ELSEVIER

Computer Communications 25 (2002) 874–883

computer communications

www.elsevier.com/locate/comcom

Design of a fuzzy controller for active queue management

Ren Fengyuan*, Ren Yong, Shan Xiuming

Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

Received 19 April 2001; revised 28 September 2001; accepted 3 October 2001

Abstract

As an effective mechanism acting on the intermediate nodes to support end-to-end congestion control, active queue management (AQM) takes a trade-off between link utilization and delay experienced by data packets. From the point of view of the control theory, it is rational to regard AQM as a typical regulating system. Although proportional integral (PI) controller for AQM outperforms traditional RED algorithm, the mismatches in simplified TCP flow model inevitably degrades the performance because the design of PI controller is heavily dependent of the accuracy of the plant, such as, for small buffer the system tends to perform poorly. In this paper, the fuzzy logic controller (FLC) for AQM is designed based on the fuzzy logical control. Its superiority is independent of the model of the plant, which is very suitable to the high variability and uncertainty networks. We present the guidelines and highlights to design the parameters of the FLC. We then compare its performance with the PI controller through simulations, and investigate the impact of the network configuration and operating parameters on the stability and responsibility. The results show that the FLC is rather robust against the noise and disturbance caused by the round-trip time, the link capacity, the number of the active flows and the non-responsive UDP flows, etc., which badly degrade the performance of the PI controller. The transient response and tracking capability of the FLC is superior to that of the PI controller, which is beneficial to achieve the goals of AQM scheme. © 2002 Published by Elsevier Science B.V.

Keywords: Active Queue Management (AQM); Congestion Control; Flow Control; Fuzzy Logic Control

1. Introduction

TCP congestion control mechanism, while necessary and powerful, are not sufficient to provide good service in all circumstances, especially with the rapid growth in size and the strong requirement to QoS support, because there is a limit to how much control can be accomplished at endpoint. It is needed to implement some measures in the routers to complement the endpoint congestion avoidance mechanisms. Active queue management (AQM), as one class of packet dropping/marking mechanism in the router queue, has been recently proposed to support the end-to-end congestion control in the Internet [1]. The principle of AQM is very simple: implicit/explicit feedback of the congestion to the hosts by dropping/marking packets at router queues. The end hosts then react to the packet dropping/marking by reducing the amount of data sent. The goals of AQM are to (1) reduce the average length of queue in routers and thereby decrease the end-to-end delay experienced by packets, and (2) ensure the network resources to be used efficiently by reducing the packet loss that occurs when queues overflow. Prior to AQM, 'drop front on full' and 'random drop on full' [5] were used pervasively in the routers, they can solve the 'lock-out', the condition in which a small sub-set of the flow sharing the link can monopolize the queue during periods of congestion, caused by drop tail scheme, but neither solves the full-queue problem. AQM further provides the solution to the full-queue problem by dropping packets before buffers overflow, so it is called proactive queue management. AQM highlights the tradeoff between delay and throughput. By keeping the average queue size small, AQM will have the ability to provide greater capacity to accommodate nature-occurring burst without dropping packets, at the same time, reduce the delays seen by flow. This is very particularly important for interactive real-time applications.

Since AQM is proposed, a series of implementation algorithms are deployed. RED was only a candidate for AQM in RFC2309 [1]. Most of the other algorithms are either improvement or variation of RED, such as, RED with 'gentle_', Balanced-RED [2], SRED [3] and FRED [4] belong to the former, but self-configuring gateway [6], and so on are examples of the later. At the same time, many studies verified that RED algorithm is unstable and too sensitive to parameters configuration [7–9]. In Ref. [10],

* Corresponding author.
E-mail address: renfy@spd.ee.tsinghua.edu.cn (R. Fengyuan).

0140-3664/02/$ - see front matter © 2002 Published by Elsevier Science B.V.
PII: S0140-3664(01)00417-0

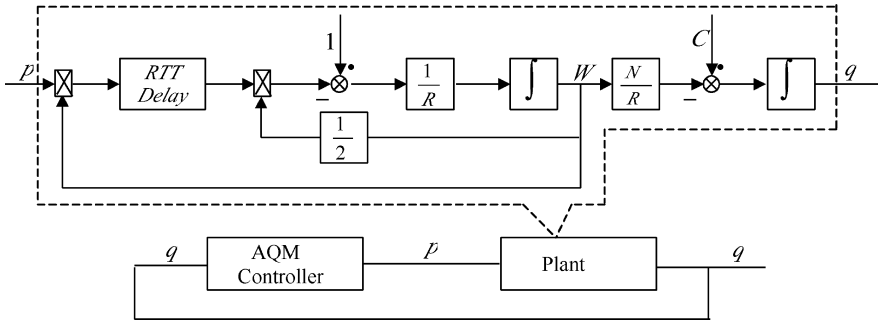

Fig. 1. The block diagram of TCP control.

Hollot et al. thoroughly analyzed the effect on stability caused by network parameters based control theory, and then designed a proportional integral (PI) controller for AQM [11]. Although this new AQM algorithm appear to improvement on performance and more robustness, we believed that it is problematic and unrealistic, at least inaccurate, to take the network as the linear and constant system because the actual network is very changeful. Thus, the algorithm based on this assumption should have the limited validity. In this study, we intend to design an AQM algorithm based on fuzzy control theory. Because the fuzzy controller is independent of the mathematical model of the controlled plant, it is very valuable to complex and time-varying network system. The remainder of this paper is organized as follow. Section 2 provides an introduction to the background on control theory applying to the router queue management, and points out the limitation of the control method based on constant model in rapidly changeable network environment. Section 3 presents the principle of fuzzy control in brief, develops the fuzzy logic controller (FLC), and explains design guidelines. In Section 4, we will testify the validity of FLC for AQM, and then compare its performance with PI controller through numerical simulation results. Finally, the conclusion is drawn in Section 5.

