

# Adaptive Route Selection Policy Based on Back Propagation Neural Networks

Fang Jing \*

Department of Computer Science and Engineering Graduate School of Engineering, Nagoya Institute of Technology  
Gokiso, Showa-ku, Nagoya, Japan

Email: [jingfang@moss.elcom.nitech.ac.jp](mailto:jingfang@moss.elcom.nitech.ac.jp)

R.S.Bhuvaneshwaran, Yoshiaki Katayama and Naohisa Takahashi

Department of Computer Science and Engineering Graduate School of Engineering, Nagoya Institute of Technology  
Gokiso, Showa-ku, Nagoya, Japan

Email: [{bhuvan,katayama,naohisa}@moss.elcom.nitech.ac.jp](mailto:{bhuvan,katayama,naohisa}@moss.elcom.nitech.ac.jp)

**Abstract**—One of the key issues in the study of multiple route protocols in mobile ad hoc networks (MANETs) is how to select routes to the packet transmission destination. There are currently two route selection methods: primary routing policy and load-balancing policy. Many ad hoc routing protocols are based on primary (fastest or shortest but busiest) routing policy from the self-standpoint of traffic transmission optimization of each node. Load-balancing protocols equalize transmission load among multiple routes in the network. However, the lack of global perspective can cause congestion in primary policy and prolong delay time in load-balancing policy. So, although they are sometimes efficient, these two types of policies cannot adapt to intricately changing network conditions. We propose a new multiple route protocol with an Adaptive route selection Policy based on a Back propagation Neural network (APBN) to optimize selection policy. In our study, we used a gradient ascent algorithm to determine the relationship between different optimum route selection policies and varying conditions in the communication network and to make a neural network that learns this relationship using the Back Propagation (BP) algorithm to predict the next optimum route selection policy. We evaluated our protocol using Omnet simulator. The results show that the proposed scheme performs better than current protocols.

**Index Terms**—mobile ad hoc network, multiple route, back propagation, neural network, gradient ascent algorithm

## I. INTRODUCTION

Mobile ad hoc networks (MANETs) are characterized as topological dynamic communication networks due to their node mobility, limited channel bandwidth, and limited battery power of nodes. Now that long-term traffic reliability in MANETs is within the realm of possibility, many researchers are proposing single route protocols to attain it [1-3] but there is still a fatal problem.

When broken routes are initiated, single route protocols have to make frequent route discovery attempts, incurring high route discovery latency and increasing fault overhead. Multiple route protocols can solve these problems [4-12]. Some protocols adopt multiple routes based on route probabilities [9-12]. For example, Ants Routing with routing History (ARH) is a routing algorithm that uses a reinforcement learning method [10]. It randomly selects routes from the set of potential routes according to probabilities calculated based on learned routing history. Given sufficient learning time, an available route will converge on the steadiest route, showing that this protocol is based on primary routing policy.

However, there is a problem with primary routing policy. When each node chooses its own best route based on transmission history, the selection does not take account of the influence exerted by other nodes on the whole communication network, that is to say, from its own perspective, which can cause congestion potentially in the future, which in turn causes a long end-to-end delay. This results in serious transmission failure on the routes for packet transmission routes. Abrahamsson [13] found that the lack of global perspective can hinder local transmission late in transmission processes. Load-balancing policy protocols can prevent the transmission congestion caused by primary routing policy by equalizing multiple route loads. Multipath On-demand Routing (MOR) is representative of multiple route load-balancing policy protocols [11]. In MOR, available routes are chosen by rotation, which is why this load-balancing policy cannot adapt to unequal changes in dynamic communication networks. This circular policy also does not take global variability into account and misses transmission opportunities on the primary route even when there is no danger of congestion. In MOR, by contrast, longer routes with more danger of transmission failure caused by node mobility and limited battery power are selected circularly, even though this is counterintuitive.

---

\*Corresponding author: Fang Jing  
E-mail: [jingfang@moss.elcom.nitech.ac.jp](mailto:jingfang@moss.elcom.nitech.ac.jp)

Relatively fixed communication network load ranges have always been assumed and used in simulations in previous policy evaluations. These policies are advantageous only within certain ranges of communication network load conditions. Therefore, it is necessary to find an estimate and control approach that is able to update route selection policy on its own, thereby adapting to intricately changing communication network conditions. We propose a multiple route protocol with an Adaptive route selection Policy based on Back propagation Neural network (APBN) in which a gradient ascent algorithm of nodes keeps finding the optimum route selection policy based on transmission performance, and a neural network is trained to predict and switch to the next optimum route selection policy using a back propagation (BP) algorithm while taking account of global transmission conditions.

We evaluated the performance of our protocol using Omnet simulator [14] and compared its performance to those of the ARH and MOR protocols. The simulation results show that our adaptive route selection policy offers a higher packet delivery ratio with lower average end-to-end delay and routing overhead ratio in changing communication network conditions.

The paper is organized as follows. Section 2 presents related work. In Section 3, the mechanism of the proposed protocol, APBN, is discussed. We present the results of our evaluation of APBN's performance in Section 4 and make concluding remarks in Section 5.

## II. RELATED WORK

### A. ARH

ARH is a routing algorithm that uses a reinforcement learning method [10] and operates by stochastically selecting good routes and coping with dynamic environments by learning routing history.

