Hybrid Intelligent Systems: Evolving Intelligence in Hierarchical Layers

A. Abraham

Natural Computation Lab
Department of Computer Science
Oklahoma State University, USA
ajith.abraham@ieee.org
ajith.softcomputing.net

Hybridization of intelligent systems is a promising research field of modern computational intelligence concerned with the development of the next generation of intelligent systems. A fundamental stimulus to the investigations of hybrid intelligent systems is the awareness in the academic communities that combined approaches might be necessary if the remaining tough problems in artificial intelligence are to be solved. The integration of different learning and adaptation techniques to overcome individual limitations and to achieve synergetic effects through the hybridization or fusion of these techniques has, in recent years, contributed to a large number of new intelligent system designs. Most of these hybridization approaches, however, follow an ad hoc design methodology, justified by success in certain application domains.

This chapter introduces the designing aspects and perspectives of two different hybrid intelligent system architectures involving fuzzy inference system, fuzzy clustering algorithm, neural network learning and evolutionary algorithm. The first architecture introduces a Takagi-Sugeno fuzzy inference system, which is optimized using a combination of neural network learning and evolutionary algorithm. In the second architecture a fuzzy clustering algorithm is used to segregate the data and then a fuzzy inference system is used for function approximation. Both architectures are validated using real world examples. Some conclusions are also provided towards the end.

8.1 Introduction

In recent years, several adaptive hybrid soft computing [27] frameworks have been developed for model expertise, decision support, image and video segmentation techniques, process control, mechatronics, robotics and complicated automation tasks. Many of these approaches use a combination of different knowledge representation schemes, decision making models and learning strategies to solve a computational task. This integration aims at overcoming the limitations of individual techniques through hybridization or the fusion of various techniques. These ideas have led to the emergence of several different kinds of intelligent system architectures [15].

It is well known that intelligent systems, which can provide human-like expertise such as domain knowledge, uncertain reasoning, and adaptation to a noisy and time-varying environment, are important in tackling practical computing problems. Experience has shown that it is crucial, in the design of hybrid systems, to focus primarily on the integration and interaction of different techniques rather than to merge different methods to create ever-new techniques. Techniques already well understood should be applied to solve specific domain problems within the system. Their weaknesses must be addressed by combining them with complementary methods. Nevertheless, developing hybrid intelligent systems is an open-ended concept rather than restricting to a few technologies. That is, it is evolving those relevant techniques together with the important advances in other new computing methods.

We broadly classify the various hybrid intelligent architectures into 4 different categories based on the system's overall architecture namely stand-alone, transformational, hierarchical hybrid and integrated hybrid systems [5, 17]. Fused architectures are the first true form of integrated intelligent systems. They include systems which combine different techniques into one single computational model. They share data structures and knowledge representations. Another approach is to put the various techniques side-by-side and focus on their interaction in a problem-solving task. This method can allow integration of alternative techniques and exploiting their mutuality. The benefits of integrated models include robustness, improved performance and increased problem-solving capabilities. Finally, fully integrated models can provide a full range of capabilities such as adaptation, generalization, noise tolerance and justification. The two architectures presented in this chapter belong to integrated model.

Section 8.2 presents the hybrid framework for the adaptation of fuzzy inference system using a combination of neural network learning and evolutionary computation. An application example is also included in this section. In Sect. 8.3, we present a hybrid combination of fuzzy clustering algorithm and a fuzzy inference system for a Web mining task. Conclusions are given towards the end.

8.2 Adaptation of Fuzzy Inference Systems

A conventional fuzzy inference system makes use of a model of the expert who is in a position to specify the most important properties of the process. Expert knowledge is often the main source for designing fuzzy inference systems. According to the performance measure of the problem environment, the membership functions, the knowledge base and the inference mechanism are to be adapted. Several research works continue to explore the adaptation of fuzzy inference systems [2, 4, 16]. These include the adaptation of membership functions, rule bases and the aggregation operators. They include but are not limited to:

- The self-organizing process controller by Procyk et al. [19] which considered the issue of rule generation and adaptation.
- The gradient descent and its variants which have been applied to fine-tune the parameters of the input and output membership functions [25].
- Pruning the quantity and adapting the shape of input/output membership functions [23].
- Tools to identify the structure of fuzzy models [21].
- In most cases the inference of the fuzzy rules is carried out using the *min* and *max* operators for fuzzy intersection and union. If the T-norm and T-conorm operators are parameterized then the gradient descent technique could be used in a supervised learning environment to fine-tune the fuzzy operators.

The antecedent of the fuzzy rule defines a local fuzzy region, while the consequent describes the behavior within the region via various constituents. The consequent constituent can be a membership function (Mamdani model) or a linear equation (first order Takagi-Sugeno model) [22].

Adaptation of fuzzy inference systems using evolutionary computation techniques has been widely explored [2, 4, 7, 8, 18, 20]. The automatic adaptation of membership functions is popularly known as self-tuning. The genome encodes parameters of trapezoidal, triangle, logistic, hyperbolic-tangent, Gaussian membership functions and so on.

The evolutionary search of fuzzy rules can be carried out using three approaches [10]. In the first (Michigan approach), the fuzzy knowledge base is adapted as a result of the antagonistic roles of competition and cooperation of fuzzy rules. Each genotype represents a single fuzzy rule and the entire population represents a solution. The second method (Pittsburgh approach) evolves a population of knowledge bases rather than individual fuzzy rules. Genetic operators serve to provide a new combination of rules and new rules. The disadvantage is the increased complexity of the search space and the additional computational burden, especially for online learning. The third method (iterative rule learning approach) is similar to the first, with each chromosome representing a single rule, but contrary to the Michigan approach, only the best individual is considered to form part of the solution, the remaining chromosomes in the population are discarded. The evolutionary learning process builds up the complete rule base through an iterative learning process.

In a neuro-fuzzy model [4], there is no guarantee that the neural network-learning algorithm will converge and the tuning of fuzzy inference system be successful (determining the optimal parameter values of the membership functions, fuzzy operators and so on). A distinct feature of evolutionary fuzzy systems is their adaptability to a dynamic environment. Experimental evidence had indicated cases where evolutionary algorithms are inefficient at fine tuning solutions, but better at finding global basins of attraction [2, 6, 14]. The efficiency of evolutionary training can be improved significantly by incorporating a local search procedure into the evolution. Evolutionary algorithms are used to first locate a good region in the space and then a local search procedure is used to find a near optimal solution in this region. It is interesting to consider finding good initial parameter values as locating a good region in the space. Defining that the basin of attraction of a local minimum is composed of all the points, sets of parameter values in this case, which can converge to the local minimum through a local search algorithm, then a global minimum can easily be found by the local search algorithm if the evolutionary algorithm can locate any point, i.e, a set of initial parameter values, in the basin of attraction of the global minimum. Referring to Fig. 8.1, G_1 and G_2 could be considered as the initial parameter values as located by the evolutionary search and W_A and W_B the corresponding final parameter values fine-tuned by the meta-learning technique.

