ELSEVIER

Fuzzy Sets and Systems 112 (2000) 141–154

FUZZY sets and systems

www.elsevier.com/locate/fss

Integrating membership functions and fuzzy rule sets from multiple knowledge sources

Ching-Hung Wang[a], Tzung-Pei Hong[b,*], Shian-Shyong Tseng[c]

[a] *Chunghwa Telecommunication Laboratories, Chung-Li, 32617, Taiwan, ROC*

[b] *Department of Information Management, I-Shou University, Kaohsiung, 84008, Taiwan, ROC*

[c] *Institute of Computer and Information Science, National Chiao-Tung University, Hsin-Chu, 30050, Taiwan, ROC*

Received June 1997; received in revised form November 1997

Abstract

In this paper, we propose a GA-based fuzzy knowledge-integration framework that can simultaneously integrate multiple fuzzy rule sets and their membership function sets. The proposed two-phase approach includes fuzzy knowledge encoding and fuzzy knowledge integration. In the encoding phase, each fuzzy rule set with its associated membership functions is first transformed into an intermediary representation, and further encoded as a string. The combined strings form an initial *knowledge population*, which is then ready for integration. In the knowledge-integration phase, a genetic algorithm is used to generate an optimal or nearly optimal set of fuzzy rules and membership functions from the initial knowledge population. The hepatitis diagnostic problem was used to show the performance of the proposed knowledge-integration approach. Results show that the fuzzy knowledge-base resulting from using our approach performs better than every individual knowledge base. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Expert systems; Fuzzy machine learning; Fuzzy sets; Knowledge acquisition; Membership functions; Operators

1. Introduction

Expert systems have been successfully applied to many fields and have shown excellent performance. Knowledge-base construction remains, however, one of the major costs in building an expert system even though many tools have been developed to help with knowledge acquisition. Building a knowledge-based system usually entails constructing new knowledge bases from scratch. The cost of the effort is high and will become prohibitive as we attempt to build larger and larger systems. Reusing and integrating available knowledge from a variety of sources, such as domain experts, historical documentary evidence, current records, or existing knowledge bases, thus plays an important role in building effective knowledge-based systems [1, 10, 13, 19]. Especially for complex application problems, related domain knowledge is usually distributed among multiple sites, and no single site may have complete domain knowledge. The use of knowledge integrated from multiple knowledge sources is thus especially important to ensure comprehensive coverage.

Many knowledge acquisition and integration systems [2, 9, 21] based on the Personal Constructs Psychology (PCP) model [15] or Integrity Constraints [1, 19] have been developed. Recently, genetic

[*] Corresponding author.
E-mail address: tphong@csa500.isu.edu.tw (T.-P. Hong)

PII: S0165-0114(97)00385-0

algorithms have also been used to derive knowledge from training instances [3, 6, 12]. In [28, 29], Wang et al. proposed a GA-based knowledge-integration strategy that automatically integrates multiple rule sets in a distributed-knowledge environment. A self-integrating knowledge-based brain-tumor diagnostic system that uses this method was also developed [25]. In this paper, we attempt to generalize it to fuzzy domains.

Most knowledge sources or actual instances in real-world applications contain fuzzy or ambiguous information. Especially in domains such as medical or control domains, the boundaries of a piece of information used may not be clearly defined. Expressions of the domain knowledge using fuzzy descriptions are thus seen more and more frequently. Several researchers have recently investigated automatic generation of fuzzy classification rules and fuzzy membership functions using evolutionary algorithms [3, 16, 18, 23]. These methods can be categorized into the following four types:

1. learning fuzzy membership functions with fixed fuzzy rules [14];
2. learning fuzzy rules with fixed fuzzy membership functions [23, 30];
3. learning fuzzy rules and membership functions in stages [16] (i.e., first evolving good fuzzy rule sets using fixed membership functions, then tuning membership functions using the derived fuzzy rule sets);
4. learning fuzzy rules and membership functions simultaneously [3, 18].

In this paper, we propose a GA-based fuzzy knowledge-integration framework that can effectively integrate multiple fuzzy knowledge sources into a single knowledge base. The hepatitis diagnostic problem [4] was used to show the performance of the proposed knowledge-integration approach. Results show that the fuzzy knowledge base that results from using our approach performs better than every individual knowledge base. Knowledge integration is thus a successful application of genetic algorithms.

The remainder of this paper is organized as follows. Some GA-based classifier systems are reviewed in Section 2. A GA-based fuzzy knowledge-integration framework is proposed in Section 3. The fuzzy knowledge encoding approach used in the proposed framework is explained in Section 4. Our fuzzy knowledge integration approach is proposed in Section 5. Experiments on the diagnosis of hepatitis are stated in Section 6. Conclusions and future work are given in Section 7.

2. Review of GA-based classifier systems

In this section, we review two famous approaches commonly used by genetic algorithms as classifier systems, the Michigan approach and the Pittsburgh approach.

2.1. The Michigan approach

Cognitive system one (CS-1) was the first *Michigan genetic classifier system.* It was devised by Holland and Reitman in 1983 [12]. CS-1 maintains a population in which each individual is a rule, and is encoded as a fixed-length string. A fitness function is defined to evaluate the goodness (called the strength) of each rule in the population. The genetic algorithm then operates on the level of individual rules and selects good parent rules for mating according to their strength values.

The major problem with the Michigan approach is the simultaneous cooperation and competition of the individual rules within the population. During evolution, rules within the population compete with each other, and the ones with high strength values are selected for mating to generate new offspring rules. At the same time, rules within the population must cooperate to solve the given problem. Maintenance and evolution of such a set of co-adapted rules by considering both factors mentioned above is thus vital to the success of the Michigan approach.

2.2. The Pittsburgh approach

Learning system one (LS-1) was the first *Pittsburgh genetic classifier system.* It was proposed by Smith in 1980 [22]. LS-1 maintains a population in which each individual is a rule set, and is encoded as a *variable-length* string. A fitness function is defined to evaluate the goodness of each rule set in the population. The genetic algorithm then operates on the level of individual rule sets and selects good parent rule sets for mating according to their fitness values.

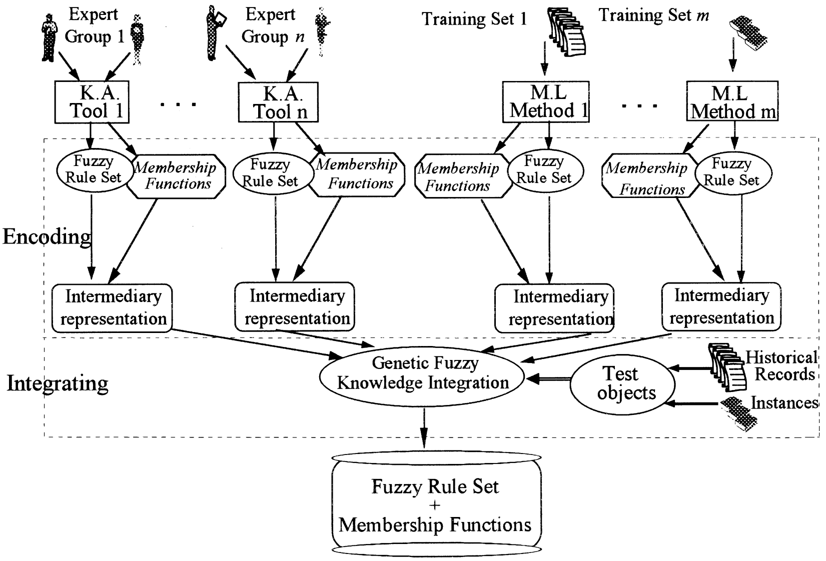

Fig. 1. A GA-based fuzzy knowledge-integration framework.

