

Wavelet Neural Network Controller for AQM in a TCP Network: Adaptive Learning Rates Approach

Jae Man Kim, Jin Bae Park*, and Yoon Ho Choi

Abstract: We propose a wavelet neural network (WNN) control method for active queue management (AQM) in an end-to-end TCP network, which is trained by adaptive learning rates (ALRs). In the TCP network, AQM is important to regulate the queue length by passing or dropping the packets at the intermediate routers. RED, PI, and PID algorithms have been used for AQM. But these algorithms show weaknesses in the detection and control of congestion under dynamically changing network situations. In our method, the WNN controller using ALRs is designed to overcome these problems. It adaptively controls the dropping probability of the packets and is trained by gradient-descent algorithm. We apply Lyapunov theorem to verify the stability of the WNN controller using ALRs. Simulations are carried out to demonstrate the effectiveness of the proposed method.

Keywords: Adaptive learning, AQM, congestion control, wavelet neural network.

1. INTRODUCTION

Congestion control for a TCP network has been widely studied since data transmission through the Internet has been increased [1-4]. In the TCP network, data transmission over the allocated capacity of link causes packet drops, which results in the retransmissions of lost packets. Therefore, the core of congestion control for the TCP network is to inhibit an incipient congestion and to realize retrieval from the congested network condition.

Active queue management (AQM) was introduced as a methodology to control the end-to-end congestion in the Internet. AQM primarily responds to network congestion when a queue begins to increase and then maintains a queue size at a predefined level in the router. By keeping the average queue size small, AQM has the ability to preserve the efficient queue utilization and to reduce the delays occurred by the network flow, which is particularly important for real-time interactive applications. RED algorithm has been

proposed for AQM scheme, which is the first practical and most well-known method. RED algorithm lessens end-to-end delay time and prevents consecutive packet drops, which results in oscillation in link utilization and global synchronization [5,6]. However, it shows drawbacks in that the performance of RED algorithm is very sensitive to traffic load and its parameter setting, moreover, it is hard to reduce the queue fluctuation by only adjusting parameters in the RED algorithm [7].

PI and PID control methods also have been proposed for AQM scheme. They are control-theoretic AQM approaches, which are mostly used in classical linear control problems. Hollot [8] analyzed the effect on stability caused by network parameters based PI control method. This AQM scheme was compared with RED algorithm and demonstrated improvement on performance in [3]. The PID control method also shows better performance than RED in [10]. However, PI and PID control algorithms are problematic and unrealistic as well because they take the network as a linear and constant system even though the actual network environment is a nonlinear system [9].

On the other hand, wavelet neural network (WNN) has been used as the tool of identification or control for nonlinear systems. It has the advantage of multi-resolution of wavelets and fast convergence compared to multi-layer perceptron (MLP). The weights of WNN are usually trained by gradient-descent (GD) method, and the learning rates in the GD method are fixed arbitrary constants. However, since the fixed learning rates (FLRs) are not optimal learning rates, the system performance is sensitive to parameters.

Therefore, we propose a novel WNN control

Manuscript received July 11, 2007; accepted February 19, 2008. Recommended by Editor Young-Hoon Joo. This work was supported in part by Yonsei University Institute of TMS Information Technology, a Brain Korea 21 program and in part by MOCIE through EIRC program with Yonsei Electric Power Research Center (YEPRC) at Yonsei University, Seoul, Korea.

Jae Man Kim and Jin Bae Park are with the Department of Electrical and Electronic Engineering, Yonsei University, 262 Seongsanno, Seodaemun-gu, Seoul 120-749, Korea (e-mails: {utopiaro, jbpark}@yonsei.ac.kr).

Yoon Ho Choi is with the School of Electronic Engineering, Kyonggi University, Suwon, Kyonggi-do 443-760, Korea (e-mail: yhchoi@kyonggi.ac.kr).

* Corresponding author.

method based on adaptive learning rates (ALRS) for AQM in the TCP network. The proposed WNN controller can be applied to linear systems as well as time-varying nonlinear systems, since it is not sensitive to its parameters setting. The proposed method further has an advantage in that it repeatedly searches the optimal learning rates using Lyapunov theory, and changes the weighting vectors. The proposed WNN controller using optimal learning rates minimizes the error between the desired queue length and the network queue length.

This paper is organized as follows. In Section 2, we present the TCP nonlinear dynamic model and WNN based control theory for AQM. We also describe the learning method of WNN. Section 3 discusses the stability analysis of the system via ALRs. Simulation results are shown in Section 4, and the conclusions are presented in Section 5.

