Congestion Control for Infrastructure-based CRNs: A Multiple Model Predictive Control Approach

Kefan Xiao, Shiwen Mao, and Jitendra K. Tugnait
Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 34849-5201 USA
Email: kzx0002@auburn.edu, smao@ieee.org, tugnajk@eng.auburn.edu

***Abstract*—In this paper, we investigate the problem of robust congestion control in infrastructure-based cognitive radio networks (CRN). We develop an active queue management (AQM) algorithm, termed MAQ, based on multiple model predictive control (MMPC). The goal is to stabilize the TCP queue at the base station (BS) under disturbances from the varying service capacity for secondary users (SU). The proposed MAQ scheme is validated with extensive simulation studies under various types of background traffic and system/network parameters. It outperforms two benchmark schemes with considerable gains in all the scenarios considered.**

I. Introduction

The wide availability of mobile devices and services has triggered unprecedented demand for wireless capacity, and the demand is predicted to increase drastically in the near future. Cognitive radio network (CRN) is an emerging solution for more capacity by exploiting underutilized licensed spectrum [1]. In CRNs, licensed users (or, Primary User (PU)) share spectrum with CRN users (or, Secondary User (SU)). With the dynamic spectrum access (DSA) approach, SUs detect and access unused licensed channels, where the tension between PU protection and SU capacity gain should be carefully balanced. In particular, the SUs have a lower priority for channel access and their service capacity usually varies over time as affected by the PU transmission behavior.

Although CRN has been well-studied in the past decade, the mainstream research effort has been focused on spectrum sensing, access, and leasing aspects. Some other important network-level problems, such as congestion control, have not been well studied. Since most network applications (e.g., even video streaming from many commercial video-sharing websites) are based on TCP, supporting TCP in CRNs is critical for enabling such applications in CRNs, which is crucial for the success of CRNs.

However, this problem has only been investigated in a few prior works [2]–[4]. In [2], [3], the TCP congestion control mechanism is modified for the CRN environment, while in [4], a partially observable Markov decision process (POMDP) based approach is proposed to maximize the SU throughput by jointly adjusting spectrum sensing, access, and the physical-layer modulation and coding scheme. Although interesting results are derived, the existing schemes are either offline [4] or based on heuristics [2], [3].

Motivated by these interesting works, we investigate the problem of robust congestion control in CRNs. For wireless network, effective solutions have been proposed in the literature, such as split-TCP or making the wireless link more reliable [5]. In CRNs, the challenge is the time-varying capacity for SU TCP sessions due to the activity of PUs, which may even be zero during sensing periods when all the SUs stop transmission to sense the channels. During the transmission period, the capacity that is available for SU TCP sessions also varies over time due to PU transmissions, and the variation could be large and occur at various timescales from session to packet levels. These pose considerable disturbance to the TCP feedback control mechanism. There is a critical need for robust congestion control mechanisms in such a highly dynamic network environment.

In this paper, we consider an infrastructure-based CRN, where multiple SUs maintain TCP connections with various TCP servers in the Internet. The TCP sessions share a buffer at the CRN base station (BS) and the last hop, and bottleneck, of the TCP connections is the downlink of the CRN. The BS advertises a zero-size receive window on behalf of the SUs to inform the TCP sender of the sensing periods and to freeze the TCP server during the sensing period [2]. Since the classic additive-increase-multiple-decrease (AIMD) algorithm does not deal well with the frozen period and capacity variations, we aim to develop new congestion control mechanisms that is robust to such disturbances in this paper.

Specifically, we exploit active queue management (AQM) to deal with the congestion control problem in CRN. AQM is a class of packet dropping/marking mechanisms implemented at the router queue to support end-to-end congestion control. We develop a robust AQM mechanism to stabilize the queue length at the CRN BS, which can not only yield a relative stable queueing delay, but also absorb the disturbances caused by busty background traffic or capacity variation. The proposed scheme, termed MAQ, is based on multiple model predictive control (MMPC) that integrates the estimation and prediction of multiple models with different weights, which can significantly enhance the robustness of the controller to reject disturbances from the environment [6]. We evaluate the performance of the proposed scheme with extensive NS2 simulations, such as under responsive and non-responsive background traffic, varying number of TCP connections, various propagation delays, and various reference queue lengths. The simulation study shows that MAQ can stabilize the TCP buffer in all the scenarios simulated, and outperforms two benchmark schemes with considerable gains.

