A Genetic-Fuzzy Mining Approach for Items with Multiple Minimum Supports

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

Abstract—**In the past, we proposed a genetic-fuzzy data-mining algorithm for extracting both association rules and membership functions from quantitative transactions under a single minimum support. In real applications, different items may have different criteria to judge their importance. In this paper, we thus propose an algorithm which combines clustering, fuzzy and genetic concepts for extracting reasonable multiple minimum support values, membership functions and fuzzy association rules form quantitative transactions. It first uses the k-means clustering approach to gather similar items into groups. All items in the same cluster are considered to have similar characteristics and are assigned similar values for initializing a better population. Each chromosome is then evaluated by the criteria of requirement satisfaction and suitability of membership functions to estimate its fitness value. Experimental results also show the effectiveness and the efficiency of the proposed approach.**

I. Introduction

Data mining is commonly used for inducing association rules from transaction data. An association rule is an expression X→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 [1]. 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.

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 [2, 6, 7, 9, 16] have been proposed for the SSFM problem. Chan and Au proposed an F-APACS algorithm to mine fuzzy association rules [2]. They first transformed quantitative attribute values into linguistic terms and then used the adjusted difference analysis to find interesting associations among attributes. Kuok et al. proposed a fuzzy mining approach to handle numerical data in databases and derived fuzzy association rules [9]. At nearly the same time, Hong et al. proposed a fuzzy mining algorithm to mine fuzzy rules from quantitative transaction data [6]. 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. then extended the above concept to find fuzzy association rules with weighted items from transaction data [16]. They adopted Kohonen self-organized mapping to derive fuzzy sets for numerical attributes. As to the MSFM problem, Lee et al. proposed a mining algorithm which used multiple minimum supports to mine fuzzy association rules [10]. They assumed that items had different minimum supports and the minimum support for an itemset was set as the maximum of the minimum supports of the items contained in the itemset. Under the constraint, the characteristic of level-by-level processing was kept, such that the original Apriori algorithm could easily be extended to finding large itemsets.

In the above approaches, the membership functions were assumed to be known in advance. Although many approaches for learning membership functions were proposed [3, 12, 13, 14, 15], 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 [8]. Hong et al. also proposed a genetic-fuzzy data-mining algorithm for extracting both association rules and membership functions from quantitative transactions [5]. 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.

Most of the mentioned approaches were proposed for the SSFM problem. In this paper, we thus extend our previous approach [5] to solving the MSFM problem. We propose an algorithm which combines the clustering, fuzzy and genetic concepts to derive minimum support values, membership functions and association rules. The proposed approach first

Chun-Hao Chen is with the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, 701, Taiwan. (e-mail: chchen@idb.csie.ncku.edu.tw).

Tzung-Pei Hong is with the Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung, 811, Taiwan. (corresponding author; phone: +886+7+5919191; fax: +886+7+5919374; e-mail: tphong@nuk.edu.tw).

Vincent S. Tseng is with the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, 701, Taiwan. (e-mail: tsengsm@mail.ncku.edu.tw).

Chang-Shing Lee is with the Department of Computer Science and Information Engineering, National University of Tainan, 701, Taiwan. (leecs@mail.nutn.edu.tw)

1-4244-1210-2/07/$25.00 ©2007 IEEE.

uses the k-means clustering approach to gather similar items into groups. All items in the same cluster are considered to have similar characteristics and are assigned similar values for initializing a better population. The values include an appropriate number of linguistic terms for each item, its reasonable membership functions, and a range of its possible minimum support values. The proposed approach then generates and encodes each set of minimum support values and membership functions into a fixed-length string. Each chromosome is then evaluated by the criteria of requirement satisfaction and suitability of membership functions to estimate its fitness value. The proposed algorithm thus has two main advantages. The first one is that the proposed approach can derive an acceptable minimum support value and membership functions of each item for fuzzy association-rule mining. The second one is that the proposed approach can get a better initial population, including an appropriate number of linguistic terms and the minimum support value and membership functions of each item by using the clustering technique.

