caltechCSTR:2002.008

Cost of AQM in stabilizing TCP *

Ki Baek Kim and Steven H. Low

Departments of CS and EE
California Institute of Technology
kkb@cisl.snu.ac.kr, slow@caltech.edu

Abstract

In this paper, we propose a unified mathematical framework based on receding horizon control for analyzing and designing AQM (Active Queue Management) algorithms in stabilizing TCP (Transfer Control Protocol). The proposed framework is based on a dynamical system of the given TCP and a linear quadratic cost on transients in queue length and flow rates. We derive the optimal receding horizon AQMs (RHAs) that stabilizes the linearized dynamical system with the minimum cost. Conversely, we show that any AQM with an appropriate structure solves the same optimal control problem with appropriate weighting matrix. We interpret existing AQM's such as RED, REM, PI and AVQ as different approximations of the optimal AQM, and discuss the impact of these approximations on performance.

1 Introduction

Congestion control is a distributed iterative procedure to share network resources among competing sources. It consists of local algorithms executed dynamically at sources (TCP) and at links (active queue management, or AQM). Links update, implicitly or explicitly, a measure of congestion, and feed it back to sources, by dropping or marking arrival packets. In response, sources adjust their rates based on the feedback information from links in its path. Popular TCP algorithms include Reno (and its variants) and Vegas, and popular AQM algorithms include DropTail, RED [3] and its variants. Recently, new TCP algorithms have been proposed in [7], [9], [8], [14], [13], and new AQMs have been proposed, e.g., Adaptive Virtual Queue (AVQ) [15], REM [1], and PI controller [6], etc.

Internet probably represents the largest engineered feedback system ever deployed. We wish to understand fundamental properties, both equilibrium and dynamic structures, of a network where sources and links interact according to a TCP/AQM algorithm pair. It turns out that one can understand the equilibrium structure of such a system by regarding TCP/AQM as a distributed primal-dual algorithm carried out over the Internet in real time by sources and links, to maximize aggregate utility [17]. Given (almost) any TCP algorithm, one can derive the

*We submitted this paper to Sigcomm, Feb. 1, 2002 (Paper ID: 150) but withdrew it because of many typographical errors. The corresponding author, Ki Baek Kim is visiting Caltech from Engineering Research Center, Seoul National University, Korea utility function that it implicitly optimize. What is more interesting is that the utility functions depends *solely* on the TCP algorithm and not on the AQM algorithm. As long as the AQM algorithm satisfies the property (condition C3 below) that input flow rate at a bottleneck link is matched to the link capacity, the equilibrium of the TCP/AQM *pair* will maximize the aggregate utility, with utility functions determined only by the TCP algorithm. The solution of the utility optimization determines equilibrium bandwidth allocation, performance, and fairness. Hence we can interpret the design of TCP algorithm as choosing the equilibrium operating points. These results are reviewed in Section 2.

The goal of AQM, we will argue in this paper, is then to stabilize these equilibrium points. Indeed, it has been shown that RED parameters can be tuned to stabilize TCP Reno, but at a cost in terms of equilibrium queue and of transient response [6, 19]. In this paper, we propose a model based on receding horizon control [22, 12, 2, 4, 16, 10], that formalize these ideas. We will derive the optimal AQM to stabilize a given TCP, here, focusing on TCP Reno. This optimal AQM is not implementable as it requires global information that will not be available in practice. However, it serves as a performance limit to practical AQM's. We will interpret existing AQM proposals as different approximations to the optimal AQM, as a way to understand their respective properties. These are developed in Sections 3 and 4.

While the duality model provides a unified framework to understand different TCP algorithms, there lacks a similar model to compare and understand various AQM's. As a result, AQM algorithms are only compared in the literature through simulations. Our model represents a first step towards developing a mathematical model for the systematic analysis and synthesis of AQMs.

2 A duality model of TCP/AQM

In this section, we describe a general model for congestion control that allows us to study the equilibrium structure, the dynamics and stability of TCP/AQM for arbitrary network topology, routing and delays. We then summarize some recent advances within this model, mainly concerning the equilibrium structure for general network and local stability for a single link. This motivates, and put in context, the subject of this paper.

