Neuron PID: A Robust AQM Scheme

Jinsheng Sun*, Sammy Chan†, King-Tim Ko†, Guanrong Chen† and Moshe Zukerman‡
*School of Automation, Nanjing University of Science and Technology, Nanjing, 210094, China
Email: sunjs@mail.njust.edu.cn
†Department of Electronic Engineering, City University of Hong Kong, Hong Kong SAR, China
Email: {eeschan,eektko,eegchen}@cityu.edu.hk
‡Australia Research Council Special Research Centre for Ultra-Broadband Information Networks (CUBIN),
EEE Department, The University of Melbourne, Victoria, 3010, Australia
Email: m.zukerman@ee.unimelb.edu.au

Abstract— **We introduce a novel and robust active queue management (AQM) scheme based on a Proportional-Integral-Derivative (PID) controller, called Neuron PID, that uses an adaptive neuron to tune its parameters. We demonstrate by simulations that Neuron PID stabilizes the router queue length regardless of the round trip delay better than other well-known AQM algorithms.**

I. INTRODUCTION

Internet congestion control relies on the end-to-end transmission control protocol (TCP) that adapts its send rate according to congestion indications and an active queue management (AQM) mechanism that signals congestion by dropping or marking packets at the congested router. AQM has been a very active research area [1], [2], [3], [4], [6], [7], [8], [10], [11], [12]. AQM aims to: 1) stabilize the queue lengths of routers at a given target, thereby reducing packet loss and making packet latency predictable, and 2) ensure high link utilization. AQM schemes normally rely on a set of parameters that indicate when to drop/mark a packet. Despite extensive research, optimizing these parameters adaptively is still a challenge. In this paper, a novel AQM scheme based on a Proportional-Integral-Derivative (PID) controller that uses an adaptive neuron to tune its parameters is proposed. It is called *Neuron PID*. The need for such an intelligent controller is justified because the Internet congestion control system is complex and highly nonlinear. Accordingly, we apply the ideas of [5], [13], where an adaptive neuron PID controller is designed for a multi-model plant. Simulation results show that Neuron PID can control the queue length and it quickly converges to a desirable target even after significant changes in the system, and even for long delay networks. We will show that it is more efficient and stable than other well-known AQM schemes.

II. NEURON PID

A PID controller-based AQM scheme can be described by,

$$p(k) = K_p e(k) + K_i \sum_{i=1}^{k} e(i) + K_d(e(k) - e(k-1)) \quad (1)$$

where Proportional Gain K_p, Integration Gain K_i and Derivative Gain K_d are the parameters of the PID controller, p(k) is the packets dropping/marking probability,

$$e(k) = q(k) - Q_T \quad (2)$$

the queue length error, $q(k)$ the queue length, and Q_T the target buffer occupancy. Eq. (1) can be written in an incremental form

$$\begin{aligned} p(k) \quad = \quad & p(k-1) + K_p(e(k) - e(k-1)) + K_i e(k) + \\ & K_d(e(k) - 2e(k-1) + e(k-2)). \end{aligned} \quad (3)$$

On the other hand, the neuron output $s(k)$ is written as

$$s(k) = K \sum_{i=1}^{m} w_i(k) x_i(k) \quad (4)$$

where, $K > 0$ is the neuron proportional coefficient, $x_i(k)(i = 1, 2, ..., m)$ denote the neuron inputs, and $w_i(k)$ are the connection weights of $x_i(k)$ determined by the learning rule. According to Hebb [5], the learning rule of a neuron is formulated by

$$w_i(k+1) = w_i(k) + d_i y_i(k) \quad (5)$$

where $d_i > 0$ is the learning rate, and $y_i(k)$ is the learning strategy. The associative learning strategy given in [5] is as follows:

$$y_i(k) = z(k)s(k)x_i(k) \quad (6)$$

where $z(k)$ is teacher's signal. This implies an adaptive neuron, which uses integrated Hebbian Learning and Supervised Learning. It means that the neuron self-organizes the surrounding information under supervision of the teacher's signal $z(k)$. It also implies a critic on the neuron actions.