The remainder of this paper is organized as follows. We

978-1-5090-1328-9/16/$31.00 ©2016 IEEE

review related work in Section II and present the system model in Section III. We discuss the MAQ design in Section IV and validate it in Section V. Section VI concludes this paper.

II. Related Work

This work is closely related to the literature on AQM design in the context of the Internet. The authors in [7] present a control theoretic analysis of random early detection (RED) [8] and find that RED is very sensitive to its parameters. In [9], a Proportional-Integral (PI) AQM scheme is developed that can work under capacity variation. However, the PI scheme reacts slowly to outside disturbances. Model predictive control (MPC) has been used in network congestion control [10], [11]. One important issue is that they do not explicitly take disturbance (e.g., varying service capacity) into consideration.

Chavan, et al. [12] proposed a solution to address the capacity variation due to a fading wireless channel based on the H_∞ technique. However, its conservative policy makes it unsuitable for CRNs where the capacity may change at various/fast timescales. In [13], the authors propose a latency control algorithm, CoDel, which provides a solution to the bufferbloat problem; but the design does not directly apply to CRNs. An AQM algorithm, termed DSM [14], based on discrete sliding mode control (DSMC) theory is proposed to deal with high-frequency oscillation of capacity. It suffers from the over control problem and it is hard to choose a proper control gain.

The problem of enhancing TCP performance in CRNs has been addressed in only a few papers [2]–[4]. In [2], the authors proposed a new network management framework, DSASync, for DSA based WLANs. Its design is largely based on heuristic methods, while smoothing ACKs would slow down the growth of window size during the valuable transmission period. In [3], a window based transport control protocol for CR Ad Hoc Networks is proposed. The proposed protocol is mainly based on heuristics rather than a rigorous theoretic analysis, and the enhancements may not be compatible with existing TCP protocols that have been widely used in both wireless and wireline networks. In [4], a cross-layer approach is proposed based on a POMDP framework and solved by dynamic programming. This is an offline scheme that requires full prior knowledge of all system statistics, while the POMDP based solution has a high complexity.

III. System Model and Problem Formulation

A. Network Model

We consider a primary network with M licensed channels. There is a CRN collocated with the primary network, consisting of a CRN base station (BS) and N active SUs. The CRN BS and SUs cooperatively sense the licensed channels to identify transmission opportunities for SUs. Since spectrum sensing is a well-studied problem, we assume that an effective spectrum sensing scheme is in force and the sensing results are accurate with a high probability.

In the CRN, each active SU maintains a TCP connection with a TCP server in the Internet (e.g., downloading a large file). We assume the channel bonding/aggregation technique is used, such that the BS and SUs can transmit and receive on multiple assigned channels simultaneously to make use of all the available spectrum [15]. Let the aggregated usable capacity be $C(t)$, which is time-varying as availability of the licensed channels vary over time. All the TCP connections share the downlink capacity of the BS with time division multiplexing (TDM), assuming negligible uplink feedback. Due to the mismatch of capacity between wired and wireless links, it is reasonable to assume that the CRN BS is the bottleneck of the TCP connections.

We follow the approach in [2] to freeze the TCP senders during the sensing period, assuming that the TCP servers obtain the sensing schedule of the CRN during the session setup phase and the zero-size receive window mechanism is used [2]. Thus the sensing period has no impact on congestion control except for a fixed small cost to the overall throughput.

B. Fluid Flow Model for TCP Sessions

Ignoring the sensing period of the CRN, during the transmission period, the dynamics of the multiple-TCP system can be modeled as a fluid flow system [16]. Different from many existing AQM algorithms, we adopt the *drop-from-front* mechanism, i.e., when there is incipient congestion, the head-of-line packet in the bottleneck queue will be dropped (instead of the packet at the tail). This way, the TCP server can infer congestion more quickly (i.e., without the queueing delay) and react faster to congestion as well as the changing capacity in the CRN.