The major problem with the Pittsburgh approach is the maintenance and evaluation of a population of rule sets. It often leads to a much greater computational burden (in terms of both memory and processing time). Also, since credit assignment occurs on the level of rule sets by a predefined evaluation function, we may obtain only the fitness value of rule sets. It cannot help us promote the performance of individual rules. This is another problem of the Pittsburgh approach.

Application of genetic algorithms to the Pittsburgh approach is apparently quite different applying them to the Michigan approach. For the knowledge integration task, representation based on the Pittsburgh approach is preferred since knowledge at each different site is a rule set. In [28, 29], we proposed a GA-based knowledge-integration framework based on the Pittsburgh approach for integrating unambiguous knowledge. In this paper, we extend it to managing fuzzy knowledge integration.

3. A GA-based fuzzy knowledge-integration framework

Here, we propose a GA-based fuzzy knowledge-integration framework that integrates information from various fuzzy knowledge sources into a single knowledge base. The proposed framework can integrate multiple fuzzy rule sets and membership function sets at the same time. The proposed framework is shown in Fig. 1.

Fuzzy rule sets, membership functions, and test objects including instances and historical records may be distributed among various sources. Knowledge from each site might be directly obtained by a group of human experts using knowledge-acquisition tools, or derived using machine-learning methods. Here, we assume that all knowledge sources are represented by fuzzy rules since almost all knowledge derived by knowledge-acquisition tools or induced by machine-learning methods may easily be translated into or represented by rules.

The proposed framework maintains a population of fuzzy rule sets with their membership functions, and uses the genetic algorithm to automatically derive the resulting fuzzy knowledge base. It operates in two phases: *fuzzy knowledge encoding* and *fuzzy knowledge integration*. The encoding phase first transforms each fuzzy rule set and its associated membership functions into an intermediary representation, which is further encoded as a variable-length string. The integration phase then chooses appropriate strings for “mating”, gradually creating good offspring fuzzy

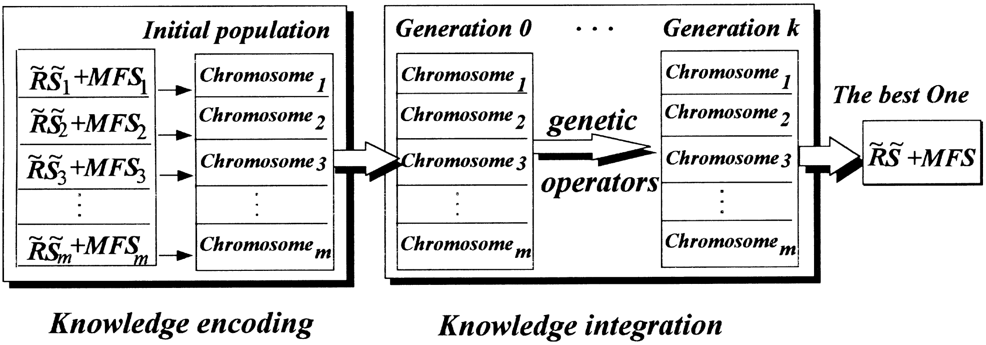

Fig. 2. Genetic fuzzy knowledge integration.

rule sets and membership function sets. The offspring fuzzy rule sets with their associated membership functions then undergo recursive “evolution” until an optimal or nearly optimal set of fuzzy rules and membership functions has been obtained. Fig. 2 shows the two-phases process, where $\tilde{RS}_1 + MFS_1$, $\tilde{RS}_2 + MFS_2, \ldots, \tilde{RS}_m + MFS_m$ are the fuzzy rule sets with their associated membership function sets, as obtained from different sources for integration.

4. Fuzzy knowledge encoding

In order to apply GAs to integration of multiple fuzzy rule sets, we need a powerful description language to represent complex rule sets and to map them easily into string representations. One of the most popular representation of rules, the conjunctive normal form, is then chosen as our description language to express each fuzzy rule set. Since the fuzzy rule sets with their associated membership functions are obtained from different sources, they may differ in size. Representation of variable-length rule sets is thus preferred here. Each fuzzy rule set with its associated membership functions is encoded as a variable-length chromosome by the *Pittsburgh approach.* However, each fuzzy rule set must first be translated into a uniform intermediary representation to preserve the syntactic and semantic constraints of the fuzzy rule sets before encoding. The steps for translating fuzzy rule sets into intermediary representations are described below.

1. Collect the features and possible values occurring in the condition parts of the fuzzy rule sets. All features gathered together comprise the global feature set.
2. Collect classes occurring in the conclusion parts of the fuzzy rule sets. All classes gathered together comprise the global class set.
3. Translate each fuzzy rule into an intermediary representation that retains its essential syntax and semantics. If some features in the global feature set are not used by the fuzzy rule, *dummy* tests are inserted into the condition part of the fuzzy rule. Each rule in the intermediary representation is then composed of *N feature tests* and one *class pattern*, where N is the number of global features collected.
4. Concatenate all intermediary representations of rules to form the representation of a rule set.

An example is given below to demonstrate the translation process of forming intermediary representations.

Example 1. Fisher’s Iris data [8] are used to demonstrate the translation process of forming intermediary representations. There are three species of Iris Flowers to be distinguished: *Setosa*, *Versicolor* and *Virginica.* A class domain D_{flower} is defined as $D_{\text{flower}} = \{Setosa,\ Versicolor,\ Virginica\}$. Each rule is described by four features: *Sepal Length* (*S.L.*), *Sepal Width* (*S.W.*), *Petal Length* (*P.L.*), and *Petal Width* (*P.W.*). Each feature has the possible linguistic values shown below.

$$D_{S.L.} = \{Short,\ Medium,\ Long\},$$

$$D_{S.W.} = \{Narrow,\ Medium,\ Wide\},$$

$$D_{P.L.} = \{Short,\ Medium,\ Long\},$$

$$D_{P.W.} = \{Narrow,\ Medium,\ Wide\}.$$

Assume a fuzzy rule set $\tilde{R}\tilde{S}_q$ obtained from a fuzzy knowledge source has the following four rules:

$\tilde{r}_{q1}$: IF (*P.L.* = *Short*), then *Class* is *Setosa.*

$\tilde{r}_{q2}$: IF (*P.L.* = *Long*), then *Class* is *Virginica.*

$\tilde{r}_{q3}$: IF (*P.W.* = *Medium*), then *Class* is *Versicolor.*

$\tilde{r}_{q4}$: IF (*P.W.* = *Wide*), then *Class* is *Virginica.*

After translation, the intermediary representations of these rules would then be constructed as follows:

$\tilde{r}'_{q1}$: IF ($\underline{S.L. = Short\ or\ Medium\ or\ Long}$) and ($\underline{S.W. = Narrow\ or\ Medium\ or\ Wide}$) and (*P.L.* = *Short*) and ($\underline{P.W. = Narrow\ or\ Medium\ or\ Wide}$), then *Class* is *Setosa.*

$\tilde{r}'_{q2}$: IF ($\underline{S.L. = Short\ or\ Medium\ or\ Long}$) and ($\underline{S.W. = Narrow\ or\ Medium\ or\ Wide}$) and (*P.L.* = *Long*) and ($\underline{P.W. = Narrow\ or\ Medium\ or\ Wide}$), then *Class* is *Virginica.*

$\tilde{r}'_{q3}$: IF ($\underline{S.L. = Short\ or\ Medium\ or\ Long}$) and ($\underline{S.W. = Narrow\ or\ Medium\ or\ Wide}$) and ($\underline{P.L. = Short\ or\ Medium\ or\ Long}$) and (*P.W.* = *Medium*), then *Class* is *Versicolor.*

$\tilde{r}'_{q4}$: IF ($\underline{S.L. = Short\ or\ Medium\ or\ Long}$) and ($\underline{S.W. = Narrow\ or\ Medium\ or\ Wide}$) and ($\underline{P.L. = Short\ or\ Medium\ or\ Long}$) and (*P.W.* = *Wide*), then *Class* is *Virginica.*

The tests with underlines are *dummy tests.* Also, $\tilde{r}'_{qi}$ is logically equivalent to $\tilde{r}_{qi}$, for $i = 1, \ldots, 4$. After translation, each intermediary representation of the rule is then composed of four feature tests and one class pattern.

