A Comparative Study of Space Search Algorithm and Particle Swarm Optimization in the Design of ANFIS-based Fuzzy Models

Wei Huang[1], Lixin Ding[1], Sung-Kwun Oh[2]
1- State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China.
Email: huangwabc@163.com
2- Department of Electrical Engineering, The University of Suwon, San 2-2, Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea.
Email: ohsk@suwon.ac.kr (Corresponding author)

Received: July 2010 Revised: December 2010 Accepted: January 2011

ABSTRACT:
In this study, we propose a space search algorithm (SSA) and then introduce a hybrid optimization of ANFIS-based fuzzy models based on SSA and information granulation (IG). In comparison with conventional evolutionary algorithms (such as PSO), SSA leads not only to better search performance to find global optimization but is also more computationally effective. In the hybrid optimization of ANFIS-based fuzzy inference system, SSA is exploited to carry out the parametric optimization of the fuzzy model as well as to realize its structural optimization. IG realized with the aid of C-Means clustering helps to determine the initial values of the apex parameters of the membership function of fuzzy model. The overall hybrid identification of ANFIS-based fuzzy models comes in the form of two optimization mechanisms: structure identification (such as the number of input variables to be used, a specific subset of input variables, the number of membership functions, and polynomial type) and parameter identification (viz. the apexes of membership function). The structure identification is developed by SSA and C-Means while the parameter estimation is realized via SSA and a standard least square method. The evaluation of the performance of the proposed model was carried out by using three representative numerical examples such as Non-linear function, gas furnace, and Mackey-Glass time series. A comparative study of SSA and PSO demonstrates that SSA leads to improved performance both in terms of the quality of the model and the computing time required. The proposed model is also contrasted with the quality of some conventional fuzzy models already encountered in the literature.

KEYWORDS: Space Search Algorithm, Particle Swarm Algorithm, Information Granulation, ANFIS-based Fuzzy Inference System.

1. INTRODUCTION

Recently, fuzzy modeling has been utilized in many fields for engineering, medical engineering, and even social science [1]. As for fuzzy model construction, identification of fuzzy rules is one of most important parts in the design of rule-based fuzzy modeling.

Many identification methods for fuzzy models have been studied over the past decades. In the early 1980s, linguistic modeling [2] was proposed as primordial identification methods for fuzzy models. Then Tong et.al [3], C.W.Xu et.al [4] studied different approaches for fuzzy models. While appealing with respect to the basic topology (a modular fuzzy model composed of a series of rules) [5] , these models still await formal solutions as far as the structure optimization of the model is concerned, say a construction of the underlying fuzzy sets – information granules being viewed as basic building blocks of any fuzzy model. Oh and Pedrycz [6] have proposed some enhancements to the model, yet the problem of finding "good" initial parameters of the fuzzy set in the rules remains open. To solve this problem, several genetically identification methods for fuzzy models have been proposed. Liu et.al [7], Chung and Kim [8] and others have discussed employing genetic algorithms to fuzzy models, respectively. In a word, evolutionary identification methods have proven to be useful in optimization of such problems.

In this study, we propose a space search algorithm (SSA) and then introduce a hybrid optimization of ANFIS-based fuzzy models based on SSA and information granulation (IG). SSA is exploited here to carry out the parameter estimation of the fuzzy models as well as to realize structure optimization. The identification process is comprised of two phases, namely a structure optimization (the number of input variables to be used, a specific subset of input variables, and the number of membership functions) and parametric optimization (apexes of membership function). The structural optimization is completed first and then we proceed with the parametric phase. The SSA and the least square method (LSE) are used in each phase of this sequence. IG realized with the aid of HCM, SSA and LSM. HCM is used to help determine the initial parameters of the fuzzy model such as the initial location of apexes of the membership functions and the prototypes of the polynomial functions being used in the premise and consequence parts of the fuzzy rules, while SSA and LSM is employed to adjust the initial values of the parameters. To evaluate the performance of the proposed model, we exploit two kinds of well-known data set. A hybrid optimization of ANFIS-based fuzzy models based on Particle Swarm Optimization (PSO) and IG is also implemented for the comparative study.

2. IG-BASED FUZZY MODEL

In essence, information granules are viewed as highly related collections of objects (data points, in particular) drawn together by some criteria of proximity, similarity, or functionality. Granulation of information is an inherent and omnipresent activity of human beings carried out with intent of gaining a better, more effective insight into a problem under consideration and arriving at its efficient solution. In particular, granulation of information is aimed at transforming the problem at hand into several smaller and therefore more manageable sub-problems. In this way, we partition the task into a series of well-defined subproblems (modules) of far lower computational complexity than the original one. The identification procedure for fuzzy models is split into the identification activities dealing with the development of the premise and the consequence part of rules. The identification completed at the premise level consists of two main steps. First, we select the input variables x_1, x_2, …, x_k of the rules. Second, we form fuzzy partitions (by specifying fuzzy sets of well-defined semantics such as e.g., Low, High, etc.) of the spaces over which these individual variables are defined. In such a sense, this phase is all about information granulation as the elements of the fuzzy partitions we are interested in when developing any rule-based model. The number of the fuzzy sets constructed there implies directly the number of the rules of the model itself. In addition, one has to determine membership functions of the information granules.

The identification of the premise part is completed in the following manner.

Given is a set of data $\mathbf{U}=\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l ; \mathbf{y}\}$, where $\mathbf{x}_k=[x_{1k}, \ldots, x_{mk}]^T$, $\mathbf{y}=[y_1, \ldots, y_m]^T$, where l is the number of variables and m is the number of data.

