doi:10.3772/j.issn.1006-6748.2017.01.002

Incremental learning of the triangular membership functions based on single-pass FCM and CHC genetic model①

Huo Weigang (霍纬纲)②, Qu Feng, Zhang Yuxiang
(Department of Computer Science and Technology, Civil Aviation University of China, Tianjin 300300, P. R. China)

Abstract

In order to improve the efficiency of learning the triangular membership functions (TMFs) for mining fuzzy association rule (FAR) in dynamic database, a single-pass fuzzy c means (SPFCM) algorithm is combined with the real-coded CHC genetic model to incrementally learn the TMFs. The cluster centers resulting from SPFCM are regarded as the midpoint of TMFs. The population of CHC is generated randomly according to the cluster center and constraint conditions among TMFs. Then a new population for incremental learning is composed of the excellent chromosomes stored in the first genetic process and the chromosomes generated based on the cluster center adjusted by SPFCM. The experiments on real datasets show that the number of generations converging to the solution of the proposed approach is less than that of the existing batch learning approach. The quality of TMFs generated by the approach is comparable to that of the batch learning approach. Compared with the existing incremental learning strategy, the proposed approach is superior in terms of the quality of TMFs and time cost.

Key words: incremental learning, triangular membership function (TMFs), fuzzy association rule (FAR), real-coded CHC

0 Introduction

Many studies have proposed the method of mining fuzzy association rules (FAR) from quantitative data to avoid unnatural boundaries[1]. The result of mining FAR is strongly influenced by the quantitative attribute's membership function. The problem of learning membership function is one of the most critical preprocessing steps. Much research work has been done to this problem. These approaches can be classified into two categories. The first is based on clustering method[2-4]. In Ref. [4], the method based on CLARANS clustering algorithm was proposed to find fuzzy set for mining fuzzy association rule. The membership functions of quantitative attribute were generated according to centroids of clusters. Ref. [2] presented a more efficient approach based on CURE cluster algorithm. Their method used less centroids than the approach described in Ref. [4]. Ref. [3] proposed an efficient algorithm for mining FAR in high-dimensional datasets. They used fuzzy c-means (FCM) clustering to create fuzzy partitions for numerical attribute.

The second category employs evolutionary algorithms, particularly genetic algorithms (GAs)[5-10]. Ref. [5] presented a GAs-based clustering method to derive membership functions. In this method, centroids of clusters which were regarded as the midpoint of TMFs were adjusted by GAs. The optimization objective was to maximize the number of large fuzzy frequent itemsets. Ref. [6] proposed a GA-based framework to learn membership functions. The value of fitness function was calculated by the number of frequent 1-itemsets and suitability of membership functions. The suitability measure was used to guarantee a good shape of membership functions. In order to reduce the time cost of calculating frequent 1-itemsets, Ref. [7] used k-means clustering algorithm to divide the chromosomes in a population into k clusters. The number of frequent 1-itemsets derived from the representative chromosome in a cluster was regarded as the number of frequent 1-itemsets resulting from each chromosome in the cluster. In Ref. [8], the fitness of the chromosome was evaluated by the fuzzy-supports of frequent 1-itemsets and the suitability of derived membership functions, which made them adopt the divide-and-conquer strategy to derive membership functions for different quantitative attributes. Ref. [9] also proposed the multi-level ant

① Supported by the National Natural Science Foundation of China (No. 61301245, U1533104).
② To whom correspondence should be addressed. E-mail: wghuo@cauc.edu.cn
Received on Dec. 28, 2015

colony framework to learn suitable membership functions for mining FAR. In this method, membership functions were encoded as a binary bits string. The fitness function was the same as the function used in Refs[6,7]. Multi-level processing was used to solve the problem in which the maximum quantities of item values in the transactions may be large. Ref. [10] presented an approach for genetic learning of membership functions based on the 2-tuples linguistic representation model and CHC genetic algorithm. The advantage of this method lied in the fact that the search space is reduced with respect to the classic tuning[6-8]. Recently, Ref. [11] proposed a method for genetic learning of the membership functions from imprecise data, which was based on the 3-tuples linguistic representation model to reduce the search space and to learn the adequate context for fuzzy linguistic terms.

All of the above approaches are executed in a batch way. In a dynamic database where the new numeric values are inserted into the original dataset, these algorithms need to be executed from scratch. This is a huge computational workload. Ref. [12] proposed an approach for incremental generating membership function. The approach was based on the CLUSTERDB* cluster algorithm. Each cluster was represented by a trapezoidal function. When a new data object was inserted into the original dataset, the approach first determined the cluster for the new data object according to the clustering property of coherent partition. In the second step, the parameters of trapezoidal function which represented the cluster including the new data object were updated correspondingly. If the new data object can't be absorbed by any cluster, CLUSTERDB* algorithm must be re-applied to the whole dataset. As far as we know, the research on incremental learning membership functions for mining FAR is very rare.

