Soft Comput (2011) 15:2319–2333
DOI 10.1007/s00500-010-0664-1

FOCUS

Genetic-fuzzy mining with multiple minimum supports based on fuzzy clustering

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

Published online: 19 November 2010

Abstract Data mining is the process of extracting desirable knowledge or interesting patterns from existing databases for specific purposes. Most of the previous approaches set a single minimum support threshold for all the items and identify the relationships among transactions using binary values. In real applications, different items may have different criteria to judge their importance. In the past, we proposed an algorithm for extracting appropriate multiple minimum support values, membership functions and fuzzy association rules from quantitative transactions. It used requirement satisfaction and suitability of membership functions to evaluate fitness values of chromosomes. The calculation for requirement satisfaction might take a lot of time, especially when the database to be scanned could not be totally fed into main memory. In this paper, an enhanced approach, called the fuzzy cluster-based genetic-fuzzy mining approach for items with multiple minimum supports (FCGFMMS), is thus proposed to speed up the evaluation process and keep nearly the same quality of solutions as the previous one. It divides the chromosomes in a population into several clusters by the fuzzy *k*-means clustering approach and evaluates each individual according to both their cluster and their own information. Experimental results also show the effectiveness and the efficiency of the proposed approach.

Keywords Data mining · Fuzzy set · Genetic algorithm · Genetic-fuzzy mining · Fuzzy *k*-means · Clustering · Multiple minimum supports

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. 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 (Agrawal and Srikant 1994). 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; William Siler and James 2004; Zhang and Liu 2006). The theory has been applied in fields such as manufacturing, engineering,

This is a modified and expanded version of the paper "Speeding up genetic-fuzzy mining by fuzzy clustering," The IEEE International Conference on Fuzzy Systems, pp. 1695–1699, 2009.

C.-H. Chen
Department of Computer Science and Information Engineering,
Tamkang University, Taipei 251, Taiwan, ROC
e-mail: chchen@mail.tku.edu.tw

T.-P. Hong (✉)
Department of Computer Science and Information Engineering,
National University of Kaohsiung, Kaohsiung 811,
Taiwan, ROC
e-mail: tphong@nuk.edu.tw

T.-P. Hong
Department of Computer Science and Engineering,
National Sun Yat-sen University, Kaohsiung 804, Taiwan, ROC

V. S. Tseng
Department of Computer Science and Information Engineering,
National Cheng-Kung University, Tainan 701, Taiwan, ROC
e-mail: tsengsm@mail.ncku.edu.tw

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 and Lee 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; Fu et al. 1998; Hong et al. 1999, 2001; Kuok et al. 1998; Mohamadlou et al. 2009; Mangalampalli and Pudi 2009; Ouyang and Huang 2009; 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 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 aforementioned approaches, the membership functions were assumed to be known in advance. Although many approaches for learning membership functions were proposed (Cordón et al. 2001; Roubos and Setnes 2001; Setnes and Roubos 2000; Wang et al. 1998, 2000), most of them were usually used for classification or control problems. For fuzzy mining problems, Kaya et al. 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 (Kaya and Alhajj 2005). 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. Its fitness function considered the number of large 1-itemsets and the suitability of membership functions. The suitability measure was used to reduce the occurrence of bad types of membership functions. Other modified approaches based on Hong's approach can also be found in (Alcala-Fdez et al. 2009; Hong et al. 2008).

Most of the mentioned approaches were proposed for the SSFM problem. Chen et al. thus proposed a genetic approach to solve the MSFM problem (Chen et al. 2009). It evaluated each chromosome by the criterion of requirement satisfaction which was composed of the number of 1-itemsets and the suitability of membership functions. Although the evaluation only by 1-itemsets was much faster than that by all itemsets or interesting association rules, it is still time-consuming since the database must be scanned once for each chromosome.

In the past, many clustering techniques were proposed (Ben-Dor et al. 1999; Dunn 1973; Ester et al. 1996; McQueen 1967). The purpose of clustering is to gather similar objects into clusters for further analysis. Among the approaches, the *k*-means (also called *c*-means) clustering approach is well known (McQueen 1967). It, however, requests that each data point belongs to only one group. Since the property is not suitable for all applications, the fuzzy *k*-means (also called fuzzy *c*-means) clustering approach was then proposed for getting more flexible clustering results (Dunn 1973). In this paper, the clustering technique will be used to reduce the execution time in solving the MSFM problem. An enhanced approach, called the fuzzy cluster-based genetic-fuzzy mining algorithm for items with multiple minimum supports (FCGFMMS), is proposed to speed up the evaluation process and keep nearly the same quality of solutions as that in (Chen et al. 2009). In the proposed approach, each chromosome represents a set of minimum supports and membership functions used in fuzzy mining. The proposed algorithm first divides the chromosomes in a population into clusters by using the fuzzy *k*-means clustering approach. All the chromosomes then use the requirement satisfaction derived only from the representative chromosomes in the clusters and from their own suitability of membership functions to calculate the fitness values. The evaluation cost can thus be greatly reduced due to the cluster-based time-saving process. Experimental results also show the effectiveness of the proposed algorithm.

The remaining parts of this paper are organized as follows: The proposed cluster-based 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 supports, 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 The proposed cluster-based 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 supports and membership functions from quantitative transactions. A cluster-based 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 maintains a population of sets of minimum support values and membership functions and uses the genetic algorithm to automatically derive the resulting one. Data preprocessing is first done to get initialization information. It then generates and encodes each set of minimum support values and membership functions into a fixed-length string according to the initialization information. It then uses the clustering technique to gather similar chromosomes into groups. Here, the fuzzy k-means clustering approach is used for this purpose. All the chromosomes then use the requirement satisfaction derived from the representative chromosomes in the clusters and their own suitability of membership functions to calculate their fitness values. Since the number for scanning a database decreases, the evaluation cost can thus be reduced. The evaluation results are utilized to choose appropriate chromosomes for mating. The offspring sets of minimum support values and membership functions then undergo recursive evolution until a good set (the highest fitness value) has been obtained.

