ABOUT US

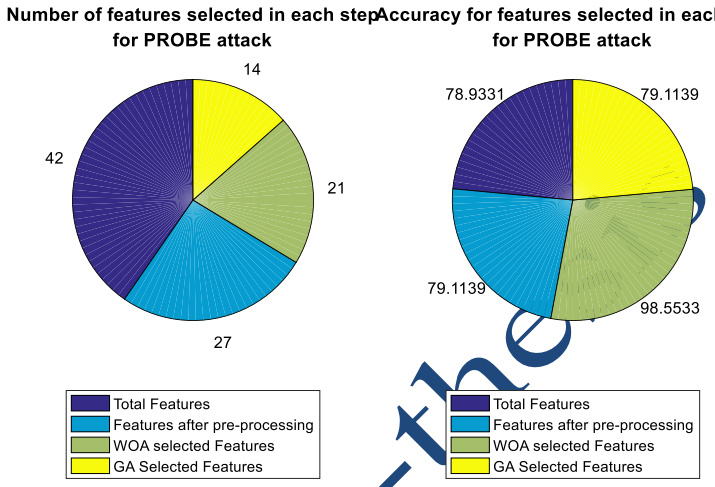
SysMat Soft Solutions is helping research scholars to solve their research issues since 2013 and provide a platform as https://www.free-thesis.com. This website is primarily handled to share the MATLAB codes of our previous researches. Some are freely available and also few are paid with a nominal price. Complete thesis making and writing service is also part of our portfolio. Besides it we are working towards the usage of machine learning algorithms to improve the credibility of thesis done with us. So far we have provided the solution to over 1000 scholars across the globe. The identity and work of client is kept private always till he/she don't permit it to share openly. Also if we share any research work on our free-thesis platform, corresponding client gets monetary benefit from it.

Contents

List of Figures

List of Tables

Thesis-1
An Intrusion Detection System in MANET Using Whale Optimization Algorithm (WOA)



Abstract

Present day scenario the MANET is the very important tool for the control of network with the help nodes present in it. The formation of this type of node is more dynamic in nature the reason behind it various nodes present in the network. There are various attacks are present in the MANET like Dos attack, probe attack, R2L and U2R. These type of attacks harm the MANET. So for the detection of attack and remove from MANET many techniques are used. The optimization plays an important role in this for minimizing the attacks possibilities. The optimization is Whale Optimization Algorithm is used it provide better accuracy for the attack detection in the MANET. This algorithm is applied on the SVM (support vector machine) which is a machine learning approach. The data set is taken by the WOA is NSL-KDD data set. It is a standard data set which is available easily. The experiment is done on this type of data set NSL-KDD and results are analyzed with the other optimization method in the end. The comparison between WOA and GA results will be done in the end of the work. There are three parameters mainly used for the comparison and observation. These parameters are accuracy, sensitivity and specificity.

PROPOSED WORK

The IDS system is the main component of MANET. It provides the security to the system. The main purpose of the intrusion detection system is found out the type of attack in the network. The latest technology Machine learning approach commonly used for predictive models and detected the types of attacks. In this work the predictive model of IDS is prepare by the feature reduction using WOA (whale optimization algorithm) with respect to the machine learning algorithm like SVM (support vector machine). The data set is NSL-KDD used for the intrusion detection system. There are mainly two types of attacks are present in the data set first one is denial of service attack (DOS) and second one is probing attacks the focus only on these two types of attack. The whale optimization algorithm is applied to the NSL-KDD data set and then compare with the other algorithm called Genetic algorithm. The results are obtained by WOA and GA compares the accuracy in both the cases.

We have divided our work in six sub cases which are

1. Take the NSL-KDD data set and consider the training and test data set from the master one data set.
2. After the step one then applied WOA for the feature reduction. The maine purpose is identified the attack with higher accuracy.
3. Apply training and testing data in to the SVM classifier with the reduce feature
4. Test data using SVM trained model and check accuracy of predicted labels
5. Compare results with GA reduced feature to proposed method

1.1 NSL KDD Dataset.

Intrusion dataset is taken from standard NSL KDD dataset from website of university of New Brunswick (UNB) http://www.unb.ca/cic/research/datasets/nsl.html. This dataset is already explained in previous chapters. This dataset is very large having 125973 training data set and 22543 testing data set with total 41 features out of which 3 features are symbolic and output is subcategories of major types of attacks. Subcategories of major four attacks are given in previous chapter also.

1.2 Feature reduction using WOA

The NSL-KDD have the large number of feature in case of training and testing the the data the features are 41. Before using the data into machine learning (ML) model, we need ot preprocess it as data is in raw format and not ready to use as it is. Every ML model understand the langaugae of numerics only. Data has strings in the features, we need to first

convert them to analytics. Many attributes has maximum number of zeros which don't contribute in classification and biase the network training. So we select them programmetically and removed from the features set. Some features have high numeric values and some of them have comparatively small values. This large difference also bias the machine learning model. So we normalised them as:

$$normalised\ attribute = \sum_{i=1}^{n} \frac{f_i - \min(f)}{\max(f) - \min(f)}$$

Where 'n' is the total number of samples in an attribute, f_i is the sample value of i^{th} attribute.

After removing the columns with 50% of samples with zero values, we only left with 16 attributes in the data. These all attributes still not contribute in the accuracy but now we don't know which set of attributes should be chosen. For that purpose we opted the novel optimization algorithm based on Wolf's hunting behavior. This optimization is known as Wolf optimization algorithm(WOA) and discussed in previosu chapter. When a large data provided by using the more number of feature the predictive model take lot of times in making traing model. After that more time is consume in the testing and classification. MANET have some applications in the tough catagory so for the more accuracy prediction and reduced the time we used an optimization technique.

The WOA is an iterative type of algorithm which can maximize and minimize any type of objective function. The principle of Whale optimization algorithm explained in the previous chapter. The algorithmis work on the principle of whale catching the food. It detects the food position with respect to the whale movement. The whale position can maximize and minimize with respect to the food location. On the basis of WOA the attack detection process accuracy will be incresed with respect to the attack. The attack can detect in less time in case of WOA.

Whale optimization algorithm and fetaure selection are two isolated algorithgms but these work in a closed loop scenario. A blog diagram representing their communication is shown in figure 1.1.

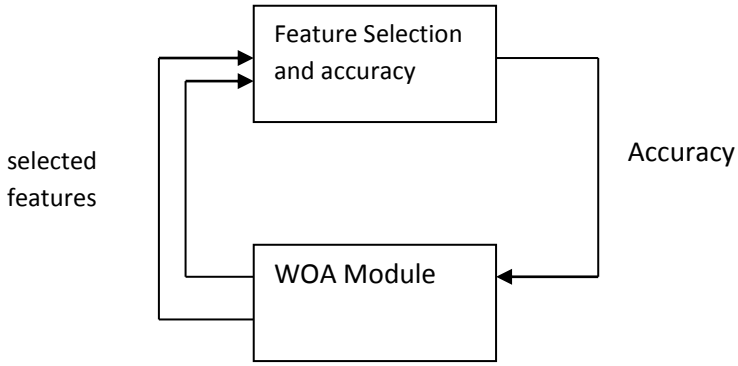


Figure 1.1: Relation between feature selection/ML training and WOA optimisation.

Both module function in equilibrium, WOA gives the input as binatry matrix to Ml module and gets the accuracy in its input from the ML module. This binary matrix is the set of features which must be included. The '1' in the matrix representes the attribute selected and '0' represents that this feature is not selected. Ml module calculates the accuracy for this set of selected features by training and testing the SVM model. This accuracy is fedback to WOA module which on the basis of it changes the feature set. This change in arrangement of 1 and 0 in the binary matrix will be done by WOA equations discussed in previosu chapter from 3.1-3.5. For this new set of selected features, accuracy will be calculated and compared with last step accuracy. Higher will be selected. The process goes on till last iterations and highest accuracy feature set is selected as the final set of selected features. The complete pipeline of optimisation is shown in figure 2. Parameters of WOA is shown in table 1.1

Table 1.1: WOA parameters

Parameter Name	Value
No. of positions (N)	10
Size of problem	41
Max iterations	100
Objective function	Accuracy

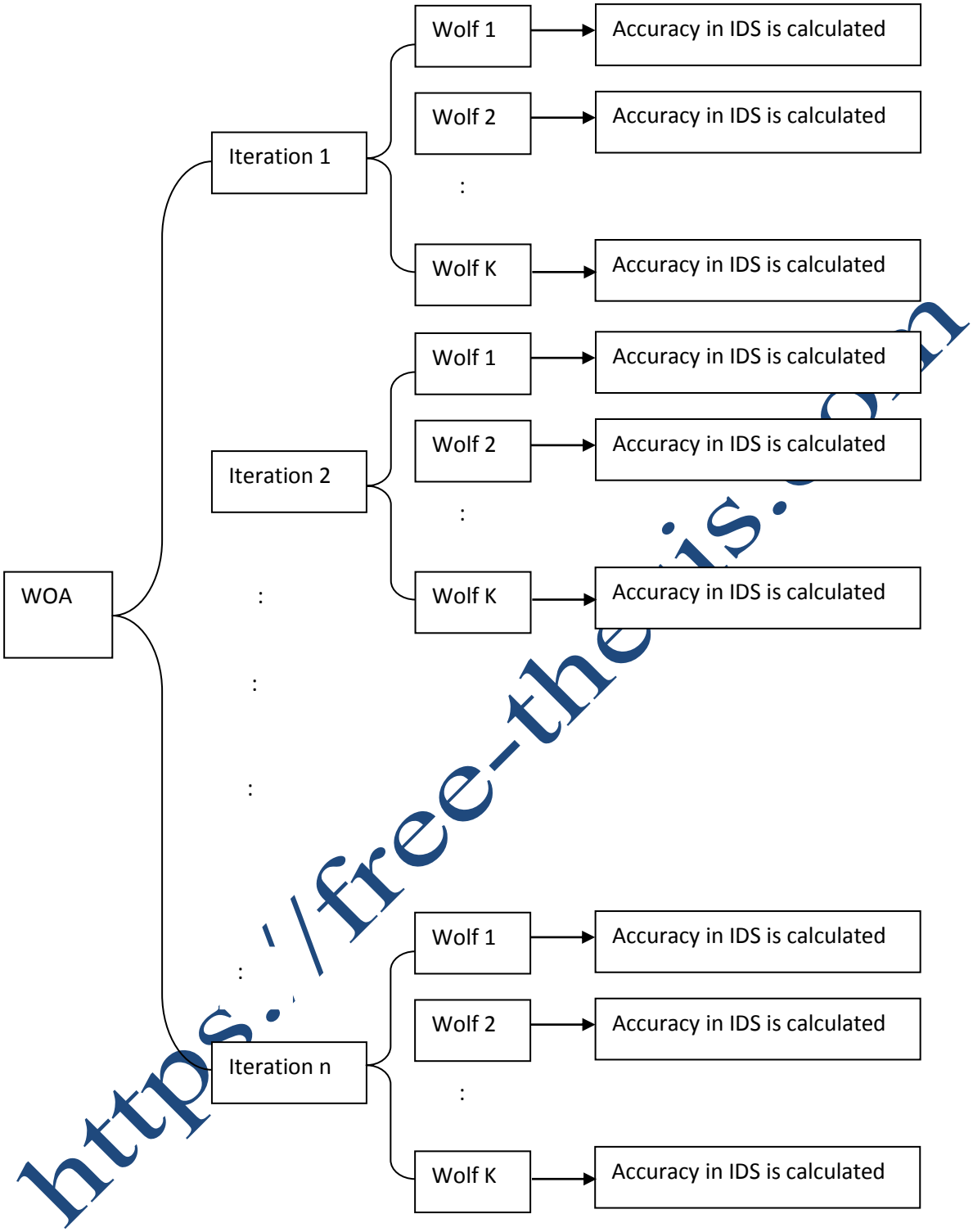


Figure 1.2: Pipeline for whole process for the feature selection using WOA optimization

1.3 Classification using SVM

Support vector machine (SVM) is a machine learning model which works on supervised learning models associated with adaptive learning algorithm that analyzed data used for classification and regression analysis. SVM classifier build a training model using

training dataset. As we have various subcategories of one single type of attack, so we have used multi-class SVM classifier. A predictive model is used for classification and validating training dataset results using testing dataset. Proposed methodological architecture of our method is shown in figure 1.3

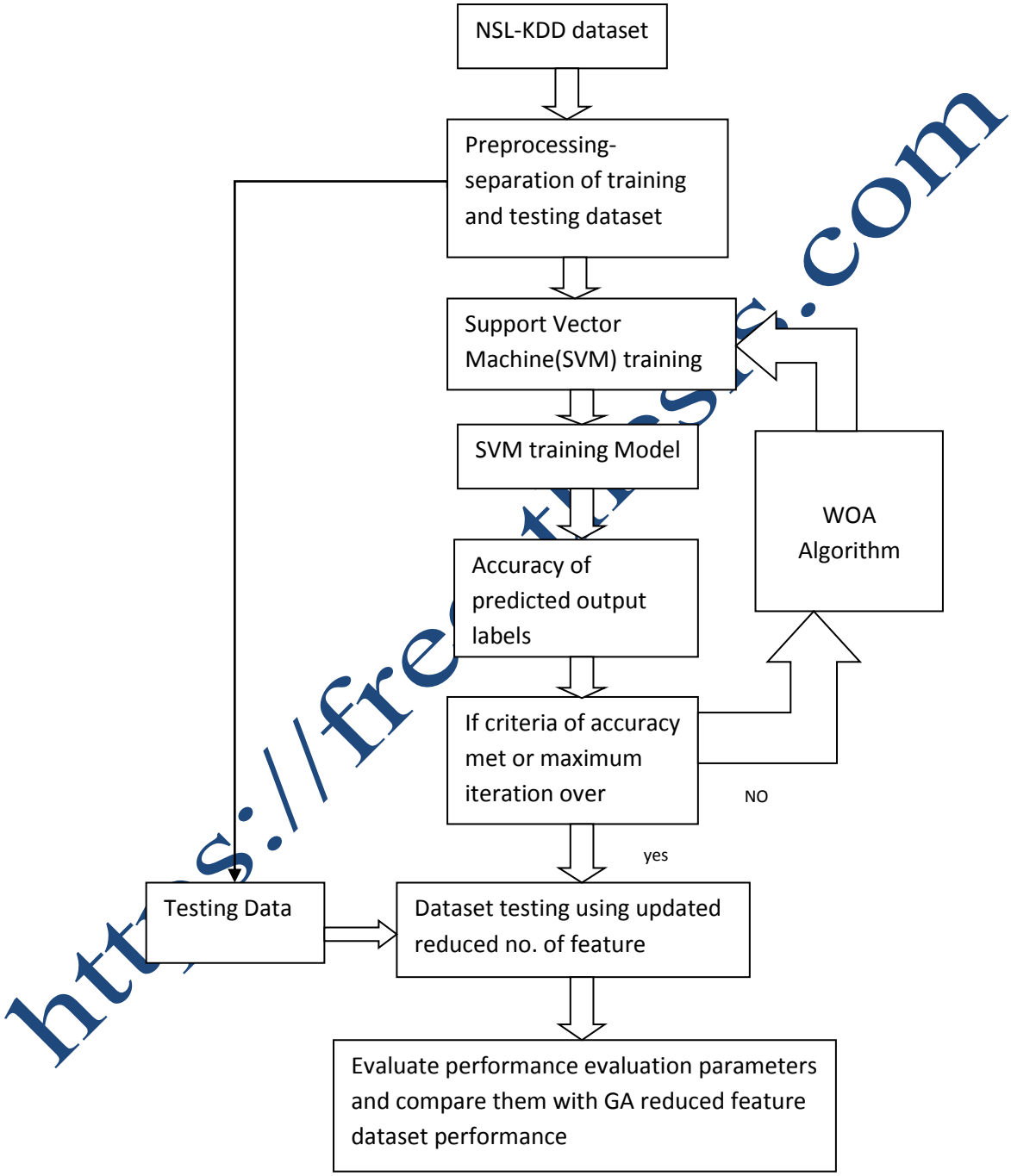


Figure 1.3 Methodology of proposed work

1.4 Performance Evaluation Parameters

Following parameters are used to compare performance of different techniques.

1. Accuracy
2. Sensitivity
3. Specificity
4. Precision
5. Recall
6. F Score

Definitions of performance parameters:-

(tp) True Positive means correctly identified

(fp) False Positive means incorrectly identified

(tn) True Negative means correctly rejected

(fn) False negative means incorrectly rejected

Accuracy-The accuracy is defined as the nearest value of the measurement in case of true value. The formula of accuracy is

$$\text{Accuracy}=\frac{tp+fn}{tp+fn+fp+fn}$$

It is also used as a statistical measure of how well a binary classification test correctly identifies or excludes a condition.

Sensitivity may be referred as the model's ability to correctly predict the correct output label. Sensitivity is also known as True positive rate (TPR)

$$\text{Sensitivity} = \frac{\text{number of true positives}}{\text{number of true positives} + \text{number of false negatives}}$$

$$\text{Sensitivity}=\frac{tp}{tp+fn}$$

Specificity is referring to as model's ability to correctly predict outputs which do not match to real output label. It measures the proportion of negatives that are correctly identified as such.

$$\text{Specificity}=\frac{tn}{tn+fp}$$

specificity is also known as True negative rate (TNR)

$$\text{Specificity} = \frac{\text{number of true negatives}}{\text{number of true positives} + \text{number of false negatives}}$$

Precision is the ratio of retrieved documents to the relevant documents.

$$\text{Precision} = \frac{|\{\text{relevantdocuments}\} \cap \{\text{retrieveddocuments}\}|}{|\{\text{retrieveddocuments}\}|}$$

$$\text{Precision} = \frac{tp}{tp+fp}$$

Recall is the ratio of relevant documents to the successfully retrieved.

$$\text{Recall} = \frac{|\{\text{relevantdocuments}\} \cap \{\text{retrieveddocuments}\}|}{|\{\text{relevantdocuments}\}|}$$

$$\text{Recall} = \frac{tp}{tp+fn}$$

$$\text{F\_Score} = 2\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

F Score is a measure of a test's accuracy. It considers both the precision *p* and the recall *r* of the test to compute the score where *p* is the number of correct positive results divided by the number of all positive results, and *r* is the number of correct positive results divided by the number of positive results that should have been returned. The F_1 score can be interpreted as a weighted average of the precision and recall, where an F_1 score reaches its best value at 1 and worst at 0.

1.5 Overall workflow diagram

Overall work can be collectively reminded as work flow diagram where each step will be shown.Pre-processing step consists of making availability of dataset dividing it in training and testing data set randomly. WOA is applied to reduce features so that accuracy of output label maintained maximum. SVM is used to classify and make a trained model using reduced features set. This trained SVM model is used to test the testing data and evaluate performance with parameters like accuracy, specificity, sensitivity.

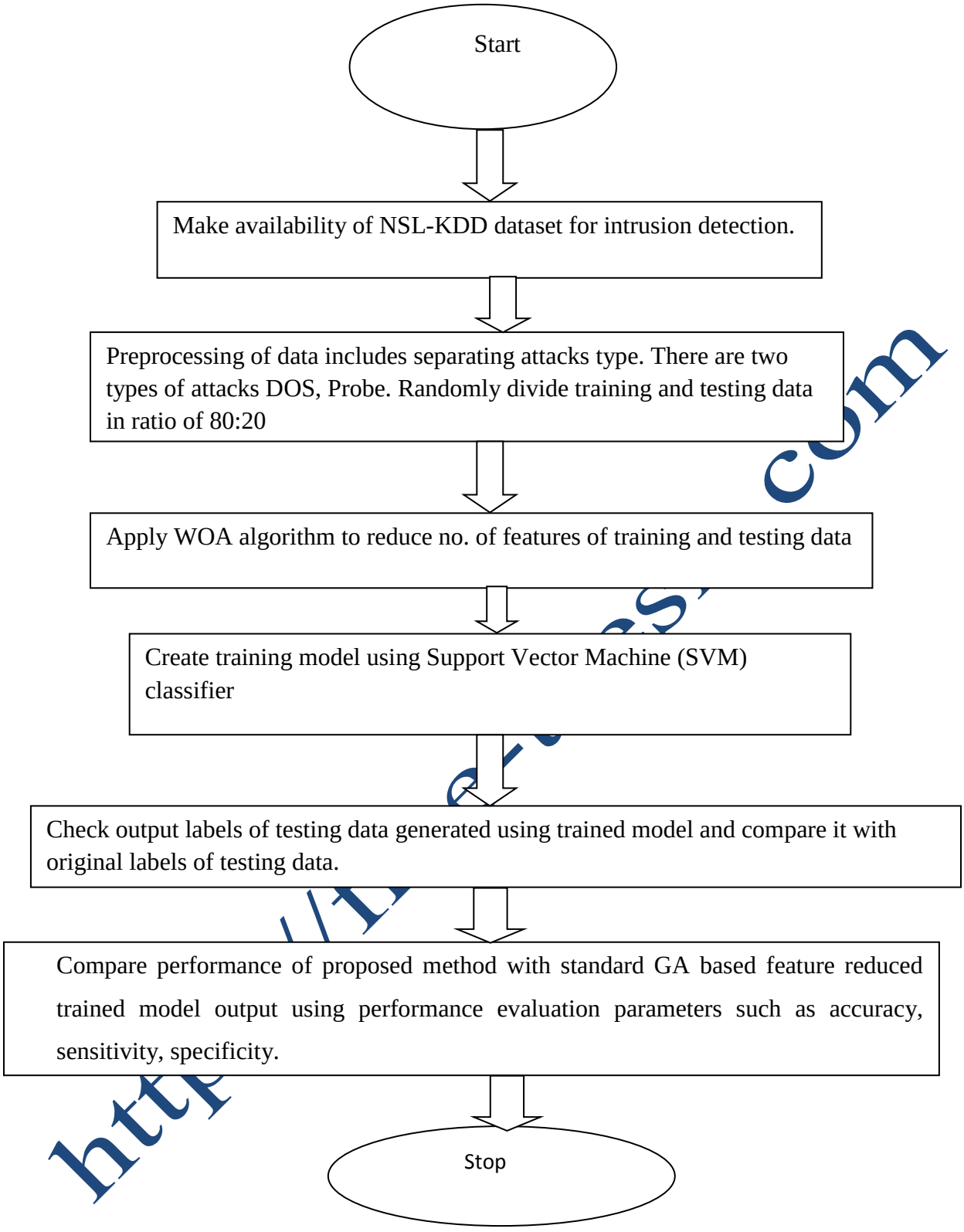
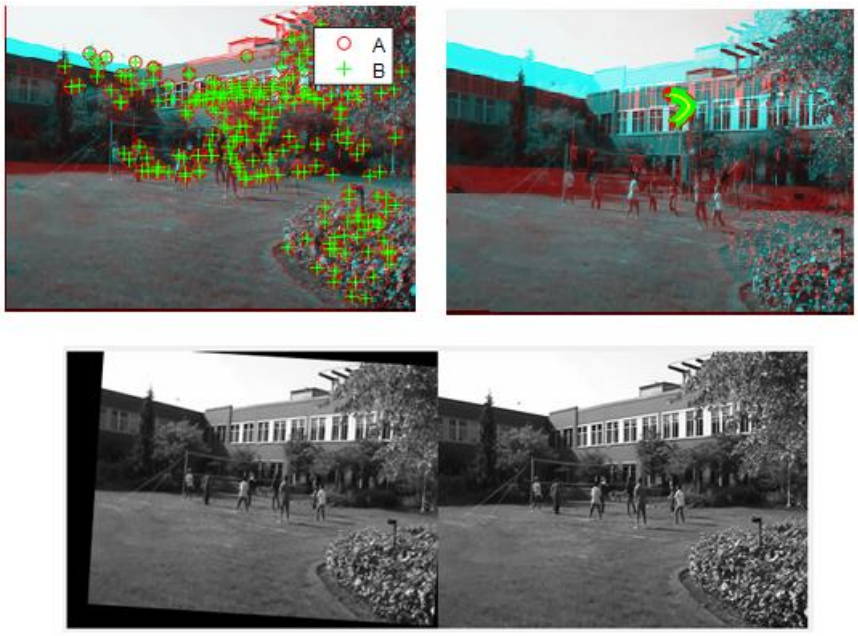


Figure 1.4 overall flow digram

Thesis-2
Digital Video Stablization using Firefly Optimization



Abstract- The problem of digital video stabilization is addressed in this thesis. Videos captured by hand-held devices (eg., cell-phones or DVs) often appear remarkably shaky. Digital video stabilization improves the video quality by removing unwanted camera motions. Our work is based upon the beizer curve smoothing which is following the KLT features tracking method. Beizer curve smoothens and reduces the ouliers in each frame's features. But beizer curve method chooses the control points randomly which doesn't guarantee a converged solution. We updated it by controlling the control points of beizer curve, resulting in reduction of more number of outliers and less feature points which are coinciding in consecutive frames. Initial frame of video is considered as reference and rest frames are rotated as per angle difference from that. We introduced a novel firefly optimization for bizer curve smoothing and evaluated the results of stabilized video on the ground of PSNR, IFT and SSIM. Several test videos are used for testing to check the efficiency of proposed solution, it has been noted that in each case proposed solution is improved upto 25% than beizer curve and 26% from the current shaky frames.

Proposed Work

Digital Video stabilization is a field with promising scope for research. Due to camera movement or shake or jitter, digital video frames shake, which results in false fine details of video. These issues needs the algorithm which can remove the shake form each frame. Since it will be used for each frame so computational complexity will increase with video size. The main concept of video stabilization is to transform the image back to angel at which camera shake while shooting. For this purpose, robust features form each frame are extracted and tracked using KLT feature tracker. We initialized the KLT tracker by SURF features reason being that these are very fast in computation than other feature extraction techniques like SIFT.

The work is based on smoothing of Beizer curve in the feature points. this curve is used to model smooth curve and depend upon some unknown points in computer graphics. Our work is based on smoothing the extracted features of frames in video by optimization function which reduce the number of points on biezer curve by firefly optimization algorithm. These points will be feature points which will be same in every consecutive frame. based on the tracking of these points frames are stabilized by gradient descent based image warping method which is out of context of this thesis. Beizer curve smoothens the trajectory of feature points so that irregularities between them can be avoided. Its a curve on imag esurface and feature points are KLT features. It is an iterative process and works towards the minimisation of mean square difference.

In some applications, it might be reasonable to use evenly spaced t but, for hand trajectories, we know that the hand does not move at constant speed. We need to allow for non-constant speeds and so we define a sequence of relative times,$0 \leq s_1 \leq \cdots s_{n-1} \leq 1$ representing the relative progress of the hand along the trajectory. We define the sum of squares for the fit as

$$SS = \sum_{i=1}^{n-1} Z_i - C(s_i)^2$$

We want to choose the control points of C to minimize SS . Z_i is the set of features observed though KLT tracker. Good minimization is dependent upon the control points selection. So in our work we choose these control points after tuning by firefly optimization algorithm. In it an objective function is defined which calculates the SS above in each iteration for each firefly. Care has to be taken that first and last control points are always fixed and firefly

optimizes only surface points within the range of 0 and 1. To calculate the SS as is clear from the formula, feature points are also required which are inliers KLT feature points. Points which are not relevant or have a large variation from other points are outlier points and are removed using MATLAB's RANSAC algorithm. The objective function designed in MATLAB is shown in table 2.1.

```
Function op= objf(ti,Mat)


%%% Default Values %%%
ibi=[1; size(Mat,1)]; % first & last
defaultValues = {ibi};
[ibi] = deal(defaultValues{:});
% % %----------------------------------------
datatype=class(Mat); %original data type
Mat=double(Mat);   %converte to double (necessary for computation)


% % %----------------------------------------
if( ~isvec(ibi) )
    error('arg3 must be row OR column vector');
end
ibi=getcolvector(ibi);
ibi=[ibi; 1; size(Mat,1)]; % make sure first & last are included
ibi=unique(ibi);           % sort and remove duplicates if any
[p0mat,p1mat,p2mat,p3mat,~]=FindBzCP4AllSeg(Mat,ibi,ti,'u');


[MatI]=BezierInterpCPMatSegVec(p0mat,p1mat,p2mat,p3mat,ibi,ti);


[sqDistAry,indexAryGlobal]=MaxSqDistAndInd4EachSegbw2Mat(Mat,MatI, ibi);
sqDistMat=[sqDistAry',indexAryGlobal'];
% localIndex is index of row in sqDistMat that contains MxSqD
[MxSqD, localIndex]=max(sqDistMat(:,1));
```

```
op=MxSqD;
```

Video stabilisation and firefly are two isolated fields, then how firefly is optimizing the beizer curve points? The firefly agents in the algorithm are attracted towards each other by their light. More the light intensity, more fireflies will move towards it. This light intensity is the value of objective function in our video stabilization case. The lower the objective value for a given set of control points on the curve, higher will b e the intensity and also the probability of convergence. Every firefly is mapped into a searching space with some co-ordinates value. The number of co ordinates for a firefly is equal to the number of points to be tuned. In our case we need to tune the surface points equal to the number of KLT features points which are correlated. So each firefly will have that number of co ordinates values equal to inliers KLT feature points. the whole process can be depicted as:

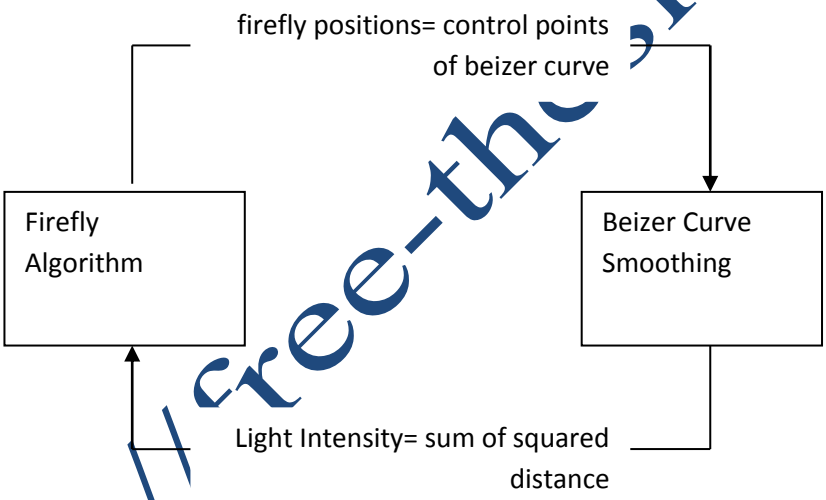


Figure 2.1: Representation of equilibrium of firefly optimisation and beizer curve smoothing

The whole process is explained in steps as:

Step1. divide the input video into frames to avoid jitter and complexity

Step2. extract the SURF features form the first frame which is considered as the reference frame for complete simulation

Step3. Initialize the KLT feature tracker by SURF features of reference frame, which will be tracked in subsequent frames.

Step4. remove the outliers points using RANSAC method from each frame's features

Step5. apply beizer curve smoothing on these inliers points and use geometrical transformation to transform the image by the angle at which these points are aligned to reference frame.

Step6. initialize the firefly algorithms initial point randomly which will be the control points of beizer curve.

Step7. make a beizer surface using these points and calculate the sum of squared difference between visible points (feature points) and control points.

Step8. update the position of each firefly using the formula

$$x_i = x_i + \beta_0 e^{-\gamma r^2}(x_j - x_i) + \alpha\left(rand - \frac{1}{2}\right)$$

Step9. calculate the objective function value again for this new updated position.

Step10. is new_objvalue < prev_objval

if yes {update the new position}

else {update the previous positions}

Step11. continue this process till all iterations are not finished and save the positions of final best firefly for which objective function is minimum.

Step12. this final point will be smoothed beizer curve's control points.

A flow chart for the proposed method is shown in figure below.

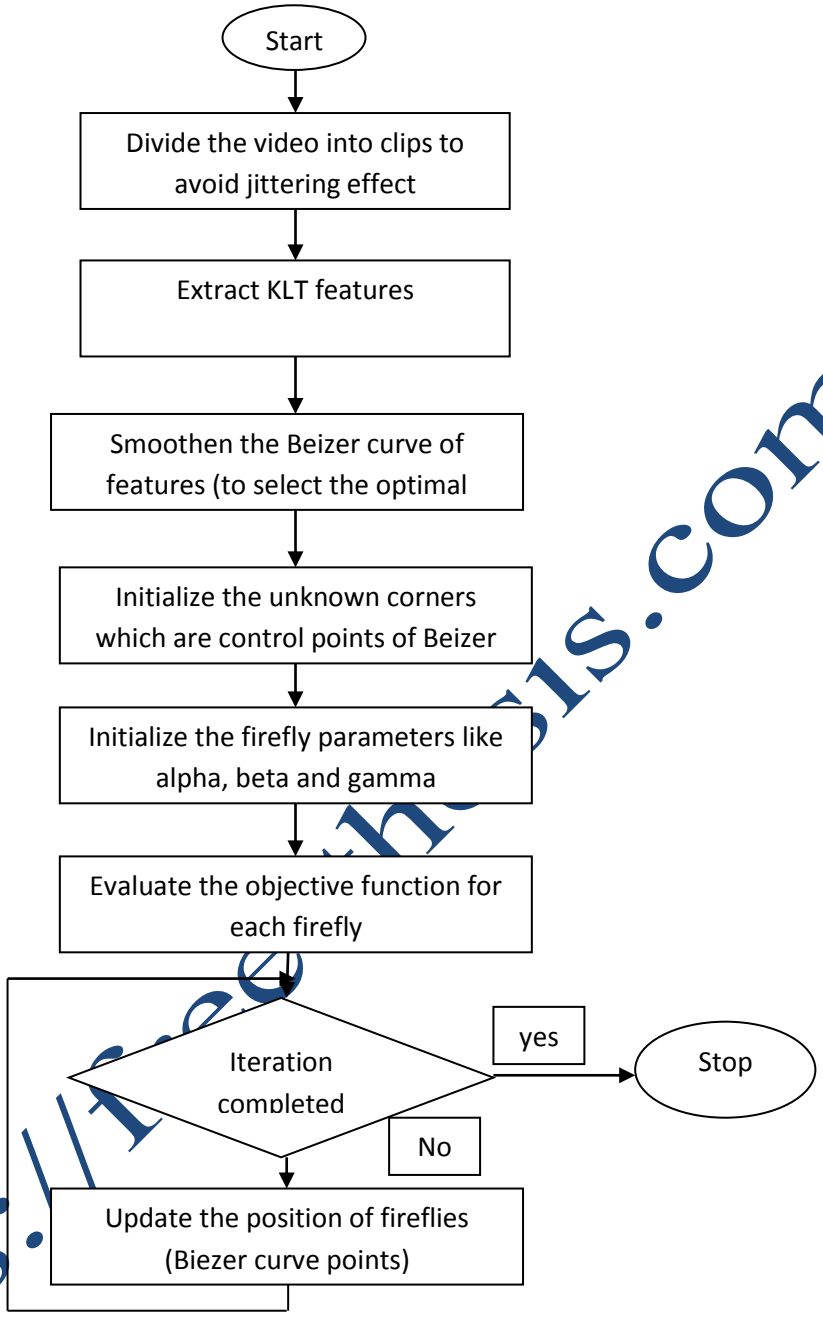
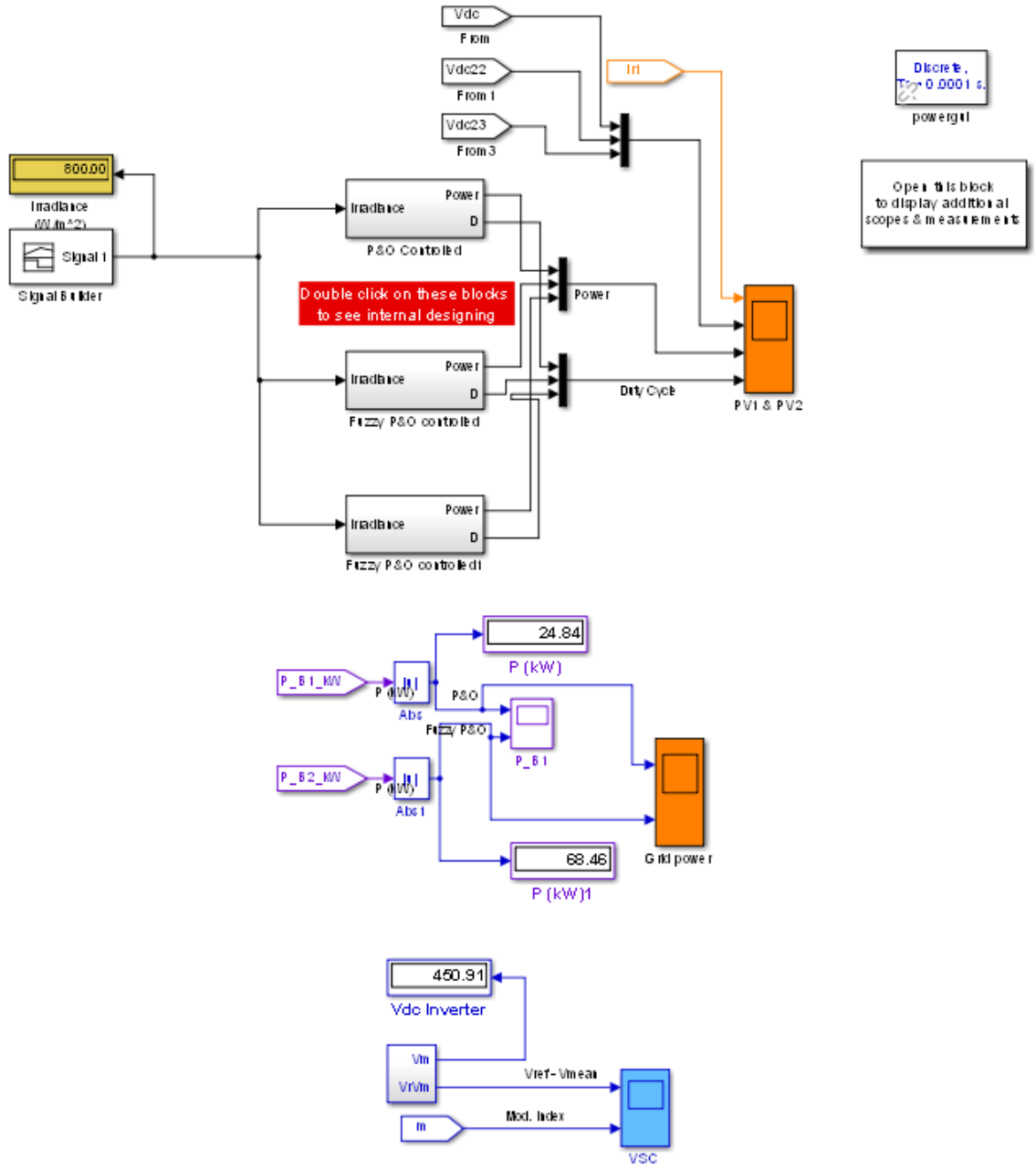


Figure 2.2 Flowchart of complete method

Thesis -3
Fuzzy Logic Controlled MPPT in PV-Grid System

In This work one objective is completed which includes the fuzzy logic to boost the MPPT



Abstract

Due to uncontrolled emission of gases by the burning of resources, global warming is increasing. This is also depleting the natural resources which can't be regenerated again. So every country is moving towards the renewable energy resources like solar cell, wind power etc. But countries can't rely them completely as they have many constraint. So in our work we analysed the PV array connected grid for power distribution. In this system harmonics in the transmission power is added which reduces the quality. So we propose a system with fuzzy logic controlled MPPT controller to boost and stabilise the harmonics along with harmonic filters at grid side. We tested the results with varying irradiation on solar cell and results are quite good. A comparison is done with fuzzy logic controlled system without harmonic filter and P&O controlled system without harmonics. Our proposed combination is able to reduce the 80% harmonics than P& O controlled system.

Proposed Work

Our work is focused on reducing the harmonics in PV connected grid transmission system. We adapted the method of using a set of series and parallel active filters. The harmonics reduction and maximum power transferred to grid from PV array should have trade-off. We are in need to transfer the maximum power with reduced harmonics in the output. So the proposed work is divided into two modules: first one is focusing on transferring the maximum power using fuzzy logic as MPPT controller and other is reducing the harmonics by the use of a set of series and parallel filters. Both modules are discussed separately.

Here we have connected the PV array with grid. To operate the system in balanced condition their voltage and frequency must be synchronised. A VV inverter at the coupling point fulfils this purpose. PV array changes the due to change in irradiations or other environmental conditions which can disturb the synchronisation of PV grid module, so a proper power generation system at PV side must be managed which can cope up with these changes. So we used fuzzy logic controlled MPPT boost converter to manage these changes. A single PV cell can generate only a power upto 3 W at 0.5-0.6 volts. This much less power can't be used to supply to grid. So we used a array of cells with 5*66 cells. 5 rows and 66 columns of PV cells are used to make an array which makes use of 330 PV cells in an array. A block diagram of PV connected grid is shown in figure 3.1.

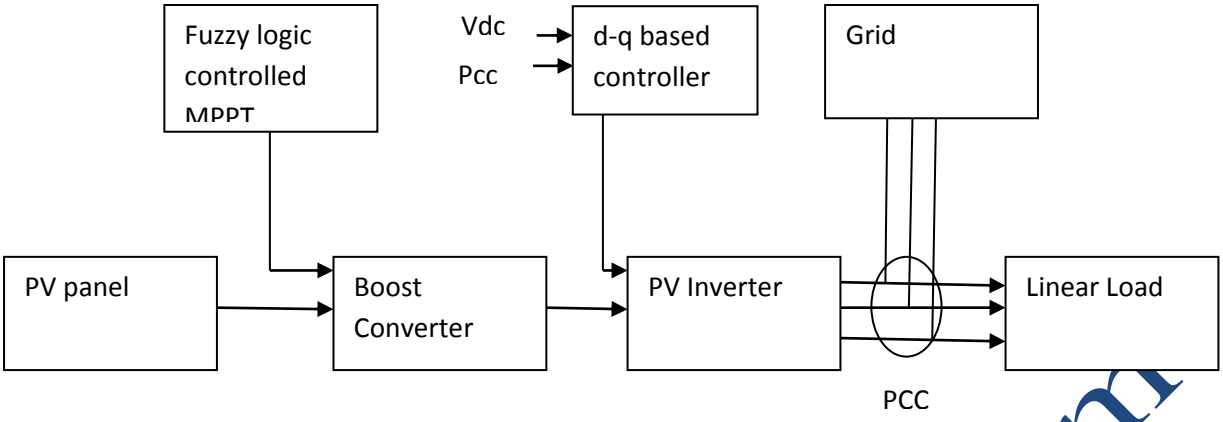


Figure 3.1: Block Diagram of PV-grid connected System

At the gird side controller, a PV inverter is used which converts the DC into AC with complete synchronisation. This converter is controlled by a feedback controller based on d-q theory.

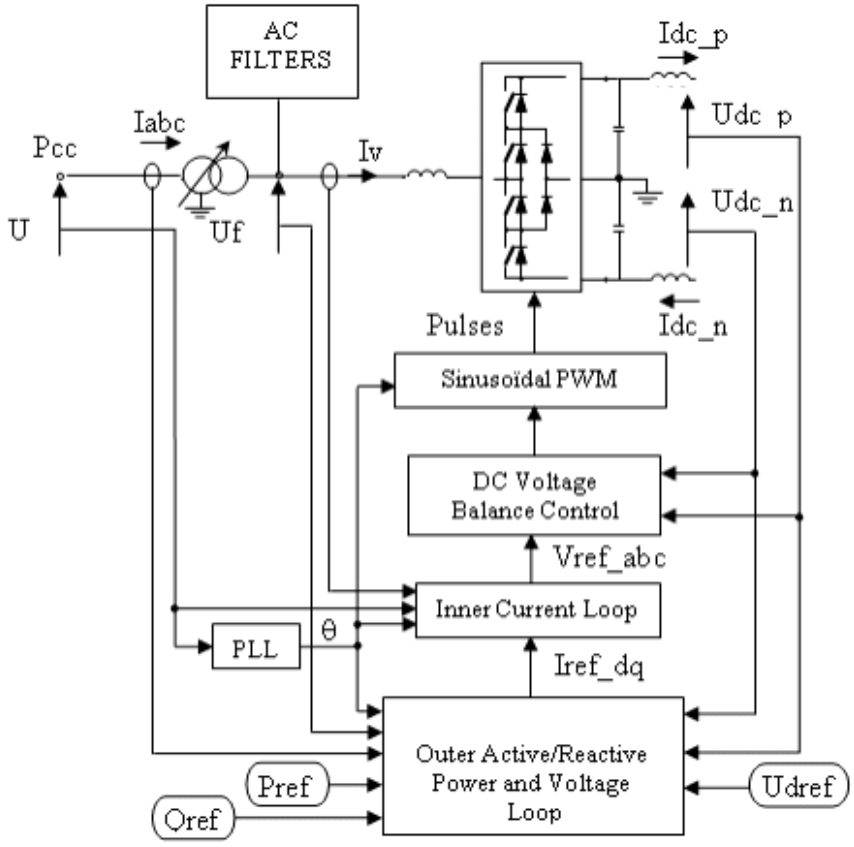


Figure 3.2: Block Diagram of VSC controller for PV inverter at grid side

For three phase signals, six controlling switches are used which are fed by SPWM (sinusoidal pulse width modulation scheme). A block diagram of this voltage source controller is shown

in figure 3.2 which is designed in ORCAD schematic. Fortunately we have a block in sim power toolbox of MATLAB which serves our purpose of controller.

