Expert Systems with Applications 37 (2010) 4560–4567

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Linguistic data mining with fuzzy FP-trees ☆

Chun-Wei Lin [a], Tzung-Pei Hong [b,c,*], Wen-Hsiang Lu [a]

[a] Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 701, Taiwan, ROC
[b] Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC
[c] Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan, ROC

ARTICLE INFO

Keywords:
Fuzzy data mining
Fuzzy-set
Quantitative value
Fuzzy FP-trees
Frequent fuzzy patterns

ABSTRACT

Due to the increasing occurrence of very large databases, mining useful information and knowledge from transactions is evolving into an important research area. In the past, many algorithms were proposed for mining association rules, most of which were based on items with binary values. Transactions with quantitative values are, however, commonly seen in real-world applications. In this paper, the frequent fuzzy pattern tree (fuzzy FP-tree) is proposed for extracting frequent fuzzy itemsets from the transactions with quantitative values. When extending the FP-tree to handle fuzzy data, the processing becomes much more complex than the original since fuzzy intersection in each transaction has to be handled. The fuzzy FP-tree construction algorithm is thus designed, and the mining process based on the tree is presented. Experimental results on three different numbers of fuzzy regions also show the performance of the proposed approach.

1. Introduction

Years of effort in data mining have produced a variety of efficient and effective techniques. Depending on the classes of the knowledge derived, the mining approaches may be classified as finding association rules, classification rules, clustering rules and sequential patterns (Agrawal & Srikant, 1995), among others. Especially, finding association rules in transaction databases is most commonly seen in data mining (Agrawal, Imielinski, & Swami, 1993a; Agrawal, Imielinski, & Swami, 1993b; Agrawal & Srikant, 1994; Chen, Han, & Yu, 1996; Cheung, Lee, & Kao, 1997).

In the past, many algorithms for mining association rules from transactions were proposed. Most of the approaches were based on the *Apriori* algorithm (Agrawal et al., 1993a), which generated and tested candidate itemsets level by level. This may cause iterative database scans and high computational costs. Han et al. thus proposed the Frequent-Pattern-tree (FP-tree) structure for efficiently mining association rules without generation of candidate itemsets (Han, Pei, & Yin, 2000). The FP-tree was used to compress a database into a tree structure which stored only large items. It was condensed and complete for finding all the frequent patterns. The construction process was executed tuple by tuple, from the first transaction to the last one. After that, a recursive mining procedure called FP-growth was executed to derive frequent patterns from the FP-tree.

In these years, the fuzzy-set theory (Zadeh, 1965) has been used more and more frequently in intelligent systems because of its simplicity and similarity to human reasoning (Kandel, 1992). Several fuzzy learning algorithms for inducing rules from given sets of data have been designed and used to good effect with specific domains (Hong & Chen, 1999, 2000). As to fuzzy data mining, several approaches have been proposed. For example, Hong et al. proposed a fuzzy mining algorithm for managing quantitative data (Hong, Kuo, & Chi, 1999b). It was based on the Apriori algorithm. Basically, it first used membership functions to transform each quantitative value into a fuzzy set in linguistic terms. It then calculated the cardinality of each linguistic term on all the transaction data. The mining process based on the cardinalities was then performed to find linguistic frequent itemsets and association rules.

Papadimitriou et al. proposed an approach based on FP-trees to find fuzzy association rules (Papadimitriou & Mavroudi, 2005). In their approach, each item in the transactions was transferred into only two fuzzy regions with individual fuzzy values. A threshold was set and a fuzzy region in a transaction would be removed if its fuzzy value was smaller than the threshold. In this process, only the local frequent fuzzy 1-itemsets kept in each transaction were used for mining. The fuzzy regions which were close to but below the threshold would provide no contribution at all to the mining. Thus, some fuzzy regions would not be frequent even the summation of its fuzzy values in the database was larger than or equal to

☆ This is a modified and expanded version of the paper "Mining fuzzy association rules based on fuzzy FP-trees", presented at The 16th National Conference on Fuzzy Theory and its Applications, Taiwan, 2008.

* Corresponding author. Address: Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC.
E-mail addresses: p7895122@mail.ncku.edu.tw (C.-W. Lin), tphong@nuk.edu.tw (T.-P. Hong), whlu@mail.ncku.edu.tw (W.-H. Lu).

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.12.052

the minimum support. Besides, the expression of fuzzy patterns with more fuzzy regions was straight. The approach did not use any fuzzy operation to combine fuzzy regions together. It made the mined fuzzy rules a little hard to understand.

In this paper, we attempt to extend the FP-tree mining process for handling quantitative data from the global values of fuzzy regions. A new fuzzy FP-tree is thus proposed, which is a data structure keeping frequent fuzzy regions. Besides, fuzzy operations are considered in forming itemsets with more than one fuzzy region. For achieving this purpose, the proposed fuzzy FP-tree is a little more complex than the original one and than that proposed by Papadimitriou and Mavroudi (2005). Based on the proposed approach, the frequent itemsets are efficiently expressed and mined out in linguistic terms, which are more natural and understandable for human beings.

The remainder of this paper is organized as follows. Related works are reviewed in Section 2. The notation used in the algorithm is explained in Section 3. The proposed fuzzy FP-tree construction algorithm is described in Section 4. An example to illustrate the proposed algorithm is given in Section 5. Experimental results for showing the performance of the proposed algorithm are provided in Section 6. Conclusions are finally given in Section 7.

