International Journal of Information Technology and Knowledge Management
July-December 2010, Volume 2, No. 2, pp. 287-293

GENETIC OPTIMIZATION OF FUZZY RULE-BASE SYSTEM

Mukesh Kumar[1], Ajay Jangra[2] & Chander Diwaker[3]

A fuzzy rule-based system consists of fuzzy if-then rules such as "If x1 is small and x2 is small than y is large". The problem with existing fuzzy rule-based systems is that the size of the rule-base (number of rules) increases exponentially with the increase of the number of fuzzy sets involved in the rules. This exponential increase in size of the rule-base increases the search time and hence the problem solving time, and also the memory space required. In this paper a fuzzy rule-base compaction using genetic algorithm is proposed. The proposed approach consists of three phases: Knowledge acquisition phase, Encoding phase, and Compaction phase or Optimization phase. In knowledge acquisition phase information from various knowledge sources i.e. experts and machine learning methods is integrated into a single knowledge base. In encoding phase rule set and corresponding membership functions from different knowledge sources is encoded into a variable length string or chromosomes so that they can contribute to the genetic optimization approach. In optimization phase genetic algorithm that results in an optimal or nearly optimal set of fuzzy rules and membership functions from the initial set of rules and membership function is proposed.

Keywords: Fuzzy Rules, Genetic Algorithm, Optimal Rule-base, Rule-based Systems.

1. Introduction

Fuzzy logic improves classification and decision support of rule based systems by allowing the use of overlapping class definitions and interpretability of the used results by providing more insight the systems and decision making process. The automatic determination of fuzzy rules from data has been approached by several different techniques: neuro-fuzzy methods, fuzzy clustering in combination with GA. It is an open question as to what is the best way to extract rules from trained rule base in domains involving classification. The complete rule base for an expert system can be seen as an collection of different classification problem rules. The previous approaches were based on the exhaustive analysis of rule base have already been demonstrated to be intractable in that scale up factor increases exponentially with the increase in number of attributes.

The genetic algorithm (GA) is a combinatorial optimizer that is domain-independent; it is applicable to all functions that can be evaluated. Whereas hill-climbing and its relatives require domain-specific information (e.g., partial derivatives) to guide their searches, the genetic algorithm requires only two things: (1) a means of representing possible solutions and (2) an objective function evaluator—a function which maps a value from the domain of possible solutions to a scalar value. In the simplest terms, the genetic algorithm starts with an initial population of individuals each representing a point in the search space of a given function. Using an individual's objective function as a measure of how "fit" that individual is within its environment, the genetic algorithm simulates nature's survival of the fittest, essentially forcing the evolution of an optimal creature. This optimal creature is then the solution to the corresponding optimization problem.

2. Fuzzy Knowledge Acquisition Process

A fuzzy knowledge acquisition framework [1], shown in Fig 1, that accepts information from various fuzzy knowledge sources and converts them into a single knowledge base. Fuzzy rule set membership function 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 tools or delivered using machine learning methods may easily be translated into or represented by rules.

3. Rule Encoding Process

Since all the fuzzy rule set with their membership functions are obtained from different sources, they may differ in size, so a variable size Rule Representation [1] corresponding to a particular rule set with its membership function is used. Before encoding each fuzzy rule set is translated into a intermediary representation to preserve its semantic and syntactic constraints. This translation process involves the following steps:

[1]Computer Science and Engineering, UIET Department, Panjab University, Chandigarh, INDIA

[2, 3]Computer Science and Engineering, UIET Department, Kurukshetra University, Kurukshetra, INDIA