II. A Genetic-Fuzzy Mining Framework for Items with Multiple Minimum Supports

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

The proposed framework maintains a population of sets of minimum support values and membership functions, and uses the genetic algorithm to automatically derive the resulting one. It first uses the *k*-means clustering approach to gather similar items into groups. All items in the same cluster are considered to have similar characteristics and are assigned similar values when a population is initialized. The values (or initialization information) include an appropriate number of linguistic terms for each item, its reasonable membership functions, and a range of its possible minimum support values. It then generates and encodes each set of minimum support values and membership functions into a fixed-length string according to the initialization information. Each chromosome is then evaluated by the requirement satisfaction and the suitability of membership functions to estimate its fitness value. The evaluation results are utilized to choose appropriate chromosomes for mating. The offspring sets of membership functions and minimum support values then undergo recursive "evolution" until a good set (the highest fitness value) has been obtained. Finally, the derived minimum support values and membership functions are used to mine fuzzy association rules by the approach in [10]. The details are described in the next section.

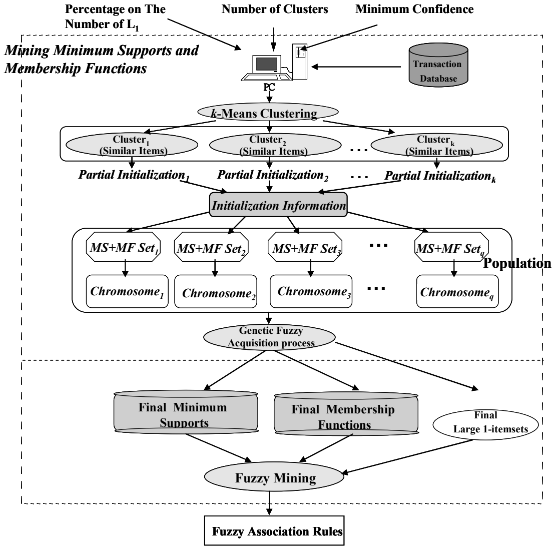
Percentage on The Number of L_1
Number of Clusters
Minimum Confidence
Mining Minimum Supports and Membership Functions
PC
Transaction Database
k-Means Clustering
Cluster_1 (Similar Items)
Cluster_2 (Similar Items)
...
Cluster_k (Similar Items)
Partial Initialization$_1$
Partial Initialization$_2$
...
Partial Initialization$_k$
Initialization Information
MS+MF Set$_1$
MS+MF Set$_2$
MS+MF Set$_3$
...
MS+MF Set$_q$
Population
Chromosome$_1$
Chromosome$_2$
Chromosome$_3$
...
Chromosome$_q$
Genetic Fuzzy Acquisition process
Final Minimum Supports
Final Membership Functions
Final Large 1-itemsets
Fuzzy Mining
Fuzzy Association Rules

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

III. The Proposed Genetic-Fuzzy Mining Approach

A. Chromosome Representation

It is important to encode minimum support values and membership functions as string representation for GAs to be applied to our problem. Several possible encoding approaches were described in the past [3, 11, 14, 15]. In our approach, each individual consists of two parts, respectively for minimum support values and membership functions. The first part encodes minimum support values by the real-number schema. Each real number represents the minimum support value of a certain item. Assume the minimum support value of item I_j is encoded with a real number α_j. The entire set of the minimum support values for all items is then formed by concatenating α_1, α_2, ..., α_m together, where *m* is the number of items. The second part handles the sets of membership functions for all the items. It also adopts the real-number schema. Here we assume the membership functions are isosceles-triangular for simplicity and use only two parameters to represent each membership function as Parodi and Bonelli [11] did. Fig. 2. shows the membership functions for item I_j, where R_{jk} denotes the membership function of the *k*-th 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.

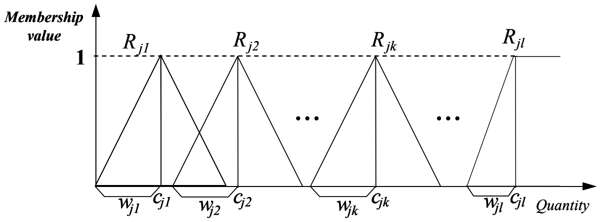

Fig. 2. Membership Functions of item I_j

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. An example is given below to demonstrate the process of encoding a set of minimum support values and membership functions.

