Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Using Lyapunov function to design optimal controller for AQM routers

ZHANG Peng†, YE Cheng-qing, MA Xue-ying, CHEN Yan-hua, LI Xin

(*School of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*)

†E-mail: zhang_peng_zju@yahoo.com.cn

Received Feb. 10, 2006; revision accepted Aug. 23, 2006

Abstract: It was shown that active queue management schemes implemented in the routers of communication networks supporting transmission control protocol (TCP) flows can be modelled as a feedback control system. In this paper based on Lyapunov function we developed an optimal controller to improve active queue management (AQM) router's stability and response time, which are often in conflict with each other in system performance. Ns-2 simulations showed that optimal controller outperforms PI controller significantly.

Key words: TCP flows, Active queue management (AQM), Lyapunov function, Optimal controls

doi:10.1631/jzus.2007.A0113 **Document code:** A **CLC number:** TP393

INTRODUCTION

Internet congestion occurs when the aggregated demand for a resource (e.g., link bandwidth exceeds the available capacity of the resource). Resulting effects from such congestion include long delays in data delivery, waste of resources due to lost or dropped packets, and even possible congestion collapse (Athuraliya *et al*., 2001). Therefore it is very necessary to avoid or control network congestion. The active queue management (AQM) scheme implemented in routers is designed for this purpose. AQM can maintain stable queuing and higher throughput by purposefully dropping packets at intermediate nodes (Sun *et al*., 2003a).

Recent literature reported several AQM schemes to provide early congestion notification to users. Among them random early detection (RED), which was originally proposed to achieve fairness among sources with different burstness and to control queue length, is probably the best known (Floyd and Jacobson, 1993). The performance of RED had been evaluated through simulations and experiments in real networks (May *et al*., 1999). And it is well known that RED is quite sensitive to parameter settings. Floyd, the designer of RED, and other researchers gave several guidelines for parameter settings, such as ARED (Floyd *et al*., 1997), SRED (Ott *et al*., 1999), "gentle_RED" (Floyd, 2000), etc. Unfortunately with improvement of its performance, RED becomes more and more complex to implement.

Another approach for AQM design is through control theory. Many kinds of controllers have been proposed, such as PI (Hollot *et al*., 2001a), PD (Sun *et al*., 2003b), PID controller (Fan *et al*., 2003), and so on. Among these schemes, proportional-integral (PI) controller designed by Hollot is the most representative, and others can be looked upon as variations of it. This controller is expected better responsiveness by calculating packet drop probability based on the current queue length instead of the average queue length (Hollot *et al*., 2001b). The results of various simulations showed that PI outperforms RED in regulating steady state queue length to a desired reference value with changing levels of congestion. Although this controller improves the performance of AQM, there are many experiments and simulations and calculations indicating that PI is sluggish and the regulating time is too long (Fan *et al.*, 2003). Apart from PI controller, REM (Athuraliya *et al.*, 2001) and AVQ (Kunniyur and Srikant, 2001) also use integral factor in their algorithm. They have similar disadvantage with PI controller in response speed.

In fact stability and response time often mutually conflict in system performance. It is an intractable problem to find tradeoff between them. To solve this problem, in this paper we apply time optimal control theory to design a novel controller that can decrease the response time and improve the stability of TCP/AQM congestion control. Different from PI and other variations of it, time optimal controller focuses on finding an satisfactory control policy which transfers control variable from current value to desired reference value in minimum time. Moreover, we design this time optimal controller through Lyapunov stability theory, so we can expect this algorithm has better performance in response time and stability than PI controller. Note that our work is only for a single link. There is a large body of literature now on control-theoretical analysis of congestion management mechanisms in a network, rather than in a single link. These are also optimization-based approaches, and their details can be found in (Srikant, 2004).

The remainder of the paper is organized as follows. In Section 2, we present a novel algorithm called time optimal controller (TOC) and give a theoretical law for choosing the parameters to achieve system stability. Simulation results showed that TOC has better network performance compared with PI scheme under different network conditions by using ns-2 in Section 3. Finally, we conclude our work in Section 4.