$P(x)$  denotes route selection probability to a downstream node  $x$  as the next hop. When node  $x$  sends a packet to an upstream node,  $P(x)$  is strengthened and the route selection probability values of the rest downstream nodes are weakened.  $\Delta p$  denotes change in the probability value. The value of  $\Delta p$  is inversely proportional to transmission delay time. The route selection probabilities of more frequently used routes increase and those of less used routes decrease after multiple transmission learning processes. This shows that ARH applies a primary route selection policy. Frequently used routes face considerable potential for transmission congestion.

### B. MOR

Shu Chen designed the MOR protocol [11] for MANETs. MOR maintains multiple routes to each destination. Each node remembers all next-hop nodes closer to a given destination for which a route exists. MOR sends packets to each next hop node using a circular policy (by rotation), which automatically equalizes the transmission loads of the multiple routes. Thus, MOR is representative of load-balancing policy protocols.

## III. A MULTIPLE ROUTE PROTOCOL WITH ADAPTIVE ROUTE SELECTION POLICY

In this section, we propose a multiple route protocol with an adaptive route selection policy based on back propagation neural network (APBN).

### A. Optimum Route Selection Policy Problem

Although the primary routing and load-balancing policies sometimes perform well, neither of them fully fits communication networks and constantly shows good performance in any load condition range. For example, when the transmission load and node mobile velocity are low, the primary route is maintained for a long time without congestion. Hence, the primary route, that is, the fastest or shortest route with the fewest intermediate nodes, has lower potential for transmission failure than inefficient backup routes with longer transmission delay times. There usually is no need to use all routes relatively equally, so the primary routing policy is predominant over load-balancing policy in these conditions.

When the transmission load is increasing, bottlenecks caused by limited channel bandwidth will appear during periods of high data communication. Primary routing policy, which assigns the highest selection probability to primary routes, faces potential congestion on the route. The congestion becomes the leading cause of transmission delay. Load-balancing policy, in contrast, equalizes loads on multiple routes, meaning that the load on the primary route is not greatly increased. There is less potential for congestion in increasing load conditions when load-balancing policy is used than when primary routing policy is used. In this area, load-balancing policy is better than primary routing policy.

The transmission failure probability of a route, TFPR, is calculated by transmission failure probability of each intermediate node and number of intermediate nodes on the route as:

$$TFPR = 1 - (1 - (B * V * t))^E \tag{1}$$

Where  $E$  is the number of intermediate nodes on the route,  $B$  is failure probability parameter,  $V$  is each intermediate node mobility velocity,  $t$  is one hop transmission delay time of one packet, and  $(B * V * t)$  indicates the transmission failure probability of each intermediate node and is proportional to  $V$  and  $t$ .

$$\text{When } V > \frac{t - E}{B t^2} ;$$

$$\frac{\partial TFPR}{\partial E} > \frac{\partial TFPR}{\partial t} \tag{2}$$

When the node mobility velocity increases, the partial derivative of TFPR over  $E$  is larger than the partial derivative of TFPR over  $t$ . This indicates that the transmission failure caused by more intermediate nodes plays a major role. When this happens, the inefficient backup route, a longer route with more intermediate nodes, has a higher potential for transmission failure than the primary route. Therefore, the primary routing policy has a stronger effect on load-balancing policy in these conditions.

Therefore, dynamic policy control is necessary for an intricately changing communication network.

**B. Switching Route Selection Policy**

In APBN, the source node initially transmits an RREQ packet toward a destination node. In response to the arrived RREQ packet, the destination node returns an RREP packet to the source node. The first RREP packet to arrive is unconditionally accepted by source node *s*, and a route is established. The backup routes are established using later RREP packets from different neighbor nodes. There are *k* routes between source node *s* and destination node *d*.

When a source or intermediate node sends a packet to one of its destinations, the node saves the packet departure time in its list. After the destination node returns an ACK (ACKnowledgement) or RREP packet to the source node, the source or intermediate node remembers the packet arrival time. Subtracting the departure time from the arrival time gives end-to-end packet delay time. Node throughput is the number of packets processed by the node per second. While ACK or RREP packets are returned to the source node by the destination node, the packet records number of nodes *E* and the sum of node throughputs on the route.

For destination *d*, the source node selects route *i* based on route selection probability  $P_i(s, d)$ .  $P_i(s, d)$  is calculated by  $PL_i$  and  $PG_i$ .

$$P_i(s, d) = PL_i + PG_i \tag{3}$$

The  $PL_i(s, d)$  is inversely proportional to the impact parameter, which is defined as end-to-end packet delay time  $T_i(s, d)$  or number of intermediate nodes  $E_i(s, d)$  of route *i*. When node mobility velocity is low,  $T_i(s, d)$  is used as the impact parameter, and  $PL_i$  is calculated as follows:

$$PL_i(s, d) = \frac{1}{T_i(s, d)} \bigg/ \sum_{g=1}^k \frac{1}{T_g(s, d)} \quad 1 \leq i \leq k \tag{4}$$

When node mobility velocity is high, the number of intermediate nodes of the route  $E_i(s, d)$  is used as the impact parameter, and  $PL_i$  is obtained as below:

$$PL_i(s, d) = \frac{1}{E_i(s, d)} \bigg/ \sum_{g=1}^k \frac{1}{E_g(s, d)} \quad 1 \leq i \leq k \tag{5}$$

$PG_i$  is controlled by the  $PL_i$  and degree of preference  $\beta$ . Our proposal for degree of preference  $\beta$ , which is placed into the route selection probability function,  $PG_i$ , to switch route selection policy, is shown in Fig. 1. Here, the primary route with the least end-to-end delay time or intermediate nodes is selected as route 1. The inefficient backup route with the maximum transmission delay time or number of intermediate nodes is selected as route *k*. Route selection policies can be dominated by controlling degree of preference  $\beta$ . When the source node wants to prioritize the primary route, that is, to utilize the primary routing policy, the node uses a high

value of  $\beta$  to assign a high selection probability to the primary route. If the nodes hope to switch to a load balancing policy based on communication network conditions at that time, the node uses a low value of  $\beta$  to avoid overusing the primary route, and the utilization of inefficient backup routes tends to increase.