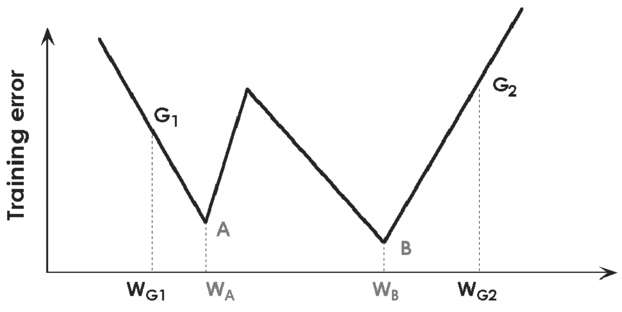

Fig. 8.1. Fine tuning of parameters using hybrid learning

We present the Evolving Neuro Fuzzy (EvoNF) model which optimizes the fuzzy inference system using a meta-heuristic approach combining neural network learning and evolutionary computation. The proposed technique could be considered as a methodology to integrate neural network learning, fuzzy inference systems and evolutionary search procedures [2, 8].

The evolutionary search of membership functions, rule base, fuzzy operators progress on different time scales to adapt the fuzzy inference system according to the problem environment. Figure 8.2 illustrates the general interaction mechanism with the evolutionary search of a fuzzy inference system (Mamdani, Takagi-Sugeno etc.) evolving at the highest level on the slowest time scale. For each evolutionary search of fuzzy operators (for example, best combination of T-norm, T-conorm and defuzzification strategy), the search for the fuzzy rule base progresses at a faster time scale in an environment decided by the fuzzy inference system and the problem. In a similar manner, the evolutionary search of membership functions proceeds at a faster time scale

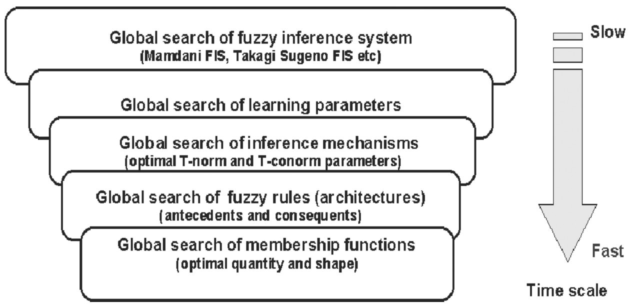
Global search of fuzzy inference system
(Mamdani FIS, Takagi Sugeno FIS etc)
Global search of learning parameters
Global search of inference mechanisms
(optimal T-norm and T-conorm parameters)
Global search of fuzzy rules (architectures)
(antecedents and consequents)
Global search of membership functions
(optimal quantity and shape)
Slow
Fast
Time scale

Fig. 8.2. EvoNF general computational framework

(for every rule base) in the environment decided by the fuzzy inference system, fuzzy operators and the problem. Thus, the evolution of the fuzzy inference system evolves at the slowest time scale while the evolution of the quantity and type of membership functions evolves at the fastest rate. The function of the other layers could be derived similarly. The hierarchy of the different adaptation layers (procedures) relies on prior knowledge. For example, if there is more prior knowledge about the knowledge base (*if-then* rules) than the inference mechanism then it is better to implement the knowledge base at a higher level. If a particular fuzzy inference system best suits the problem, the computational task could be reduced by minimizing the search space. The chromosome architecture is depicted in Fig. 8.3.

The architecture and the evolving mechanism could be considered as a general framework for adaptive fuzzy systems, that is a fuzzy model that can change membership functions (quantity and shape), rule base (architecture), fuzzy operators and learning parameters according to different environments without human intervention.

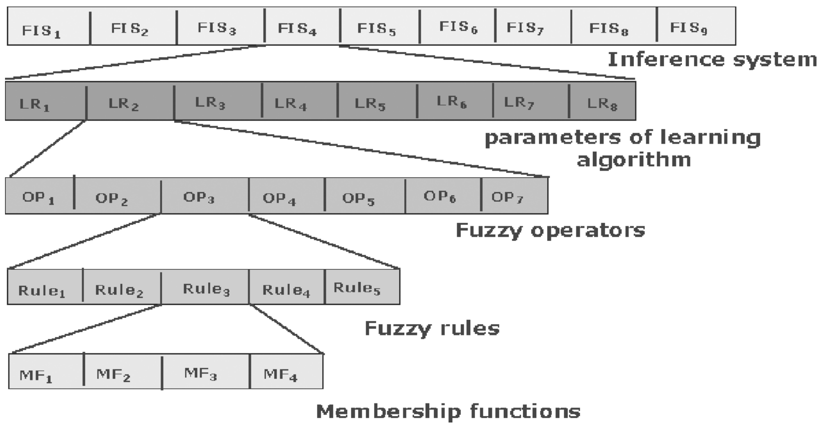
FIS₁ FIS₂ FIS₃ FIS₄ FIS₅ FIS₆ FIS₇ FIS₈ FIS₉
Inference system
LR₁ LR₂ LR₃ LR₄ LR₅ LR₆ LR₇ LR₈
parameters of learning algorithm
OP₁ OP₂ OP₃ OP₄ OP₅ OP₆ OP₇
Fuzzy operators
Rule₁ Rule₂ Rule₃ Rule₄ Rule₅
Fuzzy rules
MF₁ MF₂ MF₃ MF₄
Membership functions

Fig. 8.3. Chromosome structure of EvoNF model

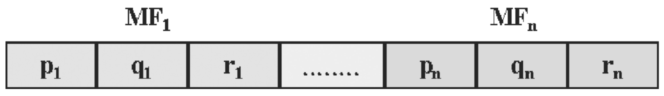

Fig. 8.4. Representation of n membership functions of a bell shape MF

Referring to Fig. 8.3 each layer (from fastest to slowest) of the hierarchical evolutionary search process has to be represented in a chromosome for successful modelling of EvoNF. The detailed functioning and modelling process is as follows.

Layer 1: The simplest way is to encode the number of membership functions per input variable and the parameters of the membership functions. Figure 8.4 depicts the chromosome representation of *n bell* membership functions specified by its parameters p, q and r. The optimal parameters of the membership functions located by the evolutionary algorithm will be further fine tuned by the neural network-learning algorithm. Similar strategy could be used for the output membership functions in the case of a Mamdani fuzzy inference system. Experts may be consulted to estimate the MF shape forming parameters to estimate the search space of the MF parameters.

In our experiments angular coding method proposed by Cordón et al. were used to represent the rule consequent parameters of the Takagi-Sugeno inference system [10].