DESIGNING TIME OPTIMAL CONTROLLER

The simplified dynamic model of TCP behavior is shown in the following equations (Misra *et al.*, 2000):

$$\left.\begin{aligned} \frac{\mathrm{d}W_i}{\mathrm{d}t} &= \frac{1}{R_i(q)} - \frac{W_i \cdot W_i(t-\tau)}{2R_i(q(t-\tau))} p(t-\tau), \\ \frac{\mathrm{d}q(t)}{\mathrm{d}t} &= \frac{W(t)}{R(t)} N(t) - C, \end{aligned}\right\} \quad (1)$$

where

$W \triangleq$ expected TCP window size (packets);
$q \triangleq$ expected queue length (packets);
$T_p \triangleq$ propagation delay (s);
$C \triangleq$ link capacity (packets/s);
$R \triangleq$ round-trip time=$(q/C)+T_p$ (s);
$N \triangleq$ load factor (number of TCP sessions);
$P \triangleq$ probability of packet mark/drop.

In (Hollot *et al.*, 2001b), linearized equation of Eq.(1) is proposed as follows:

$$\left.\begin{aligned} \delta\dot{W}(t) &= -\frac{2N}{R_0^2 C} \cdot \delta W(t) - \frac{R_0 C^2}{2N^2} \cdot \delta p(t-R), \\ \delta\dot{q}(t) &= \frac{N}{R_0} \cdot \delta W(t) - \frac{1}{R_0} \cdot \delta q(t), \end{aligned}\right\} \quad (2)$$

where $\dot{x}$ denotes the time-derivative of x and

$$\delta W \triangleq W - W_o,$$
$$\delta q \triangleq q - q_o,$$
$$\delta p \triangleq p - p_o.$$

Here, operating point (W_o,q_o,p_o) is defined by $\dot{W}=0$, $\dot{q}=0$ (Hollot *et al.*, 2001b), that is

$$W_o = \frac{R_0 C}{N},\ R_0 = \frac{q_o}{C} + T_p.$$

Let $\alpha = \frac{1}{R_0}$, $\beta = \frac{2N}{R_0^2 C}$, $\gamma = \frac{N}{R_0}$, $\eta = -\frac{R_0 C^2}{2N^2}$, $x_1=\delta q$, $x_2=\delta W$, $u=\delta p(t-R_0)$, we get the following matrix equation:

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} -\alpha & \gamma \\ 0 & -\beta \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ \eta \end{pmatrix} u. \quad (3)$$

Let us consider the following Lyapunov function:

$$V(X) = \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} 1 & b \\ b & a \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},\ a - b^2 > 0.$$

It is easy to verify that $V(X)$ satisfies the necessary conditions to enable it to be used as a Lyapunov function. According to the Lyapunov stability theory, if there exists appropriate parameters to make $\dot{V}(X) < 0$, the equilibrium state of the dynamic system is asymptotically stable.

$$\dot{V}(X) = -\begin{pmatrix} x_1 & x_2 \end{pmatrix}\begin{pmatrix} 2\alpha & b(\alpha+\beta)-\gamma \\ b(\alpha+\beta)-\gamma & 2(a\beta-b\gamma) \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + 2(bx_1+ax_2)\eta u. \quad (4)$$

If the following inequality holds

$$a > \left\{\frac{[b(\alpha+\beta)-\gamma]^2}{4\alpha} + b\gamma\right\} \Big/ \beta, \quad (5)$$

the first item on the right hand side of Eq.(4) is negative-definite, meaning that:

$$-\begin{pmatrix} x_1 & x_2 \end{pmatrix}\begin{pmatrix} 2\alpha & b(\alpha+\beta)-\gamma \\ b(\alpha+\beta)-\gamma & 2(a\beta-b\gamma) \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} < 0.$$

