Soft Comput (2009) 13:521–533
DOI 10.1007/s00500-008-0366-0

FOCUS

A genetic-fuzzy mining approach for items with multiple minimum supports

Chun-Hao Chen · Tzung-Pei Hong · Vincent S. Tseng · Chang-Shing Lee

Published online: 26 August 2008

Abstract Data mining is the process of extracting desirable knowledge or interesting patterns from existing databases for specific purposes. Mining association rules from transaction data is most commonly seen among the mining techniques. Most of the previous mining approaches set a single minimum support threshold for all the items and identify the relationships among transactions using binary values. In the past, we proposed a genetic-fuzzy data-mining algorithm for extracting both association rules and membership functions from quantitative transactions under a single minimum support. In real applications, different items may have different criteria to judge their importance. In this paper, we thus propose an algorithm which combines clustering, fuzzy and genetic concepts for extracting reasonable multiple minimum support values, membership functions and fuzzy association rules from quantitative transactions. It first uses the *k*-means clustering approach to gather similar items into groups. All items in the same cluster are considered to have similar characteristics and are assigned similar values for initializing a better population. Each chromosome is then evaluated by the criteria of requirement satisfaction and suitability of membership functions to estimate its fitness value. Experimental results also show the effectiveness and the efficiency of the proposed approach.

Keywords Data mining · Genetic-fuzzy algorithm · *k*-means · Clustering · Multiple minimum supports · Requirement satisfaction

C.-H. Chen · V. S. Tseng
Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan 701, Taiwan, ROC
e-mail: chchen@idblab.csie.ncku.edu.tw

V. S. Tseng
e-mail: tsengsm@mail.ncku.edu.tw

T.-P. Hong
Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC

C.-S. Lee
Department of Computer Science and Information Engineering, National University of Tainan, Tainan 701, Taiwan, ROC
e-mail: leecs@mail.nutn.edu.tw

T.-P. Hong (✉)
Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan, ROC
e-mail: tphong@nuk.edu.tw

1 Introduction

Data mining is commonly used for inducing association rules from transaction data. An association rule is an expression $X \rightarrow Y$, where X is a set of items and Y is a single item (Agrawal and Srikant 1994). It means in the set of transactions, if all the items in X exist in a transaction, then Y is also in the transaction with a high probability. Most previous studies focused on binary-valued transaction data. Transaction data in real-world applications, however, usually consist of quantitative values. Designing a sophisticated data-mining algorithm able to deal with various types of data presents a challenge to workers in this research field.

Fuzzy set theory has been used in intelligent systems for a long time because of its simplicity and similarity to human reasoning (Chen et al. 2000; Siler and James 2004; Zhang and Liu 2006). The theory has been applied in fields such as manufacturing, engineering, diagnosis, economics, among others (Heng et al. 2006; Ishibuchi and Yamamoto 2005; Liang et al. 2002). Several fuzzy learning algorithms for inducing rules from given sets of data have been designed and used to good effect with specific domains (Casillas et al. 2005; Hong et al. 2001; Rasmani and Shen 2004).

Most of the previous approaches set a single minimum support threshold for all the items or itemsets and identify the relationships among binary transactions. In real applications, different items may have different criteria to judge their importance and quantitative data may exist. We can thus divide the fuzzy data mining approaches into two kinds, namely single-minimum-support fuzzy-mining (SSFM) and multiple-minimum-support fuzzy-mining (MSFM) problems. Several mining approaches (Chan and Au 1997; Hong et al. 1999; Kuok et al. 1998; Yue et al. 2000) have been proposed for the SSFM problem. Chan and Au proposed an F-APACS algorithm to mine fuzzy association rules (Chan and 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. (1998) proposed a fuzzy mining approach to handle numerical data in databases and derived fuzzy association rules. At nearly the same time, Hong et al. (1999) proposed a fuzzy mining algorithm to mine fuzzy rules from quantitative transaction data. Basically, these fuzzy mining algorithms first used membership functions to transform each quantitative value into a fuzzy set in linguistic terms and then used a fuzzy mining process to find fuzzy association rules. Yue et al. (2000) then extended the above concept to find fuzzy association rules with weighted items from transaction data. They adopted Kohonen self-organized mapping to derive fuzzy sets for numerical attributes. As to the MSFM problem, Lee et al. (2004) proposed a mining algorithm which used multiple minimum supports to mine fuzzy association rules. They assumed that items had different minimum supports and the minimum support for an itemset was set as the maximum of the minimum supports of the items contained in the itemset. Under the constraint, the characteristic of level-by-level processing was kept, such that the original Apriori algorithm could easily be extended to finding large itemsets.

In the above approaches, the membership functions were assumed to be known in advance. Although many approaches for learning membership functions were proposed (Cordón et al. 2001; Mucientes et al. 2006; Roubos and Setnes 2001; Setnes and Roubos 2000; Wang et al. 2000), most of them were usually used for classification or control problems. For fuzzy mining problems, Kaya and Alhajj (2006) proposed a GA-based approach to derive a predefined number of membership functions for getting a maximum profit within an interval of user specified minimum support values. Hong et al. (2006) also proposed a genetic-fuzzy data-mining algorithm for extracting both association rules and membership functions from quantitative transactions. It maintained a population of sets of membership functions and used the genetic algorithm to automatically derive the resulting one. Hong et al. (2008) then further used the divide-and-conquer strategy to speed up the solution convergence under the strict condition of using the number of large 1-itemsets and the suitability of membership functions in fitness evaluation.

Most of the mentioned approaches were proposed for the SSFM problem. In this paper, we extend our previous approach (Hong et al. 2006) in order to solve the MSFM problem. We propose an algorithm which combines the clustering, fuzzy and genetic concepts to derive minimum support values, membership functions and association rules. The proposed approach first uses the *k*-means clustering approach to gather similar items into groups. All the items in the same cluster are considered to have similar characteristics and are assigned similar values for initializing a better population. The values include an appropriate number of linguistic terms for each item, its reasonable membership functions and possible minimum support values (more details can be found in Sect. 3.2). The proposed approach then generates and encodes each set of minimum support values and membership functions into a fixed-length string. Each chromosome is then evaluated by the criteria of requirement satisfaction and suitability of membership functions to estimate its fitness value. The proposed algorithm has two main advantages. The first one is that the proposed approach can derive an acceptable minimum support value and membership functions of each item for fuzzy association-rule mining. The second one is that the proposed approach can get a better initial population, including an appropriate number of linguistic terms and minimum support values and membership functions of items by using the clustering technique.

The remaining parts of this paper are organized as follows. The proposed genetic-fuzzy mining framework for items with multiple minimum supports is introduced in Sect. 2. The adjustment process of membership functions is explained in Sect. 3. The details of the proposed algorithm for mining multiple minimum support values, membership functions and association rules are described in Sect. 4. An example to illustrate the proposed algorithm is given in Sect. 5. Experiments to demonstrate the performance of the proposed algorithm are stated in Sect. 6. Conclusions and future works are given in Sect. 7.

