Intelligent Techniques for Home Electric Load Forecasting and Balancing

Submitted by

LING SAI HO, BEng(Hons)

For the Degree of Master of Philosophy in the Department of Electronic and Information Engineering at the Hong Kong Polytechnic University

October 2002

ABSTRACT

The work presented in this thesis is related to an intelligent power system in a modern home. Results in the following areas will be reported: a power line based data network infrastructure with hardware modules for an intelligent home, a home electric load forecasting system, and a home electric load balancing system.

A power line data network based on the spread-spectrum technology is proposed and implemented. It facilitates digital data communications at a rate of 10 Kbps in the noisy and signal-distorting environment of the AC power line. This power line data network serves as a backbone of communication in an intelligent home through which electrical appliances can be controlled via line/mobile phones, personal digital assistants (PDAs), keypads or personal computers anytime and anywhere, inside or outside the home. It can provide a basis for the plug-and-play features of electrical appliances without the need of installing additional cables.

Short-term electric load forecasting (STELF) is essential to improve the reliability of the AC power line data network and provide optimal load scheduling in an intelligent home. Three computational intelligence techniques are developed to realize STELF. The first approach is by using a fuzzy genetic algorithm (GA)-based neural network (NN). It can forecast the electric load accurately with respect to different day types and weather information. The proposed fuzzy GA is modified from a published GA with arithmetic crossover and non-uniform mutation. Fuzzy logic is used to incorporate expert knowledge and experience (in terms of linguistic rules) into the crossover and mutation operations. With fuzzy logic, the number of iteration and the rate of change of fitness value can be fine-tuned. By using some benchmark test functions, it can be shown that the fuzzy GA performs better than the traditional GA. In many applications of NNs, the networks are fully connected. However, the performance of a fully connected NN may not be better than that of a partly connected NN with the same number of hidden nodes. This is because some links in an NN could be redundant. A three-layer NN with a switch introduced to each link is proposed to facilitate the tuning of the network structure. By turning on or off these link-switches during the training process, the optimal neural network structure can be obtained. This implies that the cost of implementing the proposed NN in terms of hardware, processing and simulation time can be reduced.

The second approach for realizing STELF involves a fuzzy GA-based neural fuzzy network (NFN). The optimal NFN structure can be found by the fuzzy GA when switches in the links of the network are introduced. The membership functions and the number of rules of the NFN can be generated automatically. Results for implementing STELF in an intelligent home by using the proposed NFN will be given.

The third approach for realizing STELF is based on a modified fuzzy GA-based neural network, which involves a new neuron model. Under this model, the neuron has two activation transfer functions and exhibits a node-to-node relationship within the hidden layer. The proposed neural network can offer a better performance and a smaller number of hidden nodes than the traditional feed-forward neural network. This network is trained by the fuzzy GA.

The electric load forecasting system is further applied in a home electric load balancing system. After an electric load forecasting system can successfully forecast the power consumption profile of a home, the load balancing system can adjust the amount of energy stored in batteries accordingly and prevent it from reaching some practical limits. A steady consumption pattern can then be obtained which will benefit both the power users and the utility company. An example will be given to illustrate the accuracy of the forecaster, and its ability of achieving load balancing.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my chief supervisor, Dr Frank H. Leung for his guidance, support and valuable advice. Thanks are extended to my second supervisor, Dr Peter K.S. Tam for his motivations and helpful advice. I would also like to thank Dr H.K. Lam, Dr L.K. Wong, Mr T.H. Lee and Mr K.F. Leung for their continual encouragement. Last but not least, I truly appreciate the help from the staff in the General Office of the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University.

The work described in this thesis was substantially supported by a grant from the Hong Kong Polytechnic University of the Hong Kong Special Administrative Region (Project no. G-V954).

STATEMENT OF ORIGINALITY

The following contributions reported in this thesis are claimed to be original.

1. *An AC power line data network for an intelligent home is implemented (Chapter 3.)* With this home system, any alteration or expansion imposes no extra cost on the installation of network cables. The AC power line data network is based on the spread-spectrum technology and is applied as a wired network in an intelligent home system.
2. *A fuzzy genetic algorithm is proposed (Chapter 4, Section 4.2.)* This algorithm is modified from the published GA with arithmetic crossover and non-uniform mutation. By implementing fuzzy logic in the genetic operations, the proposed algorithm performs better and converges faster than the tradition GA.
3. *A fuzzy GA-based neural network with link switches is proposed (Chapter 4, Section 4.3.)* By introducing link switches to the neural network, the optimal network structure can be obtained. This implies that the cost of implementing the proposed neural network can be reduced.
4. *The application of the fuzzy GA-based neural network to short-term electric load forecasting is presented (Chapter 4, Section 4.3.2.)* A short-term electric load forecasting system has been realized by the fuzzy GA-based neural network. It can forecast the electric load accurately with respect to different day types and weather information.
5. *A fuzzy GA-based neural fuzzy network with rule switches is proposed (Chapter 4, Section 4.4.)* The optimal NFN structure can be found by the fuzzy GA when rule

switches are introduced. The membership functions and the number of rules can be generated automatically.

6. *The application of the fuzzy GA-based neural fuzzy network to short-term electric load forecasting is presented (Chapter 4, Section 4.4.2.)*
7. *A modified fuzzy GA-based neural network with two activation transfer functions in the neuron is proposed (Chapter 4, Section 4.5)* A new neuron model is introduced. The proposed modified network can offer a better performance and a smaller number of hidden nodes.
8. *The application of the modified fuzzy GA-based neural network to short-term electric load forecasting is presented (Chapter 4, Section 4.5.3.)*
9. *A home electric load balancing system is proposed (Chapter 5.)* The electric load balancing system can adjust the amount of energy stored in batteries accordingly and prevent it from reaching some practical limits. A relatively steady consumption pattern can then be obtained which will benefit both the power users and the utility company.

TABLE OF CONTENTS

AUTHOR'S PUBLICATIONS

INTERNATIONAL JOURNAL PAPERS

[1] F.H.F. Leung, H.K. Lam, S.H. Ling and P.K.S. Tam, "Tuning of the Structure and Parameters of Neural Network using an Improved Genetic Algorithm," *IEEE Trans. Neural Network* (accepted).

[2] S.H. Ling, H.K. Lam, F.H.F. Leung and P.K.S. Tam, "A Novel GA-Based Neural Network for Short-Term Load Forecasting," *IEEE Trans. Industrial Electronics* (under revision).

[3] S.H. Ling, H.K. Lam, F.H.F Leung and P.K.S. Tam, "Short-Term Electric Load Forecasting based on a Neural Fuzzy Network," *IEEE Trans. Industrial Electronics* (under review).

[4] S.H. Ling, F.H.F Leung and P.K.S. Tam, "Daily Load Forecasting with a Fuzzy-Neural Network in an Intelligent Home," *International Journal of Approximate Reasoning* (under review).

[5] L.K. Wong, S.H. Ling, F.H.F. Leung and Y.S. Lee, "Design and control of a home electrical load balancing system," *IEEE Trans. Industrial Electronics* (under review).

INTERNATIONAL CONFERENCE PAPERS

[1] S.H. Ling, H.K. Lam, F.H.F. Leung and P.K.S. Tam, "Parameter Learning of Neural Network Using Fuzzy Genetic Algorithm," *Congress on Evolutionary Computation (CEC2002), World Congress on Computational Intelligence (WCCI 2002)*, Honolulu, Hawaii, May 2002, pp. 1928-1933.

[2] S.H. Ling, F.H.F. Leung, H.K. Lam and P.K.S. Tam, "Short-Term Daily Load Forecasting in an Intelligent Home with GA-Based Neural Network," *2002 International Joint Conference on Neural Networks (IJCNN2002), World Congress on Computational Intelligence (WCCI 2002)*, Honolulu, Hawaii, May 2002, pp. 997-1001.

[3] S.H. Ling, H.K. Lam, F.H.F. Leung and P.K.S. Tam, "A Novel GA-Based Neural Network for Short-Term Load Forecasting," *2002 International Joint Conference on Neural Networks (IJCNN2002), World Congress on Computational Intelligence (WCCI 2002)*, Honolulu, Hawaii, May 2002, pp. 2761-2766.

[4] S.H. Ling, F.H.F Leung and P.K.S. Tam, "Daily Load Forecasting with a Fuzzy-input-Neural Network in an Intelligent Home," in *Proc. 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001*, Australia, Dec 2001, pp. 449-452.

[5] S.H. Ling, H.K. Lam, F.H.F Leung and P.K.S. Tam, "A Neural Fuzzy Network with Optimal Number of Rules for Short-Term Load Forecasting in an Intelligent Home," in *Proc. 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001*, Australia, Dec 2001, pp. 1456-1459.

[6] H.K. Lam, S.H. Ling, F.H.F. Leung and P.K.S. Tam, "Tuning of the Structure and Parameters of Neural Network using an Improved Genetic Algorithm," in *Proc. 27th Annual Conference of the IEEE Industrial Electronics Society, IECON'2001,* USA, Nov 2001, pp. 25-30.

[7] L.K. Wong, S.H. Ling, F.H.F. Leung, Y.S. Lee, S.H. Wong, T.H. Lee, H.K. Lam, K.H. Ho, D.P.K. Lun, T.C. Hsung, "An intelligent home," in *Proc. Workshop on Service Automation and Robotics,* Hong Kong, June 2000, pp. 111-119.

[8] H.K. Lam, K.F. Leung, S.H. Ling, F.H.F. Leung and P.K.S. Tam, "On Interpretation of Graffiti Digits and Commands for eBooks: Neural-Fuzzy Network and Genetic Algorithm Approach," *2002 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2002, World Congress on Computational Intelligence (WCCI 2002),* Honolulu, Hawaii, May 2002, pp. 443-448.

[9] H.K. Lam, S.H. Ling, K.F. Leung, F.H.F. Leung and P.K.S. Tam, "On interpreting Graffiti Commands for eBooks using a Neural Network and an Improved Genetic Algorithm," in *Proc. 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001,* Australia, Dec 2001, pp. 1464-1467.

[10] H.K. Lam, S.H. Ling, F.H.F. Leung and P.K.S. Tam, "Optimal and Stable Fuzzy Controllers for nonlinear Systems subject to Parameter Uncertainties using Genetic Algorithm," in *Proc. 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001,* Australia, Dec 2001, pp. 908-911.

LIST OF SYMBOLS

b	Shape parameter of non-uniform mutation
b_j^1, b_k^2	Biases for neural network
C_{upperl}	Battery storage capacity
$g(\cdot)$, $q(\cdot)$	Unknown nonlinear functions
K	Scaling factor
κ_j	Output of SATF
$logsig(\cdot)$	Logarithmic sigmoid function
$L^d(\cdot), L(\cdot)$	Electric loads
M_j^i	Fuzzy term
n_{in}	Number of input
n_h	Number of hidden node
n_{out}	Number of output
$net_s^j(\cdot)$	Static activation transfer function
$net_d^j(\cdot)$	Dynamic activation transfer function
$net_o^l(\cdot)$	Activation transfer function of the output neuron
$\mathbf{o}_s$	Offspring
P	Set of population
p_m	Probability of fuzzy mutation
$\mathbf{p}_i$	Chromosome
r_r	Probability of acceptance
s_j^1, s_k^2, s_{ij}^1, s_{jk}^2	Link switches of the neural network
T	Total iteration
t	Time
τ	Current iteration
v_{ij}, w_{jk}	Weights of neural network
w_a, w_{m_t}	Weighting factors for fuzzy GA operator
$y(t), \mathbf{y}(t)$	Output, output vector
$z(t), \mathbf{z}(t)$	Input, input vector
$\mu_{\mathrm{M}_j^i}(\cdot)$	Membership function
$\delta(\cdot)$	Unit step function
ς_g	Rule switch of neural fuzzy network
ζ_j	Output of modified neuron

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

CEBus	Consumer Electronics Bus
DATF	Dynamic Activation Transfer Function
FAM	Fuzzy Associative Memory
MAPE	Mean Absolute Percentage Error
MIMO	Multi-Input Multi-Output
NFN	Neural Fuzzy Network
GA	Genetic Algorithm
NN	Neural Network
SATF	Static Activation Transfer Function
STELF	Short-Term Electric Load Forecasting

CHAPTER ONE

INTRODUCTION

In an information age, homes should take full advantage of modern technology and offer intelligent features in order to enhance the comfort and security of their residents. A data network for the communication between the home users and the home should be present. In particular, the AC power line has gradually been accepted as the backbone of the home network thanks to its low cost and readiness of interfacing with domestic appliances. To enhance the reliability of AC power line data network, an accurate short-term electric load forecasting should be realized. Furthermore, the short-term electric load forecasting can form the basis for the development of an intelligent load balancing system. At present, the peak demand of electricity are generally met by operating costly auxiliary generators, or by purchasing power from other utility companies. The cost for supplying peak power is therefore much higher than that for supplying the average power. A reduction in the peak value of electricity demand can be achieved if we can realize load forecasting, and schedule the demands on the utility company accordingly. This has to be supported by batteries installed in the intelligent home that are responsible for balancing the load demand.

Short-term electric load forecasting and balancing systems exhibit non-linear, non-stationary, and non-Gaussian characteristics that are difficult to model and forecast using traditional mathematical methods. Computational Intelligence (CI) [Pedrycz 97] techniques are known to be capable of solving complex and non-linear problems. They are proposed in this thesis to realize short-term electric load forecasting and balancing.

The main components of CI encompass fuzzy logic [Zadeh 65], neural network [Haykin 94] and genetic algorithm [Michalewicz 94].

The aims of this thesis are to report home electric load forecasting systems and a load balancing system based on CI techniques. An AC power line data network infrastructure for an intelligent home will be discussed. Three different CI approaches for short-term home electric load forecasting are proposed. Based on one of the home electric load forecasting systems, a home electric load balancing system will be designed and presented. The achievements reported in this thesis are summarized as follows:

1.1 POWER LINE DATA NETWORK INFRASTRUCTURE

One of the requirements for realizing an intelligent home system is the establishment of a communication channel [Ferrerira 96, Amitava 99] among home appliances and users. Without having to consult the manufacturers of electrical appliances and install a LAN, one simple way to realize this communication channel is to make use of the AC power line. However, the electric power line at home has many appliances connected to it, and each appliance has different characteristics that affect the power line conditions. When using an AC power line as a networking medium, one has to deal with problems such as electromagnetic interference, varying impedance, narrow frequency impairments (owing to noise), and signal attenuation. On the other hand, video or voice signals that require a high data rate may be transmitted in an intelligent home. These bring difficulties to the design problem of an intelligent home system based on a power line data network.

In this thesis, we propose a power line data network based on the spread-spectrum technology [Radford 96], which facilitates communications at 10 Kbps in the noisy and signal-distorting environment of the AC power line. This power line network serves as a backbone of communication in an intelligent home through which electrical appliances can be controlled via line/mobile phones, PDAs, keypads or personal computers anytime and anywhere, inside or outside the home. The power line data network can provide a basis for the plug-and-play features of electrical appliances without the need of installing additional cables. Various sensors are employed to monitor the home's real time conditions in order to enhance its security and comfort. Details about the intelligent home system will be given in Chapter 3.

1.2 ELECTRIC LOAD FORECASTING SYSTEM

Three intelligent techniques for short-term home electric load forecasting are proposed in this thesis. First, a fuzzy genetic algorithm (GA)-based neural network is proposed. A three-layer neural network with a switch introduced in each link is proposed to facilitate the tuning of the optimal network structure. A proposed fuzzy GA is used to help tuning the structure as well as the parameters of the proposed neural network. Such a fuzzy GA is modified from the published GA with arithmetic crossover and non-uniform mutation operations [Michalewicz 94]. Fuzzy logic [Zadeh 65] is good in representing expert knowledge and experience with some linguistic rules, which can be easily understood by human beings. In the fuzzy GA, a fuzzy crossover is used to exchange information in two selected parents. The offspring to be generated is governed by some fuzzy rules. The rules of this fuzzy crossover should be set such that the offspring will look closer to the parent with a larger fitness value. On applying the traditional GA, the non-uniform mutation operation on the chromosomes is only governed by the iteration number. In the proposed fuzzy mutation, the operation is not only governed by the iteration number but also the rate of change of the fitness value with respect to the mutated gene of the offspring. Consequently, human knowledge on the crossover and mutation can be incorporated using fuzzy rules. It will be shown that the proposed fuzzy GA is more efficient and provides a faster convergence than the traditional GA in some benchmark test functions [De Jong 75, Yao 99 and Araujo 00].

Second, a short-term electric load forecasting realized by a fuzzy GA-based neural fuzzy network (NFN) is proposed. In this thesis, a neural fuzzy network (NFN) with switches is proposed. By using the proposed fuzzy GA, the optimal number of fuzzy rules can be found. This implies a lower cost of implementing the proposed NFN.

Third, a modified fuzzy GA-based neural network is proposed for short-term electric load forecasting. In this modified neural network, two different activation transfer functions are used in the neuron and a node-to-node relationship is proposed in the hidden layer. This network model is found to be able to give a better performance with a smaller number of hidden nodes than the traditional feed-forward neural network [Bryson 69 and Lecun 85]. The proposed fuzzy GA is used to tune the parameters of the neural network. Details about the computational intelligence techniques, the electric load forecasting systems, and the simulation results will be given in Chapter 4. A comparison for the three approaches will also be given.

1.3 ELECTRIC LOAD BALANCING SYSTEM

A home electrical load balancing system based on the proposed fuzzy GA-based neural network forecaster will be reported. It is realized with the installation of batteries for energy storage at home. In this system, the power supplied to the home is regulated to a predicted reference value. If the home needs more power than the reference, the batteries will discharge to provide the extra power. If the home draws less power than the reference, the energy in excess will be used to charge the batteries. As a result, the power drawn from the mains will approach a constant despite the presence of large fluctuations in the power consumption. The proposed system not only solves the overloading problem, but also helps power utility companies reduce the budget for peak-hour power generation. To make the system work probably, the capacity of the batteries should be high enough so that they are not fully charged or discharged during the operation. In this thesis, the proposed short-term electric load forecasting system forecasts the loading of the home and suggests a suitable reference amount of energy to be stored in the batteries. The system will spare some control power to regulate the amount of energy stored in the batteries. By doing so, the performance of the home load balancing is sacrificed a bit for the system reliability. Details about the electric load balancing system will be given in Chapter 5. A conclusion to the whole thesis will be given in Chapter 6. The achievement will be summarized and the direction for further development will be discussed.

CHAPTER TWO

LITERATURE REVIEW

A review on intelligent home, computational intelligence techniques, home load forecasting and load balancing will be given in this chapter. The advantages and disadvantages of them will be discussed.

2.1 INTELLIGENT HOME

At present, many researchers and companies are developing intelligent home systems. Some researchers have designed a phone-based remote controller through which home users can issue control commands to their home appliances using a telephone [Wong 94]. In UK, a small two-arm mobile robot in an intelligent home can be controlled via an ISDN link [Gray 96]. Existing intelligent home systems usually are implemented with a wired local area network (LAN). This involves extra cabling, and the systems have to be well designed before it is built. Moreover, additions and removals of network components might entail costly re-cabling. In the U.S., X-10 systems [Ferrerira 96] are commonly used in intelligent home systems. The AC power line is used as the medium for data transmission. As AC power lines exist in nearly every corner of a house, the installation cost for the data network can be significantly reduced. However, the data transmission rate of the X-10 system is very low (typically 60 bps).

2.2 COMPUTATIONAL INTELLIGENCE TECHNOLOGY

The main components of computational intelligence technology encompass fuzzy logic, neural network and genetic algorithm.

2.2.1 Fuzzy Logic

Since the initial work by Zadeh [Zadeh 65] and Gogien [Gogien 67], many theoretical advances on fuzzy set theory have been made in many areas. These include fuzzy algebra, fuzzy subset, fuzzy logic and reasoning, fuzzy inference, fuzzy relation and equation, fuzzy number, fuzzy computing theory [Nauck 97, Jang 97, Wang 97 and Kaimal 97] etc. Fuzzy logic offers a paradigm for representing and processing linguistic or non-numeric information. It is a logic system that is much closer in spirit to human thinking and natural language than the traditional logic systems [Lee 90]. By processing fuzzy information, reasoning with respect to a linguistic knowledge base can be done. These features have made fuzzy logic useful for dealing with complicated decision-making problems with multiple objectives. A typical fuzzy rule has the following format:

Rule j: IF x_1 is N_1^j AND x_2 is N_2^j ... AND x_n is N_n^j

$$\text{THEN}\, y = w_j, j = 1, 2, \ldots, r_m \tag{2.1}$$

where x_i (i = 1, 2, ..., n) are the input variables to the fuzzy system, y is the output variable of the fuzzy system, N_n^j is a fuzzy term of rule j, j = 1, 2, ..., r_m; r_m denotes the number of rules, w_j is a singleton output.

$$\sum_{j=1}^{r_m} m_j = 1,\ m_j \in [0,\ \ 1] \text{ for all } j \tag{2.2}$$

$$m_j = \frac{\mu_{\mathrm{N}_1^j}(x_1) \times \mu_{\mathrm{N}_2^j}(x_2) \times \cdots \times \mu_{\mathrm{N}_n^j}(x_n)}{\sum_{j=1}^{r_m} \left(\mu_{\mathrm{N}_1^j}(x_1) \times \mu_{\mathrm{N}_2^j}(x_2) \times \cdots \times \mu_{\mathrm{N}_n^j}(x_n) \right)} \tag{2.3}$$

$\mu_{\mathrm{N}_i^j}(x_i)$ is the membership function corresponding to N_i^j. The output of the fuzzy system y is defined as,

$$y = \sum_{j=1}^{r_m} m_j w_j \tag{2.4}$$

One limitation of a fuzzy logic system is that the fuzzy rules and the parameters inside must be available, which may not be obtained directly through some self-learning or tuning algorithms.

2.2.2 Neural Network

Neural networks mimic the biological information processing mechanism. They are typically designed to perform a nonlinear mapping from a set of inputs to a set of outputs. Neural networks attempt to achieve a biological system type performance using a dense interconnection of simple processing elements that are analogous to biological neurons. They are adaptive information processing systems that can autonomously develop operational capabilities in response to an information environment. Hence, neural networks can learn from experience and generalize from previous examples, and are ideal in cases where the required mapping algorithm is not known and tolerance to faulty input information is required. The processing elements (PEs) are connected in a particular fashion. The behaviour of a trained neural network depends on the weights, which are also referred to as the strengths of connections between the PEs. Neural networks offer some advantages over conventional electronic processing techniques. These advantages include generalization capability, parallelism, distributed memory, redundancy, and learning. Currently, many neural network models and learning algorithms [Nauck 97, Jang 97, Kaimal 97 and Haykin 94] are being used. The famous ones include the back propagation [Rumelhart 86], the self-organizing map [Kohonen 88], and the Hopfield nets [Hopfield 84]. The main drawback of a neural network is its "black box" nature, i.e. the way that the weights affect the input-output relationship is difficult to know.

2.2.3 Genetic Algorithm

Genetic algorithm (GA) [Michalewicz 94] is a directed random search technique that is powerful in handling optimisation problems [Michalewicz 94]. It is especially useful for complex optimisation problems with a large number of parameters such that analytical solutions are difficult to obtain. GA can help to find out the globally optimal solution over a domain [Michalewicz 94]. It has been widely applied in different areas such as fuzzy control [Lam 01a], neural network [Lam 01b], forecasting [Ling 01a], path planning [Juidette 00], greenhouse climate control [Caponetto 00], modelling and classification [Setnes 00], recognition [Lam 01c] etc.

```
Procedure simple GA
begin
        $\tau \rightarrow 0$    // $\tau$ : iteration generation
        initialize $P(\tau)$              // $P(\tau)$: population for iteration $t$
        evaluate ($P(\tau)$)
while (not termination condition) do
      begin
        $\tau \rightarrow \tau + 1$
           select 2 parents $\mathbf{p}_1$ and $\mathbf{p}_2$ from $P(\tau - 1)$
           perform genetic operations (crossover and mutation)
           reproduce a new $P(\tau)$
           evaluate ($P(\tau)$)
      end
end
```

Fig. 2.1. Simple GA process in pseudo-codes.

The basic structure of a simple GA is shown in Fig. 2.1. To implement a typical GA, a population of chromosomes P is initialised and then evolves from generation τ to $\tau+1$ by repeating the following procedures: (1) Two parents are selected from P in such a way that the probability of selection is proportional to their fitness values. (2) A new offspring is generated from these parents using crossover and mutation operations, which are governed by the probabilities of crossover and mutation. (3) The population thus generated replaces the current population. The above procedures are repeated until a certain termination condition is satisfied. The termination condition may be that the algorithm stops when a predefined number of generations has been processed.

Traditional binary GA [Michalewicz 94] has some drawbacks when applying to multidimensional, high-precision numerical problems. For example, if 100 variables in the range [−500, 500] are involved, and a precision of 6 digits after the decimal point is required, the length of the binary solution vector is 3000. This, in turn, generates a search space of about 10^{1000}. The performance of the binary GA will then be poor. The situation can be improved if GA in floating-point numbers is used. Each chromosome is coded as a vector of floating point numbers of the same length as the solution vector. A large domain can thus be handled (e.g. the parameter space of a neural network.)

Different genetic operators have been proposed to improve the efficiency of the GA. Genetic operators usually refer to crossover and mutation. Traditionally, random mutation and crossover are employed. Yet, different modifications in the crossover and mutation operations have been reported. For the crossover operation, arithmetic crossover and heuristic crossover have been proposed [Michalewicz 94 and Wang 96]. For the mutation operation, uniform mutation and non-uniform mutation can be found [Michalewicz 94 and Wang 96]. The details about these genetic operations are given in Appendix. In the published GA with non-uniform mutation, the operation on the chromosomes is governed by the generation number only. Some other factors (such as the rate of change of the fitness value) that may slow down the convergence are not considered.

