IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 12, December 2015.
www.ijiset.com
ISSN 2348 – 7968

Interval Type-2 Fuzzy logic and GA Techniques: A Review

***Manoj Kumar Jha, **Ruchi Trivedi, ***Shilpa Sharma**

*Department of Applied Mathematics, Rungta Engg. College, Raipur, India (manojjha.2010@rediffmail.com)

**Ph.D. Scholar, Department of Mathematics, Dr. C.V. Raman University, Bilaspur, India
(trivedi.ruchi2201@gmail.com)

***Department of Applied Mathematics, Rungta Engg. College, Raipur, India
(shilpa.creative@gmail.com)

Abstract: Interval Type-2 Fuzzy Logic Controller (IT2FLC) where the fuzzy parameters, e.g. Fuzzy Membership Functions and Fuzzy Rule Bases are tuned by Genetic Algorithm (GAs) known as Genetic Interval Type-2 Fuzzy System (GIT2FS). The GFS are discussed briefly, later, the proposed design is explained.

Key words- GA, Interval Type-2 Fuzzy Logic Controller (IT2FLC), Genetic Interval Type2 Fuzzy System (GIT2FS).

1. Introduction

GA is main evolutionary algorithms for Interval Type-2 Fuzzy System (GIT2FS). GAs is general purpose search algorithms. Mainly there are three technologies, Neural Network (NN), Fuzzy Logic (FL), and Genetic Algorithms (GA) and their hybrid combinations. These thechnologies individually and in combination can be employed to solve problems. The combinations include neuro-fuzzy, GA-fuzzy, neuro-GA, and neuro-fuzzy-GA technologies. It is referred as artificial intelligence (AI), which is an area of computer science concerned with designing intelligent computer systems. It is like human behaviour.

In the field of soft computing Genetic Algorithms are unorthodox search and optimization algorithms, which mimic some of the processes of natural evolution. GA is evolutionary algorithms under guided random, which is search technique. Engineering, computational science, manufacturing, mathematics, physics, and other fields are application of Genetic Algorithms. GA is based on natural genetic. It provides healthy and strong search capabilities in complex spaces. GA offer a valid approach to effective search process and problem ability efficient. GA is very different from most of the traditional optimization methods. GA need design space to be converted into genetic space. GA works with a coding of variables. The advantage of working with a coding of variable space is that coding discretizes the search space even though the function may be continuous. GA uses a population of points at one time in contrast to the single point approach by traditional optimization methods. It processes a number of designs at the same time. Algorithm is started with a set of solution represented by chromosomes, called populations which develop slowly over time through a process of controlled variation and competition. Solution of one population is taken and used to form a new population. This is motivated by hope that new population will be better than the old one.

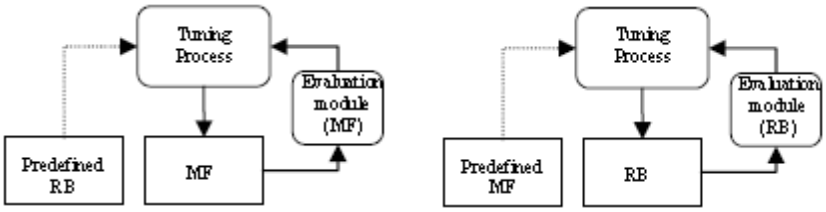
Solution which are selected to form new population (offspring), are selected according to their fitness and must be planed for each problem to be solved. The more suitable they are, the more chances they have to reproduce. This is repeated until some conditions for improvement of best solution are satisfied. The new population is further evaluated and tested for termination. If the termination criteria are not met, the population is iteratively operated by the three operators and evaluated until the termination criteria are met. One cycle of these operations and the subsequent evaluation procedure is known as a generation in GA terminology. Crossover is a recombination operator. In GA criteria are required to be expressed in terms of an objective function which is usually called to as a fitness function.

ISSN 2348 – 7968

2. Genetic Interval Type-2 Fuzzy System (GIT2FS)

GAs are general purpose search algorithms, based on natural genetics, that provide robust search capabilities in complex spaces, and thereby offer a valid approach to problem requiring efficient and effective search process. The basic idea is to maintain a population of chromosomes that evolves over time through a process of competition and controlled variation. A ***chromosome*** is representing candidate solutions to the concrete problem being solved. A GA starts with a ***population*** of randomly generated chromosomes, and advance towards better chromosomes by applying genetic operators modeled on the genetic process occurring in nature. The population undergoes evolution in a form of natural selection. During successive iterations, called ***generation***, chromosomes in the population are rated for their adaptation as solutions, and on the basic of these evaluation, a new population of chromosomes is formed using a selection mechanism and specific genetic operator such as ***crossover*** and ***mutation***. A ***fitness function*** must be devised for each problem to be solved. Given a particular chromosome, the fitness function returns a single numerical value, which is supposed to be proportional to the utility or adaptation of the solution represented by that chromosome.

As previously stated, GIT2FS is basically a fuzzy system augmented by a learning process based on a genetic algorithm (GA). In GIT2FS, GAs operates to search an appropriate Knowledge Base (KB) of a fuzzy system for a particular problem and to make sure those parameter values that are optimal with respect to the design criteria. The KB parameters constitute the optimization space, which is transformed into suitable genetic representation on which the search process operates. The KB is composed by interval type-2 membership functions (IT2MF), shortly (MF), and fuzzy rule base (RB), as mentioned before. So, there are some options to design Genetic IT2 Fuzzy System, e.g. tuning or learning membership functions, or fuzzy rule base or both of them, sequentially or concurrently. When tuning membership functions, an individual population represents parameters of the membership function shapes at which fuzzy rule base is predefined in advance. In contrast, if be desired to tune fuzzy rules base, the population represents all of fuzzy rules possibility that membership functions is assumed before. Fig.1 shows these conceptions.