Layer 2. This layer is responsible for the optimization of the rule base. This includes deciding the total number of rules, representation of the antecedent and consequent parts. Depending on the representation used (Michigan, Pittsburg, iterative learning and so on), the number of rules grow rapidly with an increasing number of variables and fuzzy sets. The simplest way is that each gene represents one rule, and "1" stands for a selected and "0" for a non-selected rule. Figure 8.5 displays such a chromosome structure representation. To represent a single rule a position dependent code with as many elements as the number of variables of the system is used. Each element is a binary string with a bit per fuzzy set in the fuzzy partition of the variable, meaning the absence or presence of the corresponding linguistic label in the rule. For a three input and one output variable, with fuzzy partitions composed of 3, 2, 2 fuzzy sets for input variables and 3 fuzzy sets for output variable, the fuzzy rule will have a representation as shown in Fig. 8.6.

Layer 3. In this layer, a chromosome represents the different parameters of the T-norm and T-conorm operators. Real number representation is adequate

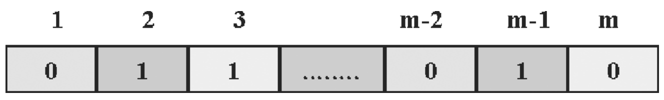

Fig. 8.5. Representation of the entire rule base consisting of m fuzzy rules

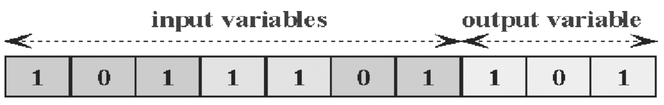

Fig. 8.6. Representation of an individual fuzzy rule

to represent the fuzzy operator parameters. The parameters of the operators could be even fine-tuned using gradient descent techniques.

Layer 4. This layer is responsible for the selection of optimal learning parameters. Performance of the gradient descent algorithm directly depends on the learning rate according to the error surface. Real number representation may be used to represent the learning parameters. The optimal learning parameters decided by the evolutionary algorithm will be used in the learning algorithm to tune the membership functions and the inference mechanism.

Layer 5. This layer basically interacts with the environment and decides which fuzzy inference system (Mamdani type and its variants, Takagi-Sugeno type, Tsukamoto type etc.) will be the optimal according to the environment.

Once the chromosome representation, C, of the entire EvoNF model is done, the evolutionary search procedure could be initiated as follows:

1. *Generate an initial population of N numbers of C chromosomes. Evaluate the fitness of each chromosome depending on the problem.*
2. *Depending on the fitness and using suitable selection methods reproduce a number of children for each individual in the current generation.*
3. *Apply genetic operators to each child individual generated above and obtain the next generation.*
4. *Check whether the current model has achieved the required error rate or the specified number of generations has been reached. Go to Step 2.*
5. *End*

8.2.1 Application of EvoNF – Export Behavior Modelling

In this section, we will examine the application of the proposed EvoNF model to approximate the export behavior of multi-national subsidiaries. Several specific subsidiary features identified in international business literature are particularly relevant when seeking to explain Multi-National Company (MNC) subsidiary export behavior. Our purpose is to is to model the complex export pattern behavior using a Takagi-Sugeno fuzzy inference system in order to determine the actual volume of Multinational Cooperation Subsidiaries (MCS) export output (sales exported) [11]. Malaysia has been pursuing an economic strategy of export-led industrialization. To facilitate this strategy, foreign investment is courted through the creation of attractive incentive packages. These primarily entail taxation allowances and more liberal ownership rights for investments. The quest to attract foreign direct investment (FDI) has proved to be highly successful. The bulk of investment has gone into

Table 8.1. Parameter settings of EvoNF framework

Population Size	40
Maximum no of generations	35
FIS	Takagi Sugeno
Rule antecedent MF	2 MF (parameterised Gaussian)/input
Rule consequent parameters	Linear parameters
Gradient descent learning	10 epochs
Ranked based selection	0.50
Elitism	5%
Starting mutation rate	0.50

export-oriented manufacturing industries. For simulations we have used data provided from a survey of 69 Malaysian MCS. Each corporation subsidiary data set were represented by product manufactured, resources, tax protection, involvement strategy, financial independence and suppliers relationship.

We used the popular grid partitioning method to generate the initial rule base [24]. This partition strategy works well when only few number of inputs are involved since it requires only a small number of MF for each input. We used 90% of the data for training and remaining 10% for testing and validation purposes. The initial populations were randomly created based on the parameters shown in Table 8.1. We used an adaptive mutation operator, which decreases the mutation rate as the algorithm greedily proceeds in the search space 0. The parameters mentioned in Table 8.1 were decided after a few trial and error approaches. Experiments were repeated 3 times and the average performance measures are reported. Figure 8.7 illustrates the meta-learning approach for training and test data combining evolutionary learning and gradient descent technique during the 35 generations.

The 35 generations of meta-learning approach created 76 *if-then* Takagi-Sugeno type fuzzy *if-then* rules compared to 128 rules using the conventional

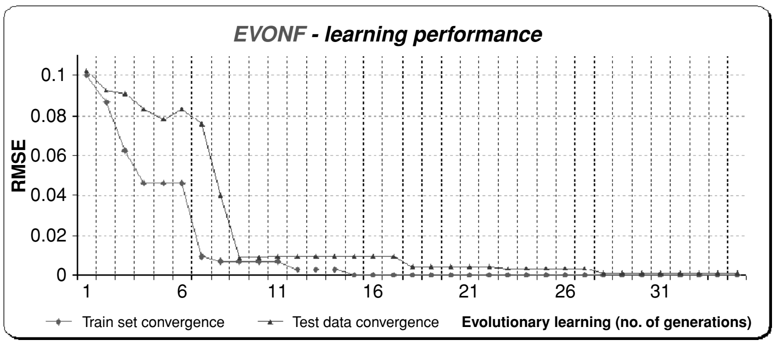

Fig. 8.7. Meta-learning performance (training and test) of EvoNF framework

Table 8.2. Training and test performance of the different intelligent paradigms

	Intelligent Paradigms					
	EvoNF			Neural Network		
Export Output	RMSE		CC	RMSE		CC
	Train	Test		Train	Test	
	0.0013	0.012	0.989	0.0107	0.1261	0.946

grid-partitioning method. We also used a feed forward neural network with 12 hidden neurons (single hidden layer) to model the export output for the given input variables. The learning rate and momentum were set at 0.05 and 0.2 respectively and the network was trained for 10,000 epochs using BP. The network parameters were decided after a trial and error approach. The obtained training and test results are depicted in Table 8.2 (RMSE = Root Mean Squared Error, CC = correlation coefficient).

Our analysis on the export behavior of Malaysia's MCS reveals that the developed EvoNF model could learn the chaotic patterns and model the behavior using an optimized Takagi Sugeno FIS. As illustrated in Fig. 8.8 and Table 8.2, EvoNF could approximate the export behavior within the tolerance limits. When compared to a direct neural network approach, EvoNF performed better (in terms of lowest RMSE) and better correlation coefficient.