The dynamics of the TCP connections sharing the CRN BS queue can be derived by modifying the model in [16] as

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

$$\dot{q}(t) = \begin{cases} N(t)\frac{W(t)}{R(t)} - C(t), & q > 0 \\ \max\left\{0, N(t)\frac{W(t)}{R(t)} - C(t)\right\}, & q = 0, \end{cases} \tag{2}$$

where $\dot{z}(t)$ donates the time-derivative of $z(t)$. At time t,

- $q(t)$ is the queue length at the CRN BS and $q(t) \in [0, B]$;
- $C(t)$ is the downlink capacity of the CRN BS (packets/s);
- $R(t)$ is the RTT of a TCP connection;
- $p(t)$ is the drop probability of AQM;
- T_p is the propagation delay of a TCP connection, i.e., RTT minus queueing delay;
- $N(t)$ is the number of active TCP connections;
- $W(t)$ is the congestion window size of the TCP source;
- β is the window multiplicative decrease parameter, which is usually set to $1/2$ (sometimes as $\ln(2)$).

This model is based on the fluid flow traffic model and is a system of delayed stochastic differential equations. It ignores the TCP time out mechanism to make the problem manageable. Since drop-from-front is adopted, the drop probability p is actually delayed by a propagation delay T_p to take effect at the TCP server.

C. Linearization and Discretization

1) Linearization: We first build the model bank for the proposed MMPC algorithm by choosing *multiple* operating points (each corresponding to a model) and linearizing the system around the operating points.

For this problem, we can obtain the analytic form solution (demonstrated in Section IV). The computation of each model is not time consuming. Thus, the complexity of a multiple models will only increase linearly with the number of models. For the scenarios with very large bandwidth variations, it is safe to use more models to capture the dynamics. However, due to the sound robustness of MMPC, for practical systems, usually three or five models should be sufficient. We choose π different reference capacity values, denoted as C_i, $i = 1, 2, \ldots, \pi$.

The operating points of the system, denoted as five-tuples $(W_i, q_i, p_i, C_i, R_i)$, can be solved from the system equations (1) and (2) by setting $\dot{W} = 0$ and $\dot{q} = 0$. Assuming the system is stabilized at operating point i, we have $W(t) = W(t - T_p)$ and $R(t) = R(t - T_p)$. It follows that

$$\beta W_i^2 p_i = 1; \;\; W_i = \frac{R_i C_i}{N}; \;\; R_i = T_p + \frac{q_i}{C_i}. \tag{3}$$

Denote the deviations from the operating point as $\delta W_i = W(t) - W_i$, $\delta q_i = q(t) - q_i$, $\delta p_i = p(t) - p_i$, and $\delta C_i = C(t) - C_i$. Denote model i state as $x_i = [\delta q_i, \delta W_i]^T$ and output as $y_i = \delta q_i$. Define $\delta p_i(t - T_p)$ as $u_i(t - T_p)$. The linearized system around operating point i can be obtained as follows.

$$\dot{x_i}(t) = A_i x_i(t) + B_i u_i(t - T_p) + D_i \delta C_i(t) \tag{4}$$

$$y_i(t) = Q_i x_i(t) + v(t). \tag{5}$$

Equation (4) is the state equation and (5) is output equation, where $v(t)$ accounts for the disturbance from the uncertainty nature of control input $u_i(t - T_p)$.

2) Discretization: Since packets arrive and depart in a discrete manner, the entire system can be viewed as sampling from the continuous-time model with a finite sampling rate. Denote the sampling period as T_s, which is a constant and sufficiently small, such that every propagation delay T_p is an integer multiple of T_s as $T_p = n_0 T_s$.

Let $x[k]$ represent the kth sample $x(kT_s)$. Then the ith linear model (around operating point i) can be discretized as

$$x_i[k+1] = \Phi_i x_i[k] + \Gamma_i u_i[k - n_0] + \Delta_i \delta C_i(k) \tag{6}$$

$$y_i[k+1] = Q_i x_i[k+1] + v_i[k+1]. \tag{7}$$

The new coefficients Φ_i, Γ_i, and Δ_i can be derived with the standard discretization algorithm.

We also consider $\delta C[k]$, the variation of link capacity, as a state variable, which accounts for both capacity changing and other disturbances and evolves as follows.

$$\delta C[k] = \delta C[k-1] + \omega[k-1], \tag{8}$$

where $\omega[\cdot]$ is the external disturbance. Due to the uncertainty such as modeling error or unknown disturbance, it is usually hard, if not impossible, to drive the system to converge to the operating point without an offset. In order to ensure convergence, augmenting the system state with the disturbance is a feasible approach, since it would be possible to estimate the disturbance and then suppress its impact. The augmented state is $x_i^a[k] = [\delta q_i, \delta W_i, \delta C_i]^T$. Finally, the discretized system can be rewritten as

