INT J COMPUT COMMUN, ISSN 1841-9836
8(6):838-844, December, 2013.

Active Queue Management of TCP Flows with Self-scheduled Linear Parameter Varying Controllers

C. Kasnakoglu

Cosku Kasnakoglu
TOBB University of Economics and Technology
Department of Electrical and Electronics Engineering
Ankara, Turkey, 06560
kasnakoglu@etu.edu.tr

Abstract:
Control-theoretic approaches to Active Queue Management (AQM) are typically based on linearizations of fluid flow models around design conditions. These conditions depend on the Round Trip Time (RTT), and the AQM performance is known to degrade if RTT values during actual operation depart substantially from design values. To overcome this difficulty a self-scheduled LPV controller for AQM is considered in this paper, where the controller is modified in real-time based on RTT. Simulations show that the self-scheduled LPV controller has good performance for both constant and time-varying RTTs, and outperforms two other common control-theoretic approaches to AQM.
Keywords: AQM, congestion control, control theory, fluid model, LPV systems, self-scheduled controller

1 Introduction

Congestion is one of the most important problems faced in communications networks. Congestion occurs when a link or node is carrying so much data that its quality of service deteriorates. This results in queueing delay, packet loss or the blocking of new connections, leading to low throughput and eventually to congestion collapse. On the other hand, links carrying less data than a certain level are also not desired as this implies that the link capacities are being underutilized [1]. Modern networks try to avoid these situations using congestion control techniques, among which Active Queue Management (AQM) is of particular interest. AQM operates by dropping or ECN-marking packets before the queue is full, according to a probabilistic rule. Earlier AQM disciplines such as RED [2] and REM [3] required careful tuning of parameters in order to provide good performance, while modern AQM disciplines such as ARED [4] and Blue [5] are self-tuning. With the development of dynamical models such as the fluid flow model in [6], control theoretic approaches for AQM have gained interest, including PI/PID controllers [7, 8] and robust $\mathcal{H}_\infty$ controllers [9–11]. These are based on the linearization of the fluid model, which produces a transfer function from probability of package mark p to queue length q. One difficulty is that the transfer function is valid only for a given Round Trip Time (RTT), and a new transfer function must be obtained for a different RTT. In [12] switching between multiple controllers designed for different RTTs is considered and it is seen that higher number of controllers results in improved performance. However, if the number of controllers is too high, the implementation becomes very difficult and complicated. In addition, since RTT takes values on an interval, designing a controller for each RTT value requires an infinite number of controllers. In this paper a self-scheduled control design for AQM is considered to overcome these difficulties. The controller is parameterized in RTT and achieves stability and small tracking error for both fixed RTT as well as time varying RTT within a prescribed range.

Copyright © 2006-2013 by CCC Publications

2 Mathematical Model of TCP/AQM

A dynamical model for TCP congestion control was developed in [6] using fluid flow approximation. The dynamical model represents a bottleneck with multiple TCP flows sharing the link. The congestion avoidance is modeled as AIMD (additive-increase multiplicative-decrease). The dynamical model consists of the following nonlinear time-delayed differential equations

$$\dot{W}(t) = \frac{1}{\theta(t)} - \frac{W(t)}{2}\frac{W(t-\theta(t))}{\theta(t-\theta(t))}p(t-\theta(t)) \tag{1}$$

$$\dot{q}(t) = \frac{N(t)}{\theta(t)}W(t) - C(t) \quad \text{for } q(t) > 0 \tag{2}$$

where W is the TCP window size, q is the queue length, N is the number of TCP flows, C is the link capacity, p is the probability of packet mark and θ is the RTT. Let C and N be constants for simplicity. The nonlinear system (1)-(2) can be linearized around an operating point to generate a transfer function from p to q. Let $\delta p := p(t) - p_o$ and $\delta q := q(t) - q_o$ where q_o and p_o are the values at the operating point. Then a transfer function from δp to δq can be obtained as [6]

$$\frac{\delta Q(s)}{\delta P(s)} = \frac{(C^3\theta^3 N)e^{-\theta s}}{2N^2\theta^3 C s^2 + (2N^2 C\theta^2 + 4N^3\theta)s + 4N^3} \tag{3}$$

where θ is regarded as a parameter. One then has a different transfer function for each value of the RTT in the range $[\theta_{\min}, \theta_{\max}]$, where $\theta_{\min}$ and $\theta_{\max}$ are the minimum and maximum values of RTT for the link.

