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 wired link which is assumed to have a fixed capacity, a wireless link has a capacity that is time varying due to multipath fading and mobility. 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.**

Keywords: Active Queue Management, Congestion Control, Wireless Network, Robust Control, $\mathcal{H}_\infty$ control, $\mathcal{H}_2$ control, discretization

I. Introduction

In packet switched networks, such as Transmission Control Protocol/Internet Protocol (TCP/IP) networks, data from the source travels through a number of intermediate nodes to reach the destination node. If the service rate at the intermediate nodes is less than the arrival rate of the packets, then the packets are queued in a buffer at the intermediate nodes. In practice, since the buffer's size is finite, packets may be dropped or experience high queuing delays. This phenomenon of congestion is detrimental to the performance of the network. TCP sources decrease their packet sending rates when packets are dropped. If the intermediate nodes drop packets only when their queues are full (Drop Tail technique), then it would cause global synchronization between the TCP flows, see [1]. Furthermore, they would not be able to effectively handle bursty traffic flows. Active Queue Management (AQM) methods drop packets randomly, even before the queue is full; thereby alleviating the problems due to Drop Tail. Hence, these techniques play an important role in controlling congestion in TCP networks.

An AQM algorithm comprises of two basic components: the first one monitors the fluctuations in queue length, while the other component 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. As a modification to the Drop Tail technique, Random Early Detection (RED) [1] was one of the earliest algorithms for AQM. In RED, when the average queue length varies between a pre-specified minimum threshold ($\min_{\text{th}}$) and a maximum threshold ($\max_{\text{th}}$), the probability of drop is varied linearly with the queue length from zero to a maximum packet drop probability ($\max_{\text{p}}$); for queue lengths greater than $\max_{\text{th}}$, the drop probability is set to 1. The 'gentle' option in RED ensures a smoother increase of drop probability from $\max_{\text{p}}$to 1 for queue lengths greater than $\max_{\text{th}}$. 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 off a lot of interest in other modifications of RED that addressed similar shortcomings of RED; see [4], [5], [6], [7], [8] for example.

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], [10], [11], [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 wired link which is assumed to have a fixed capacity, a wireless link has a capacity that is time varying due to multipath fading and mobility [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

The authors are with the Department of Electrical Engineering, Indian Institute of Technology Bombay, Powai Mumbai 400076, India. Corresponding author's email: belur@ee.iitb.ac.in

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 work was reported at National Conference on Communications, Jan. 2009 held at IIT Guwahati, India

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], [18], [19].

While it is important to design a controller that meets performance objectives *without* ignoring the delay 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 design of controller for various performance objectives. In this paper, 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 [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 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.

In order to ensure that the controller stabilizes the router queue inspite of the inaccuracies in the linearized discretized model, we design a *robust* controller using $\mathcal{H}_\infty$ control methods. We treat variations in the capacity as disturbances to the system and seek to design a controller which reduces the maximum effect that the disturbances can cause to fluctuations in the queue length. The robustness of this controller is verified using simulations. In this sense, the active queue management technique proposed in this paper can be termed as Robust Queue Management (RQM). The same discrete time model of the TCP network with a wireless link can be used for meeting other control objectives as well.

While we address the effect on throughput and 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 queue more predictable and controllable: this helps in Quality of Service guarantee for many internet applications.

The rest of the paper is organized as follows. We study the system model in the next section. 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.

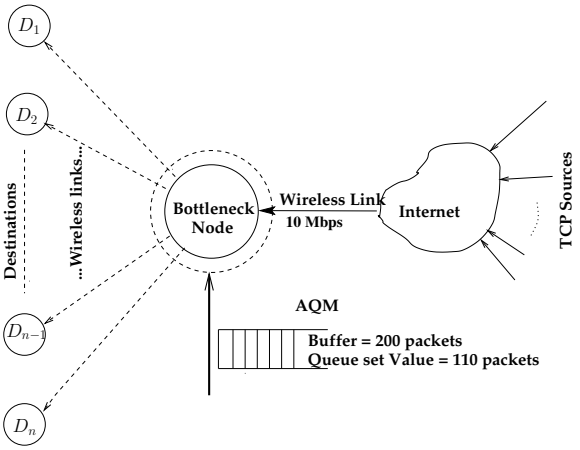

Fig. 1. TCP network model

II. System model and problem formulation

In this section, 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 Figure 1 where TCP sources send data towards their destinations in Internet through a router which represents the bottleneck node in the network. With 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. When a packet drop is detected by the TCP source, it 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 capacity of the router outgoing link and the round trip time (RTT). Note that RTT is the time required for the packet to reach the destination plus the time it takes for acknowledgement to reach the sender. Hence, it comprises of transmission delay, queuing 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 round trip time $R = \frac{q}{C} + T_p$.