A network is modeled as a set L of links (scarce resources) with finite capacities $c = (c_l, l \in L)$. They are shared by a set S of sources indexed by s. Each source s uses a set $L_s \subseteq L$ of links. The sets L_s define an $L \times S$ routing matrix[1]

$$R_{ls} = \begin{cases} 1 & \text{if } l \in L_s \\ 0 & \text{otherwise} \end{cases}$$

Associated with each source s is its transmission rate $x_s(t)$ at time t, in packets/sec. Associated with each link l is a scalar congestion measure $p_l(t) \geq 0$ at time t. Following the notation of [21], let

$$y_l(t) = \sum_s R_{ls} x_s(t - \tau_{ls}^f) \tag{1}$$

be the aggregate source rate at link l at time t, where τ_{ls}^f is the (equilibrium) forward delays from sources s to link l, which are assumed constant. Let

$$q_s(t) = \sum_l R_{ls} p_l(t - \tau_{ls}^b) \tag{2}$$

[1]We abuse notation to use L and S to denote sets and their cardinalities.

be the end-to-end congestion measure for source s, where τ^b_{ls} are the (equilibrium) backward delays from links l to source s, assumed constant.

TCP is modeled by a function F_s that specifies how source rate $x_s(t)$ is adjusted in response to end-to-end congestion measure $q_s(t)$:

$$\dot{x}_s(t) \quad = \quad F_s(x_s(t), q_s(t)) \tag{3}$$

Note that F_s does not depend on other source rates nor congestion measure not on its path. Different TCP algorithms are modeled by different F_s functions. AQM is modeled by functions (G_l, H_l) that describes how congestion measure $p_l(t)$ is updated, implicitly or explicitly, based on the aggregate flow rate $y_l(t)$ and possibly some internal variables $v_l(t)$:

$$\dot{p}_l(t) = G_l(y_l(t), v_l(t)) \tag{4}$$
$$\dot{v}_l(t) = H_l(y_l(t), v_l(t)) \tag{5}$$

Different protocols use different metrics as congestion measures [17]; e.g., Reno uses loss probability as a congestion measure, and Vegas uses queueing delay. We will often refer to an AQM by G_l, *without explicit reference* to the internal variables $v_l(t)$ and their adaptation H_l.

In summary, a TCP/AQM protocol pair is modeled by a certain $(F, G) = (F_s, G_l, s \in S, l \in L)$. We now look at how the system (3–5) behave, in equilibrium and during transient.

2.1 TCP F_s: maximize utility

The equilibrium structure of (3–5) depends largely on the TCP functions F_s in (3). Equilibrium properties include performance metrics, such as throughput (equilibrium rates), average loss and delay, and fairness (property of the equilibrium rate vector). We show in [17] that these properties can be understood by interpreting (F, G) as distributed primal-dual algorithms over the Internet to solve a global optimization problem, where the objective function depends only on F_s. We summarize this result here.

Consider an equilibrium (x, p) of (3–4). The fixed point of (3) defines an implicit relation between equilibrium rate x_s and end-to-end congestion measure q_s:

$$x_s \quad = \quad F_s(x_s, q_s)$$

Assume F_s is continuously differentiable and $\partial F_s / \partial q_s \neq 0$. Then, by the implicit function theorem, there exists a unique continuously differentiable function f_s such that

$$q_s \quad = \quad f_s(x_s) \; > \; 0 \tag{6}$$

Define the utility function of each source s as

$$U_s(x_s) \quad = \quad \int f_s(x_s) dx_s, \qquad x_s \geq 0 \tag{7}$$

that is unique up to a constant. Being an integral, U_s is a continuous function. Since $f_s(x_s) = q_s \geq 0$ for all x_s, U_s is nondecreasing. It is reasonable to assume that f_s is a nonincreasing function – the more severe the congestion, the smaller the rate. This implies that U_s is concave. If f_s is *strictly* decreasing, then U_s is strictly concave since $U''_s(x_s) < 0$. An increasing utility function U_s implies a greedy source – a larger rate yields a higher utility – and concavity implies diminishing return.

Now consider the problem of maximizing aggregate utility:

$$\max_{x \geq 0} \quad \sum_s U_s(x_s) \;\; \text{subject to } Rx \leq c \tag{8}$$

The constraint says that, at each link l, the flow rate y_l does not exceed the capacity c_l. An optimal rate vector x^* exists since the objective function in (8) is continuous and the feasible solution set is compact. It is unique if U_s are *strictly* concave. The key to understanding the equilibrium of (3–5) is to *regard* $x(t)$ *as primal variables,* $p(t)$ *as dual variables, and* $(F, G) = (F_s, G_l, s \in S, l \in L)$ *as a distributed primal-dual algorithm to solve the primal problem (8) and its Lagrangian dual* (see [18]):

$$\begin{aligned} \min_{p \geq 0} \quad D(p) \quad &:= \sum_s \max_{x_s \geq 0} (U_s(x_s) - x_s q_s) \\ &+ \sum_l p_l c_l \end{aligned} \tag{9}$$

Hence, the dual variable is a precise measure of congestion in the network. We will interpret the equilibrium (x^*, p^*) of (3–5) as the solutions of the primal and dual problem, and that (F, G) iterates on both the primal and dual variables together in an attempt to solve both problems.

