Chapter 7

Soft Computing based WSN

This chapter briefs about the Soft Computing Techniques such as Artificial Neural Network (ANN), Adaptive Neuro Fuzzy Inference System (ANFIS). Performance of WSN is evaluated based on these Soft Computing Techniques. Optimization using ANN and ANFIS decide the Packet size of data transmitted by WSN Motes.

7.1 Soft Computing Models

Soft computing is an emerging approach to computing which parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision [1]. It is a parasol term for a collection of computing techniques. Soft Computing (SC), a term originally coined by Zadeh (1994) is a sub-field of Artificial Intelligence. In his words: "Soft computing differs from conventional (hard) computing in that, unlike hard computing, it is tolerant of imprecision, uncertainty and partial truth. In effect, the role model for soft computing is the human mind. The guiding principle of soft computing is: Exploit the tolerance for imprecision, uncertainty and partial truth to achieve tractability, robustness and low solution cost". The main techniques in soft computing are evolutionary computing, artificial neural networks and fuzzy logic. Each technique can be used separately, but a powerful advantage of soft computing is the complementary nature of the techniques. Used together they can produce solutions to problems that are too complex or inherently noisy to tackle with conventional mathematical methods. Soft computing can be used to address a very wide range of problems in all industries and business sectors. In general Soft Computing is a good option for complex systems [2], where:

- The system is non-linear, time-variant or ill defined.
- The variables are continuous.
- A mathematical model is either too difficult to encode, does not exist or is too complicated and expensive to be evaluated.
- There are noisy or numerous inputs.
- An expert is available who can outline the rules-of-thumb that should determine the system behaviour.

7.2 Artificial neural networks

One type of network sees the nodes as 'artificial neurons'. These are called artificial neural networks (ANNs). An artificial neuron is a computational model inspired in the natural neurons. Natural neurons receive signals through *synapses* located on the

dendrites or membrane of the neuron. When the signals received are strong enough (surpass a certain *threshold*), the neuron is *activated* and emits a signal though the *axon*. This signal might be sent to another synapse, and might activate other neurons. The complexity of real neurons is highly abstracted when modelling artificial neurons. These basically consist of *inputs* (like synapses), which are multiplied by *weights* (strength of the respective signals), and then computed by a mathematical function which determines the *activation* of the neuron. Another function (which may be the identity) computes the *output* of the artificial neuron (sometimes in dependence of a certain *threshold*). ANNs combine artificial neurons (as shown in Figure 7.1) in order to process information [3].

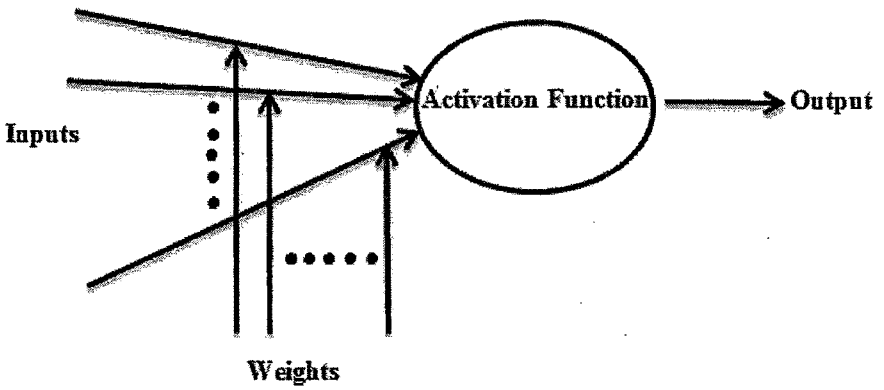

Figure 7.1: an Artificial Neuron

There are different ways of defining what the ANN are, from short and generic definitions to the ones that try to explain in a detailed way what means a neural network or neural computing. For this situation, the definition that was proposed by Teuvo Kohonen appears below:

"Artificial Neural Networks are massively interconnected networks in parallel of simple elements (usually adaptable), with hierarchic organization, which try to interact with the objects of the real world in the same way that the biological nervous system does."

Basic building block of every artificial neural network is artificial neuron, that is, a simple mathematical model (function) [4, 5, 6]. Such a model has three simple sets of rules: multiplication, summation and activation. At the entrance of artificial neuron the inputs are weighted what means that every input value is multiplied with individual weight. In the middle section of artificial neuron is sum function that sums all weighted inputs and bias. At the exit of artificial neuron the sum of previously weighted inputs and bias is passing through activation function that is also called transfer function which defines the properties of artificial neuron and can be any mathematical function. We

choose it on the basis of problem that artificial neuron (artificial neural network) needs to solve and in most cases we choose it from the set of functions: *Step function, Linear function* and *Non-linear (Sigmoid) function.*

7.2.1 Basics of Artificial Neural Networks (ANN)

ANN consists of two or more number of artificial neurons. ANN solves complex real-life problems by processing information in a non-linear, distributed, parallel and local way. The way that individual artificial neurons are interconnected is called topology under layers viz., input, hidden, output which represents architecture or graph of an ANN.

7.2.2 Types of ANN Learning

There are three major learning paradigms; supervised learning, unsupervised learning and reinforcement learning. Usually they can be employed by any given type of artificial neural network architecture. Each learning paradigm has many training algorithms.