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

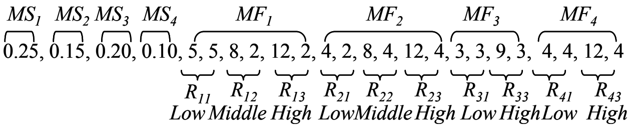
MS1 MS2 MS3 MS4 MF1 MF2 MF3 MF4
0.25, 0.15, 0.20, 0.10, 5, 5, 8, 2, 12, 2, 4, 2, 8, 4, 12, 4, 3, 3, 9, 3, 4, 4, 12, 4
R11 R12 R13 R21 R22 R23 R31 R33 R41 R43
Low Middle High Low Middle High Low High Low High

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

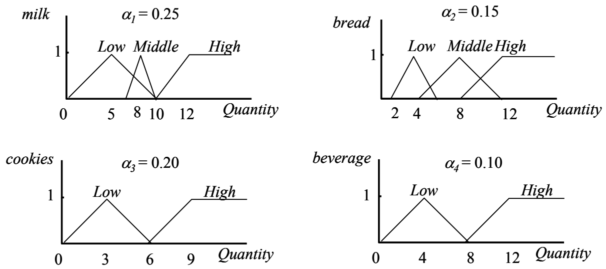

Fig. 4. The minimum support values and the membership functions for the chromosome represented in Fig. 3.

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

B. Initial Population

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

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

$$AQV_j = [\sum_{i=1}^{n} Quantity_{ij}]/(AN_j)\,,$$

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

$$SV_j = AN_j / n.$$

The clustering procedure for generating an initial population is stated as follows.

The clustering procedure for generating an initial population:

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

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

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

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

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

STEP 6: For each cluster $cluster_g$, g = 1 to k, calculate the appearing probability of each quantitative value in its corresponding interval.

STEP 7: Generate the second part of a population of P individuals according to the number of linguistic terms found in STEP 4 and the appearing probabilities of the quantitative values of each item found in STEP 6. Each interval corresponds to a linguistic term and the center of the membership function for the term is probabilistically selected according to the appearing probabilities of the quantitative values in the interval. Half the span (width) of the membership function is randomly determined from the range between 1 to the interval width.

After STEP 7, an initial population of individuals can thus be generated.

C. 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 [5].

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

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

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

D. 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:

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

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

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

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

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

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

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

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

where m is the number of items, *overlap_factor*(C_{qj}) represents the overlapping factor of the membership functions for an item I_j in the chromosome C_q, *coverage_factor*(C_{qj}) represents the coverage ratio of the membership functions for I_j, and *NumItem* is the number of items in the dataset. *overlap_factor*(C_{qj}) is the same as that in [5] and defined as follows:

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

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

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

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

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

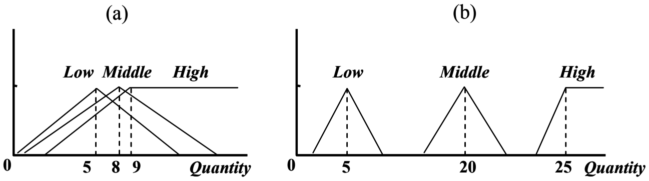

Fig. 5. Two bad membership functions

The overlap factor in *suitable*(C_q) is designed for avoiding the first bad case, and the coverage factor is for the second one.

E. 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 [4] and the *one-point mutation*, are used in the proposed genetic-fuzzy mining framework. Assume there are two parent chromosomes:

$$C_u^t = (c_1, \ldots, c_h, \ldots, c_z), \text{ and}$$
$$C_w^t = (c_1', \ldots, c_h', \ldots, c_z').$$

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

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

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

The one-point mutation operator will create a new fuzzy membership function by adding a random value ε (between $-w_{jk}$ to $+w_{jk}$) 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 $c + \varepsilon$ or $w + \varepsilon$ 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.

IV. The Proposed Mining Algorithm

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

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

STEP 1: Generate a population of *P* individuals by the clustering procedure stated in Section 3; each individual is a set of minimum support values and membership functions for all the *m* items.

STEP 2: Calculate the fitness value of each chromosome by the following substeps:

Substep 2.1: For each transaction datum D_i, i=1 to *n*, and for each item I_j, j=1 to *m*, transform the quantitative value $v_j^{(i)}$ into a fuzzy set $f_{jk}^{(i)}$ represented as:

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

using the corresponding membership functions represented by the chromosome, where R_{jk} is the *k*-th fuzzy region (term) of item I_j, $f_{jl}^{(i)}$ is $v_j^{(i)}$'s fuzzy membership value in region R_{jk}, and l (= $|I_j|$) is the number of linguistic terms for I_j.