2. PRELIMINARIES

2.1. TCP network model

The nonlinear model for TCP flow control was proposed [6-8]. The simplified version that ignores the TCP timeout mechanism is as follows:

$$\begin{aligned} W(n+1) &= W(n) + \frac{1}{R(n)} \\ &\quad - \frac{W(n)W(n-R(n))}{2R(n-R(n))} p(n-R(n)), \quad (1) \\ q(n+1) &= q(n) + \frac{W(n)}{R(n)} N(n) - C, \end{aligned}$$

where $W(n)$ denotes TCP window size, $q(n)$ is the queue length at a router, $R(n)$ is round trip time (RTT) calculated by $q(n)/C + T_p$, C is link capacity, T_p is propagation delay, N is the number of TCP sessions (load factor), and $p(n)$ is the probability of packet loss. The AQM controller for the simplified and inaccurate linear TCP model is not optimal, because the real TCP network is rapidly changed, that is, the network parameters, the number of TCP sessions, and the capacity of the link are hardly kept at constant values for a long time. Therefore, it is necessary to design an adaptive controller for nonlinear network systems.

2.2. WNN structure

The WNN is designed as a four-layer structure [11,12]. Each layer has one or more nodes. Since the objective is to design a WNN controller which adjusts the queue length of the TCP network to the desired queue length, that is, the error $e_q = q^* - q$ should be minimized, where q and q^* represent the actual

queue length and the desired queue length, respectively. The inputs of WNN are e_q and p , which are previous step values. In the mother wavelet layer, each node performs a wavelet ϕ_j that is derived from its mother wavelet. For the j th node,

$$\phi(z_{jk}) = \phi\left(\frac{x_k - m_{jk}}{d_{jk}}\right), \quad (2)$$

where m_{jk} and d_{jk} are the translation and dilation in the j th term of the k th input to the node of the mother wavelet layer, respectively. There are many kinds of mother wavelets that can be used in WNN. In this paper, the first derivative of a Gaussian function, $\phi(z) = -z \exp(-0.5z^2)$, is selected as a mother wavelet. In addition, each node l in the wavelet layer is denoted by Π which multiplies the mother wavelet outputs. For the l th rule node,

$$\Phi_l(x) = \Pi \phi(z_{jk}). \quad (3)$$

The output of WNN, which is a control input of the TCP model, is composed by each wavelet and parameters as follows:

$$y = \Phi(x, \theta) = \sum_{l=1}^{N_w} c_l \Phi_l(x) + \sum_{k=1}^{N_i} a_k x_k + a_0, \quad (4)$$

where a_0 and a_k are connection weight between input nodes and output nodes, respectively. N_i and N_w represent the number of nodes in the input layers and the product layers, respectively. c_l is a connection weight between wavelet nodes and output nodes, and Ω is the set of adjustable parameters:

$$\Omega = \{m_{jk}, d_{jk}, c_l, a_k, a_0\}. \quad (5)$$

2.3. Training algorithm for WNN weights

The core part of the training algorithm for WNN concerns how to obtain gradient vectors. Each of the elements in the training algorithm are defined as the derivative of cost function with respect to a parameter of the network, and this is done by means of the chain rule and the back-propagation (BP) learning rule.

A direct adaptive controller based on WNN is considered in Fig. 1. The error, which is the difference between the output of AQM of the TCP network and the desired queue length, is the input of a direct adaptive controller.

To describe the training algorithm of WNN using the GD method, the cost function is defined as follows:

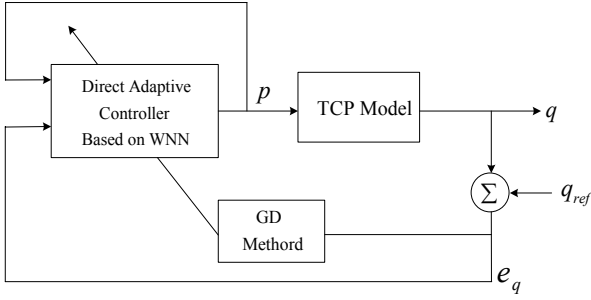


Fig. 1. Control structure for TCP model.

$$J(\Omega(n)) = \frac{1}{2} (q_{ref}(n) - q(n))^2 = \frac{1}{2} e_q^2(n), \quad (6)$$

where $q(n)$ is the current output value of the TCP model and $q_{ref}(n)$ is the desired output value. The training method is based on the minimization of the cost function. The minimization is performed by the following GD method:

$$\begin{aligned} \Omega(n+1) &= \Omega(n) - \Delta\Omega(n) \\ &= \Omega(n) - \eta \frac{\partial J(\Omega(n))}{\partial \Omega(n)}, \end{aligned} \quad (7)$$

where η is the learning rate of a WNN.