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:

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

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

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 Equation (1) corresponds to additive increase of the window size after each round trip time (i.e., with receipt of acknowledgement packet), while the second term corresponds to the multiplicative decrease in the window size due to drops of packet. The queue length q and window-size W take their values in $[0, \bar{q}]$ and $[0, \bar{W}]$, where $\bar{q}$ and $\bar{W}$ denote buffer capacity and maximum window size respectively. 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 Equations (1) and (2). We call these values as an equilibrium point. Like in [11], the equations that have to be satisfied by the variables at an equilibrium point are

$$W_0^2 p_0 = 2, \quad W_0 N = R_0 C_0 \text{ and } R_0 = \frac{q_0}{C_0} + T_p. \quad (3)$$

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 also 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 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:

$$\begin{aligned} \delta\dot{W}(t) &= \tfrac{-NC_0}{(q_0+C_0T_p)^2}(\delta W(t) + \delta W(t-R_0)) \\ &\quad -\tfrac{C_0(q_0+C_0T_p)}{2N^2}\delta p(t-R_0), \\ \delta\dot{q}(t) &= \tfrac{C_0 N\delta W(t) - C_0\delta q(t) + q_0\delta C(t)}{q_0+C_0T_p}. \end{aligned}$$

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 RTT time scale. Accordingly, we have not ignored the RTT R_0 in this linearized model.

B. Transfer Function of 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 Figure 2. The inputs to the plant are probability of packet drop

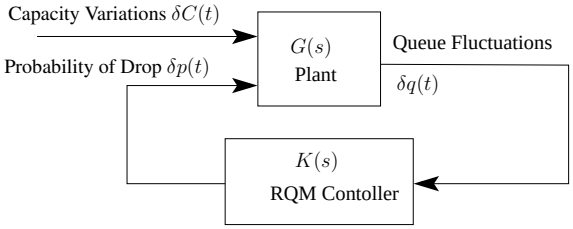

Fig. 2. AQM Controller Model for a Wireless Network

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.

In this section, we derive the transfer function of the controller. The fluid flow model is defined for a continous 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 Equations (1) and (2) in state space like form as follows. Here the state x at time t is $(\delta W(t), \delta q(t))$ and the input u is $(\delta p, \delta C)$.

$$\begin{aligned} \dot{x} &= \begin{bmatrix} \frac{-NC_0}{(q_0+C_0T_p)^2} & 0 \\ \frac{NC_0}{q_0+C_0T_p} & \frac{-C_0}{q_0+C_0T_p} \end{bmatrix} \begin{bmatrix} \delta W(t) + \delta W(t-R_0) \\ \delta q(t) \end{bmatrix} \\ &\quad + \begin{bmatrix} \frac{-C_0(q_0+C_0T_p)}{2N^2} & 0 \\ 0 & \frac{q_0}{q_0+C_0T_p} \end{bmatrix} \begin{bmatrix} \delta p(t-R_0) \\ \delta C(t) \end{bmatrix}, \quad (4) \\ y &= \delta q(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \delta W(t) \\ \delta q(t) \end{bmatrix}. \quad (5) \end{aligned}$$

After defining the constants

$$\begin{aligned} a_{11} &:= \tfrac{-NC_0}{(q_0+C_0T_p)^2}, \quad a_{21} := \tfrac{NC_0}{q_0+C_0T_p}, \\ a_{22} &:= \tfrac{-C_0}{q_0+C_0T_p}, \quad b_{11} := \tfrac{-C_0(q_0+C_0T_p)}{2N^2} \text{ and} \\ b_{22} &:= \tfrac{q_0}{q_0+C_0T_p}, \end{aligned}$$

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

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

The above system is discretized and the theorem below relates the order of the resulting finite dimensional discrete time system.