We summarize the assumptions on (F, G, H):

C1: For all $s \in S$ and $l \in L$, F_s and G_l are non-negative functions.

C2: For all $s \in S$, F_s are continuously differentiable and $\partial F_s / \partial q_s \neq 0$; moreover, f_s in (6) are nonincreasing.

C3: If $p_l = G_l(y_l, p_l, v_l)$ and $v_l = H_l(y_l, p_l, v_l)$, then $y_l \leq c_l$ with equality if $p_l > 0$.

C4: For all $s \in S$, f_s are strictly decreasing.

Condition C1 guarantees that $(x(t), p(t)) \geq 0$ and $(x^*, p^*) \geq 0$. C2 guarantees the existence and concavity of utility function U_s. C3 guarantees the primal feasibility and complementary slackness of (x^*, p^*). Finally condition C4 guarantees the uniqueness of optimal x^*.

Theorem 1 ([17]) *Suppose assumptions C1 and C2 hold. Let* (x^*, p^*) *be an equilibrium of (3–4). Then* (x^*, p^*) *solves the primal (8) and the dual problem (9) with utility function given by (7) if and only if C3 holds. Moreover, if assumption C4 holds as well, then* U_s *are strictly concave and the optimal rate vector* x^* *is unique.*

Hence, various TCP/AQM protocols can be modeled as different distributed primal-dual algorithms (F, G, H) to solve the global optimization problem (8) and its dual (9), with different utility functions U_s. This computation is carried out by sources and links over the Internet in real time in the form of congestion control. Theorem 1 characterizes a large class of protocols (F, G, H) that admits such an interpretation. This class includes, in particular, TCP Reno, TCP Vegas, REM, PI, AVQ, etc.

Example 2: Utility functions of Reno and Vegas

It is shown in [8, 17] that the utility function of Reno, and its variants such as NewReno and SACK, is

$$U_s(x_s) \quad = \quad \frac{\sqrt{2}}{\tau_s} \tan^{-1}\left(\frac{x_s \tau_s}{\sqrt{2}}\right)$$

The utility function of Vegas is [20]

$$U_s(x_s) \quad = \quad \alpha_s d_s \log x_s$$

Since both utility functions are strictly concave, the equilibrium rate vector is unique under either Reno or Vegas. The log utility function of Vegas implies that Vegas achieves weighted proportional fairness [9]. ■

2.2 AQM G_l: minimize stabilization cost

The equilibrium structure of (3–5) depends largely on TCP functions F_s, in the sense that the underlying optimization problem (8) are defined solely by F_s. As long as the AQM functions G_l satisfy condition C3, an equilibrium (x^*, p^*) will be primal-dual optimal. But C3 says that aggregate flow rate in equilibrium is equalized to link capacity at every bottleneck link, which is satisfied by any practical AQM that stabilizes the queue, e.g., RED, REM [1], PI [5] and AVQ [13], etc. *Hence we can interpret the choice of TCP functions F_s as designing the equilibrium structure (e.g., bandwidth allocation and fairness), and the role of AQM functions G_l as stabilizing the equilibrium points.* This view is taken by [5] and extended in [19].

More concretely, the analysis in [19] shows that the stability of TCP/AQM relies on bounding a convex set of the form $K \cdot C$ to the right of $(-1, 0)$ in the complex plane. Here, K is a constant gain and C is a convex set in the complex plane that contains the origin. Hence, stability can be guaranteed if K is sufficiently small. The gain K and the set C depend on both the TCP functions F_s and the AQM functions G_l. For instance, for the case of a single link with capacity c shared by N identical sources with delay τ, the overall gain is a product of two factors, $K = K_{tcp} \cdot K_{aqm}$, one due to TCP and the other due to AQM. TCP (together with network delay) contributes a factor

$$K_{tcp} \quad = \quad \frac{c^2 \tau^2}{2N} \tag{10}$$

to the overall gain K. This high gain (10) is mainly responsible for instability of TCP Reno at high delay τ, high capacity c, or low load N. AQM compensates for these effects by scaling down the TCP gain (and reshaping the set C). With RED, for instance,

$$K_{aqm} \quad = \quad \frac{c\tau\alpha\rho}{1-\beta}$$

where α and ρ are RED parameters and $\beta \in (0, 1)$ is a characteristic of the link. Specifically α is the weight in queue averaging and $\rho =$ (`max_p/(max_th-min_th)`) is the slope of RED marking probability function, as a function of average queue length. Hence to scale down K and stabilize TCP, RED must keep the product $\alpha\rho$ small. A small α leads to a sluggish response as current queue length is incorporated into the marking probability very slowly. A small ρ leads to a large equilibrium queue length. Note that adapting the RED parameter `max_p` dynamically with fixed `max_th` and `min_th` is equivalent to changing ρ, and hence it cannot avoid the inevitable choice between stability (requiring small ρ) and performance (requiring large ρ). This is the cost of RED in stabilizing TCP Reno. Different AQM algorithms, such as REM [1], PI [5], AVQ [13], can also be tuned to stabilize TCP, at different costs.