- **Supervised learning**

Supervised learning is a machine learning technique that sets parameters of an artificial neural network from training data. Supervised learning is a process of training a neural network by giving it examples of the task we want it to learn. i.e., it is learning with a teacher. The way this is done is by providing a set of pairs of vectors (patterns), where the first pattern of each pair is an example of an input pattern that the network might have to process and the second pattern is the output pattern that the network should produce for that input which is known as a target output pattern for whatever input pattern.

Thus, supervised leaning means "a learning process in which, change in a network's weights and biases are due to the intervention of any external teacher. The teacher typically provides output targets."

- **Unsupervised learning**

Unsupervised learning is a machine learning technique that sets parameters of an artificial neural network based on given data and a cost function which is to be minimized. Cost function can be any function and it is determined by the task formulation. Unsupervised learning is mostly used in applications that fall within the domain of estimation problems such as statistical modelling, compression, filtering, blind source separation and clustering. In unsupervised learning we seek to determine how the

data is organized. It differs from supervised learning and reinforcement learning in that the artificial neural network is given only unlabelled examples.

- **Reinforcement learning**

Sometimes if it is not require computing exact error between the desired and the actual network response, and for each training example the network is given a pass/fail signal by the teacher, then it is called Reinforcement learning which is a special type of supervised learning. If a fail is assigned, the network continues to readjust its parameters until it achieves a pass or continues for a predetermined number of tries, whichever comes first.

7.2.3 Back Propagation Network (BPN)

Back propagation is a systematic method for training multi-layer artificial networks [7]. It has a mathematical foundation that is strong if not highly practical. It is a multi-layer forward network using extend gradient descent based delta learning rule, commonly known as back propagation (of errors) rule.

Back propagation provides a computationally efficient method for changing the weights in a feed forward network, with differentiable activation function units, to learn training a set of input-output examples. Being a gradient descent method it minimizes the total error of the output computed by the net. The network is trained by supervised learning method. The aim of this network is to be train the net to achieve a balance between the ability to respond correctly to the input patterns that are used for training and the ability to provide good response to the input that are similar. The training algorithm of back propagation involves four stages, viz.

1. Initialize of weights
2. Feed forward
3. Back propagation
4. Updating of the weights and biases.

7.3 Adaptive Neuro Fuzzy Inference System (ANFIS)

The acronym ANFIS derives its name from adaptive neuro-fuzzy inference system. Using a given input/output data set, the toolbox function anfis constructs a fuzzy inference system (FIS) whose membership function parameters are tuned (adjusted) using either a back-propagation algorithm alone or in combination with a least squares type of method.

ANFIS is a network-type structure similar to that of a neural network, which maps inputs through input membership functions and associated parameters, and then through output membership functions and associated parameters to outputs, can be used to interpret the input/output map. The parameters associated with the membership functions will change through the learning process. The computation of these parameters (or their adjustment) is facilitated by a gradient vector, which provides a measure of how well the fuzzy inference system is modelling the input/output data for a given set of parameters.

7.4 Matlab Development tools

The development tools such as MATLAB, SIMULINK and various tools boxes are covered in the following section.

7.4.1 MATLAB

MATLAB [9] is a high-level language and interactive environment that enables you to perform computationally intensive tasks faster than with traditional programming languages such as C, C++, and FORTRAN.

- **Desktop Tools and Development Environment:** - This is the set of tools and facilities that help to use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.
- **The MATLAB Mathematical Function Library:** - This is a vast collection of computational algorithms ranging from elementary functions, like sum, sine, cosine, and complex arithmetic.
- **The MATLAB Language:** - This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features.
- **External Interfaces:** - The eternal interfaces library allows to write c/c++ and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), for calling MATLAB as a computational engine, and for reading and writing MAT-files.
- **Graphics:** - MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image

processing, animation, and presentation graphics. It also includes low-level functions that allow to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

The main MATLAB window is given in Figure 7.2

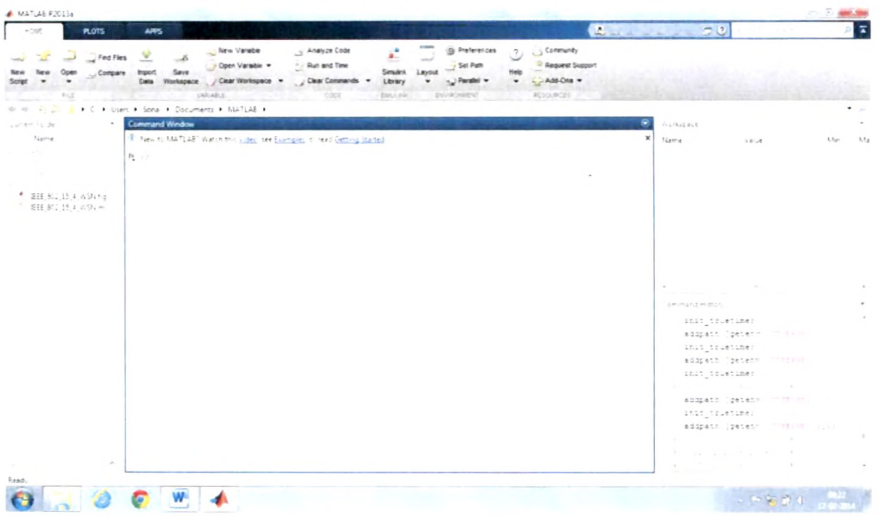

Figure 7.2: MATLAB Command Window

7.4.2 Simulink

Simulink [9] is an environment for multidomain simulation and Model-Based Design for dynamic and embedded systems. It provides an interactive graphical environment and a customizable set of block libraries that let you design, simulate, implement, and test a variety of time-varying systems, including communications, controls, signal processing, video processing, and image processing.

