CHAPTER 7

Design of Hybrid Intelligent System for IDMT Relay

7.1 Introduction
7.2 Concept of Neuro - Genetic System
7.3 Neuro – Genetic Models
7.4 Resonance the Performance of IDMT Relay using Neuro-Genetic Models
7.5 Simulation Result
7.6 Discussion

"I too can see the stars on a desert night, and feel them. But do I see less or more? The vastness of the heavens stretches my imagination - stuck on this carousel my little eye can catch one-million-year-old light. A vast pattern - of which I am a part ---- What is the pattern, or the measuring, or the why? It does not do harm to the mystery to know a little about it. For a far more marvelous is the truth than any artists of the past imagined it."

Richard P. Feynman
7.1 Introduction

Research over the past two decades has shown that among all the available technologies none, is totally better than the other. Each has its own strength or weakness. One therefore needs to look for a hybridization of various technologies such that the whole offsets the weakness of the parts. For example classical AI models have a number of advantages that neural network do not have.

Hybrid system combines more than one intelligent paradigm in a synergistic frame work, during the past decades, we have increasingly seen neural network approaches being combined with existing computational techniques to produce a variety of applications. Neural networks are easily integrated into existing frameworks, such as pattern recognition, statistical inference and symbol processing.

Considerable research effort has been expended on bringing the gap between symbolic (Rule based ) AI systems and neural networks models. Hybrid symbolic – connectionist system employ neural network for low level information processing or as sub – system for adoption. Both paradigm have their own advantages and disadvantages and finding a suitable combination of both these technologies for a particular applications remains a difficult task.

More recently the soft computing paradigm, which includes neural networks, fuzzy logic system and genetic algorithm as its principal components, has found its way into numerous important commercial applications. Considerable stress is being laid on the seamless integration of these three technologies in order to exploit advantages of each in the design of intelligent system. For example various way of combining neural network with fuzzy systems are already discussed in our previous chapters (Chapter 6). In this chapter a hybrid system comprising of artificial neural network (ANN) and genetic algorithm is proposed for further study of behavior of IDMT relay. A Neuro – Genetic model is designed and the performance of IDMT relay is studied by developing reasoning.
7.2 Concept of Neuro – Genetic System

The neural networks can effectively be used for learning membership functions. An ANN is a computational structure that is inspired by observed process of biological neurons in the brain [111]. The basic capability of neural networks is to learn pattern for any specific objective (here fuzzy sensor) [30]. In general the learning can be supervised or unsupervised as discussed in section 2.7.2.

Genetic algorithms (GA) are unorthodox search or optimizations algorithm, which were first suggested by John Holland (University of Michigan, Ann Arbor ,1975). In general, Genetic Algorithms perform directed random searches through a given set of alternatives with the aim of finding the best alternative with respect to given criteria of goodness. These criteria are required to be expressed in terms of an objective function, which is usually referred to as a fitness function . It is further required that the alternatives be coded in strings of some specific finite length which consist of symbols from some finite alphabet. These strings are called chromosomes, the symbols that form them are called genes, and their set called a gene pool (Like universe of discourse in fuzzy set). The ANN training may be carried out by converting chromosomes into weight and genetic algorithm by weight into chromosomes .Fig. 7.1 shows generalized model of Neuro – Genetic system of IDMT relay.

In this research work the learning algorithm of ANN and optimization algorithm of genetic system are entangled together to form new Neuro- Genetic hybrid system to learn and simultaneously to optimize the fuzzy parameters of fuzzy sensor for IDMT relay . The following Neuro – Genetic system configurations have been suggested.
Class A: Neural Network + Genetic learning method.
Class B: Neural Network + Genetic chromosomes as weights.
Class C: Genetic algorithm + Training by neural network.

Fig. 7.2, 7.3 and 7.4 show the architecture of class A, class B and class C type Neuro-Geneic system respectively.

![Class A Neuro-Geneic system](image)

![Fig. 7.2 Class A Neuro - Genetics system](image)

![Class B Neuro-Geneic system (Superimposition)](image)

![Fig. 7.3 Class B Neuro - Genetic system (Superimposition)](image)

![Class C Neuro-Geneic system](image)

![Fig. 7.4 Class C Neuro - Genetic system](image)
7.3. Neuro - Genetic Models

Considering only supervised learning and special neural network, usually called multilayer feed forward networks or multiplayer perceptrons (MLP). The most common supervised learning algorithm is error back propagation learning algorithm (EBPA).

