Robust Active Queue Management for Wireless Networks

Kanchan Chavan, Ram G. Kumar, Madhu N. Belur, and Abhay Karandikar

***Abstract*—Active Queue Management (AQM) algorithms have been extensively studied in the literature in the context of wired networks. In this paper, we study AQM for wireless networks. Unlike a wired link, which is assumed to have a fixed capacity, a wireless link has a capacity that is time-varying due to fading. Thus, the controller is required to meet performance objectives in the presence of these capacity variations. We propose a robust controller design that maintains the queue length close to an operating point. We treat capacity variations as an external disturbance and design a robust controller using $\mathcal{H}_\infty$ control techniques. We also consider the effect of round-trip time in our model. Our method of incorporating the delay into the discretized model simplifies controller design by allowing direct use of systematic controller design methods and/or design packages. We demonstrate the robustness of the controller to changes in the load condition and in the round-trip time through $ns-2$ simulations.**

***Index Terms*—Congestion control, discretization, $\mathcal{H}_2$ control, $\mathcal{H}_\infty$ control, robust control, wireless network.**

I. Introduction

Active queue management (AQM) [1] plays an important role in controlling congestion in packet switched networks, such as Transmission Control Protocol/Internet Protocol (TCP/IP) networks. An AQM algorithm comprises of two basic components: the first one monitors the fluctuations in queue length, while the other conveys any incipient congestion by dropping a packet (or marking it dropped) with a calculated probability. The TCP congestion control mechanism responds to this feedback by adaptively modifying its window size, thereby reducing its sending rate. Various AQM techniques differ in the way they perform queue measurements for detecting congestion and in the mechanism for packet drop.

Random Early Detection (RED) [1] was one of the earliest algorithms for AQM. In RED, when the average queue length varies between a prespecified minimum threshold $(\min_{\rm th})$ and a maximum threshold $(\max_{\rm th})$, the probability of drop is varied linearly with the queue length from zero to a maximum packet drop probability $(\max_{\rm p})$; for queue lengths greater than $\max_{\rm th}$, the drop probability is set to 1. The "gentle" option in RED ensures a smoother increase of drop probability from $\max_{\rm p}$ to 1 (for queue lengths greater than $\max_{\rm th}$). These parameters and thresholds, in addition to the options gentle, wait, etc., require setting and tuning: this is one of the drawbacks of RED, since considerable effort in tuning its parameters is required to achieve good performance [2]. A careful adaptive tuning of the parameters has been proposed in [3], resulting in "Adaptive RED" achieving significantly better queue control than that of RED. In addition to Adaptive RED, the development of RED sparked a great deal of interest in other modifications of RED that addressed similar shortcomings of RED; see, for example, [4]–[8].

It was subsequently recognized that TCP with an AQM router can be considered as a feedback control system. Thus, control theoretic models of AQM have been developed in [9]–[12], among others, and these models have helped in a better understanding of router queue dynamics in TCP networks. This has resulted in systematic and scalable techniques for controller design. See [13], [14] and the references therein for a detailed overview and analysis.

While these papers have provided significant insight into AQM, most of them deal with the case of wired networks. In this paper, we study AQM for the case of wireless links. Unlike a wired link, which is assumed to have a fixed capacity, a wireless link has a capacity that is time-varying due to fading [15]. Thus, the controller is required to meet performance objectives in the presence of these capacity variations.

AQM for wireless networks has been studied recently in [16]. Capacity variations of the wireless link has been considered as a disturbance, and an $\mathcal{H}_\infty$ control technique has been used for controller design to achieve disturbance attenuation. Our approach differs in the way the delay is handled during controller design (this is elaborated below in Section II). As indicated by literature on AQM techniques, one of the important challenges posed by AQM of both wired and wireless networks is the delay term in the differential equations. Due to the consequent ease in controller design, many popular approaches resort to ignoring the delay and designing a controller. Noting that the packet drop/mark probability has its effect on the TCP sources only after the round-trip time, modifying the model by ignoring the delay results in the modified model, suggesting an exaggerated ability to control the queue length using drop probability for control. The importance of considering the time delay has also been noted in [17]–[19].

While it is important to design a controller that meets performance objectives *without* ignoring the delay that is inevitable in the effect of drop probability on the window size of TCP sources, one would also like that controller design is possible using standard controller design packages like SCILAB/MATLAB for quick simulations and fine-tuning of parameters, if required. Our approach addresses this requirement that standard controller design packages can be used for the design of a controller for various performance objectives. In this paper,

Manuscript received September 21, 2009; revised May 10, 2010; accepted September 14, 2010. Manuscript received in final form September 22, 2010. Date of publication November 01, 2010; date of current version September 16, 2011. This work was supported in part by the Tata Teleservices-IIT Bombay Center for Excellence in Telecom and by SERC Division, DST. A preliminary version of this was presented at the National Conference on Communications, January 2009, Guwahati, India. Recommended by Associate Editor Bram de Jager.

The authors are with the Department of Electrical Engineering, Indian Institute of Technology Bombay, Powai Mumbai 400076, India (e-mail: belur@ee.iitb.ac.in).

Digital Object Identifier 10.1109/TCST.2010.2081671

we assume that the *nominal* to-be-controlled system has a constant round-trip time. This constant time delay within the system model is incorporated in a discretized model of the continuous-time TCP network system (see Section II). The actual system that can have varying round-trip time values can be controlled using a *robust* controller to meet any of various performance objectives like linear quadratic regulator (LQR), linear quadratic Gaussian (LQG), $\mathcal{H}_2$, $\mathcal{H}_\infty$, mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control, or just pole placement. In this paper, we use the $\mathcal{H}_2$ and $\mathcal{H}_\infty$ controllers for queue stabilization with capacity variations treated as a disturbance. In this sense, the AQM technique proposed in this paper can be termed as Robust Queue Management (RQM).

In [16], the solution to the $\mathcal{H}_\infty$ control of the (linearized) time-delay system obtained there requires the solution of a matrix inequality that is not a *linear* matrix inequality (LMI). Consequently, as pointed out there, a search/sweeping process is required to get a clue of suitable parameters (using the heuristic methods described in [20]), after which an LMI problem is to be solved to find a controller that meets the performance objective there ($\mathcal{H}_\infty$-optimization). In contrast to [16] and other papers that do not ignore the delay, our approach simplifies controller design.

While we address the effect on the percentage of packets dropped due to the proposed RQM controller, we focus on queue control, i.e., keeping queue fluctuations about a specified queue length sufficiently small. In addition to better utilization of queue buffer, minimizing queue fluctuations also makes the delay of packets within the queue more predictable and controllable: this helps in the Quality of Service (QoS) guarantee for many Internet applications.

