Innovating Based on R tree and Artificial Neural Network for Hierarchical Clustering in Order to Make QoS Routes in MANET

Nguyen Thanh Long[1(✉)], Nguyen Duc Thuy[2], and Pham Huy Hoang[3]

[1] Software Development Division III, Informatics Center of Hanoi Telecommunications, Hoan Kiem, Hanoi, Vietnam
Ntlptpml@yahoo.com
[2] Center for Applied Research and Technology Development, Research Institute of Posts and Telecommunications, Hanoi, Vietnam
Nguyenducthuy07@gmail.com
[3] Information Technology Institute, Hanoi University of Science Technology, Hanoi, Vietnam
Hoangph@soict.hut.edu.vn

Abstract. The advanced routing protocol not only operates on lower levels of a network protocol, but it also operates on upper layers such as the application layer of OSI model. The routing task can be operated on a wider scale. It can process based on results of some other protocols for example service based protocol can be operated based on the service discovery protocol. So this kind of routing protocols may be determinized as upper layer routing protocols. Such as the service based routing protocol can operate based on content based protocol and combines some service filters. In service based routing protocol as well as content based routing protocol, subscriber and publisher can communicate with each other but they don't know the other's address. So it is more flexible in processing and more comfortable for mobile ad-hoc network. In mobile ad-hoc networks, nodes usually move, so bandwidth of connection between them may be not stable. Therefore transmission delay, overhead and packet loss may be larger than other kinds of networks. The paper aims at purpose to increase QoS of routing by hierarchical clustering routing by using R^+ tree in addition with some advanced techniques such as multicast routing, multiple paths, use can ACO to optimize routes to transmit data. By using R tree structure, the network topology are managed by bottom-up model from leaf level to root of the tree. All the leaf nodes, inner nodes and root of this tree have two roles: (i) Manage a cluster that consisting all nodes that have direct connections with this node; (ii) Operate as a normal node. The paper introduces and analyzes: (i) Establish hierarchical clustering network by using R tree structure; (ii) Make multicast tree from some cluster heads for fast routing; (iii) Make optimized route by Ant Colony Optimization. The paper also uses the Artificial Neural Network to choose optimal cluster head and members for cluster of network.

Keywords: MANET · R^+ · Service · Routing · Multi-paths · Bandwidth · Cluster · Tree · Multicast · QoS · Overhead · Ant · ACO · ANN

T.F. Bissyande and O. Sie (Eds.): AFRICOMM 2016, LNICST 208, pp. 217–231, 2018.
https://doi.org/10.1007/978-3-319-66742-3_21

1 Introduction

1.1 History of R tree Applications

R-tree are tree data structure used for spatial access methods, for example, it can used for indexing multi-dimensional information such as geographical coordinates, rectangles or polygons [12]. The R-tree was invented by Guttman in 1984, focuses on handling geometrical data, such as points, line segments, surfaces, volumes, and hyper volumes in high-dimensional spaces [11]. R-trees are accepted as an additional access method to process multi-dimensional data. Today, spatial databases and geographical information systems have been established as a mature field, spatiotemporal databases and manipulation of moving points and trajectories are being studied extensively, and finally image and multimedia databases able to handle new kinds of data, such as images, voice, music, or video, are being designed and developed. An application in all these cases should rely on R-trees as a necessary tool for data storage and retrieval.

1.2 Main Features of R tree

The key idea of the data structure is to group nearby objects and represent them with their minimum in the next higher level of the tree; the "R" in R-tree is for rectangle. Since all objects lie within this bounding rectangle, a query that does not intersect the bounding rectangle also cannot intersect any of the contained objects. At the leaf level, each rectangle describes a single object; at higher levels the aggregation of an increasing number of objects. This can also be seen as an increasingly coarse approximation of the data set [12].

R tree has a typical feature that is each leaf/inner/root node has number of children in the predefined range [m, M].