2 A genetic-fuzzy mining framework for items with multiple minimum supports

In this paper, the fuzzy, the genetic and the clustering concepts are used together to discover useful fuzzy association rules, suitable minimum support values and membership functions from quantitative transactions. A genetic-fuzzy mining framework shown in Fig. 1 is first proposed for achieving the above purpose. It can be divided into two phases. The first phase searches for suitable minimum support values and membership functions of items and the second phase uses the final best set of minimum support values and membership functions to mine fuzzy association rules. The proposed framework is shown in Fig. 1.

The proposed framework maintains a population of sets of minimum support values and membership functions, and uses the genetic algorithm to automatically derive the resulting one. It first uses the *k*-means clustering approach to gather similar items into groups. All items in the same cluster are considered to have similar characteristics and are assigned similar values when a population is initialized. The values (or initialization information) include an appropriate number of linguistic terms for each item, its reasonable membership functions, and a range of its possible minimum support values. It then generates and encodes each set of minimum support values and membership functions into a fixed-length string according to the initialization information. Each chromosome is then evaluated by the requirement satisfaction and the suitability of membership functions to estimate its fitness value. The evaluation results are utilized to choose appropriate chromosomes for mating. The offspring sets of membership functions and minimum support values then undergo recursive evolution until a good set (the highest fitness

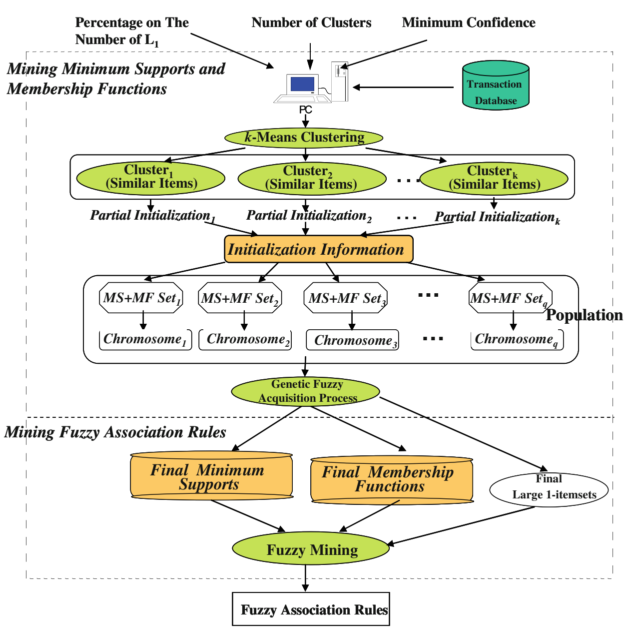

Fig. 1 The proposed genetic-fuzzy mining framework for items with multiple minimum supports

value) has been obtained. Finally, the derived minimum support values and membership functions are used to mine fuzzy association rules by the approach in Lee et al. (2004). The details are described in the next section.

3 The proposed genetic-fuzzy mining approach

3.1 Chromosome representation

It is important to encode minimum support values and membership functions as string representation for GAs to be applied to our problem. Several possible encoding approaches were described in the past (Cordón et al. 2001; Parodi and Bonelli 1993; Wang et al. 2000). In our approach, each individual consists of two parts, respectively, for minimum support values and membership functions. The first part encodes minimum support values by the real-number schema. Each real number represents the minimum support value of a certain item. Assume the minimum support value of item I_j is encoded with a real number α_j. The entire set of the minimum support values for all items is then formed by concatenating $\alpha_1, \alpha_2, \ldots, \alpha_m$ together, where m is the number of items. The second part handles the sets of membership functions for all the items. It also adopts the real-number schema. Here we assume the membership functions are isosceles-triangular for simplicity and use only two parameters to represent each membership function as Parodi and Bonelli (1993) did. Figure 2 shows the membership functions for item I_j, where R_{jk} denotes the membership function of the kth linguistic term of I_j, c_{jk} indicates the center abscissa of fuzzy region R_{jk}, and w_{jk} represents half the spread of fuzzy region R_{jk}. As Parodi and Bonelli did, we then represent each membership function as a pair (c, w). Thus, all pairs of (c, w)'s for a certain item are concatenated to represent its membership functions.

The set of membership functions MF_j for the first item I_j is then represented as a substring of $c_{j1}w_{j1} \ldots c_{jl}w_{jl}$, where l is the number of linguistic terms of I_j. The entire set of membership functions that contains m items is then encoded by concatenating substrings of $MF_1, MF_2, \ldots, MF_m$. An example is given below to demonstrate the process of

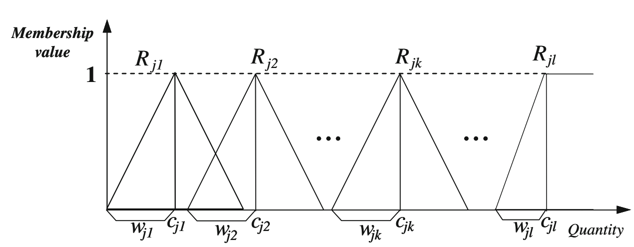

Fig. 2 Membership functions of item *Ij*

encoding a set of minimum support values and membership functions.

Example 1 Assume there are four items in a transaction database: *milk*, *bread*, *cookies* and *beverage*. Also assume the number of linguistic terms of *milk*, *bread*, *cookies* and *beverage* are 3, 3, 2 and 2, respectively. Assume there exists a chromosome shown in Fig. 3. The minimum support value and the membership functions of each item for the chromosome in Fig. 3 are shown in Fig. 4 according to the encoding scheme mentioned above.

Note that other types of membership functions (e.g. non-isosceles trapezes) can also be adopted in our approach. For coding non-isosceles triangles and trapezes, three and four points are needed instead of two for isosceles triangles. Besides, the number of fuzzy sets for each item may be different. In this paper, the *k*-means clustering approach is used to decide an appropriate number of linguistic terms for each item.

3.2 Initial population

A genetic algorithm requires a population of feasible solutions to be initialized and updated during the evolution process. As mentioned above, each individual within the population is a set of minimum support values and isosceles-triangular membership functions. Each membership function corresponds to a linguistic term of a certain item. In this paper, the initial set of chromosomes is generated according to the initialization information derived by the *k*-means clustering approach on the transactions. It includes an appropriate number of linguistic terms, the range of possible minimum support values and membership functions of items.