Email: mukesh_rai9@yahoo.com, er_jangra@yahoo.co.in, chander_d@rediffmail.com

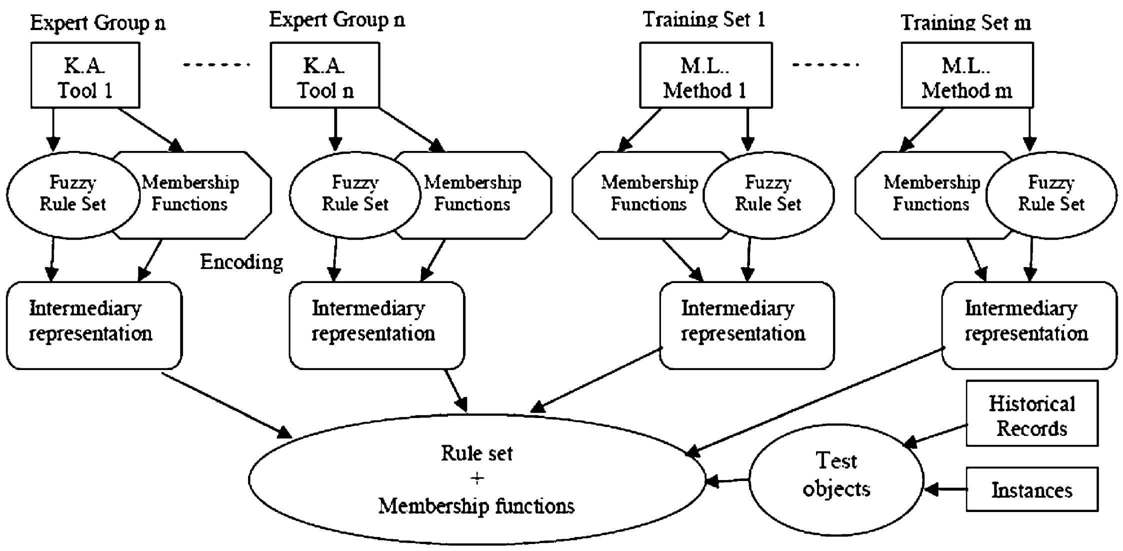

Fig. 1: Knowledge Acquisition Process

3.1. 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.

3.2. Collect classes occurring in the conclusion parts of the fuzzy rule sets. All classes gathered together comprise global class set.

3.3. Translate each fuzzy rule into an intermediary representation that retains its essential syntax and semantics. If some features in the 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 thus then composed of N feature tests and one class pattern. Where N is number of global features collected.

3.4. Concatenate all intermediary representations of rules to form the representation of rule set.

This intermediary representation can be well understood with the example explained below:

There are three species of iris flowers to be distinguished: Setosa, Versicolor, and Virginica. A class domain D_{flower} = { Setosa, Versicolor, Virginica}.Each rule is described by four features: Sepal Length(S.L.), Sepal Width(S.W.), Petal Length(P.L.), Petal Width(P.W.). Each feature has a domain given below:

$D_{S.L.}$ = {Short, Medium, Low}

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

$D_{P.L.}$ = {Short, Medium, Low}

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

Assume a fuzzy rule set RS_q obtained from a fuzzy knowledge source has the following four rules:

R_{q1}: IF (P.L. =Short) then Iris is Setosa;

R_{q2}: IF (P.L. =Long) then Iris is Virginica;

R_{q3}: IF (P.W. =Medium) then Iris is Versicolor;

R_{q4}: IF (P.W. =Wide) then Iris is Virgiinca;

After translation, the intermediary representation of the rules set would be constructed as follows:

R'_{q1}: IF (S.L. = Short or Medium or Long) and IF (S.W. = Narrow or Medium or Wide) and IF (P.L. = Short) and IF (P.W. = Narrow or Medium or Wide) then Consq is Srtosa.

R'_{q2}: IF (S.L. = Short or Medium or Long) and IF (S.W. = Narrow or Medium or Wide) and IF (P.L. = Long) and IF (P.W. = Narrow or Medium or Wide) then Consq is Virginica

R'_{q3}: IF (S.L. = Short or Medium or Long) and IF (S.W. = Narrow or Medium or Wide) and IF (P.L. = Short or Medium or Long) and IF (P.W. = Medium) then Consq is Versicolor.

R'_{q4}: IF (S.L. = Short or Medium or Long) and IF (S.W. = Narrow or Medium or Wide) and IF (P.L. = Short or Medium or Long) and IF (P.W. = Wide) then Consq is Virginica.