In this paper, we may utilize the R^+ to organize the network topology in hierarchical clustering model. The procedures for processing R^+ tree is already defined in some previous papers. The R^+ can be expressed by the recursive formula:

$$R^+ = \text{Root} \cup R^+(C_1) \cup R^+(C_2) \cup \ldots \cup R^+(C_n). \quad (1)$$

The root of the R^+ tree has n children: $C_1, C_2, \ldots, C_n$. In which, every C_i is a child that has direct link to the root of R^+. In R^+ tree, every inner and leaf node can connect directly to a rather stable number of children that is belonged to predefined range [m, M], m, M are lower and upper bound of this range respectively. The lower bound m is usually belonged to the range $(0, \lfloor M/2 \rfloor]$. When number of nodes is increased and the overhead also increases very fast. The normal topologies are not effectively, they must be innovated by some artificial algorithms. For example, OLSR protocol uses MPR to reduce overhead, control packet flooding and route loop. In the recent time the multicast topology is usually used to reduce delay and transmit packet to all destinations at the same time. Especially, on a larger network we may use hierarchical clustering model to organize the network topology to reduce overhead and easily to apply artificial algorithms to find optimal routes to control packets and transmit data. In the R^+ tree, the network nodes in a level and are managed by a local root that are called

a cluster. The network of some cluster heads may cooperate to establish an upper cluster. In that way, the root node manages to highest level cluster of the network.

$$\text{Upper(Cluster)} = \text{NETWORK(U}(CH_i)) = CH_1 \,\text{U}\, CH_2 \,\text{U}\, CH_3 \,\text{U}\ldots\text{U}\, CH_n \quad (2)$$

This NETWORK(U(CH_i)) is established that is called the upper level cluster, as lower level cluster that choose a node to manage this cluster which is called cluster head. In this way, up to the root we have the root cluster that has root of tree is the cluster head of the highest level cluster of the whole network. At the first time, the root cluster has only three nodes, the root of tree that manages two cluster heads of two one level lower clusters that are just established by dividing original cluster into two parts when the number of nodes of this cluster reaches M. Therefore, it is easy to realize that in the leaf level, there are some basic clusters, in which each cluster has some nodes that are only managed by a cluster head; they don't manage any other node. The cluster head of each leaf cluster is a leaf node of the R tree. The member nodes communicate with their cluster head for receiving and sending data and control information in this cluster and from outside networks. In R tree, we define a level of the tree is collection of nodes that have the same height to the root of the tree. The zero level consists of nodes are managed by leaf nodes, the first level contains leaf nodes, the root level contains only root of the tree. The number of levels of tree is equal to the height of tree plus one. Therefore, it is easy to realize that the root of an upper level cluster manages all nodes of all branches that connect to this local root. Then it is also the cluster head of this upper level of some cluster heads of one level lower some clusters.

2 Use R^+ tree for Hierarchical Cluster Routing

The algorithms are used for making, updating, deleting, regulating, cluster dividing … of R^+ tree is introduced in previous papers. This paper introduces technical specification details on using this tree to make hierarchical clustered routing. R tree can be defined by a four items tuple:

$$\text{R} = \{\text{m},\ \text{M},\ \text{root},\ \{\text{F}\}\} \quad (3)$$

{F} is the set of functions to insert, update, delete and regulate R tree.

The root often transmits control messages to their children to coordinate its cluster. If root stops then the tree will be must to renew. In that way these child members also manage their clusters by periodically sending controlling messages. So the overhead for transmitting one control message is:

$$\text{Overhead(Control)} = \prod_{i=1}^{h} n_i \quad (4)$$

It is approximate overhead to transmit a control packet to each node in the network. In that h is number of levels, n_1 is average number of nodes of each cluster in level 1 of R^+ tree, n_2 is average number of children nodes of each cluster in level 2…, level h has only root node. As mention above, $n_i \in$ [m, M], with one level of the tree, h is usually

not large because number of nodes of the tree is less than M^{h-1}. For example, with network, choose m = 5, number of nodes is 100000, h is less than or equal to 7. Especially controlling messages are sent simultaneously from a cluster head to all its child nodes. So its delay is reduced and all managed nodes can receive at one time, satisfies the QoS request. Beside requested bandwidth is reduced significantly because at one time all child nodes can receive a control packet from their cluster head. Cluster head can transmit data by multicast protocol, it can decrease flood of data over the network. When a node is covered by a cluster head with satisfied bandwidth, it may join this cluster by receiving one invitation message of this cluster head. Then all of this network local change is spread over network locally in its managed region by multicast tree of combination of some clusters to reduce packet loss. At first, this control message (M) is transmitted to the cluster head (CH) of parent cluster (PC) of the cluster head of current cluster. The cluster head of PC transmits this M to all members of PC and to CH of its cluster. By this way, any change of network configuration is very fast to be notified to all CHs. On the other hand, all child nodes of network leaf nodes or managed nodes don't need to receive this information, for example, some network policy information only are needed by cluster heads, so it reduces much of required bandwidth.