As we know that weighted sum in a neuron is

\[ \text{Net} = \sum_{i=1}^{n} W_i \cdot X_i \]  

...(7.3)

Where \( W_i \) is weight vector and \( X_i \) is input vector.

in terms of a non linear function, which is called activation function or threshold function. The sigmoidal function defined by the formulae [14].

\[ S_{\beta}(a) = \left(1 + e^{-\beta a}\right)^{-1} \]  

...(7.4)

where \( \beta \) is steepness parameter (positive constant) and \( a \) is activation input. The output of the neuron is defined by [14].

\[ y = S_{\beta}\left(\sum_{i=1}^{n} W_i \cdot X_i\right) \]  

...(7.5)

Considering the Neuro - Genetic configuration of Fig. 7.2 (i.e. class A), one can assume \( y \) as an initial population of chromosomes \( p^{(k)} \) of a given size \( m \) with \( k = 1 \), the iterative genetic algorithm may be entangled with the neural network equation 7.5 in following six steps for optimization of output [30].

1. Select initial population \( p^{(k)} \) from set of gene pool \( G_0 \).
2. Evaluate each chromosomes in population \( p^{(k)} \) in terms of fitness. This is done by determining for each chromosomes \( g \) in the population the value of the fitness function \( f(y) \).
3. Generate a new population \( p^{(k+1)} \) from the given population (input pattern of NN) \( p^{(k)} \) such that the value \( e(y) = mg(y) \) for each \( y \) in \( p^{(k)} \), where \( g(y) \) is a relative fitness given by -

\[ g(y) = \frac{f(y)}{\sum_{y \in p^{(k)}} f(y)} \]  

...(7.6)

The number of copies of each chromosomes \( y \) in \( p^{(k)} \) that is chosen for \( p^{(k)} \) is given by the integer part of \( e(y) \). If \( i < m \), then select remaining chromosomes by fraction part of \( e(x) \).

4. If stopping criteria are not met, go to step 5, otherwise stop.
5. Produce a population of new chromosomes \( p^{(k+1)} \) by operating on chromosomes i population \( p^{(k)} \), these operations are given in further steps.
6. Simple crossover: Given two chromosomes of output gene 'y' of NN.

\[ y_1 = (y_{11}, y_{12}, y_{13} \ldots y_{1n}) \]

97
and an integer \( i \in N_{n-1} \), which is called crossover position, the operation of simple crossover applied to \( y_1 \) and \( y_2 \) replaces these chromosomes with their offspring,
\[
y_1' = (y_{11}, \ldots, y_{1i}, y_{2i+1} \ldots, y_{2n}) \\
y_2' = (y_{21}, \ldots, y_{2i}, y_{1i+1} \ldots, y_{1n})
\]
...(7.8)

chromosomes \( y_1 \) and \( y_2 \), to which this operation is applied, are called mates.

7. **Double crossover**: For two crossover positions \( i, j \in N_{n-1} \) \((i < j)\), the operation of double crossover applied to \( y_1 \) and \( y_2 \) replaces these chromosomes with their offspring,
\[
y_1' = (y_{11}, \ldots, y_{1i}, y_{2i+1} \ldots, y_{2j}, y_{1j+1} \ldots, y_{1n}) \\
y_2' = (y_{21}, \ldots, y_{2i}, y_{1i+1} \ldots, y_{1j}, y_{2j+1} \ldots, y_{2n})
\]
...(7.9)

8. **Mutation**: With a given chromosome \( y_1 = (y_{11}, y_{12}, y_{13}, \ldots, y_{1n}) \) and an integer \( i \in N_n \), which is called a mutation position, the operation of mutation replaces \( y_1 \) with
\[
y_1' = (y_{11}, \ldots, y_{1i}, x, y_{i+1} \ldots, y_{1n})
\]
...(7.10)

where \( x \) is a randomly chosen gene from the gene pool \( G^n \).