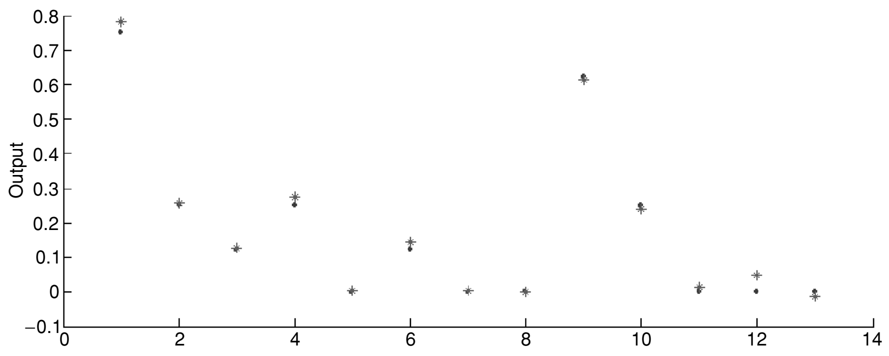

Fig. 8.8. Test results showing the export output (scaled values) for 13 MNC's with respect to the desired values

8.2.2 Hints and Tips to Design EvoNF Architecture

- Incorporate prior knowledge to minimize the number of layers. This will reduce the computational complexity and development time.
- The learning parameter layer could be the first to abolish. By prefixing a low learning rate, we could almost ensure proper meta-learning performance.

- For many function approximation problems, Takagi-Sugeno inference method seems to work very well compared to a Mamdani inference system.
- Computational complexity could be minimized by selecting an appropriate initial rule base which has also minimal number of rules. Instead of the grid partition method, other partition methods (tree partition, scatter partition etc.) could be explored.
- For most function approximation problems Gaussian membership functions seems to work very well. In most cases it is not worth to explore more than 3 membership functions for an input variable.
- Adequate genotype representation of the different layers is very important. Try to use simple representation as much as possible.
- It is efficient to start with a high mutation rate (as high as 0.80) and let the adaptive algorithm pick up the mutation rate according to the generated error.

In the following section we present an extension of the EvoNF architecture discussed in Sect. 8.2. The advanced hybrid framework uses a fuzzy c-means algorithm to segregate the data and a fuzzy inference system for data mining purposes. The parameters of the fuzzy c-means algorithm and the fuzzy inference system are represented in hierarchical layers and are optimized using evolutionary algorithm and gradient decent method.

8.3 Hybrid Fuzzy Clustering and Fuzzy Inference Method for Data Mining

8.3.1 Fuzzy Clustering Algorithm

One of the widely used clustering methods is the fuzzy c-means (FCM) algorithm developed by Bezdek [9]. FCM partitions a collection of n vectors $x_i, i = 1, 2 \ldots, n$ into c fuzzy groups and finds a cluster center in each group such that a cost function of dissimilarity measure is minimized. To accommodate the introduction of fuzzy partitioning, the membership matrix U is allowed to have elements with values between 0 and 1. The FCM objective function takes the form

$$J(U, c_1, \ldots c_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d_{ij}^2$$

Where u_{ij}, is a numerical value between [0,1]; c_i is the cluster center of fuzzy group i; $d_{ij} = \|c_i - x_j\|$ is the Euclidian distance between ith cluster center and jth data point; and m is called the exponential weight which influences the degree of fuzziness of the membership (partition) matrix. For every data object the sum of membership values with all the associated clusters should add up to one.

8.3.2 Optimization of Fuzzy Clustering Algorithm

Optimization of usually a number of cluster centers are randomly initialized and the FCM algorithm provides an iterative approach to approximate the minimum of the objective function starting from a given position and leads to any of its local minima. No guarantee ensures that FCM converges to an optimum solution (can be trapped by local extrema in the process of optimizing the clustering criterion). The performance is very sensitive to initialization of the cluster centers. An evolutionary algorithm is used to decide the optimal number of clusters and their cluster centers. The algorithm is initialized by constraining the initial values to be within the space defined by the vectors to be clustered. A very similar approach is used by Hall et al. [12] and [3].

8.3.3 Intelligent Miner (i-Miner) Framework for Web Mining

We propose an integrated framework (*i-Miner*) which optimizes the FCM using an evolutionary algorithm and a Takagi-Sugeno fuzzy inference system using a combination of evolutionary algorithm and neural network learning [1, 3]. The developed framework is used for a Web usage mining problem. Web usage mining attempts to discover useful knowledge from the secondary data obtained from the interactions of the users with the Web. The rapid e-commerce growth has made both business community and customers face a new situation. Due to intense competition on one hand and the customer's option to choose from several alternatives business community has realized the necessity of intelligent marketing strategies and relationship management. Web usage mining has become very critical for effective Web site management, creating adaptive Web sites, business and support services, personalization, network traffic flow analysis and so on. We present how the (*i-Miner*) approach could be used to optimize the concurrent architecture of a fuzzy clustering algorithm (to discover web data clusters) and a fuzzy inference system to analyze the Web site visitor trends. Figure 8.9 illustrates the *i-Miner* framework. A hybrid evolutionary FCM algorithm is used to optimally segregate similar user interests. The clustered data is then used to analyze the trends using a Takagi-Sugeno fuzzy inference system learned using EvoNF approach.

In Web usage mining, data pre-processing involves mundane tasks such as merging multiple server logs into a central location and parsing the log into data fields followed by data cleaning. Graphic file requests, agent/spider crawling etc. could be easily removed by only looking for HTML file requests. Normalization of URL's is often required to make the requests consistent. For example requests for www.okstate.edu and www.okstate.edu/index.html, are all for the same file. All these tasks could be achieved by conventional hard computing techniques which involves text processing, string matching, association rules, simple statistical measures etc. The cleaned and pre-processed raw data from the log files is used by evolutionary FCM algorithm to identify the optimal number of clusters and their centers. The developed clusters of