3 Controller Design

The controller must achieve stability of the closed-loop system and the tracking of a desired queue length, under the presence of constant or time-varying RTT within the specified range.[1] A parameter dependent dynamic controller of the following form will be sought

$$\begin{aligned} \dot{\zeta} &= A_K(\theta)\zeta + B_K(\theta)e \\ \delta p &= C_K(\theta)\zeta + D_K(\theta)e \end{aligned} \tag{4}$$

where ζ is the controller's internal state and $e := \delta q_d - \delta q$ is the error between the desired and the actual queue sizes. A systematic method for the selection of the controller matrices A_K, B_K, C_K and D_K in (4) was derived in [13]. For this purpose the system to be controlled, i.e. (3), must be transformed into affine-parameter dependent form. Approximating the time delay $e^{-\theta s}$ in (3) with a second order Pade approximation

$$e^{-\theta s} \approx \frac{12 - 6\,\theta\, s + \theta^2 s^2}{12 + 6\,\theta\, s + \theta^2 s^2} \tag{5}$$

and transforming into state space form yields

$$\dot{x} = A_P(\theta)x + B_P(\theta)\delta p \tag{6}$$

$$\delta q = C_P(\theta)x \tag{7}$$

[1] Under ideal operation, RTT will be small for small queue size. However no such assumption will be made here, since unexpected changes in link conditions (e.g. increased propagation delay) can create longer RTTs even at smaller queue lengths. Thus, the controller is expected to be prepared for any RTT at any queue size (within the limits allowed).

where

$$A_P(\theta) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -a_3 & -a_2 & -a_1 & -a_0 \end{bmatrix}, \tag{8}$$

$$B_P(\theta) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T, \tag{9}$$

$$C_P(\theta) = \begin{bmatrix} b_2 & b_1 & b_0 & 0 \end{bmatrix} \tag{10}$$

and

$$\begin{aligned} a_0 =& 7\theta^{-1} + \tfrac{2N}{C}\theta^{-2},\ a_1 = \tfrac{14N}{C}\theta^{-3} + 18\theta^{-2}, \\ a_2 =& 12\theta^{-3} + \tfrac{36N}{C}\theta^{-4},\ a_3 = \tfrac{24N}{C}\theta^{-5}, \\ b_0 =& \tfrac{C^2}{2N},\ b_1 = \tfrac{-3C^2}{N}\theta^{-1},\ b_2 = \tfrac{6C^2}{N}\theta^{-2}\ . \end{aligned}$$

Letting $\Theta := \theta^{-1}$, and performing a series expansion around point $\Theta = \Theta_o$, one can write A_P, B_P, C_P in terms of Θ as

$$A_P(\Theta) = A_{P0} + A_{P1}\Theta + \mathcal{O}(\Theta^2) \tag{11}$$

$$B_P(\Theta) = B_{P0} + B_{P1}\Theta + \mathcal{O}(\Theta^2) \tag{12}$$

$$C_P(\Theta) = C_{P0} + C_{P1}\Theta + \mathcal{O}(\Theta^2) \tag{13}$$

where A_{P0} has the same structure as in (8) but with

$$\begin{aligned} a_0 =& \tfrac{-2N\Theta_o{}^2}{C},\ a_1 = \tfrac{-28N\Theta_o{}^3}{C} - 18\Theta_o{}^2, \\ a_2 =& \tfrac{-108N\Theta_o{}^4}{C} - 24\Theta_o{}^3,\ a_3 = \tfrac{-96N\Theta_o{}^5}{C}, \end{aligned}$$