To manage the proper synchronisation between PV array and grid, the DC voltage at point of common coupling (PCC) as shown in figure 4.1 must be managed to constant rating. In our work it is 500 Volt. The more it disturbs, the more system is in unsynchronised state. The dq controller in figure 4.2 converts the current into direct axis current and quardature axis current. The phase locked loop (PLL) is provide the synchronous angle Ɵ by measuring the system frequency for the dq transformation block. A brief mathematical formulation fo controller is discussed here. After conversion of reference current into d-q part, they are estimated in sequence as follows:

The unit vectors in-phase with va, vb and vc, are derived as:

Ua = Va / Vm; Ub = Vb / Vm; uc =Vc /Vm

where Vm is the amplitude of the AC terminal voltage at the PCC and can be computed as:

$$V_m = \frac{2}{3}\sqrt{V_a^2 + V_b^2 + V_c^2}$$

where va, vb, and vc are the instantaneous voltages at PCC and can be calculated as:

$$V_a = V_{san} - R_s I_{sa} - L_s p I_{sa}$$

$$V_b = V_{sbn} - R_s I_{sb} - L_s p I_{sb}$$

$$V_c = V_{scn} - R_s I_{sc} - L_s p I_{sc}$$

where p is time differential operator (d/dt) and Ls and Rs are per phase source inductance and resistance respectively. vsan, vsbn, and vscn are the three phase instantaneous input supply voltages at PCC and are expressed as:

$$V_{san} = V_{sm}\sin(\omega t)$$

$$V_{sbn} = V_{sm}\sin(\omega t - 2\pi/3)$$

$$V_{cbn} = V_{sm}\sin(\omega t + 2\pi/3)$$

where V_{sm} is the peak value and $\omega = 2\pi f$ is the angular frequency of the supply.

The unit vectors in-quadrature with Va, Vb and Vc may be derived by taking a quadrature transformation of the in-phase unit vectors ua, ub and uc as:

$$w_a = -\frac{u_b}{\sqrt{3}} + \frac{u_c}{\sqrt{3}}$$

$$w_b = \sqrt{3}\frac{u_a}{2} + \frac{(u_b - u_c) \times 2}{\sqrt{3}}$$

$$w_b = -\sqrt{3}\frac{u_a}{2} + \frac{(u_b - u_c) \times 2}{\sqrt{3}}$$

The quadrature component of the reference source currents is computed as:

The voltage error Ver at PCC at the nth sampling instant is as:

$$V_{er}(n) = V_{ref}(n) - V_m(n)$$

The output of the PI controller at the nth sampling instant is expressed as:

$$I^*_{smq(n)} = I^*_{smq(n-1)} + k_p\big(V_{er}(n) - V_{er}(n-1)\big) + k_i V_{er}(n)$$

Where Kp and K, are the proportional and integral constants, respectively of the proportional integral (PI) controller and the superscript (*) represents the reference quantity.

The quadrate components of the reference source currents are estimated as:

$$I^*_{saq} = I^*_{smq} w_a$$

$$I^*_{sbq} = I^*_{smq} w_b$$

$$I^*_{scq} = I^*_{smq} w_c$$

The in-phase components of the reference source currents are computed as:

$$I^*_{sad} = I^*_{smd} u_a$$

$$I^*_{sbd} = I^*_{smd} u_b$$

$$I^*_{scd} = I^*_{smd} u_c$$

where, I*smd is considered fixed value corresponding to the constant source current for load leveling. The total reference source currents are the sum of the inphase components of the reference source currents and the quadrature components of the reference source currents are given as:

$$I_{sa}^{*} = I_{sad}^{*} + I_{saq}^{*}$$

$$I_{sb}^{*} = I_{sbd}^{*} + I_{sbq}^{*}$$

$$I_{sc}^{*} = I_{scd}^{*} + I_{scq}^{*}$$

The PWM output is thus fed into the gating terminal of inverter which control the power to grid.

3.1 Module I: fuzzy controlled MPPT

Maximum power point tracking is the concept of transferring the maximum power to grid. It depends upon the duty cycle fed into boost converted connected to the PV side. Boost converter step up the voltage generated form PV array and makes use of IGBT switches for that. A MATLAB developed schematic diagram of boost converter is shown in figure 3.3. A PWM block is used to generate the pulses which are feed into the gate control of switch of boost converter.

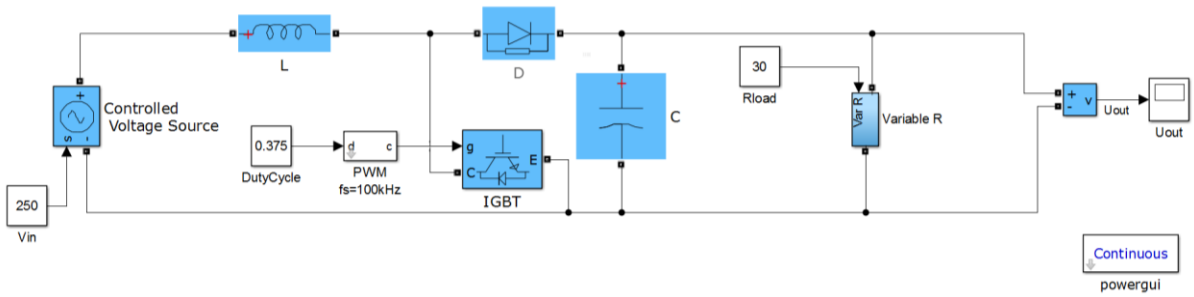


Figure 3.3: Boost Converter in MATLAB

To control the MPPT point, this duty cycle of PWM is to be controlled which will change the pulse on/off time. Maximum the on time in a pulse, maximum the power transferred to grid. A PWM pulse wave with on off time is shown in figure 3.4. Here D is the duty cycle and T is the time period of full cycle. Initially this controlling work was done by P&O controller which is acronym of perturbation & observation controller.

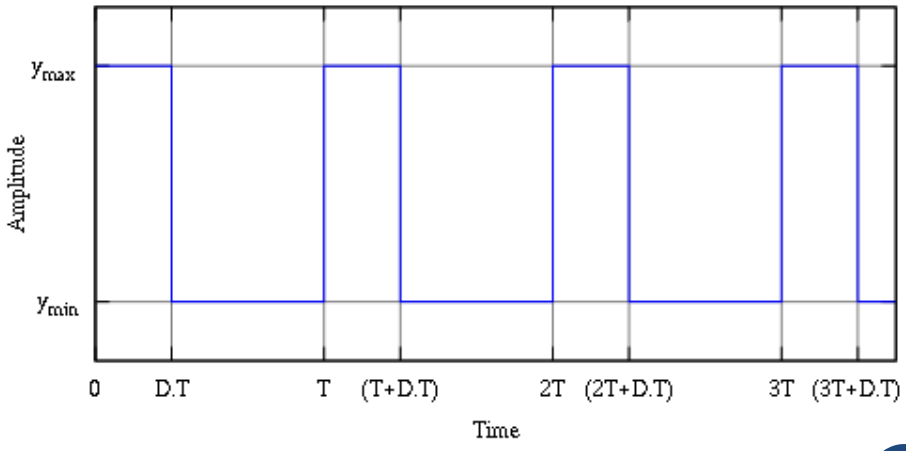


Figure 3.4: Pulse wave, showing the definitions of y_{min}, y_{max} and D.

This algorithm uses simple feedback arrangement and little measured parameters. In this approach, the module voltage is periodically given a perturbation and the corresponding output power is compared with that at the previous perturbing cycle. In this algorithm a slight perturbation is introduce to the system. This perturbation causes the power of the solar module various. If the power increases due to the perturbation then the perturbation is continued in the same direction. After the peak power is reached the power at the MPP is zero and next instant decreases and hence after that the perturbation reverses as. When the stable condition is arrived the algorithm oscillates around the peak power point. In order to maintain the power variation small the perturbation size is remain very small. The technique is advanced in such a style that it sets a reference voltage of the module corresponding to the peak voltage of the module. The flow chart of this algorithm is shown into figure 3.5.

To improve the result from P&O controller, we used artificial intelligence technique. The fuzzy logic controller is now used in our proposed work to enhance the decision levels. Fuzzy logic is explained in previous chapter. here we will discuss how its rules and inputs are defined for our application area.

The tracking of MPP is divided into two steps: first will increase the response of MPP and other will increase the stability after MPP. The fuzzy controller consists of three sub-blocks: fuzzyfication in which real environment variable is converted to fuzzy variables, inference model which inherits the rule set or decision variables and last one is defuzzyfication which reverse the fuzzy variables to environment variables. The fuzzy logic controller for the MPPT has two real time inputs measured at every sampling time, named ‘E’ and ‘CE’ and one

output named ‘Duty’ for each of the phases. The ‘E’ stands for error and ‘CE’ is the change in error. The error at sample time k is calculated by

$$E(k) = \frac{\partial P}{\partial V} = \frac{[P(k)-P(k-1)]}{[V(k)-V(k-1)]} \qquad (3.1)$$

And the $$CE = E(k) - E(k-1) \qquad (3.2)$$

Where P(k)= output power of PV panel at time instant k

P(k-1)= output power of PV panel at time instant (k-1)

V(k)= output voltage of PV panel at time instant k

V(k-1)= output voltage of PV panel at time instant (k-1)

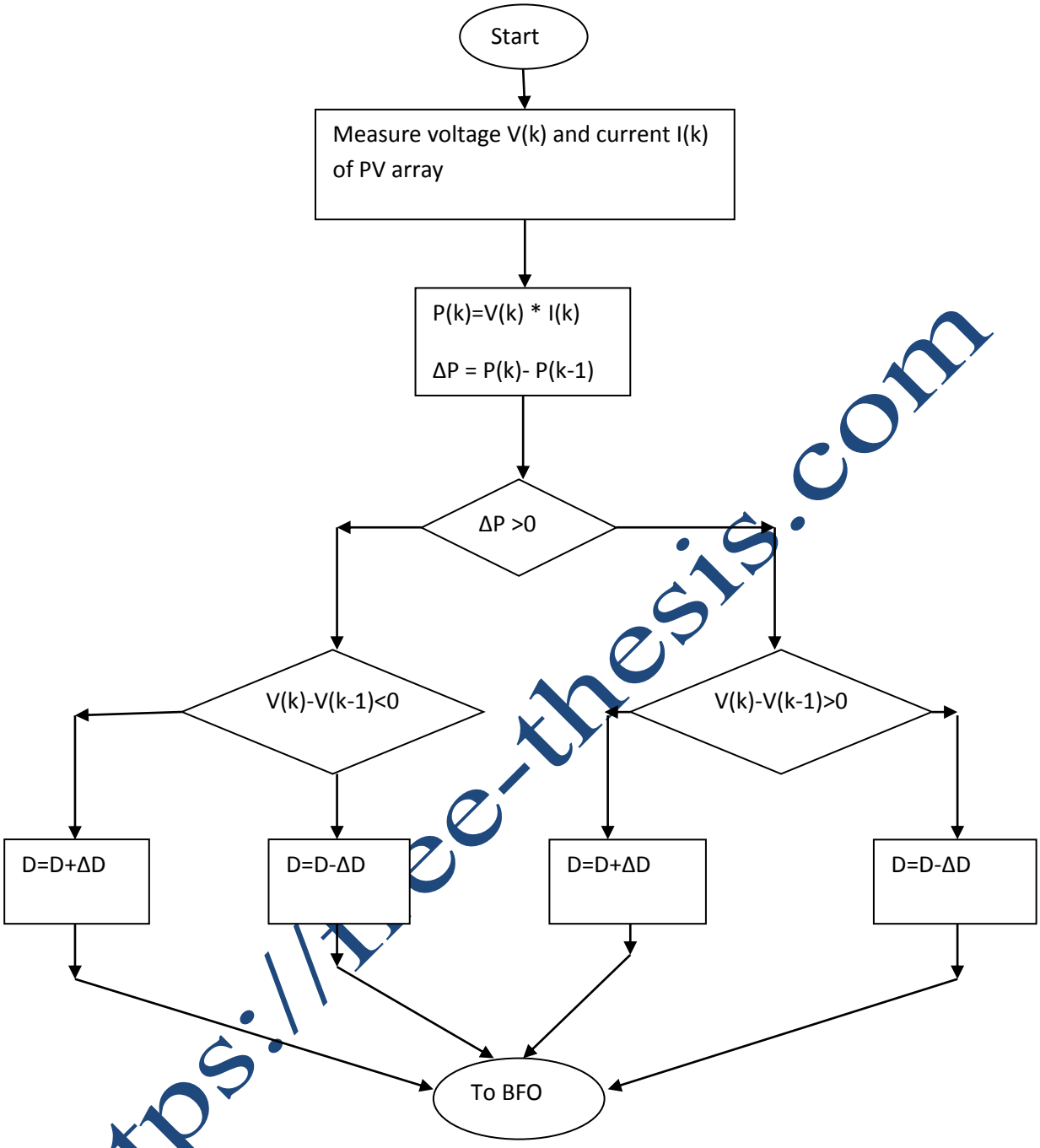


Figure 3.5: P&O algorithm flow chart

The input signals are fuzzified and represented in fuzzy set notations by membership functions. The defined rules produce the linguistic variables and these variables are defuzzified into control signals for comparison. Fuzzy logic control involves three steps: fuzzification, decision-making and defuzzification. Fuzzification transforms the non-fuzzy (numeric) input variable measurements into the fuzzy set (linguistic) variable that is a clearly defined boundary. In the proposed controller, the ‘E’ and ‘CE’ are defined by linguistic

variables such as NB, NS, ZE, PS, PB characterized by memberships. The memberships are curves that define how each point in the input space is mapped to a membership value between -0.032 to 0.032 and -100 to 100 for ‘E’ and ‘CE’ respectively. The membership functions belonging to the other phases are identical Membership functions for the inputs are shown in Fig.3.6 and Fig.3.7. The membership function of output variable is shown in Fig.3.8.

The surface viewer of our fuzzy logic is shown in figure 3.9. It is a three dimensional representation of mapping of error and output of fuzzy logic. These rules can be visualised in MATLAB's toolbox by using command $fuzzy('name.fis')$. here name.fis is the fuzzy logic name designed in MATLAB which is stored with extension '.fis'.

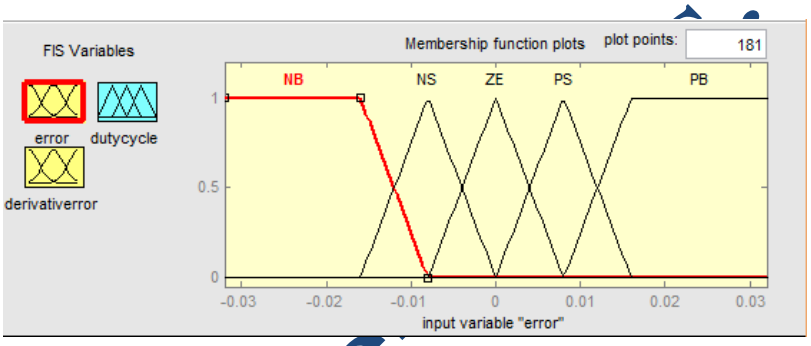


Figure 3.6: Membership function of input ‘SNR’

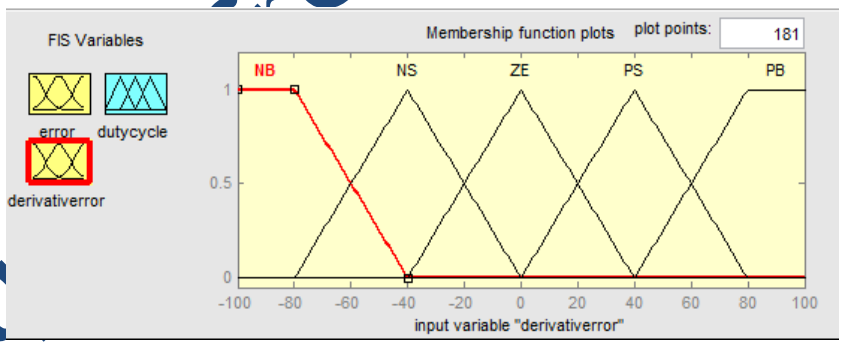


Figure 3.7: Membership function of input ‘mod’

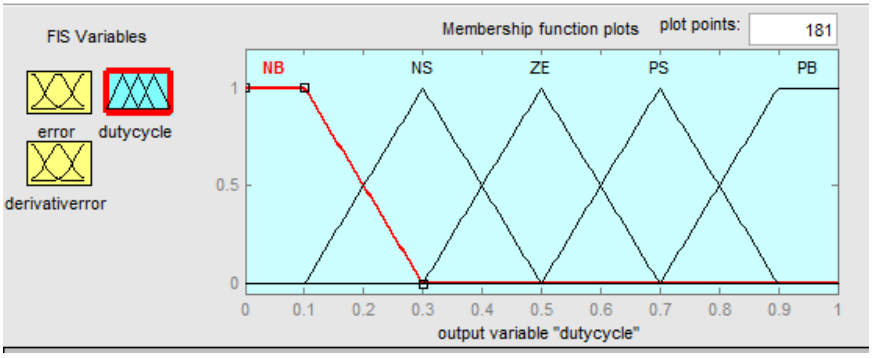


Figure 3.8: Membership function of output modulation technique

Z axis represents the output modulation technique. Defining only membership function doesn't complete fuzzy logic designing. Rule sets for taking decision have to be designed also. A set of 25 rules in our case is designed and table 3.1 represents that along with the representation of rules in rule viewer in figure 3.10.

Table 3.1: Fuzzy logic rules sets

E/CE	NB	NS	ZE	PS	PB
NB	ZE	ZE	PB	PB	PB
NS	ZE	ZE	PS	PS	PS
ZE	PS	ZE	ZE	ZE	NS
PS	NS	NS	NS	ZE	ZE
PB	NB	NB	NB	ZE	ZE

The Rule Viewer displays a roadmap of the whole fuzzy inference process. It is based on the fuzzy inference diagram. You see a single figure window with 25 plots nested in it. The three column plots represent rules of SNR, mod and output. Each rule is a row of plots, and each column is a variable.

The rule numbers are displayed on the left of each row. You can click on a rule number to view the rule in the status line.

- The first two columns of plots (the six yellow plots) show the membership functions referenced by the antecedent, or the if-part of each rule.

- The third column of plots (the three blue plots) shows the membership functions referenced by the consequent, or the then-part of each rule.

This decision will depend on the input values for the system. The defuzzified output is displayed as a bold vertical line on this plot. The Rule Viewer allows you to interpret the entire fuzzy inference process at once. The Rule Viewer also shows how the shape of certain membership functions influences the overall result. Based on these rules output duty cycle range is decided.

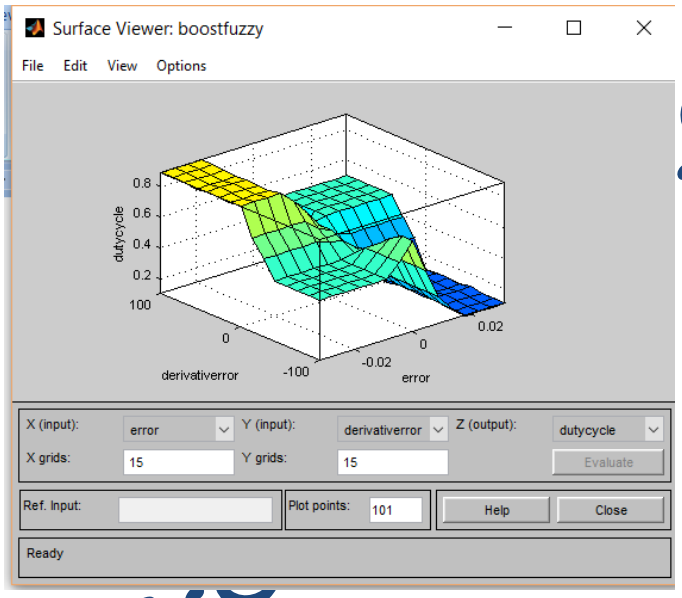


Figure 3.9: Surface viewer plot of fuzzy logic

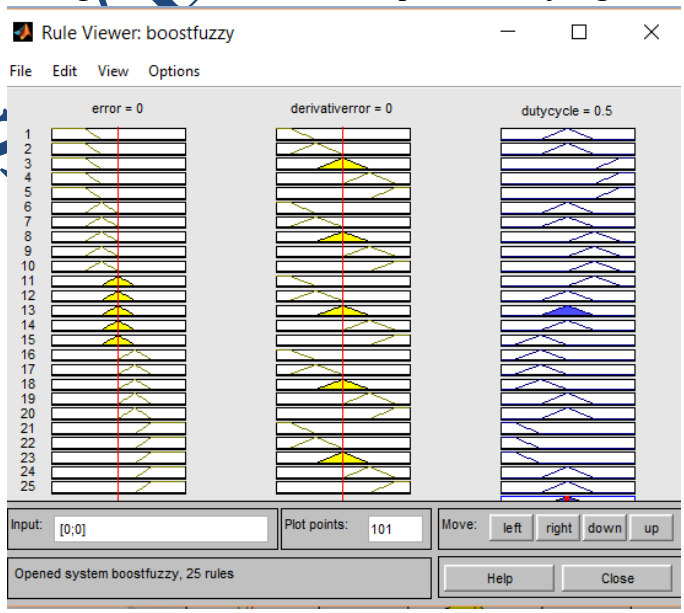


Figure 3.10: Rule viewer of membership functions

3.2 Module II: Harmonics Distortion reduction

We have used harmonic filters along with fuzzy logic controller to stabilise the power output. The filters are composed of R.L& C. A series RL filter along with shunt filter is used. Three shunt filters are connected in parallel. The shunt filter is made up of

- one capacitor banks (C1) of 150 Mvar modeled by a "Three-Phase Series RLC Load",

- three filters modeled using the "Three-Phase Harmonic Filter"

(1) one C-type high-pass filter tuned to the 3rd (F1) of 150 Mvar

(2) one double-tuned filter 11/13 th (F2) of 150 Mvar

(3) one high-pass filter tuned to the 24th (F3) of 150 Mvar

A total 600 MVar rating of filters is used and these are connected to AC line of system.

3.3 Modelling of PV array

3.3.1 Mathematical Formulation

PV cell is a semiconductor p-n intersection that transforms sunlight to electrical power. To model a solar cell, it is imperative that we assess the effect of different factors on the solar panels and to consider the characteristics given by the manufacturers in the datasheet. It is to be noted that to form a PV module, a set of cells are connected in series or in parallel. To form a PV array, a set of PV modules are connected in series and in parallel. Thus, the mathematical models for PV array are attained while utilizing the basic description equivalent circuit of the PV cells.

A PV cell is usually embodied by an electrical equivalent of one-diode, resistance series Rs and resistance parallel Rp as shown in Figure 3.11.

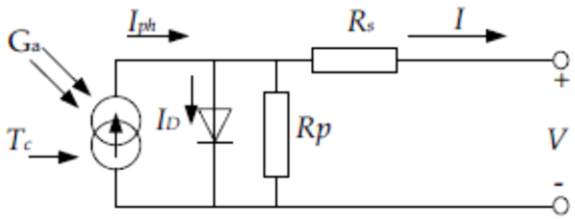


Figure 3.11: Equivalent circuit of solar cell with one diode

From the figure 3.11, the different parameters characteristics of the PV cells are:

Iph: currents generated by the solar cells (A)

Rs: resistance series (Ω)

Rp: resistance parallel (Ω)

Ga: irradiance from the sunlight (W/m2)

T: cell temperature (K)

Id: diode current (A)

I: output current of the PV (A)

V: output voltage of the PV (V)

Manufacturer of the solar module gives the another parameters needed to model the solar cells. The datasheet which gives the electrical characteristics is calculated under standard test condition STC when the temperature T is 25℃ and the irradiance G is 1000 W/m2. The parameters that can be found inside the datasheet are:

Voc: open circuit voltage (V)

Isc: short-circuit current (A)

Pmp: power at maximum power point,

Vmp: voltage at maximum power point

Imp: current at maximum power point

The solar cell is model first, then extends the model to a PV module, and finally models the PV array. From figure 3.11, the output current of the PV cell is

$$I = I_{ph} - I_d$$

where

Iph: photon produced by the cell,

Id: diode current

By Shockley equation, the diode current Id is given by

$$I_d = I_o(e^{\frac{qV_d}{kT}} - 1)$$

where

I0: reverse saturation current of diode,

q: elementary electron charge (1.602x10-19 C),

Vd: diode voltage,

k: Boltzmann constant 1.381x10-23 (J/K)

T: temperature in kelvin (K)

The relation between voltage and current result by replacing the diode current

$$I = I_{ph} - I_o(e^{\frac{qV_d}{kT}} - 1)$$

where Vd is the output voltage of the PV cell.

The reverse saturation I0 is found by using the above equation. By setting the current I equal to zero and calculating at temperature T1

$$I_o(T_1) = \frac{I_{ph}T_1}{e^{\frac{qV_{oc}}{kT}} - 1}$$

The current generated by the solar cells Iph can be approximated with the short circuit current Isc. The current generated can be calculated for other irradiance. The standard current, temperature and irradiance from the datasheet are used to determine the current at different condition.

$$I_{sc} \approx I_{ph}$$

$$I_{sc}(T1) = \left(\frac{G}{G_{nom}}\right) I_{sc}(T_{1,nom})$$

where

Isc(T1): current at temperature T1

T1,nom the temperature of cell from datasheet at STC

Gnom: irradiance from datasheet at STC

After calculation, the equation of the PV are

$$I = I_{ph} - I_o\left(e^{q\left(\frac{V+IR_s}{akT}\right)}\right) - \frac{v + IR_s}{R_p}$$

V is the cell voltage. For a PV module, the cell voltage is multiplied by the total amount of the cells found within the series. The model is completed by using the following recursive equations to find the currents. The recursive equation is used to calculate the current for a PV cell. It is more convenient to solve numerically. The equation introduces a simplified method to calculate resistance series and neglect the resistance parallel.

$$I_{n+1} = I_n - \frac{I_{ph} - I_n - I_o[e^{q\left(\frac{V+I_nR_s}{akT}\right)} - 1]}{-1 - I_o(\frac{qR_s}{akT})e^{q\left(\frac{V+I_nR_s}{akT}\right)}}$$

3.3.2 MATLAB modelling of PV array

The equations used in PV array designing are discussed. As is clear the designing of array is dependent upon the irradiation and temperature. For our work we have considered the room temperature as fixed and irradiation intensity is changed. The main four PV model parameters are: photo-generated current (I_{ph}), diode saturation current (I_{sat}), parallel resistance and series resistance are adjusted to fit the four parameters of model like short circuit current, open circuit voltage and voltage and current at maximum power point. The parameters value of SunPower SPR-305WT module is used which is extracted from the database of website solardesinedtool.com [23]. In appendix 1 parameters values of this module are provided. PV array consists of 66 strings of module connected in parallel and 5 string of modules connected in series. The designing of PV array is shown in figure 3.12. The reason for select of this module is out of our work scope as our focus is on improving the MPPT in weblink given in reference [24] comparison between various solar panels can be checked by curious readers.

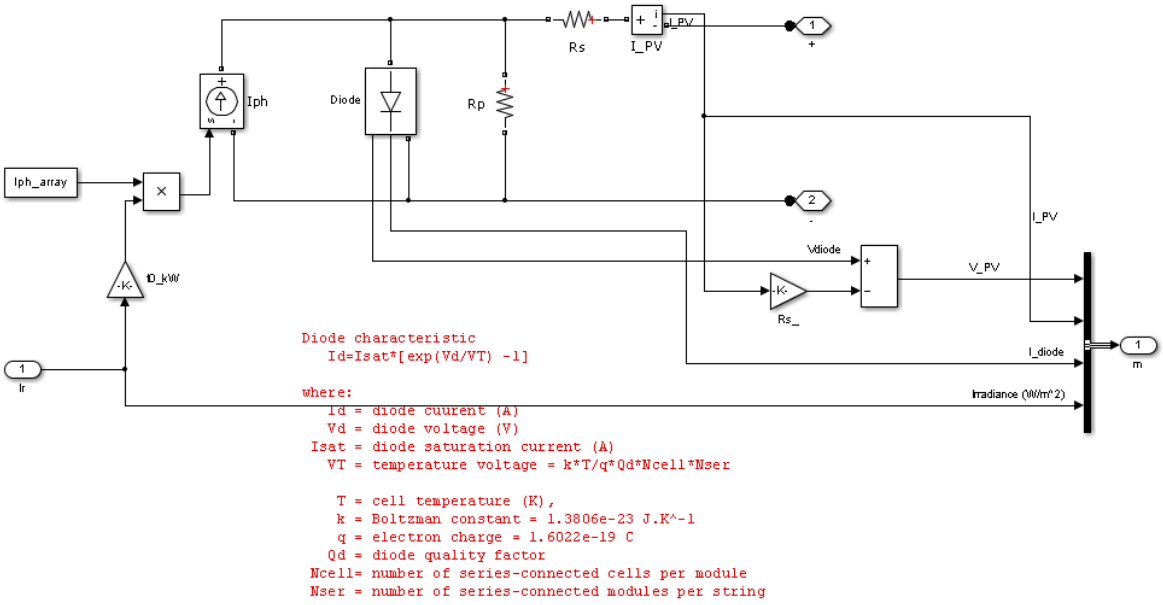


Figure 3.12: PV array module designed in MATLAB

3.4 Modelling of Grid

The grid designing is quite simple as compared to PV array as it doesn't require any specific parameters. The MATLAB designed model for utility grid is shown in figure 3.13 which consists of a total 19 KM feeder with a 2MW load at the 14 KM distance and 30 MW load at the generating point.

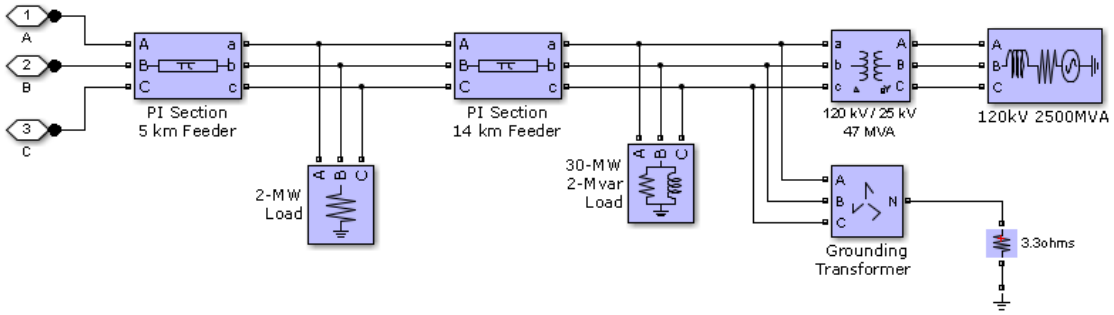
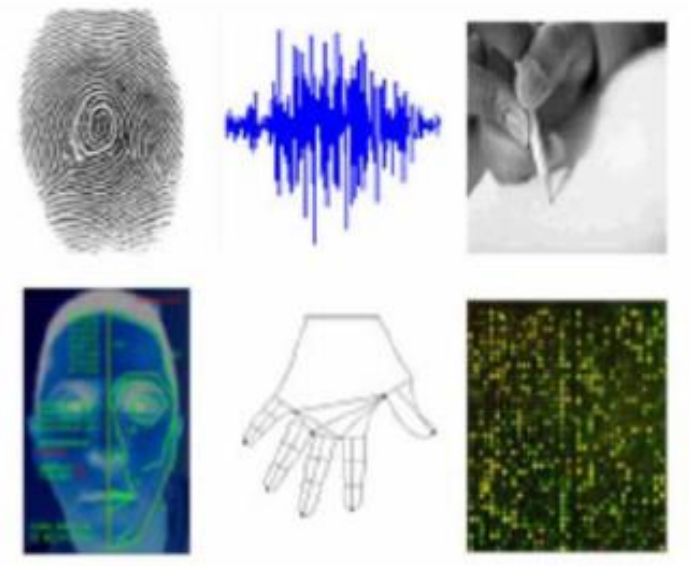


Figure 3.13: Utility grid designed in MATLAB

Thesis -4
Fingerprint Identification by GSA Optimized ANFIS



ABSTRACT

The Automatic Fingerprint Recognition System plays a significant role in forensics and law enforcement applications. The fingerprint scanned are not always ready to use or extract from images and poses a challenge to automatic fingerprint extraction methods. Fingerprints in images can be overlapped, noisy, weak ridges etc. These issues are obstacles in extracting and recognizing fingerprints automatically. The objective of the proposed system in the current study is to identify and separate fingerprint images automatically using a Fuzzy Inference System (FIS) with heuristic algorithm Grey Wolf Optimizer (GWO).Fingerprint database is taken from public access data base FVC2002_DB1_B. we have implemented our work by doing firstly image refinement and segmentation using block based segmentation, second feature extraction to create feature table, third testing and training data using fuzzy logic with GWO and compare our method's performance using ANFIS classifier.

In preprocessing step, image is converted into gray scale and filtered through median filter followed by histogram equalization. Then block based segmentation method is used. The block based segmentation method is to remove the fingerprint background and strengthen the weak ridges in fingerprint by normalizing it. It is based on the coefficient of variation and morphological open and close operations. The coefficient of variation calculates the intensity

change in the image block. This block wise operation is performed over whole image in the block of 4*4 or 8*8.

The intention of feature selection approach is to select a small subset of features. The features extracted from fingerprint images are second order and third order features such as homogeneity, ridge orientation etc. These features are classified with neural network trained fuzzy logic classifier. We proposed it to do by Gray wolf optimized (GWO) fuzzy logic membership functions. Since fuzzy logic is based on the rules and decisions, membership functions, but these are not dynamic in nature and don't perform as desired as in case for which these rules are developed. So we need to tune those for every new test case. We used gaussian bell shape membership functions in it which changes the range after optimization. Five features from a pre processed finger print image are extracted so five input to fuzzy logic are to be used and each input has three bell shaped gaussian membership functions.

This work finds finger prints recognition for fingerprint obtained with label or class, so that during testing, error can be minimized. Three test cases are analyzed with the available dataset. The extracted features are divided into 60:40, 70:30 and 80:20 for training and testing respectively. Our proposed method of fingerprint recognition fits good enough when compared to fingerprint using ANFIS classifier only.

Proposed Work

The Automatic Fingerprint Recognition System plays a significant role in forensics and law enforcement applications. In this work, we proposed fingerprint recognition using fuzzy logic with Grey Wolf Optimizer algorithm (GWO). The objective of the proposed system in the current study is to identify and separate fingerprint images automatically using a Fuzzy Inference System (FIS) with meta-heuristic algorithm Grey Wolf Optimizer (GWO).We have compared our results with ANFIS classifier to check and validate our results. Fingerprint database is taken from public access data base FVC2002_DB1_B available at http://bias.csr.unibo.it/fvc2002/ .Different categories of features are extracted from the segmented fingerprint images. The features extracted from fingerprint images are real values. Overall working can be divided in following points.

1. Pre-processing of fingerprint Images
2. Feature extraction from fingerprint images
3. Testing data using trained data with ANFIS based classifier
4. Testing data using trained data with FIS and GWO based classification

5. Comparison and discussion.

4.1 Pre-processing of Fingerprint Image

Pre-processing of finger print image includes following steps

a) Input a finger print image
b) Conversion to gray scale image
c) Binarization
d) Histogram base equalization
e) Noise removal using median filter
f) Segmentation using block based technique

The fingerprint images under test acquired by the fingerprint scanner are always susceptible by the environment. The image restoration tries to minimize the effects of degradations, by means of a filter. It is a fundamental problem in the image processing to improve quality through the reduction of the noise. A wide variety of techniques dedicated to carry out this task already exist. In the fingerprint image compression system, significant features detection depends on the regions of interest which are usually of noisy nature and low contrast. Hence an image enhancement and denoising may be required to preserve the image quality, highlighting the image features and suppressing the noise. Noise in image not only lowers its quality but also can cause feature extraction algorithms to be unreliable. The denoising and feature enhancement techniques presented in this work will improve the reliability of image processing. In the past many methods for denoising or image enhancement have been developed. The median filter is a non-linear digital filtering technique, generally used to remove noise from images or other signals. The Median Filter is performed by taking the magnitude of all the vectors within a mask and sorting the magnitudes. The pixel with the median magnitude is then used to replace the pixel studied. The Simple Median Filter has an advantage over the Mean filter in that it relies on median of the data instead of the mean. Figure 4.1 shows steps of pre-processing of input finger print image

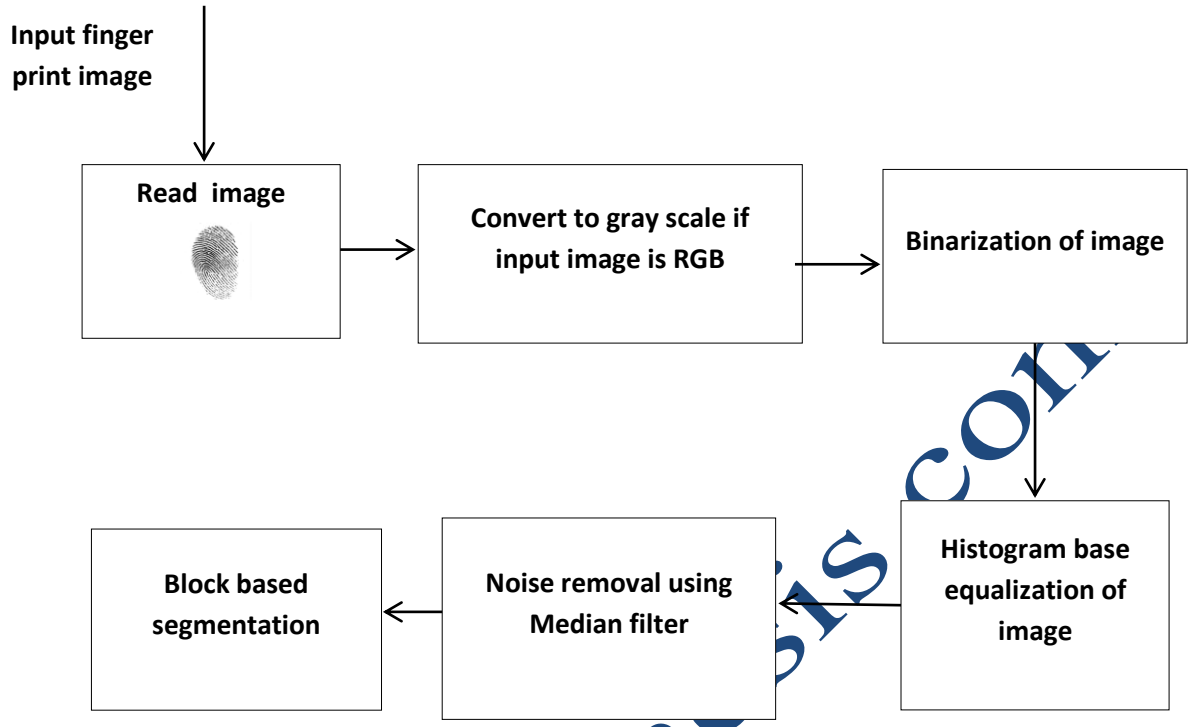


Figure 4.1: Pre-processing steps

In pre-processing, segmentation plays a key role, so we used block based segmentation technique. Fingerprint image Segmentation is a challenging task especially if it contains low-quality foreground regions, unwanted high-frequency regions (background noise) or remaining ridge regions. Here, we have proposed a block based segmentation algorithm for fingerprint images that employs morphological filters and a statistical measure called coefficient of variation (CV). The proposed method is performed in two passes where the foreground region, i.e., the region of interest (ROI), is coarsely separated from the background region in pass-1 based on CV and morphological open-close filters with reconstruction. In pass-2, region shrinking and merging is employed to provide further refinement in the shape of the ROI with the use of CV and average gray value. Figure 4.3 shows the stepwise output of block based segmentation for pass-1 of algorithm and Figure 4.4 shows the stepwise output of block based segmentation for pass-2 of algorithm.

Figure 4.2 explains in details the step followed to implement block based segmentation.

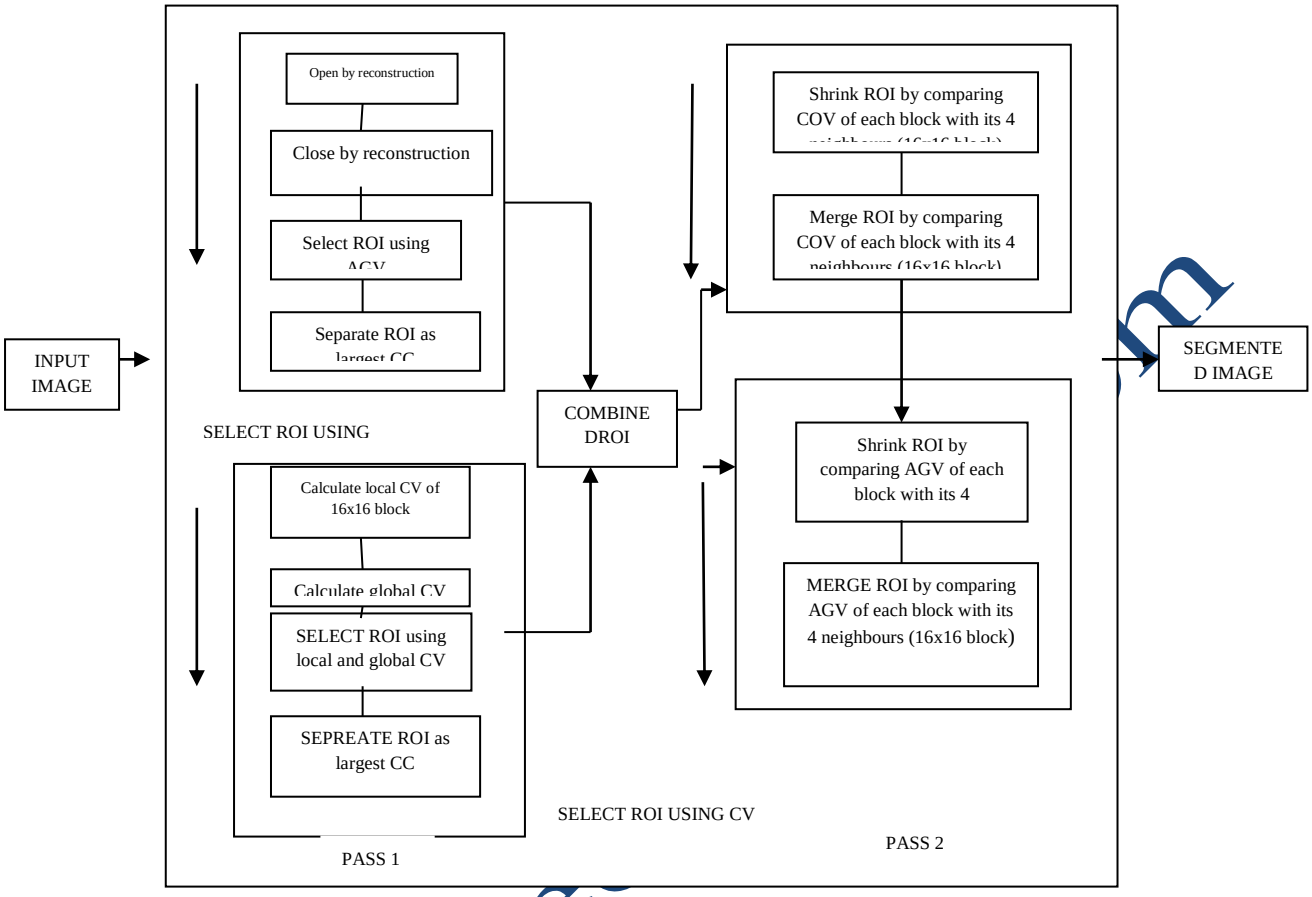


Figure 4.2 Block based segmentation technique for finger print pre-processing and segmentation

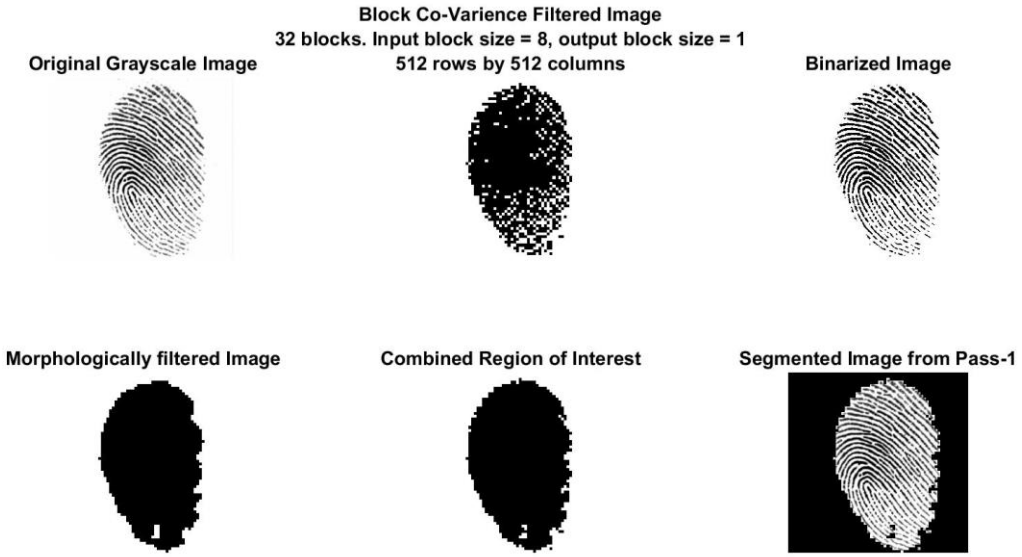


Figure 4.3 Stepwise output of block based segmentation for pass-1 of algorithm

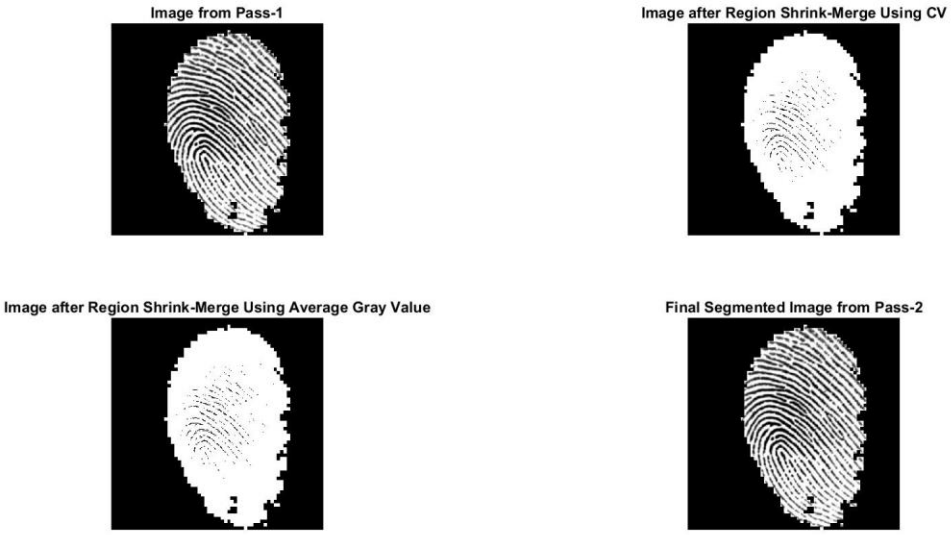


Figure 4.4 Stepwise output of block based segmentation for pass-2 of algorithm

4.2 Feature Extraction from fingerprint Image

There are various features of segmented fingerprint image but to avoid much complexities we have taken essential 5 features for comparison purpose. These features are second order and third order features.