2.3 ELECTRIC LOAD FORECASTING SYSTEM

Computational intelligence techniques have been applied in short-term electric load forecasting (STELF). In particular, artificial neural networks have been considered as a very promising tool to short-term electric load forecasting [Hsu 91, Part 91, Lee92, Lu93, Kiartzis 97, Bakirtzls 96, Momoh 97, Rewagad 98 and Drezga 99]. In recent years, fuzzy logic has been used to deal with variable linguistic information in load forecasting [Hse 92]. By processing fuzzy information, reasoning with respect to a linguistic knowledge base can be done. When the back-propagation feed-forward neural networks were used [Hsu 91, Part 91, Lee92, Lu93, Kiartzis 97, Bakirtzls 96, Momoh 97, Rewagad 98 and Drezga 99], the common problems of convergence to a local minima and sensitivity to initial values persist. Most of the reported NNs for STELF are fully connected. However, the performance of a fully connected NN may not be better than that of a partly connected NN with the same number of hidden nodes. This is because some links in an NN could be redundant. In [Hse 92], a fuzzy system has been applied for electric load forecasting. In this electric load forecasting system, the fuzzy rules or the fuzzy membership functions are determined by trial and error. The resulting system is not necessarily optimal.

2.4 ELECTRIC LOAD BALANCING SYSTEM

The basic purpose of an electric load balancing system is to maintain a relatively steady pattern of power demand on the utility company. Schemes serving a similar purpose were proposed under different names such as time-of-use, load management program, demand-side management, and load balancing. Among them, some researchers proposed pricing schemes for electricity supply in order to achieve the aim [David 94, 96, He 97, Sheen 94, 95 and Maeda 92]. Based on charging more in peak hours, these schemes employ complex mathematical methods to optimise the electricity pricing and attract a reduction of loading at peak hours. However, this method is not applicable in the areas where the utility companies charge the same at all time. Some researchers [Naga Raj 95 and Lee 92] focus on the method of implementation, like the load balancing that uses feeders in a distribution system. Although these methods are suitable for large power systems, it is too expensive to be implemented at home.

CHAPTER THREE

POWER LINE DATA NETWORK

3.1 INTRODUCTION

In this chapter, a power line data network based on the spread-spectrum technology [Radford 96], which facilitates communications at 10 Kbps in the noisy and signal-distorting environment of AC power lines, will be reported. This power line network serves as a backbone of communication among home appliances and users in an intelligent home. In this way, a low-cost data network can be built without resorting to manufacturers of electrical appliances and the installation of a LAN. Besides the developed power line data network, this chapter will also discuss the features offered by the intelligent home system.

3.2 AC POWER LINE DATA NETWORK

3.2.1 Background of AC Power Line Data Network

Wireless and wired LAN are two common physical media for data transmission. Besides the acceptable data rate and the high reliability, a wireless LAN has a number of advantages. First, no extra cost is required for the installation of cables. Second, every hardware component can be portable. Third, it requires no amendment to the infrastructure on expanding the system. However, it is difficult to limit the coverage of the transmission media strictly within a house and not to affect or be affected by other networks. It is an especially serious problem for the highly populated residential environment. A wired LAN with dedicated cables gives the fastest data rate and most reliable data transmission. Unfortunately, a wired LAN requires an extra cabling cost, and the system has to be well designed before building it. Moreover, additions or removals of components may imply a costly re-cabling.

To compromise the above advantages and disadvantages, we propose to use the existing AC power line as a networking media for an intelligent home system. The AC power lines have already existed in every corner of a house. By using the AC power line network, all electrical appliances plugged to a power socket will not only obtain electric power but also digital data. Still, we have to face some problems. Two main concerns are the noisy environment and the low impedance of the power line in the operating bandwidth [Liu 99 and Schickhuber 97]. Noise in an AC power line will disturb the data signals and decrease the network reliability. When the line impedance is low in the operating bandwidth, the maximum transmission speed will be affected. To alleviate this problem, the spread-spectrum technology [Radford 96] is applied in signal transmission that can facilitate a data rate of 10 Kbps.

3.2.2 Power Line Data Network Design

With the power line network, electrical appliances can be controlled via line/mobile phones, PDAs, keypads or personal computers anytime and anywhere, inside or outside the home. The network employs the spread spectrum carrier Consumer Electronics Bus (CEBus) standard (which will be discussed in sub-section 3.2.3). A diagram of the power line data network in the intelligent home is shown in Fig. 3.1.

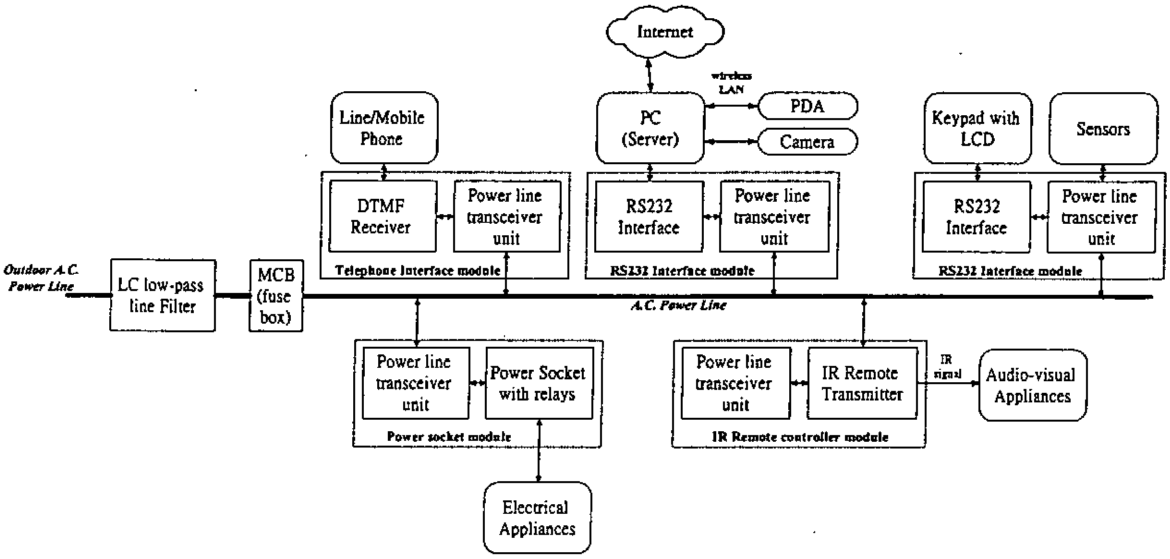


Fig. 3.1. Diagram of the power line data network.

The system contains two types of units: control units and slave units. A control unit can be a telephone interface module accessed by a line/mobile phone, or a serial (RS232) interface module in a PC accessed by a PDA or a keypad. Signals input to a control unit is captured by a video camera or sensors. A slave unit can be a power socket module for on/off control of electrical appliances, or an infrared (IR) remote control module for AV equipment. The details about each module and the features of the intelligent system will be described in next section.

A data packet in the power line data network is either a control packet (generated by a control unit) or a status packet (generated by a slave unit) as shown in Fig. 3.2. The control packet has three bytes defined as follows:

Destination Address (byte 0): defining the device address of the receiving node.

Control Command (byte 1): defining the action or the request for feedback status information (from the slave unit).

Reserved (byte 2): reserved for future development. Normally, we set this byte to 0xFF so that the transmission rate is the fastest (a property of the spread spectrum CEBus standard).

The feedback packet has one byte only, which contains the electrical appliance's on/off status or other information.

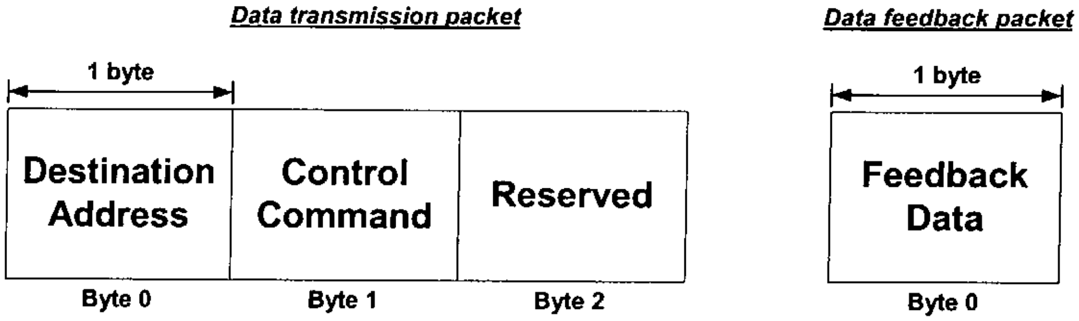


Fig. 3.2. Data packet format.

To interface a unit into the power line, a power line transceiver has to be developed which is the basic unit of every functional module. There are many transceivers employing different modulation techniques available in the market [Schickhuber 97]. The simplest modulation techniques are amplitude shift keying (X10 and Philips TDA5051A) and frequency shift keying (SGS Thomson ST7537). These techniques provide a low-cost solution for power line networking but give a very low bit rate (typically 1.2 Kbps). Moreover, with a higher noise level as compared with UTP network cables, the power lines may suffer from a high error rate. Some chips applying the spread spectrum carrier technique can give a better performance. For example, Adaptive Networks AN1000, LonWorks, and Intellon P300 all contain chipsets employing spread spectrum techniques. The AN1000 can achieve a data rate of 100 Kbps, while the other two can achieve 10 Kbps. The cost of AN1000 and LonWorks are relatively expensive. To trade off performance against cost, the Intellon P300 is selected in the development.

A block diagram of the power line transceiver unit embedded with a P300 chip is shown in Fig. 3.3. The prototyped power line transceiver is shown in Fig. 3.4. The P300 chip is the power line network interface controller chip that serves as a power line transceiver and channel access interface for CEBus [Douligenis 93] compatible products. It is a host interface transmitting and receiving data to and from the power line via the Serial Peripheral Interface (SPI). There are a filter, a driving amplifier, and a coupling circuit between the unit and the power line. The 8051 MCU is to support data or command (defined by the manufacturer) transfer between the SPI and the application unit (control unit or slave unit).

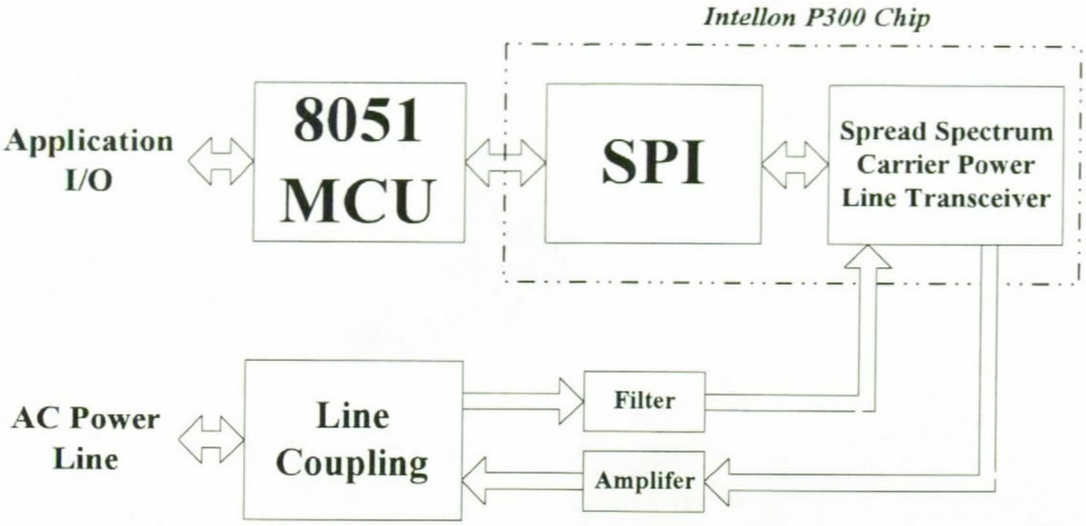


Fig. 3.3. Block diagram of the power line transceiver.

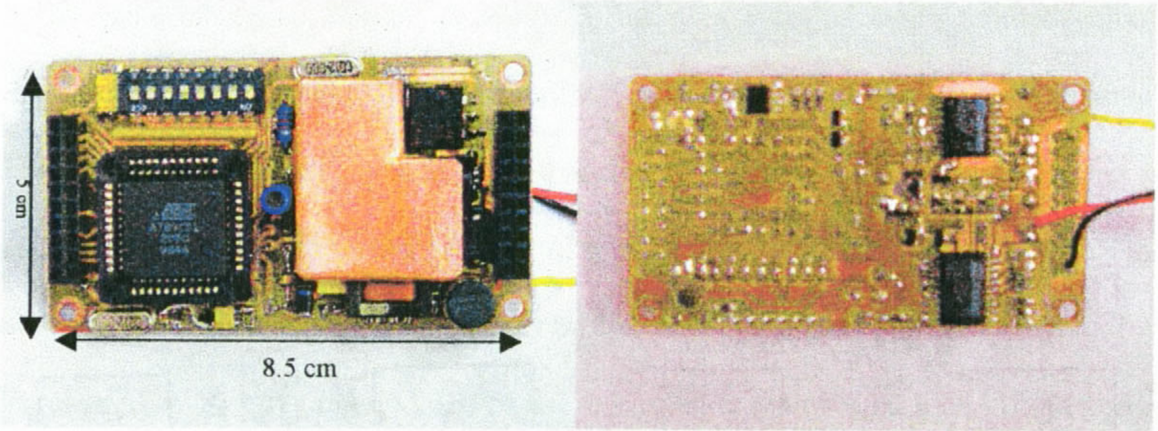


Fig. 3.4. The outlook of the power line transceiver.

3.2.3 Spread Spectrum Carrier Consumer Electronics Bus Standard

Consumer Electronics Bus (CEBus) [Douligenis 93] is the Electronic Industry Association's (EIA) standard for home automation. It is an easy-to-install and effective standard that uses four of the seven open systems interconnection (OSI) layers and omits the transport, session and presentation layers. These omissions reduce the packet length and the node complexity. The interface between different layers in the CEBus is defined as a set of service primitives. Each layer provides service to the layer above it; higher layers subscribe to the service of the lower layers.

CEBus allows communication among any CEBus compatible devices regardless of their manufacturing companies. It uses a broad-spectrum frequency, which is swept from 100 kHz to 400 kHz over a 100 μs unit symbol time (UST). The UST is the building block of a data stream. Each UST consists of a frequency swept chirp of the range of 100 kHz to 400 kHz. The sweeping time is 100 μs for the packet body and 114 μs for the preamble.

Unlike most transmission formats, CEBus uses a set of four medium symbols instead of the more common binary symbols. The symbols are: 1 (binary one), 0 (binary zero), EOF (end of field), which is used to separate packet fields, and EOP (end of packet), which is used to identify the end of a transmitted packet. The four symbols are encoded on each medium by using four different frequencies. The 1 symbol is represented by 10 kHz (a period of 100 μs), 0 symbol is represented by 5 kHz (a period of 200 μs), EOF is represented by 3.3 kHz (a period of 300 μs), and EOP is represented by 2.5 kHz (a period of 400 μs.) Spread spectrum is used in the CEBus power line standard. Its signalling works by spreading a transmitted signal over a range of frequencies, rather than using a single frequency.

3.3 FEATURES OF THE INTELLIGENT HOME SYSTEM

To make the proposed intelligent home system highly flexible and easily expandable, a modular base solution is employed. This means that every module can work independently without relying on others. In this case, users can install a minimum of one control unit and one slave unit to implement the intelligent system. If users want to include more features, their systems can be expanded at any time without significant alternations to the existing system.

Fig. 3.1 illustrates some modules which can be included in the intelligent home system. As a control unit, a telephone interface module enables the whole system to be connected and accessed via a mobile/line telephone or a PDA with a GSM/CDMA card. As a mobile phone is common to everyone nowadays, we can easily control home appliances through an outdoor access. If a PDA is used, the images captured by a video camera connected to a PC can be monitored via a wireless LAN or a GSM/CDMA network. It allows users to have surveillance of their home anywhere and anytime. An RS232 interface module can interface with any device that supports the RS232 standard. In particular, a personal computer having a serial port that supports RS232 can connect to the power line data network via this module. Application programs have been written to control the intelligent home system. A home server (the PC) can be included to enhance the intelligent features within the home, including the load forecasting and balancing to be discussed in Chapters 4 and 5 respectively. A keypad connected with the RS232 module can also be used to control the intelligent home system inside the home. Sensor modules are used to sense motion, temperature, smoke etc. so that the system can be informed of any change via the power line data network for suitable actions. As a slave unit, the power socket module consists of relay-controlled AC power sockets. It can turn on or off any appliance connected to it. This module can also report the on/off status of each socket at the request of a control unit. With a data rate of 10 Kbps, not only the on/off control but also other programming features can be realized in real time. The IR remote control module consists of a multi-function remote controller. Ideally, an intelligent appliance should support communications within the network according to certain hardware and software protocols. However, a widely adopted protocol has not come yet. The IR remote control module is therefore an intermediate solution before any real intelligent appliance that supports communications within a network is available in the market.

An emulated intelligent home [Wong 00] has been built in a flat of about 20m^2 inside the Hong Kong Polytechnic University (Fig. 3.5). Different modules have been developed to illustrate the proposed features. These features will be introduced in the following sub-sections.

Fig. 3.5. Intelligent home.

3.3.1 Telephone Interface Module

A telephone interface (Fig. 3.6) enables the whole system to be connected and accessed via a mobile or a line phone. This telephone interface module provides a complete solution for outdoor control of home appliances without relying on the Internet. The heart of this module is a DTMF decoder (MT8870), which is a dialing system with a 4-bit DTMF decoder. On dialing in, the module will connect the phone line and wait for commands in DTMF from the caller. Caller can check the status of the electrical appliance through audio feedback from the module to the phone.

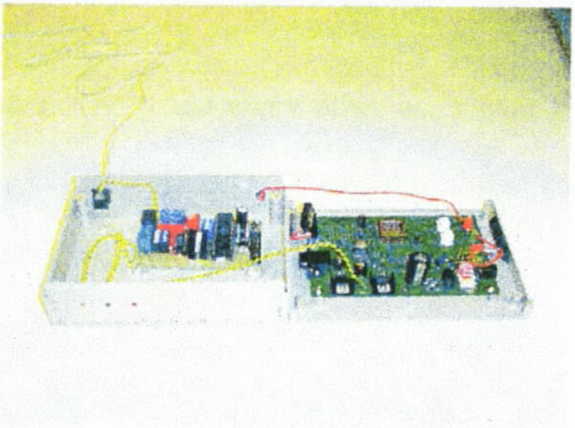


Fig. 3.6. Telephone interface module.

3.3.2 Power Socket Module

The power socket module (Fig. 3.7) consists of four relay-controlled AC power sockets. The four relays can be accessed individually via the power line network. Any appliance requiring on/off control can be connected to this module. For example, we have connected a lamp and a TV set in the room. This module can report the on/off status of each socket at the request of other control units in the network.

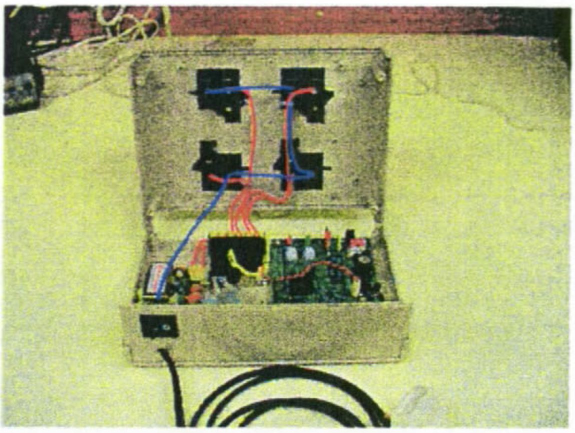


Fig. 3.7. Power socket module.

3.3.3 *RS232 Interface Module*

Personal computers (PCs) and many other equipment have a standard RS232 serial port for communication. To link them to the power line network, an RS232 module is developed. This module makes use of the build-in serial port of the MCU 8051 inside the power line transceiver unit, with a MAX232 to interface. The module is connected to a PC or a keypad (with an LCD display) such that the PC or keypad can send control commands and receive data through the power line network.

3.3.4 *IR Remote Interface Module*

The heart of this module (Fig. 3.8) is a power line transceiver unit connecting with an IR remote controller unit. The IR remote controller unit consists of an RS232 interface and a universal remote IR controller. An audio-visual appliance can be controlled by this module when the power line transceiver transfers the control code from a control unit via the power line.

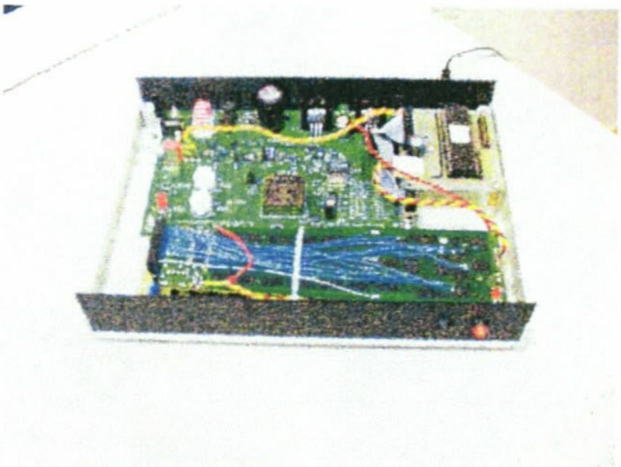


Fig. 3.8. IR remote interface module.

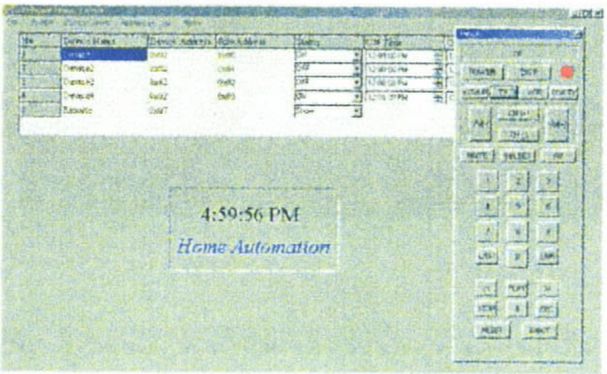
3.3.5 PDA

To control the electrical appliances inside the intelligent home, a control interface in a handheld Pocket PC (*Compaq iPAQ H3630*) has been implemented. An additional PCMCIA GSM card phone can be inserted into the Pocket PC to allow connection to the power line data network through the GSM data channel. The Pocket PC can also connect to a video server inside the intelligent home through a wireless LAN, or outside the intelligent home through the GSM network. Users can watch the captured real-time video signal using the Pocket PC, and remotely control the pan and tilt motion of the camera to adjust the viewing angle. The Pocket PC with the developed user interface is shown in Fig. 3.9.

Fig. 3.9. Pocket PC with the developed user interface.

3.3.6 Personal Computer with Application Programs

In the intelligent home, an RS232 module is connected with a PC so that the PC can send control commands and receive data through the power line network. An application program (Fig. 3.10) that can control the power socket and IR remote control modules has been developed. This program has a timer feature that can pre-set the operation period of the electrical appliances, and allows the real time control of audio-visual appliances through the IR remote control module. The PC is also the platform for the future implementation of the home load forecasting and balancing system.



4:59:56 PM

Home Automation

Fig. 3.10. Application program.

3.3.7 LC Low Pass Line Filter

The digital signals in the power line data network can be isolated from the outside network by using a two-pole LC low-pass line filter installed near the main circuit breaker (MCB). Interference with other house can then be prevented. The schematics of the LC low-pass line filter and its effect are shown in Fig. 3.11. The output of the CRO's channel 1 is a data signal at the input of this filter. The output of the CRO's channel 2 shows that the data signals can be blocked after passing through this filter.

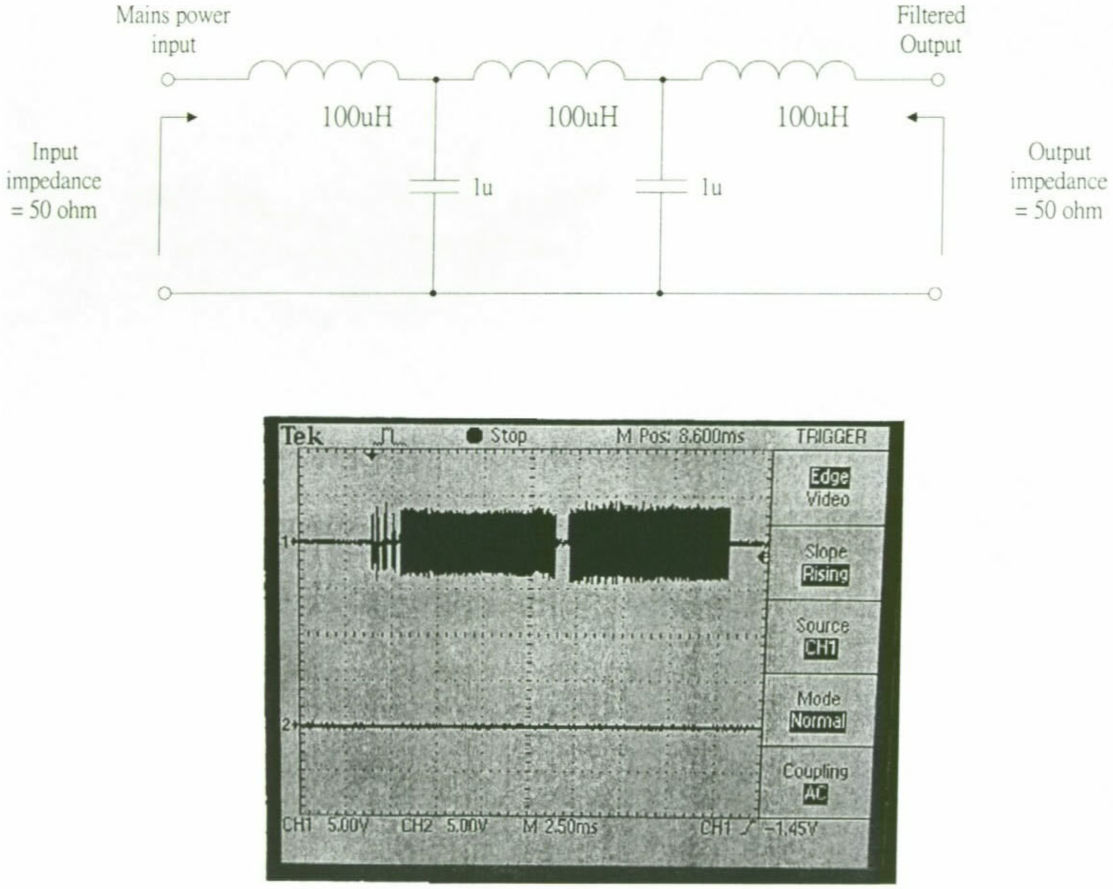


Fig. 3.11. LC low-pass line filter and its effect.