The partial derivative of the cost function with respect to weighting vector can be represented as follows:

$$\frac{\partial J(\Omega(n))}{\partial \Omega(n)} = -e_q(n) \frac{\partial q(n)}{\partial \Omega(n)} = -e_q(n) \frac{\partial q(n)}{\partial u(n)} \frac{\partial u(n)}{\partial \Omega(n)}, \quad (8)$$

where $u(n) = p(n)$, and $\frac{\partial q(n)}{\partial u(n)}$ is the system sensitivity. The partial derivatives of the control input $u(n)$ with respect to weighting vector Ω are as follows:

$$\frac{\partial u(n)}{\partial a_0(n)} = 1, \quad (9)$$

$$\frac{\partial u(n)}{\partial a_k(n)} = x_k, \quad (10)$$

$$\frac{\partial u(n)}{\partial c_l} = \Phi_l(x), \quad (11)$$

$$\frac{\partial u(n)}{\partial m_{jk}} = -\frac{c_l}{d_{jk}} \frac{\partial \Phi_l(x)}{\partial z_{jk}}, \quad (12)$$

$$\frac{\partial u(n)}{\partial d_{jk}} = -\frac{c_l}{d_{jk}} z_{jk} \frac{\partial \Phi_l(x)}{\partial z_{jk}}, \quad (13)$$

where

$$\frac{\partial \Phi_l(x)}{\partial z_{jk}} = \phi(z_{j1}) \phi(z_{j2}) \cdots \dot{\phi}(z_{jk}) \cdots \phi(z_{jN_i}), \quad (14)$$

$$\dot{\phi}(z_{jk}) = \frac{\partial \phi(z_{jk})}{\partial z_{jk}} = (z_{jk}^2 - 1) \exp\left(-\frac{1}{2} z_{jk}^2\right).$$

3. STABILITY ANALYSIS VIA ALRS

The update rule of weighting vectors calls for the proper choice of learning rate η [13]. For a small value of η , the convergence is guaranteed but the convergence speed is very slow. On the other hand if η is too large, the algorithm becomes unstable. Hence, this section develops a guideline for the selection of proper learning rates.

Let us define a discrete Lyapunov function as follows:

$$V(n) = \frac{1}{2} [e_q(n)]^2, \quad (15)$$

where $e_q(n)$ represents the error between the network queue length and the reference queue value.

The error difference due to the learning rate can be expressed as follows:

$$\Delta e_q(n) = e_q(n+1) - e_q(n) \approx \left[\frac{\partial e_q(n)}{\partial \Omega} \right] \Delta \Omega, \quad (16)$$

where

$$\Delta \Omega = \eta e_q(n) \frac{\partial q(n)}{\partial u(n)} \frac{\partial u(n)}{\partial \Omega(n)}, \quad (17)$$

denotes the change in an arbitrary weighting vector, and η represents the corresponding learning rate in WNN.

Then, the change of Lyapunov function due to the training process can be represented by

$$\begin{aligned} \Delta V(n) &= V(n+1) - V(n) \\ &= \frac{1}{2} [e_q^2(n+1) - e_q^2(n)] \\ &= \Delta e_q(n) \left[e_q(n) + \frac{1}{2} \Delta e_q(n) \right] \\ &= \left[\frac{\partial e_q(n)}{\partial \Omega(n)} \right] \eta e_q(n) \frac{\partial q(n)}{\partial u(n)} \frac{\partial u(n)}{\partial \Omega(n)} \\ &\quad \times \left[e_q(n) + \frac{1}{2} \left[\frac{\partial e_q(n)}{\partial \Omega(n)} \right]^T \eta e_q(n) \frac{\partial q(n)}{\partial u(n)} \frac{\partial u(n)}{\partial \Omega(n)} \right] \\ &= - \left[\frac{\partial u(n)}{\partial \Omega(n)} \right]^T G \eta e_q(n) G \frac{\partial u(n)}{\partial \Omega(n)} \\ &\quad \times \left[e_q(n) - \frac{1}{2} \left[\frac{\partial u(n)}{\partial \Omega(n)} \right]^T G \eta e_q(n) G \frac{\partial u(n)}{\partial \Omega(n)} \right] \end{aligned}$$

$$\begin{aligned}
&= -e_q^2(n)G^2 \left[\eta \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \right] \left(1 - \frac{1}{2} \eta G^2 \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \right) \\
&= -\lambda e_q^2(n)G^2,
\end{aligned} \tag{18}$$

where

$$\lambda = \eta \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \left(1 - \frac{1}{2} \eta G^2 \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \right), \tag{19}$$

$$G = \frac{\partial q(n)}{\partial u(n)}. \tag{20}$$

Theorem 1: Let

$$\eta = [\eta^1, \eta^2, \eta^3, \eta^4, \eta^5]^T = [\eta^{a_0}, \eta^{a_k}, \eta^c, \eta^d, \eta^m]^T$$

be the learning rates for the tuning parameters of WNN, and define U_{\max} as

$$\begin{aligned}
U_{\max} &= [U_{1,\max} \ U_{2,\max} \ U_{3,\max} \ U_{4,\max} \ U_{5,\max}]^T \\
&= \left[\max \left\| \frac{\partial u(n)}{\partial a_0(n)} \right\| \ \max \left\| \frac{\partial u(n)}{\partial a_k(n)} \right\| \ \max \left\| \frac{\partial u(n)}{\partial c(n)} \right\| \right. \\
&\quad \left. \max \left\| \frac{\partial u(n)}{\partial d(n)} \right\| \ \max \left\| \frac{\partial u(n)}{\partial m(n)} \right\| \right].
\end{aligned} \tag{21}$$