1. Correlation: Correlation is a statistics term which is defined as a mutual relationship or connection between two or more things. This technique is a promising approach for fingerprint matching using matching of ridge shapes, breaks etc.General expression for correlation of two images is shown below.

$$Correlation = \sum_{i,j=0}^{N-1} pi,j \left[\frac{i - \mu i(j - \mu j}{(\sigma i^2)(\sigma j^2)}\right]$$

2. Energy: It gives the sum of square elements in GLCM (Gray Level Co-occurrenceMatrix) generated in MATLAB. Its range is [0,1].The equation of energy is

$$\boldsymbol{energy} = \sum_{\boldsymbol{i-1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j-1}}^{\boldsymbol{k}} \boldsymbol{P^2}_{\boldsymbol{ij}}$$ Type equation here.

3. Homogeneity: It gives the value that calculates the tightness of distribution of the elements in the GLCM to the GLCM diagonal. For diagonal GLCM its value is 1 and its range is [0,1]. The equation of homogeneity is

$$\boldsymbol{homogeneity} = \sum_{i-1}^{k}\sum_{j-1}^{k} \boldsymbol{P_{ij}}$$

4. Ridge frequency: Ridge frequency, is used as a global feature of fingerprint It is very important in image pre-processing methods used for fingerprint matching. It is calculated using ridge distance. Fingerprint ridge distance may be defined as the distance from a given ridge to adjacent ridges. It can be measured as the distance from the centre of one ridge to the centre of another. Ridge frequency is the reciprocal of ridge distance and indicates the number of ridges within a unit length.

Ridge Distance: $\boldsymbol{d} = (\boldsymbol{m_i}, \boldsymbol{m_{j)}} = \sqrt{(\boldsymbol{x_i} - \boldsymbol{x_j})^2 + (\boldsymbol{y_i} - \boldsymbol{y_j})^2}$

Ridge frequency: $\boldsymbol{F(i,j)} = \dfrac{\boldsymbol{1}}{\boldsymbol{d(m_{i,}\hat{m}_j)}}$

5. Ridge orientation: The orientation field of a fingerprint image represents the directionality of ridges. Fingerprint image assumingly divided into number of non-overlapping blocks and an orientation representative of the ridges in the block is assigned to the block based on greyscale gradients in the block. The block size depends on the inter-ridge distance, i.e. it should include at least one ridge and one valley in a block. The block orientation can be determined from the pixel gradients by averaging. It is well known that an estimation of the local orientation θ using block operation can be computed by using

   $\theta = \frac{1}{2} tan^{-1}\left(\sum_{u,v \in w\times w} 2g_x(u,v)g_y(u,v) / \sum_{u,v \in w\times w} g_x^2(u,v) - g_y^2(u,v)\right)$

   Where $u, v \in w \times w$ defines all the points in the 2D region of size $w \times w$ while g_x and g_y are the gradients in the directions x and y, respectively.

The features described above are extracted from fingerprints and normalized using normalization function i.e.

$$\overline{x} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Decision labels for each fingerprint are taken from fingerprint database considering the fact that fingerprint database is made by fingerprints of 10 person having 8 fingerprints each. So we have 10 labels which signify person identity.

4.3 Fuzzy Logic Model using FIS

A fuzzy inference system (FIS) is a system that uses fuzzy set theory to map inputs (*features* in the case of fuzzy classification) to outputs (*classes* in the case of fuzzy classification). Two FIS, the Mamdani and the Sugeno are generally used in FIS. Fuzzy inference system can achieve a better combination of the all input parameters to obtain the optimal output. we constructed a Sugeno FIS in MATLAB. The fuzzy controller consists of three parts: first is fuzzyfication in which real environment variables are converted to fuzzy variables, second is inference model which inherits the rule sets or decision variables and third is de-fuzzyfication which reverse the fuzzy variables to environment variables. Figure 4.6 show snapshot of our sugeno FIS with five features as input and one output as label or class of image.

We have taken five features as input and every input is presented with 3 membership function, so total membership function rules , 3^5 =243 rules are to be made which is represented in figure 4.7.

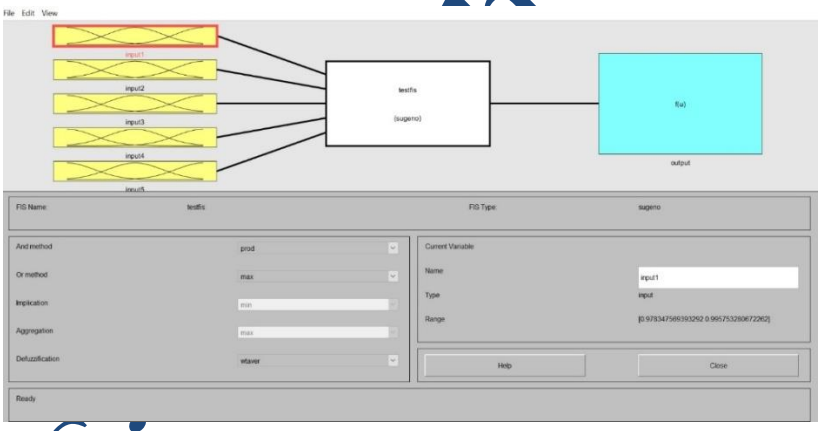


Figure 4.5 fuzzy logic controller with 5 input as features of fingerprint

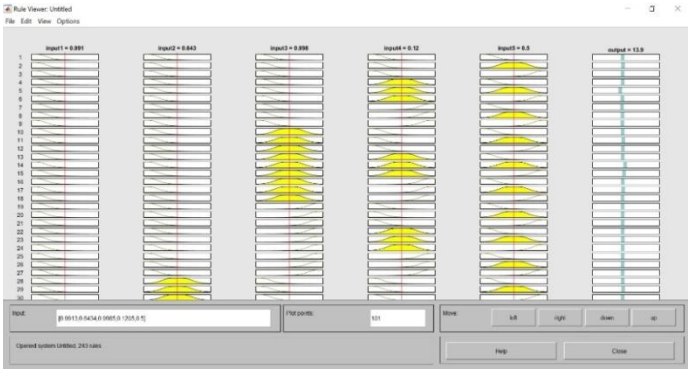


Figure 4.6 Fuzzy Logic Rules set

4.4 Grey Wolf Optimization (GWO) tuning of Fuzzy membership functions:

In previous chapter working and mathematical models of the social hierarchy, tracking, encircling, and attacking prey in GWO are already provided. GWO is used here to tune the membership functions of Sugeno FIS. GWO can be complex if taken more no. of wolves, iteration or alpha wolves size. However, we have kept the GWO algorithm as simple as possible with the fewest operators to be adjusted.

GWO is meta-heuristic optimization algorithm which is used to minimize a cost function for optimum values which are to be tuned. We have tuned all membership functions of all 5 inputs.

Total of 45 parameters are tuned because we have taken gbellmf type membership function which depends upon 3 boundary values and for 5 input with each input having 3 membership functions and each function here gbellmf depends upon 3 boundary values, it means 5x3x3=45 values. We have taken wolf group size also the same as that of parameters to be tuned but it is not essential that size is same , it may be different also. Cost function is taken as MSE (Mean Square Error) which is to be minimized. For fast implementation of algorithm we compromised between iteration size and decided maximum iteration as 50. Table 4.1 shows working parameters of GWO.

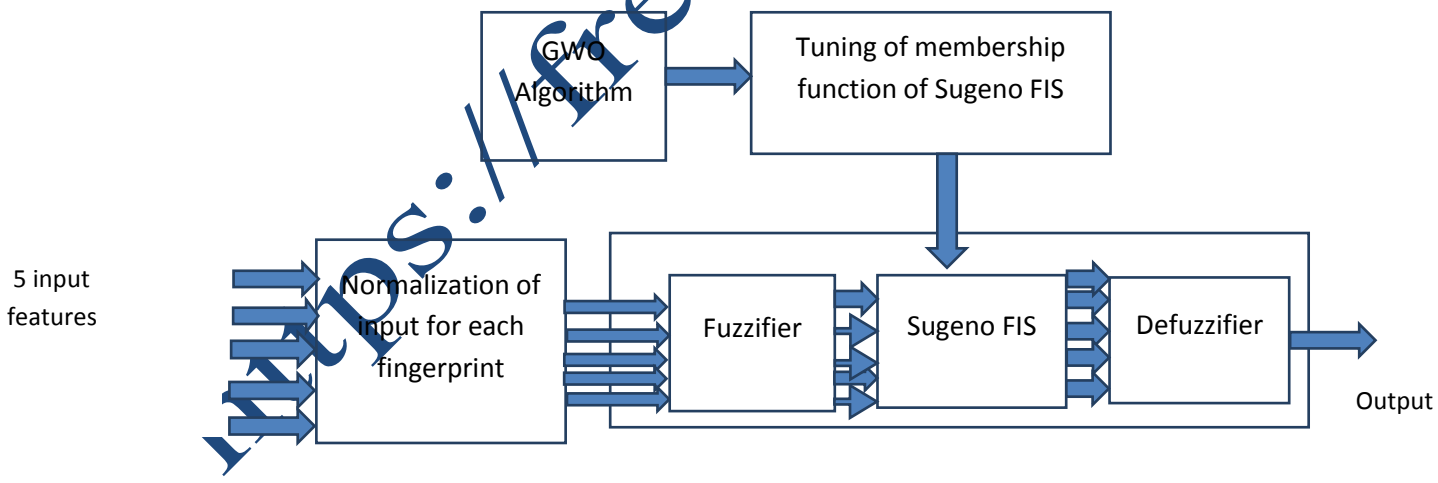


Figure 4.7 Block diagram of FIS membership function tuned using GWO

1. **Fuzzification:** The purpose of fuzzification is to map the inputs from a set of sensor or database to values from 0 to 1 using a set of inputs membership functions.
2. **Fuzzy Inference System (Sugeno):** The primary difference between Sugeno and Mamdani FIS is that the output consequence is not computed by clipping an output membership function at the rule strength. In fact, in Sugeno FIS there is no output

membership function at all. Instead the output is a crisp number computed by multiplying each input by a constant and then adding up the results.

3. **De-fuzzification:** In many instances, it is desired to come up with a single crisp output from a FIS. For example, if one was trying the FIS would have to come up with a crisp number to tell the computer which letter was drawn. This crisp number is obtained in a process known as de-fuzzification.

Table 4.1 GWO parameters

S.No.	GWO Parameters	Value
1	Number of Wolves	45
2	Dimension of problem	45
3	Type of membership function	Gbellmf
4	Number of iterations	50
5	Cost Function	Mean Square Error (MSE)

4.5 Performance Evaluation Parameters:

In order to evaluate performance of FIS-GWO and ANFIS, we have chosen two performance measures

1. Mean square error of trained and tested output
2. Accuracy of output w.r.t. testing input labels.

Details of classifiers are out of scope of this study so we put off this segment for reader as self-exploration. Before applying to ANFIS, we divided our feature data in

1. 80:20 ratio and 80% data is used as training data and remaining 20% is used as testing data.
2. 70:30 ratio and 70% data is used as training data and remaining 30% is used as testing data.
3. 60:40 ratio and 60% data is used as training data and remaining 40% is used as testing data.

Following parameters are used to compare performance of different techniques.

1. Accuracy

2. Mean Square Error (MSE)

The accuracy of fuzzy inference system is the precision of the fuzzy logic model. It is highly related to the measurement of correct estimation of a fuzzy model for a real system. Commonly, accuracy is evaluated as percentage of correct classifications of a given data set, the mean square error (MSE) is also considered as second important performance evaluation measure. It is for the reason we have taken MSE as our cost function in GWO.

$$\text{Accuracy\%} = \frac{\sum_{j=1}^{N} \delta_{y_j,t_j}}{N} \times 100$$

$$\text{Mean Square Error (MSE)} = \frac{1}{N}\sum_{j=1}^{N}(y_j - t_j)^2$$

Where $\delta_{kl} = 1\ or\ 0$ according to whether k=l or k≠ l respectively; N is the size of data set,y_j is the system output for *j*th data point, t_j is true, desired output for the *j*th data point.

4.6 Complete work flow diagram:

We have implemented FIS theory with GWO for fingerprint matching efficiently and speedily. Following the framework given in figure 4.1, we have implemented steps and found that GWO using FIS for fingerprint matching is giving better results than ANFIS based classification. We have checked our results on various performance parameters.

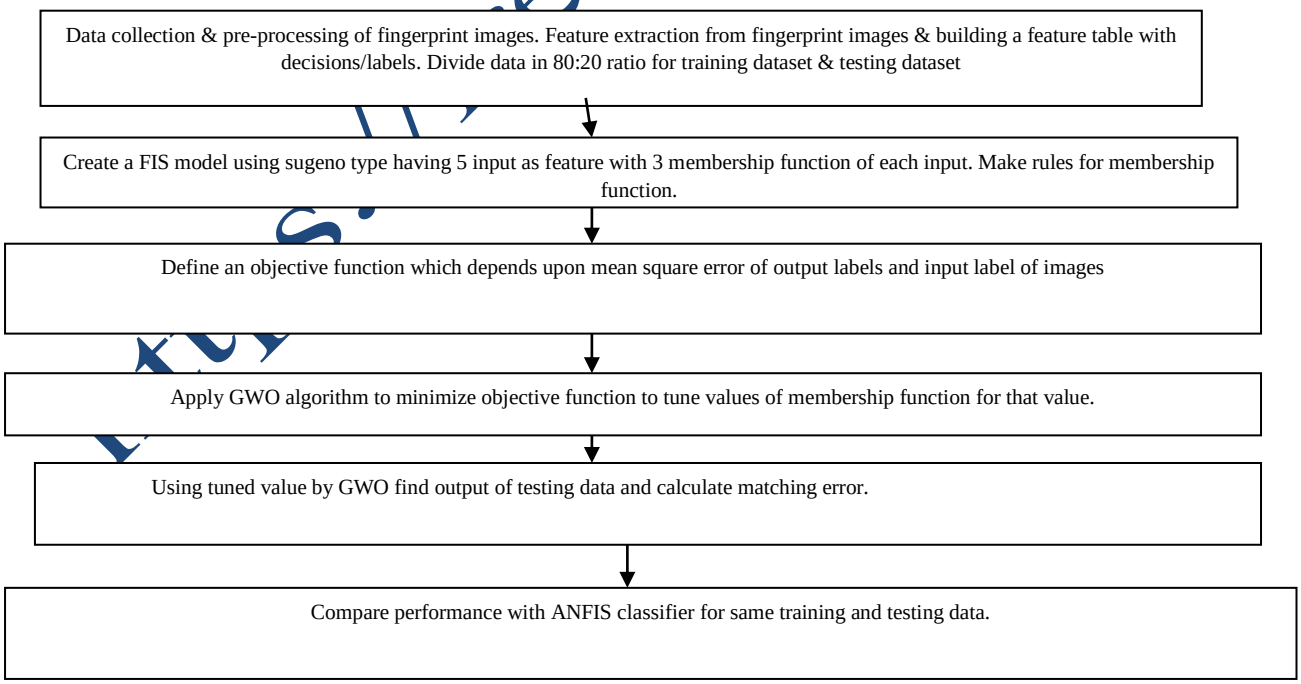
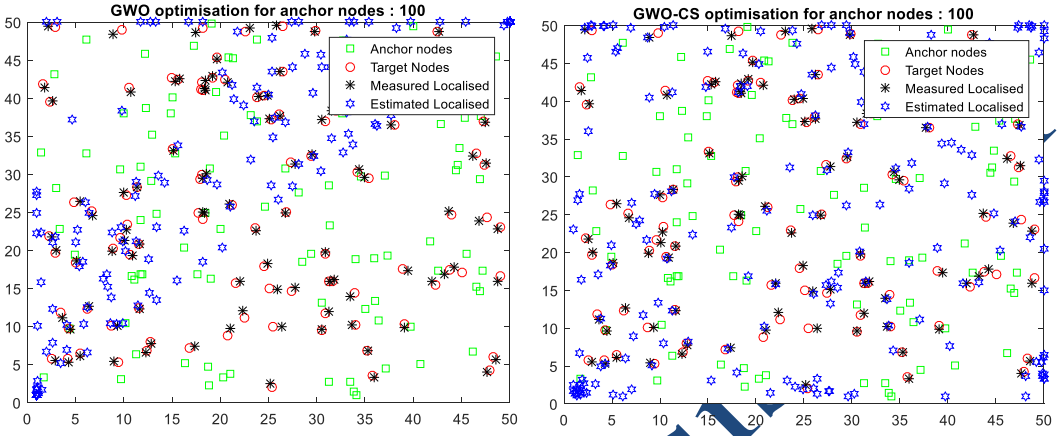


Figure 4.8 Overall work flow diagram of FIS with GWO algorithm

Thesis-5
WSN Localization using GWO and CS Optimization



Abstract

In WSN field, accurate location of sensor nodes is highly required and some of the sensing operations require the location of target node. These are used usually to sense the temperature, moisture etc and fault tolerant. This makes their applications in military, civil and industrial applications.

Since some of the sensing applications are based on the location of node, our work is focused on that research. We used a hybrid heuristic optimisation method to accurately locate the host sensor node which is the hybrid of Grey wolf optimisation algorithm (GWO) and cuckoo search optimisation (CS). Both algorithms are combined to use the advantages of both and results are compared with GWO optimisation only. The hybrid function is also tested over few unconstrained linear and non linear benchmark functions and GWO-CS is the winner in terms of minimum fitness value in each function. The work is evaluated on the basis of mean localization error, number of localized nodes and computation time in each case. The nodes localization is a complex problem and backed by several constraints. This needs the division of nodes into anchor, reference and unknown nodes. Anchor nodes are equipped with GPS to know their location exactly. These help to find out other nodes' positions. In our test cases which differ by geographical area in which nodes are placed, we found a optimal area of 100 m^2 for the maximum number of localized nodes and minimum localization error. Each test case is tested with different number of anchor nodes and it has been found that in 100 m^2

area, 80 anchor nodes give maximum number of localized nodes. By our proposed hybrid optimization algorithm, an improvement of 13.8% is achieved over GWO localized nodes.

Proposed Work

Wireless sensor nodes have been very useful and able to attract a large community of researchers which are working on to solve their issues. In case of unusual disaster or locations where man can't go, sensors are thrown by some means like form air or drones and to bring those sensors in network communication, their locations must be known. GPS can also be attached to each sensor but it will be very costly and also GPS consumes a lot of power, so sensor node will exhaust soon. So unknown node localization is the important part in WSN research. Our work is focused on node localization. To know the locations, some nodes' positions must be known or it can be said only a few nodes with higher battery capacity must have GPS installed. A unknown node can be located by measuring the distance by received signal strength index (RSSI) which is fine tuned by heuristic optimization algorithms. We focused on grey wolf optimization (GWO) for this purpose but it suffers from behavior of falling into local minima. So the fine tuned distance of unknown localized nodes by this method can't be reliable. So we introduced a hybrid method which overcome the weakness of GWO and give it the strength of global optimization. We used cuckoo search optimization algorithm along with the GWO. Cuckoo search (CS) is discussed in previous chapter. CS updates the best three solutions of GWO which are alpha_wolf, beta_wolf, gamma_golf.

5.1 How CS Makes Hybrid with GWO

Once all wolves are initialized with some random feed then fitness function is calculated for each wolf. In a group 10-20 wolves are considered. Out of them the one with minimum fitness function (as GWO works to reduce the distance between prey and wolf and optimal position is the prey position whereas CS works for maximization of profit) is considered as leader of the group and α_wolf, followed by two more wolf with corresponding decreasing fitness function as β_wolf and γ_wolf. The mean of these positions is considered as optimal position of wolf in that iteration.

$$GWO\ optimal\ position = \frac{\alpha_{wolf} + \beta_{wolf} + \gamma\_wolf}{3} \qquad \ldots\ldots (5.1)$$

. In GWO, to move towards the prey, the distance between prey and golf is minimized and changed over time. The step size by which wolf moves is randomly weighted by a constant which leads to fall it into local optima. This problem is solved by cuckoo search algorithm

which update the current position based on the best position so far. CS optimality is more relied on other habitat groups rather than only time. To make it hybrid we updated the best three locations of wolves in the group by CS method which update it by a step of λ with angle ω. The step size is updated as:

```
stepsize=w*step.*(s-best);                    ...(5.2)
```

where 's' is the position of alpha_wolf, beta_wolf and gamma_golf

'step' is the previous step size of cuckoo movement

'stepsize' is the updated step size

'w' is the weighting factor = `0.001`

The position of cuckoo is now updated as:

$$s = s + stepsize \times \omega \quad \ldots.(5.3)$$

where ω is the deviation of cuckoo and a random quantity.

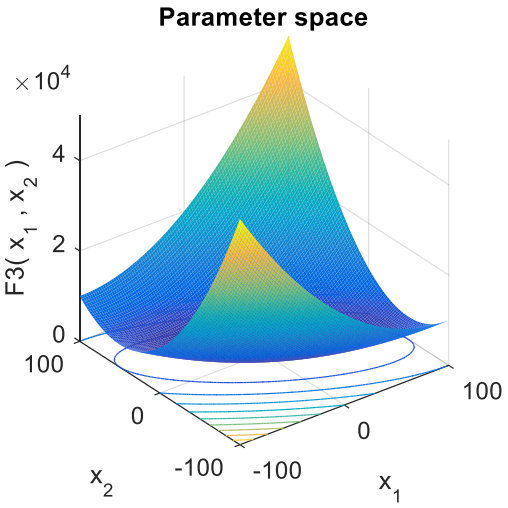
Using equation 5.3 the $\alpha_{wolf}, \beta_{wolf}, \gamma_{wolf}$ are updated to new positions and handle will get back to GWO form CS. Now GWO takes mean of all three best positions again and tradeoff the local optima error in this hybrid.

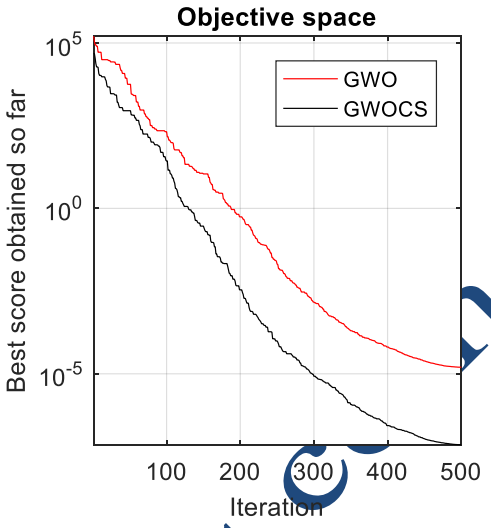
We tested this hybrid GWO-CS function on benchmark functions too and compared with GWO algorithm as can be checked in table 4.1. We took benchmark function reference from GWO paper [13]. We used few random functions from different benchmark function categories. It has been observed that hybrid approach of GWO-CS is performing better than GWO only and removes the local optima issue in previous work. Every objective function's 3D surface view is also shown in table along with the convergence curve. The lower the curve better is the algorithm as parameter space 3D view is pointing/converging towards a minima point.

Table 5.1: Benchmark functions results for GWO-CS and GWO optimization

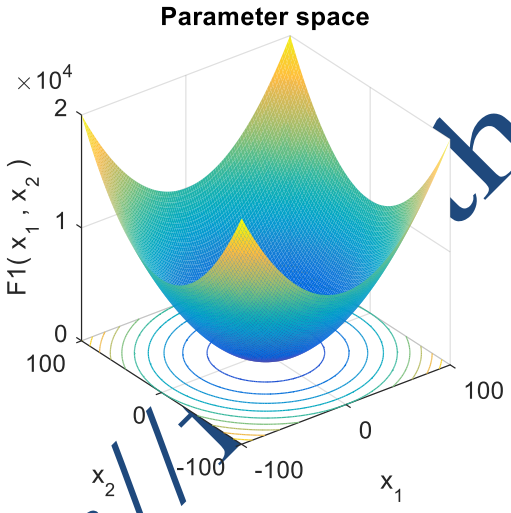
Functio

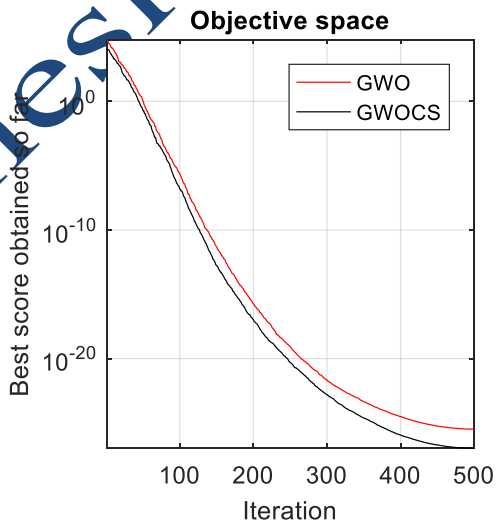
n f_3



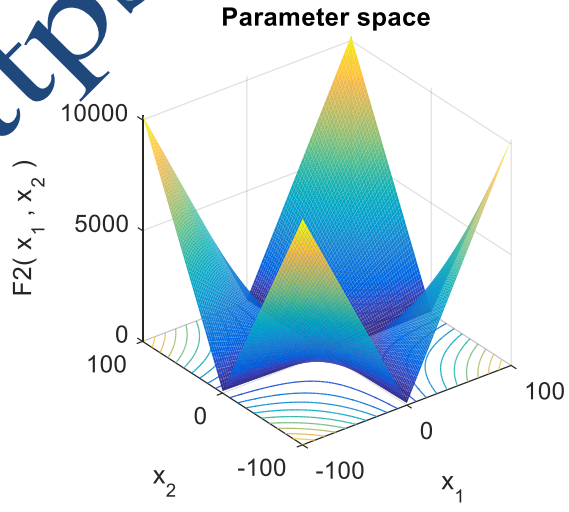


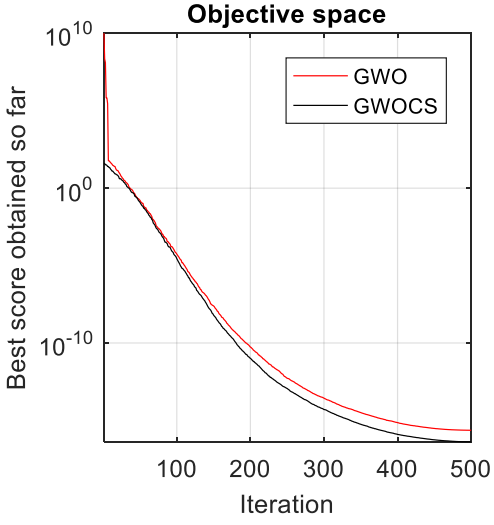
f_2



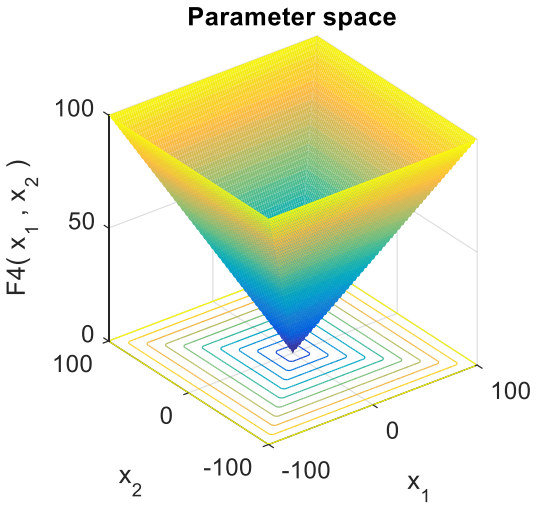


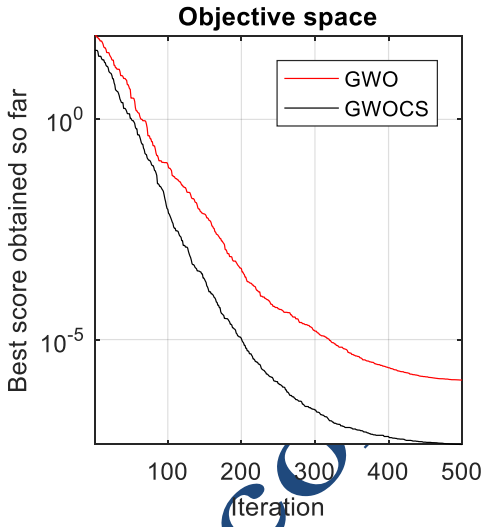
f_1



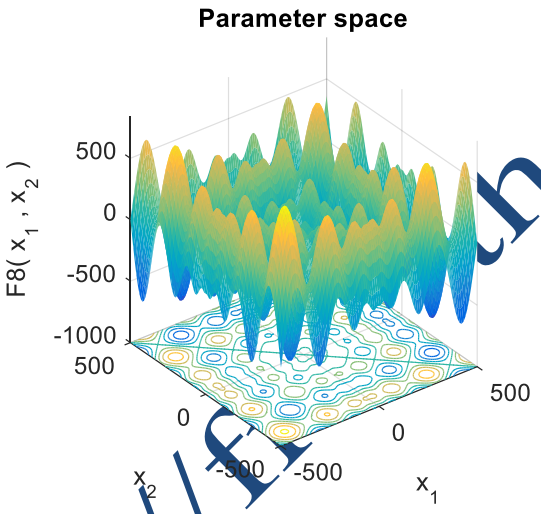


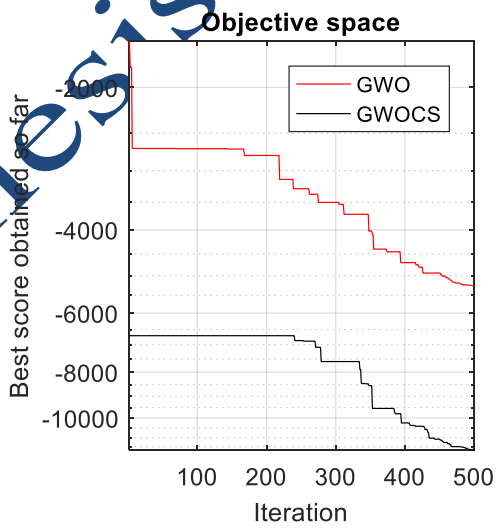
f_4



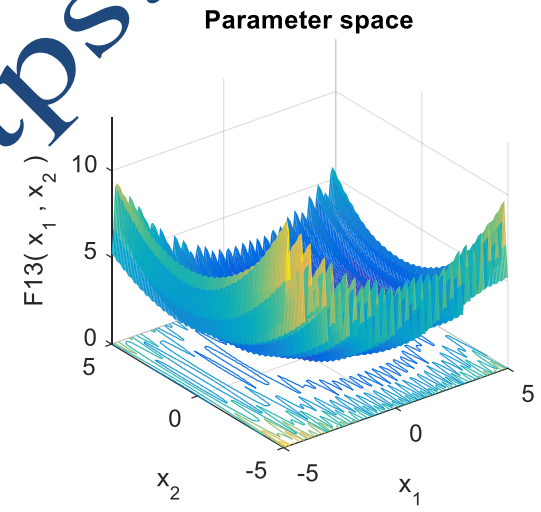


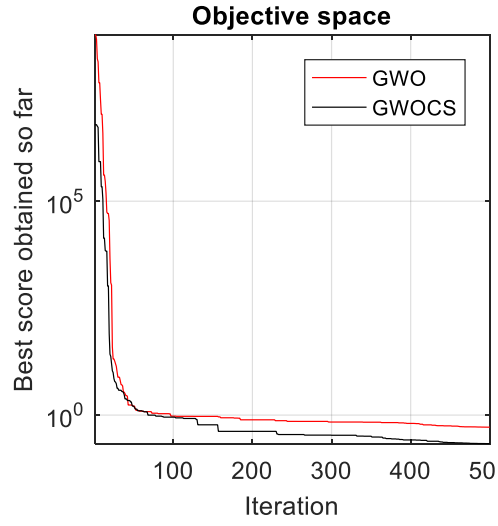
f_8



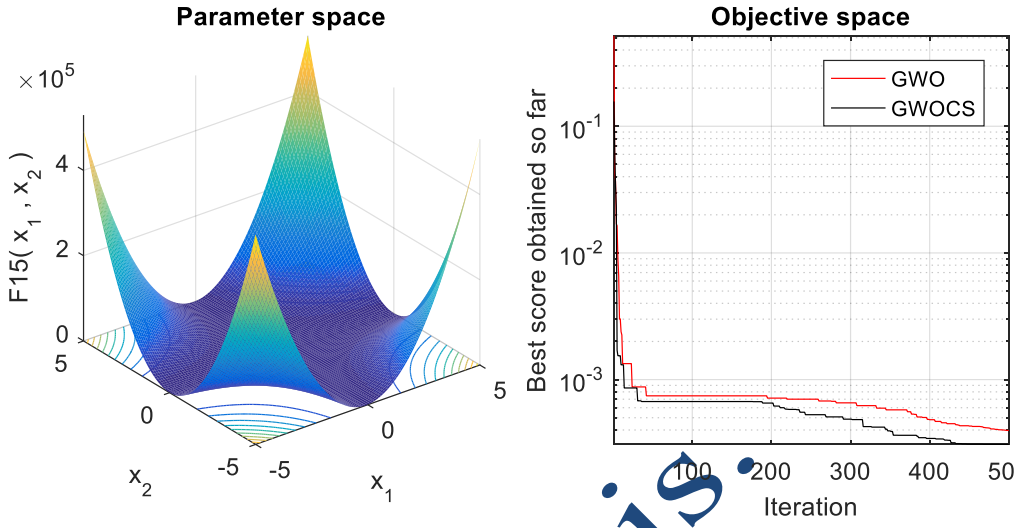


f_{13}

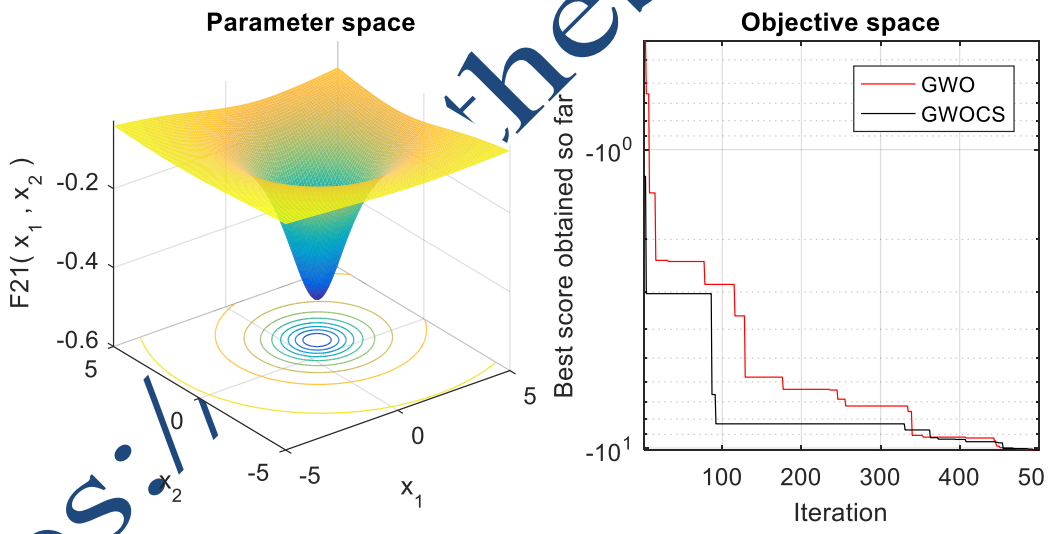




f_{15}



f_{21}



5.2 Node localization using GWO-CS

To location the location of unknown nodes in WSN, we are fine tuning the location considered by the RSSI method. All nodes are categorised into three categories: anchor nodes, reference nodes and unlocalized nodes. Anchor nodes are those nodes whose position is known to base station, those unlocalised nodes whose positions are determined by the algorithm are reference nodes and can take part in locating the other un-localised nodes. Hybrid method finds the location of sensor nodes optimally and compares the distance with the actual one. The minimum is the error, better is the position. The GWO-CS and node localisation are two isolated system but their dependency can be depicted as in figure 4.1.

These collectively form a feedback loop. The module of GWO-CS gets the relative error of distance of nodes in the input and gives the updated nodes' positions to the WSN module.

There are two tuning variables for each unlocalised node in the area which represents the x and y co-ordinates of the node. These tuning variables are the positions of wolves and updated by hybrid optimisation to find a node with minimal distance from the actual location. The error must be minimum.

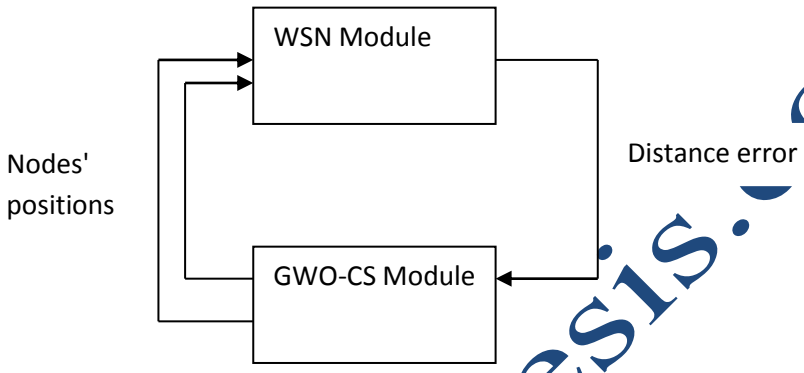


Figure 5.1: Relation between node localisation and GWO-CS optimisation

The optimisation process is the iterative process and in each iteration the each wolf position will be updated and sent to WSN module in figure 5.1 which calculates the distance error.

5.2.1 WSN Module

There is constraint in the WSN module, only that node can be localised which is in range of at least three anchor nodes. If node is not in range of three anchor nodes, either that is dropped or a new position is assigned to that node. The distance of those un-localised nodes is calculated from all anchor nodes in vicinity and relative difference of sum of measured distance and sum of calculated distance (RSSI method) is observed which is passed back to optimisation module.

If any un-localised node is localised then it acts as reference node and helps the anchor nodes to find out the other nodes' position as anchor nodes.

5.2.2 Optimisation Module

The relative error is feedback to hybrid optimisation for each wolf in each iteration. The minimum error position wolf is determined by comparing it in group and that corresponding wolf is assigned as leader of group along with two other wolf with lesser fitness value and

assigned and beta and gamma wolf. These three optimal position for present node which is termed as wolf inside optimisation module is again updated by cuckoo method as discussed in section 5.1. A chart for it can be shown as in figure 5.2.

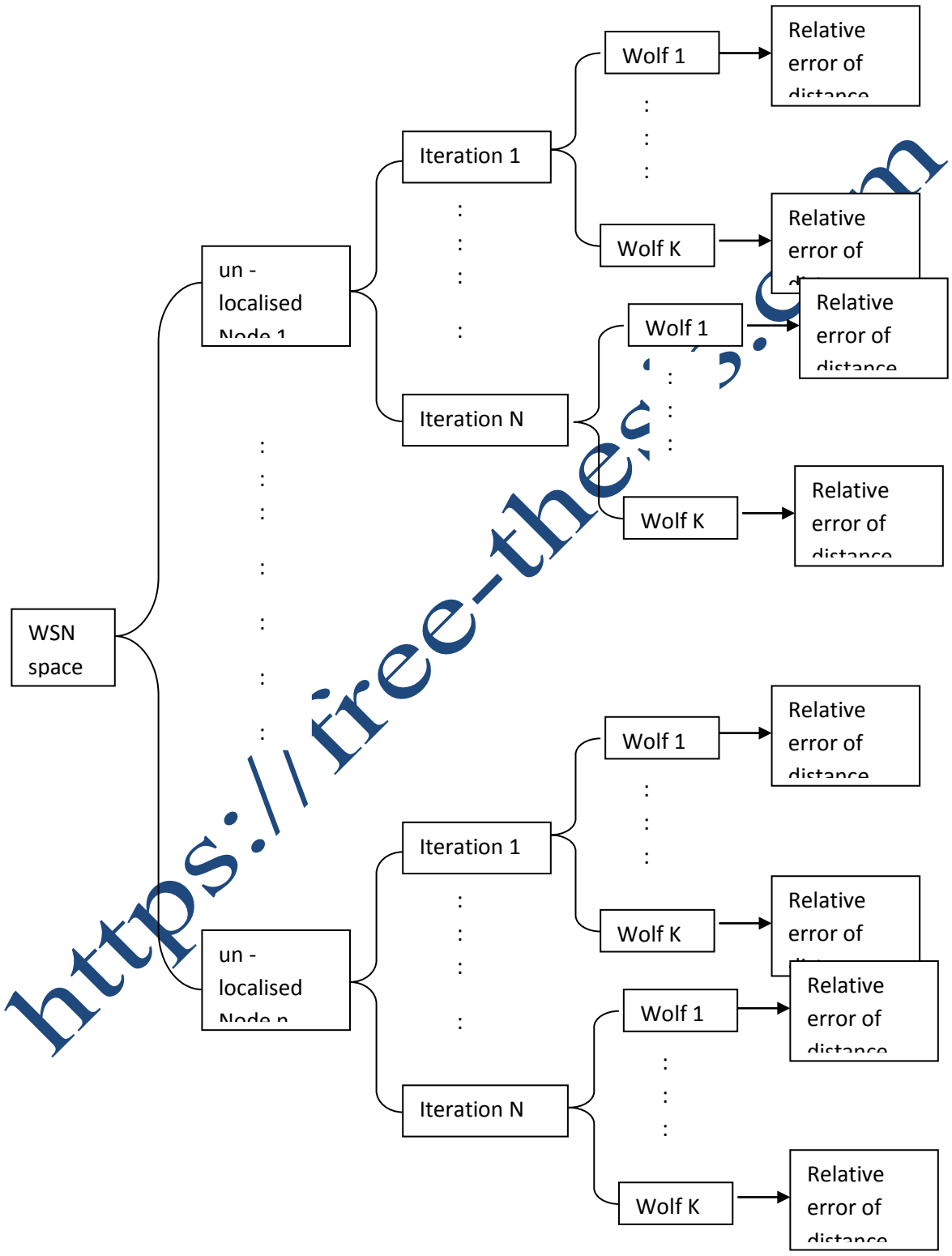


Figure 5.2: Pipeline for whole process for the interaction of WSN module with hybrid optimisation module

5.3 Steps for Unknown Node Localisation

Step1. randomly place the nodes in an area of $100 \times 100\ m^2$

Step2. Fix the few sensor nodes as anchor nodes whose locations are known and rest are to be localized.

Step3. Measure the unknown node's locations using RSSI method. These locations will be used to compare the calculated distance by hybrid optimization algorithm.

Step4. start a loop for each node and call the optimization module. Inside this module, initiate the wolf position randomly within the WSN geographical area which is [1,100].

Step5. for each wolf, optimisation module (objective function) is called which checks if that node is in the vicinity of at least three anchor nodes. if yes, then distance from those anchor nodes is calculated and compared with the distance of corresponding measured node (step 3) with anchor nodes.

Step6. Relative error is calculated and sum of Relative error is passed back to GWO-CS optimization. This step is repeated for every grey wolf in the group.

Step7. Top three best wolves are selected for which error comes out to be minimum in present iteration and these wolf's position are updated by cuckoo search optimization by equation:

$$s = s + stepsize \times \omega$$

notations are given in section 4.1. The step size is calculated as in equation 4.2.

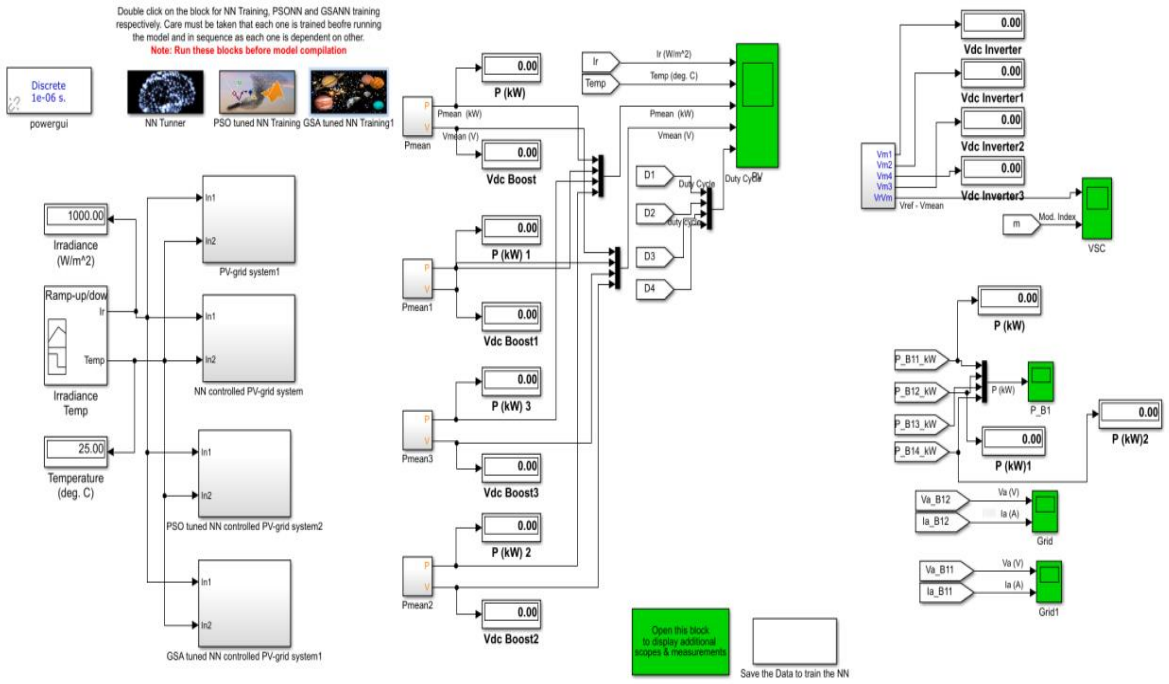
Step8. Updated positions for three best wolves are used to calculate the overall best position for present iteration and step5 is called again to calculate the relative error.