3.4. CHAPTER CONCLUSION

In this chapter, an intelligent home system realized by a spread spectrum AC power line data network has been discussed. The network and the system features have been presented. The points of .consideration include cost, reliability, speed of communication and effectiveness. Using this system, communications among electrical appliances and home users can take place at any time and place. This serves as a low cost backbone of communication to enhance the security and comfort of the home users. One advantage of this system is that every module can be easily added or removed, and placed anywhere in a house without too much influence to the whole system and the need of re-routing any cable. Hence a prudent plan for the intelligent home system before building is not needed. Any existing homes can implement this system.

Chapter Four

ELECTRIC LOAD FORECASTING SYSTEM

4.1 INTRODUCTION

Computational intelligence techniques for short-term electric load forecasting in an intelligent home will be presented in this chapter. The intelligent home system has already been discussed in Chapter 3. In this system, the AC power line serves as a data communication channel for electrical appliances. With this AC power line data network, a short-term electric load forecasting can be realized. An accurate load forecasting can bring the following benefits to the intelligent home.

1) Increasing the reliability of the AC power line data network - On using the AC power line as the networking medium, we may suffer from the possible low impedance of the power line in the operating bandwidth [Liu 1999] [Schickhuber 1997] for data transmission. When this occurs, the maximum transmission rate, the reliability and the throughput of the AC power line data network will decrease. The attenuation of the data signal in an AC power line is proportional to the load connected to it. The reliability of the power line data network can be enhanced if the load is kept at an optimal level through forecasting and power backup (load balancing). We can also adaptively set a suitable data transmission rate based on the forecasted load condition in order to reduce the overhead of data retransmission.

2) Optimal electric load - At present, the peak demand of electricity is met by

operating costly auxiliary generators, or by purchasing power from other utility companies. The cost for supplying peak power is therefore much higher than that for supplying the average power. A reduction in the peak value of electricity demand can be achieved if we can realize electric load forecasting, and schedule the demands on the utility company accordingly. This has to be supported by batteries installed in the intelligent home that are responsible for sharing the load demand. The design of the home load balancing system based on an electric load forecaster will be presented in Chapter 5.

Existing electric load forecasting systems using computational intelligence methods and their difficulties have been discussed in Chapter 2. In this chapter, three computational intelligence techniques for short-term electric load forecasting (STELF) will be reported. The first approach for realizing STELF is by using an optimal neural network. A neural network with a switch introduced in each link is proposed. By introducing the switches to the links, the proposed neural network is able to learn the input-output relationships of an application as well as the network structure. The second approach for realizing STELF is by using a neural fuzzy network (NFN). The optimal network structure (number of fuzzy rules) can be found when switches in the links of the network are introduced. This implies that the cost of implementing the proposed NFN can be reduced. The third approach for realizing STELF is by using a modified neural network. Two different activation transfer functions are used in the neuron and a node-to-node relationship is proposed in the hidden layer. This network model is found to be able to give a better performance with a smaller number of hidden nodes. All the three approaches are trained by a proposed fuzzy genetic algorithm (GA).

The proposed fuzzy GA is modified from the traditional GA with arithmetic crossover and non-uniform mutation. Fuzzy logic can express expert knowledge and experience in some linguistic rules, which can be easily understood by human beings. In the proposed fuzzy GA, fuzzy logic is used to help realize the crossover and mutation operations. Consequently, the human knowledge on the crossover and mutation can be incorporated using fuzzy rules. It will be shown that the proposed fuzzy GA performs more efficiently and provides a faster convergence than the traditional GA in some benchmark test functions.

This chapter is organized as follows. The proposed fuzzy GA and its benchmark tests will be introduced in Section 4.2. The short-term electric load forecasting with the proposed fuzzy GA-based neural network will be presented in Section 4.3. In Section 4.4, the design and implementation of the short-term electric load forecasting using the proposed fuzzy GA-based neural fuzzy network will be discussed. The short-term electric load forecasting realized by the modified fuzzy GA-based neural network will be introduced in Section 4.5. A chapter conclusion will be drawn in Section 4.6.

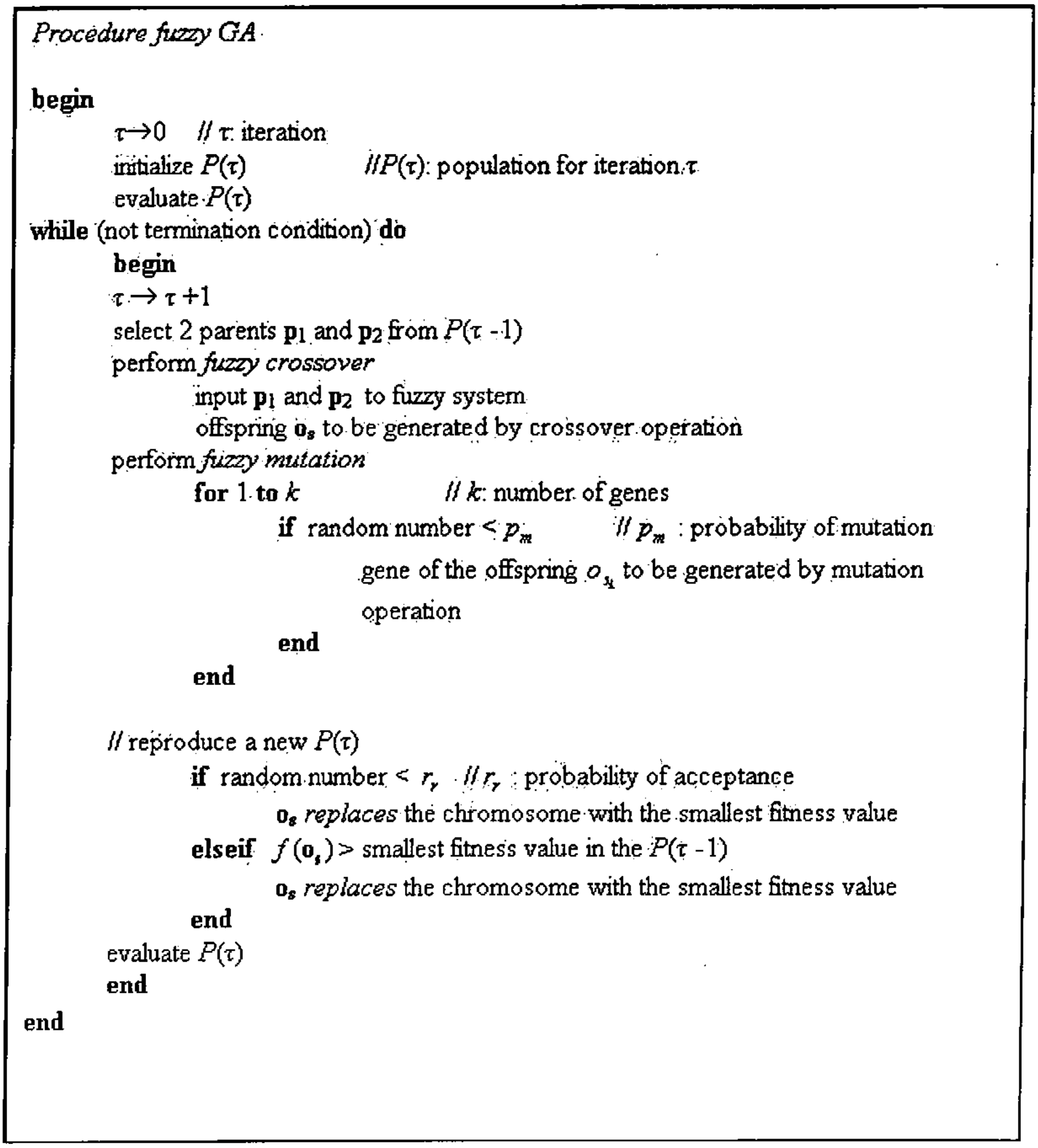
4.2 FUZZY GENETIC ALGORITHM

Genetic Algorithm (GA) is a directed random search technique, which is a powerful searching algorithm for complex optimisation problems. It helps to find out the globally optimal solution over a domain.

In this section, a fuzzy GA is proposed which is developed from the traditional GA with arithmetic crossover and non-uniform mutation [Michalewicz 94]. Fuzzy logic is applied in the crossover and mutation operations. As a result, human knowledge on the crossover and mutation can be incorporated using fuzzy rules. It will be shown that the proposed fuzzy GA performs more efficiently and provides a faster convergence than the traditional GA in some benchmark test functions [De Jong 75 and Yao 99]. This section is organized as follows. The proposed fuzzy GA is presented in sub-section 4.2.1. The applicability and efficiency of the fuzzy GA are tested by some benchmark functions in sub-section 4.2.2.

4.2.1 Fuzzy Genetic Algorithm

The pseudo-codes of the proposed fuzzy GA are shown in Fig. 4.1. The details are presented as follows.



```
Procedure fuzzy GA

begin
        $\tau \rightarrow 0$    // $\tau$: iteration
        initialize $P(\tau)$               //$P(\tau)$: population for iteration $\tau$
        evaluate $P(\tau)$
while (not termination condition) do
        begin
        $\tau \rightarrow \tau + 1$
        select 2 parents $p_1$ and $p_2$ from $P(\tau - 1)$
        perform fuzzy crossover
                input $p_1$ and $p_2$ to fuzzy system
                offspring $o_s$ to be generated by crossover operation
        perform fuzzy mutation
                for 1 to $k$                  // $k$: number of genes
                        if random number < $p_m$         // $p_m$ : probability of mutation
                                gene of the offspring $o_s$ to be generated by mutation
                                operation
                        end
                end

        // reproduce a new $P(\tau)$
                if random number < $r_r$   // $r_r$ : probability of acceptance
                        $o_s$ replaces the chromosome with the smallest fitness value
                elseif $f(o_s)$ > smallest fitness value in the $P(\tau - 1)$
                        $o_s$ replaces the chromosome with the smallest fitness value
                end
        evaluate $P(\tau)$
        end
end
```

Fig. 4.1. Fuzzy GA process in pseudo-codes.

4.2.1.1. Initial population

The initial population is a potential solution set *P*. The first set of population is usually generated randomly.

$$P = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{pop\_size}\} \quad (4.1)$$

$$\mathbf{p}_i = \begin{bmatrix} p_{i_1} & p_{i_2} & \cdots & p_{i_j} & \cdots & p_{i_{no\_vars}} \end{bmatrix}, i = 1, 2, \ldots, pop\_size; j = 1, 2, \ldots, no\_vars \quad (4.2)$$

$$para^j_{\min} \le p_{i_j} \le para^j_{\max}, i = 1, 2, \ldots, pop\_size; j = 1, 2, \ldots, no\_vars \quad (4.3)$$

where pop_size denotes the population size; no_vars denotes the number of variables to be tuned; p_{i_j}, i = 1, 2, ..., pop_size; j = 1, 2, ..., no_vars, are the parameters to be tuned; $para^j_{\min}$ and $para^j_{\max}$ are the minimum and maximum values of the parameter p_{i_j} respectively. It can be seen from (4.1) to (4.3) that the potential solution set P contains some candidate solutions $\mathbf{p}_i$ (chromosomes). The chromosome $\mathbf{p}_i$ contains some variables p_{i_j} (genes).

4.2.1.2 Evaluation

Each chromosome in the population will be evaluated by a defined fitness function. The better chromosomes will return higher values in this process. The fitness function to evaluate a chromosome in the population can be written as,

$$fitness = f(\mathbf{p}_i) \quad (4.4)$$

The fitness value should have a non-negative nature and the form of the fitness function depends on the application.

4.2.1.3. Selection

Two chromosomes in the population will be selected to undergo fuzzy genetic operations for reproduction. It is believed that the high potential parents will produce better offspring (survival of the best ones). The chromosome having a higher fitness value should therefore have a higher chance to be selected. The selection can be done by assigning a probability q_i to the chromosome $\mathbf{p}_i$ such that:

$$q_i = \frac{f(\mathbf{p}_i)}{\sum_{j=1}^{pop'\_size} f(\mathbf{p}_j)}, \; i = 1, 2, \ldots, pop\_size \tag{4.5}$$

The cumulative probability $\hat{q}_i$ for the chromosome $\mathbf{p}_i$ is defined as,

$$\hat{q}_i = \sum_{j=1}^{i} q_j \,, \; i = 1, 2, \ldots, pop\_size \tag{4.6}$$

The selection process starts by randomly generating a nonzero floating-point number, $d \in [0 \quad 1]$. Then, the chromosome $\mathbf{p}_i$ is chosen if $\hat{q}_{i-1} < d \leq \hat{q}_i$, i = 1, 2, ..., *pop_size* ($\hat{q}_0 = 0$). It can be observed from this selection process that a chromosome having a larger $f(\mathbf{p}_i)$ will have a higher chance to be selected. Consequently, the best chromosomes will get more copies, the average will stay and the worst will die off. In the selection process, only two chromosomes will be selected to undergo the fuzzy genetic operations.

4.2.1.4. Fuzzy genetic operations

The fuzzy genetic operations are to generate some new chromosomes (offspring) from their parents after the selection process. They include the fuzzy crossover and the fuzzy mutation operations.

A. Fuzzy Crossover

The fuzzy crossover operation is mainly for exchanging information from the two parents obtained in the selection process. If the two selected chromosomes are $\mathbf{p}_1$ and $\mathbf{p}_2$ with fitness values $f(\mathbf{p}_1)$ and $f(\mathbf{p}_2)$ respectively, the offspring to be generated is governed by the following fuzzy rules,

$$\text{Rule } i\text{: IF } f(\mathbf{p}_1)-f(\mathbf{p}_2) \text{ is } \mathrm{M}_i \text{ THEN } w_a = w_{c_i} \tag{4.7}$$

where M_i is a fuzzy term of rule i, i = 1, 2, …, r_c; r_c denotes the number of rules. $w_{c_i} \in [0 \quad 1]$ is a singleton to be determined. The value of w_a determines the offspring $\mathbf{o}_s$ after the fuzzy crossover:

$$\mathbf{o}_s = w_a \mathbf{p}_1 + (1-w_a)\mathbf{p}_2 \tag{4.8}$$

and

$$w_a = \sum_{i=1}^{r_c} w_i w_{c_i} \tag{4.9}$$

where

$$w_i = \frac{\mu_{\mathrm{M}^i}\left(f(\mathbf{p}_1) - f(\mathbf{p}_2)\right)}{\sum_{k=1}^{r_c}\left(\mu_{\mathrm{M}^i}\left(f(\mathbf{p}_1) - f(\mathbf{p}_2)\right)\right)} \tag{4.10}$$

$\mu_{\mathrm{M}^i}\left(f(\mathbf{p}_1) - f(\mathbf{p}_2)\right)$ is the membership function corresponding to M^i. From (4.10),

$$\sum_{i=1}^{r_c} w_i = 1,\ w_i \in [0,\ \ 1] \text{ for all } i \tag{4.11}$$

The rule of this fuzzy crossover should be set such that the offspring will look closer to the parent with a larger fitness value.

B. Fuzzy Mutation

The offspring (4.8) may then undergo a fuzzy mutation operation. The mutation operation changes the genes of the offspring chromosomes. Every gene of the offspring $\mathbf{o}_s$ of (4.8) will have a chance to mutate governed by a probability of mutation, $p_m \in [0\ \ 1]$, which is defined by the user. This probability gives an expected number ($p_m \times no\_vars$) of genes that undergo the mutation. For each gene, a random number between 0 and 1 will be generated such that if it is less than or equal to p_m, the operation of mutation will take place on that gene. The gene of the offspring of (4.8) is then mutated by:

$$\hat{o}_{s_k} = \begin{cases} o_{s_k} + \Delta o_{s_k}^U \text{ if } f(\mathbf{o}_s + \Delta \mathbf{o}_{s_k}^U) \geq f(\mathbf{o}_s - \Delta \mathbf{o}_{s_k}^L) \\ o_{s_k} - \Delta o_{s_k}^L \text{ if } f(\mathbf{o}_s + \Delta \mathbf{o}_{s_k}^U) < f(\mathbf{o}_s - \Delta \mathbf{o}_{s_k}^L) \end{cases},\ k = 1, 2, \ldots, no\_vars \tag{4.12}$$

where

$$\Delta o_{s_k}^U = r^{\frac{1}{w_{mk}}}\left(para_{\max}^k - o_{s_k}\right) \tag{4.13}$$

$$\Delta o_{s_k}^L = r^{\frac{1}{w_{mk}}}\left(o_{s_k} - para_{\min}^k\right) \tag{4.14}$$

$$\Delta \mathbf{o}_{s_k}^U = \begin{bmatrix} 0 & 0 & \cdots & \Delta o_{s_k}^U & \cdots & 0 \end{bmatrix} \tag{4.15}$$

$$\Delta \mathbf{o}_{s_k}^L = \begin{bmatrix} 0 & 0 & \cdots & \Delta o_{s_k}^L & \cdots & 0 \end{bmatrix} \tag{4.16}$$

$r \in [0 \quad 1]$ is a randomly generated number; $w_{m_k} \in [0 \quad 1]$ is a weight governing the magnitudes of $\Delta o_{s_k}^U$ and $\Delta o_{s_k}^L$. The value of weight w_{m_k} is determined by two factors: the rate of change of the fitness with respect to o_{s_k}, i.e. $\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|$, and the value of $\frac{\tau}{T}$. A large value of $\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|$ implies the gene o_{s_k} has a large search space. A large weight w_{m_k} is thus necessary when $\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|$ is large in order to obtain a significant mutation (large $\Delta o_{s_k}^U$ or $\Delta o_{s_k}^L$). On the other hand, τ and T denote the current iteration number and the total number of iterations respectively, and $\frac{\tau}{T}$ is used for the fine-tuning. The value of weight w_{m_k} should approach 0 as $\frac{\tau}{T}$ increases in order to reduce the significance of the mutation. Based on these two factors, the weight w_{m_k} is governed by the following fuzzy rules:

Rule j: IF $\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|$ is N_1^j AND $\frac{\tau}{T}$ is N_2^j THEN $w_{m_k} = w_{s_j}$, $j =$ 1, 2, …, r_m; $k = 1$, 2, …, *no_vars* (4.17)

where N_1^j and N_2^j are fuzzy terms of rule j, r_m denotes the number of rules, $w_{m_k} \in [0 \quad 1]$, $w_{s_j} \in [0 \quad 1]$ is a singleton to be determined. The final value of w_{m_k} is given by

$$w_{m_k} = \sum_{j=1}^{r_m} m_j w_{s_j}\text{, } k = 1, 2, \ldots, no\_vars \tag{4.18}$$

where

$$m_j = \frac{\mu_{\mathrm{N}_1^j}\left(\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|\right) \times \mu_{\mathrm{N}_2^j}\left(\frac{\tau}{T}\right)}{\sum_{j=1}^{r_m}\left(\mu_{\mathrm{N}_1^j}\left(\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|\right) \times \mu_{\mathrm{N}_2^j}\left(\frac{\tau}{T}\right)\right)} \tag{4.19}$$

$\mu_{\mathrm{N}_1^j}\left(\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|\right)$ and $\mu_{\mathrm{N}_2^j}\left(\frac{\tau}{T}\right)$ are the membership functions corresponding to N_1^j and N_2^j respectively.

$$\sum_{j=1}^{r_m} m_j = 1\text{, } m_j \in [0,\quad 1] \text{ for all } j \tag{4.20}$$

4.2.1.5. Reproduction

After going through the fuzzy mutation process, the new offspring will be evaluated using the fitness function of (4.4). This new offspring will replace the chromosome with the smallest fitness value among the population if a randomly generated number within 0 to 1 is smaller than $r_r \in [0 \quad 1]$, which is the probability of acceptance defined by users. Otherwise, the new offspring will replace the chromosome with the smallest fitness value if the fitness value of the offspring is greater than the fitness value of that chromosome in the population.

After the operation of selection, fuzzy crossover and fuzzy mutation, a new population is generated. This new population will repeat the same process to produce another offspring. Such an iterative process can be terminated when a defined condition is reached, e.g. a sufficiently large number of iterations has been reached.

4.2.1.6. Choosing the parameters

We can regard the GA as a balance between the exploration of new regions and the exploitation of already sampled regions in the search space. This balance, which critically controls the performance of the GA, is governed by the right choices of control parameters: the probability of fuzzy mutation (p_m), the probability of acceptance (r_r) and the population size (*pop_size*). Some views about these parameters are included as follows:

- Increasing the probability of fuzzy mutation tends to transform the genetic search into a random search. This probability gives us an expected number ($p_m \times no\_vars$) of genes that undergo the mutation. When $p_m = 1$, all genes will mutate. The value of the fuzzy mutation probability depends on the desirable number of genes that undergo the mutation operation.

- Increasing the probability of acceptance will increase the chance that a poor offspring joins the population. This reduces the probability that the GA prematurely converges to a local optimum. From experience, a probability of acceptance of 0.1 is a good enough choice for many optimization problems.

- Increasing the population size will increase the diversity of the search space, and reduce the probability that the GA prematurely converges to a local optimum. However, it also increases the time required for the population to converge to the optimal region in the search space. From experience, a population size of 10 is an acceptable choice.

4.2.2 Benchmark Test Functions

Benchmark test functions [De Jong 75 and Yao 99] are used to examine the applicability and efficiency of the proposed fuzzy GA. Six test functions, $f_i(\mathbf{x})$, i = 1, 2, 3, 4, 5, 6, will be used, where $\mathbf{x} = \begin{bmatrix} x_1 & x_1 & \cdots & x_{no\_x} \end{bmatrix}$. no_x is an integer denoting the dimension of the vector **x**.

The six test functions are defined as follows,

$$f_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 \text{ , } -5.12 \le x_i \le 5.12 \tag{4.21}$$

where $n = 3$ and the minimum point is at $f_1(0, 0, 0) = 0$.

$$f_2(\mathbf{x}) = \sum_{i=1}^{n-1}\left(100\left(x_{i+1} - x_i^{\,2}\right)^2 + (x_i - 1)^2\right), \; -2.048 \le x_i \le 2.048 \tag{4.22}$$

where $n = 2$ and the minimum point is at $f_2(0, 0) = 0$.

$$f_3(\mathbf{x}) = 6n + \sum_{i=1}^{n} floor(x_i) \text{ , } -5.12 \le x_i \le 5.12 \tag{4.23}$$

where $n = 5$ and the minimum point is at $f_3([-5.12, \ -5], \ldots, [-5.12, \ -5]) = 0$. The floor function, $floor(\cdot)$, is to round down the argument to an integer.

$$f_4(\mathbf{x}) = \sum_{i=1}^{n} i x_i^4 + Gauss(0,1) \text{ , } -1.28 \le x_i \le 1.28 \tag{4.24}$$

where $n = 3$ and the minimum point is at $f_4(0, 0, 0) = 0$. *Gauss*(0, 1) is a function to generate uniformly a floating-point number between 0 and 1 inclusively.

$$f_5(\mathbf{x}) = \left[\frac{1}{k} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right]^{-1}, \quad -65.356 \le x_i \le 65.356 \qquad (4.25)$$

where

$$\mathbf{a} = \{a_{ij}\} = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 \\ 32 & 32 & 32 & 32 & 32 & -16 & -16 & -16 & -16 & -16 \end{bmatrix}$$

$$\begin{matrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 \\ 0 & 0 & 0 & 0 & 0 & 16 & 16 & 16 & 16 & 16 & 32 & 32 & 32 & 32 & 32 \end{matrix} \Bigg] ,$$

$k = 500$ and the minimum point is at $f_5(-32, -32)$ 1.

$$f_6(\mathbf{x}) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right], \ -5.12 \le x_i \le 5.12 \qquad (4.26)$$

where $n = 3$ and the minimum point is at $f_6(0, 0, 0) = 0$

The benchmark test functions belong to three main classes: unimodal functions, multimodal functions with only a few local minima, and multimodal functions with many local minima problem. In this thesis, functions 1 to 4 are unimodal function, function 5 is a multimodal function with only a few local minima and function 6 is amultimodal function with many local minima. A brief description of each function and the problem it represents are given as follows:

- f_1 is a sphere function, which is probably the most widely used test function. It is smooth, unimodal and symmetric. The performance on this function is a measure of the general efficiency of an algorithm.

- f_2 is a Rosenbrock function of which the optimum is located in a very narrow ridge. The tip of the ridge is very sharp, and it runs around a parabola. Algorithms not able to discover good directions will perform poorly in this problem.

- f_3 is a step function that is a representative of flat surfaces. Flat surfaces are obstacles for optimisation algorithms because they do not give any information about the search direction. Unless the algorithm has a variable step size, it can get stuck in one of the flat surfaces.

- f_4 is a quartic function, which is a simple unimodal function, padded with noise. The Gaussian noise causes the algorithm never getting the same value at the same point. Algorithms that do not do well in this function will perform poorly on noisy data.

- f_5 is a foxholes function that has many local minima (25 in this case). Many standard optimisation algorithms get stuck in the first maximum they find.

- f_6 is a Rastrigin function, which is similar to the foxholes function. However, this function has even more local minima.

It should be noted that the minimum values of all functions in the defined domain are zero except for $f_5(\mathbf{x})$. The fitness functions for f_1 to f_4 and f_6 are defined as,

$$fitness = \frac{1}{1+f_i(\mathbf{x})},\ i = 1, 2, 3, 4, 6. \tag{4.27}$$

and the fitness function for f_5 is defined as,

$$fitness = \frac{1}{f_5(\mathbf{x})} \tag{4.28}$$

The proposed fuzzy GA goes through these six test functions. The results are compared with those obtained by the traditional GA with arithmetic crossover and non-uniform mutation [Michalewicz 94]. For each test function, the simulation takes 500 iterations and the population size is 10. The probability of crossover is set at 0.8 for all functions, and the probabilities of mutation for functions f_1 to f_6 are 0.8, 0.8, 0.7, 0.8, 0.8 and 0.35 respectively. The shape parameters b of the traditional GA for non-uniform mutation are set at b = 5 for function f_1, f_2 and f_5; b = 1 for function f_4 and f_6; b = 0.1 for function f_3. For the proposed fuzzy GA, the probability of acceptance r_r is set at 0.1 for all functions, and the probabilities of mutation p_m for functions f_1 to f_6 are 0.5, 0.8, 0.7, 0.8, 0.8, and 0.35 respectively. The probabilities of the genetic operators (r_r and p_m) are selected by trial and error. Fig. 4.2 and Fig. 4.3 show the membership functions for the fuzzy crossover and fuzzy mutation respectively. Bell-shaped membership functions are chosen. Considering the fuzzy rules for genetic operators as discussed in sub-section 4.2.1, the rule tables are designed and shown in Fig. 4.4. From these two rule tables, the output singleton values for the fuzzy crossover are 0 for L, 0.5 for M and 1.0 for H. The output singleton values for the fuzzy mutation are 0.2 for L, 0.7 for M and 1.0 for H. All the initial genes, i.e. the initial values of **x**, in the population for a test function are set to be the same. For tests 1 to 6, the initial values are $[1 \quad 1 \quad 1]$, $[0.5 \quad 0.5]$, $[1 \quad \cdots \quad 1]$, $[0.5 \quad \cdots \quad 0.5]$, $[10 \quad \cdots \quad 10]$ and $[1 \quad 1 \quad 1]$

respectively. The results of the average fitness values over 100 times of simulations (500 iterations each) based on the proposed and traditional GAs are shown in Fig. 4.5 and tabulated in Table 4.1. It can be seen that the performance of the proposed fuzzy GA is better than that of the traditional GA.