The remainder of this paper is organized as follows. We study the system model in Section II. The control problem and the performance objective of the controller are stated in Section III. Section IV contains simulation results of the proposed RQM controller and comparison with RED and Adaptive RED performance. A few conclusive remarks follow in Section V.

II. System Model and Problem Formulation

Here, we first describe the fluid flow model of TCP window and router dynamics in a wireless network.

A. Fluid Flow Model of TCP for Wireless Network

We model the TCP network by a simple topology as shown in Fig. 1, where TCP sources send data towards their destinations in Internet through a router which represents the bottleneck node in the network. With a receipt-of-acknowledgement packet, each source increases its rate of transmission, and this eventually causes the outgoing link capacity of the bottleneck node to be exceeded. Upon detection of a packet drop, a TCP source reduces its window size to half. In this way, each source attempts to determine the available capacity in the network. We consider the fluid flow model of TCP behavior as proposed in the classical paper [11]. This model relates window and queue dynamics with a capacity of the router outgoing link and the round-trip time (RTT). Note that RTT is the time required for

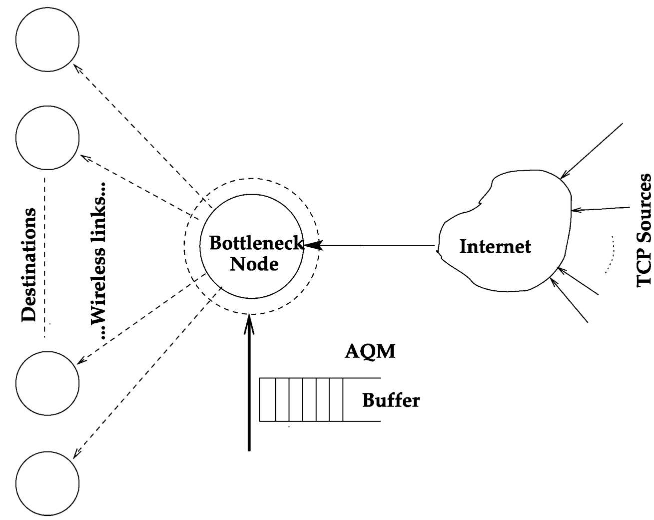

Fig. 1. TCP network model.

the packet to reach the destination plus the time it takes for acknowledgement to reach the sender. Hence, it comprises transmission delay, queueing delay, and propagation delay. Let T_p denote the sum of propagation and transmission delays (in seconds), and q and C denote the queue length (in packets) and the link capacity (in packets/second) respectively. Then, the RTT $R = (q/C) + T_p$.

Though originally proposed for wired networks, we use the fluid flow model for the wireless scenario where we assume that the capacity of the bottleneck node is time-varying. With these assumptions, we can write the fluid flow model in terms of W: the TCP window size (in packets); N: the load factor (number of TCP sources); and p: the probability of packet drop, and q, T_p, C, defined above as

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

$$\dot{q}(t) = \frac{W(t)}{R(t)}N(t) - C(t). \tag{2}$$

The above model captures the standard Additive Increase Multiplicative Decrease phenomenon in typical TCP/IP networks: the first term on the right-hand side of (1) corresponds to additive increase of the window size after each round-trip time (i.e., with receipt of an acknowledgement packet), while the second term corresponds to multiplicative decrease in the window size due to packet drop. The queue length and window size take their values in [0, $\bar{q}$] and [0, $\bar{W}$], where $\bar{q}$ and $\bar{W}$ denote buffer capacity and maximum window size. The packet drop probability p takes values in [0, 1].

Using the above nonlinear model of TCP dynamics, we obtain a linearized model about an equilibrium point. The time-varying capacity affects the rate at which packets depart on the router link. This in turn affects the queue length at the router buffer. Accordingly, we consider this capacity variation also during the process of linearization.

Suppose $W_0, C_0, R_0, p_0, q_0, T_p$, and N have values such that $\dot{W}(t)$ and $\dot{q}(t)$ are zero, then W and q will remain unchanged according to (1) and (2). We call these values an equilibrium

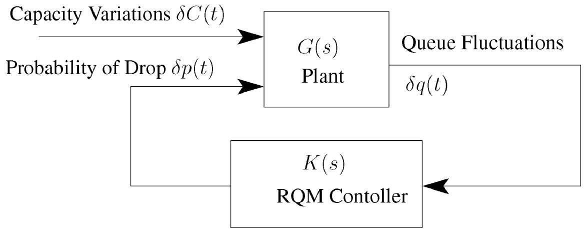

Fig. 2. AQM controller model for a wireless network.

point. As in [11], the equations that have to be satisfied by the variables at an equilibrium point are

$$W_0^2 p_0 = 2 \quad W_0 N = R_0 C_0 \quad R_0 = \frac{q_0}{C_0} + T_p. \tag{3}$$

For the purpose of controller design, we obtain a *nominal* to-be-controlled system's linearized model, for which we assume the round-trip time to be constant: R_0. The designed controller is robust with respect to varying time delays and meets the performance objectives. This is demonstrated below in our simulation results. If the variables W, p, N, q, and C have values satisfying the above equations (constituting the requirement for being an equilibrium point), then these variables remain constant with time. Consider deviations of variables W, q, p, and C about such an equilibrium point; let δW, δq, and δC, respectively, denote these deviations, i.e., $\delta W := W - W_0$, $\delta q := q - q_0$, $\delta p := p - p_0$, and $\delta C := C - C_0$. We obtain the following linearized model for the dynamical system:

$$\begin{aligned} \delta \dot{W}(t) &= \frac{-NC_0}{(q_0 + C_0 T_p)^2}\left(\delta W(t) + \delta W(t - R_0)\right) \\ &\quad - \frac{C_0(q_0 + C_0 T_p)}{2N^2}\delta p(t - R_0) \\ \delta \dot{q}(t) &= \frac{C_0 N \delta W(t) - C_0 \delta q(t) + q_0 \delta C(t)}{q_0 + C_0 T_p}. \end{aligned}$$

In contrast with [10], where, under suitable assumptions of network parameters, the RTT delay R_0 has been ignored, in the wireless scenario considered here, we assume that the capacity variations occur on an RTT time scale. Accordingly, we have not ignored the RTT R_0 in this linearized model.

B. Transfer Function of the AQM Controller

The objective of an AQM controller is to control the queue length of the router at the desired set value. The controller is also required to be robust with respect to variations in network parameters like queue set point and number of TCP sources. We can represent the entire system by the model as shown in Fig. 2. The inputs to the plant are probability of packet drop δp and capacity variations δC, while the output is the queue length deviation δq. Note that the controller takes queue length as the input and produces probability of packet drop as output.