Inspired by Ref. [13] and Ref. [14], a new approach to incrementally learn the TMFs for mining FAR is presented, which consists of the following two steps:

(1) First, the values of quantitative attribute in the origin database are clustered by SPFCM[15]. The TMFs are encoded as three real-number schema and are learned by the CHC genetic model[15]. In order to enhance the quality of TMFs, the cluster centers resulting from SPFCM are regarded as the midpoint of TMFs. The other two endpoints of TMFs are generated sequentially according to its rational value range. In every generation of CHC, the chromosome with the best fitness value will be stored.

(2) When the new numeric values are inserted into the original dataset, the cluster's centers are adjusted by SPFCM. The new initial population consists of two parts. One part is randomly generated by the adjusted center as the first step. The other part is extracted from the excellent chromosomes maintained in the first step. In this genetic process, only new inserted numeric values are used to calculate the fitness value of the chromosomes. The chromosome with the best fitness value in the new population will be output as the membership functions of updated quantitative attribute.

In order to assess the performance of the proposed approach, an experimental study using a public database, FAM95 is presented. First, the performance of our approach is compared with other batch learning methods in terms of the number of generation converging to a solution and the quality of TMFs. These batch learning methods include the batch CHC model proposed by Ref. [10] and the batch FCM-CHC model whose principle is the same as the approach proposed by Ref. [5]. Second, the proposed approach is compared with CLUSTERDB* based incremental learning approach. Finally, the influence of the number of new inserted numeric value on the proposed incremental genetic learning process is analyzed.

1 The SPFCM algorithm and the CHC genetic components to incrementally learn the membership functions

In this paper, the SPFCM algorithm and the CHC genetic model are used to design the proposed method. The CHC genetic algorithm is an advanced evolutionary algorithm which has a good trade-off between exploration and exploitation. It makes use of an incest prevention mechanism and a restarting process to provoke diversity in the population. Ref. [10] verified that the CHC genetic model was superior to usual genetic algorithm in learning membership functions for mining FAR. The SPFCM algorithm can produce data partitions in a single pass through the dataset, which is used to reduce the search space of the CHC genetic model and enhance the quality of the TMFs. The details of the SPFCM algorithm and the components designed in the CHC genetic model are described as follows.

1.1 SPFCM algorithm

SPFCM is an incremental clustering method designed based on FCM. It processes the data chunk by chunk. For each chunk, a set of weighted centroids are calculated to represent the chunk with one centroid per cluster. The weight for the centroid of each cluster in each chunk is calculated as follows:

$$w_c = \sum_{i=1}^{n_p+q} (u_{ci}) \cdot w_i,\ 1 \leqslant c \leqslant k \quad (1)$$

In Eq. (1), w_c is the weight of centroid of the cth cluster, n_p is the number of data objects in the pth chunk, k is the number of clusters, u_{ci} is the membership of data object i belonging to cluster c, and w_i is the weight of data object i. For the first chunk of the data object ($p=1$), w_i is assigned to 1 for every data object and $q = 0$. From the second chunk of data ($p \neq 1$), $q = k$ and the k-weighted centroids are combined with the pth chunk of data. There union constitutes dataset $\boldsymbol{X}$. The $n_p + k$ data objects in $\boldsymbol{X}$ will be clustered by weighted FCM (wFCM) in which the weights of the n_p objects in pth chunk are all set to 1 and the weights of k centroids are calculated from previous chunk according to Eq. (1).

The wFCM modifies the objective function of FCM to take the effect of the weighted points into consideration. The iterative clustering principle of wFCM is the same as that of FCM. The cluster centroids and membership matrix for the wFCM are calculated as follows:

$$v_i = \frac{\sum_{j=1}^{n_p+k} w_j \cdot (u_{ij})^m \cdot x_j}{\sum_{j=1}^{n_p+k} w_j \cdot (u_{ij})^m},\ 1 \leqslant i \leqslant k,\ x_j \in \boldsymbol{X} \quad (2)$$

$$u_{ij} = \left[\sum_{l=1}^{k} \left(\frac{\| x_j - v_i \|}{\| x_j - v_l \|} \right)^{\frac{2}{m-1}} \right]^{-1},\ 1 \leqslant i \leqslant k,\ 1 \leqslant j \leqslant n_p + k \quad (3)$$

In Eqs(2) and (3), m is a fuzziness coefficient.

1.2 Chromosome representation and initial population

The triangular membership functions (TMFs) are used in the proposed approach. It is used most widely in the fuzzy system. Each set of TMFs of a quantitative attribute is encoded as a chromosome and handled as an individual with real-number string. The membership functions for a quantitative attribute y having k fuzzy sets are shown in Fig. 1, where $R_i (1 \leqslant i \leqslant k)$ denotes the membership function of the ith linguistic term for y, L and R indicate the parameters of fuzzy region R_1 and R_k respectively, C_{ij} indicates the parameter of fuzzy region $R_i (i = 2, 3, \cdots, k-1;\ j = 1, 2, 3)$, $y^{\min}$ and $y^{\max}$ indicate the minimum and maximum value of quantitative attribute y. A chromosome representing the membership functions for y is encoded in the following form:

$$L, C_{21}, C_{22}, C_{23}, C_{31}, C_{32}, C_{33}, \cdots, R$$

Except for the left and right side of the membership functions R_1 and R_k, three integer numbers are used to represent the $R_i (2 \leqslant i \leqslant k-1)$. Each gene value in the chromosome must lie between $y^{\min}$ and $y^{\max}$. The inequality conditions of the value of the parameter $C_{ij} (i = 2, 3, \cdots, k-1;\ j = 1, 2, 3)$ are $C_{i1} \leqslant C_{i2} \leqslant C_{i3}$, and $y^{\min} < C_{22} < C_{32} < \cdots < C_{(k-1)2} < y^{\max}$.

