A New Efficient Fast Routing Protocol for MANET

Hazem M. El-Bakry, and Mamoon H. Mamoon

Faculty of Computer Science & Information Systems,
Mansoura University, EGYPT
helbakry20@yahoo.com

Abstract: In this paper, an efficient fast secure routing protocol for MANET is presented. Fast packet detection is a critical issue to overcome intrusion attack. Here, a new approach for fast packet detection during data transmission is presented. Such algorithm uses fast time delay neural networks (FTDNNs). The operation of these networks relies on performing cross correlation in the frequency domain between the input data and the input weights of neural networks. It is proved mathematically and practically that the number of computation steps required for the presented FTDNNs is less than that needed by conventional time delay neural networks (CTDNNs). Simulation results using MATLAB confirm the theoretical computations. This security algorithm is involved with Ad hoc Dynamic Source Routing (DSR) protocol based on mobility prediction, named as MDSR. Such protocol controls route discovery, route keeping and route switching according to the distance and mobility estimation of the neighbor nodes. The proposed protocol is implemented by using a modified NS2. The performance of the proposed protocol is described. Simulation results are given.

Keywords: Fast Neural Network; Cross Correlation, Frequency Domain, Packet Detection, Routing (DSR) Protocol.

1. Introduction

For intrusion detection, it is important to detect packets during transmission of data sequence through computer networks. Recently, time delay neural networks have shown very good results in different areas such as automatic control, speech recognition, blind equalization of time-varying channel and other communication applications. Here, the response time of time delay neural networks is reduced. This is done by performing the operations in the frequency domain instead of the time domain. This approach was successfully applied for sub-image detection using fast neural networks (FNNs) as proposed in [5]. Furthermore, it was used for fast face detection, and fast iris detection. Another idea was presented to further increase the speed of FNNs through image decomposition [6].

FNNs for detecting a certain code in one dimensional serial stream of sequential data were described in [7,8]. Compared with conventional neural networks, FNNs based on cross correlation between the tested data and the input weights of neural networks in the frequency domain showed a significant reduction in the number of computation steps required for certain data detection [5-8]. Here, we make use of the theory of FNNs implemented in the frequency domain to increase the speed of time delay neural networks for packet detection [5-8]. The idea of moving the testing process from the time domain to the frequency domain is applied to time delay neural networks. Theoretical and practical results show that the proposed FTDNNs are faster than CTDNNs.

Mobile Ad hoc Network (MANET) [10,11,12,17] comes from the Defense Advanced Research Projects Agency (DARPA), which is a network that each node processes packet routing without infrastructure. The fast equipping and mobile self-organization characters push MANET the hot point in wireless communication, especially the route protocol, e.g., now Internet Engineering Task Force (IETF)'s MANET working Group is searching for the comments [14,15].

Today routing protocol is approximately divided into table-driven routing protocol and On-Demand routing protocol [11,13] according to its trigger mechanism. Table-driven routing protocol is based upon the established routing tables, thus owns small delay. Its virtue is to not wait for establishing route while information are transmitted and delay is small. Meanwhile, on-demand routing protocol looks for routing when demanded, where the node usually needn't maintain routing technically. Hence it is suitable for MANET and is widely researched recently.

On-demand routing is usually made up of two processes named routing set up and routing maintain. When nodes need transmit information, they broadcast route set up packets and search the optimal path without storage of the routing information, resulting in

effective bandwidth and memory save. But it is apparently at the cost of delay due to the set up procedure. Accordingly, this style represents in Ad hoc Dynamic Source Routing (DSR) [15].

This paper presents a new DSR routing protocol based on mobility prediction (MDSR). It can control node route by keeping and switching route through estimating the neighbor node's distance and predicting the neighbor node's mobility to fit fast changed network topology. This mechanism reduces the end-to-end delay effectively and enhances the real-time character, which is very important to the voice communication and the video communication.

Theory of neural networks for pattern detection are described in section 2. Section 3 presents FTDNNs for detecting certain packet in serial data. Section 4 presents a DSR routing protocol. We investigate the proposed routing protocol (MDSR) in section 5. Section 6 elaborates on the simulation environment and the experimental results. In section 7, we present our conclusion.