The tests with underlines are Dummy tests. After translation each intermediary rule representation then consists of four features set and one consequent pattern. We then concatenate all intermediary rules to form an intermediary rule set RS'_q. After each rule set has been translated into intermediary representation, each representation and membership functions are to be encoded. We are using the encoding structure:

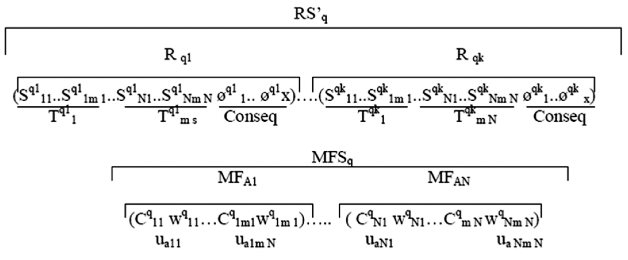
RS'q
R q1
R qk
$(S^{q1}_{11}..S^{q1}_{1m1}..S^{q1}_{N1}..S^{q1}_{NmN}\, \text{ø}^{q1}_{1}..\text{ø}^{q1}_{x})....(S^{qk}_{11}..S^{qk}_{1m1}..S^{qk}_{N1}..S^{qk}_{NmN}\, \text{ø}^{qk}_{1}..\text{ø}^{qk}_{x})$
T^{q1}_{1}
T^{q1}_{ms}
Conseq
T^{qk}_{1}
T^{qk}_{mN}
Conseq
MFSq
MF_{A1}
MF_{AN}
$(C^{q}_{11} W^{q}_{11}...C^{q}_{1m1}W^{q}_{1m1})....(C^{q}_{N1} W^{q}_{N1}...C^{q}_{mN}W^{q}_{NmN})$
u_{a11}
u_{a1mN}
u_{aN1}
u_{aNmN}

Fig. 2: Representation of a Fuzzy set RS'$_q$ with its Membership Functions

Each chromosome is divided into two parts: RS'q and MFS$_q$. RS'$_q$ denotes an intermediary form of fuzzy rule set and MFS$_q$ denotes the set of membership functions associated with RS'$_q$. In Fig 2, R$_{qi}$ represents the ith rule in RS'$_q$. Tqij is the jth test in rule R'$_{qi}$. a$_{ij}$ denotes the jth possible linguistic value of feature A$_i$. MF$_A$ denotes the set of membership functions for feature A$_i$, and uaij denotes the membership function of a$_{ij}$. Each feature test T^{qi}j is an intermediary fuzzy rule set is then encoded into a fixed length binary substring s^{qi}j1..........s^{qi}jm, where m$_j$ is the number of possible values for A$_i$. For example, assume the set of possible linguistic for feature A$_j$ is {a$_{j1}$, a$_{j2}$, a$_{j3}$}. Three bits are then used to represent this feature. The conclusion pattern in each rule is encoded as a bit string ($_1$, $_2$ … $_x$) where x is the number of possible conclusions. When the rule concludes to l, then F$_l$ is set as one and other as zero. N features tests and one conclusion pattern are then encoded and concatenated as a fixed length rule substring. All rules set substring are then concatenated to represent a variable length rule set string since each rule set differs in size.

For the above said example, the fuzzy rule R'q$_1$ is encoded as:

	S.L.	S.W.	P.L.	P.W.	Conseq
R'$_{q1}$:	111	111	100	111	1001

Fig. 3: Bit-string representation of R'$_{q1}$

Since feature S.L. in R'$_{q1}$ has three disjunctive test values, Short, Medium, Long, the test for S.L. is encoded as "111". S.W. also has three disjunctive test values, Narrow, Medium, Wide, and thus is encoded as "111". Similarly, P.W. is encoded as "111". But P.L. has only one test value Short. It is thus encoded as "100". Since the Consequent pattern has three possible values {Setosa, Versicolor, Virginica}, so the consequent pattern for Setosa is "100".

All substrings of intermediary rules are then concatenated to represent the entire fuzzy rule set. The result is as shown in Fig 4.

