A Robust Nonlinear PI Controller for Improving AQM Performance

XiaoLin Chang[a], Jogesh K. Muppala[a], Jen-te Yu[b]
[a]Dept. of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
Email: {changxl,muppala}@cs.ust.hk
[b]895 St. Charles Dr. #5, Thousand Oaks, CA 91360, USA

***Abstract*—In this paper a simple robust Proportional-Integral (R-PI) controller is proposed for Active Queue Management (AQM). We assume that TCP/AQM dynamics can be described by the linearized TCP/AQM model [7]. R-PI aims to address the tradeoff between responsiveness and stability and the tradeoff between responsiveness and high link utilization over a large range of structured and unstructured uncertainties. This controller achieves these goals by varying its control parameters according to the system state. We show that the closed-loop system is asymptotically stable as long as the control parameters are time-invariant and varying in a range. Extensive simulation results demonstrate the robust ability of R-PI compared with some other AQM mechanisms in the literature.**

Keywords- *Asymptotically Stable; Robust; Adaptive control; Popov criterion*

I. INTRODUCTION

Active Queue management (AQM) mechanisms are employed in the Internet by the routers to provide good congestion control and provide support for Quality of Service (QoS) [1]. Random Early Detection (RED)[2] was among the first AQM to be introduced into the Internet routers. Since then several enhancements [3] to RED and new AQM mechanisms [4], which are based on heuristic techniques, have been proposed. Recently, some controllers for AQM based on feedback control theory have been developed, such as Integral (I) controller in [5], Proportional-Derivative (PD) controller in [6], PI controller in [7] and PID controller in [9]. These static P/I/D controllers can achieve satisfactory transient response (including small rise time, small settling time and small overshoot), and small steady-state error simultaneously in some situations. However, they are unable to satisfy these performance specifications under varying network conditions. Specifically, they do not address the tradeoff between responsiveness and stability.

The authors in [10] and [11] propose self-tuning schemes to improve the ability of these P/I/D controllers in the varying networks. These schemes use on-line algorithms to estimate the network parameters, which are used as real values to tune the control parameters in the controllers. Such adaptive schemes significantly improve the system performance. However, they may be time-consuming and bring in additional complexity. The authors in [12] propose a switching controller, called P^2I, in order to speed-up the transient response. But it does not address the conflict between responsiveness and high link utilization in the varying networks.

This paper assumes that TCP/AQM dynamics can be described by the linearized TCP/AQM model [7] and a static PI controller for AQM as in [8] has been designed to stabilize this closed-loop system over a range of network scenarios. A simple robust Proportional-Integral (R-PI) controller is introduced, which aims to address (1) the tradeoff between responsiveness and stability, (2) the tradeoff between responsiveness and high link utilization over a wide range of network dynamics. This R-PI controller consists of an automatic gain adjustment and the static PI controller, which are in cascade. R-PI controller improves the system performance by adapting its control response according to the current system states, like the deviation of the instantaneous queue length from its target value. According to *Popov stability criterion* [13], we show that as long as the control parameters in R-PI is time-invariant and sector-bounded, the closed-loop system employing R-PI controller is asymptotically stable. We claim that this R-PI controller is (1) simple, because no on-line algorithm is employed; (2) robust, because extensive simulation results display that, compared other AQM mechanisms in the literature, R-PI exhibits the superior ability of R-PI in alleviating the two tradeoffs over a large range of structured uncertainties, such as parameter variations, and unstructured uncertainties, such as un-modeled dynamics and disturbances,.

The rest of this paper is organized as follows. First, we describe R-PI controller in Section II. In this section, we also give stability analysis of the nonlinear system and some design considerations. Then the experimental evaluation of R-PI, and comparison with some other mechanisms are presented in Section III. Finally we present the conclusions in Section IV.

II. SIMPLE ROBUST PI CONTROLLER

In this section, we first describe the linear closed-loop system [7] with the static PI controller [8]. Afterwards, we present the R-PI controller. Finally the conditions to make the nonlinear system stable and some design considerations are given.

A. The linearized system with static PI controller

We assume that the coupled, linear differential equations in Eq. (1) describes the TCP/AQM dynamics. The details of linearization are given in [7]. Figure 1. shows the block diagram of the linearized AQM control system, including a linear plant and a static PI controller as AQM. $P(s)$ is the transfer function of the linear plant, expressed in Eq. (2). $F(s)= k_p+k_I/s$ is the transfer function of the PI controller.