Step9. This process continues till whole iterations for each nodes are not finished.

Step10. results are evaluated on the basis of minimum localization error, number of localized nodes and computation time.

Step11. A comparison is done with the GWO results for all these three parameters.

Thesis- 6
GSA Optimized NN Controlled MPPT in PV-Grid System



Abstract

The findings in this work highlight the MMPT tracking to boost the power transmission from PV array to grid in conjunction with an established neural network fraud detection system. The principles of neural networking are motivated by the functions of the brain especially pattern recognition and associative memory. The neural network recognizes similar patterns, predicting future values or events based upon the associative memory of the patterns it has learned. The advantages neural networks is that these models are able to learn from the past and thus, improve results as time passes. They can also extract rules and predict future activity based on the current situation. In this thesis, Neural network is tuned with gravitational search algorithm and compared with PSO tuned NN. The improvement in mean square error is noticed in purposed optimisation. GSA is performing well because it's a global meta optimisation technique and PSO is local optimisation algorithm which converges prematurely unlike GSA. Our system achieves less MSE than PSO tuned neural network and highest maximum power is tracked from the PV array.

Proposed Work

The MPPT control in grid connected PV array has always been a research area so that maximum power can be transferred in distribution lines. With the evaluation of artificial intelligence it is getting a new level. In this work we used neural network initially to track the maximum power generated form PV array and then NN is updated with optimisation method which is gravitational search algorithm (GSA). Previously particle swarm optimisation (PSO) was used for this purpose. Optimisation process changes the input weights and biases values of NN to achieve more less error. As discussed in previous chapter NN is also an iterative process which changes its input weights and biases to achieve the minimum mean square error (MSE). It is using feedback propagation loop which is using Lquenberg algorithm. This algorithm iterates locally which means it doesn't guarantee the convergence of all minima points. it may skip some combinations of input weights and biases which may reduce the MSE more. To avoid this issue we have adapted the optimisation method named Gravitational Search Algorithm (GSA) which is explained in previous chapter. It is based on the movement of celestial bodies and position of these agents are input weights and biases in our case. The output of NN is calculated by formula in 6.1.

$$output = input * IW_i + B_i \qquad (6.1)$$

where IW_i are the input weights and B_i are the biases. The number of input weights and biases depends upon the number of hidden layers. The GSA algorithm is supposed to tune these values. For this purpose first the Neural network is created in MATLAB. That network will be used further for optimisation algorithm.

MPPT Dataset Generation for NN training

Dataset for this purpose is generated from our simulink model in ideal case. MPPT control is dependent upon the duty cycle fed into the boost converter. This duty cycle is controlled by neural network. For this purpose the solar irradiation and temperature are considered as input to neural network and duty cycle is the output as shown in figure 6.1.

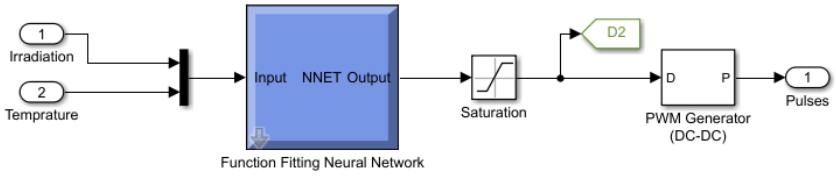


figure 6.1: NN input and output for grid connected PV array

We fix variable temperature and irradiation and save the duty cycle generated by incremental conductance method. A total of 25000 samples are saved as the data. A graph for saved data is shown in figure 6.2.

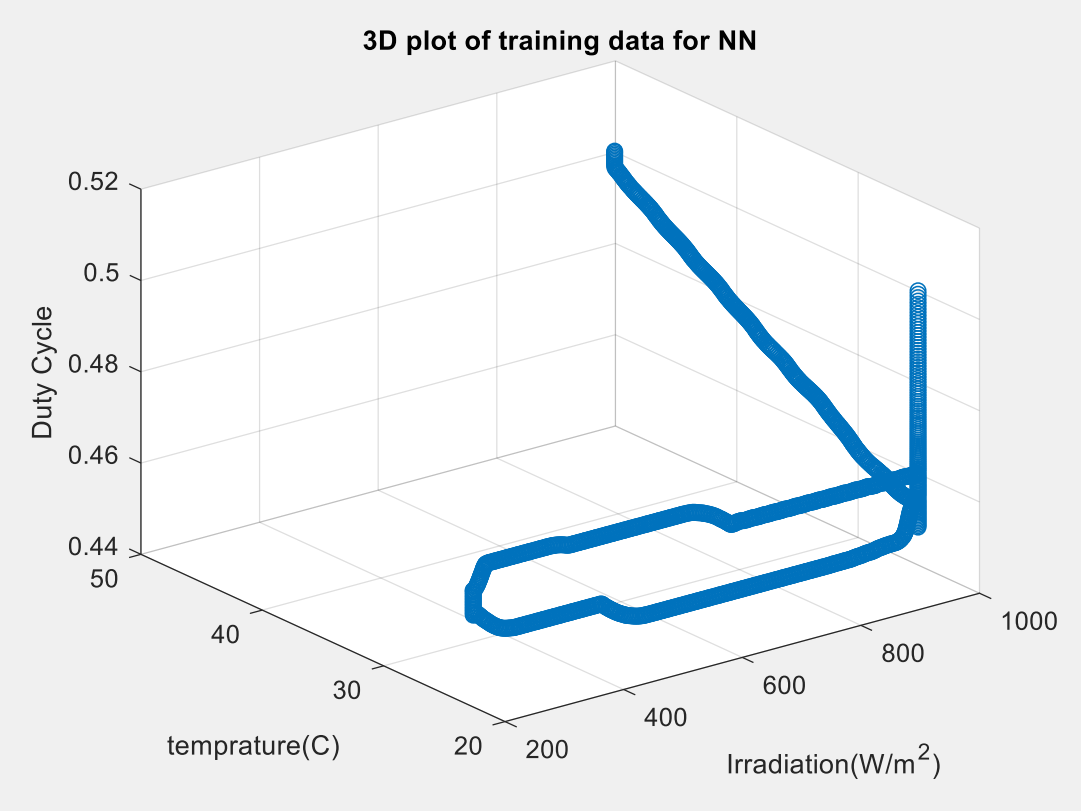


Figure 6.2: Data set plot with variable temperature and irradiation as input to NN and duty cycle as output of NN

The NN training get better with more number of samples in the dataset. This can be done by adding the output duty cycle of NN in this dataset and use the whole dataset again for training.

Now further we divide our this chapter in two sections. First section will discuss and show how NN code is generated for our purpose using MATLAB's neural network toolbox and second section will explain how GSA optimises the NN's weights and biases.

6.1 MATALB Script generation of NN for PV grid Data

Neural network toolbox requires a training dataset for training purpose so that it can get trained and learn the behaviour of data. We don't have separate dataset for testing and training so we divided the present dataset randomly in 70/30 ratio to use 70% for network training and 30% for testing.

MATLAB provides a neural network toolbox which can be used for several purposes and network this trained can be deployed as standalone application or can generate a script for further use of modifications. We used this facility to speed up our work. A user interface of NN toolbox can be opened by using command *nnstart* in MATLAB's command window.

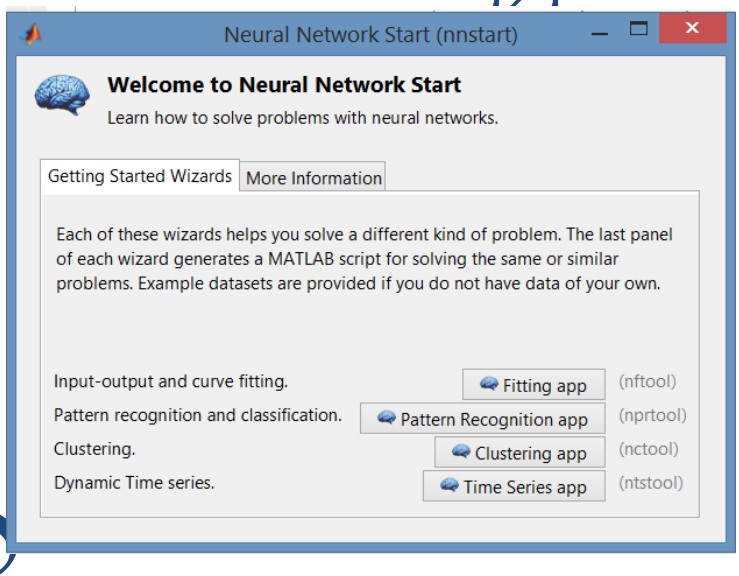


Figure 6.3: MATLAB's NN toolbox interface

Figure 6.3 shows the interface opened from this command. Since our work is recognising the pattern of previous temperature and irradiance as input and duty cycle as output, so we will use it in pattern recognition app which lands to a page to chose the input data and target data. These datasets are picked from workspace of MATLAB, so these must be there already.

After choosing the data division for training and testing of network, network is created which further leads to a page where user can input the number of hidden neurons. We have set hidden neurons at 20. Figure 6.4 shows that landed page.

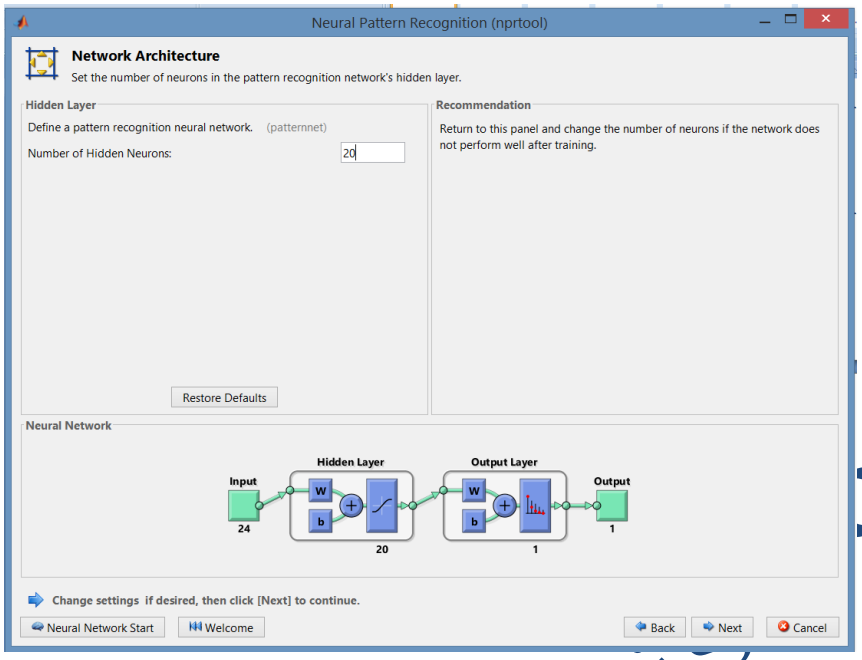


Figure 6.4: NN toolbox UI for entering hidden neurons

Then this network is trained for loaded dataset and tested with rest 30% of data. After training mean square error is generated and displayed on the user interface. Thus trained and tested NN by this toolbox can be converted in the form of MATLAB script which is require in our work. Figure 6.5 shows the option in NN toolbox interface for it.

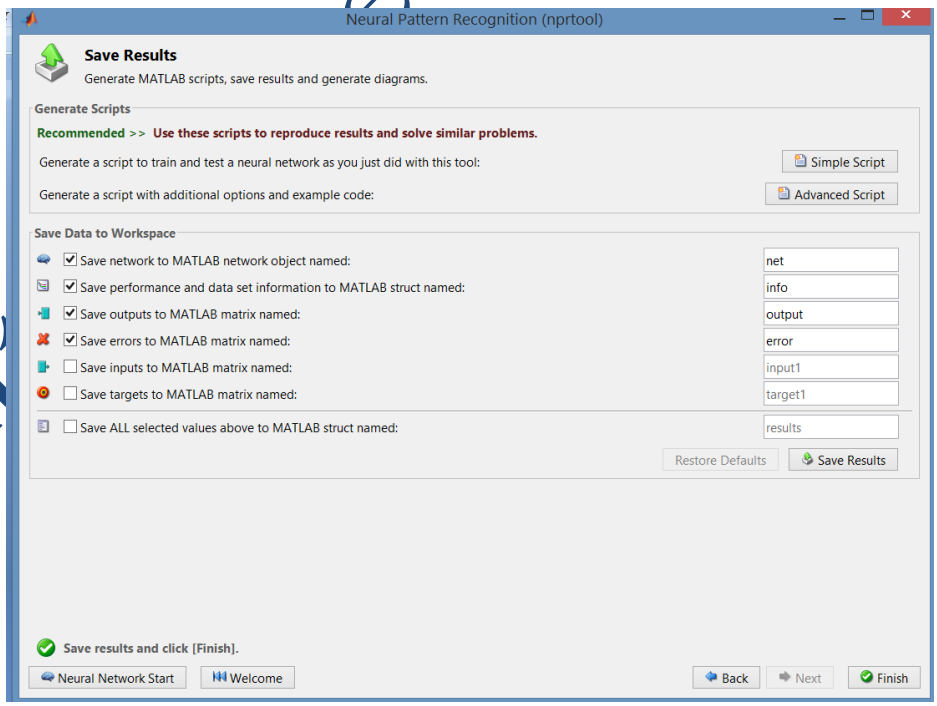


Figure 6.5: NN toolbox interface to generate the require NN script

6.2 Neural Network optimisation by GSA

The proposed work is to tune NN to get the high accuracy and less mean square error to generate the duty cycle such that maximum power can be transferred from PV array to grid. To achieve this aim we use GSA optimisation and tuned NN's weights and biases. in every optimisation task, it is required that an objective function must be set which calculates the target value like MSE in our case. This objective function will be called in each iteration and for each agent in that iteration. since the neural network is already created and trained in previous step so it is not required to create again every time when objective function is called as our objective function updates the pre trained NN's weights and biases which are 41 in numbers and calculate the MSE for those set of weights and biases. The developed objective function snippet is shown in table 6.2.

Table 6.2: Objective function snippet for GSA tuned NN optimisation

```
function [performance,net]=Objective_function(L,~,net,input, target)


%   input - input data.

%   target - target data.


x = input';

t = target';

t(t==2)=0;

net=setwb(net,L); % set the input weights and biases of NN using values in 'L'

% Test the Network

y = net(x);

e = gsubtract(t,y);

performance = perform(net,t,y);
```

end

GSA is based on its agents' movements and an agent's position is represented by the weights and biases values. The number of co- ordinates of an agent's position is equal to total number of input weights and biases. In our case this number is 41. These weights and biases can be fetched from generated neural network by using a MATLAB function 'getwb' and after updating these are set back to NN by 'setwb'. The significance of GSA terminology with NN tuning is provided in table 6.3.

Table 6.3: Significance of GSA terminology in NN tuning

GSA terms	In NN tuning significance
Agents Position	Input weights and biases
Dimension for optimisation/ number of variables to be tuned	Total number of input weights and biases
Update in the position of agents	Change the values of weights and biases to move towards minimum MSE

A complete step by step algorithm is explained below.

Step1. Load the pre-generated input/output data of MPPT in numerical format and divide that into random 70/30 ratio for training and testing of neural network.

Step2. generate the NN script to create and train the network whose weights and biases are to be optimised.

Step3. initialise the GSA parameters like number of iterations, number of agents, initial G0 and alpha. Pass the previously created network into GSA to get the dimension of weights and biases.

Step4. randomly initialise the new input weights and biases to give an initial seed to GSA optimisation. These must be within a boundary as given in next chapter.

Step5. call the objective function to update the neural network's weights and biases and calculate the MSE for those values by using the testing dataset.

Step6. to update the random positions of agents, force and mass has to be calculated by using the equations

$F_{ij}^{d}(t)$ = G(t) $\frac{M_{pi}(t)\times M_{aj}(t)}{R_{ij}(t)+\varepsilon} X_j^d(t) - X_i^d(t))$,

$m_{it}=\frac{fit_i(t)-worst(t)}{best(t)-worst(t)}$

the respective notations are given in previous chapter

Step7. The new updated position is obtained from the formula

$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1)$

the velocity in this case is calculated by using acceleration which is based on force and mass calculated in previous step.

Step8. For this new updated position or values of weights and biases, objective function is again called and MSE is saved.

Step9. The weights and biases for which minimum of MSE is obtained out of previous two set of values, is further considered for updating.

Step10. This process continues till all iterations are not completed.

Step11. The final minimum MSE is obtained and weights ad biases set for them is used as final NN weights and biases which gives less MSE than conventional NN and PSO tuned NN which was done previously.

Thesis-7
Maximization of Profit in Multiple EV Charging by Optimization

Abstract

Electric vehicles are increasing day by day and with these demands, charging stations are also increasing. Researchers are continuously working to minimize the cost of charging so that EV usage can be motivated more for the greener environment. For this purpose smarter charging stations are required which can sense the status of vehicle and schedule the power distribution for minimal cost. We proposed ant lion optimisation (ALO) algorithm for this purpose which is a global optimization algorithm and doesn't fall into local minima like genetic algorithm which is used in previous work. The intelligent charging of EV is a multi-objective problem defied by constraints. Stochastic algorithm like ALO can perform better to schedule this problem. We converted this multi-objective problems into a single objective optimization algorithm. Electric vehicle when attached to charging point then the load demand on grid also increases if several are starts charging at once. This increase in load demand at grid also increase the cost of per unit in charging. So in our work we have optimally distributed the load demand over all vehicles so that it gets charged till it desired SoC and also the load demand doesn't exceed much. Then the scheduling is done in a way so that per unit price is distributed optimally among all vehicles. Our proposal shows the improvement upto 51% whereas genetic algorithm optimally tuned schedule showed it upto 12% only.

Proposed Work

7.1 Research Gap

Electric vehicle (EV) charging has been always a research interest with the advancement in the electric vehicle. The objective of intelligent EV charging is to charge the vehicle battery to customer desired level which is 95% of battery at minimal cost and less time. The charging time is also an issue since with the increase in charging time, cost of charging increases. Also the power of vehicle battery for different companies make vary which needs different charging schedule for different vehicles. Along with these issues in intelligent electric vehicle charging, the electricity supply rate from the distribution grid is also important factor as during high demand peak times, the per unit price of electricity increases. To deal with

these kind of issues, intelligent charging of electric vehicle is required which can consider these constraints. These all makes the charging of EV, a multi-objective optimization problem which requires heuristic optimization algorithms. Previously this problem was dealt with Genetic algorithm which is a local optimization algorithm. This method has the problem of falling into local minima and not giving the accurate results because the stochastic evolution is not done in this. Stochastic evolution parameter make the nature of optimization global and jumps off when the optimization stuck into local minima. We proposed the ant lion optimization (ALO) algorithm here which is a global optimization technique and converges accurately than GA.

7.2 Problem Formulation

When an electric vehicle connected to the charger then charger gets all information like initial state of charge (SoC), desired SoC, plug in time, departure time etc. These all information are passed to the controller which estimate the charging time and adjust the cost of charging too to put less burden on customer's pocket. Controller also gets the information from distribution grid about the daily forecast and if it changes controller updates its database. Once an electric vehicle attached with the charger, controller immediately calculates the maximum charging time for desired SoC. If the $SoC_{desired}$ is not attained in that charging time then vehicle will not take part in the scheduling and it charges fully without pool till desired SoC for customer's satisfaction. A block diagram for it is shown in figure 7.1 which depicts the flow.

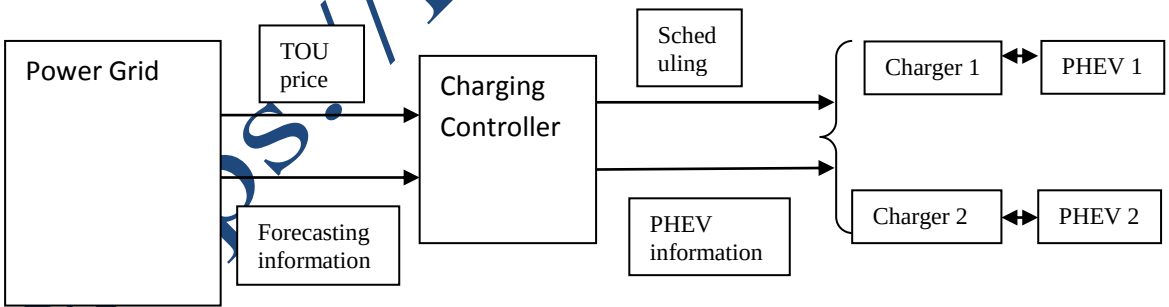


Figure 7.1: Proposed Pipe line of the electric vehicle charging

Our proposed work will revolve around the charging controllers by ALO optimization.

7.3 Objective function

This is a multi-objective problem and two objective functions are combined to convert the multi-objective problem into a single objective. These are dependent upon the cost of charging and load variance at grid.

7.3.1 Objective function to minimize the charging cost

The charging cost of the vehicle varies with heavy load and less load. With the high demand of electricity, the cost will be high and vice versa. But the intelligent charging system should not let his affect the customer's pocket. An objective function in [1] is defined for this as:

$$\min F_1 = \min\left(\sum_{i=1}^{n}\sum_{k=1}^{m} C_k \cdot p_{i,k} \cdot \Delta t\right) \quad (7.1)$$

where n *is the total number of plug in electic vehcile*

m *is the total number of time steps*

C_k *is the TOU price at time stamp k*

$p_{i,k}$ *is the* $i-th$ *PHEV charging power at each time*

Δt *is the time stamp*

7.3.2 Minimizing the load variance

The load variance minimization is necessary as cost minimization inserts the load peak in the grid which results in power losses. So load variation is defined as:

$$\min F_2 = \frac{1}{m}\sum_{k=t}^{m+t}\left[\left(p_k + \sum_{i=1}^{n} p_{i,k}\right) - \frac{1}{m}\left(p_k + \sum_{i=1}^{n} p_{i,k}\right)\right]^2 \quad (7.2)$$

p_k is the grid forecasted load at time stamp k

t is the time stamp

7.3.3 Multi-objective function converted to single objective

We converted this multi-objective problem into single objective as:

$$\min(0.001XF_1 + 0.0001 \times F_2) \quad (7.3)$$

This objective function is optimized with the help of ant lion optimization considering few constraints.

7.3.4 Optimization constraints for PHEV charging

The charging of electric vehicle abide by the real world constraints. Main constraint is the maximum power allotted to each electric vehicle which should not be exceeded more than

the power allotted to charger, also the consumer must go satisfied with the desired SoC. Charging efficiency must also be considered which is fixed in our case. Following are the constraints of charging:

$$0 \leq p_{i,k} \leq \eta . p_{max}$$

$$SOC_{i,desire} \leq SOC_{i,departure} \leq SOC_{max}$$

$$0 \leq SOC_i(k+1) - SOC_i(k) \leq \Delta SOC_{i,max}$$

7.4 Ant Lion optimization of Objective function

7.4.1 How ALO intelligently control the charging controller

In the intelligent charging of electric vehicle the dependent factor is the power supplied to vehicle for the charging at each time stamp. This power is optimally tuned by ALO optimization considering the constraints of maximum, minimum power, state of charge etc as discussed ion section 7.3. At the charging station, many number of vehicles can be charged and scheduled power to every EV must be provided for fast charging at lowest cost.

The ALO and PHEV charging are two isolated system but their dependency can be depicted as in figure 7.2. These collectively forms a feedback loop. The module of ALO gets the power for all EV at each time stamp for charging in the input and gives the updated objective function as in equation 7.3 in the output which is feedback to ALO which changes the power of charging to decrease the objective function value.

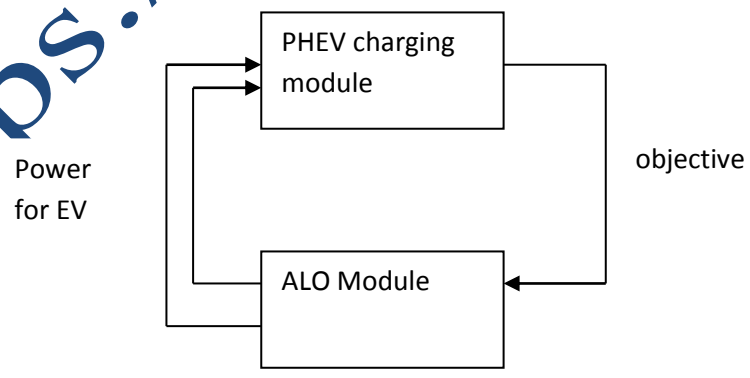


Figure 7.2: Relation between PHEV charging and ALO optimization

The optimization process is the iterative process and in each iteration the each Ant's position will be updated and sent to PHEV module in figure 7.2 which calculates the objective function.

The optimization algorithm is dependent upon the nature of problem and number of tuning variables to be used to achieve the optimal solution. In our case we have considered 50 electric vehicles for a time stamp of 24 hrs. We need to optimize the power transmitted to these 50 electric vehicles for 24 hrs which make the number of tuning variables equal to 24X50=1200. The searching space dimension in ALO is the number of tuning variables. Each set of tuning parameter represents the ant lion's ($M_{antlion}$) and ant's position (M_{ant}) as discussed in chapter 3. The optimization method works to converge at the minima point which is achieved when ant lion consumes the ant and this is depicted in our PHEV charging application when the objective function in equation 7.3 reaches to its minimum state. Every new position should satisfy all constraints of section 7.3.4. For the first iteration the position of all ants and lion ants are initialized randomly within the limits of maximum and minimum power. A new matrix is saved for the objective value in every iteration for all ants and lion ants. Minimum value index in the matrix is the best value so far which is further updated by ALO. After few iterations the saturation value of objective function is reached and no more minimum value is obtained. This is the termination criteria of ALO optimization algorithm which indicates that a best power set of all electric vehicle in each time stamp is achieved. Earlier is this convergence better is the optimization. A chart for complete optimization process is shown in figure 7.3.

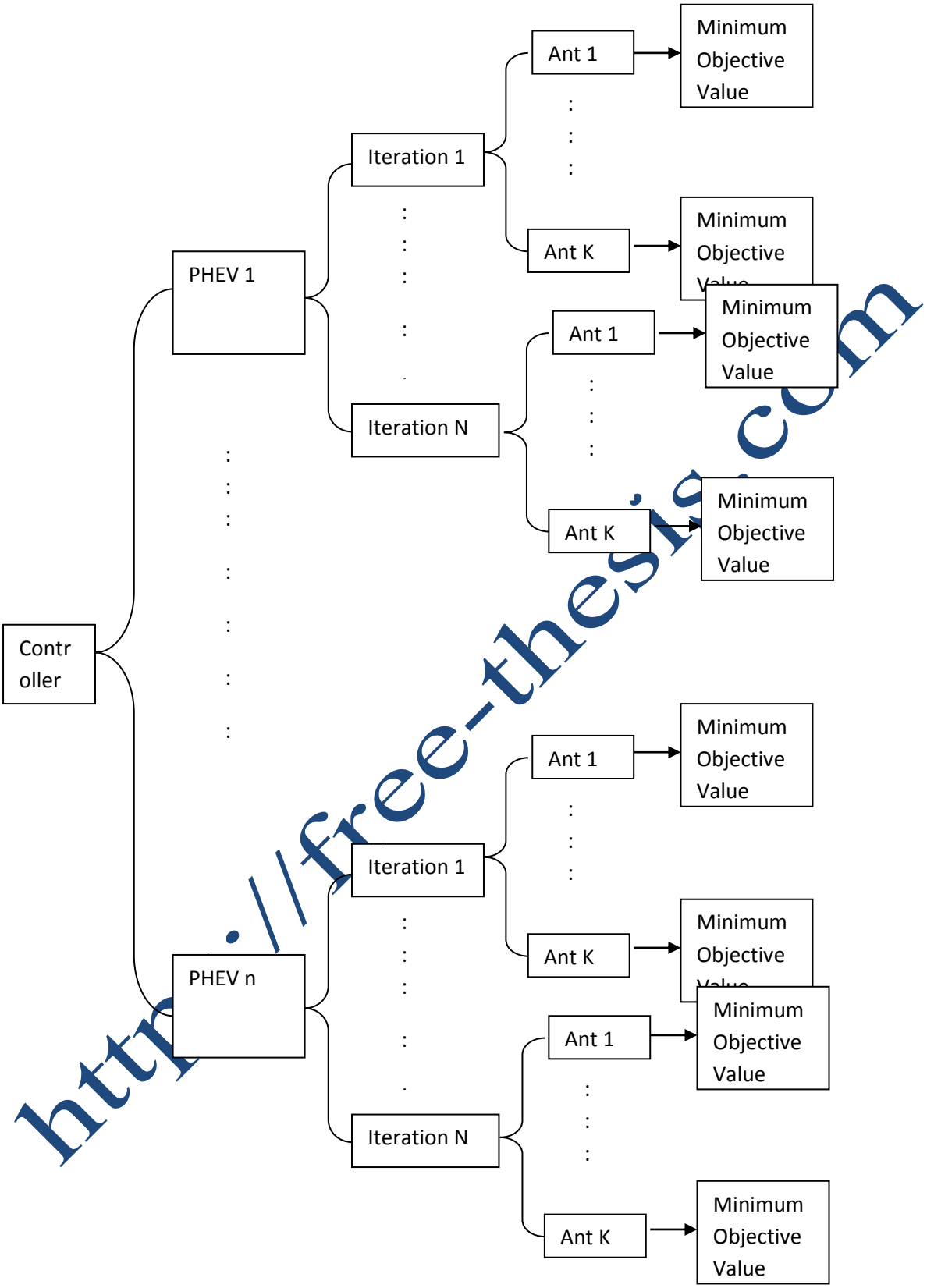


Figure 7.3: Pipeline for whole process for the interaction of PHEV module with ALO optimisation module

The equivalent terminology of ALO with the PHEV charging schedule application is shown in table 7.1.

Table 7.1: Terminology for PHEV in ALO

ALO terminology	Equivalent terminology in PHEV charging scheduling
Searching space of ants and ant lions	Minimum and maximum constraint of charging power for each vehicle
Dimension of searching space	Number of tuning variables which is equal to $\boldsymbol{number\ of\ EV \times time\ stamps\ for\ each\ EV}$
Fitness value	Objective function value

7.4.2. Steps for controlling of EV charging

Step1. consider the input attributes like maximum/ minimum power, maximum /minimum SoC, number of electric vehicles etc.

Step2. initialise the ant's and ant lion's positions randomly which is the power applied to each vehicle for the charging.

Step3. this power applied is sent to objective function module which considers the electric load in summer and winter separately with corresponding TOU price.

Step4. The parking time of each EV is randomly decided between 1-8 hrs.

Step5. Based on the charging rate, charging of each EV during that parking time is calculated by the charging power received.

Step6. check the constraints of state of charge (SoC). If it violates the limit then change the violated value with either minimum or maximum boundary which keeps them at the boundary as penalty.

Step7. calculate the cost of charging for the given charging power based on per unit price from the grid.

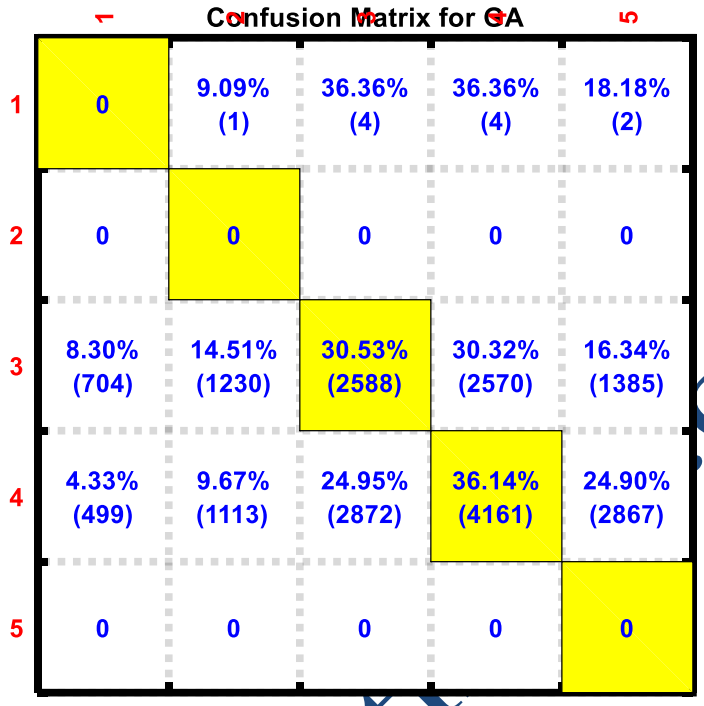
Step8. calculate the load variation in due to the all EV charging

Step9. combine both values in step 7 and step 8 using equation 7.3 and pass these values to ALO module.

Step10. ALO update the previous charging power to minimize the objective function value in step 9 using equations from 3.2 to 3.8.

Step11. The minimum values out of all iterations are considered as the best power set supplied to the EV which occurs the minimum cost on customer with least load variation in supply and with EV charged till desired SoC.

Thesis -8
Improved Recommendation Engine



Abstract

In the recent years, the Web has undergone a tremendous growth regarding both content and users. This has lead to an information overload problem in which people are finding it increasingly difficult to locate the right information at the right time. Recommender systems have been developed to address this problem, by guiding users through the big ocean of information. Until now, recommender systems have been extensively used within e-commerce and communities where items like movies, music and articles are recommended. More recently, recommender systems have been deployed in online music players, recommending music that the users probably will like.

This thesis presents the design and implementation of recommendation system for movies. our work is based on the fact that collaborative recommendation system is suitable for this application as it considers user's behaviour as well as movie data. The attributes considered for users are his age, demographic information and gender whereas the movie data has its rating by different users, its genre and time stamp information. The rating of movies is given from 1-5 by the users, so we used multiclass Support vector machine (SVM) as machine learning model. It is noticed from previous works that it is not necessary that all attributes in the data contributes towards accuracy. Optimal attributes must be selected, For example the

time stamp in our case don't carry any useful information so can be dropped. To choose more optimal attributes we chose Gravitational Search Algorithm (GSA) as its a global optimisation algorithm and converge early with maximum accuracy as in our case. It gives an optimal features set out of 8 in total. We compared the results with PSO and Genetic Algorithm to prove GSA selects the best set of attributes and all selected features are trained and tested by SVM to give high rating recommendation of movies.

Proposed Work

8.1 Algorithm

Collaborative filtering (CF) for recommendation is widely used for music recommendation, movie recommendation, news recommendation etc. In this work we used it for movie recommendation engine. It is purely based on movies rating whether explicit or implicit by the user. A set of user's behaviour data is collected and any machine learning model is trained to learn user's behaviour. Using that trained model, movies are recommended to any user. In our case Support Vector Machine (SVM) which has advantage of an overall optimum and a strong generalization ability over other ML algorithms. The setback point here is the data attributes to create a hurdle in ML modelling to recommend effectively as not all attributes contribute into recommendation everytime. Some of them can reduce the accuracy too. So we must choose the attributes which contribute the most and also reduction of dataset will train the model faster. For this purpose we used optimisation to choose the optimum feature set of input attributes. Previously this is done by using particle swarm optimisation (PSO) but this optimisation method has the issue of premature convergence since it is a local search heuristic algorithm. The optimum solution searched by it is not necessarily best due to skipping of some minima/maxima points. So we replaced this optimisation with the Gravitational Search Algorithm (GSA). GSA overcomes the drawback of PSO and based on motion of celestial bodies called agents as discussed. Since recommendation engine's attribute selection and GSA, both are at different poles and no similarity or relation exists between them. out task is to bring them on a same pole and optimise the attributes selection for better accuracy using GSA optimisation. The equilibrium condition between them can be shown in figure 8.1.

The data used here is movie lens dataset which has a total of 8 attributes to be optimally selected. Dataset description is detailed in next section of this chapter. Out of these attributes, GSA will choose all those only which contributes more in the accuracy improvement in recommendation. The number of agents in GSA acts as the number of

options available for attributes selection at an iteration and their position is the column choice in the data. The position of agents will be either 0 or 1. '1' stands for selection of that attribute and '0' stands for that attribute is not selected in that iteration. For an iteration,10 different set of attributes choice is chosen which means 10 agents are used for GSA to optimise attributes choice.

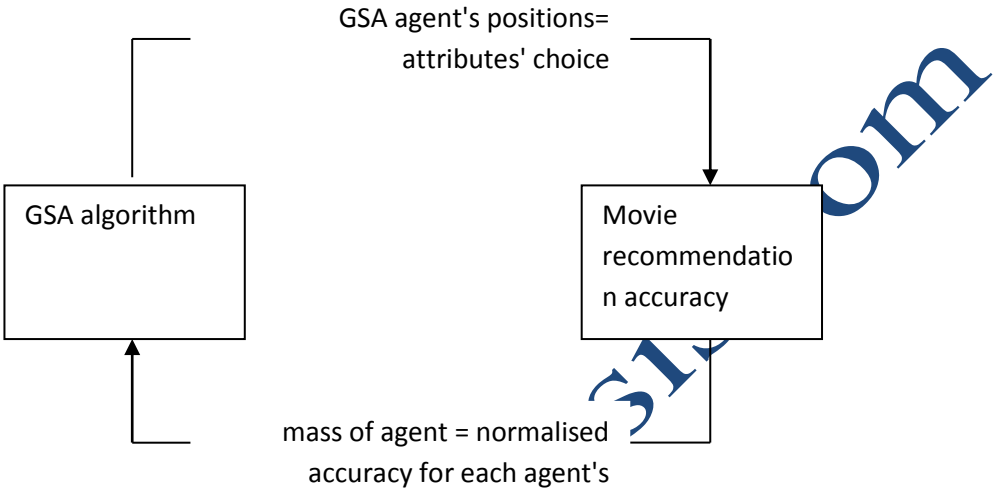


Figure 8.1: Representation of equilibrium of GSA optimisation and Movie recommendation engine's attribute selection

The whole data is first divided into testing and training datasets. 80% of data is used for training and rest is used for testing. For the first iteration, each position of agent is chosen randomly which is for number of attributes set for an iteration. The selected attributes for them are used to train the multiclass SVM and tested for the query data. This accuracy is noted down in a matrix for each GSA agent. An example matrix is shown in table 8.1.

Table 8.1: Attributes selection for 1st iteration along with corresponding accuracy

Agent	Attribute1	Attribute2	Attribute3	Attribute4	Attribute5	Attribute6	Attribute7	Attribute8	Accuracy
1	0	1	1	1	0	1	1	1	0.868990731678713
2	1	0	1	1	1	1	0	1	0.883867770312039
3	0	0	1	0	0	1	1	1	0.819391091939580

4	1	1	0	0	0	0	0	1	0.877404322372083
5	1	0	1	1	1	1	1	1	0.995419791334546
6	0	0	1	1	1	1	1	1	0.975249713143506
7	0	1	1	0	1	0	0	1	0.984408540860446
8	1	1	1	1	1	1	1	1	0.998033199069239
9	0	1	1	1	1	1	1	1	0.985513595030705
10	1	1	1	1	0	1	1	1	0.807961686464962

out of these accuracy values, the best one is saved. Now the positions of agents are updated or in other words the attributes selection is changed and again accuracy for the new selection is calculated and saved for the best value. This process keeps going on and a n-dimesnional matrix is formed out of which corresponding row to highest accuracy is considered to be the best selection for attributes as in figure 8.2.

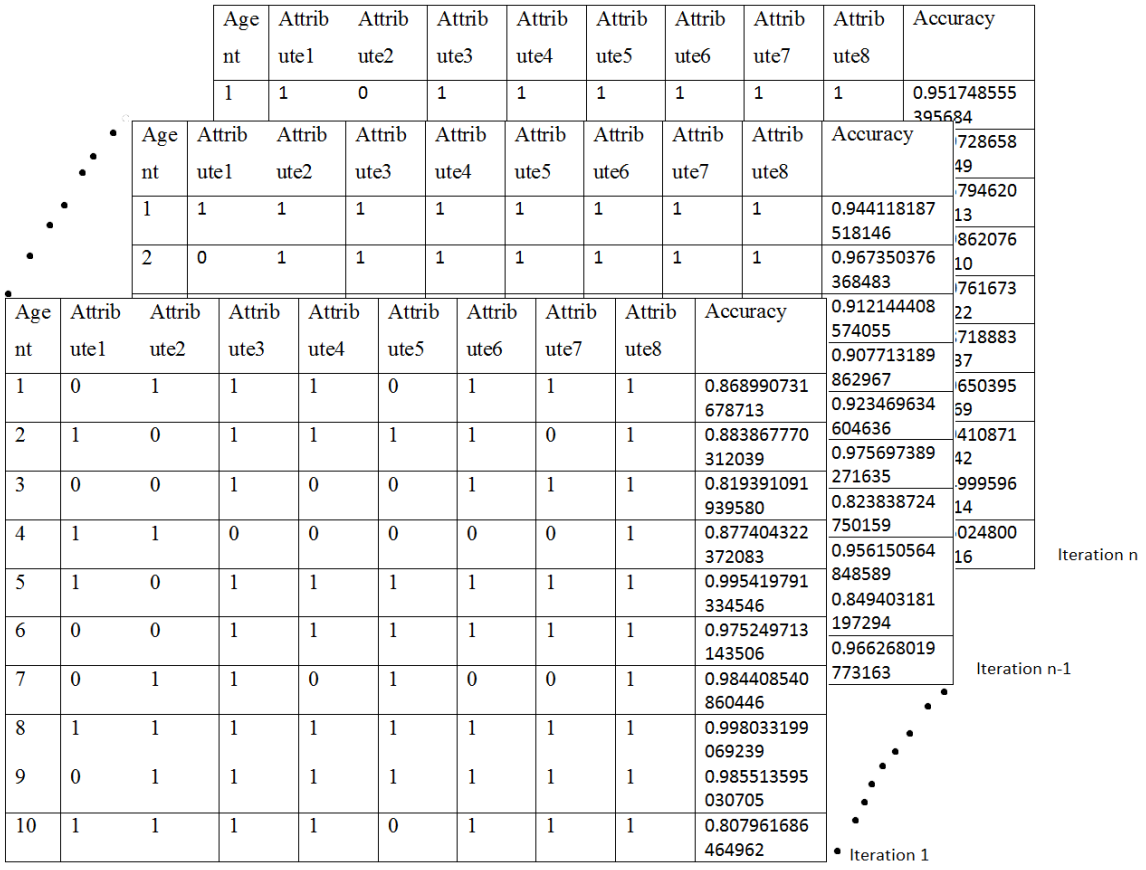


Figure 8.2: Matrix created for accuracy and attributes selection for each agent for n number of iterations

The significance of GSA terminology with recommendation engine is tabulated in table 8.2.

Table 8.2: Significance of GSA terminology in NN tuning

GSA terms	In Recommendation engine's feature selection significance
Agents Position	Selection of attributes' option
Dimension for optimisation/ number of variables to be tuned	Total number of attributes
Update in the position of agents	Change selection of attributes of the data to get higher accuracy

A complete step by step algorithm is explained below.

Step1. Load the movie lens dataset in numeric format and divide that into random 80:20 ratio for training and testing of recommendation engine.

Step2. initialise the GSA parameters like number of iterations, number of agents, initial G0 and alpha.

Step3. randomly initialise the agents new positions which must be either 1 or 0 and will choose the attributes out of 8 in total.

Step4. call the objective function to train the model for selected attributes in training data and test the model for testing data to get the recommendation accuracy.

Step5. to update the random positions of agents, force and mass has to be calculated by using the equations

$$F_{ij}^{d}(t) = \mathrm{G(t)} \frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon} X_j^d(t) - X_i^d(t)),$$

$$m_{it} = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}$$

the respective notations are given in previous chapter

Step6. The new updated position is obtained from the formula

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1)$$

the velocity in this case is calculated by using acceleration which is based on force and mass calculated in previous step.

Step7. For this new updated position or values of weights and biases, objective function is again called and accuracy is saved.

Step8. The attributes' positions for which minimum of accuracy is obtained out of previous two set of values, is further considered for updating.

Step9. This process continues till all iterations are not completed.

Step10. The final maximum accuracy is obtained and attributes selected for them are used as final set of attributes which gives higher accuracy.

A flow chart of work is shown in figure 8.3.

8.2 Dataset Description

We used movielens dataset for academic purpose. This dataset has 1 million sample points and is distributed for open to use. Data is divided into four different files.

u.data -- The full u data set, 100000 ratings by 943 users on 1682 items. Each user has rated at least 20 movies. Users and items are numbered consecutively from 1. The data is randomly ordered. This is a tab separated list of user id | item id | rating | timestamp. The time stamps are unix seconds since 1/1/1970 UTC

u.info -- The number of users, items, and ratings in the u data set.

u.item -- Information about the items (movies); this is a tab separated list of movie id | movie title | release date | video release date | IMDb URL | unknown | Action | Adventure | Animation | Children's | Comedy | Crime | Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller | War | Western | The last 19 fields are the genres, a 1 indicates the movie is of that genre, a 0 indicates it is not; movies can be in several genres at once. The movie ids are the ones used in the u.data data set. An example dataset is shown below.