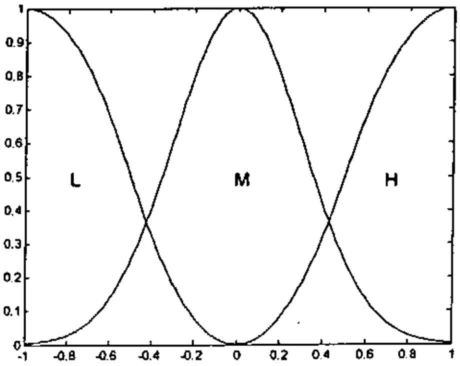


Fig. 4.2. Membership functions for fuzzy crossover operation (x-axis: $f(\mathbf{p}_1)-f(\mathbf{p}_2)$, y-axis: $\mu_M(f(\mathbf{p}_1)-f(\mathbf{p}_2))$).

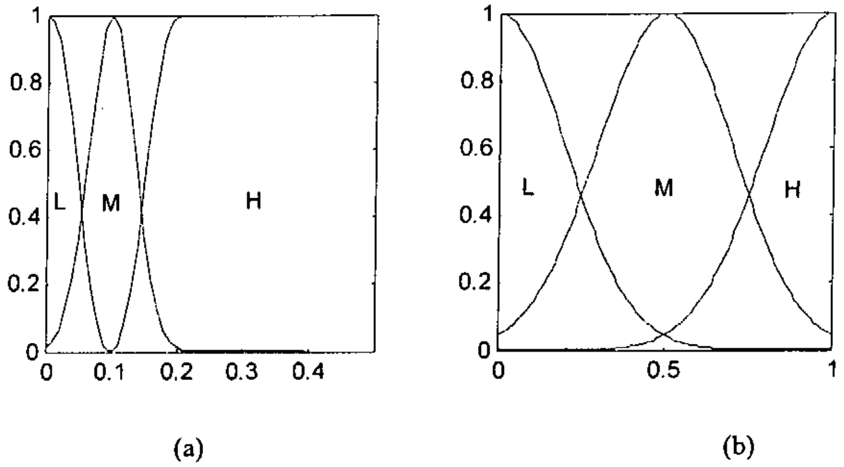


Fig. 4.3. Membership functions for fuzzy mutation operation: (a) (x-axis: $\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|$, y-axis: $\mu_{N_1}\left(\left|\frac{\partial f(\mathbf{o}_s)}{\partial o_{s_k}}\right|\right)$) and (b) ($x$-axis: $\frac{\tau}{T}$, y-axis: $\mu_{N_2}\left(\frac{\tau}{T}\right)$).

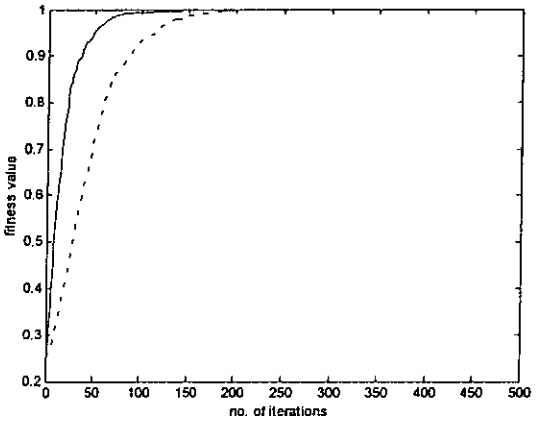
$f(p_1)-f(p_2)$	L	M	H
w_a	L	M	H

(a)

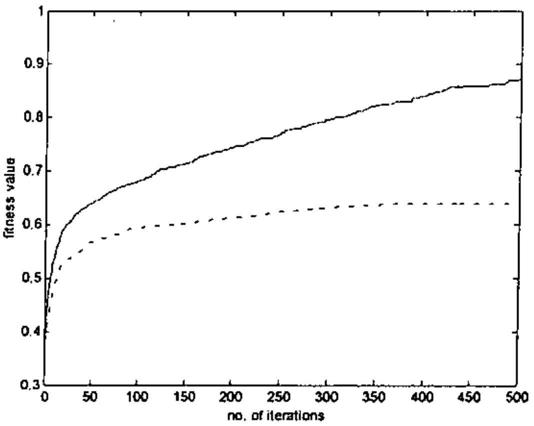
w_s $\frac{\partial f(o_s)}{\partial o_s}$ $\frac{t}{T}$	L	M	H
L	H	H	H
M	M	H	H
H	L	M	H

(b)

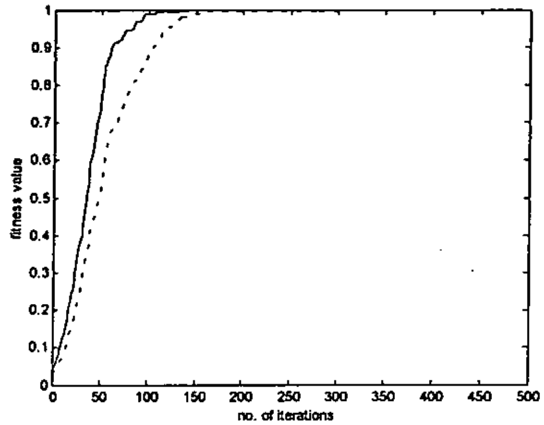
Fig. 4.4. Fuzzy rule tables: (a) for fuzzy crossover (b) for fuzzy mutation.



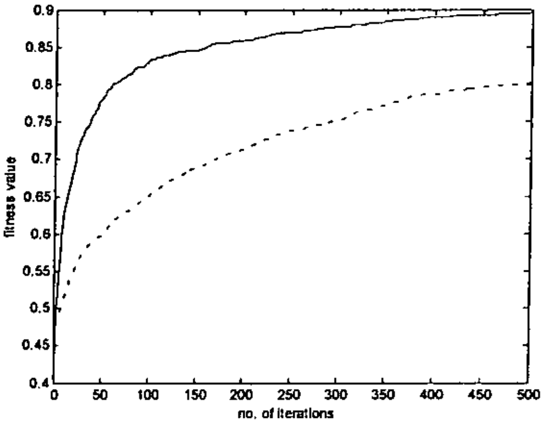
(a) The averaged fitness value of the test function $f_1(x)$ obtained by the proposed fuzzy GA (solid line) and traditional GA (dotted line).



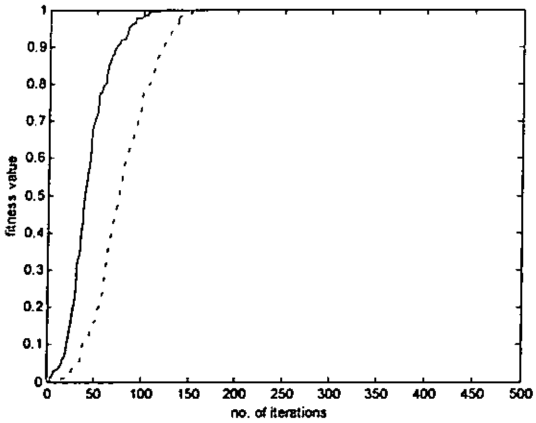
(b) The averaged fitness value of the test function $f_2(x)$ obtained by the proposed fuzzy GA (solid line) and traditional GA (dotted line).



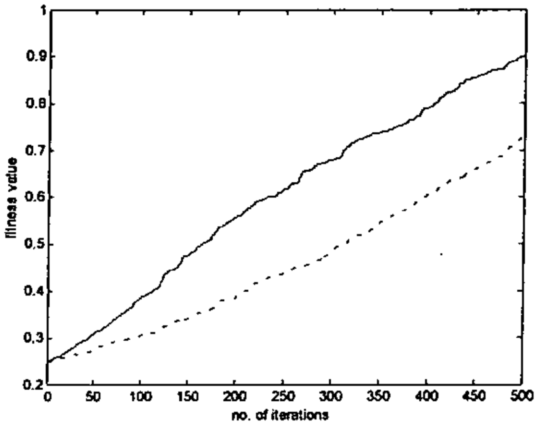
(c) The averaged fitness value of the test function $f_3(x)$ obtained by the proposed fuzzy GA (solid line) and traditional GA (dotted line).



(d) The averaged fitness value of the test function $f_4(x)$ obtained by the proposed fuzzy GA (solid line) and traditional GA (dotted line).



(e) The averaged fitness value of the test function $f_5(x)$ obtained by the proposed fuzzy GA (solid line) and traditional GA (dotted line).



(f) The averaged fitness value of the test function $f_6(\mathbf{x})$ obtained by the proposed fuzzy GA (solid line) and traditional GA (dotted line).

Fig. 4.5. Simulation results of the proposed and traditional GAs.

Test function	Proposed Fuzzy GA	Traditional GA
$f_1(\mathbf{x})$	1.0000	1.0000
$f_2(\mathbf{x})$	0.8724	0.6393
$f_3(\mathbf{x})$	1.0000	1.0000
$f_4(\mathbf{x})$	0.8956	0.8037
$f_5(\mathbf{x})$	1.0000	1.0000
$f_6(\mathbf{x})$	0.8989	0.7297

Table 4.1. Average fitness values obtained from the proposed fuzzy GA and the traditional GA for the benchmark test functions.

4.3 Electric Load Forecasting with Fuzzy GA-Based Neural Network

Electric load forecasting with a fuzzy GA-based neural network will be reported in this section. Thanks to its specific structure, neural networks can realize learning [Rumelhart 86] by executing some algorithms such as GA [Michalewicz 94] and back-propagation [Rumelhart 86]. Usually, the structure of a neural network is fixed for a learning process. However, a fixed structure may not provide the best performance within a given training period. Some network links could be redundant. If the network structure is too complicated, the training period will be long and the implementation cost will be high.

In this section, a three-layer neural network with a switch introduced in each link is proposed to facilitate the tuning of the optimal network structure. Fuzzy GA (as discussed in Section 4.2) is used to help tuning the structure as well as the parameters of the proposed neural network. The proposed neural network is then used to forecast the daily electric load in an intelligent home. Simulation results will be given to illustrate the performance of the proposed neural network.

4.3.1 Fuzzy GA Based Neural Network with Link Switches

A neural network with link switches is to be presented. The tuning of the network parameters and structure using the fuzzy GA will also be discussed.

4.3.1.1. Neural network with link switches

The proposed three-layer network is shown in Fig. 4.6. Specifically, a unit step function is introduced to each link. This unit step function is defined as,

$$\delta(\alpha) = \begin{cases} 0 \text{ if } \alpha < 0 \\ 1 \text{ if } \alpha \geq 0 \end{cases}, \ \alpha \in \Re \tag{4.29}$$

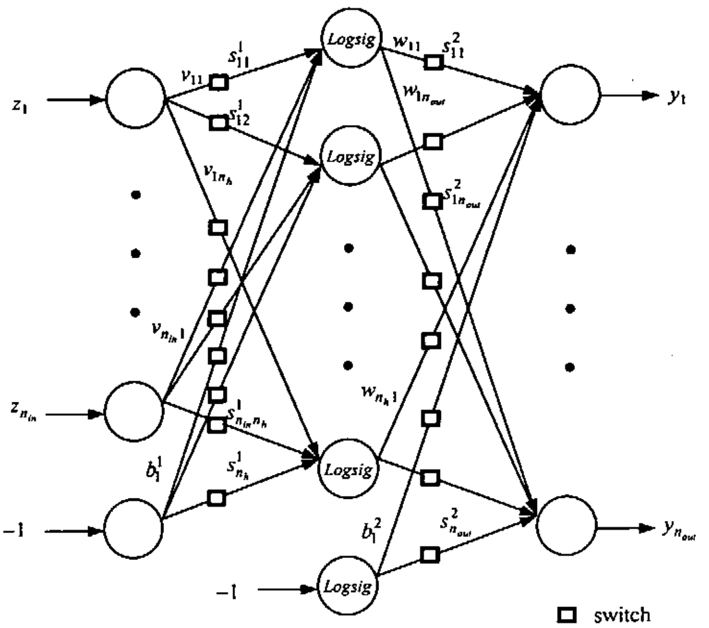


Fig. 4.6. Proposed three-layer neural network.

The introduction of the step function is equivalent to adding a switch to each link of the neural network. Referring to Fig. 4.6, the input-output relationship of the proposed multiple-input-multiple-output three-layer neural network is given by,

$$y_k(t) = \sum_{j=1}^{n_h} \delta(s^2_{jk}) w_{jk} logsig\left[\sum_{i=1}^{n_{in}} \left(\delta(s^1_{ij}) v_{ij} z_i(t) - \delta(s^1_j) b^1_j\right)\right] - \delta(s^2_k) b^2_k,\ k = 1, 2, \ldots, n_{out} \tag{4.30}$$

where $z_i(t)$, $i = 1, 2, \ldots, n_{in}$, are the inputs which are functions of a variable t; n_{in} denotes the number of inputs; v_{ij}, $j = 1, 2, \ldots, n_h$, denotes the weight of the link between the i-th input node and the j-th hidden node, n_h denotes the number of hidden nodes; w_{jk}, $k = 1, 2, \ldots, n_{out}$, denotes the weight of the link between the j-th hidden node and the k-th output node, n_{out} denotes the number of outputs; s_{ij}^1 denotes the parameter of the link switch from the i-th input to the j-th hidden node; s_{jk}^2 denotes the parameter of the link switch from the j-th hidden node to the k-th output node; b_j^1 and b_k^2 denote the biases for the hidden nodes and output nodes respectively; s_j^1 and s_k^2 denote the parameters of the link switches of the biases to the hidden and output layers respectively; $logsig(\cdot)$ denotes the logarithmic sigmoid function:

$$logsig(\alpha) = \frac{1}{1+e^{-\alpha}}, \ \alpha \in \Re \qquad (4.31)$$

$y_k(t)$, $k = 1, 2, \ldots, n_{out}$, is the k-th output of the proposed neural network. The weights v_{ij} and the switch states are to be tuned. It can be seen that the weights of the links govern the input-output relationship of the neural network while the switches of the links govern the structure of the neural network.

4.3.1.2 Tuning

In this sub-section, the proposed neural network is employed to learn the input-output relationship of an application using the proposed fuzzy GA. The input-output relationship is described by,

$$\mathbf{y}^d(t) = \mathbf{g}\left(\mathbf{z}^d(t)\right),\ t = 1, 2, \ldots, n_d \tag{4.32}$$

where $\mathbf{y}^d(t) = \left[y_1^d(t) \quad y_2^d(t) \quad \cdots \quad y_{n_{out}}^d(t)\right]$ is the desired output corresponding to the input $\mathbf{z}^d(t) = \left[z_1^d(t) \quad z_2^d(t) \quad \cdots \quad z_{n_{In}}^d(t)\right]$ of an unknown nonlinear function $\mathbf{g}(\cdot)$; n_d denotes the number of input-output data pairs. The fitness function is defined as,

$$fitness = \frac{1}{1 + err} \tag{4.33}$$

$$err = \frac{1}{n_d}\sum_{t=1}^{n_d}\frac{1}{n_{out}}\sum_{k=1}^{n_{out}}\frac{\left|y_k^d(t) - y_k(t)\right|}{y_k^d(t)} \tag{4.34}$$

The objective is to minimize the mean absolute percentage error (MAPE) of (4.34) using GA by setting the chromosome to be $\left[s_{jk}^2 \quad w_{jk} \quad s_{ij}^1 \quad v_{ij} \quad s_j^1 \quad b_j^1 \quad s_k^2 \quad b_k^2\right]$ for all i, j, k. The range of (4.33) is from 0 to 1. A larger value of the fitness function indicates a smaller value of (4.34).

4.3.2 Short-Term Daily Electric Load Forecasting System

The idea of daily load forecasting is to construct seven multi-input multi-output neural networks, one for each day of a week. Each neural network has 24 outputs representing the expected hourly load for a day. One important job in designing the forecasting system is the selection of input variables. In this electric load forecasting system, there are three main kinds of input variables:

1. Historical data of loads: hourly loads of the previous day were collected and used as historical load inputs. The historical load data reflect the habit of the

family on power consumption.

2. Temperature inputs: the average temperatures of the previous day and the present day are used as two inputs in this forecasting system. The value of the average temperature of the present day is got from the temperature forecast of the weather observatory.

3. Rainfall index inputs: the average rainfall indexes of the previous day and the present day are used as two inputs in this forecasting system. The range of the rainfall index is from 0 to 1. 0 represents no rain and 1 represents heavy rain.

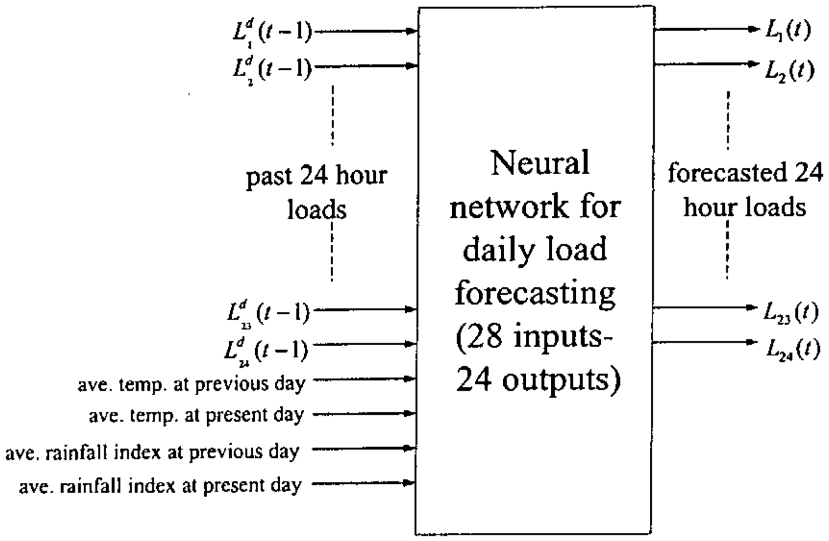

$L_1^d(t-1)$
$L_2^d(t-1)$
past 24 hour loads
$L_{23}^d(t-1)$
$L_{24}^d(t-1)$
ave. temp. at previous day
ave. temp. at present day
ave. rainfall index at previous day
ave. rainfall index at present day
Neural network for daily load forecasting (28 inputs-24 outputs)
$L_1(t)$
$L_2(t)$
forecasted 24 hour loads
$L_{23}(t)$
$L_{24}(t)$

Fig. 4.7. Proposed neural network for daily load forecasting.

A diagram of one of the seven NNs for the daily load forecasting is shown in Fig. 4.7. Each neural network has 28 inputs, 24 outputs with link switches. Among the 28 inputs nodes, the first 24 input nodes (z_1, ..., z_{24}) represent the previous 24 hourly loads [Momoh 97] and are denoted by $z_i = L_i^d(t-1)$, i = 1, 2, ..., 24. Node 25 (z_{25}) and node

26 (z_{26}) represent the average temperatures of the previous day and the present day respectively. Node 27 (z_{27}) and node 28 (z_{28}) represent the average rainfall indexes at the previous day and the present day respectively. The output layer consists of 24 output nodes that represent the forecasted 24 hourly loads of a day, and are denoted by $y_k(t) = L_i(t)$, $i = 1, 2, \ldots, 24$. Such a network structure is chosen based on the assumption that the consumption patterns of the seven days within a week would differ significantly among each other, while the patterns among the same day of weeks are similar. By using the past 24 hourly loads as the inputs, the relationship between a given hour's load and the 24 hourly loads of the pervious day can be considered. The accuracy of the forecasting can then be increased. The first-time off-line training is a time consuming process. However, once trained, the system can do the forecast quickly. In this example, 12 sets of historical data are used for off-line training with 1000 iterations. After the first-time off-line training, the forecasting system can operate and continue to be off-line trained daily using updated data. During the daily off-line training, the system will update the weights of the neural network with 200 iterations. From (4.30), the proposed neural network used for the daily load forecasting is governed by,

$$y_k(t) = \sum_{j=1}^{n_h} \delta(s_{jk}^2) w_{jk} logsig\left[\sum_{i=1}^{28} \left(\delta(s_{ij}^1) v_{ij} z_i(t) - \delta(s_j^1) b_j^1\right)\right] - \delta(s_k^2) b_k^2, \quad k = 1, 2, \ldots, 24. \quad (4.35)$$

The number of hidden nodes (n_h) is changed from 8 to 13 in order to compare the learning performance. The fitness function is defined as follows,

$$fitness = \frac{1}{1 + err} \quad (4.36)$$

$$err = \frac{1}{12}\sum_{t=1}^{12}\frac{1}{24}\sum_{k=1}^{24}\frac{\left|y_k^d(t) - y_k(t)\right|}{y_k^d(t)} \tag{4.37}$$

The proposed fuzzy GA is employed to tune the parameters and structure of the neural network of (4.35). The objective is to maximize the fitness function of (4.36). The best fitness value is 1 and the worst value is 0. The population size used for the GA is 10. The lower and the upper bounds of the link weights are defined as $\frac{-3}{\sqrt{n_{in}+1}} \geq v_{ij}, w_{jk}, b_j^1, b_1^2 \geq \frac{3}{\sqrt{n_h+1}}$ and $-1 \geq s_{j1}^2, s_{ij}^1, s_j^1, s_1^2 \geq 1$, i = 1, 2, …, n_{in} ; j = 1, 2, …, n_h , k = 1, 2, …, 24 [Brown 94]. The chromosomes used for the GA are $\begin{bmatrix} s_{j1}^2 & w_{jk} & s_{ij}^1 & v_{ij} & s_j^1 & b_j^1 & s_k^2 & b_1^2 \end{bmatrix}$ for all i, j, k. The initial values of the link weights are set at $\frac{1}{n_h}$. For comparison purpose, the proposed neural network trained by the traditional GA with arithmetic crossover and non-uniform mutation [Michalewicz 94], a traditional neural network trained by the proposed fuzzy GA, and a traditional neural network trained by the traditional GA, are also applied in electric load forecasting. The working conditions are kept the same. For the traditional GA with arithmetic crossover and non-uniform mutation, the probability of crossover and mutation are set at 0.8 and 0.01 respectively, and the shape parameter b of the non-uniform mutation operation is set at 5. For the proposed fuzzy GA, the probabilities of mutation p_m and acceptance r_r are 0.01 and 0.1 respectively. The learning results of the daily electric load forecasting systems using the proposed neural network trained by the fuzzy GA and the traditional GA on Wednesday and Sunday are tabulated in Table 4.2 and Table 4.3 respectively. These tables show the fitness values and the percentages of reduction of the number of links of the neural network. From these two tables, the best result is obtained when the number of hidden node n_h is equal to 12. From Table 4.2, n_h=12, the number of links being reduced is 69 after training. The number of links of a fully connected network is 660. It is about a 10.45% reduction. Fig. 4.8 and Fig. 4.9 show the simulation results when the proposed fuzzy GA (solid lines) and the traditional GA (dotted lines) are used respectively. From these two figures, we can see that the time of convergence of the proposed fuzzy GA is faster. The learning results of the daily electric load forecasting systems using a traditional neural network trained by the fuzzy GA and the traditional GA on Wednesday and Sunday are tabulated in Table 4.4 and Table 4.5 respectively. Comparing the results in Table 4.2 to Table 4.5, the proposed fuzzy GA-based neural network gives the best result.

Once the first-time off-line training is done, the forecasting system is put to daily operation. The system will then update the weights of the neural network daily with 200 iterations. The training error and the forecasting error in term of MAPE from Monday to Sunday under n_h=12 are shown in Table 4.6. This table shows the off-line training error and the forecasting error for week 13 to week 15. The average errors of training and forecasting are 1.6572 and 1.9878 respectively. Fig. 4.10 and Fig. 4.11 show the simulation results of the daily electric load forecasting on Wednesday and Sunday respectively using the proposed network. In Fig. 4.10 and Fig. 4.11, the dashed line represents the forecasted result and the solid line is the actual load.

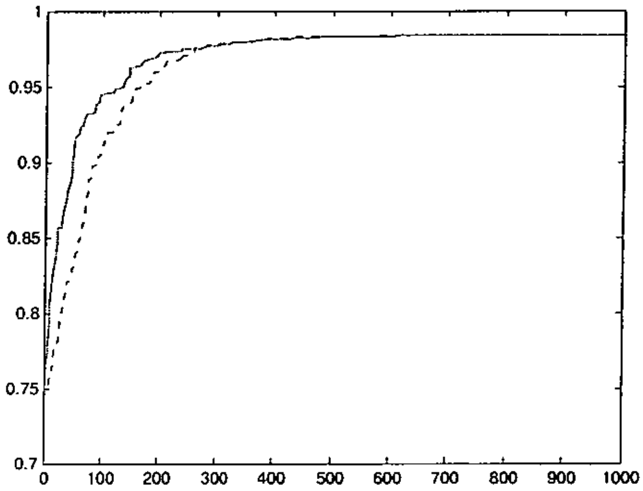


Fig. 4.8. Simulation results of training the daily electric load forecasting system on Wednesday with traditional GA (dotted line) and the proposed fuzzy GA (solid line) for 1000 iterations at $n_h = 12$.