Finally, the derived minimum support values and membership functions are used to mine fuzzy association rules. Any fuzzy mining approach for items with multiple minimum supports can be used in the framework. Here the approach proposed by Lee et al. (2004) is used as an example. The details are described in the next section.

3 Details of the proposed approach for the framework

In this section, some implementation details about the proposed approach are described. The chromosome representation and the initial population adopted are first stated. The concept of the required number of large 1-itemsets used in fitness evaluation is then explained. The fitness function and the selection procedure are then stated. The fuzzy clustering procedure for chromosomes is then designed, and finally the genetic operators are depicted.

3.1 Chromosome representation and initial population

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. 1998, 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 scheme. 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 total number of items. The second part handles the sets of membership functions for all the items. It also adopts the real-number scheme. Here, we assume the membership functions are isosceles-triangular for simplicity and can thus use only two parameters to represent a 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, …, MF_m. 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.

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

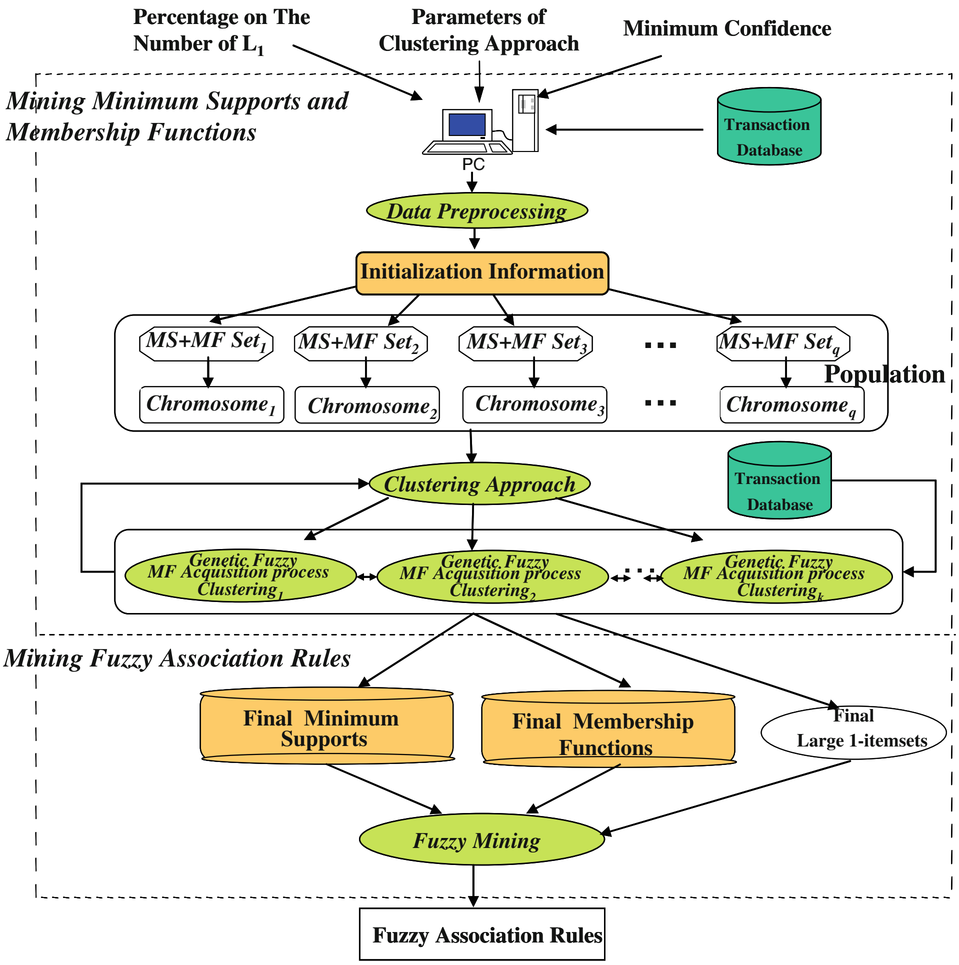
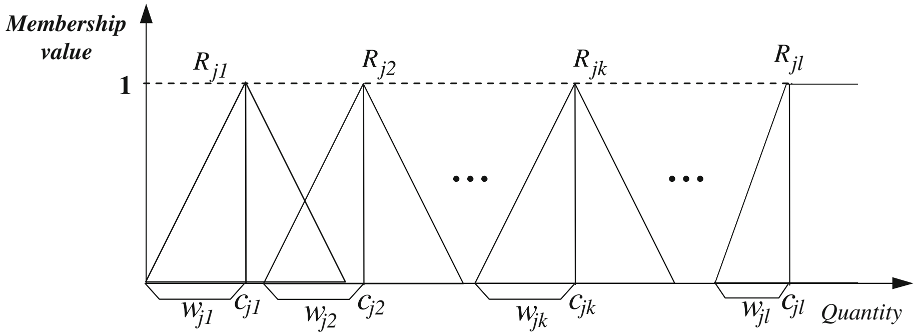

Percentage on The Number of L_1
Parameters of Clustering Approach
Minimum Confidence
Mining Minimum Supports and Membership Functions
PC
Transaction Database
Data Preprocessing
Initialization Information
MS+MF Set$_1$
MS+MF Set$_2$
MS+MF Set$_3$
MS+MF Set$_q$
Chromosome$_1$
Chromosome$_2$
Chromosome$_3$
Chromosome$_q$
Population
Clustering Approach
Transaction Database
Genetic Fuzzy MF Acquisition process Clustering$_1$
Genetic Fuzzy MF Acquisition process Clustering$_2$
Genetic Fuzzy MF Acquisition process Clustering$_k$
Mining Fuzzy Association Rules
Final Minimum Supports
Final Membership Functions
Final Large 1-itemsets
Fuzzy Mining
Fuzzy Association Rules

Fig. 2 Membership functions of item I_j



Besides, the number of fuzzy sets for each item may be different.