2. Theory of ANNs for Pattern Detection

Artificial neural network (ANN) is a mathematical model, which can be set one or more layered and occurred from many artificial neural cells. The wide usage of the ANN may be due to the three basic properties: (1) the ability of the ANN as a parallel processing of the problems, for which if any of the neurons violate the constraints would not affect the overall output of the problem; (2) the ability of the ANN to extrapolate from historical data to generate forecasts; and (3) the successful application of the ANN to solve non-linear problems. The history and theory of the ANN have been described in a large number of published literatures and will not be covered in this paper except for a very brief overview of how neural networks operate.

The ANN computation can be divided into two phases: learning phase and testing phase. The learning phase forms an iterative updating of the synoptic weights based upon the error back propagation algorithm. Back propagation algorithm is generalized of least mean square learning rule, which is an approximation of steepest descent technique. To find the best approximation, multi-layer feed forward neural network architecture with back propagation learning rule is used. A schematic diagram of typical multi-layer feed-forward conventional time delay neural network architecture is shown in Fig. 1. The network has five inputs and one output neuron in its linear output layer. The number of neurons in the hidden layer is varied to give the network enough power to solve the problem. Each neuron computes a weighted sum of the individual inputs (I_1, I_2, ..., Ij) it receives and adding it with a bias (b) to form the net input (x). The bias is included in the neurons to allow the activation function to be offset from zero.

$$sum = w_{1,1}.I_1 + w_{1,2}.I_2 + ... + w_{1,j}.I_j + b \quad (1)$$

Where, wji is the connection weight between neuron j and neuron i. The net input (sum) is then passed to the subsequent layer through a non linear sigmoid function to form its own output (yj).

$$y_j = 1/\left(1 + e^{-sum}\right) \quad (2)$$

Afterward, the output yj was compared with the target output tj using an error function of the form:

$$\delta_k = \left(t_j - y_j\right) y_j \left(1 - y_j\right) \quad (3)$$

For the neuron in the hidden layer, the error term is given by the following equation:

$$\delta_j = y_j \left(1 - y_j\right) \sum_k \delta_k w_k \quad (4)$$

where δk is the error term of the output layer, and wk is the weight between the hidden layer and output layer. The error was then propagated backward from the output layer to the input layer to update the weight of each connection at iteration (t + 1) as follows:

$$w_{ji}(t+1) = w_{ji}(t) + \eta \delta_j y_j + \alpha \left(w_{ii}(t) - w_{ii}(t-1)\right) \quad (5)$$

Choosing a small learning rate η leads to slow rate of convergence, and too large η leads to oscillation. The term α is called momentum factor and determines the effect of past weight changes on the current direction of movement. Both of these constant terms are specified at the start of the training cycle and determine the speed and stability of the network. The process was repeated for each input pattern until the error was reduced to a threshold value.

3. Real-Time Packet Detection by using FTDNNs

Finding a certain packet of information, in the incoming serial data, is a searching problem. First neural networks are trained to classify the defected packets which contain intrusion codes from others that do not and this is done in time domain. In packet detection phase, each position in the incoming packet is tested for intrusion code. At each position in the input one dimensional matrix, each sub-matrix is multiplied by a window of weights, which has the same size as the sub-matrix. The outputs of neurons in the hidden layer are multiplied by the weights of the output layer. When the final output is high, this means that the sub-matrix under test contains the required information in data sequence and vice versa. Thus, we may conclude that this searching problem is a cross correlation between the incoming serial data and the weights of neurons in the hidden layer.

The convolution theorem in mathematical analysis says that a convolution of f with h is identical to the result of the following steps: let F and H be the results of the Fourier Transformation of f and h in the frequency domain. Multiply F and H* in the frequency domain point by point and then transform this product into the spatial domain via the inverse

Fourier Transform. As a result, these cross correlations can be represented by a product in the frequency domain. Thus, by using cross correlation in the frequency domain, speed up in an order of magnitude can be achieved during the detection process [5-25]. Assume that the size of the attack code is 1xn. In attack detection phase, a sub matrix I of size 1xn (sliding window) is extracted from the tested matrix, which has a size of 1xN. Such sub matrix, which may be an attack code, is fed to the neural network. Let W_i be the matrix of weights between the input sub-matrix and the hidden layer. This vector has a size of 1xn and can be represented as 1xn matrix. The output of hidden neurons h(i) can be calculated as follows [5]:

$$h_i = g\left(\sum_{k=1}^{n} W_i(k)I(k) + b_i\right) \quad (6)$$

where g is the activation function and b(i) is the bias of each hidden neuron (i). Equation 1 represents the output of each hidden neuron for a particular sub-matrix I. It can be obtained to the whole input matrix Z as follows [5]:

$$h_i(u) = g\left(\sum_{k=-n/2}^{n/2} W_i(k)\, Z(u+k) + b_i\right) \quad (7)$$

Eq.6 represents a cross correlation operation. Given any two functions f and d, their cross correlation can be obtained by [4]:

$$d(x) \otimes f(x) = \left(\sum_{n=-\infty}^{\infty} f(x+n)d(n)\right) \quad (8)$$

Therefore, Eq. 7 may be written as follows [5]:

$$h_i = g\left(W_i \otimes Z + b_i\right) \quad (9)$$

where h_i is the output of the hidden neuron (i) and h_i (u) is the activity of the hidden unit (i) when the sliding window is located at position (u) and (u) $\in$ [N-n+1].

Now, the above cross correlation can be expressed in terms of one dimensional Fast Fourier Transform as follows [5]:

$$W_i \otimes Z = F^{-1}\left(F(Z) \bullet F^{*}\left(W_i\right)\right) \quad (10)$$

Hence, by evaluating this cross correlation, a speed up ratio can be obtained comparable to conventional neural networks. Also, the final output of the neural network can be evaluated as follows:

$$O(u) = g\left(\sum_{i=1}^{q} W_O(i)\, h_i(u) + b_O\right) \quad (11)$$

where q is the number of neurons in the hidden layer. O(u) is the output of the neural network when the sliding window located at the position (u) in the input matrix Z. W_o is the weight matrix between hidden and output layer.

The complexity of cross correlation in the frequency domain can be analyzed as follows:

1- For a tested matrix of 1xN elements, the 1D-FFT requires a number equal to $N\log_2 N$ of complex computation steps [13]. Also, the same number of complex computation steps is required for computing the 1D-FFT of the weight matrix at each neuron in the hidden layer.

2- At each neuron in the hidden layer, the inverse 1D-FFT is computed. Therefore, q backward and (1+q) forward transforms have to be computed. Therefore, for a given matrix under test, the total number of operations required to compute the 1D-FFT is $(2q+1)N\log_2 N$.

3- The number of computation steps required by FTDNNs is complex and must be converted into a real version. It is known that, the one dimensional Fast Fourier Transform requires $(N/2)\log_2 N$ complex multiplications and $N\log_2 N$ complex additions [3]. Every complex multiplication is realized by six real floating point operations and every complex addition is implemented by two real floating point operations. Therefore, the total number of computation steps required to obtain the 1D-FFT of a 1xN matrix is:

$$\rho = 6((N/2)\log_2 N) + 2(N\log_2 N) \quad (12)$$

which may be simplified to:

$$\rho = 5N\log_2 N \quad (13)$$

4- Both the input and the weight matrices should be dot multiplied in the frequency domain. Thus, a number of complex computation steps equal to qN should be considered. This means 6qN real operations will be added to the number of computation steps required by FTDNNs.

5- In order to perform cross correlation in the frequency domain, the weight matrix must be extended to have the same size as the input matrix. So, a number of zeros = (N-n) must be added to the weight matrix. This requires a total real number of computation steps = q(N-n) for all neurons. Moreover, after computing the FFT for the weight matrix, the conjugate of this matrix must be obtained. As a result, a real number of computation steps = qN should be added in order to obtain the conjugate of the weight matrix for all neurons. Also, a number of real

computation steps equal to N is required to create butterflies complex numbers ($e^{-jk(2\Pi n/N)}$), where 0<K<L. These (N/2) complex numbers are multiplied by the elements of the input matrix or by previous complex numbers during the computation of FFT. To create a complex number requires two real floating point operations. Thus, the total number of computation steps required for FTDNNs becomes:

$$\sigma=(2q+1)(5N\log_2 N)+6qN+q(N-n)+qN+N \quad (14)$$

which can be reformulated as:

$$\sigma=(2q+1)(5N\log_2 N)+q(8N-n)+N \quad (15)$$

6- Using sliding window of size 1xn for the same matrix of 1xN pixels, q(2n-1)(N-n+1) computation steps are required when using CTDNNs for certain attack detection or processing (n) input data. The theoretical speed up factor η can be evaluated as follows:

$$\eta=\frac{q(2n-1)(N-n+1)}{(2q+1)(5N\log_2 N)+q(8N-n)+N} \quad (16)$$

CTDNNs and FTDNNs are shown in Figures 1 and 2 respectively.