Here, we derive the transfer function of the controller. The fluid flow model is defined for a continuous plant whereas the actual network transmits packets at discrete intervals. Moreover, the state of the system is also monitored in actual practice at discrete intervals. Hence, we present a discretized linear model.

We begin by first writing (1) and (2) in state-space-like form as follows:

$$\begin{aligned} \dot{x} &= \begin{bmatrix} \frac{-NC_0}{(q_0+C_0T_p)^2} & 0 \\ \frac{NC_0}{q_0+C_0T_p} & \frac{-C_0}{q_0+C_0T_p} \end{bmatrix} \begin{bmatrix} \delta W(t) + \delta W(t - R_0) \\ \delta q(t) \end{bmatrix} \\ &\quad + \begin{bmatrix} \frac{-C_0(q_0+C_0T_p)}{2N^2} & 0 \\ 0 & \frac{q_0}{q_0+C_0T_p} \end{bmatrix} \begin{bmatrix} \delta p(t - R_0) \\ \delta C(t) \end{bmatrix} \end{aligned} \tag{4}$$

$$y = \delta q(t) = [0 \quad 1] \begin{bmatrix} \delta W(t) \\ \delta q(t) \end{bmatrix}. \tag{5}$$

Here, the state x at time t is $(\delta W(t), \delta q(t))$ and the input u is $(\delta p, \delta C)$.

After defining the constants

$$a_{11} := \frac{-NC_0}{(q_0 + C_0 T_p)^2} \quad a_{21} := \frac{NC_0}{q_0 + C_0 T_p}$$
$$a_{22} := \frac{-C_0}{q_0 + C_0 T_p} \quad b_{11} := \frac{-C_0(q_0 + C_0 T_p)}{2N^2}$$
$$b_{22} := \frac{q_0}{q_0 + C_0 T_p}$$

we get the following state-space-like set of equations:

$$\begin{aligned} \dot{x} &= \begin{bmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} \delta W(t) + \delta W(t - R_0) \\ \delta q(t) \end{bmatrix} + \begin{bmatrix} b_{11}\delta p(t - R_0) \\ b_{22}\delta C(t) \end{bmatrix} \\ y &= [0 \quad 1] \begin{bmatrix} \delta W(t) \\ \delta q(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta p(t - R_0) \\ \delta C(t) \end{bmatrix}. \end{aligned} \tag{6}$$

The following theorem relates the order of the discrete time system obtained by discretization of the above system.

Theorem: Consider the infinite dimensional linear delay differential system in (6). Discretize this system with any sampling period T_s such that the round trip R_0 and T_s are related by $R_0 = nT_s$ for some positive integer n. Suppose one of the following schemes is used for discretization:

1) the zeroth-order hold (ZOH) discretization;
2) bilinear transformation;
3) the backward Euler method.

Then, the discretized system is a finite-dimensional linear discrete-time system of order $(n + 2)$.

The significance of the theorem is as follows. As observed from (4), loosely speaking, the control action δp has an effect on the to-be-regulated output δq only after time R_0, while δC influences $\dot{\delta}\, q$ immediately. Under this circumstance, ignoring the delay would lead to the resulting model suggesting an exaggerated ability of the control action δp to cancel the effect of disturbance δC; controller design using the resulting plant model would not perform satisfactorily on the actual plant system. However, the delay term in the delay-differential equation [(6)] poses difficulties to controller design routines using standard controller design packages. One could consider designing a controller for the plant by first ignoring the delay and then ensuring sufficient phase margin to counter *all* delays up to R_0 (see [10]). In this paper, we have considered the situation where the controller achieves good queue control for a sufficiently wide range of values around R_0, which is the nominal round-trip time. (Our simulations in Section IV-B demonstrate that queue control is achieved for nonnominal values of time delay also.) A topic for future study is to calculate the "jitter margin" of this system:

this is relevant due to the varying time delay in the model and the destabilizing effects this can cause. See [21] for an analysis into this.

On the other hand, one could approximate the irrational transfer function e^{-sR_0} with a suitably high-degree Padé approximation (with or without imposing a difference in the degrees of numerator and denominator, see [22]); this would cause a large increase in the dimension of the approximated model and, hence, the controller.

Our approach of discretizing the system incorporates the delay e^{-sR_0} as z^{-n}, and thus the resulting discrete-time system is amenable for controller design systematically using standard controller design packages. Discretizing the continuous-time system is reasonable given that eventually AQM techniques are implemented digitally.

We now proceed to prove the theorem. We prove this only for the bilinear transformation, since it is similar for the other two discretization methods. The bilinear transformation corresponds to piecewise linear approximation of the signals between sampling instants, i.e., the trapezoidal rule of approximating integration. See [23] for more information about the bilinear and other discretization schemes.

Proof: The fact that the resulting discretized system is finite-dimensional follows due to the discretization of the (infinite-dimensional) delay operator e^{-sR_0} to get the finite-dimensional operator z^{-n}. It is the order that remains to be calculated.

Consider the system as described by (6). Taking the Laplace transform of the signals and incorporating the time delay R_0 by introducing the operator e^{-sR_0}, the transfer function $G(s)$ of the plant from the input signals $(\delta p, \delta C)$ to the output signal δq evaluates to

$$G(s) = [0 \quad 1]\left(sI - \begin{bmatrix} a_{11}(e^{-sR_0}+1) & 0 \\ a_{21} & a_{22} \end{bmatrix}\right)^{-1} \times \begin{bmatrix} b_{11}e^{-sR_0} & 0 \\ 0 & b_{22} \end{bmatrix}.$$

Simplifying this, we get

$$G(s) = \frac{\left[a_{21}b_{11}e^{-sR_0} \quad \left(s - a_{11}(1+e^{-sR_0})\right)b_{22}\right]}{(s-a_{22})(s-a_{11}-a_{11}e^{-sR_0})}. \quad (7)$$

Note that $G(s)$ acts on input $\begin{bmatrix} \delta p \\ \delta C \end{bmatrix}$ to give output δq.

