, the transferred database, and the predefined minimum support threshold s .

OUTPUT: The fuzzy frequent itemsets.

STEP 1: Process the fuzzy regions in Header_Table one by one and bottom-up using the following steps. Let the currently processed fuzzy region be R_{ji} .

STEP 2: Find all the nodes with the fuzzy region R_{ji} in the UBMFFP tree through the links, starting from Header_Table.

STEP 3: Set the variable $k = 2$, where k is used to keep the number of items in the currently processed itemsets.

STEP 4: Trace the paths with the currently processed fuzzy region R_{ji} to generate the fuzzy k -itemsets except for any of its super-items associated with the same I_j and set their upper-bound fuzzy counts as the minimum of the values in the corresponding nodes, which contain the fuzzy regions in the itemsets.

STEP 5: Sum the upper-bound fuzzy counts of the same fuzzy itemsets from different paths.

STEP 6: If the summed upper-bound fuzzy count of a fuzzy itemset after STEP 5 is larger than or equal to the

predefined minimum count ($n*s$), add it to the candidate set C which will be used to generate fuzzy candidate $(k+1)$ -itemsets.

STEP 7: Set $k = k + 1$ and repeat STEPs 4 to 6 to generate the fuzzy candidate k -itemsets in C based on the downward-closure property recursively. The process is repeated until no more fuzzy itemsets are generated.

STEP 8: Repeat STEPs 2 to 7 for another fuzzy region until all the fuzzy regions in Header_Table are processed.

STEP 9: Rescan the transferred database to get the actual fuzzy counts of the fuzzy candidate itemsets in C and check them against the predefined minimum count.

STEP 10: Output the candidate fuzzy itemsets with their fuzzy counts equal to or larger than the predefined minimum count as the fuzzy frequent itemsets.

B. An Example

For the constructed UBMFFP tree in Figure 2, the proposed UBMFFP-growth mining algorithm finds all the fuzzy frequent itemsets as follows. The variable k is initially set at 2, where k is used to keep the number of items in the currently processed itemsets. The paths with the currently processed fuzzy region are then used to form the candidate fuzzy k -itemsets. The upper-bound fuzzy counts of the same fuzzy itemsets are summed up. The summed upper-bound fuzzy counts of the derived fuzzy itemsets with the currently processed item are checked against the predefined minimum count, which is 2.4. The above steps are repeated for another fuzzy region until all the fuzzy regions in Header_Table are processed. The final candidate fuzzy itemsets with their upper-bound fuzzy counts in C are shown $\{A.Low, C.Middle: 2.6\}$, and $\{C.Middle, E.High: 2.8\}$.

The transferred database is then re-scanned to find the actual fuzzy counts of the derived candidate fuzzy itemsets. The final derived fuzzy frequent itemsets are shown in Table V.

TABLE V. FINAL FUZZY FREQUENT ITEMSETS

1-itemset	Count
<i>A.Low</i>	3.6
<i>B.Middle</i>	4.2
<i>C.Middle</i>	3.6
<i>C.High</i>	2.4
<i>E.High</i>	2.8
2-itemset	Count
<i>A.Low, C.Middle</i>	2.4

V. EXPERIMENTAL RESULTS

A real dataset called FOODMART from an anonymous chain store was used in the experiments [17]. The quantitative database was then transformed into fuzzy 2-regions by the

predefined membership functions. Figure 3 shows the comparisons of execution times of the Apriori-based algorithm [1] for mining multiple fuzzy-term frequent itemsets (MF-Apriori), the multiple fuzzy FP tree (MFFP-tree) algorithm [11], and the proposed UBMFFP-tree algorithm. It was obvious to see from Figure 3 that the proposed UBMFFP-tree algorithm ran faster than the other two for the various minimum support thresholds. The experimental results in Figure 4 show the numbers of tree nodes generated from the two algorithms for two fuzzy regions. It reveals that the numbers of nodes in the UBMFFP tree are fewer than the MFFP tree.

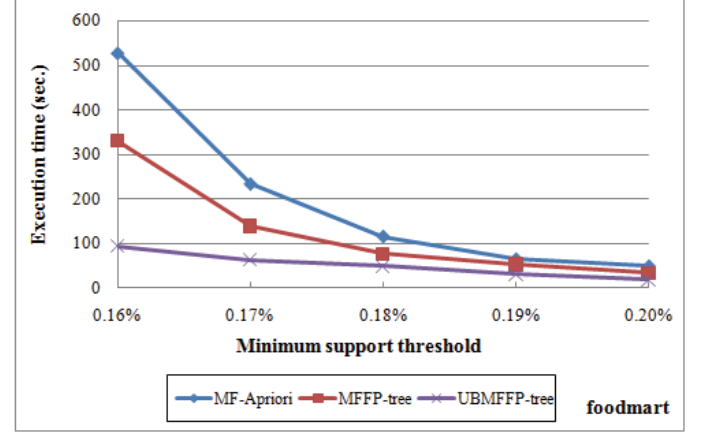


Figure 3. Execution time for three approaches.