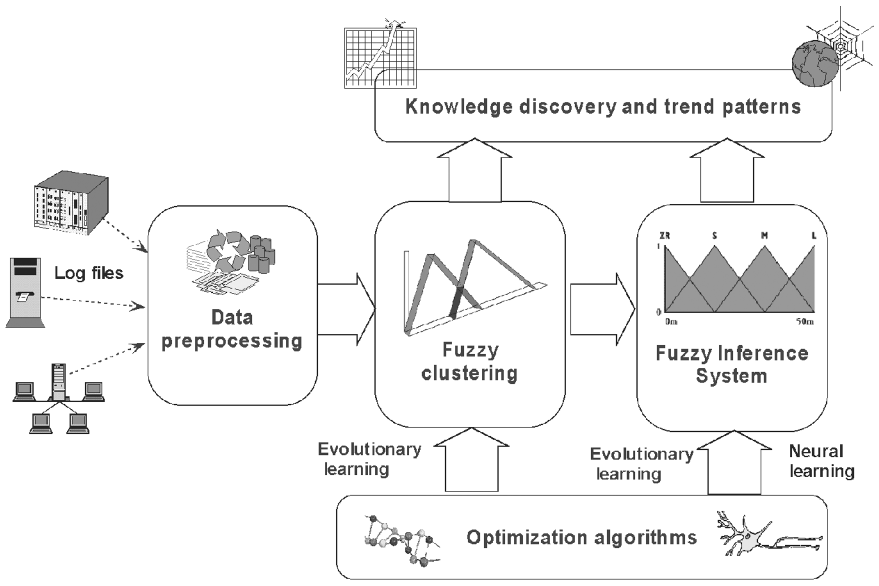

Fig. 8.9. *i-Miner* framework for Web usage mining

data are fed to a Takagi-Sugeno fuzzy inference system to analyze the Web server access trend patterns. The *if-then* rule structures are learned using an iterative learning procedure [10] by an evolutionary algorithm and the rule parameters are fine-tuned using gradient decent algorithm. The hierarchical distribution of *i-Miner* is depicted in Fig. 8.10. The arrow direction indicates the hierarchy of the evolutionary search. In simple words, the optimization of clustering algorithm progresses at a faster time scale at the lowest level in an environment decided by the fuzzy inference system and the problem envi-

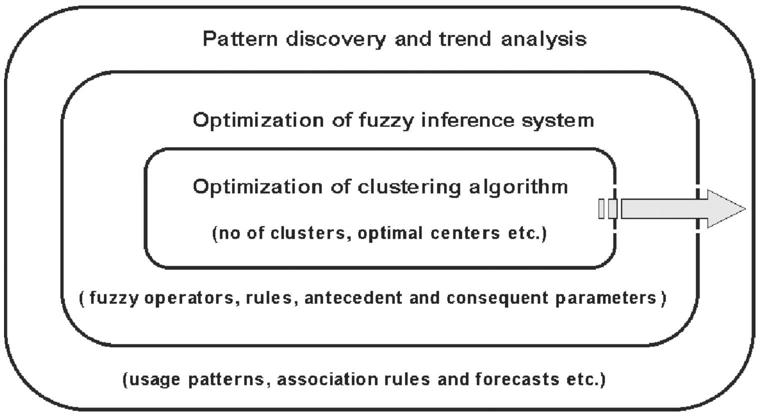
Pattern discovery and trend analysis
Optimization of fuzzy inference system
Optimization of clustering algorithm
(no of clusters, optimal centers etc.)
(fuzzy operators, rules, antecedent and consequent parameters)
(usage patterns, association rules and forecasts etc.)

Fig. 8.10. Hierarchical architecture of *i-Miner*

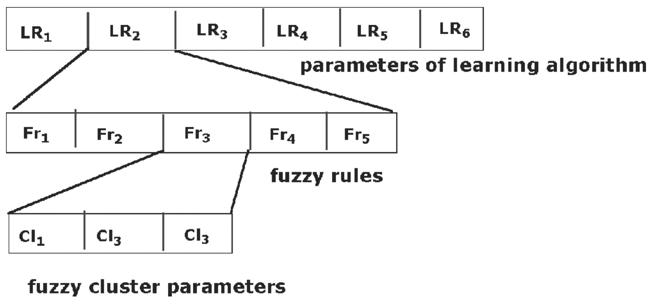

Fig. 8.11. Chromosome structure of the *i-Miner*

ronment. The evolution (optimization) of the fuzzy inference system proceeds at a slower time scale at a higher level with respect to the fuzzy clustering algorithm.

8.3.4 Chromosome Modelling and Representation

Hierarchical evolutionary search process has to be represented in a chromosome for successful modelling of the *i-Miner* framework. A typical chromosome of the *i-Miner* would appear as shown in Fig. 8.11 and the detailed modelling process is as follows.

Layer 1. The optimal number of clusters and initial cluster centers is represented this layer.

Layer 2. This layer is responsible for the optimization of the rule base (same function defined in Fig. 8.3). We used the grid-partitioning algorithm to generate the initial set of rules. An iterative learning method is then adopted to optimize the rules. The existing rules are mutated and new rules are introduced. The fitness of a rule is given by its contribution (strength) to the actual output. To represent a single rule a position dependent code with as many elements as the number of variables of the system is used.

Layer 3. This layer is responsible for the selection of optimal learning parameters. Performance of the gradient descent algorithm directly depends on the learning rate according to the error surface. The optimal learning parameters decided by this layer will be used to tune the parameterized rule antecedents/consequents and the fuzzy operators. In the *i-Miner* approach, rule antecedent/consequent parameters and the fuzzy operators are fine tuned using a gradient descent algorithm to minimize the output error

$$E = \sum_{k=1}^{N} (d_k - x_k)^2 \tag{8.1}$$

where d_k is the kth component of the rth desired output vector and x_k is the kth component of the actual output vector by presenting the rth input vector to the network. The gradients of the rule parameters to be optimized, namely the consequent parameters (P_n) $\frac{\partial E}{\partial P_n}$ for all rules R_n and the premise parameters $\frac{\partial E}{\partial \sigma_i}$ and $\frac{\partial E}{\partial c_i}$ for all fuzzy sets F_i (σ and c represents the MF width and center of a Gaussian MF)are to be computed. As far as rule parameter learning is concerned, the key difference between *i-Miner* and the EvoNF approach is in the way the consequent parameters were determined.

Once the three layers are represented in a chromosome structure C, then the learning procedure could be initiated as defined in Sect. 8.2.

8.3.5 Application of *i-Miner*: Web Usage Mining

The hybrid framework described in Sect. 8.3 was used for Web usage mining [1]. The statistical/text data generated by the log file analyzer from 1 January 2002 to 7 July 2002. Selecting useful data is an important task in the data pre-processing block. After some preliminary analysis, we selected the statistical data comprising of domain byte requests, hourly page requests and daily page requests as focus of the cluster models for finding Web users' usage patterns. It is also important to remove irrelevant and noisy data in order to build a precise model. We also included an additional input "*index number*" to distinguish the time sequence of the data. The most recently accessed data were indexed higher while the least recently accessed data were placed at the bottom. Besides the inputs "*volume of requests*" and " *volume of pages (bytes)*" and "*index number*", we also used the *"cluster information"* provided by the clustering algorithm as an additional input variable. The data was re-indexed based on the cluster information. Our task is to predict the Web traffic volume on a hourly and daily basis. We used the data from 17 February 2002 to 30 June 2002 for training and the data from 1 July 2002 to 6 July 2002 for testing and validation purposes.

The performance is compared with self-organizing maps (alternative for FCM) and several function approximation techniques like neural networks, linear genetic programming and Takagi-Sugeno fuzzy inference system (to predict the trends). The results are graphically illustrated and the practical significance is discussed in detail.