(a) (b)

Fig.1 Some designs of Genetic Fuzzy Systems

Recently, there are some successful applications of GIT2FS to real world problems, e.g. control, robotics, manufacturing, consumer products, transportation, modeling and decision making. Further we will describe application a GIT2FS in behavior based mobile robot. Afterwards, a GIT2FS is then applied as a searching algorithm to tune the value of knowledge bases that described in next subsection. where *ki*, *ji* are adjustment coefficient, *Cx*, and *Wx* are set of centre and width of each fuzzy membership function. It means *ki* makes each center of membership function move to the right or left and *ji* makes them wider or sharper, as shown in Fig. 2. After that, the adjustment coefficients are encoded to form the population, as presented as below.

Gene $|1|,\ldots,|5|,|6|,|7|,..|11|,|12|,|13|,\ldots,|17|,|18|$

Chromosome $|subchrom_1|,|subchrom_2|,|sumchrom_3|$

Chromosome $|\ldots k_1 j_1 \ldots\ldots|,|\ldots k_2 j_2 \ldots\ldots|,|\ldots\ldots k_3 j_3 \ldots\ldots|$

Parameter $|\ldots.(MF_1)\ldots|,|\ldots(MF_2)|\ldots|,|\ldots(MF_3)\ldots\ldots|$

ISSN 2348 – 7968

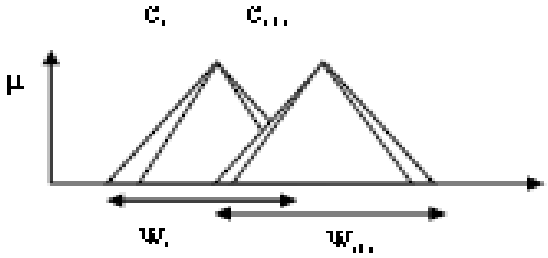


Fig. 2 Principle in tuning of membership function

In another side, for rule base searching, each of the parameter is encoded into integer codes that are based on number of output membership function. The coded parameters are arranged as show in the following equation to form chromosome of the population, as shown below.

Gene	|1|, |2|, … ,|26|,|27|, |28|,|29|, …, |53|, |54|
Chromosome	$\|sub\text{-}chromosome_1\|$, $\|sub\text{-}chromosome_2\|$
Parameter	$\|\ldots\ldots(RB_1)\ldots\ldots\|$, $\|\ldots\ldots(RB_2)\|\ldots\ldots\|$

GA's process starts with randomly generated initial populations. Then, all chromosomes are evaluated and associated base on fitness function with linear ranking method to determine the members of the new generation population. The fitness functions for faulty parts avoiding as:

$$f = \sum_{i=0}^{I}[\sum_{k=0}^{K}(100\omega^2(k) + 0.5/v(k) + 100c(k))$$

where I is the total number of start position, K is the number of step simulation for each start position, $\omega(k)$, and $v(k)$ are the rotational speed and the faulty speed at k, respectively, and c is constant for health check of IM, 0 if there is no fault and 1 if there is fault. This function is minimized in order to achieve the condition than the motor run by avoiding fault, higher speed, and mostly reliable speed. After that, three operators of GA are carried out, namely *recombination*, *crossover* and *mutation*, with fixed crossover probability rate (*Pc*) and probability mutation rate (*Pm*), that are 0.7, and 0.7/parameter numbers, respectively. The number of new generation is adjusted by Generation Gap constant (*GGAP*), which is 0.9. The procedure is repeated until the termination condition is reached. It has been presented Interval Type2Fuzzy Logic Controller (IT2FLC) where the fuzzy knowledge based, i.e. membership functions and rule bases, are tuned by Genetic Algorithm (GAs), known as Genetic Fuzzy System (GIT2FS), to generate individual command action. The model is designed in order to detect faults in IM. The best fitness knowledge base is obtained by learning the RB in advance and then tuning the MF after. Besides that, the motor has improved its performance, for instance it can generate motor control for individual fault.

3. Interval Type-2 Fuzzy Logic Controllers

The interval type-2 FLC uses interval type-2 fuzzy sets (such as those shown in Fig. 1(a) to represent the inputs and/or outputs of the FLC. In the interval type-2 fuzzy sets all the third dimension values equal to one. The use of interval type-2 FLC helps to simplify the computation (as opposed to the general type-2 FLC which is computationally intensive) which will enable the design of a robot FLC that operates in real time. The structure of an interval type-2 FLC is depicted in Fig. 2(b), it consists of a Fuzzifier, Inference Engine, Rule Base, Type-Reducer and a Defuzzifier.