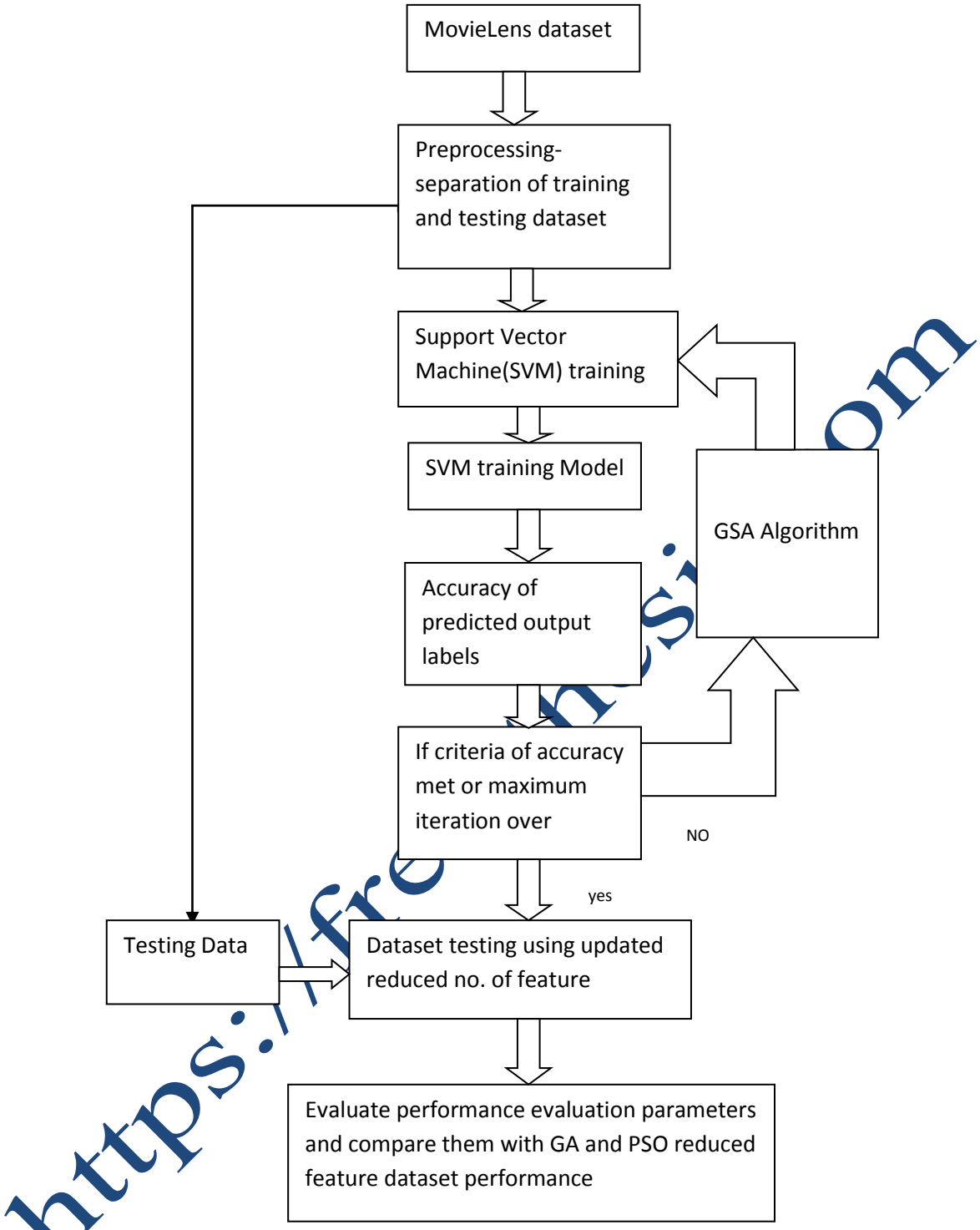


Figure 8.3: Flow chart of proposed method

8.2.1 Dataset Linking

The link between all these files in the dataset can be best explained by the figure 8.4. We collected all these information to make a common file to be used in our experiment. In actual our data combines the movie information and user's behaviour to work in collaborative filtering category of recommendation engine. These all information makes a total of 8

attributes. The data has to be converted from qualitative to quantitative as no machine learning model accepts qualitative data, so all genres, genders etc are encoded to their corresponding numeric counterpart and used for experiment. For example if user is a male then it is replaced by 1 otherwise 2. The data is also filtered to remove the undefined values to improve the accuracy and we don't use all 100K sample points of the data rather we took 30K of whole data, chosen randomly and divided that into a set of 20K for training and 10 K for testing. All results in next chapter are based on this filtered dataset.

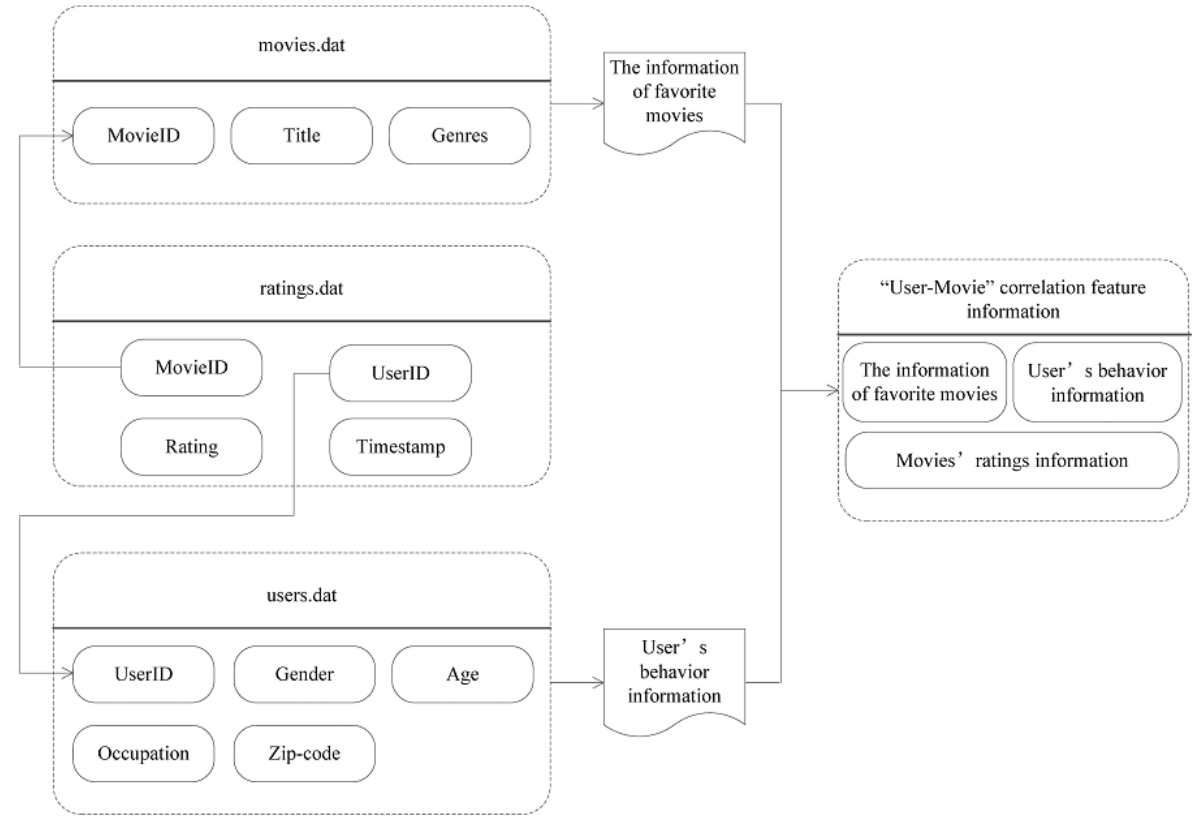
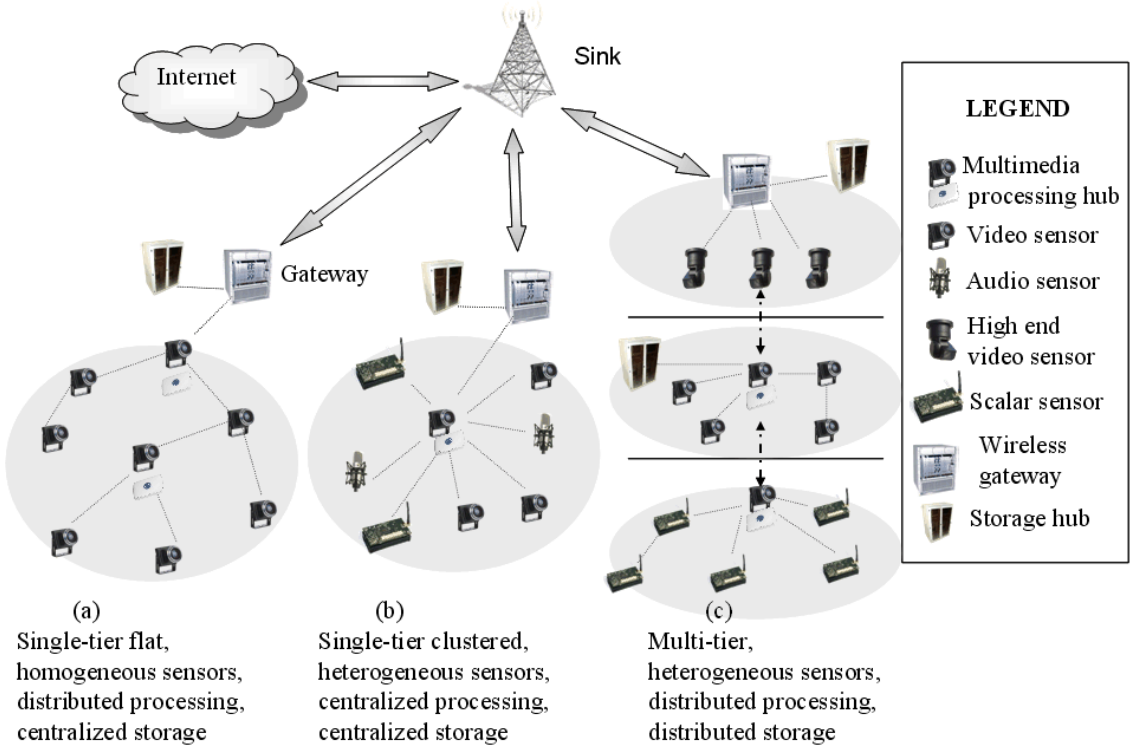


Figure 8.4: Linking of all files in dataset

Thesis-9
PSOGSA Based Improved LEACH Protocol



Abstract

The recent advances in information and communication technologies enable fast development and practical applications of wireless sensor networks (WSNs). The operation of the WSNs including sensing and communication tasks needs to be planned properly in order to achieve the application-specific objectives. The WSNs consist of a number of sensor nodes equipped with microprocessor, wireless transceiver, sensing components and energy source. These sensor nodes operate as autonomous devices to perform different tasks including sensing, communication and data processing. As the members of a network, the sensor nodes are required to cooperate with each other to perceive environmental parameters and transfer data. Commonly, the sensor nodes are left unattended in the environment after being deployed with limited resources such as computational ability, memory and energy. In order to serve for a long lifespan, the resources, especially energy, need to be utilized appropriately. Efficient energy usage is an essential requirement for each individual node as well as for the overall network. A number of energy efficient protocols have been proposed in the literature. The

cluster-based protocol is one classification which has the advantage of scalability, efficient communication and energy savings. This protocol organizes the network into clusters, each cluster has one cluster head (CH) that gathers and aggregates data from all the cluster members, and then send to a base station (BS). Hence, the amount of transferred data is reduced that conserves the energy. This is called LEACH protocol. it has its some set of rules once a cluster head elected, can't be elected again until all nodes in that cluster gets a chance.

We made this protocol more efficient by using optimisation algorithm to choose the cluster head optimally amongst all nodes in the cluster. A hybrid optimisation Particle Swarm Optimisation (PSO) and Gravitational Search Algorithm (GSA) is used in which advantages of both individual algorithms are clubbed together to give rise to a more fast converging and accurate algorithm. We optimised the cluster head based on energy and distance from other neighbouring nodes by this PSOGSA algorithm and achieves high residual energy than PSO optimised LEACH and conventional LEACH protocol for the same network parameters.

Proposed Work

In our work, we focused on energy consumption minimisation of sensor nodes in WSN. In WSN, while transmitting the data, a sensor node has to consume energy and same is the case in reception of data. The problem is that sensor node is battery powered device and it has limited power source. So it is required to increase the life span of a sensor node, which is possible by reduction of energy consumption. To reduce the energy consumption a hierarchy is followed. Sensor nodes select a head with maximum energy residual in each round out of nearby nodes. This selected sensor node transmits the data to sink node. This transmission is done in TDMA fashion. This kind of process is followed in LEACH (Low-energy adaptive clustering hierarchy) protocol. In it two tier topology is followed in which clusters of nodes are formed with cluster head in the first tier. These cluster heads are responsible to pass the sensor information to the sink node. Clusters are formed on the basis of distance amongst nodes. Nodes with minimum distance are considered in a single cluster and these nodes are farther from nodes in other cluster. In second tier communication cluster head communicates with sink node, again in TDMA fashion. The architecture of WSN is shown in figure 4.1.

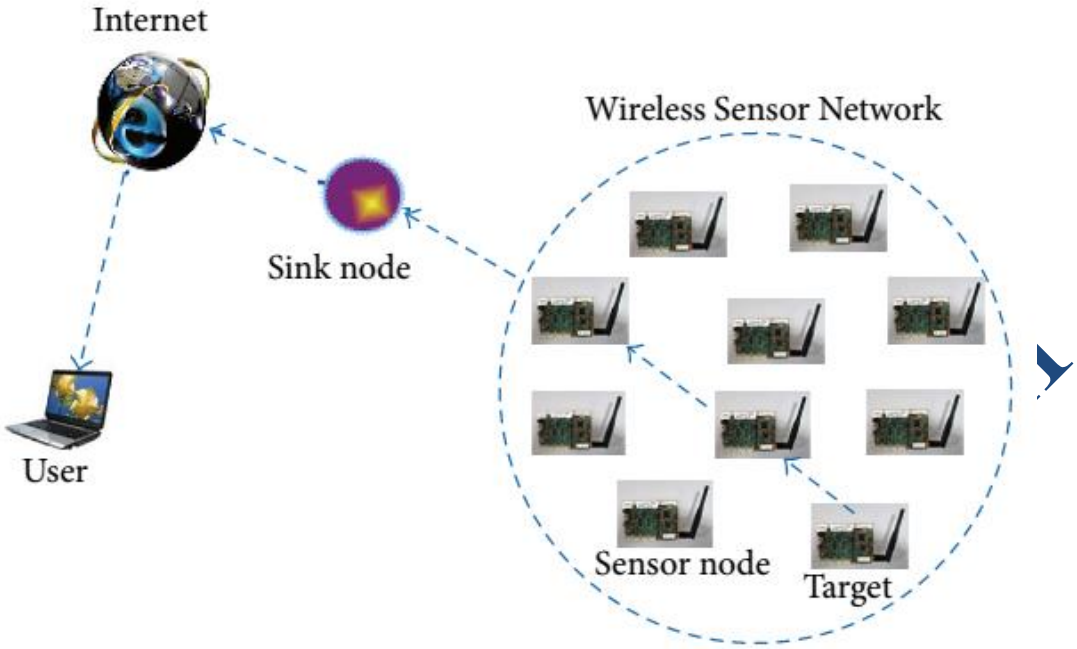


Figure 9.1: Architecture of WSN

Every time in the cluster a cluster head is selected with maximum residual energy and another cluster head is chosen for next round with same criteria. Care has to be taken that no node is repeated as cluster head till all nodes are elected once. The energy residual of each node is considered as the initial energy for that for the next round and after all round completion, dead nodes, alive nodes, residual energy is checked and analysed. The cluster head selection in LEACH protocol is done by using the formula:

$$T(n) = \begin{cases} \frac{P}{1-P[r*mod(\frac{1}{P})]} & if\ n \in G \\ 0 & Otherwsie \end{cases} \qquad (9.1)$$

where n is the given node, p is the probability, r is the current round, G is the set of nodes that were not cluster heads in the previous round, T(n) is the Threshold.

The energy model followed by LEACH protocol is dependent upon two channel model: one is free space (d^2) for the purpose of one hope and other is for multihop path with multipath fading (d^4). Thus the energy consumption for *l* bit packets over distance *d* is calculated as:

$$E_{TX}(l,d) = \begin{cases} lE_{elec} + l\varepsilon_{fs}d^2 & d < d_0 \\ lE_{elec} + l\varepsilon_{mp}d^4 & d > d_0 \end{cases} \qquad (9.2)$$

where ε_{fs} = free space energy loss

ε_{mp}= multipath fading loss

d = distance between source and destination node

d_0 = crossover distance = $\sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}}$

The energy spent in reception of this message is

$$E_{RX}(l) = lE_{elec} \quad (9.3)$$

This transmission and reception of power is designed in physical and MAC layer.

As is clear from equation 9.2, the energy consumption is dependent upon the square of distance for the single hop communication, so it's the inter cluster distance which is the affecting variable. If this distance is minimised then energy residual in each node can be increased. Previously researcher used Particle swarm optimisation (PSO) to optimally select the cluster head based on minimum distance which proved efficient than conventional LEACH protocol, but PSO is a local optimisation algorithm and has premature convergence tendency. So convergence point reached by PSO is not the absolute point, so we updated the PSO algorithm in our work. We mix it with another global optimisation algorithm which has good convergence accuracy. Gravitational Search Algorithm (GSA) is used along with PSO in this work to optimally select the cluster head position to have maximum residual energy.

9.1 Fitness Function

Every optimisation algorithm needs an objective function which must satisfy and match all constraints and requirements of the network. In our case the energy is the dependent variable which are dependent upon the node distance, so objective function must include the relation between these two variables. The fitness function used for our work is as:

$$ObjVal = \alpha_1 * \frac{\sum_{i=0}^{n} d(current\ node, member\ i)}{n} + \alpha_2 * \frac{\sum_{i=0}^{n} E(member\ i)}{E(current\ node)} + (1 - \alpha_1 + \alpha_2) * \frac{1}{no\ of\ members\ covered\ by\ current\ node} \quad (9.4)$$

here α_1 and α_2 are tradeoff factors and decide the weightage of distance and energy variable. In our work we chose both as 0.4.

9.2. PSOGSA optimisation of LEACH protocol

In our proposed scheme PSOGSA optimized technique is used which is hybrid of PSO (Particle Swarm optimisation) and GSA (Gravitational Search Algorithm) optimisation algorithm which requires an objective function to minimize. GSA is a global search optimization technique which has very less probability of premature termination and when it is hybridized with PSO, convergence speed is also increased along with more intensive search of global minima point. It performs well in case of multi objective function or multi constraint function. GSA algorithm is based on the motion of celestial bodies in the universe and PSO is based on behavior of swarms as discussed in previous chapter. The counterpart of agents and particles in GSA and PSO respectively in our work is the tuning variables. The position of a single agent or single particle is defined by the number of tuning variables. Co ordinates used to define the position in a searching space are equal to the variables to be tuned. For example in our case, we need to tune the values of frequency channels, so the tuning variable will be frequency of each channel and these variables number will be equal to number of channels.

The hybrid GSA+PSO algorithms work in the manner that PSO becomes alive to update the direction in the GSA algorithm. The update in position requires knowledge of step size and direction and in GSA the direction of movement of bacteria is random. Due to it, it takes time to converge of to reach at an optimal solution. This random direction is controlled by the PSO algorithm in our work. This make the convergence faster with each minima point checked. Initialization of direction is random but later on once for every particle, fitness function is evaluated, the direction is controlled by PSO. Output of fitness function becomes the local best for the PSO and based on that local best position is calculated, which is updated by the velocity update equation in PSO. This velocity is added into the old position of particle which was local best position obtained from GSA, to get the new position. This new position is updated as direction of bacteria. This way PSO tunes the direction of bacteria in gives them a direction to look for the food.

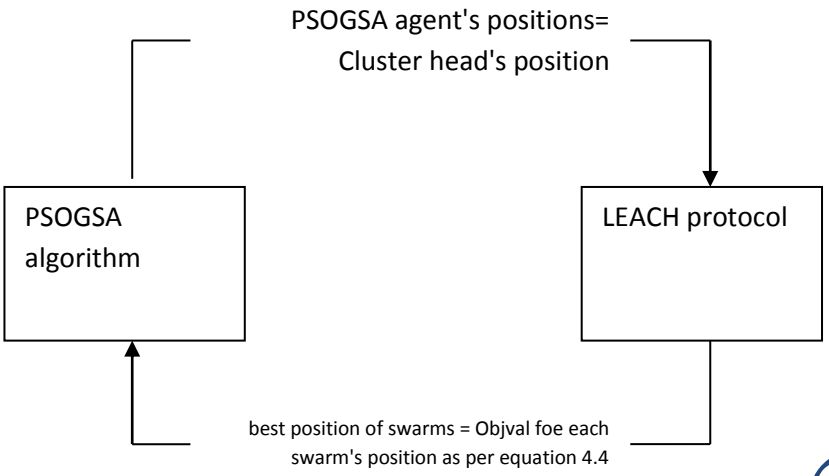


Figure 9.2: Representation of equilibrium of PSOGSA optimisation and LEACH protocol's cluster head selection

Steps of proposed algorithms are described as:

Step1. Initialize all initial parameters for the LEACH protocol like number of nodes, their position, channel bandwidth, frequency etc. to model it. All these network values are indicated in table 5.3.

Step2. Place the nodes randomly in geographical region of 100*100

Step3. divide all nodes into clusters. We take 5% of total nodes as clusters to be formed. So for 100 nodes, 5 clusters will be created using k-means algorithm.

PSOGSA Initialization

Step4. Initialize the random positions of particles in PSO.

Step5. Consider the searching space dimension as number of available channels

Step6. Initialize the weighting parameters of PSO as 1.2 and 0.5.

Step7. Compare the fitness value of each particle with the previous best position of bacteria. If fitness function value is less for this new position than previous position then it will be assigned as new.

Step8. The present bets position is termed as current position of particle for PSO and output of fitness function is Jlocal for the PSO.

GSA Starts here:

Step9. The current position selected in previous step is used to get the mass for each agent as per GSA algorithm. The minimum value of fitness function is selected as best and maximum as worst position and using the formulas, mass of each agent can be calculated as:

$$m_i(t) = \frac{fit(t) - worst(t)}{best(t) - worst(t)}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}$$

Step10. Gravitational force is calculated as:

$$F_{ij}^d(t) = G(t).\left(M_{pi}(t) \times \frac{M_{ai}(t)}{R_{ij}(t)} + \varepsilon\right)\left(x_j^d(t) - x_i^d(t)\right)$$

The formula is described in section 3.1.

Step11. This new velocity is the direction of particle in PSO is updated as

$$new\ velocity = old\ velocity + c1 * acceleration + c2(gbest - current\ position)$$

Here gbest is the global best position of particles in PSO and acceleration is calculated in GSA as $a_i^d(t) = F_i^d(t)/M_{ii}(t)$.

GSA ends here

Step12. The final position of agents which is achieved either by matching the condition of power reduction or by reaching the maximum iterations.

Step13. Final positions of agents thus settled are considered as the final cluster head's position in each cluster which has maximum RE amongst all.

Step14. these steps will keep on repeating for every round considering the RE energy in previous round as the initial energy in current round.

Significance of PSO-GSA algorithm in LEACH protocol

The main task of our work is to allocate the cluster heads in each cluster which consumes very less energy in data transmission and reception. This work is done by hybrid PSOGSA algorithm which is a bio inspired algorithm and fired by the change in agents positions each time. Every change in position of agents is the change in cluster head's co-ordinates to achieve the minimum energy loss component. Table 9.1 shows the significance of bio terms in PSOGSA with our proposed technical terms. This table will correlate proposed optimisation with channel allocation task.

Table 9.1: Related terms of PSOGSA algorithm with channel allocation task

Bio terms of PSOGSA algorithm	Corresponding meaning in channel allocation
Position of agents/swarms	Position of cluster heads
Number of dimension of searching space	Total number of positions to be tuned for cluster head
Update in positions of patcicles/agents	Change in the cluster head's positions

The flow chart of proposed work is shown in figure 9.3.

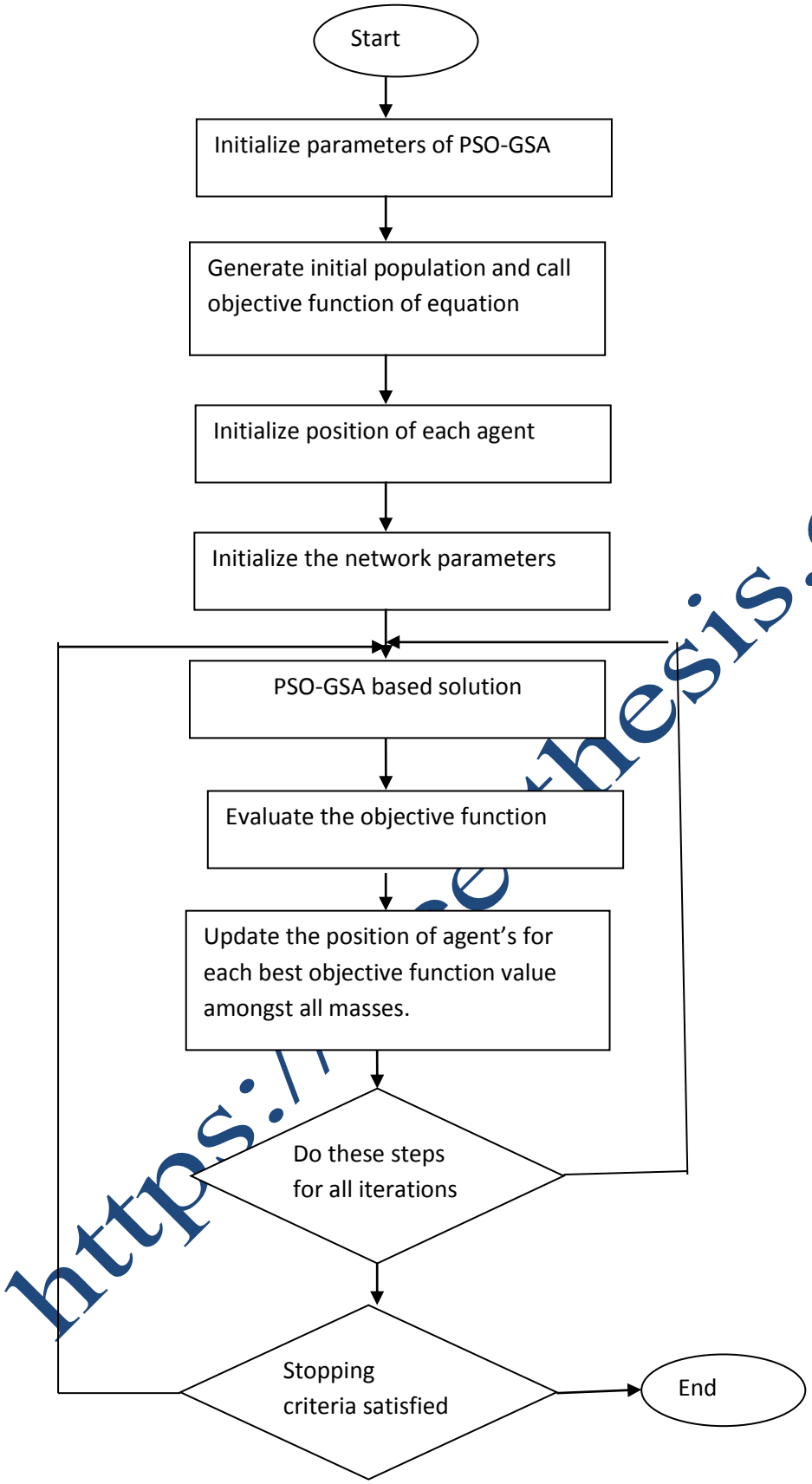
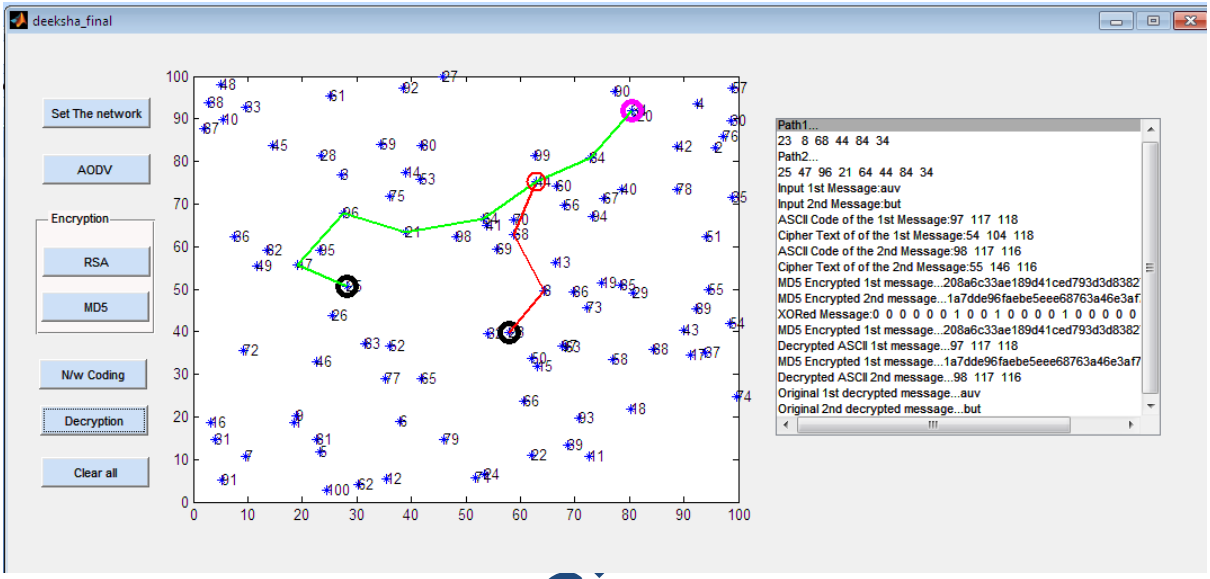


Figure 9.3 Flowchart of Proposed Work

Thesis- 10
Cascaded Encryption algorithm in WSN for data Dissemination



Abstract

The popularity of Wireless Sensor Networks (WSN) has increased rapidly and tremendously due to the vast potential of the sensor networks to connect the physical world with the virtual world. Since sensor devices rely on battery power and node energy and may be placed in hostile environments, so replacing them becomes a difficult task. Thus, improving the energy of these networks i.e. network lifetime becomes important.

Proposed Work

In our work we have emphasized over the encryption of data in network coding. The environment for the NW has been created using MATLAB tool. We have applied here two level encryption algorithms at the sender nodes' level. At the first level MD5 encryption algorithm stand still to protect the data and after that RSA encryption decryption algorithm is used. A block diagram of the whole process is shown in figure 10.1. the ciphered message by RSA is in the form of sequence if digits which are converted into hash values by MD5 algorithm to provide second level security.

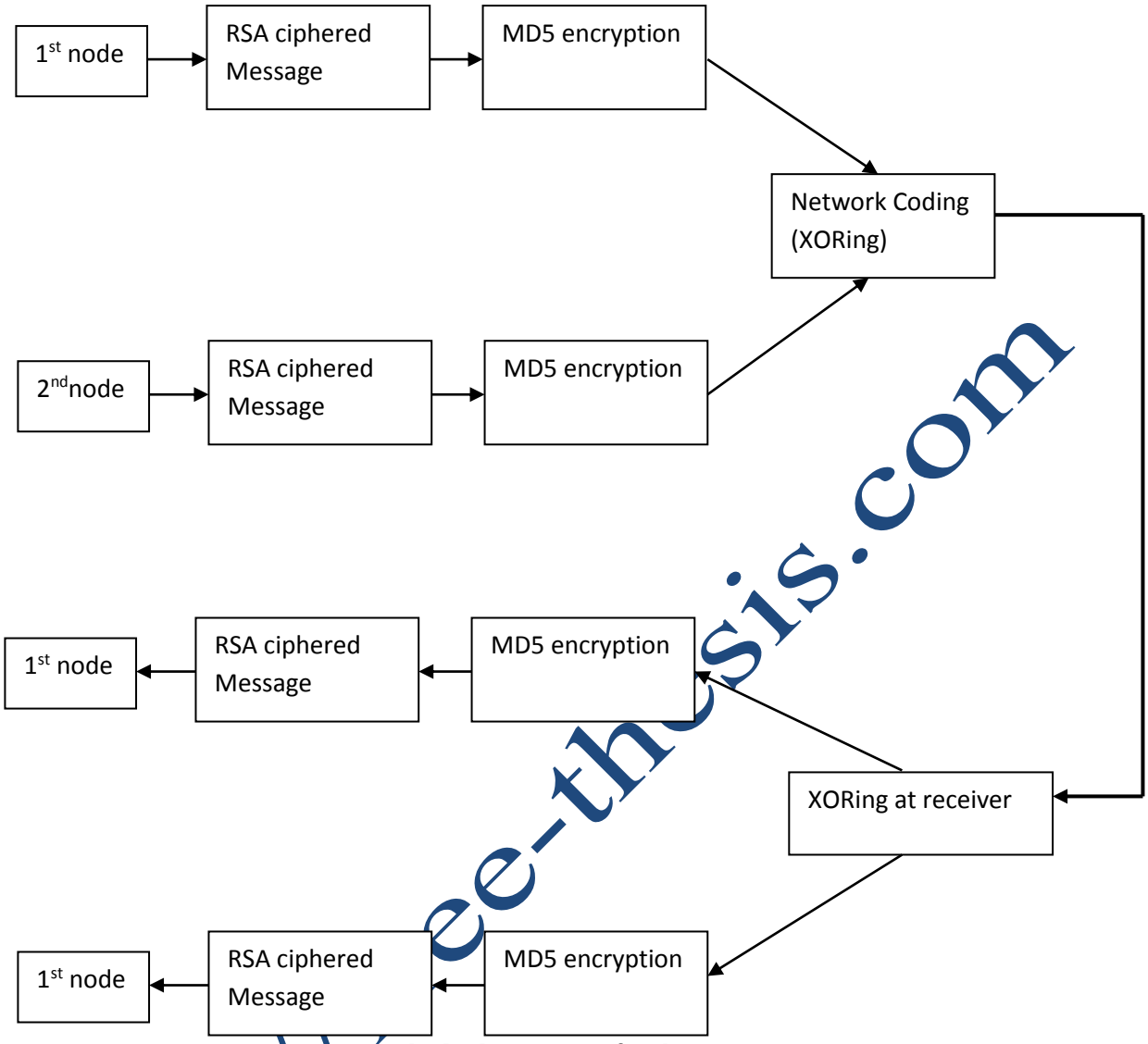


Figure 10.1: proposed Block Diagram for the encryption process

The proposed work is divided into two modules. In the first part encryption of data is discussed and second part is about network coding.

Module 1: Encryption of Data

As discussed above RSA and MD5 algorithms are the backbone of security in our work and MD5's decryption is not possible so the key message generated by RSA is secure. At the receiver end RSA deciphered message is again MD5 decrypted to check whether hash code generated by deciphered message is same as the ciphered' has code at the transmitting end. If both hash codes are matched then we believe that no interruption with message is one and it travelled through transmission channel safely. Both encryption algorithms are described below.

RSA algorithm

The RSA cryptosystem is a public-key cryptosystem that offers both encryption and digital signatures (authentication). The RSA algorithm works as follows: take two large primes, p and q, and compute their product n = pq; n is called the modulus. Choose a number, e, less than n and relatively prime to (p-1)(q-1), which means e and (p-1)(q-1) have no common factors except 1. Find another number d such that (ed - 1) is divisible by (p-1)(q-1). The values e and d are called the public and private exponents, respectively. The public key is the pair (n, e); the private key is (n, d). The factors p and q may be destroyed or kept with the private key.

It is currently difficult to obtain the private key d from the public key (n, e). However if one could factor n into p and q, then one could obtain the private key d. Thus the security of the RSA system is based on the assumption that factoring is difficult.

Here is how the RSA system can be used for encryption and digital signatures.

Encryption

Suppose Alice wants to send a message m to Bob. Alice creates the ciphertext c by exponentiating: $c = m^e \bmod n$, where e and n are Bob's public key. She sends c to Bob. To decrypt, Bob also exponentiates: $m = c^d \bmod n$; the relationship between e and d ensures that Bob correctly recovers m. Since only Bob knows d, only Bob can decrypt this message.

Digital Signature

Suppose Alice wants to send a message m to Bob in such a way that Bob is assured the message is both authentic, has not been tampered with, and from Alice. Alice creates a digital signature s by exponentiating: $s = m^d \bmod n$, where d and n are Alice's private key. She sends m and s to Bob. To verify the signature, Bob exponentiates and checks that the message m is recovered: $m = s^e \bmod n$, where e and n are Alice's public key.

Thus encryption and authentication take place without any sharing of private keys: each person uses only another's public key or their own private key. Anyone can send an encrypted message or verify a signed message, but only someone in possession of the correct private key can decrypt or sign a message.

RSA Algorithm

The RSA algorithm involves three steps: key generation, encryption and decryption.

Key generation

RSA involves a *public key* and a *private key*. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

- Choose two distinct prime numbers p and q. For security purposes, the integers p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primarily test.
- Compute $n = pq$.

  - n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
- Compute $\varphi(n) = \varphi(p)\varphi(q) = (p - 1)(q - 1) = n - (p + q - 1)$, where φ is Euler's function.
- Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$; i.e., e and $\varphi(n)$ are coprime. e is released as the public key exponent having a short bit-length and small Hamming weight results in more efficient encryption – most commonly $2^{16} + 1 = 65{,}537$. However, much smaller values of e (such as 3) have been shown to be less secure in some settings.
- Determine d as $d \equiv e^{-1} \pmod{\varphi(n)}$; i.e., d is the multiplicative inverse of e (modulo $\varphi(n)$).
- This is more clearly stated as: solve for d given $d \cdot e \equiv 1 \pmod{\varphi(n)}$
- This is often computed using the extended Euclidean algorithm. Using the pseudo code in the *Modular integers* section, inputs a and n correspond to e and $\varphi(n)$, respectively.
- d is kept as the private key exponent.

The *public key* consists of the modulus n and the public (or encryption) exponent e. The *private key* consists of the modulus n and the private (or decryption) exponent d, which must be kept secret. p, q, and $\varphi(n)$ must also be kept secret because they can be used to calculate d.

Encryption

Alice transmits her public key (n, e) to Bob and keeps the private key secret. Bob then wishes to send message M to Alice. He first turns M into an integer m, such that $0 \leq m < n$ by using

an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext *c* corresponding to

$$c = m^e (mod\ n)$$

This can be done quickly using the method of exponentiation by squaring. Bob then transmits *c* to Alice. Note that at least nine values of *m* will yield a ciphertext *c* equal to *m*, but this is very unlikely to occur in practice.

Decryption

Alice can recover *m* from *c* by using her private key exponent *d* via computing

$$m = c^d (mod\ n)$$

Given *m*, she can recover the original message *M* by reversing the padding scheme.

MD5 Algorithm

The major part of MD5 algorithm described here has been picked form the source http://www.ietf.org/rfc/rfc1321.txt. MD5 algorithm consists of 5 steps:

Step 1. Appending Padding Bits. The original message is "padded" (extended) so that its length (in bits) is congruent to 448, modulo 512. The padding rules are:

- The original message is always padded with one bit "1" first.
- Then zero or more bits "0" are padded to bring the length of the message up to 64 bits fewer than a multiple of 512.

Step 2. Appending Length. 64 bits are appended to the end of the padded message to indicate the length of the original message in bytes. The rules of appending length are:

- The length of the original message in bytes is converted to its binary format of 64 bits. If overflow happens, only the low-order 64 bits are used.
- Break the 64-bit length into 2 words (32 bits each).
- The low-order word is appended first and followed by the high-order word.

Step 3. Initializing MD Buffer. MD5 algorithm requires a 128-bit buffer with a specific initial value. The rules of initializing buffer are:

- The buffer is divided into 4 words (32 bits each), named as A, B, C, and D.
- Word A is initialized to: 0x67452301.
- Word B is initialized to: 0xEFCDAB89.
- Word C is initialized to: 0x98BADCFE.
- Word D is initialized to: 0x10325476.

Step 4. Processing Message in 512-bit Blocks. This is the main step of MD 5 algorithm, which loops through the padded and appended message in blocks of 512 bits each. For each input block, 4 rounds of operations are performed with 16 operations in each round.

Step 5. Output. The contents in buffer words A, B, C, D are returned in sequence with low-order byte first.

Network Coding

As discussed in previous chapter network coding increase the throughput by XORing the data into a single message and thus reducing the waiting time in case of multicasting network. This network coding is implemented in our work and comparison is done in terms of throughput, time consumed and waiting time. For this a network of 10 nodes has been considered. The positions of nodes have been kept fixed as network coding work only in case where a common route between multiple sources to destination can exist. So a fixed network for simulation is better solution to avoid this. These nodes are distributed in a geographical area of 100 square meter and have fixed range of transmission. At the very first node of their common rout the XORing of messages take place. XORing is the only way by which two messages which take part in this process and lose their identity to new one, can be reverted back to their identity just by XORing the XORed output with one of them. Example 4.1 below proves my point.

Example 10.1 consider two binary inputs [1 1 1 0] and [0 1 0 1]. These are XORed as shown in table below. XORing comes with the formula as

$$Y = A\bar{B} + \bar{A}B$$

Where A & B are inputs and Y is the output. Here the 1st term is the result of ORing the input A and complement of other input and vice versa for the second term. Consider for the first single bit of both inputs, the output will be

$$Y = 1 \times 0 + 0 \times 0 = 1$$

A	B	Y
1	0	1
1	1	0
1	0	1
0	1	0

When the output Y is again XORed with any of the input then other input will be the output. Here in this example to get back the input message B, Y is XORed with A as shown in table.

Y	A	XORed
1	1	0
0	1	1
1	1	0
0	0	1

The above table proves our point as the XORed output is same as input B.

Because of this unique property XORing is used for encoding purpose as the destination nodes receive their message by XORing the other on with XORed message. The message length transmitted from all sources must be same and same length of message will be transmitted after operation. This reduces the message packets load carried over bandwidth, thus increase the throughput of the network. For a moment if length of message at two sources is 4 bits then after XORing too both message will be transmitted maintaining the size of packet as four bits. Considering this the network operation with coding can be described as:

Consider the instance $\{G = (V,E),S,R\}$. Let In(v) and Out(v) denote the set of incoming and outgoing edges for vertex v. For simplicity, we assume binary communication, that is, we can send only bits through the unit capacity edges of the network. Similar analysis goes through in the case of larger finite alphabets. The network operates as follows:

(1) A source S of rate ωS produces B packets (messages), each containing $n\omega S$ information bits. It also selects |Out(S)| functions $\{f_i^s\}$,

$$f_i^s: 2^{n\omega s} \rightarrow 2^n$$

which map the $n\,\omega S$ information bits into n coded bits to be transmitted over its outgoing edges.

(2) Similarly, each vertex $v \in V$ selects |Out(v)| functions f_i^v one for each of its outgoing edges. Each function $fv\ i$ has as its inputs |In(v)| packets of length n, and as its output one packet of length n

$$f_i^v: 2^{nIn(v)} \rightarrow 2^n$$

The functions f_i^s and f_i^v are chosen uniformly at random among all possible such mappings. For example, if we were to restrict the network nodes operation to linear processing over F2, each function f_i^v would be defined by a randomly selected binary matrix of dimension $|\text{In}(v)|n \times n$.

(3) The network is clocked: at time k of the clock, for $1 \leq k \leq B$, the source S maps one of its B information packets denoted by m_k to |Out(S)| packets, and sends these packets through its outgoing edges. After time B, the source stops transmission.

(4) For each information packet mk, $1 \leq k \leq B$, each other (non source) vertex $v \in V$ waits until it receives all |In(v)| incoming packets that depend only on packet mk. It then (by the means of functions $\{f_i^v$computes and sends |Out(v)| packets through its outgoing edges, also depending only on packet mk.

(5) For each information packet mk, $1 \leq k \leq B$, each receiver Rj gets |In(Rj)| packets, which it uses to decode the source packet mk, based on its knowledge of the network topology and all the mapping functions that the source and the intermediate nodes employ.

(6) The receivers decode all B source packets during at most $B + |V|$ time slots. Consequently, the received information rate is

$$\frac{n\omega_s B}{n(B + |V|)} \rightarrow \omega_s$$

10. 1 The Main Network Coding Theorem

We now consider a multicast scenario over a network $G = (V,E)$ where h unit rate sources $S1,...,Sh$ located on the same network node S (source) simultaneously transmit information to N receivers $R1,...,RN$. We assume that G is an *acyclic directed* graph with *unit capacity* edges, and that the value of the min-cut between the source node and each of the receivers is h. For the moment, we also assume *zero delay*, meaning that during each time slot all nodes simultaneously receive all their inputs and send their outputs.

What we mean by unit capacity edges: The unit capacity edges assumption models a transmission scenario in which time is slotted, and during each time-slot we can reliably (with no errors) transmit through each edge a symbol from some finite field Fq of size q. Accordingly, each unit rate source Si emits σi, $1 \leq i \leq h$, which is an element of the same field Fq. In practice, this model corresponds to the scenario in which every edge can reliably carry one bit and each source produces one bit per time-unit. Using an alphabet of size, say, $q = 2m$, simply means that we send the information from the sources in packets of m bits, with m time-units being defined as one time-slot. The m bits are treated as one symbol of Fq and processed using operations over Fq by the network nodes. We refer to such transmission mechanisms as transmission schemes over Fq. When we say that a scheme exists "over a large enough finite field Fq," we effectively mean that there exists "a large enough packet length."