2. Review of related works

In this section, some related researches are briefly reviewed. They are fuzzy-set concepts, mining association rules from quantitative data, and FP-tree algorithm.

2.1. Fuzzy-set concepts

A fuzzy set is an extension of a crisp set. Crisp sets allow only full membership or no membership at all, whereas fuzzy sets allow partial membership. Besides, each element may belong to more than one set. In a crisp set, the membership of an element x in set A is described by a characteristic function $u_A(x)$, where

$$u_A(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \notin A. \end{cases}$$

The fuzzy-set theory extends this concept by defining partial memberships, which can take values ranging from 0 to 1. A membership function is formally defined as follows (Kosko, 1997; Zadeh, 1965):

$$u_A : X \to [0,1],$$

where X refers to the universal set for a specific problem. Assuming that A and B are two fuzzy sets with membership functions $u_A(x)$ and $u_B(x)$, respectively. The following common fuzzy operators can be defined as follows:

(1) The intersection operator:

$$u_{A\cap B}(x) = u_A(x)\ \tau\ u_B(x),$$

where τ is a *t-norm* operator. That is, τ is a function of $[0,1]*[0,1] \to [0,1]$ and must satisfy the following conditions for each $a,b,c \in [0,1]$:

(i) $a\ \tau\ 1 = a$;
(ii) $a\ \tau\ b = b\ \tau\ a$;
(iii) $a\ \tau\ b \geqslant c\ \tau\ d$ if $a \geqslant c, b \geqslant d$;
(iv) $a\ \tau\ b\ \tau\ c = a\ \tau\ (b\ \tau\ c) = (a\ \tau\ b)\ \tau\ c$.

Two instances of a *t-norm* operator for $a\ \tau\ b$ are $min(a, b)$ and $a * b$.

(2) The union operator:

$$u_{A\cap B}(x) = u_A(x)\ \rho\ u_B(x),$$

where ρ is an *s-norm* operator. That is, ρ is a function of $[0,1]*[0,1] \to [0,1]$ and must satisfy the following conditions for each $a,b,c \in [0,1]$:

(i) $a\ \rho\ 0 = a$;
(ii) $a\ \rho\ b = b\ \rho\ a$;
(iii) $a\ \rho\ b \geqslant c\ \rho\ d$ if $a \geqslant c, b \geqslant d$;
(iv) $a\ \rho\ b\ \rho\ c = a\ \rho\ (b\ \rho\ c) = (a\ \rho\ b)\ \rho\ c$.

Two instances of an *s-norm* operator for $a\ \rho\ b$ are $max(a,b)$ and $a + b - a * b$.

(3) The α -cut operator:

$$A_\alpha(x) = \{x \in X | u_A(x) \geqslant \alpha\},$$

where A_α is an α-*cut* of a fuzzy set A. A_α thus contains all the elements in the universal set X that have membership grades in A greater than or equal to the specified value of α. These fuzzy operators will be used in our fuzzy FP-tree mining algorithm to derive fuzzy association rules.

2.2. Mining algorithms for fuzzy association rules

It is useful to extract knowledge via data from the real world and to represent it in a comprehensible form. Linguistic representation is popular and may help knowledge more understandable to human beings. It is also easily implemented by fuzzy sets, since the fuzzy-set theory is concerned with quantifying and reasoning using natural language. Several fuzzy mining approaches have been proposed to find interesting linguistic association rules or sequential patterns from transaction data with quantitative values.

For example, Chan et al. proposed an F-APACS algorithm to mine fuzzy association rules (Chan & Au, 1997). They first transformed quantitative attribute values into linguistic terms and then used the adjusted difference analysis to find interesting associations among attributes. Kuok et al. proposed a fuzzy mining approach to handle numerical data in databases and to derive fuzzy association rules (Kuok, Fu, & Wong, 1998). At nearly the same time, Hong et al. proposed a fuzzy mining algorithm to mine fuzzy rules from quantitative transaction data (Hong & Chen, 1999, 2000; Hong et al., 1999b; Hong, Kuo, & Chi, 1999a). Besides, many mining methods for finding fuzzy association rules have also been proposed (Kaya & Alhajj, 2004; Shen, Wang, & Yang, 2004; Srikant & Agrawal, 1996; Yue, Tsand, Yeung, & Shi, 2000), and some related researches are still in progress.

2.3. The FP-tree mining algorithm

Han et al. proposed the Frequent-Pattern-tree structure (FP-tree) for efficiently mining association rules without generation of candidate itemsets (Han et al., 2000). The FP-tree mining algorithm consists of two phases. The first phase focuses on constructing the FP-tree from a database, and the second phase focuses on deriving frequent patterns from the FP-tree. Three steps are involved in FP-tree construction. The database is first scanned to find all items with their counts. The items with their supports equal to or larger than a predefined minimum support are selected as frequent 1-itemsets (items). Next, the frequent items are sorted in descending frequency. At last, the database is scanned again to construct the FP tree according to the sorted order of frequent items. The construction process is executed tuple by tuple, from the first transaction to the last one. After all transactions are processed, the FP tree is completely constructed.

After the FP tree is constructed from a database, a mining procedure called FP-growth (Han et al., 2000) is executed to find all frequent itemsets. FP-growth does not need to generate candidate itemsets for mining, but derives frequent patterns directly from the FP tree. A conditional FP tree is generated for each frequent item,

and from the tree the frequent itemsets with the processed item can be recursively derived.