A genetic algorithm requires a population of feasible solutions to be initialized and updated during the evolution process. 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 supports and membership functions of each item. The initialization process is stated as follows (Chen et al. 2009). The items are first divided into clusters according to the two attributes, average quantitative values (AQV) and support values (SV), which are calculated from the given transactions. Items in the same cluster are considered to have similar characteristics and are assigned similar values for initializing a better population. The appearing number of each quantitative value is then found 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. The derived break points are then used to generate intervals. 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. The appearing probability of each quantitative value in its corresponding interval is then used for generating membership functions. The first part of each individual in a population of P is generated 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. The second part of each individual in a population is generated according to the found number of linguistic terms and the appearing probabilities of the quantitative values of each item.

3.2 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).

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:

$$\mathrm{RNL}_j = \lfloor l_j * p \rfloor,$$

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.

3.3 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 a probabilistic way. 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:

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

where $\mathrm{RS}(C_q)$ 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. $\mathrm{RS}(C_q)$ is defined as follows:

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

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

$$\mathrm{RS}(C_{qj}) = \begin{cases} \dfrac{|L_1^j|}{\mathrm{RNL}_j}, & \text{if } L_1^j| \le \mathrm{RNL}_j; \\ \dfrac{\mathrm{RNL}_j}{|L_1^j|}, & \text{if } \mathrm{RNL}_j < |L_1^j|; \end{cases}$$

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. $\mathrm{RS}(C_{qj})$ 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:

$$\mathrm{Suitability}(C_q) = \left[\sum_{j=1}^{m} \mathrm{overlap\_factor}(C_{qj})\right] + \mathrm{weight}_1 \left[\left(\sum_{j=1}^{m} \mathrm{coverage\_factor}(C_{qj})\right)\right],$$

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 weight_1 is the coefficient to represent the relative weight ratio of the two factors.

The factor overlap_factor(C_{qj}) is the same as that in (Hong et al. 2006) and defined as follows:

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

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

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

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. 3, 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. Below, an example is given to illustrate the above idea.

3.4 Estimated requirement satisfaction by the fuzzy clustering approach

From the above section, it is known that the large 1-itemsets should be found first before the requirement satisfaction for each chromosome is calculated. The transactions must thus be scanned once for each chromosome to get its requirement satisfaction. Although the evaluation only by 1-itemsets is much faster than that by all itemsets or interesting association rules, it is still time-consuming since the database must be scanned once for each chromosome. In the past, we proposed a method based on the clustering technique to reduce the evaluation time of large 1-itemsets (Chen et al. 2008). It first used the coverage factors and overlap factors of all the chromosomes to form appropriate clusters. For each cluster, the chromosome which was the nearest to the cluster center was thus chosen as the representative chromosome to derive its number of large 1-itemsets. All chromosomes in the same cluster then used the number of large 1-itemsets derived from the representative chromosome as their own. Finally, each chromosome was evaluated by this number of large 1-itemsets divided by its own suitability value.

In this paper, we will modify the above approach to solve the MSFM problem. Since in MSFM, each item has its own minimum support, using only the two factors (coverage factor and overlap factor) for clustering is not enough. For instance, assume there are two membership functions with the same coverage and overlap factors but different minimum supports, which are shown in Fig. 4.

In Fig. 4, although the two chromosomes have the same coverage and overlap factors, they have different numbers of large 1-itemsets due to their different minimum supports. The minimum support values of items should thus be considered as additional attributes for clustering chromosomes. Since using all minimum support values of items as attributes will cause high dimensions, the average minimum support values of items (called the support factor) will be used as an additional attribute for clustering chromosomes. That is

$$\text{support\_factor}(C_q) = \frac{100\sum_{j=1}^{m}\alpha_j}{m},$$

where m is the number of items and α_j is the minimum support value of item I_j.

The clustering process is thus executed according to the coverage factors, the overlap factors and the support factors of chromosomes. In this paper, the fuzzy k-means clustering approach is adopted. Since the chromosomes with similar coverage, overlap and support factors will form a cluster, their minimum supports and shapes of membership functions will be close, thus generating about the same requirement satisfaction. For each cluster, the chromosome which is the nearest to the cluster center (with the highest membership value to the cluster) is thus chosen as the representative and used to derive its requirement satisfaction. Each chromosome then estimates its requirement satisfaction by the requirement satisfaction of its

Fig. 3 Two bad sets of membership functions

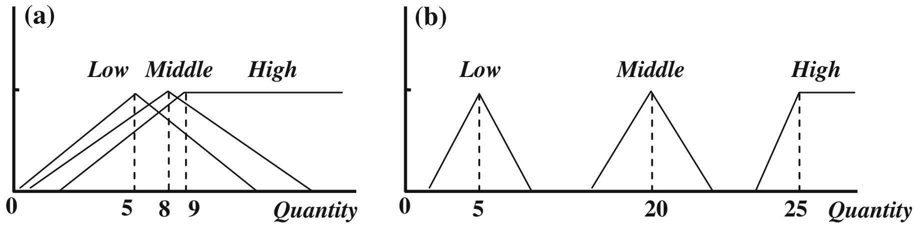
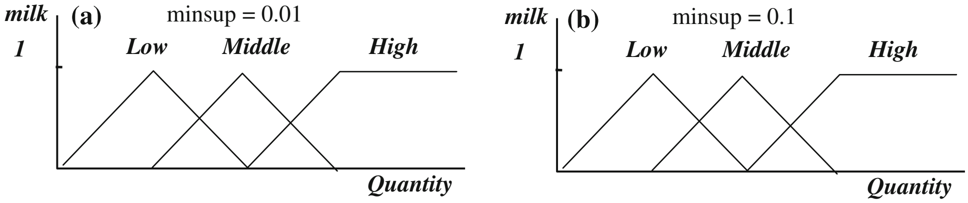


Fig. 4 Two membership functions with the same coverage and overlap factors but different minimum supports



representative chromosome and its membership value to the cluster. That is

$$\text{EstimatedRS}(C_q) = \sum_{g=1}^{k} \text{RS}(\text{RepChro}_g) * \mu_{qg},$$

where RepChro_g is the representative chromosome of cluster_g, $\text{RS}(\text{RepChro}_g)$ is the requirement satisfaction of the representative chromosome RepChro_g, and μ_{qg} is the membership value of chromosome C_q belonging to cluster_g. Finally, each chromosome uses its estimated requirement satisfaction and own suitability of membership functions to calculate its fitness value. The details of the process will be further illustrated later.