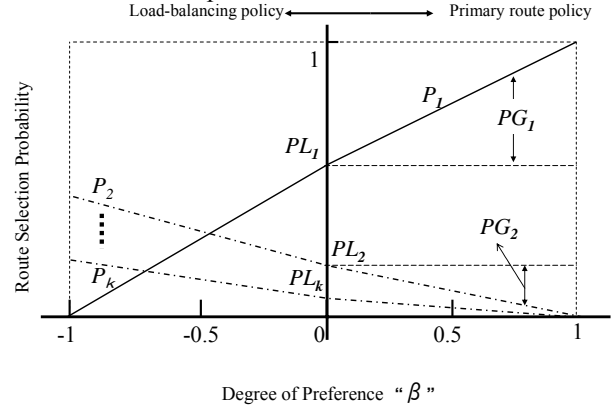


Figure 1. Route selection probability and degree of preference

When *i* is 1 and  $\beta$  is positive,  $\beta$  increases the  $PG_1(s, d)$  of the primary route. When *i* is between 2 and *k*,  $PG_i(s, d)$  is the route selection probability for the inefficient backup routes.  $PG_1(s, d)$  and  $PG_i(s, d)$  can be computed as:

$$PG_i(s, d) = -\beta * PL_i(s, d) ; 2 \leq i \leq k$$

$$PG_1(s, d) = -\sum_{g=2}^k PG_g(s, d) ; 0 \leq \beta \leq 1 \tag{6}$$

When  $\beta$  is negative,  $PG_1(s, d)$  and  $PG_i(s, d)$  is obtained as:

$$PG_1(s, d) = \beta * PL_1(s, d) ; -1 \leq \beta < 0$$

$$PG_i(s, d) = -\frac{PG_1(s, d)}{(k - 1)} ; 2 \leq i \leq k \tag{7}$$

$PG_1(s, d)$  decreases as  $\beta$  decreases. However, the value of the inefficient backup routes  $PG_i(s, d)$  increases.

APBN confirms the utilization route using a stochastic confirmation method [10]. As node mobility is low during incipient packet transmission,  $PL_i(s, d)$  is obtained as formula (5). When current performance of the route selection policy deteriorates gradually,  $\xi$  times,  $PL_i(s, d)$  is switched to formula (6) and the impact parameter is switched to  $E_i(s, d)$ . If deterioration continues,  $PL_i(s, d)$  is switched from formula (6) to formula (5).  $PL_i(s, d)$  is switched between formula (5) and formula (6) in real time in response to transmission performance.

**C. Gradient Ascent Algorithm for Finding the Optimum Route Selection Policy**

Gradient ascent is an optimization algorithm. In control theory, gradient ascent refers to the algorithm used to find a maximum of a function in which steps proportional to the gradient (or the approximate gradient) of the function at the current point are taken [15]. Currently, the gradient ascent algorithm is more generally applied [16].

In Fig. 2,  $f$  times data transmission to the destination node is considered one iteration. Degree of preference  $\beta(t_x)$  of  $x$  iteration determines the current route selection policy between  $t_x$  time and  $t_{x+1}$  time. The packet delivery ratio is the ratio of correctly delivered data packets to total packets transmitted to the current destination node. Packet delivery ratio  $Q(\beta(t_x))$  of the total  $f$  times data transmission reflects the current performance of the route selection policy between  $t_x$  and  $t_{x+1}$ .  $\theta$  is step length of gradient ascent,  $\lambda(t_x)$  is an  $\beta(t_x)$  improved based on transmission performance, and  $Q(\beta(t_x))$  is calculated at time  $t_{x+1}$ . A source node keeps  $\beta(t_{x-1})$ ,  $Q(\beta(t_{x-1}))$ ,  $\beta(t_x)$  and  $Q(\beta(t_x))$  to each destination node at time  $t_{x+1}$ .

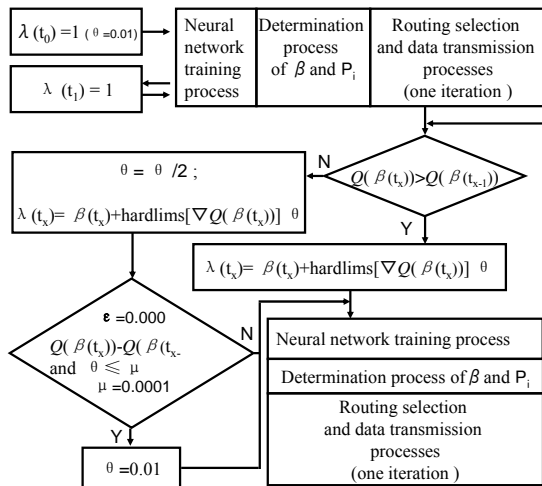


Figure 2.Gradient ascent algorithm application

The hardlims function is a symmetric hard limit transfer function.  $\nabla Q(\beta(t_x))$  is the approximate gradient of  $Q(\beta(t_x))$  at  $\beta(t_x)$  point.

$$\text{hardlims}(G) = \begin{cases} +1 & G > 0 \\ -1 & G \leq 0 \end{cases}$$

$$\nabla Q(\beta(t_x)) = \frac{\partial Q}{\partial \beta(t_x)} = \frac{Q(\beta(t_x)) - Q(\beta(t_{x-1}))}{\beta(t_x) - \beta(t_{x-1})} \quad (8)$$

Because the primary routing policy performs well in incipient communication network transmission conditions that are commonly considered to be low transmission load conditions, incipient  $\lambda(t_0)$  and  $\lambda(t_1)$  are initialized as 1.