Traditional PID controller has an intuitive structure. Unfortunately, it is not very efficient for nonlinear and time-varying systems. In order to improve the efficiency of the PID controller, we use a neuron to adaptively modify its parameters. This way, we overcome the limitations of PID with fixed coefficients. According to the neuron model and its learning strategy, let $x_1(k) = e(k) - e(k-1)$, $x_2(k) = e(k)$, and $x_3(k) = e(k) - 2e(k-1) + e(k-2)$ be the inputs of the neuron, let $w_1(k)$ be the adaptive Proportional Gain, $w_2(k)$

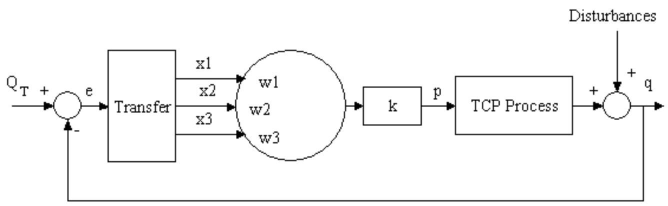

Fig. 1. Block diagram of the closed-loop system with Neuron PID

be the adaptive Integration Gain and $w_3(k)$ be the adaptive Derivative Gain, the Neuron PID AQM scheme is formulated as follows:

$$p(k) = p(k-1) + \frac{K\sum_{i=1}^{3} w_i(k)x_i(k)}{\sum_{i=1}^{3} w_i(k)} \quad (7)$$

where

$$w_i(k+1) = w_i(k) + d_i e(k)p(k)x_i(k) \quad (8)$$

where $e(k)$ is used as the teacher's signal $z(k)$, d_1 is the learning rate of Proportional Gain $K_p(w_1(k))$, d_2 is the learning rate of Integration Gain $K_i(w_2(k))$, d_3 is the learning rate of Derivative Gain $K_d(w_3(k))$.

A simplified block diagram that represents the various components of the overall closed-loop feedback control system including Neuron PID is shown in Figure 1.

Thus, for the Neuron PID AQM scheme, there are six parameters: 1) sampling time interval δt, 2) target queue occupancy level Q_T, 3) the neuron proportional coefficient K, 4) the learning rate of Proportional Gain d_1, 5) The learning rate of Integration Gain d_2, and 6) The learning rate of Derivative Gain d_3. Research has shown [13] that the neuron PID has strong robustness and adaptability which are not very sensitive to the choice of parameters. Based on our simulation experience we propose to use the following parameter values: $\delta t = 0.001s$, $K = 0.01$, $d_1 = 0.000002$, $d_2 = 0.0000001$, and $d_3 = 0.0000001$. We select d_1 to be an order of magnitude greater than the learning rates of other gains (d_2,d_3), because the controller should be more responsive (and therefor track closely) to the current changes in buffer size. The initial values of w_1, w_2 and w_3 do not affect the performance significantly; they can be set as $w_1 = w_2 = w_3 = 0.001$.

III. Simulation Results

Here we present *ns2* [9] simulation results demonstrating superior performance for our Neuron PID over other AQMs. We consider a dumbbell topology based on a single common bottleneck link of 45 Mb/s capacity with many identical, greedy long-lived TCP/Reno flows. This widely used model is applicable to many network topologies whereby a group of sources share a single bottleneck during a period of time that they have data queued for transmission. The receiver's advertised window size is set sufficiently large so that the TCP connections are not constrained at the destination. The ack-every-packet strategy is used at the TCP receivers. Unless specified, the packet size is set at 1000 bytes, the round trip propagation delay of the TCP connections is 100 ms, and the buffer size is 1125 packets, the target queue occupancy is 400 packets. The parameters of Neuron PID are set to the values given in the previous section.

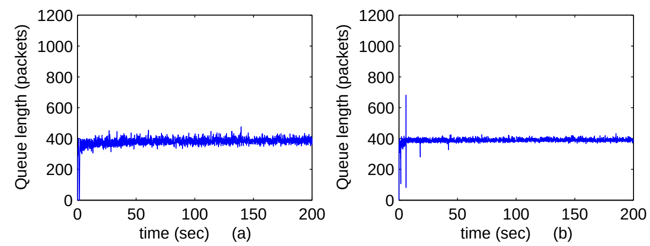

Fig. 2. Queue lengths for Neuron PID with different connections (a) 200 connections (b) 2000 connections

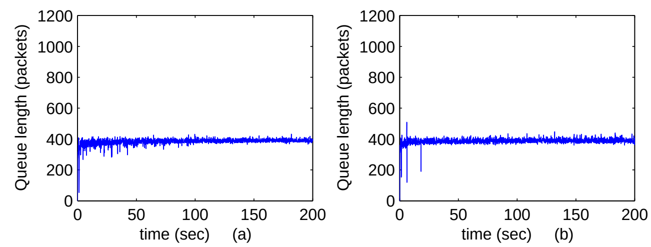

Fig. 3. Queue lengths for Neuron PID with varying number of TCP connections (a) TCP connections varies from 500 to 2000 (b) TCP connections varies from 2000 to 500

First, we test the performance of Neuron PID with constant number of connections. Figure 2 presents the queue length for the cases of 200 and 2000 connections. We can see that Neuron PID is effective at stabilizing and keeping the queue length around the control target Q_T.