This work described in this paper has been supported by the Research Grants Council of Hong Kong SAR, China (Project No. DAG02/03.EG20)

$$\begin{cases} \delta\dot{W}(t) = -\delta W(t)\dfrac{2N_0(t)}{C_0(t)R_0^2(t)} - \delta p(t)\dfrac{C_0^2(t)R_0(t)}{2N_0^2(t)} \\ \delta\dot{q}(t) = \delta W(t)\dfrac{N_0(t)}{R_0(t)} - \delta q(t)\dfrac{1}{R_0(t)} \end{cases} \quad (1)$$

where

$C(t)$	bottleneck link capacity (packets/sec)
$N(t)$	number of adaptive connections at time t
$R(t)$	round-trip delay at time t (second)
$p(t)$	dropping/marking probability at time t, $\in [0,1]$
$q(t)$	instantaneous queue length at time t (packets)
q_{ref}	target queue size at time t (packets)
$q$lim	queue buffer size (packets)
$e(t)$	$q(t)$- q_{ref}
k_p, k_I	the proportional gain and integral gain in PI controller, respectively.
T_s	sampling interval
$W(t)$	window size of a TCP connection at time t (packets)
$(W_0(t), q_0(t), p_0(t), C_0(t))$	system operating points in steady state
$\delta w \doteq W-W_0$	$\delta q \doteq q-q_0$ $\delta p \doteq p-p_0$

$$P(s) = \frac{K}{a_1 * s^2 + a_2 * s + a_3} \quad (2)$$

where $K = \dfrac{C^3 R_0^3}{(2N_0)^2}$, $a_1 = \dfrac{R_0^3 C}{2N_0}$, $a_2 = R_0 + \dfrac{R_0^2 C}{2N_0}$, $a_3 = 1$

B. R-PI controller

As mentioned previously, the main disadvantage of employing a static controller is the difficulty in addressing the tradeoff between responsiveness and stability in the varying network. That is, the static controller may exhibit either a fast response with large queue fluctuations or a slow response with small queue fluctuations. Both these two results could result in low link utilization. It is preferable to design a controller with varying gains.

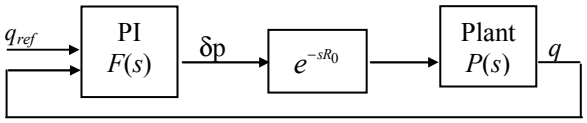


Figure 1. Linear system with static PI controller

Based on the above heuristic arguments, a gain adjustment, $\varphi(t)$, is developed. This adjustment plus the static PI controller constitutes the R-PI controller. Figure 2. describes the system employing this R-PI controller. We define $\varphi(t)$ in Eq. (3). Thus, the R-PI controller is a nonlinear controller in which the gains are not constant. The input to $\varphi(t)$ is the queue dynamics $e(t)=\delta q$ and the output is the "scaled" error $v(t)= \varphi(t)* e(t)$. The scaled error $v(t)$ is the input to the conventional PI controller, $F(s)$, which generates the control action $p(t)$ to drive the system. R-PI controller is a generalization of the static PI controller. This can be seen by setting $\varphi(t)$ to 1.0 for all $e(t)$. Here k_0, k_1, k_2 are positive constants and k_0 must satisfy a condition which is explained in the next subsection.

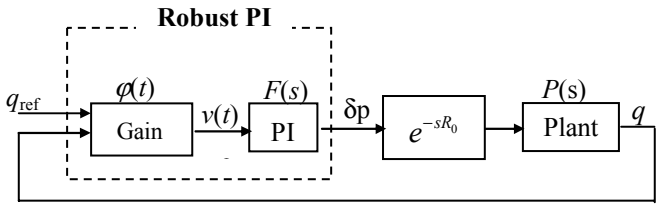


Figure 2. Linear System Employing Robust PI Controller

$$\varphi(t) = \begin{cases} 1 + k_0\left(\dfrac{2}{1+\exp\left(-\left(\dfrac{|e(t)|-k_1}{k_2}\right)^2\right)} - 1.0\right) & k_1 < |e(t)| < \max\{2*q_{ref}, q\text{lim} - q_{ref}\} \\ 1.0 & \text{otherwise} \end{cases} \quad (3)$$

When R-PI is applied, the formula of computing marking probability is shown in Eq. (4).

$$p(t) = k_p\left[\varphi(t)e(t)\right] + k_1\int_0^t\left[\varphi(t)e(t)\right]dt \quad (4)$$

C. Stability analysis

We give the following proposition to specify the condition under which the nonlinear system in Figure 2. is stable. This condition gives the guidelines for selecting k_0.

