

A Solution of Short Term Unit Commitment in Smart Grid Using Artificial Neural Network and Fuzzy Logic

^aC.Sailaja , ^bR.Sadhasivam

^aPG Student, ^bAssistant Professor

Department of Electrical and Electronics Engineering M.P.N.M.J Engineering College Chennimalai, Erode,
Tamilnadu – 638112, India

Abstract: The unit commitment problem is proposed to be solved by the Artificial Neural Network and fuzzy systems in smart grid using distributed energy storage systems. The main objective of short term load forecasting is to provide load predictions for generation scheduling, economic load dispatch and security assessment at any time. With the increased attraction of medium and smaller scale systems the expansion of the integration of intermittent renewable resources in smart grids. The effectiveness of the economic dispatch is well understood where the objective is to schedule the committed generators to meet the load. This paper is intended to offer a tool for analyzing potential advantages of distributed energy storages in Smart Grids with reference to both different possible renewable resources at the distributed energy environment. The objective of this paper is to commit the units with the AI techniques back propagation neural network and Fuzzy logic without violating the constraints and to compare the results in order to find the best suitable method.

Index Terms: Unit commitment, Smart grids, Distributed energy storage, network, Artificial Neural Network and Fuzzy System .

I. Introduction

Among these innovative alternatives, Distributed Energy Storage Systems (DESS) is nothing but the grid offer interesting features and have received considerable attention lately. However, they still remain capital-intensive and raise cost-competitiveness issues. This paper deals with the potential of DESS to support and to optimize distribution system operation. First of all, some general information about DESS and their technical performances is given. Then, an overview of their applications in liberalized power systems is drawn (for DSO and other stakeholders). In the last part, a new approach to the combination of storage benefits is introduced and its use within the framework of this project is briefly discussed.

In any power system, the load is dynamic in nature. It is higher during the daytime and early evening when industrial loads are high; lights are on, and so for the and lower during the late evening and early morning when most of the population is asleep. The load variation is continuous and the load must be met with the available resources economically. This is done by committing (switching ON) and decommitting (switching OFF) of the units in power station. By running only the most economic units, the load can be supplied to the best efficiency of unit operators.

Thus committing the correct number and kind of units such that the load is met at least operating cost. There have been many methods that are available to solve the unit commitment problem such that the Lagrangian method, Dynamic Programming (DP), branch and bound technique, simulated annealing, Priority listing and Advanced Priority listing method. The DP method based on priority list is flexible, but its computational time suffers from dimensionality. Lagrangian relaxation for UCP is superior to DP due to its higher solution quality and faster computational time. However, numerical convergence and solution quality of LR do not give satisfactory results in case identical units exist. These methods though may give results but does not give a qualitative interpretation of the results in terms of input variables. The neural network computing enhanced by expert systems has opened up a new route for the optimization of generation scheduling. With proper and sufficient training, the information regarding the optimal operation of a system can be stored in the network as such, and the output can be obtained in a much shorter time. In the problem, multilayer feed forward network using back propagation error of learning determine variables corresponding to the operating level of generators and production cost. Load demand profile is input to the neurons in the input layer. Generation is the output to the neurons in the output layer. The fuzzy system involves the fuzzification and defuzzification process with the certain membership functions to obtain the predicted load demand.

II. Modelling Of Neural Network

Neural networks are a very popular data mining and image processing tool. Their origin stems from the attempt to model the human thought process as an algorithm which can be efficiently run on a computer. The human brain consists of neurons that send activation signals to each other (figure on the left) thereby creating

intelligent thoughts. The algorithmic version of a neural network (called an ANN) also consists of neurons which send activation signals to one another (figure below). The end result is that the ANN can approximate a function of multiple inputs and outputs. As a consequence, neural networks can be used for a variety of data mining tasks, among which are classification, descriptive modeling, clustering are the various function approximation, and time series prediction. Neural networks are also commonly used for image processing.

We will first see how to compute the output of an already learned neural network which means that the entire network (topology, activation functions, weights, and biases) is known at this point. In a later section, we will see how to construct a neural network to approximate any given function, but for now just assume that the neural network is given and that it has been trained. It might be helpful to refer to the figure. 2. While reading this section. Recall that the purpose of a feed-forward neural network is to approximate a function of multiple inputs and outputs. The inputs and outputs are continuous (real values). Each input to the neural network is fed as input into a different input neuron. The input neurons are connected to hidden neurons which perform transformations on the input. At the end of the graph are the output neurons. The output of these neurons is defined to be the output of the entire network. It is also possible to connect an input neuron directly to an output neuron (without any hidden neurons in between) but that is usually not a good idea because there are no hidden neurons to perform transformations on the data and then it would not be possible to approximate a function accurately.