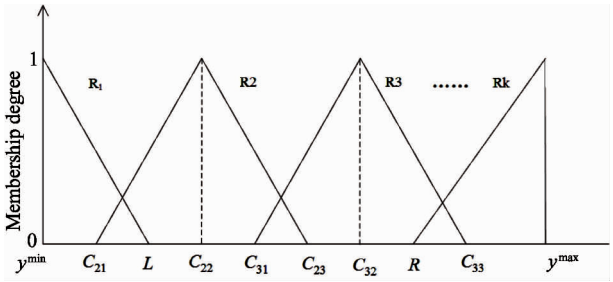

Fig. 1 An example of membership functions for quantitative attribute y

The initial set of chromosomes in an evolving population is randomly generated according to constraints mentioned above. To illustrate the procedure of initializing genetic population, it is assumed that the number of the membership function for quantitative attribute y is 4. So the length of every chromosome is 8. The size of evolving population is denoted as N_{pop}. Then a two dimension array $a[N_{\text{pop}}]$[8] is used to represent the evolving population for generating high quality membership functions. The pseudo-code of the procedure of initializing evolving population in this study is described as follows.

(1) Run the SPFCM clustering algorithm on the attribute values of quantitative attribute y, the clustering number is set as 4, produce the cluster center $\boldsymbol{C} = \{C(1), C(2), C(3), C(4)\}$.

(2) For $i = 1$ to N_{pop}

$a[i][3] = C(2)$; //$a[i][3]$ corresponds to C_{22} in the chromosome.

$a[i][6] = C(3)$; //$a[i][6]$ corresponds to C_{32} in the chromosome.

$a[i][2] =$ randint $(y^{\min}, C(2))$, which means randomly generating integer number between $y^{\min}$ and $C(2)$. // $a[i][2]$ corresponds to C_{21} in the chromosome. The meaning of following randint is the same as that of here.

$a[i][7] =$ randint $(C(3), y^{\max})$; // $a[i][7]$ corresponds to C_{33} in the chromosome.

$a[i][4] =$ randint $(C(2), a[i][7])$; // $a[i][4]$ corresponds to C_{23} in the chromosome.

$a[i][1] =$ randint $(y^{\min}, a[i][4])$; //$a[i][1]$ corresponds to L in the chromosome.

$a[i][5] =$ randint $(a[i][2], C(3))$; // $a[i][5]$ corresponds to C_{31} in the chromosome.

$a[i][8] =$ randint $(a[i][5], y^{\max})$; // $a[i]$

[8] corresponds to R in the chromosome.

End for

It must be pointed out that the principle of initializing evolving population is the same as the procedure mentioned above when the number of quantitative attribute's fuzzy set is not 4.

1.3 Chromosome evaluation

In this study, the fitness function defined in Ref. [8] is used to evaluate a chromosome. The fitness value of chromosome C_q is defined as

$$fitness(C_q) = \frac{\sum_{x \in L_1} fuzzy\_support(x)}{suitability(C_q)} \quad (4)$$

Where L_1 is the set of fuzzy frequent 1-itemsets obtained by using the membership functions which are represented by the chromosome C_q, $fuzzy\_support\ (x)$ is the fuzzy support of fuzzy frequent 1-itemset x for the given transaction database, and $suitability(C_q)$ represents the shape suitability of the membership functions. The suitability of the set of membership functions in the chromosome C_q is defined as

$$suitability(C_q) = overlap\_factor(C_q) + coverage\_factor(C_q) \quad (5)$$

where $overlap\_factor(C_q)$ and $coverage\_factor(C_q)$ represent the overlap factor and the coverage factor of the membership functions represented by C_q respectively.

The overlap factor of the membership functions in C_q is defined as

$$overlap\_factor(C_q) = \sum_{i=1}^{k-1} \left[\max\left(\frac{overlap(R_i, R_{j=i+1})}{\min(C_{i3} - C_{i2}, C_{j2} - C_{j1})}, 1 \right) - 1 \right] \quad (6)$$

where $overlap(R_i, R_{j=i+1})$ is the overlap length of adjacent membership function R_i and R_j, k is the number of membership function for quantitative attribute y. According to the encoding style, if $i = 1$, then $C_{i3} = L$ and $C_{i2} = y^{\min}$. If $j = k$, then $C_{j2} = y^{\max}$ and $C_{j1} = R$.