Time delay neural networks accept serial input data with fixed size (n). Therefore, the number of input neurons equals to (n). Instead of treating (n) inputs, the proposed new approach is to collect all the incoming data together in a long vector (for example 100xn). Then the input data is tested by time delay neural networks as a single pattern with length L (L=100xn). Such a test is performed in the frequency domain as described before.

The theoretical speed up ratio for searching short successive (n) code in a long input vector (L) using time delay neural networks is listed in tables 1, 2, and 3. Also, the practical speed up ratio for manipulating matrices of different sizes (L) and different sized weight matrices (n) using a 2.7 GHz processor and MATLAB is shown in table 4.

Table 1: The theoretical speed up ratio for packet detection (packet length=400).

Length of serial data	Number of computation steps required for CTDNNs	Number of computation steps required for FTDNNs	Speed up ratio
10000	2.3014e+008	4.2926e+007	5.3613
40000	0.9493e+009	1.9614e+008	4.8397
90000	2.1478e+009	4.7344e+008	4.5365
160000	3.8257e+009	8.8219e+008	4.3366
250000	5.9830e+009	1.4275e+009	4.1912
360000	8.6195e+009	2.1134e+009	4.0786
490000	1.1735e+010	2.9430e+009	3.9876
640000	1.5331e+010	3.9192e+009	3.9119

Table 2: The theoretical speed up ratio for packet detection (packet length=625).

Length of serial data	Number of computation steps required for CTDNNs	Number of computation steps required for FTDNNs	Speed up ratio
10000	3.5132e+008	4.2919e+007	8.1857
40000	1.4754e+009	1.9613e+008	7.5226
90000	3.3489e+009	4.7343e+008	7.0737
160000	0.5972e+010	8.8218e+008	6.7694
250000	0.9344e+010	1.4275e+009	6.5458
360000	1.3466e+010	2.1134e+009	6.3717
490000	1.8337e+010	2.9430e+009	6.2306
640000	2.3958e+010	3.9192e+009	6.1129

Table 3: The theoretical speed up ratio for packet detection (packet length=900).

Length of serial data	Number of computation steps required for CTDNNs	Number of computation steps required for FTDNNs	Speed up ratio
10000	4.9115e+008	4.2911e+007	11.4467
40000	2.1103e+009	1.9612e+008	10.7600
90000	4.8088e+009	4.7343e+008	10.1575
160000	0.8587e+010	8.8217e+008	9.7336
250000	1.3444e+010	1.4275e+009	9.4178
360000	1.9381e+010	2.1134e+009	9.1705
490000	2.6397e+010	2.9430e+009	8.9693
640000	3.4493e+010	3.9192e+009	8.8009

Table 4: Practical speed up ratio for packet detection.

Length of serial data	Speed up ratio (n=400)	Speed up ratio (n=625)	Speed up ratio (n=900)
10000	8.94	12.97	17.61
40000	8.60	12.56	17.22
90000	8.33	12.28	16.80
160000	8.07	12.07	16.53
250000	7.95	17.92	16.30
360000	7.79	11.62	16.14
490000	7.64	11.44	16.00
640000	7.04	11.27	15.89