Now considering the second item on the right hand side of Eq.(4),

$$2[bx_1(t)+ax_2(t)]\cdot\eta\cdot\delta p(t-R_0), \quad (6)$$

x_1 and x_2 can be rewritten as follows:

$$x_1(t) = x_1(t-R_0) + \int_{t-R_0}^{t} \dot{x}_1 \mathrm{d}\tau,$$

$$x_2(t) = x_2(t-R_0) + \int_{t-R_0}^{t} \dot{x}_2 \mathrm{d}\tau.$$

According to TCP window-based additive increase and multiplicative decrease (AIMD) control scheme, the TCP transmission window size does not change its value within the RTT interval (R_0). So we approximately have

$$\dot{x}_2(t-R_0) = \frac{\Delta x_2(t-R_0)}{R_0} = \frac{x_2(t)-x_2(t-R_0)}{R_0}$$

in this model, i.e., $x_2(t)=x_2(t-R_0)+\dot{x}_2(t-R_0)\cdot R_0$.

From Eq.(1), we know that $\dot{x}_1$ is constant within the RTT interval as long as $\delta W(t)$ does not change its value. So $x_1(t)=x_1(t-R_0)+\dot{x}_1(t-R_0)\cdot R_0$ holds.

Furthermore, according to Eq.(1), we have

$$\frac{R_0\cdot\dot{x}_1(t-R_0)}{N} = x_2(t-R_0),$$

and according to Eq.(2), we have

$$\dot{x}_2(t-R_0) = -\frac{2N}{R^2C}\cdot x_2(t-R_0) - \frac{RC^2}{2N^2}\cdot\delta p(t-2R_0).$$

Rewriting expression (6) as follows:

$$2\left\{b\left[x_1(t-R_0)+\frac{N}{R_0}\cdot x_2(t-R_0)\cdot R_0\right]+a\left[x_2(t-R_0)-\frac{2N}{R_0C}\cdot x_2(t-R_0)-\frac{R_0^2C^2}{2N^2}\cdot\delta p(t-2R_0)\right]\right\}\cdot\eta\cdot\delta p(t-R_0)$$
$$=2\left[b\cdot x_1(t-R_0)+x_2(t-R_0)\cdot\left(bN+a\frac{R_0C-2N}{R_0C}\right)\right]\cdot\eta\cdot\delta p(t-R_0)-\frac{aR_0^2C^2}{N^2}\cdot\delta p(t-2R_0)\cdot\eta\cdot\delta p(t-R_0).$$

Expression (6) can be rewritten as follows:

$$2\left[b\cdot x_1(t-R_0)+x_2(t-R_0)\cdot\left(bN+a\frac{R_0C-2N}{R_0C}\right)-\frac{aR_0^2C^2}{2N^2}\cdot\delta p(t-2R_0)\right]\cdot\eta\cdot\delta p(t-R_0).$$

If the following inequality holds

$$2\left[b\cdot x_1(t-R_0)+x_2(t-R_0)\cdot\left(bN+a\frac{R_0C-2N}{R_0C}\right)-\frac{aR_0^2C^2}{2N^2}\cdot\delta p(t-2R_0)\right]\cdot\eta\cdot\delta p(t-R_0)<0, \quad (7)$$

it can be assured that expression (6)<0. According to Lyapunov stability theory, this means the system is asymptotically stable. So according to inequality (7), the control policy can be chosen as follows:

$$\delta p(t)\begin{cases} >0, & \text{if } \eta X<0, \\ \le 0, & \text{if } \eta X>0, \end{cases}$$

where

$$X = b\cdot x_1(t)+x_2(t)\cdot\left(bN+a\frac{R_0C-2N}{R_0C}\right)-\frac{aR_0^2C^2}{2N^2}\cdot\delta p(t-R_0).$$

Note that $x_2=\delta W=W-RC/N$, $NW/R=flow\_rate$, which means $x_2=\delta W=R(flow\_rate-C)/N$.