S.L.	S.W.	P.L.	P.W.	Conseq	S.L.	S.W.	P.L.	P.W.	Conseq	S.L.	S.W.	P.L.	P.W.	Conseq	S.L.	S.W.	P.L.	P.W.	Conseq
111	111	100	111	100	111	111	001	111	001	111	111	111	010	010	111	111	111	001	001

Fig 4: Bit-string Representation of RS$_q$

We used two parameters to encode associated membership functions. Membership functions applied to a fuzzy rule set are then assumed to be isosceles-triangle functions as shown in Fig 5, where aij denotes the jth linguistic value of feature A$_i$. ua$_{ij}$ denotes the membership function of a$_{ij}$, c$_{ij}$ represents the center of abscissa of membership function ua$_{ij}$ and w$_{ij}$ represents half the spread of membership function ua$_{ij}$.

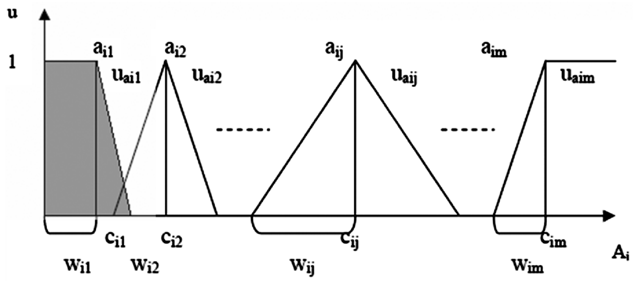

Fig 5: Membership Function for Feature A$_i$

Each membership function is represented as a pair (c, w). Thus all pairs of (c, w)'s for a certain feature are concatenated to represent its membership functions. MF$_{qi}$ is then represented as a substring of c$^q_{i1}$ w$^q_{i1}$,.......c$^q_{imi}$ w$^q_{imi}$ where mi is the number of possible linguistic values of A$_i$ (Fig 2). The entire set of membership functions MFS$_q$ is encoded by concatenated substrings of MF$_{A1}$, MF$_{A2}$, ...MF$_{AN}$. Since c and w are both numeric values, MFS$_q$ is thus encoded as a fixed-length real-number than a bit string. Let the membership functions for each feature in RS$_q$ are given in Fig 6.

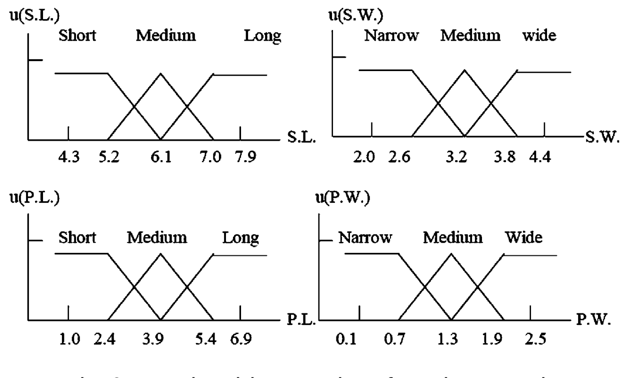

Fig. 6: Membership Functions for Iris Example.

According to the given encoding scheme MFS$_q$ is represented as shown in Fig 7.

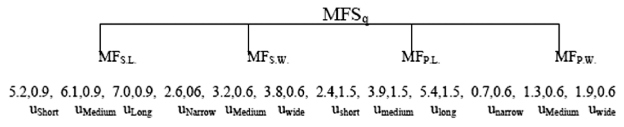

Fig. 7: String Representation of MFS$_q$.

Since feature S.L. has three linguistic values, Short, Medium, Long, membership functions for S.L. are encoded as (5.2, 0.9, 6.1, 0.9, 7.0, 0.9) according to Fig 6, S.W. also has three linguistic values Narrow, Medium, Wide, and its associated membership functions are thus encoded as (2.6, 0.6, 3.2, 0.6, 3.8, 0.6). Similarly membership function for P.L. and P.W. are respectively, encoded as (2.4, 1.5, 3.9, 1.5, 5.4, 1.5) and (0.7, 0.6, 1.3, 0.6, 1.9, 0.6).

The fuzzy rule set RS_q with associated membership function set MFS_q is then encoded as shown in Fig 8.

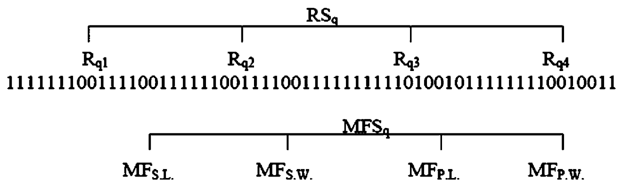

Fig. 8: String Representation of RS_q with its Associated MFS_q.