From time  $t_1$ , the circle of gradient ascent begins to run along with sequence  $x=1,2,3,\dots$ .  $\beta(t_x)$  is defined by the trained neural network. At time  $t_{x+1}$ , if  $Q(\beta(t_x))$  is better than  $Q(\beta(t_{x-1}))$ , the current gradient is thought to be accurately directed toward maximum  $Q$ . In this case,  $\lambda(t_x)$  is also defined along the same direction. Otherwise,  $\lambda(t_x)$  is defined as a value that moves back a half step ( $\theta/2$ ) along the current direction from  $\beta(t_x)$ . The next circle of gradient ascent optimization begins from  $\beta(t_{x+1})$ , but with a new step length that is half as long ( $\theta/2$ ).

When both the step length is less than step boundary value  $\mu$  and the difference between  $Q(\beta(t_{x-1}))$  and  $Q(\beta(t_x))$  is less than convergence limit  $\epsilon$ ,  $Q$  is

considered to converge on the optimization value. However, when step length is less than step boundary value  $\mu$ , but the difference between  $Q(\beta(t_{x-1}))$  and  $Q(\beta(t_x))$  is not less than  $\epsilon$ , the convergence limit value, the step length  $\theta$  will be recovered to 0.01 to make  $\beta(t_{x+1})$  jump out of current search range. Then, the node begins a new  $\lambda$  optimization process from the  $\beta(t_{x+1})$  point. After the gradient ascent algorithm of the source node finds the target,  $\lambda(t_x)$ , based on their transmission performance, the neural network is trained to remember the relationship between global transmission conditions,  $\lambda(t_x)$ , and improved  $\beta(t_x)$  using the back propagation (BP) algorithm and to predict the next degree of preference  $\beta$  while taking account of global transmission conditions.

#### D. Neural Network Predicting the Next Degree of Preference

A source node with a back propagation neural network keeps predicting the degree of preference  $\beta$  for every destination node. A two-layer neural network with a sigmoid first layer and a linear second layer can be trained to approximate most relations well [17]. Currently, two-layer neural networks trained by back propagation techniques are more generally applied and are called back propagation neural networks [18-21]. Use of this neural network to predict optimum degree of preference  $\beta$ , as shown in Fig. 3, is a solvable problem. The neural network can give an output as a predicted degree of preference  $\beta(t_x)$  at time  $t_x$  for a destination  $d$ .

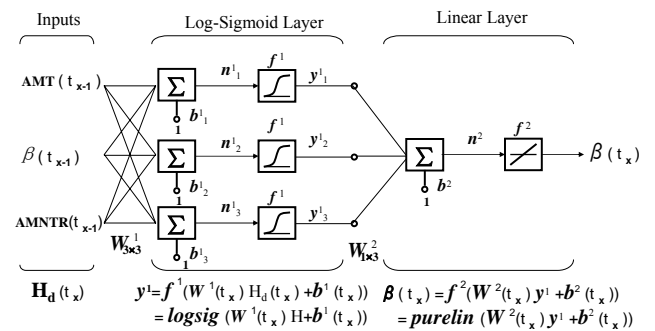


Figure 3. Two-layer neural network

$H_d(t_{x-1})$  is an input vector for destination node  $d$  that includes three inputs to the neural network. The average multiple route end-to-end packet delay time,  $AMT(t_{x-1})$ , is an average of the latest end-to-end packet delay times of the multiple routes between  $t_{x-1}$  and  $t_x$ . Degree of preference  $\beta(t_{x-1})$  reflects the policy between  $t_{x-1}$  and  $t_x$ . Average node throughput of route,  $ANTR_i(t_{x-1})$ , is the latest average value of node throughputs of all nodes along route  $i$  between  $t_{x-1}$  and  $t_x$ . Average multiple route node throughput of route  $AMNTR(t_{x-1})$  is an average of multiple routes  $ANTR_i(t_{x-1})$ .  $H_d(t_{x-1})$  can be computed as:

$$ANTR_g(t_{x-1}) = \text{Sum of node throughputs } (g, t_{x-1}) / E_g$$

$$AMT(t_{x-1}) = \left( \sum_{g=1}^k T_g(s,d) \right) / k$$

$$AMNTR(t_{x-1}) = \left( \sum_{g=1}^k ANTR_g(t_{x-1}) \right) / k \quad (9)$$

We append a superscript representing the number of the layer (1 or 2) to the names of each of these variables. Thus, the weight matrix for the first layer is written as  $W^1$ , and the weight matrix for the second layer is written as  $W^2$ . The present weight matrix,  $W^1(t_x)$ , is a  $3 \times 3$  matrix, and the present bias vector  $b^1(t_x)$  and output vector  $y^1(t_x)$  are also vectors of length 3 on the first layer.  $n^1(t_x)$  and  $y^1(t_x)$  can be computed as:

$$n^1(t_x) = W^1(t_x) H_d(t_x) + b^1(t_x)$$

$$y^1(t_x) = f^1 \left( \begin{bmatrix} n^1_1(t_x) \\ n^1_2(t_x) \\ n^1_3(t_x) \end{bmatrix} \right) = \text{logsig} \left( \begin{bmatrix} n^1_1(t_x) \\ n^1_2(t_x) \\ n^1_3(t_x) \end{bmatrix} \right)$$

$$= \begin{bmatrix} \frac{1}{1 + e^{-n^1_1(t_x)}} \\ \frac{1}{1 + e^{-n^1_2(t_x)}} \\ \frac{1}{1 + e^{-n^1_3(t_x)}} \end{bmatrix} \quad (10)$$