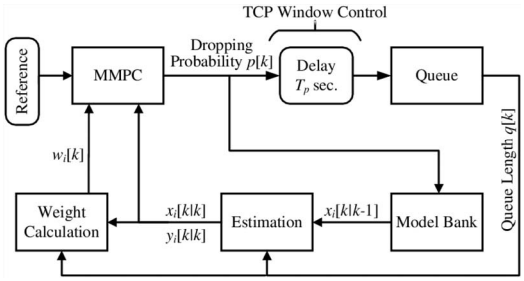

Fig. 1. The control system structure.

$$x_i^a[k+1] = \Phi_i^a x_i^a[k] + \Gamma_i^a u[k - n_0] + \Theta_i^a w[k] \tag{9}$$

$$y_i[k] = Q_i^a x_i^a[k] + v_i[k], \tag{10}$$

where the coefficient matrices are

$$\Phi_i^a = \begin{bmatrix} \Phi_i & \Delta_i \\ 0 & I \end{bmatrix}, \; \Gamma_i^a = \begin{bmatrix} \Gamma_i \\ 0 \end{bmatrix}, \; \Theta_i^a = \begin{bmatrix} 0 \\ I \end{bmatrix}, \; Q_i^a = [Q_i, 0].$$

IV. MAQ Design

The design of the MAQ scheme is presented in this section. Due to the propagation delay T_p, the control output, i.e., the dropping probability p, will take effect at the TCP server after T_p. The entire control system is shown in Fig. 1. When there is incipient congestion, packets are dropped from front of the BS queue with a certain probability at the CRN BS.

A. Estimation

In the model bank we developed, the noise is usually not known in practice. It is necessary to utilize the previous information to estimate both the augmented state and output. Let $q[k]$ be the measured queue length at step k. We use Kalman Filter for state and output estimation as follows.

$$\hat{x}_i^a[k|k-1] = \Phi_i^a \hat{x}_i^a[k-1|k-1] + \Gamma_i^a u_i[k - n_0 - 1] \tag{11}$$

$$\hat{x}_i^a[k|k] = \hat{x}_i^a[k|k-1] + L_i[k](q[k] - q_i - Q_i^a \hat{x}_i^a[k|k-1]) \tag{12}$$

$$\hat{y}_i^a[k|k] = Q_i^a \hat{x}_i^a[k|k], \tag{13}$$

where $\hat{x}_i^a[k|k-1]$ is the estimation of step k based on step $(k-1)$ information, $\hat{x}_i^a[k|k]$ represents the updated state at step k based on the measurement $q[k]$ at time step k, and $L_i[k]$ is the Kalman gain of model i. The "hat" notation indicates that they are estimations. Eq. (13) is the output estimation. The Kalman gain $L_i[k]$ can be derived from the following process.

$$L_i[k] = P_i[k](Q_i^a)^T (Q_i^a P_i[k-1](Q_i^a)^T + G_i)^{-1} \tag{14}$$

$$P_i[k] = \Phi_i^a P_i[k-1](\Phi_i^a)^T + \Theta_i^a O_i (\Theta_i^a)^T - \Phi_i^a P_i[k-1](Q_i^a)^T (Q_i^a P_i[k-1](Q_i^a)^T + G_i)^{-1} Q_i^a P_i[k-1](\Phi_i^a)^T, \tag{15}$$

where $P_i[k]$ is the estimation error covariance of model i at step k.

This algorithm works in an iterative way. Parameters O_i and G_i in (15) are stochastic terms that account for the variance of state interference and output, respectively.

B. Model Weight Calculation

The model weights are calculated by normalizing the probability of each model that represents the real model, as

$$w_i[k] = \begin{cases} \frac{\rho_i[k]}{\sum_{j=1}^{\pi} \rho_j[k]}, & \rho_i[k] > \mu \\ 0, & \rho_i[k] \leq \mu. \end{cases} \tag{16}$$

With the general assumption that the weights are Gaussian, the probability that model i represents the system at step k can be derived with the Bayesian theorem as

$$\rho_i[k] = \frac{\exp(-0.5\lambda\xi_i^2[k])\rho_i[k-1]}{\sum_{j=1}^{\pi} \exp(-0.5\lambda\xi_j^2[k])\rho_j[k-1]}. \tag{17}$$

in which $\xi_i[k] = q[k] - q_i - \hat{y}_i[k|k]$. In the initialization phase, all the weights are set to $1/\pi$ since there is no a priori information about the system. In (17), the exponential term is beneficial that the rejecting speed of the unsuitable models is exponential. The coefficient λ determines the sensitivity to the residual.