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:

$$C_u^t = (c_1, \ldots, c_h, \ldots, c_Z) \quad \text{and} \quad C_w^t = (c_1, \ldots, c_h, \ldots, c_Z)$$

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

1. $C_1^{t+1} = \left(c_{11}^{t+1}, \ldots, c_{1h}^{t+1}, \ldots, c_{1Z}^{t+1}\right)$, where $c_{1h}^{t+1} = dc_h + (1-d)c_h^{'}$,
2. $C_2^{t+1} = \left(c_{21}^{t+1}, \ldots, c_{2h}^{t+1}, \ldots, c_{2Z}^{t+1}\right)$, where $c_{2h}^{t+1} = dc_h^{'} + (1-d)c_h$,
3. $C_3^{t+1} = \left(c_{31}^{t+1}, \ldots, c_{3h}^{t+1}, \ldots, c_{3Z}^{t+1}\right)$, where $c_{3h}^{t+1} = \min\left\{c_h, c_h^{'}\right\}$,
4. $C_4^{t+1} = \left(c_{41}^{t+1}, \ldots, c_{4h}^{t+1}, \ldots, c_{4Z}^{t+1}\right)$, 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 of each jth 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 adding 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.

4 The proposed mining algorithm: FCGFMMS

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

The proposed fuzzy cluster-based genetic-fuzzy mining algorithm for items with multiple minimum supports (FCGFMMS):

INPUT A body of n quantitative transactions, a set of m items, a parameter k for fuzzy k-means clustering, a coefficient w_1 to represent the relative weight ratio, a population size P, a crossover rate P_c, a mutation rate P_m, a crossover parameter d, a percentage of the required number of large 1-itemsets p, 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 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 coverage_factor, overlap_factor and support_factor of each chromosome. The three factors are calculated using the formulas defined in Sect. 3.

STEP 3 Divide the chromosomes into k clusters by the fuzzy k-means clustering approach based on the three attributes (coverage_factors, overlap_factors and support_factors). For each cluster g, find the chromosome with the highest membership value to the cluster as the representative RepChro_g, $1 \le g \le k$.

STEP 4 Calculate the requirement satisfaction of each representative chromosome by the following substeps:

SUBSTEP 4.1 For each transaction datum D_i, $i = 1$–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:

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

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 4.2 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 4.3 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:

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

SUBSTEP 4.4 Set the requirement satisfaction of each representative chromosome using the formulas defined in Sect. 3.

STEP 5 Calculate the fitness values of the representative chromosomes by the formula defined in Sect. 3.3. Calculate the estimated requirement satisfaction of the other chromosomes by the requirement satisfaction of the representative chromosomes and their membership values to the clusters. That is

$$\text{EstimatedRS}(C_q) = \sum_{g=1}^{k} \text{RS}(\text{RepChro}_g) * \mu_{qg},$$

where RepChro_g is the representative chromosome of cluster_g, $\text{RS}(\text{RepChro}_g)$ represents the requirement satisfaction of RepChro_g, and μ_{qg} is the membership value of chromosome C_q belonging to cluster_g.

STEP 6 Calculate the fitness value of each chromosome by the following formula:

$$f(C_q) = \frac{\text{EstimatedRS}(C_q)}{\text{Suitability}(C_q)}.$$

STEP 7 Execute the crossover operation on the population.

STEP 8 Execute the mutation operation on the population.

STEP 9 Calculate the fitness values of chromosomes by using STEPs 2–6.

STEP 10 Use the selection operation to choose appropriate individuals for the next generation.

STEP 11 If the termination criterion is not satisfied, go to Step 2; otherwise, do the next step.

STEP 12 Find the chromosome with the highest fitness value and get the set of minimum supports and membership functions contained in it.

STEP 13 Mine fuzzy association rules using the set of minimum supports and membership functions.

The set of minimum supports and membership functions are thus used to mine fuzzy association rules from the given database. Any fuzzy mining approach for items with multiple minimum supports can be used in the framework. Here the approach proposed by Lee et al. (2004) is used as an example to achieve this purpose.

5 An example

In this section, a simple example is given to illustrate the proposed FCGFMMS algorithm. 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 proposed FCGFMMS algorithm proceeds as follows.

STEP 1 P individuals are 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. Assume the ten individuals generated are shown below, where the first four numbers are minimum supports and the others are the parameters for membership functions:

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;

C_2: 0.26, 0.06, 0.01, 0.12, 4, 3.41, 6, 1.67, 10, 7.42, 3, 2.81, 8, 4.71, 10, 1.42, 6, 4.03, 12, 3.63, 7, 1.65, 12, 2.24;

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)

C_3: 0.2, 0.3, 0.16, 0.2, 2, 1.97, 8, 7.48, 11, 10.66, 2, 1.68, 7, 1.05, 11, 6.85, 6, 4.1, 12, 5.73, 6, 5.57, 11, 10.21;

C_4: 0.33, 0.09, 0.09, 0.05, 3, 2.9, 7, 1.37, 10, 7.92, 3, 1.13, 8, 5.99, 10, 8.2, 7, 3.93, 12, 2.22, 7, 1.01, 10, 9.78;

C_5: 0.12, 0.18, 0.06, 0.22, 2, 1.76, 8, 4.71, 11, 6.66, 3, 1.88, 6, 5.68, 11, 8.86, 7, 6.97, 10, 6.99, 7, 2.12, 12, 5.62;