Let the appearing number (AN_j) of the *j*th item be the number of transactions in which the *j*th item appears. The average quantitative value (AQV_j) is the average value of the appearing quantities for the *j*th item and is defined as:

$$AQV_j = \left[\sum_{i=1}^{n} \text{Quantity}_{ij} / (AN_j) \right]$$

where n is the number of transactions and Quantity_{ij} is the quantity of the j-th item in the ith transaction. The support value (SV_j) of the jth item is the ratio of the transactions in which the jth item appears and is defined as:

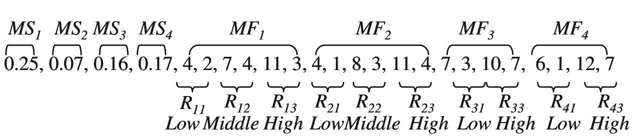

Fig. 3 A chromosome representation of minimum support values and membership functions

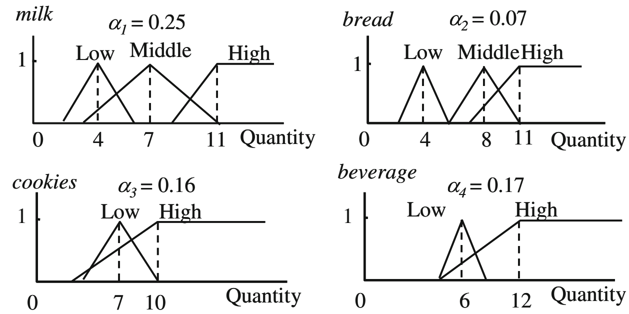

Fig. 4 The minimum support values and the membership functions for the chromosome represented in Fig. 3

$$SV_j = AN_j / n.$$

The two attributes, AQV and SV, are then used to divide items into clusters since they can reflect users' behavior. The AVQ value represents the quantity that consumers may buy for a certain item in a transaction. The SV value represents the ratio of the transactions in which a certain item appears (frequency of an item). Items in the same cluster are thus considered to have similar characteristics and are then assigned similar values for initializing a better population. The clustering procedure for generating an initial population is stated as follows.

The clustering procedure for generating an initial population:

Step 1 Calculate the average quantitative value AQV_j and the support value SV_j for each item I_j from given transactions.

Step 2 Divide the items into k clusters by the k-means clustering approach based on the two attributes (AQV, SV).

Step 3 For each cluster $cluster_g$, $g = 1$ to k, find the distribution of the quantitative values in the transactions. That is, find the appearing number of each quantitative value from the items in the same cluster. If the appearing number of a quantitative value is less than or equal to a break threshold, then it is thought of as a break point.

Step 4 For each cluster $cluster_g$, $g = 1$ to k, generate intervals according to the break points. If the total quantity in an interval is less than or equal to an interval threshold, it is removed. The number of the remaining intervals is then set as the number of linguistic terms for each item in the cluster.

Step 5 Generate the first part of a population of P individuals according to the support values of the items. That is, the minimum support of an item in an individual is randomly generated in the range between 0 and its support value.

Step 6 For each cluster $cluster_g$, $g = 1$ to k, calculate the appearing probability of each quantitative value in its corresponding interval. Restated, it is the appearing number of the quantitative value divided by the appearing number of all the quantitative values in the same interval.

After *Step 7*, an initial population of individuals can thus be generated. Below, an example is given to illustrate the above procedure.

Example 2 Assume there are four items in a transaction database: *milk*, *bread*, *cookies* and *beverage*. Also assume the data set includes the ten transactions shown in Table 1. The initial population is generated as follows:

Step 1 The average quantitative value and the support value of each item I_j are calculated from the ten transactions and represented as a pair (AQV_j, SV_j). Take the item *milk* as an example. Its average quantitative value is (6 + 7 + 2 + 3 + 6 + 10 + 11)/7, which is 6.42, and its support value is 7/10, which is 0.7. The results for all the four items are shown in Table 2.

Step 2 The *k*-means clustering approach is then executed to divide the four items into *k* clusters. In this example, assume the parameter *k* is set at 2. The two clusters found are shown in Table 3.

Step 3 The distribution of the quantitative values from the items in the same cluster is calculated. Let $q_v{}^g$ represent the appearing number of the quantitative value *v* for the *g*th cluster. Take the quantitative value "3" in $cluster_1$ as an example. It appears twice, one for milk in T5 and the other for bread in T4. $q_3{}^1$ is thus 2. The distribution of the quantitative values for the first cluster is shown in Table 4. The distribution of the quantitative values for the second cluster can be similarly found.

Assume the break threshold is set at 0. The quantitative values 1, 5, 9 and 12 are the break points. Since 1 and 12 are the minimum and the maximum values in the cluster, the actual break points are 5 and 9.

Table 1 The ten transactions in this example

TID	Items
T1	(milk, 6); (bread, 4); (cookies, 7); (beverage, 7).
T2	(milk, 7); (bread, 7); (cookies, 12).
T3	(bread, 8); (cookies, 12); (beverage, 6).
T4	(milk, 2); (bread, 3).
T5	(milk, 3); (bread, 8).
T6	(milk, 6); (beverage, 6).
T7	(milk, 10); (cookies, 6).
T8	(milk, 11); (bread, 11).
T9	(beverage, 11).
T10	(beverage, 10).

Table 2 The average quantitative value and the support value of each item

Item	(AQV_j, SV_j)	Item	(AQV_j, SV_j)
Milk	(6.42, 0.7)	Cookies	(9.25, 0.4)
Bread	(6.83, 0.6)	Beverage	(8, 0.5)

Table 3 The two clusters found from the ten chromosomes

$Cluster_i$	Items
$Cluster_1$	Milk, bread
$Cluster_2$	Cookies, beverage

Table 4 The distribution of the quantitative values for $cluster_1$

$q_v{}^g$	Value	q_v^g	Value	q_v^g	Value
q_1^1	0	q_5^1	0	q_9^1	0
q_2^1	1	q_6^1	2	q_{10}^1	1
q_3^1	2	q_7^1	2	q_{11}^1	2
q_4^1	1	q_8^1	2	q_{12}^1	0

Step 4 The intervals for each cluster is generated according to the break points. There are three intervals generated for $cluster_1$. Assume the interval threshold is set at 2. The quantities in all the three intervals are larger than the threshold and no intervals are removed. The number of the remaining intervals is still 3 and each item in $cluster_1$ is thus assigned three linguistic terms. Each item in $cluster_2$ is assigned two linguistic terms in the same way.

Step 5 The first part of a population of *P* individuals is generated according to the support values of the items. For example, the minimum support of the item milk in an individual is randomly generated in the range between 0 and 0.7.

Step 6 The appearing probability of each quantitative value in its corresponding interval is calculated. Take the quantitative value "3" in $cluster_1$ as an example. Its probability in the interval is 0.5 (=2/4). The probabilities of all the quantitative values in $cluster_1$ are shown in Table 5.

Step 7 The second part of a population of *P* individuals is generated according to the number of linguistic terms found in Step 4 and the appearing probabilities of the quantitative values of each item found in Step 6. Take the item *milk* in $cluster_1$ as an example. Since its possible minimum support range is [0, 0.7] (see Table 2), the initial minimum support value of milk is generated randomly from the range. Assume the generated value is 0.3 in this example. From Step 4, the number of membership functions is three. The procedure

Table 5 The probabilities of all the quantitative values in $cluster_1$

Pro_v^g	Value	Pro_v^g	Value	Pro_v^g	Value
Pro_1^l	0	Pro_5^l	0	Pro_9^l	0
Pro_2^l	0.25	Pro_6^l	0.33	Pro_{10}^l	0.33
Pro_3^1	0.5	Pro_7^l	0.33	Pro_{11}^l	0.66
Pro_4^1	0.25	Pro_8^l	0.33	Pro_{12}^l	0

thus first generates the center values of the three membership functions. For instance, the possible center values of the first membership function are 2, 3 and 4 (quantitative values in the first interval), which have their probabilities, 0.25, 0.5 and 0.25, respectively, (see Table 5). Assume the generated value is 4 in this example. Half the spread of the fuzzy region is then 2 (from the range [1, 4]). The center values and half the spread of the second and the third membership functions are generated as 7, 4 and 11, 3 in the same way. An individual for items milk, bread, cookies and beverage is thus generated as follows:

C_1: 0.3, 0.07, 0.16, 0.17, 4, 2, 7, 4, 11, 3, 4, 1, 8, 3, 11, 4, 7, 3, 10, 7, 6, 1, 12, 7, where the first four numbers represent the minimum supports of the four items and the others represent the membership functions.