[Step 1] Arrange a set of data **U** into data set $\mathbf{X}_k$ composed of the corresponding input and output data.

$$X_k=[x_k ; y] \tag{1}$$

[Step 2] Run the K-Means to determine the centers (prototypes) $\mathbf{v}_{kg}$ within the data set $\mathbf{X}_k$.

[Step 2-1] Arrange data set $\mathbf{X}_k$ into c-clusters (in essence this is effectively the granulation of information)

[Step 2-2] Calculate the centers $\mathbf{v}_{kg}$ of each cluster.

$$v_{kg} = \{v_{k1},\ v_{k2}, \ldots, v_{kc}\} \tag{2}$$

[Step 3] Partition the corresponding input space using the prototypes of the clusters $\mathbf{v}_{kg}$. Associate each cluster with some meaning (semantics), say Small, Large, etc.

[Step 4] Set the initial apexes of the membership functions using the prototypes $\mathbf{v}_{kg}$.

As for a part of the consequence identification, we consider the initial values of the polynomial functions based upon the information granulation realized for the consequence and premise part.

[Step 1] Find a set of data included in the fuzzy space of the j-th rule.

[Step 2] Compute the prototypes $\mathbf{V}_j$ of the data set by taking the arithmetic mean of each rule.

$$V_j = \{V_{1j}, V_{2j}, \ldots, V_{kj}; M_j\} \tag{3}$$

[Step 3] Set the initial values of polynomial functions with the center vectors $\mathbf{V}_j$.

The identification of the conclusion parts of the rules deals with a selection of their structure (Type 1, Type 2, Type 3 and Type 4) that is followed by the determination of the respective parameters of the local functions occurring there. The consequence part of the rule that is extended form of a typical fuzzy rule in the TSK (Takagi-Sugeno-Kang) fuzzy model has the form

$$\begin{aligned} &R^j : If\ x_1\ is\ A_{1c}\ and \cdots and\ x_k\ is\ A_{kc} \\ &\quad then\ y_j - M_j = f_j(x_1, \cdots, x_k) \end{aligned} \tag{4}$$

Type 1 (Simplified Inference): $f_j = a_{j0}$

Type 2 (Linear Inference):

$$f_j = a_{j0} + a_{j1}(x_1 - V_{j1}) + \ldots + a_{jk}(x_k - V_{jk})$$

Type 3 (Quadratic Inference):

$$f_j = a_{j0}+a_{j1}(x_1-V_{1j})+\ldots+a_{jk}(x_k-V_{kj})+a_{j(k+1)}(x_1-V_{1j})^2+\ldots+ a_{j(2k)}(x_k-V_{kj})^2+a_{j(2k+1)}(x_1-V_{1j})(x_2-V_{2j})+\ldots+ a_{j((k+2)(k+1)/2)}(x_{k-1}-V_{(k-1)j})(x_k-V_{kj})$$

Type 4 (Modified Quadratic Inference):

$f_j = a_{j0} + a_{j1}(x_1 - V_{1j}) + \ldots + a_{jk}(x_k - V_{kj}) + a_{j(k+1)}(x_1 - V_{1j})(x_2 - V_{2j})$

$+\ldots + a_{j(k(k+1)/2)}(x_{k-1} - V_{(k-1)j})(x_k - V_{kj})$

where, R^j is the j-th fuzzy rule, x_k represents the input variables, A_{kc} is a membership function of fuzzy sets, a_{jk} is a constant, V_{kj} and M_j is a center value of the input and output data, respectively, n is the number of fuzzy rules.

We use two performance indexes as the standard root mean squared error (RMSE) and mean squared error (MSE)

$$PI(or\ E\_PI) = \begin{cases} \sqrt{\dfrac{1}{m}\sum_{i=1}^{m}(y_i - y_i^*)^2}, & (RMSE) \\ \dfrac{1}{m}\sum_{i=1}^{m}(y_i - y_i^*)^2. & (MSE) \end{cases} \quad (5)$$

where y^* is the output of the fuzzy model, m is the total number of data, and i is the data number.

The calculation of the numeric output of the model, based on the activation (matching) levels of rules there, relies on the following expression

$$y^* = \frac{\sum_{j=1}^{n} w_{ji} y_i}{\sum_{j=1}^{n} w_{ji}} = \frac{\sum_{j=1}^{n} w_{ji}(f_j(x_1,\cdots,x_k) + M_j)}{\sum_{j=1}^{n} w_{ji}} \quad (6)$$
$$= \sum_{j=1}^{n} \hat{w}_{ji}(f_j(x_1,\cdots,x_k) + M_j)$$

Here, y^* is the inferred output value, w_{ji} is the premise level of matching R^j (activation level). As the normalized value of w_{ji}, we use an abbreviated notation to describe an activation level of rule R^j to be in the form

$$\hat{w}_{ji} = \frac{w_{ji}}{\sum_{j=1}^{n} w_{ji}}, \quad \hat{w}_{ji} = \frac{A_{j1}(x_{1i}) \times \cdots \times A_{jk}(x_{ki})}{\sum_{j=1}^{n} A_{j1}(x_{1i}) \times \cdots \times A_{jk}(x_{ki})} \quad (7)$$

The consequence parameters a_{jk} can be determined by the standard least-squares method, which leads to the expression

$$\hat{\mathbf{a}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \quad (8)$$

In the case of Type 2 scheme, we have

$\hat{\mathbf{a}} = [a_{10} \cdots a_{n0}\ a_{11} \cdots a_{n1} \cdots a_{1k} \cdots a_{nk}]^T$,