Several other algorithms based on the FP-tree structure have been proposed. For example, Qiu et al. proposed the QFP-growth mining approach to mine association rules (Qiu, Lan, & Xie, 2004). Mohammad proposed the COFI-tree structure to replace the conditional FP-tree (Zaiane & Mohammed, 2003). Ezeife constructed a generalized FP-tree, which stored all the frequent and infrequent items, for incremental mining without rescanning databases (Ezeife, 2002). Koh et al. adjusted FP trees with a complex procedure (Koh & Shieh, 2004). Some related researches are still in progress.

3. Notations

The notation used in the proposed fuzzy mining algorithm is first described below:

D	the original quantitative database
n	the number of transactions in D
T	the ith transaction in D, $1 \leqslant i \leqslant n$
m	the number of items in D
I_j	the jth item, $1 \leqslant j \leqslant m$
h_j	the number of fuzzy regions for I_j
R_{jl}	the lth fuzzy region of $I_j, 1 \leqslant l \leqslant h_j$
v_{ij}	the quantitative value of I_j *in* T
f_{ijl}	the membership value of v_{ij} in region R_{jl}
$count_{jl}$	the count of the fuzzy region R_{jl} *in* D;
$max\text{-}count_j$	the maximum count value among the fuzzy regions of I_j
$max\text{-}R_j$	the fuzzy region of $I_j with max - count_j$
s	the predefined minimum support threshold

Besides, a special notation is often used to represent fuzzy sets. Assume that x_1 to x_n are the elements in fuzzy set A, and μ_1 to μ_n are, respectively, their membership grades in A. A is usually represented as follows:

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

4. The proposed fuzzy fp-tree mining algorithm

The proposed fuzzy FP-tree mining algorithm integrates the fuzzy-set concepts and the FP-tree-like approach to find frequent fuzzy itemsets from quantitative transaction data. The fuzzy FP-tree construction algorithm is designed to generate the tree structure for frequent fuzzy regions (terms). It first transforms quantitative values in transactions into linguistic terms based on Hong et al.'s approach (Hong et al., 1999a, 1999b). Each term uses only the linguistic term with the maximum cardinality in later processes, thus making the number of fuzzy regions processed equal to the number of the original items for reducing the processing time. The frequent fuzzy itemsets, represented by linguistic terms, are then derived from the fuzzy FP tree.

The fuzzy FP-tree structure could thus help mine fuzzy association rules from quantitative data efficiently and effectively. However, when extending the crisp FP tree to the fuzzy one, the processing becomes much more complex than the original since fuzzy intersection in each transaction has to be handled. The fuzzy FP-tree construction algorithm is stated below.

4.1. The fuzzy FP- tree construction algorithm

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

OUTPUT: A fuzzy FP tree.

STEP 1: Transform the quantitative value v_{ij} of each item I_j in the ith transaction into a fuzzy set f_{ij} represented as $(f_{ij1}/R_{j1} + f_{ij2}/R_{j2} + \cdots + f_{ijn}/R_{jh})$ using the given membership functions, where h is the number of fuzzy regions for I_j, R_{jl} is the lth fuzzy region of $I_j, 1 \leqslant l \leqslant h$, 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: Find $max\text{-}count_j = MAX_{l=1}^{h}(count_{jl})$ for $j = 1$–m, where h is the number of fuzzy regions for item I_j and m is the number of items. Let $max\text{-}R_j$ be the region with $max\text{-}count_j$ for item I_j. It will then be used to represent the fuzzy characteristic of item I_j in the later mining process.

STEP 4: Check whether the value $max\text{-}count_j$ of a kept fuzzy region $max\text{-}R_j$, $j = 1$–m, is larger than or equal to the predefined minimum count $n * s$. If the count of a fuzzy region $max\text{-}R_j$ is equal to or greater than the minimum count, put the fuzzy region in the set of frequent fuzzy regions (L_1). That is:

$$L_1 = \{max\text{-}R_j | max\text{-}count_j \geqslant n * s,\ 1 \leqslant j \leqslant m\}.$$

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

STEP 6: Remove the fuzzy regions of the items not in L_1 from the transactions of the transformed database.

STEP 7: Sort the remaining frequent fuzzy regions in each transaction by their membership values in a descending order.

STEP 8: Initially set the root node of the fuzzy FP tree as {*root*}.

STEP 9: Insert the transactions in the transformed database into the fuzzy FP tree tuple by tuple. The following two cases may exist.

Substep 9-1: If a fuzzy region $max\text{-}R_j$ in a transaction has been at the corresponding branch of the fuzzy FP tree for the transaction, add the membership value of $max\text{-}R_j$ in the transaction to the node of $max\text{-}R_j$ in the branch.

Substep 9-2: Otherwise, add a node of $max\text{-}R_j$ at the end of the corresponding branch, set the count of the node as the membership value of $max\text{-}R_j$, and insert a link from the node of $max\text{-}R_j$ in the last branch to the current node. If there is not such a branch with the node of $max\text{-}R_j$, insert a link from the entry of $max\text{-}R_j$ in the Header-Table to the added node.

In STEP 9, a corresponding branch is the branch generated from a transformed transaction according to the descending order of the membership values of the fuzzy regions in it. After STEP 9, the final fuzzy FP tree is constructed. With the fuzzy FP tree, the desired frequent fuzzy itemsets can then be found in a way similar to the FP-growth mining approach (Han et al., 2000), but much more complex. The process is stated as follows.