Proposition: Assume that a static PI controller has been designed to stabilize the system in Fig. 1 and $\varphi(t)$ is defined as given in Eq. (3). Then, there must exist a positive value k_{max} such that as long as $k_0+1<k_{max}$, the system in Figure 2. is be asymptotically stable.

Proof: We define $L(s)= F(s)*P(s)*\exp(-sR)$. We borrow the method of selecting k_p and k_I in [8] to stabilize the system in Figure 1. . Thus, we obtain the real and imaginary parts of $L(s)$ as shown in Eq. (5).

$$\begin{cases} \mathrm{Re}L(j\omega) = \dfrac{Kk_1\left[\sin(\omega R) + \omega R\cos(\omega R)\right]}{-\omega(1+\omega^2R^2)} \\ \omega\,\mathrm{Im}L(j\omega) = \dfrac{Kk_1\left[\cos(\omega R) - \omega R\sin(\omega R)\right]}{-(1+\omega^2R^2)} \end{cases} \quad (5)$$

Eq. (5) shows that $\omega \mathrm{Im}L(j\omega) \to -Kk_1$ when $\omega \to 0$. So $L(s)$ is stable. Assuming $\beta = \mathrm{arctg}(\omega R)$, then we obtain

$$\begin{cases} \mathrm{Re}L(j\omega) = -\dfrac{Kk_1 \sin(\beta + \omega R)}{\omega\sqrt{1+\omega^2 R^2}} \\ \omega \mathrm{Im}L(j\omega) = -\dfrac{Kk_1 \cos(\beta + \omega R)}{\sqrt{1+\omega^2 R^2}} \end{cases} \quad (6)$$

Given (C,R,N), $\mathrm{Re}L(j\omega)$ and $\omega \mathrm{Im}L(j\omega)$ must be bounded when ω is varying from 0 to infinite. This means that in the plane $(\mathrm{Re}L(j\omega), \omega \mathrm{Im}L(j\omega))$ there must exist a line intersecting the negative horizontal axis to make the Popov plot to the right of this line for all $\omega \geq 0$. We define $-1/k_{\max}$ as a value less than the intersection point of the curve and the horizontal axis. This means that, for any stable linear system in Fig. 1, there exists a real number η such that $\mathrm{Re}[(1+j\omega\eta)L(j\omega)]+1/k_{\max} > 0$ for all $\omega \geq 0$. Let $k_0+1<k_{\max}$. Thus, $\varphi(t)$ satisfies the sector condition [13]. According to the *Popov stability criterion*, the system consisting of Eq. (1) and Eq. (2) is asymptotically stable.

As a numerical illustration consider the case of (N_0=60, C_0=15Mbps, R_0=0.246ms) used in [8]. Figure 3. gives the *Popov* plot of $L(j\omega)$ and a line $y=3*(x+1/k_{\max})$, where $x=\mathrm{Re}L(j\omega)$, $y=\omega \mathrm{Im}L(j\omega)$and $k_{\max}$ is 8.0. The intersection point of the curve and the horizontal axis is -0.11394. That means that if the system in Figure 1. has been stable in the vinicity around the equilibrium point (N_0=60, C_0=15Mbps, R_0=0.246ms), the system in Figure 2. with Eq. (3) is asymptotically stable in this vinicity as long as k_0, k_1, k_2 are positive constants and $k_0+1<8.0$. In the following simualtions, we choose k_0=7, k_1=100, k_2=50.

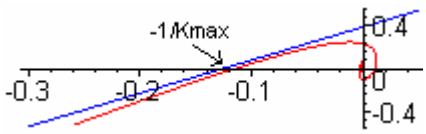


Figure 3. Popov plot of $\mathrm{Re}L(j\omega)$ versus $\omega \mathrm{Im}L(j\omega)$

D. *Design considerations*

This subsection provides several design considerations.