For $\delta p(t)=p(t)-p_o$ ($0<p_o, p(t)<1$), we have

$$\delta p(t)\begin{cases} <0, & \text{if } p(t)=0, \\ >0, & \text{if } p(t)=1. \end{cases}$$

This leads to the following result:

$$p(t)=\begin{cases}1, & \text{if } Y>0,\\ 0, & \text{if } Y<0,\end{cases}$$

where

$$Y=b\cdot x_1(t)+x_2(t)\cdot\left(bN+a\frac{R_0C-2N}{R_0C}\right)$$
$$-\frac{aR_0^2C^2}{2N^2}\cdot\delta p(t-R_0),$$

or to say

$$p(t)=\begin{cases}1, & \text{if } Z>0,\\ 0, & \text{if } Z<0,\end{cases}$$

where

$$Z=b\cdot\delta q(t)+\frac{R_0}{N}\cdot\left(bN+a\frac{R_0C-2N}{R_0C}\right)$$
$$\cdot(flow\_rate-C)-\frac{aR_0^2C^2}{2N^2}\cdot\delta p(t-R_0).$$

Letting $a_0=\frac{R_0}{N}\cdot\left(bN+a\frac{R_0C-2N}{R_0C}\right)$, $a_1=\frac{aR_0^2C^2}{2N^2}$, and using the algorithm proposed in (Wang *et al.*, 2004) to compute p_o, our control algorithm can be described as the pseudo-code as shown in Fig.1. From this pseudo-code we can see that TOC algorithm regulates the system to the expected value in nature by appropriately weighting the values of queue length deviation from q_o, rate deviation from C and drop probability deviation from p_o and then deciding whether or not to drop the packet.

```
if (now>last_time+sample_time) {
   if (b(q_length−qo)+a0(estimate_flow_rate−C)
      −a1(last_drop_prob[n−1]−po)>0)
         drop_prob[n]=1;
   else
         drop_prob[n]=0;
   last_time=now;
}
else
   return;
```

Fig.1 Pseudo-code of TOC algorithm

To compute a_0, a_1, we set b at 30, then according to Eq.(5),

$$a=1+\left\{\frac{[b(\alpha+\beta)-\gamma]^2}{4\alpha}+b\gamma\right\}\Big/\beta.$$

So a_0, a_1 can be easily computed. Note here that R_0=(q_0/C)+*average_propagation_delay*. To avoid dropping too many packets (which will result in the system vibrating sharply), *last_drop_prob*[n−1] should be set at the largest value in the last RTT interval. Through experiments we find this is a useful method to improve our controller's performance.

Now we explain why we call this algorithm time optimal controller (TOC). According to control theory analysis, $|\dot{V}(X,u)|$, the absolute value of Lyapunov function's time derivative along the trajectories of the system approximately reflects how fast this system converges to equilibrium state. If we choose $\dot{V}(X,u_o)=\min_{u\in U}\dot{V}(X,u)$, the system will achieve the desired reference value at the shortest regulating time (Wang, 1980; Lupfer and Berger, 1959). So this is a fast-response controller.

PERFORMANCE EVALUATION

We evaluate the effectiveness and performance of the TOC algorithm by simulations using ns-2.1b9 simulator. The network topology is a simple dumbbell topology based on a single common bottleneck link of 10 Mbps capacity with many identical, long-lived and saturated TCP/Reno flows. The round-trip propagation delay is randomly chosen from 1 ms to 20 ms.

Experiment 1 In this experiment, bottleneck link was 10 Mbps, target queue length q_o was 80 packets, and buffer size for AQM was 300 packets, average packet size was 1040 bytes. There were 100 FTP connected all together.

The performance of TOC was compared with that of PI controller. The parameters used for the controller were as follows: for PI scheme, the parameters were set the same as that in (Hollot *et al.*, 2001a), that is, a=1.822e-5 and b=1.816e-5 respectively. For TOC, the parameters were set at b=30, a_0=1.147, a_1=670.4188. Fig.2 shows the dynamic change of the real queue length of the PI and TOC algorithms respectively. It can be seen that TOC shows higher performance than PI.