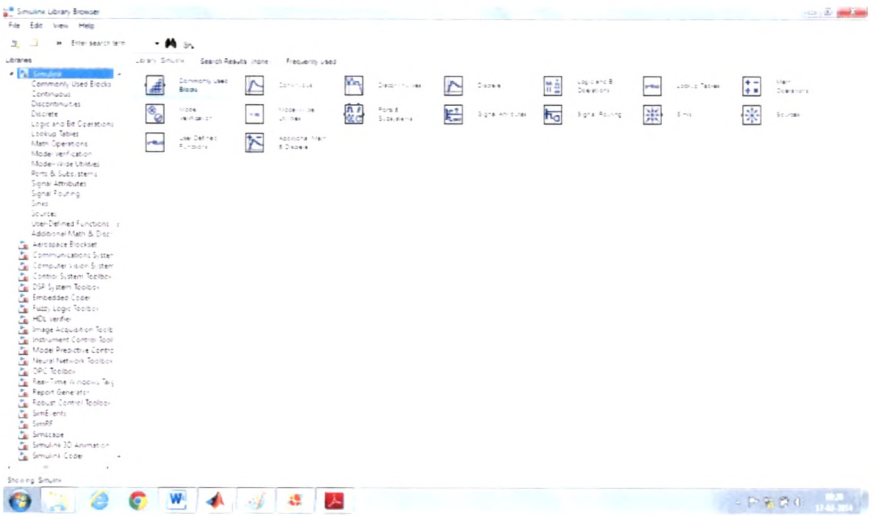

Figure 7.3: Simulink View

The default Simulink view is given in Figure 7.3.

7.4.3 Toolboxes

✂ Neural network toolbox

There are four different levels at which the Neural Network Toolbox software can be used.

- ❖ The first level is represented by the Graphical User Interfaces. These provide a quick way to access the power of the toolbox for many problems of function fitting, pattern recognition, clustering and time series analysis.
- ❖ The second level of toolbox use is through basic command-line operations. The command-line functions use simple argument lists with intelligent default settings for function parameters.
- ❖ A third level of toolbox use is customization of the toolbox. This advanced capability allows you to create your own custom neural networks, while still having access to the full functionality of the toolbox.
- ❖ The fourth level of toolbox usage is the ability to modify any of the M-files contained in the toolbox. Every computational component is written in MATLAB code and is fully accessible.

✂ ANFIS Editor

To start ANFIS editor [10] GUI, on the MATLAB command prompt type the following.

>>anfisedit

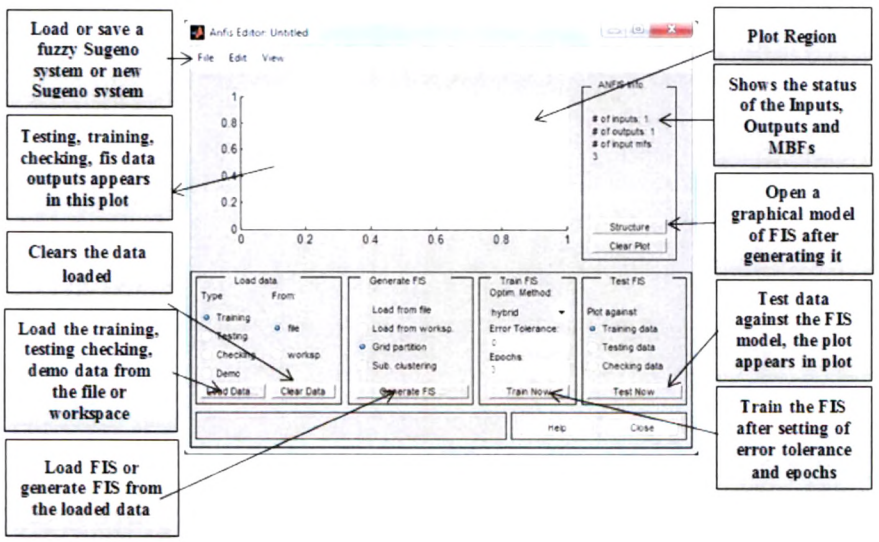

Figure 7.4: ANFIS Editor GUI

The ANFIS Editor GUI is shown in Figure 7.4. The main functions in the anfisedit are the loading of the training data from the work space. Then after by loading the training data into the anfiseditor select the sub clustering button to generate the FIS model. After doing these steps now select the training algorithm and error tolerance and number of epochs. Now press the train button which will train the Neuro-fuzzy controller and it generates the training rules itself.