The initial populations were randomly created based on the parameters shown in Table 8.3. Choosing good reproduction operator values is often a challenging task. We used a special mutation operator, which decreases the mutation rate as the algorithm greedily proceeds in the search space [3]. If the allelic value x_i of the ith gene ranges over the domain a_i and b_i the mutated gene $x_i^{'}$ is drawn randomly uniformly from the interval $[a_i,\, b_i]$.

$$x_i^{'} = \begin{cases} x_i + \mathit{\Delta}(t, b_i - x_i), & if\ \omega = 0 \\ x_i + \mathit{\Delta}(t, x_i - a_i), & if\ \omega = 1 \end{cases} \tag{8.2}$$

where ω represents an unbiased coin flip $p(\omega = 0) = p(\omega = 1) = 0.5$, and

$$\mathit{\Delta}(t, x) = x\ \left(1 - \gamma^{\left(1 - \frac{t}{t_{\max}}\right)^b}\right) \tag{8.3}$$

Table 8.3. Parameter settings of *i-Miner*

Population Size	30
Maximum no of generations	35
Fuzzy inference system	Takagi Sugeno
Rule antecedent membership functions	3 membership functions per input variable (parameterized Gaussian)
Rule consequent parameters	linear parameters
Gradient descent learning	10 epochs
Ranked based selection	0.50
Elitism	5%
Starting mutation rate	0.50

and t is the current generation and t_{max} is the maximum number of generations. The function Δ computes a value in the range $[0, x]$ such that the probability of returning a number close to zero increases as the algorithm proceeds with the search. The parameter b determines the impact of time on the probability distribution Δ over $[0, x]$. Large values of b decrease the likelihood of large mutations in a small number of generations. The parameters mentioned in Table 8.3 were decided after a few trial and error approaches (basically by monitoring the algorithm convergence and the output error measures). Experiments were repeated 3 times and the average performance measures are reported. Figures 8.12 and 8.13 illustrates the meta-learning approach combining evolutionary learning and gradient descent technique during the 35 generations.

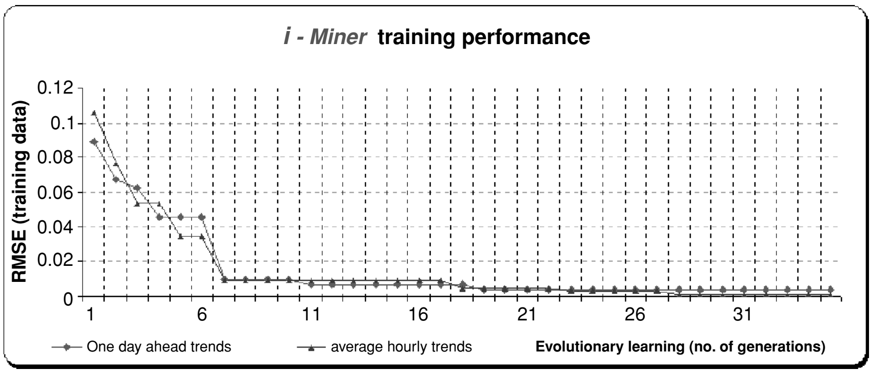

Fig. 8.12. Meta-learning performance (training) of *i-Miner*

Table 8.4 summarizes the performance of the developed *i-Miner* for training and test data. *i-Miner* trend prediction performance is compared with ANFIS [13], Artificial Neural Network (ANN) and Linear Genetic Programming (LGP). The Correlation Coefficient (CC) for the test data set is also

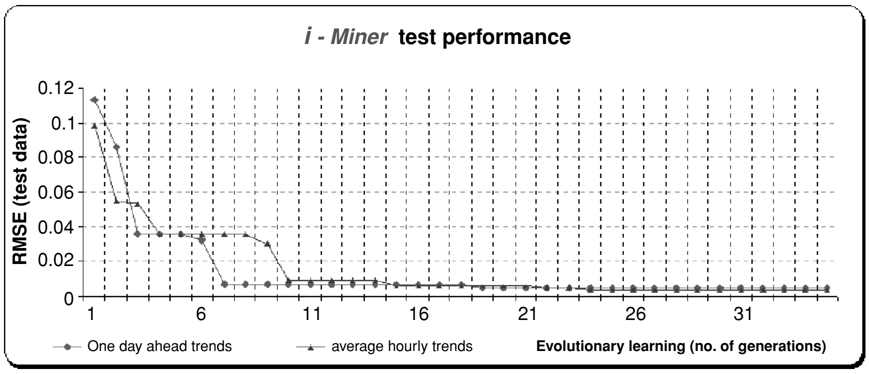

Fig. 8.13. Meta-learning performance (testing) of *i-Miner*

Table 8.4. Performance of the different paradigms

	Period					
	Daily (1 day ahead)			Hourly (1 hour ahead)		
	RMSE			RMSE		
Method	Train	Test	CC	Train	Test	CC
i-Miner	0.0044	0.0053	0.9967	0.0012	0.0041	0.9981
ANFIS	0.0176	0.0402	0.9953	0.0433	0.0433	0.9841
ANN	0.0345	0.0481	0.9292	0.0546	0.0639	0.9493
LGP	0.0543	0.0749	0.9315	0.0654	0.0516	0.9446

given in Table 8.4. The 35 generations of meta-learning approach created 62 *if-then* Takagi-Sugeno type fuzzy rules (daily traffic trends) and 64 rules (hourly traffic trends) compared to the 81 rules reported in [26] using ANFIS [13]. Figures 8.14 and 8.15 illustrate the actual and predicted trends for the test data set. A trend line is also plotted using a least squares fit (6th order polynomial). Empirical results clearly show that the proposed Web usage-mining framework (*i-Miner*) is efficient.

As shown in Fig. 8.16, Evolutionary FCM approach created 7 data clusters for the average hourly traffic according to the input features compared to 9 data clusters for the daily Web traffic (Fig. 8.17). The previous study using Self-organizing Map (SOM) created 7 data clusters for daily Web traffic and 4 data clusters for hourly Web traffic respectively. Evolutionary FCM approach resulted in the formation of additional data clusters.