Each cluster has to store two main kinds of connections: (i) one to the cluster head of cluster that it is belonged; (ii) one to all its cluster members. This number of connections is approximate to: $n_i + 1$.

The algorithm for choosing cluster head as introduced by our previous papers [1–7]. In fact, a node may be accepted to some clusters simultaneously because all connections to these clusters also satisfy bandwidth request. So it may be reasonable to use an ANN for choosing the best cluster to accept this node. Exceptionally, at the time to join network with very high density of nodes. For example, many upper levels of clusters may accept this node at a very short time duration. If this node stores all connections to these cluster heads then the time to update network topology is reduced by multicast tree from this node to all these CHs. In this way the root of R^+ can be connected to some another trees easily. It establishes a daisy chain R^+ trees. Because this model of networks can avoid the constraint of [m, M] in each level of R^+.

3 Find the Optimized Route by Multiple Paths and Multicast Routing

The temporary backbone of network of R^+ daisy chain can be established, so some multicast trees or meshes of cluster heads of some clusters of this network can established for data transmission. Therefore it overcomes constraint of R^+ structure in some critical circumstances to directly transmitting data from a source node to many destinations. This concept is very comfortable on upper layer routing protocol in advanced networks. The algorithms to establish multiple optimal multicast trees and optimal multiple paths routes that are found by GEN/ACO are introduced in [3, 4].

4 Acceptance of a Node

As any normal wireless routing protocol, when a node wants to join a network, it has to create HELLO packet to broadcast over network. When a cluster head receives this packet, it checks some context aware conditions and node's capabilities. In a general circumstance, these conditions are node's energy, strength of signal, processing capacity, CPU usage, and network bandwidth. CH will make a join invitation to send to this node. This node will send this CH its authentication information to join this cluster. As it is introduced in [6], this node information is processed and it will be propagate to a basically cluster from this cluster. When the best congruous cluster accepts this node, all needed changes will be processed on this cluster to propagate to the root of R^+ tree, after that the changes of network are transmitted to all CH in the network. Because every CH also receives these changes so it can manage its cluster better. So the new node may be cluster head based on its capability and the procedures for accepting this node as the new cluster head of this cluster is carried out. Hence at first it will be cluster of current cluster. If new node is chosen as CH then this node's information is processed by upper level clusters. This process is executed until the new node will not be accepted as CH on any upper cluster. The some cluster heads with the same level with this cluster head will vote new cluster head for the next level of clusters. The new node may be root of R^+ tree, if it satisfies the conditions.

4.1 Procedure for Accepting Node

When a node is invited to join R^+ tree or the root of an R^+ that can be a branch of the large R^+ to receive the HELLO message of this node. The node's information is spread back to the root of the R^+ tree. The node's information is processed that begins from root node to choose the best child node to transmit node information to. The child node continuously processes node information to choose the best grandchild to relay node information to. This process executes continuously until it reaches leaf node. The leaf node processes the node information by the same algorithms or any optimal algorithm such as an artificial algorithm. If the node is accepted by this CH to join its cluster then this cluster has to regulate to follow R tree structure. The regulation processing begins from current CH up to the root, until there is no change to the structure or cluster head of the processing cluster. The algorithm is used to evaluate a node for a CH to accept a node based on some context conditions and candidate node's information. For example, it can use fuzzy logic controller to decide, these parameters of candidate node are fuzzed to be put into a fuzzy logic controller to get the fuzzed output result to make the decision. Besides, it may use ANN to process the parameters of node with condition is that ANN has been trained to choose the best result.

4.2 Procedure for Choosing Cluster Head

The procedure is used to choose a cluster head that depends on some metrics of nodes in each cluster. After a cluster is formed, it have to choose a node that has strong capacity and/or many good characteristics and fit best to context conditions to manage

the cluster that is called cluster head. Then we can use a neuron network or fuzzy logic controller to choose the cluster head or a range of several cluster heads. After that, the network established by found cluster heads of R^+ tree is optimized by GENETIC algorithm or some other algorithms.

5 Artificial Neural Network

5.1 Basic Concepts

The Artificial Neural Network (ANN) has some layers, which are input layer, hidden layers and output layer (Fig. 1).