Then,  $n^2(t_x)$  can be computed as:

$$n^2(t_x) = W^2(t_x) y^1(t_x) + b^2(t_x) \quad (11)$$

The transfer function of the linear second layer is a pure-line transfer function.

$$\beta(t_x) = f^2(n^2(t_x)) = \text{purelin}(n^2(t_x)) = n^2(t_x) \quad (12)$$

Because a differentiable transfer function is necessary for the back propagation algorithm, the log-sigmoid transfer function is commonly used as transfer function in multilayer neural networks that are trained using the back propagation algorithm, partly because it is differentiable [18].

These incipient values of weight matrix  $W(t_x)$  and bias  $b(t_x)$  are small random values. In the initial  $m$  iterations, we do not use output of neural network as  $\beta$ . During this period, the node trains the neural network to converge on the optimum initial state. The  $m$  was set to 60 in our experiment.

After the two layer neural network predicts a degree of preference  $\beta(t_x)$  of  $x$  iteration and sends  $\beta(t_x)$  to the node at time  $t_x$ , a gradient ascent algorithm of source nodes identifies the improved  $\beta(t_x)$  as target  $\lambda(t_x)$  based on the transmission results for the practical utilization at time  $t_{x+1}$ .

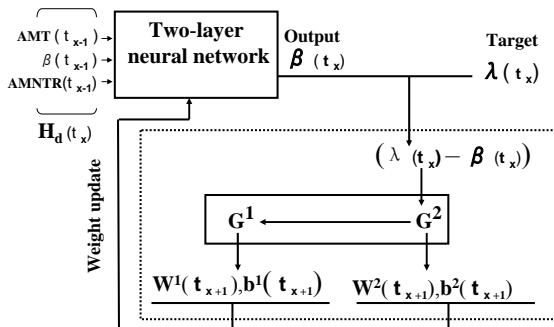


Figure. 4. Back propagation learning process

### E. Back Propagation Algorithm Training Neural Network

At time  $t_{x+1}$ , the back propagation method is used to train the neural network with target  $\lambda(t_x)$  and  $H_d(t_x)$ , as shown in Fig. 4. Then, new weight  $W(t_{x+1})$  and bias  $b(t_{x+1})$  enable the neural network to adjust output  $\beta$ . These weight matrices and biases memorize the learning experience, so the neural network becomes increasingly cognitive as it learns.

The back propagation method computes new weights and biases in the following steps.

First, it computes sensitivities  $G^2$  and  $G^1$ , shown in Fig. 4, as follows:

$$G^2(t_x) = -2\dot{F}^2(n^2(t_x))(\lambda(t_x) - \beta(t_x))$$

$$G^1(t_x) = \dot{F}^1(n^1(t_x))(W^2(t_x))^T G^2(t_x) \quad (13)$$

Weights and biases are computed starting from the last layer and propagate backwards to the first layer. Here,  $\dot{F}^2(n^2)$  and  $\dot{F}^1(n^1)$  are the derivatives of the transfer function and are obtained as follows:

$$\dot{F}^2(n^2(t_x)) = \dot{f}^2(n^2(t_x)) = \frac{d(n^2(t_x))}{d(n^2(t_x))} = 1$$

$$\dot{F}^1(n^1(t_x)) = \begin{bmatrix} \dot{f}^1(n^1_1(t_x)) & 0 & 0 \\ 0 & \dot{f}^1(n^1_2(t_x)) & 0 \\ 0 & 0 & \dot{f}^1(n^1_3(t_x)) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{d(f^1(n^1_1(t_x)))}{d(n^1_1(t_x))} & 0 & 0 \\ 0 & \frac{d(f^1(n^1_2(t_x)))}{d(n^1_2(t_x))} & 0 \\ 0 & 0 & \frac{d(f^1(n^1_3(t_x)))}{d(n^1_3(t_x))} \end{bmatrix} \quad (14)$$

The new weight and bias are computed as:

$$W^2(t_{x+1}) = W^2(t_x) - \delta G^2(t_x) (y^1(t_x))^T$$

$$b^2(t_{x+1}) = b^2(t_x) - \delta G^2(t_x)$$

$$W^1(t_{x+1}) = W^1(t_x) - \delta G^1(t_x) (H_d(t_x))^T$$

$$b^1(t_{x+1}) = b^1(t_x) - \delta G^1(t_x) \quad (15)$$

At time  $t_{x+1}$ , the neural network can give a  $\beta(t_{x+1})$  of  $x+1$  iterations for destination  $d$  with the input of  $H_d(t_{x+1})$  under the back propagation learning of  $x$  iterations. After output of a neural network is used to control practical route selection, the gradient ascent algorithm finds a  $\lambda(t_{x+1})$ , which is an improved  $\beta(t_{x+1})$ , based on transmission performance at next time point  $t_{x+2}$ . The back propagation learning method adjusts the weights and biases of the neural network. This relationship between  $H_d(t_{x+1})$  and  $\lambda(t_{x+1})$  can be memorized by the neural network.

## IV. PERFORMANCE EVALUATION

### A. Simulation Conditions

We evaluated our proposal using Omnet simulator [14]. Parameters are shown in Table I.