Theorem: *Consider the infinite dimensional linear delay differential system described by Equation* (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 any one of the following schemes is used for discretization:*
1) the Zeroth Order Hold (ZOH) discretization,
2) Bilinear Transformation, or
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 Equation (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 (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 upto R_0 (see [10]). Since we are required to assure controller performance only for delay of values around R_0 and not for all delays upto R_0, this phase margin would amount to too much conservatism in controller design, possibly compromising on other performance objectives.

On the other hand, one could approximate the irrational transfer function e^{-sR_0} (i.e. the delay) with a suitably high degree Padé approximation (with or without imposing a difference in the degrees of numerator and denominator, see [21]); 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 [22] 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 Equation (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 $G(s) =$

$$\begin{bmatrix} 0 & 1 \end{bmatrix} \left(sI - \begin{bmatrix} a_{11}(e^{-sR_0}+1) & 0 \\ a_{21} & a_{22} \end{bmatrix} \right)^{-1} \begin{bmatrix} b_{11}e^{-sR_0} & 0 \\ 0 & b_{22} \end{bmatrix}$$

Simplifying this, we get

$$G(s) = \frac{\begin{bmatrix} a_{21}b_{11}e^{-sR_0} & (s - a_{11}(1+e^{-sR_0}))b_{22} \end{bmatrix}}{(s-a_{22})(s-a_{11}-a_{11}e^{-sR_0})} \qquad (7)$$

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 $\frac{2}{T_s}(\frac{z-1}{z+1})$, and, e^{-sR_0} with z^{-n}, where the sampling period T_s satisfies $R_0 = nT_s$. By doing this, we get $\hat{G}(z) =$

$$\frac{1}{d(z)} \begin{bmatrix} a_{21}b_{11}z^{-n} & b_{22}(\frac{2}{T_s})(\frac{z-1}{z+1}) - a_{11}b_{22}(z^{-n}+1) \end{bmatrix}$$

where $d(z)$ is

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

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

$$\begin{bmatrix} \frac{a_{21}b_{11}(z+1)^2}{z^n(z+1)^2 d(z)} & \frac{b_{22}(\frac{2}{T_s})(z-1)(z+1)z^n - a_{11}b_{22}(z^n+1)(z+1)^2}{z^n(z+1)^2 d(z)} \end{bmatrix}$$

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

$$\begin{aligned} &\tfrac{4}{T_s^2}(z-1)^2 z^n - a_{22}z^n(z+1)(z-1)\tfrac{2}{T_s} \\ &- a_{11}(1+z^n)(z+1)(z-1)\tfrac{2}{T_s} + a_{22}a_{11}(z^n+1)(z+1)^2, \end{aligned}$$

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. Due to causality, $K(z)$ is proper, i.e. the degree of the numerator is at most the degree of the denominator. Most control design packages ensure properness of the controller; this is required for practical implementation. Express $K(z)$ as a ratio of polynomials in z as follows.

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

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. In this paper, we consider RTT and sampling period such that n is 2. Note that without loss of generality, the coefficient of the denominator's leading term can be assumed to be one. 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

$$\begin{aligned} \delta p(k) := &-d_{m-1}\delta p(k-1) - d_{m-2}\delta p(k-2) - \cdots \\ &-d_0\delta p(k-m) + n_m\delta q(k) + n_{m-1}\delta q(k-1) + \cdots \\ &+ n_0\delta q(k-m) \end{aligned}$$

i.e., the present value of $\delta p(k)$ is calculated using past values of δp and present/past values of δq. Zero initial conditions can be assumed for the first few steps; due to stability of the controller, influence of these few initial steps is transient/short-lived.

III. Robust queue management

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

[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 all integers. The space of such square summable sequences is also denoted by ℓ_2.

output signal Pd to that of the input signal d to define the $\mathcal{H}_\infty$ norm of a stable transfer function $P(z)$:

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

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 elaborated 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 Robust Queue Management (RQM) of buffer in the wireless router to address congestion.

Thus this scheme uses instantaneous queue measurement and calculates a suitable 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 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 a $\mathcal{H}_\infty$ 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 was 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 Linear Quadratic Guassian control. This is elaborated below.