TOC settled down to the reference point within 3 s, while PI controller took more than 7 s to settle down to the reference point. Note how the overshoot and oscillation were essentially eliminated when we use

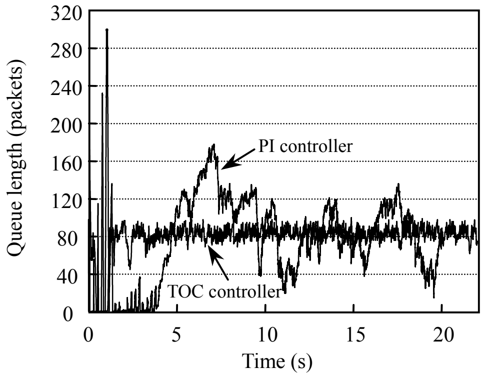

Fig.2 Queue evolution in Experiment 1

optimal controller. All together, the system has become more stable than the system using PI controller.

Experiment 2 In this experiment the bottleneck link was 10 Mbps. There were 150 FTP connected, and q_o was 80 packets, buffer size was 300 packets. PI controller's parameter setting was the same as that in Experiment 1, and parameters of TOC were set at b=30, a_0=0.7050, a_1=308.0166.

Fig.3 shows the experiment result. The queue length evolution of PI controller shows it varies sharply. On the other hand, TOC controller regulates the queue length to the equilibrium point more rapidly and more stably.

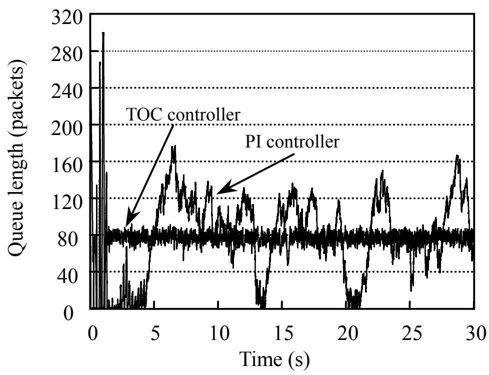

Fig.3 Queue evolution in Experiment 2

Experiment 3 To evaluate TOC controller performance at different target queue length and R_0 settings, in this experiment we set q_o to 150, 100 and 50 each, with 150 FTP connections. Note here that different target queue length will result in different R_0's.

Fig.4 shows the result of this experiment. It can be seen that all queue length were stabilized at target value with short response time. This proves that TOC can force the system to rapidly achieve the desired reference value, and reduce the overshoot and oscillation evidently.

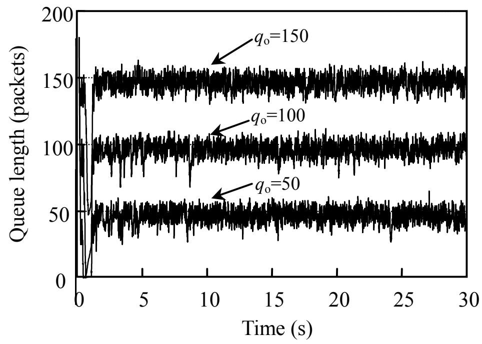

Fig.4 Queue evolution in Experiment 3

Experiment 4 In this experiment we tested the robustness and capability of resisting model mismatches. At 0 second, there were 80 FTP connections to start, at 10 second there were another 80 FTP connections to start, at 20 second the other 80 FTP connections started, and at 30 second the first group connections stopped, at 45 second the second group stopped. The target queue length was 100 packets.

We compared the performance of TOC with PI controller. Parameters of these two controllers were set the same as those in Experiment 1. Result of Experiment 4 is plotted in Fig.5. It shows evidently that queue evolution under PI controller oscillates at large magnitude, while TOC algorithm operates in a relatively stable state. We can draw a conclusion from this experiment that TOC is more robust than PI algorithm and can resist very well the model mismatches.