Several meaningful information could be obtained from the clustered data. Depending on the no of visitors from a particular domain, time and day of access etc. data clusters were formulated. Clusters based on hourly data show the visitor information at certain hour of the day. Some clusters accounted for the visitors during the peak hour and certain weekdays and so on. For example,

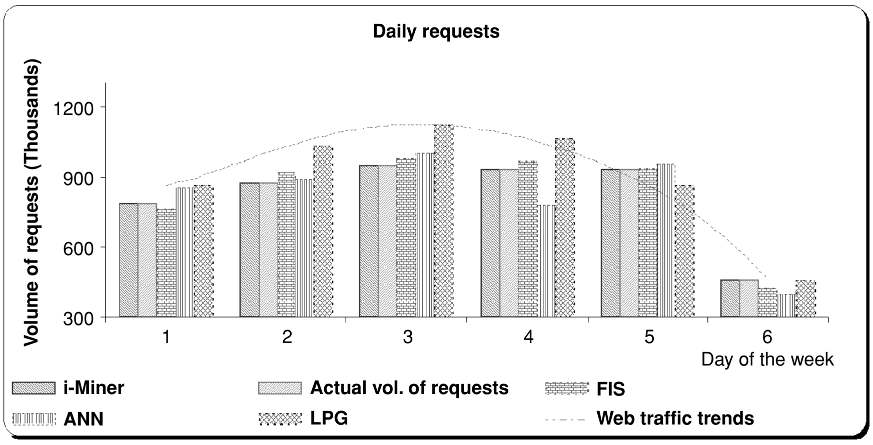

Fig. 8.14. Test results of the daily trends for 6 days

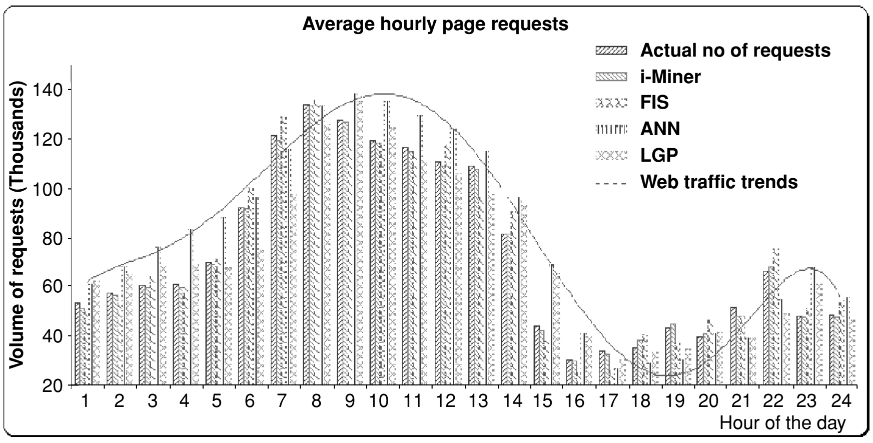

Fig. 8.15. Test results of the average hourly trends for 6 days

Fig. 8.18 depicts the volume of visitors according to domain names from a cluster developed using the evolutionary FCM approach. Detailed discussion on the knowledge discovered from the data clusters is beyond the scope of this chapter.

8.3.6 Hints and Tips to Design *i-Miner*

- Most of the hints and tips given for the design of EvoNF is applicable for *i-Miner*.
- Data pre-processing is an important issue for optimal performance. In most cases, normalization or scaling would suffice.

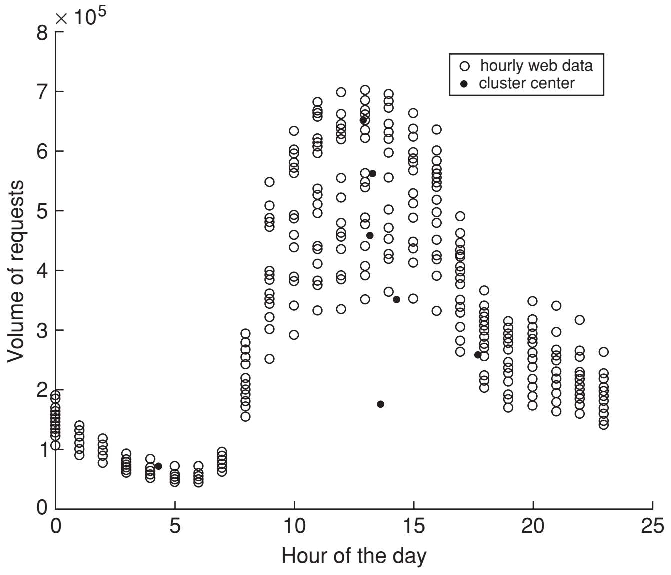

Fig. 8.16. FCM clustering of hourly Web traffic

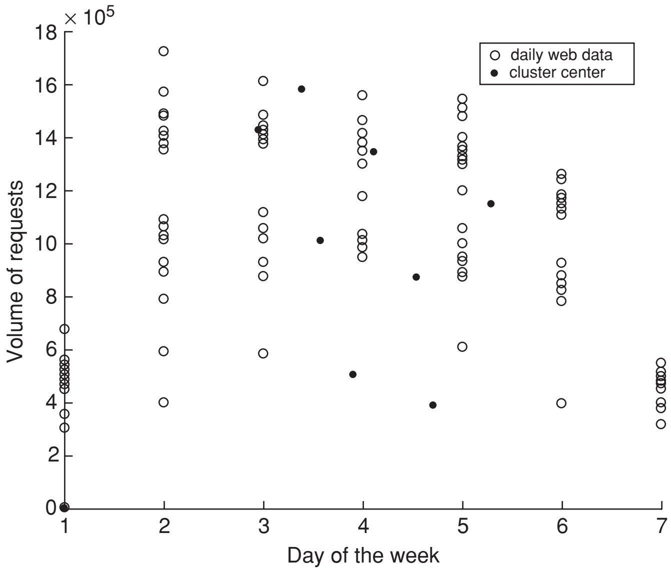

Fig. 8.17. FCM clustering of daily Web traffic

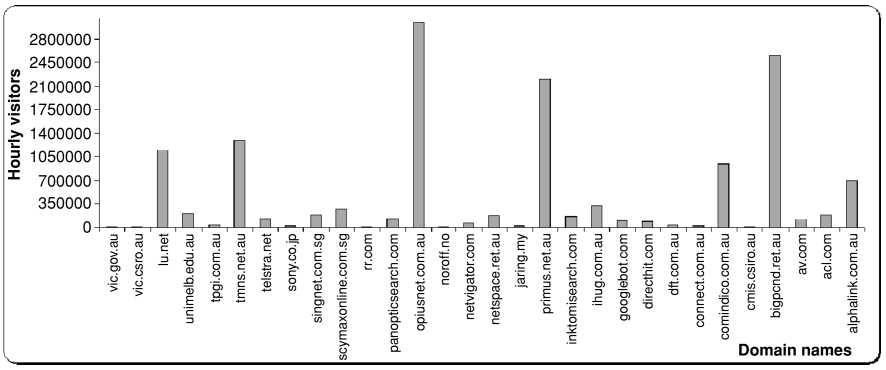

Fig. 8.18. Hourly visitor information according to the domain names from an FCM cluster

8.4 Conclusions

This chapter has presented some of the architectures of hybrid intelligent systems involving fuzzy clustering algorithms, neural network learning, fuzzy inference systems and evolutionary computation. The key idea was to demonstrate the evolution of intelligence in hierarchical layers. The developed hybrid intelligent systems were applied to two real world applications illustrating the importance of such complicated approaches. For the two applications considered, the hybrid models performed better than the individual approaches. We were able to improve the performance (low RMSE and high CC) and at the same time we were able to substantially reduce the number of rules. Hence these approaches might be extremely useful for hardware implementations.

