A Robust Active Queue Management Algorithm Based on Sliding Mode Variable Structure Control

Ren Fengyuan Lin Chuang
Department of Computer Science
Tsinghua University
Beijing 100084, China
renfy@csnet1.cs.tsinghua.edu.cn

Ying Xunhe Shan Xiuming
Department of Electroinc Engineering
Tsinghua University
Beijing 100084, China
yinxh@sdp.ee.tsinghua.edu.cn

Wang Fubao
Department of Computer Science
Northwestern Polytechnical University
Xi'an 710072,China
wangfb@nwpu.edu.cn

***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. Most of existed AQM algorithms are heuristic, and lack systematic and theoretical design and analysis approach. From the viewpoint of the control theory, it is rational to regard AQM as a typical regulating system. Although PI controller for AQM outperforms RED algorithm, the mismatches in simplified TCP flow model inevitably degrades the performance of controller designed with classical control theory. In this paper, a robust SMVS controller for AQM is put forward based on Sliding Mode Variable Structure Control (SMVS), its superiority is insensitive to noise and variance of the parameters, thus it very suitable to time-varying network system. The principle and guidelines on design of SMVS controller are presented in details. The integrated performance is evaluated using *ns* simulations. The results show that the SMVS is very responsive and robust against the disturbance. At the same time, a complete comparison between SMVS controller and PI controller is made. The conclusion is that both transient and steady performance of SMVS controller is superior to that of PI controller, thus the SMVS controller is in favor of the achievement to AQM objectives.**

A. 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 end system. It is needed to implement some measures in the intermediate nodes to complement the end system congestion avoidance mechanisms. Active Queue Management, 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]. It has been a very active research area in the Internet community. 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. 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 real-time interactive applications. RED[2] was originally proposed to achieve fairness among sources with different burst attributes and to control queue length, which just meets the requirements of AQM, so RED was recommended as only candidature algorithm for AQM in RFC 2309. However, many subsequent studies verified that RED is unstable and too sensitive to parameter configuration [3][4], and tuning of RED has proven to be a difficult job [5]. Numerous variants of RED have been proposed to work around some problems existed in RED, such as RED with "gentle_", Balanced RED [6], SRED [7], FRED [8], BLUE [9] and Self-Configuring Gateway [10] etc. Most of them, including RED, are heuristic algorithms depending on the intuition. The partial simulations and experiments on the special network configuration is only measure to validate the algorithm performance, and the systematic and theoretic analysis and evaluation are neglected. Thus we lack the whole understanding about the algorithm performance and efficiency. Once the problems occur in practice, the remediation was made through new simulations or experiments. For instance, the original paper about RED presented the impact of the configuration parameter on the performance, and suggested the guidelines towards the appropriate values based on heuristic and simulation. Subsequently, some further investigations found that they were not optimal, and then gave the current suggestion [11], in which w_q (weighted factor) is increased from 0.001 to 0.002 because w_q is too low to timely detect the congestion in relative high speed network; max_p (maximum packet dropping probability) is set to 0.1 instead of to 0.02 because 2% drop probability is not enough to force multiple TCP sources to sufficiently reduce their window sizes, especially in many short-lived and burst HTTP sessions. The intuition and heuristic design is not always scientific and reasonable under any conditions. Of course, since the Internet is a rather complex huge system, it is very difficult to have a full-scale and systematic comprehension, but importance has been considerably noted. The mathematical modeling of Internet is the first step to have an in-depth understanding, and the algorithms designed based on the rational model should be more reliable than one originated from intuition. The non-linear dynamic model for TCP flow control [12] inspired C. Hollot et al to design the PI controller for AQM [13], S. Kunniyur and R. Srikant also proposed the Adaptive Virtual Queue (AVQ) algorithm for AQM based on this non-linear model [14]. Kelly et al constructed a unified framework [15], the problem of congestion control was formulated as a convex program, with the aggregate source utility being maximized subject to bandwidth constrain. REM [16] was a controller designed in a dual formulation to obtain optimal source rates. Although PI controller successfully revealed some limitations of RED, for instance, the queue length and dropping/marking probability are decoupled, whenever the queue length can be easily controlled to the desired value; the system has relatively high stability margin. The shortcomings of PI controller are also obvious. The modification of probability excessively depends on buffer size. As a result, for small buffer the system exhibits sluggishness. Secondly, for small reference queue length, the system tends to performance poorly, which is unfavorable to achieve the goal of AQM because small queue length implies small queue waitting delay. Thirdly, the status of actual network is rapidly changeable, so we believe that it is problematic and unrealistic, at least inaccurate, to take the network as a linear and constant system just like the designing of PI controller. Affirmatively, the algorithm based on this assumption should have limited validity, such as inability against disturbance or noise. We need more robust controller to adapt complex and mutable network environment, which will be our motivation and aim in this study. In this paper, we will apply one of advance robust control theory, Sliding Mode Variable Structure control, to design the AQM controller, and expect it to have the perfect performance and be better suited for AQM than PI controller. The reminder of the paper is organized as follows. In Section II, we provide an introduction to the background on TCP flow control model and Sliding Mode Variable Structure control. Section III develops the SMVS controller, and presents design guidelines. In the next Section, we testify the validity of SMVS controller, and then compare its performance with that of PI controller by numerical simulation results. Finally the conclusion is drawn in Section V.