Although the intermediary representation may include irrelevant tests and increase search space during integration, it can easily map each intermediary rule into a fixed-length string representation. The condition part of each intermediary rule is a conjunctive form with internal disjunctions that can describe complex rules. Irrelevant tests can also be removed by the knowledge decoding process after integration.

After each rule set has been translated into an intermediary representation, an appropriate data structure must then be designed to encode both the fuzzy rule sets and their membership function sets. Several strategies for representing fuzzy knowledge structures in conceptual learning were proposed [3, 20]. Here, we represent each membership function with two parameters as Parodi and Bonelli [20] did. Membership functions applied to a fuzzy rule set are assumed to be isosceles-triangle functions, as shown in Fig. 3, where c_{ij} is the abscissa with the highest membership value of the jth linguistic value (a_{ij}) of feature A_i, and w_{ij} represents half the spread of the membership function. A linguistic value a_{ij} of the feature A_i is then represented as a pair (c_{ij}, w_{ij}).

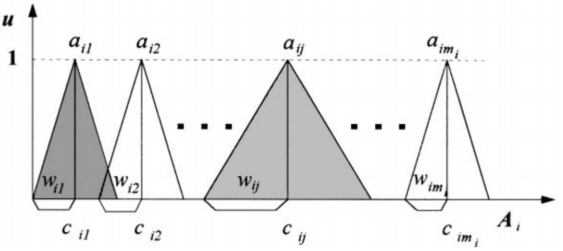

Fig. 3. Membership functions of feature A_i.

Assume that an intermediary rule $\tilde{r}'_{qk}$ is composed of N feature tests and one class pattern. Each feature test in a fuzzy rule is then encoded as m_i pairs of (c, w)'s, where m_i is the number of possible linguistic values of A_i, and the pair of (c, w) represents one possible linguistic value. If a fuzzy test value "$A_i = a_{ij}$" exists in a rule, the test is then encoded as $(c_{i1}, -w_{i1})(c_{i2}, -w_{i2}) \ldots (c_{ij}, w_{ij}) \ldots (c_{i_{m_i}}, -w_{i_{m_i}})$. Only w_{ij} is positive, the other w values have minus signs added to them. Similarly, if the fuzzy test is ("$A_i = a_{ij}$" or "$A_i = a_{ik}$"), then only w_{ij} and w_{ik} are positive and the other w values are negative. From the sign of w, the encoded string can then correctly represent the test condition of a fuzzy rule. The condition part of each fuzzy rule is then encoded as $\sum_{i=1}^{N} m_i$ pairs of (c, w)'s using the proposed encoding methods.

Next, the conclusion part of each fuzzy rule is encoded as a bit substring $(\varphi_1 \cdots \varphi_x)$, where x is the number of possible classes. When the rule points to class j, then φ_j is set as 1 and the others are set as 0. The rule $\tilde{r}_{qk}$ is then encoded as shown in Fig. 4.

In Fig. 4, the substring $(c_{i1}^{qk} w_{i1}^{qk} \cdots c_{ij}^{qk} w_{ij}^{qk} \cdots c_{i_{m_i}}^{qk} w_{i_{m_i}}^{qk})$ represents the membership functions of m_i possible linguistic values for feature A_i in rule $\tilde{r}_{qk}$, and

$$\tilde{r}'_{qk}: \overbrace{(\underbrace{c^{qk}_{11} w^{qk}_{11}}_{a_{11}} \cdots \underbrace{c^{qk}_{1m_1} w^{qk}_{1m_1}}_{a_{1m_1}})}^{A_1} \cdots \overbrace{(\underbrace{c^{qk}_{i1} w^{qk}_{i1}}_{a_{i1}} \cdots \underbrace{c^{qk}_{ij} w^{qk}_{ij}}_{a_{ij}} \cdots \underbrace{c^{qk}_{im_i} w^{ik}_{im_i}}_{a_{im_i}})}^{A_i} \cdots \overbrace{(\underbrace{c^{qk}_{N1} w^{qk}_{N1}}_{a_{N1}} \cdots \underbrace{c^{qk}_{Nm_N} w^{qk}_{Nm_N}}_{a_{Nm_N}})}^{A_N} \overbrace{(\varphi^{qk}_1, \cdots \varphi^{qk}_x)}^{Class}$$

Fig. 4. String representation of $\tilde{r}'_{qk}$.

$$\overbrace{\overbrace{\overbrace{(\underbrace{c^{q1}_{11} w^{q1}_{11}}_{a_{11}} \cdots \underbrace{c^{q1}_{1m_1} w^{q1}_{1m_1}}_{a_{1m_1}})}^{A_1} \cdots \overbrace{(\underbrace{c^{q1}_{N1} w^{q1}_{N1}}_{a_{N1}} \cdots \underbrace{c^{q1}_{Nm_N} w^{q1}_{Nm_N}}_{a_{NmN}})}^{A_N} \overbrace{(\varphi^{q1}_1 \cdots \varphi^{q1}_x)}^{Class}}^{\tilde{r}'_{q1}} \cdots \overbrace{\overbrace{(\underbrace{c^{qk}_{11} w^{qk}_{11}}_{a_{11}} \cdots \underbrace{c^{qk}_{1m_1} w^{qk}_{1m_1}}_{a_{1m_1}})}^{A_1} \cdots \overbrace{(\underbrace{c^{qk}_{N1} w^{qk}_{N1}}_{a_{N1}} \cdots \underbrace{c^{qk}_{Nm_N} w^{qk}_{Nm_N}}_{a_{NmN}})}^{A_N} \overbrace{(\varphi^{qk}_1 \cdots \varphi^{qk}_x)}^{Class}}^{\tilde{r}'_{qk}}}^{\tilde{R}\tilde{S}_q}$$

Fig. 5. String representation of fuzzy rule set $\tilde{R}\tilde{S}_q$.

the substring $(\varphi^{qk}_1 \cdots \varphi^{qk}_x)$ represents its output class. Since c_{ij} and w_{ij} are both numeric values, fuzzy rules and their fuzzy membership function sets are then encoded as fixed-length real-number strings rather than bit strings. Each fuzzy rule set $\tilde{R}\tilde{S}_q$ that contains k fuzzy rules is then encoded by concatenating strings of k fuzzy rules (Fig. 5).

An example is given below to demonstrate the encoding of fuzzy knowledge.

Example 2. Continuing from Example 1, assume the fuzzy rule set $\tilde{R}\tilde{S}_q$ is to be encoded. Assume that the membership functions used for each feature are as shown in Fig. 6.

Using the proposed intermediary representation, the fuzzy rule $\tilde{r}'_{q1}$ in Example 1 is encoded as shown in Fig. 7.

Since feature *S.L.* in $\tilde{r}'_{q1}$ has three disjunctive test values, *Short*, *Medium* and *Long*, the tests for *S.L.* are then encoded as “5.2, 0.9, 6.1, 0.9, 7.0, 0.9” according to the membership functions given in Fig. 6. *S.W.* also has three disjunctive test values, *Narrow*, *Medium* and *Wide*, and is then encoded as “2.6, 0.6, 3.2, 0.6, 3.8, 0.6”. Similarly, *P.W.* is encoded as “0.7, 0.6, 1.3, 0.6, 1.9, 0.6”. But *P.L.*, has only one test value, *Short*. It is then encoded as “2.4, 1.5, 3.9, −1.5, 5.4, −1.5”, where the two negative w values indicate “*P.L.* = *Medium*” and “*P.L.* = *Long*” are not in the condition part of rule $\tilde{r}'_{q1}$.