2. Background

In Ref. [12], the dynamic model of a TCP connection through a congested AQM router was developed using fluid flow analysis, the following is a simplified version of that model.

$$\begin{cases} \dfrac{\mathrm{d}W(t)}{\mathrm{d}t} = \dfrac{1}{R(t)} - \dfrac{W(t)W(t-R(t))}{2R(t)}p(t-R(t)) \\ \dfrac{\mathrm{d}q(t)}{\mathrm{d}t} = \dfrac{N(t)}{R(t)}W(t) - C(t) \end{cases} \tag{1}$$

Fig. 1 illustrates these coupled differential equations through the block diagram, which shows that the problem of AQM algorithm can be converted into the design of the controller. Regarding the components embraced in dash line as the plant, it will be a classical feedback control system. This understanding is very important itself, because the algorithm designed based on control theory is certainly superior to the initial intuitive RED algorithm on the performance, such as robustness, stability and responsibility etc. Both Refs. [11,12] approximate this non-linear and time-varying dynamic as a linear constant system by small-signal linearization about an operating point. It is useful and helpful to analyze and explain the instability of RED under some network parameter configuration. Nevertheless, we believe that the AQM controller designed using the simplified and inaccurate linear constant model should not be optimal, because the real network is rapidly changeful, the network parameters, including round-trip time $R(t)$, the number of active connections $N(t)$, and the capacity of the link $C(t)$ are hardly kept at the constant values for a long time. Moreover, Eq. (1) only takes consideration into the fast retransmission and fast recovery, but ignore the timeout mechanism caused by lacking of enough duplicated ACK, this is very usual in

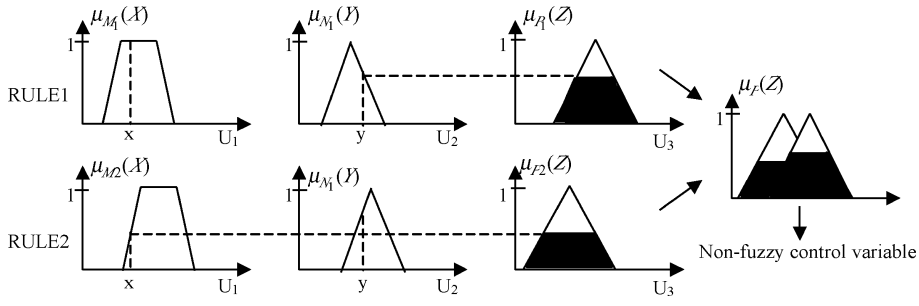

Fig. 2. Fuzzy inference procedure.

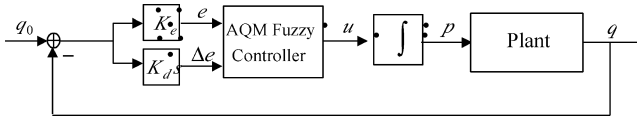

Fig. 3. FLC for AQM.

burst and short services, such as TELNET and HTTP. In addition to, there are many non-responsive UDP service flows besides TCP connections in the network. They are also not included in Eq. (1).

The effective AQM controller should be adaptable. Both P and PI controller [11] hardly work under the general network environment because of their dependence on the linear model, which will be further confirmed in later simulations. Thus, the perfect scheme is to explore an approach to design the AQM controller independent of the model of the controlled plant. The FLC should be very applicable.

3. Fuzzy queue controller

3.1. Fuzzy control theory

Traditional control engineering uses mathematical models of a system and its inputs to design control actions and/or to analyze their effectiveness. Fuzzy control, introduced by Mamdani for the control complex process [13], denotes the field within control engineering in which fuzzy sets and fuzzy inference are used to derive control laws. It is especially useful for situations in which no precise model of the processes exits and most of the a priori information is available only in qualitative form. The basic idea of fuzzy control is to make use of expert knowledge and experience to build a rule base with linguistic if-then rules. Proper control actions are then derived from the rule base. The basic principle used in fuzzy control is the notion of a fuzzy sets and fuzzy inference. A fuzzy rule is a conditional statement, expressed in the form if-then. The deduction of the rule is called the inference and requires the definition of a membership function characterizing this inference. This function allows us to determine the degree of the truth of each proposition. The fuzzy controller operation is typically divided into the following three phases:

Table 1
Control rules base

Q	RQ						
	NB	NM	NS	Z	PS	PM	PB
NB	NH	NH	NH	NB	NB	NM	NM
NM	NH	NB	NB	NM	NM	NS	NS
NS	NB	NM	NS	Z	Z	Z	PS
Z	NM	NS	Z	Z	Z	PS	PM
PS	NS	Z	Z	Z	PS	PM	PB
PM	PS	PS	PM	PM	PB	PB	PB
PB	PM	PM	PB	PB	PB	PH	PH

(a) *Fuzzifization* is a procedure to define fuzzy sets based on system input measurements. That is fuzzification defines the membership mappings from the measured values of each input to a set of linguistic values for that input, such as:

If X is M_1 and Y is N_1 then Z is P_1 (Rule 1)

If X is M_2 and Y is N_2 then Z is P_2 (Rule 2)

where X and Y are variables of the condition part, and Z is the variable of the action part. M_i, N_i and P_i are fuzzy parameters characterized by membership functions. The condition parts of control rules make use of measurements that are usually real numbers. According the their respective value domain, e.g. U_1 and U_2, these real-valued measurements, e.g. x and y, are matched to their membership values defined as (Fig. 2):

$\mu_{M_1}(x),\ \mu_{N_1}(y),\ \mu_{M_2}(x),\ \mu_{N_1}(y)$

(b) *Inference* is an interface that produces a new fuzzy set from the result of the fuzzification (the linguistic values)