B. BACKGROUND

Ⅰ. TCP Flow Control Model

In [12], a non-linear dynamic model for TCP flow control was developed based on fluid-flow theory. The following is a simplified version of that model.

$$\begin{cases} \dfrac{dW(t)}{dt} = \dfrac{1}{R(t)} - \dfrac{W(t)W(t-R(t))}{2R(t)}\, p(t-R(t)) \\ \dfrac{dq(t)}{dt} = \dfrac{N(t)}{R(t)} W(t) - C(t) \end{cases} \tag{1}$$

Subsequently, C. Hollot et al a this nonlinear and time-varying system was approximated as a linear constant system by small-signal linearization about an operating point [17]. The Fig. 1 illustrates this linearized model. The details about modeling and linearization can be seen in [12] and [17]. In the block diagram, C(s) is the compensator or controller, G(s) is the plant, namely controlled object. The meanings of parameters presented in Fig. 1 are following:

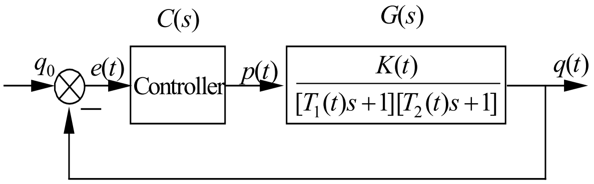

Fig.1 Block Diagram of AQM Control System

$$K(t) = \frac{[R(t)C(t)]^3}{[2N(t)]^2} \quad T_1(t) = R(t) \quad T_2(t) = \frac{R^2(t)C(t)}{2N(t)}$$

where

$C(t)$: Link capacity (packets/sec)

q_0 : Queue reference value

$N(t)$: Load factor, i.e. number of active sessions

$R(t)$: Round-Trip Time (RTT), $R(t)=2(q(t)/C(t)+T_p)$, T_p is the fixed propagation delay.

$p(t)$: Dropping/marking probability.

$q(t)$: Instantaneous queue.

The control system depicted in Fig. 1 is basically consistent with that presented in [13], both of them are linear second-order system. The only difference is that the above system is time-varying, but the system parameters described in [13] are constant. The linear constant model is useful and helpful to analyze and explain the instability of RED under some network parameter configuration [17]. Nevertheless, we believe that the AQM controller designed with the simplified and inaccurate linear constant model should not be optimal, because the actual network is very changeful, the state parameters are hardly kept at the constant values for a long time. Moreover, the equations (1) only take consideration into the fast retransmission and fast recovery, but ignore the timeout mechanism caused by lacking of enough duplicated ACK, which is very usual in burst and short-lived services, such as Telnet and HTTP. In addition to, there are many non-responsive UDP flows besides TCP connections in networks, they are also not included in equations (1). These mismatches in model will have negative impact on the performance of controller designed with the approach depending on the accurate model. For the changeable network, the robust control should be an appropriate approach to design controller for AQM. The sliding model variable structure control is one of the best techniques in robust control theory.

Ⅱ. Sliding Mode Variable Structure Control [20]

In the formulation of any control problem, there will typically be discrepancies between the actual plant and the mathematical model developed for controller design. This mismatch may be due to unmodelled dynamics, variation in system parameters or the approximation of complex plant behavior by a straightforward model. The perfect controller should have the ability to produce the required performance in practice despite such plant/model mismatches. As evidenced by its name, the structure of SMVS control is not constant but is varied during the control process, the controllers are designed to drive and then constrain the system state to lie within a neighborhood of the switching function. There are two main advantages to this approach.