A_{P1} has the same structure as in (8) but with

$$\begin{aligned} a_0 =& \tfrac{4N\Theta_o}{C} + 7,\ a_1 = \tfrac{42N\Theta_o{}^2}{C} + 36\,\Theta_o, \\ a_2 =& \tfrac{144N\Theta_o{}^3}{C} + 36\,\Theta_o{}^2,\ a_3 = \tfrac{120N\Theta_o{}^4}{C}, \end{aligned}$$

B_{P0} is equal to B_P in (9), B_{P1} is zero, C_{P0} has the same structure as in (10) but with

$$b_0 = \tfrac{C^2}{2N},\ b_1 = 0,\ b_2 = \tfrac{-6C^2\Theta_o{}^2}{N}$$

and C_{P1} has the same structure as in (10) but with

$$b_0 = 0,\ b_1 = \tfrac{-3C^2}{N},\ b_2 = \tfrac{12C^2\Theta_o}{N}\ .$$

Approximating A_P, B_P and C_P with the constant and linear terms in (11)-(13) produces an affine parameter dependent state space system (in Θ) as desired. Self-scheduled controller design methods [13] can then be used to design the parameter dependent controller in (4) to meet the desired control objectives. The performance objective is to have a fast and well-damped response over the entire range of parameter values for step-like references. The control law is also required to achieve robustness by avoiding high-gain feedback at high-frequencies. This will prevent the excitation of high frequency modes and nonlinearities that were unmodeled or neglected.

Consider the feedback control structure depicted in Figure 1. The control problem described

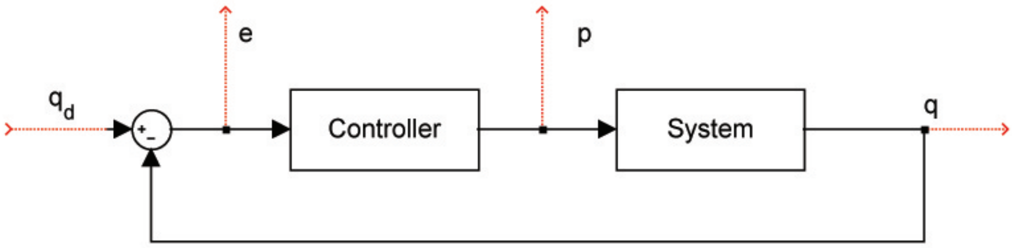

Figure 1: Feedback control structure for AQM

above can be formulated as the minimization of the $\mathcal{L}_2$-induced norm of the operator mapping the signal δq_d to signals e and δp. The former map is the called the sensitivity function and denoted by S, and the latter map is denoted by KS. Following standard $\mathcal{H}_\infty$ design procedure, the performance objective and robustness objectives are specified through weighting filters $W_1(s)$ and $W_2(s)$, where $W_1(s) = 1/s$ for good tracking of step-like references, and $W_2(s)$ is a third order high pass Butterworth filter. This yields to an $\mathcal{H}_\infty$ optimization problem where the goal is to find a stabilizing controller K for which the inequality

$$\left\| \begin{bmatrix} W_1(s)S(s) \\ W_2(s)K(s)S(s) \end{bmatrix} \right\|_\infty \leq \gamma \tag{14}$$

can be satisfied with γ in $\mathbb{R}_+$ as small as possible. Let A_{cl}, B_{cl}, C_{cl} and D_{cl} be the state-space matrices corresponsing to the one-input two-output system defined by the transfer function matrix $[W_1S,\ W_2KS]^T$. An LPV controller satisfying (14) can be synthesized and implemented as follows [13]

1. Find a matrix $X_{cl} > 0$, and controller matrices A_{Ki}, B_{Ki}, C_{Ki}, D_{Ki} so that

$$\begin{bmatrix} A_{cl}(\Theta_i)^T X_{cl} + X_{cl}A_{cl}(\Theta_i) & X_{cl}B_{cl}(\Theta_i) & C_{cl}^T(\Theta_i) \\ B_{cl}^T(\Theta_i)X_{cl} & -\gamma I & D_{cl}^T(\Theta_i) \\ C_{cl}(\Theta_i) & D_{cl}(\Theta_i) & -\gamma I \end{bmatrix}$$

   is negative-definite for $i = 1, 2$, where $\Theta_1 = \theta_{\min}^{-1}$ and $\Theta_2 = \theta_{\max}^{-1}$.