Then, asymptotic convergence is guaranteed if η^i is chosen to satisfy

$$0 < \eta^i < \frac{2}{G^2 (U_{i,\max})^2}, \quad i = 1, \dots, 5. \tag{22}$$

Proof: In (19),

$$\begin{aligned}
\lambda &= \eta \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \left(1 - \frac{1}{2} \eta G^2 \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \right) \\
&\geq \eta \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \left(1 - \frac{1}{2} \eta G^2 (U_{i,\max})^2 \right).
\end{aligned} \tag{23}$$

Since the change of Lyapunov function is

$$\Delta V(n) = -\lambda e_q^2(n)G^2, \tag{24}$$

the convergence of WNN is guaranteed if $\lambda > 0$. This completes the proof. \square

Corollary 1: The maximum learning rates which guarantee the convergence are

$$\eta_{\max}^i = \frac{1}{G^2 (U_{i,\max})^2}, \quad i = 1, \dots, 5. \tag{25}$$

Proof: Since in Theorem 1,

$$\begin{aligned}
\lambda &= \eta \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \left(1 - \frac{1}{2} \eta G^2 \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \right) \\
&\geq - \left\| \frac{\partial u(n)}{\partial \Omega(n)} \right\|^2 \left[\frac{1}{2} (U_{i,\max})^2 G^2 \left(\eta - \frac{1}{G^2 (U_{i,\max})^2} \right) \right. \\
&\quad \left. - \frac{1}{2G^2 (U_{i,\max})^2} \right] > 0.
\end{aligned} \tag{26}$$

Then, from (22) and (26), we can obtain (25). This completes the proof. \square

Theorem 2: Let η^{a_0} be the learning rate for weight a_0 . Then, the asymptotic convergence is guaranteed if the learning rate satisfies

$$0 < \eta^{a_0} < \frac{2}{G^2}. \tag{27}$$

Proof: Since

$$U_1(n) = \frac{\partial u(n)}{\partial a_0} = 1, \tag{28}$$

then, $\|U_1(n)\|^2 = 1 \leq \|U_{a_0,\max}\|^2$. Therefore, from Theorem 1, we find (27). This completes the proof. \square

Theorem 3: Let η^{a_k} be the learning rate for the weight a_k . Then, the asymptotic convergence is guaranteed if the learning rate satisfies

$$0 < \eta^{a_k} < \frac{2}{G^2 N_i |x_{\max}|^2}. \tag{29}$$

Proof: Since

$$U_2(n) = \frac{\partial u(n)}{\partial a_k} = X, \tag{30}$$

where $X = [x_1, x_2, \dots, x_{N_i}]$ is the input of WNN, we have $\|U_2(n)\|^2 \leq N_i \|U_{2,\max}\|^2$. Therefore, from Theorem 1, we have (29). This completes the proof. \square

Theorem 4: Let η^c be the learning rate for the weight c . Then, the asymptotic convergence is guaranteed if the learning rate satisfies

$$0 < \eta^{a_0} < \frac{2}{N_w}. \tag{31}$$

Proof: Since

$$U_3(n) = \frac{\partial u(n)}{\partial c} = \Phi, \tag{32}$$

where $\Phi = [\Phi_1, \Phi_2, \dots, \Phi_{N_w}]$ is the output of the wavelet layer, we have $\Phi_l \leq 1$ for all l , $\|U_3(n)\| \leq \sqrt{N_w}$. Therefore, from Theorem 1, we have (31). This completes the proof. \square

In order to prove Theorem 5, the following lemmas are used.

Lemma 1:

Let $f(t) = t \exp(-t^2)$, then $|f(t)| < 1, \forall t \in R$.

Lemma 2:

Let $g(t) = t^2 \exp(-t^2)$, then $|g(t)| < 1, \forall t \in R$.

Theorem 5: Let η^m, η^d be the learning rates for the translation and dilation weights, respectively. Then, the asymptotic convergence is guaranteed if the learning rates satisfy

$$0 < \eta^m < \frac{2}{G^2 N_i N_w} \left[\frac{1}{|c_{\max}| \left(\frac{2 \exp(-0.5)}{|d_{\min}|} \right)} \right]^2, \quad (33)$$

$$0 < \eta^d < \frac{2}{G^2 N_i N_w} \left[\frac{1}{|c_{\max}| \left(\frac{2 \exp(0.5)}{|d_{\min}|} \right)} \right]^2, \quad (34)$$

where N_w is the number of nodes in the product layer of WNN.