✂ GENERAL DESIGN METHODOLOGY

1) Analyze the problem and find whether it has sufficient elements for a neural network solution. Consider alternative approaches. A simple DSP/ASIC based direct solution may be satisfactory.
2) If the ANN is to represent a static function, then a three-layer feed forward network should be sufficient. For a dynamic function select either a recurrent network or a time delayed network. Information about the structure and order of the dynamic system is required.
3) Select input nodes equal to the number of input signals and output nodes equal to the number of output signals and a bias source. For a feed forward network, select the initially hidden layer neurons typically mean of input and output nodes.
4) Create an input/output training data table. Capture the data from an experimental plant or simulation results, if possible.
5) Select input scale factor to normalize the input signals and the corresponding output scale factor for denormalization.
6) Select generally a sigmoidal transfer function for unipolar output and a hyperbolic tan function for bipolar output.
7) Select a development system, such an s Neural Network Toolbox in MATLAB.
8) Select appropriate learning coefficients and momentum factor.
9) Select an acceptable training error and a number of epochs. The training will stop whichever criterion was met earlier.
10) After the training is complete with all the patterns, test the network performance with some intermediate data points.
11) Finally, download the weights and implement the network by hardware or software.

7.5 ANN based GTS Mechanism

Analysis of GTS mechanism (discussed in chapter 6) is evaluated for enhanced GTS throughput using Artificial Neural Network (ANN) [8, 11] soft computing technique in OPNET Modeler, which can be achieved due to the packet size based on data rate and Interarrival time.

Here two inputs are considered to ANN for optimizing the GTS throughput as shown in Figure 7.5. A packet interarrival time and data rate input decides the best packet size output through trained ANN. Since it is possible to generate I/O training pairs from the existing environment and classical procedures, a feed forward ANN with multi layers Perception model, with 2 nodes in the input Layer, 10 nodes in hidden layer and 1 node in the output layer is selected. The input layer, hidden and output layers have a sigmoid tan-type activation function to produce outputs.

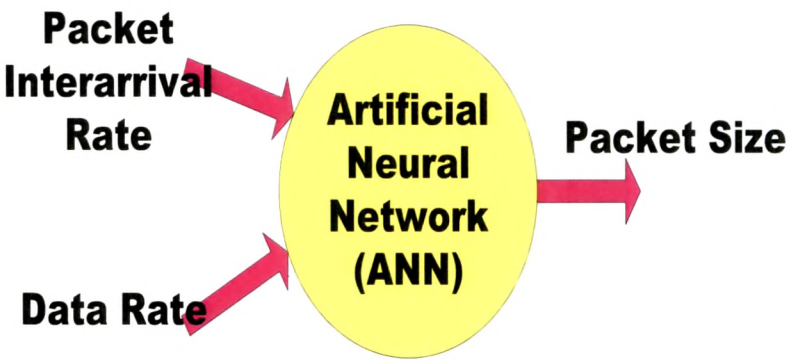

Figure 7.5: ANN with respective inputs and outputs

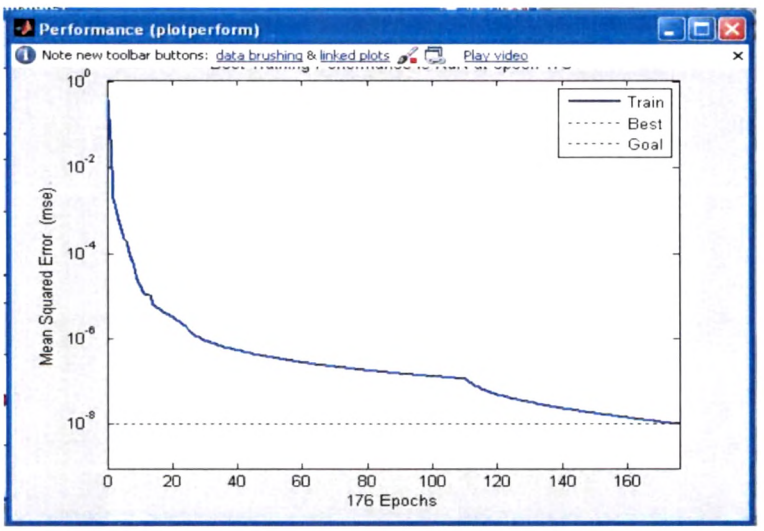

Figure 7.6 Training of Artificial Neural Network for Packet size

Training of neural network tries to achieve the goal of 1E-5 within 20 epochs; the goal performance is 9.95 e- 5.The Figure 7.6 shows the training of the neural network.

7.5.1 Implementation of ANN GTS MECHANISM[1,2]

The simulation results were carried out using OPNET simulator and MATLAB to evaluate the performance of the GTS Mechanism where Simulation model has one pan coordinator and one end device (GTS enabled). This configuration is discussed in chapter 6.

The impact of data Rate, Packet Size , Packet Interarrival time, Buffer Size is evaluated on GTS throughput using traditional and soft computing method for different values of SO (=BO) [12]. The intense of this section is to optimize the GTS throughput during one time slot of GTS when inter arrival time is set to 0.001824 and for different values of the SO [13]. Since the frames are transmitted without acknowledgements, the underutilized bandwidth can only result from IFS or from intermittent data arrival at the buffer.