2. For a given value of $\Theta = \theta^{-1}$, compute the matrices $A_K(\Theta)$, $B_K(\Theta)$, $C_K(\Theta)$ and $D_K(\Theta)$ defining the LPV controller as

$$\begin{bmatrix} A_K(\Theta) & B_K(\Theta) \\ C_K(\Theta) & D_K(\Theta) \end{bmatrix} = \sum_{i=1}^{2} \alpha_i(\Theta) \begin{bmatrix} A_{Ki} & B_{Ki} \\ C_{Ki} & D_{Ki} \end{bmatrix}$$

   where (α_1, α_2) is a convex decomposition of Θ such that $\Theta = \alpha_1\Theta_1 + \alpha_2\Theta_2$ and $\alpha_1 + \alpha_2 = 1$.

4 Simulation Results

The performance of the closed-loop system with the LPV controller is tested using MATLAB/Simulink and discrete event simulations.[2] The number of TCP flows is taken to be $N = 150$, the link capacity $C = 500$ packets/sec, the desired queue size $q_o = 150$ packets, and the buffer

[2]The results from both were consistent so only those from the former are shown for better presentation and to save space.

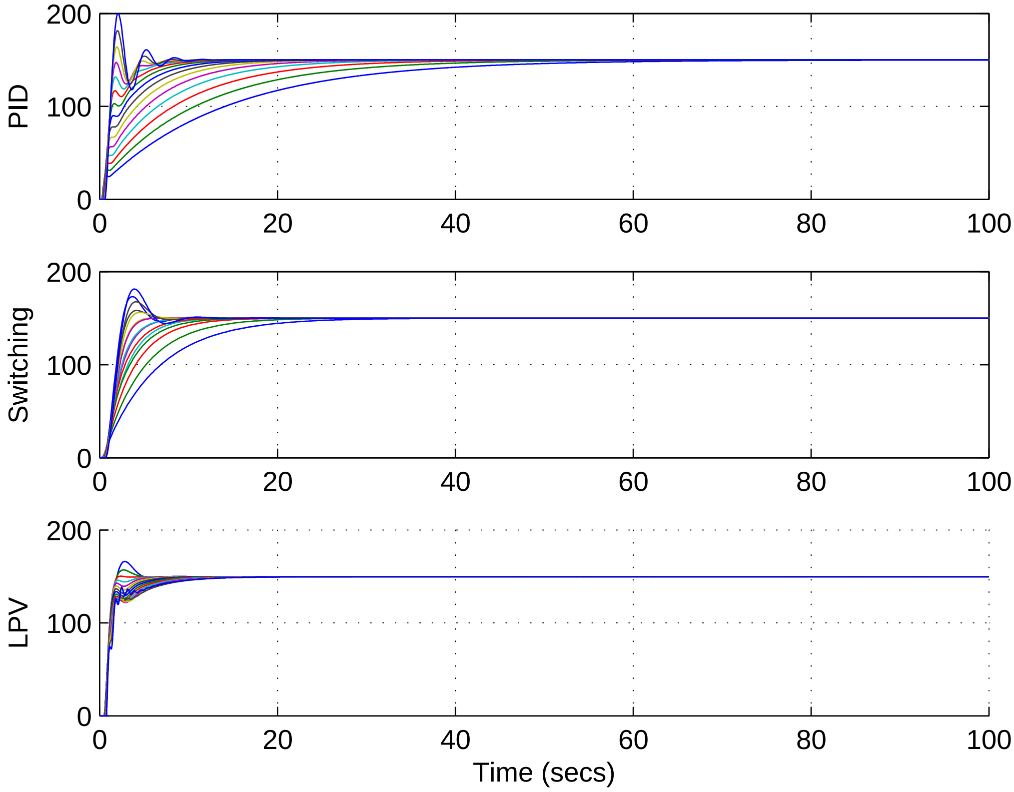

Figure 2: Simulation results for AQM based on PID (top), switching $\mathcal{H}_\infty$ (middle) and LPV controllers (bottom), for 15 fixed values of RTT.