$\mathbf{X} = [\mathbf{x}_1\ \ \mathbf{x}_2\ \ \cdots\ \ \mathbf{x}_i\ \ \cdots\ \ \mathbf{x}_m]^T$,

$\mathbf{x}_i^T = [\ \hat{w}_{1i} \cdots \hat{w}_{ni}\ (x_{1i} - V_{11})\hat{w}_{1i} \cdots (x_{1i} - V_{1n})\hat{w}_{ni} \cdots$
$(x_{ki} - V_{k1})\hat{w}_{1i} \cdots (x_{ki} - V_{kn})\hat{w}_{ni}\ ]$,

$$\mathbf{Y} = \left[y_1 - \left(\sum_{j=1}^{n} M_j w_{j1}\right)\ \ y_2 - \left(\sum_{j=1}^{n} M_j w_{j2}\right)\ \cdots\ y_m - \left(\sum_{j=1}^{n} M_j w_{jm}\right) \right]^T$$

3. OPTIMIZATION

In this section, we provide a very brief description of the essence of the PSO and SSA.

3.1. Particle Swarm Optimization

Generally, Particle Swarm Optimization is utilized as a useful optimization vehicle to deal with the optimization problem. PSO is an example of a modern search heuristics belonging to the category of Swarm Intelligence methods. PSO [9, 10] is an example of a modern search heuristics belonging to the category of Swarm Intelligence methods. We provide a very brief description of the essence of the algorithm and then show its direct use in feature selection.

The underlying principle of PSO involves a population-based search in which individuals representing possible solutions carry out a collective search by exchanging their individual findings while taking into consideration their own experience and evaluating their own performance. PSO involves two competing search strategy aspects [11, 12]. One deals with a social facet of the search; according to this, individuals ignore their own experience and adjust their behavior according to the successful beliefs of individuals occurring in their neighborhood. The cognition aspect of the search underlines the importance of the individual experience where the element of population is focused on its own history of performance and makes adjustments accordingly.

PSO is conceptually simple, easy to implement, and computationally efficient. Unlike many other heuristic techniques, PSO has a flexible and well-balanced mechanism to enhance the global and local exploration abilities [11]. The basic elements of PSO technique are briefly introduced as follows:

Performance index (fitness). Each particle is characterized by some value of the underlying performance (objective) index or fitness. This is a tangible indicator stating how well the particle is doing in the search process. The fitness is reflective of the nature of the problem for which an optimal solution is being looked for. Depending upon the nature of the problem at hand, the fitness is either minimized or maximized.

Particles. The vectors of the variables (particles) in the n-dimensional search space will be denoted by p_1, p_2, …, p_N. In the search, a swarm is composed of “N” particles involved leading to the concept of a swarm. The performance of each particle is described by some objective function referred to as a fitness (objective) function.

Best particles. As a particle wanders through the search space, we compare its fitness at the current position with the best fitness value it has so far attained. This is done for each element in the swarm. The location of the particle at which it has attained the best fitness is denoted by pbest. Similarly, by gbest we denote the best location attained among all pbest.

Velocity. The particle is moving in the search space with some velocity which plays a pivotal role in the search process. Denote the velocity of the i-th particle by vi. From iteration to iteration, the velocity is governed by the following expression

$$\begin{aligned}\mathbf{v}_i &= w\mathbf{v}_i + c_1 r_1 (\text{pbest}_i\text{-}p_i) \\ &+ c_2 r_2 (\text{gbest-}p_i)\end{aligned} \tag{9}$$

Or equivalently

$$\begin{aligned}v_{ik} &= w \cdot v_{ik} + c_1 r_1 (\text{pbest}_{ik} - p_{ik}) \\ &+ c_2 r_2 (\text{gbest}_k - p_{ik})\end{aligned} \tag{10}$$

i=1,2,…,N; k=1,2,….,n where, r_1 and r_2 are random values in [0,1], and c_1 and c_2 are positive constants, called the acceleration constants and referred to as the cognitive and social parameters, respectively. As the above expression shows, c_1 and c_2 reflect the weighting of the stochastic acceleration terms that pull the *i*-th particle toward pbest_i and gbest positions. Low values allow particles to roam far from the target regions before being tugged back. High values of c_1 and c_2 result in abrupt movement toward, or past, target regions. Typically, the values of these constants are set to 2.0. The inertia factor "w" is a control parameter that is used to establish the impact of the previous velocity on the current velocity. Hence, it influences the tradeoff between the global and local exploration abilities of the particles. For initial stages of the search process, large values enhancing the global exploration of the space are recommended. As the search progresses, the values of "w" are gradually reduced to achieve better exploration at the local level.

As PSO is an iterative search strategy, we proceed until there is no substantial improvement of the fitness or we have exhausted the number of iterations allowed in this search. Overall, the algorithm can be outlined as the following sequence of steps:

Step 1: Randomly generate "N" particles, p_i, and their velocities v_i. Each particle in the initial swarm (population) is evaluated using the objective function. For each particle, set $\text{pbest}_i=p_i$ and search the best particle of pbest. Set the best particle associated with the global best, gbest.

Step 2: Adjust the inertia weight, w. Typically, its values decrease linearly over the time of search. We start with w_{max}=0.9 at the beginning of the search and move down to w_{min}=0.4 at the end of the iterative process,

$$w(t) = w_{max} - \frac{w_{max} - w_{min}}{\text{iter}_{max}} \times t \tag{11}$$

Where iter_{max} denotes the maximum number of iterations of the search and "t" stands for the current index of the iteration.

Step 3: Given the current values of gbest and pbest_i, the velocity of the *i*-th particle is adjusted following (10). If required, we clip the values making sure that they are positioned within the required region.