10.1.1 Statement of the Theorem

We state the main theorem in network coding as follows:

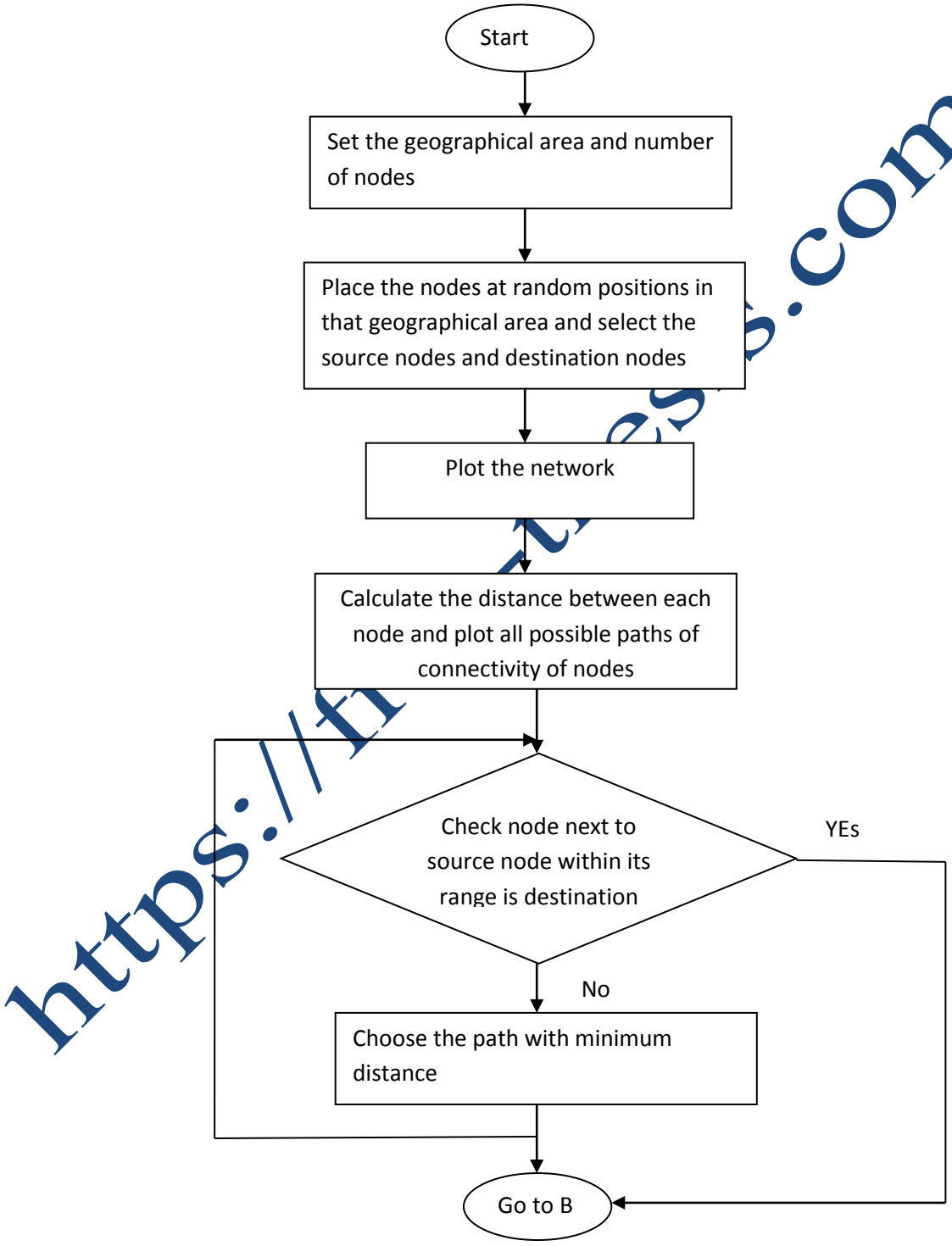
"Consider a directed acyclic graph $G = (V,E)$ with unit capacity edges, h unit rate sources located on the same vertex of the graph and N receivers. Assume that the value of the min-cut to each receiver is h. Then there exists a multicast transmission scheme over a large enough finite field Fq, in which intermediate network nodes linearly combine their incoming

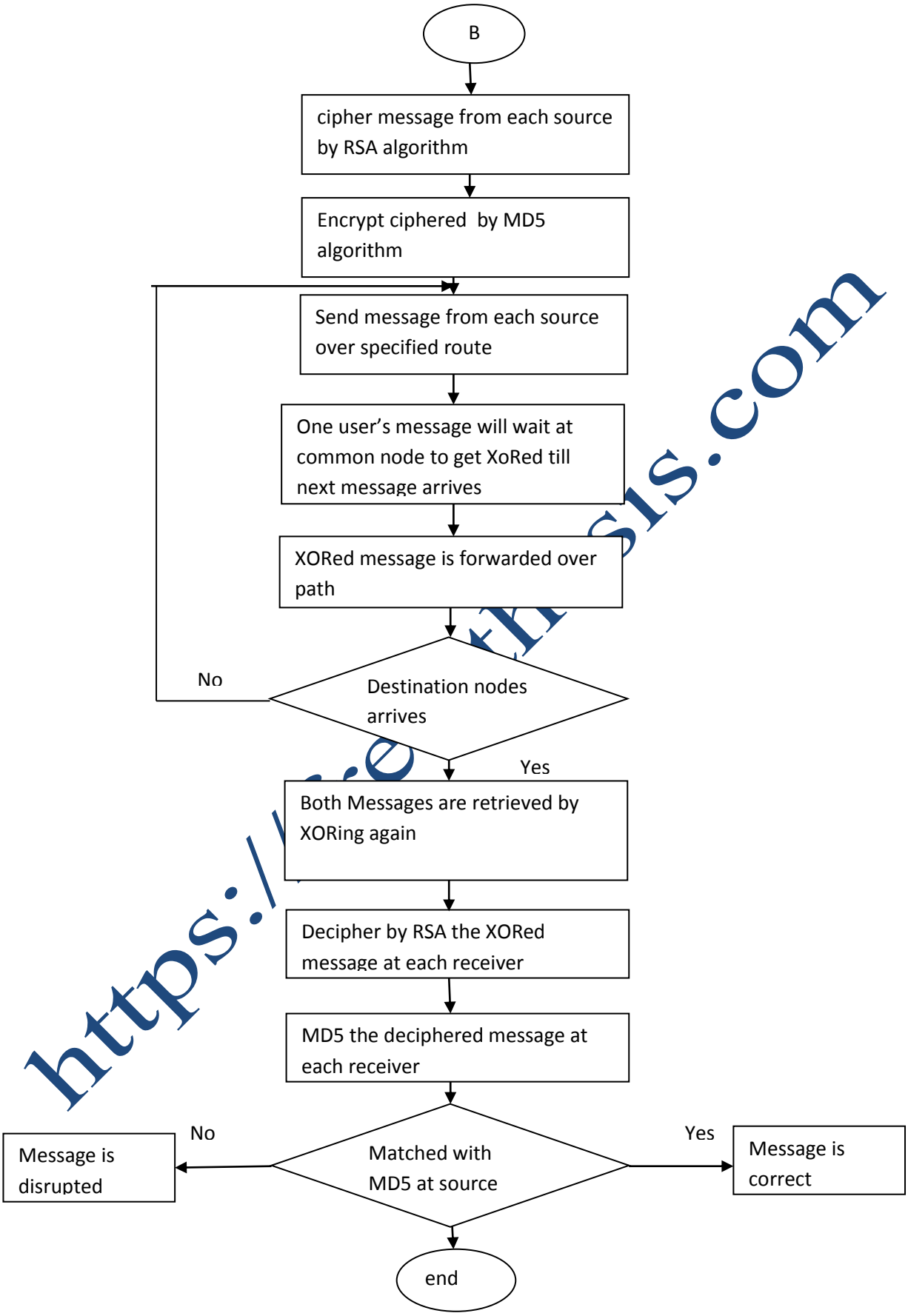
information symbols over F*q*, that delivers the information from the sources simultaneously to each receiver at a rate equal to *h*. “

From the min-cut max-flow theorem which is stated in appendix, we know that there exist exactly *h* edge-disjoint paths between the sources and each of the receivers. Thus, if any of the receivers, say, *Rj* , is using the network by itself, the information from the *h* sources can be routed to *Rj* through a set of *h* edge disjoint paths. When multiple receivers are using the network simultaneously, their sets of paths may overlap. The conventional wisdom says that the receivers will then have to share the network resources, (e.g., share the overlapping edge capacity or share the access to the edge in time), which leads to reduced rates. However, Theorem tells us that, if we allow intermediate network nodes to not only forward but also combine their incoming information flows, then each of the receivers will be getting the information at the same rate as if it had sole access to network resources.

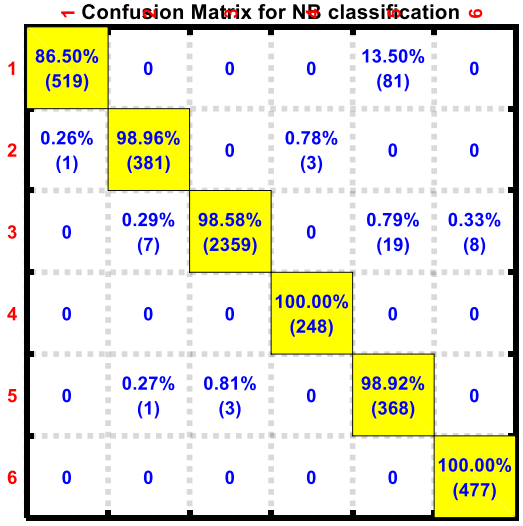
The theorem additionally claims that it is sufficient for intermediate nodes to perform linear operations, namely, additions and multiplications over a finite field F*q*. We will refer to such transmission schemes as *linear network coding*. Thus the theorem establishes the existence of linear network codes over some large enough finite field F*q*. To reduce computational complexity, the field F*q* should be chosen as small as possible.

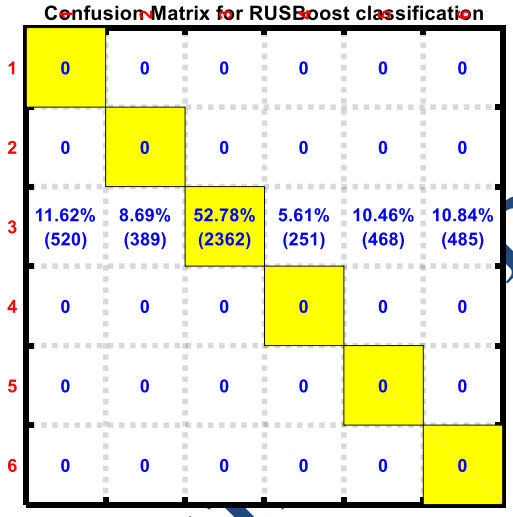
10.2 Flow Chart





Thesis -11
Packet sniffing using Machine learning





Abstract

The objective of this work is to assess the utility of four supervised learning algorithms KNN, SVM, Naïve Bayes and RUSBoost classifying SSH traffic from log files. Pre-processing is applied to the network traffic data to express as traffic flows. It is possible to detect SSH traffic such as local tunnelling, remote tunnelling, SCP, SFTP, Shell and X11 and distinguish from regular traffic types such as DNS, FTP, HTTP, TELNET, Lime (P2P) with high accuracy. In past port no. identifies type of traffic but in present scenario machine learning techniques applied to statistical flow properties, allowing accurate classification. We used NIMS and MAWI dataset for analysis. We find KNN and SVM classifiers has an accuracy of over 95% and error less than 0.02% for SSH data classification. We have implemented our methodology for various combinations of traffic data types and find SVM has overall better performance than rest three classifier which are KNN, Naïve Bayes, RUSBoost. Total of 22 statistical flow parameters are used to identify the data types in NIMS dataset. We built training models using four classifiers and for testing first we used NIMS dataset and secondly, we used MAWI dataset without exact labels and predicting true labels for MAWI datasets. Our predicted labels for MAWI dataset accurately matches the type of data viz SSH and NOT SSH data type.

Proposed Work

This work proposes supervised machine learning algorithm for improved classification of known and unknown network traffic flows. In this work we have taken two datasets NIMS

and MAWI datasets which are used for machine learning algorithm and for testing the machine learning model using both datasets using four main classifiers which are K Nearest Neighbour (KNN) Classifier, Support Vector Machine (SVM) classifier, Naïve Bayes classifier and RUSBoost classifier. The training model made by using NIMS datasets is used to identify unknown labels of MAWI dataset such that it performs cross dataset testing and labelling for unknown network traffic flows.

Overall work can be divided into following steps for better understanding.

We have divided our work in eight sub cases which are

1. Extracting database from web links and make dataset usable for learning algorithm.
2. Analysing dataset, its features and labels.
3. Divide NIMS datasets in different combination which are more probable to exist in network traffic flows.
4. Dividing data into testing and training randomly with ratio 80:20for each combination.
5. Create training model using four classifiers.
6. Test the testing data using training model created by all four classifiers
7. Test the MAWI dataset for unknown labels using trained model.
8. Comparison of output test labels from real labels and discussion.

11.1 Dataset Preparation

Network Information Management and security Group (NIMS) datasets and *MAWI* (*Measurement and Analysis* on the WIDE Internet) dataset are downloaded from following weblinks.

1. For NIMS (http://www.cs.dal.ca/~riyad/DataSets/NIMS/NIMS.arff.zip)
2. For MAWI (http://www.cs.dal.ca/~riyad/DataSets/MAWI/MAWI.zip)

NIMS and MAWI datasets have very large dimension (NIMS-713851x23), (MAWI-500000x23), which means it has 22 features and last coloumn is label. These fetaures are listed in table 11.1 below

Table 11.1 Network traffic flow features

S.No.	Feature Name	S.No.	Feature Name	S.No.	Feature Name	S.No.	Feature Name
1	Min_fpktl	7	Max_bpktl	13	Min_biat	19	Total_fpackets
2	Mean_fpktl	8	Std_bpktl	14	Mean_ biat	20	Total_fvolume
3	Max_fpktl	9	Min_fiat	15	Max_ biat	21	Total_bpackets
4	Std_fpktl	10	Mean_ fiat	16	Std_ biat	22	Total_bvolume
5	Min_bpktl	11	Max_ fiat	17	Duration		
6	Mean_bpktl	12	Std_ fiat	18	Proto		

Fpktl-forward packet length

Bpktl-backward packet length

Fiat-*forward inter-arrival time*

Biat-backward inter-arrival time

Total_fpackets-Total Forward packets

Total_fvolume- Total forward packet volume

Proto-protocol

Total_bpackets-Total backward packets

Total_bvolume- Totalbackward packet volume

Output classes in MAWI dataset are given in the form of two classes which are SSH and NOTSSH. In NIMS datatset output classes are given in further details which are shown in table below.

Table 11.2 NIMS dataset output classes

S.No. Main Class Types of traffic flows

1	SSH (Secure Shell Services)	local tunnelling
2		remote tunnelling
3		SCP(Secure copy)
4		SFTP (Secure File Transfer Protocol)
5		Shell
6		X11
7	NOT SSH Services	DNS (Domain Name System)
8		FTP (File Transfer Protocol)
9		HTTP (Hyper Text Transfer Protocol)
10		TELNET
11		Lime (P2P)

11.2 Classification using four classifiers

We have used four classifers which are K Nearest Neighbour (KNN) Classifier, Support Vector Machine (SVM) classifier, Naïve Bayes classifier and RUSBoost classifier.

The K nearest neighbours algorithm is a non-parametric method which is used for classification and regression. In both cases, input must consists of the k closest training examples in feature table. For KNN classification, the output is a class membership. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its K nearest neighbours. If K=1, then the object is simply

assigned to the class of that single nearest neighbour. The KNN algorithm is among the simplest of all machine learning algorithms.

In machine learning , SVM (Support Vector Machines) are used as supervised learning models with associated learning algorithms that analyse dataset which is used for classification. We have a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm build a model which is representation of the examples as points in space properly mapped so that the examples of the separate categories are divide by a clear gap that is as wide as possible. New examples are then mapped into that very same space and predicted examples belong to a category based on which side of the gap they fall.

Other two classifiers are Naïve Bayes and RUSBoost classifiers which are simple probabilistic classifier. Naïve Bayes classifiers are a family of simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions between the features. Once traffic data feature table is formed it is divided randomly into training and tetsing data in the ratio of 80:20. All four multi class classifier is used to create a training model using training data and testing data is used to test the label using that trained model.

Output labels of testing data is compared with predicted labels from trained model created using our proposed method with all features of image and calculates all four performance evaluation parameters.

11.3 Performance Evaluation Parameters

Following parameters are used to compare performance of different techniques.

1. Accuracy
2. Sensitivity
3. Error
4. Prevalence

Definitions of performance parameters:-

(tp) True Positive means correctly identified

(fp) False Positive means incorrectly identified

(tn) True Negative means correctly rejected

(fn) False negative means incorrectly rejected

Accuracy can be described as the closeness of a measurement to the true value. It is also used as a statistical measure of how well a binary classification test correctly identifies or excludes a condition.

Accuracy $= \frac{tp+fn}{tp+fn+fp+fn}$

Sensitivity may be referred as the model's ability to correctly predict the correct output label.

$$Sensitivity = \frac{number\ of\ true\ positives}{number\ of\ true\ positives + number\ of\ false\ negatives}$$

Sensitivity is also known as True positive rate (TPR)

Sensitivity$= \frac{tp}{tp+fn}$

Prevalence may be defined as ratio of positive condition to the total population size.

$$Prevalence = \frac{\sum Condition\ Positive}{\sum Total\ Population}$$

Error is Mean square error between actual label and predicted label value.

11.4 Overall workflow diagram

Overall work can be collectively reminded as work flow diagram where each step will be shown. In pre-processing of dataset we arrange dataset features in usable format using Net make or MATLAB. Then different probable combinations for training models are created using four classifiers which are further used for testing and labelling.

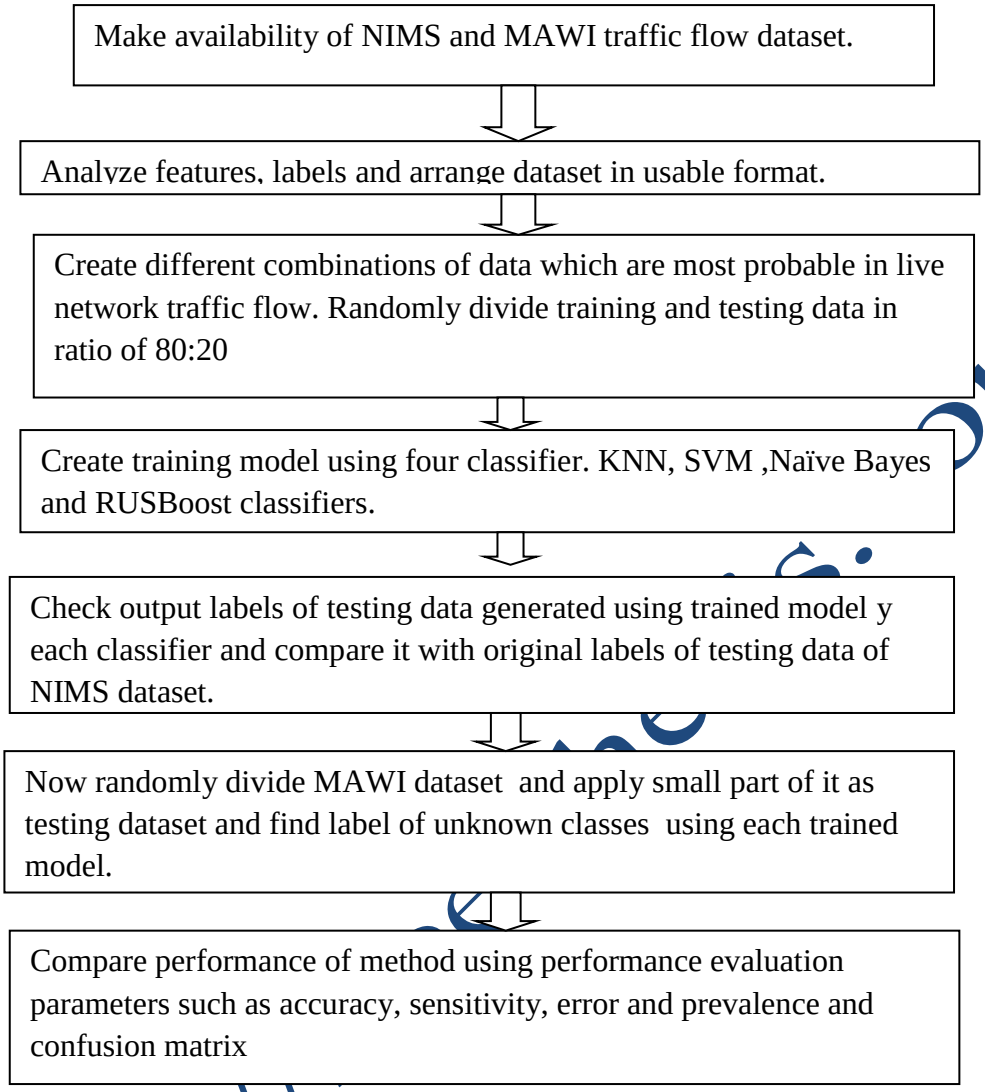
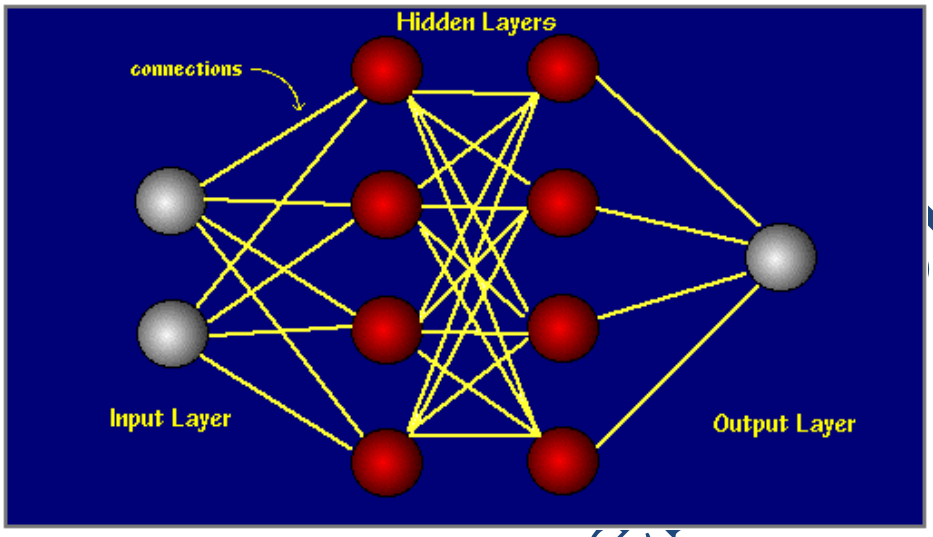


Figure 11.1 overall work flow diagram of proposed work

Thesis-12
Credit Card Fraud Detection using GSA and NEURAL Network



Abstract

Billions of dollars of loss are caused every year by fraudulent credit card transactions. The design of efficient fraud detection algorithms is key for reducing these losses, and more and more algorithms rely on advanced machine learning techniques to assist fraud investigators. The design of fraud detection algorithms is however particularly challenging due to the non-stationary distribution of the data, the highly unbalanced classes distributions and the availability of few transactions labelled by fraud investigators. At the same time public data are scarcely available for confidentiality issues, leaving unanswered many questions about what is the best strategy. We found German credit card fraud detection database available publically which is having 1000 data points. This dataset is divided into 70/30 ratio for training and testing the neural network. The famous and efficient machine learning neural network algorithm is used to get a trained NN. This network is further updated for more classification accuracy using Gravitational Search Algorithm (GSA) which is an optimisation algorithm. It tunes NN's weights and biases and check for the mean square error which is an evaluation parameter also in our work. Complete work is simulated in MATLAB R 2016a. Results are compared with previously used simulated annealing (SA) algorithm and proposed method is giving better results in term of area under curve (AUC) of ROC (receiver operating characteristics) and MSE.

Proposed Work

The credit card fraud detection is emerging risk field with more and more presence of user's on internet. With the introduction of Digital India movement, online payments and money transfer is increased. This all raises a group of people who defraud the online activities. So the need of credit card fraud detection and prevention is utmost required. In our work we proposed a novel algorithm to detect the credit card. The method is using machine learning algorithm as main along with evolutionary optimisation algorithm to improve the performance of neural network (NN). As discussed in previous chapter NN is also an iterative process which changes its input weights and biases to achieve the minimum mean square error (MSE). It is using feedback propagation loop which is using Lquenberg algorithm. This algorithm iterates locally which means it doesn't guarantee the convergence of all minima points. it may skip some combinations of input weights and biases which may reduce the MSE more. To avoid this issue we have adapted the optimisation method named Gravitational Search Algorithm (GSA) which is explained in previous chapter. It is based on the movement of celestial bodies and position of these agents are input weights and biases in our case. The output of NN is calculated by formula in 12.1.

$$output = input * IW_i + B_i \qquad (12.1)$$

where IW_i are the input weights and B_i are the biases. The number of input weights and biases depends upon the number of hidden layers. The GSA algorithm is supposed to tune these values. For this purpose first the Neural network is created in MATLAB. That network will be used further for optimisation algorithm. We have use the German dataset downloaded from UCI machine learning repository [30]. This dataset contains 20 attributes along with a label of good and bad. If label is 1, those attributes are for non fraud case and vice versa. A snapshot of data is shown in figure 4.1. The description of this data is given in appendix A.1. In out proposed algorithm of optimised neural network we need the numeric dataset, so this dataset in numeric format is also available on the same web link.

Now further we divide our this chapter in two sections. First section will discuss and show how NN code is generated for our purpose using MATLAB's neural network toolbox and second section will explain how GSA optimises the NN's weights and biases.

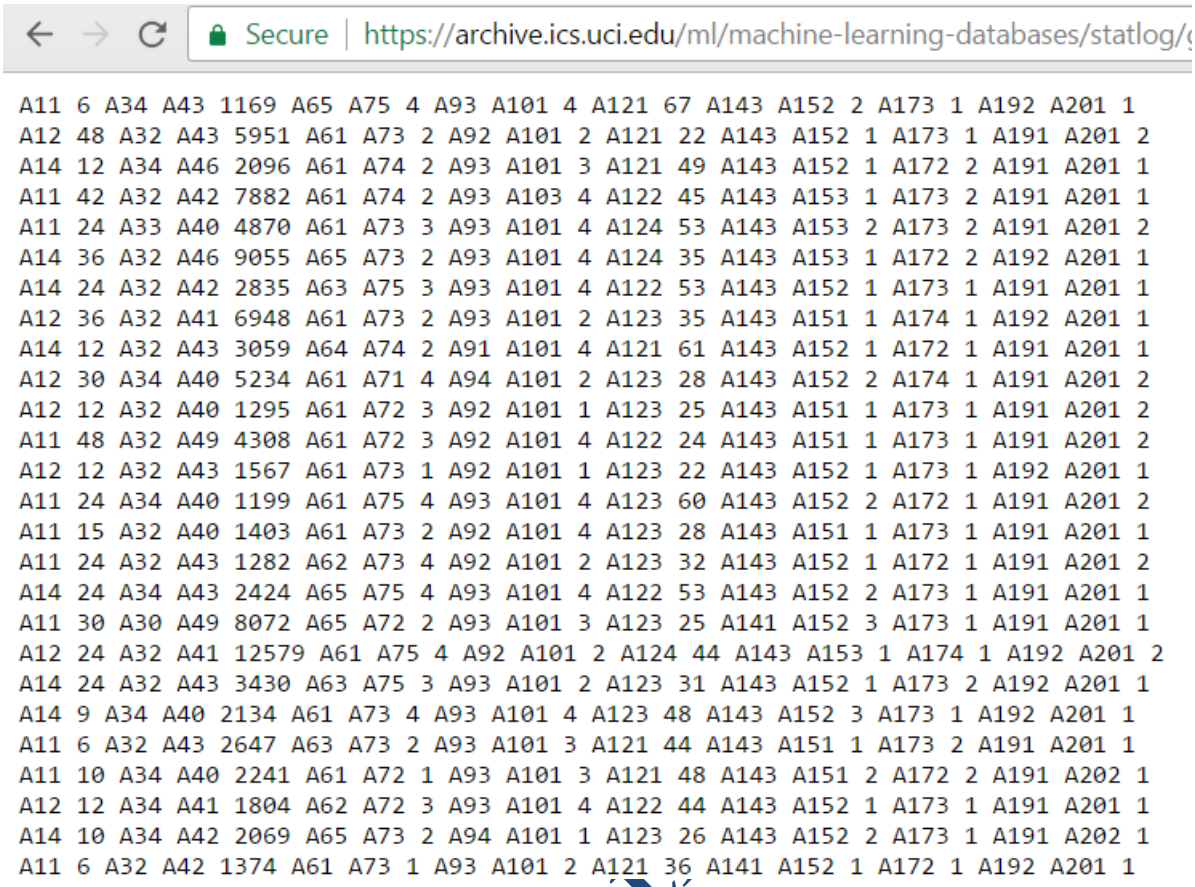


Figure 12.1: A snapshot of German credit card fraud dataset. Last column in the data tells whether it is a fraud or not

12.1 NN code generation for credit card fraud detection

Neural network toolbox requires a training dataset for training purpose so that it can get trained and learn the behaviour of data. We don't have separate dataset for testing and training so we divided the present dataset randomly in 70/30 ratio to use 70% for network training and 30% for testing.

MATLAB provides a neural network toolbox which can be used for several purposes and network this trained can be deployed as standalone application or can generate a script for further use of modifications. We used this facility to speed up our work. A user interface of NN toolbox can be opened by using command *nnstart* in MATLAB's command window. figure 12.2 shows the interface opened from this command. Since our work is recognising the pattern of previous credit card frauds, so we will use it pattern recognition app which lands to a page to chose the input data and target data. These datasets are picked from workspace of MATLAB, so these must be there already.

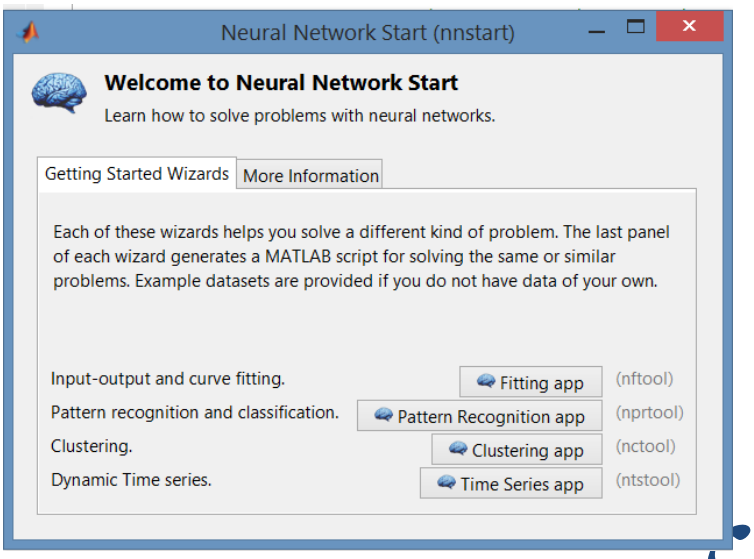


Figure 12.2: MATLAB's NN toolbox interface

After choosing the data division for training and testing of network, network is created which further leads to a page where user can input the number of hidden neurons. We have set hidden neurons at 20. Figure 12.3 shows that landed page.

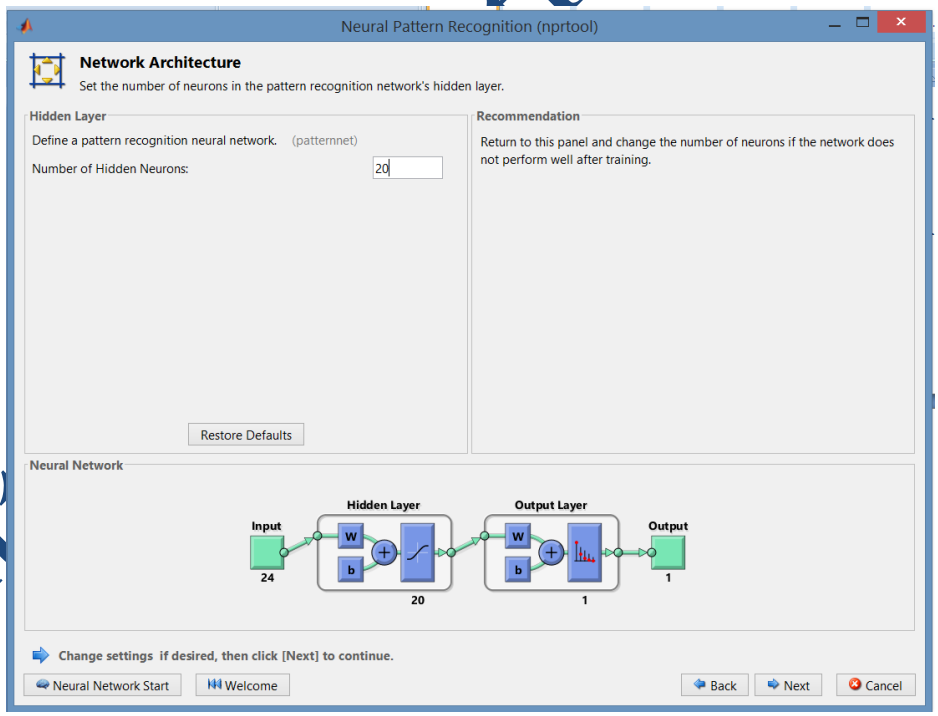


Figure 12.3: NN toolbox UI for entering hidden neurons

Then this network is trained for loaded dataset and tested with rest 30% of data. After training mean square error is generated and displayed on the user interface. Thus trained and

tested NN by this toolbox can be converted in the form of MATLAB script which is require in our work. Figure 12.4 shows the option in NN toolbox interface for it.

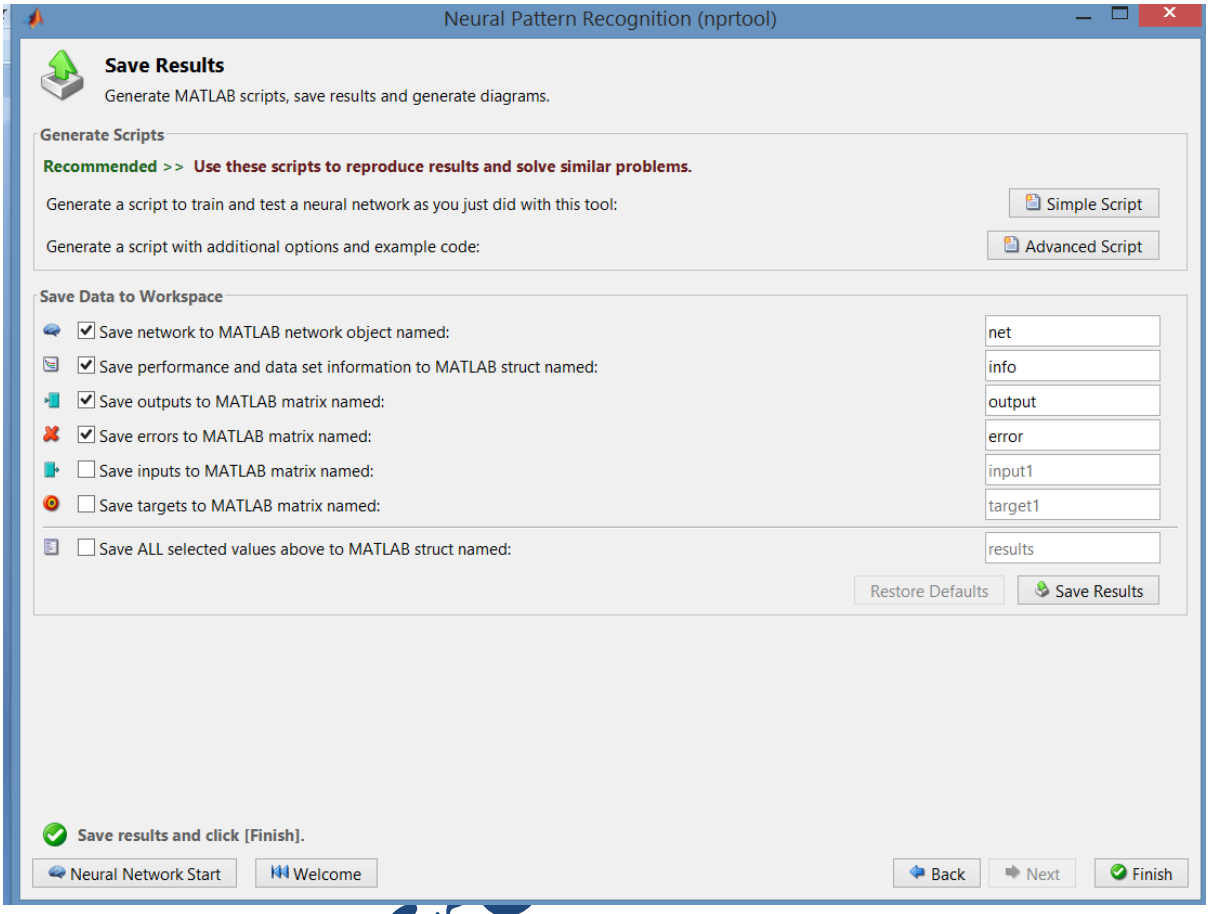


Figure 12.4: NN toolbox interface to generate the require NN script

This way we can easily get our MATLAB code for NN network for our credit card fraud analysis. The script thus generated is shown in table 12.1.

Table 12.1: MATLAB generated script by NN toolbox

```
function [net,NN]=nntrain(input,target)
% This script assumes these variables are defined:
%
%   input - input data.
%   target - target data.


x = input';
```

```
t = target';
t(t==2)=0;
% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainbr';  % Scaled conjugate gradient backpropagation.

% Create a Pattern Recognition Network
hiddenLayerSize = 20;
net = patternnet(hiddenLayerSize);

% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nnprocess
net.input.processFcns = {'removeconstantrows','mapminmax'};
net.output.processFcns = {'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivide
net.divideFcn = 'dividerand';  % Divide data randomly
net.divideMode = 'sample';  % Divide up every sample
net.divideParam.trainRatio = 70/100;
% net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 30/100;

% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
net.performFcn = 'mse';  % Cross-Entropy

% Choose Plot Functions
% For a list of all plot functions type: help nnplot
net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
    'plotconfusion', 'plotroc'};
```

```
% Train the Network
[net,tr] = train(net,x,t);

% Test the Network
y = net(x);
e = gsubtract(t,y);
NN.performance = perform(net,t,y);
tind = vec2ind(t);
yind = vec2ind(y);
percentErrors = sum(tind ~= yind)/numel(tind);

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
NN.trainPerformance = perform(net,trainTargets,y);
NN.valPerformance = perform(net,valTargets,y);
NN.testPerformance = perform(net,testTargets,y);
NN.y=y;NN.e=e;NN.tr=tr;
[NN.tpr,NN.fpr,~] = roc(t,y);
% View the Network
view(net)

% Plots
% Uncomment these lines to enable various plots.
%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, ploterrhist(e)
%figure, plotconfusion(t,y)
%figure, plotroc(t,y)

End
```

12.2 Neural Network optimisation by GSA

The proposed work is to tune NN to get the high accuracy and less mean square error. To achieve this aim we use GSA optimisation and tuned NN's weights and biases. in every optimisation task, it is required that an objective function must be set which calculates the target value like MSE in our case. This objective function will be called in each iteration and for each agent in that iteration. since the neural network is alreayd created and trained in previous step so it is not required to create again every time when objective function is called as our objective function updates the pre trained NN's weights and biases which are 251 in numbers and calculate the MSE for those set of weights and biases. The developed objective function snippet is shown in table 12.2.

Table 12.2: Objective function snippet for GSA tuned NN optimisation

```
function [performance,net]=Objective_function(L,~,net,input, target)


%   input - input data.

%   target - target data.


x = input';

t = target';

t(t==2)=0;

net=setwb(net,L); % set the input weights and biases of NN using values in 'L'

% Test the Network

y = net(x);

e = gsubtract(t,y);

performance = perform(net,t,y);


end
```

GSA is based on its agents' movements and an agent's position is represented by the weights and biases values. The number of co- ordinates of an agent's position is equal to total number of input weights and biases. In our case this number is 251. These weights and biases are the positions of all agent used in the optimisation and updated as per the equations quoted in previous chapter . These weights and biases can be fetched from generated neural network by using a MATLAB function 'getwb' and after updating these are set back to NN by 'setwb'. The significance of GSA terminology with NN tuning is provided in table 12.3.

Table 12.3: Significance of GSA terminology in NN tuning

GSA terms	In NN tuning significance
Agents Position	Input weights and biases
Dimension for optimisation/ number of variables to be tuned	Total number of input weights and biases
Update in the position of agents	Change the values of weights and biases to move towards minimum MSE

A complete step by step algorithm is explained below.

Step1. Load the German credit card fraud dataset in numeric format and divide that into random 70/30 ratio for training and testing of neural network.

Step2. generate the NN script to create and train the network whose weights and biases are to be optimised.

Step3. initialise the GSA parameters like number of iterations, number of agents, initial G0 and alpha. Pass the previously created network into GSA to get the dimension of weights and biases.

Step4. randomly initialise the new input weights and biases to give an initial seed to GSA optimisation. These must be within a boundary as given in next chapter.

Step5. call the objective function to update the neural network's weights and biases and calculate the MSE for those values by using the testing dataset.

Step6. to update the random positions of agents, force and mass has to be calculated by using the equations

$$F_{ij}^{d}(t) = \mathrm{G(t)} \frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon} X_j^d(t) - X_i^d(t)),$$

$$m_{it} = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}$$

the respective notations are given in previous chapter

Step7. The new updated position is obtained from the formula

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1)$$

the velocity in this case is calculated by using acceleration which is based on force and mass calculated in previous step.

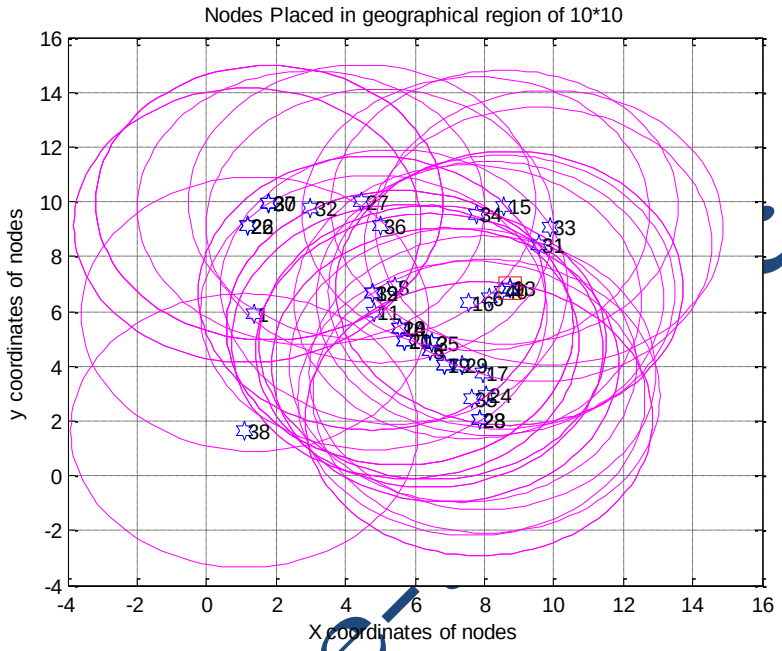
Step8. For this new updated position or values of weights and biases, objective function is again called and MSE is saved.

Step9. The weights and biases for which minimum of MSE is obtained out of previous two set of values, is further considered for updating.

Step10. This process continues till all iterations are not completed.

Step11. The final minimum MSE is obtained and weights ad biases set for them is used as final NN weights and biases which gives less MSE than conventional NN and Simulated Annealing tuned NN which was done previously by Khan, A. et.al [26].

Thesis-13
PSO-GSA Tuned Dynamic Channel Allocation in Wireless Video Sensor Networks for IOT



Abstract

We are witnessing the formation of an Internet of Things (IoT), where real-world entities (e.g., people, plants, cars) augmented with computing devices (e.g., smart phones, tablets, sensor no des), sensors (e.g., humidity sensors, microphones, cameras), and actuators (e.g., motors, LED) are connected to the Internet, enabling the m to publish their generated data on the Web. By mashing up these “Smart Things” with the services and data available on the Web, novel IoT applications can b e created. Two main characteristics of the IoT are its large scale interconnecting billions of Smart Things in the next decade, and the resource limitations of Smart Things (i.e., of their embedded computing devices). For many IoT applications, routing and sensor search are essential services. The sensor search service allows for quickly finding Smart Things in the IoT based on the real-world states perceived by their embedded sensors. To facilitate sensor search and also other IoT applications, a routing service is required to enable efficient communication of information among Smart Things and other Internet nodes. Due to the resource limitations of Smart Things, the design of these two services is challenging. Our thesis is that despite the large scale of the IoT and the resource

limitations of Smart Things, efficient solutions for the routing and sensor search services can actually be provided.

We support this thesis by proposing, implementing, and evaluating routing algorithm for high-scale wireless sensor networks (which are a building block of the IoT), and channel allocation for those searched path. Particle Swarm Optimisation (PSO) and Gravitational Search Algorithm (GSA) are used to assign the optimal channel bandwidth to all paths out of total assigned bandwidth to minimise the total path loss so that all data can be transferred to destination.

We have tested the algorithm on video transmission in sensor networks of IoT which are wirelessly connected which is the backbone of IoT network.

Proposed Work

This work is targeted to minimise the power consumption in wireless video application of IOT. The mathematical formulation of this, converted the solution in to a non linear equation which can't be solved linearly. Optimisation algorithms have to be applied on them.

The mathematical problem for this work is represented as:

$$obj = \sum P_{TX} \times UR$$

Where P_{TX}= transmitted power and UR is link utilisation ratio. The Ptx and UR will be calculated between two sensor nodes termed as edge. These nodes must be alive only when any activity is detected and remaining will be in sleep mode to save power. These active nodes only take part in routing of video data to destination. This work is in support of multi hop data transmission. The choice of optimal edges satisfying the above equation is subject to constraint that

I. $e \in E$; where 'e' is the edge and 'E' is the set of all alive edges
II. The remaining link energy at each edge must be to hold the new sensor data.

Keeping these constraints into consideration, PSO-GSA algorithm looks for optimal edge which consumes least power in transmission. A flow chart of this is shown in figure 4.1.