3.3 The required number of large 1-itemsets

In our approach, the minimum support values of the items may be different. It is hard to assign the values. As an alternative, the values can be determined according to the required number of rules. It is, however, very time-consuming to obtain the rules for each chromosome. Usually, a larger number of 1-itemsets will result in a larger number of all itemsets with a higher probability, which will thus usually imply more interesting association rules. The evaluation by 1-itemsets is faster than that by all itemsets or interesting association rules. Using the number of large 1-itemsets can thus achieve a trade-off between execution time and rule interestingness (Hong et al. 2006). Of course, the criterion can also be based on the number of all large itemsets or all association rules. Their relationship for single minimum supports was discussed in (Chen et al. 2007).

A criterion should thus be specified to reflect the user preference on the derived knowledge. In this paper, the required number of large 1-itemsets (*RNL*) is proposed for this purpose. It is the number of large 1-itemsets that a user wants to get from an item and can be defined as follows:

$$RNL_j = \lfloor l_j^* p \rfloor,$$

where l_j is the number of linguistic terms of item I_j and p is the predefined percentage to reflect users' preference on the number of large 1-itemsets. The minimum support value from which the number of large 1-itemsets for an item is close to its *RNL* value is thought of as a good one. For example, assume there are three linguistic terms for an item and the predefined percentage p is set at 80%. The *RNL* value is then set as $\lfloor 3*0.8 \rfloor$, which is 2. *RNL* is thus used in the fitness function described in the next section to evaluate the goodness of a chromosome.

3.4 Fitness and selection

In order to develop a good set of minimum support values and membership functions from an initial population, the genetic algorithm selects parent chromosomes for mating in an appropriate way. In this paper, the elitism selection strategy or the roulette selection strategy can be used. An evaluation function is thus used to qualify the derived minimum support values and membership functions. The fitness function of a chromosome C_q is defined as follows:

$$f(C_q) = \frac{RS(C_q)}{\text{Suitability}(C_q)},$$

where $RS(C_q)$ is the requirement satisfaction defined as the closeness of the number of derived large 1-itemsets for chromosome C_q to its *RNL*, suitability(C_q) represents the suitability of the membership functions for C_q. $RS(C_q)$ is defined as follows:

$$RS(C_q) = \sum_{j=1}^{m} Rs(C_{qj}),$$

where m is the number of items and $RS(C_{qj})$ represents the closeness of the number of derived linguistic large 1-itemsets for the jth item in chromosome C_q to its *RNL*. $RS(C_{qj})$ is defined as follows:

$$RS(C_{qj}) = \begin{cases} \frac{|L_1^j|}{RNL_j} & \text{if } |L_1^j| \le RNL_j; \\ \frac{RNL_j}{|L_1^j|} & \text{if } RNL_j < |L_1^j|; \end{cases}$$

where RNL_j is the required number of large 1-itemsets for item j and $|L_1^j|$ is the number of derived large 1-itemsets. $RS(C_{qj})$ is used to reflect the closeness degree between the number of derived large 1-itemsets and the required number of large 1-itemset.

Suitability(C_q) represents the shape suitability of the membership functions from C_q and is defined as follows:

$$\sum_{j=1}^{m} overlap\_factor(C_{qj}) + \left(\sum_{j=1}^{m} coverage\_factor(C_{qj})/NumItem \right),$$

where m is the number of items, $overlap\_factor(C_{qj})$ represents the overlapping factor of the membership functions for an item I_j in the chromosome C_q, *coverage_factor*(C_{qj}) represents the coverage ratio of the membership functions for I_j, and *NumItem* (=*m*) is the number of items in the dataset. *overlap_factor*(C_{qj}) is the same as that in Hong et al. (2006) and defined as follows:

$$overlap\_factor(C_{qj}) = \sum_{k \neq i} \left[\max\left(\left(\frac{overlap(R_{jk}, R_{ji})}{\min(w_{jk}, w_{ji})} \right), 1 \right) - 1 \right],$$

where *overlap*(R_{jk}, R_{ji}) is the overlap length of R_{jk} and R_{ji}. *coverage_factor*(C_{qj}) represents the coverage ratio of a set of membership functions for an item I_j and is defined as:

$$coverage\_factor(C_{qj}) = \frac{1}{\frac{range(R_{j1},\ldots,R_{jl})}{\max(I_j)}},$$

where *range*(R_{j1}, R_{j2},…, R_{jl}) is the coverage range of the membership functions, l is the number of membership functions for I_j, and max(I_j) is the maximum quantity of I_j in the transactions.

The suitability factor used in the fitness function can reduce the occurrence of the two bad kinds of membership functions shown in Fig. 5, where the first one is too redundant, and the second one is too separate.

The overlap factor in *suitable*(C_q) is designed for avoiding the first bad case, and the coverage factor is for the second one. Note that a lower suitability value represents a set of better membership functions. Below, an example is given to illustrate the above idea.

Example 3 Continue Example 1, assume the maximum quantitative values of the four items are all 12. The suitability of the chromosome C_1 is computed as follows:

$$\text{Suitability}(C_1) = [0.5 + 0.33 + 1 + 1] + [1.5/4 + 1.33/4 + 1.33/4 + 1.7/4] = 4.22.$$

3.5 Genetic operators

Genetic operators are very important to the success of specific GA applications. Two genetic operators, the max–min–arithmetical (MMA) crossover proposed in Herrera et al. (1997) and the *one-point mutation*, are used in the proposed genetic-fuzzy mining framework. Assume there are two parent chromosomes:

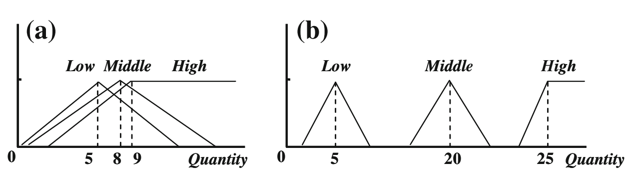

Fig. 5 Two bad membership functions

$C_u{}^t$ = (c$_1$,…, c_h,…, c_Z), and $C_w{}^t$ = (c_1',…, c_h',…, c_Z').

The max–min–arithmetical (MMA) crossover operator will generate the following four candidate chromosomes from them:

1. $C_1{}^{t+1}$ = (c_{11}^{t+1},… c_{1h}^{t+1},…, c_{1Z}^{t+1}), where $c_{1h}^{t+1} = dc_h + (1-d)c_h'$,
2. $C_2{}^{t+1}$ = (c_{21}^{t+1},…, c_{2h}^{t+1},…, c_{2Z}^{t+1}), where $c_{2h}^{t+1} = dc_h' + (1-d)c_h$,
3. $C_3{}^{t+1}$ = (c_{31}^{t+1},…, c_{3h}^{t+1},…, c_{3Z}^{t+1}), where $c_{3h}^{t+1} = \min\{c_h, c_h'\}$,
4. $C_4{}^{t+1}$ = (c_{41}^{t+1},…, c_{4h}^{t+1},…, c_{4Z}^{t+1}), where $c_{4h}^{t+1} = \max\{c_h, c_h'\}$,

where the parameter d is either a constant or a variable whose value depends on the age of the population. The best two chromosomes of the four candidates are then chosen as the offspring.