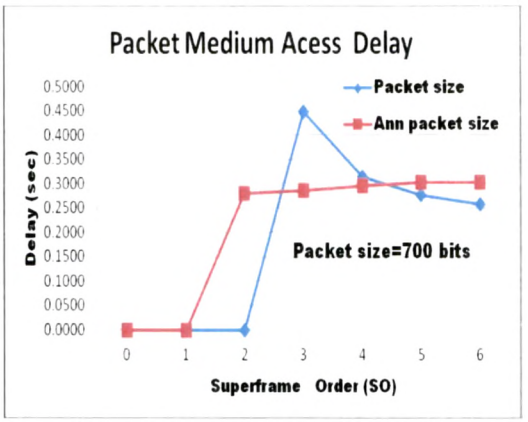

Figure 7.7(a): Packet Medium Access Delay v/s Superframe Order for 700 bits Packet Size

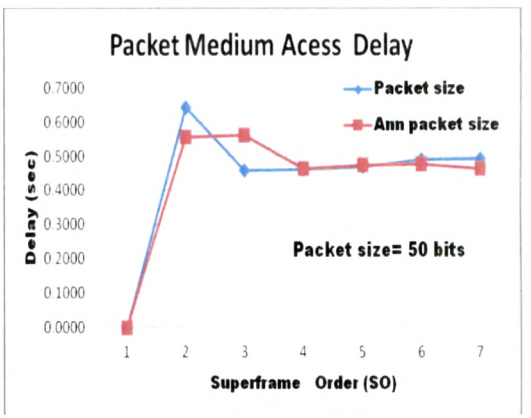

Figure 7.7(b): Packet Medium Access Delay v/s Superframe Order for 50 bits Packet Size

Figure 7.7(a) and (b) plot the packet medium access delay (sec). The lowest delay is achieved for SO values equal to 2, 3. For SO values higher or equal to 5, all frames

[1] **Present a paper: Ms. Sonal J. Rane, Prof. Satish K. Shah, Ms. Dharmistha D Vishwakarma, "Soft Computing Technique Based Throughput Optimization of GTS mechanism for IEEE 802.15.4 Standard" in International Conference on Soft Computing and Software Engineering 2013 (SCSE'13), in San Francisco, California, USA.**

[2] **Published a paper: Ms. Sonal J. Rane, Prof. Satish K. Shah, Ms. Dharmistha D Vishwakarma, " Soft Computing Technique Based Throughput Optimization of GTS mechanism for IEEE 802.15.4 Standard" in International Journal of Soft Computing and Software Engineering [JSCSE], Vol. 3, No. 3, pp. 668-673, 2013, Doi: 10.7321/jscse.v3.n3.102**

stored in the buffer and transmitted during one GTS and the delay grows with SO. For higher values of SO, delay will increase.

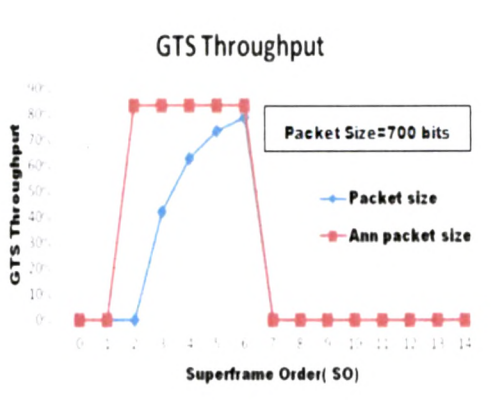

Figure 7.8(a): GTS Throughput v/s Superframe Order for 700 bits Packet Size

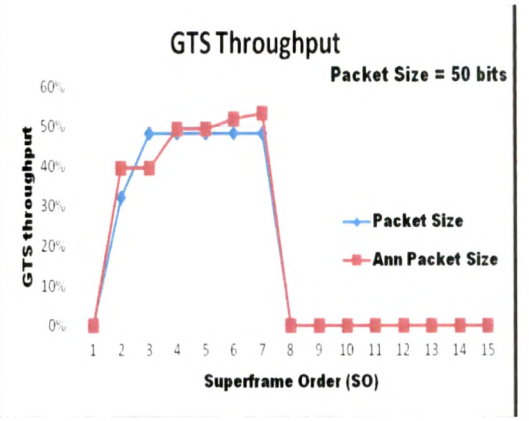

Figure 7.8(b): GTS Throughput v/s Superframe Order for 50 bits Packet Size

Figures 7.8(a) and (b) plot the GTS throughput for traditional GTS mechanism and Soft GTS mechanism. Figure 7.9 (a) and (b) show the Wasted Bandwidth for Traditional and Soft GTS mechanism.

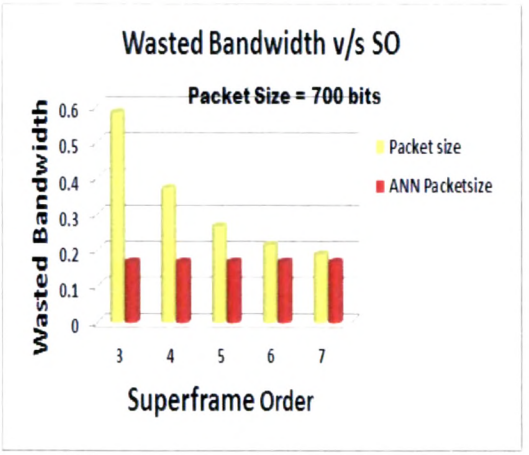

Figure 7.9(a): Wasted B.W. v/s Superframe Order for 700 bits Packet Size

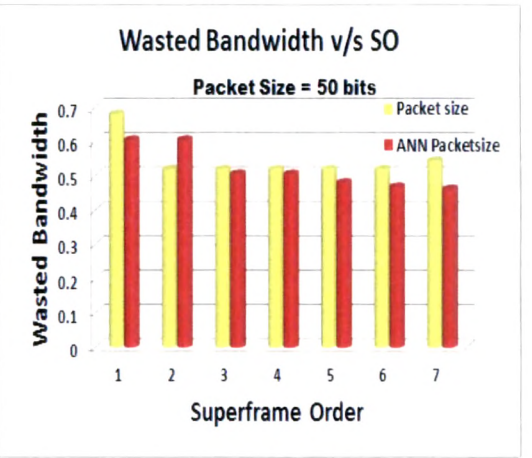

Figure 7.9(b): Wasted B.W. v/s Superframe Order for 50 bits Packet Size

It can observe that when Soft GTS mechanism is applied, GTS throughput is increased, Wasted Bandwidth and PMAD decrease for different SO values compare to Traditional Simulation method. That means considered parameters are optimized after applying the soft computing method to predict the packet size.