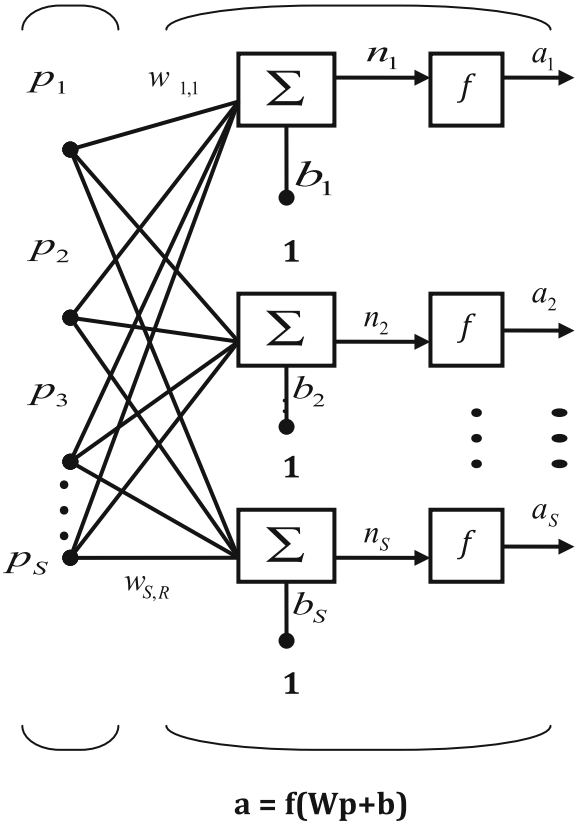

Fig. 1. ANN model with one layer.

The input data is executed by the input layer, each data of input is received by a neuron of input layer (Fig. 2).

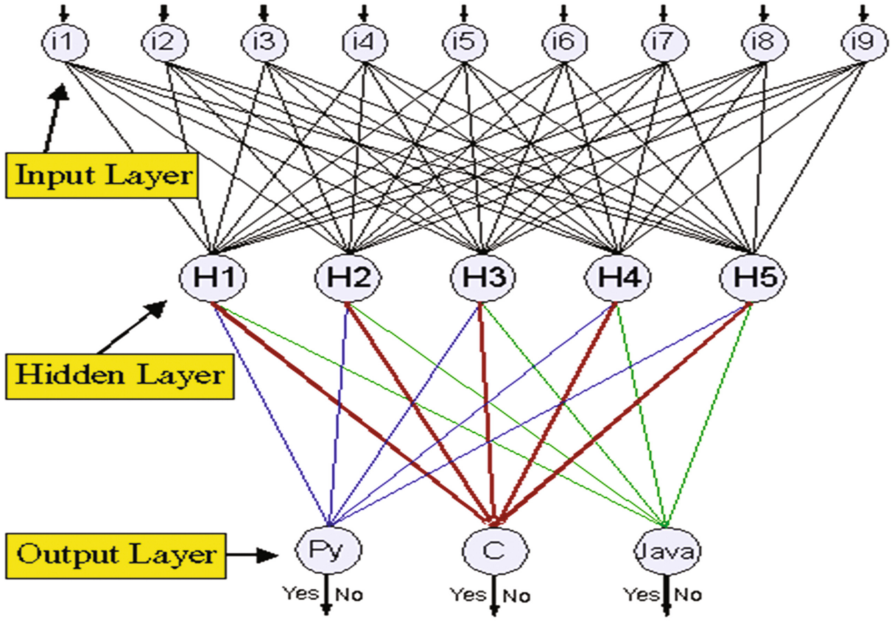

Fig. 2. ANN model with multiple layers.

These data of neurons of input layer are the inputs of the first hidden layer; each input is multiplied with a weight value. So each neuron stores a weight vector, we get the sum of weighted input data with a bias of each layer to get a sum:

$$S = \sum_{i=1}^{n} I_i W_i + B \tag{5}$$

The transfer function is used for this sum to get a result of each neuron:

$$R = \text{Transfer Function}(S) \tag{6}$$

The results of each hidden layer are inputs for the next hidden layer. Finally, the results of the last hidden layer are inputs for the output layer. The output layer calculates the output vector for the neural network. In order to tune the result of the neural network in the training stage for the ANN, We calculate the error vector by subtracting the desired vector to result vector. We use the error vector to tune the weight vector of hidden layers individually from the last hidden layer to the first hidden layer. After some predefined steps or the output vector is converged we stop the training process to get neural network to use for clustering network as the next section will introduce. We must use a transfer function that can be derived. So we can use derivation of the

transfer function to tune weight vector. The formula for updating weight vector of each neuron in output layer is:

$$w_{ij}(1+1) = w_{ij}(1) + \eta.e_i(1).y_j.\mathrm{f}'(y_i(1)) \tag{7}$$

In which: η: training coefficient,

e_i(l): deviation value of ith neuron in output layer in lst round of processing.
f′: derivation of the transfer function, $\mathrm{f}' = \frac{1-x^2}{2}$.
y_i(l): output value of ith neuron in the output layer in the lst round of processing.
y_j(l): output value of jth neuron of hidden layer in the lst round of processing.