TABLE I.

SIMULATION PARAMETERS

Map nodes and time	
Map size	700 meters x 700 meters
Number of nodes	25
Number of source nodes enabled to transmit	One of 5, 10, 15, 20, or 25
Initial node arrangement	Random location in map
Experiment run repeat	3 times
Simulation time/one run	300 seconds
Experiment time/one run	20-40 hours
Node mobility model (one movement)	
Angle	Randomly distributed in $[0, 2\pi]$
Velocity	(1) Randomly distributed in $[0, 1]$ meter/sec (2) Randomly distributed in $[8, 9]$ meter/sec
Distance	Randomly distributed in $[0, 200]$ meters
Pause time	Randomly distributed in $[0, 5]$ seconds
Physical layer	
Radio range	40 meters
Channel bandwidth	11Mb/s (IEEE 802.11a)
Channel error probability	1 bit on $10^6$
Channel delay	10 $\mu$ seconds
Mac layer: promiscue mode	
Input buffer size	1 MB
Application	
Message packet size	512 bytes
Burst length	64 packets
Send packet rate	5 packets/sec
Control message size	64 bytes
Burst appear	Randomly distributed over $[0, 25]$ nodes
Burst interval	Randomly distributed over $[0.5, 1]$ seconds
Gradient ascent and back propagation learning algorithm	
Initial step length $\theta$	0.01
Convergence limit $\epsilon$	0.0001
Step boundary value $\mu$	0.0001
Back propagation learning rate $\delta$	0.8
$\xi$	6

**B. Performance Metrics**

The packet delivery ratio, average end-to-end delay, and routing overhead ratio were used as performance metrics. The packet delivery ratio is the ratio of the correctly delivered data packets to total packets transmitted and is obtained as follows:

$$\text{Packet Delivery Ratio} = \frac{\text{No. of packets delivered}}{\text{No. of packets sent}} \quad (16)$$

The number of delivered data packets is the sum of total delivered data packets received by all nodes. The number of sent data packets is the sum of total numbers of data packets sent by each node.

The average end-to-end delay is the average of all end-to-end delay of packet transmission.

The routing overhead ratio is the ratio of control packets to correctly delivered data packets, and is obtained as:

$$\text{Routing Overhead Ratio} = \frac{\text{No. of control packets sent}}{\text{No. of packets delivered}} \quad (17)$$

The control packets are route discovery packets (RREQ, RREP) and route maintenance packets (RERR).

**C. Simulation Results**

We simulated ARH, MOR, and APBN and made three comparisons within each of the three protocols at two kinds of node mobility speed. Figures 5 and 7 show that APBN is the best of the tested methods in terms of packet delivery ratio during changing conditions. APBN can observe intelligently and adopt optimum route selection policy.

As shown in Fig. 5, in the node mobility velocity range within  $[0, 1]$  meter/sec, with between five and ten traffic source nodes, ARH performs better than MOR. Our assertion is that the primary routing policy is predominant over load-balancing policy when the transmission load and node mobility velocity are low. As shown in Fig. 6, MOR, which also has between five and ten traffic source nodes and uses longer routes than ARH, has long end-to-end delay resulting in transmission failure more serious than ARH's.

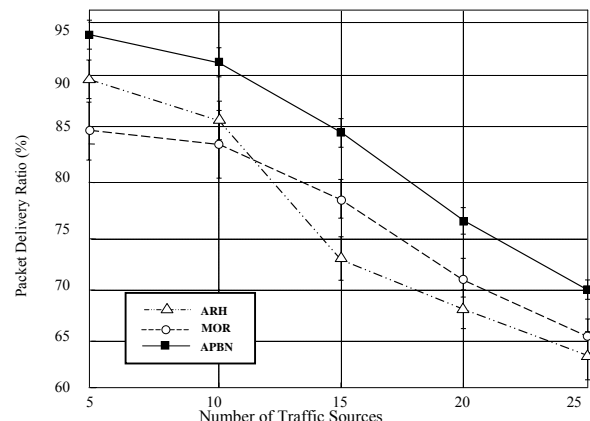


Figure 5. Comparison of packet delivery ratios during node mobile velocity distribution of  $[0, 1]$  meter/sec

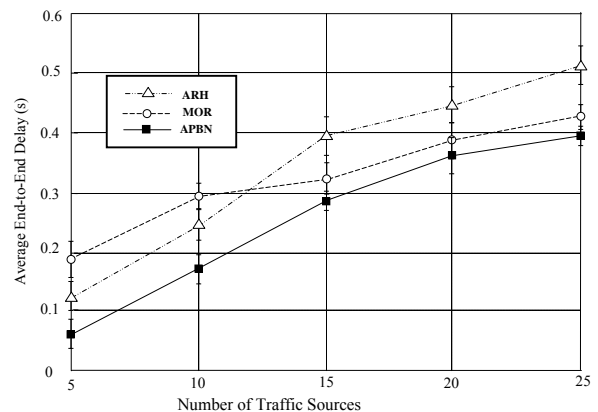


Figure 6. Comparison of average end-to-end delays during node mobile velocity distribution of  $[0, 1]$  meter/sec

As shown in Fig. 5, MOR surprisingly performs better than ARH when there are between 15 and 25 traffic source nodes, that is to say, the load-balancing policy is

predominant over primary routing policy when node mobility velocity is low but the transmission load is increasing. As shown in Fig. 6, when there are between 15 and 25 traffic source nodes, ARH shows longer end-to-end delay than MOR. This is due to the higher potential for congestion, not to utilization of longer routes. In low node mobility velocity conditions, the seriousness of the transmission failure caused by node mobility increases as end-to-end delay increases. The serious transmission failure caused by longer end-to-end delay plays a major role in these conditions.