(1) How to accommodate the possible traffic burst and traffic noise. Some AQM mechanisms like [2], [5], and [6] use $\overline{q}(t)$ to adapt $p(t)$ in order to filter the traffic burst and noise. The problems with using $\overline{q}(t)$ have been pointed out by some others like [8]. For example, $\overline{q}(t)$ may not be able to reflect the impact of changes caused by control decisions in a timely manner and may lead to control in the wrong direction. Our simulation results in the next section also show this. Actually, those AQM mechanisms using $q(t)$ to adapt $p(t)$ periodically can accommodate some traffic burst and noise. The process of periodically computing $p(t)$ is a process of averaging $p(t)$. Compared with directly averaging $q(t)$, averaging $p(t)$ can accommodate some burst and noise without compromising system performance.

(2) How to address the tradeoff between responsiveness and stability and the tradeoff between responsiveness and high link utilization. Note that (i) the premise of R-PI is that a static PI controller that stabilizes the system in Figure 1. has been developed; (ii) the network stability is a kind of quasi-stability. Thus, when $|e(t)|$ is small, $\varphi(t)$ is set to 1.0 in order to maintain the performance with the static PI controller. We use k_1 to define the range in which the static PI is active. When $|e(t)|$ is large, $\varphi(t)$ amplifies $|v(t)|$ substantially to generate a large corrective action to drive $q(t)$ to q_{ref} quickly.

However, we observe that it is possible that noise or burst may cause $|e(t)|$ above k_1 temporarily. In these situations $\varphi(t)$ should produce small gain. We use k_1 and k_2 to define the range of small increase. It is known that there is a time delay before the end hosts know the occurrence of incipient congestion. That means that some more undesirable packets are sent out by the sources after incipient congestion has occurred. This may cause the unnecessary increase in $p(t)$. This problem exists in PI controller [8]. In order to prevent from worsening in R-PI, when $|e(t)|$ is above a level $\varphi(t)$ is set back to 1.0. In the following simulations, we set this level as $\max\{2*q_{\mathrm{ref}}, q_{\mathrm{lim}} - q_{\mathrm{ref}}\}$. In addition, we use $|e(t)|$ rather than $e(t)$ in Eq. (3) in order to compensate the overshoot in marking probability to some degree, especially when the traffic load is suddenly decreased.

III. SIMULATION RESULTS

In this section, we carry out simulations with *ns-2* [14] simulator to evaluate R-PI controller. We compare it with ARED [3], PD [6], PI [8], and P^2I [12], among which ARED and PD use $\overline{q}(t)$ to adapt $p(t)$, PI, R-PI and P^2I use $q(t)$. By comparing with PI and PD, we aim to examine the improvement with R-PI in fast transient response while maintaining small steady-state error. By comparing with P^2I, we aim to examine the ability of R-PI in addressing the tradeoff between fast response and high link utilization. In order to investigate the advantage of directly controlling $q(t)$ over controlling $\overline{q}(t)$, we do simulations with ARED and a variant of PD which uses $q(t)$ to adapt $p(t)$. In the following, PD_a represents PD controller using $\overline{q}(t)$ as control input; PD_c represents PD controller using $q(t)$as control input.

TABLE I. lists some notations and the default parameter settings in the corresponding mechanisms. We set these control parameters according to the authors' recommendation. Considering the difficulty in setting parameters, we do simulations using the network scenarios in [6] and [8], respectively. Figure 4. is the network topology. Unless otherwise specified, in all the simulations, the adaptive sources use TCP/Reno; the initial window size of each TCP connection is set to 100 packets; all the adaptive connections and the routers are ECN-enabled. Performance is evaluated using the instantaneous queue length $q(t)$.