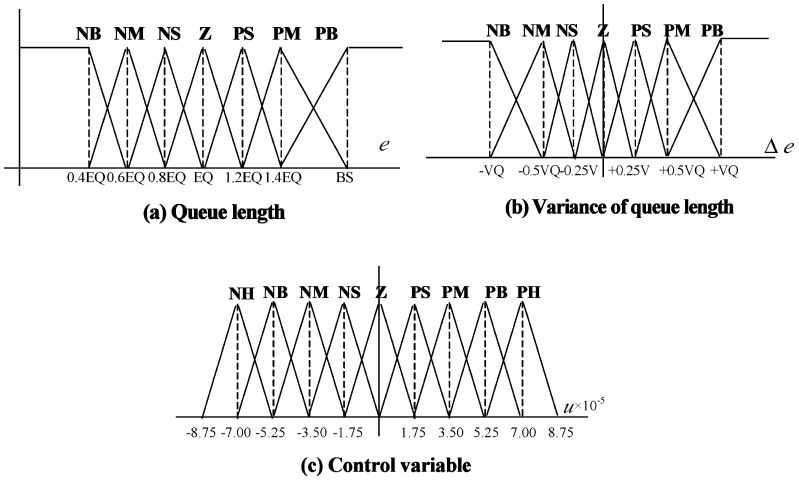

Fig. 4. The membership function for fuzzy sets: (a) queue length, (b) variance of queue length, and (c) control variable.

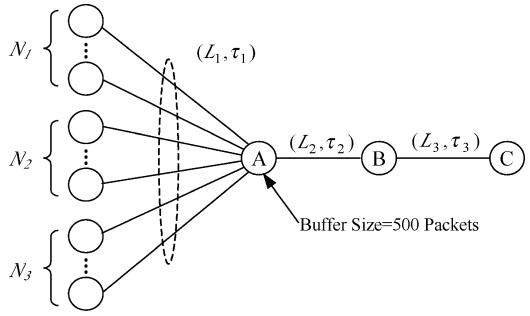

Fig. 5. The simulation network topology.

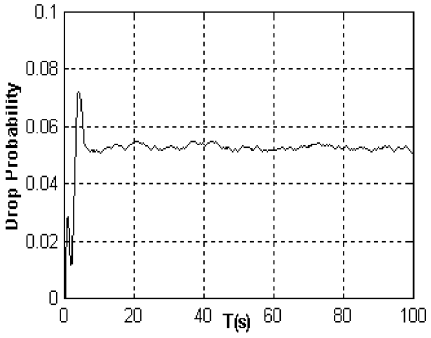

Fig. 7. (Experiment 1)

using a set of rules. The result of the inference is a fuzzy set that can be called the fuzzy control action. Suppose that $X = x$ and $Y = y$, the inference is derived as follows.

For all the control rules in rule base, we derive the truth-value of each rule in the premise by building the conjunctions of the matching membership values:

$$\mu_{P_1} = \mu_{M_1}(x) \wedge \mu_{N_1}(y) \qquad \text{(Rule 1)}$$

$$\mu_{P_2} = \mu_{M_2}(x) \wedge \mu_{N_2}(y) \qquad \text{(Rule 2)}$$

Generally, where the operator $\wedge$ can be the 'min' function [13]. In Fig. 2, U_3 represents the value domain of the output variable Z. The fuzzy control output $P(Z)$ is represented by the aggregation of all fuzzy subsets $P_i(Z)$, whose membership values $\mu_P(Z)$ are determined by the disjunction of all the membership values $\mu_{P_i}(Z)$ as following:

$$\mu_p = \mu_{p_1}(Z) \vee \mu_{p_2}(Z)$$

Here, the disjunction operator $\vee$ is the 'max' function when used with Mamdani's implication.

(c) *Defuzzzifization* is the procedure that produces a real value from the result of the inference, which could be used as control input. Several methods can be applied to defuzzification, the most widely used is the *center of gravity* method [13]. Fig. 2 summarizes the general architecture of the fuzzy controller.

3.2. Fuzzy logic controller for active queue management

Fuzzy logic systems have been widely applied to control non-linear, time-varying and ill-defined systems in which they can provide simple and effective solutions. The system model of AQM FLC is shown in Fig. 3, which is a standard regulating system. The parameter q_0 is the expected queue (EQ) length. The instantaneous queue length is obtained by sampling. Generally speaking, the sampling frequency is higher, the control is more accurate, but sampling at every packet arrival, just like in RED implementation, is overkill and provides no perceptible benefit. For the 12 500 packets/s (100 Mbps, the default size of packet is 1000 bytes) bottleneck link, the maximum queue variance could be 125 packets when the sampling frequency is 100 Hz. In order to observe this maximum variance, only requirement is to have enough buffer space. In a general router, it should be very usual that the buffer accommodates hundreds of packets. In FLC implementation, we fix the sampling frequency at 160 Hz. Of course, another consideration is convenient to compare with PI controller. The error of queue length e ($e = q_0 - q$) and the error varying rate Δe are used as the input linguistic variable of the FLC, the latter can effectively predicate and describe the local dynamic of the difference between the arrival rate and the service rate. The chosen output is linguistic variable u that represents the increment of the packet dropping or marking probability p.

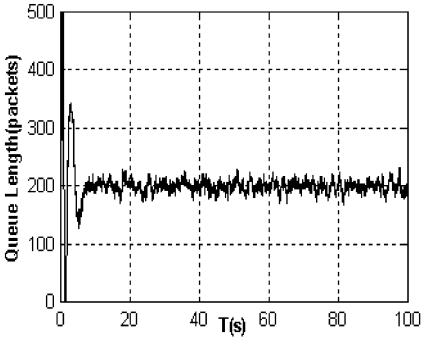

Fig. 6. EQ = 200 (Experiment 1).