Step 4: Based on the updated velocities, each particle changes its position using the expression

$$p_{ik} = v_{ik} + p_{ik} \tag{12}$$

Furthermore, we keep the particle within the boundaries of the search space, that is

$$p_k^{min} \le p_{ik} \le p_k^{max} \tag{13}$$

Step 5: Move the particles in the search space and evaluate their fitness both in terms of pbest_i and gbest.

Step 6: Repeat from Step 2 to Step 5 until the termination criterion has not been met. Otherwise return gbest as the solution found.

3.2. Space Search Algorithm

SSA is a heuristic algorithm whose search method comes with the analysis of the solution space. In essence, the solution space is the set of all feasible solutions for the optimization problem (or mathematical programming problem), which is stated as the problem of determining the best solution coming from the solution space. To illustrate the idea of the SSA, let us consider why an evolutionary algorithm (such as the well-known genetic algorithm) can find the optimal solution. In fact, a precondition should be satisfied when evolutionary algorithm can find the optimal solution. The precondition is that, in most of local areas, a point (solution) and the other points located in the point's adjacent space have the similar values of the objective function (fitness values). In other words, in most of local areas, a solution with better fitness is closer to the optimal solution. Moreover, if we take the entire space as the biggest local area into consideration, the precondition can be satisfied for any target optimization problems. Based on this observation, we may give rise to a space search mechanism to update the current solutions. The role of space search is to generate new solutions from old ones. The search method is based on the operator of space search, which generates two basic steps: generate new subspace (local area) and search the new space. Search in the new space is realized by randomly generating a new solution (individual) located in this space. Regarding the generation of the new space, we consider two cases: (a) space search based on M selected solutions (denoted here as Case I), and (b) space search based on one selected solution (Case II).

In Case I, the new subspace (local area) is generated by M selected solutions (individuals). The core issue is how to determine the adjacent space based on the M solutions. For convenience, a solution X can be presented in another way $X=(x_1,x_2,......,x_n)$, where n is the index of the dimension. Regarding the M solutions, we use the following representations: $X^k=(x_1^k,x_2^k,......,x_n^k)$, $k=1,2,......,M$. To adjust the size of the new generating subspace, we use the coefficients $a_i\in[l,u]$ as parameters, where l and u are given numbers. Suppose that V is the new generating space, X^{new} is a new feasible solution randomly generated on a basis of V, where $X^{new}=(x_1^{new},x_2^{new},......,x_n^{new})$, S is the entire feasible solution space. The new space can be determined by the following expression:

$$V=\{X^{new}\;\Big|\; x_i^{new}=\sum_{k=1}^{M}a_i x_i^k \cup X^{new}\in S,\quad where\sum_{i=1}^{M}a_i=1, l\le a_i\le u\} \tag{14}$$

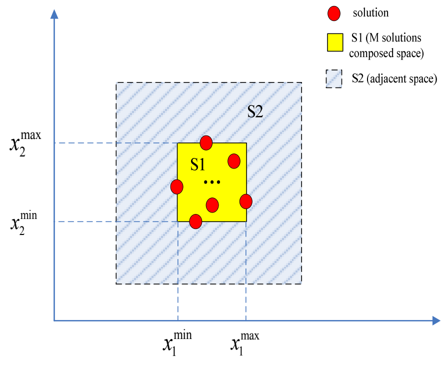


Fig. 1. Different spaces generated from M solutions using different parameters (Case I)

Figure 1 depicts different subspaces generated by M solutions using different parameters when the index of the dimension of a feasible solution is equal to 2. In this case, M solutions can presented as $X^k=(x_1^k,x_2^k)$, where $x_i^k\in[l_i,u_i], i=1,2$, $k=1,2,......,M$. The four points x_1^{min}, x_2^{min}, x_1^{max}, x_2^{max} are the minimum of x_1^k, the minimum of x_2^k, the maximum of x_1^k, the maximum of x_2^k, respectively. It is clear that the search space V is equal to S1 in case of $a_i\in[0,1]$. S2 is the search space of V in case of $a_i\in[l,u]$, where $l<0$, and $u>1$. In this study, we search the adjacent space S2 and set $a_i\in[-1,2]$. We generate the new space V_1 based on the following expression:

$$V_1=\{X^{new}\;\Big|\; x_i^{new}=\sum_{k=1}^{M}a_i x_i^k \cup X^{new}\in S,\quad where\sum_{i=1}^{M}a_i=1,-1\le a_i\le 2\} \tag{15}$$

In Case II, the space search operation is based on a given solution. In this case, the given solution is the best solution in the current solution set (population). The role of this operator is to adjust the best solution by searching its adjacent space. In the SSA, we generate the new space V_1 based on the following expression:

$$V_1=\{(x_1^{new},x_2^{new},......,x_n^{new})\,|\,x_j^{new}=x_j(j\ne i)\;\bigcup x_i^{new}\in[l_i,u_i]\} \tag{16}$$

where the value of x_i^{new} is the same as x_i which is range from l_i to u_i.

Suppose that the fitness value of a solution x is denoted by $f(x)$. Assume a function

$$better(x,y)=\begin{cases} true, & if\ f(x)\ge f(y),\\ false, & if\ f(x)<f(y).\end{cases} \tag{17}$$

where both x and y are the feasible solutions in the solution space. The overall algorithm can be outlined as the following sequence of steps.

Step 1. Initialize (randomly generate) solution set $P=(X^1,X^2,......,X^m)$, where $X^i\in S$.

Step 2. Evaluate each solution X^i, where $i=1,2,......,m$.