As shown in Fig. 7, in the node mobility velocity ranges within [8,9] meter/sec, ARH performs better than MOR. In ARH, the primary route is the steadiest route with the least intermediate nodes. Our assertion is that the primary routing policy is predominant over load-balancing policy when node mobility velocity is high. When there is high node mobility velocity, the seriousness of transmission failure caused by node mobility increases as the number of intermediate nodes increases. The serious transmission failure caused higher numbers of intermediate nodes plays a major role in these conditions.

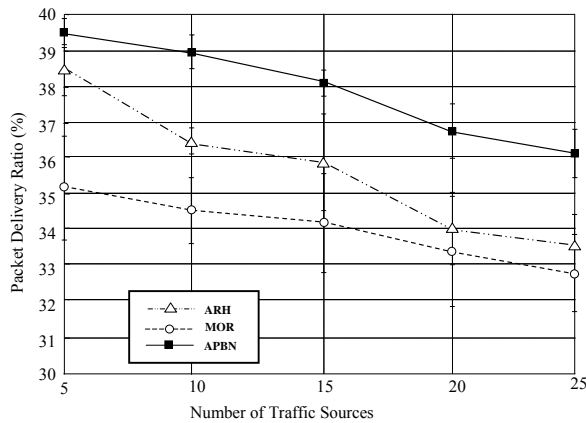


Figure 7. Comparison of packet delivery ratios at the node mobility speed distributed in [8,9]meter/sec

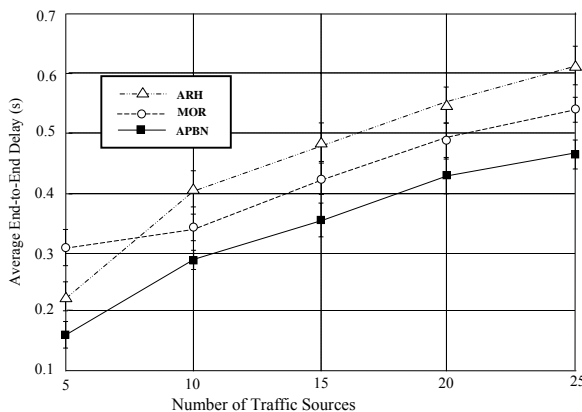


Figure 8. Comparison of average end-to-end delays during node mobile velocity distributed in [8,9]meter/sec

As shown in Fig. 8, when there are between ten and 25 traffic source nodes, although ARH has longer end-to-end delay than MOR, MOR has a lower packet delivery ratio than ARH, even with more intermediate nodes. These results support our assertion that one fixed route selection policy cannot fully adapt to various

communications network conditions. Only APBN fully adapts to changing communication network conditions and constantly shows good performance.

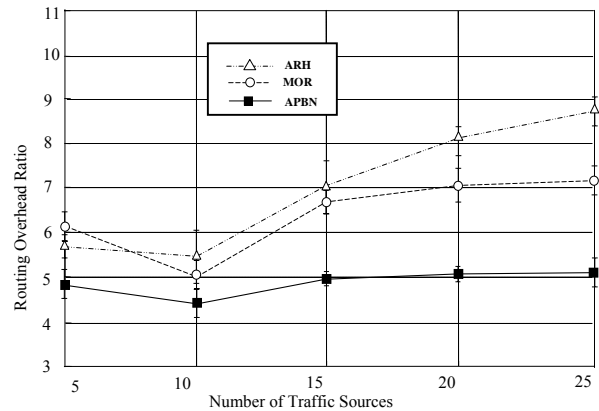


Figure 9. Comparison of routing overhead ratios during node mobile velocity distribution of [0,1]meter/sec

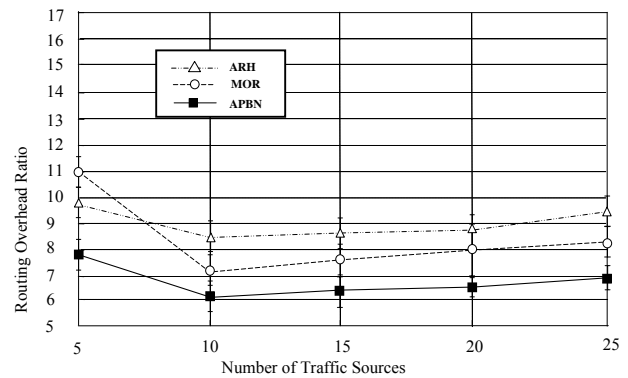


Figure 10. Comparison of routing overhead ratios at the node mobility speed distributed in [8,9]meter/sec

Figures 9 and 10 also show that the APBN protocol has lower routing overhead ratios and higher control packet restraints than the other three protocols in the node mobility velocity ranges of [0,1] meter/sec and [8,9] meter/sec, respectively.

### V. CONCLUSION

We proposed a new multiple route protocol with an adaptive route selection policy based on back propagation neural networks. The neural networks were used to predict the optimum route selection policy for adapting to changing network load conditions. The back propagation method was used to train the neural network to learn the relationship between different optimum route selection policies and the varying conditions of the communication network. Our simulations indicated that the new protocol has higher reliability (packet delivery ratio) and lower overhead (average end-to-end delay and routing overhead ratio) in intricately changing communication network conditions. Our results support our assertion that adaptive route selection policy control, rather than a fixed route selection policy, is necessary for routing protocols to perform well in varying communication network load conditions.