The one-point mutation operator will add a random value ω to the minimum support value α_j in a chromosome. The newly derived minimum support value will thus be changed to $\alpha_j \pm \omega$. A new fuzzy membership function will also be created by addition of a random value ε to the center or to the spread of an existing linguistic term, say R_{jk}. Assume that c and w represent the center and the spread of R_{jk}. The center or the spread of the newly derived membership function will be changed to $c \pm \varepsilon$ or $w \pm \varepsilon$ by the mutation operation. Mutation at the center of a fuzzy membership function may, however, disrupt the order of the resulting fuzzy membership functions. These fuzzy membership functions then need rearrangement according to their center values. For example, assume the membership functions of *milk* are 4, 2, 7, 4, 11, 3. Also assume the mutation point is set at the first value “4” and the random value ε is set at 5. After mutation, the membership functions became 9, 2, 7, 4, 11, 3. The order of the first two center values, 9 and 7, are disrupted in the resulting membership functions. After rearrangement, the membership functions become 7, 2, 9, 4, 11, 3.

4 The proposed mining algorithm

According to the above description, the proposed genetic-fuzzy mining algorithm for mining minimum support values, membership functions and fuzzy association rules is described below.

The proposed genetic-fuzzy mining algorithm for items with multiple minimum supports:

INPUT: A body of n quantitative transactions, a set of m items, a parameter k for k-means clustering, a population size P, a crossover rate P_c, a mutation rate P_m, a

crossover parameter d, a percentage of required number of large 1-itemsets p, a break threshold, an interval threshold, and a confidence threshold λ.
OUTPUT: A set of fuzzy association rules with its associated set of minimum support values and membership functions.
Step 1 Generate a population of P individuals by the clustering procedure stated in Sect. 3; each individual is a set of minimum support values and membership functions for all the m items.
Step 2 Calculate the fitness value of each chromosome by the following substeps:
Substep 2.1 For each transaction datum D_i, i = 1 to n, and for each item I_j, j = 1 to m, transform the quantitative value $v_j^{(i)}$ into a fuzzy set $f_{jk}^{(i)}$ represented as:

$$\left(\frac{f_{j1}^{(i)}}{R_{j1}}+\frac{f_{j2}^{(i)}}{R_{j2}}+\ldots+\frac{f_{jl}^{(i)}}{R_{jl}}\right),$$

using the corresponding membership functions represented by the chromosome, where R_{jk} is the kth fuzzy region (term) of item I_j, $f_{jl}^{(i)}$ is $v_j^{(i)}$'s fuzzy membership value in region R_{jk}, and l (=|I_j|) is the number of linguistic terms for I_j.
Substep 2.2 For each item region R_{jk}, $1 \le j \le m$, calculate its scalar cardinality on the transactions as follows:

$$\text{count}_{jk} = \sum_{i=1}^{n} f_{jk}^{(i)}.$$

Substep 2.3 For each R_{jk}, $1 \le j \le m$ and $1 \le k \le l$, check whether its count$_{jk}$ is larger than or equal to the minimum support value represented in the chromosome. If R_{jk} satisfies the above condition, put it in the set of large 1-itemsets (L_1). That is:

$$L_1 = \left\{R_{jk}|\text{count}_{jk} \ge \alpha_j, 1 \le j \le m \text{ and } 1 \le k \le 1\right\}.$$

Substep 2.4 Set the fitness value of the chromosome as the requirement satisfaction ($RS(C_q)$) divided by *suitability*(C_q). That is:

$$f(C_q) = \frac{RS(C_q)}{\text{suitability}(C_q)}.$$

Step 1 Execute the crossover operation on the population.
Step 2 Execute the mutation operation on the population.
Step 3 Use the selection operation to choose appropriate individuals for the next generation.
Step 4 If the termination criterion is not satisfied, go to Step 2; otherwise, do the next step.
Step 5 Get the set of minimum support values and membership functions with the highest fitness value.
Step 6 Mine fuzzy association rules using the set of minimum support values and membership functions.

The set of minimum support values and membership functions are thus used to mine fuzzy association rules from the given database. The fuzzy mining algorithm proposed in Lee et al. (2004) is then adopted to achieve this purpose. All the large 1-itemsets L_1 have been found previously from the process of finding the minimum supports and the membership functions. Candidate 2-itemsets C_2 can then be formed from L_1. The minimum support for an itemset is set as the maximum of the minimum supports of the items contained in the itemset. Under the constraint, the characteristic of level-by-level processing is kept, such that the original Apriori algorithm can be easily extended to find the large itemsets.

5 An example

In this section, a simple example is given to illustrate the proposed genetic-fuzzy mining algorithm for finding minimum support values, membership functions and fuzzy association rules. Assume the items and the data set are the same as before. The proposed genetic-fuzzy mining algorithm proceeds as follows.

Step 1 P individuals are first generated as the initial population by the clustering procedure. Assume P is set at 10. Each individual is a set of minimum support values and membership functions for all the four items: *milk*, *bread*, *cookies* and *beverage*. Each item is expected to have a good initial minimum support value, an appropriate number of linguistic terms, and suitable membership functions. Assume ten individuals are generated and the first one is shown below:
C_1: 0.25, 0.07, 0.16, 0.17, 4, 2, 7, 4, 11, 3, 4, 1, 8, 3, 11, 4, 7, 3, 10, 7, 6, 1, 12, 7;
Step 2 The fitness value of each chromosome is calculated. The quantitative value of each transaction datum is transformed into a fuzzy set according to the membership functions in each chromosome. Take the first item in transaction *T6* using the membership functions in chromosome C_1 as an example. The membership functions for *milk* in C_1 are represented as (4, 2, 7, 4, 11, 3), which are shown in Fig. 6.

The amount "6" of item *milk* is then converted into the fuzzy set (0.75/*milk.Middle*) using the membership functions for *milk* in C_1. The results for all the items are shown in Table 6, where the notation *item.term* is called a fuzzy region.

The scalar cardinality of each fuzzy region in the transactions is calculated as the *count* value. The count of

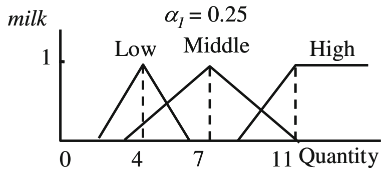

Fig. 6 The membership functions for milk in C_1

any fuzzy region is then checked against the minimum support value in C_1. The minimum support values of the four items *milk*, *bread*, *cookies* and *beverage* are 0.25, 0.07, 0.16 and 0.17, respectively. Take *milk* as an example. Its minimum support is 0.25. Since the count value of *milk.Middle* is larger than 2.5 (=0.25*10), it is put in L_1. The results for the other regions are shown in Table 7.