7.6 ANFIS based GTS Mechanism

ANFIS is a network-type structure similar to that of a neural network, which maps inputs through input membership functions and associated parameters, and then through output membership functions and associated parameters to outputs, can be used to interpret the input/output map. The parameters associated with the membership functions will change through the learning process. The computation of these parameters (or their adjustment) is facilitated by a gradient vector, which provides a measure of how well the fuzzy inference system is modelling the input/output data for a given set of parameters [13].

Once the gradient vector is obtained, any of several optimization routines could be applied in order to adjust the parameters so as to reduce some error measure (usually defined by the sum of the squared difference between actual and desired outputs). ANFIS uses either back propagation or a combination of least squares estimation and gradient method for membership function parameter estimation. The GTS Mechanism is also optimized using ANFIS [14] with same inputs. The implementation of ANFIS is accomplished using ANFIS editor available in MATLAB. The Sugeno type of inference system is used.

A packet Interarrival time and data rate input decides the best packet size output through trained ANFIS. ANFIS is used offline with OPNET modeler for IEEE 802.15.4 GTS mechanism and evaluated the performance. Total (53x53) 2809 pairs are used to minimize the error between trained output and desired output as shown in Figure 7.10.

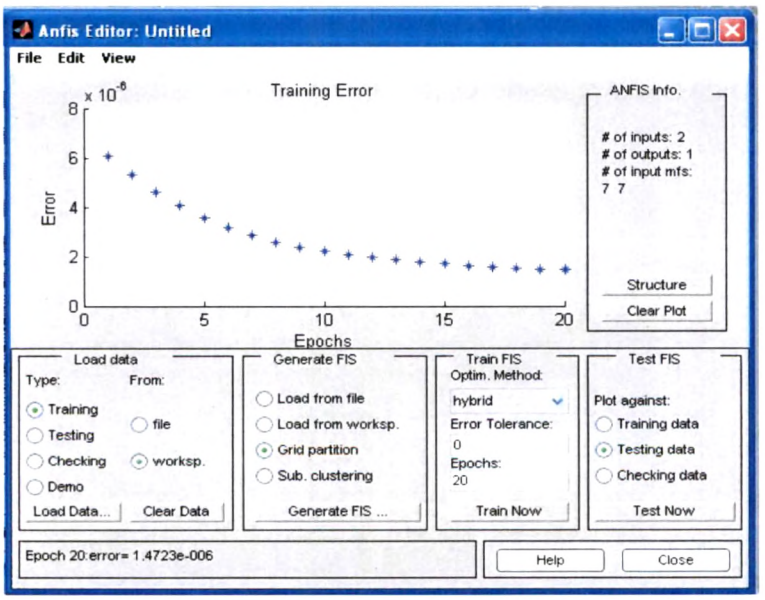

Figure 7.10: ANFIS training

ANFIS architecture was trained with Number of nodes are 131 with training data pairs of 2809 using 49 fuzzy rules.

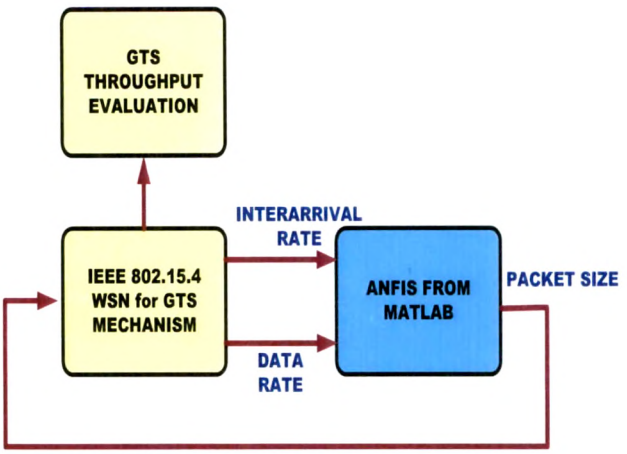

Figure 7.11: System Setup for ANFIS based GTS Mechanism

The system setup for evaluation of performance of GTS mechanism for WSN using ANFIS in OPNET and MATLAB is shown in Figure 7.11.

7.7 Comparison of Traditional method, ANN, ANFIS

Figure 7.12 shows the results analysis among traditional, ANN and ANFIS.