ISSN 2348 – 7968

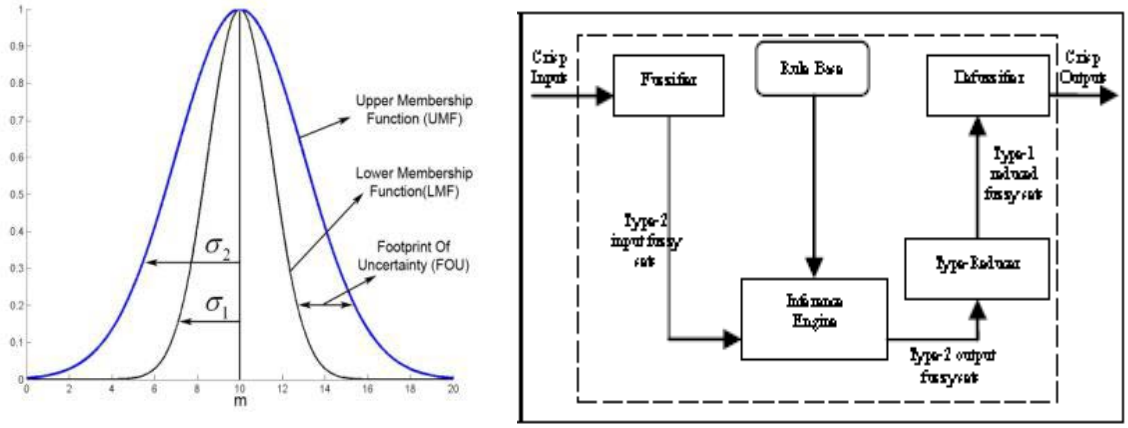


Fig. 2. (a) An interval type-2 fuzzy set. (b) Structure of the type-2 FLC.

The interval type-2 FLC works as follows: the crisp inputs from the input sensors are first fuzzified into input type-2 fuzzy sets; singleton fuzzification is usually used in interval type-2 FLC applications due to its simplicity and suitability for embedded processors and real time applications. The input type-2 fuzzy sets then activate the inference engine and the rule base to produce output type-2 fuzzy sets. The type-2 FLC rules will remain the same as in a type-1 FLC but the antecedents and/or the consequents will be represented by interval type-2 fuzzy sets. The inference engine combines the fired rules and gives a mapping from input type-2 fuzzy sets to output type-2 fuzzy sets. The type-2 fuzzy outputs of the inference engine are then processed by the type-reducer which combines the output sets and performs a centroid calculation which leads to type-1 fuzzy sets called the type-reduced sets. There are different types of type-reduction methods. In this paper we will be using the Center of Sets type-reduction as it has reasonable computational complexity that lies between the computationally expensive centroid type-reduction and the simple height and modified height type-reductions which have problems when only one rule fires . After the type-reduction process, the type-reduced sets are defuzzified (by taking the average of the type-reduced set) to obtain crisp outputs that are sent to the actuators.

4. The GA based Evolutionary System

The GA chromosome includes the interval type-2 MFs parameters for both the inputs and outputs of the induction motor (IM) interval type-2 FLC. We have used the interval type-2 fuzzy set which is described by a Gaussian primary MF with uncertain standard deviation as shown in Fig. 1(a). The GA based system uses real value encoding to encode each gene in the chromosome. Each GA population consists of 30 chromosomes. The GA uses an elitist selection strategy. The GA based system procedure can be summarized as follows:

Step 1: 30 chromosomes are generated randomly while taking into account the grammatical correctness of the chromosome. The “Chromosome Counter” is set to 1 – the first chromosome. The “Generation Counter” is set to 1 – the first generation.

Step 2: Type-2 FLC is constructed using the chromosome “Chromosome Counter” and is executed on the robot for 400 control steps to provide a fitness for the chromosome. During fitness evaluation, a healthy condition is in place to avoid the motor fault. After a controller has been executed for 400 control steps, a fixed controller takes over control.

Step 3: If “Chromosome Counter” < 30, increment “Chromosome Counter” by 1 and go to Step 2, otherwise proceed to Step 4.

Step 4: The best individual-so-far chromosome is preserved separately.

ISSN 2348 – 7968

Step 5: If "Generation Counter" = 1 then store current population, copy it to a new population *P* and proceed to Step 6. Else, select 30 best chromosomes from population "Generation Counter" and population "Generation Counter"-1 and create a new population *P*.

Step 6: Use roulette wheel selection on population *P* to populate the breeding pool.

Step 7: Crossover is applied to chromosomes in the breeding pool and "chromosome consistency" is checked. (*)

Step 8: "Generation Counter" is incremented. If "Generation Counter" < the number of maximum generations or if the desired performance is not achieved, reset "Chromosome Counter" to 1 and go to Step 2, else go to Step 9.

Step 9: Chromosome with best fitness is kept and solution has been achieved; END. (*)The crossover operator employed computes the arithmetic average between two genes. It is used with a probability of 100% to force the GA to explore the solution space in between the previously discovered, parental solutions. With chromosome consistency we refer to the correctness of the chromosome's genes in relation to their function in the IT2FLC.

Each input type-2 MF is represented by three parameters (one certain mean and two standard deviations). Thus, 12 genes are used to represent the type-2 FLC inputs (The FLC has 2 input sensors, each represented by two type-2 MFs and each MF is represented by 3 parameters). The type-2 FLC has 1 output governing the motor health condition of speed. Only the standard deviations of the output type-2 MFs are evolved and the means are fixed to guarantee that the FLC outputs are within the allowed domain of the outputs. Hence as shown in Fig. 3(a), the GA chromosome for this type-2 FLC comprises 12(inputs) + 4(outputs) = 16 genes.

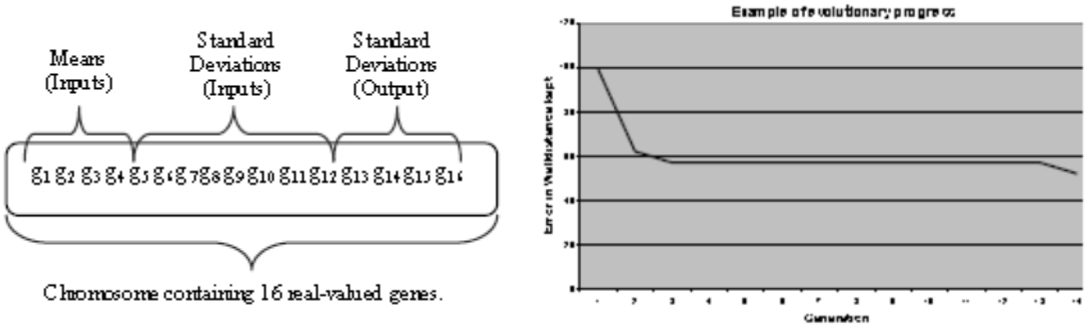


Figure 3. (a) Chromosome Structure, (b) Progress of best individual.