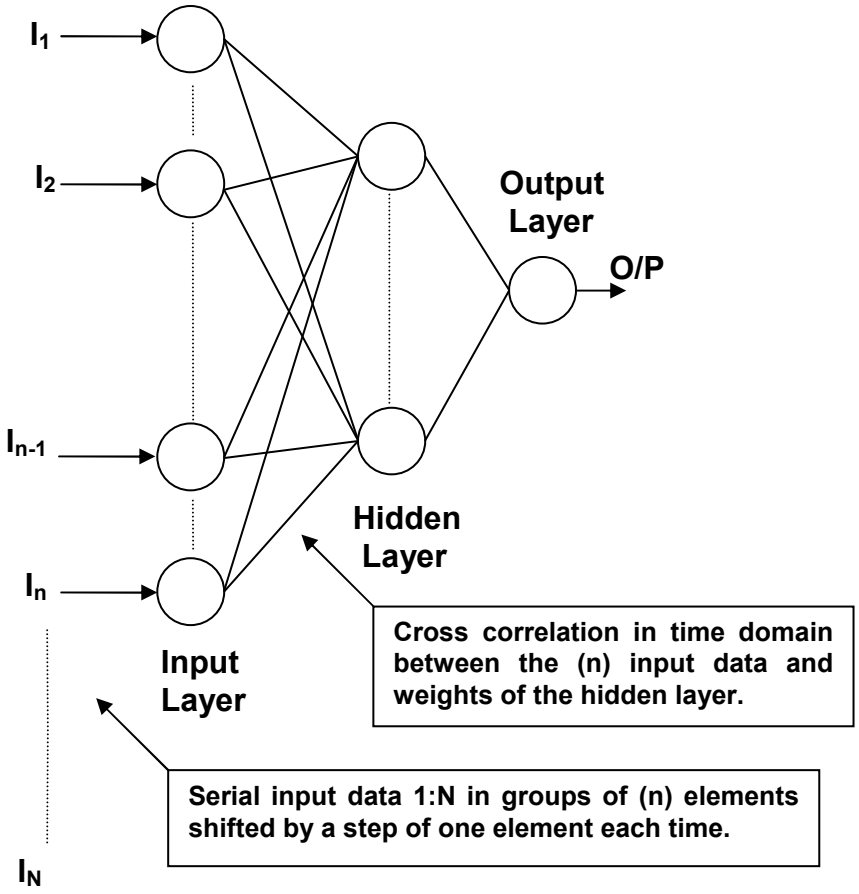

Fig. 1. CTDNNs.

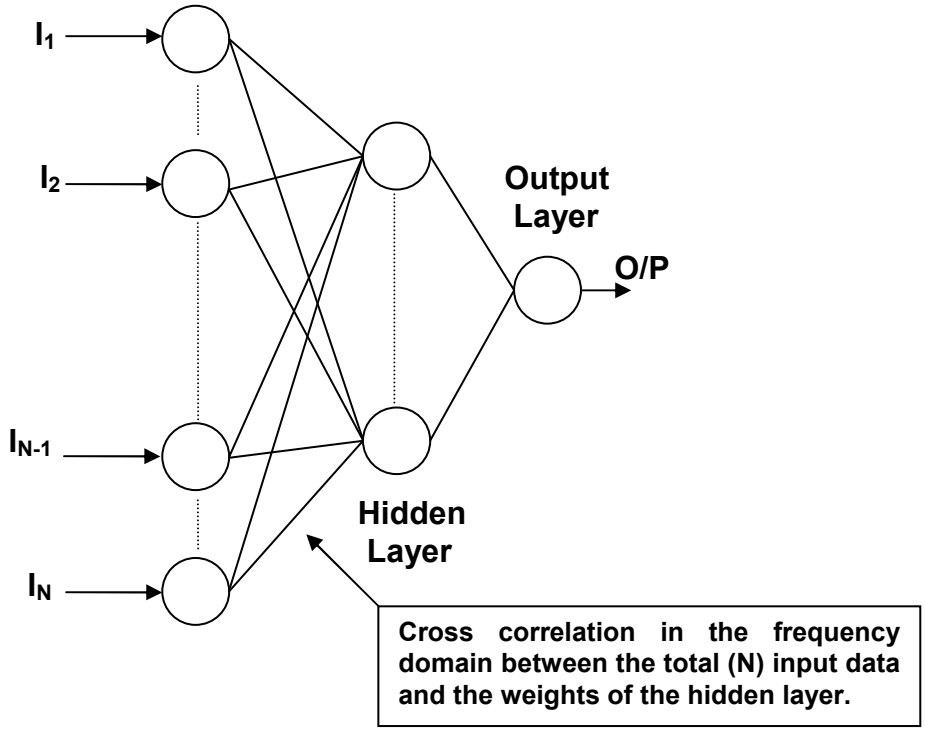

Fig.2. FTDNNs.

An interesting point is that the memory capacity is reduced when using FTDNN. This is because the number of variables is reduced compared with CTDNN.