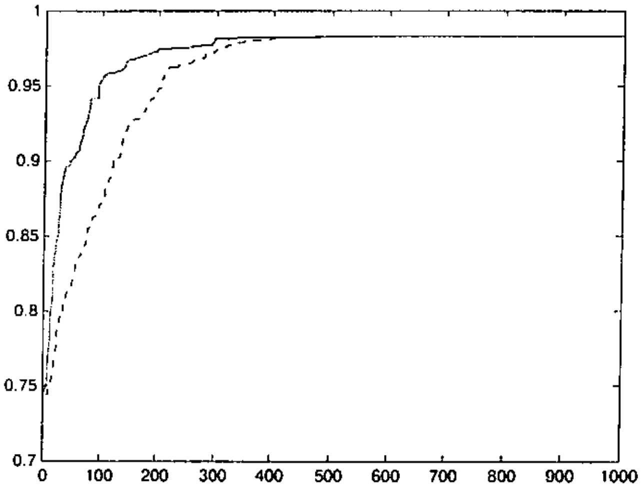


Fig. 4.9. Simulation results of training the daily electric load forecasting system on Sunday with traditional GA (dotted line) and the proposed fuzzy GA (solid line) for 1000 iterations at $n_h = 12$.

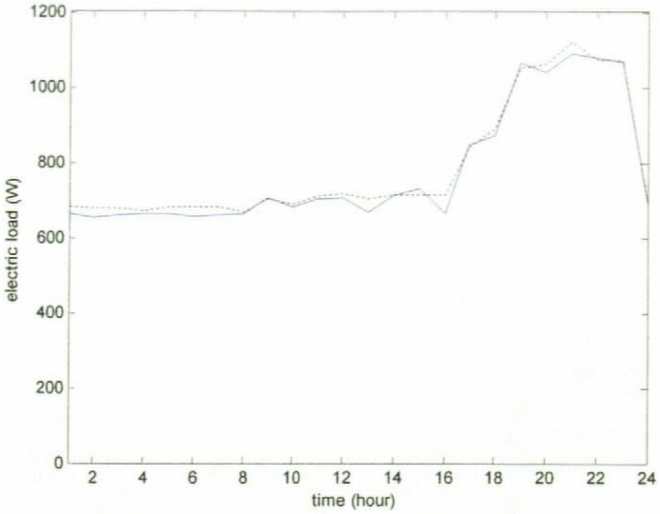


Fig. 4.10. Load forecasting result for Wednesday (Week13) with the proposed network (dashed line), and the actual load (solid line).

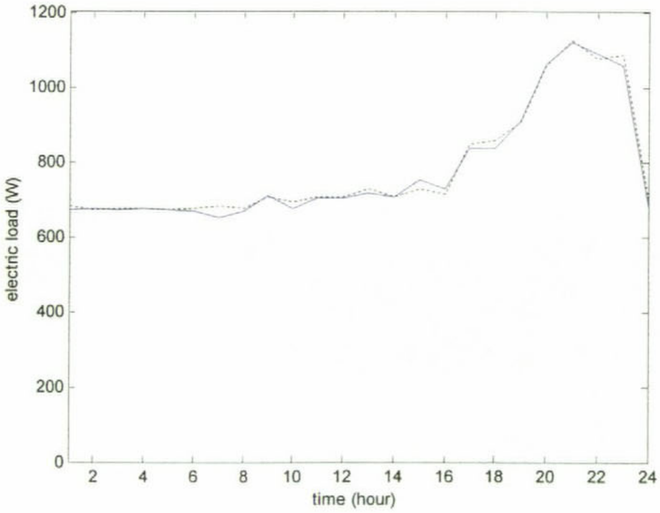


Fig. 4.11. Load forecasting result for Sunday (Week13) with the proposed network (dashed line), and the actual load (solid line).

	Fuzzy GA			Traditional GA		
n_h	Fitness Values	Training error (MAPE)	Ratio of the number of links being reduced to the total number of links (Percentage of reduction)	Fitness Values	Training error (MAPE)	Ratio of the number of links being reduced to the total number of links (Percentage of reduction)
9	0.9834	1.6880	22/501 (4.3%)	0.9832	1.7087	33/501 (6.59%)
10	0.9840	1.6260	30/554 (5.42%)	0.9831	1.7191	29/554 (5.03%)
11	0.9829	1.7397	43/607 (7.08%)	0.9791	2.1346	33/607 (5.44%)
12	0.9842	1.6054	69/660 (10.45%)	0.9835	1.6777	25/660 (3.79%)
13	0.9837	1.6570	30/713 (4.21%)	0.9823	1.8019	27/713 (3.79%)

Table 4.2. Learning results of the daily electric load forecasting system with the proposed neural network for Wednesday.

	Fuzzy GA			Traditional GA		
n_h	Fitness Values	Training error (MAPE)	Ratio of the number of links being reduced to the total number of links (Percentage of reduction)	Fitness Values	Training error (MAPE)	Ratio of the number of links being reduced to the total number of links (Percentage of reduction)
9	0.9825	1.7812	40/501 (7.98%)	0.9794	2.1033	29/501 (5.79%)
10	0.9827	1.7605	37/554 (6.68%)	0.9781	2.2390	22/554 (3.97%)
11	0.9826	1.7708	39/607 (6.43%)	0.9820	1.8330	34/607 (5.60%)
12	0.9833	1.6984	35/660 (5.30%)	0.9825	1.7812	22/660 (3.33%)
13	0.9822	1.8123	40/713 (5.61%)	0.9819	1.8434	32/713 (4.49%)

Table 4.3. Learning results of the daily electric load forecasting system with the proposed neural network for Sunday.

	Fuzzy GA		Traditional GA	
n_h	Fitness Values	Training error (MAPE)	Fitness Values	Training error (MAPE)
9	0.9830	1.7294	0.9793	2.1138
10	0.9833	1.6984	0.9823	1.8019
11	0.9835	1.6777	0.9811	1.9264
12	0.9838	1.6467	0.9831	1.7191
13	0.9827	1.7605	0.9826	1.7708

Table 4.4. Learning results of the daily electric load forecasting system with a traditional neural network for Wednesday.

	Fuzzy GA		Traditional GA	
n_h	Fitness Values	Training error (MAPE)	Fitness Values	Training error (MAPE)
9	0.9810	1.9368	0.9781	2.2390
10	0.9813	1.9056	0.9780	2.2495
11	0.9824	1.7915	0.9821	1.8226
12	0.9826	1.7708	0.9823	1.8019
13	0.9815	1.8849	0.9808	1.9576

Table 4.5. Learning results of the daily electric load forecasting system with a traditional neural network for Sunday.

	First-time off-line training error for Week 1-Week 12	Forecasting error at Week 13	Daily off-line training error for Week 2-Week 13	Forecasting error at Week 14	Daily off-line training error for Week 3-Week 14	Forecasting error at Week 15
Monday	1.6502	2.4174	1.6198	1.3578	1.4907	1.9942
Tuesday	1.7321	1.9180	1.6858	1.6747	1.6646	2.2151
Wednesday	1.6075	2.0265	1.6347	2.0004	1.6621	2.4037
Thursday	1.7012	1.6767	1.6486	2.1213	1.7119	1.5012
Friday	1.7177	2.1913	1.6718	2.4073	1.8705	2.0092
Saturday	1.6717	1.6646	1.5506	1.9754	1.5779	2.2299
Sunday	1.7094	1.4369	1.6065	2.2732	1.5854	2.2492

Table 4.6. Off-line training error and forecasting error in terms of MAPE from Monday to Sunday.

4.4 ELECTRIC LOAD FORECASTING WITH FUZZY GA-BASED NEURAL FUZZY NETWORK

Electric load forecasting using a fuzzy GA-based neural fuzzy network will be reported in this section. By introducing switches in the links of the neural fuzzy network (NFN), the optimal network structure can be found by the proposed fuzzy GA (as discussed in Section 4.2.) The membership functions and the number of rules of the NFN can be generated automatically. This implies that the cost of implementing the proposed NFN can be reduced. The proposed NFN will be presented in sub-section 4.4.1. A short-term electric load forecasting realized by the proposed NFN tuned by the proposed fuzzy GA will be presented in sub-section 4.4.2. Simulation results will be given.

4.4.1 Fuzzy GA-Based Neural Fuzzy Network with Rule Switches

In this section, a neural fuzzy network with link switches will be presented. The tuning of the network parameters and structure is done by the proposed fuzzy GA. The optimal number of rules can be found by introducing switches to the fuzzy rules, which are realized as switches in some links of the NFN.

4.4.1.1 Neural fuzzy network with rule switches

A fuzzy associative memory (FAM) [Kosko 91] rule base is adopted. An FAM is formed by partitioning the universe of discourse of each fuzzy variable according to the level of fuzzy resolution chosen for the antecedents. A grid of FAM elements is then generated. The entry at each grid element in the FAM corresponds to a fuzzy premise. The FAM is therefore a geometric or tabular representation of a fuzzy logic rule base. For an NFN, the number of possible rules may be too large. This makes the network complex while some rules may be unnecessary. The implementation cost is also unnecessarily high. Thus, a multi-input-single-output NFN is proposed which can have an optimal number of rules and membership functions. The main difference between the proposed network and the traditional network is that a unit step function is introduced to each rule. The unit step function is defined as,

$$\delta(\varsigma) = \begin{cases} 0 \text{ if } \varsigma \le 0 \\ 1 \text{ if } \varsigma > 0 \end{cases}, \varsigma \in \Re \tag{4.38}$$

This is equivalent to adding a switch to each rule in the NFN. Referring to Fig. 4.11, the input and output variables of the NFN are z_i and y respectively; where i = 1, 2, ..., n and n is the number of input variables. The behaviour of the NFN is governed by p fuzzy rules in the following format:

R_g: IF $z_1(t)$ is $A_1^{g_1}(z_1(t))$ AND $z_2(t)$ is $A_2^{g_2}(x_2(t))$ AND ... AND $z_n(t)$ is $A_n^{g_n}(z_n(t))$ THEN $y(t)$ is w_g, t = 1, 2, ..., u (4.39)

where u denotes the number of input-output data pairs; g = 1, 2, ..., p, is the rule number; w_g is the output singleton of rule g. From Fig. 4.11, it can be seen that

$$p = \prod_{i=1}^{n} m_i \tag{4.40}$$

where m_i is the number of membership functions of the input variable z_i and $g_i \in [1,\ldots,m_i]$, $i = 1, \ldots, n$.

In this network, the membership function is a bell-shaped function as given by,

$$\mu_{A_i^{g_i}}\left(z_i(t)\right) = e^{\frac{-(z_i(t)-\bar{z}_i^{g_i})^2}{2\sigma_i^{g_i 2}}} \tag{4.41}$$

where the parameter $\bar{z}_i^{g_i}$ and $\sigma_i^{g_i}$ are the mean value and the standard deviation of the membership function $\mu_{A_i^{g_i}}$ respectively. The grade of membership of each rule is defined as,

$$\mu_g(t) = \mu_{A_1^{g_1}}(z_1(t)) \cdot \mu_{A_2^{g_2}}(z_2(t)) \cdot \ldots \cdot \mu_{A_n^{g_n}}(z_n(t)) \tag{4.42}$$

The output of the neural fuzzy network $y(t)$ is defined as,

$$y(t) = \frac{\sum_{g=1}^{p} \mu_g(t) w_g \delta\left(\varsigma_g\right)}{\sum_{g=1}^{p} \mu_g(t)} \tag{4.43}$$

where ς_g denotes the rule switch parameter of the g-th rule.

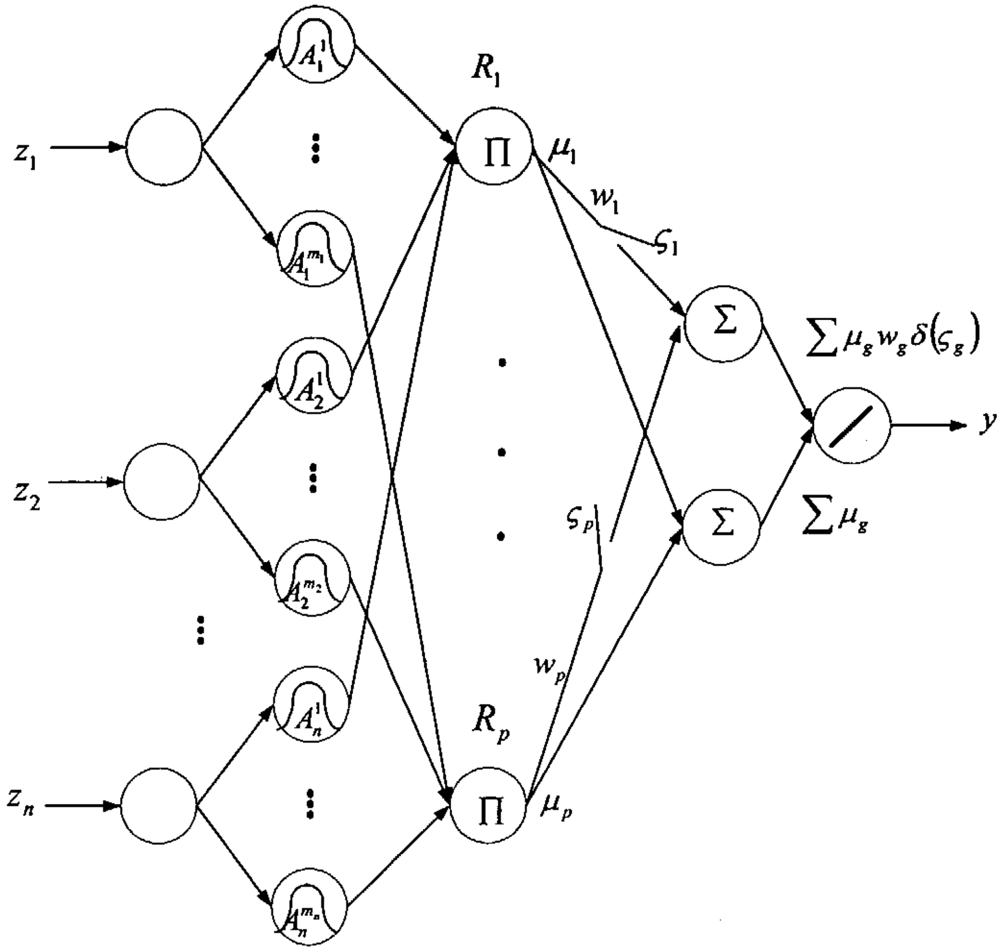


Fig. 4.11. Proposed neural fuzzy network.

4.4.1.2. Tuning

The proposed fuzzy GA can be employed to learn the multi-input-single-output relationship of the proposed NFN in an application. The desired input-output relationship is described by,

$$y^d(t) = q(\mathbf{z}^d(t)),\ t = 1, 2, \ldots, u \tag{4.44}$$

where $y^d(t)$ is the desired output, $\mathbf{z}^d(t) = \begin{bmatrix} z_1^d(t) & z_2^d(t) & \cdots & z_v^d(t) \end{bmatrix}$ is the desired input vector and $q(\cdot)$ is an unknown non-linear function. The fitness function is defined as,

$$fitness=\frac{1}{1+err} \tag{4.45}$$

where

$$err=\frac{1}{u}\sum_{t=1}^{u}\frac{\left|y^{d}(t)-y(t)\right|}{y^{d}(t)} \tag{4.46}$$

The objective is to minimize the value of (4.46) using the proposed fuzzy GA by setting the chromosome to be [$\bar{z}_i^{g_i}$ $\sigma_i^{g_i}$ ς_g] for all i, g_i, g. The value of (4.46) is the mean absolute percentage error (MAPE). The range of *fitness* in (4.45) is [0, 1]. A larger value of *fitness* indicates a smaller *err*. By using the proposed neural fuzzy network and the proposed fuzzy GA, an optimal neural fuzzy network in terms of the number of rules and the membership functions can be obtained.

4.4.2 Short-Term Electric Load Forecasting System

It is desired to forecast the load demand in a home with respect to the week's day type number and the hour number. The load forecasting system involves 168 multi-input-single-output NFNs, one for a given hour number in a week's day type number ($7\times 24=168$). If the network structure described in Section 4.3, which has 28 inputs and 24 outputs, is applied to this NFN, the system will become complex because the number of rules is large. Thus, 168 NFNs with seven inputs and one output per network are used. With a larger number of NFNs, a smaller number of rules in each NFN can be used. One important task in the short-term load-forecasting problem is to select the input variables. In this system, the input variables are historical load data, temperature inputs and rainfall index inputs. One of the 168 NFNs for daily load forecasting is shown in Fig. 4.12. It is a 7-input-1-output network with rule switches. The inputs, z_i, of the proposed NFN are: $z_1 = L^d(d-1,h-1)$, which represents the load value at the previous hour of the previous day; $z_2 = L^d(d-1,h)$, which represents the load value at the forecasting hour of the previous day; $z_3 = L^d(d-1,h+1)$, which represents the load value at the next hour of the previous day; z_4 = average temperature at the previous day, z_5 = average temperature at the present day, z_6 = average rainfall index at the previous day, z_7= average rainfall index at the present day. The output $y(t) = L(d,h)$, where d = 1, 2, …, 7 is the week's day type number (e.g. d = 1 for Monday, d = 7 for Sunday), h = 1, 2, …, 24 is the hour number. One should note the special case that if d = 1, (d – 1) should be 7. $L(d,h)$ is the forecasted load for day-d and hour-h.

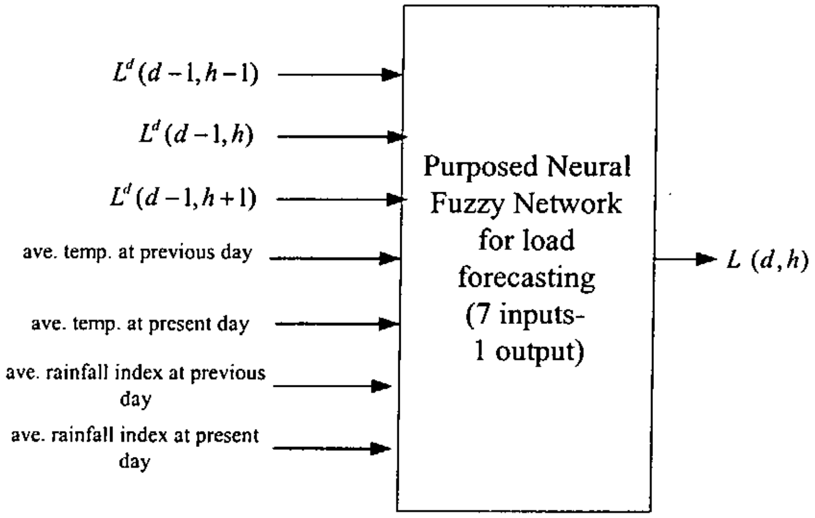

$L^d(d-1,h-1)$
$L^d(d-1,h)$
$L^d(d-1,h+1)$
ave. temp. at previous day
ave. temp. at present day
ave. rainfall index at previous day
ave. rainfall index at present day
Purposed Neural Fuzzy Network for load forecasting (7 inputs- 1 output)
$L(d,h)$

Fig. 4.12. Proposed neural fuzzy network for electric load forecasting.

Before putting the daily load forecasting system into operation, the NFN should be trained by sampled data. The first-time off-line training is a time consuming process. However, once trained, the NFN can forecast quickly. Comparing with the NN discussed in section 4.3, this NFN requires a smaller number of iterations in the first-time off-line training. For this example, data of 12 weeks (week 1 to week 12) for training and data of 2 weeks (week 13 to week 14) for testing are prepared. The number of membership function for each input variable is chosen to be 2 ($m_i = 2$, $i = 1, 2, \ldots, 7$) such that the number of rules is $p = 2^7 = 128$. Referring to (4.43), the proposed NFN used for the load forecasting of a particular hour is governed by,

$$y(t) = \frac{\sum_{g=1}^{128} \mu_g(t) w_g \delta(\varsigma_g)}{\sum_{g=1}^{128} \mu_g(t)} \tag{4.47}$$

The fitness function is defined as follows,

$$fitness = \frac{1}{1+err} \tag{4.48}$$

$$err = \frac{1}{12}\sum_{t=1}^{12} \frac{\left|y^d(t) - y(t)\right|}{y^d(t)} \tag{4.49}$$

It should be noted that (4.47) describes one of the 168 NFNs in the proposed load forecaster.

The proposed fuzzy GA is employed to tune the parameters and structure of the NFN of (4.47). The population size is 10. The bounds of the parameters are set at $0 \le \bar{z}_i^{g_i} \le 1$, $0 \le \sigma_i^{g_i} \le 0.4$ and $-1 \le \varsigma_g \le 1$. The chromosomes used for the proposed fuzzy GA are [$\bar{z}_i^{g_i}$ $\sigma_i^{g_i}$ ς_g], $i = 1,\ldots,7;\ g_i = 1,2;\ g = 1,\ldots,128$. The initial values of $\bar{z}_i^{g_i}$ $\sigma_i^{g_i}$ ς_g are arbitrarily set at 0.5, 0.2 and 1 respectively. The probabilities of mutation p_m and acceptance r_r are set at 0.1 and 0.1 respectively. The number of iterations to train the NFN is 500. Once the first-time off-line training is done, the forecasting system will be put to daily operation. During the operation, the system will update the weights of the NFN with 100 iterations daily. For comparison, a 7-input-1-output NFN without rule switches trained by the traditional GA with arithmetic crossover and non-uniform mutation [Michalewicz 94] is also applied for the load forecasting. The probabilities of crossover and mutation are 0.8 and 0.03 respectively.

The results are tabulated in Table 4.7 and Table 4.8. Table 4.7 shows the simulation results of electric load forecasting for Wednesday and Table 4.8 shows the results for Sunday. From Table 4.7 and Table 4.8, we observe that the proposed NFN trained by the proposed fuzzy GA provides better results than those of the traditional NFN trained by the traditional GA in terms of fitness value and number of rules. From Table 4.7 and Table 4.8, the average numbers of rules for the proposed NFNs are 67.29 and 70.25 respectively. They imply a 47.4% and 45.11% reduction in the number of rules after training.

Table 4.9 show the average training error (MAPE) and the average forecasting error (MAPE) for Monday to Sunday. Fig. 4.13 and Fig. 4.14 show the forecasted daily load curve on Wednesday (weekday) and Sunday respectively at Week 13. We can observe that the proposed NFN offers a satisfactory performance in load forecasting.

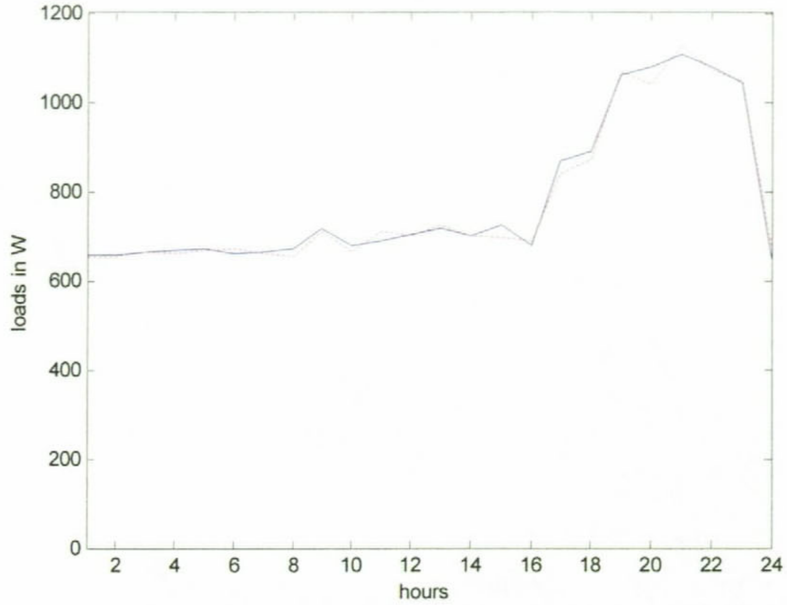


Fig. 4.13. Actual load (solid line) and forecast results for Wednesday (Week13) from the proposed forecasting system (dashed line).

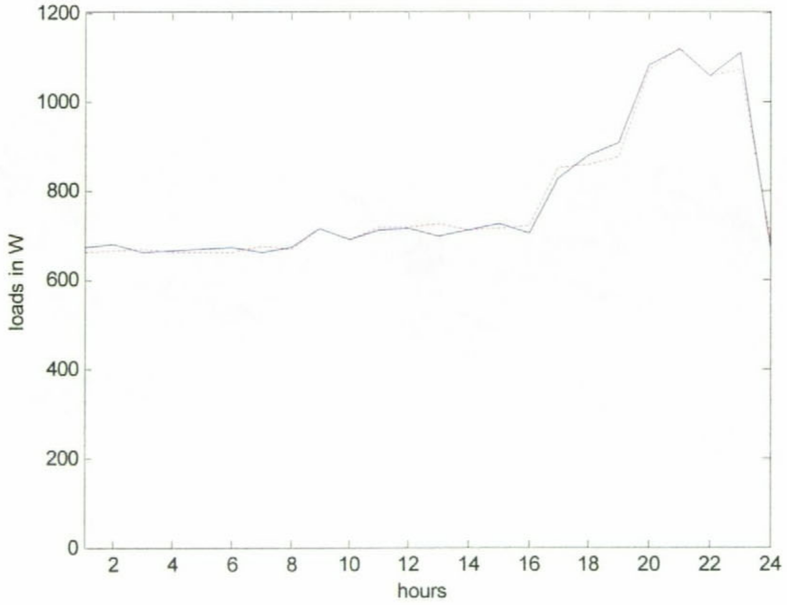


Fig. 4.14. Actual load (solid line) and forecast results for Sunday (Week13) from the proposed forecasting system (dashed line).