Consider the linear system shown in Figure 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 Guassian process. Due to the system being Linear & Time-invariant (LTI), the output δq is also a Guassian 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 [22, Section 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 also. 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 Figure 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 relative importance of usage of packet drop probability in comparison to the 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 in the following subsection.

B. Controller Design

We briefly review the methods for controller design when using a standard package, say Matlab. Consider Figure 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

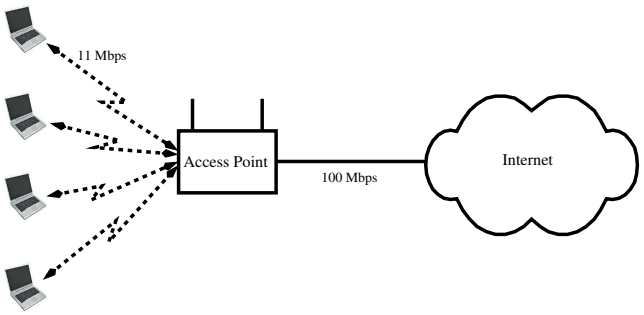

Fig. 3. Wireless Network topology for simulations

"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 controller design routines "hinf" and "h2lqg". The inbuilt routines for these controllers use Loop-shifting and Riccati equation based methods (adapted from [23]) to arrive at the sub-optimal/optimal controller, if one exists. Design procedures in other controller design packages follow along similar lines.

IV. Simulations and Performance Results

In this section, we demonstrate the performance of our algorithm through simulations using the *ns-2* network simulator [24]. In order to implement the proposed schemes in *ns-2*, appropriate changes have been made in the configurations. 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 IEEE 802.11 Wireless Local Area Network (WLAN) scenario with channel data rate of 11 Mbps ([25]). The wired network (local area network (LAN) or Internet) is connected to the access point with a link of capacity 100 Mbps. As shown in Figure 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 to 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 channel between the wireless nodes and the access point is time varying due to fading. These capacity variations are performed using the model supported in *ns-2* for wireless networks.

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 non-homogenous sources that have different window-sizes.

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

We compare the proposed methods with adaptive RED and RED algorithm with the gentle option. For fairness of the comparisons, we first tune RED parameters for its 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:

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

The 'gentle' option in RED ensures a smoother graph of p at $\max_{\rm th}$: see [1] for details. 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 110 for both the proposed algorithms- $\mathcal{H}_\infty$ and $\mathcal{H}_2$. Note that the queue set point can be explicitly chosen in the 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 IV. The chosen parameters give rise to equilibrium point (W_0, q_0, p_0, C_0) as below,

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

For these values, the $\mathcal{H}_\infty$ and $\mathcal{H}_2$ controllers are designed using Matlab [26]. The command *"hinf"* is used to generate the $\mathcal{H}_\infty$ controller whereas the command *"h2lqg"* is used for the $\mathcal{H}_2$ controller design. The obtained controllers are as follows :

$$K_{\mathcal{H}_\infty}(z) = \frac{11.22z^4 + 0.79z^3 - 10.03z^2 + 0.79z + 0.40}{10^3(z^4 + 2.55z^3 + 2.43z^2 + 1.11z + 0.24)},$$

and

$$K_{\mathcal{H}_2}(z) = \frac{18.26z^3 - 16.97z^2 + 0.64z + 0.64}{10^3(z^4 + 2.55z^3 + 2.58z^2 + 1.42z + 0.39)}.$$

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	110 packets
Nominal capacity C_0	1250 pps
Propagation delay T_p	0.8 ms
Sampling period T_s	40 ms

Both controllers have all their poles within the unit circle.

Even though we design 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 a load 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 3 new flows every 10 seconds. The interval of 10 seconds is chosen so that Adaptive RED algorithm has sufficient time to stabilize the queue (see [3]). For the second scenario, we decrease the number of TCP flows every 10 seconds by 3 flows starting from a fairly high initial number of flows. For the final scenario, we vary the flows as a slowly transpiring burst, i.e oscillating flows.