This representation allows genetic operators (defined later) to easily integrate multiple fuzzy rule sets and their fuzzy membership function sets at the same time. Furthermore, since fuzzy membership functions are encoded together with each rule (as opposed to a global collection of membership functions for all rules), rules are permitted to evolve to different degrees of vagueness as Carse et al. proposed [3]. The advantage of this representation is the expressive power for the derived rules to possess their own specificity in terms of the fuzzy sets they relate to. Especially for multi-dimensional domains that cannot generally use a global collection of membership functions for all rules, the use of local fuzzy sets to perform a linguistic interpretation of individual rules seems valid to overcome “curse of dimensionality” [3]. However, this advantage is at the cost of an increase of the search space. A more detailed discussion about using local fuzzy sets can be found in [3, 5, 18].

5. Fuzzy knowledge integration

After each fuzzy rule set with its associated membership functions has been encoded as a variable-length string (an individual in the initial population), the genetic-fuzzy knowledge-integration process starts. It chooses good individuals in the population for “mating”, gradually creating better *offspring* fuzzy rule sets. During evolution, a measure function and a set of test objects are used to evaluate the fitness value of each “offspring” fuzzy rule set. The *offspring* fuzzy rule sets then undergo recursive “evolution” until a really good *fuzzy knowledge base* has been produced. Domain experts thus need not intervene in

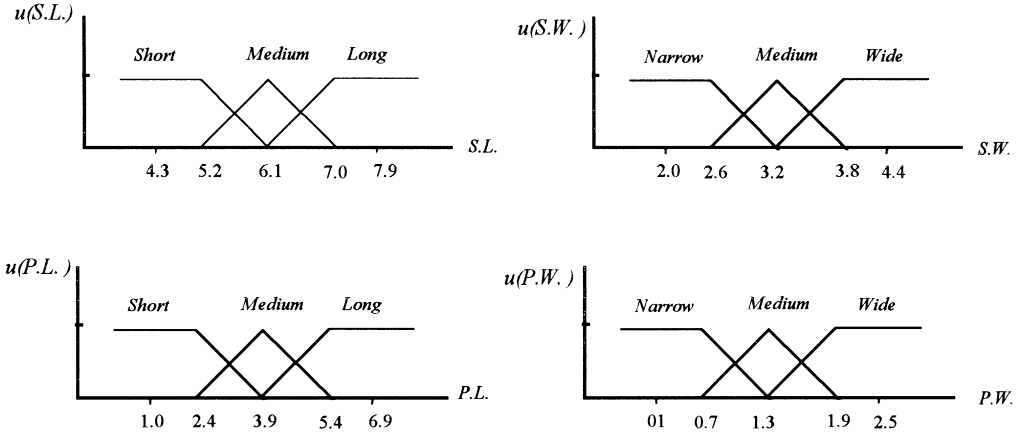

Fig. 6. Membership functions for Example 2.

$$\tilde{r}'_{q1}: \overbrace{5.2,0.9,6.1,0.9,7.0,0.9}^{S.L}\ \overbrace{2.6,0.6,3.2,0.6,3.8,0.6}^{S.W}\ \overbrace{2.4,1.5,3.9,-1.5,5.4,-1.5}^{P.L}\ \overbrace{0.7,0.6,1.3,0.6,1.9,0.6}^{P.W}\ \overbrace{100}^{Class}$$

Fig. 7. String representation of $\tilde{r}'_{q1}$.

the integration process. Notation and definitions used are given as below.

5.1. Notation and definitions

Definition 1. A fuzzy test $\tilde{s}_k$ is represented as $[A_k\ r\ v]$, where A_k is a feature, r is a relationship, and v is a fuzzy linguistic value. For example "color = reddish" and "height = tall" are both fuzzy tests.

Definition 2. $u_{\tilde{s}_k}(\tilde{e})$ represents the degree to which object $\tilde{e}$ is matched by $\tilde{s}_k$. The value of $u_{\tilde{s}_k}(\tilde{e})$ ranges between 0 and 1; 0 indicates complete exclusion and 1 indicates complete inclusion.

Definition 3. Assume the condition part $\tilde{c}_j$ of rule $\tilde{r}_j$ consists of j_m tests, $\tilde{s}_{j_1} \wedge \tilde{s}_{j_2} \wedge \cdots \wedge \tilde{s}_{j_m}$. The degree of object $\tilde{e}$ matched by $\tilde{c}_j$ is evaluated as

$$u_{\tilde{c}_j}(\tilde{e}) = u_{\tilde{s}_{j_1}}(\tilde{e}) \wedge u_{\tilde{s}_{j_2}}(\tilde{e}) \wedge \cdots \wedge u_{\tilde{s}_{j_m}}(\tilde{e}),$$

or more generally,

$$u_{\tilde{c}_j}(\tilde{e}) = u_{\tilde{s}_{j_1}}(\tilde{e})\, \tau\, u_{\tilde{s}_{j_2}}(\tilde{e})\, \tau \cdots \tau\, u_{\tilde{s}_{j_m}}(\tilde{e}),$$

where τ is a *t-norm* operator.

Definition 4. The classification of an object $\tilde{e}$ judged by a rule $\tilde{r}_j$ $(\tilde{c}_j \Rightarrow \tilde{\delta}_j)$ is $\tilde{\delta}_j$, with a membership value $u_{\tilde{c}_j}(\tilde{e})$.

Definition 5. The classification of an object $\tilde{e}$ judged by a rule set $\tilde{R}\tilde{S}$ is $\tilde{\delta}_j$, if a rule concluding to $\tilde{\delta}_j$ $(\tilde{c}_j \Rightarrow \tilde{\delta}_j)$ has the highest $u_{\tilde{c}_j}(\tilde{e})$ among all rules. If $\tilde{e}$ is classified by $\tilde{R}\tilde{S}$ into several classes that have the highest degree, the classification of an object $\tilde{e}$ is then shared among them.

Definition 6. An object is *correctly matched* by rule set $\tilde{R}\tilde{S}$ if the original object class is equal to the class judged by $\tilde{R}\tilde{S}$.

5.2. Initial population

The proposed fuzzy knowledge-integration method uses a genetic algorithm for integration and optimization of fuzzy rule sets. The genetic algorithm requires a population of feasible solutions to be initialized and updated during the evolution process. In our approach, the initial population of fuzzy rule sets with their associated membership functions comes from multiple knowledge sources. Each individual in the initial population represents a fuzzy rule set with its associated membership functions. If the initial number of knowledge sources is small, some *dummy* initial rule sets that are randomly generated or duplicated from original rule sets, are inserted into the population to increase the population size.

5.3. Fitness and selection

In order to develop a "good" fuzzy knowledge base from the initial population, the genetic algorithm selects *parent* fuzzy rule sets with high fitness values for mating. An evaluation function and a set of test objects including instances or historical records, are then used to qualify the derived fuzzy rule set. Rule set performance is then fed back to the genetic algorithm to control how the solution space is searched to promote fuzzy rule set quality. Two important factors are used in evaluating derived fuzzy rule sets, the accuracy and the complexity of the resulting knowledge structure. Accuracy of a fuzzy rule set $\tilde{R}\tilde{S}$ is evaluated using test objects as follows:

$$\text{Accuracy}(\tilde{R}\tilde{S}) = \frac{\text{total number of objects correctly matched by } \tilde{R}\tilde{S}}{\text{total number of objects}}.$$

The more data used, the more objective and accurate the evaluation is. The complexity of the resulting rule set ($\tilde{R}\tilde{S}$) is the ratio of rule increase, defined as follows:

$$\text{Complexity}(\tilde{R}\tilde{S}) = \frac{\text{Number of rules in the integrated rule set } \tilde{R}\tilde{S}}{[\sum_{i=1}^{m} (\text{Number of rules in the initial } \tilde{R}\tilde{S}_i)]/m},$$

where $\tilde{R}\tilde{S}_i$ is the ith initial fuzzy rule set, and m is the number of initial rule sets. Accuracy and complexity are combined to represent the fitness value of the rule set. The evaluation function is then defined as follows:

$$\text{fitness}(\tilde{R}\tilde{S}) = \frac{[\text{Accuracy}(\tilde{R}\tilde{S})]}{[\text{Complexity}(\tilde{R}\tilde{S})]^{\sigma}},$$

where σ is a control parameter, representing a trade-off between accuracy and complexity.