The formula for updating weight vector of each neuron in the hidden layer is:

$$w_{ij}(1+1) = w_{ij}(1) + \eta.\mathrm{x_j}.\mathrm{f}'(\mathrm{y_i}).\sum_{k=1}^{m} w_{ki}(l+1).e_k(l).f'(y_k(l)) \tag{8}$$

In which: η: training coefficient,

x_j: output value of j^{th} neuron of the input layer.
y_i: output value of i^{th} neuron in the hidden layer.
w_{ki}(l + 1): weight value of the connection between the k^{th} neuron in the output layer and the i^{th} neuron in hidden layer in the $(l+1)^{th}$ round of processing.
y_k(l): output value of the kth neuron of output layer in the l^{th} round of processing.

5.2 Apply Artificial Neural Network to Cluster Network

The network is clustered by two processes stages: the first step is for choosing a cluster head for each cluster. The second stage is for choosing cluster members for each cluster. We use ANN concepts as analyzing above for training an ANN for each cluster head. After that we use found ANNs for accepting cluster member for each cluster. We use back propagation algorithm to update weight matrix of ANN. The algorithm consist of two steps: (i) At the first step: starting from output layer, the output value (OVE) error is used to update weight matrix of the previous layers:

$$[\mathrm{OVE}] = [\text{Desired Output Value}] - [\text{Calculated Output Result}] \tag{9}$$

We calulate the delta for each neuron in hidden layers and input layer by summing the multiple of weight vector of the current neuron and delta of each neuron in the next layer:

$$\mathrm{Delta_j^i} = \sum\nolimits_{\mathrm{k}=1}^{\mathrm{n}} \mathrm{w_k^{ij}} \mathrm{Delta_k^{i+1}} \tag{10}$$

In which: Delta is the deviation between desired result and real got result, $Delta_k^{i+1}$ is delta of neuron k in the layer i + 1, w_k^{ij} is the weight kth in the weight matrix of neuron j of layer i; (ii) At the second step, we start at the input layer to the last hidden layer. The weight matrix of each neuron of each layer is added by summing the multiplication of three items: the learning rate of the current neuron, the delta of respective neuron with current weight in the next layer and result of the neuron:

$$w_k^{ij} = w_k^{ij} + \sum\nolimits_{k=1}^{n} LR_i^j * Delta_k^{i+1} * NR_j^i \tag{11}$$

In this formula, n is number of neurons in the next layer i + 1, w_k^{ij} is the kth weight in of weight vector of the neuron j in the layer i, LR_i^j is learning rate of this neuron, $Delta_k^{i+1}$ is delta of the neuron k of the layer i + 1, NR_j^i is result of this neuron. We also update weight vector for bias in each layer:

$$w_k^i = w_k^i + \sum\nolimits_{k=1}^{n} B_i * Delta_k^i * LR_k^i \tag{12}$$

In which, B_i is bias of the layer i, n is number of neurons of this layer. $Delta_k^i$ is delta of the neuron k^{th} of the layer i, LR_k^i is learning rate of the neuron k in the layer i.

5.3 Use the Genetic Algorithm to Innovate the Weight Updating Algorithm

In order to update weight matrices of layers of ANN fast, we use the Genetic (GEN) algorithm. In particular, we use the crossover operator to update weight matrices of ANN. The crossover operator is:

We assume each neuron network after training is a solution in the population. We interchange weight vector of neurons between two networks randomly, with condition, two neurons have the same position in each network (the layers and position in each layer). We change weights for each neuron respectively. By simulations, we see that the training process is improved with time required is reduced.

6 Selection of QoS Route by ACO

The paper introduces an algorithm to find QOS routes based on ACO which consists of two phases: (i) route discovery; (ii) route maintenance. In the discovery phase, it uses two kinds of packets that like route request and route reply as used in DSR protocol which are Ant_Route_Request and Ant_Route_Reply packets. Some QOS demands are stored in Ant_Route_Request in combination with visited node's list field, this field stores nodes that are visited by this packet.

6.1 Introduction

Pheromone evaporation is some condition to assess the probability to choose the optimal route. On the route to find food, ants emit pheromone evaporations to sign routes which are traversed. So other ants easy to find and focus to the food source very fast.