Proof: The learning rate η^m of the translation weight m :

$$\begin{aligned} U_5(n) &= \frac{\partial u(n)}{\partial m(n)} \\ &< \sum_{l=1}^{N_w} c_l \sum_{k=1}^{N_i} \max \left(\frac{\partial \phi(z_{jk})}{\partial z_{jk}} \frac{\partial z_{jk}}{\partial m} \right) \\ &< \sum_{l=1}^{N_w} c_l \sum_{k=1}^{N_i} \max \left(2 \exp(-0.5) \left(-\frac{1}{d} \right) \right). \end{aligned} \quad (35)$$

According to Lemma 2, we obtain

$$\left| \left(\frac{1}{2} z_{jk}^2 - \frac{1}{2} \right) \exp \left\{ - \left(\frac{1}{2} z_{jk}^2 - \frac{1}{2} \right) \right\} \right| < 1, \quad (36)$$

$$\begin{aligned} \|U_5(n)\| &< \sum_{l=1}^{N_w} c_l \sqrt{N_i} \left(\frac{-2 \exp(-0.5)}{d_{\min}} \right) \\ &< \sqrt{N_w} \sqrt{N_i} |c_{\max}| \left| \frac{2 \exp(-0.5)}{d_{\min}} \right|. \end{aligned} \quad (37)$$

Therefore, from Theorem 1, we find (33).

The learning rate η^d of the translation weight d :

$$\begin{aligned} U_4(n) &= \frac{\partial u(n)}{\partial d(n)} \\ &< \sum_{l=1}^{N_w} c_l \sum_{k=1}^{N_i} \max \left(\frac{\partial \phi(z_{jk})}{\partial z_{jk}} \frac{\partial z_{jk}}{\partial d} \right) \\ &< \sum_{l=1}^{N_w} c_l \sum_{k=1}^{N_i} \max \left(2 \exp(0.5) \left(\frac{1}{d} \right) \right). \end{aligned} \quad (38)$$

According to Lemmas 1 and 2, we obtain

$$|z_{jk} \exp(-z_{jk}^2)| < 1, \quad (39)$$

$$\left| \left(\frac{1}{2} - \frac{1}{2} z_{jk}^2 \right) \exp \left\{ - \left(\frac{1}{2} - \frac{1}{2} z_{jk}^2 \right) \right\} \right| < 1, \quad (40)$$

$$\begin{aligned} \|U_4(n)\| &< \sum_{l=1}^{N_w} c_l \sqrt{N_i} \left(\frac{2 \exp(0.5)}{d_{\min}} \right) \\ &< \sqrt{N_w} \sqrt{N_i} |c_{\max}| \left| \frac{2 \exp(0.5)}{d_{\min}} \right|. \end{aligned} \quad (41)$$

From Theorem 1, we obtain (34). This completes the proof. \square

Remark 1: From Corollary 1, the maximum learning rates of the WNN are as follows:

$$\eta^{a0} = \frac{1}{G^2}, \quad (42)$$

$$\eta^{ak} = \frac{1}{G^2 N_i |x_{\max}|^2}, \quad (43)$$

$$\eta^c = \frac{1}{N_w}, \quad (44)$$

$$\eta^m = \frac{1}{G^2 N_i N_w} \left[\frac{1}{|c_{\max}| \left(\frac{2 \exp(-0.5)}{|d_{\min}|} \right)} \right]^2, \quad (45)$$

$$\eta^d = \frac{1}{G^2 N_i N_w} \left[\frac{1}{|c_{\max}| \left(\frac{2 \exp(0.5)}{|d_{\min}|} \right)} \right]^2. \quad (46)$$

4. SIMULATIONS

In this section, we verify the performances of the proposed WNN controller using ALRs through computer simulations, and then compare the results

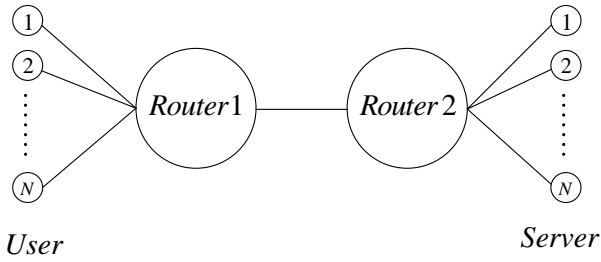


Fig. 2. TCP network topology.

with the performances of the PID controller and WNN controller using FLRs. We use a simple bottleneck network topology as shown in Fig. 2. The capacity of bottleneck link between routers 1 and 2 is 15Mbps. The propagation delay is 15ms. All links between users and router 1 have 10Mbps capacity and those with propagation delay are also 15ms. The size of all queues is 300 packets. The number of TCP sessions is 120, and the reference queue length is 200 packets [14]. We get the parameters of PID controller in [10] and set the values of initial ALRs and FLRs to 0.0001.

We carry out the simulations on two TCP network conditions. In static network condition, the simulations are performed by fixing the desired queue length and the number of TCP sessions to a constant value. And, in dynamic network condition, the simulations are performed by changing the desired queue length and the number of TCP sessions.