The fuzzy regions in the Header_Table are processed one by one and bottom–up. A fuzzy conditional pattern tree is first built for each frequent fuzzy region. The counts of the itemsets containing the fuzzy region are then recursively calculated. Since each branch in the FP tree is built according to the membership values of the fuzzy regions in a transaction, the count of a fuzzy k-itemset (with k fuzzy terms) obtained by the fuzzy intersection (minimum) operator can be easily achieved without rescan of the database. If the

total membership value of a fuzzy k-itemset from all transactions is larger than or equal to the predefined minimum count $n * s$, the k-itemset is thought of as frequent.

5. An example

In this session, an example is given to illustrate how to construct a fuzzy FP tree and generate frequent fuzzy itemsets based on the proposed approach from the quantitative transaction data. Assume the quantitative transaction database shown in Table 1 is used as the example. It consists of six transactions and five items, denoted A–E.

Assume the fuzzy membership functions are the same for all the items and are shown in Fig. 1. In this example, amounts are represented by three fuzzy regions: *Low*, *Middle* and *High*. Thus, three fuzzy membership values are produced for each item in a transaction according to the predefined membership functions. Note that our proposed approach also works when the membership functions of the amounts for the items are not the same.

The proposed approach constructs the fuzzy FP tree for this example as follows:

STEP 1: The quantitative values of the items in the transactions are represented as fuzzy sets. Take the first item in transaction 1 as an example. The amount "5" of A is converted into the fuzzy set $\left(\frac{0.2}{A.Low} + \frac{0.8}{A.Middle}\right)$ using the given membership functions in Fig. 1. This step is repeated for the other items, and the results are shown in Table 2, where the notation *item.term* is called a fuzzy region.

STEP 2: The scalar cardinality of each fuzzy region in the transaction is calculated as the *count* value. Take the fuzzy region *A.Low* as an example. Its scalar cardinality is $(0.2 + 0.0 + 0.0 + 0.0 + 0.2 + 0.6)$, which is 1.0. This step is repeated for the other regions, and the results are shown in Table 3.

STEP 3: The fuzzy region with the maximum count among the three possible regions for each item is found. Take item A as an example. Its count is 1.0 for *Low*, 3.4 for *Middle*, and 0.6 for *High*. Since the count for *Middle* is the maximum among the three counts, the region *Middle* is thus used to represent item A in the later mining process. This step is repeated for the other items. Thus, region *Low* is chosen for B, D, and E, and region *High* is chosen for C.

STEP 4: The count of any region selected in STEP 3 is checked against the predefined minimum support value s. Assume in this example, s is set at 30%. Since the count values of *A.Middle, B.Low, C.High*, and *D.Low* are larger than $6 * 30\% (= 1.8)$, these fuzzy region are put in the set of L_1, which will be used to construct the fuzzy FP tree later. The results are shown in Table 4.

Table 1
Six quantitative transactions in the example.

Transaction no.	Items
1	(A:5) (C:10) (D:2) (E:9)
2	(A:8) (B:2) (C:3)
3	(B:3) (C:9)
4	(A:7) (C:9) (D:3)
5	(A:5) (B:2) (C:5)
6	(A:3) (C:10) (D:2) (E:2)

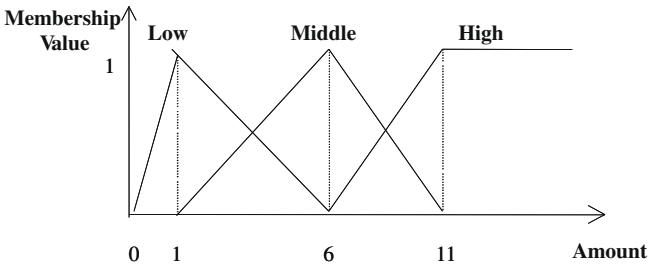

Fig. 1. The membership functions used in this example.

Table 2
The fuzzy sets transformed from the data in Table 1.

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

Table 3
The counts of fuzzy regions.

Item	Count	Item	Count	Item	Count
A.Low	1.0	*C.Low*	0.8	*E.Low*	0.8
A.Middle	3.4	*C.Middle*	2.4	*E.Middle*	0.6
A.High	0.6	*C.High*	2.8	*E.High*	0.1
B.Low	2.2	*D.Low*	2.2		
B.Middle	0.8	*D.Middle*	0.8		
B.High	0.0	*D.High*	0.0		

Table 4
The set of frequent fuzzy regions in the example.

Frequent fuzzy regions	Count
A.Middle	3.4
B.Low	2.2
C.High	2.8
D.Low	2.2

Header_Table

Fuzzy Region	*Count*
A.Middle	3.4
C.High	2.8
B.Low	2.2
D.Low	2.2

Fig. 2. The Header_Table formed after STEP 5.

Table 5
The transformed transactions with frequent fuzzy regions.

TID	Frequent fuzzy regions
1	$\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.8}{C.High}\right)\left(\frac{0.8}{D.Low}\right)$
2	$\left(\frac{0.6}{A.Middle}\right)\left(\frac{0.8}{B.Low}\right)$
3	$\left(\frac{0.6}{B.Low}\right)\left(\frac{0.6}{C.High}\right)$
4	$\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.6}{C.High}\right)\left(\frac{0.6}{D.Low}\right)$
5	$\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.8}{B.Low}\right)$
6	$\left(\frac{0.4}{A.Middle}\right)\left(\frac{0.8}{C.High}\right)\left(\frac{0.8}{D.Low}\right)$

STEP 5: The frequent fuzzy regions in L_1 are then sorted in the descending order of their counts and are put into the Header_Table. The results are shown in Fig. 2.