Next, we examine the performance of Neuron PID for a traffic scenario involving random start and stop times, thus simulating staggered connection setup and termination. We performed two simulations. In the first, the initial number of connections is set to 500 and, in addition, 1500 connections have their start-time uniformly distributed over a period of 100 seconds. In the second simulation, the initial number of connections is set to 2000 out of which 1500 connections have their stop-time uniformly distributed over a period of 100 seconds. Figure 3 presents the queue length for these two simulations. We can clearly see that Neuron PID is able to stabilize the queue length around the control target.

Recalling that simulations in [10] have shown that AQMs, such as PI, RED and REM, are unstable when RTT is 400 ms, we test here the Neuron PID scheme for a long delay network. Two simulation tests have been performed. In both simulations, there are 1000 TCP connections. In the first, the RTTs are 500 ms, and in the second they are uniformly distributed between 50 and 950 ms. Figure 4 presents the queue lengths for these two simulations. The results demonstrate that Neuron PID is still effective in stabilizing the queue length

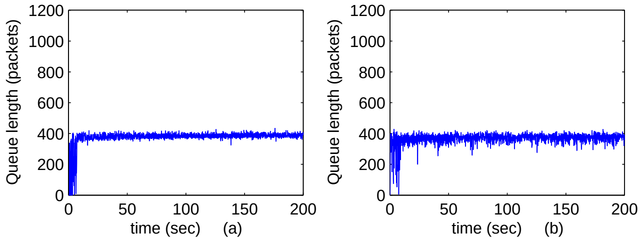

Fig. 4. Queue lengths for Neuron PID with long delay (a) $RTT = 500ms$ (b) RTTs range between 50 and 950 ms

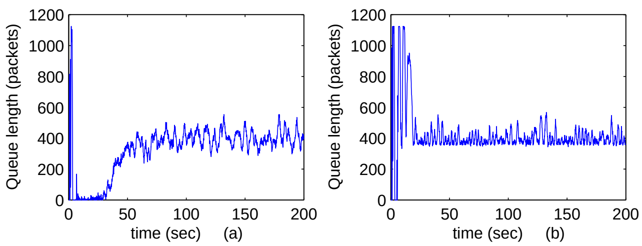

Fig. 5. Queue lengths for (a) IM*PID* (b) LQR*PID*, with RTT=500ms

around the target for TCP connections with large RTTs.

In order to compare with other AQMs under large RTT, Figure 5 presents the simulation results of the AQM algorithm of [10] which we call IM*PID* and the one proposed in [1] which we call LQR*PID*. In the simulations, we consider 1000 TCP connections, and the RTTs are 500 ms. The parameters of IM*PID* are set as: $T = 0.00625$ s, $K_p = 2.268 \times 10^{-5}$, $K_i = 4.193 \times 10^{-6}$, $K_d = 4.1225 \times 10^{-7}$. The parameters of LQR*PID* are set as: $T = 0.00625$ s, $K_p = 3.482 \times 10^{-10}$, $K_i = 1.563 \times 10^{-7}$, $K_d = 8.695 \times 10^{-13}$. As shown, these two AQMs can stabilize their queue length at the given target level, but their response is slow.

We now present simulation results to compare the performance of Neuron PID with Adaptive RED [6], PI controller [8], REM [2], IM*PID* [10] and LQR*PID* [1]. The network topology used in the simulation is the same as that described above. For Adaptive RED, we set the parameters: $min_{th} = 15$, $max_{th} = 785$ and $w_q = 0.002$, and other parameters are set as in [6]: $\alpha = 0.01$, $\beta = 0.9$, $intervaltime = 0.5\ s$. For PI controller and REM, we use the default parameters in *ns2*. For Neuron PID, the parameters are identical to above. For IM*PID* and LQR*PID*, the parameters are set as above.

In the simulation, the initial number of connections is set to 500, and 1500 additional connections have their start-times uniformly distributed over a period of 100 seconds. Figure 6 presents the queue lengths for all six AQMs. We can see that Neuron PID reacts and converges to the target queue occupancy of 400 faster than all other five AQMs.

In order to evaluate the performance in steady-state, we calculate the mean and the standard deviation of the queue length for the last 150 seconds, and the link utilization. These results are presented in Table I. We observe that Neuron PID queue length has the mean of 388.5, which is the closest to the