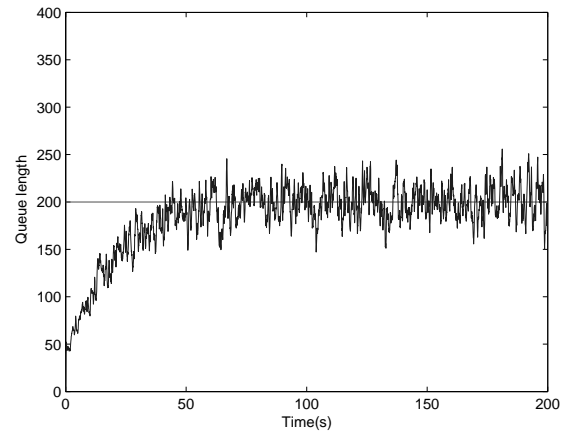
4.1. Static TCP network

First, we compare the performances of three controllers: PID, WNN controllers using FLRs, and ALRs. The instantaneous queue lengths for these three controllers are shown in Fig. 3. As presented in Fig. 3, the plot for the PID controller extremely oscillates at the desired queue length of 200 packets. That is, the performance of PID controller is not adaptive to change upon the environment of the TCP network. The WNN controller using FLRs makes the system more stable than the PID controller. However, it also shows a drawback in that there exists a gap between the queue length and the desired value of 200 packets. The WNN controller using ALRs demonstrates the acceptable performance. It decreases the packets drop, therefore transmits more packets through the router in TCP network than other controllers.

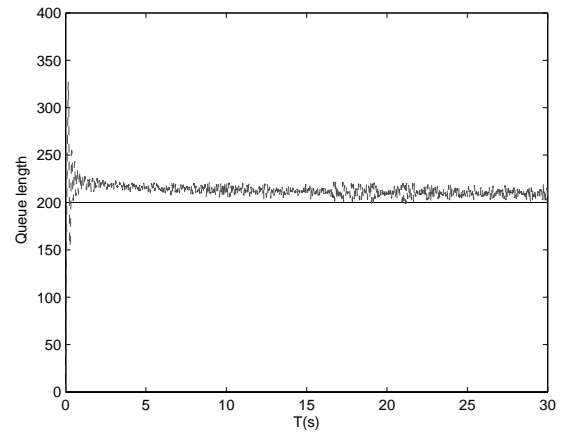
4.2. Dynamic TCP network

4.2.1 Dynamic reference queue length

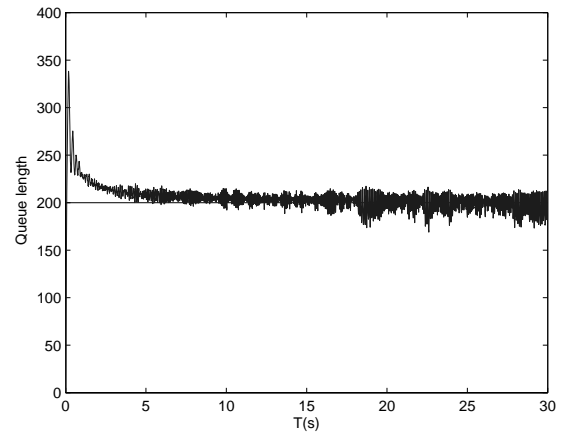
In this simulation, we investigate the ability of each scheme when the reference queue length (q_{ref}) varies from 200 packets to 150 packets. The router is more congested by the reduction of reference length. Fig. 4 shows the queue lengths for the PID controller, WNN controllers using FLRs, and ALRs. For PID controller and WNN controller using FLRs, more



(a) PID controller.



(b) WNN controller using FLRs.



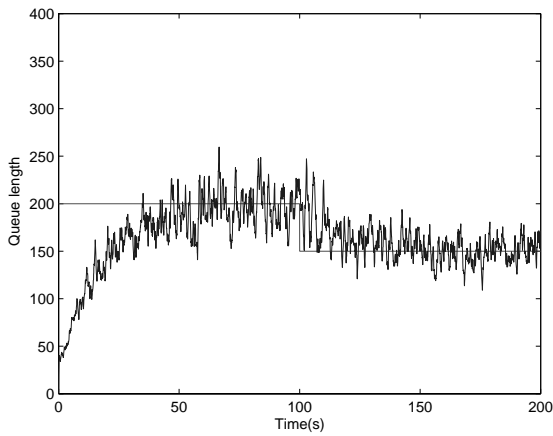
(c) WNN controller using ALRs.

Fig. 3. Queue length for static TCP network.