The model of fuzzy system, comprising the control rules and the term sets of the variables with their related fuzzy sets, was obtained through a tuning process that started from a set of the initial insight considerations and progressively modified the parameters of the system until it reached a level of performance considered to be adequate. In particular, both of the input variables have seven fuzzy term sets, which are negative big (NB), negative medium (NM), negative small (NS), zero (Z), positive small (PS), positive medium (PM) and positive big (PS). The output variable is greater than two term sets in number, they are negative huge (NH) and positive huge (PH). The membership functions chosen for the fuzzy sets are shown in Fig. 4. EQ stands for the expected value of queue length, i.e. q_0. BS is the router buffer size. VQ represents the estimated value of the maximum range that the queue length variation can reach during one sample interval. Through many simulation experiments, it is finally fixed at 0.5BS. When

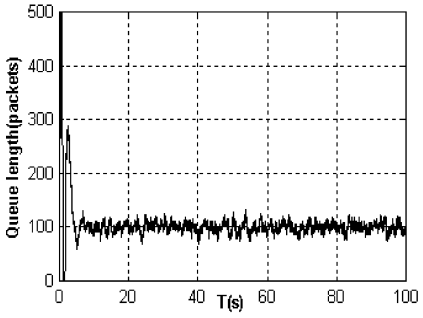

Fig. 8. EQ = 100 (Experiment 1).

BS = 500 packets, VQ = 250 packets, which is sufficient for observing queue variance according to earlier discussion.

During the designing of AQM FLC, the two conflicting requirements must be taken into consideration at the same time. The first requires the controller to have good transient response, such as, the regulating time is rather short and the overshot is very small. The second emphasizes the steady performance, such as small steady error and accurately tracking capability. For the perfect trade-off between two requirements, we eventually chose the sets depicted in Fig. 4(c) as the membership function of the control variable u. The section $[-8.75\times10^{-5}, 8.75\times10^{-5}]$ is evenly divided into 11 intervals. All triangles used to define the membership functions of nine fuzzy term sets have the same shape, whose bottom equals to 3.5×10^{5}. As well as, the control rules are also very crucial. Moreover, they are coupled with the parameters of the membership functions, so the parallel and cooperative tuning is needed in order to get better AQM FLC.

Table 1 describes all the 49 possible control rules. Each one can be presented as the following:

$$R_k:\ \text{if } Q_i \text{ and } \mathrm{QR}_j \text{ then } U_{ij} \qquad i=1,2,\ldots,7\ \ j=1,2,\ldots,7$$

So we have:

$$R=\bigcup_{k}^{N=49} Q_i\otimes \mathrm{QR}_j\otimes U_{ij} \tag{2}$$

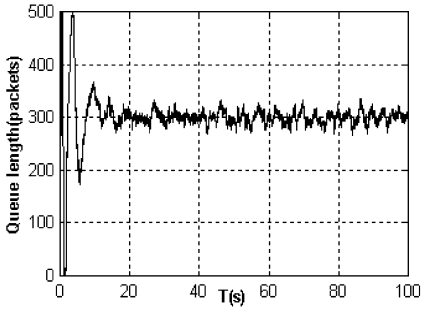

Fig. 9. EQ = 300 (Experiment 1).

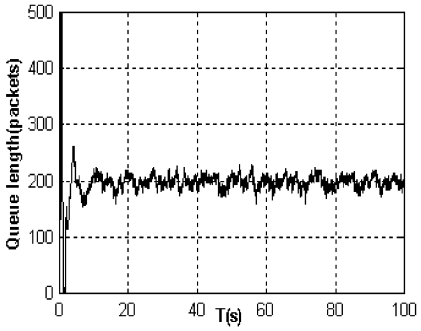

Fig. 10. FLC (Experiment 2).

The membership function of the fuzzy relationship R is:

$$\mu_R(q,\mathrm{qr},u)=\bigvee_{i=1;j=1}^{i=7;j=7}\mu_{Q_i}(q)\wedge\mu_{\mathrm{QR}_j}(\mathrm{qr})\wedge\mu_{U_{ij}}(u) \tag{3}$$

The fuzzy output variable is got using 'max–min' reasoning:

$$U=(Q\otimes \mathrm{QR})\oplus R \tag{4}$$

The membership degree of the control variable U is:

$$\mu_U(u)=\bigvee_{\substack{q\in Q\\ \mathrm{qr}\in \mathrm{QR}}}\mu_R(q,\mathrm{qr},u)\wedge[\mu_Q(q)\wedge\mu_{\mathrm{QR}}(\mathrm{qr})] \tag{5}$$

In order to obtain the final output control value, the *center of gravity method* [15] is used:

$$u=\frac{\int u\cdot\mu(u)\mathrm{d}u}{\int\mu(u)\mathrm{d}u} \tag{6}$$

4. Performance evaluation via simulation

This section validates the effectiveness and performance of the FLC by simulation. We still use ns2 [14] as simulator. The network topology is shown in Fig. 5. The only bottleneck link lies between node A and node B. The buffer size of the node A is 500 packets, whose default size is 1000 bytes. The PI controller (or FLC) controls the queue in node A, and the others are drop tail. All sources are classed into three groups. The first one includes N_1 greedy sustained FTP

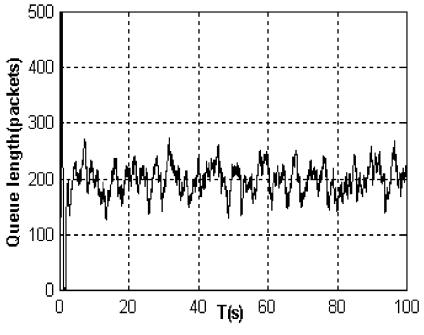

Fig. 11. PI (Experiment 2).