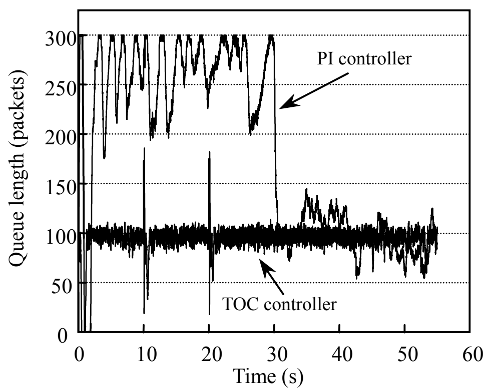

Fig.5 Queue evolution in Experiment 4

CONCLUSION

In this work we focus on improving the TCP/AQM performance in stability and response time in the router. In fact stability and response time are often in conflict with each other in system performance. To find a tradeoff between them, we propose to regulate the system with time optimal controller (TOC) designed through Lyapunov stability theory. We made a complete comparison with PI controller under various scenarios. Results of extensive simulations by ns-2 showed that the overall performance of the optimal controller is superior to that of the PI controller. TOC is responsive, stable and robust.

References

Athuraliya, S., Low, S.H., Li, V.H., Yin, Q., 2001. REM: active queue management. *IEEE Network*, **15**(3):48-53. [doi:10.1109/65.923940]

Fan, Y.F., Ren, F.Y., Lin, C., 2003. Design a PID Controller for Active Queue Management. Proc. ISCC, p.985-990.

Floyd, S., Jacobson, V., 1993. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, **1**(4):397-413. [doi:10.1109/90.251892]

Floyd, S., Gummadi, R., Shenker, S., 1997. Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management. Http://www.icir.org/floyd

Floyd, S., 2000. Recommendations on Using the Gentle "Variant of RED". Http://www.aciri.org/floyd/red/gentle.html

Hollot, C., Misra, V., Towsley, D., Gong, W.B., 2001a. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. Proc. IEEE INFOCOM. Anchorage, Alaska, USA, p.1726-1734.

Hollot, C., Misra, V., Towsley, D., Gong, W.B., 2001b. A Control Theoretic Analysis of RED. Proc. IEEE INFOCOM. Anchorage, Alaska, USA, p.1510-1519.

Kunniyur, S., Srikant, R., 2001. Analysis and design of an adaptive virtual queue algorithm for active queue management. *ACM Computer Communication Review*, **31**(4): 123-134. [doi:10.1145/964723.383069]

Lupfer, D.E., Berger, D.E., 1959. Computer control of distillation reflux. *ISA Journal*, p.34-39.

May, M., Bolot, J., Diot, C., Lyles, B., 1999. Reasons Not to Deploy RED. Proc. IWQoS. London, UK, p.260-262.

Misra, V., Gong, W.B., Towsley, D., 2000. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. *ACM Computer Communication Review*, **30**(4):151-160. [doi:10.1145/347057.347421]

Ott, T., Lakshman, T., Wong, L., 1999. SRED: Stabilized RED. Proc. IEEE INFOCOM. New York, USA, p.1346-1355.

Srikant, R., 2004. The Mathematics of Internet Congestion Control. Birkhauser, p.40-47.

Sun, J., Ko, K., Chen, G., Chan, S., Zukerman, M., 2003a. PD-RED: to improve the performance of RED. *IEEE Communications Letters*, **7**(8):406-408. [doi:10.1109/LCOMM.2003.815653]

Sun, J., Chen, G., Ko, K., Chan, S., Zukerman, M., 2003b. PD-Controller: A New Active Queue Management Scheme. Proc. IEEE GLOBECOM. San Francisco, USA, **22**(1):3103-3107.

Wang, Y.C., 1980. Foundation of Design Optimal Controller System. Science Publishing House, Beijing, p.434-437 (in Chinese).

Wang, C., Bin, L., Hou, Y.T., Sohraby, K., Yu, L., 2004. LRED: A Robust Active Queue Management Scheme Based on Packet Loss Ratio. Proc. IEEE INFOCOM, p.1-12.