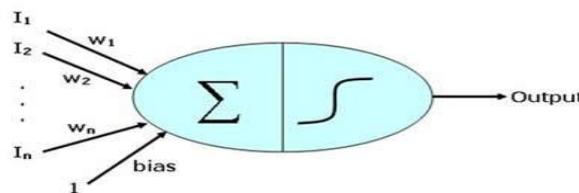


Figure 1. Processing of Output With Weights And Biases

III. Training Of Network Model

The steps for computing the output of a single neuron are as follows:

- (1) Compute the weighted sum of inputs to the neuron.
- (2) Add the bias to the sum.
- (3) Feed the sum as an input to the activation function of the neuron.

The output of the activation function is defined to be the output of the neuron. These steps can be summarized in the following the formula: $Output = A(\sum w_k * I_k + bias)$ where A is the activation function of the neuron, w_k is the weight of the kth in-edge, I_k is the input carried across the kth in-edge, and bias is the bias of the neuron.

IV. Formation Of Weights & Biases

The algorithm for learning an instance $(X_i, Y_i) = ((x_1, x_2, \dots, x_u), (y_1, y_2, \dots, y_v))$ can be split into three phases: Calculation of Z_i , the output of the network for the training instance's input set X_i . The bias can be calculated with Y_i is the desired value for Z_i , so when they are different, there is some error. From this error, we wish to compute a bias value for each neuron. Blames will later be used to decide by how much to adjust weights and biases. The figure above illustrates this step. The calculation of bias for Output Neurons, The bias of each output neuron is calculated as $y_i - z_i$. Note that even though our goal is to minimize the sum squared error of the entire network, the blames are just the linear error. This linearity is due to the fact that we need the bias to be proportional to partial derivatives which are needed for the gradient descent. Calculation of bias for input and hidden neurons. The bias for an input or hidden neuron is calculated in terms of the blames of its out-neurons (the ones that it is pointing to). Because of this dependency, the blames need to be calculated in a topologically sorted order of the reverse of the network topology graph. This is similar to the order in which the output of neurons is calculated except that the topology graph must first be reversed before topologically sorting. The formula for the bias of an input or hidden neuron is $\sum w_k * e_k$ where e_k is the bias of an out-neuron and w_k is the weight of the edge that connects to the out-neuron. This step of propagating the error backwards is why the learning algorithm is called back-propagation. The adjustment of weights and biases are this last step performs the actual gradient descent by adjusting the weights with the formula: $w_{ij} = w_{ij} + r * e_j * A_j'(I_j) * O_i$ where r is the learning rate, e_j is the bias of neuron j, A_j' is the derivative of neuron j's activation function, I_j is the input fed to neuron j during calculation of the output in the first step, and O_i is the output of neuron i during the first step. The biases are similarly adjusted using the formula: $bias_i = bias_i + r * e_i$. After all training instances are run, learning is complete. However, sometimes we may wish to run the training data multiple times through the learning algorithm to achieve better accuracy on the training data. When using this technique, it is important to make sure that over fitting does not occur.

V. Formulation Of Back Propagation Algorithm

STEP 1 : Read the input values(X),target values(t),learning rate coefficient ,tolerance limit of error, momentum constant, maximum number of iterations(nitr),neurons of hidden layer(n),input nodes(m),output nodes(z),maximum value of input(Xmax),maximum value of target(tmax).

STEP 2: Normalise the input and target vectors

STEP 3: Activate the node by input of the given set of the data.

STEP 4: Generate the random weights

STEP 5: Set all the elements of error matrices dW1 and dW2 equal to zero.

STEP 6: Set epoch=0

STEP 7: Set serr=0

STEP 8: Calculate the output Y1 using

$$Y1(k) = X(1, i) * W1(i, k)$$

Where $i=1, 2, \dots, m$ and $k=1, 2, \dots, n$

STEP 9: Process the output of the hidden layer through a nonlinear activation function to produce output of hidden layers.

$$Y11(k) = 1 / (1 + \exp(-Y1(k)))$$

Where $k=1, 2, \dots, n$

STEP 10: Calculate the final output using

$$Y2(j) = Y11(1, k) * W2(k, j)$$

Where $j=1, 2, \dots, z$ and $k=1, 2, \dots, n$

STEP 11: Process again through the sigmoid function to find output of the network.