size $q_{\max} = 200$ packets. The RTT is assumed to take values between 0.3 secs and 0.7 secs. The controller design as outlined in Section 3 is carried out with the help of MATLAB Robust Control Toolbox.

Figure 2 shows the simulation results for 15 fixed values of RTT, linearly equally spaced between 0.3 and 0.7. Two other common control-theoretic approaches to AQM design are also implemented for comparison: a PID controller, and a switched $\mathcal{H}_\infty$ controller. After subsequent trials, the best results that could be obtained with the PID controller were those based on a linearization around RTT $= 0.6$. The switched $\mathcal{H}_\infty$ controller is based on two operating points RTT_1 and RTT_2, the best results for which were obtained when $\text{RTT}_1 = 0.4320$ and $\text{RTT}_2 = 0.6080$. The simulations illustrate that all controllers eventually succeed in achieving and maintaining the desired queue size of 150 packets. However, the LPV controller response is faster, with less overshoot and better damping over the entire parameter range.

The controllers were also tested for the case when RTT is time varying. The PID controller and the switched controller did not produce a stable closed-loop so these responses are not shown. The result for the LPV controller is shown in Figure 3. The top subfigure shows the variation in RTT versus time, which is a sinusoid with increasing frequency taking values between 0.3 and 0.7.[3] The bottom subfigure shows the response of the closed-loop system. It can be observed that the LPV controller produces a stable closed-loop and is capable of maintaining the queue size very close to the desired value, even when RTT is time varying.

[3]Different cases for the variation of RTT were also tested with successful results, including square, triangle and sawtooth waveforms, as well as the case where RTT varies randomly according to various probability densities. However, only one case is shown in the paper to save space.