A. *Constant TCP flows + burst HTTP+ burst UDP*

This simulation aims to investigate the ability of each scheme in the network with constant large number of long-lived TCP flows and with burst HTTP+UDP traffic as noise. We use one of the network scenarios in [6]. The bottleneck link capacity is 45Mbps, the packet size is 1000 bytes, q_{ref}

=400packets, $q$lim=1125packets. So in ARED, w_g=0.00017776, min_{th}=200packets, max_{th}=600packets. In P^{2}I, q_{thresh}=650, q'_{ref} =250packets. There are 1000 TCP sources (connections), 400 HTTP sessions (connections) and 500 UDP sources. Each TCP source generates an infinite FTP bulk data transfer with different RTTs uniformly distributed in (50, 150) ms. The packet size is 1000bytes. The RTTs of HTTP connections are uniformly distributed in (50, 300) ms. Each session has 250 pages and each page has one object. The page interval in a session follows an Exponential Distribution with mean of 0.4s. The RTTs of the UDP flows are uniformly distributed in (30, 150) ms. Each UDP source follows an exponential ON/OFF traffic model, both the idle and the burst times have mean of 0.5 s. The packet size is 210 bytes, and the sending rate during on-time is 64 kbps. At the network nodes, the mean packet size is 500 bytes. The simulation lasts 100s. All the UDP and long-lived TCP connections start randomly in the first second. The HTTP connections start in sequence. The first starts at the 0th second. The next starts after 0.1s, and so on. Other settings are as default. Figure 5. is $q(t)$ variation. The dark curves represent $q(t)$ variation and the light curves represent $\overline{q}(t)$ variation.

TABLE I.

Controller	Parameters
PI [8]	a=1.816*10^{-6}, b=1.822*10^{-6}, T_s =1/160 s
R-PI	a=1.816*10^{-6}, b=1.822*10^{-6}, k_0=7.0, k_1=100.0, k_2 =50, T_s =1/160 s
P^{2}I [12]	a=1.816*10^{-6}, b=1.822*10^{-6}, T_s=1/160 s, α_p=β_p =2.0
PD_a PD_c [6]	L= q_{ref} *0.8, k_p = 0.0002 k_d = 0.08, filter_gain = 0.0025, T_s =1/100 s
ARED [3]	min_{th}= q_{ref} *0.5; max_{th}= q_{ref} *1.5, w_g=1-exp(-1/C), all other parameters are set to the default values in NS$_2$

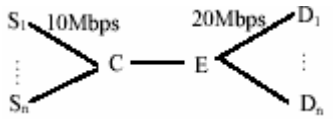


Figure 4. Network topology

B. *Large variation in traffic load*

This experiment aims to examine the performance of each scheme when the number of long-lived TCP flows is varying in (200, 1000) in the network, where there exist disturbances caused by 200 HTTP and 200 UDP burst connections. Network settings and control parameter settings in each scheme are same as in Simulation A. Each TCP source generates an infinite FTP bulk data transfer with different RTTs uniformly distributed in (160, 240) ms. The packet size is 1000bytes. The RTTs of HTTP connections are uniformly distributed in (50, 300) ms. Each HTTP session has 400 pages and the page interval in a session follows an Exponential Distribution with mean of 1.0s. The RTTs of the UDP flows are uniformly distributed in (30, 150) ms. Each UDP source follows an exponential ON/OFF traffic model, both the idle and the burst times have mean of 0.5 s. The packet size is set to 210 bytes, and the sending rate during on-time is 50kbps. The mean packet size at the router is 500 bytes. The simulation lasts 450s. All the UDP and long-lived TCP connections start randomly in the first second. The HTTP connections start in sequence. The first starts at the 0th second. The next starts after 0.0005s, and so on. For FTP flows, 200 flows are started at 0th second, and increased in steps of 200 flows at each 50 second interval until t=250th second. From t=200th second, the number of flows is reduced by 200 at each 50 second interval until t=400th second. Figure 6. is queue length variation. The dark curves represent $q(t)$ variation and the light curves represent $\overline{q}(t)$ variation.