C_6: 0.28, 0, 0.08, 0.16, 4, 2.3, 6, 3.46, 11, 9.71, 2, 1.78, 7, 4.62, 10, 5.86, 6, 4.33, 10, 8.24, 6, 2.99, 11, 9.93;

C_8: 0.33, 0.06, 0.01, 0.11, 3, 2.49, 8, 6.32, 11, 6.8, 4, 1.07, 7, 1.81, 10, 8.85, 6, 5.53, 12, 2.45, 7, 5.37, 12, 10.01;

C_8: 0.33, 0.06, 0.01, 0.11, 3, 2.49, 8, 6.32, 11, 6.8, 4, 1.07, 7, 1.81, 10, 8.85, 6, 5.53, 12, 2.45, 7, 5.37, 12, 10.01;

C_9: 0.27, 0.02, 0.16, 0.1, 3, 2.31, 7, 4.27, 11, 10.41, 3, 1.6, 6, 3.5, 10, 3.08, 6, 5.9, 12, 4.89, 7, 3.41, 12, 3.46;

C_{10}: 0.26, 0.17, 0.13, 0.15, 3, 2.61, 8, 5.6, 11, 8.55, 3, 1.87, 6, 5.8, 11, 5.98, 6, 2.56, 11, 5.31, 6, 5.01, 12, 8.51.

STEP 2 The coverage_factor, overlap_factor and support_factor of each chromosome are calculated by the formulas defined in Sect. 3. The results are shown in Table 2, where the column "*Attributes*" represents the tuples (coverage_factor, overlap_factor, support_factor).

STEP 3 The fuzzy *k*-means clustering approach is executed to divide the ten chromosomes into clusters. In this example, assume the parameter *k* is set at 3. The membership values of the chromosomes to each cluster and the representative chromosomes are shown in Table 3. The representative chromosomes (with the highest membership value to each cluster) in the three clusters are C_1, C_3 and C_6.

STEP 4 The requirement satisfaction of each representative chromosome is calculated by the following substeps.

Table 2 The coverage_factor, overlap_factor and support_factor of each chromosome

Chromosome	Attributes	Chromosome	Attributes
C_1	(5.58, 2.88, 16.25)	C_6	(4.39, 5.23, 13.12)
C_2	(5.43, 3.26, 11.06)	C_7	(4.72, 6.08, 14.78)
C_3	(4.29, 4.53, 21.70)	C_8	(4.34, 4.06, 12.58)
C_4	(4.54, 5.19, 13.79)	C_9	(4.61, 2.43, 13.86)
C_5	(4.73, 3.77, 14.74)	C_{10}	(4.54, 3.22, 17.94)

SUBSTEP 4.1 The quantitative value of each transaction datum is transformed into a fuzzy set according 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 amount "6" of item milk is then converted into the fuzzy set $\left(\frac{0.75}{\text{milk.Middle}}\right)$ using the membership functions for milk in C_1. The results for all the items are shown in Table 4, where the notation item.term is called a fuzzy region.

SUBSTEP 4.2 The scalar cardinality of each fuzzy region in the transactions is calculated as the count value. Take the fuzzy region milk.Middle as an example. Its scalar cardinality = (0.75 + 1.0 + 0 + 0 + 0 + 0.75 + 0.25 + 0 + 0 + 0) = 2.75. The counts for all the fuzzy regions are shown in Table 5.

SUBSTEP 4.3 The count of any fuzzy region is checked against the minimum support value in C_1. The minimum support values of 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 values of milk.Low is larger than 2.5 (=0.25*10), these items are put in L_1. The results for other items are shown in Table 6.

SUBSTEP 4.4 Assume the percentage *p* of the required number of large 1-itemsets is set at 0.8.

Table 3 The membership values of chromosomes and the representative chromosomes

	Cluster_1	Cluster_2	Cluster_3
C_1	**0.917**	0.025	0.056
C_2	0.220	0.042	0.736
C_3	0.002	**0.996**	0.001
C_4	0.111	0.013	0.874
C_5	0.731	0.018	0.249
C_6	0.045	0.006	**0.948**
C_7	0.352	0.065	0.581
C_8	0.049	0.006	0.943
C_9	0.536	0.035	0.427
C_{10}	0.621	0.245	0.132
Representative chromosome	C_1	C_3	C_6

Table 4 The fuzzy sets transformed from the data in Table 1

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

Table 5 The counts of the fuzzy regions

Item	Count	Item	Count
milk.Low	0.5	bread.High	1.5
milk.Middle	2.75	cookies.Low	1.66
milk.High	1.66	cookies.High	2.99
bread.Low	1.0	beverage.Low	2.00
bread.Middle	2.66	beverage.High	2.14

Table 6 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 RNL values of the four items milk, bread, cookies and beverage are $2(=\lfloor 3*8 \rfloor), 2(=\lfloor 3*8 \rfloor), 1(=\lfloor 3*8 \rfloor)$ and $1(=\lfloor 2*8 \rfloor)$, respectively. The number of L_1 of the four items milk, bread, cookies and beverage are 1, 3, 2 and 2 from Table 6. 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). The requirement satisfaction of C_1 is thus 2.16 (=0.5 + 0.66 + 0.5 + 0.5). The results for the three representative chromosomes are shown in Table 7.

Table 7 The requirement satisfaction for the three representative chromosomes

Cluster_i	Representative chromosome	RS
Cluster_1	C_1	2.16
Cluster_2	C_3	2.0
Cluster_3	C_6	2.66

STEP 5 and 6 The estimated requirement satisfaction of each chromosome is calculated. Take C_5 as an example. Its membership values to cluster_1, cluster_2 and cluster_3 are 0.731, 0.018 and 0.249, respectively. Its estimated requirement satisfaction is thus 2.28 (=2.16*0.731 + 2*0.018 + 2.66*0.249). The fitness value of each chromosome is calculated. Take C_5 as an example. Its estimated requirement satisfaction is 2.28 and its suitability is calculated as 4.96 when the weight weight_1 is set at 0.25. The fitness value of C_5 is thus 2.28/4.96 (=0.46). The evaluation results for the other chromosomes can be similarly derived and are shown in Table 8.