Firstly, the dynamic behavior of the system may be tailored by the particular choice of switching function. Secondly, the closed-loop respond becomes totally insensitive to a particular class of uncertainty. The sliding mode design approach consists of two components. The first involves the design of a switching function so that the sliding motion satisfies design specification. The second is concerned with the selection of a control law that makes the switching function attract the system state. For the sake of clarity, we briefly illustrate by the system in Fig.2. Suppose that at some time t_0 the system lies on the initial point A_0, and then moves along one phase trajectory under the control law $u = k_1 x_1$. At time $t_1 > t_0$ the system reaches and crosses the line $\sigma = w x_1 + x_2$, the control law is switched to $u = k_2 x_1$, the system will move along another phase trajectory, but it can not be remain there, since the system will immediately cross the line $\sigma = 0$ again. The control law $u = k_1 x_1$ intends to make the system cross the line $\sigma = 0$ at the third time, and so on. The control law is frequently switched back and forth, the system oscillations about the switching line $\sigma = 0$ (also called as sliding mode line), eventually moves towards to the origin. Because the sliding mode line is designed to be irrelative to the system itself, the SMVS control has an important property: the corresponding steady status of the system is independent of changes in the plant parameters and of external disturbance, which is very valuable for the changeable networks.

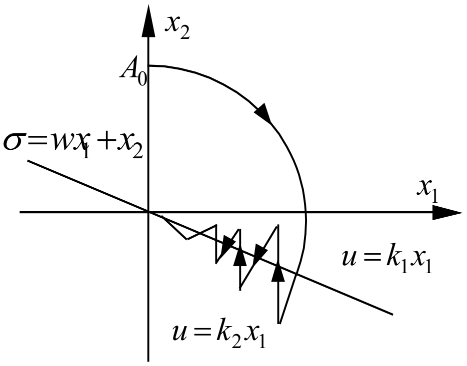

Fig. 2 Typical SMVS System

C. SMVS ALGORITHM

In this section, we discuss the design of SMVS controller for AQM. Firstly, suppose that $x_1 = e$, $x_2 = \frac{d}{dt} e = \frac{d}{dt} x_1$, the plant depicted in Fig. 1 is described by a second-order system of differential equations:

$$\begin{cases} \dfrac{dx_1}{dt} = x_2 \\ \dfrac{dx_2}{dt} = -a_1(t) x_1 - a_2(t) x_2 - b(t) p + F(t) \end{cases} \quad (2)$$

$$a_{2\min} \le a_2(t) \le a_{2\max}$$
$$a_{1\min} \le a_1(t) \le a_{1\max}$$
$$0 < b_{\min} \le b(t) \le b_{\max}$$

where

$$a_1(t) = \frac{1}{T_1(t) T_2(t)}$$
$$a_2(t) = \frac{T_1(t) + T_2(t)}{T_1(t) T_2(t)}$$
$$b(t) = \frac{K(t)}{T_1(t) T_2(t)}$$
$$F(t) = \frac{d^2}{dt^2} q_0 + \frac{T_1(t) + T_2(t)}{T_1(t) T_2(t)} \frac{d}{dt} q_0 + \frac{1}{T_1(t) T_2(t)} q_0$$

F (t) is regarded as the system disturbance. For the convenience of implementation, we adopt the following control law form:

$$p = \psi x_1 \quad (3)$$

where

$$\psi = \begin{cases} \alpha, & if \quad x_1 \sigma > 0 \\ \beta, & if \quad x_1 \sigma < 0 \end{cases}$$

The switching function σ is defined as following:

$$\sigma = w x_1 + x_2 = 0 \qquad w > 0 \quad (4)$$

At the points of the switching line $\sigma = 0$, we have:

$$\lim_{\sigma \to 0} \frac{d\sigma}{dt} = (w a_2(t) - w^2 - a_1(t) - b(t)\psi) x_1$$

Therefore

$$\lim_{\sigma \to +0} \frac{d\sigma}{dt} = \begin{cases} (w a_2(t) - w^2 - a_1(t) - \alpha b(t)) x_1 & if \quad x_1 > 0 \\ (w a_2(t) - w^2 - a_1(t) - \beta b(t)) x_1 & if \quad x_1 < 0 \end{cases} \quad (5)$$

$$\lim_{\sigma \to -0} \frac{d\sigma}{dt} = \begin{cases} (w a_2(t) - w^2 - a_1(t) - \beta b(t)) x_1 & if \quad x_1 > 0 \\ (w a_2(t) - w^2 - a_1(t) - \alpha b(t)) x_1 & if \quad x_1 < 0 \end{cases} \quad (6)$$

According to the existence condition for a sliding line: $\lim_{\sigma \to 0} \sigma \frac{d\sigma}{dt} < 0$ [18], we have:

$$\alpha \ge \max_{a_1, a_2, b} \frac{1}{b(t)} [w a_2(t) - w^2 - a_1(t)]$$
$$\beta \le \min_{a_1, a_2, b} \frac{1}{b(t)} [w a_2(t) - w^2 - a_1(t)] \quad (7)$$

Here, if conditions (7) are satisfied, the system must have a sliding regime on the switching line $\sigma = 0$, but we are unable to determine if the system cloud hit this sliding line. The following theorem can answer this question.