This view leads to the natural questions of what the 'optimal' AQM G_l is to stabilizing a given TCP function F_s, and how different AQM functions G_l can be compared. The purpose of this paper is to propose a model within which these questions can be rigorously studied.

The basic idea is to treat TCP F_s as a dynamical system with congestion measure $p(t)$ as its control input. The problem of optimal AQM design is to choose an input that stabilizes TCP with the minimum cost. In this paper, we study a simplified version of this problem, simplified in three regards. First, we consider the linearized version of (3), so the variables denote perturbations around an equilibrium and the cost measures the deviation from the equilibrium point. For example, a slower transient will incur a higher cost. Second, we consider the case of a single link, so $q_s(t) = p_l(t) = p(t)$. Finally, instead of general TCP functions F_i, we focus on TCP Reno.

3 Receding horizon formulation of AQM design

In this section, we describe a unified model, based on receding horizon control, to analyze and synthesize AQM algorithms. In the next two sections, we derive the structure of the optimal stabilizing AQM, in the sense of minimizing the transient around an equilibrium, and interpret existing AQMs, such as RED, REM/PI, and AVQ, within this model as different approximations to the optimal AQM.

Consider the simple case of a single link with capacity c shared by N TCP Reno sources with identical delay. As in [5], we assume forward delay $\tau^f = 0$ so that the equilibrium round trip time is $\tau = \tau^b$. Let $w(t)$ be the common equilibrium window of each source at time t. The common source rate is then defined as $x(t) = w(t)/(d + b(t)/c)$ where d is the common end-to-end propagation delay, and $b(t)$ is the queue occupancy at the link. Let (w^*, b^*, p^*) be the equilibrium point. Then τ is related to b^* by $\tau = d + b^*/c$. The linearized model of TCP Reno (or its variants such as NewReno and SACK) derived in [19] is

$$\delta\dot{w}(t) = -x^* p^* \delta w(t) - \frac{1}{\tau p^*}\delta p(t-\tau) \tag{11}$$

$$\delta\dot{b}(t) = N\frac{\delta w(t)}{\tau} - \frac{1}{\tau}\delta b(t) \tag{12}$$

where $(\delta w(t), \delta b(t), \delta p(t))$ are perturbations around the equilibrium (w^*, b^*, p^*). The equilibrium quantities are given by

$$x^* = \frac{c}{N}, \quad w^* = \tau x^*, \quad p^* = \frac{2N^2}{2N^2 + (\tau c)^2}$$

and b^* depends on the AQM employed.

Note that given $(\delta b(t), \delta\dot{b}(t))$, the window dynamics $\delta w(t)$ can be obtained from (12). Hence we do not need to include $\delta w(t)$ in the state. Instead, we use $(\delta b(t), \delta\dot{b}(t), \delta\ddot{b}(t))$ as the state variable; we will see below that this can be used to model various AQM's, including RED, REM/PI and AVQ, as special cases. Then the linearized TCP (11–12) can be equivalently modeled as

$$\dot{z}(t) = Az(t) + B\dot{u}(t-\tau) \tag{13}$$

where $z(0)$ and $\{u(\sigma), \sigma \in [-\tau, 0]\}$ are given, and

$$\begin{aligned} z(t) &= \begin{bmatrix} \delta b(t) \\ \delta \dot{b}(t) \\ \delta \ddot{b}(t) \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & A_1 & A_2 \end{bmatrix} \\ B &= \begin{bmatrix} 0 \\ 0 \\ B_1 \end{bmatrix}, \quad \dot{u}(t) = \delta \dot{p}(t), \quad A_1 = -\frac{x^* p^*}{\tau} \\ A_2 &= -(x^* p^* + \frac{1}{\tau}), \quad B_1 = -\frac{N}{\tau^2 p^*} \end{aligned}$$

Here $z(t)$ and $\dot{u}(t)$ in (13) are respectively state and input variables of the linearized model around the equilibrium point. It is easy to check that the pair (A, B) is stabilizable.