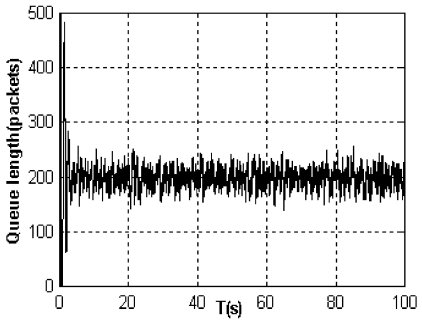

Fig. 12. FLC (Experiment 3).

application sources, the second one is composed of N_2 burst HTTP connections, and the third one has N_3 UDP sources. We introduce short-lived HTTP flows and non-responsive UDP services into the router in order to generate a more realistic traffic scenario, because it is very important for a perfect AQM scheme to achieve full bandwidth utilization in the presence the noise and disturbance introduced by these flows. The links between node A and all sources have the same capacity and propagation delay pair (L_2, τ_1). The pair (L_2, τ_2) and (L_2, τ_3) define the parameters of link AB and link BC.

4.1. Experiment 1

In this experiment, we will use the most general network configuration to testify whether the FLC can reach the goals of AQM, and freely control the queue length to stabilize at the arbitrary expected value. Therefore, given that $(L_1, \tau_1) = (L_2, \tau_2) = (15\ \text{Mbps}, 5\ \text{ms})$, $(L_3, \tau_3) = (30\ \text{Mbps}, 5\ \text{ms})$. $N_1 = 60$, $N_2 = N_3 = 0$. Let the EQ equal to 200 packets, the instantaneous queue length and the drop probability are plotted in Figs. 6 and 7, respectively. After a transient regulating process, both of them settle down the stable operating point. In order to adequately show this ability of FLC, setting EQ = 100 and 300, the results are depicted in Figs. 8 and 9. This attribute is necessary to achieve the AQM objectives. The stable queue length is beneficial to control the queuing delay to satisfy with the special QoS requirements.

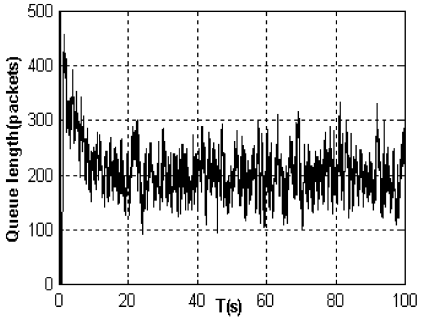

Fig. 13. PI (Experiment 3).

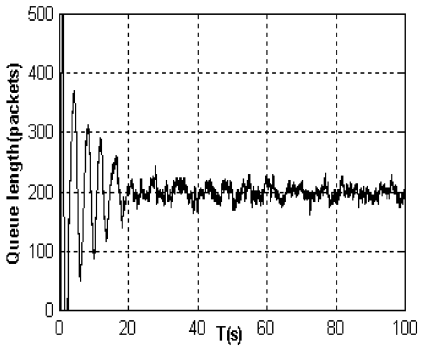

Fig. 14. FLC (Experiment 4).

4.2. Experiment 2

In order to analyze the stability of RED and its variation algorithms, C. Hollot approximated the inherently nonlinear dynamics presented in the coupled differential Eq. (1) by their small-signal linearization about an operating point [10]. The conclusion drawn from the linear model is that three parameters mainly affect the system stability. They are link capacity C, the round-trip time R and network load N. Because, from the plant transfer function [11]

$$G(s) = \frac{\dfrac{C^2}{2N}}{\left(s + \dfrac{2N}{R^2C}\right)\left(s + \dfrac{1}{R}\right)} \tag{7}$$

We can easily obtain the DC[1] gain of the plant

$$\text{DC plant gain} = \frac{(RC)^3}{(2N)^2} \tag{8}$$

Obviously, the DC gain, which is crucial to the stability of the closed-loop system of the plant, is related to network parameters N, C and R. No matter what algorithm is adopted, firstly, the system must have the acceptable transient and steady responses, and secondly, the algorithm should be robust enough to variations in the plant parameters because they are very changeable in real network environment. In the following experiments, we will make a comparison between FLC with PI controller about the ability against noise. According to Ref. [11], the load level N of the PI controller has low bound with some stability margins. Here, given that $N_1 = 10$, the others parameters are kept the same value in Experiment 1. The EQ is fixed at 200 packets in both controllers. Their respective instantaneous queue lengths are plotted in Figs. 10 and 11, respectively. We can observe the queue controlled by the PI controller ranges between 150 and 250 packets, on the other hand the FLC is still controlling the queue length at around 200 packets. Thus, the FLC appears to be much more robust in face of light loads.

[1] The DC gain of the transfer function $G(s)$ equals $G(0)$, i.e. the amplifying coefficient at 0 (DC) frequency [14].

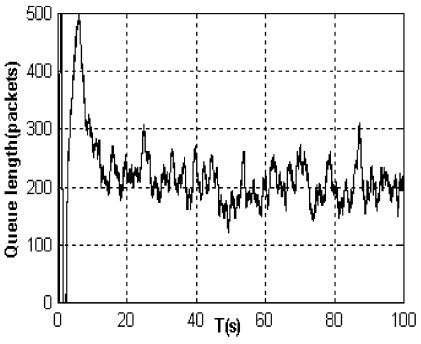

Fig. 15. PI (Experiment 4).

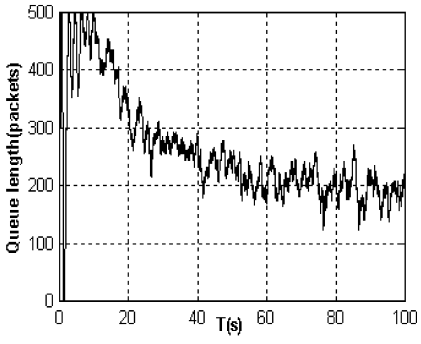

Fig. 17. PI (Experiment 5).