The hybrid intelligent systems has many important practical applications in science, technology, business and commercial. Compared to the individual intelligent constituents hybrid intelligent frameworks are relatively young. As the strengths and weakness of different hybrid architectures are understood, it will be possible to use them more efficiently to solve real world problems. Integration issues range from different techniques and theories of computation to problems of exactly how best to implement hybrid systems. Like most biological systems which can adapt to any environment, adaptable intelligent systems are required to tackle future complex problems involving huge data volume. Most of the existing hybrid soft computing frameworks rely on several user specified network parameters. For the system to be fully adaptable, performance should not be heavily dependant on user-specified parameters.

The real success in modelling the proposed hybrid architectures will directly depend on the genotype representation of the different layers. The population-based collective learning process, self-adaptation, and robustness are some of the key features. Evolutionary algorithms attract considerable computational effort especially for problems involving complexity and huge data volume. Fortunately, evolutionary algorithms work with a population of independent solutions, which makes it easy to distribute the computational load among several processors.

References

1. Abraham A., Business Intelligence from Web Usage Mining, Journal of Information and Knowledge Management (JIKM), World Scientific Publishing Co., Singapore, Volume 2, No. 4, pp. 1-15, 2003.
2. Abraham A., EvoNF: A Framework for Optimization of Fuzzy Inference Systems Using Neural Network Learning and Evolutionary Computation, 2002 IEEE International Symposium on Intelligent Control (ISIC'02), Canada, IEEE Press, pp. 327-332, 2002.
3. Abraham A., i-Miner: A Web Usage Mining Framework Using Hierarchical Intelligent Systems, The IEEE International Conference on Fuzzy Systems FUZZ-IEEE'03, IEEE Press, pp. 1129-1134, 2003.
4. Abraham A., Neuro-Fuzzy Systems: State-of-the-Art Modeling Techniques, Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence, LNCS 2084, Mira J. and Prieto A. (Eds.), Springer-Verlag Germany, pp. 269-276, 2001.
5. Abraham A., Intelligent Systems: Architectures and Perspectives, Recent Advances in Intelligent Paradigms and Applications, Abraham A., Jain L. and Kacprzyk J. (Eds.), Studies in Fuzziness and Soft Computing, Springer Verlag Germany, Chap. 1, pp. 1-35, 2002.
6. Abraham A., Meta-Learning Evolutionary Artificial Neural Networks, Neurocomputing Journal, Elsevier Science, Netherlands, 2003 (in press).
7. Abraham A. and Nath B., Evolutionary Design of Fuzzy Control Systems – An Hybrid Approach, The Sixth International Conference on Control, Automation, Robotics and Vision, (ICARCV 2000), CD-ROM Proceeding, Wang J.L. (Ed.), ISBN 9810434456, Singapore, 2000.
8. Abraham A. and Nath B., Evolutionary Design of Neuro-Fuzzy Systems – A Generic Framework, In Proceedings of The 4-th Japan-Australia Joint Workshop on Intelligent and Evolutionary Systems, Namatame A. et al (Eds.), Japan, pp. 106-113, 2000.
9. Bezdek J.C., Pattern Recognition with Fuzzy Objective Function Algorithms, New York: Plenum Press, 1981.
10. Cordón O., Herrera F., Hoffmann F., and Magdalena L., Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases, World Scientific Publishing Company, Singapore, 2001.
11. Edwards R., Abraham A. and Petrovic-Lazarevic S., Export Behaviour Modeling Using EvoNF Approach, The International Conference on Computational Science (ICCS 2003), Springer Verlag, Lecture Notes in Computer Science- Volume 2660, Sloot P.M.A. et al (Eds.), pp. 169-178, 2003.
12. Hall L.O., Ozyurt I.B., and Bezdek J.C., Clustering with a Genetically Optimized Approach, IEEE Transactions on Evolutionary Computation, Vol. 3, No. 2, pp. 103-112, 1999.
13. Jang J.S.R., ANFIS: Adaptive-Network-BasedFuzzy Inference System, IEEE Transactions in Systems Man and Cybernetics, Vol. 23, No. 3, pp. 665-685, 1993.

14. Jayalakshmi G.A., Sathiamoorthy S. and Rajaram, An Hybrid Genetic Algorithm – A New Approach to Solve Traveling Salesman Problem, International Journal of Computational Engineering Science, Vol. 2, No. 2, pp. 339-355, 2001.
15. Kandel A. and Langholz G. (Eds.), Hybrid Architectures for Intelligent Systems, CRC Press, 1992.
16. Lotfi A., Learning Fuzzy Inference Systems, PhD Thesis, Department of Electrical and Computer Engineering, University of Queensland, Australia, 1995.
17. Medsker L.R., Hybrid Intelligent Systems, Kluwer Academic Publishers, 1995.
18. Pedrycz W. (Ed.), Fuzzy Evolutionary Computation, Kluwer Academic Publishers, USA, 1997.
19. Procyk T.J. and Mamdani E.H., A Linguistic Self Organising Process Controller, Automatica, Vol. 15, no. 1, pp. 15-30, 1979.
20. Sanchez E., Shibata T. and Zadeh L.A. (Eds.), Genetic Algorithms and Fuzzy Logic Systems: Soft Computing Perspectives, World Scientific Publishing Company, Singapore, 1997.
21. Stepniewski S.W. and Keane A.J., Pruning Back-propagation Neural Networks Using Modern Stochastic Optimization Techniques, Neural Computing & Applications, Vol. 5, pp. 76-98, 1997.
22. Sugeno M. and Tanaka K., Successive Identification of a Fuzzy Model and its Applications to Prediction of a Complex System, Fuzzy Sets Systems, Vol. 42, no. 3, pp. 315-334, 1991.
23. Wang L.X. and Mendel J.M., Backpropagation Fuzzy System as Nonlinear Dynamic System Identifiers, In Proceedings of the First IEEE International conference on Fuzzy Systems, San Diego, USA, pp. 1409-1418, 1992.
24. Wang L.X. and Mendel J.M., Generating Fuzzy Rules by Learning from Examples, IEEE Transactions in Systems Man and Cybernetics, Vol. 22, pp. 1414–1427,1992.
25. Wang L.X., Adaptive Fuzzy Systems and Control, Prentice Hall Inc, USA, 1994.
26. Wang X., Abraham A. and Smith K.A, Soft Computing Paradigms for Web Access Pattern Analysis, Proceedings of the 1st International Conference on Fuzzy Systems and Knowledge Discovery, pp. 631-635, 2002.
27. Zadeh L.A., Roles of Soft Computing and Fuzzy Logic in the Conception, Design and Deployment of Information/Intelligent Systems, Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications, Kaynak O. et al (Eds.), pp. 1-9, 1998.