	Our Approach			Traditional Approach		
Hour	Fitness value	Training error (MAPE)	Number of rules	Fitness value	Training error (MAPE)	Number of rules
1	0.9943	0.5733	64	0.9895	1.0611	128
2	0.9946	0.5429	63	0.9879	1.2248	128
3	0.9952	0.4823	65	0.9812	1.9160	128
4	0.9883	1.1839	66	0.9808	1.9576	128
5	0.9926	0.7455	77	0.9838	1.6467	128
6	0.9925	0.7557	65	0.9841	1.6157	128
7	0.9945	0.5530	71	0.9793	2.1138	128
8	0.9827	1.7605	73	0.9831	1.7191	128
9	0.9894	1.0714	59	0.9803	2.0096	128
10	0.9853	1.4919	72	0.9801	2.0304	128
11	0.9885	1.1634	68	0.9875	1.2658	128
12	0.9889	1.1225	64	0.9792	2.1242	128
13	0.9835	1.6777	70	0.9751	2.5536	128
14	0.9856	1.4610	71	0.9848	1.5435	128
15	0.9844	1.5847	73	0.9781	2.2390	128
16	0.9813	1.9056	64	0.9733	2.7432	128
17	0.9857	1.4507	67	0.9776	2.2913	128
18	0.9889	1.1225	70	0.9829	1.7397	128
19	0.9801	2.0304	56	0.9764	2.4170	128
20	0.9827	1.7605	70	0.9783	2.2181	128
21	0.9851	1.5125	68	0.9850	1.5228	128
22	0.9899	1.0203	71	0.9887	1.1429	128
23	0.9791	2.1346	63	0.9801	2.0304	128
24	0.9834	1.6880	65	0.9791	2.1346	128
Average:	**0.9874**	**1.2761**	**67.29**	**0.9815**	**1.8849**	**128**

Table 4.7. Simulation results of electric load forecasting for Wednesday after training.

	Our Approach			Traditional Approach		
Hour	Fitness value	Training error (MAPE)	Number of rules	Fitness value	Training error (MAPE)	Number of rules
1	0.9941	0.5935	63	0.9918	0.8268	128
2	0.9903	0.9795	65	0.9911	0.8980	128
3	0.9889	1.1225	81	0.9792	2.1242	128
4	0.9960	0.4016	82	0.9817	1.8641	128
5	0.9928	0.7252	75	0.9887	1.1429	128
6	0.9897	1.0407	69	0.9836	1.6673	128
7	0.9906	0.9489	67	0.9889	1.1225	128
8	0.9825	1.7812	77	0.9800	2.0408	128
9	0.9890	1.1122	74	0.9871	1.3069	128
10	0.9811	1.9264	71	0.9773	2.3227	128
11	0.9837	1.6570	68	0.9825	1.7812	128
12	0.9895	1.0611	61	0.9798	2.0616	128
13	0.9801	2.0304	81	0.9733	2.7432	128
14	0.9781	2.2390	65	0.9748	2.5851	128
15	0.9835	1.6777	70	0.9781	2.2390	128
16	0.9791	2.1346	73	0.9720	2.8807	128
17	0.9775	2.3018	75	0.9717	2.9124	128
18	0.9821	1.8226	60	0.9721	2.8701	128
19	0.9779	2.2599	69	0.9751	2.5536	128
20	0.9885	1.1634	62	0.9788	2.1659	128
21	0.9921	0.7963	71	0.9808	1.9576	128
22	0.9884	1.1736	67	0.9824	1.7915	128
23	0.9802	2.0200	68	0.9819	1.8434	128
24	0.9863	1.3890	72	0.9823	1.8019	128
Average:	**0.9859**	**1.4302**	**70.25**	**0.9806**	**1.9784**	**128**

Table 4.8. Simulation results of electric load forecasting for Sunday after training.

	First-time off-line training error from Week 1-Week 12	Forecasting error at Week 13	Daily off-line training error from Week 2-Week 13	Forecasting error at Week 14	Daily off-line training error from Week 3-Week 14	Forecasting error at Week 15
Monday	1.3013	1.5211	1.2951	1.2059	1.2833	1.5037
Tuesday	1.3354	1.4893	1.3523	1.3215	1.3329	1.4988
Wednesday	1.2831	1.3522	1.2983	1.4021	1.2865	1.6129
Thursday	1.3152	1.3095	1.3028	1.4253	1.3107	1.2553
Friday	1.4257	1.4753	1.3851	1.4953	1.4055	1.5951
Saturday	1.4019	1.3992	1.2897	1.4059	1.3033	1.6803
Sunday	1.4316	1.3515	1.3503	1.5201	1.3012	1.4741

Table 4.9. Off-line training error and forecasting error in term of MAPE from Monday to Sunday.

4.5 ELECTRIC LOAD FORECASTING WITH MODIFIED FUZZY GA-BASED NEURAL NETWORK

Electric load forecasting with a modified fuzzy GA-based neural network will be reported in this section. A modified neural network model is to be proposed. Two different activation transfer functions are used in the neurons of the hidden layer, and a node-to-node relationship is proposed. This network model is found to be able to give a better performance with a smaller number of hidden nodes than the traditional feed-forward neural network [Lecun 85]. The proposed fuzzy GA (in Section 4.2) can help tuning the parameters of the proposed network. The tuned network is then used to forecast the short-term daily electric load in an intelligent home. Simulation results show that the proposed network can forecast the short-term daily electric load successfully. The proposed modified neural network will be presented in sub-section 4.5.1. Training of the neural network with the proposed fuzzy GA will be presented in sub-section 4.5.2. The short-term daily load forecasting will be presented in sub-section 4.5.3.

4.5.1 Neural Network Model

Fig. 4.15 shows the proposed neuron. It has two activation transfer functions to govern the input-output relationships of the neuron. The two activation transfer functions are called static activation transfer functions (SATF) and dynamic activation transfer functions (DATF). For the SATF, the parameters are fixed and its output depends on the input of the neuron. For the DATF, the parameters of the activation transfer function depend on the outputs of other neurons in the same layer and its SATF.

The modified neural network as shown in Fig. 4.16 is a 3-layer network. The first layer is the input layer, which simply distributes the input. The second layer is the hidden layer, and two different activation transfer functions are used in the neuron. The two different activation transfer functions are, namely, static and dynamic activation transfer functions (SATF and DATF). The SATF determines hyper-planes as switching surfaces. It can select convex regions in the input space for firing and/or forecasting. Owing to the DATF, the upper neighbor's output concerns the bias term while the lower neighbor's output influences the sharpness of the edges of the hyper-planes. They eventually combine into convex regions by the output layer.

By introduced this modified neural network, a node-to-node relationship is introduced in the hidden layer. Comparing with the traditional feed-forward neural network [Lecun 85], the proposed neural network can offer a better performance.

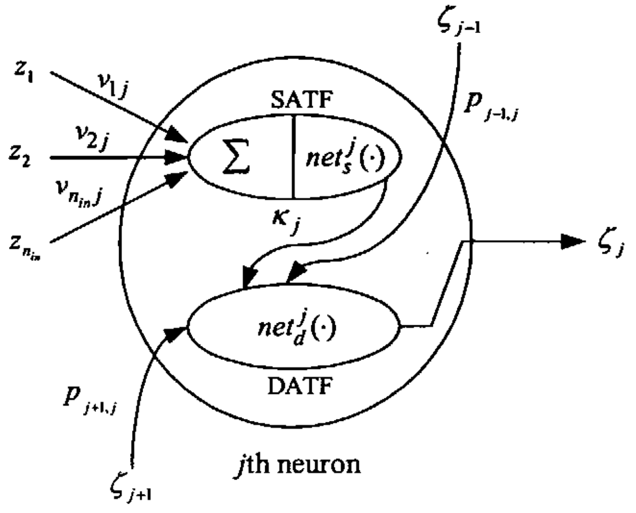


Fig. 4.15. Model of the proposed neuron.

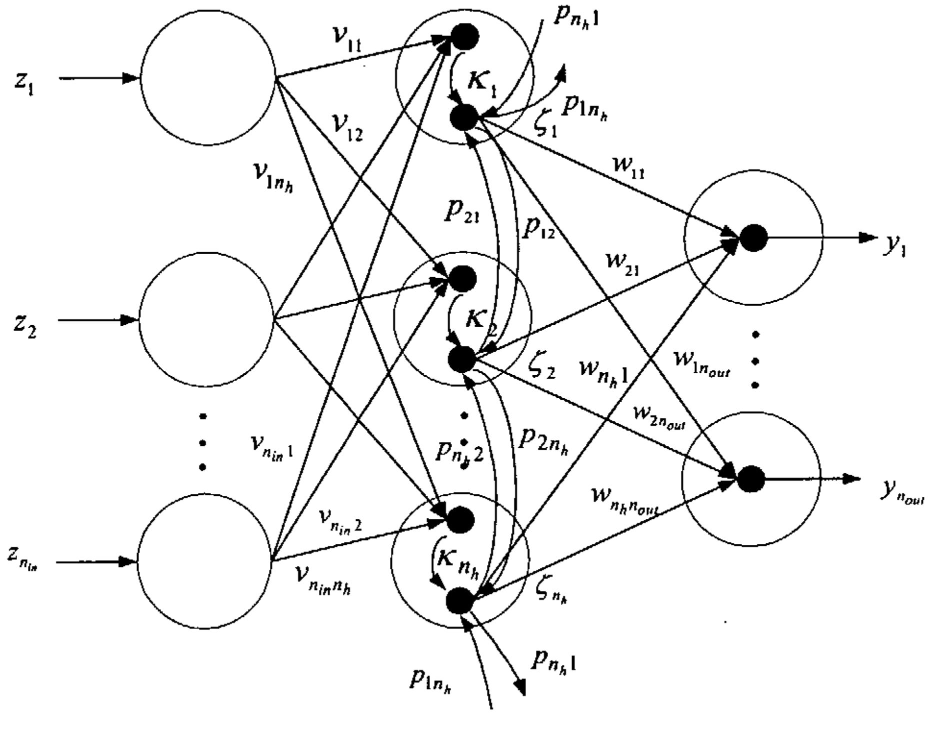


Fig. 4.16. Connection of the modified neural network.

4.5.1.1 Proposed neuron model

For the SATF, let v_{ij} be the synaptic connection weight from the i-th input node z_i to the j-th neuron. The output κ_j of the j-th neuron's SATF is defined as,

$$\kappa_j = net_s^j(\sum_{i=1}^{n_{in}} z_i v_{ij}),\ i = 1, 2, \ldots, n_{in},\ j = 1, 2, \ldots, n_h \tag{4.50}$$

where n_{in} denotes the number of input and $net_s^j(\cdot)$ is a static activation transfer function. The activation transfer function is defined as:

$$net_s^j(\sum_{i=1}^{n_{in}} z_i v_{ij}) = \begin{cases} e^{\frac{-\left(\sum_{i=1}^{n_{in}} z_i v_{ij} - m_s^j\right)^2}{2\sigma_s^{j^2}}} - 1 & \text{if } \sum_{i=1}^{n_{in}} z_i v_{ij} \le m_s^j \\ 1 - e^{\frac{-\left(\sum_{i=1}^{n_{in}} z_i v_{ij} - m_s^j\right)^2}{2\sigma_s^{j^2}}} & \text{otherwise} \end{cases},\quad j = 1, 2, \ldots, n_h \tag{4.51}$$

where m_s^j and σ_s^j are the static mean and static standard deviation for the j-th SATF respectively. The parameters (m_s^j and σ_s^j) are fixed after the training process. By using the proposed activation function in (4.51), the output value is ranged from –1 to 1. The shape of the proposed activation transfer function is shown in Fig. 4.17 and Fig. 4.18. In Fig. 4.17, the effect of the mean value to the activation transfer function is shown. The standard deviation σ_s of the function is fixed at 0.2 and the mean value m_s is chosen from –0.4 to 0.4. In Fig. 4.18, the effect of the standard deviation to the activation transfer function is shown. The mean value m_s is fixed at 0. The standard deviation σ_s is chosen from 0.1 to 0.5. It can be observed from these two figures that

$net_s(f) \to 1$ as $f \to \infty$ and $net_s(f) \to -1$ as $f \to -\infty$.

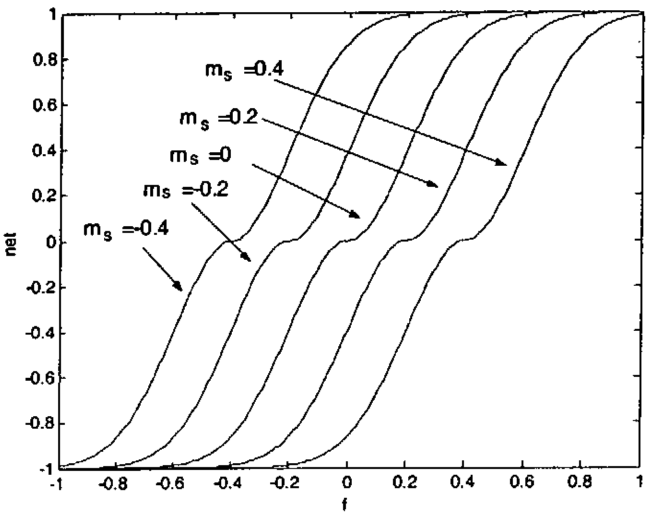


Fig. 4.17. Sample activation transfer functions of the proposed neuron (σ_s = 0.2).

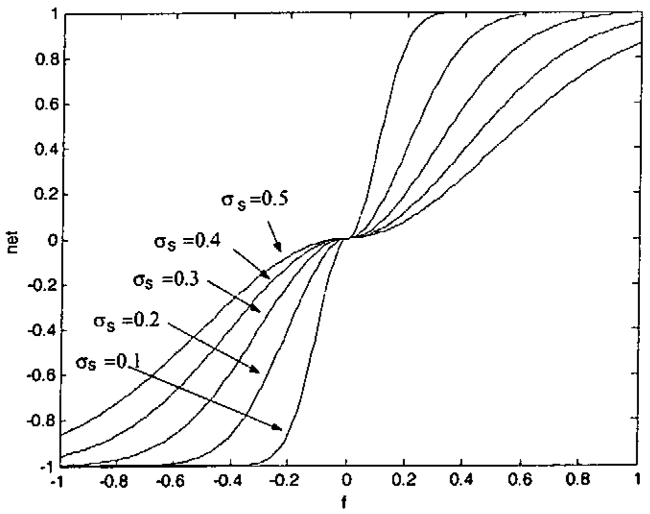


Fig. 4.18. Sample activation transfer functions of the proposed neuron (m_s = 0).

For the DATF, the neuron output ζ_j of the j-th neuron is defined as,

$$\zeta_j = net_d^j(\kappa_j),\ j = 1, 2, \ldots, n_h \tag{4.52}$$

and,

$$net_d^j(\kappa_j, m_d^j, \sigma_d^j) = \begin{cases} e^{\frac{-(\kappa_j - m_d^j)^2}{2\sigma_d^{j^2}}} - 1 & \text{if } \kappa_j \le m_d^j \\ 1 - e^{\frac{-(\kappa_j - m_d^j)^2}{2\sigma_d^{j^2}}} & \text{otherwise} \end{cases},\ j = 1, 2, \ldots, n_h \tag{4.53}$$

where

$$m_d^j = p_{j+1,j} \times \zeta_{j+1} \tag{4.54}$$

$$\sigma_d^j = p_{j-1,j} \times \zeta_{j-1} \tag{4.55}$$

m_d^j and σ_d^j are the dynamic mean and dynamic standard deviation for the j-th DATF; ζ_{j-1} and ζ_{j+1} represent the output of the $j-1$-th and j+1-th neurons respectively, $p_{j+1,j}$ denotes the weight of the link between the j+1-th node and the j-th node, and $p_{j-1,j}$ denotes the weight of the link between the $j-1$-th node and the j-th node. It should be noted from Fig.4.15 that if $j=1$, $p_{j-1,j}$ is equal to $p_{n_h,j}$, and if $j = n_h$, $p_{j+1,j}$ is equal to $p_{1,j}$.

Unlike the SATF, the DATF is dynamic as the parameters of its activation transfer function depend on the outputs of the $j-1$-th and j+1-th neurons. Referring to (4.50 - 4.53), the input-output relationship of the proposed neuron is as follows:

$$\zeta_j = net_d^j(net_s^j(\sum_{i=1}^{n_{in}} z_i v_{ij})),\, j = 1, 2, \ldots, n_h \tag{4.56}$$

4.5.1.2. Connection of the proposed neural network

The proposed MIMO neural network has three layers with n_{in} nodes in the input layer, n_h nodes in the hidden layer, and n_{out} nodes in the output layer. In the hidden layer, the neuron model presented in sub-section 4.5.1.1 is employed. A node-to-node relationship is introduced in the hidden layer. In the output layer, a static activation transfer function is employed. Considering an input-output pair $(\mathbf{z}, \mathbf{y})$, the output of the j-th node of the hidden layer is given by

$$\zeta_j = net_d^j(net_s^j(\sum_{i=1}^{n_{in}} z_i v_{ij})),\, j = 1, 2, \ldots, n_h \tag{4.57}$$

where v_{ij}, $i = 1, 2, \ldots, n_{in}; j = 1, 2, \ldots n_h$, denotes the weight of the link between the i-th input and the j-th hidden nodes. The output of the proposed neural network (Fig. 4.19) is defined as,

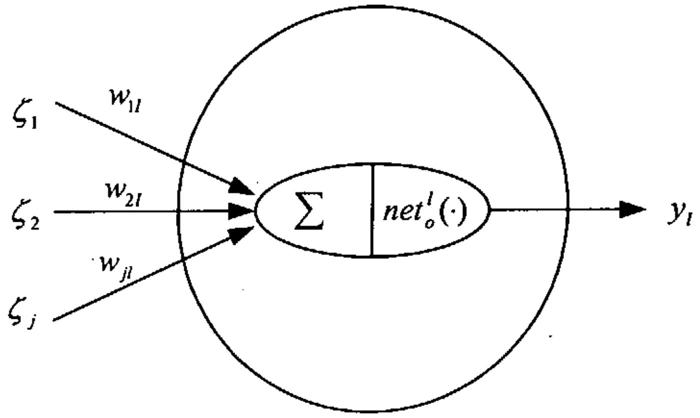
$$y_l = net_o^l(\sum_{l=1}^{n_{out}} \zeta_j w_{jl}),\ l = 1, 2, \ldots, n_{out} \tag{4.58}$$

$$= net_o^l(\sum_{l=1}^{n_{out}} net_d^j(net_s^j(\sum_{i=1}^{n_{in}} z_i v_{ij}))w_{jl}) \tag{4.59}$$

where $w_{jl}, j = 1, 2, \ldots, n_h; l = 1, 2, \ldots n_{out}$ denotes the weight of the link between the j-th hidden and the l-th output nodes; $net_o^l(\cdot)$ denotes the activation transfer function of the output neuron:

$$net_o^l(\sum_{j=1}^{n_h}\zeta_j w_{jl}) = \begin{cases} e^{\frac{-\left(\sum_{j=1}^{n_h}\zeta_j w_{jl}-m_o^l\right)^2}{2\sigma_o^{l^2}}}-1 & \text{if } \sum_{l=1}^{n_{out}}\zeta_j w_{jl} \le m_o^l \\ 1-e^{\frac{-\left(\sum_{j=1}^{n_h}\zeta_j w_{jl}-m_o^l\right)^2}{2\sigma_o^{l^2}}} & \text{otherwise} \end{cases} \qquad (4.60)$$

where m_o^l and σ_o^l are the mean and the standard deviation of the output node activation transfer function respectively. The parameters of the proposed modified neural network can be trained by the proposed fuzzy GA.



*k*th output neuron

Fig. 4.19. Model of the proposed output neuron.

4.5.2 Training with Fuzzy Genetic Algorithm

In this sub-section, the proposed neural network is employed to learn the input-output relationship of an application using the proposed fuzzy GA (as discussed in Section 4.2). The input-output relationship of the NN can be described by,

$$\mathbf{y}^d(t) = \mathbf{g}\left(\mathbf{z}^d(t)\right),\ t = 1, 2, \ldots, n_d \qquad (4.61)$$

where $\mathbf{y}^d(t) = \begin{bmatrix} y_1^d(t) & y_2^d(t) & \cdots & y_{n_{out}}^d(t) \end{bmatrix}$ is the desired output corresponding to the input $\mathbf{z}^d(t) = \begin{bmatrix} z_1^d(t) & z_2^d(t) & \cdots & z_{n_{in}}^d(t) \end{bmatrix}$ of an unknown nonlinear function $\mathbf{g}(\cdot)$, n_d denotes the number of input-output data pairs. The fitness function is defined as,

$$fitness = \frac{1}{1 + err} \tag{4.62}$$

$$err = \sum_{k=1}^{n_{out}} \frac{\sum_{t=1}^{n_d} \left| y_k^d(t) - y_k(t) \right|}{n_d \times n_{out}} \tag{4.63}$$

The objective is to maximize the fitness value of (4.62) using the proposed fuzzy GA by setting the chromosome to be $[v_{ij} \quad m_s^j \quad \sigma_s^j \quad p_{j+1,j} \quad p_{j-1,j} \quad w_{jl} \quad m_o^l \quad \sigma_o^l]$ for all i, j, l. The range of fitness function in (4.62) is [0, 1]. A larger value of *fitness* indicates a smaller *err*.

4.5.3 Short-Term Daily Electric Load Forecasting System

The structure of the short-term daily electric load forecasting system is the same as that in sub-section 4.3.2. Twelve sets of historical data are used for first-time off-line training with 1000 iterations. Once the first-time off-line training is done, the forecasting system can be put to daily operation. During the operation, the system will update the weights of neural network based on new data with 200 iterations daily. From (4.59), the proposed neural network used for the daily load forecasting is governed by,

$$y_l(t) = net_o^l\Big(\sum_{j=1}^{n_h} net_d^j\big(net_s^j\big(\sum_{i=1}^{24} z_i v_{ij}\big)\big)w_{jl}\Big), \ l = 1, 2, \ldots, 24 \tag{4.64}$$

The number of hidden node (n_h) is changed from 3 to 9 in order to test the learning performance. The fitness function is defined as follows,

$$fitness = \frac{1}{1 + err} \tag{4.65}$$

$$err = \frac{1}{12}\sum_{t=1}^{12}\frac{1}{24}\sum_{k=1}^{24}\frac{\left|y_k^d(t) - y_k(t)\right|}{y_k^d(t)} \tag{4.66}$$

The proposed fuzzy GA is employed to tune the parameters of the modified neural network of (4.64). The objective is to maximize the fitness function of (4.65). The value of *err* indicates the mean absolute percentage error (MAPE) of the forecasting result. The best fitness value is 1 and the worst value is 0. The population size used for the fuzzy GA is 10. The probability of acceptance r_r is set at 0.1 and the probability of mutation p_m is set at 0.03. The chromosomes used for the fuzzy GA are $[v_{ij} \;\; m_s^j \;\; \sigma_s^j \;\; p_{j+1,j} \;\; p_{j-1,j} \;\; w_{jl} \;\; m_o^l \;\; \sigma_o^l]$ for all *i*, *j*, *l*. The number of iterations to train the proposed neural network for the first time is 1000. For comparison purpose, a traditional neural network trained by the proposed fuzzy GA is also applied to do the electric load forecasting. The working conditions are kept the same. The results are tabulated in Table 4.10 and Table 4.11. Table 4.10 shows the simulation results of the daily load forecasting for Wednesday, and Table 4.11 shows the daily load forecasting for Sunday. These tables show the fitness value and the number of parameters of the network. We can observe that the performance of the proposed modified neural network is better than that from a traditional neural network. The worst result offered by the proposed neural network is even better than the best result offered by the traditional neural network. Table 4.12 shows the off-line training error and the forecasting error (week 13-week 15) on using the proposed modified neural network at $n_h = 4$. The average errors of training and forecasting are 1.6488 and 1.9604 respectively. Fig. 4.20 and Fig. 4.21 show the simulation results of the daily load forecasting on Wednesday (week 13) and Sunday (week 13) using the proposed neural network. In these figures, the dotted line in black represents the forecasted result using the proposed neural network, and the dashed line in red is the forecasted result using the traditional neural network. The actual load is represented by a solid line in blue. We can see that the forecasting result using the proposed modified neural network is better.

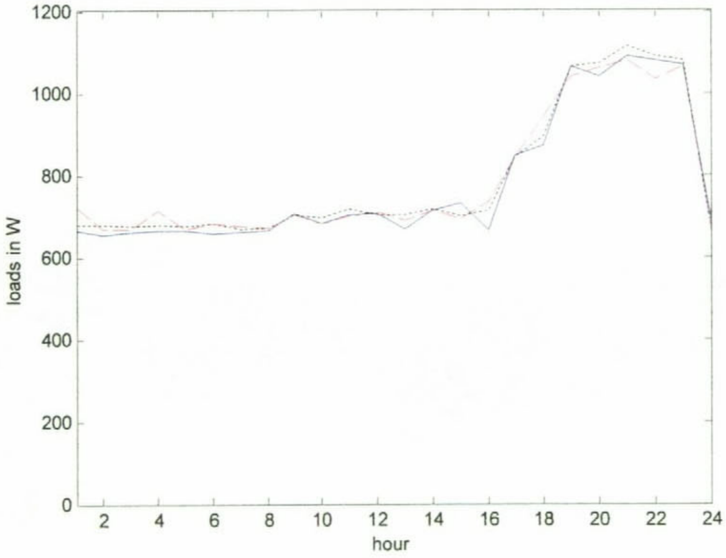


Fig. 4.20. Daily load forecast results on Wednesday (Week13) with the proposed neural network (dotted line in black) and the traditional neural network (dashed line in red), as compared with the actual load (solid line in blue).

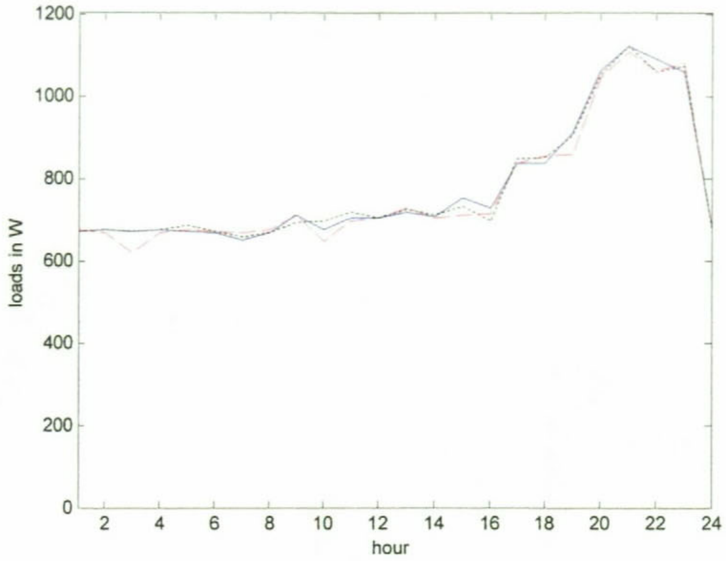


Fig. 4.21. Daily load forecast results on Sunday (Week13) with the proposed neural network (dotted line in black) and the traditional neural network (dashed line in red), as compared with the actual load (solid line in blue).