$$Y22(j) = 1 / (1 + \exp(-Y2(j)))$$

Where $j=1, 2, \dots, z$

STEP 12: Calculate the square error as

$$e(s, 1) = t(s, 1) - Y22(s, 1)$$

$$serr += e(s, 1) * e(s, 1)^r$$

Where $s=0$

STEP 13: Adjust the weight of the output layer

$$d12(j) = Y22(j) * (1 - Y22(j)) * e(s, 1)^r$$

$$dW2(j) = n * d12(j) * Y22(j) + a * dW2(j, 1)$$

Where $j=1, 2, \dots, z$

STEP 14: Adjust the weight of the hidden layers

$$Sum_w = 0$$

$$Sum_w += d12(j) * W2(j, 1)$$

$$d11(k) = Sum_w * Y11(k) * (1 - Y11(k))$$

$$dW1(i, k) = n * d11(k) * X(1, i)$$

$$+ a * dW1(i, k)$$

$$W1(i, k) += dW1(i, k)$$

STEP 15: Check for tolerances if serr is greater than the convergence rate then goto step

STEP 16: Test the stopping condition. The stopping condition may be minimization of errors, number of epochs etc.

VI. Fuzzy Logic Implementation

Fuzzy logic provides not only a meaningful and powerful representation for measurement of uncertainties but also a meaningful representation of blurred concept expressed in normally language. Fuzzy logic is a mathematical theory, which encompasses the idea of vagueness when defining a concept or a meaning. For example, there is uncertainty or fuzziness in expressions like `large` or `small`, since these expressions are imprecise and relative. Variables considered thus are termed `fuzzy` as opposed to `crisp`. Fuzziness is simply one means of describing uncertainty. Such ideas are readily applicable to the unit commitment problem.

VII. FUZZY SET ASSOCIATED WITH THE UNIT-COMMITMENT

After identifying the fuzzy variables associated with unit commitment, the fuzzy sets defining these variables are selected and normalized between 0 and 1. This normalized value can be multiplied by a selected scale factor to accommodate any desired variable.

LCG = {Low, Below Average, Average, Above Average, High}

The incremental cost is stated by the following sets:

IC = {Zero, Small, Large}

The sets representing the start-up cost are shown below:

SUC = {Low, Medium, High}

The production cost, chosen as the objective function, is given by:

PRC= {Low, Below Average, Average, Above Average, High}

S.N O	Pmax	Pmin	a	b	c	Incremental fuel cost
1	455	150	1000	16.19	0.00048	16.7
2	455	150	970	17.26	0.00031	16.26
3	130	20	700	16.6	0.002	16.19
4	130	20	680	16.5	0.00211	16.2

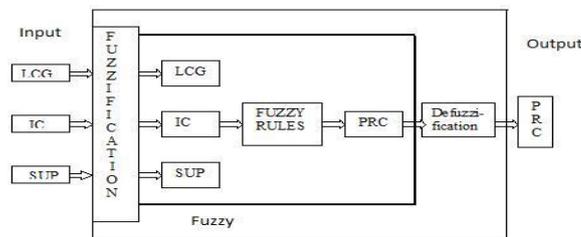


Figure.2 Block Diagram of Unit Commitment

VIII Experimental Results And Discussions

In this case, number of units system has been tested in order to prove the applicability of the proposed method for solving the UC problem with mixed generating units system. The intermittent data of number of units system is obtained using distributed energy stored system has been proposed and given in Table1. The constraints are formulated using the distributed energy system and trained neural network and fuzzy system model has been shown. Scheduling of the proposed method for various combinations of units system is given in output of the simulation results of combination of different units in terms of the overall fuel cost are given as output. It is clear from the simulation results that the proposed approach provides better solution. Here four units are taken for case study.