Step 3. Find the best solution x_{best} and the worst solution x_{worst} in the current solution set.

Step 4. If $better(x_{worst},x_{best})=true$, goto step 13.

Step 5. Randomly select M solutions from P, where M is a given number.

Step 6. Generate a new subspace V according to the M solutions (Case I).

Step 7. Generate a new solution x_{new} from the new subspace V.

Step 8. Update the current best and worst solutions in the following two cases:

(a) if $better(x_{new}, x_{worst}) = true$, set $x_{worst} = x_{new}$; and

(b) if $better(x_{new}, x_{best}) = true$, set $x_{best} = x_{new}$.

Step 9. Generate new subspaces V_1 based on the current best solution x_{best} (Case II).

Step 10. Generate a new solution x_{new1} from the subspace V_1.

Step 11. Update the current best and worst solutions in the following two cases:

(a) if $better(x_{new1}, x_{worst}) = true$,

set $x_{worst} = x_{new1}$; and

(b) if $better(x_{new1}, x_{best}) = true$, set $x_{best} = x_{new1}$.

Step 12. Repeat steps 4-11.

Step 13. Report the optimal solution.

The features of the SSA are highlighted as follows.

(1) The SSA leads to better performance when finding global optimization than PSO, especially in the optimization problems with larger solution spaces. The SSA searches the same size of solution space as PSO. However, the SSA searches the solutions based on the relative adequate analyzing space while PSO searches the solutions without such adequate analyzing space.

(2) The SSA leads to shorter computing time when being compared with the conventional PSO. Each solution is updated in PSO while SSA generates only two new solutions in each generation. That is in one generation, individuals which correspond to lots of new solutions are evaluated in PSO while only two new solutions (individual) are evaluated in the SSA. This operation procedure enables us to carry out the rapid CPU operation for hybrid identification of fuzzy systems.

3.3. Hybrid Optimization of ANFIS-based Fuzzy models

The standard gradient-based optimization techniques might not be effective in the context of rule based systems given their nonlinear character (in particular the form of the membership functions) and modularity of the systems. This suggests us to explore other optimization techniques. Figure 2 depicts the arrangement of chromosomes commonly used in fuzzy modeling [6, 13]. Genes for structure optimization are separated from genes used for parameter optimization. The size of the chromosomes for structure identification of the IG-based fuzzy model is determined according to the number of all input variables of the system. The size of the chromosomes for parameter identification depends on structurally optimized ANFIS-based fuzzy inference system.

The objective function (performance index) is a basic mechanism guiding the evolutionary search carried out in the solution space. The objective function includes both the training data and testing data and comes as a convex combination of the two components.

$$f(PI,\ E\_PI) = \theta \times PI + (1 - \theta) \times E\_PI \qquad (18)$$

Here, PI and E_PI denote the performance index for the training data and testing data, respectively. θ is a weighting factor that allows us to form a sound balance between the performance of the model for the training and testing data. Depending upon the values of the weighting factor, several specific cases of the objective function are worth distinguishing.

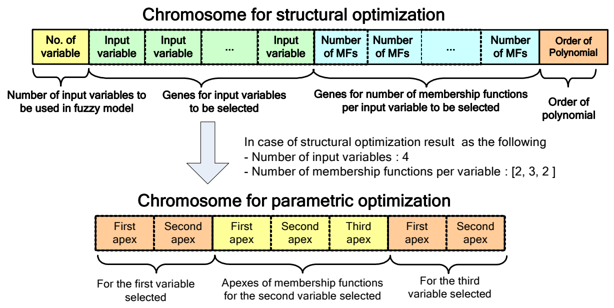

Chromosome for structural optimization
No. of variable
Input variable
Input variable
...
Input variable
Number of MFs
Number of MFs
...
Number of MFs
Order of Polynomial
Number of input variables to be used in fuzzy model
Genes for input variables to be selected
Genes for number of membership functions per input variable to be selected
Order of polynomial
In case of structural optimization result as the following
- Number of input variables : 4
- Number of membership functions per variable : [2, 3, 2]
Chromosome for parametric optimization
First apex
Second apex
First apex
Second apex
Third apex
First apex
Second apex
For the first variable selected
Apexes of membership functions for the second variable selected
For the third variable selected

Fig. 2. Arrangement of chromosomes for the optimization of fuzzy model

(i) If $\theta = 1$ then the model is optimized based on the training data. No testing data is taken into consideration.

(ii) If $\theta = 0.5$ then both the training and testing data are taken into account. Moreover it is assumed that they exhibit the same impact on the performance of the model.

(iii) The case $\theta = \alpha$ where $\alpha \in [0, 1]$ embraces both the cases stated above. The choice of α establishes a certain tradeoff between the approximation and generalization aspects of the fuzzy model.

4. EXPERIMENTAL STUDIES

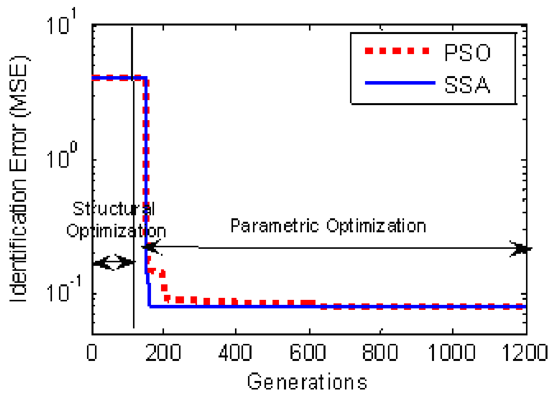
This section includes comprehensive numeric studies illustrating the design of the fuzzy model. We use three well-known data sets. PI denotes the performance index for training data and E_PI for testing data. The weighting factor $\theta = 0.5$ is taken into consideration. The numeric values of the parameters of the PSO were either predetermined or selected experimentally. More specifically, we used the following values of the parameters: maximum number of generations is 150; maximal velocity, v_{max}, is 20% of the range of the corresponding variables; w=0.4 and acceleration constants c_1 and c_2 are set to 2.0. The maximal velocity was set to 0.2 for the search carried out in the range of the unit interval [0,1]. The algorithm terminates after running 1000 generations. The parameters of SSA are as follows. We use 150 generations and a size of 100 populations (individuals) for structure identification and run the method for 1,000 generations. The population size is 60 for parameter identification. In each generation, we first search the space based on 8 solutions generated randomly and then search the space based on the best solution.