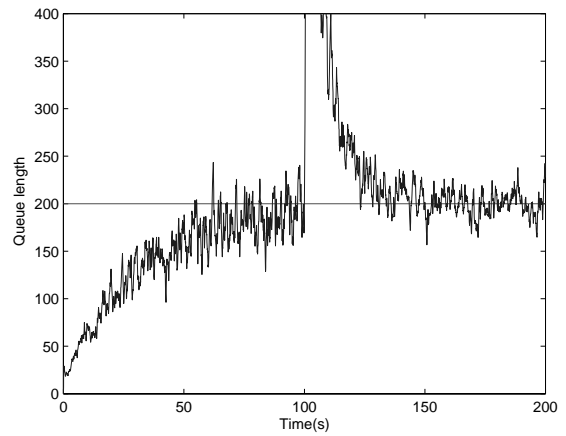
packets are dropped. However, WNN controller using ALRs make the system adaptive to the change of reference queue length, therefore, the packet drop is uniform before and after the change of reference queue length.

4.2.2 Dynamic TCP sessions

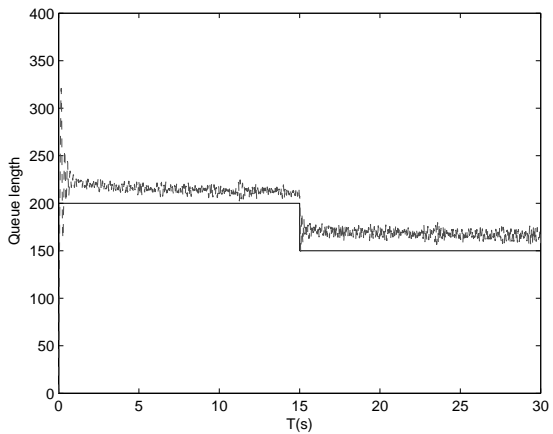
We consider the case that the number of TCP sessions changes from 100 to 200, and the value of other parameters is equal to the simulations



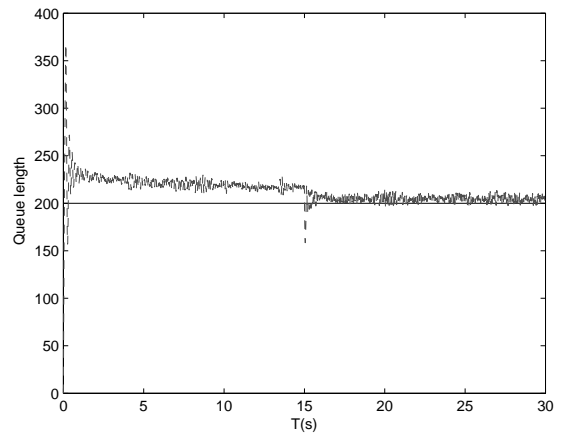
(a) PID controller.



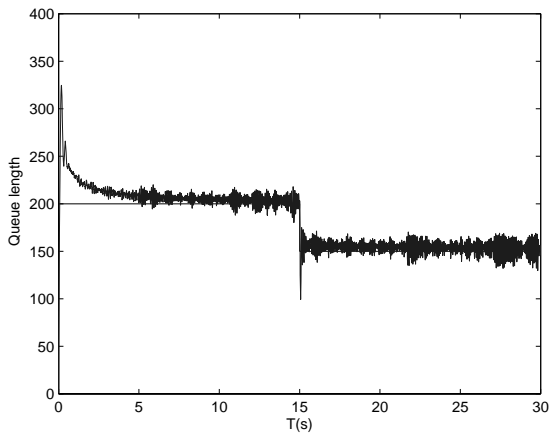
(a) PID controller.



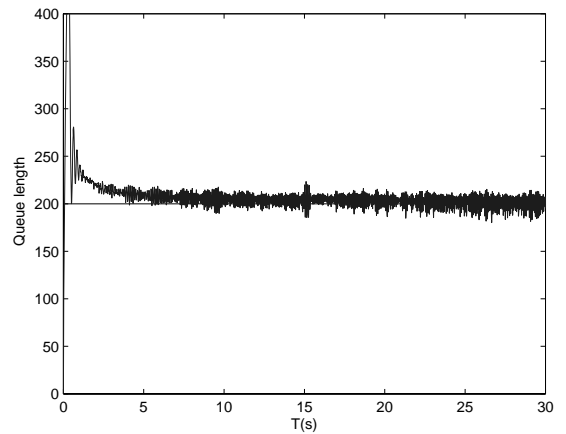
(b) WNN controller using FLRs.



(b) WNN controller using FLRs.



(c) WNN controller using ALRs.



(c) WNN controller using ALRs.

Fig. 4. Queue length for dynamic TCP network (Changing reference queue length).

Fig. 5. Queue length for dynamic TCP network (Changing TCP sessions).

considered in 4.1 Fig. 5 shows the queue lengths for three controllers. In Fig. 5, we can see that the PID controller has the overshoot at the time when the TCP session changes to 200. But WNN controllers using FLRs and ALRs do not have the occurrence of overshoot and show the uniform queue length unrelated to the number of TCP sessions. These results also demonstrate that the WNN controller using ALRs has faster adaptability for the dynamic

TCP sessions than others.