According to this representation each chromosome consists of an intermediary fuzzy rule set and its associated membership functions. This representation allows the application of genetic operators to the multiple rules set with their membership functions at the same time.

4. Rule Base Optimization

4.1. Proposed Genetic Algorithm

After encoding of the rules in variable length chromosomes, a genetic algorithm is applied to optimize the number of rules (chromosomes) within the rule base. The proposed approach is shown in Fig 9.The various steps involved in the approach are explained below.

4.2. Initial Population

The genetic algorithm requires a population of feasible solutions to be initialized and updated during evaluation process of the proposed approach, the initial population of fuzzy rules comes from multiple knowledge sources. Each individual in the initial population consists of certain rules and membership functions from different knowledge sources.

4.3. Fitness and Selection

To develop a compact rule base from an initial population, the genetic algorithm selects parent fuzzy rules with high Fitness values for mating. An evaluation function is a set of test objects including the instances and historical records, which is then used to qualify the derived rule set. The performance of the derived rule sets their Fitness value is fed back to the genetic algorithm to continue. Now, how the solution space is searched to promote the quality fuzzy rules. Two factors are used in evaluating the derived fuzzy rule, Accuracy and Complexity which are defined below:

Accuracy (R_i) = Total Number of objects currently matched Ri / Total number of objects

Complexity = Number of rules into current population / Total number of rules in the initial population.

Then using these two functions we can determine the Fitness function as

Fitness (R_i) = Accuracy (R_i) / [Complexity]

Because our goal is to increase Accuracy and to decrease the Complexity of rule base Here is a control parameter representing a trade-off between Accuracy and Complexity.

4.4. Genetic Operators

Genetic operators are very important to the success of a specific genetic algorithm. Two genetic Operators: Two-substring crossover and two-point mutation are used in the proposed approach.

4.4.1. Two Point Crossover

T h e two-substring crossover operator used, exchanges the substrings of the parents to generate offspring. It selects each crossover point with a probability of 1 / L-1, where L is the number of bits of the longest rule. The substring exchange may result in the change of both antecedent and consequent of the rule, so there are better chances of generation of better rules from the crossover operator. The three points chosen cp_{rs}, c_{pd} and cp_{mf} are chosen to accomplish this task, cprs is located in the rule-set part. And it may occur with in the rule string or at a rule boundary. The cprs crossover points need not be located at the same point positions for both parent chromosomes. The only requirement for cprs is that they must "match up syntactically" that means if one rule is cut at rule boundary the other parent must also be cut at a rule boundary. Similarly, if one parent is cut at a point p bits to left of a rule boundary, then the other parent must also be cut at a point p bits to left of some other rule boundary. cp_d is always chosen at the boundary between the rule set and membership function set. cp_{mf} is located in the membership function part and it may occur at any membership function center or at spreads. The cpmf crossover points for both parent chromosomes must however be same distances to the ends of both parent chromosomes. The crossover operator thus produces two offspring by exchanging the two substrings between cprs and cp_d and cpmf and the end points of both parents. This is explained in Fig 10.