STEP 7 The crossover operation is executed on the population. Assume the crossover rate is set at 0.8. According to the MMA crossover operator, totally eighty offspring chromosomes are generated. The offspring chromosomes are put into the population.

Table 8 The fitness values of the chromosomes

Chromosome	f	Chromosome	f
C_1	0.512	C_6	0.421
C_2	0.547	C_7	0.336
C_3	0.356	C_8	0.511
C_4	0.410	C_9	0.662
C_5	0.460	C_{10}	0.502

STEP 8 The mutation operation is executed on the population to generate possible offspring. The operation is the same as the traditional one except that rearrangement may need to be done. Here, the offspring chromosomes are also put into the population.

STEP 9 Steps 2–6 is executed to calculate the fitness values of the chromosomes in the population.

STEPs 10–13 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 rules. After the minimum support values and membership functions are derived, the fuzzy mining method proposed in (Lee et al. 2004) is then 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 10000 transactions were used in the experiments. The initial population size P was set at 50, the cluster number k was set at 5, 10, 15 and 20, the crossover rate p_c was set at 0.8, and the mutation rate p_m was set at 0.001. The parameter d of the crossover operator was set at 0.35 according to Herrera et al.'s paper (Herrera et al. 1997). The percentage of the required number of large 1-itemsets was set at 0.8 and the weight weight_1 was set at 1/64.

In the following sections, we first give a description of the experimental dataset. We then analyze the performance of the proposed approach according to the designed fitness function. The comparison of the proposed approach (FCGFMMS) and the previous approach (GFMMS) (Chen et al. 2009) are then made to show the efficiency of the proposed algorithm.

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 followed exponential distribution and another one followed uniform distribution. 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 (Agrawal and Srikant 1994; Hong et al. 1999) for both the two datasets. The purchased items in each transaction were then selected from the 64 items in a uniform distribution of the range [1, 64] for the uniform dataset and in an exponential distribution with the rate parameter set at 16 for the exponential dataset. Their quantities were then assigned from a uniform distribution of the range (Agrawal and Srikant 1994; Dunn 1973) for the uniform dataset and from an exponential distribution with the rate parameter set at 5 for the exponential dataset. The simulation process was repeated 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 were apparently much better than the original ones. For example, the initial minimum supports and membership functions of some two items among the 64 items are shown in Fig. 5a. The membership functions have the two bad types of shapes according to the definition in the previous section. After 500 generations, the final minimum support values and membership functions for the same items are shown in Fig. 5b. It is easily seen that the membership functions in Fig. 5b is better than those in Fig. 5a. The two bad kinds of membership functions are improved in the final results.

The average fitness values of the chromosomes along with different numbers of generations by the proposed approach were then found. The average fitness value was calculated by the following formula:

$$\text{AvgFitness} = \sum_{q=1}^{P} f(C_q)/P,$$

where P is the size of the population and $f(C_q)$ is the fitness value of chromosome C_q. The results are shown in Fig. 6. As expected, the curves for the dataset gradually went upward, finally converging to a fixed value.

Next, experiments were made for providing a comparative analysis of the proposed approach with different clustering approaches, including hard c-means (HCM) and fuzzy c-means (FCM) clustering approaches. The average fitness values of the chromosomes along with different

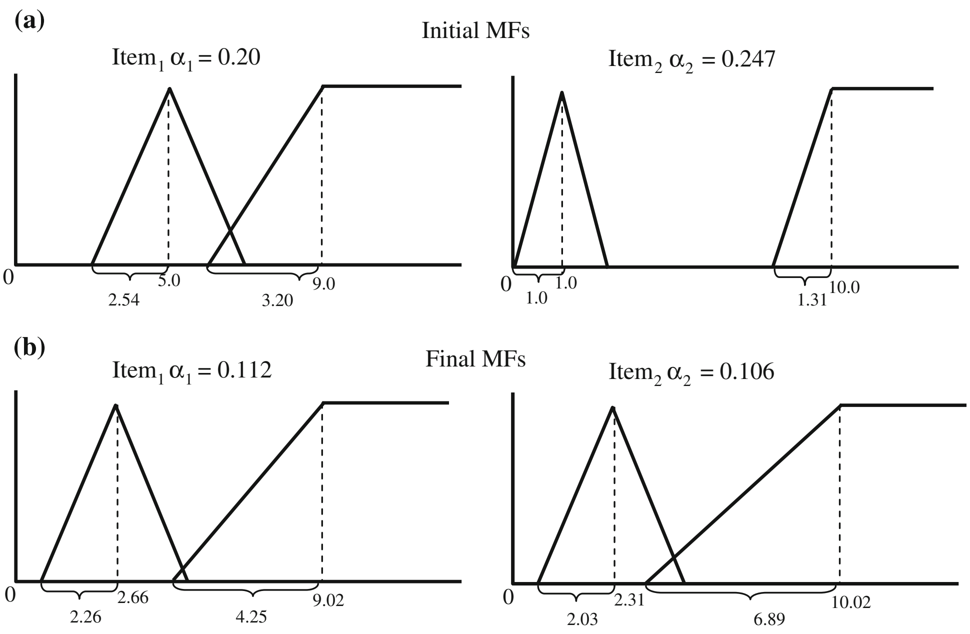

(a)
Initial MFs
Item$_1$ α_1 = 0.20
0
2.54
5.0
3.20
9.0
Item$_2$ α_2 = 0.247
0
1.0
1.0
1.31
10.0
(b)
Final MFs
Item$_1$ α_1 = 0.112
0
2.26
2.66
4.25
9.02
Item$_2$ α_2 = 0.106
0
2.03
2.31
6.89
10.02

Fig. 5 The initial and the final minimum support values and membership functions of some items for the exponential dataset