C. MMPC Control Law

We are now ready to derive the MMPC control law. Due to the propagation delay, the control signal $u_i[k]$ affects the queue length at time $(k + n_0)$. It is thus necessary to estimate the state and output at time $(k+n_0)$. We estimate state $x_i^a[k+n_0]$ as

$$\begin{aligned} \hat{x}_i^a[k + n_0|k] &= \Phi_i^a \hat{x}_i^a[k + n_0 - 1|k] + \Gamma_i^a u_i[k-1] \\ &= (\Phi_i^a)^{n_0}\hat{x}[k|k] + \sum_{j=1}^{n_0} (\Phi_i^a)^{j-1}\Gamma_i^a u_i[k-j]. \end{aligned} \tag{18}$$

The $(k + n_0 + j)$th step prediction is the weighted average of all model outputs at the same step, as

$$\bar{y}[k + n_0 + j|k] = \sum_{i=1}^{\pi} w_i[k]\hat{y}_i[k + n_0 + j|k] \tag{19}$$

The objective is to stabilize the queue length around the set point while keeping the difference between each control move as small as possible. The first goal is obvious, and the reason for the second one would be explained in the later section. We define the objective function as

$$\min \ (Y_{ref} - \bar{Y})^T W_y (Y_{ref} - \bar{Y}) + \Delta U^T W_u \Delta U, \tag{20}$$

where Y_{ref} is the modified reference output based on the reference queue length. The reference queue length is given as q_{ref} and the estimated queue length is $\hat{q}[k + n_0 + j]$. It follows that

$$Y_{ref} = \begin{bmatrix} q_{ref} - \sum_{i=1}^{\pi} w_i[k]q_i \\ q_{ref} - \sum_{i=1}^{\pi} w_i[k]q_i \\ \vdots \\ q_{ref} - \sum_{i=1}^{\pi} w_i[k]q_i \end{bmatrix}. \tag{21}$$

Further, $\bar{Y}$ is a vector of the weight estimations of the model outputs over the prediction horizon and ΔU is a vector of dropping probability increments over the control horizon, i.e.,

$$\bar{Y} = \begin{bmatrix} \bar{y}[k + n_0 + 1|k] \\ \bar{y}[k + n_0 + 2|k] \\ \vdots \\ \bar{y}[k + n_0 + s|k] \end{bmatrix}, \ \Delta U = \begin{bmatrix} \Delta p[k] \\ \Delta p[k+1] \\ \vdots \\ \Delta p[k+m] \end{bmatrix}. \tag{22}$$

Finally, W_y and W_u are diagonal weight matrices of queue length differences and control moves, respectively.

In (20), the first term minimizes the difference between the real queue length and reference queue length. The second term in the objective function is the penalty for changing the dropping probability. The optimization variables of problem (20) is ΔU, which can be solved as

$$\begin{aligned} \Delta U = (M_e^T M_c^{aT} W_y M_c^a M_e + W_u)^{-1} M_e^T M_c^{aT} \\ W_y(\tilde{Y}_{ref} - M_x^a - M_c^a U_0), \end{aligned} \tag{23}$$

where $\tilde{Y}_{ref} = Y_{ref} + M_{cp}$. The matrices in (23) are given as

$$M_x^a = \sum_{j=1}^{\pi} w_j[k] \begin{bmatrix} Q_j^a(\Phi_j^a) \\ \vdots \\ Q_j^a(\Phi_j^a)^m \end{bmatrix} \hat{x}_j^a[k + n_0|k]$$

$$M_e = \begin{pmatrix} 1 & & 0 \\ \vdots & \ddots & \\ 1 & \cdots & 1 \end{pmatrix}, \ U_0 = \begin{bmatrix} p[k-1] \\ \vdots \\ p[k-1] \end{bmatrix}$$

$$M_c^a = \sum_{j=1}^{\pi} w_j[k] \begin{bmatrix} Q_j^a\Gamma_j^a & & 0 \\ \vdots & \ddots & \\ Q_j^a(\Phi_j^a)^{m-1}\Gamma_j^a & \cdots & Q_j^a\Gamma_j^a \end{bmatrix}$$

$$M_{cp}^a = \sum_{j=1}^{\pi} w_j[k] \begin{bmatrix} Q_j^a\Gamma_j^a & & 0 \\ \vdots & \ddots & \\ Q_j^a(\Phi_j^a)^{m-1}\Gamma_j^a & \cdots & Q_j^a\Gamma_j^a \end{bmatrix} \begin{bmatrix} p_j \\ \vdots \\ p_j \end{bmatrix},$$

where M_{cp} is the compensation for the deviation from operation point p_i of each model i.