Discretization of the transfer function $G(s)$ using the bilinear transformation translates to replacing s with $(2/T_s)(z-1/z+1)$ and e^{-sR_0} with z^{-n}, where the sampling period T_s satisfies $R_0 = nT_s$. By doing this, we get

$$\hat{G}(z) = \frac{1}{d(z)}\left[a_{21}b_{11}z^{-n} \quad b_{22}\left(\frac{2}{T_s}\right)\left(\frac{z-1}{z+1}\right) - a_{11}b_{22}(z^{-n}+1)\right]$$

where $d(z)$ is

$$\frac{\left(\frac{2}{T_s}(z-1) - a_{11}(z^{-n}+1)(z+1)\right)\left(\frac{2}{T_s}(z-1) - a_{22}(z+1)\right)}{(z+1)^2}.$$

On rewriting the equation for $\hat{G}(z)$, we get

$$\hat{G}(z) = \left[\frac{a_{21}b_{11}(z+1)^2}{z^n(z+1)^2d(z)} \times \frac{b_{22}\left(\frac{2}{T_s}\right)(z-1)(z+1)z^n - a_{11}b_{22}(z^n+1)(z+1)^2}{z^n(z+1)^2d(z)}\right].$$

The expression $z^n(z+1)^2d(z)$ simplifies to

$$\frac{4}{T_s^2}(z-1)^2z^n - a_{22}z^n(z+1)(z-1)\frac{2}{T_s} - a_{11}(1+z^n)(z+1)(z-1)\frac{2}{T_s} + a_{22}a_{11}(z^n+1)(z+1)^2$$

which is of order $n+2$ and has no common factors with the numerators in $\hat{G}(z)$. This proves the theorem. □

Once we have obtained a discrete-time model for the system, the transfer matrix $\hat{G}(z)$ is used for the purpose of designing a controller that meets a suitable control objective. This digital controller is used for simulations on a sample network, and the performance is evaluated. Let $K(z)$ be the transfer function of the controller from the controller's input δq to the controller's output δp. Express $K(z)$ as a ratio of polynomials in z as follows:

$$K(z) = \frac{n_0 + n_1 z + \cdots + n_m z^m}{d_0 + d_1 z + \cdots + d_{m-1}z^{m-1} + z^m} \quad (8)$$

where m denotes the order of the controller. For the controllers designed in this paper ($\mathcal{H}_\infty$ and LQG), the order equals the plant order: $n+2$, as shown in the above theorem. The controller $K(z)$, acting on its input $\delta q(\cdot)$ and giving output $\delta p(\cdot)$, is implemented in the time domain as follows:

$$\delta p(k) := -d_{m-1}\delta p(k-1) - d_{m-2}\delta p(k-2) - \cdots - d_0\delta p(k-m) + n_m\delta q(k) + \cdots + n_0\delta q(k-m).$$

Initial conditions at the start of controller implementation can be assumed to be zero; due to closed-loop stability, the effect of these few initial steps is short-lived.

III. Robust Queue Management

The AQM problem can now be formulated for the linear system $\hat{G}(z)$ as a disturbance attenuation problem: design a controller $K(z)$ that takes input δq and gives output δp such that the "effect" of the disturbance input δC on δq is sufficiently small. There are several ways in which this effect can be quantified; one of the ways is to use the so-called induced ℓ_2 norm of the transfer function. We use the ratio of the ℓ_2 norm[1] of the output signal Pd to that of the input signal d to define the $\mathcal{H}_\infty$ norm of a stable transfer function $P(z)$ as

$$\|P\|_{\mathcal{H}_\infty} = \sup_{d\in\ell_2,\ d\neq 0} \frac{\|Pd\|_2}{\|d\|_2}$$

[1]The ℓ_2 norm of a sequence $\{d(\cdot)\}$ is defined as $\|d\|_2^2 := \sum_{k\in\mathbb{Z}}|d(k)|^2$, where $\mathbb{Z}$ denotes the set of all integers. The space of such square summable sequences is also denoted by ℓ_2.

where Pd is the output for input d. Thus, choosing a controller to reduce the closed-loop system's $\mathcal{H}_\infty$ norm amounts to reducing the gain that the worst disturbance input signal δC can have on the queue length deviation δq in the closed-loop system; this is called $\mathcal{H}_\infty$ control. Alternatively, instead of $\mathcal{H}_\infty$ control, one can shape the closed-loop system such that the energy in the impulse response is minimized; this is called $\mathcal{H}_2$ control. This is made precise and is elaborated on in the following subsection. The importance of $\mathcal{H}_\infty$ control is in the robustness property of the controller. More precisely, the nominal plant $\hat{G}$ is only an approximation of the actual TCP network and hence the actual system can be considered as a perturbation of $\hat{G}$. Our controller (designed using the nominal plant) is required to be robust to these perturbations if the objective of queue stabilization is to be achieved for the actual plant also. Moreover, due to the time-varying nature of wireless link capacity, we have considered capacity as an exogenous disturbance input. It is in this sense that the method proposed in this paper provides for RQM of the buffer in the wireless router to address congestion.

Thus, this scheme uses instantaneous queue measurement to calculate the probability of randomly dropping packets from the queue to achieve a robust control of queue length. In short, it is AQM with a robust instantaneous queue measurement algorithm and random packet drop policy. We end this section with a quick summary of the major steps involved in the RQM scheme, given here, and then proceed to evaluate performance results of simulations using this method in the following section.

1) Knowing the operating points (W_0, q_0, p_0, C_0), fix the desired queue length in buffer to q_0 and the probability of drop to p_0, using the equilibrium equation.
2) Design an $\mathcal{H}_\infty$ controller (using a standard controller design routine) to get the digital controller $K(z)$.
3) Measure the instantaneous queue length of the router at every packet arrival and calculate the deviation from the set point value q_0 which gives rise to δq.
4) Use $K(z)$ to compute the incremental probability of drop δp corresponding to δq by $K(z)\widehat{\delta q}(z) =: \widehat{\delta p}(z)$ where $\widehat{\delta q}(z)$ and $\widehat{\delta p}(z)$ are the Z-transforms of signals δq and δp after sampling.
5) Update the probability of drop for the current instant, $p(k) = p_0 + \delta p(k)$.
6) Drop the packets randomly from the router buffer with the above calculated drop probability $p(k)$.

A. LQG and $\mathcal{H}_2$ Control

The controller $K(z)$ designed and implemented as described above has been chosen to obtain robustness, and $\mathcal{H}_\infty$ control has been used for that purpose. The advantage of our method (of discretizing the plant thus incorporating the delay) is visible by the ease with which we can handle other performance objectives like LQG control. This is elaborated below.