This work is divided into two sections. In one part route selection is done and in another part channel allocation to each route is done using hybrid PSO-GSA algorithm. in channel

allocation part. Since there are many nodes available but to save energy only some nodes are active at a time and transmission of video data takes place only through those active nodes. This route selection is based on minimum distance between active nodes. Route selected by data packet is that which has minimum distance between source and destination. The mathematical approach for it is discussed as:

Each edge's link rate varies depending on the distance between two nodes, and the radio channel condition. From the measurement of wireless modem modules, we can observe the following relations. Let *D* is the distance from a source node to a destination node. A general formula for the path loss for a wireless link is given by (in unit of dB) :

$$Path\ Loss, L_p = 20log_{10}(4\pi D/\lambda)$$

Here λ is wavelength of the RF signal. The TX power *PTx(e)* for each edge can then be represented as follows (in unit of *mW):*

$$TX\ power\ P_{TX}(e) = \frac{10^{LP}}{10\alpha}$$

Here α is a channel factor. As described above, the maximum link rate is assumed as 100Mbps, which is R_e^{max} Then with distortion factor β, a possible link rate for each edge e can be defined by

$$link\ rate\ R(e) = \frac{R_e^{max}}{P_{TX}(e) \times \beta}$$

The total data rate traversing an edge e is defined as U(e). Then the link utilization ratio *UR(e)* for edge e is defined by

$$utilisation\ ratio\ UR(e) = \frac{U(e)}{R(e)}$$

Here *R(e)* is a possible link rate for edge e. For each edge e, the effective TX power $P_{eff}(e)$ is defined by

$$P_{eff}(e) = P_{TX}(e) \times UR(e)$$

This reflects the important condition that each wireless modem turns on Tx power only when its link transmits data, and otherwise it goes down to power saving mode. Hence the total effective power consumption P_{eff} of the entire network is defined as

$$P_{eff}^{net} = \sum P_{eff}(e)$$

Using these set of equations route selection between active nodes are searched. Once route is finalised then all edges of these paths are used to allocate channels. Each edge or path gets a dedicated channel for transmission once it is requested. Dynamic channel allocation is used to allocate channels and each channel is of same bandwidth which is a part of total bandwidth available in our work. We have used 5MHz channel bandwidth to allocate and a total of 6 channels are available with us. At a time only certain number of nodes are active and will take part in communication.

We have considered that each channel can be repeated number of times but a constraint has been put which restricts the use of a single channel for all edges. Each channel must be used. Firefly algorithm works in this direction considering the equation 4.1 as main objective function.

Dynamic Channel Allocation by PSO-GSA

In our proposed scheme PSOGSA optimized technique is used which is hybrid of PSO (Particle Swarm optimisation) and GSA (Gravitational Search Algorithm) optimisation algorithm which requires an objective function to minimize. GSA is a global search optimization technique which has very less probability of premature termination and when it is hybridized with PSO, convergence speed is also increased along with more intensive search of global minima point. It performs well in case of multi objective function or multi constraint function. GSA algorithm is based on the motion of celestial bodies in the universe and PSO is based on behavior of swarms as discussed in previous chapter. The counterpart of agents and particles in GSA and PSO respectively in our work is the tuning variables. The position of a single agent or single particle is defined by the number of tuning variables. Co ordinates used to define the position in a searching space are equal to the variables to be tuned. For example in our case, we need to tune the values of frequency channels, so the tuning variable will be frequency of each channel and these variables number will eb equal to number of channels.

The hybrid GSA+PSO algorithms work in the manner that PSO becomes alive to update the direction in the GSA algorithm. The update in position requires knowledge of step size and direction and in GSA the direction of movement of bacteria is random. Due to it, it takes time to converge of to reach at an optimal solution. This random direction is controlled by the PSO algorithm in our work. This make the convergence faster with each minima point checked. Initialization of direction is random but later on once for every bacteria, fitness function is evaluated, the direction is controlled by PSO. Output of fitness function becomes the local best for the PSO and based on that local best position is calculated, which is updated by the velocity update equation in PSO. This velocity is added into the old position of particle which was local best position obtained from GSA, to get the new position. This new position is updated as direction of bacteria. This way PSO tunes the direction of bacteria in gives them a direction to look for the food.

Steps of proposed algorithms are described as:

1. Initialize all initial parameters like active nodes, their position, channel bandwidth, frequency to model it.
2. Place the nodes randomly in geographical region of 10*10
3. Mark the transmission range of each node to be 5 meter and plot a circular region around each node.
4. Manage a sink node and select the best path for each node to sink based on minimum hopes and distance.
5. Pass these paths for each node to hybrid PSO-GSA optimization algorithm to get the tuned bandwidth allocation for minimization of power consumption.

PSOGSA Initialization

6. Initialize the random positions of particles in PSO.
7. Consider the searching space dimension as number of available channels
8. Initialize the weighting parameters of PSO as 1.2 and 0.5.
9. Compare the fitness value of each particle with the previous best position of bacteria. If fitness function value is less for this new position than previous position then it will be assigned as new.
10. The present bets position is termed as current position of particle for PSO and output of fitness function is Jlocal for the PSO.

GSA Starts here:

11. The current position selected in previous step is used to get the mass for each agent as per GSA algorithm. The minimum value of fitness function is selected as best and maximum as worst position and using the formulas, mass of each agent can be calculated as:

$$m_i(t) = \frac{fit(t) - worst(t)}{best(t) - worst(t)}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}$$

12. Gravitational force is calculated as:

$$F_{ij}^d(t) = G(t).\left(M_{pi}(t) \times \frac{M_{ai}(t)}{R_{ij}(t)} + \varepsilon\right).\left(x_i^d(t) - x_i^d(t)\right)$$

The formula is described in section 3.1.

13. This new velocity is the direction of particle in PSO is updated as

$$new\ velocity = old\ velocity + c1 * acceleration + c2(gbest - current\ position)$$

Here gbest is the global best position of particles in PSO and acceleration is calculated in GSA as $a_i^d(t) = F_i^d(t)/M_{ii}(t)$.

GSA ends here

14. The final position of agents which is achieved either by matching the condition of power reduction or by reaching the maximum iterations.

15. Final positions of agents thus settled are considered as the final weighted sequence of PTS algorithm and multiplied with input sub blocks and PAPR is calculated.

Significance of PSO-GSA algorithm in channel allocation

The main task of pour work is allocating the channels to edges. This work is done by hybrid PSOGSA algorithm which is a bio inspired algorithm and fired by the change in agents positions each time. Every change in position of agents is the change in channel bandwidth to achieve the minimum path loss component. Table 13.1 shows the significance of bio terms in

PSOGSA with our proposed technical terms. This table will correlate proposed optimisation with channel allocation task.

Table 13.1: Related terms of firefly algorithm with channel allocation task

Bio terms of PSOGSA algorithm	**Corresponding meaning in channel allocation**
Position of agents/swarms	Bandwidth of channels to be allocated
Number of dimension of searching space	Number of channels to be allocated
Update in positions of patcicles/agents	Change in the bandwidth allocated to channels

A MATLAB script has been developed for the proposed task and its objective function is tabulated in table 13.2

Table 13.2: MATLAB script for objective function to be used in PSO-GSA algorithm

```
function objv=effective_power(lambda)

global E dist

lambda=round(lambda);

for ii=1:length(E.path)-1

    if length(lambda)<length(E.path{ii})

        lambda=[lambda, repmat(lambda,1,length(E.path{ii})-
length(lambda))];

    end

    for kk=1:length(E.path{ii})-1


        wavlngth=lambda(kk); %assign channel to each link

        maxR=100; % maximum link rate in Mbps

        pathloss=20*log10((4*pi*dist(kk,kk+1))/wavlngth);

        Ptx=(10^pathloss)/10;

        R=maxR/Ptx;
```

```
%          hop=length(E.path{ii})-1;
          U=10*kk;
          UR=U/R;
          Peff_edge(kk)=Ptx*UR;
     end
     Peff(ii)=sum(Peff_edge);
end
objv=sum(Peff);
end
```

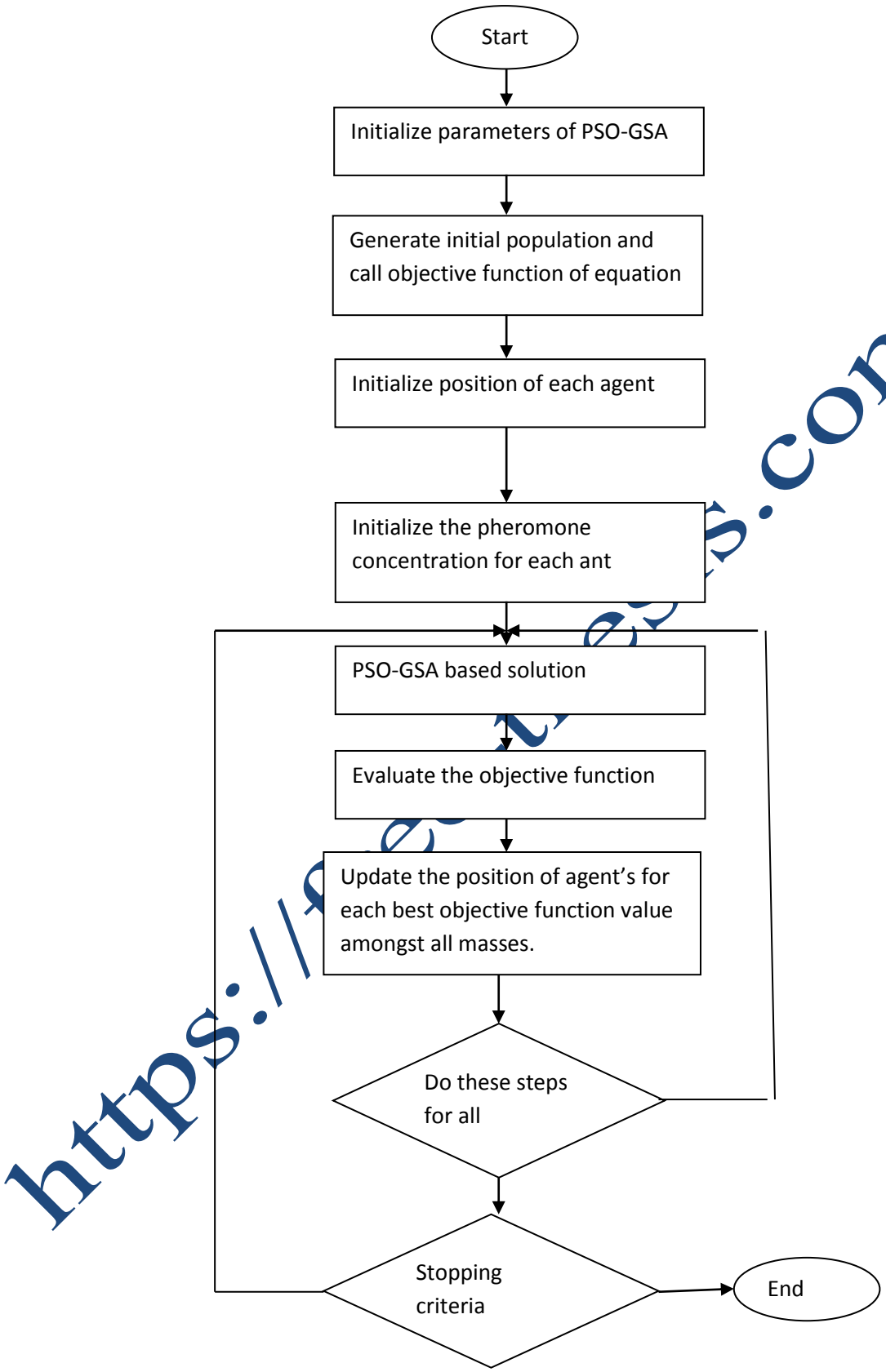
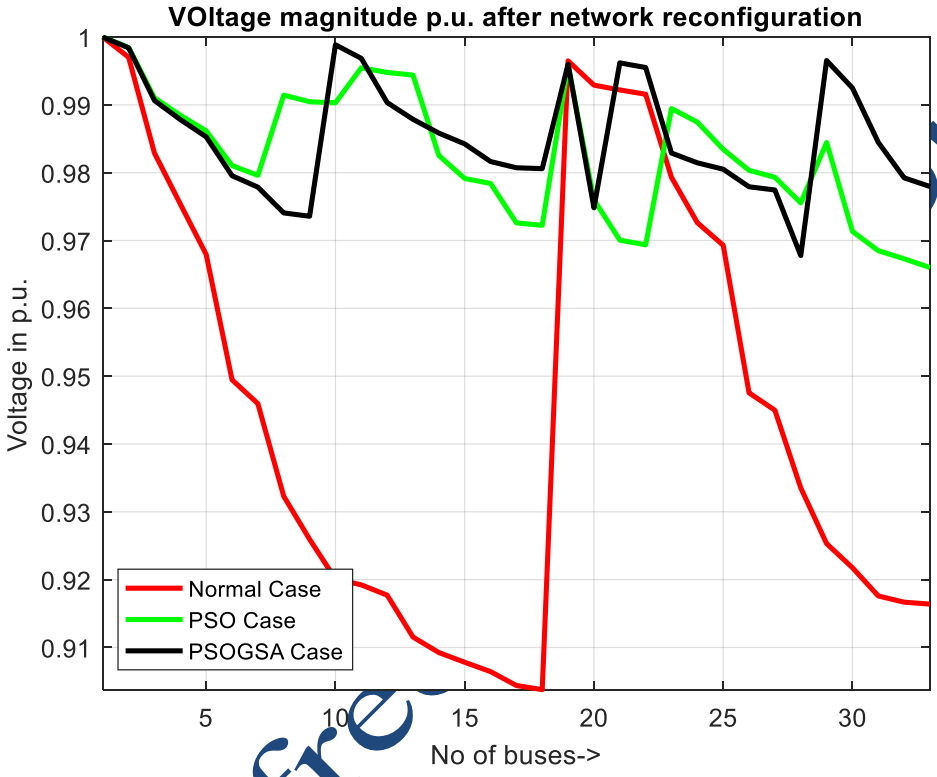


Figure 13.1 Flowchart of proposed work

Thesis-14
Multi-Objective Network reconfiguration Using Hybrid GSA-PSO Optimization



Abstract

Power distribution systems have tie and sectionalizing switches whose states determine the configuration of the network. Reconfiguration of distribution network is achieved through switching operation on switches of distribution network branches. Power companies are interested in finding the most efficient configuration for minimization of real power losses and load balancing among distribution feeders to save the energy and enhance the operation performance of distribution system. The objective of this thesis is to show that the hybrid PSO and GSA optimisation algorithm can be used successfully in the reconfiguration of electrical distribution networks to minimize the power losses of the system and to balance the loading of the feeders.

This work is to provide a basis for power companies to use it in the reconfiguration of the distribution networks to reduce the operational costs and to

enhance the performance of their networks. The network reconfiguration problem is usually formulated as a single objective optimization problem with equality and inequality constraints whereas it is a complex combinatorial optimization problem. This is because there are multiple constraints which must not to be violated while finding an optimal or near optimal solution to the distribution network reconfiguration problem. As a result, more efficient approaches are required to handle this combinatorial problem. In this work hybrid a PSOGSA is adopted in solving the optimization problem. This is based on the movement of swarms in a defined searching space whose velocity is changed towards convergence by GSA agents. The searching space is the number of tie switches to be tuned. Basically the optimisation and power distribution system are two separate isolated fields which are connected through an equilibrium of inputting the tie switch number to distribution system from PSOGSA optimisation and get in return the power loss calculated for those tie switches.

The proposed algorithm is coded using MATLAB. The performance of IEEE 33 radial distribution network with different load has been evaluated using MATLAB to test the effectiveness and validity of the proposed algorithm.

PROPOSED WORK

The distribution network reconfiguration is an important application in line distribution system as with the increase in number of DGs in the system, it is required to use them optimally. With the high demand, the demand of reactive power is going to increase in future. This demand can be matched by using extra DGs in distribution system but that also reduces the active power, resulting in increase in losses. To keep the losses in acceptable range and to meet the demand of load, network reconfiguration is used. Tie switches are opened/closed to maintain the demand and to keep any DG in radial loop and other out of loop. It is an optimisation problem which is considered a highly non-linear, mixed integer type and non-differentiable multi-objective problem which renders it impractical to solve with traditional optimisation approaches. This is due to the discrete nature of the switches and the characteristics of different constraints and objective functions of the network reconfiguration problem. The optimal reconfiguration is targeted to use the supply from extra DG's as minimum as possible and maximum from the available resources, while maintaining the radial property of the network.

In it we have used the test case of IEEE 33 radial bus system which is having five tie switches to be optimally opened/closed. A typical single line diagram of 33 bus distribution system is shown in figure 14.1. The mesh network of that is shown in figure 4.2. While opening and closing various tie switches in the network for optimal reconfiguration, following are the constraints:

- The total number of switches selected must be equal to the number of loops formed when all tie switches are closed.
- Only one switch at a time can be opened/closed which is common to two loops. For example in L12 in figure 14.2, out of three switches available in the branch only one can be opened/closed.
- Similarly only one switch from a continuous branch or vector can be opened/closed. For example only one switch in vector L33 can be opened in figure 14.2.
- In figure 14.2, L12,L23 and L13 are intersecting at a common node, so all of them can't have any open switch at a time, only one branch can have.

This network reconfiguration task is an optimisation task, so we used a hybrid combination of optimisation algorithms.

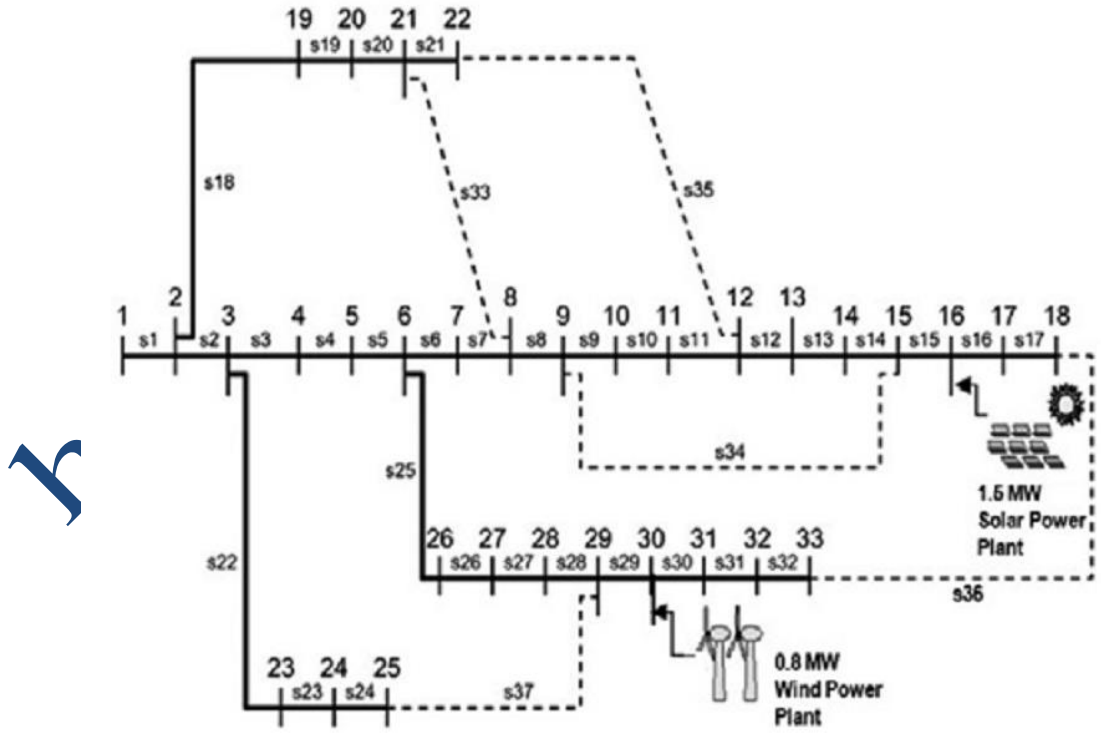


Figure 14.1: IEEE 33 radial bus distribution system with tie switches [6]

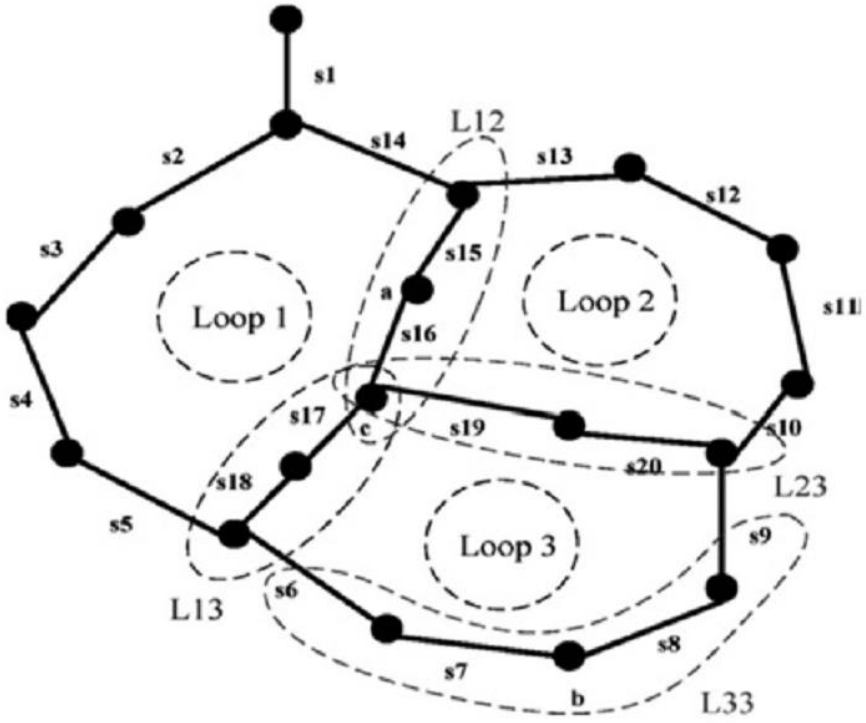


Figure 14.2: Mesh network of IEEE 33 distribution system

We used Particle Swarm Optimisation (PSO) and Gravitational Search Algorithm (GSA) in combination. There are two ways to combine two optimisation algorithms, either cascade them or merge them. We have adopted the second method. Since PSO is local optimisation algorithm so its convergence rate is faster but it skips some points which are to be checked for convergence, so optimal converge point is compromised whereas GSA is a global optimisation technique which doesn't skip any point which can be checked to find converging point but due to no skipping, it is compromise by convergence speed.

To add the advantages of both, we merged the algorithms and gained advantages. Here in our application, the convergence means, the optimal combination of tie switches which leads to minimum power losses. The IEEE 33 bus system is used whose bus & line data is tabulated in appendix A.1. To get the power flow and losses, load flow analysis has to be done prior to optimisation. Since it is radial distribution system, so conventional load flow analysis using Newton Raphson can't be used. Radial network differs from the conventional distribution network as

- They usually have a radial network structure, whereas a transmission network is usually a mesh network.

- The load they serve is usually non-uniformly distributed which may result in unbalanced operation.
- Distribution networks usually have an extremely large number of branches/nodes, with lesser number of connections for each bus/ node.
- They may have a wide variation in resistance and reactance values for all branches, with usually higher value of resistance to reactance ratio(R/X) than for transmission lines

Due to these limitations, conventional load flow analysis can't be used. The backward/forward load flow analysis will be used for this purpose which is based on two steps; forward sweep and backward sweep, and these run iteratively. The backward sweep is primarily a current or power flow summation with possible voltage updates. The forward sweep is primarily a voltage drop calculation with possible current or power flow updates.

Optimisation of tie switches by PSO-GSA

In our proposed scheme PSOGSA optimized technique is used which is hybrid of PSO (Particle Swarm optimisation) and GSA (Gravitational Search Algorithm) optimisation algorithm which requires an objective function to minimize. GSA is a global search optimization technique which has very less probability of premature termination and when it is hybridized with PSO, convergence speed is also increased along with more intensive search of global minima point. It performs well in case of multi objective function or multi constraint function. GSA algorithm is based on the motion of celestial bodies in the universe and PSO is based on behavior of swarms as discussed in previous chapter. The counterpart of agents and particles in GSA and PSO respectively in our work is the tuning variables. The position of a single agent or single particle is defined by the number of tuning variables. Co ordinates used to define the position in a searching space are equal to the variables to be tuned. For example in our case, we need to tune the tie switches, so the tuning variable will be tie switch positions in the distribution system and these variables number will be equal to number of tie switches .

The hybrid GSA+PSO algorithms work in the manner that PSO becomes alive to update the direction in the GSA algorithm. The update in position requires knowledge of step size and direction and in GSA the direction of movement of bacteria is random. Due to it, it takes time to converge of to reach at an optimal solution. This random direction is controlled by the PSO

algorithm in our work. This make the convergence faster with each minima point checked. Initialization of direction is random but later on once for every bacteria, fitness function is evaluated, the direction is controlled by PSO. Output of fitness function becomes the local best for the PSO and based on that local best position is calculated, which is updated by the velocity update equation in PSO. This velocity is added into the old position of particle which was local best position obtained from GSA, to get the new position. This new position is updated as direction of bacteria. This way PSO tunes the direction of bacteria in gives them a direction to look for the food.

Steps of proposed algorithms are described as:

Step1. Input he tie switches locations in the bus system based on the mesh networking of IEEE33 radial system. It gives 5 set of tie switches combinations, out of which optimal tie switches are to be chosen by PSOGSA for loss minimization. These tie switch options are obtained by KVL,KCL loops and are as

Tie Switches=[8 9 10 11 21 33 35 0 0
2 3 4 5 6 7 18 19 20
12 13 14 34 0 0 0 0 0
15 16 17 29 30 31 36 32 0
22 23 24 25 26 27 28 37 0]

Step2. Select the loading on buses. As we have tested the work for different load cases.

PSOGSA Initialization

Step3. Initialize the random positions of particles in PSO.

Step4. Consider the searching space dimension as number of available tie switches

Step5. Initialize the weighting parameters of PSO as 0.5 and 1.5.

Step6. Compare the fitness value of each particle with the previous best fitness value of particles. If fitness function value is less for this new position than previous position then it will be assigned as new.

Step7. The present best position is termed as current position of particle for PSO and output of fitness function is Jlocal for the PSO.

GSA Starts here:

Step8. The current position selected in previous step is used to get the mass for each agent as per GSA algorithm. The minimum value of fitness function is selected as best and maximum as worst position and using the formulas, mass of each agent can be calculated as:

$$m_i(t) = \frac{fit(t) - worst(t)}{best(t) - worst(t)}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}$$

Step9. Gravitational force is calculated as:

$$F_{ij}^{d}(t) = G(t).\left(M_{pi}(t) \times \frac{M_{ai}(t)}{R_{ij}(t)} + \varepsilon\right).\left(x_j^d(t) - x_i^d(t)\right)$$

The formula is described in section 3.1.

Step10. This new velocity is the direction of particle in PSO is updated as

$$new\ velocity = old\ velocity + c1 * acceleration + c2(gbest - current\ position)$$

Here gbest is the global best position of particles in PSO and acceleration is calculated in GSA as $a_i^d(t) = F_i^d(t)/M_{ii}(t)$.

GSA ends here

Step11. The final position of agents which is achieved either by matching the condition of power reduction or by reaching the maximum iterations.

Step12. Final positions of agents thus settled are considered as the final tie switches options to be opened for power loss minimization.

Significance of PSO-GSA algorithm in network Reconfiguration

The main task of our work is optimally select the tie switches in a mesh network. This work is done by hybrid PSOGSA algorithm which is a bio inspired algorithm and fired by the change in agents positions each time. Every change in position of agents is the change in tie switches options to achieve the minimum power losses. Table 14.1 shows the significance of bio terms in PSOGSA with our proposed technical terms. This table will correlate proposed optimisation with channel allocation task.

Table 14.1: Related terms of firefly algorithm with channel allocation task

Bio terms of PSOGSA algorithm	Corresponding meaning in network reconfiguration
Position of agents/swarms	Tie switches positions

Number of tie switches in the network

Number of dimension of searching space

Change in position of tie switches

Update in positions of particles/agents

Objective Function

An objective function is the soul of any optimisation algorithm. It calculates the dependent variables' value for various independent variables. In our case our dependent variable is power loss which depends upon independent locations of tie switches. Since optimisation algorithms are iterative methods, so this objective function gives the power loss value in each iteration for every particle. This loss is calculated by closing the five tie switches which are particle's position and load flow analysis is done for updated configuration of distribution system. We have to keep a check on the opening switches locations so that radiality of the network don't get disturbed. To check it determinant of Branch Incidence Branch Current (BIBC) matrix will be calculated and that should be not be zero.

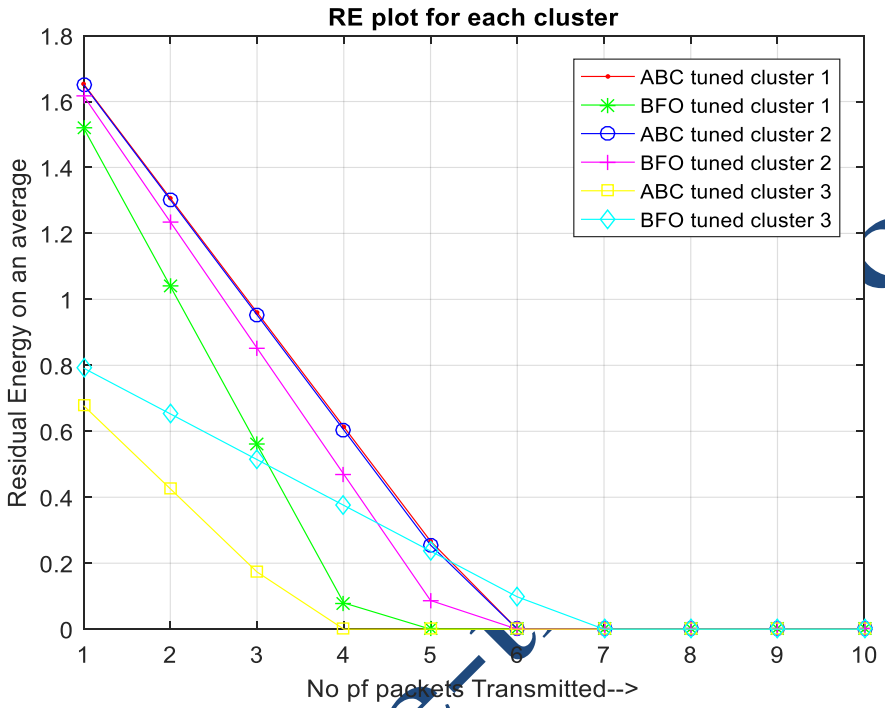
A MATLAB script has been developed for the proposed task and its objective function is tabulated in table 14.2

Table 14.2: MATLAB script for objective function to be used in PSO-GSA algorithm

```
function Pl=objFcn(BrStatus,ta,loadlevel)


[baseMVA,bus,branch,tie]=case33radial; % calling data from data file

branch(:,5)=ones(numel(branch(:,5),1));

% open the tie switch of branch in BrStatus

branch(BrStatus,5)=0;

[DistLoadFlowSolution,BIBC]=powerflow (baseMVA,bus,branch,loadlevel); % calling load flow analysis function


% Check on constraint of radial distribution network

%%%%% network will be radial if determinant of Branch incidence matrix will
```

```
%%%%% be either 1 or -1
while det(BIBC)==0

    for jj=1:dim
        maxL=length(nonzeros(ta(:,jj)));
        BrStatus(jj) = ta(round(1+(maxL-1)*rand),jj);
    end
    [baseMVA,bus,branch,tie]=case33radial; % calling data from data file
    branch(:,5)=ones(numel(branch(:,5),1));
    % open the tie switch of branch in BrStatus
    branch(BrStatus,5)=0;
    [DistLoadFlowSolution,BIBC]=powerflow (baseMVA,bus,branch,loadlevel); % calling
load flow analysis function

end
Pl=DistLoadFlowSolution.PtLosskW;
```

Thesis-15
Energy Consumption Minimization in WSN using BFO



Abstract

The popularity of Wireless Sensor Networks (WSN) have increased rapidly and tremendously due to the vast potential of the sensor networks to connect the physical world with the virtual world. Since sensor devices rely on battery power and node energy and may be placed in hostile environments , so replacing them becomes a difficult task. Thus, improving the energy of these networks i.e. network lifetime becomes important. The thesis provides methods for clustering and cluster head selection to WSN to improve energy efficiency using fuzzy logic controller. It presents a comparison between the different methods on the basis of the network lifetime. It compares existing ABC optimization method with BFO algorithm for different size of networks and different scenario. It provides cluster head selection method with good performance and reduced computational complexity. In addition it also proposes BFO as an algorithm for clustering of WSN which would result in improved performance with faster convergence.

Proposed Work

Wireless sensor network (WSN) is a very fast evolving technological platform having tremendous applications scopes in several domains for example health monitoring, agriculture, military, structural monitoring, home networks and many more. A WSN contains substantial number of small-size sensor nodes with low power consumption and must be capable of detecting physical phenomena for example constrained in energy supply, bandwidth and processing power. In our work, we implemented Bacterial Foraging Optimization (BFO) algorithm in a cluster-based routing protocol based on Sugeno fuzzy inference system. As in a given populations of nodes clustering can be done by different techniques such as FCM, Kmeans, Cmeans etc. We have used Kmeans for clustering as it is efficient and fast. After clustering centroid of clusters chosen. In the cluster-based protocols Cluster Heads (CHs) are generally selected among all sensor nodes from pool of nodes who is reliable to maintain cluster work, and then, clusters are made byassigning each node to the nearest CH. The major limitation is to generate an inappropriate distribution of CHs over WSN. The main steps of our work can be summarized as follows:

• An optimized Sugeno fuzzy inference system (FIS) is proposed as an efficient and fast, application specific routing protocol in Wireless Sensor Network environment. We have designed three embership function with 27 set of rules in Sugeno• K-means algorithm is utilized to form balanced clusters over the network.

• An objective function is made to calculate residual energy (RE), distance of node from sink (DNS), distance of node from centroid (DNC). Position of centroid is calculated by Kmeans algorithm. Objective function also find position of Cluster Head on the basis of fuzzy inference system.

• Bacterial Foraging Optimization (BFO) algorithm is implemented to optimize the fuzzy rules of FIS file in order to prolong the network lifetime, based on the different application specifications. Flow chart of our work is given below for easy understanding.

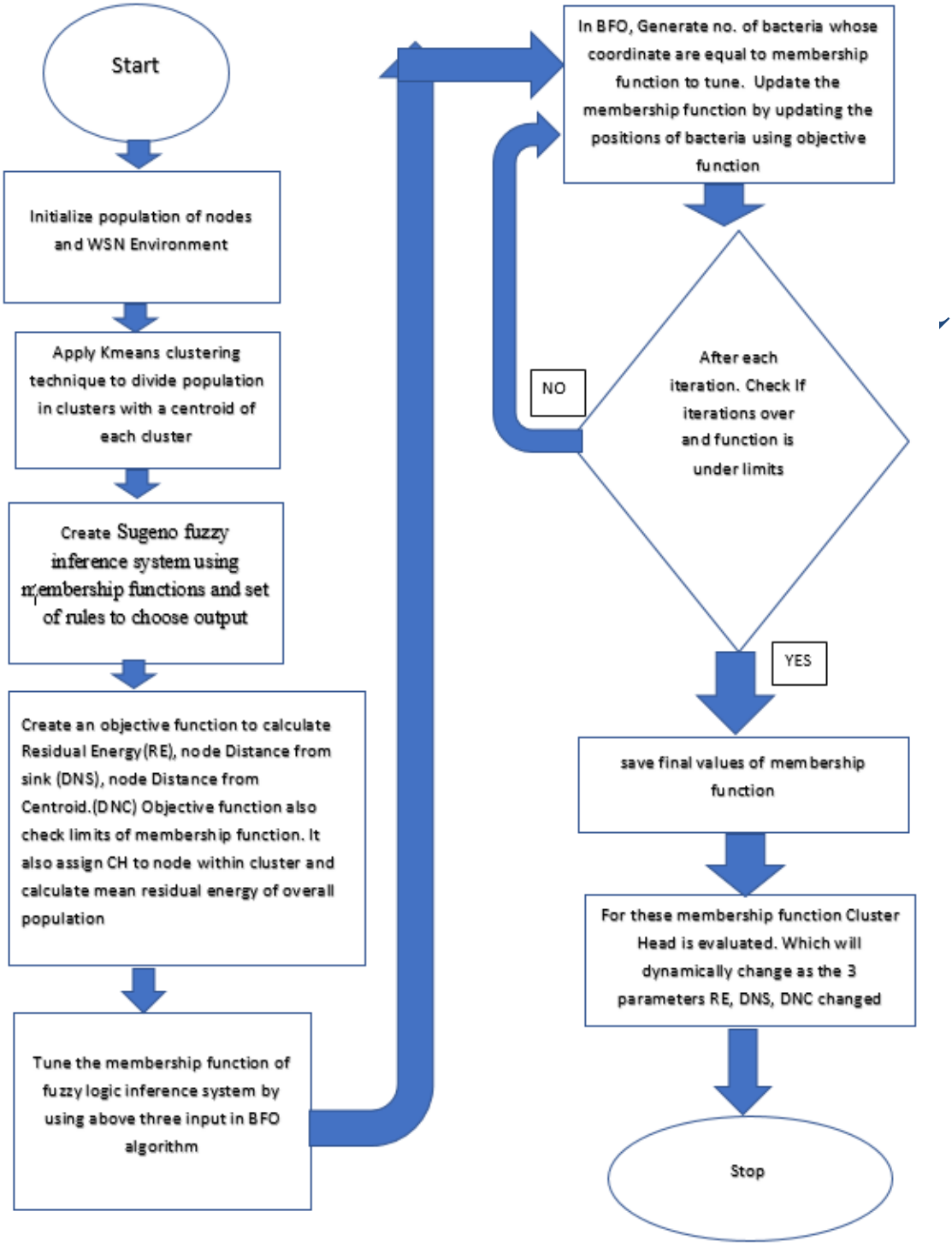


Figure 15.1 Work flow diagram

15.1 FUZZY LOGIC INFERENCE SYSTEM

As the fuzzy inference system FIS can achieve a better combination of the all input parameters to obtain the optimal output. So a Sugeno FIS is constructed in MATLAB. The fuzzy controller consists of three parts: first is fuzzification in which real environment

variables are converted to fuzzy variables, second is inference model which inherits the rule sets or decision variables and third is defuzzification which reverse the fuzzy variables to environment variables. The fuzzy logic controller for the our case has three real time inputs measured by objective function for each node in a cluster. These are:

1. Residual energy of node (RE)
2. Distance of node from sink of cluster (DNS)
3. Distance of node from centroid of cluster (DNC)

The residual energy (RE) is the most important variable which can affect the network lifetime. If a node with low RE decides to become a CH, it would be soon dead. For higher RE node, priority to become CH is more. The distance between a node and the sink (DNS) determines the energy consumption of the node. The distance between node and the centroid of cluster (DNC) is also important, because the shorter the distance from the centroid of cluster, the smaller the summation of intra-cluster distances. These input variables are considered to avoid selecting low energy nodes (RE) to minimize the total energy consumption of nodes (DNS and DNC).

A typical Sugeno fuzzy rule with three inputs *x*, *y* and *z*, (RE,DNS,DNC)and one output *w* can be shown as

$$IF = p \times x + q \times y + r \times z$$

Where p, q, r, are weightage of three variables, RE,DNS,DNC. We considered there values as 0.5,0.3,0.2 in the order of priorities.

Normalization function used to normalize input variable with in required range is given below.

$$Normalized\ x_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

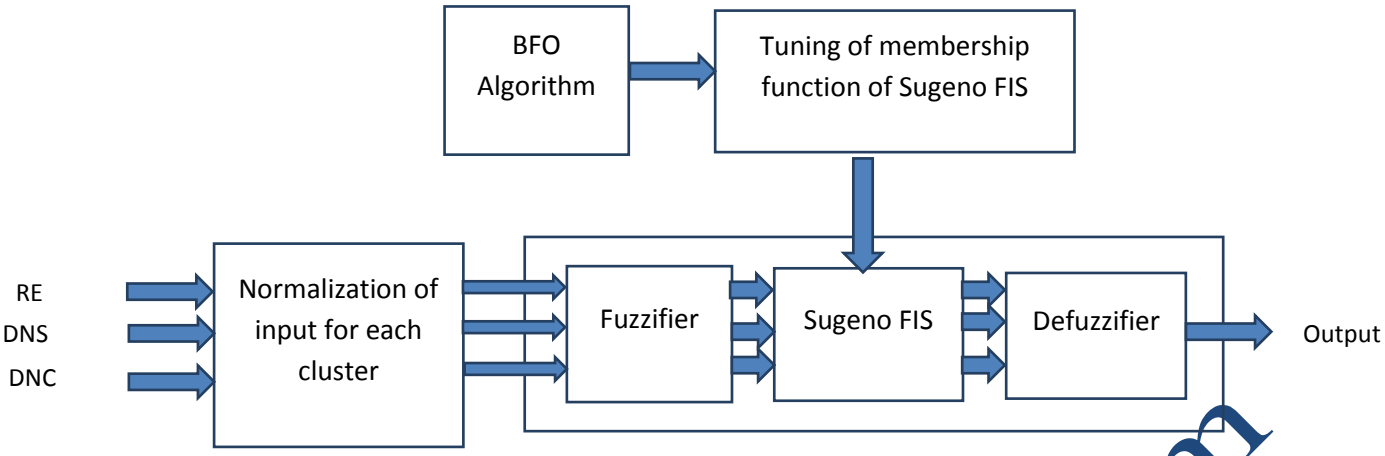


Figure 15.2Flow chart of cluster head selection procedure using Fuzzy Logic Controller

Membership functions for input to FIS is designed by taking some assumptions. These are:

1. Limits of RE input after normalization is 0 to1
2. Limits of DNS and DNC input is -1 to 0
3. Trapezoidal membership functions are used for each input
4. Initial values of range for Low membership input considered 0-0.1, for Medium membership input considered .026-0.65 and for High membership input considered 0.66-1, for RE input.
5. Initial values of range for Low membership input considered -0.25 to 0, for Medium membership input considered -0.64 to -0.25 and for High membership input considered -1 to -0.65, for DNS input.
6. Initial values of range for Low membership input considered -0.25 to 0, for Medium membership input considered -0.64 to -0.25 and for High membership input considered -1 to -0.65, for DNC input.

The surface viewer of our fuzzy logic is shown in figure 15.3 . It is a three-dimensional representation of mapping of error and output of fuzzy logic.

The three input signals are fuzzified and represented in fuzzy set notations by membership functions. The defined ‘if … then …’ rules produce the linguistic variables and these variables are defuzzified into control signals for comparison. Fuzzy logic control involves three steps: fuzzification, decision-making using FIS and defuzzification. Fuzzification transforms the non-fuzzy (numeric) input variable measurements into the fuzzy set (linguistic) variable that is a clearly defined boundary. In the proposed controller, the RE, DNS and DNC are defined by linguistic variables such as LOW, MED, HIGH characterized

by memberships. The memberships are curves that define how each point in the input space is mapped to a membership value between 0 to 1 for RE and -1 to 1 for DNC and DNS. The membership functions belonging to the other phases are trapezoidal Membership functions for the inputs are shown in Fig.15.4 and Fig.15.5. In sugeno FIS type we need to declare type of output on the basis of input membership functions.