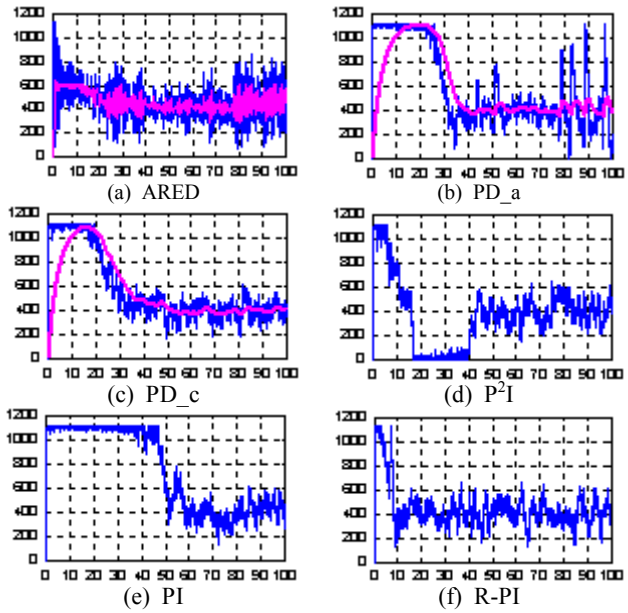
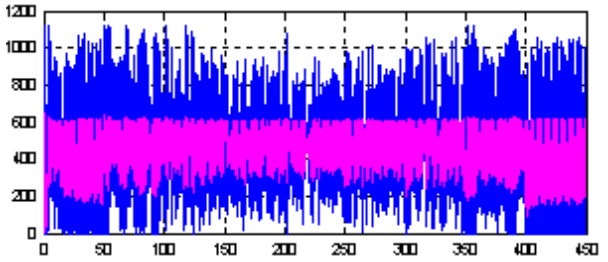


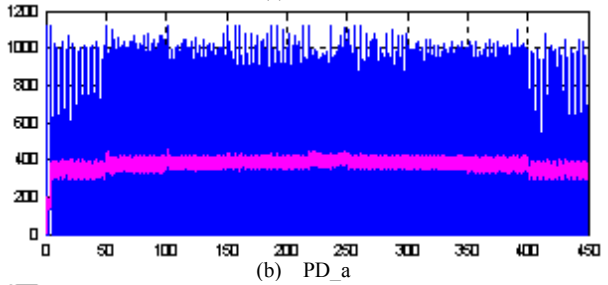
Figure 5. Simulation 3: Queue Length Variation

C. *Summary and discussion*

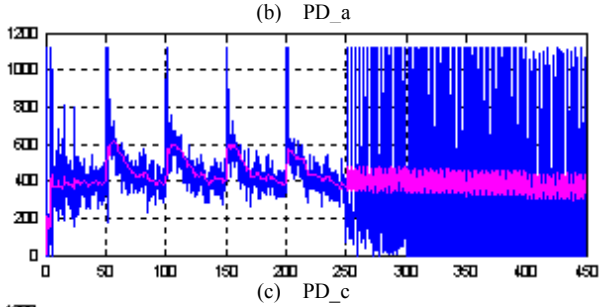
The above simulation results show that (1) ARED and PD have a fast transient response than PI. However both of them do not address the conflict between responsiveness and stability in all the above network scenarios. (2) When PD controller is applied, using $\overline{q}(t)$ does not produce better performance than using $q(t)$. (3) PI controller can lead to a stable system in all the network scenarios but it has a slow transient response. (4) P^{2}I controller has a fast transient response. It can reach the steady state but it does not solve the conflict between responsiveness and high link utilization. In the above two groups of experiments, the performance with P^{2}I is worse than that of PI, especially in terms of link utilization. Due to space limit, we donot show the results of link utilization with each scheme. (5) In all the simulations, R-PI controller performs the best among the five mechanisms. It addresses the conflicts existing in PD and P^{2}I controllers and makes great improvement of the ability of PI over a wide range of network dynamics.