The fitness of each chromosome is generated by monitoring how the generated type-2 FLC has succeeded in following the edge at the desired distance over 400 control steps. An example of the evolutionary progress is shown in Fig. 3(b) which shows the performance of the best individual found so far against the number of generations. We have compared the performance of the evolved type-2 FLC against a manually designed type-2 FLC as well as a series of three type-1 FLCs with 4, 9 and 25 rules.

5. Tuning a modified Mamdani fuzzy rule base system (FRBS) with a genetic algorithm

GAs are useful for tuning both the database and the rule-base of FRBS (Maniadakis and Surmann 1999; Cordón et al. 2001). The subject of this work is the impact of a Interval Type2 Fuzzy System GA (IT2FS-GA) for tuning the database in a Mamdani FRBS on the outcomes of a GA which operates on the rule-base (RB-GA). The rest of this work first introduces a specific trip scheduling problem which has previously been analysed with a Mamdani FRBS. It then introduces both the standard Mamdani FRBS and the modifications made for this case including the structure of the rules and the database. It very briefly summarises the RB-GA since this is the test bed for the IT2FS - GA.

ISSN 2348 – 7968

6. MAMDANI FRBS

The Selection interface takes the fuzzy partition from the Inference system, and outputs a discrete value, as illustrated in Figure 4.

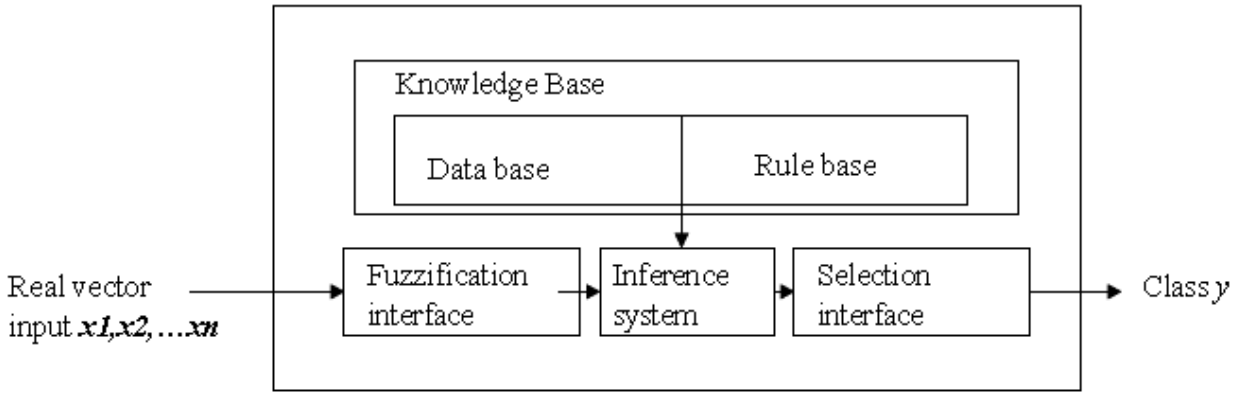


Fig.4 The modified Mamdani FRBS with Selection replacing de-fuzzification.

(After Cordón, Herrera et al. 2001)

7. Rules GA

This GA uses a population of 100 variable length chromosomes. The system is a Pittsburgh GA (Cordón et al. 2001; Cordon et al. 2004) with a population consisting of individuals encoding rule sets. The chromosome is translated into a rule set of a Mamdani FRBS, ***Fitness*** is the proportion of test cases correctly classified by a FRBS using that rule set with an additional weighting factor preferring shorter chromosome lengths.

Chromosome Coding the consequent conditions by position and thus storing only their adjectives, there are only (768) different possible antecedent rule sets each with a single consequent (one of 8). Each gene is a pair of integers; the antecedent part and the consequent part. Each individual chromosome consists of a variable length string of up to 100 genes. ***Crossover*** is asymmetric single point where a random crossover point is computed for each parent separately and two new strings are composed from the head of one and the tail of the other after duplicate antecedents are deleted. Thus the two offspring are generally of different lengths. A number of mutation operators are defined. No mutator is allowed to create a chromosome that has duplicate antecedent parts. ***Point Mutation*****:** A randomly selected locus in the antecedent part *or* the consequent of one gene is incremented or decremented so that it refers to one of the overlapping, neighboring fuzzy-sets. ***Delete*****:** Delete a sequence of up to a quarter of genes. ***Extend:*** extend the chromosome with a random gene. ***Inversion*****:** a strategy for regrouping genes so that crossover has a chance to test different building blocks. ***Breeding*** consists of selection of pairs from the population, and reproduction using crossover and mutation producing new pairs of offspring. ***Selection* is *elitist***, using a form of ranking selection. Individuals are first ranked by fitness and then working from fittest to least fit, individuals mate with selected members of the old population, offspring replacing least fit individuals. The majority of the populations are subjected to a bout of mutation using a random selection of mutation operators. One feature of ranking selection is that all objective functions which are monotonic on each other are equivalent in effect.