Theorem A necessary and sufficient condition for hitting to occur in the system consisting with (2) and (3) is that the characteristic equation (8) has no nonnegative real roots

$$p^2 + a_{2\min} p + a_{1\min} + \inf_t\{\alpha b(t)\} = 0 \qquad (8)$$

The detail proof can be found in [18]. The conservative condition that equation (8) has no nonnegative roots is it has complex roots, and so

$$a_{2\min}^2 - 4(a_{1\min} + \inf_t\{\alpha b(t)\}) < 0$$

Namely

$$\alpha > \frac{a_{2\min}^2 - 4a_{1\min}}{4b_{\min}} \qquad (9)$$

For a variable-parameters and single-input system, the excellent performance could be reached if the conditions (9) and (7) would be satisfied, at the same time, the control law (3) should also be implemented. For the sliding mode parameter *w*, only requirement is to keep it more than zero [18]. Theoretically speaking, *w* should be relatively large since it is larger the transient process is shorter. However, for the network queue management system, *w* is limited. Taking *p*(t) into consideration, its meaningful value is in section [0,1], i.e. the control variable is limited. Therefore, we need to choose the combination of constant control and proportional control as new control law:

$$p = \begin{cases} \psi x_1, & |x_1| < M \\ \theta \ , & |x_1| > M \end{cases} \qquad (10)$$

where

$$\theta = \begin{cases} M, & \sigma > 0 \\ m\,, & \sigma < 0 \end{cases}$$

M and m respectively denotes the maximum and minimum of control variable, i.e. M=1 and m=0. The other parameters are same with (3). For the sake of convenience, suppose that $k = \alpha > 0 > \beta = -k$. Since the control variable is limited, the sliding mode regime in phase space will be restricted in a certain scope, so that the choice of sliding mode parameter *w* is also restricted, otherwise, the controlled variable (queue length) would rush out of the sliding mode regime, and sharp overshoot and great oscillation will occur, which degrades the performance of router. According to the conclusion drawn in [18], we have $w \le w_{\max} = \sqrt{a_1(t)}$.

Subsequently, we start to design SMVS controller for AQM based on the above theory and approach. For the purpose of SMVS controller being suitable for most of dynamic system, the varying scope of parameters in TCP/AQM system is assumed as following:

N(t): 1~300 T_p: 0.02*sec*
q_0: 0~300 (*packets*) (packet default size of is 500bytes)
C (t): 1250~7500 (*packets/sec*)

Calculating the values of parameters in (2), we yield:

$$a_{2\min} = 3.8501\,,\ a_{2\max} = 1250\,;$$
$$a_{1\min} = 0.015\,,\quad a_{1\max} = 60000\,;$$
$$b_{\min} = 2604.2\,,\ b_{\min} = 28125000$$

Let the sliding mode parameter *w*=2. According to the existence condition (7), we have:

$$\alpha > 0.958; \quad \beta < -0.0021 \qquad (11)$$

With the reachable condition of sliding mode (9), we also have:

$$\alpha > 0.0015 \qquad (12)$$

Combining (11) with (12), the parameter in control laws is determined:

$$k = \alpha = -\beta = 0.96 \qquad (13)$$

So far, a novel AQM algorithm based sliding mode variable structure control theory is obtained, and named as SMVS algorithm. The following pseudo-code describes its implementation.

```
/* SMVS Algorithm */
q(k): queue length            q0:expected length
T  : sample time              x1(k)=q(k)-q0
x2(k)=(x1(k)- x1(k-1))/T      z(k)=2.0x1(k)+ x2(k)

At each packet arrival epoch do
 If (now_time()>last_time+T)
     If (x1(k)>1.0)
         If (z(k)>0.0)
             p(k)=0.0
         else
             p(k)=1.0
     else
         If (x1(k)z(k)>0.0)
             p(k)=0.96x1(k)
         else
             p(k)=-0.96x1(k)
     last_time=now_time()
else
     null
If (p(k)>random()/maxRandom)
     mark or drop packet
else
     put packet into queue
```

Fig. 3 Pseudo-code of SMVS Algorithm

D. SIMULATIONS

We evaluate the effectiveness and performance of the SMVS controller by simulations using ns2 simulator [19]. The network topology is shown in Fig 4. The only bottleneck link lies between node A and node B. The buffer size of the node A is 300 packets, and default size of the packet is 500 bytes. Queue A is SMVS scheme, and the others are Drop Tail. All sources are classed into three groups. The first one includes N_1 greedy sustained FTP application sources, the second one is composed of N_2 burst HTTP connections, each connection has 10 sessions, and the number of pages per session is 3. The third one has N_3 UDP sources, which follow the exponential ON/OFF service model, the idle and burst time are 10000ms and 1000ms respectively, and the sending rate during "on" duration is 40Kbps.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_1, τ_1). The pair (L_2, τ_2) and (L_3, τ_3) define the parameter of link AB and link BC.