We define the optimal AQM design as the problem of choosing an input $\dot{u}(\cdot)$ that minimizes the cost of transient around an equilibrium:

$$\begin{aligned} \min_{\dot{u}(\cdot)} J(\dot{u}(\cdot)) = \int_0^\infty & [Q_1 \delta b^2(t) + Q_2 \delta \dot{b}^2(t) \\ & + Q_3 \delta \ddot{b}^2(t) + \delta \dot{p}^2(t)] dt \end{aligned} \tag{14}$$

subject to (13). The first term in the integrand penalizes deviation of the queue length from its equilibrium, the second term penalizes the deviation of the aggregate rate from link capacity ($\dot{b}(t) = y(t) - c$), and the last term penalizes the fluctuation of the marking probability. Hence the cost is a weighted sum of transients in queue, aggregate rate, and fluctuation in probability, weighted by $Q_1 > 0$, $Q_2 > 0$, and $Q_3 \geq 0$. The cost function in (14) can also be written in terms of the state variable and the diagonal matrix $Q = \text{diag}(Q_1, Q_2, Q_3)$:

$$\min_{\dot{u}(\cdot)} J(\dot{u}(\cdot)) = \int_0^\infty [z^T(t) Q z(t) + \dot{u}^2(t)] dt$$

Then the pair $(A, Q^{\frac{1}{2}})$ is observable.

In the following two subsections, we will study two simplified cases: the case without delay compensation and the case of second order control. In the first case, we assume $\tau = 0$ in the control input $\dot{u}(t - \tau)$. This represents a design that does not compensate for delay that is inherent in a real network. We derive the structure of the optimal AQM, interpret various AQM algorithms as different approximations of the optimal AQM and discuss implications on their performance. In the second case, we take $\tau > 0$ in $\dot{u}(t - \tau)$ and explicitly compensate for delay in optimal AQM design. We focus on RED and consider second order control (the general problem in (14) is third order).

4 AQM without delay compensation

4.1 Optimal AQM

The problem of optimal AQM design is to find the minimizing input $\dot{u}(t)$ for (14) subject to

$$\dot{z}(t) = Az(t) + B\dot{u}(t) \tag{15}$$

We will call the minimizing $\dot{u}(t)$ the *optimal AQM* or the *RHA* (Recending Horizon AQM).

Theorem 2 *The RHA that minimizes cost $J(\dot{u}(\cdot))$ is given by*

$$\dot{u}^*(t) \quad = \quad k_1 \ \delta b(t) + k_2 \ \delta \dot{b}(t) + k_3 \ \delta \ddot{b}(t) \tag{16}$$

where $k_1 > 0$, $k_2 > -\frac{A_1}{B_1}$, $k_3 > 0$, and can be obtained by solving a fourth order polynomial. Moreover, the closed-loop system (15) with RHA $\dot{u}^*(t)$ as input is asymptotically stable.

Proof (sketch): The optimal closed-loop control that minimizes (14) is given by

$$\dot{u}^*(t) \quad = \quad -B^T K z(t) \tag{17}$$

where K satisfies the algebraic Riccati equation $0 = A^T K + KA + Q - KBB^T K$. K is a symmetric matrix and the resulting closed-loop system is asymptotically stable since the pairs (A, B) and $(A, Q^{\frac{1}{2}})$ are controllable and observable, respectively. Expanding the algebraic Riccati equation and using (17), it can be shown that

$$\begin{aligned} k_1 &= \sqrt{Q_1} \\ k_2 &= -\frac{B_1}{2}(B_1^2 k_3^2 - 2A_2 k_3) \end{aligned}$$

and k_3 is the positive solution of the following fourth order polynomial:

$$\begin{aligned} -B_1^3 k_3^4 - 4A_2 B_1^2 k_3^3 + (4A_1 B_1 - 4A_2^2 B_1 + 2B_1^3 Q_3) \\ k_3^2 + (8B_1\sqrt{Q_1} + 8A_1 A_2 + 4A_2 B_1^2 Q_3)k_3 \\ +8A_2\sqrt{Q_1} + 4B_1 Q_2 - 4A_1 B_1 Q_3 - B_1^3 Q_3^2 = 0 \end{aligned}$$

■

The minimum cost can be computed as $J^* = z^T(0)Kz(0)$ in terms of the solution K of the algebraic Riccati equation and the initial state $z(0)$. Moreover, the closed-loop system (15) with (16) as input is

$$\dot{z}(t) \quad = \quad \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ B_1 k_1 & A_1 + B_1 k_2 & A_1 + B_1 k_3 \end{bmatrix} z(t).$$

The eigenvalues of the closed-loop system and the buffer dynamics $\delta b(t)$ can be derived explicitly in terms of entries of the matrix K and the initial state $z(0)$. These details are provided in [11].