5. CONCLUSIONS

The adaptive WNN controller using ALRs has been proposed for AQM in the TCP network. The WNN controller using ALRs is operated as a direct adaptive controller, where the output is a dropping probability of packets at the router. The proposed controller

maintains the actual queue size close to a reference queue value, which is trained by the GD method with ALRs. By using the Lyapunov theorem, we obtain the ALRs for the stable queue length, and show the stability of the whole control scheme. Through simulation results, we show that the proposed control method is more adaptive than PID and WNN using FLR controllers in both the static and the dynamic TCP network systems.

REFERENCES

- [1] R. Johari and H. T. Kim, "End-to-end congestion control for the internet: Delays and stability," *IEEE/ACM Trans. on Networking*, vol. 9, no. 6, pp. 818-832, December 2001.
- [2] S. Deb and R. Srikant, "Global stability of congestion controllers for the internet," *IEEE Trans. on Automatic Control*, vol. 48, no. 6, pp. 1055-1060, June 2003.
- [3] C. V. Hollet, V. Misra, D. Towsley, and W. Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Trans. on Automatic Control*, vol. 47, no. 6, pp. 945-959, June 2002.
- [4] F. Yanfei, R. Fengyuan and L. Chuang, "Design a PID controller for active queue management," *Proc. of IEEE International Symp. on Computers and Communication*, vol. 2, pp. 985-990, June 2003.
- [5] B. Braden and D. Clark, "Recommendations on queue management and congestion avoidance in the internet," *IETF Request for Comments*, RFC 2309, 1989.
- [6] V. Misra, W. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," *Proc. of ACM/SIGCOMM*, pp. 151-160, August 2000.
- [7] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, "Tuning RED for web traffic," *Proc. of ACM/SIGCOMM*, pp. 139-150, August 2000.
- [8] C. V. Hollet, V. Misra, D. Towsley, and W. Gong, "On designing improved controllers for AQM routers supporting TCP flows," *Proc. of IEEE INFOCOM*, vol. 3, pp. 1726-1734, April 2001.
- [9] H. C. Cho, M. S. Fadali, and H. Lee, "Neural network control for TCP network congestion," *Proc. of American Control Conference*, vol. 5, pp. 3480-3485, June 2005.
- [10] S. Ryu, C. Rump, and C. Qiao, "A predictive and robust active queue management for internet congestion control," *Proc. of IEEE Sym. on Computers and Communication*, vol. 2, pp. 991-998, June 2003.
- [11] F. J. Lin, R. J. Wai, and P. K. Huang, "Two-axis motion control system using wavelet neural network for ultrasonic motor drives," *IEE Proc. Electric Power Applications*, vol. 151, no. 5, pp. 613-621, September 2004.
- [12] C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control," *IEEE Trans. on Neural Networks*, vol. 6, no. 1, pp. 144-156, January 1995.
- [13] S. J. Yoo, J. B. Park, and Y. H. Choi, "Direct adaptive control using self recurrent wavelet neural network via adaptive learning rates for stable path tracking of mobile robots," *Proc. of American Control Conference*, vol. 1, pp. 288-293, June 2005.
- [14] R. Fengyuan, R. Yong, and S. Xiuming, "Design of a fuzzy controller for active queue management," *Computer Communications*, vol. 25, no. 9, pp. 874-883, June 2002.



Jae Man Kim received the B.S. and M.S. degrees from Yonsei University, Seoul, Korea, in 2005 and 2007, respectively, both in Electrical and Electronic Engineering. He is currently working toward a Ph.D. degree at Yonsei University. His research interests include neural networks, congestion control, time-delayed

systems, nonlinear theories, and adaptive control.



Jin Bae Park received the B.S. degree in Electrical Engineering from Yonsei University, Seoul, in 1977 and the M.S. and Ph.D. degrees in Electrical Engineering from Kansas State University, Manhattan, in 1985 and 1990, respectively. Since 1992 he has been with the Department of Electrical and Electronic Engineering, Yonsei

University, Seoul, where he is currently a Professor. His research interests include robust control and filtering, nonlinear control, mobile robots, fuzzy logic control, neural networks, and genetic algorithms. He had served as Vice-President for the Institute of Control, Automation, and Systems Engineers. He currently serves as Editor-in-Chief for the International Journal of Control, Automation, and Systems.



Yoon Ho Choi received the B.S., M.S., and Ph.D. degrees in Electrical Engineering from Yonsei University, Seoul, Korea, in 1980, 1982, and 1991, respectively. Since 1993, he has been with the School of Electronic Engineering at Kyonggi University, where he is currently an Associate Professor. From 2000 to 2002, he was

with the Department of Electrical Engineering, Ohio State University, where he was a Visiting Scholar. His research interests include intelligent control, mobile robots, web-based control systems, and wavelet transform. He was the Director for the Institute of Control, Automation, and Systems Engineers from 2003 to 2004.