Experiment 1

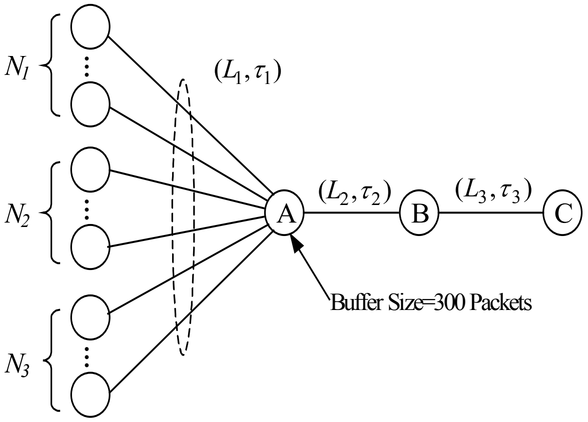

Fig. 4 The Simulation Network Topology

In this experiment, we will use the most general network configuration to testify whether the SMVS controller can reach the goals of AQM, and freely control the queue length to stabilize at the arbitrary expected value. Therefore, given that (L_1, τ_1) =(10Mbps, 15ms), (L_2, τ_2)= (15Mbps, 15ms), (L_3, τ_3) =(45Mbps, 15ms). N_1=270, N_2=N_3=0. Let the expected queue length equal to 75 packets, the instantaneous queue length is plotted in Fig. 5. After a very short regulating process, the queue settles down its stable operating point. This ability is vital to achieve the AQM objectives, at the same time, beneficial to control the queuing delay to satisfy with the special QoS requirements. RED algorithm is unable to accurately control the queue length to the desired value. Under the given parameters settings, the queue length varies with network loads. The load is heavier the queue length is longer. Attempting to control queue length through decreasing the interval between high and low thresholds, then it is likely to lead queue oscillation. Although PI controller could regulate the queue to the fixed point, the integrated performance needs to be improved, such as the transient process is too long and the fluctuation in steady state is great, for small reference queue length, which lows the link utilization. That is conflicting with the essential objectives of AQM. The queue evolution of router A, controlled by PI scheme (q_0=75packets), is presented in Fig.6. Obviously, the PI controller takes the longer time to settle down the reference point, it approximates to 35 seconds. For the sake of clearness, the curves of probability are plotted in Fig.7.

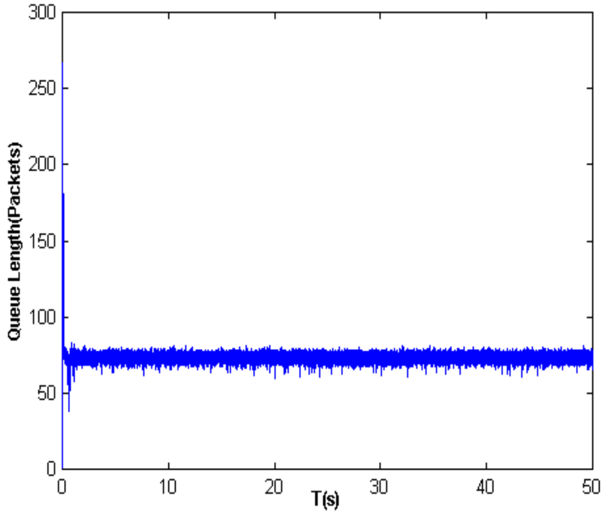

Fig.5 Queue Evolution (SMVS)

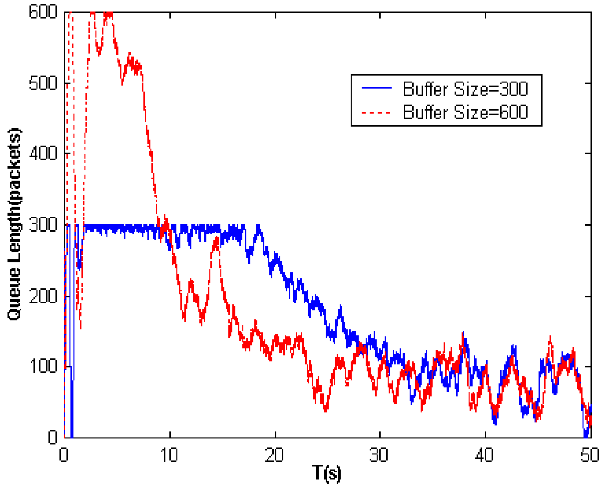

Fig.6 Queue Evolution (PI)