Consider the linear system shown in Fig. 2, where the controller $K(z)$ is to be designed for the plant $\hat{G}(z)$ that is influenced by disturbances δC and the to-be-regulated variable is δq. The first objective is that the controller $K(z)$ has to stabilize the closed-loop system, i.e., the variable δq has to go to zero for any initial condition in the absence of disturbance. In addition to stabilization assuming disturbance is zero, it is also often required that the "effect" of disturbance on δq is minimized in the following sense. Assume δC is a zero-mean white Gaussian process. Due to the system being linear and time-invariant (LTI), the output δq is also a Gaussian process, and the steady-state variance of δq depends on which controller K is used for stabilization. The square-root of the ratio of the steady state variance of δq to the variance of δC turns out to be equal to the $\mathcal{H}_2$ norm of the closed-loop transfer function (see [23, Sec. 4.3]). The $\mathcal{H}_2$ norm of a stable single-input single-output (SISO) transfer function $f(z)$ is defined as

$$\|f\|_{\mathcal{H}_2} := \left[\frac{1}{2\pi}\int_0^{2\pi} f(e^{i\theta})\overline{f(e^{i\theta})}d\theta\right]^{1/2}.$$

Thus, to minimize the effect of disturbance input δC on the variations in the queue fluctuations, one can consider the design of a controller $K(z)$ that minimizes the $\mathcal{H}_2$ norm of the closed-loop transfer function from δC to δq. In the situation of using packet drop probability as the controlled input, the constraint that the probability is required to be within the range 0 to 1 can be incorporated by penalizing δp. This is done by including δp (with a suitable weight) together with δq as the to-be-regulated output. This brings us to LQG control. Consider the objective of finding an input sequence δp to the linear system (see Fig. 2 again) such that, for a fixed integer T and a constant $\rho > 0$, the summation

$$\frac{1}{T}\sum_{k=0}^{T} E\left[(\delta q(k))^2 + \rho\,(\delta p(k))^2\right]$$

is to be minimized. The LQG problem (for the simplified scalar case) is to minimize the above summation in the limit $T \to \infty$. This also translates to minimizing the steady state expected value of $(\delta q)^2 + \rho(\delta p)^2$ by a suitable choice of the input sequence δp. The parameter ρ decides the importance of usage of packet drop probability relative to that of regulation of the queue length: high value of ρ causes less usage of δp by accordingly tolerating higher deviations δq.

The control law that results in the above minimization can be determined by standard routines using the model $\hat{G}(z)$ as above. This is elaborated on in the following subsection.

B. Controller Design

We briefly review the methods for controller design when using a standard package, say MATLAB. Consider Fig. 2, where the plant's inputs are δC and δp: the disturbance and the control inputs, respectively. Accordingly, the plant transfer matrix $\hat{G}(z)$ (and $G(s)$) has two columns, and specifying them in the correct order is required for use of Matlab commands "hinf" or "h2lqg": the first input here is δC and the second is δp. Further, in this case, the to-be-regulated variable δq is also the measured output which the controller uses as its input. This situation is expressed in MATLAB by repeating the rows of C and D matrices in the standard state-space model. The command "mksys" builds the system model as required by the routines "hinf" and "h2lqg." The in-built routines for these controllers use Loop-shifting and Riccati equation based methods (adapted from [24]) to arrive at

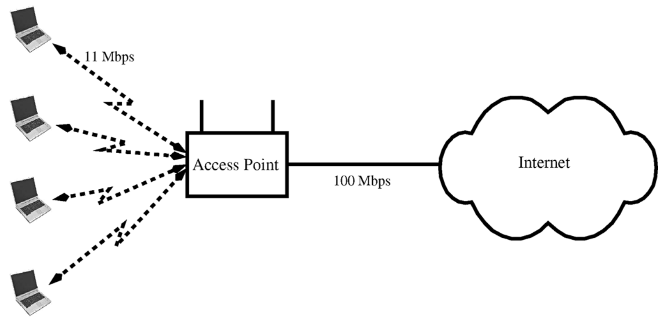

Fig. 3. Wireless network topology for simulations.

TABLE I
SUMMARY OF PARAMETERS FOR RED

Parameter	Value
$\max_{\rm p}$	0.1
$\min_{\rm th}$	30
$\max_{\rm th}$	90
w_q	7.27×10^{-4}
wait	on
gentle	on

TABLE II
SUMMARY OF PARAMETERS FOR OUR PROPOSED ALGORITHMS

Parameter	Value
Nominal round trip delay, R_0	80 ms
Number of TCP sources, N	18
Data rate of the congested link	10 Mbps
Queue buffer size, $\bar{q}$	200 packets
Desired queue set point, q_0	90 packets
Nominal capacity C_0	1250 pps
Propagation delay T_p	0.8 ms
Sampling period T_s	40 ms

the suboptimal/optimal controller, if one exists. Design procedures in other controller design packages follow along similar lines.

IV. SIMULATIONS AND PERFORMANCE RESULTS

Here, we demonstrate the performance of our algorithm through simulations using the $ns-2$ network simulator [25]. We compare packet drop, throughput, and queue stability for various AQM algorithms: RED with the gentle option, Adaptive RED, and our proposed methods: $\mathcal{H}_2$ and $\mathcal{H}_\infty$.

We simulate the IEEE 802.11 Wireless Local Area Network (WLAN) scenario with channel data rate of 11 Mb/s ([26]). The wired network (local area network (LAN) or Internet) is connected to the access point with a link of capacity 100 Mb/s. As shown in Fig. 3, there are TCP flows between the Internet and the wireless nodes; the precise number of flows is varied for different scenarios as elaborated below. The round-trip times of these flows range between 80–160 ms (while nominal RTT R_0 is 80 ms). We consider fixed-length packets, each of 1000 bytes. The maximum window size of the TCP flows is kept at the default $ns-2$ value of 20 packets. We implement the AQM at the access point (router); this is assumed to have a buffer of 200 packets. The wireless nodes in the simulation use $ns-2$'s wireless channel model. The large-scale fading effects in a wireless channel are simulated by using the log-normal shadowing as the radio propagation model. The model was configured with a path-loss exponent value of 2 and shadowing mean of 0 and deviation of 4, which are the values recommended as typical for an outdoor environment. See the $ns-2$ manual [25]. In order to implement the proposed AQM schemes, the default $ns-2$ was modified by extending the queue class to incorporate our control algorithm.

Though we have assumed only one state for the window size in the nominal system model to capture the influence that packet drop probability has on the window size, the simulations have been performed using several TCP Reno sources which have possibly different window sizes. The amount of data to be transmitted at any instant is chosen randomly for each source destination pair; accordingly they exhibit different trajectories for window size. Thus, at any time instant, the queue management is being done for nonhomogeneous sources that have different window sizes.