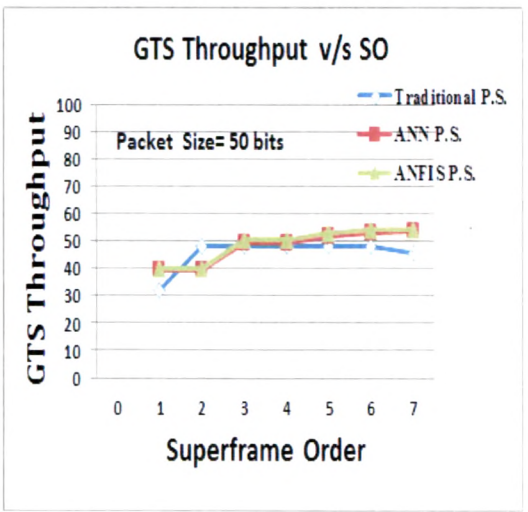

Figure 7.12(a): Result analyses among traditional, ANN and ANFIS GTS Mechanism

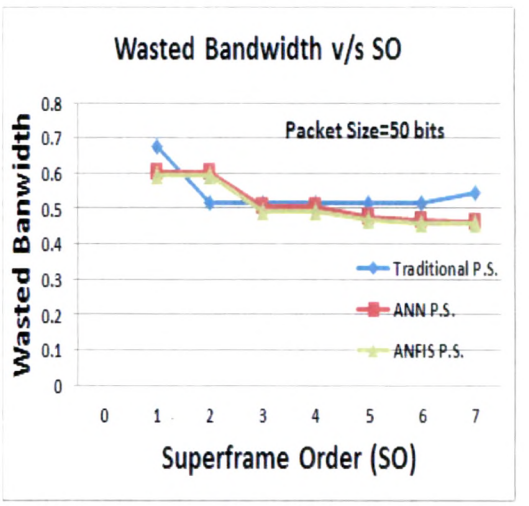

Figure 7.12(b): Result analyses among traditional, ANN and ANFIS GTS Mechanism

From Figure 7.12(a) and (b) it can observe that by applying ANFIS soft computing technique better performance (high throughput and low Wasted Bandwidth) is achieved for packet sizes decided by ANFIS.

7.8 Soft GTS Mechanism Simulator[3]

MATLAB Graphical User Interface [15] development environment provides a set of tools for creating graphical user interfaces (GUIs). These tools simplify the process of laying out and programming GUIs.

Using the GUIDE Layout Editor, you can populate a GUI by clicking and dragging GUI components—such as axes, panels, buttons, text fields, sliders etc.—into the layout area. You can also create menus and context menus for the GUI. From the Layout Editor, you can size the GUI, modify component look and feel, align components, set tab order, view a hierarchical list of the component objects, and set GUI options.

GUIDE automatically generates an M-file that controls how the GUI operates. This M-file provides code to initialize the GUI and contains a framework for the GUI callbacks—the routines that execute when a user interacts with a GUI component. Using the M-file editor, you can add code to the callbacks to perform the functions you want.

Figure 7.13 shows the snap shot of Graphical User Interface SOFT GTS MECHANISM SIMULATOR, designed to do the performance analysis of GTS Mechanism using IEEE 802.15.4 OPNET Simulation model and Adaptive soft computing method having following facilities [16]:

- To change the IEEE 802.15.4 Standard parameters i.e., SO (Superframe duration) and BO (Beacon Order).
- To change the different nodes considered in input, hidden and output layers in soft computing technique.
- To view the different structures, literatures which are taken to carry out this research work.
- To view results of different statistics i.e., GTS throughput, wasted bandwidth by varying different parameters.

3

Published a paper: Ms. Sonal J. Rane, Prof. Satish K. Shah, Ms. Dharmistha D Vishwakarma," Soft GTS Mechanism, Simulator in IEEE 802.15.4 WSN" in International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X ,Volume 3, Issue 9, September 2013.