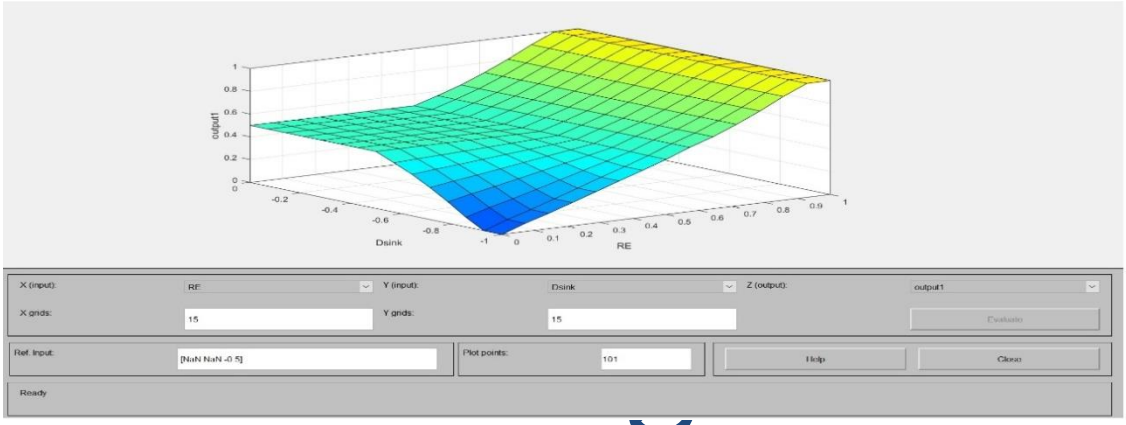


Figure 15.3: Surface viewer plot of fuzzy logic

Figure 15.4: Membership function of input RE

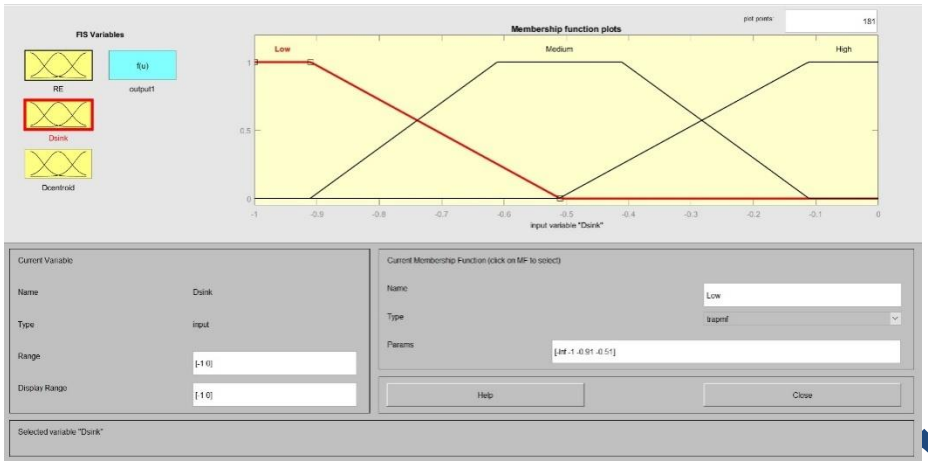


Figure 15.5: Membership function of input DNS

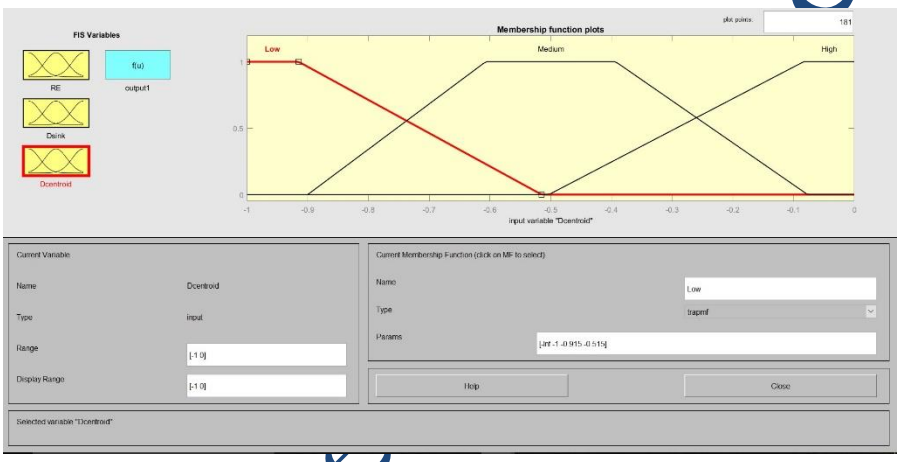


Figure 15.6: Membership function of input DNC

This decision will depend on the input values for the system. The defuzzified output is displayed as a bold vertical line on this plot. The Rule Viewer allows you to interpret the entire fuzzy inference process at once. The Rule Viewer also shows how the shape of certain membership functions influences the overall result. Based on these rules output duty cycle range is decided. For three inputs and each having three levels, we have made 27 rules for FIS. Rules set for FIS is given in figure 15.8

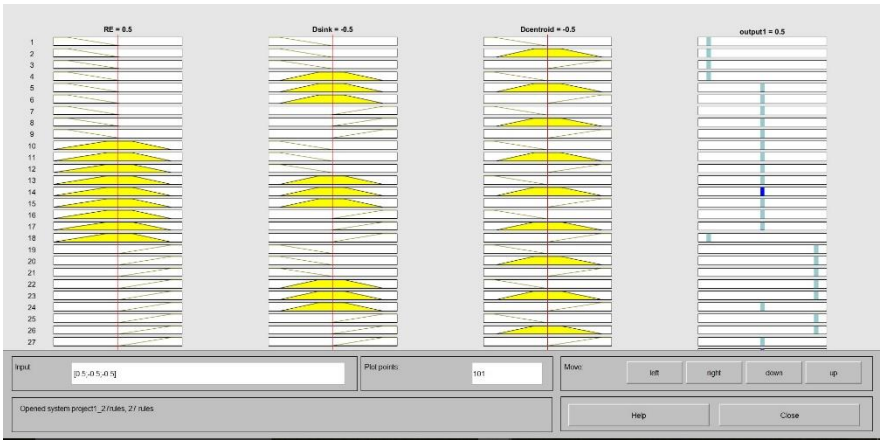


Figure 15.7: Rule viewer of membership functions

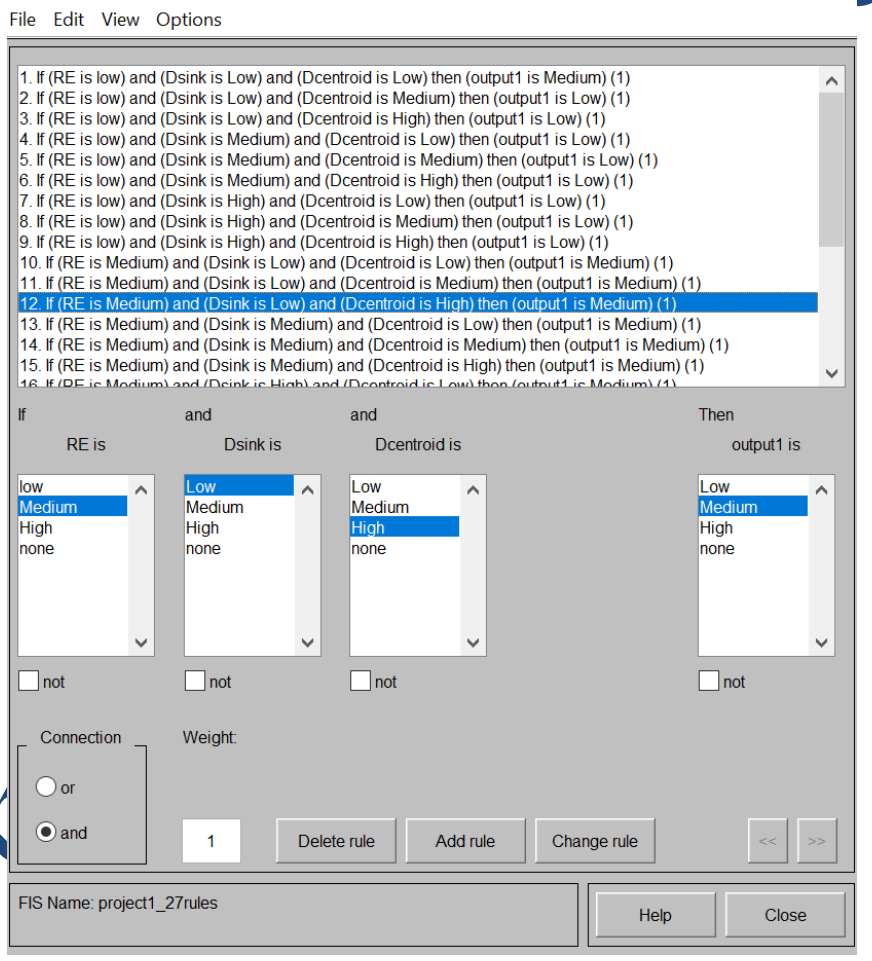


Figure 15.8 Rules set for Fuzzy Logic controller

15.1.1 Calculation of Residual Energy of each node

In every round, the nodes sense the WSN environment and send the obtained information to the sink. The sink is responsible for receiving the data from nodes, and sending this information to the user end. All nodes have the same capabilities of sensing, processing and communicating. Each node can communicate directly with the sink and with other nodes. All

nodes are equipped with GPS devices, so they are aware of their own location as well as the location of the sink. The first order radio communication is used to model the dissipated energy. In this model, a radio dissipates $E_{elec} \times l$ to run either the transmitter or the receiver circuitry. The energy consumption for transmitter an l bit data packet with distance d can be formulated as equation 15.1. Depending on the transmission distance, the free space and multipath fading channel model is used for transmitter amplifier, as seen in the right term of the eq.15.1. Also the energy consumption to the receiver an $l\ bit$ data packet is calculated according to eq.15.2

$$E_{TD}(l \times d) = \begin{cases} l \times E_{elec} + l \times \varepsilon_{fs} \times d^2 & if\ d \le d_0 \\ l \times E_{elec} + l \times \varepsilon_{amp} \times d^4 & if\ d > d_0 \end{cases} \quad (15.1)$$

$$E_{RX}(l) = l \times E_{elec} \quad (15.2)$$

Where E_{elec} is the dissipated energy (per bit) in every transmitter and receiver circuit, and depends on such electronics factors as digital coding, modulation, filtering and spreading of the signal. The amplifier parameter used for free space and multipath environment ε_{fs} and ε_{amp}respectively. The distance threshold d_0 is defined as $d_0 = \sqrt{\varepsilon_{fs}/\varepsilon_{amp}}$

15.1.2 Performance evaluation:

An overall clustering is firstly performed via Kmeans and then a CH is selected within the each cluster. In order to demonstrate the effectiveness of the proposed, the maximum and standard deviation of intra-cluster distance are used to compare by BFO and ABC algorithms. The standard deviation of intra-cluster distance can be expressed as

$$STD_{cl} = \sqrt{\frac{1}{N}\sum_{J=1}^{M}\sum_{i=1}^{N}(x_i\, a_{ij} - c_j)^2} \quad (15.3)$$

Where, a_{ij} is binary parameter determining whether node i belongs to cluster j or not.

In order to demonstrate the effectiveness of the BFO, simulation results of the both algorithm are compared for 50mx50m , 100mx100m and 200x200m geographical area. The base station located at the centre of the network. All sensors have the same initial energy. Table 15.1 presents the network details.

Table 15.1Network details

Parameter	Value

Initial energy	1 J
E_{elec}	50 nJ/bit
E_{fs}	100 pJ/bit/m^2
E_{amp}	0.013pJ/bit/m^2
Data packet size	4000 bit
Control packet size	50 bit

15.3 FUZZY LOGIC TUNED WITH BFO

15.3.1 Description

In previous section we discuss the fuzzy logic decided output on the basis of three inputs RE, DNS, DNC membership functions. In figures 15.2-15.4 membership range of three inputs are defined. In the given fuzzy system, the number of fuzzy memberships for theeach input is 3 (Low, Medium, and High). So, the number of fuzzy rules is 3×3×3=27.We tuned the membership function range based on the input conditions and bacterial foraging optimisation (BFO algorithm. In BFO Bacteria move in random direction in search of its food which takes time into convergence of BFO. The positions of bacteria are the initial position of FIS membership functions and direction updated on these positions constitute the direction of bacteria in BFO which is now tuned rather than random. We need to tune the range values of membership functions of fuzzy logic. We have three membership functions for each input a total of 9 membership function. Figure 15.2-15.4 depicts the answer to this. The initial and final range of trapezoidal function is fixed which is at –infinity to +infinity. Moreover, two points of each membership function are common to others. It is shown in table 15.2-15.4

Table15 .2 Membership function range and type of first input RE

Membership functions Type	Range parameters
Trapezoidal	`[-inf 0 x(1) x(3)]`

Trapezoidal	[x(1) x(2) x(4) x(5)]
Trapezoidal	[x(3) x(5) 1 inf]

Table 15.3 Membership function range and type of second input DNS

Membership functions Type	Range parameters
Trapezoidal	[-Inf -1 x(6) x(8)]
Trapezoidal	[x(6) x(7) x(9) x(10)]
Trapezoidal	[x(8) x(10) 0 inf]

Table 15.4 Membership function range and type of third input DNC

Membership functions Type	Range parameters
Trapezoidal	[-Inf -1 x(11) x(13)]
Trapezoidal	[x(11) x(12) x(14) x(15)]

Trapezoidal `[x(13) x(15) 0 inf]`

Common values can be easily predicted form table 15.2-15.4. So for a single variable we have to tune only five values and for three variables, a total of 15 values are to tuned. There are some constraints which should be considered while tuning these values.

Constraints:

1. Every value should satisfy the following inequality criteria
   x(1)<x(2)<x(3)<x(4)<x(5)
2. For input 'RE'
   0<x<1
   For input 'DNS'
   -1<x<1
   For output 'DNC'
   -1<x<1
   For these constraints the membership function values of fuzzy controller is changed and model is executed for these values.

15.3.3 Algorithm Steps

A step by step algorithm for the proposed work is given as:

STEP1. Initialize the node population random positions and directions of bacteria.

STEP2. Apply Kmeans clustering technique to make clusters of nodes and their centroids.

STEP3. Create an objective function which can calculate RE, DNS. DNC and also choose CH on the basis of RE and calculates mean RE of clusters and total node population.

STEP4. Create a Fuzzy Inference System FIS using Sugeno function for three inputs RE,DNS,DNC and make their membership function and rule set to decide output.

STEP5. Initialize random positions and directions of bacteria in BFO

STEP6. Consider the searching space dimension as number of membership function values to be tuned which is 15 in our case.

STEP7. Initialize the chemotactic, swarming, reproduction and dispersion steps. The initial step size of bacteria is taken as 0.005.

STEP8. In each chemotactic step, for every bacteria fitness function is and position of bacteria is updated by position updating formula. It is

$$new\ pos = old\ pos + step\ size \times \frac{direction}{\sqrt{direction * direction}}$$

STEP9. In swarming step the previous fitness function output is compared with the next position output of same bacteria. If found less then position of bacteria is updated again by formula given in step 5.

STEP10. The present position of bacteria is termed as current values of membership functions.

STEP11. The chemotactic and swarming loop continues till all initialized steps are completed. In each loop BFO updates the direction of bacteria and move the bacteria into the direction of fast convergence.

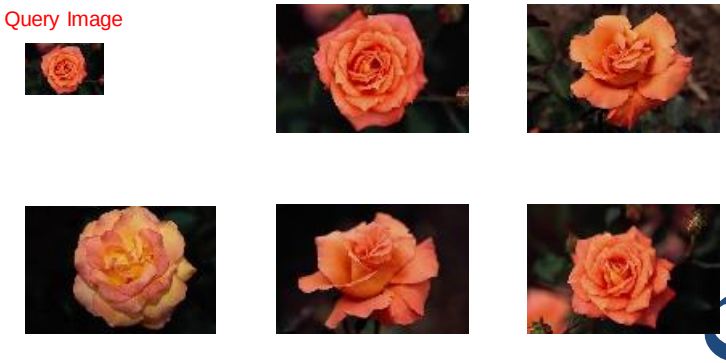
STEP12. Reproduction steps take place for bacteria with high fitness function values.

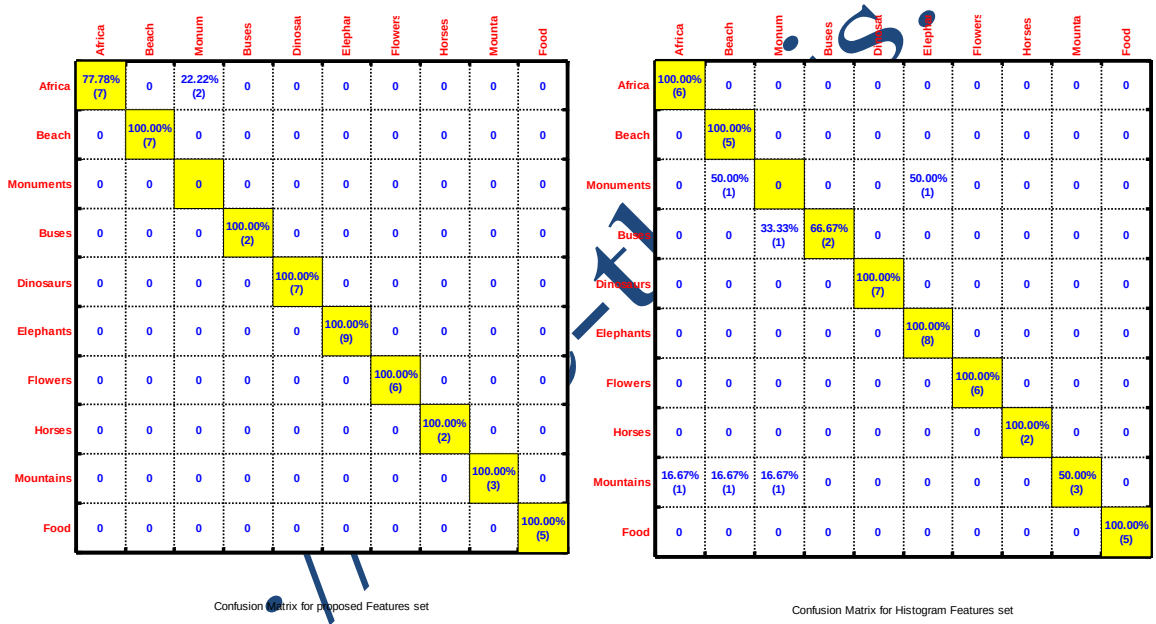
STEP13. To disperse or kill the weak bacteria, a probability of 0.25 is defined as the deciding probability. If random probability is higher than it, bacteria is dispersed or vice versa.

STEP14. Result will be positions of bacteria with minimum fitness function output. These positions are membership function's tuned variables for fuzzy logic controller.

Following these steps in optimization of BFO, optimal values of fuzzy controller membership function is achieved in our work.

Thesis-16
CBIR Based on Spatial, Temporal & Statistical Features





Abstract

Due to rapid increase in volume of image and video collections, traditional ways of classification and retrieval using solely keywords have become superannuated. Therefore, alternative strategies to describe images using their visual content are developed. Most of the existing image retrieval systems perform reasonably once using color features. However, retrieval accuracy using form or texture features will not produce nearly as good results. Therefore, this thesis investigates different strategies of representing shape and texture in content-based image retrieval. We have combined 5 features set in our work and these are trained and classified with SVM (support vector machine) classifier which makes use of machine learning technology. We combined histogram features, texture features (GLCM features), wavelet features, gabor features, and statistical features, which makes use of global

and local features. A database of a thousand pictures (Wang database) of ten completely different categories is used to extract all features vector for every image and keep in our database in order that SVM will use it to classify the query image. By using these features set, we square measure in a position to reach upto ninety seven.53% classification accuracy.

Proposed Work

This work of content based image retrieval is progressed in our thesis. We have focused on features based image classification followed by SVM machine learning algorithm. previously cascading of two or three features based method to enhance classification accuracy were used by researchers. Either they combined spatial features with color based features (discussed in chapter3) or color based features with statistical features. In our work we have combined four kind of features set to include every kind of feature in our feature vector. Color based features (histogram), statistical features (mean, median, standard deviation), spatial features (gabor feature and wavelet features) and texture features (GLCM).

In machine learning algorithms we need to have a database which will be used to train the system about type of inputs. In ours we have picked Wangle's database of 1000 images with 10 different classes of images. Features of each image using above defined features extraction methodology are extracted and combined to form a feature vector. Thus we created databases of features of all 1000 images each with 195 features after combining all. Whenever any image is queried, same set of features extraction is applied to that image and tested with SVM method which predicts the class form which image belongs. A flow chart for proposed work is shown in figure 16.1. Each feature set is discussed separately in next column.

Gabor features

Gabor based features are extracted by filtering the image form gabor filter which are statistical features. The impulse response of this filter is defined by a harmonic function multiplied by a Gaussian function. This filter is a linear filter. Because of the multiplication-convolution property (Convolution theorem), the Fourier transform of a Gabor filter's impulse response is the convolution of the Fourier transform of the harmonic function and the Fourier transform of the Gaussian function. These filters are directly related to Gabor wavelets.These filters can be designed for number of dilations and rotations. In our work we have used 6 orientations and 4 scales for gabor features filter. Since the orientation or angle for filtration is different so the out image will be different. An example for our test image is shown in figure 16.2.

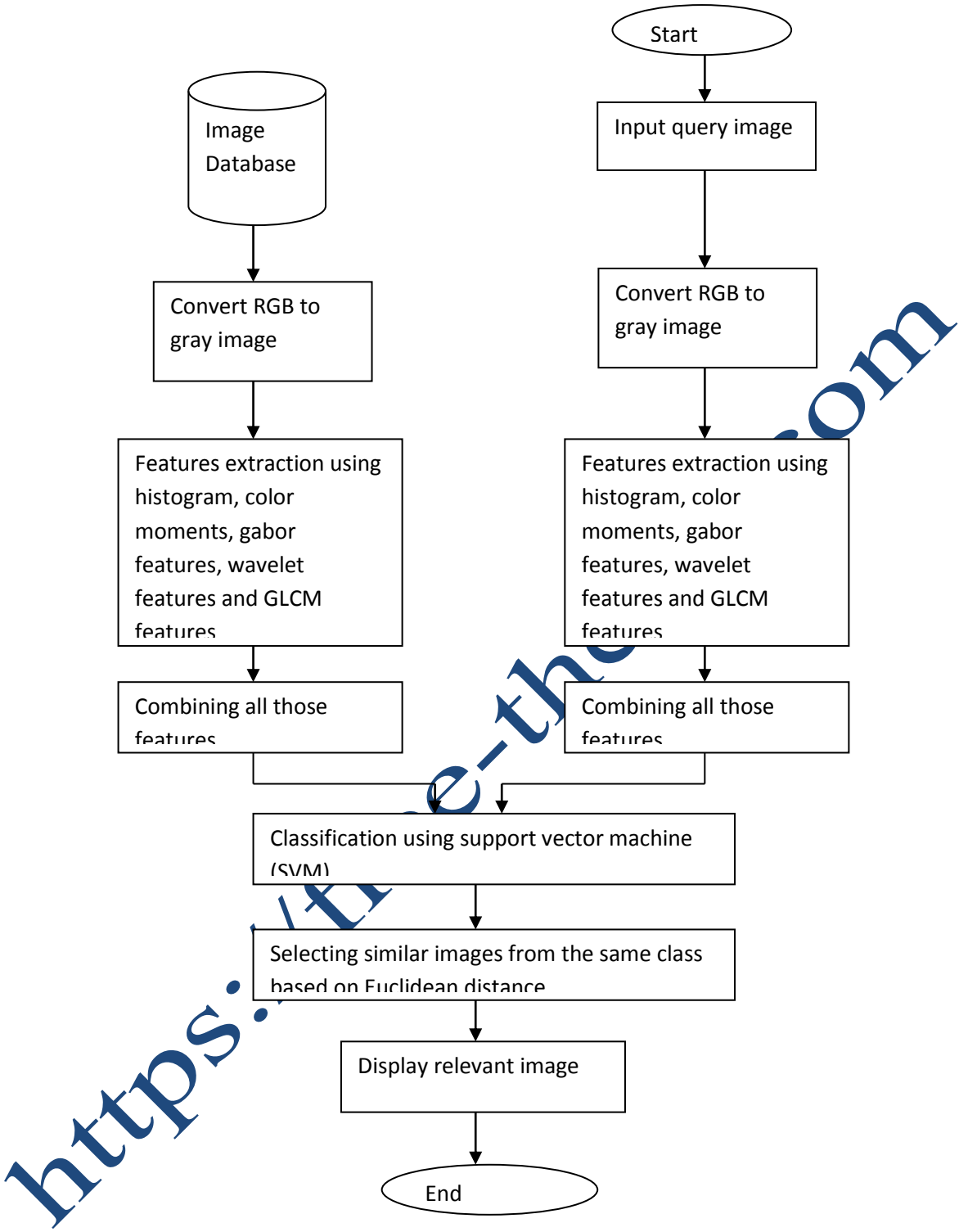


Figure 16.1: flow chart for proposed methodology

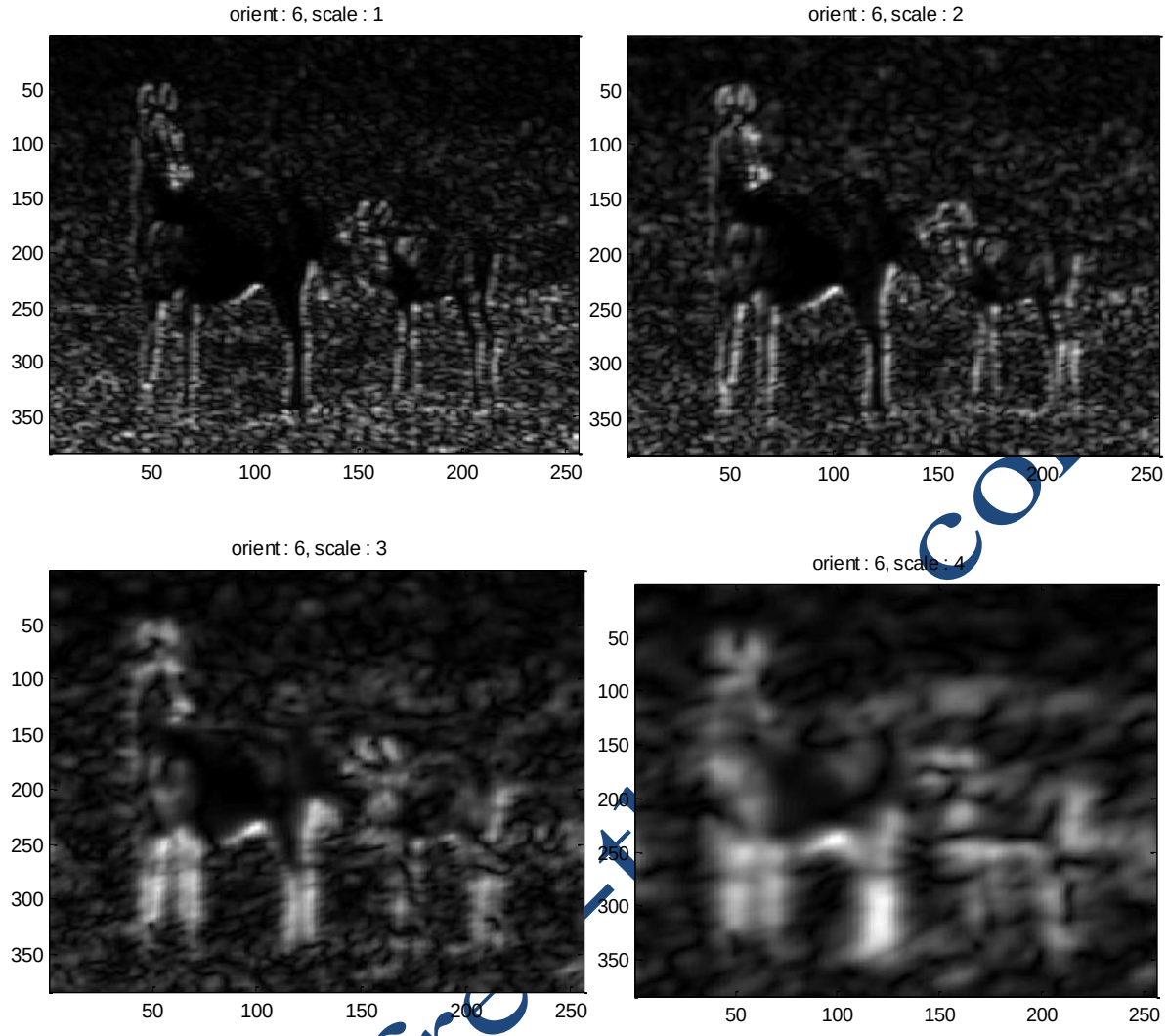


Figure 16.2: gabor features visualisation for 6th orientation for a test image

h(x, y) = s(x, y)g(x, y)

s(x, y): Complex sinusoid

g(x, y) : 2-D Gaussian shaped function, known as envelope

$$s(x,y) = e^{-j2\pi(\omega_0 x + v_o y)}, g(x,y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2})}$$

A complex Gabor function in space domain can be formulated as

g(x, y) = s(x, y)w_r (x, y)

where s(x, y) = complex sinusoid, known as the carrier,

w_r (x, y) = 2-D Gaussian-shaped function, known as the envelope.

The complex sinusoid carrier

The complex sinusoid is represented as,

$$s(x,y) = \exp(j(2\pi(u_o x + v_o y) + P)$$

Where (u_o, v_o) and P define the spatial frequency and the phase of the sinusoid respectively.

This sinusoid can be defined as two separate real functions, which are allocated in the real and imaginary part of a complex function. The real part and imaginary part of this sinusoid are

$$Re\big(s(x,y)\big) = \cos(2\pi(u_o x + v_o y) + P)$$

$$Im\big(s(x,y)\big) = \sin(2\pi(u_o x + v_o y) + P)$$

The parameter (u_o, v_o) defines the spatial frequency of the sinusoid in Cartesian coordinates. This spatial frequency can also be expressed in polar coordinates as magnitude

$$F_0 = \sqrt{u_o^2 + v_o^2}$$

$$\omega_0 = tan^{-1}(\frac{v_0}{u_0})$$

$$u_0 = F_0 cos\omega_0$$

$$v_0 = F_0 sin\omega_0$$

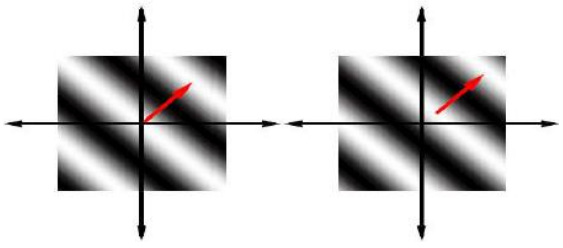


Figure. 16.3: The real and imaginary parts of a complex sinusoidal

The real and imaginary parts of a complex sinusoidal Using this representation, the complex sinusoid is

$$s(x,y) = \exp(j(2\pi F_0(xcos\omega_0 + ysin\omega_0) + P)$$

The Gaussian envelope

The Gaussian envelope looks as follows:

$$\omega_r(x,y) = K \exp(-\pi(a^2(x-x_0)_r^2 + b^2(y-y_0)_r^2))$$

where $(x_0,\ y_0)$ is the peak of the function , a and b are scaling parameters of the Gaussian, and the r subscript stands for a rotation operation such that

$$(x-x_0) = (x-x_0)cos\theta + (y-y_0)sin\theta$$

$$(y-y_0) = -(x-x_0)sin\theta + (y-y_0)cos\theta$$

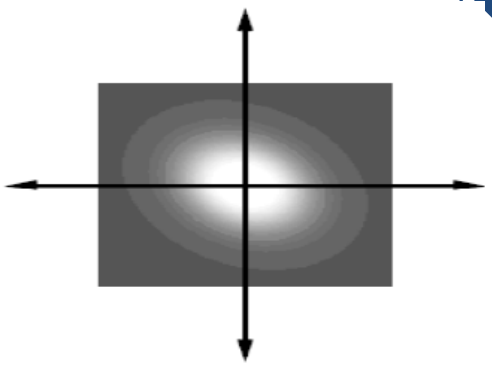


Figure. 16.4: A Gaussian envelope

The complex Gabor function

The complex Gabor function is defined by the following 9 parameters;

K : Scales the magnitude of the Gaussian envelope

(a,b) : Scale the two axis of the Gaussian envelope.

θ: Rotation angle of the Gaussian envelope.

$(x_0,\ y_0)$: Location of the peak of the Gaussian envelope.

(u_o, v_o): Spatial frequencies of the sinusoid carrier in Cartesian coordinates.

P: Phase of the sinusoid carrier

Each complex Gabor has two functions in quadrature (out of phase by 90 degrees), conveniently located in the real and imaginary parts of a complex function.

Now we have the complex Gabor function in space domain:

$$g(x,y) = K\exp(-\pi(a^2(x-x_0)_r^2 + b^2(y-y_0)_r^2))\exp(j(2\pi(u_o x + v_o y) + P)$$

Or in polar coordinates

$$g(x,y) = K\exp(-\pi(a^2(x-x_0)_r^2 + b^2(y-y_0)_r^2))\exp((2\pi F_0(xcos\omega_0 + ysin\omega_0) + P)$$

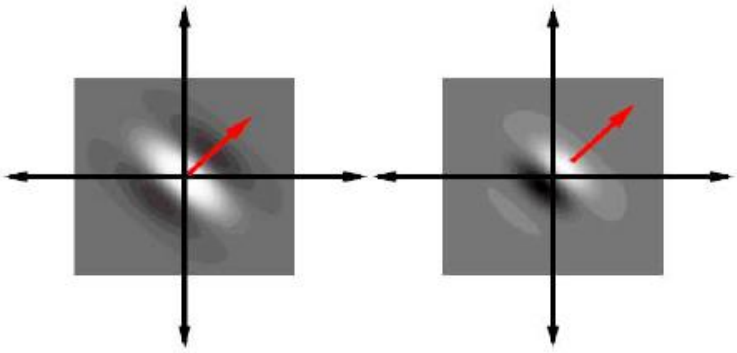


Figure. 16.5: The real and imaginary parts of a complex Gabor function in space domain.

The 2-D Fourier transform of this Gabor is as follows:

$$\hat{g}(u,v) = \frac{k}{ab}\exp(j(-2\pi(x_0((u-u_0) + y_0((v-v_0) + P))\exp(-\pi(\frac{(u-u_0)_r^2}{a^2} + \frac{(v-v_0)_r^2}{b^2}))$$

Texture features

Gray Level Co-occurrence Matrix (GLCM) a statistical methodology for examining texture features that take into account the spatial relationship of pixels, also famed as grey Level spatial Dependence. In this a GLCM matrix is made by calculating however typically a picture element with the intensity value i happens in a specific spatial relationship to a picture element with the value j. GLCM consists of frequencies at which 2 pixels are separated by a bound vector occur within the image.

The human visual system is more sensitive to distinction than absolute luminosity, we will understand the planet equally notwithstanding the massive changes in illumination over the

day or from place to place. Contrast is the distinction in luminosity or color that produces an object (or its illustration in a picture or display) distinguishable. In visual perception of the real world, contrast is determined by the distinction within the color and brightness of the thing and different objects inside an equivalent field of view. The distinction of image will be categorised as : international contrast and local contrast. Global distinction measures the brightness distinction between the darkest and brightest part in the entire image. Tools like Curves and Levels only modification international distinction as they treat all pixels with the same brightness levels identical.

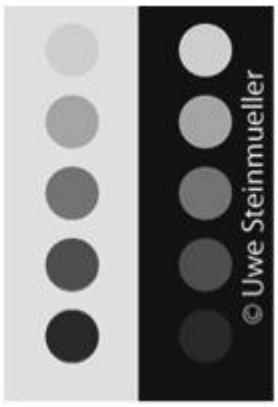
The global distinction has 3 main regions:

- Mid-tones
- Highlights
- Shadows

The sum of the contrast amounts of these 3 regions defines the global distinction. This means if you pay additional global contrast on the mid-tones (very ordinarily needed) you can spend less global contrast on highlights and shadows at any given international contrast level.

The mid-tones normally show the main subject. If the mid-tones show low contrast the image lacks “snap”. Adding more international contrast to the mid-tones (“snap”) typically results in compressed shadows and highlights. Adding some local distinction will facilitate to improve the general image presentation.

The local distinction is based mostly on the retinex theory consistent with that our eyes sees the difference in relevance surroundings, a color map below can prove this purpose.



The circles in each row have specifically the identical brightness levels. Yet the prime right circle appearance plenty brighter than the one on the left. This is because our eyes see the distinction to the native close. The right circle looks a lot of brighter with the dark grey background compared to a brighter background on the left. Just the opposite is true for the 2 circles on very cheap. For our eyes the absolute brightness is of less importance than the relative relevancy other shut areas. So, local distinction is terribly necessary for process or sweetening of any image.

In our work because of this human visual system local contrast map is extracted from an image and then on the basis of that a local thresholding approach will be used to convert the image onto binary format. Previously image gradient and normalize image gradient were used to extract local contrast of image, these methods are quite good, although the variation of bright to weak contrast can be compensated by these methods yet these don't perform well in case of document which have bright text. This is because a weak contrast will be calculated for stroke edges of the bright text. Calculation of local contrast and then global thresholding algorithm like otsu is used and then local image edge detection is used in paper published by Bolan Su (2013). We have followed the same line of action but rather than using global thresholding, we use local thresholding, it removes the need of using again local edge detection algorithm like canny edge detection. Gray level co –occurrence matrix (GLCM) also called texton co- occurrence matrix (TCM) fulfills our purpose. It is a local contrast mapping method. Here basically TCM serves two purposes: make image's local contrast map, unaffected by the illumination variation of image and local edge detection. Further, the GLCM functions characterize the texture of a picture by calculating however typically pairs of pixel with specific values and in a given abstraction relationship occur in a picture, creating a GLCM, and then extracting statistical measures from this matrix. A gray-level co-occurrence matrix (GLCM) is generated by calculating however typically a pixel with the intensity (gray-level) value i happens in a specific abstraction relationship to a pixel with the value j. By default, the spatial relationship is outlined as the element of interest and also the element to its immediate right (horizontally adjacent), but you will specify alternative abstraction relationships between the two pixels. Each component (i,j) within the resultant glcm is solely the sum of the amount of times that the pixel with value ioccurred within the given abstraction relationship to a element with price j in the input image. The number of grey levels within the image determines the dimensions of the GLCM. GLCM of an image is computed employing a displacement vector d, defined by its radius δ and orientation θ. To

illustrate, the following figure shows how graycomatrix calculates the primary 3 values in an exceedingly GLCM. In the output GLCM, element (1,1) contains the value one as a result of there's only one instance within the input image wherever 2 horizontally adjacent pixels have the values one and 1, respectively. glcm(1,2) contains the value two as a result of there area unit 2 instances wherever 2 horizontally adjacent pixels have the values one and a pair of. Element (1,3) in the GLCM has the worth 0 as a result of there are not any instances of 2 horizontally adjacent pixels with the values one and three. graycomatrix continues processing the input image, scanning the image for other element pairs (i,j) and recording the sums in the corresponding elements of the GLCM. Figure 16.6 shows this thought.

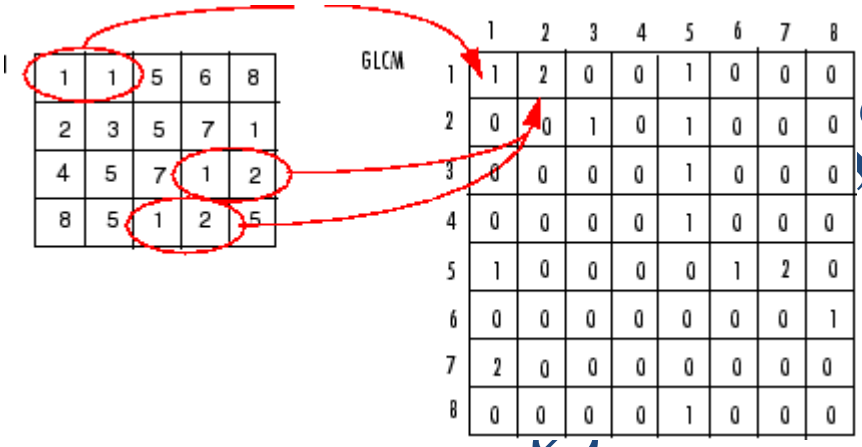


Figure 16.6: GLCM output of a test matrix.

A single GLCM matrix might not able to define all texture features of image, so multiple GLCM at different orientations are calculated. Above given example was with 0^{o} orientation i.e. horizontally matching pairs are checked. Further it can be done at angle 45^{o}, 90^{o}, 135^{o} as shown in figure 15.7.

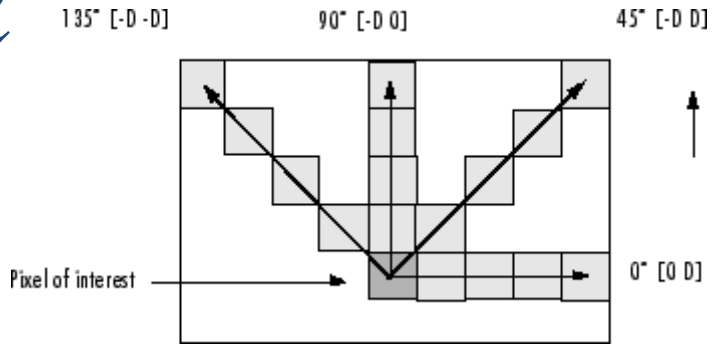


Figure 16.7: Multiple orientations to calculate GLCM

In actual every picture element has eight neighboring pixels permitting eight decisions for θ, which square measure 0°, 45°, 90°, 135°, 180°, 225°, 270° or 315°. However, taking into consideration the definition of GLCM, the co-occurring pairs obtained by selecting θ up to 0° would be the same as those obtained by choosing θ equal to 180°. This concept extends to 45°, 90° and 135° as well. Hence, one has four choices to choose the worth of θ.

Above we have a tendency to have mentioned a term radius concerning GLCM. In the last example matching pairs are taken upto one distance, this constitutes the radius of GLCM. Various analysis studies show δ values move from one, 2 to 10. Applying large displacement worth to a fine texture would yield a GLCM that will not capture elaborate textural data. From the previous studies, it has been concluded that overall classification accuracies with δ = one, 2, 4, 8 square measure acceptable with the best results for δ = one and a pair of. This conclusion is justified, as a picture element is additional probably to be related to alternative closely placed pixel than the one placed so much away. Also, displacement value equal to the dimensions of the feel component improves classification.

The dimension of a GLCM is determined by the utmost gray worth of the picture element. Number of grey levels is an necessary think about GLCM computation. More levels would mean additional correct extracted textural data, with increased machine prices. The computational quality of GLCM technique is extremely sensitive to the quantity of grey levels. As in above example in figure four.1, the size of GLCM is eight by eight matrix as 8 grey levels are thought of. Thus for a preset worth of G, a GLCM is required for every distinctive combine of δ and θ. GLCM is a second-order texture measure. The GLCM's lower left triangular matrix is forever a mirrored image of the higher right triangular matrix and also the diagonal always contains even numbers. Another test matrix and its GLCM at all four rotations for radius one are shown in figure 16.8.

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

Figure 4.8(a) test matrix

4	2	1	0
2	4	0	0
1	0	6	1
0	0	1	2

Figure 4.8(b) test matrix with angle 0

6	0	2	0
0	4	2	0
2	2	2	2
0	0	2	0

Figure 4.8(c) test matrix with angle 45

4	1	0	0
1	2	2	0
0	2	4	1
0	0	1	0

Figure 4.8(d) test matrix with angle 90

2	1	3	0
1	2	1	0
3	1	0	2
0	0	2	0

Figure 4.8(e) test matrix with angle 135

Various GLCM parameters are connected to specific first-order statistical ideas. For instance, contrast would mean pixel pair repetition rate, variance would mean spatial frequency detection etc. Association of a textural meaning to every of those parameters is incredibly vital. Traditionally, GLCM is dimensioned to the number of grey levels G and stores the co-occurrence probabilities gij. To determine the texture features, selected statistics are applied to every GLCM by iterating through the whole matrix. The textural features are based mostly on statistics that summarize the ratio distribution that describes how usually one grey tone will seem in a very given abstraction relationship to a different grey tone on the image.

Following notations are used to justify the assorted textural features:

gij = (i, j)th entry in GLCM

gx(i) = ith entry in marginal probability matrix obtained by summing rows of gij=

Ng = range of distinct grey levels in the image

gy(i) = ∑_(i=1)^(N_g)▒〖g(i,j)〗

Contrast (con) = ∑_i▒∑_j▒〖〖(i-j)〗^2 g_(i,j) 〗

This statistic measures the abstraction frequency of an image and is distinction moment of GLCM. It is the difference between the best and therefore the lowest values of a contiguous set of pixels. It measures the amount of native variations present within the image. A low contrast image presents GLCM concentration term round the diagonal and options low abstraction frequencies.

Wavelet Transform

Wavelet transform is another spatial feature set. In it small parts of test image are extracted by wavelet filters, and mean and standard deviation of these filtered images are considered as features in our work.

Wavelets are functions generated from one single function (basis function) called the prototype or mother wavelet by dilations (scalings) and translations (shifts) in time (frequency) domain.

If the mother wavelet is denoted by y (t), the other wavelets $\varphi_{a,b}(t)$ can be represented as

$$\varphi_{a,b}(t) = (1 * \varphi(\left(\frac{t-b}{a}\right)/\sqrt{a})$$

where a and b are two arbitrary real numbers. The variables 'a' and 'b' represent the parameters for dilations and translations respectively in the time axis. The mother wavelet can be essentially represented as

$$\varphi_{a,b}(t) = \varphi_{1,0}(t)$$

For any arbitrary a ≠1 and b = 0, we can derive that

$$\varphi_{a,0}(t) = (1 * \varphi(\left(\frac{t}{a}\right)/\sqrt{a})$$

As shown above, $\varphi_{a,0}(t)$ is nothing but a time-scaled (by a) and amplitude-scaled (by a) version of the mother wavelet function y (t).

Decomposition Process

To start with, the image is high and low-pass filtered along the rows and the results of each filter are down- sampled by two. Those two sub-signals correspond to the high and low frequency components along the rows and are each of size N by N/2. Then each of these sub-signals is again high and low-pass filtered, along the column data. The results are again down-sampled by two.

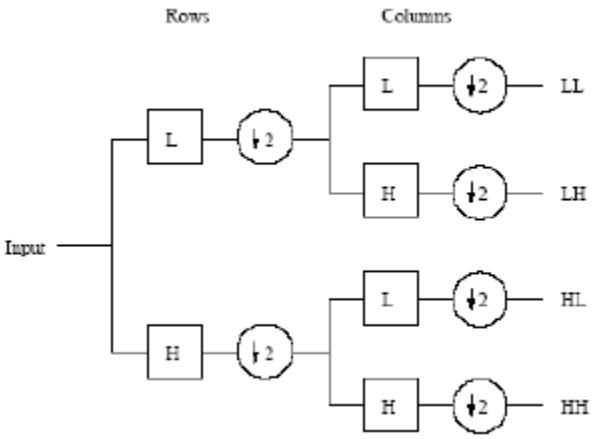


Figure 16.9 One decomposition step of the two dimensional image

As a result the original data is split into four sub-images each of size N/2 by N/2 containing information from different frequency components.

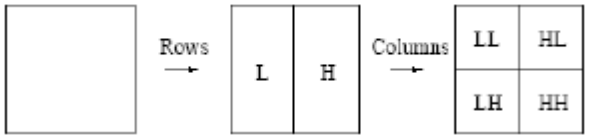


Figure 16.10 One DWT decomposition step

The LL subband is the result of low-pass filtering both the rows and columns and it contains a rough description of the image as such. Hence, the LL subband is also called the approximation subband. The HH subband is high-pass filtered in both directions and contains the high-frequency components along the diagonals as well. The HL and LH images are the result of low-pass filtering in one direction and high-pass filtering in another direction. LH contains mostly the vertical detail information that corresponds to horizontal edges. HL represents the horizontal detail information from the vertical edges. All three subbands HL, LH and HH are called the detail subbands, because they add the high-frequency detail to the approximation image.Further to construct the feature vectors in our case we take mean and standard deviations of these sub-bands are calculated and saved in a matrix of single row.

Note:

1. **This manuscript is only to provide the guidance to the researcher.**
2. **Content is not plagiarism free.**
3. **Only provided the brief discussion related to the thesis.**