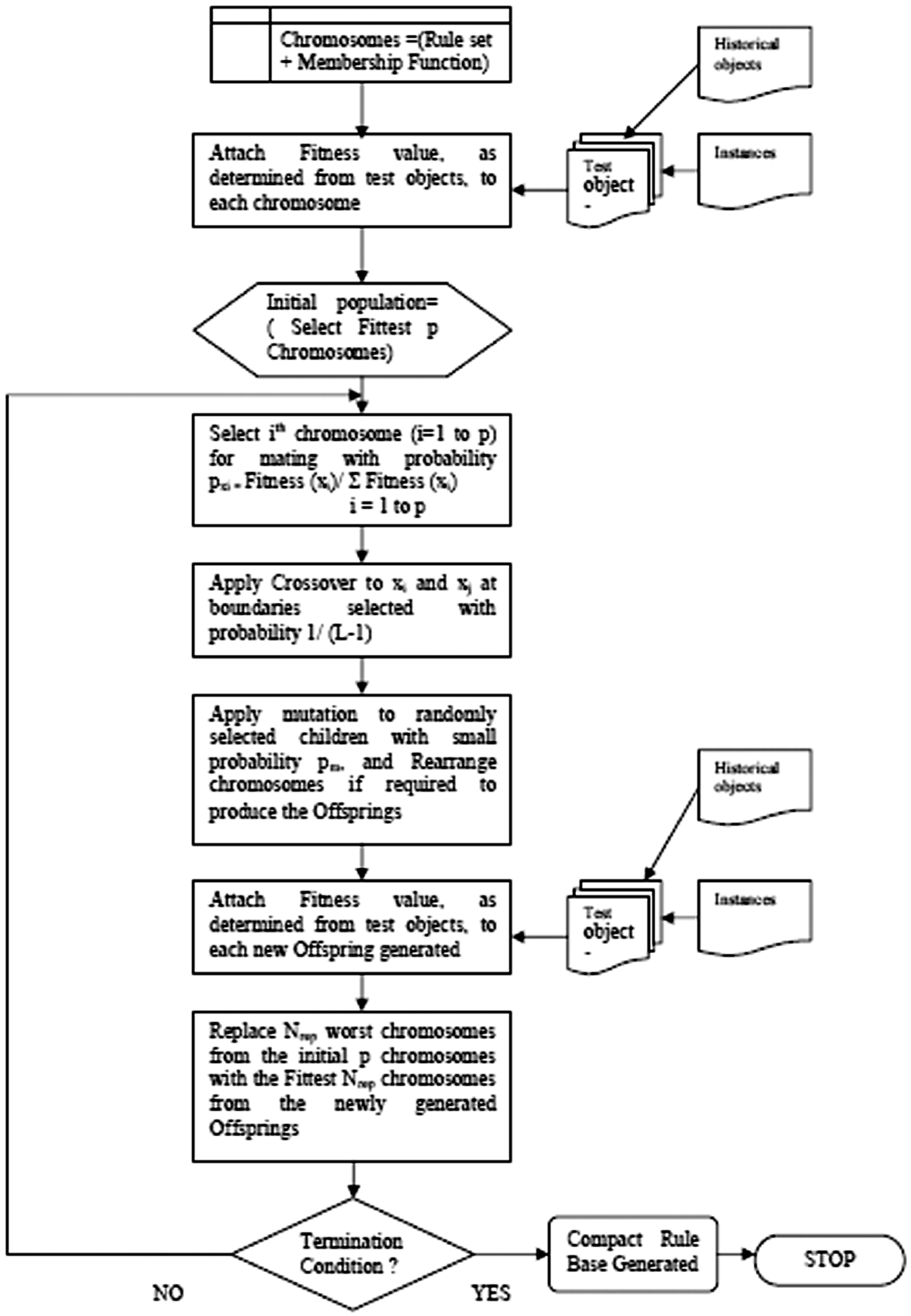
Chromosomes =(Rule set + Membership Function)
Historical objects
Test object -
Instances
Attach Fitness value, as determined from test objects, to each chromosome
Initial population= (Select Fittest p Chromosomes)
Select ith chromosome (i=1 to p) for mating with probability p_{xi} = Fitness (x_i)/ Σ Fitness (x_i) i = 1 to p
Apply Crossover to x_i and x_j at boundaries selected with probability 1/ (L-1)
Apply mutation to randomly selected children with small probability p_m, and Rearrange chromosomes if required to produce the Offsprings
Historical objects
Test object -
Instances
Attach Fitness value, as determined from test objects, to each new Offspring generated
Replace N_{rep} worst chromosomes from the initial p chromosomes with the Fittest N_{rep} chromosomes from the newly generated Offsprings
Termination Condition ?
NO
YES
Compact Rule Base Generated
STOP

Fig. 9: Proposed Genetic Rule Base Compaction Approach

We have assumed that two parent rule set RS_1 and RS_2 respectively, contain k and h rules for classifying objects with two linguistic features (A_1 and A_2). Features A_1 and A_2 both have two possible linguistic values. Two classes are to be determined. Also assume that RS_1 and RS_2 are encoded with their membership functions as shown in Fig 10(a).

```
              R11          R11      R1k     MFA1                  MFA2
RS1 + MFS1 : 101001…011001…110101 5.7 0.8 6.9 1.1, 1.6 0.7 1.8 0.9
              R21          R21      R2h     MFA1                  MFA2
RS2 + MFS2 : 111001…101010…010101 6.1 0.9 7.0 0.9, 1.3 0.5 1.5 0.6
```