We compare the proposed methods with adaptive RED and RED algorithms with the gentle option. For fairness of the comparisons, we first tune RED parameters for better performance. For more details on tuning RED, the readers are referred to [3]. The RED model used for queue averaging and packet-drop probability computation is standard. The exponentially weighted moving average $q_{\rm avg}$ is computed from the current queue length $q(k)$ using a suitable weight w_q and the following equation:

$$q_{\rm avg}(k+1) = w_q q(k) + (1 - w_q) q_{\rm avg}(k).$$

Using the maximum and minimum thresholds, $\max_{\rm th}$ and $\min_{\rm th}$, respectively, the average queue $q_{\rm avg}$, and the maximum drop probability $\max_{\rm p}$, the following equation is used to compute the packet-drop probability:

$$p(q_{\rm avg}) = \begin{cases} 0 & \text{for } q_{\rm avg} \in [0, \min_{\rm th}) \\ \frac{\max_{\rm p}(q_{\rm avg} - \min_{\rm th})}{\max_{\rm th} - \min_{\rm th}} & \text{for } q_{\rm avg} \in [\min_{\rm th}, \max_{\rm th}] \\ 1 & \text{for } q_{\rm avg} \in (\max_{\rm th}, \bar{q}]. \end{cases}$$

The "gentle" option in RED ensures a smoother graph of p at $\max_{\rm th}$: see [1] for further elaboration. The simulations parameters for RED are chosen as shown in Table I.

Adaptive RED automatically tunes its parameters and maintains its average queue length at half of the buffer size. In order to have a reasonable comparison, we choose the queue set point as 90 for both the proposed algorithms—$\mathcal{H}_\infty$ and $\mathcal{H}_2$. Note that the queue set point can be explicitly chosen in our proposed method; this is an advantage as compared to the existing AQM techniques. Nevertheless, choosing the correct set point for an environment is tricky as there exists a tradeoff between average delay and throughput.

The nominal design parameters used in the simulations are given in Table II. The chosen parameters give rise to the equilibrium point (W_0, q_0, p_0, C_0) as follows:

$$W_0 = \frac{q_0 + C_0 T_p}{N} = 6.1667 \quad p_0 = \frac{2}{W_0^2} = 0.0525.$$

For these values, the $\mathcal{H}_\infty$ and $\mathcal{H}_2$ controllers are designed using MATLAB. The command "$hinf$" is used to generate the $\mathcal{H}_\infty$

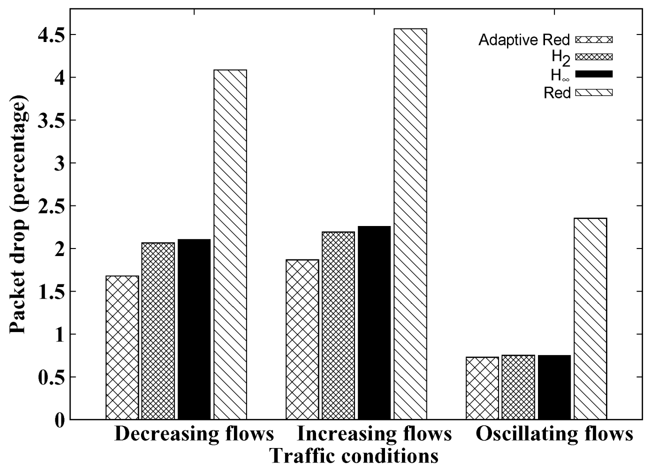

Fig. 4. Packet-drop percentage of the AQM algorithms for various load scenarios.

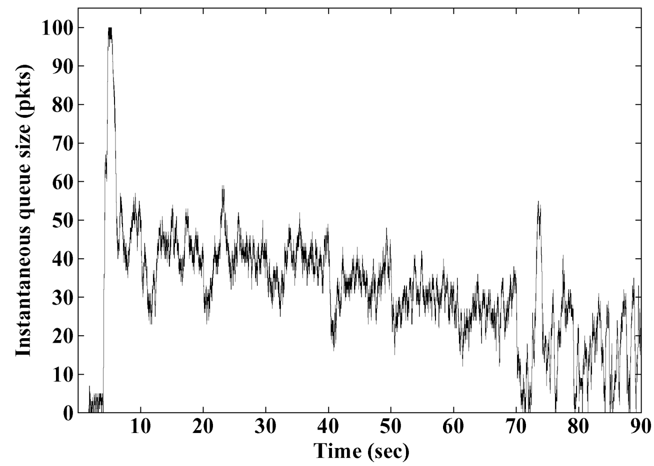

Fig. 5. Queue control: RED (decreasing load).

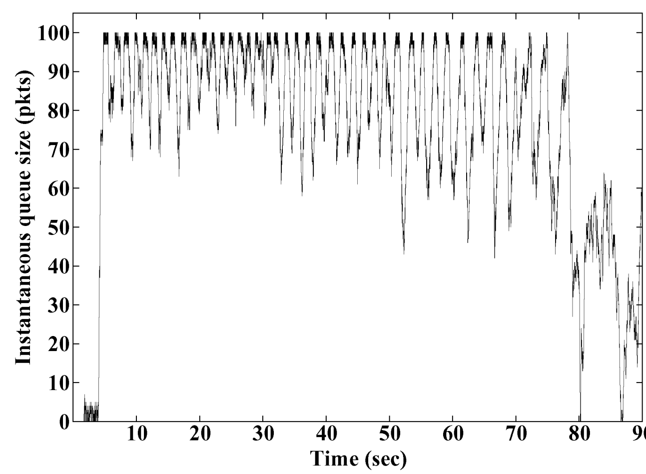

Fig. 6. Queue control: Adaptive RED (decreasing load).

controller, whereas the command "$h2lqg$" is used for the $\mathcal{H}_2$ controller design. The obtained controllers are as follows:

$$K_{\mathcal{H}_\infty}(z) = \frac{11.22z^4 + 0.79z^3 - 10.03z^2 + 0.79z + 0.40}{10^3(z^4 + 2.55z^3 + 2.43z^2 + 1.11z + 0.24)}$$
$$K_{\mathcal{H}_2}(z) = \frac{18.26z^3 - 16.97z^2 + 0.64z + 0.64}{10^3(z^4 + 2.55z^3 + 2.58z^2 + 1.42z + 0.39)}.$$

Both controllers have all poles within the unit circle.

In spite of designing the controller for the above-mentioned parameters, due to the inherent robustness property, the controllers perform satisfactorily under other conditions as well.

We perform the simulations under a fluctuating network load. The objective of such an experiment is to evaluate the ability of various AQM algorithms to control the queue fluctuations and maintain the queue stability. Each simulation experiment has been performed for 90 seconds. In the first scenario, we increase the number of TCP flows by three new flows every 10 s: we call this "increasing load." The interval of 10 s is chosen so that Adaptive RED algorithm has sufficient time to stabilize the queue (see [3]). For the second scenario, called "decreasing load," we decrease the number of TCP flows every 10 s by three flows starting from a fairly high initial number of flows. For the final scenario, we vary the number of TCP flows as a slowly transpiring burst, i.e., oscillating load.