In the proposed MMPC scheme, the computation complexity is linear with the control horizon length. It's worth to notice that there are several parameters that need to be tuned to achieve good performance. The two weight matrices W_y and W_u in the objective function are directly related to the optimal control signal. The control move is the dropping probability in the range of $[0, 1]$. We can use the saturation function to enforce this range. To avoid offset or unstability, it is necessary to choose a large W_u. Usually, the larger the ratio W_y/W_u, the more aggressive the control move. For the CRN TCP system, we recommend $W_y = \text{diag}(1, 1, \ldots, 1)$ and $W_u = \text{diag}(10^6, 10^6, \ldots, 10^6)$. In addition, parameters of Kalman Filter estimation will be based on the recommendation of standard procedure. The coefficient λ, on the other hand, determines the sensitivity of weights to estimation error.

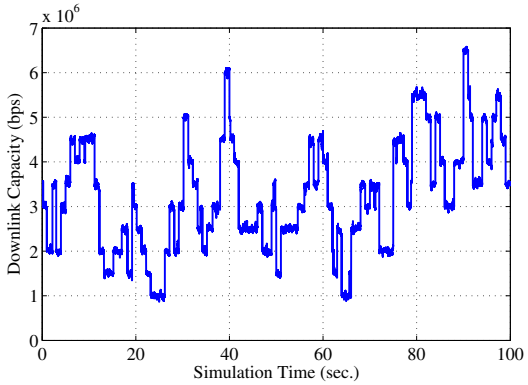

Fig. 2. The CRN downlink capacity over time used in one of the simulations.

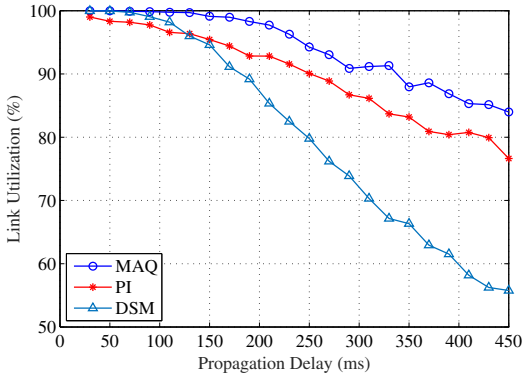

Fig. 3. Average link utilization versus propagation delay.

V. Simulation Study

In this section, we evaluate the performance of MAQ with NS2 simulations. We extend the NS2 simulator with the multiple-channel extension, TDM module, and channel bonding/aggregation. We assume 10 licensed channels: five of them with a capacity of 500 Kbps and the other five with a 1 Mbps capacity, to account for heterogeneous conditions. As in prior work [3], each channel is modeled as an on-off process. Both the on and off periods are exponentially distributed with mean $\eta(\text{on}) = 4$ seconds and $\eta(\text{off}) = 5$ seconds, respectively. The sensing phase of the CRN is 20 ms and the transmission phase is 200 ms with periodical sensing. If there is no channel available after the sensing period, the frozen state will continue until there is an idle channel.

The CRN maintains a large number of TCP connections. The RTT and propagation delay are key factors influencing the performance of congestion control. In real networks, different users may have different propagation delays. So we choose propagation delays uniformly distributed in $(100, 150)$ ms (unless otherwise stated). The capacity of the wireline link between the BS to a TCP server is 20 Mbps. The average capacity for the CRN downlink is $\bar{C} = 3.2$ Mbps with variance $\sigma_C = 1.3$ Mbps. The capacity used in one of the simulations is shown in Fig. 2. The packet size is 540 Bytes with 500 Bytes data and 40 Bytes header. The reference queue length is set to 90 packets, corresponding to a queueing delay around 100 ms. The queue size is set to 500 packets (about 2 Mb).