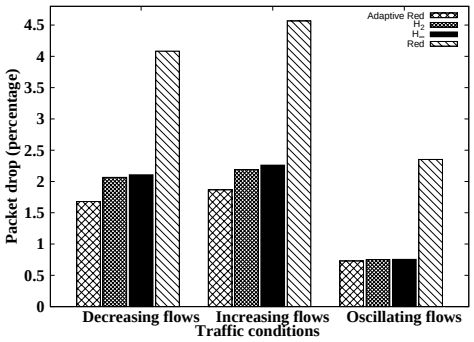

Fig. 4. Packet drop percentage of the AQM algorithms for the various traffic scenarios

A. Packet Drop

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 Figure 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 to Adaptive RED and RED (with the gentle option).

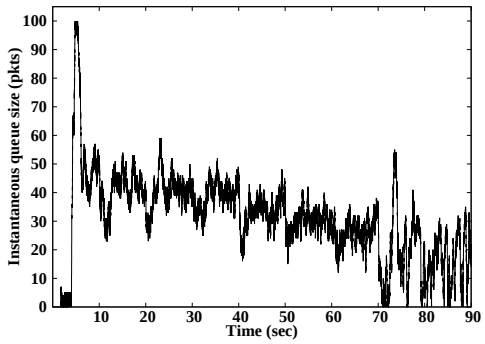

Fig. 5. Queue control: RED (decreasing flows)

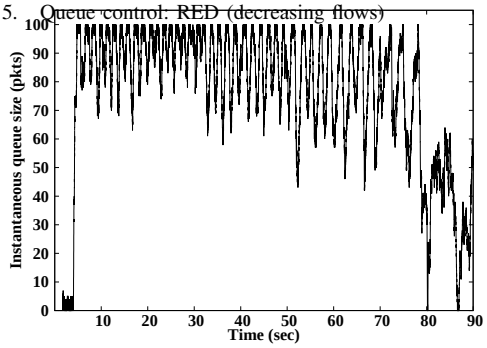

Fig. 6. Queue control: Adaptive RED (decreasing flows)

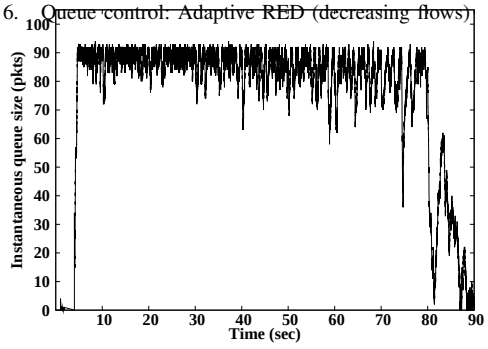

Fig. 7. Queue control: $\mathcal{H}_2$ (decreasing flows)

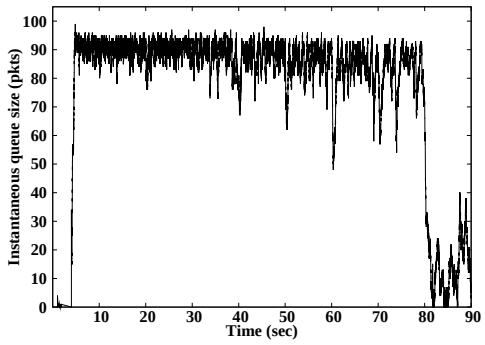

Fig. 8. Queue control: $\mathcal{H}_\infty$ (decreasing flows)

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.

Figures 5-8 depict the instantaneous queue size (in packets) for a scenario where flows are decreased gradually. We observe

that our proposed algorithms have better queue control than Adaptive RED. The results for the scenario of increasing flows has not been shown; as in terms of queue control, the results observed are very similar to those of decreasing flows. 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 $\mathcal{H}_\infty$ control objectives (as described in Section III). In all 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.

Figures **??**-**??** contain plots of the instantaneous queue for oscillating 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 deviations in queue size range between 100 and 60 in the case of Adaptive RED whereas in the case of our proposed algorithms the range is between 90 and 80.

Refering to Figures 4-**??**, we notice that better queue control has been acheived with only a marginal increase in percentage packet drop.