	Proposed Neural Network			Traditional Neural Network		
n_h	Fitness Value	Training error (MAPE)	Number of parameters	Fitness Value	Training error (MAPE)	Number of parameters
3	0.9830	1.7294	216	0.9640	3.7344	183
4	0.9842	1.6054	272	0.9684	3.2631	236
5	0.9831	1.7191	328	0.9708	3.0078	289
6	0.9832	1.7087	384	0.9620	3.9501	342
7	0.9815	1.8849	440	0.9742	2.6483	395
8	0.9818	1.8537	496	0.9738	2.6905	448

Table 4.10. Simulation results of the proposed neural network and the traditional neural network for daily load forecasting for Wednesday.

	Proposed Neural Network			Traditional Neural Network		
n_h	Fitness Value	Training error (MAPE)	Number of parameters	Fitness Value	Training error (MAPE)	Number of parameters
3	0.9810	1.7294	216	0.9567	4.5260	183
4	0.9833	1.9368	272	0.9683	3.2738	236
5	0.9815	1.6984	328	0.9705	3.0397	289
6	0.9801	1.8849	384	0.9610	4.0583	342
7	0.9799	2.0304	440	0.9712	2.9654	395
8	0.9791	2.0512	496	0.9701	3.0822	448

Table 4.11. Simulation results of the proposed neural network and the traditional neural network for daily load forecasting for Sunday.

	First-time off-line training error from Week 1-Week 12	Forecasting error at Week 13	Daily off-line training error from Week 2-Week 13	Forecasting error at Week 14	Daily off-line training error from Week 3-Week 14	Forecasting error at Week 15
Monday	1.6417	2.3821	1.6188	1.3327	1.5022	2.0012
Tuesday	1.7035	1.8925	1.6803	1.6713	1.6388	2.1056
Wednesday	1.6063	2.0117	1.6331	1.9982	1.6682	2.4321
Thursday	1.6952	1.6602	1.6415	2.0299	1.7099	1.4894
Friday	1.7036	2.0985	1.6682	2.3055	1.8601	1.9827
Saturday	1.6682	1.6536	1.5411	1.9034	1.5793	2.3021
Sunday	1.7049	1.4331	1.6102	2.2753	1.5714	2.2077

Table 4.12. Off-line training error and forecasting error in terms of MAPE from Monday to Sunday.

4.6 CHAPTER CONCLUSION

Short-term home electric load forecasting systems realized by computational intelligence techniques have been discussed in this chapter. These techniques involve a GA-based neural network, a GA-based neural fuzzy network, and a modified GA-based neural network. All parameters of the proposed networks are tuned by a proposed fuzzy genetic algorithm. The proposed fuzzy GA is modified from the published GA with arithmetic crossover and non-uniform mutation. Modified genetic operators have been introduced. Based on the benchmark test functions, it has been shown that the fuzzy GA performs better than the traditional GA.

On implementing these home electric load forecasting systems, the objectives are to minimize the training errors and forecast the electric load. In a normal environment, the home power consumption pattern of a given week should not deviate too much from that of its next week. Thus, a small training error can be obtained and yet given a small forecasting error. In some special cases, when the habit of the family on power consumption changes suddenly, the forecasting error may be affected.

By introducing a switch to each link, a GA-based neural network that facilitates the tuning of its structure has been proposed. Using the proposed fuzzy GA, the neural network is able to learn both the input-output relationship of an application and the optimal network structure. As a result, a given fully connected neural network will become a partly connected neural network after training. This implies a lower cost of implementation.

A GA-based neural fuzzy network has been proposed in which a switch is introduced in each fuzzy rule. Thus, the number of rules can be optimized using the proposed fuzzy GA. The cost of implementing the proposed NFN can be reduced, while the network parameters can be optimised.

A proposed modified neural network trained by the proposed fuzzy GA has also been presented. A neuron model with two activation transfer functions has been introduced. With this proposed neuron, the connection of the proposed neural network exhibits a node-to-node relationship in the hidden layer. On using this network, the performance is found to be better than that of a traditional feed-forward neural network.

The performance of these three proposed computational intelligence techniques are satisfactory when applying to short-term home electric load forecasting. Comparing the neural network approaches and the neural fuzzy network approach, the training and forecasting errors of the neural fuzzy network are found to be smaller than those of neural network approaches. However, more networks are necessary (7 networks in the neural network approaches against 168 networks in the neural fuzzy network approach). The modified neural network discussed in Section 4.5 can be regarded as an improved version of the neural network discussed in Section 4.3. By using the modified neural network, the number of the hidden nodes can be reduced and a better performance can be obtained.

CHAPTER FIVE

HOME ELECTRIC LOAD BALANCING SYSTEM

5.1 INTRODUCTION

With the recent technological advancement in consumer electronics and the improvement in the quality of life, a significant increase in the consumption of domestic electrical power is seen [Lee 99]. This increase, which causes overloading in existing power systems, may not occur at all time but only at some peak hours. The average power consumption is not increased drastically. Reinstalling a more sophisticated power system is not an economic solution to solve this type of overloading problem. Rather, implementing schemes that schedule the power consumption in order to decrease/increase the load demand from the utility company during peak/off-peak period is preferred. It is because an evenly distributed power consumption profile from the point of view of the utility company can result in an efficient way of using power. It can reduce the operation cost of the power system and relieve the need of adding new generator units. To achieve this goal, rechargeable batteries are needed to share the loading. Thanks to the advancement in the power electronics technology, it is not too difficult to install low-cost high-power rechargeable batteries at our homes.

This chapter proposes a home electric load balancing system that employs rechargeable batteries at home. The power drawn by the home loads will be monitored and the power drawn from the utility company will be regulated to a reference value. If the home needs more power than the reference value, the batteries will provide the extra power. If the home draws less power than the reference value, the energy in excess will be used to charge the batteries. As a result, the power drawn from the mains will be kept near to a constant. The proposed system not only alleviates the overloading problem in the user side, but also relieves the instantaneous demand to the utility company. The reference value can be determined by a home electric load forecasting system discussed in Chapter 4.

Apart from the primary objective of regulating the power drawn from the utility company, a secondary control objective is to prevent the amount of energy stored in the batteries from reaching the practical limits. If the upper/lower limit is reached, the batteries can no longer be charged/discharged and the regulation will fail. The short-term home electric load forecasting system may help in suggesting a suitable amount of energy to be stored in the batteries with respect to time. Whenever necessary, the system will spare some control power to prevent the battery-stored energy from reaching the limits by sacrificing a bit of the performance in the regulation.

The proposed home electrical load balancing system will be described in Section 5.2. An example will be shown in Section 5.3. A chapter conclusion will be drawn in Section 5.4.

5.2 DESIGN AND CONTROL

5.2.1 System Architecture

The system block diagram is shown in Fig. 5.1. Let P_A be the power drawn from the AC mains, which is to be regulated; L_A be the power consumption of the home, R be the reference power consumption which is the mean value of the forecasted daily power consumption of the home, E_A be the difference between R and L_A, B_{fore} and B_{act} be the forecasted stored energy of the batteries and the actual stored energy of the batteries respectively, E_B be the difference between B_{fore} and B_{act}. The primary control objective of the proposed system is to regulate P_A to the reference value R under a varying L_A. The control signal U controls the amount of power flowing to or from the batteries via the bi-directional converter (U is the reference input power of the bi-directional converter). A positive U makes the bi-directional converter act as a power converter drawing power $P_B = U$ from the batteries to the power line. A negative U makes the bi-directional converter act as a charger to charge the batteries with power $L_B = -U$. Theoretically, the storage capacity of the batteries should be as large as possible to prevent them from fully charged or discharged during operation. However, the physical size and cost of the batteries are practical constraints. The value of U depends on two values: U_A and U_B. U_A is for reducing E_A while U_B attempts to prevent the amount of energy stored in the batteries from reaching the limits. The latter, in fact, is the secondary control objective of the system. The values of U_A and U_B are governed by the gain values of K_1 and K_2 respectively.

It can be seen from Fig. 5.1 that the values of L_B and P_B can be controlled and the values of the disturbance input L_A are measured. Since the dynamics of the bi-directional converter is fast, its transfer function can be approximated as a constant. The transfer functions from the disturbance input (L_A) to the output (P_A), and from the control input (U) to the output (P_A) are also constants.

5.2.2 Battery Capacity

The capacity of the installed batteries for the home load balancing system can be determined based on the historical daily power consumption profiles of L_A. Assuming that L_A is roughly daily periodic in some sense, a set of historical daily power consumption profiles can be sampled. A typical power consumption profile and the corresponding required energy profile stored in the batteries are shown in Fig. 5.2. From the sampled profiles, the profile with the maximum mean value of power consumption is selected. Let this value be R_a, then the total discharging energy A_+ (the sum of areas above R_a) equals the total charging energy A_- (the sum of areas below R_a). In practice, we do not want the stored energy of the batteries to reach zero or a specified maximum value. An upper limit and a lower limit for the stored energy should be set as shown in Fig. 5.2c. The capacity of the batteries is effectively the upper limit C_{upper} as required by the proposed system:

$$C_{upper} = B'_{act_{max}} K \qquad (5.1)$$

where $B'_{act_{max}}$ is the maximum required energy stored in the batteries for the load balancing, and K is a scaling factor greater than unity for the tolerance of any discrepancy between the predicted and the actual profiles. Then, the lower limit for the stored energy of the batteries is given by:

$$C_{lower} = B'_{act_{max}}(\frac{K-1}{2}) \tag{5.2}$$

5.2.3 System Design

The roles of the home electric load forecasting inside the load balancing system are to determine R and B_{fore}. A typical forecasted home electric load consumption profile and the corresponding forecasted stored energy of the batteries are shown in Fig. 5.3. As seen from Fig. 5.3, R is the mean power consumption of the forecasted daily load. B_{fore} is the forecasted stored energy of the batteries given by:

$$B_{fore} = B'_{fore} + C_{lower} \tag{5.3}$$

where B'_{fore} is the forecasted required energy stored in the batteries for supporting the load balancing. Based on the primary objective (to regulate P_A), an output control signal U_A should be designed:

$$U_A = -K_1(R - L_A) \tag{5.4}$$

or

$$U_A = -K_1 E_A \tag{5.5}$$

where K_1 is a positive gain. If the forecast error is equal to zero, only the primary objective in the balancing system has to be satisfied. Unfortunately, the forecast error is seldom equal to zero. With forecast errors, the amount of energy stored in the batteries may reach the upper/lower limit of the batteries. To alleviate this problem, the secondary objective (to prevent the amount of energy stored in the batteries from reaching the practical limits) has to be achieved, and an output control signal U_B should be designed:

$$U_B = -K_2(B_{fore} - B_{act}) \tag{5.6}$$

or

$$U_B = -K_2 E_B \tag{5.7}$$

where K_2 is a positive gain.

It can be seen that if E_B is positive, $B_{fore} > B_{act}$, and the amount of the stored energy may not be enough for future use. Therefore, the control signal U_B should be more negative so that the charging of the batteries will be made faster. On the other hand, if E_B is negative, $B_{fore} < B_{act}$, and the amount of the stored energy is more than enough for future use. Therefore, the control signal U_B should be more positive so that the charging of the batteries will be made slower.

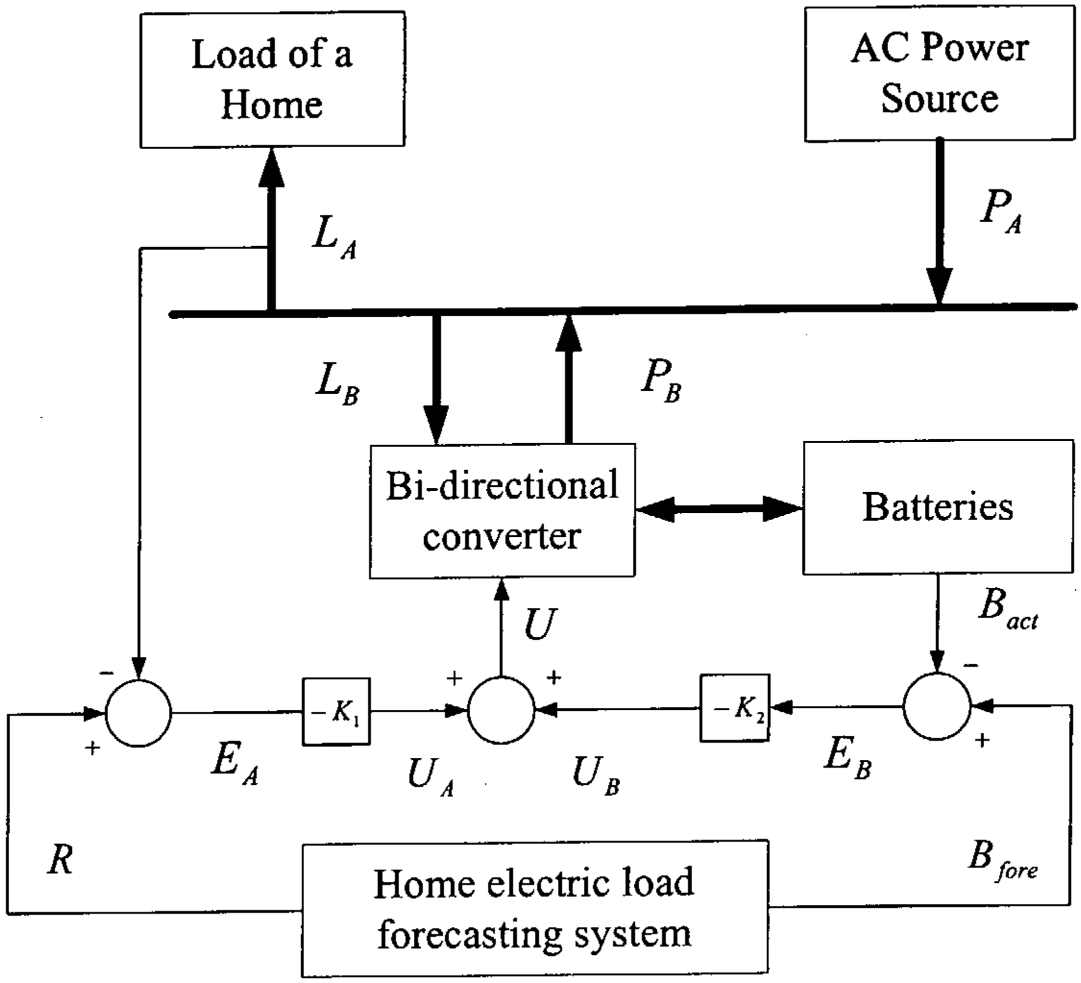
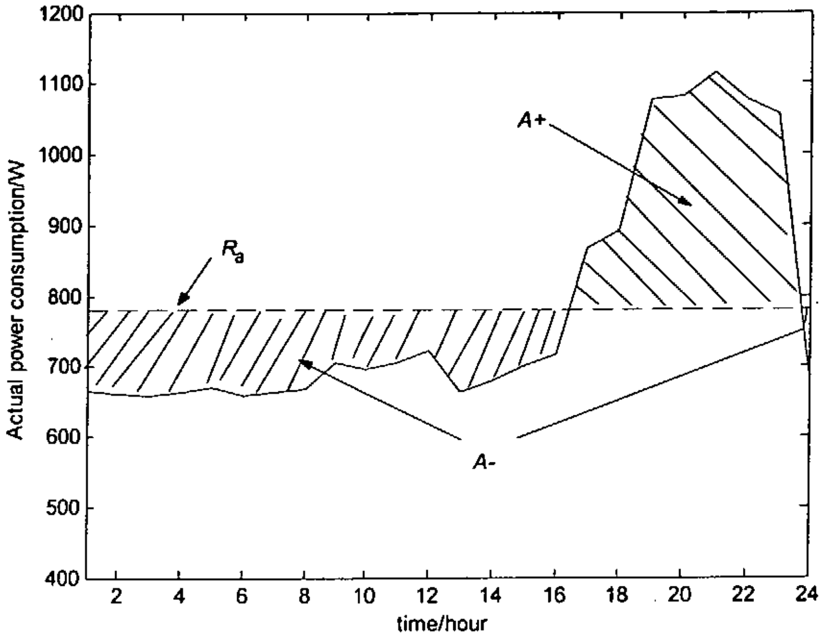
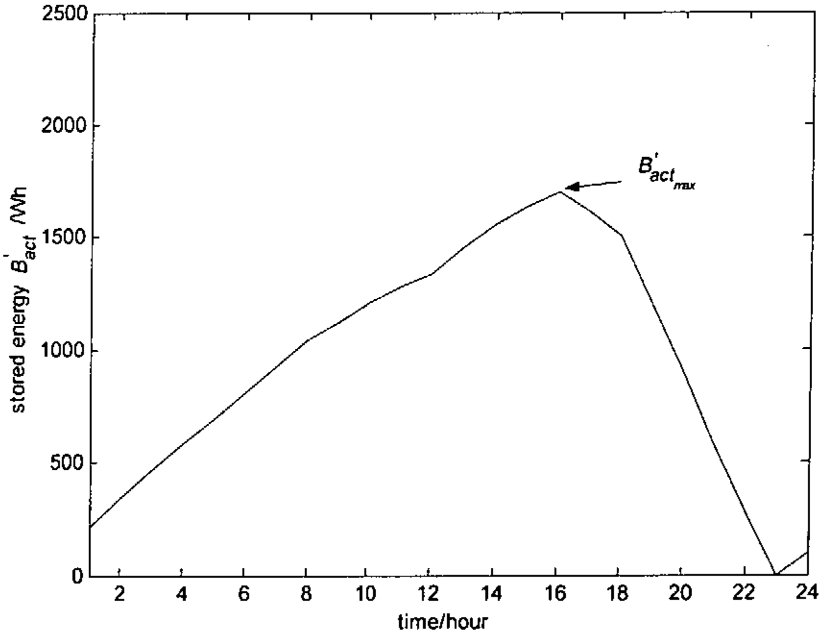


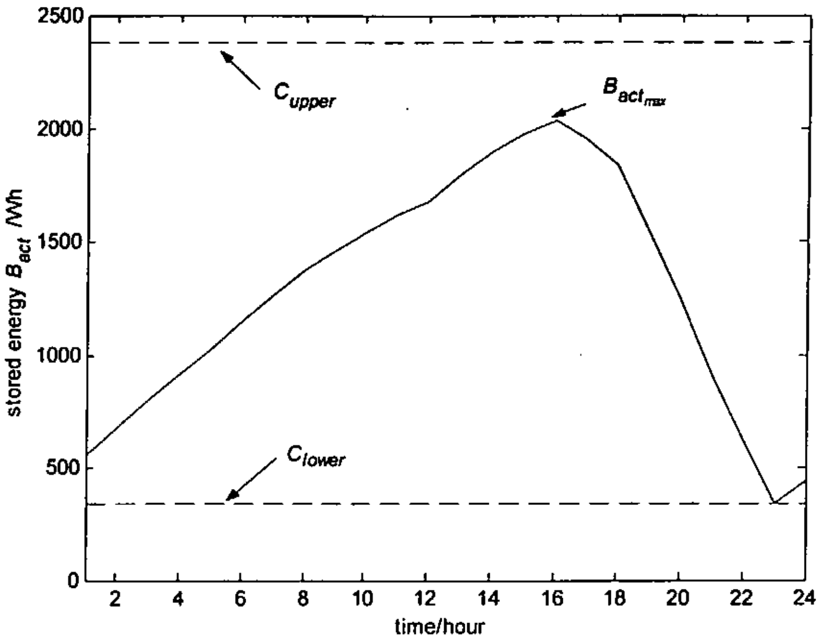
Fig. 5.1. System block diagram.


1200
1100
1000
900
800
700
600
500
400
Actual power consumption/W
2
4
6
8
10
12
14
16
18
20
22
24
time/hour
A+
R_a
A-

(a)

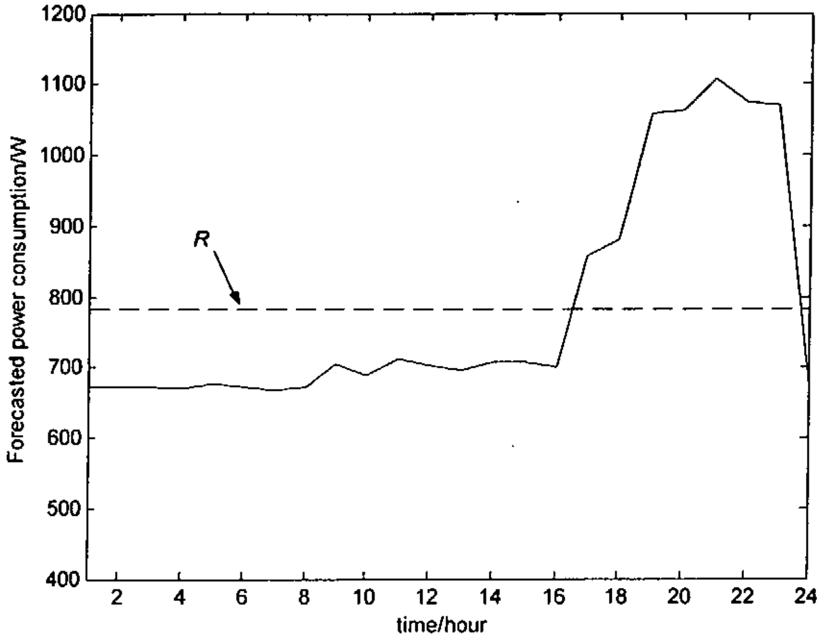

2500
2000
1500
1000
500
0
stored energy B'_{act} /Wh
2
4
6
8
10
12
14
16
18
20
22
24
time/hour
$B'_{act_{max}}$

(b)

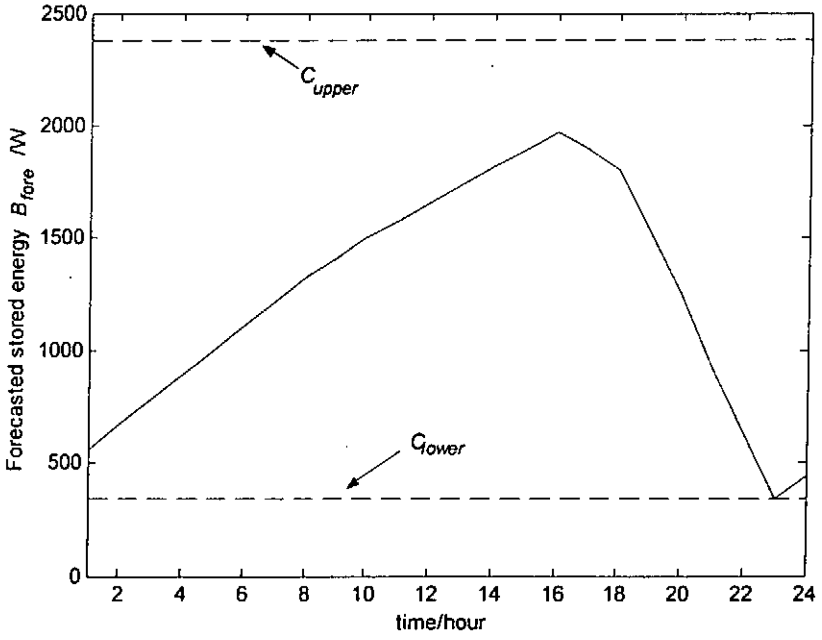


(c)

Fig. 5.2. (a) Actual home daily power consumption profile (b) required stored energy of batteries, (c) practical stored energy of batteries with upper and lower limits.



(a)



(b)

Fig. 5.3. (a) Forecasted home power consumption profile, (b) forecasted stored energy of batteries.

5.3 SIMULATION EXAMPLE

A home load balancing system based on the GA-based neural network forecaster presented in section 4.3 will be used. The system diagram is shown in Fig. 5.1. The actual and forecast week-13 results are considered in this simulation example. Details about the forecast result have been presented in section 4.3. Referring to Fig. 5.1, the values of K_1 and K_2 are 1 and 0.02 respectively, which are tuned by trail-and-error. From a set of actual consumption profiles, the one with the maximum mean power consumption R_a is selected. From (5.2), it is found that R_a = 780.8W, and $B'_{act_{max}}$ = 1698Wh. Let K = 1.4, from (5.1), C_{upper} = 2377.2 kWh and C_{lower} = 339.6Wh. Fig. 5.4 shows the regulation results of the home load balancing system for 168 hours (7 days). The home power consumption L_A has large fluctuations as shown in Fig. 5.4a. It can be seen from Fig. 5.4b that the regulated power drawn from the AC mains P_A under the proposed home load balancing system does not have a large fluctuation. Fig. 5.4c shows that the amount of energy stored in the battery is not over the limits, and its mean value tends to keep at a constant. Fig. 5.4d shows the control signal U.

To show the ability of the load balancing system on meeting the secondary objective, another system with K_2 = 0 is also tested. Fig. 5.5 shows the results. Fig. 5.5a shows that the control result of P_A with K_2 = 0 is better than that with K_2 = 0.02, but the amount of energy stored in batteries will be over the upper limit as shown in Fig. 5.5b.