8. Fitness and objective function

The aim of the fitness function is to assess how well the partitioning of the fuzzy sets also partitions the data. There are four axes each with a number of partitions which thus forms a partitioned hyper-space with multiple “hyper-regions”. Intuitively one would like all of the classified situations with a particular class to fall within one hyper-region. In this case just eight hyper-regions would be filled, each with a single class and just eight rules would suffice to

ISSN 2348 – 7968

classify the data fully. For each class represented within a hyper-region, a *specificity* and a *selectivity* of that region for that class can be assigned. *Specificity* of a situation S in region R and having class C is the probability of S being of class C given having non-zero membership of the region R.. *Selectivity* of a situation S in region R and having class C is the probability of S having non-zero membership of region R given class C. S may have a non-zero degree of membership within more than one region, so to compute fitness we first compute the specificity of S multiplied by log of selectivity of S within each of the regions for which it has a degree of membership, take the maximum of absolute value of all these as the contribution of the situation to the fitness function and sum over all situations.

$$Specificity_S^R = \Pr(C_S \mid R_S)$$

$$Selectivity_S^R = \Pr(R_S \mid C_S)$$

$$F = \sum_{S=1}^{Number\,Situations} \max_R \left(\mathrm{abs}\left(Specificity_S^R \cdot \log\left(Selectivity_S^R \right) \right) \right)$$

9. Experiments and Results

The IT2FS-GA was run 13 times for 100 generations. The elite from each run was retained and the value of its associated objective function was recorded. This produced a spectrum of fuzzy sets, each with a different value. This constituted a test set of fuzzy-sets. Using each of the obtained fuzzy sets in turn, the RB-GA was then run against the full travel decisions data set. The RB-GA ran with single-best composition, and was halted at 10,000 generations. The process was repeated in triplicate to give 39 results, each yielding a collection of rules sets, each of these having a classification rate, a rule count, and an associated value for the objective function assigned to the interval type2 fuzzy-set by the IT2FS-GA. Pooling the three runs, and performing a linear regression of the interval type2 fuzzy-set fitness v classification rate and rule counts obtained, we find that there is a significant positive trend for the classification rates and no trend for the rule counts as seen in Figure 3.

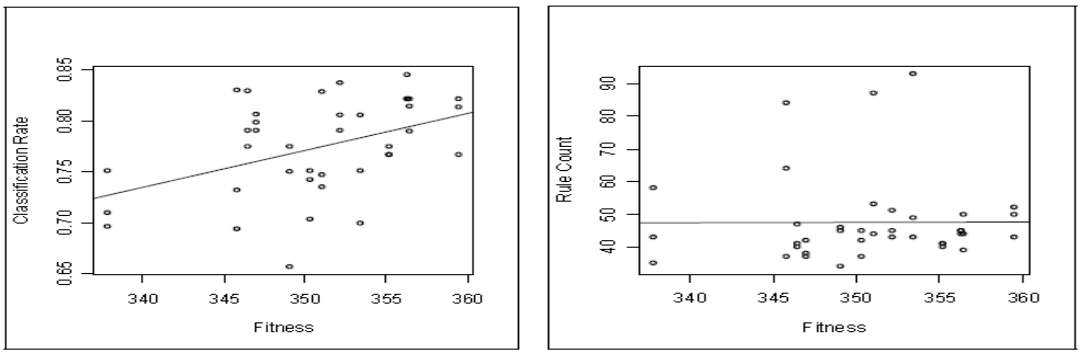


Fig.5 Regressions of Classification Rate and of Rule Count against the IT2FS-GA fitness achieved by each of thirteen independent runs of the IT2FS-GA.

The classification rate shows a significant trend, whereas the rule count does not. There is considerable variance. Furthermore fitness and rule counts are not independent. The RB-GA optimizes for high classification rates and only reduces rule counts within classification rate. As shown in Figure 4, there is a non-significant positive trend between IT2FS-GA objective function value and the smallest per-replicate size of the rule set obtained, indicating that the GA may have proceeded faster (since all runs of the RB-GA halted after the same number of generations). There is a more significant negative trend with the maximum per-replicate ratio of classification rate to rule count, indicating that less compact rule sets obtained running with the higher values of the IT2FS-GA fitness, are likely less generalized.

ISSN 2348 – 7968

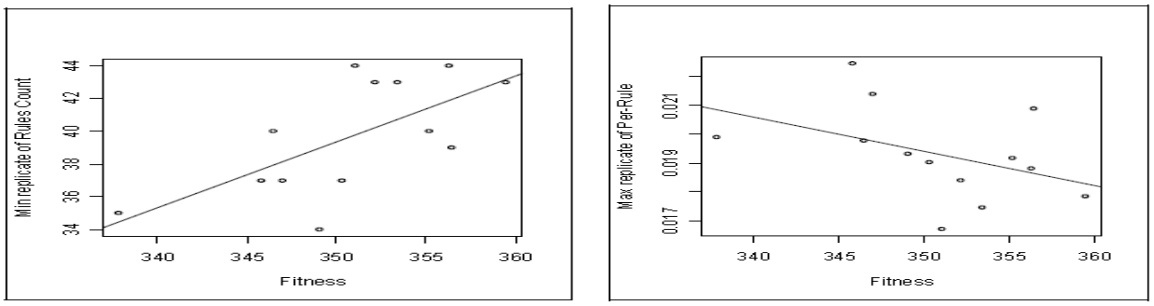


Fig.6 The second analysis tests fuzzy-set fitness against the minimum rule count, and maximum ratio of classification rate to rule count, for the replicates at each fitness level.

10 Simulation Results

A set of different trainings for the modular neural networks was performed to test the proposed type-2 fuzzy logic approach for response integration in modular neural networks. We show in Table 1 some of these trainings with different numbers of modules, layers and nodes. The training times are also shown in this table to illustrate the performance with different training algorithms and conditions.