Assume the percentage p of the required number of large 1-itemsets is set at 0.8. The *RNL* values of the four items *milk*, *bread*, *cookies* and *beverage* are 2 (=$\lfloor 3*0.8 \rfloor$), 2 (=$\lfloor 3*0.8 \rfloor$), 1 (=$\lfloor 2*0.8 \rfloor$) and 1 (=$\lfloor 2*0.8 \rfloor$), respectively. The numbers of L_1 of the four items *milk*, *bread*, *cookies* and *beverage* are 1, 3, 2 and 2, as shown in Table 7. The requirement satisfaction of *milk*, *bread*, *cookies* and *beverage* are thus 0.5 (=1/2), 0.66 (=2/3), 0.5 (=1/2) and 0.5 (=1/2). Since the requirement satisfaction of C_1 is 2.16 (=0.5 + 0.66 + 0.5 + 0.5) and its suitability is calculated as 4.22, the fitness value of C_1 is thus 2.16/4.22 (=0.51).

Table 6 The fuzzy sets transformed from the data in Table 1

TID	Fuzzy Set
T1	$\left(\frac{0.75}{milk.Middle}\right)\left(\frac{1.0}{bread.Low}\right)$
T2	$\left(\frac{1.0}{cookies.Low}+\frac{0.571}{cookies.High}\right)\left(\frac{0.285}{beverage.High}\right)$
	$\left(\frac{1.00}{milk.Middle}\right)\left(\frac{0.66}{bread.Low}\right)\left(\frac{1.0}{cookies.High}\right)$
T3	$\left(\frac{1}{bread.Middle}+\frac{0.25}{bread.High}\right)$
T4	$\left(\frac{1}{cookies.High}\right)\left(\frac{1.0}{beverage.Low}+\frac{0.142}{beverage.High}\right)$
	$\left(\frac{0.0}{milk.Low}\right)\left(\frac{0.0}{bread.Low}\right)$
T5	$\left(\frac{0.5}{milk.Low}\right)\left(\frac{1.0}{bread.Middle}+\frac{0.25}{bread.High}\right)$
T6	$\left(\frac{0.75}{milk.Middle}\right)\left(\frac{1.0}{beverage.Low}+\frac{0.142}{beverage.High}\right)$
T7	$\left(\frac{0.25}{milk.Middle}+\frac{0.666}{milk.High}\right)\left(\frac{0.666}{cookies.Low}+\frac{0.428}{cookies.High}\right)$
T8	$\left(\frac{1.0}{milk.High}\right)\left(\frac{1.0}{bread.High}\right)$
T9	$\left(\frac{0.857}{beverage.High}\right)$
T10	$\left(\frac{0.714}{beverage.High}\right)$

Table 7 The set of large 1-itemsets (L_1) in this example

Itemset	Count	Itemset	Count
milk.Middle	2.75	*cookies.Low*	1.66
bread.Low	1.0	*cookies.High*	2.99
bread.Middle	2.66	*beverage.Low*	2.00
bread.High	1.5	*beverage.High*	2.14

The fitness values of all the other chromosomes can be similarly calculated.

Steps 3 to 8: The crossover and the mutation operations are executed on the population to generate possible offspring. The elitist selection operation is executed to generate ten chromosomes as the next population. The same procedure is then executed until the termination criterion is satisfied. The best chromosome (with the highest fitness value) is output as the minimum support values and membership functions for deriving fuzzy association rules. After the minimum support values and membership functions are derived, the fuzzy mining method proposed in Lee et al. (2004) is used to mine fuzzy association rules.

6 Experimental results

In this section, experiments made to show the performance of the proposed approach are described. They were implemented in Java on a personal computer with Intel Pentium IV 3.20 GHz and 512 MB RAM. 64 items and 10,000 transactions were used in the experiments. The initial population size P is set at 50, the cluster number k is set at 10, the interval threshold is set at 30%, the break threshold is set at 5%, the crossover rate p_c is set at 0.8, and the mutation rate p_m is set at 0.001 (for each gene). The parameter d of the crossover operator is set at 0.35 according to Herrera et al.'s paper (1997). The percentage of the required number of large 1-itemsets is set at 0.8. In the following subsections, we first give a description of the experimental datasets. We then analyze the performance of the proposed approach according to the fitness function.

6.1 Description of the experimental datasets

Two Simulated datasets with 64 items and with 10,000 transactions were used in the experiments. One dataset follows uniform distribution and another one follows exponential distribution. The two datasets are described as follows. The factors for the two datasets included the transaction length, the purchased items and their quantities. In the experiments, the number (transaction length) of purchased items in a transaction was randomly generated in a uniform distribution of the range [1, 19] for both two datasets. The purchased items in each transaction were then selected from the 64 items in a uniform distribution of the range [1, 64] and in an exponential distribution with the rate parameter set at 16. Their quantities were then assigned from a uniform distribution of the range [1, 11] and from an exponential distribution with the rate parameter set at 5. The simulation process was terminated until the dataset size was reached. An item could not be generated twice in a transaction.

6.2 The performance of the proposed approach

After 500 generations, the final membership functions are apparently much better than the original ones in both the datasets. For example, the initial minimum support values and membership functions of some two items among the 64 items are shown in Fig. 7a.

After 500 generations, the final minimum support values and membership functions for the same items are shown in Fig. 7b. It is easily seen that the membership functions in Fig. 7b is better than those in Fig. 7a. The two bad kinds of membership functions are improved in the final results. The same results can also be derived for the exponential dataset. The initial minimum support values and membership functions of some two items among the 64 items are shown in Fig. 8.

The average fitness values of the chromosomes along with different numbers of generations of the proposed approach for the two datasets are shown in Figs. 9 and 10. The average fitness value is calculated by the following formula:

$$avgFitness = \sum_{q=1}^{p} f(C_q)/p,$$

where P is the size of a population, and $f(C_q)$ is the fitness value of chromosome C_q. As expected, the curves for the two datasets gradually go upward, finally converging to a fixed value.

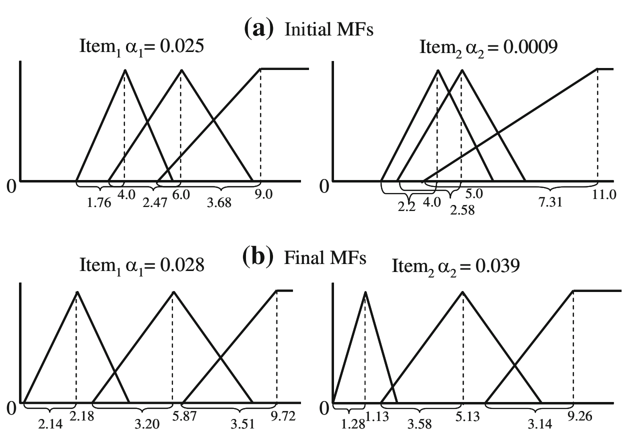

Fig. 7 The initial and final minimum support values and membership functions of some items for a uniform dataset