4.3. Experiment 3

The heterogeneity is ubiquitous in the current network, especially obvious for link capacity, such as 128 kbps ISDN line, T3 (45 Mbps) link, and OC48 (622 Mbps) link, etc. The perfect AQM scheme should be adaptable to most of them. Eq. (8) shows that the DC gain of the plant is proportional to the cube of link capacity, so it is very necessary to investigate the performance of AQM scheme under the higher link bandwidth. Therefore, we change (L_1, L_2, L_3) from (15, 15, 30 Mbps) in Experiment 1 to (100, 100, 200 Mbps), and keep the other parameters unchangeable. The queue lengths controlled by the two controllers are shown in Figs. 12 and 13. The FLC is largely insensitive to the link capacity variations, but the performance of PI controller degrades with the bandwidth increase, the oscillation magnitude reaches 100 packets, which will introduce considerable jitter to the round-trip time experienced by the data packets.

4.4. Experiment 4

Designing the algorithm of the network flow and congestion control, the delay should not be neglected, especially in WAN environment, because the large delay easily leads to instability and degradation on the performance. In this experiment, we will draw a comparison between the two controllers about the performance under the large delay. The network configuration parameters in Experiment 1 are still used except that the propagation delay is approximately tripled, i.e. τ_1 is increased from 5 to 40 ms. The simulation results are plotted in Figs. 14 and 15. The superior steady performance of FLC is observed, but the larger delay makes the FLC have the longer response time, approximates to 20 s. Of course, it is right for the PI controller, which also takes about 20 s to settle down. However, just considering the steady performance, the FLC is more robust against delay variances.

4.5. Experiment 5

In the earlier experiments, we concentrated on the robustness of the controller, which is crucial for highly variability and uncertainty network environment. From the point of view of the control theory, beside the perfect steady performance, the system compensated by the controller must have an acceptable transient respond. In this experiment, we will explore the transient performance of FLC by comparing it with the PI controller. Increasing N_1 from 60 to 100, the other parameters are kept same as in Experiment 1. Figs. 16 and 17 plot the evolution of the queue size controlled by the FLC and the PI controller, respectively. We observe that the PI controller takes the longer time to settle down the equilibrium point. For the sake of clearness, the curves of the dropping probability are plotted in Fig. 18. The regulating time of PI controller is six times larger than that of FLC.

We analyze in brief why the PI controller is so sluggish.

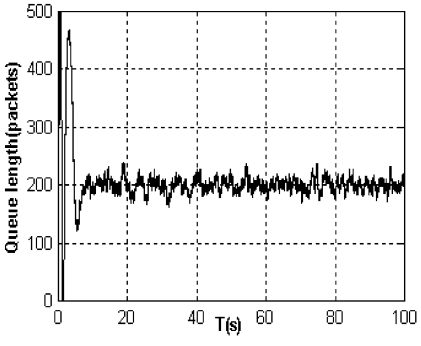

Fig. 16. FLC (Experiment 5).

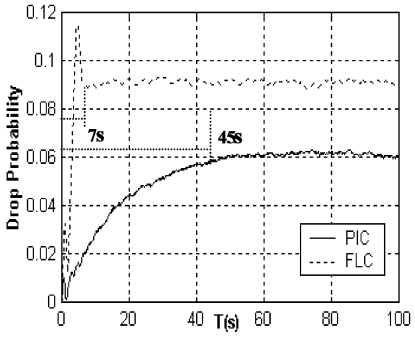

Fig. 18. Buff_size = 500 (Experiment 5).

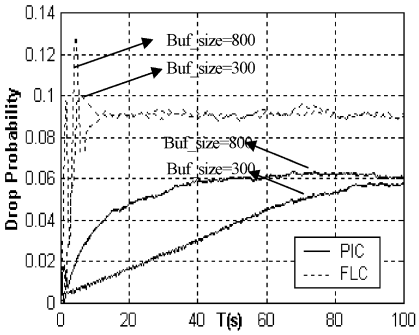

Fig. 19. Buff_size = 300, and 800 (Experiment 5).

The following is the control law of the PI controller [11]

$$p(k) = (a - b)(q(k) - q_{\text{ref}}) + b(q(k) - q(k-1)) + p(k-1) \quad (9)$$

the coefficients a and b were fixed at 1.822×10^{-5} and 1.816×10^{-5}, respectively, the sampling frequency is 160 Hz [11], the control variable p is accumulative. Because the parameter b is very small, and the sample interval is very short, the negative contribution to p made by the second item in the right can be omitted in initial process, thus the positive contribution mainly comes from the first item. Assume that $q(k)$ is average value of the transient process, i.e. 350 packets, it is not difficult to calculate the lasting time that p increases from 0 to 0.06.

$$\text{estimated regulating time} \approx \frac{p_0}{(q(k) - q_{\text{ref}}) \times (a - b)} \times \frac{1}{160}$$

$$= \frac{0.06}{(350 - 200) \times (1.822 - 1.816) \times 10^{-5}} \times \frac{1}{160} = 42 \text{ (s)}$$

This result is very close to the simulation. Considering the requirement of the steady performance, it is impractical to increase the difference between a and b to speed up the response of the PI controller. With the higher sampling frequency, the computation will be significantly exhausted. The only feasible way is to add the buffer size. In order to illustrate this ability, we redo the earlier simulation with 800 and 300 packets buffer size, respectively, and the results are plotted in Fig. 19. Indeed, the large buffer is able to enhance the responsibility of the PI controller, but this ability is limited, moreover it seems to be wasteful. Conversely, the FLC has the ideal transient performance without any additional regulating mechanism.

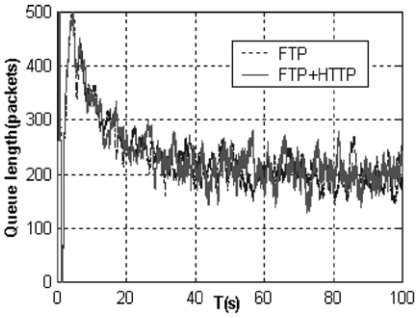

Fig. 21. PI (Experiment 6).

In addition to, one fact should be noted, that is the two controllers settle the drop probability p down the different values. The FLC regulates p to 0.09, while dropping/marking probability keeps at 0.06 with the PI controller. This means that FLC possibly drops more data packets. However, when marking instead of dropping packets, i.e. ECN [16] mechanism is used in conjunction with AQM scheme, the efficient performance is still able to obtain, because the system operates in almost lossless regimes, thereby diminishing retransmission as well as reducing unnecessary.