STEP 6: The fuzzy regions which are not in L_1 are removed from each transaction in Table 2. The results are shown in Table 5.

STEP 7: The remaining fuzzy regions in each transaction of Table 5 are then sorted according to their membership values in a descending order. After this step, the transactions with only the sorted frequent fuzzy regions (1-itemsets) are shown in Table 6.It can be seen from Table 6 that the order of *A.Middle* and *C.High* are different from that in Table 5.

STEP 8: The corresponding fuzzy FP tree is to be constructed. The root of the fuzzy FP tree is initially set as {*root*}.

STEP 9: The transactions in the transformed database are inserted into the fuzzy FP tree tuple by tuple. In Table 6, the first transaction is $\left(\frac{0.8}{A.Middle}, \frac{0.8}{C.High}, \frac{0.8}{D.Low}\right)$. This transaction is then inserted into the fuzzy FP tree as the first branch. The first node, which is, *A.Middle* is thus inserted as the child of the root. The next node, which is, *C.High* is then inserted as the child of the first node *A.Middle*. The same procedure is processed for inserting the node of *D.Low*. Each node in the branch is attached with the membership value of the corresponding fuzzy region. Besides, since each node in the branch is the first one for the fuzzy region, a link is then connected from the Header_Table to the node. The results after the first transaction is processed are shown in Fig. 3.

The second transaction is next processed. Its content is $\left(\frac{0.8}{B.Low}, \frac{0.6}{A.Middle}\right)$. It is then inserted into the fuzzy FP tree as the second branch because it does not share the same prefix with the first transaction. Besides, a link is created between the two nodes of *A.Middle* in the two branches. The results are shown in Fig. 4.

Table 6
The transactions with only the sorted frequent fuzzy regions.

TID	Frequent fuzzy regions
1	$\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.8}{C.High}\right)\left(\frac{0.8}{D.Low}\right)$
2	$\left(\frac{0.8}{B.Low}\right)\left(\frac{0.6}{A.Middle}\right)$
3	$\left(\frac{0.6}{B.Low}\right)\left(\frac{0.6}{C.High}\right)$
4	$\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.6}{C.High}\right)\left(\frac{0.6}{D.Low}\right)$
5	$\left(\frac{0.8}{A.Middle}\right)\left(\frac{0.8}{B.Low}\right)$
6	$\left(\frac{0.8}{C.High}\right)\left(\frac{0.8}{D.Low}\right)\left(\frac{0.4}{A.Middle}\right)$

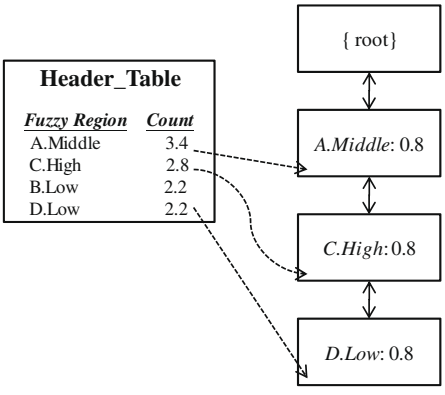

Fig. 3. The fuzzy FP tree after the first transaction is processed.

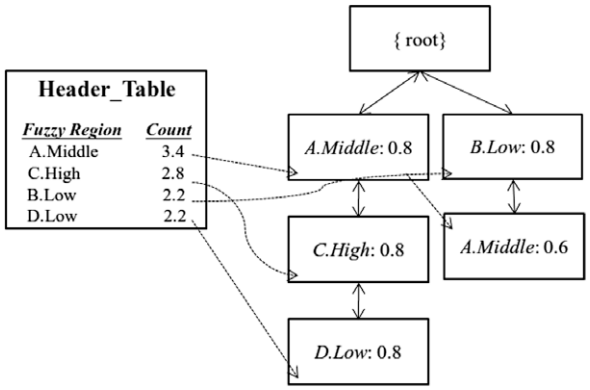

Fig. 4. The fuzzy FP-tree after the second transaction is processed.

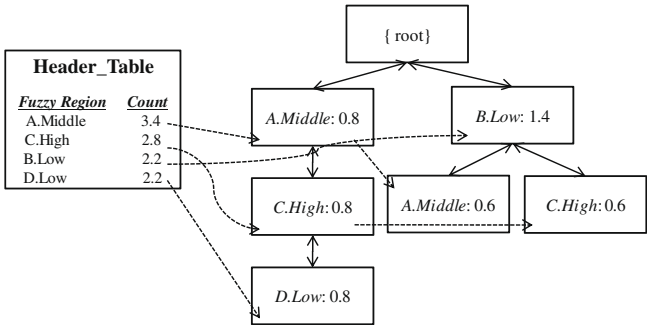

Fig. 5. The fuzzy FP-tree after the third transaction is processed.

The third transaction is next processed. It contains $\left(\frac{0.6}{B.Low}, \frac{0.6}{C.High}\right)$, which shares the same prefix (*B.Low*) with the second branch in the current fuzzy FP tree. The count of the node for *B.Low* in the second branch is then incremented by 0.6, and the new node (*C.High*) with count 0.6 is created and linked to (*B.Low*) as its child. A link is also created between the two nodes of *C.High* in different branches. The results are shown in Fig. 5.

The same process is then executed for the other transactions. After all the six transactions are processed, the resulting Header_Table and fuzzy FP tree are shown in Fig. 6.