5.4. Genetic operators

Genetic operators are very important to success of specific GA applications. Two genetic operators, *crossover* and *mutation*, are used in the genetic fuzzy knowledge-integration framework.

5.4.1. Crossover operator

The crossover operator used here selects crossover points differently from the way used in the simple genetic algorithm. The crossover operator in the simple genetic algorithm chooses the same points for both parent chromosomes, but, the crossover operator used here need not use the same point positions for both parent chromosomes. The crossover points may occur within rule strings or at rule boundaries. The only requirement for crossover points is that they must "match up semantically". That means, if one parent is cut at a rule boundary, then the other parent must also be cut at a rule boundary. Similarly, if one parent is cut at a point p units to left of a rule boundary, then the other parent must also be cut at a point p units to the left of some other rule boundary. An example of crossover operation is shown below.

Example 3. Assume that parent rule sets $\tilde{R}\tilde{S}_1$ and $\tilde{R}\tilde{S}_2$, contain, respectively, n and m rules for classifying test objects with two linguistic features (F_1 and F_2). Features F_1 and F_2 both have two possible linguistic values. Two classes are to be determined. Assume that $\tilde{R}\tilde{S}_1$ and $\tilde{R}\tilde{S}_2$ are encoded as shown in Fig. 8.

As mentioned above, the crossover points on both parents must "match up semantically". If crossover point cp_1 is the sixth unit to the left of $\tilde{r}_{1i}$ in $\tilde{R}\tilde{S}_1$ (denoted as $cp_1 = (1i, 6)$), then crossover point cp_2 for $\tilde{R}\tilde{S}_2$ must be the sixth unit to the left of a certain rule $\tilde{r}_{2j}$ (denoted as $cp_2 = (2j, 6)$). Thus, the crossover operator generates two offspring rule sets, $\tilde{O}_1$ and $\tilde{O}_2$, as shown in Fig. 9.

After offspring fuzzy rule sets have been generated using the crossover operation, the order of fuzzy membership functions may be destroyed, and may need rearrangement according to their center values. An example is given below to demonstrate the rearrangement of membership functions.

Example 4. Assume that two fuzzy rule sets, $\tilde{R}\tilde{S}_1$ and $\tilde{R}\tilde{S}_2$, for the Iris Flower domain, are encoded as shown in Fig. 10.

The crossover operator generates two offspring rule sets, $\tilde{O}_1$ and $\tilde{O}_2$. The pairs (2.6, 0.6)(2.5, -0.4)(3.8, -0.9) for *S.W.* in $\tilde{O}_1$ are out of sequence. They are then

$$\widetilde{RS}_1: \overbrace{\overbrace{c_{11}^{11} w_{11}^{11} c_{12}^{11} w_{12}^{11}}^{F1} \overbrace{c_{21}^{11} w_{21}^{11} c_{22}^{11} w_{22}^{11}}^{F2} \overbrace{\varphi_1^{11} \varphi_2^{11}}^{Class}}^{\tilde{r}'_{11}} \cdots \overbrace{\overbrace{c_{11}^{1i} w_{11}^{1i} c_{12}^{1i} w_{12}^{1i}}^{F1} \overbrace{c_{21}^{1i} w_{21}^{1i} c_{22}^{1i} w_{22}^{1i}}^{F2} \overbrace{\varphi_1^{1i} \varphi_2^{1i}}^{Class}}^{\tilde{r}'_{1i}} \cdots \overbrace{\overbrace{c_{11}^{1n} w_{11}^{1n} c_{12}^{1n} w_{12}^{1n}}^{F1} \overbrace{c_{21}^{1n} w_{21}^{1n} c_{22}^{1n} w_{22}^{1n}}^{F2} \overbrace{\varphi_1^{1n} \varphi_2^{1n}}^{Class}}^{\tilde{r}'_{1n}}$$

$$\widetilde{RS}_2: \overbrace{\overbrace{c_{11}^{21} w_{11}^{21} c_{12}^{21} w_{12}^{21}}^{F1} \overbrace{c_{21}^{21} w_{21}^{21} c_{22}^{21} w_{22}^{21}}^{F2} \overbrace{\varphi_1^{21} \varphi_2^{21}}^{Class}}^{\tilde{r}'_{21}} \cdots \overbrace{\overbrace{c_{11}^{2j} w_{11}^{2j} c_{12}^{2j} w_{12}^{2j}}^{F1} \overbrace{c_{21}^{2j} w_{21}^{2j} c_{22}^{2j} w_{22}^{2j}}^{F2} \overbrace{\varphi_1^{2j} \varphi_2^{2j}}^{Class}}^{\tilde{r}'_{2j}} \cdots \overbrace{\overbrace{c_{11}^{2m} w_{11}^{2m} c_{12}^{2m} w_{12}^{2m}}^{F1} \overbrace{c_{21}^{2m} w_{21}^{2m} c_{22}^{2m} w_{22}^{2m}}^{F2} \overbrace{\varphi_1^{2m} \varphi_2^{2m}}^{Class}}^{\tilde{r}'_{2m}}$$

Fig. 8. String representation of $\widetilde{RS}_1$ and $\widetilde{RS}_2$.

$$\widetilde{RS}_1: \mathbf{c_{11}^{11} w_{11}^{11} c_{12}^{11} w_{12}^{11} c_{21}^{11} w_{21}^{11} c_{22}^{11} w_{22}^{11}} \varphi_1^{11} \varphi_2^{11} \cdots \mathbf{c_{11}^{1i} w_{11}^{1i} c_{12}^{1i} w_{12}^{1i}} \underset{cp_1}{\vee} \underbrace{\mathbf{c_{21}^{1i} w_{21}^{1i} c_{22}^{1i} w_{22}^{1i}} \varphi_1^{1i} \varphi_2^{1i}}_{6\ units} \cdots \mathbf{c_{11}^{1n} w_{11}^{1n} c_{12}^{1n} w_{12}^{1n} c_{21}^{1n} w_{21}^{1n} c_{22}^{1n} w_{22}^{1n}} \varphi_1^{1n} \varphi_2^{1n}$$

$$\widetilde{RS}_2: c_{11}^{21} w_{11}^{21} c_{12}^{21} w_{12}^{21} c_{21}^{21} w_{21}^{21} c_{22}^{21} w_{22}^{21} \varphi_1^{21} \varphi_2^{21} \cdots c_{11}^{2j} w_{11}^{2j} c_{12}^{2j} w_{12}^{2j} \underset{cp_2}{\vee} \underbrace{c_{21}^{2j} w_{21}^{2j} c_{22}^{2j} w_{22}^{2j} \varphi_1^{2j} \varphi_2^{2j}}_{6\ units} \cdots c_{11}^{2m} w_{11}^{2m} c_{12}^{2m} w_{12}^{2m} c_{21}^{2m} w_{21}^{2m} c_{22}^{2m} w_{22}^{2m} \varphi_1^{2m} \varphi_2^{2m}$$