The coverage factor of the membership function in the chromosome C_q is defined as

$$coverage\_factor(C_q) = \frac{\max(y)}{range(R_1, \cdots, R_k)} \quad (7)$$

where $\max(y)$ is the maximum quantity of quantitative attribute y, range $(R_1, \cdots, R_k)$ is the coverage range of the membership functions.

1.4 Crossover operator

The Parent Centric BLX operator (PCBLX)[16] is used, which allows the offspring genes to be around the genes of one parent or around a wide zone determined by both parent genes. The PCBLX operator is described as follows. Assume that $\boldsymbol{X} = (x_1, \cdots, x_n)$ and $\boldsymbol{Y} = (y_1, \cdots, y_n)$, $(x_i, y_i \in [a, b] \subset R, i = 1, \cdots, n)$ are two real-coded chromosomes going to be crossed. The two following offsprings are generated:

(1) $\boldsymbol{P} = (p_1, \cdots, p_n)$, where p_i is a uniformly distributed random number from the interval $[l_i^1, u_i^1]$, with $l_i^1 = \max(a, x_i - I_i \cdot \alpha)$, $u_i^1 = \min(b, x_i + I_i \cdot \alpha)$, and $I_i = | x_i - y_i |$.

(2) $\boldsymbol{Q} = (q_1, \cdots, q_n)$, where q_i is a uniformly distributed random number from the interval $[l_i^2, u_i^2]$, with $l_i^2 = \max(a, y_i - I_i \cdot \alpha)$, $u_i^2 = \min(b, y_i + I_i \cdot \alpha)$. (In this paper, $\alpha = 0.5$).

The new generated offspring chromosome may not meet the inequality condition described in subsection 1.2. So the gene value of the new chromosome must be sorted. Assume that the number of membership function is 4. The new generated chromosome is denoted as $a[i][1:8]$. The pseudo-code of the procedure of sorting the chromosome is described as follows.

```
a[i][2:4] = sort (a[i][2:4]); //sort the gene value of a[i][2], a[i][3] and a[i][4] in ascending order;
a[i][5:7] = sort (a[i][5:7]); //sort the gene value of a[i][5], a[i][6] and a[i][7] in ascending order;
While (a[i][3] > a[i][6])
    swap (a[i][3], a[i][6]); //swapping the gene value of a[i][3] and a[i][6].
  a[i][2:4] = sort (a[i][2:4]);
  a[i][5:7] = sort (a[i][5:7]);
end
```

1.5 Restart approach

The CHC genetic model makes use of an incest prevention mechanism and a restarting process to provoke diversity in the population. During the reproduction step, two parents are crossed if their Hamming distance divided by 2 is over predefined threshold M. In order to calculate the Hamming distance of the two chromosomes, each gene value is transformed into a binary string. The length of each binary string is denoted as $Bitlengene$, which is the number of bit corresponding to the maximum value of the quantitative attribute y. The number of genes in the chromosome is denoted as Num_{genes}. The threshold value M is defined as

$$M = (Num_{genes} \cdot Bitlengene)/4 \quad (8)$$

If no offspring is obtained in one generation of CHC scheme, the threshold value M will be decremented by a 10% of its initial value in the approach. When M is below zero, the restart procedure is per-

formed. The evolving population is reinitialized by considering the best individual as the first chromosome of the new population and generating the remaining N_{pop} − 1 by randomly flipping 35% of their bits. When the new individual generated by the restart procedure does not meet the inequality constraint condition, it must execute the sorting procedure described in subsection 1.4.

2 The proposed incremental learning algorithm

The proposed learning approach consists of two phases. In phase 1, the best chromosomes during the execution of the CHC algorithm on the original dataset are collected. In phase 2, the incremental CHC algorithm on the new inserted dataset is executed. The initial population is composed of the chromosomes saved in phase 1 and the randomly generated chromosomes. The detailed steps of our approach are described below.

Input: $\boldsymbol{y} = \{y_1, y_2, \cdots, y_n\}$ the original dataset for quantitative attribute y, new inserted dataset $\{y_{n+1}, y_{n+2}, \cdots, y_{n+m}\}$, the number of membership function k, fuzzy support threshold β, and population size N_{pop}

Output: the best chromosome which represents the membership functions on the updated dataset $\{y_1, \cdots, y_{n+1}, \cdots, y_{n+m}\}$

Phase 1: genetic learning of the membership functions on $\{y_1, y_2, \cdots, y_n\}$

Step 1.1: Run the SPFCM clustering algorithm on $\{y_1, y_2, \cdots, y_n\}$, the number of cluster is k, generate N_{pop} chromosomes according to the principle described in subsection 1.2.

Step 1.2: Evaluate the population. For each chromosome C_q:

1.2.1 Each quantitative value $y_i (1 \leqslant i \leqslant n)$ is transferred into a fuzzy set f_i represented as $f_i = \left\{\frac{f_{i1}}{R_1} + \cdots + \frac{f_{ik}}{R_k}\right\}$ using the membership functions represented by the chromosome C_q, where R_k is the kth linguistic term of y, f_{jk} is the fuzzy membership value in region R_k.