Fig. 10(a): String Representation of RS_1 + MFS_1 and RS_2 + MFS_2

```
             R11          R11     R1k    MFA1                  MFA2
RS1 + MFS1: 101001…011 001…110101 5.7 0.8 6.9 1.1, 1.6 0.7 1.8 0.9
              cprs1        cpd1                   cpmf1 cpend1 (Six units from cpend1)
             R21          R21     R2h    MFA1                  MFA2
RS2 + MFS2: 111001…101 010…010101 6.1 0.9 7.0 0.9, 1.3 0.5 1.5 0.6
              cprs2        cpd2                   cpmf2 cpend2 (Six units from cpend2)
```

Fig. (10b-1): RS_1 + MFS_1 and RS_2 + MFS_2 before Crossover

O1: 101001……011 010….010101 5.7, 0.8, 7.0, 0.9, 1.3, 0.5, 1.5, 0.6
O2: 111001...101 001…110101 6.1, 0.9, 6.9, 1.1, 1.6, 0.7, 1.8, 0.9

Fig. (10b-2): Offsprings after Crossover of (RS_1 + MFS_1) and (RS_2 + MFS_2)

```
             R11    R11          R1k    MFA1            MFA2
RS1 + MFS1: 101001…011   001…110101 5.7 0.8, 6.9 1.1, 1.6   0.7 1.8 0.9
                         cprs1       cpd1                 cpmf1 cpend1 (3 units from cpend1)
             R21    R21          R2h    MFA1            MFA2
RS2 + MFS2: 111001…101 010…010101 6.1 0.9, 7.0 0.9, 1.3 0.5 1.5 0.6
                         cprs2       cpd2            cpmf2 cpend2 (3 units from cpend2)
```

Fig. (10b-3): RS_1 + MFS_1 and RS_2 + MFS_2 before Crossover (with different C_p, Cd and cp_{mf})

Crossover
O1: 101001……011 010....010101 5.7, 0.8, 6.9, 1.1, <u>1.6, 0.5, 1.5, 0.6</u>
Out of sequence
O2: 111001...101 001...110101 6.1, 0.9, 7.0, 0.9, 1.3 0.7, 1.8, 0.9

Fig. (10b-4): Offsprings after Crossover of (RS_1 + MFS_1) and (RS_2 + MFS_2)

Rearrange
O1: 101001……011 010….010101 5.7, 0.8, 6.9, 1.1, 1.5, 0.6, 1.6, 0.5

O2: 111001...101 001…110101 6.1, 0.9, 7.0, 0.9, 1.3 0.7, 1.8, 0.9

Fig. 10(b): An Example of Two-substring Crossover Operator

Assume that third bit to the left of r_{1j} in RS_1 as randomly chosen as the crossover point $cp_{rs}1$ The crossover point cp_{rs2} for RS_2 must then be the third bit to the left of certain rule r_{2j}, where j is randomly generated. Also assume a point six units to left of the chromosome is randomly chosen as a crossover point cpmf1. The crossover point cp_{mf2} for the other parent must be chosen six units to the left of MFS_2. cp_{d1} and cp_{d2} are chosen at the boundaries between the rule set and the membership function sets. Thus the substring from cp_{rs1} to cp_{d1} is exchanged with that substring from cp_{rs2} to cp_{d2}, and the substring from cp_{mf1} to cp_{end1} is exchanged with that from cp_{mf2} to cp_{end2}. The process is illustrated in Fig 10(10b).

After offspring fuzzy rule sets and their rule sets and their membership functions have been generated by two-substring crossover operation, the order of a newly generated fuzzy membership function and its neighbor may be destroyed. These fuzzy memberships thus need rearrangement according to their center values. An example is explained in Fig 11. Let after crossover the order of newly generated fuzzy membership function (1.6, 0.5) and its neighbor (1.5, 0.6) in the offspring O1 is destroyed. The crossover operator generates two offspring rule set O1 and O2. The pairs (1.6, 0.5) and (1.5, 0.6) for MFA_2 in O1 are out of sequence. They are then rearranged in ascending order of membership function centers to (1.5, 0.5) (1.6, 0.6).