9. **Inversion**: With a given chromosome \( y_1 = (y_{11}, y_{12}, y_{13}, \ldots, y_{1n}) \) and two integers \( i, j \in N_{n-1} \) \((i < j)\) which are called inversion positions, the operation of inversion replaces \( y_1 \) with
\[
y_1' = (y_{11}, \ldots, y_{1i}, y_{1j}, y_{i+1} \ldots, y_{1j-1} \ldots, y_{1n})
\]

10. **Replace population** \( p^{(k)} \) with population \( p^{(k+1)} \) produced in step 4, increase \( k \) by one and go to Step 2.

The similar model can also be developed for class B Neuro-Genetic systems (see Fig. 7.3) equation 7.5 for ANN by superimposing the weight chromosomes on ANN may be developed for Genetic Algorithm also. The weight values may be converted into binary string to form chromosome. From equation 7.5, for three chromosomes one may write.
\[
y_1 = S_\beta \left( \sum_{i=1}^{\sigma} (w_{i1} + w_{i2} + w_{i3} + \ldots + w_{in}) x_i \right)
\]
\[
y_2 = S_\beta \left( \sum_{i=1}^{\sigma} (w_{21} + w_{22} + w_{23} + \ldots + w_{2n}) x_i \right)
\]
\[
y_3 = S_\beta \left( \sum_{i=1}^{\sigma} (w_{31} + w_{32} + w_{33} + \ldots + w_{3n}) x_i \right)
\]
...(7.11)

The genetic iteration and neural iterations are assumed to be alternative, thus Neuro-Genetic iterations may be conducted in inter-iterative state. After a simple cross over of weight chromosomes the output gene of ANN and an integer \( i \in N_{n-1} \) called crossover position of weights, the genetic iteration in its turn applied to \( y_1, y_2, \) and \( y_3 \) replaces these chromosomes offspring (see Fig. 7.5).
\[ y_1 = S_\beta \left( \sum_{i=1}^{n} (w_{1i}, \ldots, w_{ni}) \right) \]
\[ y_2 = S_\beta \left( \sum_{i=1}^{n} (w_{2i}, \ldots, w_{ni}) \right) \]
\[ y_3 = S_\beta \left( \sum_{i=1}^{n} (w_{3i}, \ldots, w_{ni}) \right) \] ...(7.12)

The values of elements of gene pool \( G^n \) may be mapped from bit to decimal by the expression suggested here. This bit string may be mapped to the value of a parameter, say \( C_i, \) \( i=1,2,\ldots,n \) by the mapping [39,61]:

\[ C_i = C_{\text{min}} + \frac{b}{2^L-1} (C_{\text{max}} - C_{\text{min}}) \] ...(7.13)

Where 'b' is the number in decimal form that is being represented in binary form, \( L \) is the length of the bit string and \( C_{\text{max}} \) and \( C_{\text{min}} \) are user defined constant.

The process of generation of strings and their evaluation is continued until a convergence to the solution within a generation is obtained. The membership function or weight values are coded as bit strings that are than concatenated. A fitness function is used to evaluate the fitness of each set of membership function. The strings with highest fitness values are accepted. Let \( t_p \) be the target output and '\( y_k \)' be the output of Neuro-Genetic algorithm then the error function may be expressed by:

\[ E_p = \frac{1}{2} \sum_{k=1}^{m} \left( t_k^p - y_k^p \right)^2 \]

\[ = \frac{1}{2} \sum_{k=1}^{m} \left[ t_k^p - S_\beta \left( \sum_{i=0}^{n} w_k x_i^p \right) \right]^2 \] ...(7.14)

**Fig. 7.5 Simple crossover in class B model**
This $E_p$ must be less than or equal to the maximum acceptable error (i.e. $E_p \leq E_{\text{max}}$). If $E_p > E_{\text{max}}$ then a new cycle is initiated using back propagation learning algorithm.