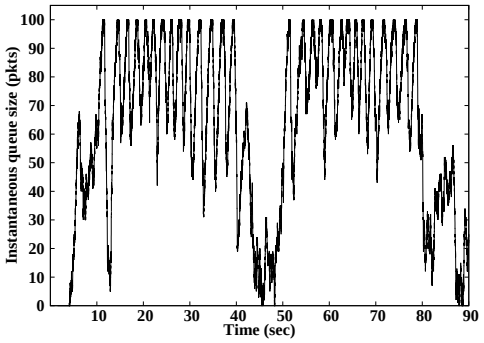

Fig. 9. Queue control: Adaptive RED (oscillating flows)

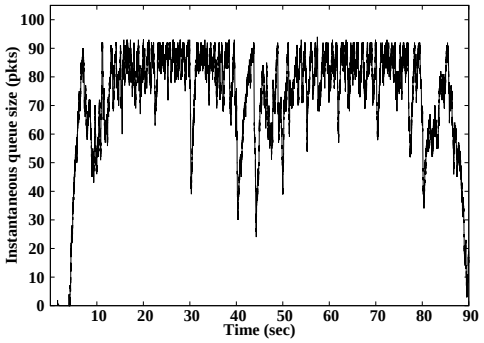

Fig. 10. Queue control: $\mathcal{H}_2$ (oscillating flows)

C. Summary of comparison with other AQM schemes

In this subsection we summarize the benefits of using RQM scheme over RED and Adaptive RED AQM schemes. The foremost advantage of using the proposed RQM scheme is that explicit control over the queue length set-point q_0 can be incorporated into controller implementation; this helps in utilization of the queue buffer for unexpected bursty traffic and/or good throughput. In [27], [28] also[2], there is a similar flexibility in explicitly choosing the set-point, 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: better queue control has been achieved. Good queue length regulation ensures that the delay in the packets is more predictable and controllable. This helps in Quality of Service guarantees when dealing with many Internet applications. Queue length regulation also results in the actual plant model being closer to the nominal plant model due to the round trip time being closer to R_0.

V. Conclusions

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 round trip time delay has been incorporated in a novel manner by discretizing the system and using the discrete time system for the design of a digital controller.

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.

References

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

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

[3] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: an algorithm for increasing the robustness of RED," http://citeseer.ist.psu.edu/floyd01adaptive.html.

[4] T. Ott, T. Lakshman, and L. Wong, "SRED: Stabilized RED," in *Proceedings of 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 *Proceedings of NOSSDAV*, 2001, pp. 41–50.

[6] ——, "A self-configuring RED gateway," in *Proceedings of 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 Transactions on Networking*, vol. 12, pp. 286–299, 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 *Proceedings of 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 Transactions on Automatic Control*, vol. 47, pp. 945–959, June 2002.

[2]We thank an anonymous reviewer for bringing this to our notice.

[11] ——, "A control theoretic analysis of RED," in *Proceedings of INFOCOM*, vol. 3, 2001, pp. 1510–1519.

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

[13] S. Low, F. Paganini, and J. Doyle, "Internet congestion control," *IEEE Control Systems Magazine*, vol. 22, pp. 28–43, December 2004.

[14] P. Quet and H. Ozbay, "On the design of AQM supporting TCP flows using robust control theory," *IEEE Transactions on Automatic Control*, vol. 49, pp. 1031–1036, 2004.

[15] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University 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," *Computer Networks: the International Journal of Computer and Telecommunications Networking*, 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 Transactions on Networking*, vol. 9, pp. 818–832, December 2001.

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

[19] G. Vinnicombe, ""on the stability of end-to-end congeston control for the Internet," University of Cambridge, Technical Report, 2001."

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

[21] M. Vajta, "Some remarks on Padé approximation," in *Proceedings of TEMPUS-INTCOM Symposium*, Veszprém, Hungary, 2000.

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

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

[24] Network Simulator NS, http://nile.wpi.edu/NS/.

[25] LAN/MAN Committee, *IEEE Std 802.11-2007: 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 Computer Society, March 2007.

[26] Matlab, http://www.mathworks.com.

[27] P. Ignaciuk and A. Bartoszewicz, "Linear quadratic optimal discrete time sliding mode controller for connnection oriented communication networks," *IEEE Transactions on Industrial Electronics*, vol. 55, pp. 4013–4021, 2008.

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