4.1. Non-linear function

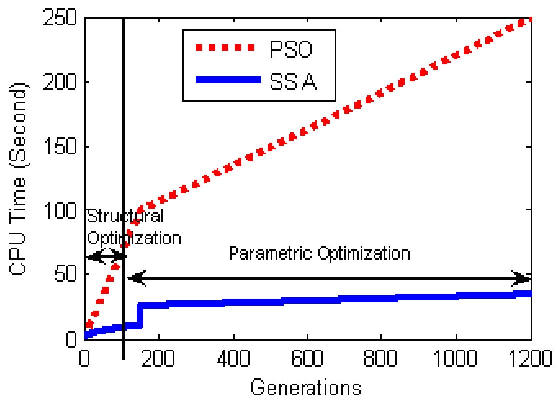
The three-input nonlinear function is given as

$$y = (1 + x_1^{0.5} + x_2^{1} + x_3^{-1.5})^2 \tag{19}$$

It is widely used to evaluate performance of various fuzzy models [14-17]. In this experiment, the data set is partitioned into two separate data sets. We use MSE defined by Eq.(5) as the performance index. The first 50% of data set (consisting of 20 pairs) is used for the design of the fuzzy model. The remaining 50% data set (consisting of 20 pairs) helps quantify the predictive quality of the model.



(a) Identification error in successive generations



(b) Computing time in successive generations

Fig. 3. Comparison of PSO with SSA (Non-linear function).

Figure 3 shows the optimization process of fuzzy model with eight fuzzy rules when running SSA and PSO, respectively. SSA and PSO exhibit the same identification error (performance index) in structural optimization, but SSA comes with the lower error than the one produced by the PSO in parametric optimization; see Figure 3(a). Moreover, Figure 3(b) shows that SSA uses less CPU time than PSO in each optimization phase.

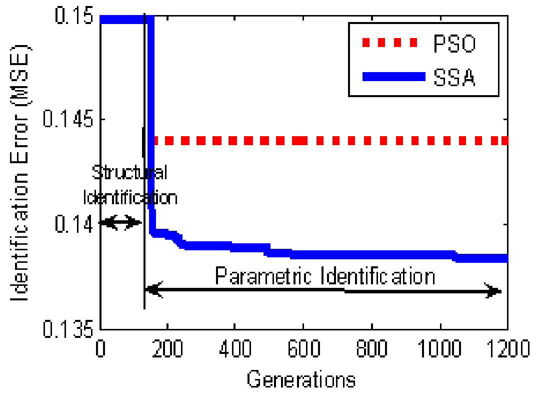
Table 1 supports a comparative analysis considering some existing models; it is evident that the proposed model compares favorably both in terms of accuracy and prediction capabilities. Notice that we compare with different types of model such as FNNs and GMDH, because there no previous fuzzy model for three-input nonlinear data.

Table 1. Comparison of identification errors for selected fuzzy models (Non-linear function)

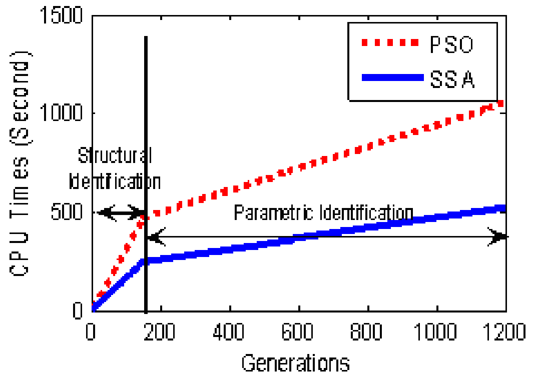
Model			PI	E_PI	No. of rules
Shinichi's model [14]		Type 1	0.84	1.22	
		Type 2	0.73	1.28	
Sugeno's model [15]		Model I	1.5	2.1	
		Model II	1.1	3.6	
Linear model [16]			12.7	11.1	
GMDH [16]			4.7	5.7	
Single-FNN [17]			2.670	3.063	
Oh et al.'s model [17]			0.174	0.689	
Our model	PSO+IG		0.000837	0.1590	8
	SSA+IG		0.000835	0.1564	8

4.2. Gas Furnace Process

The second well-known dataset is time series data of a gas furnace utilized by Box and Jenkins [2-6,13]. The time series data is comprised of 296 input-output pairs resulting from the gas furnace process has been intensively studied in the previous literature. The delayed terms of methane gas flow rate $u(t)$ and carbon dioxide density $y(t)$ are used as six input variables with vector formats such as $[u(t-3), u(t-2), u(t-1), y(t-3), y(t-2), y(t-1)]$. $y(t)$ is used as output variable. The first 148 pairs are used as the training data while the remaining 148 pairs are the testing data set for assessing the predictive performance. MSE is considered as a performance index.