Once the necessary trainings were done, a set of tests were performed with different type-2 fuzzy systems. The fuzzy systems were used as response integrators for the three modules of the modular network. In the type-2 fuzzy systems, different types of membership functions were considered with goal of comparing the results and deice on the best choice for the recognition problem.

The best type-2 fuzzy system, in the sense that it produced the best recognition results, was the one with triangular membership functions. This fuzzy system has 3 input variables and one output variable, with three membership functions per variable. We show in Figures 5 and 6 the membership functions of the type-2 fuzzy system. The recognition results of this type-2 fuzzy system for each training of the modular neural network are shown in Table 2. In Table 2 we show the results for 15 trainings of the modular neural network. In each row of this table we can appreciate the recognition rate with the type-2 fuzzy system. We can appreciate that in 8 out of 15 cases, a 100% recognition rate was achieved.

ISSN 2348 – 7968

Table1 Sample Trainings

Red	Funcion de Entrenamiento	Num. de Capas	Neuronas	F. R.	%de Reconocimiento	Error Meta	Error Alcanzado	Epocas	time
1	Trainscg	V: Mod1: 2	48,49	sse	100% (20/20)	0.001	0.000998658	3000	35 Min.
		Mod2: 2	50,60	sse	100% (20/20)	0.001	0.000998254	3000	
		Mod3: 2	60,70	sse	100% (20/20)	0.001	0.00099876	3000	
		R: Mod4: 1	350	mse	100% (20/20)	0.01	0.0099726	4000	
		Mod5: 1	400	mse	100% (20/20)	0.01	0.0099546	4000	
		Mod6: 1	420	mse	100% (20/20)	0.01	0.0098316	4000	
		H: Mod7: 1	350	mse	100% (20/20)	0.01	0.0080223	4000	
		Mod8: 1	250	mse	100% (20/20)	0.01	0.0086453	4000	
		Mod9: 1	300	mse	100% (20/20)	0.01	0.0067892	4000	
Red	**Funcion de Entrenamiento**	**Num. de Capas**	**# de Neuronas**	**F.R.**	**%de Reconocimiento**	**Error Meta**	**Error Alcanzado**	**Epocas**	**time**
10	Trainscg	V: Mod1: 2	80,90	sse	100% (20/20)	0.001	0.000999948	3000	1 Hra. 34 Min.
		Mod2: 2	90,90	sse	100% (20/20)	0.001	0.000992595	3000	
		Mod3: 2	80,90	sse	100% (20/20)	0.001	0.000997131	3000	
	Trainscg	R: Mod4: 1	20	mse	100% (20/20)	0.01	0.124015	4000	
		Mod5: 1	15	mse	100% (20/20)	0.01	0.0269689	4000	
		Mod6: 1	25	mse	100% (20/20)	0.01	0.01698	4000	
	Traingdx	H: Mod7: 1	350	mse	25% (5/20)	0.01	0.037501	4000	
		Mod8: 1	230	mse	5% (1/20)	0.01	0.0450007	4000	
		Mod9: 1	290	mse	5% (1/20)	0.01	0.0425011	4000	
Red	**Funcion de Entrenamiento**	**Num. de Capas**	**# de Neuronas**	**F.R.**	**% de Reconocimiento**	**Error Meta**	**Error Alcanzado**	**Epocas**	**time**
14	Trainscg	V: Mod1: 2	85,95	mse	100% (30/30)	0.001	0.000990206	3000	1 Hra. 23 Min.
		Mod2: 2	95,90	mse	100% (30/30)	0.001	0.000970259	3000	
		Mod3: 2	99,96	mse	93% (28/30)	0.001	0.000995249	3000	
	Trainscg	R: Mod4: 1	25	mse	0.3% (1/30)	0.01	0.0880936	4000	
		Mod5: 1	20	mse	0.3% (1/30)	0.01	0.0172602	4000	
		Mod6: 1	30	mse	0.3% (1/30)	0.01	0.0127059	4000	
	Traincsg	H: Mod7: 1	15	mse	96% (29/30)	0.01	0.148242	4000	
		Mod8: 1	10	mse	90% (27/30)	0.01	0.142394	4000	
		Mod9: 1	23	mse	96% (29/30)	0.01	0.127423	4000	

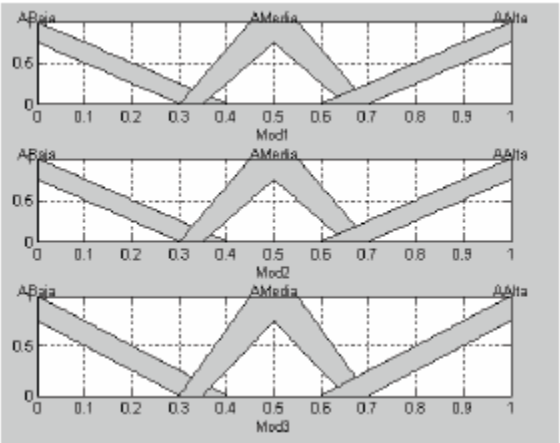


Fig.7 Input variables of the type-2 fuzzy system

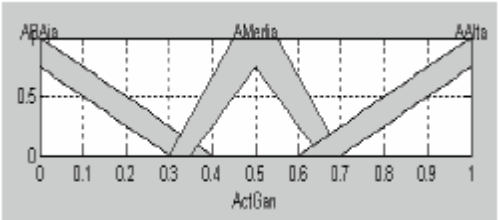


Fig.8 Output variables of the type-2 fuzzy system

ISSN 2348 – 7968

Table 2 Results of the Type-2 Fuzzy System with Triangular Membership Functions

Funciones de Membresia Triangulares(7)	
Entrenamiento	% de Reconocimiento
1	100% (20/20)
2	100% (20/20)
3	100% (20/20)
4	100% (20/20)
5	100% (20/20)
6	5% (1/20)
7	100% (20/20)
8	65% (13/20)
9	100% (20/20)
10	100% (20/20)
11	93% (28/30)
12	96% (29/30)
13	93% (28/30)
14	93% (28/30)
15	83% (25/30)
Resultados para cada uno de los entrenamientos , utilizando un Sistema Difuso con Funciones de MembresiaTriangulares	

The fuzzy systems with worst results for the modular neural network were the ones with Gaussian and Trapezoidal membership functions. We use 3 input variables and one output variable, as in the previous fuzzy system. We show in Figures 9 and 10 the Gaussian membership functions of this system.