4. DSR Protocol

DSR is an entirely on-demand ad hoc network routing protocol composed of two parts: Route Discovery and Route Maintenance. In this section, we describe the basic form of Route Discovery and Route maintenance in DSR.

In DSR, when a node has a packet to send to some destination and does not currently have a route to that destination in its Route Cache, the node initiates Route Discovery to find a route; this node is known as the initiator of the Route Discovery, and the destination of the packet is known as the Discovery's target. The initiator transmits a Route Request (RREQ) packet as a local broadcast, specifying the target and a unique identifier from the initiator. Each node receiving the Route Request, if it has recently seen this request identifier from the initiator, discards the Request. Otherwise, it appends its own node address to a list in the Request and rebroadcasts the Request. When the Route Request reaches its target node, the target sends a Route Replay (RREP) back to the initiator of the Request, including a copy of the accumulated list of addresses from the Request. When the Replay reaches the initiator of the Request, it caches the new route in its Route Cache [19].

Route Maintenance is the mechanism by which a node sending a packet along a specified route to some destination detects if that route has broken, for example because two nodes in it have moved too apart. DSR is based on source routing: when sending a packet, the originator lists in the header of the packet the complete sequence of nodes through which the packet is forwarded. Each node along the route forwards the packet to the next hop indicated in the packet's header, and attempts to confirm this by means of a link-layer acknowledgment or network layer acknowledgment. If, after a limited number of local retransmissions of the packet, a node in the route is unable to make this confirmation, it returns a Route Error to the original source of packet, identifying the link from itself to the next node was broken. The sender then removes this broken link from its Route Cache; for subsequent packets to its destination, the sender may use any other route to its destination in its Cash, or it may attempt a new Route Discovery for that target if necessary.

5. A Proposed DSR Protocol

Our proposed routing protocol based on mobility prediction combined with the DSR routing protocol.

Figure 1 shows a sketch map on mobility prediction. Node D moves to node D' in Velocity V. The radial distance varies from d to d' relative to node S and the radial velocity act as V and $V_r^{'}$ Thus the movement difference is the $\Delta d = d - d'$. After calculating the

received power P_r, d, is determined by:

$$d = \sqrt[1/4]{k.P_t / P_r} \quad (17)$$

Here, we suppose that all nodes have the same transmission power Pt The neighbor node's velocity is given by:

$$\overline{V} = \Delta d / \Delta t \quad (18)$$

where, Δt is spacing time for twice measurement.

By the positive or negative of v value, we can determine movement direction parameter of neighbor nodes that are outward node S or inward node S for instance:

$$Direction = \begin{Bmatrix} \overline{V} > 0 & outward \\ \overline{V} = 0 & static \\ \overline{V} < 0 & inward \end{Bmatrix} \quad (19)$$

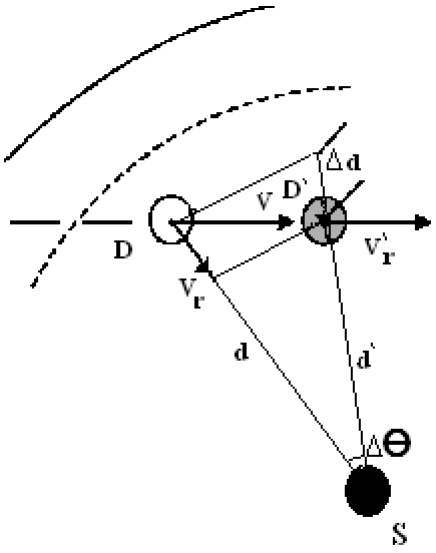

Figure 3: Sketch map on mobility prediction.

By estimating space distance d, neighbor's movement velocity v and neighbor's movement direction, we can predict effective communication area whether node leaves off one another. If node's movement direction is outward and d is already close to effective communication radius and v is relatively large, we may in advance start up routing detection and establish a new route before current path breaks off. In the suitable time path switching will be finished. So we may decrease delivery delay duo to suddenly route breaking off by prediction technique effectively.