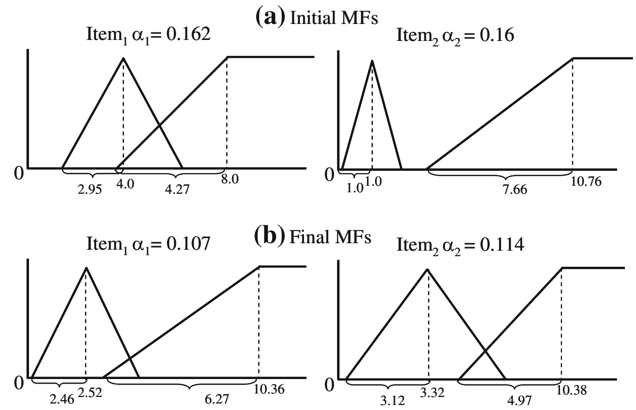

Fig. 8 The initial and final minimum support values and membership functions of some items for exponential dataset

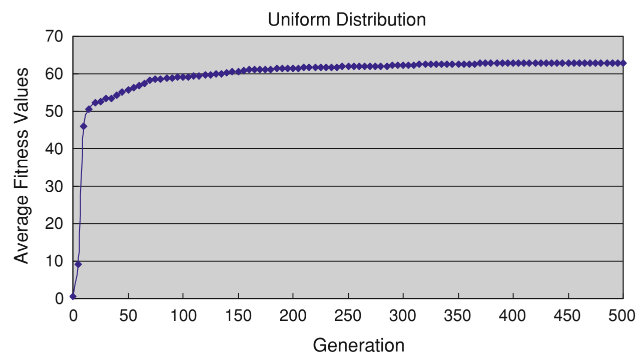

Fig. 9 The average fitness values along with different numbers of generations for uniform dataset

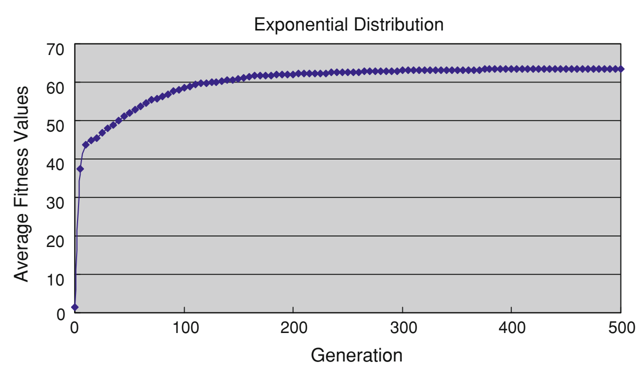

Fig. 10 The average fitness values along with different numbers of generations for exponential dataset

Next, experiments were made to verify the effect of applying the clustering approach to generate an initial population. The experimental results of the proposed approach with and without the partial information set for the two data distributions are shown in Figs. 11 and 12, respectively.

From Figs. 11 and 12, it can be observed that the average fitness values of the proposed approach with the partial information set was better than that without the partial information set for both the two datasets. The results thus showed that the initial knowledge obtained by means of the clustering strategy is actually useful.

Experiments were made to comparing the proposed approach and Lee et al.'s approach (2004) to show the advantages of the former. In Lee et al.'s approach, three uniform fuzzy partitions were used for each item and its minimum support value was set at 3%. The parameter *p* is the percentage of the required number of large 1-itemsets. The numbers of fuzzy rules by the two approaches on the uniform and the exponential datasets are shown in Figs. 13 and 14.

From Figs. 13 and 14, it is easily observed that the derived numbers of fuzzy association rules decreased along with the increase of the minimum confidence value in both the approaches. The results were obvious according to the principles of data mining. Besides, the numbers of fuzzy association rules derived by the proposed approach very depended on the value of the parameter *p,* which was used to control the desired ratio of large 1-itemsets. As can be seen from the two figures, a larger *p* value would result in more association rules. This was just what we expected.

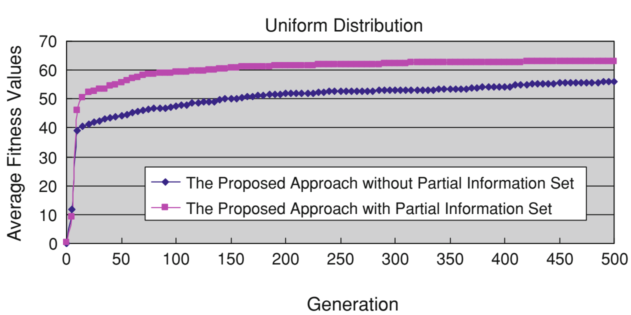

Fig. 11 The comparison results of the proposed approach with and without the partial information set for the uniform dataset

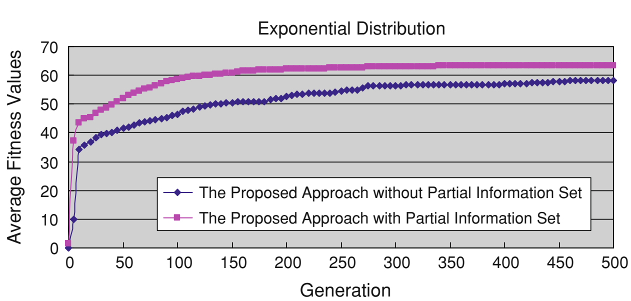

Fig. 12 The comparison results of the proposed approach with and without the partial information set for the exponential dataset

Besides, in Lee et al.'s approach, the minimum support values for all the items were unknown and thus fixed at 3% in the experiments, but the values for all the items can be found by our approaches. The proposed approach is thus more flexible than Lee et al's approach. Note that the numbers of association rules in the two approaches do not have a certain relationship since the rule number will depend on *p* in our approach and depends on the given minimum supports in Lee et al.'s approach. The derived minimum supports and membership functions of items by the proposed approach can more reflect users' preferences than those by Lee et al.'s. In order to further show this point, the comparison of the fitness values between the proposed approach and Lee et al.'s are shown in Tables 8 and 9. In Lee et al.'s approach, three uniform fuzzy partitions were used for each item, and the minimum support values derived by the proposed approach were used for items. From Tables 8 and 9, it can be observed that the proposed approach can get a better fitness value than Lee et al.'s. The proposed approach can thus more reflect users' preferences than Lee et al.'s.