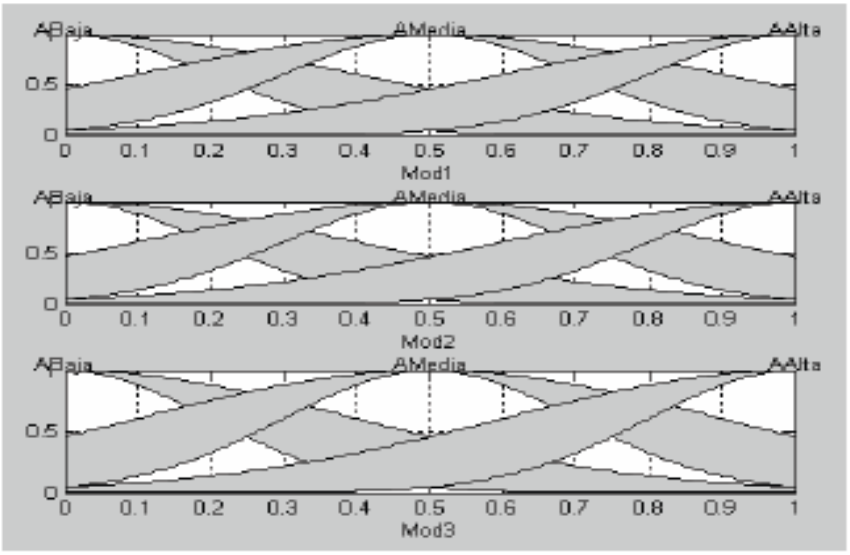
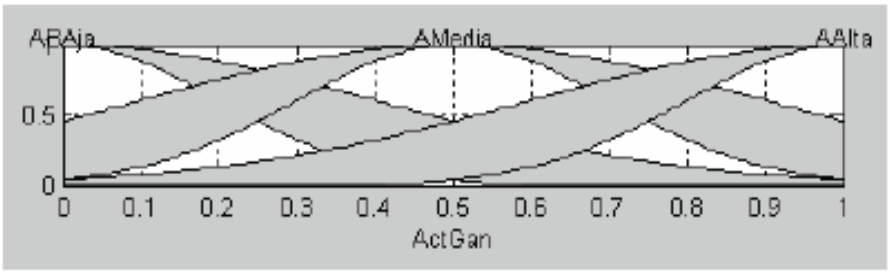


Fig.9 Input variables for type-2 fuzzy system with Gaussian membership functions



ISSN 2348 – 7968

Fig.10. Output variable for type-2 fuzzy system with Gaussian membership functions

The results that were obtained with Gaussian and Trapezoidal membership functions are similar.

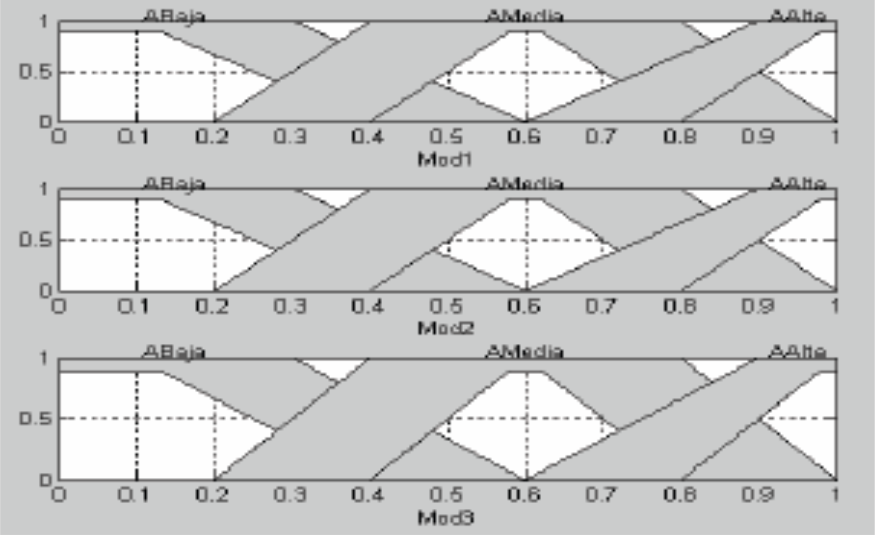


Fig.11 Output variable for type-2 fuzzy system with Trapezoidal functions

Table 3 Recognition rates with the Type-2 System and Trapezoidal Functions

Funciones de Membresia Trapezoidales (11)	
Entrenamiento	**% de Reconocimiento**
1	100% (20/20)
2	100% (20/20)
3	55% (11/20)
4	100% (20/20)
5	100% (20/20)
6	95% (19/20)
7	100% (20/20)
8	60% (12/20)
9	55% (11/20)
10	45% (9/20)
11	96% (29/30)
12	96% (29/30)
13	100% (30/30)
14	6% (2/30)
15	3% (1/30)
Resultados para cada uno de los entrenamientos, utilizando un Sistema Difuso con Funciones de Membresia Trapezoidal	

We show in Table 3 the recognition results obtained with the type-2 fuzzy system with Trapezoidal membership functions. We can appreciate from Table 3 that only in 6 out of the 15 cases a 100% recognition rate is obtained. Also, there are 4 cases with low recognition rates. We have to mention that results with a type-1 fuzzy integration of responses were performed in previous paper, in which the recognition rates were consistently lower by an average of 5%. We can state in conclusion that the type-2 fuzzy system for response integration is improving the recognition rate in the case of persons based on face, fingerprint and voice.