After the fuzzy FP tree is constructed, the frequent fuzzy itemsets with more than one fuzzy region can then be found in a way similar to the FP-growth mining approach (Han et al., 2000), but much more complex due to the intersection operation in fuzzy sets. Here, the minimum operation is used for intersection. The frequent fuzzy 1-itemsets (fuzzy regions) in the Header_Table in Fig. 6 are then processed bottom-up and one by one. In this case, item *D.Low* is first processed. The corresponding conditional fuzzy pattern tree is thus built from the prefix paths of the item in the FP tree and is further used to derive the itemsets containing the item. In this example, there are two branches with item *D.Low*, which are (*A.Middle*:2.4) (*C.High*:1.4) (*D.Low*:1.4) and (*C.High*:0.8) (*D.Low*:0.8). The conditional fuzzy pattern tree built for item *D.Low* is shown in Fig. 7.

After the *conditional fuzzy pattern tree for the fuzzy region D.Low is built*, the fuzzy itemsets containing *D.Low can* then be generated by the recursive approach of the FP-growth. In fuzzy data mining, the intersection (minimum) operator is used to count fuzzy itemsets in transactions. The operator can, however, be easily implemented by the designed fuzzy FP tree since the count of an itemset in a prefix of the tree is that of the item at the bottom of the prefix. For example, (*A.Mddile*, *D.Low*), (*C.High*, *D.Low*) and (*A.Middle*, *C.High*, *D.Low*) are the possible itemsets generated from the first branch

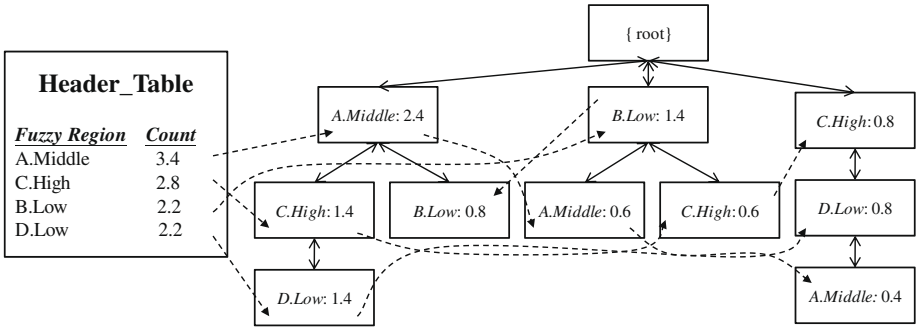

Fig. 6. The final fuzzy FP tree constructed in the example.

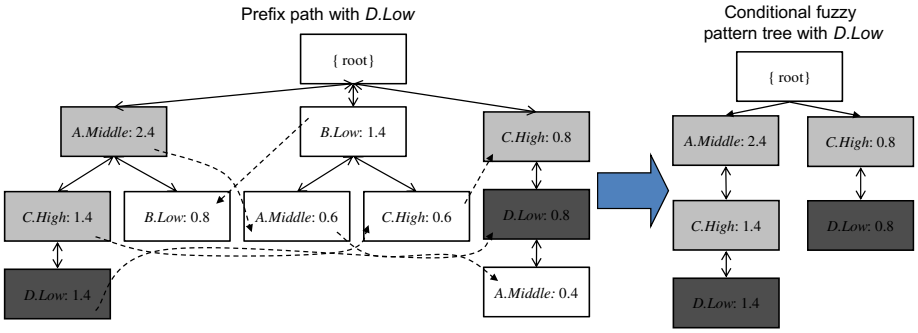

Fig. 7. The conditional fuzzy pattern tree for the fuzzy region *D.Low*.

of the conditional fuzzy pattern tree for *D.Low*. Their counts in the branch will equal to the count of *D.Low* in the branch, which is 1.4, because the fuzzy terms in the branch is sorted according to their membership values in the corresponding transactions. Similarly, the fuzzy itemset (*C.High*, *D.Low*) is generated from the second branch of the conditional fuzzy pattern tree for *D.Low* and its count is 0.8. The total count of each itemset containing *D.Low* is then calculated from the individual branches. For example, the count of the fuzzy itemset (*C.High*, *D.Low*) is 1.4 + 0.8, which is 2.2. In this way, the counts of the fuzzy itemsets can be obtained without rescan of the database.

Next, if the count of a fuzzy itemset is larger than or equal to the minimum support threshold, it will become a frequent fuzzy itemset. After *D.Low* is processed, the other frequent fuzzy regions in the Header_Table are processed one by one and bottom–up in the same way. The final results for this example are shown in Table 7.

6. Experimental results

The experiments were performed in Java on an AMD Athlon PC with a 3.0 GHz processor and 1 G main memory, running the Microsoft Windows XP operating system. A real dataset called FOODMART from an anonymous chain store was used in the experiments (Microsoft Corporation). The FOODMART dataset contained quantitative transactions about the products sold in the chain store. There were totally 21,556 transactions with 1600 items in the dataset.

In the experiments, different numbers of fuzzy regions (membership functions) for items were tested. They included one region, two regions and three regions. The purpose of using one region was for comparison with using other numbers of regions. The range of the minimum support threshold was divided into two parts due to the different scales of the resulting numbers of nodes. The first part was set at from 0.04% to 0.12% with increment 0.02% each time, and the second part was from 0.14% to 0.22% with increment 0.02% each time as well. The relationship between the number of nodes in the constructed fuzzy FP tree and the minimum support values in the two ranges were shown in Figs. 8 and 9, respectively.