6.2 Discovery Process

The Ant_Route_Request is broadcasted to 1-hop neighbors of the source node like a HELLO packet of OLSR. In each neighbor node: (i) node has to maintain its link quality table $\{L_{i\to j}\}$ of 1-hop distance nodes based on the accessing pheromone evaporation, it calculates the preference probability (P_{pref}) of each link based on information of routing table on this node. If P_{pref} is more than predefined threshold preference probability, then this link is selected for forwarding Ant_Route_Request on the network; (ii) Select a 1-hop distance node among 1-hop distance nodes that is MPR, that has connections to almost 2-hop distance nodes of this node. So in selecting MPR being next hop on the route to the destination. Obvious MPR functions as source node, it carries out the same functions: broadcast Ant_Route_Request to its 1-hop distance nodes. When packet for finding a route arrives at a destination node. The destination node instantly makes Ant_Route_Reply packet to transmit back to the source node. The Ant_Route_Reply stores the route that has just been detected by the unicast protocol.

6.3 Maintenance Process

Some routes may be discovered by discovery process, but we select the optimal route by Genetic algorithm to transmit data. The other routes may be cached in the buffer of the source node for multiple path routing purpose; they may be used for alternating or replacing route in the case of main route hasn't existed because of topology changes. But these routes are periodically updated their status for checking existence state.

7 Hidden Markov Model

We assume a node in a hierarchical network can belong to N clusters in M times. In each time t, this node can be in cluster i (i = 1..N) with probability P_i. The purpose is to find the route that passes through some member nodes which belonged to these clusters. In which each node is belonged to one cluster. The host can be the member node of one of N clusters. Applying the HMM to this problem:

N states are N cluster heads of N clusters that are established before. In which each cluster (i) connects to some cluster member (j) with some probability existence that is called A_{ij};

M outputs are M node members of N clusters that are hops on finding route from the source S. So the finding route can be denoted by:

$$R = (S, R_1, R_2, \ldots, R_{m-2}, D);$$

A node of this hierarchical cluster network that needs to find routes R that passes through M nodes. At each step we have to find a cluster that has connection to recent selected cluster. This may be known as state transition. At each state i or cluster C_i which must to choose one or more cluster members connected this cluster which are belonged to the finding route.

From these assumptions, we establish a HMM with probabilities for transitions and connection choosing between nodes are calculated by ACO as mentioned above. The transition probability is the probability of connection between two cluster heads of this network is also defined by ACO. So applying HMM formulas that are easy to get the probability to select a route from these member nodes [10].

We define probability model of HMM model system:

Transition probability from state i (cluster i) to state j (cluster j) is probability of existance of connection between cluster i and j $(i, j \in \overline{1..N})$:

$$P_{ij} = \text{probability}(\text{existance}(i,\ j)) \tag{13}$$

The probability to generate observed output j $(j \in \overline{1..M})$ from a state i $(i \in \overline{1..N})$:

$$A_{ij} = \text{probability}(\text{existance}(i,\ j)) \tag{14}$$

At first, the source node can be connected to one of clusters with some probability. So if this node is connected to cluster C_i then the probability to get the first state is the existance probability of the connection between the source node and C_i.

At the state i, with accumulation probability to reach cluster C_i is P_i, so the for reaching to cluster j P_{i+1} that has probability is:

$$P_{i+1} = P_i * P_{C_iC_j} * \prod_{k=j}^{t} A_{C_iM_k} \tag{15}$$

In which, $k \in \overline{j..t}$ is the range of cluster members that are connected to C_i.

Apply the formula (16) for all related cluster heads to get the result.

8 Assessments

8.1 Use ANN for Choosing Cluster Member for Each Cluster

As analyzing above about ANN, We make ANN for each cluster head. We have to choose model for each ANN, the number of layers, the number of neuron of each layer. The weight vector is made randomly, and then it is tuned by the back propagation process. The formulas for tuning weight vector are calculated:

The code is written in C# as follow:

```
    public void Correct_weight()
            {
                double ws = 0;
                int iCount = 0;
                for (int k = ls_Layer.Count - 2; k > 0; k--)
                {
                    iCount = ls_Layer[k + 1].ds_Neural.Count;
                    for (int j = 0; j < ls_Layer[k].ds_Neural.Count;
j++)
                    {
                        ws = 0;
                        for (int i = 0; i < iCount; i++)
                        {
                            ws +=
ls_Layer[k].ds_Neural[j].weight_vector[i] * ls_Layer[k +
1].ds_Neural[i].delta;
                        }
                        ls_Layer[k].ds_Neural[j].delta = ws;
                    }
                }
                for (int k = 0; k < ls_Layer.Count - 1; k++)
                {
                    for (int i = 0; i < ls_Layer[k].ds_Neural.Count;
i++)
                    {
                        for (int j = 0; j < ls_Layer[k +
1].ds_Neural.Count; j++)
                        {
                            ls_Layer[k].ds_Neural[i].weight_vector[j]
+= ls_Layer[k].ds_Neural[i].hs_run * ls_Layer[k +
1].ds_Neural[j].delta *
                                    ls_Layer[k].ds_Neural[i].neural_res;
                        }
                        ls_Layer[k].weighted_vector[i] +=
ls_Layer[k].bias *
                        ls_Layer[k].hs_run *
ls_Layer[k].ds_Neural[i].delta;
                    }
                }
            }
```

The simulation data is collected consisting of three dimensions that are number of layers, number of neurons of the first hidden layer and the time required for execution. The execution time consists time for training and time to recognize or detect a member. The diagram is drawn from the simulation data below indicates that the time is not increased fast when the number of layers and number of neurons increasing (Fig. 3).

8.2 Assessment of Using Genetic Algorithm to Train Neural Network

In order for training artificial neural network, as mentioned above we can use Genetic algorithm by crossover operator. We assume a neural network as a solution in each round that it is a gene of genetic population, so we can crossover between these genes.

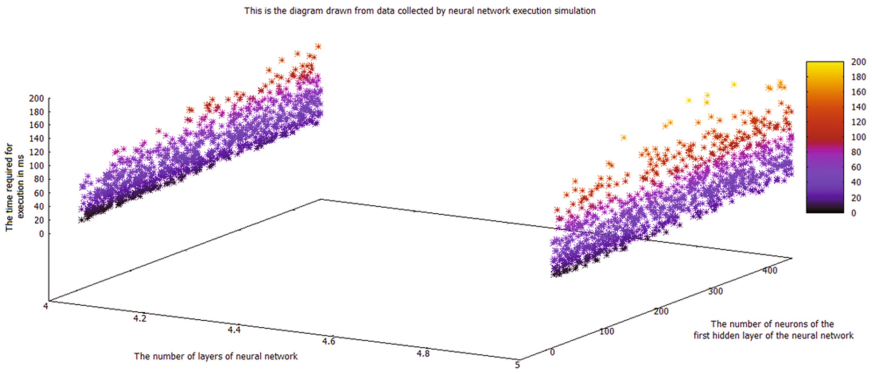

Fig. 3. Assessing results of applying ANN to get a node for a network cluster.

Each gene has some layers; each layer has some nodes, each node has some weight inputs. The procedure to apply the crossover operator as below:

```
   public void crossover_2(cNeural_Network gen_1, cNeural_Network
gen_2, int iLayer)
        {
            int iPos = -1;
            double weight = 0;

            try
            {
                rand = new Random();
                iPos =
rand.Next(gen_1.ls_Layer[iLayer].neural_count);
                weight = 0;

                for (int i = 0; i <
gen_1.ls_Layer[iLayer].prev_layer.ds_Neural.Count; i++)
                {
                    weight =
gen_1.ls_Layer[iLayer].prev_layer.ds_Neural[i].weight_vector[iPos];

gen_1.ls_Layer[iLayer].prev_layer.ds_Neural[i].weight_vector[iPos] =
gen_2.ls_Layer[iLayer].prev_layer.ds_Neural[i].weight_vector[iPos];

gen_2.ls_Layer[iLayer].prev_layer.ds_Neural[i].weight_vector[iPos] =
weight;
                }
            }
            catch { }
        }
```

In the following graph, we compare the time required to training ANN in two cases: (i) case 1: training ANN normally; (ii) case 2: training ANN using Genetic's crossover operation:

We try to test the algorithms to train ANN in 280 times, we calculate two factors: (i) Average time required to train; and (ii) the time required for normal training and training with Genetic's crossover operation. The average time of Genetic (T(Gene)) is rather lower than in normal training (T(Norm)) (Fig. 4).