ISSN 2348 – 7968

11. Conclusions

In this work, we have presented the use of GA based architecture to directly evolve the type-2 MFs of interval type-2 FLCs used for fault diagnosis of induction motor. We have shown that the genetically evolved type-2 FLCs can lead to superior performance in comparison to type-1 FLCs and the manually designed type-2 FLCs. The results indicate that the genetic evolution of type-2 MFs can provide valid and high-performance type-2 FLCs without relying on any a priori knowledge such as logged data or a previously existing model, making it suitable for diagnosis problems where no such a priori data is available such as in the IM fault domain. This work will be a step towards overcoming the problem of manually specifying pseudo-optimal MFs for type-2 FLCs, which to date is one of the main obstacles when designing type-2 FLCs. The FRBS tuning problem has a high dimension error space with dimensions relating to the following partial order of choices; (a) the selection of axes, (b) the number and form of fuzzy-sets on the axes, these points relating to the database; (c) the grammar of the rules, and (d) the selection of the rules in the rule base. This work shows some benefits to pre-tuning using a simple heuristic. It has been well established that GAs in general executes a more robust search than pure hill-climbing. The decoding scheme in the IT2FS-GA limits the shapes of the fuzzy sets and again simplifies the error landscape. The primary experiment with the RB-GA shows several important things. The GA was halted after a set number of runs, and there is a high level of variance of all performance measures. The performance measures of the IT2FS-GA, which are the same measures, thus are subject to high variance.

Reference

1. Tsaur R.C., (2005) “Fuzzy Grey GM (1,1) Model Under Fuzzy Systems”, *International Journal of Computer Mathematics*, 2, pp.141-149.
2. Zadeh L., Fuzzy Sets, *Information and Controls*, 8, 1965, pp. 338--353.(24)
3. Deng L.J., (1989) “Grey Prediction and Decision (in Chinese)”, Huazong Institute Of Technology *Press*, Wuhan, China. (25)
4. Deng L.J., (1989) “Introduction to Grey System Theory”, The Journal of Grey Systems, Vol. 1, pp.1-24,. (26)
5. Sterman J.D., *Management Science*, 35, 1989, pp.321-399.(7)
6. Tozan H., Vayvay O., The Effects of Fuzzy Forecasting Models on Supply Chain Performance, *Proceedings of 9th WSEAS International Conference on Fuzzy Systems (FS’08), WSEAS*, 2008, pp. 107--112. (4)
7. Sterman J.D, Business Dynamics: System thinking and modelling for a complex world, *McGraw-Hill*, New York, USA (2000). (44)
8. Buckley JJ., Hayashi Y., Fuzzy neural networks, In: Fuzzy sets, neural networks and soft computing by Yager L., Zadeh L., V. *Nostrand Rainhol* , NY 1994.(41)
9. Brown M., Harris C., Neuro-fuzzy adaptive modeling and control, Prentice Hall, NY, 1994. (40)
10. Seung-Kuk Paik, Prabir K. Bagchi, (2007) "Understanding the causes of the bullwhip effect in a supply chain", International Journal of Retail & Distribution Management, Vol. 35 Iss: 4, pp.308 – 324.
11. Hakan Tozan, Ozap Vayvay (2011), A Hybrid Grey & ANFIS Approch to Bullwhip Effect in Supply Chain Networks, WSEAS transactions on stystem, issue 4, Volume 8, April 2009.
12. Karnik, N.N., Mendel, J.M. (1998), Type-2 fuzzy logic systems: Type-reduction, in IEEE Syst.,Man, Cybern. Conf., San Diego, CA, pp 2046– 2051, vol.2, 11-14 Oct 1998.
13. Qilian Liang, Mendel, J.M. (2000), Interval type-2 fuzzy logic systems: theory and design, Fuzzy Systems, IEEE Transactions, Volume:8,Issue: 5, pp. 535 - 550, Oct 2000.
14. Guldemir.H., “Detection of airgap eccentricity using line current spectrum of induction motors”, Electric Power Systems Research 64 (2003), 109–117.

ISSN 2348 – 7968

15. Zouzou S.E., Laala.W, Guedidi.S and Sahraoui.M, "A Fuzzy Logic Approach for Diagnosis of Rotor Faults in squirrel Cage Induction Motors," in Proc. 2000 E Computer and electrical Engineering Conf., pp. 173-177.
16. Liu.Y., Guo.L., Wang.Q., Fu.D., “A system of detection, analysis and diagnosis on electro-mechanic systems”, in: Proceedings of the Eighth International Conference on Electronic Measurement and Instruments, vol. 1, Xi’an, China, August 2007, pp. 772–775.
17. Guldemir.H., “Detection of airgap eccentricity using line current spectrum of induction motors”, Electric Power Systems Research 64 (2003) 109–117.
18. Anant G. Kulkarni, Dr. M. F. Qureshi, Dr. Manoj Kumar Jha, Genetically Tuned Interval Type-2 Fuzzy Logic for Fault Diagnosis of Induction Motor International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753, Vol. 3, Issue 7, July 2014.