From Fig. 8, it could be seen that the cross point of the three curves was at about 0.12%. When the minimum support threshold was set lower than 0.12%, the nodes generated based on three regions were more than those based on the others. On the contrary,

Table 7
All the frequent fuzzy itemsets obtained in the example.

1-itemset	2-itemset	3-itemset
(*A.Middle*:3.4) (*C.High*:2.8) (*B.Low*:2.2) (*D.Low*:2.2)	(*A.Middle*, *C.High*:1.8) (*A.Middle*, *D.Low*:1.8) (*C.High*, *D.Low*:2.2)	(*A.Middle*,*C.High*, *D.Low*: 1.8)

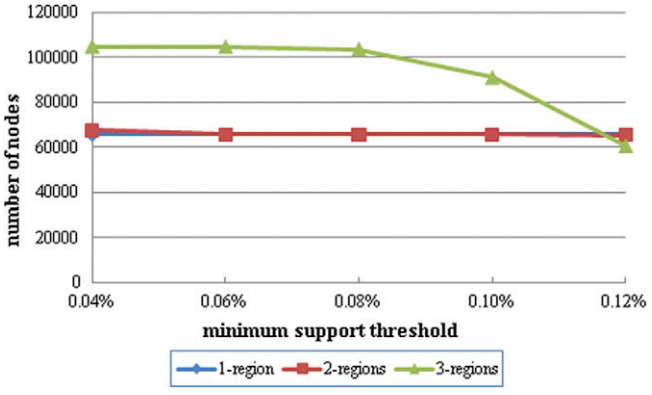

Fig. 8. The number of nodes in the fuzzy FP tree along with different minimum support values in the first part.

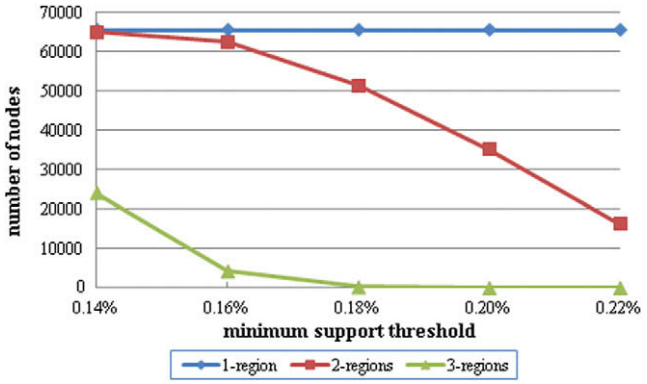

Fig. 9. The number of nodes in the fuzzy FP tree along with different minimum support values in the second part.

when the minimum support threshold was set higher than 0.12%, the nodes generated based on three regions were less than those based on the others. This was because when an item had more fuzzy regions, it might have more concentrated membership functions, thus causing a larger membership value belonging to a certain fuzzy region.

The execution time of constructing the fuzzy FP-trees along with different minimum support values in the two parts were shown in Figs. 10 and 11, respectively.

From Fig. 10, it could be observed that the execution time of constructing fuzzy FP trees decreased along with the increase of

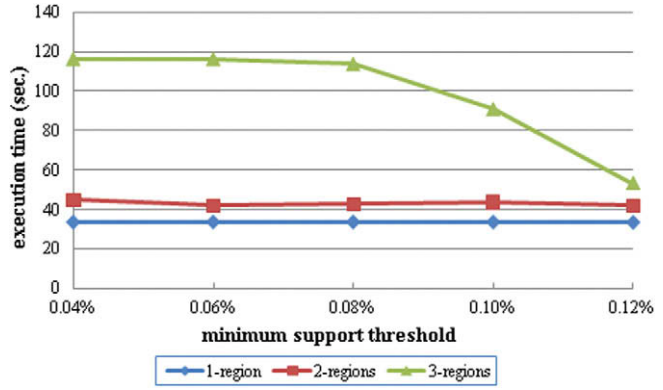

Fig. 10. The execution time along with different minimum support values in the first part.

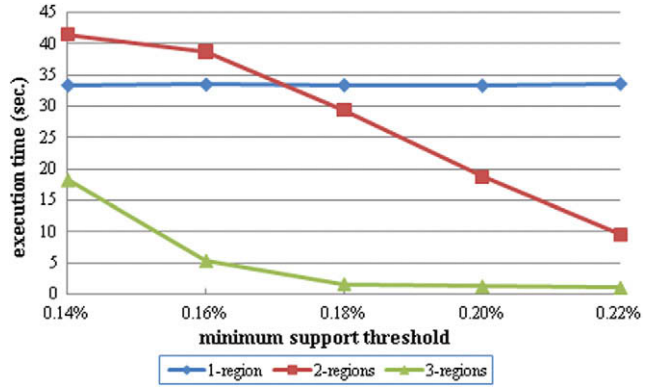

Fig. 11. The execution time along with different minimum support values in the second part.

the minimum support values for all the three different numbers of regions. It was very reasonable since less frequent fuzzy regions were generated for larger minimum support values. Besides, when more fuzzy regions were used, the algorithm did not certainly take more execution time. In Fig. 9, it could be observed that when the minimum support threshold was set lower than 0.12%, more fuzzy regions caused more execution time since more frequent fuzzy regions were generated. On the contrary, in Fig. 11, when the minimum support threshold was set higher than 0.18%, more fuzzy regions caused less execution time since less frequent fuzzy regions were generated. It was thus obvious that the number of fuzzy regions did affect the performance of the proposed algorithm.