Theorem 2 clarifies the structure of the optimal AQM that stabilizes TCP dynamics (15) at the minimum cost as defined in (14). It implies in particular that the computation of the marking probability should be based on the perturbations in queue length ($\delta b(t) = b(t) - b^*$), in aggregate rate ($\delta \dot{b}(t) = \delta y(t)$), and in the rate of change in aggregate rate ($\delta \ddot{b}(t) = \delta \dot{y}(t)$). Intuitively, excess queue and aggregate rate should lead to an increase in marking probability, and hence the dependence on $\delta b(t)$ and $\delta \dot{b}(t)$. Theorem 2 says that RHA also makes use of aggregate rate change $\dot{y}(t)$ to adjust the probability $p(t)$, in anticipation of the future; e.g., a positive $\dot{y}(t)$ predicts an excess rate or queue in the future. We will discuss in the next subsection the effect of k_i on the system behavior.

Conversely, given any AQM with this structure, specified by (k_1, k_2, k_3), it solves the receding horizon control problem (14) with appropriate weights Q_i, as the next result says. It can be easily proved from Theorem 2.

Theorem 3 *Given AQM $\dot{u}(t) = [k_1,\ k_2,\ k_3]^T$, it solves the receding horizon control problem (14) with weights*

$$\begin{aligned} Q_1 &= k_1^2 \\ Q_2 &= k_2^2 - \frac{2}{B_1}(A_2 k_1 + B_1 k_1 k_3 - A_1 k_2) \\ Q_3 &= k_3^2 + 2\frac{A_2 k_3 + k_2}{B_1} \end{aligned}$$

Alternatively, instead of specifying the control input directly as in Theorem 3, one can design the dynamics of TCP (15) with state feedback by specifying the eigenvalues $\lambda_1, \lambda_2, \lambda_3$ of the closed-loop system matrix. The next result shows that, under a suitable condition, this dynamics also solves (14) with appropriate Q_i. By combining it with Theorem 2, we can derive the optimal stabilizing AQM, (k_1, k_2, k_3), that achieves the specified dynamics. Its proof can be found in [11]. For simplicity of notations, define

$$\hat{\lambda}_1 = \lambda_1 + \lambda_2 + \lambda_3,\ \hat{\lambda}_2 = \lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_1\lambda_3$$
$$\hat{\lambda}_3 = \lambda_1\lambda_2\lambda_3$$

Theorem 4 *Given the eigenvalues $\lambda_1,\ \lambda_2,\ \lambda_3$ of the closed-loop system (15) with state feedback $\dot{u}(t) = [k_1,\ k_2,\ k_3]^T z(t)$, it solves the receding horizon control problem (14) with weights*

$$\begin{aligned} Q_1 &= \frac{\hat{\lambda}_3^2}{B_1^2} \\ Q_2 &= \frac{-A_1^2 + \hat{\lambda}_2^2 - 2\hat{\lambda}_1\hat{\lambda}_3}{B_1^2} \\ Q_3 &= \frac{-A_2^2 - 2A_1 + \hat{\lambda}_1^2 - 2\hat{\lambda}_2}{B_1^2}. \end{aligned}$$

4.2 Approximating AQM's

We now interpret AQM algorithms RED, REM/PI and AVQ as various approximations of RHA. The models we use for these schemes are highly simplified and ignore many important characteristics. They only capture the property that RED adjusts its marking probability based on queue length, and REM, PI and AVQ based on queue length and aggregate rate. We emphasize that the goal is not to propose RHA as a replacement for current AQM's, but rather as a performance limit that shed light on the behavior of practical AQMs.

The linear models of these AQM's are:

$$\text{RED:} \quad \delta\dot{p}^r(t) = k_2^r \delta\dot{b}(t) \tag{18}$$
$$\text{REM/PI/AVQ:} \quad \delta\dot{p}^m(t) = k_1^m \delta b(t) + k_2^m \delta\dot{b}(t) \tag{19}$$

for some constants k_2^r, k_1^m, k_2^m. The linear models of RED, REM and PI are ridicules of the models in the original papers [3, 1, 6]. We comment on the rational behind the model of AVQ [15]. AVQ operates on two time-scales. The fast time scale, on the order of round trip times, describes the dynamics of TCP and its interaction with marking probability $p(t)$. The probability function not only depends on aggregate rate on a fast time-scale, but also on a virtual capacity that is updated on a slow time-scale. The fast time-scale is relevant here. At this time-scale, the marking probability of AVQ is a static function of aggregate rate, $p = p(Nx(t)) = p(y(t))$

where $x(t) = w(t)/(d + b(t)/c)$. Hence $\dot{p}(t) = p'(y(t))\dot{y}(t) = p'(y(t))\ddot{b}(t)$. Linearizing, we have (19).