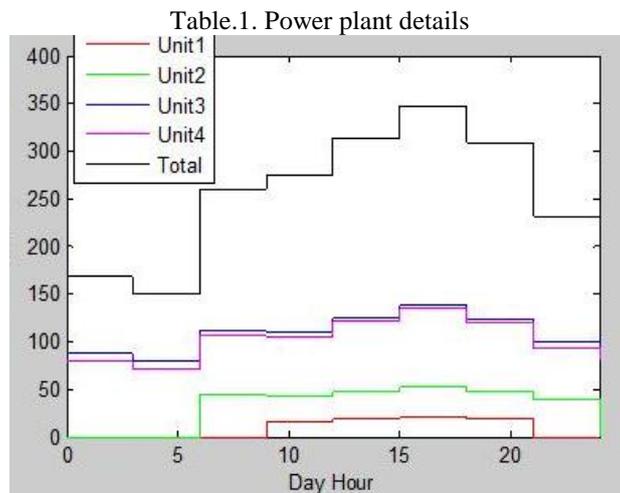


Figure.3 Unit Commitment Neural Network Approach

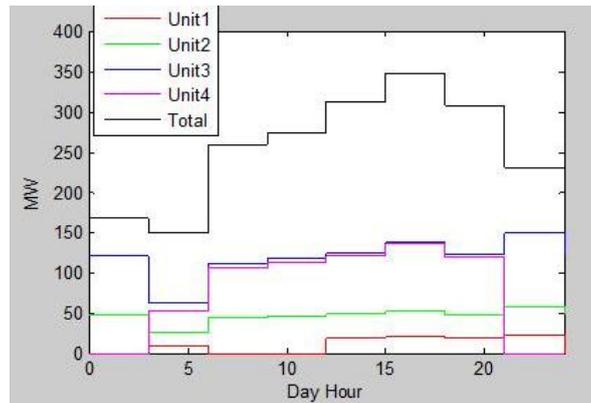


Figure.5 Unit Commitment Fuzzy-Logic-Based Approach

IX. Conclusion

A Neural network and fuzzy-logic-based approach has been developed for solving the unit commitment problem of a four-unit power system to provide feasible unit combinations for every period of time. The daily load demand is divided into eight periods of time and during each period the load demand remains constant. A Matlab code has also been developed to handle the problem and after running the code, the approach has been successfully implemented and feasible unit combinations are obtained for each time period that satisfies the system constraints.

References

- [1] Chandrasekaran.K,S. Hemamalini,Sishaj P. Simona, Narayana Prasad Padhye a National Institute of Technology, Tiruchirappalli, India,b VIT University, Chennai, India c. Indian Institute of Technology, Roorkee, India on Thermal unit commitment using binary/real coded artificial bee colony algorithm.
- [2] ChenthurPandiyam.S and Dr.K.Duriswamy ,”Fuzzy Logic Implementation for solving the unit commitment problems”,2004 International Conference on Power Systems Technology”,POWERCON 2004 Singapore,21-24 November 2004.
- [3] Chung-Ching Su and Yuan-Yih Hsu,”Fuzzy “Dynamic Programming: An Application to Unit Commitment “Transactions on power systems, Vol-6, No.3, August 1991.
- [4] Chung-Li Tseng, Shmuel S. Oren,Department of IEOR, UC Berkeley, Berkeley,CA720,USA,tseng@ieor.berkeley.edu,shmuel@euler.berkeley.edu Carol S. Cheng, Chao-an Li, Alva J. Svoboda,on A Transmission-Constrained Unit Commitment Method.
- [5] Dimitris Bertsimas, Member, IEEE, Eugene Litvinov, Senior Member, IEEE, Xu Andy Sun, Member, IEEE, Jinye Zhao, Member, IEEE, and Tongxin Zheng, Senior Member, IEEE on Adaptive Robust Optimization for the Security Constrained Unit Commitment Problem.
- [6] Elgerd.O.L , “Electric energy systems theory: an introduction “,Mc-Graw –Hill Book Company ,NewYork,1971,pp.294-296.Sayeed Salam,”Unit Commitment solution methods”, proceedings of the world academy of Sciences engineering and technology,volume 26,December 2007,ISSN 1307-6884.
- [7] George J.Klir / Boyuan ,”Fuzzy Sets and Fuzzy Logic”,Prentice Hall of India Private Limited,NewDelhi-2000.
- [8] Hadi Sadat,”Power system analysis “, Mc Graw hill publication,NewDelhi,1997.
- [9] Indumathi.C,Latha.E Mercy and S.Ravichandran,”A Fuzzy based technique for Unit commitment “,IEEE transactions power apparatus and systems,2004.
- [10] Kory W. Hedman, Student Member, IEEE, Michael C. Ferris, Richard P. O’Neill, Emily Bartholomew Fisher, Student Member, IEEE, and Shmuel S.Oren, Fellow, IEEE on Co-Optimization of Generation Unit Commitment and Transmission Switching With N-1 Reliability.
- [11] Kothari.D.P ,”Application of Neural Networks to Power Systems “,Center of Energy Studies,NewDelhi,pp 621-626.
- [12] Mohammad mohatram,Sanjay Kumar,”Application of ANN in economic generation scheduling of thermal power plants”
- [13] MohammadJamshidi,NaderVadice and T.J.Ross ,”Fuzzy Logic and Control”,New Jersey,1993.