4.6. Experiment 6

The PI controller is based on the linear model. However, the model itself is inaccurate because of its simplification, such as, the timeout and slow start mechanisms have not been taken into account, the UDP flows were also neglected. Intuitively, the PI controller is hard to accommodate itself to the complex and changeable network status, but the FLC should not be so. Subsequently, we confirm this assumption by simulation experiments. Firstly, let $N_1 = 60$, $N_2 = 400$, $N_3 = 0$. Each burst HTTP connection has 10 sessions, and the number of pages per session is three. The evolution of the queue size is plotted in Figs. 20 and 21, which show that the two controllers can stabilize the short-lived and burst,

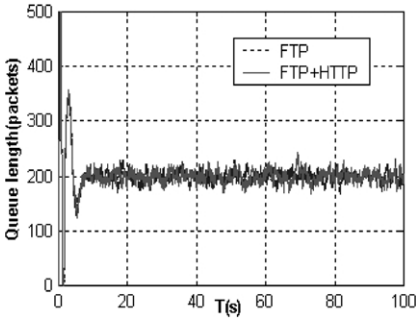

Fig. 20. FLC (Experiment 6).

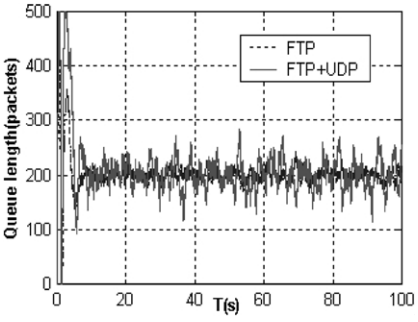

Fig. 22. FLC (Experiment 6).

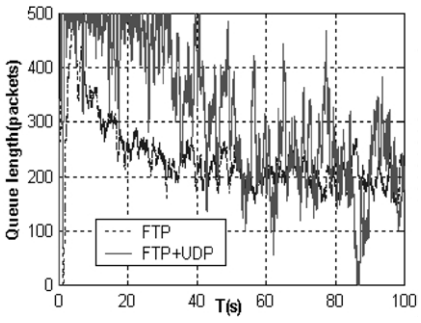

Fig. 23. PI (Experiment 6).

but responsive TCP flows. The PI controller has some stability margin after all, so the curve with HTTP and FTP flows approximately overlaps the one with only FTP connections. Observing Fig. 20, the FLC also has this attribute. Next, given that $N_1 = 60, N_2 = 0, N_3 = 10$, we further investigate the capability against the disturbance caused by the non-responsive UDP flows. UDP sources follows the exponential ON/OFF service model, the idle and burst time are 1000 and 1000 ms, respectively, and sending rate during 'on' time is 40 kbps. The experiment results are depicted in Figs. 22 and 23. Evidently, the PI controller is very sensitive to this disturbance, and the queue controlled by it exhibits oscillation with great magnitude, while the FLC operates in a relatively stable state. Of course, the queue fluctuation increases with UDP flows entering, but the variance is too much small comparing with PI controller. This experiment verifies that the FLC has more robustness than the PI controller.

4.7. Experiment 7

An important consideration in designing AQM scheme is the tradeoff between queuing delay and utilization. Intuitively, larger buffers lead to higher utilization of link, but they also result in larger queuing delays. With both of controllers, the delay is essentially tunable with a single parameter (EQ, q_0). Generally, larger values of q_0 give large delay and utilization. In order to study this tradeoff,

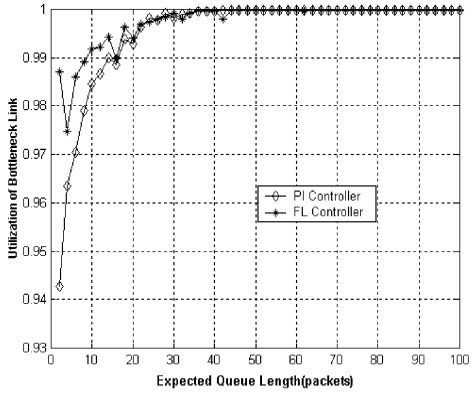

Fig. 24. Utilization vs. q_0 (FTP).

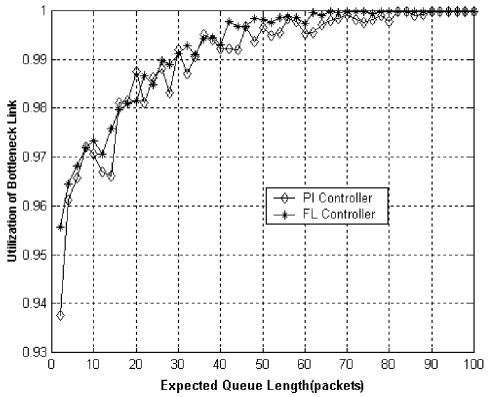

Fig. 25. Utilization vs. q_0 (FTP + UDP).

we performed experiments under two scenarios, one with purely long-lived flows (FTP), and another when the traffic flow consisted of a mixture of FTP flows and UDP (non-responsive and short-lived). We plot the EQ vs. utilization curve in Figs. 24 and 25. For both of controllers, the small q_0 can also yield very high utilization in case of pure FTP, moreover the FLC is more efficient than PI controller. When q_0 reaches at 30 packets, their utilizations are nearly full. In the experiment for mixed flows, we fixed the 200 FTP sessions and 20 burst and non-responsive UDP flows, which have the same statistic properties in Experiment 6. When the EQ is relative small, UDP sources reduce the utilization. Because the queue fluctuations caused by UDP flows in case of PI controller is more apparent than that in case of FLC, just like in Experiment 6, the FLC more efficiently utilizes the link resources than PI controller does when the q_0 is less than 80 packets, which is very important to satisfy with the goals of AQM scheme.