A. Packet Drop and Throughput

We compare the packet drops for various AQM algorithms. The fraction of packet drop is calculated as the ratio of number of packets dropped to the number of packets received at the router. The packet-drop percentages exhibited by the algorithms have been shown in Fig. 4. Adaptive RED demonstrates lower packet drop under all conditions, while the proposed $\mathcal{H}_2$ and $\mathcal{H}_\infty$ algorithms demonstrate a slightly higher packet drop percentage, and RED with gentle parameter has the highest packet drop among all algorithms for all the three traffic conditions mentioned above. Nevertheless, as the following subsections reveal, queue control is significantly better when using the $\mathcal{H}_2$ and $\mathcal{H}_\infty$ controllers as compared with Adaptive RED and RED (with the gentle option).

Throughput for the system using the four schemes has also been evaluated for various network traffic simulations. We notice that the throughput for RED is worse, while that using ARED, $\mathcal{H}_2$, and $\mathcal{H}_\infty$ schemes are very close. Due to lack of space, a plot showing this is omitted.

B. Queue Control

The $\mathcal{H}_2$ and $\mathcal{H}_\infty$ controllers have been designed with the explicit objective of queue control and our simulation experiments demonstrate that they achieve better queue control. When the deviation in the instantaneous queue length from the average queue length is small, we say good queue control has been achieved. We discuss the simulation results obtained for various scenarios in this subsection.

Figs. 5–8 depict the instantaneous queue size (in packets) for a scenario where the number of TCP flows are decreased gradually. We observe that our proposed algorithms have better queue control than Adaptive RED. The results for the scenario of increasing load has not been shown; as in terms of queue control, the results observed are very similar to those of decreasing load. Both $\mathcal{H}_2$ and $\mathcal{H}_\infty$ exhibit almost identical instantaneous queue sizes and there is no significant difference in terms of performance; this is true inspite of the varied nature of the $\mathcal{H}_2$ and

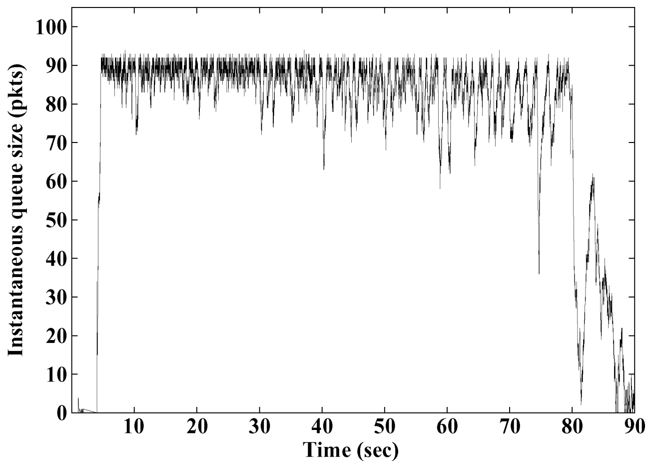

Fig. 7. Queue control: $\mathcal{H}_2$ (decreasing load).

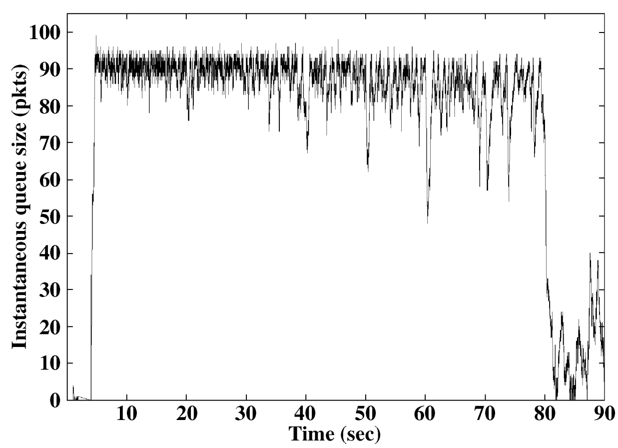

Fig. 8. Queue control: $\mathcal{H}_\infty$ (decreasing load).

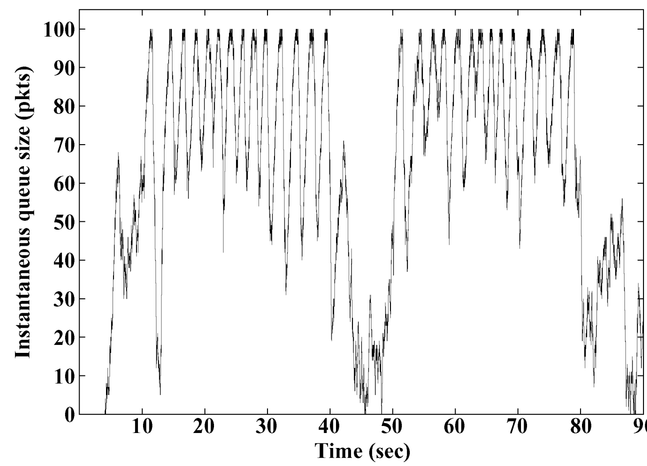

Fig. 9. Queue control: Adaptive RED (oscillating load).

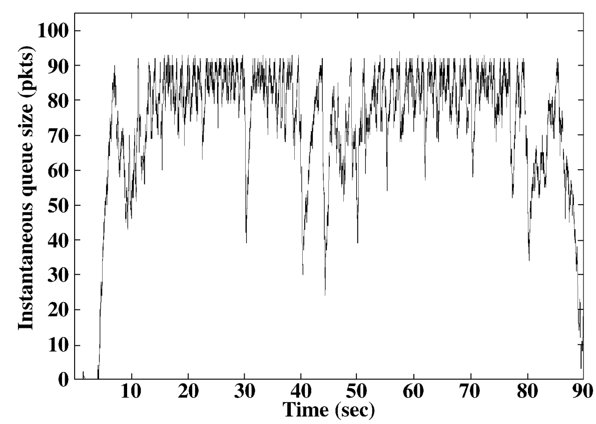

Fig. 10. Queue control: $\mathcal{H}_2$ (oscillating load).

$\mathcal{H}_\infty$ control objectives (as described in Section III). In all of the scenarios, RED with gentle option is unable to control the oscillations in the queue. Any further tuning on RED only results in worse performance.