For comparison purpose, we also simulated two existing control-theoretic schemes: (i) the proportional integral (PI) scheme, which is canonical and has been shown to be effective on dealing with varying capacities [9]; (ii) the DSM scheme, which is a most recent advance that is based on the principles of discrete sliding-mode control [14]. We set the PI parameters as: 4.839×10^{-5} and 4.346×10^{-5}, as recommended in [9]. The DSM parameters are set as recommended in [14], with $\gamma = 0.01$ and a sigmoid function with $\delta = 10$ is used. We test the proposed scheme under various parameter settings.

We analyze the impact of different propagation delay and reference queue length on the congestion control performance. We vary the average propagation delay from 30 ms to 450 ms in steps of 20 ms in a series of simulations. In each simulation, the propagation delay of each SU is the average plus a disturbance uniformly chosen from $[0, 50]$ ms. We also set different reference queue length in $[10, 115]$ packets, which is indicative of different queueing delays.

In the simulations, we compare the link utilization and delay performance of MAQ with that of PI and DSM, which is shown to be robust against system condition variations [14]. Fig. 3 shows the link utilization of each scheme under different propagation delays. A large propagation delay has a negative impact on all the three algorithms, since it is harder to predict future states and outputs. When the RTT is larger than the average capacity changing period, it will be impossible to properly control the queue. This is because when a control signal takes effect on the queue, the service capacity has already changed to some other value. We find that MAQ can still maintain a considerably higher throughput than both PI and DSM, and the DSM performance degrades more quickly than MAQ and PI with increasing propagation delays.

Reference queue length is also critical for congestion control, since a large buffer could mitigate the effect of capacity variation. The link utilization of the three schemes for increasing reference queue lengths are presented in Fig. 4. Again, MAQ achieves considerable higher link utilization than the other two schemes, especially when the reference queue length is small. For example, when the reference queue length is 10 packets, the MAQ link utilization about 20% higher than that of PI and 70% higher than that of DSM. When the reference queue length becomes large, all the three schemes achieve a high link utilization. However, as can be seen later in Fig. 6, PI and DSM achieve the high link utilization at the cost of higher delay and large delay variation than MAQ.

The queueing delay performance under different propagation delay and reference queue length are plotted in Fig. 5 and 6, respectively. We find that the average delay of MAQ is not only much smaller than PI (e.g., around 50% of that of PI), but also much stable over the entire range of propagation delay values. DSM exhibits better performance on controlling queuing delay than PI. Although its queueing delay is low

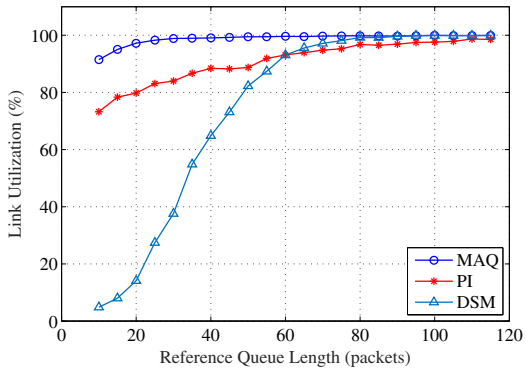

Fig. 4. Link utilization versus reference queue length.

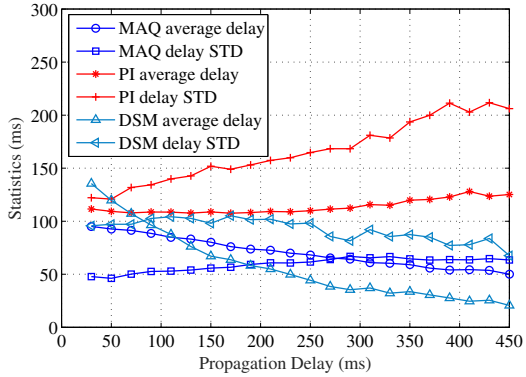

Fig. 5. Average delay and delay STD versus propagation delay.

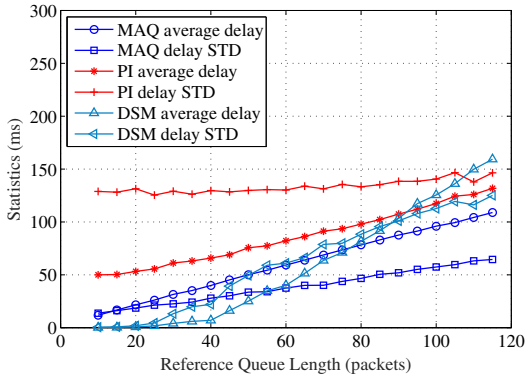

Fig. 6. Average delay and delay STD versus reference queue length.

when propagation delay is large, this is achieved at the cost of a low throughput, since its queue is empty more often than the other two schemes.