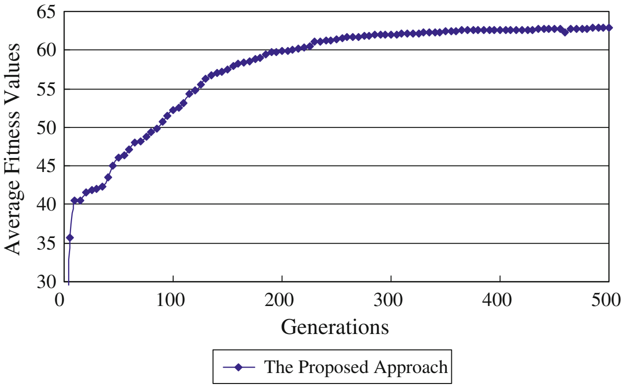


Fig. 6 The average fitness values along with different numbers of generations

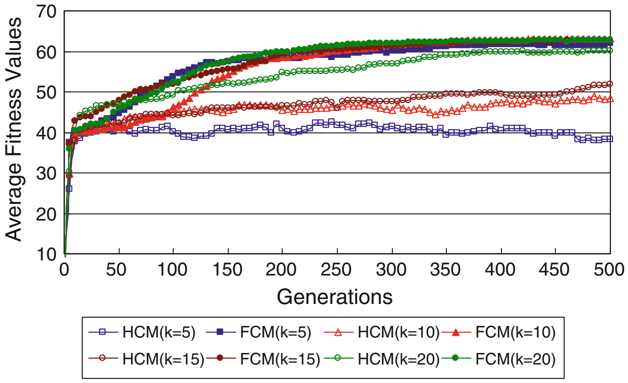


Fig. 7 The average fitness values of the chromosomes by different clustering approaches

numbers of generations of the proposed approach with HCM and with FCM for different numbers of clusters are shown in Fig. 7.

From Fig. 7, it can be first observed that the average fitness values gradually increased by both the clustering approaches when the number of clusters increased. Overall, the average fitness values of the proposed approach with fuzzy *c*-means were better than that with hard *c*-means. The results were reasonable since the fuzzy *c*-means could estimate the requirement satisfactions of the chromosomes through the property that an object could belong to more than one cluster.

6.3 The comparisons of the proposed approach and the previous approach

Experiments were then made to compare the proposed method (FCGFMMS) with our previous one (GFMMS) (Chen et al. 2009) for showing the effect of using clusters in evaluation. The average fitness values of the chromosomes along with different numbers of generations for different numbers of clusters for exponential and uniform distributions are shown in Figs. 8 and 9, respectively.

It could be observed from Fig. 8 that the average fitness values by the proposed approach were only a little less than those by the previous one. The similar experimental phenomenon could also be observed in Fig. 9, in which the results were very close when the number of clusters increased. The results were reasonable since the proposed approach just estimated the requirement satisfaction of chromosomes. The comparisons for the execution time of the two approaches with different numbers of clusters for

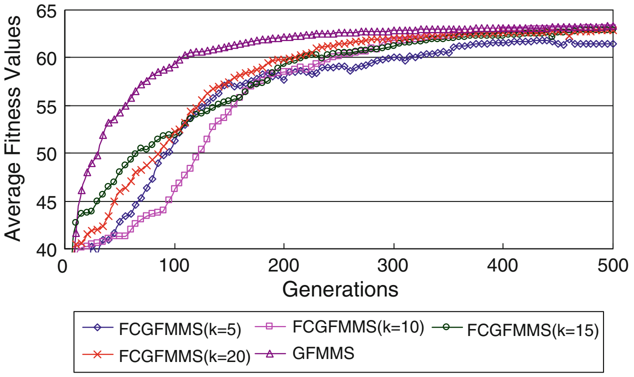


Fig. 8 The comparison results between the proposed and the previous approaches for the dataset with exponential distribution

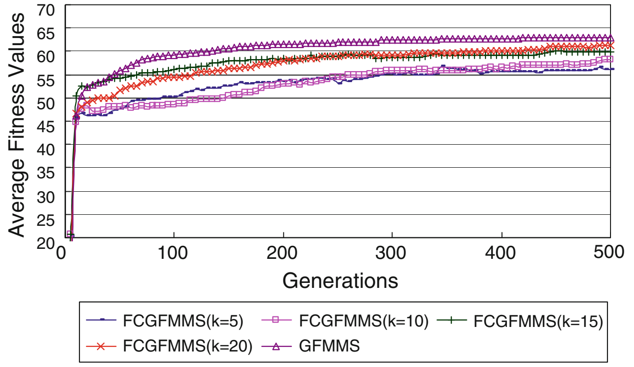


Fig. 9 The comparison results between the proposed and the previous approaches for the dataset with uniform distribution

both the exponential and uniform distributions are shown in Tables 9 and 10. The results were averaged for five runs with different seeds.

From Tables 9 and 10, it could be seen that the proposed approach (FCGFMMS) ran nearly five to ten times faster than the previous one (GFMMS) on both the datasets. Besides, based on Lee et al.'s approach (Lee et al. 2004), the number of rules, the average support and the average confidence of the rules were also shown in Tables 9 and 10. From Table 9, it was found that when k was set at 10–20, the proposed approach could get good results both on the average fitness values and the obtained rules for the dataset with exponential distribution when compared with the GFMMS approach. From Table 10, it could be found that when k was set at 15 or 20, the proposed approach had good results for the dataset with uniform distribution. Usually, the results should be better along with the increase of the number of clusters. However, due to the heuristic that the number of large 1-itemsets was used in the evaluation, the number of rules did not necessarily increase along with the number of clusters. Appropriate choice of a cluster number is thus important.

7 Conclusion and future works

In this paper, we have proposed a genetic-fuzzy mining algorithm, namely FCGFMMS, for extracting multiple minimum supports, membership functions and fuzzy association rules from quantitative transactions. The proposed algorithm can adjust the minimum support and membership functions for each item by genetic algorithms and use them to fuzzify quantitative transactions. It can also speed up the evaluation process and keep nearly the same quality of solutions by incorporating the clustering technique.