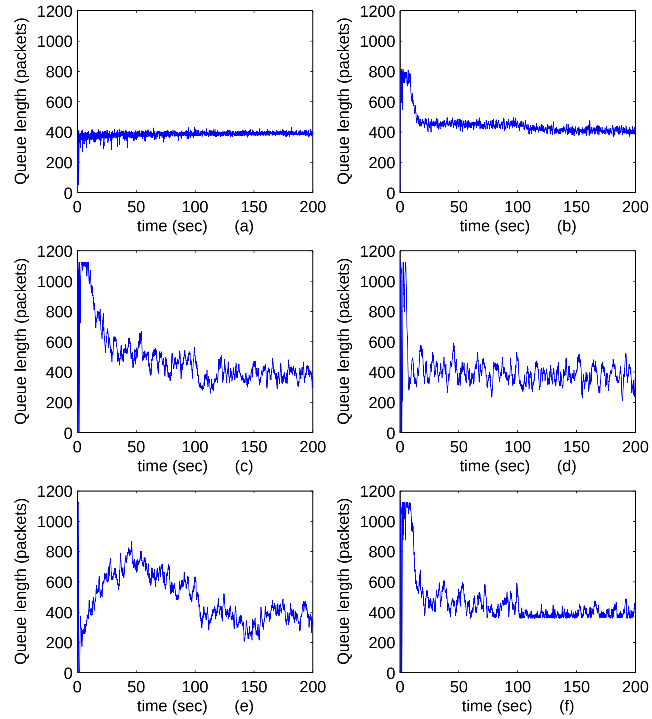

Fig. 6. Simulation results (a) Neuron PID (b) ARED (c) PI (d) REM (e) IM*PID* (f) LQR*PID*

TABLE I

MEAN AND STANDARD DEVIATION (STD) OF QUEUE LENGTHS AND LINK UTILIZATION (U) FOR DIFFERENT AQMS

	Neuron PID	ARED	PI	REM	IM*PID*	LQR*PID*
Mean	388.5	426.7	414.1	380.4	458.3	412.1
STD	15.97	35.92	70.48	59.96	148.06	49.84
U	0.999	0.999	0.997	0.997	0.998	0.980

target of 400, achieves the lowest standard deviation among all AQMs, and has the highest link utilization.

IV. CONCLUSIONS

In this study, a novel AQM scheme called Neuron PID has been proposed. Simulation results show that the Neuron PID can control the queue length at a given target in both short and long delay networks, and that its performance is superior to other well-known AQM algorithms.

ACKNOWLEDGMENT

This work was jointly supported by grants from the Research Grants Council of the Hong Kong SAR, China (Project No. CityU 110705 and CityU 1278/04E), the Natural Science Foundation of Jiangsu Province, China (No. BK2004132), and the Australian Research Council (Grant Number DP0559131).

REFERENCES

[1] D. Agrawal and F. Granelli, “Redesigning an active queue management system,” *Proc. IEEE GLOBECOM 2004*, pp. 702-706, 2004.

[2] S. Athuraliya, V. H. Li, S. H. Low and Q. Yin, "REM: Active queue management," *Proc. 17th Int. Teletraffic Congress*, Salvadorda Bahia, Brazil, Sep. 2001.

[3] J. Aweya, M. Ouellette and D. Y. Montuno, "A control theoretic approach to active queue management," *Computer Networks*, vol. 36, nos. 2-3, pp. 203-235, 2001.

[4] Q. Chen and O. W. W. Yang, "On designing self-tuning controller for AQM routers supporting TCP flows based on pole placement," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 10, pp. 1965-1974, 2004.

[5] Y. Du and N. Wang, "A PID controller with neuron tuning parameters for multi-model plants," *Proceedings of 2004 International Conference on Machine Learning and Cybernetics*, vol. 6, pp. 3408-3411, 2004.

[6] S. Floyd, R. Gummadi and S. Shenker, "Adaptive RED: An algorithm for increasing the robustness of RED's active queue management," available at http://www.icir.org/floyd/red.html

[7] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397-413, Aug. 1993.

[8] C. V. Hollot, V. Misra, D. Towsley and W. Gong, "On designing improved controllers for AQM routers supporting TCP flows," *Proc. IEEE INFOCOM 2001*, vol. 3, Anchorage, Alaska, pp. 1726-1734, April 2001.

[9] The NS simulator and the documentation, available at *http://www.isi.edu/nsnam/ns/*

[10] F. Ren, C. Lin and B. Wei, "A robust active queue management algorithm in large delay networks," *Computer Communications*, vol. 28, no. 5, pp. 485-493, 2005

[11] J. Sun, K. T. Ko, G. Chen, S. Chan and M. Zukerman, "PD-RED: to Improve the Performance of RED," *IEEE Communications Letters*, vol. 7, no. 8, pp. 406-408, August 2003.

[12] J. Sun, G. Chen, K. T. Ko, S. Chan and M. Zukerman, "PD-Controller: A new active queue management scheme,"*proc. IEEE Globecom*, vol. 6, San Francisco, pp. 3103-3107, 2003.

[13] J. Zhang,S. Wang and N. Wang "A new intelligent coordination control system for a unit power plant ," *Proceedings of the 3rd World Congress on Intelligent Control and Automation*, pp. 313-317, 2000.