## ACKNOWLEDGMENTS

This research was partially supported by the Ministry of Education, Culture, Sports, Science and Technology, Grant-in-Aid for JSPS Fellows 1604285, Scientific Research on Priority Areas 18049038 and Scientific Research (C)18500050.

## REFERENCES

- [1] C. E. Perkins et al, "Ad Hoc On Demand Distance Vector (AODV) Routing," *RFC 3561*, July.2003.
- [2] J. Broch, D. Johnson, and D. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks," *Mobile Computing*, pp. 153-181, 1996.
- [3] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *IEEE INFOCOM '97*, Apr.1997.
- [4] S. J. Lee and M. Gerla, "AODV-BR: Backup Routing in Ad Hoc Networks," *IEEE WCNC 2000*, Sep.2000.
- [5] M. Jiang and R. Jan, "An Efficient Multiple Paths Routing Protocol for Ad-Hoc Networks," *IEEE ICOIN 2001*, Feb.2001.
- [6] S. R. Das et al, "On-demand Multipath Distance Vector Routing for Ad Hoc Networks," *IEEE ICNP 2001*, Nov.2001.
- [7] S. Motegi et al, "Proposal on Multipath Routing for Ad hoc Networks," *IEICE Technical Report*, IN2002-125 Nov,2002.
- [8] A. Nasipuri, R. Castaneda, and S. R. Das, "Performance of Multipath Routing for On-Demand Protocols in Mobile Ad Hoc Networks," *Mobile Networks and Applications*, pp. 339-349, Aug ,2001.
- [9] Fang Jing, R.S. Bhuvaneshwaran, Yoshiaki Katayama, and Naohisa Takahashi, "A Multipath On-Demand Routing with Path Selection Probabilities for Mobile Ad hoc Network," *In IEEE WCNM, 2005*, vol. 2, pp. 1145-1148.
- [10] Iwata, H., and Matsuo, H., "Media streaming on MANET using probabilistic routing ARH," *IPSJ SIG Mobile Computing and Ubiquitous Communications MoMuC2006-17*, May, 2006.
- [11] Biagioni, E. and Chen, S, "A Reliability Layer for Ad-Hoc Wireless Sensor Network Routing," *Proceedings of the 37th Hawaii International Conference on System Sciences*, Jan. 2004.
- [12] Fang Jing, R. S. Bhuvaneshwaran, Yoshiaki Katayama, and Naohisa Takahashi, "On-Demand Multipath Routing Protocol with Preferential Path Selection Probabilities for MANET," *In IEEE AINA 2006*, volume 2, pp. 758-762, Apr. 2006 .
- [13] Abrahamsson, H and Gunnar, A, "Traffic Engineering in Ambient Networks," *Swedish National Computer Networking Workshop*, November, 2004.
- [14] OMNET++, 2.3b distribution, <http://www.omnetpp.org/filemgmt/viewcat.php?cid=2>.
- [15] Mordecai Avriel, "Nonlinear Programming: Analysis and Methods," *Dover Publishing*, 2003.
- [16] Jan A. Snyman, "Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms," *Springer Publishing*, 2005.
- [17] Hornik. K. M., Stinchcombe. M., and White. H., "Multilayer feedforward networks are universal approximators," *Neural Network*, vol. 2, no. 5, pp. 359-366,1989.
- [18] Martin. T. H., Howard. B. D., Mark. B., "Neural Network Design, (ISBN 0-534-94332-2)," *Published by PWS Publishing Company*, 1996.
- [19] Rumelhart. D. E., G. E. Hinton, and R .J. Williams, "Learning internal representations by back propagating error," *Nature*, vol. 323, pp. 533-536,1986.
- [20] Werbos P. J., "Beyond regression: new tools for prediction and analysis in the behavioural sciences," Harvard University, Boston, 1974.
- [21] Parker D. B., "Learning-logic: Casting the cortex of the human brain in silicon," *Technical Report TR-47, Center for Research in Computational Economics and Management Science*, 1985.

**Fang Jing** currently pursuing PhD in the Department of Computer Science & Engineering at Nagoya Institute of Technology. Since 2004, his research interests include the routing algorithms for ad-hoc networks.

**R.S.Bhuvaneshwaran** received a doctorate from Nagoya University in 2005. He had been working as a foreigner researcher in Takahashi and Katayama lab of the Electrical and Computer Engineering Department at Nagoya Institute of Technology from 2005 to 2007. Now he is an assistant professor of Ramanujan Computing Centre, College of Engineering Guindy, Anna University.

**Yoshiaki Katayama** received his B.E., M.E. and D.E. in computer science from Osaka University in Japan. He had been working at Information Technology Center, Nara Institute of Science and Technology(NAIST) from 1994 to 2002. And he has joined Nagoya Institute of Technology(NITECH) in 2002. Now, he is an associate professor of department of computer science and engineering, graduate school of engineering, NITECH. His research interests include distributed algorithms, network applications and intelligent home appliance systems. He is a member of IEICE and IPSJ.

**Naohisa Takahashi** received B.E. and M.E. degrees in electrical engineering from the University of Electro-Communications in 1974 and 1976. He received a doctorate in computer science in 1987 from Tokyo Institute of Technology. He was with NTT Electrical Communications Laboratories from 1976 to 2001, where he was engaged in research on parallel processing, software engineering and computer networks. Since 2001 he has been with Nagoya Institute of Technology as a Professor. His research areas include mobile computing, advance network model and network security.