Substep 2.2: For each item region R_{jk}, , $1 \le j \le m$, calculate its scalar cardinality on the transactions as follows:

$$count_{jk} = \sum_{i=1}^{n} f_{jk}^{(i)} .$$

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

$L_1 = \{R_{jk} \mid count_{jk} \ge \alpha_j,\ 1 \le j \le m$ and $1 \le k \le l\ \}$.

SubSTEP 2.4: Set the fitness value of the chromosome as the requirement satisfaction divided by *suitability*(C_q) as defined in previous section. That is:

$$f(C_q) = \frac{RS(C_q)}{Suitabilit\,y(C_q)} .$$

STEP 3: Execute the crossover operation on the population.

STEP 4: Execute the mutation operation on the population.

STEP 5: Use the Roulette-wheel selection operation to choose appropriate individuals for the next generation.

STEP 6: If the termination criterion is not satisfied, go to Step 2; otherwise, do the next step.

STEP 7: Get the set of minimum support values and membership functions with the highest fitness value.

STEP 8: Mine fuzzy association rules using the set of minimum support values and membership functions.

The set of minimum support values and membership functions are thus used to mine fuzzy association rules from the given database. The fuzzy mining algorithm proposed in [10] is then adopted to achieve this purpose.

V. 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 3.20GHz and 512MB RAM. 64 items and 10000 transactions were used in the experiments. In each data set, the numbers of purchased items in transactions were first randomly generated. The purchased items and their quantities in each transaction were then generated. An item could not be generated twice in a transaction. The initial population size *P* is set at 50, the cluster number *k* is set at 10, the crossover rate p_c is set at 0.8, and the mutation rate p_m is set at 0.001. The parameter *d* of the crossover operator is set at 0.35 according to Herrera *et al.*'s paper [4]. The percentage of expected number of large 1-itemsets is set at 0.8.

After 500 generations, the final membership functions are apparently much better than the original ones. For example, the initial minimum support values and membership functions of some two items among the 64 items are shown in Fig. 6.

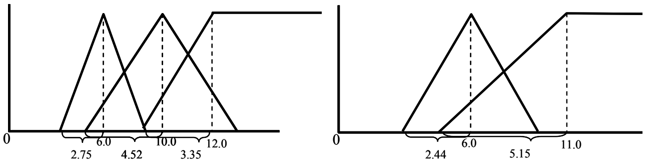

Fig. 6. The initial minimum support values and membership functions of some four items

In Fig. 6, 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 four items are shown in Fig. 7.

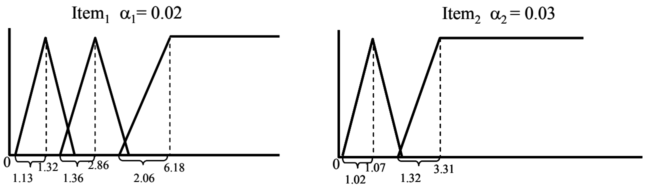

Fig. 7. The final minimum support values and membership functions of some four items after 500 generations

It is easily seen that the membership functions in Fig. 7. is better than those in Fig. 6. 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 are shown in Fig. 8. As expected, the curve gradually goes upward, finally converging to a fixed value.