1.2.2 For each linguistic term $R_j (1 \leqslant j \leqslant k)$, calculate its count as follows: $count_j = \sum_{i=1}^{n} f_{ij}$, check whether $count_j$ is larger than or equal to $\beta \cdot n$. If $R_j (1 \leqslant j \leqslant k)$ satisfies the condition, put it in the set of fuzzy frequent 1-itemsets L_1.

1.2.3 Calculate the fitness value of chromosome C_q using the fitness function described in subsection 1.3.

Step 1.3: Generate the next population Pop_{next}:

1.3.1 Shuffle the population, select the chromosomes two by two, crossover operator described in subsection 1.4 is executed if the Hamming distance between the selected chromosomes divided by 2 is over threshold value M.

1.3.2 Evaluate the generated offspring, join the parents with their offspring and select the best N_{pop} chromosomes as the next population Pop_{next}

Step 1.4: If the fitness value of the chromosome $C_{feasible}$ which has the highest fitness value in Pop_{next} is larger than all the fitness value of chromosome in feasible chromosome list (FCL), $C_{feasible}$ is added to FCL. (The initial number of chromosome in FCL is 0.)

Step 1.5: If there are no new generated offspring in Step 1.3.1, the threshold value M is decremented by 10% of its initial value.

Step 1.6: If $M < 0$, restart procedure described in subsection 1.5 is executed.

Step 1.7: If the maximum number of iteration is not reached, go to Step 1.3.

Phase 2: incremental learning of membership functions on $\{y_1, \cdots, y_{n+1}, \cdots, y_{n+m}\}$

Step 2.1: Generate the new initial population P_{new}:

2.1.1: Run the SPFCM clustering algorithm on $\{y_{n+1}, y_{n+2}, \cdots, y_{n+m}\}$ and derive the new cluster center.

2.1.2: The number of chromosome in FCL is denoted as N_{FCL}. If $N_{FCL} < N_{pop}/2$, all the chromosomes are added to P_{new}. After that, randomly generate $N_{pop} - N_{FCL}$ chromosomes according to the principle described in subsection 1.2.

2.1.3 If $N_{FCL} \geqslant N_{pop}/2$, select the best $N_{pop}/2$ chromosomes from FCL and add them to P_{new}. Randomly generate $N_{pop}/2$ chromosomes according to the principle described in subsection 1.2.

Step 2.2: Evaluate the new initial population P_{new}. Calculate the fitness value of C_q in P_{new} according to the method described in step 1.2. But only the new inserted dataset $\{y_{n+1}, y_{n+2}, \cdots, y_{n+m}\}$ is used. Each quantitative value $y_{n+i} (1 \leqslant i \leqslant m)$ is transferred into a fuzzy set f_i represented as $f_i = \left\{\frac{f_{i1}}{R_1} + \cdots + \frac{f_{ik}}{R_k}\right\}$ according to corresponding chromosome, and the count of the linguistic term $R_j (1 \leqslant j \leqslant k)$, is calculated as follows: $count_j = \sum_{i=1}^{m} f_{ij}$.

Step 2.3: Execute the CHC genetic process de-

scribed in Step 1.3, Step 1.5 to Step 1.7.

Step 2.4: Output the best chromosome in the final generation.

3 Experimental results

A real dataset called FAM95 (http://www.ics.uci.edu/~mlearn/) is used to assess the performance of the proposed incremental learning approach. There are 63,756 transactions in the dataset. Of the 23 attributes in the dataset, 3 quantitative attributes are chosen. They are age, head's personal income and family income. The experiments are performed on a computer with 3.1GHz processor and 4GB main memory and running the Microsoft Windows 7 operating system. The proposed incremental learning approach and the relevant algorithm are all implemented by Matlab 7.0.

3.1 Comparison of the proposed approach with the related batch approach

In the experiment, the proposed approach is compared with the batch CHC model proposed by Ref. [10] and the batch FCM-CHC model whose principle is the same as the approach proposed by Ref. [5]. The proposed incremental approach, the batch CHC model and the batch FCM-CHC model is abbreviated to Inc-CHC, B-CHC and B-FCM-CHC respectively. In B-FCM-CHC, FCM algorithm is used to cluster the quantitative values and CHC genetic procedure is used to adjust the cluster center which is regarded as the midpoint of the TMFs. For the convenience of comparison, the coding method and the genetic operator of the B-CHC and the B-FCM-CHC are the same as the Inc-CHC. In the Inc-CHC, the first 33,756 quantitative values are extracted from FAM95 dataset as original database. The rest of 30,000 quantitative values are inserted into the original database for the experiment. The other two approaches are executed on the 63,756 quantitative values directly. The parameter values of each CHC model are described as follows: the number of membership function for the quantitative attribute 4, the size of evolving population 50, the maximum iteration number 500, and the fuzzy support threshold value 0.1.