Crossover $\Big\downarrow$

$$\tilde{O}_1: \mathbf{c_{11}^{11} w_{11}^{11} c_{12}^{11} w_{12}^{11} c_{21}^{11} w_{21}^{11} c_{22}^{11} w_{22}^{11}} \varphi_1^{11} \varphi_2^{11} \cdots \mathbf{c_{11}^{1i} w_{11}^{1i} c_{12}^{1i} w_{12}^{1i}} c_{21}^{2j} w_{21}^{2j} c_{22}^{2j} w_{22}^{2j} \varphi_1^{2j} \varphi_2^{2j} \cdots c_{11}^{2m} w_{11}^{2m} c_{12}^{2m} w_{12}^{2m} c_{21}^{2m} w_{21}^{2m} c_{22}^{2m} w_{22}^{2m} \varphi_1^{2m} \varphi_2^{2m}$$

$$\tilde{O}_2: c_{11}^{21} w_{11}^{21} c_{12}^{21} w_{12}^{21} c_{21}^{21} w_{21}^{21} c_{22}^{21} w_{22}^{21} \varphi_1^{21} \varphi_2^{21} \cdots c_{11}^{2j} w_{11}^{2j} c_{12}^{2j} w_{12}^{2j} \mathbf{c_{21}^{1i} w_{21}^{1i} c_{22}^{1i} w_{22}^{1i}} \varphi_1^{1i} \varphi_2^{1i} \cdots \mathbf{c_{11}^{1n} w_{11}^{1n} c_{12}^{1n} w_{12}^{1n} c_{21}^{1n} w_{21}^{1n} c_{22}^{1n} w_{22}^{1n}} \varphi_1^{1n} \varphi_2^{1n}$$

Fig. 9. An example of the crossover operation.

$$\widetilde{RS}_1: \ldots \overbrace{5.2,0.9,6.1,-0.9,7.0,-0.9}^{S.L}\ \overbrace{2.6,0.6,3.2,-0.6,3.8,0.6}^{S.W}\ \overbrace{2.4,1.6,3.9,-1.4,5.4,-1.5}^{P.L}\ \overbrace{0.6,0.6,1.2,0.6,1.9,0.6}^{P.W}\ \overbrace{001}^{Class} \ldots.$$

$$\widetilde{RS}_2: \ldots \overbrace{5.2,0.8,6.1,-0.7,7.0,-0.8}^{S.L}\ \overbrace{2.0,0.3,2.5,-0.4,3.8,-0.9}^{S.W}\ \overbrace{2.4,1.6,4.0,1.4,5.4,1.5}^{P.L}\ \overbrace{0.6,0.7,1.2,0.6,1.9,0.6}^{P.W}\ \overbrace{010}^{Class} \ldots.$$

$\Big\downarrow$ *crossover*

$$\tilde{O}_1: \ldots \overbrace{5.2,0.9,6.1,-0.9,7.0,-0.9}^{S.L}\ \overbrace{\underline{2.6,0.6,2.5,-0.4,3.8,-0.9}}^{S.W}\ \overbrace{2.4,1.6,4.0,1.4,5.4,1.5}^{P.L}\ \overbrace{0.6,0.7,1.2,0.6,1.9,0.6}^{P.W}\ \overbrace{010}^{Class} \ldots.$$

out of sequence

$$\tilde{O}_2: \ldots \overbrace{5.2,0.8,6.1,-0.7,7.0,-0.8}^{S.L}\ \overbrace{2.0,0.3,3.2,-0.6,3.8,0.6}^{S.W}\ \overbrace{2.4,1.6,3.9,-1.4,5.4,1.5}^{P.L}\ \overbrace{0.6,0.6,1.2,0.6,1.9,0.6}^{P.W}\ \overbrace{001}^{Class} \ldots.$$

$\Big\downarrow$ *rearrange*

$$\tilde{O}_1: \ldots \overbrace{5.2,0.9,6.1,-0.9,7.0,-0.9}^{S.L}\ \overbrace{2.5,-0.4,2.6,0.6,3.8,-0.9}^{S.W}\ \overbrace{2.4,1.6,4.0,1.4,5.4,1.5}^{P.L}\ \overbrace{0.6,0.7,1.2,0.6,1.9,0.6}^{P.W}\ \overbrace{010}^{Class} \ldots.$$

$$\tilde{O}_2: \ldots \overbrace{5.2,0.8,6.1,-0.7,7.0,-0.8}^{S.L}\ \overbrace{2.0,0.3,3.2,-0.6,3.8,0.6}^{S.W}\ \overbrace{2.4,1.6,3.9,-1.4,5.4,1.5}^{P.L}\ \overbrace{0.6,0.6,1.2,0.6,1.9,0.6}^{P.W}\ \overbrace{001}^{Class} \ldots.$$

Fig. 10. Rearrangement of membership functions for Example 4.

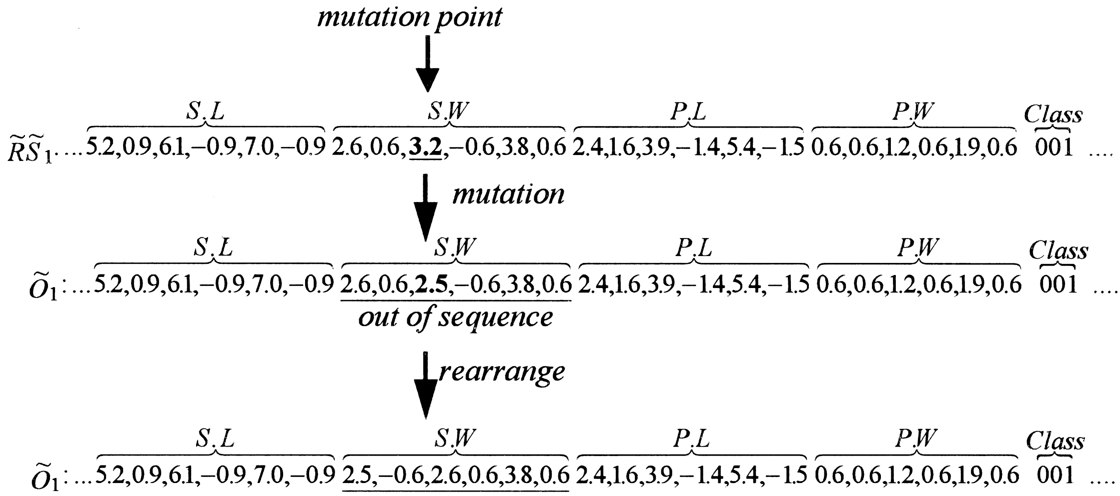
mutation point
S.L S.W P.L P.W Class
$\tilde{R}\tilde{S}_1$....5.2,0.9,6.1,−0.9,7.0,−0.9 2.6,0.6,3.2,−0.6,3.8,0.6 2.4,1.6,3.9,−1.4,5.4,−1.5 0.6,0.6,1.2,0.6,1.9,0.6 001 ....
mutation
S.L S.W P.L P.W Class
$\tilde{O}_1$:...5.2,0.9,6.1,−0.9,7.0,−0.9 2.6,0.6,2.5,−0.6,3.8,0.6 2.4,1.6,3.9,−1.4,5.4,−1.5 0.6,0.6,1.2,0.6,1.9,0.6 001 ....
out of sequence
rearrange
S.L S.W P.L P.W Class
$\tilde{O}_1$:...5.2,0.9,6.1,−0.9,7.0,−0.9 2.5,−0.6,2.6,0.6,3.8,0.6 2.4,1.6,3.9,−1.4,5.4,−1.5 0.6,0.6,1.2,0.6,1.9,0.6 001 ....

Fig. 11. The mutation process for Example 5.