4.4.2. Two-part Mutation

The two part mutation is used hare to create a new fuzzy rule and a new fuzzy membership function from one chromosome. The mutation operator in the ruleset part and membership function part are different. Two part mutation operator at rule set part randomly changes bit values from a selected rule set, to escape the local-optimum traps. Two-part mutation operator applied to the membership function part creates new fuzzy membership function say n. Assume that c and w represent the center and spread of membership function. The center or spread of newly generated membership function will be c + e or s + e by the mutation operator. Mutation at the center of a fuzzy membership function may disrupt the order of the resulting fuzzy membership functions. These fuzzy membership functions then need rearrangement according to their center values, as shown in Fig 11.

```
                R11   R11          R1k   MFA1          MFA2
RS1 + MFS1:   101001…011 001…110101 5.7, 0.8, 6.9, 1.1, 1.6, 0.7, 1.8, 0.9
                          mprs                        mpmf
                           Mutation
                    R11      R11     R1k   MFA1          MFA2
New RS1 + MFS1: 101001…011 101…110101 5.7, 0.8, 6.9, 1.1, 1.6, 0.7, 1.2, 0.9
                                                              Out of sequence
                           Rearrange
                    R11         R11      R1k      MFA1        MFA2
New RS1 + MFS1: 101001…011      101…110101  5.7, 0.8, 6.9, 1.1, 1.2, 0.7, 1.6, 0.9
```

Fig. 11: Two-part Mutation Operation

5. CONCLUSION

An optimization method for fuzzy rule base is proposed, that can be incorporated in fuzzy expert systems. Both representative data and expert knowledge are included in designing fuzzy rules, which conventionally include expert knowledge. This approach does not require any human intervention during the optimization phase. The time required is thus dependent on the 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 time consuming. The proposed approach is scalable to the increase in number of rule set. The increasing number of rule sets may increase the validity of the resulting knowledge base.

REFERENCES

[1] Ching-Hung Wang, Tzung-Pci Hong and Shian-Shyong Tseng. "Integrating Fuzzy Knowledge by Genetic Algorithms". IEEE Trans. on Evolutionary Computation. 2. No. 4. November 1998.

[2] H.Ishibuchi, K.Nozaki, and H. Tanaka. "Distributed Representation of Fuzzy Rules and its Application to Pattern Classification". Fuzzy Sets and Systems, 52, No. 1, pp. 21-32, November 1992.

[3] C.Z. Jainikow. "A Genetic Algorithm for Optimizing Fuzzy Decision Trees". In Proc. of 6th International Conf. on Genetic Algorithms, Pp. 421-428, July 15-19, 1995.

[4] T. BÄack, D. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, Oxford University Press, New York NY, 1997.

[5] E. Noda, A. A. Freitas, and H. S. Lopes. "Discovering Interesting Prediction Rules with a Genetic Algorithm". Proc. IEEE Confr. Evolutionary Comput. CEC '99, Pp. 1322–1329, July 1999.

[6] Michalewicz, Z., D. B. Fogel. How to Solve it: Modern Heuristics, Springer-Verlag. 2000.

[7] H. M., Gary G. Yen. "Multiobjective Optimization Design Via Genetic Algorithm", Proc. of the 2001 IEEE Int. Conf. on Control Applications: 1190-1195.

[8] Cordon, O., F. Herrera, et al. Recent Advances in Genetic Fuzzy Systems, Information Sciences, 136(2001): 1-5.

[9] J. Roubos, M. Setnes, J. Abonyi, Learning Fuzzy Classification Rules from Data, in:RASC Conference, Leichester, UK, 2000.

[10] A. Abraham, Evonf: A Framework for Optimization of Fuzzy Inference Systems using Neural Network Learning and Evolutionary Computation, in: The 17th IEEE International Symposium on Intelligent Control, ISIC'02, IEEE Press, ISBN 0780376218, pp.327-332, Canada, 2002.