(a) Identification error in successive generations



(b) Computing time in successive generations

Fig. 4. Comparison of PSO with SSA (GAS).

Table 2. Comparative analysis of selected models (GAS)

Model		PI_t	PI	E_PI	No. of rules
Pedrycz's model[2]		0.776			20
Tong's model[3]		0.469			19
Xu's model[4]		0.328			25
Sugeno's model[5]		0.355			6
Oh et al.'s model[6]	Simplified		0.024	0.328	4
	Linear		0.022	0.326	4
			0.021	0.364	6
HCM+GA [13]	Simplified		0.035	0.289	4
			0.022	0.333	6
	Linear		0.026	0.272	4
			0.020	0.264	6
Our model	PSO+IG		0.019	0.284	4
			0.015	0.273	6
	SSA+IG		0.017	0.266	4
			0.015	0.260	6

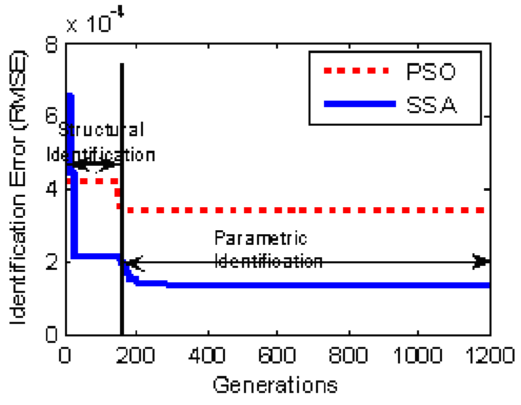
Figure 4 depicts the optimization process in SSA and PSO for the fuzzy model with six fuzzy rules. It shows that SSA has less identification error, less CPU time and rapid convergence in comparison with PSO. The identification error of the proposed model is compared with the performance of some other models; refer to Table 2. It is easy to see that the proposed model outperforms several previous fuzzy models known in the literature.

4.3. Chaotic Mackey-Glass Time Series

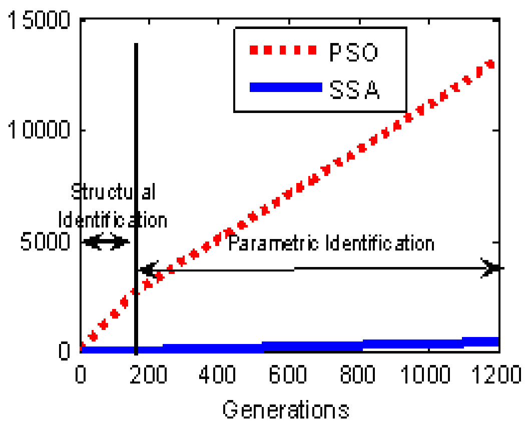
A chaotic time series is generated by the chaotic Mackey–Glass differential delay equation [18-23] of the form:

$$\dot{x}(t) = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{20}$$

The prediction of future values of this series arises is a benchmark problem that has been used and reported by a number of researchers. From the Mackey–Glass time series x(t), we extracted 1000 input–output data pairs for the type from the following the type of vector format such as: [x(t-30), x(t-24), x(t-18), x(t-12), x(t-6), x(t); x(t +6)] where t = 118–1117. The first 500 pairs were used as the training data set for IG-based FIS while the remaining 500 pairs were the testing data set for assessing the predictive performance. To come up with a quantitative evaluation of the fuzzy model, we use the standard RMSE performance index as like Eq. (5).



(a) Identification error in successive generations



(b) Computing time in successive generations

Fig. 5. Comparison of PSO with SSA (Mackey).

Figure 5 depicts the optimization process of fuzzy model with sixteen fuzzy rules when running SSA and PSO, respectively. It shows that SSA has better performance index, less Computing time and rapid convergence in comparison with PSO. Table 3 summarizes the results of comparative analysis of the proposed model with respect to other constructs. Here PI_t denotes the performance index for total process data, the non-dimensional error index (NDEI) is defined as the RMSE divided by the standard deviation of the target series.

Table 3. Comparative analysis of selected models (Mackey)

Model		PI_t	PI	E_PI	NDEI	No. of rules
Support vector regression model[18]			0.023	1.028	0.0246	
Multivariate adaptive regression splines [18]			0.019	0.316	0.0389	
Standard neural networks			0.018	0.411	0.0705	15 nodes
RBF neural networks			0.015	0.313	0.0172	15 nodes
Wang's model[19]		0.004				7
		0.013				23
ANFIS [20]			0.0016	0.0015	0.007	16
FNN model[21]			0.014	0.009		
Incremental type multilevel FRS [22]			0.0240	0.0253		25
Aggregated type multilevel FRS[22]			0.0267	0.0256		36
Hierarchical TS-FS[23]			0.0120	0.0129		28
Our model	PSO+IG		0.00346	0.00323	0.0157	8
			0.00033	0.00035	0.0057	16
	SSA+IG		0.00321	0.00302	0.0155	8
			0.00012	0.00015	0.0013	16

5. CONCLUDING REMARKS

This paper contributes to the research area of the hybrid optimization of ANFIS-based fuzzy models in the following two important aspects: 1) we proposed a space search algorithm. From the perspective of the size of the solution space, the SSA exhibits better performance in finding global optimization and less computing time than the conventional PSO; and 2) we introduced the hybrid optimization of ANFIS-based fuzzy models based on the SSA and information granulation. It is shown that the coding scheme introduced here leads to chromosomes which help decrease the number of unfeasible solutions arising in the process of evolutionary computing. Numerical experiments using three well-known data sets show that the model constructed with the aid of the SSA exhibits better performance in comparison with the PSO-constructed fuzzy model.