The three approaches are compared in terms of the number of generations converging to a solution and the quality of the solutions. Every approach is executed six times. The following experiment results are the average of six outcomes of the experiments. Figs 2 – 4 shows the average fitness values of the chromosomes in the population along with different number of generation of the Inc-CHC, B-CHC, and B-FCM-CHC for the quantitative attribute age, head's personal income and family income respectively. The variation trend for the Inc-CHC in Figs 2 – 4 represents genetic process of Phase 2 described in Section 3. Table 1 presents the number of generations converging to a solution and the maximum fitness value obtained by the three CHC model. Analyzing these experiment results, the following conclusions can be got: (1) For the quantitative age and family income, the maximum fitness value obtained by the B-FCM-CHC is higher than the value obtained by the B-CHC. The converging speed of the B-FCM-CHC is slightly faster than that of the B-CHC. So FCM algorithm plays an important role in the genetic process of learning the TMFs and it contributes to obtain high quality membership functions and decrease the size of the tuning search space. However, for the head's personal income, the difference between B-CHC and B-FCM-CHC is not very obvious. This is because that the midpoints of TMFs generated by the B-CHC and B-FCM-CHC are very close to the center generated by the FCM algorithm. (2) The Inc-CHC is superior to the B-CHC and the B-FCM-CHC in term of the number of generations converging to a solution. The maximum fitness value of the Inc-CHC is higher than that of B-CHC and is comparable to that of B-FCM-CHC even if the number of the converging generation is significantly less. The proposed incremental learning strategy is effective.

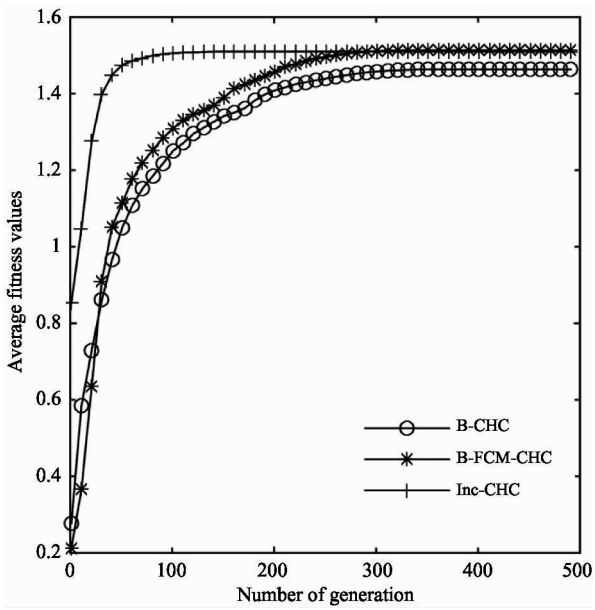

Fig. 2 The average fitness values along with numbers of generation for age

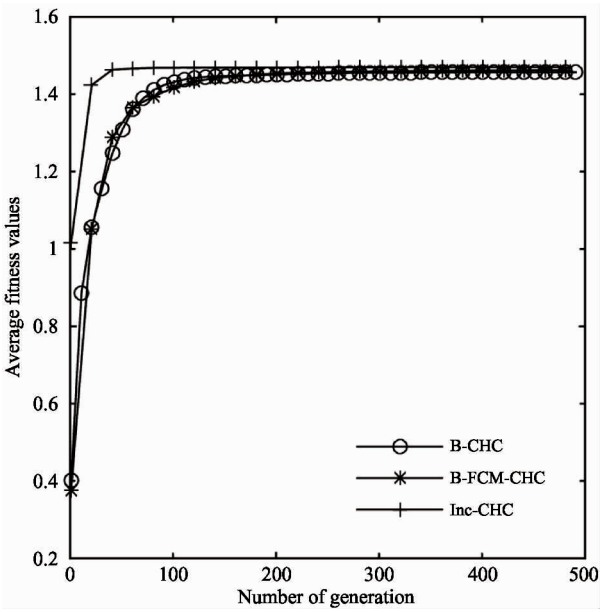

Fig. 3 The average fitness values along with numbers of generation for head's personal income

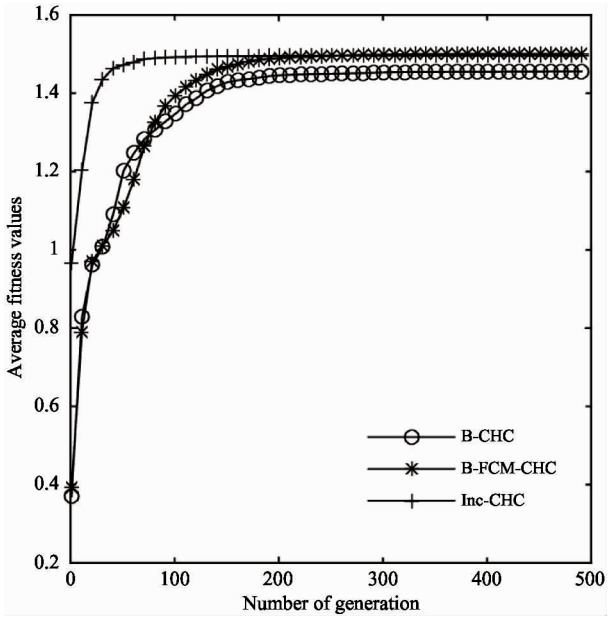

Fig. 4 The average fitness values along with numbers of generation for family income