In the proposed approach, each chromosome represents a set of minimum support values and membership

Table 9 The execution time of the two approaches for the dataset with exponential distribution

Approaches	Execution time (min)	Speed-up ratio	#Rules	Avg.Sup.	Avg.Conf.
FCGFMMS ($k = 5$)	18.775	13.372	830	0.027	0.237
FCGFMMS ($k = 10$)	25.924	9.685	1,100	0.024	0.222
FCGFMMS ($k = 15$)	33.854	7.416	1,032	0.024	0.238
FCGFMMS ($k = 20$)	39.970	6.281	1,306	0.022	0.229
GFMMS	251.060	1	1,347	0.021	0.234

Table 10 The execution time of the two approaches for the dataset with uniform distribution

Approaches	Execution time (min)	Speed-up ratio	#Rules	Avg.Sup.	Avg.Conf.
FCGAMMS ($k = 5$)	25.511	11.887	777	0.022	0.505
FCGFMMS ($k = 10$)	33.118	9.157	1,122	0.022	0.570
FCGFMMS ($k = 15$)	42.601	7.118	1,378	0.021	0.624
FCGFMMS ($k = 20$)	53.316	5.688	1,437	0.022	0.640
GFMMS	303.253	1	1,593	0.021	0.689

functions used in fuzzy mining. The proposed algorithm first divides the chromosomes in a population into clusters by using the fuzzy *k*-means clustering approach. All the chromosomes then use the requirement satisfactions derived from the representative chromosomes of the clusters and their own suitability of membership functions to calculate the fitness values. The evaluation cost can thus be significantly reduced due to the time-saving in finding requirement satisfaction.

Experimental results first show that the adopted fitness function can derive a good minimum support values and membership functions. Comparisons of the proposed approach with different clustering methods (HCM and FCM) are also made. The results show that the proposed approach with the FCM clustering approach can get better results than that with the HCM clustering approach. The experimental results also show that using the clustering technique to speed up the evaluation process can not only get nearly the same fitness values as the previous approach, but can also significantly reduce execution time. The proposed approach can thus get a good trade-off between accuracy and execution time.

The proposed approach would like to derive as much knowledge amount as possible under a set of multiple minimum support thresholds. It implies that for the same knowledge amount to be obtained, the proposed approach can get a set of higher minimum support thresholds than the others. Using a set of higher minimum supports is usually better in the meaning of relevance than using a set of lower ones in the mining process. This is an advantage of the approach. In most applications, the large majority of rules obtained are non-interesting for experts when evaluated despite they may be considered interesting by the support-confidence framework employed. Thus, how to determine appropriate minimum support thresholds or design an alternative measure to control the number of large 1-itemsets is thus very critical. This is a big challenge that we will try to solve in the future. At last, some interesting works on generic fuzzy systems and rule reduction approaches can be found as well in (Casillas and Carse 2009; Gacto et al. 2009). These works also provide possible future directions for enhancing the proposed approach.

Acknowledgments This research was supported by the National Science Council of the Republic of China under contract NSC 98-2221-E-390-033.

References

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

Alcala-Fdez J, Alcala R, Gacto M, Herrera F (2009) Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. Fuzzy Sets Syst 160(7):905–921

Ben-Dor A, Shamir R, Yakhini Z (1999) Clustering gene expression patterns. In: The annual international conference on computational molecular biology, pp 281–297

Casillas J, Carse B (2009) Genetic fuzzy systems: recent developments and future directions. Soft Comput 13(3):417–418

Casillas J, Cordon 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

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

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, Tseng VS, Hong TP (2008) Cluster-based evaluation in fuzzy-genetic data mining. IEEE Trans Fuzzy Syst 16(1):249–262

Chen CH, Hong TP, Tseng VS, Lee CS (2009) A genetic-fuzzy mining approach for items with multiple minimum supports. Soft Comput 13(5):521–533

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

Dunn JC (1973) “A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters”. J Cybern 3:32–57

Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: The international conference on knowledge discovery and data mining, pp 226–231

Fu A, Wong M, Sze S, Wong W, Wong W, Yu W (1998) Finding fuzzy sets for the mining of fuzzy association rules for numerical attributes. In: The international symposium on intelligent data engineering and learning, pp 263–268

Gacto MJ, Alcalá R, Herrera F (2009) Adaptation and application of multi-objective evolutionary algorithms for rule reduction and parameter tuning of fuzzy rule-based systems. Soft Comput 13(3):419–436

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 Info Technol Biomed 10(1):28–41

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, Lee YC (2001) Mining coverage-based fuzzy rules by evolutional computation. In: The IEEE international conference on data mining, pp 218–224

Hong TP, Kuo CS, Chi SC (1999) A data mining algorithm for transaction data with quantitative values. In: The eighth international fuzzy systems association world congress, pp 874-878

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

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 12(2):252–265

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 (2005) Genetic algorithm based framework for mining fuzzy association rules. Fuzzy Sets Syst 152(3):587–601

Kuok C, Fu A, Wong M (1998) Mining fuzzy association rules in databases. SIGMOD Rec 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

Mangalampalli A, Pudi V (2009) Fuzzy association rule mining algorithm for fast and efficient performance on very large datasets. In: The IEEE international conference on fuzzy systems, pp 1163–1168

McQueen JB (1967) Some methods of classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley symposium on mathematical statistics and probability, pp 281–297

Mohamadlou H, Ghodsi R, Razmi J, Keramati A (2009) A method for mining association rules in quantitative and fuzzy data. In: The international conference on computers & industrial engineering, pp 453–458

Ouyang W, Huang Q (2009) Mining direct and indirect weighted fuzzy association rules in large transaction databases. Int Conf Fuzzy Syst Knowl Discov 3:128–132

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

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

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

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

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

Wang CH, Hong TP, Tseng SS (1998) Integrating fuzzy knowledge by genetic algorithms. IEEE Trans Evol Comput 2(4):138–149

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, Berlin