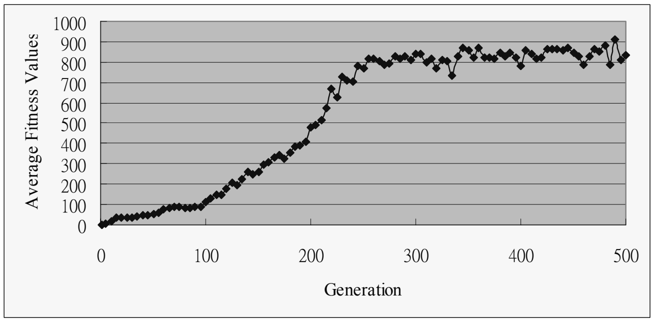

Fig. 8. The average fitness values along with different numbers of generations

VI. Conclusion and Future Works

In this paper, we have proposed a genetic-fuzzy mining algorithm for extracting multiple minimum support values, membership functions and fuzzy association rules from quantitative transactions. The proposed algorithm can adjust the minimum support value and membership functions for each item by genetic algorithms and use them to fuzzify quantitative transactions. The proposed algorithm has two main advantages. The first one is that the proposed approach can derive an acceptable minimum support value and membership functions of each item for fuzzy association-rule mining. The second one is that the proposed approach can get a better initial population, including an appropriate number of linguistic terms and the minimum support value and membership functions of each item by using the clustering technique. Experimental results also show that the effectiveness of the adopted clustering technique and the fitness function. In the future, we will continuously attempt to enhance the genetic-fuzzy mining framework for more complex problems.

Acknowledgment

This research was supported by the National Science Council of the Republic of China under contract NSC 95-2221-E-390-025.

References

[1] R. Agrawal and R. Srikant, "Fast algorithm for mining association rules," *The International Conference on Very Large Databases*, pp. 487-499, 1994.

[2] C. C. Chan and W. H. Au, "Mining fuzzy association rules," *The Conference on Information and Knowledge Management*, Las Vegas, pp. 209-215, 1997.

[3] O. Cordón, F. Herrera and P. Villar, "Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base," *IEEE Transactions on Fuzzy Systems*, Vol. 9, No. 4, pp. 667-674, 2001.

[4] F. Herrera, M. Lozano and J. L. Verdegay, "Fuzzy connectives based crossover operators to model genetic algorithms population diversity," *Fuzzy Sets and Systems,* Vol. 92, No. 1, pp. 21-30, 1997.

[5] T. P. Hong, C. H. Chen, Y. L. Wu and Y. C. Lee, "A GA-based fuzzy mining approach to achieve a trade-off between number of rules and suitability of membership functions," *Soft Computing*, Vol. 10, No. 11, pp. 1091-1101. 2006.

[6] T. P. Hong, C. S. Kuo and S. C. Chi, "A data mining algorithm for transaction data with quantitative values," The Eighth International Fuzzy Systems Association World Congress, pp. 874-878, 1999.

[7] T. P. Hong, C. S. Kuo and S. C. Chi, "Trade-off between time complexity and number of rules for fuzzy mining from quantitative data," *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems,* Vol. 9, No. 5, pp. 587-604, 2001.

[8] M. Kaya and R. Alhajj, "A clustering algorithm with genetically optimized membership functions for fuzzy association rules mining," *The IEEE International Conference on Fuzzy Systems*, pp. 881-886, 2003.

[9] C. Kuok, A. Fu and M. Wong, "Mining fuzzy association rules in databases," *SIGMOD Record*, Vol. 27, No. 1, pp. 41-46, 1998.

[10] Y. C. Lee, T. P. Hong and W. Y. Lin, "Mining fuzzy association rules with multiple minimum supports using maximum constraints", *Lecture Notes in Computer Science*, Vol. 3214, pp. 1283-1290, 2004.

[11] A. Parodi and P. Bonelli, "A new approach of fuzzy classifier systems," *The Fifth International Conference on Genetic Algorithms*, Morgan Kaufmann, Los Altos, CA, pp. 223-230, 1993.

[12] H. Roubos and M. Setnes, "Compact and transparent fuzzy models and classifiers through iterative complexity reduction," *IEEE Transactions on Fuzzy Systems*, Vol. 9, No. 4, pp. 516-524, 2001.

[13] M. Setnes and H. Roubos, "GA-fuzzy modeling and classification: Complexity and performance," *IEEE Transactions on Fuzzy Systems*, Vol. 8, No. 5, pp. 509-522, 2000.

[14] C. H. Wang, T. P. Hong and S. S. Tseng, "Integrating fuzzy knowledge by genetic algorithms," *IEEE Transactions on Evolutionary Computation,* Vol. 2, No.4, pp. 138-149, 1998.

[15] C. H. Wang, T. P. Hong and S. S. Tseng, "Integrating membership functions and fuzzy rule sets from multiple knowledge sources," *Fuzzy Sets and Systems,* Vol. 112, pp. 141-154, 2000.

[16] S. Yue, E. Tsang, D. Yeung and D. Shi, "Mining fuzzy association rules with weighted items," *The IEEE International Conference on Systems, Man and Cybernetics*, pp. 1906-1911, 2000.