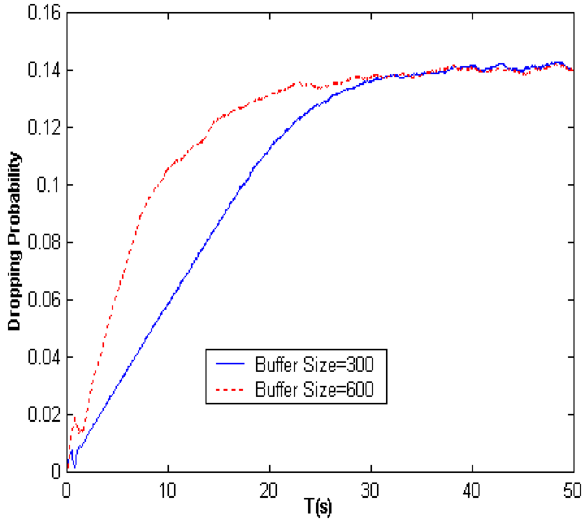

Fig.7 Probability Variance (PI)

We briefly analyze why the PI controller is so sluggish. (14) is the control law of the PI controller [13]

$$p(k)=(a-b)(q(k)-q_0)+b(q(k)-q(k-1))+p(k-1) \quad (14)$$

the coefficients a and b are fixed at 1.822e-5 and 1.816e-5 respectively, the sampling frequency is 500Hz, 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, then the positive contribution mainly come from the first item. Assume that q(k) is average value of buffer size and reference queue length in transient process, i.e. p(k)=(300+75)/2=188packets, it is not difficult to calculate the lasting time that p increases from 0 to 0.14:

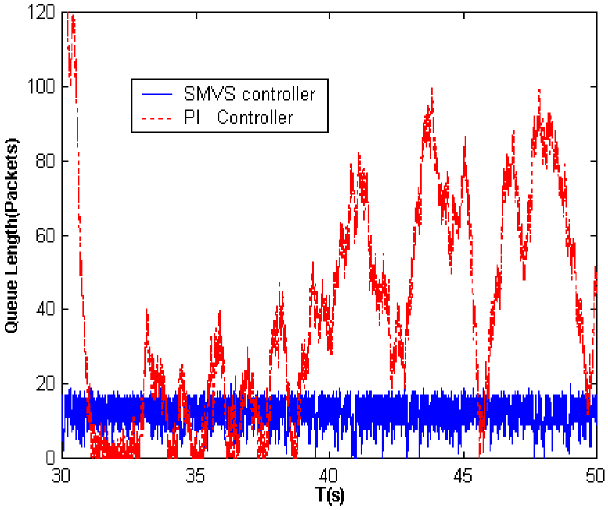

Fig.8 Small Expected Queue (q_0=15)

$$estimated \quad regulating \quad time \approx \frac{p_0}{(q(k)-q_0)\times(a-b)} \times \frac{1}{500}$$

$$= \frac{0.14}{(188-75)\times(1.822-1.816)\times 10^{-5}} \times \frac{1}{500} = 41.3(s)$$

This result is 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 above simulation with 600 packets buffer size, and the results are also plotted in Fig.6 and Fig.7. 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 SMVS controller has the ideal transient performance without any additional regulating mechanism. In order to evaluate the performance in steady state, we calculate the average and the standard deviation of the queue length in steady state. For the convenience of comparison, choose the queue length between 40 and 50 seconds as sample data. The computing results are listed in Table 1. The standard deviation of PI controller is much larger than one of SMVS controller. On the one spectrum, this lager queue deviation will introduces the delay jitter, on the other spectrum, when the reference queue length is rather small, the queue is apt to empty, so that the link utilization declines. The Fig. 8 presents the case of small reference queue length. Except q_0=15 , the other parameters are unchangeable.

Table1 Mean and standard deviation of queue length

	PI	SMVS
Mean	74.6459	72.8875
Standard Deviation	32.3336	2.5928

Experiment 2

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 environment, but the SMVS controller should be robust and capable of resisting mismatches of model and disturbances. Subsequently, we confirm this assumption by simulation experiments. Firstly, let N_1=270, N_2=400, N_3=0, the evolution of the queue size is plotted in Fig.9, which shows that both of controllers can stabilize the short-lived and burst, but responsive TCP flows because the PI controller has some stability margin after all. Of course the SMVS controller also has this ability. Next, given that N_1=270, N_2=0, N_3=50, we further investigate the performance against the disturbance caused by the non-responsive UDP flows. The experiment results are depicted in Fig. 10. Evidently, the PI controller is very sensitive to this disturbance, and the queue controlled by it exhibits oscillation with great magnitude, while the SMVS controller operates in a relatively stable state. The queue fluctuation increases with introducing the UDP flows, but the variance is too much small comparing with PI controller. This experiment verifies that the SMVS controller is much more robust than the PI controller.