6. ACKNOWLEDGEMENTS

This work was partially supported by the GRRC program of Gyeonggi province [GRRC SUWON2010-B2, Center for U-city Security & Surveillance Technology], the Power Generation & Electricity Delivery the Korea Institute of Energy Technology Evaluation, the Planning (KETEP) grant funded by the Korean government Ministry of Knowledge Economy (2009T100100563), and the National Natural Science Foundation of China (Grant No. 60975050).

REFERENCES

[1] M.C. Nataraja, M.A. Jayaram, and C.N. Ravikumar, **"Prediction of Early Strength of Concrete: A Fuzzy Inference System Model,"** *International Journal of Physical Sciences*, Vol. 1, pp. 47-56, 2006.

[2] W. Pedrycz, **"An identification algorithm in fuzzy relational system,"** *Fuzzy Sets Syst.*, Vol. 13, pp. 153-167, 1984.

[3] R. M. Tong, **"The evaluation of fuzzy models derived from experimental data,"** *Fuzzy Sets Syst*., Vol. 13, pp. 1-12, 1980.

[4] C. W. Xu. and Y. Zailu, **"Fuzzy model identification self-learning for dynamic system,"** *IEEE Trans. on System, Man, and Cybernetics*, Vol. 17(4), pp. 683-689, 1987.

[5] M. Sugeno and T. Yasukawa, **"Linguistic modeling based on numerical data,"** *in IFSA'91 Brussels, Computer, Management & System Science*, pp. 264-267. 1991.

[6] S. K. Oh. and W. Pedrycz, **"Identification of Fuzzy Systems by means of an Auto-Tuning Algorithm and Its Application to Nonlinear Systems,"** *Fuzzy Sets and Syst*., Vol. 115(2), pp. 205-230, 2000.

[7] F. Liu, P. Lu, and R. Pei, **"A new fuzzy modeling and identification based on fast-cluster and genetic algorithm,"** *Intell. Contr. Automat.*, Vol. 1, pp. 290-293, 2004.

[8] W.Y Chung, W. Pedrycz, and E.T. Kim, **"A new two-phase approach to fuzzy modeling for nonlinear function approximation,"** *IEICE Trans. Info. Syst.*, Vol. 9, pp. 2473-2483, 2006

[9] Z.-L. Gaing, **"A particle swarm optimization approach for optimum design of PID controller in AVR system, "** *IEEE Trans. Energy Conversion*, Vol. 19, pp. 384-391, 2004.

[10] K.E. Parsopoulos and M.N. Vrahatis, **"On the computation of all global minimizers through particle swarm optimization,"** *IEEE Trans. Evolutionary Computation*, Vol. 8, pp. 211-224, 2004.

[11] M.A. Abido, **"Optimal design of power-system stabilizers using particle swarm optimization, "** *IEEE Trans. Energy Conversion*, Vol. 17, pp. 406-413, 2002.

[12] C.-L. Huang and C.-J. Wang, **"A GA-based feature selection and parameters optimization for support vector machines, "** *Expert Systems with Applications*, Vol. 31, pp. 231-240, 2006.

[13] B. J. Park., W. Pedrycz., and S. K. Oh, **"Identification of Fuzzy Models with the Aid of Evolutionary Data Granulation"**. *IEE Proc.-Control Theory and Applications*, Vol. 148, pp. 406-418, 2001

[14] S. Horikawa, T. Furuhashi and Y. Uchigawa, **"On fuzzy modeling using fuzzy neural networks with the back propagation algorithm,"** *IEEE Trans. Neural*

Networks, Vol. 3, pp. 801-806, 1992.

[15] G. Kang and M. Sugeno, **"Fuzzy modeling,"** *Trans. SICE*, Vol. 23, pp. 106-108, 1987.

[16] T. Kondo, **"Revised GMDH algorithm estimating degree of the complete polynomial,"** *Trans. Soc. Instrum. Control Eng.*, Vol. 22(9), pp. 928-934, 1986.

[17] S.K. Oh, W. Pedrycz, and H.S. Park, **"Rule-based multi-FNN identification with the aid of evolutionary fuzzy granulation,"** *Knowledge-Based systems*, Vol. 17, pp. 1-13, 2004.

[18] Krishnaiah, and P.R. Kanal, editors: *Classification, Pattern Recognition, and Reduction of Dimensionality. Handbook of Statistics*, Vol. 2, North-Holland, Amsterdam, 1982.

[19] L.X. Wang and M. Mendel: **"Generating fuzzy rules from numerical data with applications,"** *IEEE Trans. on System, Man, and Cybernetics,* Vol. 22, pp. 1414-1427 (1992)

[20] J.S.R. Jang, **"ANFIS: adaptive-network-based fuzzy inference system,"** *IEEE Trans. System Man Cybernet.*, Vol. 23(3), pp. 665-685, 1993.

[21] L.P. Maguire, B. Roche, T.M. McGinnity, and L.J. McDaid, **"Predicting a chaotic time series using a fuzzy neural, network,"** *Inform. Sci.*, Vol. 112, pp. 125-136, 1998.

[22] J.C. Duan and F.L. Chung, **"Multilevel fuzzy relational systems: structure and identification,"** *Soft Computing*, Vol. 6, pp. 71-86, 2002.

[23] Y.Chen, B. Yang, and A. Abraham, **"Automatic design of hierarchical Takagi-Sugeno type fuzzy systems using evolutionary algorithms,"** *IEEE Trans. Fuzzy Systems*, Vol. 15, pp. 385-397, 2007.