4.8. Experiment 8

In the last experiment, we will evaluate the integrated performance of the FLC using one relatively real scenario, i.e. the number of the active flows is changeable, which has 300 FTP flows, 400 HTTP connections and 100 TELNET flows. The FTP flows are classified into three groups, and each one has 100 connections. The HTTP and TELNET flows are evenly divided into two groups. At time $t = 0$,

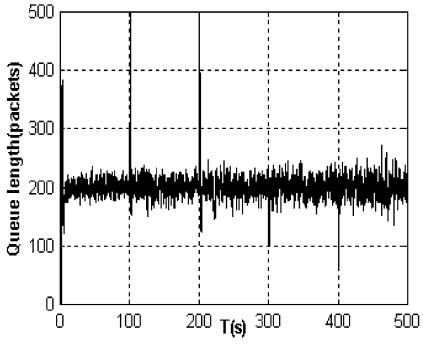

Fig. 26. FLC (Experiment 6).

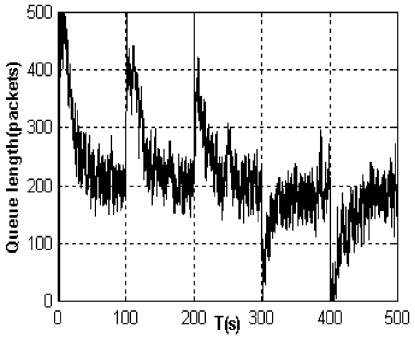

Fig. 27. PI (Experiment 6).

the first FTP flows start, the second FTP group starts at 100 s, and the third one is launched at time $t = 200$. At time $t = 300$, the first FTP group stops, but the first HTTP and TELNET groups start together. After 100 s, all the flows in the second FTP group are stopped; the other HTTP and TELNET flows are launched. The whole simulation is finished at time $t = 500$. Figs. 26 and 27 plot the evolution of the queue controlled by the FLC and the PI controller respectively. It is very clear that the FLC's integrated performance, transient and steady, is superior to the PI controller's. The queue size is always kept at the expected value, even if the network loads abruptly change, but the PI controller has the inferior adaptability, i.e. the FLC is more powerful, robust and adaptive than PI controller, which is in favor of achieving the objectives of the AQM scheme.

5. Conclusion

The AQM is an effective mechanism to support end-to-end congestion control, and necessary to health of the Internet, many researches are absorbed by AQM. Most of works mainly focus on the analysis of stability and fairness of the existed various algorithms, and the exploration of the novel schemes. According to the operating mechanism of AQM, it is reasonable to regard the TCP flow control as the typical regulating system in control theory, so the design of the AQM implementation algorithm can be naturally converted into the design of the controller with the well-developed control techniques. Hollot et al., simplified the complex non-linear model of the TCP flow control as a linear system using local linearization, and then designed the traditional PI controller using the classical control theory. We think that some assumptions needed to be contemplated. The linearization inevitably introduce into model error, in addition to, the methods used in the constant system seem to be unsuitable for the time-varying network. Thus, in this study, we have concentrated on more modern robust control methodology, i.e. the FLC, which is capable of being against the disturbances, and adapting to highly variability and uncertainty in network. We clearly presented clear guidelines towards to the designing of the FLC, implemented it in ns-2 platform, and compared performance under various scenarios with the PI controller. The FLC has the superior steady and transient performance, exhibits the great adaptability to the variances of link delay and capacity, and provides the more robustness against the noise and disturbance.

References

[1] B. Braden, et al., Recommendations on queue management and congestion avoidance in the Internet. RFC2309, April 1998.
[2] F. Anjum, L. Tassiulas, Balanced-RED: an algorithm to achieve fairness in Internet; http://www.isr.umd.edu/CSHCN/.
[3] T.J. Ott, T.V. Lakshman, L.H. Wong, SRED: Stabilized RED, Proc. INFOCOM'99, March 1999.
[4] D. Lin, R. Morris, Dynamics of random early detection, Proc. SIGCOMM'97, September 1997, pp. 127–138.
[5] T.V. Lakeshman, A. Neidhardt, T. Ott, The drop from front strategy in TCP over ATM and its interworking with other control features, INFOCOMM 1996.
[6] W. Feng, D. Kandlur, D. Saha, K. Shin, A self-configuring RED gateway, Proc. INFOCOM'99, March 1999, pp. 1320–1328.
[7] V. Firoiu, M. Borden, A study of active queue management for congestion control, Proc. INFOCOM 2000, March, 2000.
[8] M. May, T. Bonald, T. Bolot, Analytic evaluation of RED performance, Proc. INFOCOM 2000, March 2000.
[9] M. Christiansen, K. Jeffay, D. Ott, F.D. Smith, Tuning RED for web traffic, ACM SIGCOMM, August 2000.
[10] C. Hollot, V. Misra, D. Towsley, W.B. Gong, A control theoretic analysis of RED, INFOCOM 2001, Anchorage, Alaska, April 2001.
[11] C. Hollot, V. Misra, D. Towsley, W.B. Gong, On designing improved controllers for AQM routers supporting TCP Flows, INFOCOM 2001, Anchorage, Alaska, April 2001.
[12] V. Misra, W.B. Gong, D. Towsley, Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED, Proceedings of ACM/SIGCOMM 2000.
[13] E.H. Mamdani, Application of fuzzy algorithm for control of simple dynamic plant, Proc. IEEE 121 (1974) 12.
[14] UCN/LBL/VINT, Network simulator-ns2: http://www-mash.cs.berkeley.edu/ns.
[15] W. Pedrycz, Fuzzy Control and Fuzzy System (Second Edition). Research Studies Press Ltd. Taunton, Somerset, England, 1993.
[16] K.K. Ramakrishnan, S. Floyd, A proposal to add explicit congestion notification (ECN) to IP, RFC 2481, January 1999.