The mathematical model for class C Neuro - Genetic system (see Fig. 7.4) be derived from the previous algorithms. The gene pool $G^n$ consists of fuzzy universe of discourse of antecedent values of the system. The membership functions with the highest fitness function of a best generation of a population $p^{(k)}$ are used as an input pattern of neural network to obtained a precise membership function and learning of NN is achieved simultaneously [30,61]. The two chromosomes of input gene are given by:

\[ x_1 = (x_{11}, x_{12}, \ldots x_{1n}) \]
\[ x_2 = (x_{21}, x_{22}, \ldots x_{2n}) \]

An integer $i \in N_{n-1}$, which is crossover position, is operated to replace these chromosomes with their offspring as given in equation 7.7.

\[ x_1' = (x_{n1}, \ldots, x_{1i}, x_{2i+1} \ldots x_{2n}) \]
\[ x_2' = (x_{21}, \ldots, x_{2i}, x_{1i+1} \ldots x_{1n}) \]

For an integer $i \in N_n$, which is called a mutation position, the operation of mutation replaces $x_i$ with

\[ x_i' = (x_{i1}, \ldots, x_{i1-1}, y_i, x_{i1+1} \ldots x_{1n}) \]

After a final search at a convergent point the membership functions of a fuzzy set with highest fit value is obtained, as given below

\[ x_p' = (x_1, x_2, \ldots x_n) \]

The gene population of highest fitness value obtained from equation 7.18 is used as an input pattern to the neural network to train it, the output of NN would be

\[ y_p = \sum_{i=0}^{n} w_p x_p \]

7.4 Reasoning the Performance of IDMT Relay Using Neuro - Genetic Models

Plug setting current ($I_{pi}$) and operating time ($O_t$) obtained after fuzzification of characteristics curve of IDMT relay (see Fig. 4.5) in chapter 4 is considered again for reasoning the performance of IDMT relay using Neuro - Genetic model. The performance of IDMT relay reasoned by or learnt by the Neuro - Genetic models of class A, class B and class C have compared for study. First, class A is considered for reasoning. Class A system is combination of NN and genetic algorithm. The pattern of training data set (see Table 4.2) are given below [61,20].
The NN training results are tabulated in Table 7.2. Further the genetic algorithm may be applied using equation 7.6 to equation 7.11 for class A system. The performance of IDMT relay is to be defined in plug setting current interval (2-20). The interval may be approximated by 20 integer point 0,1,2,...20 and coded these point by the corresponding binary numbers. Then G = (0,1) and possible chromosomes are binary integers. Assuming m=3 and p(1) = (00010,01001,10011) in step 1 in Table 7.1.

**Table 7.1 Performance evaluation by Genetic algorithm for IDMT relay**

(a) k=1, m=3, step 2 and 3

<table>
<thead>
<tr>
<th>Chromosomes in ( p_n^{(1)} )</th>
<th>Integers</th>
<th>Fitness</th>
<th>( G^{(p_1)} )</th>
<th>4g ( (I_p) )</th>
<th>Numbers of selected copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>00010</td>
<td>2</td>
<td>3.85</td>
<td>0.069</td>
<td>0.276</td>
<td>0</td>
</tr>
<tr>
<td>01001</td>
<td>9</td>
<td>12.98</td>
<td>0.295</td>
<td>1.180</td>
<td>1</td>
</tr>
<tr>
<td>10011</td>
<td>19</td>
<td>15.94</td>
<td>0.356</td>
<td>1.424</td>
<td>2</td>
</tr>
</tbody>
</table>

(b) k=1, step 5

<table>
<thead>
<tr>
<th>Chromosomes in ( p_n^{(1)} )</th>
<th>Mate (randomly selected)</th>
<th>Cross over site (randomly selected)</th>
<th>Resulting chromosomes in ( p^{(2)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>01001</td>
<td>10011</td>
<td>3</td>
<td>01011</td>
</tr>
<tr>
<td>10011</td>
<td>01001</td>
<td>3</td>
<td>10001</td>
</tr>
<tr>
<td>10011</td>
<td>11000</td>
<td>1</td>
<td>11000</td>
</tr>
</tbody>
</table>
Using $f(I_p)$ as the fitness function, the fitness of each chromosomes in $p^{(1)}$ (Step 2) are calculated. Then using the deterministic sampling in step 3, the new population $p^{(1)}_n = (01001, 10011, 10011)$ are obtained as shown in Table 7.1(b). If given stopping criteria in step 4 are not met, proceed to step 5. Assuming that the condition $p^{(k)}_a = p^{(k)}$ is chosen as the stopping criterion, the algorithm does not stop at this point and proceed to stops. In step 5 simple crossovers are used, each of which produces one of the two possible offsprings. For each $I_p$ in $p^{(1)}_n$, a mate $I_p$ in $p^{(1)}_n$ and a crossover point are chosen randomly and, then, the offspring $I_p$ is produced (Table 7.1(b)), next in step 10, the old population $p^{(1)}_n$ is replaced with new population $p^{(2)}$ of offsprings produced in step 5, $k$ is increased by one and proceed to step 2. Step 2 and 3 are now repeated for $k=2$...
and results are shown in Table 7.1(c). The stopping criterion in step 4 is again not satisfied, consequently, proceed to step 5. The result of this step is shown in Table 7.1(d). In step 10, \( p_n^{(2)} \) is replaced with \( p_n^{(3)} \), increase \( k \) by one, and proceed to step 2. The application of step 2 and 3 for \( k=3 \) result in \( p_n^{(3)} \), shown in Table 7.1(c). Now the stopping criterion \( p_n^{(3)} = p_n^{(3)} \) is satisfied in step 4, and algorithm terminates. The genetic algorithm is represented for each consequent variables also by converting in milliseconds (i.e. operating time in \( m \) seconds).