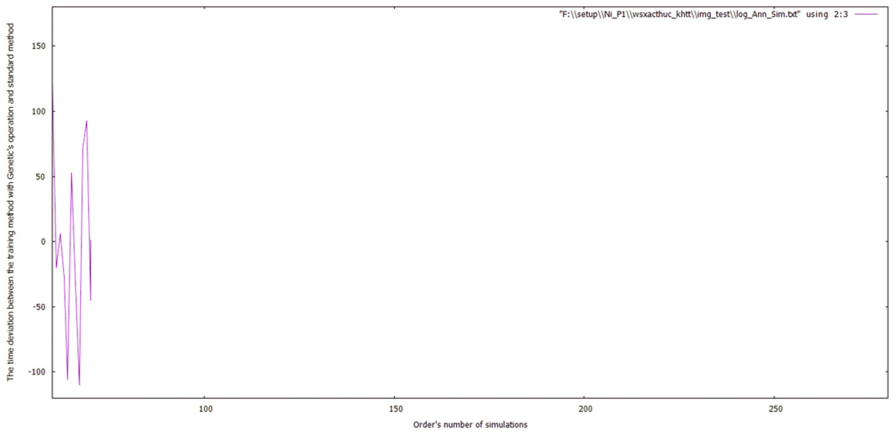

Fig. 4. Simulation results of training ANN with genetic's operation.

According the above graph, we can see that in common:

$$\mathrm{T(Gene)} < \mathrm{T(Norm)} \tag{16}$$

From the last point in the graph, we can see that:

$$\mathrm{Average(T(Gene))} - \mathrm{Average(T(Norm))} < 0 \tag{17}$$

So we can apply Genetic's operation for training process of ANN.

9 Assessing Results

In combination of some advanced technologies as introduced above of the paper to make MANET operating better and reduce overhead. We can use R tree to cluster network hierarchically effectively. ANN can be used to choose CH to manage cluster and choose child node of CH. In each cluster, We can use multicast tree to transmit information from CH to its child nodes. Besides, We can use ACO, HMM to find optimal route to transmit data.

References

1. Long, N.T., Thuy, N.D., Hoang, P.H.: Research on applying hierarchical clustered based routing technique using artificial intelligence algorithms for quality of service of service based routing. IoT Cloud Comput. **3**, 14–21 (2015). doi:10.11648/j.iotcc.s.2015030601.11. Special Issue: Quality of Service of Service Based Routing

2. Long, N.T., Thuy, N.D., Hoang, P.H.: Research on innovating and applying evolutionary algorithms based hierarchical clustering and multiple paths routing for guaranteed quality of service on service based routing. IoT Cloud Comput. **3**, 9–15 (2015). doi:10.11648/j.iotcc.s.2015030601.12. Special Issue:Quality of Service of Service Based Routing
3. Srungaram, K., Krishna Prasad, M.H.M.: Enhanced cluster based routing protocol for MANETS
4. Ferreira, C.: Gene expression programming: a new adaptive algorithm for solving problems
5. Roy, B.: Ant Colony based Routing for Mobile Ad-Hoc Networks towards Improved Quality of Services
6. Long, N.T., Thuy, N.D., Hoang, P.H., Chien, T.D.: Innovating R tree to create summary filter for message forwarding technique in service-based routing. In: Qian, H., Kang, K. (eds.) WICON 2013. LNICSSITE, vol. 121, pp. 178–188. Springer, Heidelberg (2013). doi:10.1007/978-3-642-41773-3_19
7. Long, T.N., Tam, N.T., Chien, T., Thuy, N.D.: Research on innovating, applying multiple paths routing technique based on fuzzy logic and genetic algorithm for routing messages in service - oriented routing. J. Scalable Inf. Syst. EAI **15**, e2 (2015)
8. Chen, K.-T., Fan, K., Dai, Y., Baba, T.: A particle swarm optimization with adaptive multi-swarm strategy for capacitated vehicle routing problem. EAI Endorsed Trans. Ind. Netw. Intell. Syst. **15**, e3 (2015)
9. Long, N.T., Thuy, N.D., Hoang, P.H.: Research on innovating, evaluating and applying multicast routing technique for routing messages in service-oriented routing. In: Vinh, P.C., Hung, N.M., Tung, N.T., Suzuki, J. (eds.) ICCASA 2012. LNICSSITE, vol. 109, pp. 212–228. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36642-0_22
10. Newson, P., Krumm, J.: Hidden Markov Map Matching Through Noise and Sparseness. Microsoft Research, Redmond (2009)
11. http://www.bowdoin.edu/∼ltoma/teaching/cs340/spring08/Papers/Rtree-chap1.pdf
12. https://en.wikipedia.org/wiki/R-tree