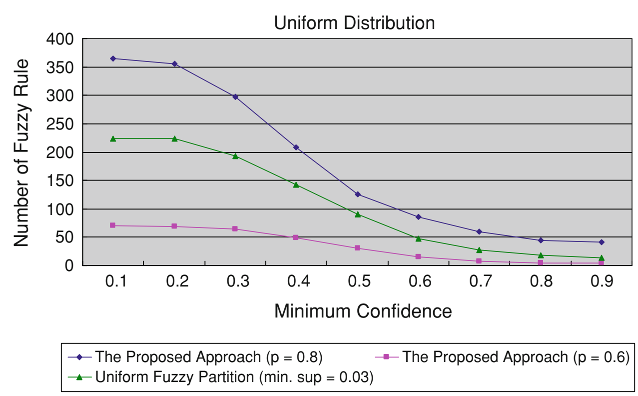

Fig. 13 The comparison results of the proposed approach and the previous fuzzy mining approach for the uniform dataset

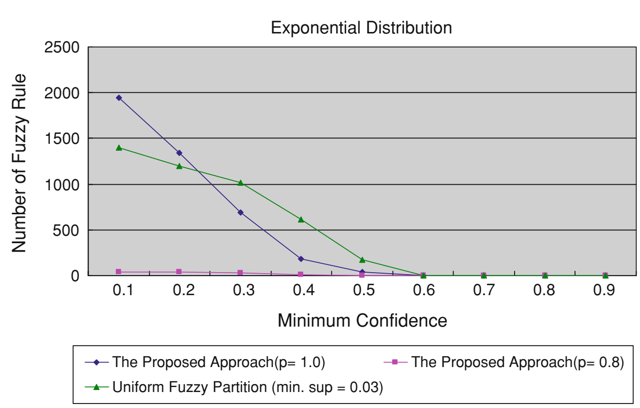

Fig. 14 The comparison results of the proposed approach and the previous fuzzy mining approach for the exponential dataset

Table 8 The comparison of the fitness values between the proposed approach and Lee et al.'s for the uniform dataset

	p = 0.6	p = 0.8
The proposed approach	61.24	62.98
Lee et al.'s approach	43.00	47.99

Table 9 The comparison of the fitness values between the proposed approach and Lee et al.'s for the exponential dataset

	p = 0.8	p = 1.0
The proposed approach	61.81	63.49
Lee et al.'s approach	33.66	49.16

7 Conclusions

In this paper, we have proposed a genetic-fuzzy mining algorithm for extracting multiple minimum support values, membership functions and fuzzy association rules from quantitative transactions. The proposed algorithm can adjust the minimum support value and membership functions for each item by genetic algorithms and use them to fuzzify quantitative transactions. In the evolution process, each chromosome represents a possible set of minimum support values and membership functions used in fuzzy mining. The proposed approach first uses the *k*-means clustering approach to gather similar items into groups. All items in the same cluster are considered to have similar characteristics and are assigned similar values when a population is initialized. Each chromosome is then evaluated by the requirement satisfaction and the suitability of membership functions to estimate its fitness value. Compared to the previous approaches, the proposed one can handle the MSFM problem. It has the following two main advantages. The first one is that the proposed approach can derive an acceptable minimum support value and membership functions of each item for fuzzy association-rule mining. The second one is that the proposed approach can get a better initial population, including an appropriate number of linguistic terms and the minimum support value and membership functions of each item by using the clustering technique. Experimental results also show that the effectiveness of the adopted clustering technique and the fitness function.

In the future, we will continuously attempt to enhance the genetic-fuzzy mining framework for more complex problems. We will also discuss the issue of optimizing membership functions and association rules at the same time. Many works have been done in the direction for control, modeling and classification. Maybe we could provide some relevance in the methodological point of view although the task of mining fuzzy association rules is a little different.

Acknowledgments This research was supported by the National Science Council of the Republic of China under contract NSC 96–2213-E-390-003.

References

Agrawal R, Srikant R (1994) Fast algorithm for mining association rules. In: The international conference on very large databases, pp 487–499

Chan CC, Au WH (1997) Mining fuzzy association rules. In: The conference on information and knowledge management, pp 209–215

Casillas J, Cordón O, del Jesus MJ, Herrera F (2005) Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction. IEEE Trans Fuzzy Syst 13(1):13–29

Chen J, Mikulcic A, Kraft DH (2000) An integrated approach to information retrieval with fuzzy clustering and fuzzy inferencing. In: Pons O, Vila MA, Kacprzyk J (eds) Knowledge management in fuzzy databases. Physica-Verlag, Heidelberg

Chen CH, Hong TP, Tseng VS (2007) A comparison of different fitness functions for extracting membership functions used in fuzzy data mining. In: IEEE symposium on foundations of computational intelligence, pp 550–555

Cordón O, Herrera F, Villar P (2001) Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base. IEEE Trans Fuzzy Syst 9(4):667–674

Herrera F, Lozano M, Verdegay JL (1997) Fuzzy connectives based crossover operators to model genetic algorithms population diversity. Fuzzy Sets Syst 92(1):21–30

Hong TP, Chen CH, Wu YL, Lee YC (2006) A GA-based fuzzy mining approach to achieve a trade-off between number of rules and suitability of membership functions. Soft Comput 10(11):1091–1101

Hong TP, Chen CH, Lee YC, Wu YL (2008) Genetic-fuzzy data mining with divide-and-conquer strategy. IEEE Trans Evol Comput (Accepted and to appear)

Hong TP, Kuo CS, Chi SC (1999) Mining association rules from quantitative data. Intell Data Anal 3(5):363–376

Hong TP, Kuo CS, Chi SC (2001) Trade-off between time complexity and number of rules for fuzzy mining from quantitative data. Int J Uncertain Fuzziness Knowl Based Syst 9(5):587–604

Heng PA, Wong TT, Rong Y, Chui YP, Xie YM, Leung KS, Leung PC (2006) Intelligent inferencing and haptic simulation for Chinese acupuncture learning and training. IEEE Trans Inf Technol Biomed 10(1):28–41

Ishibuchi H, Yamamoto T (2005) Rule weight specification in fuzzy rule-based classification systems. IEEE Trans Fuzzy Syst 13(4):428–435

Kaya M, Alhajj R (2006) Utilizing genetic algorithms to optimize membership functions for fuzzy weighted association rules mining. Appl Intell 24(1):7–15

Kuok C, Fu A, Wong M (1998) Mining fuzzy association rules in databases. SIGMOD Record 27(1):41–46

Lee YC, Hong TP, Lin WY (2004) Mining fuzzy association rules with multiple minimum supports using maximum constraints. Lect Notes Comput Sci 3214:1283–1290

Liang H, Wu Z, Wu Q (2002) A fuzzy based supply chain management decision support system. World Congr Intell Control Autom 4:2617–2621

Mucientes M, Moreno DL, Bugarin A, Barro S (2006) Evolutionary learning of a fuzzy controller for wallfollowing behavior in mobile robotics. Soft Comput 10(10):881–889

Parodi A, Bonelli P (1993) A new approach of fuzzy classifier systems. In: The fifth international conference on genetic algorithms, Morgan Kaufmann, Los Altos, pp 223–230

Roubos H, Setnes M (2001) Compact and transparent fuzzy models and classifiers through iterative complexity reduction. IEEE Trans Fuzzy Syst 9(4):516–524

Rasmani KA, Shen Q (2004) Modifying weighted fuzzy subsethood-based rule models with fuzzy quantifiers. IEEE Int Conf Fuzzy Syst 3:1679–1684

Siler W, James J (2004) Fuzzy expert systems and fuzzy reasoning. Wiley, New York

Setnes M, Roubos H (2000) GA-fuzzy modeling and classification: complexity and performance. IEEE Trans Fuzzy Syst 8(5):509–522

Wang CH, Hong TP, Tseng SS (2000) Integrating membership functions and fuzzy rule sets from multiple knowledge sources. Fuzzy Sets Syst 112:141–154

Yue S, Tsang 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

Zhang H, Liu D (2006) Fuzzy modeling and fuzzy control, Springer, Heidelberg