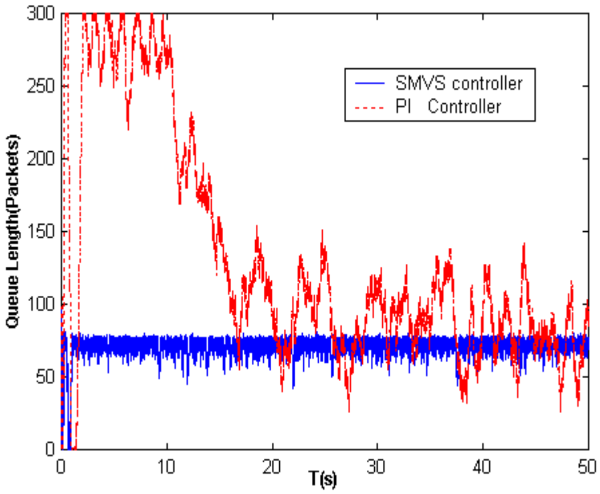

Fig.9 Queue Evolution (FTP+HTTP)

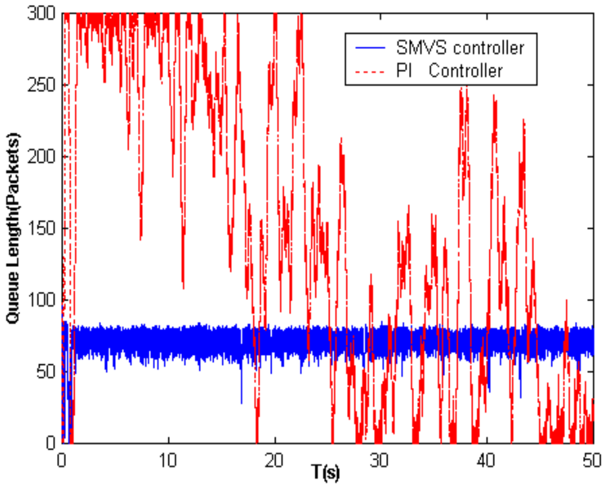

Fig.10 Queue Evolution (FTP+UDP)

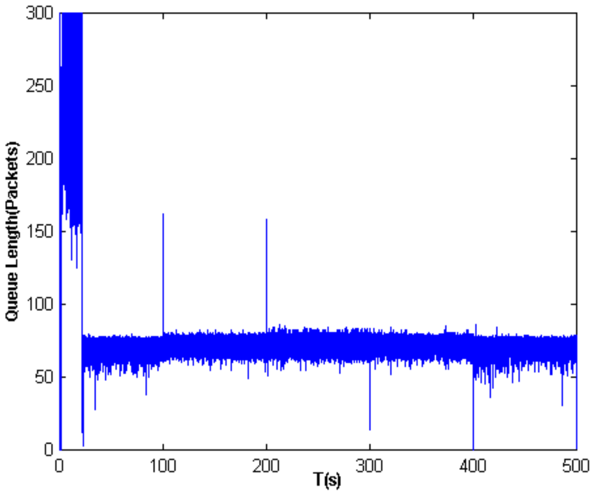

Fig.11 Queue Evolution (SMVS)

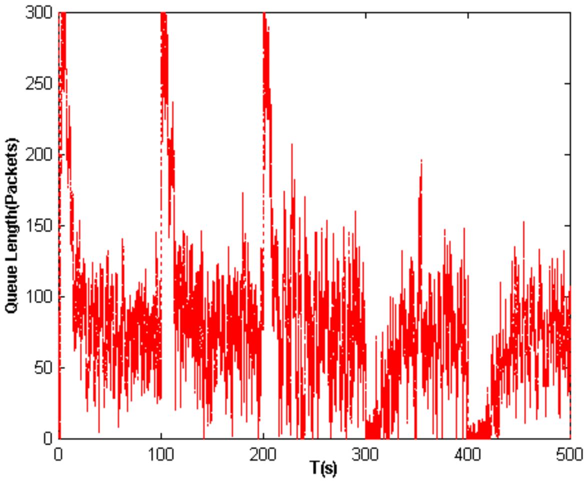

Fig.12 Queue Evolution (PI)

Experiment 3

Finally, we evaluate the integrated performance of the SMVS controller using one relatively real scenario, i.e. the number of the active flows is changeable, which has 270 FTP flows, 400 HTTP connections and 30 UDP flows. Both FTP and UDP flows are averagely divided into three groups. At time t=0, the first group flows start; the second FTP group and the second UDP group start at 100 second, and the other services (including all HTTP sessions, the third group of FTP and UDP flows) are launched at time t=200. At time t=300, the first group of FTP and UDP flows stops, After 100 seconds, all the flows in the second FTP and UDP group are terminated; The whole simulation is finished at time t=500. The Fig. 11 and Fig. 12 plot the evolution of the queue controlled by the SMVS and the PI controller respectively. Obviously, the integrated performance of SMVS controller, namely transient and steady responds, is superior to that of PI controller. The SMSV controller is always keeping the queue length at the reference value, even if the network loads abruptly change, but the PI controller has the inferior adaptability. In other words, the SMVS controller is more powerful, robust and adaptive than PI controller, which is in favor of achievement to the objectives of the AQM policy.