Table 1 Number of converging generations and the maximum fitness value obtained by the three CHC model

Method	Age		Head's person income		Family income	
	Number of generations	Max fitness value	Number of generations	Max fitness value	Number of generations	Max fitness value
B-CHC	274.5	1.46	233.75	1.46	234.25	1.45
B-FCM-CHC	248	1.51	228.5	1.47	200.75	1.51
Inc-CHC	1	1.51	26.5	1.47	77.5	1.49

3.2 Comparison of Inc-CHC with the incremental approach based on CLUSTERDB*

The proposed approach Inc-CHC is compared with the incremental learning membership function approach based on CLUSTERDB* in Ref. [12]. It is abbreviated to Inc-CLUSTERDB*. In the experiment, the quantitative attribute values of head's personal income and family income are used. As before, the first 33,756 quantitative values are regarded as original dataset. The rest of 30,000 quantitative values are inserted into the original dataset. The triangular membership functions (TMFs) is used in Inc-CLUSTERDB*. The parameter values of Inc-CHC are the same as the values determined in subsection 3.1. So the cluster number is set as 4 in Inc-CLUSTERDB*. The second and the third cluster center, the minimum and the maximum quantitative attribute value are used as parameters value of TMFs. The fitness function defined in subsection 1.3 is used to assess the quality of TMFs generated by two methods. Table 2 shows the experiment results which include fitness values and time cost of the two methods. Then it is easy to draw the conclusions:

(1) The quality of TMFs generated by Inc-CHC is better than that of Inc-CLUSTERDB*. This is because that Inc-CHC can dynamically adjust and optimize TMFs by CHC genetic process. Inc-CLUSTERDB* generates TMFs just according to data distribution of quantitative values.

Table 2 The comparison of Inc-CHC with Inc-CLUSTERDB*

Method	Head's person income		Family income	
	Fitness value	Time cost (s)	Fitness value	Time cost (s)
Inc-CLUSTERDB*	0.78	32.09	1.17	38.3
Inc-CHC	1.47	18.96	1.49	20.61

(2) Time cost of Inc-CHC is less than that of the Inc-CLUSTERDB*. In order to obtain consistent partition, CLUSTERDB* algorithm must be re-applied to the whole dataset many times when the new inserted value is not between lower and upper bound of a cluster in Inc-CLUSTERDB*. So the higher time cost is required by Inc-CLUSTERDB*.

3.3 The effect of the number of the inserted quantitative values on the proposed approach

In this experiment, the effect of the number of the inserted quantitative values on the Inc-CHC is analyzed. The first 33,756 quantitative values are extracted from FAM95 dataset as original dataset. The inserted dataset is generated by sequentially selecting 7500, 15000, 22500 quantitative values from the remainder dataset respectively. The rest parameters values of the Inc-CHC are the same as that of the experiment described in subsection 4.1. Every experiment is also executed six times repeatedly. Average results of the experiments are presented in Tables 3 – 5. Tables 3 – 5 show the maximum fitness value obtained by the Inc-CHC and the number of iterative evaluation converging to a solution with different number of the inserted quantitative values for the three quantitative attribute respectively. It also shows the experiment results of B-FCM-CHC which executes on the 41256 (33756 + 7500), 48756 (33756 + 15000), 56256 (33756 + 22500) selected quantitative values directly for the three quantitative attributes respectively.

Table 3 The comparison of B-FCM-CHC and Inc-CHC with different number of inserted values for age

Method	Maximum fitness value			Number of converging generation		
	7500	15000	22500	7500	15000	22500
Inc-CHC	1.5	1.5	1.5	1	1	1
B-FCM-CHC	1.52	1.52	1.52	266.5	241	272

Table 4 The comparison of B-FCM-CHC and Inc-CHC with different number of inserted values for Head's person income

Method	Maximum fitness value			Number of converging generation		
	7500	15000	22500	7500	15000	22500
Inc-CHC	1.46	1.47	1.47	1	174	1
B-FCM-CHC	1.47	1.47	1.47	258.5	332	279.5

Table 5 The comparison of B-FCM-CHC and Inc-CHC with different number of inserted values for Family income

Method	Maximum fitness value			Number of converging generation		
	7500	15000	22500	7500	15000	22500
Inc-CHC	1.41	1.5	1.5	168	99.75	188.75
B-FCM-CHC	1.46	1.47	1.51	188	278	248

From these experiment results, it can be easily observed to the following phenomenon:

(1) For each quantitative attribute, in the case of different number of inserted quantitative values, the maximum fitness values of the Inc-CHC are comparable to that of the B-FCM-CHC. But the number of the generation converging to a solution is less than that of the B-FCM-CHC.