Under different reference queue lengths, the average queuing delay goes up roughly linearly since it is around $q_{ref}/\bar{C}$, as shown in Fig. 6. The DSM has very small average delay when the reference queue length is lower than 40 packets. As discussed, this is achieved at the cost of very low link utilization (see Fig. 4). MAQ not only achieves low queueing delay and delay variation, the two MAQ curves are also less sensitive to the increasing reference queue lengths than the other two schemes.

VI. Conclusion

In this paper, we designed a congestion control scheme for infrastructure-base CRNs. The goal was to stabilize the TCP buffer at the CRN BS under a wide range of system/network parameters and dynamics. The proposed scheme, MAQ, was based on MMPC with enhanced state and output estimation. The proposed scheme was evaluated with extensive NS2 simulations under various background traffic types and network/system parameter settings. It was shown to achieve superior performance over two benchmark schemes.

Acknowledgment

This work is supported in part by the US National Science Foundation (NSF) under Grant CNS-0953513, and by the Wireless Engineering Research and Education Center (WEREC) at Auburn University.

References

[1] Y. Zhao, S. Mao, J. O. Neel, and J. H. Reed, "Performance evaluation of cognitive radios: Metrics, utility functions, and methodology," *Proc. IEEE*, vol. 97, no. 4, pp. 642–659, Apr. 2009.

[2] A. Kumar and K. G. Shin, "Managing TCP connections in dynamic spectrum access based wireless LANs," in *Proc. IEEE SECON 2010*, Boston, MA, June 2010, pp. 1–9.

[3] K. R. Chowdhury, M. Di Felice, and I. F. Akyildiz, "TP-CRAHN: A transport protocol for cognitive radio ad-hoc networks," in *Proc. IEEE INFOCOM 2009*, Rio de Janeiro, Brazil, Apr. 2009, pp. 2482–2490.

[4] C. Luo, F. R. Yu, H. Ji, and V. C. Leung, "Cross-layer design for TCP performance improvement in cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 59, no. 5, pp. 2485–2495, 2010.

[5] X. Chen, H. Zhai, J. Wang, and Y. Fang, *A Survey on Improving TCP Performance over Wireless Networks*. Kluwer Academic Publishers, 2005, pp. 657–695.

[6] M. Kuure-Kinsey and B. W. Bequette, "Multiple model predictive control strategy for disturbance rejection," *Ind. Eng. Chemistry Research*, vol. 49, no. 17, pp. 7983–7989, 2010.

[7] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "A control theoretic analysis of RED," in *Proc. IEEE INFOCOM 2001*, Anchorage, AK, Apr. 2001, pp. 1510–1519.

[8] 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.

[9] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "On designing improved controllers for AQM routers supporting TCP flows," in *Proc. IEEE INFOCOM 2001*, Anchorage, AK, Apr. 2001, pp. 1726–1734.

[10] X.-H. Zhang, K.-S. Zou, Z.-Q. Chen, and Z.-D. Deng, "Stability analysis of AQM algorithm based on generalized predictive control," in *Advanced Intelligent Computing Theories and Applications*, ser. LNCS. Springer, 2008, vol. 5226, pp. 1242–1249.

[11] P. Wang, H. Chen, X. Yang, and Y. Ma, "Design and analysis of a model predictive controller for active queue management," *Elsevier ISA transactions*, vol. 51, no. 1, pp. 120–131, 2012.

[12] K. Chavan, R. G. Kumar, M. N. Belur, and A. Karandikar, "Robust active queue management for wireless networks," *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 6, pp. 1630–1638, 2011.

[13] K. Nichols and V. Jacobson, "Controlling queue delay," *Commun. ACM*, vol. 55, no. 7, pp. 42–50, 2012.

[14] P. Ignaciuk and M. Karbowańczyk, "Active queue management with discrete sliding modes in TCP networks," *Bull. Pol. Ac.: Tech.*, vol. 62, no. 4, pp. 701–711, Dec. 2014.

[15] H. Salameh, M. Krunz, and D. Manzi, "Spectrum bonding and aggregation with guard-band awareness in cognitive radio networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 3, pp. 569–581, Mar. 2014.

[16] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *ACM SIGCOMM Comput. Commun. Rev.*, vol. 30, no. 4, 2000, pp. 151–160.