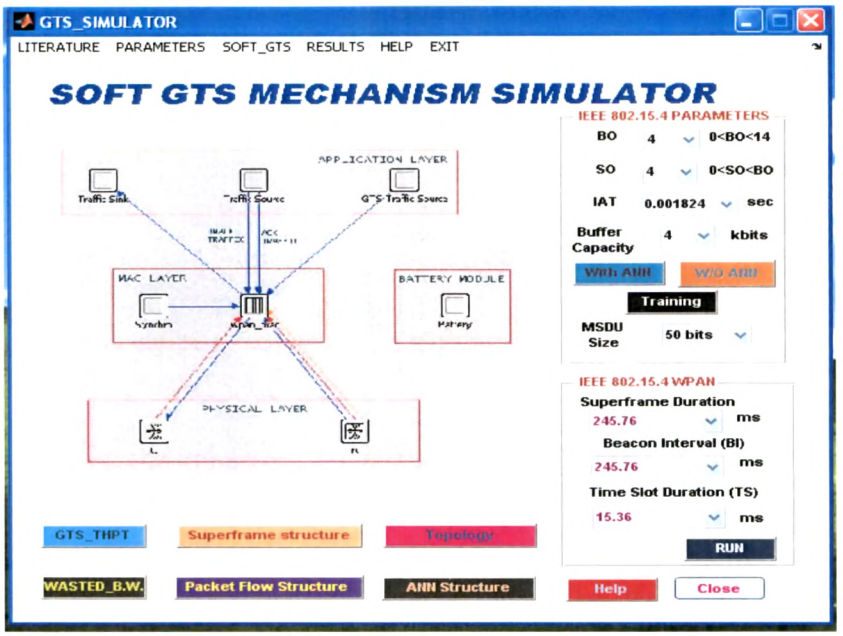

Figure 7.13: Soft GTS Mechanism Simulator

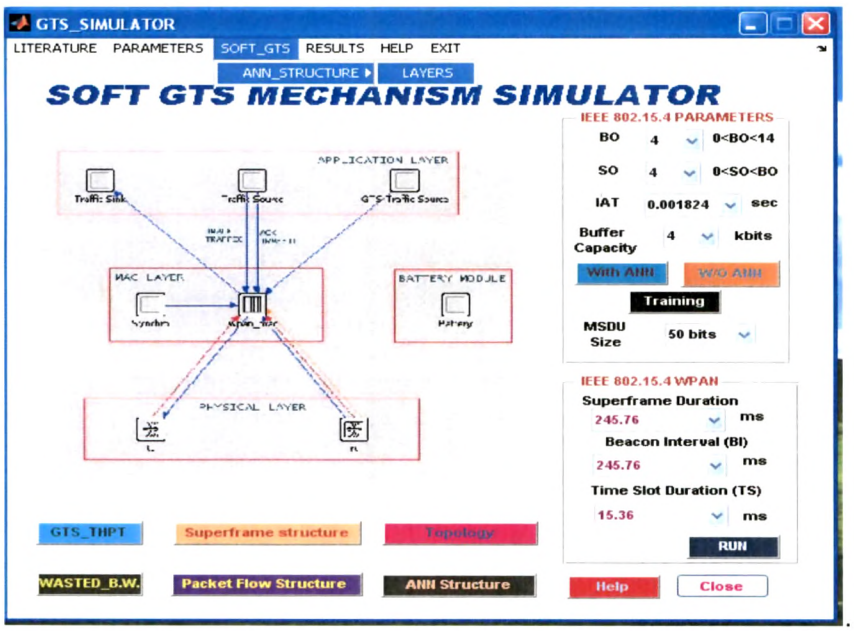

Figure 7.14: Snapshot of menus facility provided in given simulator

It has different menus in menu bar (Refer Figure 7.14), such as Literature, selecting parameters, changing layers in proposed soft computing method, results, help and exit. Literature menu provides different references, pdf files, based on which this research is carried out. SOFT GTS menu provides facility in which input, hidden and output nodes of ANN can be varied to get best output. Result menu gives the graphical result of different parameters that we have chosen. It also provides the online help as one

can require anytime and the exit facility to close the simulator only or to close the whole matlab. In GUI simulator, by clicking the push button **GTS_THPT** and **WASTED_B.W,** graphs are displayed in the display section for different values of SO and BO values. The snapshot is shown in below Figure 7.15. The SO and BO values can be varied for a given condition by clicking the push button SO and BO, designed in right upper panel of IEEE 802.15.4 Parameters.

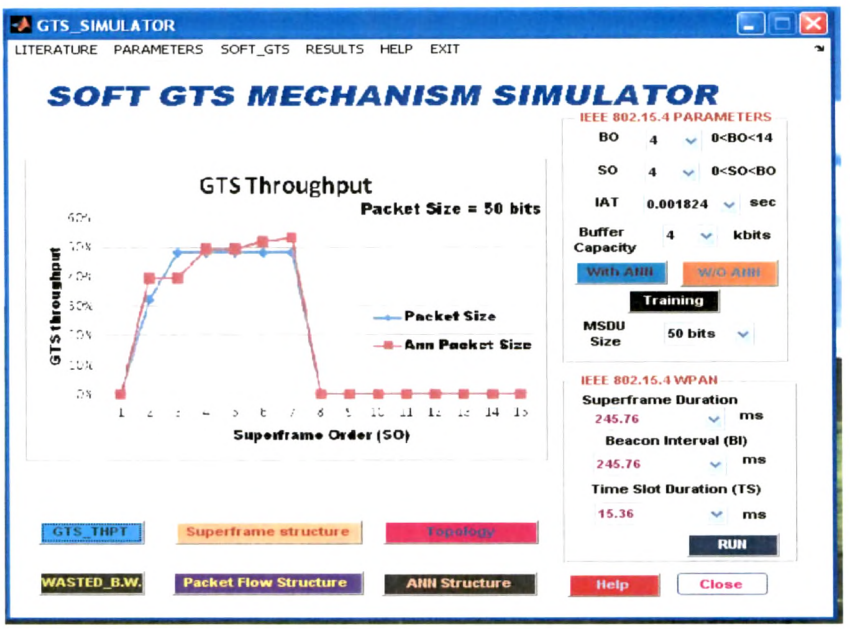

Figure 7.15: Snapshot of GTS Throughput result in designed simulator

This GUI simulator also provides the result with and without applying ANN technique. When ANN push button is selected, the training graph is displayed by clicking the Training button. In below side of GUI, Push button for superframe structure, Packet Flow Structure, ANN structure and considered topology structure are given. In upper panel-IEEE 802.15.4 Parameters (Right Side), values of parameters SO and BO can be selected within the range specified. The corresponding SI and BI values are set, in below panel-IEEE 802.15.4 WPAN shown in Figure 7.15.

Summary:

The theoretical background of soft computing techniques such as ANN and ANFIS is summarized and described. Toolboxes available for deploying soft computing techniques in MATLAB and used in our research work for the design and testing of proposed techniques are described in detail. Procedural steps to be followed in each trait are discussed in detail. Also GTS Mechanism can be optimized by deciding packet size using soft computing Techniques such as ANN and ANFIS.