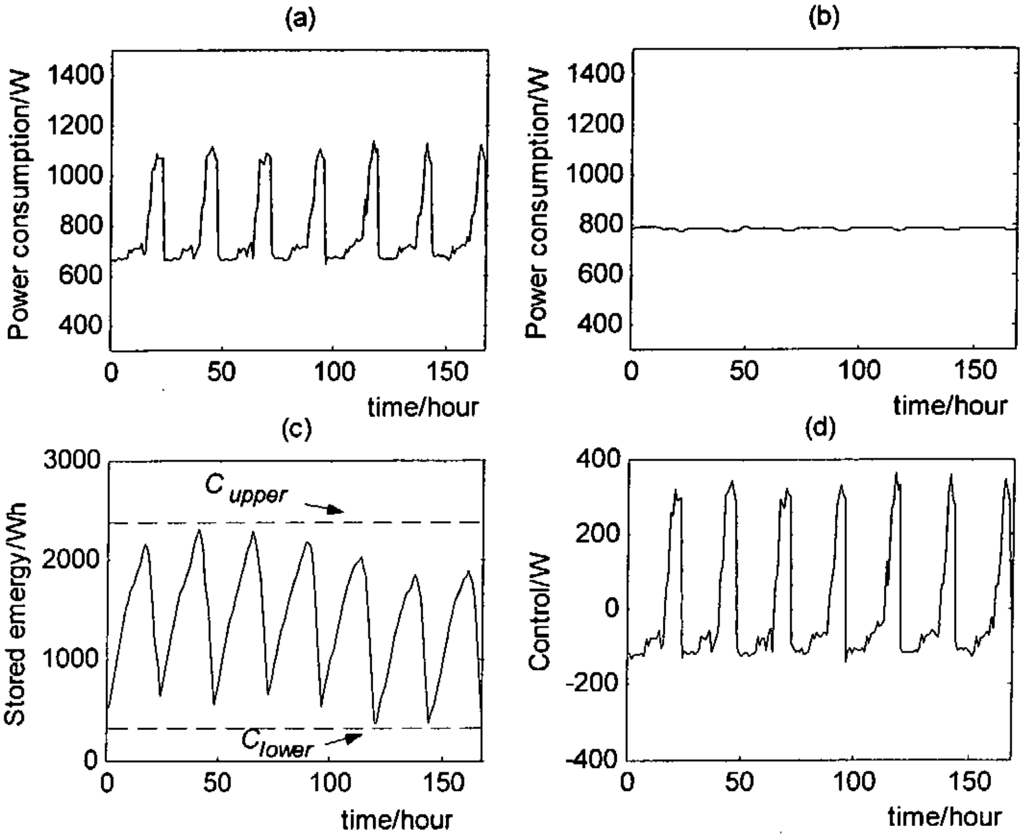
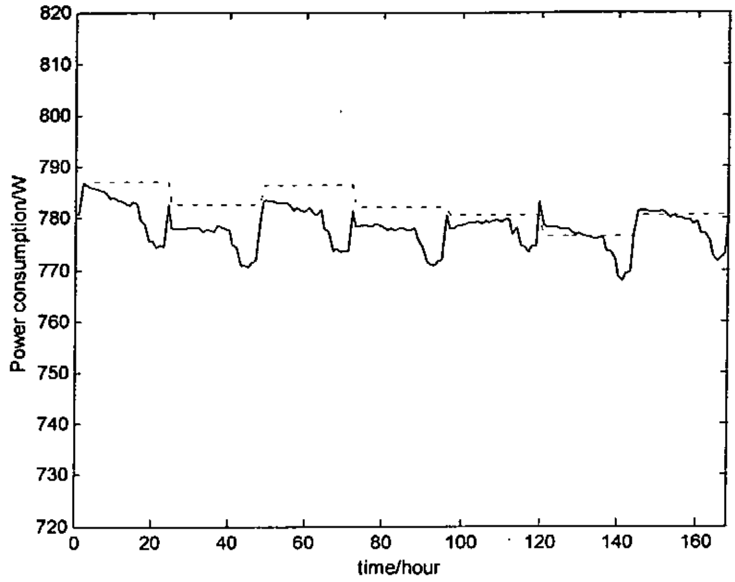
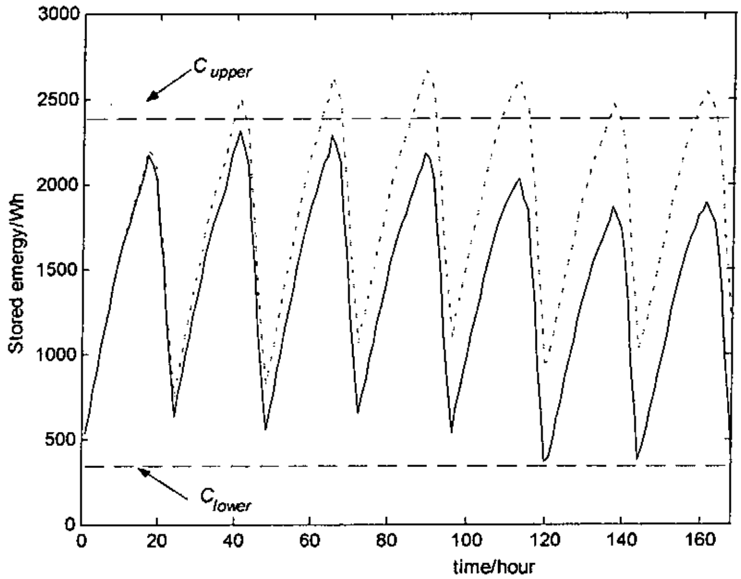


Fig. 5.4. Control results: (a) actual power consumption profile; (b) regulated power drawn from AC mains P_A; (c) stored energy of batteries; (d) control signal U.



(a)



(b)

Fig. 5.5. Control results: (a) regulated power consumption P_A with $K_2 = 0.02$ (solid line) and $K_2 = 0$ (dotted line); (b) stored energy of the batteries with $K_2 = 0.02$ (solid line) and $K_2 = 0$ (dotted line).

5.4 CHAPTER CONCLUSION

The primary objective of a home load balancing system is to maintain an even distribution of the power drawn from the utility company in order to avoid overloading the domestic power supply system. In practice, the system has to achieve also a secondary objective: to regulate the amount of energy stored in the batteries in order to prevent it from reaching the limits. To achieve the first objective, a feed-forward controller with constant gain is applied, which processes the error signal between the actual power consumption and the predicted mean power consumption based on a home electric load forecaster. To achieve the secondary objective, the electric load forecaster is also used to predict the amount of energy stored in the batteries, which is then compared with the actual stored energy and processed by another proportional controller. By using an example, it has been shown that the home load balancing system can successfully regulate the power drawn from the AC mains, and the energy stored in the batteries can be kept within the limits, despite the presence of large variations in the home consumption profile.

CHAPTER SIX

CONCLUSION

6.1 ACHIEVEMENTS

Intelligent techniques for home electric load forecasting and balancing have been presented in this thesis. A power line data network based on the spread-spectrum technology has been developed. The network features such as low cost, high reliability, fast communication and effectiveness have been discussed. With this power line data network, communications among electrical appliances and home users can take place at any time in any place. It serves as a low cost backbone to enhance the security and comfort of the home users. One advantage is that this system takes a modular structure so that every module can be added or removed, and placed anywhere in a house without resorting to special installation of cables. This is convenient to general consumers because a prudent planning for the home networking system before building it is not necessary. The system is readily installed in existing buildings.

Short-term electric load forecasting (STELF) is essential to improve the reliability of the AC power line data network and provide optimal load scheduling in an intelligent home system. Three computational intelligence approaches have been developed to realize the STELF. These three approaches involve a GA-based neural network, a GA-based neural fuzzy network and, a modified GA-based neural network.

All these three systems can forecast the electric load of a home with respect to different day types and weather information. The systems are tuned by a proposed fuzzy GA. The proposed fuzzy GA is modified from a published GA with arithmetic crossover and non-uniform mutation. Fuzzy logic is used to incorporate expert knowledge and experience (in terms of linguistic rules) into the crossover and mutation operations. Modified fuzzy genetic operators have been introduced. By using some benchmark test functions, it has been shown that the fuzzy GA performs better than the traditional GA.

The first approach for realizing the STELF involves a fuzzy GA-based neural network. In many applications of NNs, the networks are fully connected. However, the performance of a fully connected NN may not be better than that of a partly connected NN. This is because some links in an NN could be redundant. A three-layer NN with a switch introduced to each link has been proposed. Besides training for the optimal parameters, by turning on or off the link-switches during the training process, the fuzzy GA can help to find the optimal NN structure. This implies a lower cost of implementing the NN.

The second approach for realizing the STELF involves a fuzzy GA-based neural fuzzy network (NFN). The optimal NFN structure can be found by the fuzzy GA when switches in the rules of the NFN are introduced. Both the membership function parameters and the number of rules of the NFN can be obtained through training.

The third approach for realizing the STELF involves a modified fuzzy GA-based neural network. A new neuron model has been proposed. In this model, the neuron has two activation transfer functions and exhibits a node-to-node relationship within the hidden layer. This modified neural network can offer a better performance and a smaller number of hidden nodes than the traditional feed-forward neural network.

The above three approaches have been applied to realize different home electric load forecasting systems. It has been shown that these load forecasting systems can successfully forecast the power consumption pattern of a home. One of the electric load forecasting system has been further applied in a home electric load balancing system. The system can adjust the amount of energy stored in the batteries installed at a home so as to prevent it from reaching some practical limits. At the same time, the power drawn from the utility company is regulated to a nearly constant value, despite the presence of large variations in the home power consumption.

6.2 FURTHER RESEARCH

The proposed fuzzy GA [Ling 02a] can be further improved. In this thesis, the probabilities associated with the genetic operations are kept to be static. In our view, dynamic probabilities for the genetic operations (that, for example, change with the iteration number) may provide a faster convergence and help to find the global solution. Further studies on these dynamic probabilities for the genetic operations could be done. Concerning the modified GA-based neural network, the proposed neuron might also be applied in the proposed neural fuzzy network discussed in section 4.4. The properties and performance of this modified GA-based NFN could be further investigated.

APPENDIX

GENETIC OPERATIONS IN GA WITH FLOATING POINT ENCODING

The genetic operations used in GAs with floating point encoding are quite different from those in GAs with binary encoding. They work in a real valued space. Arithmetic crossover and heuristic crossover have been proposed for the crossover operation. Uniform mutation and non-uniform mutation have been proposed for the mutation operation. The following sections will give a description for these genetic operations.

A.1. CROSSOVER

A.1.1 Arithmetic crossover

Arithmetic crossover is defined as a linear combination of two selected chromosomes ($\mathbf{p}_1$ and $\mathbf{p}_2$). The resulting offspring $\mathbf{o}_s$ is defined as,

$$\mathbf{o}_s = a \times \mathbf{p}_1 + (1-a) \times \mathbf{p}_2 \qquad \text{(A.1)}$$

where $\mathbf{o}_s = \lfloor o_{s_1}, \ o_{s_2}, \ \cdots \ o_{s_k}, \ \cdots \ ,o_{s_{no\_vars}} \rfloor$, $k \in [1, 2, \dots$ *no_vars*], *no_vars* denotes the number of variables to be tuned. This operation depends on a parameter *a*, which is a constant.

A.1.2 Heuristic crossover

Heuristic crossover produces a linear extrapolation of two chromosomes. It is an operation that utilizes the fitness value information. The resulting offspring is given by,

$$\mathbf{o}_s = \mathbf{p}_1 + (\mathbf{p}_1 - \mathbf{p}_2) \times r \tag{A.2}$$

where r is a random number in [0, 1], and $\mathbf{p}_1$ is better than $\mathbf{p}_2$ in terms of the fitness value. Define:

$$feasibility(\mathbf{o}_s) = \begin{cases} 1 & \text{if } para_{\min}^{k} \le o_{s_k} \le para_{\max}^{k} \quad \forall k \\ 0 & \text{otherwise} \end{cases} \tag{A.3}$$

where $para_{\min}^{k}$ and $para_{\min}^{k}$ are the minimum and maximum values of the parameter o_{s_k} respectively. If $\mathbf{o}_s$ is infeasible, i.e., $feasibility(\mathbf{o}_s)$ = 0, a new random number r should be generated to create a new offspring using (A.2). This process is repeated until $\mathbf{o}_s$ is feasible.

A.2. MUTATION

A.2.1. Uniform mutation

For uniform mutation, each element o_{s_k} in $\mathbf{o}_s = \lfloor o_{s_1}, \ o_{s_2}, \ \ldots \ , o_{s_{no\_vars}} \rfloor$ has an equal chance of undergoing the mutation process. The result of a single operation is a vector $\hat{\mathbf{o}}_s = \lfloor o_{s_1}, \ \ldots \ , \hat{o}_{s_k}, \ \ldots \ , o_{s_{no\_vars}} \rfloor$, $k \in [1, 2, \ldots \ no\_vars]$ and $\hat{o}_{s_k}$ is a random value taken from the domain of the corresponding parameter in [$para_{\min}^{k}, para_{\max}^{k}$], where $para_{\min}^{k}$ and $para_{\min}^{k}$ are the minimum and maximum values of the parameter respectively. In brief, uniform mutation simply replaces a randomly selected element in the chromosome with a random real number in [$para_{\min}^{k}, para_{\max}^{k}$].

A.2.2. Non-uniform mutation

Non-uniform mutation is an operation with a fine-tuning capability. Its action depends on the generation number of the population. The operation takes place as follows. If $\mathbf{o}_s = \lfloor o_{s_1}, \; o_{s_2}, \; \ldots \; , o_{s_{no\_vars}} \rfloor$ is a chromosome and the element o_{s_k} is randomly selected for mutation (the value of o_{s_k} is inside [$para_{\min}^{k}, para_{\max}^{k}$]), the resulting chromosome is then given by $\hat{\mathbf{o}}_s = \lfloor o_{s_1}, \; \ldots \; , \hat{o}_{s_k}, \; \ldots \; , o_{s_{no\_vars}} \rfloor$, $k \in 1, 2, \ldots$ *no_vars*, and

$$\hat{o}_{s_k} = \begin{cases} o_{s_k} + \Delta\left(\tau, para_{\max}^{k} - o_{s_k}\right) \text{ if } r_d = 0 \\ o_{s_k} - \Delta\left(\tau, o_{s_k} - para_{\min}^{k}\right) \text{ if } r_d = 1 \end{cases} \tag{A.4}$$

where r_d is a random digit (r_d is equal to 0 or 1 only). The function $\Delta(\tau, y)$ returns a value in the range [0, y] such that $\Delta(\tau, y)$ approaches 0 as τ increases. The function is defined as follows,

$$\Delta(\tau, y) = y\left(1 - r^{\left(1 - \frac{\tau}{T}\right)^{b}}\right) \tag{A.5}$$

where r is a random number in [0, 1], τ is the present generation number of the population, T is the maximum generation number of the population, and b is a system parameter that determines the degree of non-uniformity.

REFERENCES

[Amin 97]

S. Amin and J.L. Fernandez-Villacanas, "Dynamic Local Search, " in *Proc. 2nd Int. Conf. Genetic Algorithms in Engineering Systems: Innovations and Applications*, 1997, pp129-132.

[Amitava 99]

D.R. Amitava, "Networks for homes," *IEEE Spectrum*, Vol.36, pp.26-33, Dec. 1999.

[Araujo 00]

A.L. Araujo and F.M. Assis," An improved genetic algorithm performance with benchmark functions," *Proceedings Sixth Brazilian Symposium on Neural Networks*, 2000, pp 292.

[Bakirtzls 96]

A.G. Bakirtzls, V. Petridls, S.J. Klartzis, M.C. Alexladls, and A.H. Malssls, "A neural network short term load forecasting model for Greed power system", *IEEE Transaction on Power Systems*, vol.11, no.2, pp.858-863, May1996.

[Belarbi 00]

K. Belarbi and F. Titel, "Genetic algorithm for the design of a class of fuzzy controllers: an alternative approach," *IEEE Trans. Fuzzy Systems*, vol. 8, no. 4, pp. 398-405, Aug. 2000.

[Brown 94]

M. Brown and C. Harris, *Neuralfuzzy adaptive modeling and control*, Prentice Hall, 1994.

[Bryson 1969]

A.E. Bryson and Y.C. Ho, *Applied Optimal Control*, Blaisdell, 1969.

[Caponetto 00]

R. Caponetto, L. Fortuna, G. Nunnari, L. Occhipinti and M. G. Xibilia, "Soft computing for greenhouse climate control," *IEEE Trans., Fuzzy Systems*, vol. 8, no. 6, pp. 753-760, Dec. 2000.

[David 94]

A.K. David, "Modelling risk in energy contracts with investor owned generation", *IEE Proc.-Gener. Transm. Distrib.*, vol. 141, no. 1, pp. 75-80, Jan. 1994.

[David 96]

A.K. David, "Risk modelling in energy contracts between host utilities and BOT plant investors", *IEEE Trans Energy Conversion*, vol. 11, no. 2, pp. 359-366, Jun. 1996.

[De Jong 75]

K.A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems," *Ph.D. Thesis*, University of Michigan, Ann Arbor, MI., 1975.

[Drezga 99]

I. Drezga, S. Rahman, "Short-term load forecasting with local ANN predictors," *IEEE Transactions on Power System*, Vol. 14, no. 3, pp. 844-850, Aug. 1999.

[Douligenis 93]

Douligenis C, "Intelligent Home Systems", *IEEE Communications Magazine*, Volume 31, pp. 52-61, Oct. 1993.

[Ferrerira 96]

H.C. Ferrerira, H.M. Grove, O Hooijen and A.J. HanVinc, "Power line communication: Overview," *IEEE AFRICON 4th*, Vol 2, 1996, pp. 558-563.

[Gary 96]

J. Gray, D.G. Caldwell and N. Linge, "A new perspective on the intelligent home," *Impact of Multimedia Services on the Home Environment, IEE Colloquium*, 1996, pp. 7/1-7/4.

[Goguen 67]

J.A.Goguen, "L-fuzzy sets," *Journal of Mathematics, Analysis and Applications*, Vol.18, pp.145-174, 1967.

[Haykin 94]

S. Haykin, *Neural Networks: A Comprehensive Foundation*, NJ: Macmillan, 1994.

[He 97]

Y.Q. He and A.K. David, "Time-of-use electricity pricing based on global optimization for generation expansion planning", *Proc. 4th Int. Conf. Advance in Power Syst. Control, Operation and Management APSCOM-97*, Hong Kong, November 1997, pp. 668-673.

[Heng 98]

E.T.H. Heng, D. Srinivasan and A.C. Liew, "Short term load forecasting using genetic algorithm and neural networks", *1998 International Conference on Energy Management and Power Delivery*, vol. 2, 1998, pp.576-581.

[Holland 75]

J.H. Holland, *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, MI, 1975.

[Hopfield 84]

J.J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. Of Nat. Acad. Sci. U.S.*, Vol. 81, pp. 3088-3092, 1984.

[Hse 92]

Y.Y. Hse and K.L. Ho, "Fuzzy expect systems: an application to short-term load forecasting," *IEE Proceedings-C*, vol. 139, no.6, pp. 471-477, Nov 1992.

[Hsu 91]

Y.Y. Hsu, C.C. Yang, "Design of artificial neural networks for short-term load forecasting. Part 1: Self-organizing feature maps for day type identification," *IEE Proceedings-C*, Vol.138, no.5, pp. 407-418, 1991.

[Jang 97]

R.J.S. Jang, *Neuro-Fuzzy and Soft Computing*, NJ: Prentice Hall, 1997.

[Juidette 00]

H. Juidette and H. Youlal, "Fuzzy dynamic path planning using genetic algorithms," *Electronics Letters*, vol. 36, no. 4, pp. 374-376, Feb. 2000.

[Kosko 91]

B. Kosko, *Neural Networks and Fuzzy System: A Dynamical Systems Approach to Machine Intelligence*. Prentice Hall, 1991.

[Kaimal 97]

M.R. Kaimal, *Neuro-Fuzzy Control System*, New Delhi : Narosa Publishing House, 1997.

[Kohonen 88]

T. Kohonen, *Self-organization and Associative Memory*, Berlin: Springer-Verlag, pp.132, 1988.

[Lam 01a]

H.K. Lam, S.H. Ling, F.H.F. Leung and P.K.S. Tam," Optimal and Stable Fuzzy Controllers for nonlinear Systems subject to Parameter Uncertainties using Genetic Algorithm," *Proceedings of the 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001, Australia,* Dec 2001, pp.908-911.

[Lam 01b]

H.K. Lam, S.H. Ling, F.H.F. Leung and P.K.S. Tam, "Tuning of the Structure and Parameters of Neural Network using an Improved Genetic Algorithm," *Proceedings of the 27th Annual Conference of the IEEE Industrial Electronics Society, IECON'2001, USA*, Nov 2001, pp.25-30.

[Lam 01c]

H.K. Lam, S.H. Ling, K.F. Leung, F.H.F. Leung and P.K.S. Tam, "Interpreting Graffiti Commands for eBooks using a Neural Network and an Improved Genetic Algorithm," *Proceedings of the 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001, Australia,* Dec 2001, pp.1464-1467.

[Lecun 85]

Y. Lecun, "A learning procedure for asymmetric network," *Cognitiva*, pp.599-604, 1985.

[Lee 90]

C.C. Lee, "Fuzzy logic in control systems: fuzzy logic controller –part I & II," *IEEE Transactions on System, Man and Cybernetics*, Vol.20, No. 2, pp.404-435, Aug. 1990.

[Lee 92]

K.Y. Lee, Y.T. Cha, J.H. Park, "Short-term load forecasting using an artificial neural network," *IEEE Transaction on Power Systems*, Vol.7, no.1, pp.124-132, 1992.

[Lee 99]

Po-Wa Lee, Yim-Shu Lee, and Bo-Tao Lin, "Power distribution systems for future homes", *Proc. Int. Conf. Power Electronics and Drive Syst.*, Hong Kong, 1999, vol. 2, pp. 1140-1146.

[Ling 01a]

S.H. Ling, H.K. Lam, F.H.F Leung and P.K.S. Tam, "A Neural Fuzzy Network with Optimal Number of Rules for Short-Term Load Forecasting in an Intelligent Home," *Proceedings of the 10th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'2001, Australia,* Dec 2001, pp.1456-1459.

[Liu 99]

D. Liu, E. Flint, B. Gaucher, and Y. Kwark, "Wide band AC power line characterization," *IEEE Transactions on Consumer Electronics,* Vol. 45, no.4, pp. 1087-1097, Nov.1999.

[Liu 01]

B.D. Liu, C.Y. Chen and J.Y. Tsao, "Design of adaptive fuzzy logic controller based on linguistic-hedge concepts and genetic algorithms," *IEEE Trans. Systems, Man and Cybernetics, Part B,* vol. 31 no. 1, pp. 32-53, Feb. 2001.

[Lu 93]

C.N. Lu, H.T. Wu and S. Vemuri, "Neural network based short-term load forecasting", *IEEE Transaction on Power Systems*, Vol.8, no.1, pp.336-342, Feb.1993.

[Maeda 92]

Maeda and Y. Kaya, "Game theory approach to use of non-commercial power plants under time-of-use pricing", *IEEE Trans. Power Syst.*, vol. 7, no. 3, pp. 1052-1059, Aug. 1992.

[Michalewicz 94]

Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs,* second, extended edition, Springer-Verlag, 1994.

[Momoh 97]

J.A. Momoh, Y. Wang, M. Elfayoumy, "Artifical neutal network based load forecasting," *IEEE International Conference on Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation,* Vol. 4, 1997, pp. 3443-3451.

[Naga Raj 95]

Naga Raj and K.S. Prakasa Rao, "A new fuzzy reasoning approach for load balancing in distribution system", *IEEE Trans. Power Syst.*, vol. 10, no. 3, pp. 1426-1432, Aug. 1995.

[Nauck 97]

D. Nauck, *Neuro-fuzzy systems*, Chichester, England: Wiley, 1997.

[Part 91]

D. Part, M. El-Sharkawi, R. Marks, L. Atlas, and M. Damborg, "Electric load forecasting using an artificial neural network," *IEEE Transactions on Power System*, Vol. 6, no. 2, pp.442-449, 1991.

[Radford 96]

D. Radford, "Spread-spectrum data leap through ac power wiring", *IEEE Spectrum*, pp. 48-53, Nov. 1996.

[Pedrycz 97]

W. Pedrycz, *Computational Intelligent: An Introduction*. Boca Raton, NY: CRC

Press, 1997.

[Rewagad 98]

A.P. Rewagad and V.L. Soanawane, "Artificial neural network based short term load forecasting," *TENCON '98. 1998 IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control,* Vol. 2, 1998, pp.588 –595.

[Rumelhart 86]

D.E. Rumelhart, G.E. Hinton, and R.F. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing,* Vol. 1, Cambridge: MIT Press, pp.318-362, 1986.

[Schickhuber 97]

G. Schickhuber and O. McCarthy, "Control using power lines: a European view," *Computing and Control Engineering Journal,* pp.180-184, Aug.1997.

[Setnes 00]

M. Setnes and H. Roubos, "GA-fuzzy modeling and classification: complexity and performance," *IEEE. Trans, Fuzzy Systems*, vol. 8, no. 5, pp. 509–522, Oct. 2000.

[Sheen 94]

J.N. Sheen, C.S. Chen, and J.K. Yang, "Time-of-use pricing for load management programs in Taiwan power company", *IEEE Trans. Power Syst.*, Vol. 9, no. 1, pp 388-396, Feb. 1994.

[Sheen 95]

J.N. Sheen, C.S. Chen, and T.Y. Wang, "Response of large industrial customers to electricity pricing by voluntary time-of-use in Taiwan", *IEE Proc.-Gener. Transm.*

Distrib., vol. 142, no. 1, pp. 157-166, Mar. 1995.

[Wang 96]

Xiufeng Wang and M., Elbuluk, "Neural network control of induction machines using genetic algorithm training," *Industry Applications Conference, 1996. Thirty-First IAS Annual Meeting, IAS '96., Conference Record of the 1996 IEEE*, vol. 3, 1996, pp. 1733-1740.

[Wang 97]

L.X. Wang, *A Course in Fuzzy System and Control*, USA: Prentice Hall, 1997.

[Wong 94]

E.M.C. Wong, "A phone-based remote controller for home and office automation," *IEEE Transactions on Consumer Electronics*, Vol.40, No.1, Feb. 1994, pp. 148-152.

[Wong 00]

L.K. Wong, S.H. Ling, F.H.F. Leung, Y.S. Lee, S.H. Wong, T.H. Lee, H.K. Lam, K.H Ho, D.P.K. Lun, T.C. Hsung, "An intelligent home," in *Proc. Workshop n Service Automation and Robotics, Hong Kong*, June 2000, pp. 111-119.

[Yao 99]

G.X. Yao and Y. Liu "Evolutionary Programming made Faster," *IEEE Trans., on Evolutionary Computation*, vol. 3, no. 2, pp.82-102, July 1999.

[Zadeh 65]

L.A. Zadeh, "Fuzzy sets," *Information and Control*, Vol. 8, pp.338-353, 1965.

[Zhou 97]

Q. Zhou, D. Shirmohammadi, and W.H.E. Liu, "Distribution feeder reconfiguration for service restoration and load balancing", *IEEE Trans. Power*

Syst., vol. 12, no. 2, pp. 724-729, May 1997.

[Zhou 00]

Y.S. Zhou and L.Y. Lai, "Optimal design for fuzzy controllers by genetic algorithms," *IEEE Trans., Industry Applications*, vol. 36, no. 1, pp. 93-97. Jan.-Feb, 2000.