(2) The number of iterative evaluation converging to the maximum fitness value is very different when different number of inserted quantitative values is used for the quantitative attribute head's personal income and family income. However, for the attribute age, it just needs one generation to converge to the solution for all the different number of inserted values.

The main reason is analyzed as follows. When the distribution of new inserted quantitative values is the same as that of the original dataset, the chromosomes saved in the Phase 1 of the Inc-CHC are useful for the incrementally learning the membership functions. So a few number of generations converging to the solution is required by the Inc-CHC. Otherwise, when the distribution of the new inserted quantitative vales is very different from that of the original dataset, the optimal solution is shifted too much in the search space. A lot of number of generations converging to a solution is required. So it is concluded that the Inc-CHC is not sensitive to the number of the inserted values in terms of

learning the high quality of membership functions. However, the number of the inserted quantitative values is very crucial factor in order to decrease the number of iterative generations.

4 Conclusions

In this paper, an incremental approach of learning the TMFs for mining fuzzy association rule is proposed. In the process of genetic learning of TMFs, cluster centers resulting from single-pass fuzzy c means (SPFCM) algorithm give the CHC genetic model heuristic information, which contributes to enhance the quality of TMFs. Compared with the existing incremental approach, the proposed method is superior in terms of the quality of the generated TMFs and the time cost. The number of iterative generations converging to a solution of our approach is less than that of the batch learning approach. The number of the inserted quantitative values is a crucial factor in terms of decreasing the number of iterative converging generations. In future, how to determine the appropriate number of inserted numeric values will be studied for the proposed approach.

References

[1] Kalia H, Dehuri S, Ghosh A. A Survey on Fuzzy Association Rule Mining. *International Journal of Data Warehousing and Mining*, 2013, 9(1): 1-27

[2] Kaya M, Alhajj R, Polat F, et al. Efficient automated mining of fuzzy association rules. In: Proceedings of the International Conference on Database and Expert Systems with Applications, Aix-en-Provence, France, 2002. 133-142

[3] Mangalampalli A, Pudi V. FAR-HD: A fast and efficient algorithm for mining fuzzy association rules In large high-dimensional datasets. In: Proceedings of the 2013 IEEE International Conference on Fuzzy Systems, Hyderabad, India, 2013. 1-6

[4] Fu A W, Wong M H, Sze S C, et al. Finding fuzzy sets for the mining of fuzzy association rules for numerical attributes. In: Proceedings of International Symposium on Intelligent Data Engineering and Learning (IDEAL98), Hong Kong, China, 1998. 263-268

[5] Kaya M, Alhajj R. Genetic algorithm based framework for mining fuzzy association rules. *Fuzzy Sets and Systems*, 2005, 152(3): 587-601

[6] Hong T P, Chen C H, Wu Y L, et al. A GA-based fuzzy mining approach to achieve a trade-off between number of rules and suitability of membership functions. *Soft Computing*, 2006, 10(11): 1091-1101

[7] Chen C H, Tseng V S, Hong T P. Cluster-based evaluation in fuzzy-genetic data mining. *IEEE Transactions on Fuzzy Systems*, 2008, 16(1): 249-262

[8] Hong T P, Chen C H, Lee Y C, et al. Genetic-fuzzy data mining with divide-and-conquer strategy. *IEEE Transactions on Evolutionary Computation*, 2008, 12(2): 252-265

[9] Hong T P, Tung Y F, Wang S L, et al. A multi-level ant-colony mining algorithm for membership functions. *Information Sciences*, 2012, 182(1): 3-14

[10] Alcala-Fdez J, Alcala R, Gacto M J, et al. Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. *Fuzzy Sets and Systems*, 2009, 160(7): 905-921

[11] Palacios A M, Palacios J L, Sanchez L, et al. Genetic learning of the membership functions for mining fuzzy association rules from low quality data. *Information Sciences*, 2015, 295(358-378)

[12] Hachani N, Derbel I, Ounelli H. Automatic and Incremental Generation of Membership Functions. In: Proceedings of the 10th International Conference on Artificial Intelligence and Soft Computing, Pt I, Zakopane, Poland, 2010. 97-104

[13] Mansour N, Awad M, Elfakih K. Incremental genetic algorithm. *The International Arab Journal of Information Technology*, 2006, 3(1): 42-47

[14] Hore P, Hall L O, Goldgof D B. Single pass fuzzy c means. In: Proceedings of the IEEE International Conference on Fuzzy Systems, London, UK, 2007. 240-246

[15] Eshelman L J. The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. *Foundations of Genetic Algorithms*, 1991, 1: 265-283

[16] Lozano M, Herrera F, Krasnogor N, et al. Real-coded memetic algorithms with crossover hill-climbing. *Evolutionary Computation*, 2004, 12(3): 273-302

Huo Weigang, born in 1978. He received his Ph. D degrees in College of Computer and Control Engineering of Nankai University in 2011. He received M. S. degrees from Shanxi University in 2004. He also received B. S. degrees from Shanxi Normal University in 2001. His research interests include fuzzy classifier, fuzzy association rule mining.