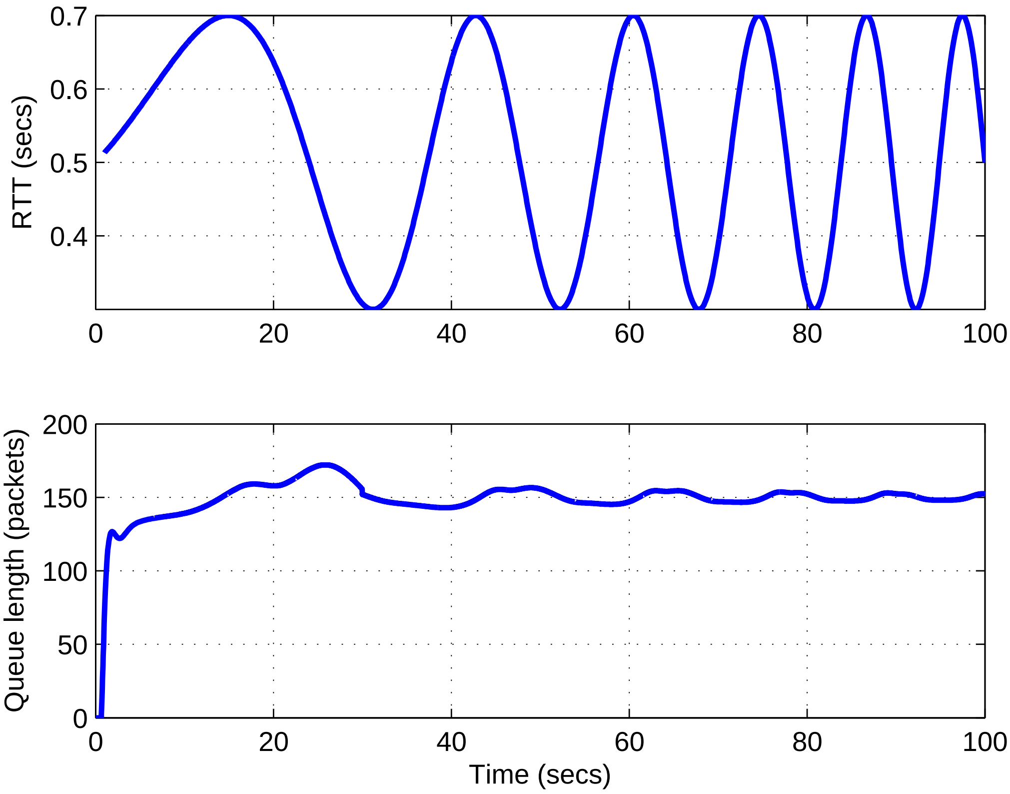

Figure 3: Simulation results for AQM based on LPV, for a time-varying RTT.

5 Conclusions

In this paper self-scheduled LPV control design was implemented for AQM. The controller design is based on a fluid flow approximation of TCP congestion control and utilizes RTT as the scheduling parameter. Simulations "were carried out to evaluate the closed-loop system in its ability to keep the queue length at a desired level. Two other control-theoretic approaches to AQM, namely PID control and switching $\mathcal{H}_\infty$ control, were also implemented for comparison. When RTT is constant, it was seen that the LPV controller outperforms the other two, producing a faster and better damped response with lesser overshoot over the entire parameter range. When RTT is time varying, the LPV controller is still capable of maintaining the desired queue size, whereas the PID and switching controllers do not produce a stable closed-loop.

Bibliography

[1] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Transactions on Networking*, 7(4):458–472, 1999.

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

[3] S. Athuraliya, SH Low, and VH Li. REM: active queue management. *IEEE Network*, 15(3):48–53, 2001.

[4] S. Floyd, R. Gummadi, and S. Shenker. Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management. *Preprint, http://www. icir. org/floyd/papers.html, August*, 2001.

[5] W. Feng, K.G. Shin, D.D. Kandlur, and D. Saha. The BLUE active queue management algorithms. *IEEE/ACM Transactions on Networking*, 10(4):513–528, 2002.

[6] V. Misra, W.B. Gong, and D. Towsley. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. In *Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 151–160. ACM Press New York, NY, USA, 2000.

[7] CV Hollot, V. Misra, and D. Towsley. Analysis and design of controllers for AQM routers supporting TCPflows. *IEEE Transactions on Automatic Control*, 47(6):945–959, 2002.

[8] D. Ustebay, H. Ozbay, and N. Gundes. A New PI and PID Control Design Method for Integrating Systems with Time Delays: Applications to AQM of TCP Flows. *Wseas Transactions On Systems And Control*, 2(2):117, 2007.

[9] P. Yan and H. Ozbay. Robust Controller Design for AQM and $\mathcal{H}_\infty$-Performance Analysis. *Advances In Communication Control Networks*, 6:7, 2005.

[10] L. Yu, M. Ma, W. Hu, Z. Shi, and Y. Shu. Design of parameter tunable robust controller for active queue management based on H-infinity control theory. *Journal of network and computer applications*, 34(2):750–764, 2010.

[11] R. Vilanova and V.M. Alfaro. Robust 2-DoF PID control for Congestion control of TCP/IP Networks. *Int J Comput Commun*,ISSN 1841-9836, 5(5):968–975, 2010.

[12] D. Ustebay and H. Ozbay. Switching Resilient PI Controllers for Active Queue Management of TCP Flows. In *IEEE International Conference on Networking, Sensing and Control,*, pages 574–578, 2007.

[13] P. Apkarian, P. Gahinet, and G. Becker. Self-scheduled H-infinity Control of Linear Parametervarying Systems: a Design Example. *Automatica*, 31(9):1251–1261, 1995.