E. CONCLUSION AND FUTURE WORKS

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 schemes, and the exploration of the heuristic algorithm. Since the intuition and partial simulations are not absolutely reliable, the precise and systematic approaches should be used in designing and evaluating algorithms for flow and congestion control. According 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 AQM implementation algorithm can be naturally converted into the design of controller with the well-developed control techniques. C.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 introduces model error. In addition to, the methods used in the constant system seem to be inappropriate to the time-varying network. In this study, we applied sliding mode variable structure control to design AQM algorithm because this advance robust control methodology is insensitive to system dynamic parameters, and is capable of being against the disturbances and noise, which is very suitable for the mutable network environment. We clearly presented the principle and guidelines to designing of SMVS controller, implemented it in ns-2 platform, and evaluated the performance by simulations. Finally, we took a complete comparison with PI controller under various scenarios. The conclusion is that the integrated performance of SMVS controller is superior to that of PI controller. The SMVS controller is very responsive, stable and robust, especially for the small reference queue system, but its performace is inferior when active TCP sessions is relatively small, it is difficult to constrain queue oscillation. Thus it will be very imperious to design the controller suitable for light load, then integrated it with SMVS controller using adaptive control technology. We are currently investigating according to this direction.

F. ACKNOWLEDGEMENTS

This work has been supported by the National Natural Science Foundation of China(No.69972040) and the Projects of Development Plan of the State Key Fundamental Research (No.G1999032707)

REFERENCES

[1] B. Braden, et al. ‘Recommendations on Queue Management and Congestion Avoidance in the Internet’. RFC2309, April 1998.

[2] S. Floyd and V. Jacobson. ‘Random Early Detection Gateway for Congestion Avoidance’. IEEE/ACM Transactions on Networking, August 1993.

[3] V. Firoiu and M. Borden. A Study of Active Queue Management for Congestion Control. Proc. INFOCOM 2000, March 2000

[4] M. May, T. Bonald and T.Bolot. Analytic Evaluation of RED Performance. Proc. INFOCOM 2000, March 2000

[5] M. Christiansen, K. Jeffay, D. Ott, and F.D. Smith, Tuning RED for Web Traffic, ACM SIGCOMM, August 2000.

[6] F. Anjum and L. Tassiulas. Balanced-RED: An Algorithm to Achieve Fairness in Internet. http://www.isr.umd.edu/ CSHCN/

[7] Teunis J. Ott, T.V. Lakshman and Larry H. Wong. SRED: Stabilized RED. Proc. INFOCOM’99, March 1999

[8] D. Lin and R. Morris. Dynamics of Random Early Detection. Proc. SIGCOMM1997.

[9] W. Feng, D. Kandlur, D. Saha and K. Shin. Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness. Proc. INFOCOM 2001

[10] W. Feng, D. Kandlur, D. Saha and K. Shin. A Self-Configuring RED Gateway. Proc. INFOCOM’99, March 1999, pp1320-1328

[11] S.Floyd,RED: Discussion of Setting Parameters, http://www.aciri.org/floyd/redparameters.txt

[12] 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. Proc. ACM/SIGCOMM 2000.

[13] C. Hollot, V. Misra, D. Towsley and W. B. Gong. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. Proc. INFOCOM 2001

[14] Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. Proc. ACM/SIGCOMM 2001

[15] F. P. Kelly, A. Maulloo and D. Tan. Rate Control in Communication Networks. Journal of the Operation Research Society, vol.49,pp.237-252,1998

[16] S. Athuraliya, D.E. Lapsley and S.H.Low. Random Early Marking for Internet Congestion Control. Proc. IEEE Globecom 1999.

[17] C. Hollot, V. Misra, D. Towsley and W. B. Gong. A Control Theoretic Analysis of RED. Proc. INFOCOM 2001.

[18] U. Itkis. Control System of Variable Structure. Keter Publishing House Jerusalem Ltd.1976

[19] UCN/LBL/VINT. Network Simulator-NS2. http://www-mash.cs.berkeley.edu/ns

[20] J. Ackermann and V. Utkin. Sliding Modes in Control Optimization. IEEE Transactions on Automatic Control. Feb. 1998