Figs. 9 and 10 contain plots of the instantaneous queue for oscillating load, i.e., periodically varying number of TCP flows. These results confirm that $\mathcal{H}_2$ and $\mathcal{H}_\infty$ have better queue control than Adaptive RED, while RED performs even worse. The queue length varies between 60–100 in the case of Adaptive RED whereas in the case of our proposed algorithms the range is between 80 and 90.

We conclude from Figs. 4–10 that better queue control has been achieved with only a marginal increase in percentage packet drop.

C. Summary of Comparison With Other AQM Schemes

Here, we summarize the benefits of using RQM scheme over RED and Adaptive RED AQM schemes. The foremost advantage of using the RQM scheme is that explicit control over the queue length set-point q_0 can be incorporated into controller implementation; this allows better buffer utilization during unexpected bursty traffic and good throughput. In [27] and [28] also, there is a similar flexibility in explicitly choosing q_0, though the controller design follows a different approach and has a different application context.

In addition to the ability to choose the queue length set point, the proposed controller also reduces fluctuations in the instantaneous queue length thus achieving better queue control. Good queue length regulation ensures that the delay of the packets is more predictable and controllable. This helps in QoS guarantees when dealing with Internet applications.

V. Conclusion

AQM algorithms have been extensively studied for wired networks. However, the design of AQM for wireless network has not been adequately addressed. In this paper, we have proposed the design of AQM for wireless networks. Specifically, we have proposed a way to address capacity variations of the wireless link by treating it as an external input.

The controller design has been done offline using a linearized model about a suitably chosen operating point. There is no online tuning or adjustment of parameters to be done by the user while monitoring the network performance. Our simulation results on the IEEE 802.11-based wireless network demonstrate that the proposed algorithm achieves significant advantage in terms of queue stability over various AQM algorithms with a minor increase in packet drop.

Further, the controller is robust to wide variations in the capacity and the simulations also demonstrate the robustness to significant variations in the round trip time.

REFERENCES

[1] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[2] M. May, J. Bolot, C. Diot, and B. Lyles, "Reasons not to deploy RED," in *Proc. IWQoS*, 1999, pp. 260–262.

[3] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED," [Online]. Available: http://citeseer.ist.psu.edu/floyd01adaptive.html

[4] T. Ott, T. Lakshman, and L. Wong, "SRED: Stabilized RED," in *Proc. IEEE INFOCOM*, 1999, pp. 1346–1355.

[5] W. Feng, D. Kandlur, D. Saha, and K. Shin, "BLUE: An alternative approach to active queue management algorithm," in *Proc. NOSSDAV*, 2001, pp. 41–50.

[6] W. Feng, D. Kandlur, D. Saha, and K. Shin, "A self-configuring RED gateway," in *Proc. IEEE INFOCOM*, 1999, pp. 1320–1328.

[7] S. L. S. Athuraliya, V. H. Li, and Q. Yin, "REM: Active queue management," *IEEE Network*, vol. 15, pp. 48–53, 1999.

[8] S. Kunniyur and R. Srikant, "An adaptive virtual queue (AVQ) algorithm for active queue management," *IEEE/ACM Trans. Netw.*, vol. 12, no. 2, pp. 286–299, Apr. 2004.

[9] V. Misra, W. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proc. SIGCOMM*, 2000, pp. 151–160.

[10] C. Hollot, V. Misra, D. Towsley, and W. Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Trans. Autom. Control*, vol. 47, no. 6, pp. 945–959, Jun. 2002.

[11] C. Hollot, V. Misra, D. Towsley, and W. Gong, "A control theoretic analysis of RED," in *Proc. INFOCOM*, 2001, vol. 3, pp. 1510–1519.

[12] F. Kelly, "Mathematics unlimited—2001 and beyond," in *Mathematical Modelling of the Internet*, B. Engquist and W. Schmid, Eds. Berlin, Germany: Springer-Verlag, 2001.

[13] S. Low, F. Paganini, and J. Doyle, "Internet congestion control," *IEEE Control Syst. Mag.*, vol. 22, no. 1, pp. 28–43, 2004.

[14] P. Quet and H. Ozbay, "On the design of AQM supporting TCP flows using robust control theory," *IEEE Trans. Autom. Control*, vol. 49, no. 6, pp. 1031–1036, Jun. 2004.

[15] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge: Cambridge Univ. Press, 2005.

[16] F. Zheng and J. Nelson, "An $\mathcal{H}_\infty$ approach to congestion control design for AQM routers supporting TCP flows in wireless access networks," *Comput. Netw.: Int. J. Comput. Telecommun. Netw.*, vol. 51, pp. 1684–1704, 2007.

[17] R. Johari and D. Tan, "End-to-end congestion control for the Internet: Delays and stability," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 818–832, Dec. 2001.

[18] L. Massoulie, "Stability of distributed congestion control with heterogenous feedback delays," Microsoft Res., 2000.

[19] G. Vinnicombe, "On the stability of end-to-end congeston control for the Internet Univ. of Cambridge, Cambridge, U.K., 2001.

[20] F. Zheng, Q. Wang, and T. Lee, "A heuristic approach to solving a class of bilinear matrix inequality problems," *Syst. Control Lett.*, vol. 47, pp. 111–119, 2002.

[21] C.-Y. Kao and B. Lincoln, "Simple stability criteria for systems with time-varying delays," *Automatica*, vol. 40, pp. 1429–1434, 2004.

[22] M. Vajta, "Some remarks on Padé approximation," in *Proc. TEMPUS-INTCOM Symp.*, Veszprém, Hungary, 2000, pp. 53–58.

[23] T. Chen and B. Francis, *Optimal Sampled-Data Control Systems*. New York: Springer-Verlag, 1995.

[24] M. Safonov, D. Limebeer, and R. Chiang, "Simplifying the $\mathcal{H}_\infty$ theory via loop shifting, matrix pencil and descriptor concepts," *Int. J. Control*, vol. 50, pp. 2467–2488, 1989.

[25] Network Simulator NS. [Online]. Available: http://nile.wpi.edu/NS/

[26] *IEEE Standard for Information Technology—Telecommunications and Information Exchange between Systems—Local and Metropolitan Area Networks—Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std 802.11-2007, LAN/MAN Committee, Mar. 2007.

[27] P. Ignaciuk and A. Bartoszewicz, "Linear quadratic optimal discrete time sliding mode controller for connnection oriented communication networks," *IEEE Trans. Ind. Electron.*, vol. 55, no. 11, pp. 4013–4021, Nov. 2008.

[28] A. Pietrabissa, "A multi-model reference control approach for bandwidth-on-demand protocols in satellite networks," *Control Eng. Practice*, vol. 16, pp. 847–860, 2008.