According to analysis above mentioned, we provide an equation 20 for estimating effective communication area that neighbor node will outward.

$$V_r T_{min} + d = D_{comm} \quad (20)$$

distance, V_r is the radial velocity and T_m denotes a new routing detective time estimated. When neighbor node's distance d satisfied with equation 5, a new routing detection is stated up. Before primary routing path breaks off, a new routing will set up.

$$d > D_{comm} - V T_{min} \quad (21)$$

6. MDSR Protocol Simulation

A network simulator (NS2) [9] was developed in order to monitor, observe and measure the performance of MDSR routing. In simulations, we choose propagation model based on 802.11 and apply lucent's WaveLAN. Under these simulation conditions, the wireless node is configured as:

Table 5: Configuration parameters

Configuration Parameter	Value
Effective comm. area	250 m
Terrain range	1500x300 m
No. of nodes	50
Channel model	Tow-ray ground model
Antenna model	Omni-antenna
Speed	1-20 m/sec
Packet size	64 and 1024 Bytes
MAC layer	IEEE 802.11
Routing protocol	DSR and MDSR
Application traffic	CBR=20

The results of experiments are shown in the following Figures. From Figure 4, we may obtain end-to-end packet delay in the MDSR protocol superior to in the DSR protocol. The decrease of delay is mainly introduced by route updating predicted. Because route updating is predicted, the delay by route breaking off will greatly reduce. So the average delay character in the MDSR will be improvement.

Figure 5 shows the packet delivery ratio versus the node's velocity, where packet delivery ratio is the ratio of received packets versus transmitted packets. As the velocity (mobility) increases, the packet delivery ratio in DSR and MDSR all decrease. But decreased value is not dramatically. Simultaneously the MDSR follows on-demand routing character and its packet delivery ratio will slightly inferior to DSR. After setting up prediction path, the MDSR don't process further for network topology. The prediction algorithm that is finds out is the temporal excellent path. By memory, the path may be do not the most stable path. Meanwhile the error between the prediction algorithm and the actual status will bring

some packet loosen ratio. Therefore under the complex network circumstances, the MDSR will decrease packet delivery ratio because the action of MDSR protocol reply network topology

Figure 5 shows the packet delivery ratio versus the node's velocity, where packet delivery ratio is the ratio of received packets versus transmitted packets. As the velocity (mobility) increases, the packet delivery ratio in DSR and MDSR all decrease. But decreased value is not dramatically.

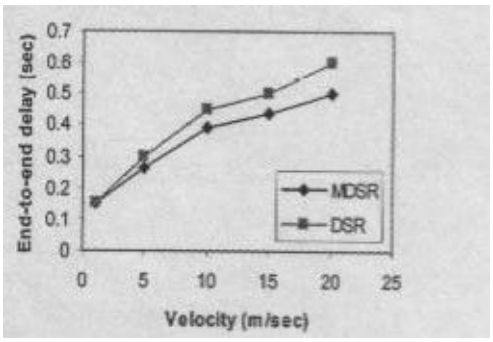

Figure 4: End-to-end delay vs. node's velocity.

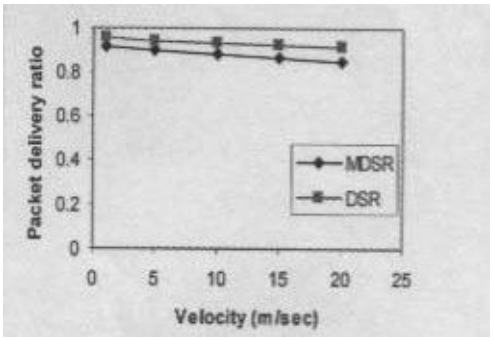

Figure 5: Packet delivery ratio vs. node's velocity

Simultaneously the MDSR follows on-demand routing character and its packet delivery ratio will slightly inferior to DSR. After setting up prediction path, the MDSR don't process further for network topology. The prediction algorithm that is finds out is the temporal excellent path. By memory, the path may be do not the most stable path. Meanwhile the error between the prediction algorithm and the actual status will bring some packet loosen ratio. Therefore under the complex network circumstances, the MDSR will decrease packet delivery ratio because the action of MDSR protocol reply network topology

7. Conclusion

A fast neural algorithm for packet detection during data transmission has been presented. Such strategy has been realized by using our design for FTDNNs. Theoretical computations have shown that FTDNNs require fewer computation steps than conventional ones. This has been achieved by applying cross correlation in the frequency domain between the input data and the weights of neural networks. Simulation results have confirmed this proof by using MATLAB. The proposed algorithm can be applied efficiently to overcome various intrusion codes.