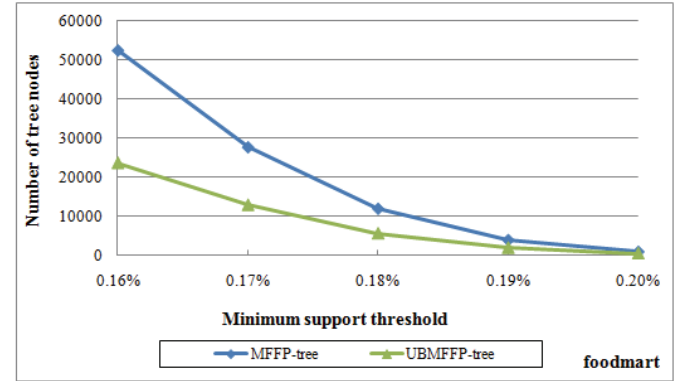


Figure 4. Number of tree nodes generated from the two approaches.

VI. CONCLUSION

In this paper, we proposed a two-phase multiple fuzzy mining algorithm for finding the complete fuzzy frequent itemsets based on the UBMFFP tree. Experimental results show that the proposed approach generates fewer nodes than the fuzzy FP-tree algorithm. In the future, we will further extend the approach to solve other fuzzy mining problems.

REFERENCES

- [1] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," The 20th International Conference on Very Large Data Bases, pp. 487–499, 1994.

- [2] R. Agrawal and R. Srikant, "Mining sequential patterns," The International Conference on Data Engineering, pp. 3–14, 1995.
- [3] K. C. C. Chan and W. H. Au, "Mining fuzzy association rules," The 6th International Conference on Information and Knowledge Management, pp. 209–215, 1997.
- [4] M. S. Chen, J. Han, and Y. P. S., "Data mining: An overview from a database perspective," IEEE Transactions on Knowledge and Data Engineering, vol. 8, pp. 866–883, 1996.
- [5] D. Dubois, E. Hullermeier, and H. Prade, "A systematic approach to the assessment of fuzzy association rules" Data Mining and Knowledge Discovery, vol. 13, pp. 167–192, 2006.
- [6] C. I. Ezeife and Y. Su, "Mining incremental association rules with generalized fp-tree," The 15th Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence, pp. 147–160, 2002.
- [7] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," Data Mining and Knowledge Discovery, vol. 8, pp. 53–87, 2004.
- [8] T. P. Hong, C. S. Kuo, and S. L. Wang, "A fuzzy aprioritid mining algorithm with reduced computational time," Applied Soft Computing, vol. 5, pp. 1–10, 2004.
- [9] T. P. Hong and J. B. Chen, "Finding relevant attributes and membership functions," Fuzzy Sets and Systems, vol. 103, pp. 389–404, 1999.
- [10] T. P. Hong, C. W. Lin, and Y. L. Wu, "Incrementally fast updated frequent pattern trees," Expert Systems with Applications, vol. 34, pp. 2424–2435, 2008.
- [11] T. P. Hong, C. W. Lin, and T. C. Lin, "Mining complete fuzzy frequent itemsets by tree structures," IEEE International Conference on Systems, Man, and Cybernetics, pp. 563–567, 2010.
- [12] K. Hu, Y. Lu, L. Zhou, and C. Shi, "Integrating classification and association rule mining: A concept lattice framework," The 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, pp. 443–447, 1999.
- [13] C. W. Lin, T. P. Hong, and W. H. Lu, "Linguistic Data Mining with Fuzzy FP-trees," Expert Systems with Applications, vol. 37, pp. 4560–4567, 2010.
- [14] C. W. Lin, T. P. Hong, and W. H. Lu, "An Efficient Tree-based Fuzzy Data Mining Approach," International Journal of Fuzzy Systems, vol. 12, no. 2, pp. 150–157, 2010.
- [15] C. W. Lin, T. P. Hong, and W. H. Lu, "A Two-Phase Fuzzy Mining Approach," IEEE International Conference on Fuzzy Systems, pp. 494–498, 2010.
- [16] F. Liu, Z. Lu, and S. Lu, "Mining association rules using clustering," Intelligent Data Analysis, vol. 5, pp. 309–326, 2001.
- [17] Microsoft. Example database foodmart of microsoft analysis services. Available: [http://msdn.microsoft.com/en-us/library/aa217032\(SQL.80\).aspx](http://msdn.microsoft.com/en-us/library/aa217032(SQL.80).aspx)
- [18] J. S. Park, M. S. Chen, and P. S. Yu, "An effective hash-based algorithm for mining association rules," SIGMOD Record, vol. 24, pp. 175–186, 1995.
- [19] S. Papadimitriou and S. Mavroudi, "The fuzzy frequent pattern tree," The 9th WSEAS International Conference on Computers, pp. 1–7, 2005.
- [20] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," The 5th International Conference on Extending Database Technology: Advances in Database Technology, pp. 3–17, 1996.