7. Conclusion

In this paper, we have proposed the fuzzy FP-tree construction algorithm for processing transaction data with quantitative values and for mining frequent fuzzy itemsets from the transactions. The fuzzy FP-tree structure is used to efficiently and effectively handle the quantitative data. When extending the FP tree to handle fuzzy data, the processing becomes much more complex than the original. For example, the membership value of a k-itemset $(k > 1)$ in a transaction has to be derived from the membership values of the items contained in the itemset by fuzzy intersection. By the proposed fuzzy FP tree, the function can be easily achieved. The tree structure, however, becomes larger than the original since the order of fuzzy regions in a transaction must be maintained. Besides, the mining process from the tree also becomes complicated. The cost is, however, needed since more detailed knowledge is derived than that in only the binary format. Of course, as an alternative, the fuzzy regions can be converted back into crisp regions by the operation of α-cut, and then processed in the traditional FP-tree approach.

Experimental results also show that the number of fuzzy regions significantly affects the performance of the proposed algorithm. When the minimum support threshold is set lower, the larger number of fuzzy regions will generate less nodes and need less execution time. On the contrary, when the minimum support threshold is set higher, the result is inverse to the above.

In this paper, we assume the database is static. In real-world applications, data may be dynamically inserted into a database. In the future, we will attempt to handle the maintenance problem of fuzzy data mining. How to further improve the fuzzy FP tree is another interesting topic.

References

Agrawal, R. & Srikant, R. (1995). Mining sequential patterns. In *The eleventh IEEE international conference on data engineering* (pp. 3–14).

Agrawal, R., Imielinski, T., & Swami, A. (1993a). Mining association rules between sets of items in large database. In *The ACM SIGMOD international conference on management of data* (pp. 207–216).

Agrawal, R., Imielinski, T., & Swami, A. (1993b). Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 914–925.

Agrawal, R., & Srikant, R. (1994). Fast algorithm for mining association rules. *The International Conference on Very Large Data Bases*, 487–499.

Chan, C. C. & Au, W. H. (1997). Mining fuzzy association rules. In *The sixth international conference on information and knowledge management* (pp. 209–215).

Chen, M. S., Han, J., & Yu, P. S. (1996). Data mining: An overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 866–883.

Cheung, D. W., Lee, S. D., & Kao, B. (1997). A general incremental technique for maintaining discovered association rules. In *The fifth international conference on database systems for advanced application* (pp. 185–194).

Ezeife, C. I. (2002). Mining Incremental association rules with generalized FP-tree. In *The 15th conference of the Canadian society for computational studies of intelligence on advances in artificial intelligence* (pp. 147–160).

Han, J., Pei, J. & Yin, Y. (2000). Mining frequent patterns without candidate generation. In *The 2000 ACM SIGMOD international conference on management of data* (pp. 1–12).

Hong, T. P., & Chen, J. B. (1999). Finding relevant attributes and membership functions. *Fuzzy Sets and Systems*, 389–404.
Hong, T. P., & Chen, J. B. (2000). Processing individual fuzzy attributes for fuzzy rule induction. *Fuzzy Sets and Systems*, 127–140.
Hong, T. P., Kuo, C. S. & Chi, S. C. (1999a). A data mining algorithm for transaction data with quantitative values. In *The eighth international fuzzy systems association world congress* (pp. 874–878).
Hong, T. P., Kuo, C. S., & Chi, S. C. (1999b). Mining association rules from quantitative data. *Intelligent Data Analysis, 3*(5), 363–376.
Kandel, A. (1992). *Fuzzy expert systems*. Boca Raton, FL: CRC Press. pp. 8–19.
Kaya, M. & Alhajj, R. (2004). Mining multi-cross-level fuzzy weighted association rules. In *IEEE international conference intelligent systems* (pp. 225–230).
Koh, J. L. & Shieh, S. F. (2004). An efficient approach for maintaining association rules based on adjusting FP-tree structures. In *The ninth international conference on database systems for advanced applications* (pp. 417–424).
Kosko, B. (1997). *Fuzzy engineering*. Upper Saddle River, New Jersey: Prentice Hall.
Kuok, C., Fu, A. & Wong, M. (1998). Mining fuzzy association rules in databases. In *ACMSIGMOD record* (pp. 41–46).
Microsoft Corporation. Example Database FoodMart of Microsoft Analysis Services.
Papadimitriou, S. & Mavroudi, S. (2005). The frequent fuzzy pattern tree. In *The 9th WSEAS international conference on computers*.
Qiu, Y., Lan, Y. J. & Xie, Q. S. (2004). An improved algorithm of mining from FP- tree. In *The third international conference on machine learning and cybernetics* (pp. 26–29).
Shen, H., Wang, S., & Yang, J. (2004). Fuzzy taxonomic, quantitative database and mining generalized association rules. In *The 4th international conference on rough sets and current trends in computing* (pp. 610–617).
Srikant, R., Agrawal, R. (1996). Mining quantitative association rules in large relational tables. In *The ACM SIGMOD international conference on management of data* (pp. 1–12).
Yue, S., Tsand, E., Yeung, D., & Shi, D. (2000). Mining fuzzy association rules with weighted items. In *The IEEE international conference on systems, man and cybernetics* (pp. 1906–1911).
Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 338–353.
Zaiane, O. R., Mohammed, E. H. (2003). COFI-tree mining: A new approach to pattern growth with reduced candidacy generation. In *The workshop on frequent itemset mining implementations, IEEE international conference on data mining*.