In addition, a novel MDSR routing protocol based on mobility prediction combined with the DSR routing protocol has been presented. The algorithm controls route discovery, route keeping and route switching by estimating the neighbor node's distance and predicting the neighbor node's mobility. Then we introduce a modified NS2 to verify the algorithm in terms of the average end-to-end packet delay and the packet delivery ratio. Therefore, the MDSR reduces the end-to-end delay effectively and enhance the data transmission rate, which is very important to real-time communication such as the voice communication and multimedia image communication. On the other hand, the MDSR will reduce packet delivery ratio.

References

[1] Alok Tongaonkar, Sreenaath Vasudevan, and R. Sekar, "Fast Packet Classification for Snort by Native Compilation of Rules," 22nd Large Installation System Administration Conference (LISA '08), San Diego, California, November 9-14, 2008.

[2] Mike Fisk, George Varghese "Fast Content-Based Packet Handling for Intrusion Detection," UCSD Technical Report CS2001-0670, May 2001.

[3] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. Comput. 19, 1965, pp. 297–301.

[4] R. Klette, and Zamperon, "Handbook of image processing operators," John Wiley & Sonsltd, 1996.

[5] Hazem M. El-Bakry, and Qiangfu Zhao, "Speeding-up Normalized Neural Networks For Face/Object Detection," Machine Graphics & Vision Journal (MG&V), vol. 14, No.1, 2005, pp. 29-59.

[6] Hazem M. El-Bakry, "Face detection using fast neural networks and image decomposition," Neurocomputing Journal, vol. 48, 2002, pp. 1039-1046.

[7] Hazem M. El-Bakry, and Qiangfu Zhao, "A Fast Neural Algorithm for Serial Code Detection in a Stream of Sequential Data," International Journal of Information Technology, vol.2, no.1, pp. 71-90, 2005.

[8] Hazem M. El-Bakry and Nikos Mastorakis, "Fast Code Detection Using High Speed Time Delay Neural Networks," Lecture Notes in Computer Science, Springer, vol. 4493, Part III, May 2007, pp. 764-773.

[9] D. Cavin, Y. Sasson and A. Schiper, " On the accuracy of MANET simulators," Proceeding ACM POMC'02, pp. 38-43, Oct. 2002.

[10] S. Giordano and W. Lu, " Challenges in mobile ad hoc networking," IEEE Network Magazine, Vol. 39, No. 6,

pp. 129-129, June 2001.

[11] J. Broch, D. A Maltz and D. B. Johnson, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," in 4th Annual *ACM/ IEEE* International Conference on Mobile Computing and Networking, Dallas, TX, USA, Oct. 1998.

[12] D. Kim, H. Bae and C. K. Toh, " TCP- vegas- Ad hoc: Improving TCP vegas performance over MANET routing protocols," IEEE Trans. Veh. Technology, Vol. 56, No. 1, pp. 372-377, Jan. 2006.

[13] S. Lee, M. Gerla and C. K. Toh, "A simulation study of table-driven and on-demand routing protocols for mobile ad hoc networks," IEEE Trans. Network, Vol. 13, No. 4, pp. 48-54, August 1999.

[14] A. Mnaouer, L. Chen, C. Foh and J. Tantra, "The OPHMR: An optimized polymorphic hybrid multicast routing protocol for MANET," IEEE Trans. Mobile Computing, Vol. 6, No. 5, pp. 551-562, May 2007.

[15] D. B. Johnson, D. A Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks," Internet-Draft, draft-ietf- manet-dsr-07.txt, Feb. 2002.

[16] C. Scaglione, D. Goeckel and J. Laneman, "Cooperative communications in mobile ad hoc networks," IEEE Signal Proc. Mag., Vol. 23, No. 5, pp. 18-29, September 2006.

[17] Y. Kim and A. Peering, "Simple and fault-tolerant key agreement for dynamic collaborative groups," in *ACM* Conference on Computer and Communication Security, pp. 235-244, May 2000.

[18] S. Marti and T. Giuli, "Mitigating Routing Misbehavior in Mobile Ad-hoc Networks," in 6th Annual ACM/ IEEE International Conference on Mobile Computing and Networking, Boston, MA, USA, Aug. 2000.

[19] Mamoon H. Mamoon, "A New DSR Routing Protocol for MANET "Journal of Convergence Information Technology Volume 4, Number 4, December 2009.