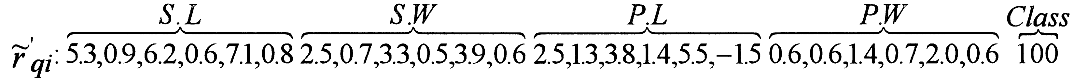
S.L S.W P.L P.W Class
$\tilde{r}'_{qi}$: 5.3,0.9,6.2,0.6,7.1,0.8 2.5,0.7,3.3,0.5,3.9,0.6 2.5,1.3,3.8,1.4,5.5,−1.5 0.6,0.6,1.4,0.7,2.0,0.6 100

Fig. 12. String representation of $\tilde{r}'_{qi}$.

rearranged in ascending order of membership function centers to (2.5, -0.4)(2.6, 0.6)(3.8,-0.9).

5.4.2. Mutation operator

The mutation operator is used to create a new fuzzy membership function by adding a random value ε to the center or the spread of an existing fuzzy membership function, say f. Assume that c and w represent the center and the spread of f. The center or the spread of the new 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 feature's fuzzy membership functions. These fuzzy membership functions then need rearrangement according to their center values. An example is given below to demonstrate the rearrangement of membership functions.

Example 5. Continuing from Example 1, assume a center of the second membership function for feature *S.W.* in $\tilde{R}\tilde{S}_1$ is chosen for mutation. Assume the random value ε is -0.7. The mutation process is shown in Fig. 11.

The mutation operator generates one offspring, rule set $\tilde{O}_1$, from $\tilde{R}\tilde{S}_1$, by adding the ε to the center of membership function (3.2, -0.6). The new membership function formed is (2.5, -0.6). The pairs (2.6, 0.6), (2.5, -0.4) and (3.8, -0.9) for *S.W.* in $\tilde{O}_1$ are then, however, out of sequence. They must be rearranged in ascending order of membership function centers; the final result is (2.5, -0.4), (2.6, 0.6), and (3.8, -0.9).

As mentioned above, the fuzzy knowledge integration phase chooses fuzzy rule sets with their associated membership functions for "mating", gradually creating good offspring. After the termination criteria are satisfied, the best offspring is decoded into the form of rules. If a fuzzy test in the final offspring is represented as $(c_{i1},w_{i1})(c_{i2},w_{i2})\cdots(c_{ij},w_{ij})\cdots(c_{i_{m_i}},w_{i_{m_i}})$ and each w is positive, the feature is irrelevant for the rule and is removed from the rule condition part. The rule can thus be succinctly interpreted. An example is given below to demonstrate the decoding of the final offspring.

Example 6. Continuing from Example 1, assume the fuzzy-rule string $\tilde{r}'_{qi}$ (shown in Fig. 12) is to be decoded into the form of rules.

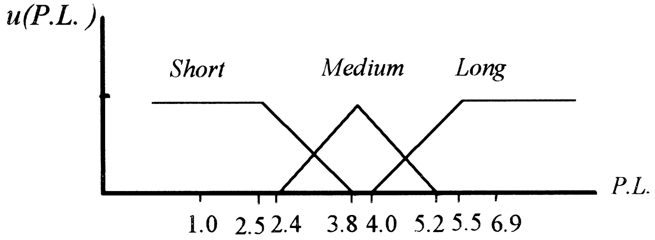

Fig. 13. Membership functions for *P.L.* in Example 6.

Since the tests for feature *S.L.* in $\tilde{r}'_{qi}$ are represented as "5.3, 0.9, 6.2, 0.6, 7.1, 0.8" and their w values are all positive, feature *S.L.* is thus irrelevant and is removed from the condition part of $\tilde{r}'_{qi}$. Similarly, the features *S.W.* and *P.W.* are also removed from the condition part of $\tilde{r}'_{qi}$. The tests in *P.L.* are represented as "2.5, 1.3, 3.8, 1.4, 5.5, −1.5". Since the last w value (-1.5) in the tests of *P.L.* is negative, the feature *P.L.* cannot be thought of an irrelevant one. The tests for *P.L.* are then decoded as "*P.L.* = *Short* or *Medium*" and their associated membership functions are decoded as shown in Fig. 13.

After decoding, the fuzzy-rule string $\tilde{r}'_{qi}$ is thus represented as below.

IF (*P.L.* = *Short* or *Medium*), then *Class* is *Setosa.*

6. Experimental results

The hepatitis diagnostic problem [4] was used as the problem domain to test the performance of the proposed fuzzy knowledge-integration approach. The 155 cases used in these experiments were obtained from Carnegie–Mellon University [4]. The goal of the experiments was to identify two possible classes, *Die* or *Live*. Table 1 shows an actual case expressed in term of 19 features and one class.

The 155 cases were first divided into two groups, a training set and a test set. The training set was used to evaluate the fitness of rule sets during the integration process; the test set acted as input events to test the derived rule set, and the percentage of correct predictions was recorded. In each run, 70% of the hepatitis cases were selected at random for training, and the remaining 30% of the cases were used for testing. Ten initial rule sets were obtained from different groups of experts or derived via machine-learning [24, 26, 27].

Table 1
A case for hepatitis diagnosis

Features	Feature values
Age	34
Sex	male
Steroid	high
Antivirals	high
Fatigue	high
Malaise	high
Anorexia	high
Liver BIG	yes
Liver Firm	yes
Spleen Palpable	yes
Spiders	yes
Ascites	high
Varices	high
Bilirubin	0.90
Alk Phosphate	95
SGOT	28
Albumin	4.0
Protime	75
Histology	no
Class: Live	

Table 2
The accuracy of the ten initial fuzzy rule sets

Rule Sets	Accuracy (%)	No. of rules
1	80.0	3
2	79.1	4
3	73.4	3
4	78.7	4
5	74.6	3
6	74.8	4
7	73.3	3
8	74.3	4
9	80.4	4
10	80.2	4

Table 3
Experimental results for the hepatitis diagnostic problem

Generation	CPU time (s)	Accuracy	Fitness values
0	1	76.88	0.7537
13	4	78.44	0.7690
69	19	81.32	0.7972
124	35	83.28	0.8164
181	52	84.32	0.8266
261	75	85.25	0.8357
414	115	86.88	0.8517
1401	427	88.76	0.5701
2110	560	89.49	0.8773
3110	822	89.65	0.8789
3550	935	89.77	0.8800
3817	1005	91.83	0.9002
4000	1056	92.90	0.9107

The accuracy of the ten initial rule sets was measured using the test instances. The results are shown in Table 2.

Although the ten initial fuzzy rule sets were not accurate enough, they could however act as a set of locally-optimal solutions that provide useful information in the search space. Beginning with these fuzzy rule sets, the genetic algorithm could then reach the (nearly) global optimal solution more rapidly than if it had nothing to refer to. Of course, each initial rule set could first have been improved by the same GA scheme before integration. We do not however favor this alternative since it cannot use information from the other rule sets, and it would take much time to get good results. Similarly, we could also have abandoned these initial rule sets and directly applied the genetic algorithm to acquire knowledge from training instances. But the same disadvantages would still have been present.

In the experiments, the operation frequency for *crossover* and *mutation* was set at 0.9 and 0.04, respectively. Table 3 shows the results for different generations as to accuracy, integration time, and fitness values.

Experimental results also show that executing the proposed approach in more generations yields more accurate results although the spent time increases. Fig. 14 shows the relationship between generations and fitness values of resulting rule sets for the proposed approach.

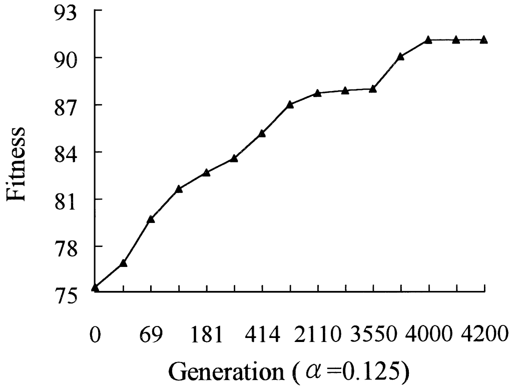

Fig. 14. Relationship between fitness values and generations.