The class B and class C systems are also simulated and a comparative simulation result has been shown in Table 7.2.

### 7.5 Simulation Result

The performance of Neuro–Genetic model of class A, Class B and class C system have been tested. Simulation results are scheduled in Table 7.2. The antecedent and consequent variables i.e. Plug setting currents \( I_p \) and operating times \( O_t \) respectively are used as input/output pattern for NN and gene pool for generic algorithm. A comparative study of performance of class A, class B and class C Neuro-Genetic system for reasoning the performance characteristics of IDMT relay shows that three systems have almost similar behavior. This conceptual piece of research work is completed to provide the information about the implementation of Neuro-Genetic theories for protective relays used in power system protection or power utility company.

### 7.6 Discussion

There is a well stabilized theory for protective relays. Performance characteristic of a protective relay of IDMT type has been fuzzified. The empirical reasoning by neural network and optimum reasoning by genetic algorithm are employed in system designing. The NN and Genetic system have been continued together to form three classes of Neuro–Genetic (NG) system which provides very rich theory of neural computing and its industrial application. The simulation results of the proposed system have been tabulated in Table 7.2. Data fro plug setting current and operating time for three classes A, B and C are obtained by simple calculation using the steps of GA. This concept of hybrid intelligent system can be implemented in power protection system. This piece of work can also be expended for simulating and modeling using software like MATLAB.
Table 7.2 The comparison of antecedent/consequent variables of Neuro-Genetic system for Class A, class B and class C

<table>
<thead>
<tr>
<th>S. NO.</th>
<th>Antecedent Variable of IDMT relay</th>
<th>Consequent Variable of IDMT relay</th>
<th>Class A NG SYSTEM</th>
<th>Class B NG SYSTEM</th>
<th>Class C NG SYSTEM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>I_p in Amp.</td>
<td>O_t in Second</td>
<td>I_p in Amp.</td>
<td>O_t in Second</td>
<td>I_p in Amp.</td>
</tr>
<tr>
<td>1</td>
<td>2.5</td>
<td>0.78</td>
<td>2.39734</td>
<td>0.773</td>
<td>2.38735</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0.65</td>
<td>3.10373</td>
<td>0.652</td>
<td>3.100021</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>0.58</td>
<td>3.98843</td>
<td>0.579</td>
<td>3.99388</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>0.52</td>
<td>5.00313</td>
<td>0.519</td>
<td>4.99937</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>0.48</td>
<td>6.10010</td>
<td>0.49</td>
<td>5.99897</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>0.46</td>
<td>6.99873</td>
<td>0.453</td>
<td>7.00321</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>0.44</td>
<td>7.98792</td>
<td>0.439</td>
<td>8.00012</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>0.41</td>
<td>9.10301</td>
<td>0.400</td>
<td>9.00307</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>0.40</td>
<td>10.01031</td>
<td>0.401</td>
<td>10.020111</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>0.35</td>
<td>14.99813</td>
<td>0.360</td>
<td>15.10001</td>
</tr>
<tr>
<td>11</td>
<td>17.5</td>
<td>0.33</td>
<td>17.53103</td>
<td>0.321</td>
<td>17.49983</td>
</tr>
<tr>
<td>12</td>
<td>20</td>
<td>0.32</td>
<td>19.99981</td>
<td>0.319</td>
<td>20.00013</td>
</tr>
</tbody>
</table>