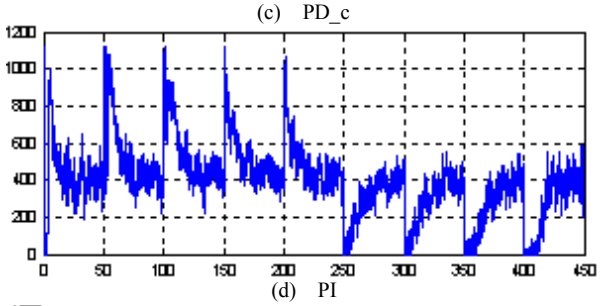
(a) ARED



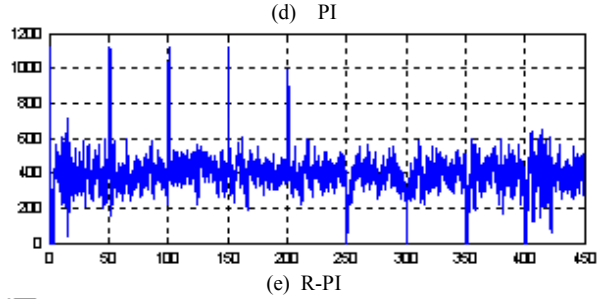
(b) PD_a



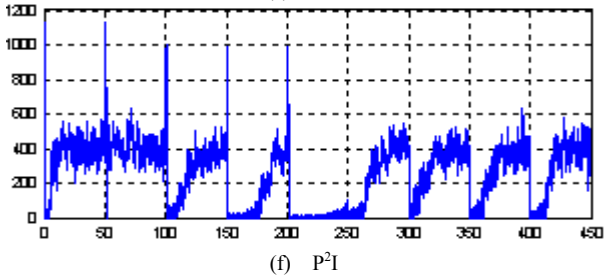
(c) PD_c



(d) PI



(e) R-PI



(f) P^2I

Figure 6. Simulation C: Queue Length Variation

IV. CONCLUSION

In this paper we present a simple robust Proportional-Integral (R-PI) controller. The varying gain enables the controller to adapt its response according to the system state. By using *Popov Stability Criterion*, we analyze the asymptotic stability of the closed-loop system. Some features demonstrate its simplicity. Extensive simulation results display its robust ability in addressing (1) the tradeoff between responsiveness and stability; (2) the tradeoff between responsiveness and high link utilization over a wide range of network uncertainties.

REFERENCES

[1] V. Misra, W.B. Gong, and D. Towsley, “Fluid-based Analysis of a Network of AQM Routers Supporting TCP flows with an Application to RED,” In ACM. Computer Communication Review, vol. 30, no.4, pp.151-160, October 2000.

[2] S. Floyd, and V. Jacobson, “Random Early Detection Gateways for Congestion Avoidance,” In IEEE/ACM Transactions on Networking, vol. 1, no. 4, pp. 397-413, October 1993.

[3] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: an Algorithm for Increasing the Robustness of RED," available at http://www.icir.org/floyd/papers/adaptiveRed.pdf, August 2001.

[4] W. Feng, K.G. Shin, D.D. Kandlur, and D. Saha, “The BLUE Active Queue Management Algorithms,” In IEEE/ACM Transactions on Networking, vol. 10, no. 4, pp. 513-528, August 2002.

[5] J. Aweya, M. Ouellette, and D.Y. Montuno, “DRED: a Random Early Detection Algorithm for TCP/IP Networks,” In International Journal of Communication Systems, vol.15, no.4, pp.287-307, May 2002.

[6] J. S. Sun, G.C.K.T. Ko, S.Chan and M. Zukerman, “PD-controller: A New Active Queue Management Scheme,” In Proc. IEEE Global Telecommunications Conference (GLOBECOM 2003), December 2003.

[7] C.V. Hollot, V. Misra, D. Towsley, and W. B. Gong, “A Control Theoretic Analysis of RED,” In Proc. IEEE Conference on Computer Communications (INFOCOM 2001), vol. 3, pp. 1510-1519, April 2001.

[8] C.V. Hollot, V. Misra, D. Towsley, and W. B. Gong, “On Designing Improved Controllers for AQM Routers Supporting TCP Flows,” In Proc. IEEE Conference on Computer Communications (INFOCOM 2001), vol. 3, pp. 1726-1734, April 2001.

[9] X. Deng, S. Yi, G. Kesidis, and C. Das, "A Control Theoretic Approach for Designing Adaptive AQM Schemes," To appear in Proc. IEEE Global Telecommunications Conference (GLOBECOM 2003), December 2003.

[10] W. Wu, Y. Ren, and X. Shan, "A Self-Configuring PI Controller for Active Queue Management", In Proc. the 7th Asia-Pacific Communication Conference (APCC'2001), September 2001.

[11] H.G. Zhang, D. Towsley, C. V. Hollot, and V. Misra, “A Self-tuning Structure for Adaptation in TCP/AQM Networks,” In Proc. ACM/SIGMETRICS 2003, June 2003.

[12] M. Zhang, J. Wu, C. Lin, and K. Xu, “Rethink the Tradeoff between Proportional Controller and PI Controller,” In Proc. International Symposium on Computers and Communications (ISCC 2002), pp.57-62, July 2002.

[13] S.M. Shinners, “Advanced Modern Control System Theory and Design,” New York : Wiley, c1998.

[14] UCB/LBNL/VINT Network Simulator – NS (version 2), http://www-mash.cs.berkeley.edu/ns