Table 4
A comparison with other learning methods

Methods	Accuracy (%)
Our approach	92.9
Assistant-86 [5]	83
Diaconis and Efron's [8]	80

As the numbers of generations were increased, the resulting fitness values also increased, and finally converged to a specific value.

The accuracy of some other learning algorithms on the Hepatitis classification problem was examined in [4, 7]. The methods studied were Diaconis and Efron's statistic method [7] and that of Cestnik et al. Assistant-86 [4]. Table 4 compares the accuracy of our proposed approach with that of the other learning methods. It can easily be seen that the accuracy of our approach is higher than those of the other learning methods.

7. Conclusions

In this paper, we have shown how fuzzy knowledge-integration can be effectively processed using a genetic algorithm. Experimental results have also shown that our genetic fuzzy knowledge-integration framework is valuable for simultaneously combining multiple fuzzy rule sets and membership function sets. Our approach needs no human experts' intervention during the integration process. The time required by our approach is thus dependent on computer execution speed, but not on human experts. Much time can thus be saved since experts may be geographically dispersed, and their discussions are usually very time-consuming [11, 17]. Also, our approach is a scalable integration method, that can be applied as well when the number of rule sets to be integrated increases. Integrating a large number of rule sets may increase the validity of the resulting knowledge base. It is also objective since human experts are not involved in the integration process.

Although the work presented here shows good results, it is only a beginning. Some future investigations are proposed below.

1. Several issues in the field of knowledge verification remain unresolved. Effectively dealing with knowledge verification issues is another interesting topic.
2. Each derived rule supplied to the classifier system has a different degree of vagueness. Design of a new genetic fuzzy knowledge-integration framework that permits rules evolve over a global collection of membership functions is our current research project.

Acknowledgements

The authors would like to thank the anonymous referees for their very constructive comments.

References

[1] C. Baral, S. Kraus, J. Minker, Combining multiple knowledge bases, IEEE Trans. Knowledge and Data Eng. 3 (2) (1991) 208–220.

[2] J.H. Boose, Rapid acquisition and combination of knowledge from multiple experts in the same domain, Future Comput. Systems 1 (1986) 191–216.

[3] B. Carse, T.C. Fogarty, A. Munro, Evolving fuzzy rule based controllers using genetic algorithms, Fuzzy Sets and Systems 80 (1996) 273–293.

[4] G. Cestnik, I. Konenenko, I. Bratko, Assistant-86: a knowledge-elicitation tool for sophisticated users, in: Bratko, Lavrac (Eds.), Machine Learning, Sigma Press, 1987, pp. 31–45.

[5] M.G. Cooper, J.J. Vidal, Genetic design of fuzzy controllers: the cart and jointed pole problem, Proc. 3rd IEEE Internat. Conf. Fuzzy Systems IEEE, Piscataway, NJ, 1994, pp. 1332–1337.

[6] K.A. DeJong, W.M. Spears, D.F. Gordon, Using genetic algorithms for concept learning, Machine Learning 13 (1993) 161–188.

[7] P. Diaconis, B. Efron, Computer-intensive methods in statistics, Scientific American 248 (1983).

[8] R.A. Fisher, The use of multiple measurements in taxonomic problems, Annual Eugenics 7 (1936) 179–188.

[9] B.R. Gaines, M.L.G. Shaw, Eliciting knowledge and transferring it effectively to a knowledge-based system, IEEE Trans. Knowledge Data Eng. 5 (1) (1993) 4–14.

[10] B.J. Gragun, H.J. Studel, A decision-table based processor for checking completeness and consistency in rule-base expert-systems, Int. J. Man–Machine Studies 26 (1987) 633–648.

[11] D.M. Hamilton, Knowledge acquisition for multiple site, related domain expert systems: Delphi process and application, Expert Systems with Applications 11 (3) (1996) 377–389.

[12] J.H. Holland, J.S. Reitman, Cognitive systems based on adaptive algorithms, in: D.A. Waterman, F. Hayes-Roth (Eds.), Pattern-directed Inference Systems, Academic Press, New York, 1978.

[13] G.J. Hwang, Knowledge elicitation and integration from multiple experts, J. Inform. Sci. Eng. 10 (1) (1994) 99–109.

[14] C. Karr, Design of an adaptive fuzzy logic controller using a genetic algorithm, Proc. 4th Int. Conf. Genetic Algorithms, 1991, pp. 450–457.

[15] G.A. Kelly, The psychology of personal constructs, Norton, New York, 1995.

[16] J. Kinzel, F. Klawonn, R. Kruse, Modifications of genetic algorithms for designing and optimising fuzzy controllers, Proc. 1st IEEE Int. Conf. Evolutionary Computation, IEEE, Piscataway, NJ, 1994, pp. 28–33.

[17] A.J. LaSalle, L.R. Medsker, Computerized conferencing for knowledge acquisition from multiple experts, Expert Systems With Applications 3 (1991) 517–522.

[18] M. Lee, H. Takagi, Integrating design stages of fuzzy systems using genetic algorithms, Proc. 2nd IEEE Int. Conf. Fuzzy Systems, IEEE, San Francisco, 1993, pp. 612–617.

[19] O.K. Ngwenyama, N. Bryson, A formal method for analyzing and integrating the rule sets of multiple experts, Inform. Systems 17 (1) (1992) 1–16.

[20] A. Parodi, P. Bonelli, A new approach of fuzzy classifier systems, Proc. 5th Int. Conf. Genetic Algorithms, Morgan Kaufmann, Los Altos, CA, 1993, pp. 223–230.

[21] M.L. Shaw, B.R. Gaines, KITTEN: Knowledge initiation and transfer tools for experts and novices, Int. J. Man–Machine Studies 27 (1987) 251–280.

[22] S.F. Smith, A learning system based on genetic adaptive algorithms, Ph.D. Thesis, University of Pittsburgh, 1980.

[23] P. Thrift, Fuzzy logic synthesis with genetic algorithms, Proc. 4th Int. Conf. Genetic Algorithms, Morgan Kaufmann, Los Altos, CA, 1991, pp. 509–513.

[24] C.J. Tsai, C.H. Wang, T.P. Hong, S.S. Tseng, An inductive learning strategy with fuzzy sets, Proc. Int. Conf. Artificial Intelligence, 1996, pp. 63–69.

[25] C.H. Wang, T.P. Hong, S.S. Tseng, Self-integrating knowledge-based brain tumor diagnostic system, Expert Systems With Applications 11 (3) (1996) 351–360.

[26] C.H. Wang, T.P. Hong, S.S. Tseng, Inductive learning fuzzy examples, Proc. 5th IEEE Int. Conf. Fuzzy Systems, IEEE, San Francisco, 1996, pp. 13–18.

[27] C.H. Wang, J.F. Liu, T.P. Hong, S.S. Tseng, A fuzzy inductive learning strategy for modular rules, Fuzzy Sets and Systems, accepted.

[28] C.H. Wang, T.P. Hong, S.S. Tseng, C.M. Liao, Automatically integrating multiple rule sets in a distributed-knowledge environment, IEEE Trans. Systems Man Cybernet., accepted.

[29] C.H. Wang, T.P. Hong, S.S. Tseng, Knowledge integration by genetic algorithms, 7th Int. Fuzzy Systems Association World Congress, 1997, pp. 404–408.

[30] Y. Yuan, H. Zhuang, A genetic algorithm for generating fuzzy classification rules, Fuzzy Sets and Systems 84 (1996) 1–19.