By Theorem 2, the optimal AQM has strictly positive gains, $(k_1, k_2, k_3) > 0$ when $Q_3 = 0$. Since this condition is satisfied by none of RED, REM/PI and AVQ, none of them can be made optimal, in the sense of minimzing (14), by tuning its parameters. Moreover, their structure implies a limitation to their equilibrium queue length and rate of convergence to equilibrium.

Specifically, RED has $k_1^r = 0$ and $k_3^r = 0$. It can be shown (see [11]) that the sum of eigenvalues of the closed-loop system is given by

$$\lambda_1 + \lambda_2 + \lambda_3 \quad = \quad A_2 + B_1 k_3^r \quad < \quad A_2$$

where the last inequality follows from that $A_2 < 0$, $B_1 < 0$, and $k_3^r \geq 0$. Since all eigenvalues have nonpositive real parts, the above inequality means that the sum of the real parts of the eigenvalues is less negative when $k_3^r = 0$ than when $k_3^r > 0$. This suggests that the decay rate is smaller with RED ($k_3^r = 0$). The implication of $k_1^r = 0$ is that at least one of the eigenvalues λ_i is zero, implying a nonzero equilibrium queue length (more precisely, steady state error in queue length). Note that $k_1^r = 0$ implies $Q_1 = 0$ in the cost (14), and hence deviation from equilibrium queue length is not penalized.

Since for REM/PI and AVQ, $k_3^m = 0$, they suffer the same structural limitation on decay rate as RED. That $k_1^m > 0$ drives the equilibrium queue length to zero or a target.

4.3 Simulations

Here, we illustrate the results of the above sections via simulation for the simple TCP/AQM model (11). For comparison, the gains of RHA (16) is obtained first by solving the optimization problem. Then, we set $H_1 = H_3 = 0$ for RED, $H_3 = 0$ for REM, and $H_1 = H_2 = 0$ for AVQ.

In the simulation, we also set the sampling time and total simulation time as $T_s = 0.002$ sec and 4 sec (2000 steps), respectively. Fig. 1 shows transient queue length trajectories ($\delta q(t)$) when H_1, H_2, and H_3 of the RHA are 0.0980, 0.0147, and 0.0000677, respectively. Fig. 2 shows transient queue length trajectories ($\delta q(t)$) when H_1, H_2, and H_3 of the RHA are 0.0123, 0.0036, and 0.0000309, respectively. As shown in Fig. 1 and Fig. 1, the queue length trajectory ($\delta q(t)$) of RHA is stable and goes to zero fast, while those of RED, REM, and AVQ are unstable, oscillate, and/or move slowly. As discussed in the previous section, trajectories of RED and AVQ go to some constant value in both figures since they assume $H_1 = 0$, while those of RHA and REM in Fig. 2 can go to zero. Trajectories of RED and REM move very slowly since they assume $H_3 = 0$, while those of RHA and AVQ move very fast. As shown in 1, REM can be unstable if we use large H_1 and H_2, since it assumes $H_3 = 0$. Thus it should use small H_1 and H_2, which makes the trajectory move slowly.

5 AQM with delay compensation

In this section, we take delay into account in designing the optimal AQM, and explain its effect on performance. For simplicity, we focus on second order system and compare the resulting optimal AQM with RED (first order system). We start with second-order RHA without delay compensation and extend it to RHA with delay compensation.

The second-order linearized TCP model is as follows:

$$\dot{z}(t) \quad = \quad Az(t) + Bu(t - \tau) \qquad (20)$$

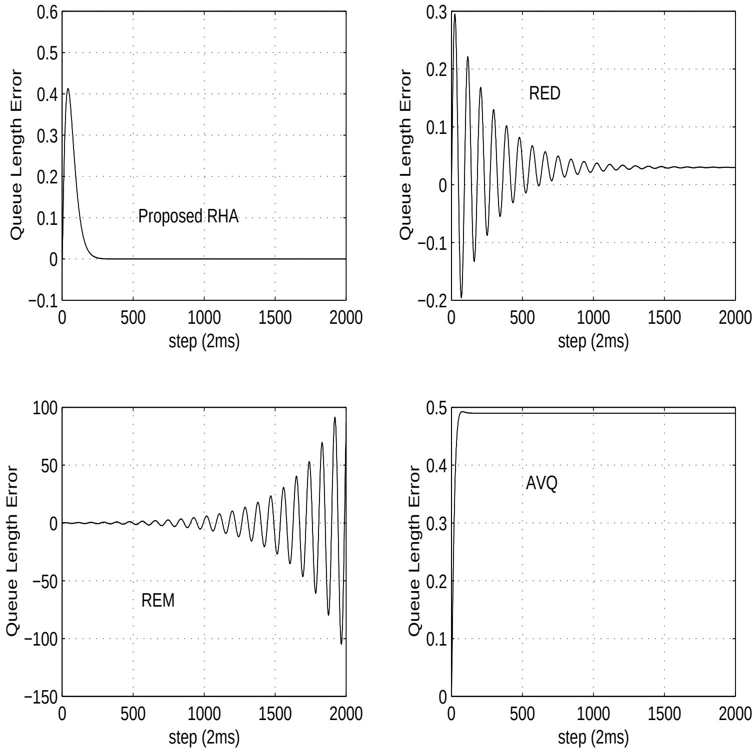

Figure 1: Queue length (δq) trajectory

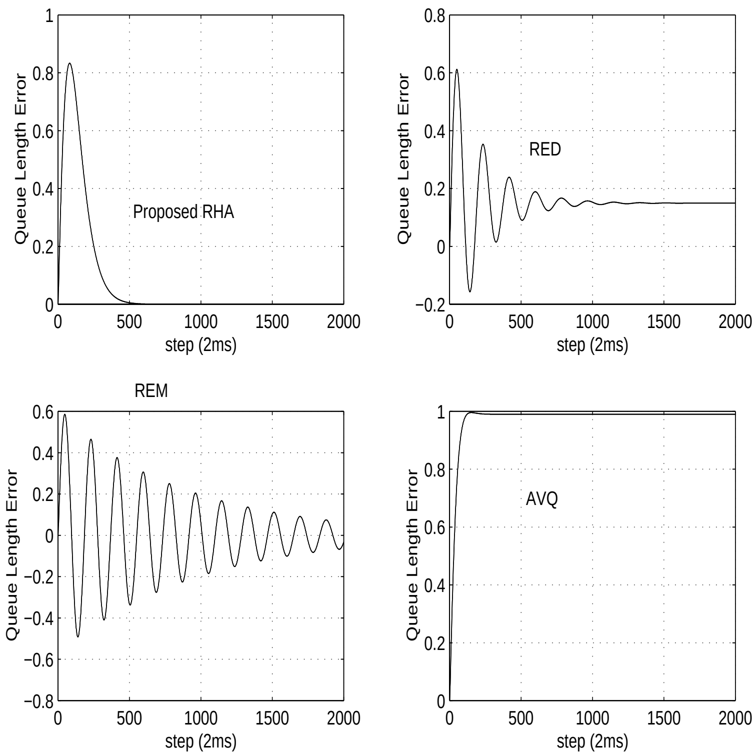

Figure 2: Queue length (δq) trajectory

where $z(0)$ and $\{u(\sigma), \sigma \in [-\tau, 0]\}$ are given,

$$z(t) = \begin{bmatrix} \delta b(t) \\ \delta \dot{b}(t) \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 \\ A_1 & A_2 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ B_1 \end{bmatrix}$$
$$u(t) = \delta p(t).$$

Throughout the rest of this section, for simplicity, we define

$$\begin{aligned} a_1 &= \frac{A_2 + \sqrt{A_2^2 + 4A_1}}{2}, \; a_2 = \frac{A_2 - \sqrt{A_2^2 + 4A_1}}{2} \\ a_3 &= \frac{1}{a_1 - a_2} \log_e \frac{a_2}{a_1} & (21) \\ e_1 &= e^{-a_1\tau} - e^{-a_2\tau}, \; e_2 = a_1 e^{-a_1\tau} - a_2 e^{-a_2\tau} \\ e_3 &= a_2 e^{-a_1\tau} - a_1 e^{-a_2\tau} & (22) \\ \hat{B}_1 &= -\frac{B_1(a_1 - a_2)}{e_1}. & (23) \end{aligned}$$

The key to analyzing the delayed system (13) is to transform the state variable from $z(t)$ into $s(t)$ that satisfies (see [11] for details):

$$\dot{s}(t) = As(t) + \begin{bmatrix} 0 \\ \hat{B}_1 \end{bmatrix} u(t) \qquad (24)$$

where

$$s_1(t) = -\frac{e_2}{e_1}(\delta b(t) + u_{1h}(t)) + \delta \dot{b}(t) + u_{2h}(t)$$
$$s_2(t) = A_1(\delta b(t) + u_{1h}(t)) + \frac{e_3}{e_1}(\delta \dot{b}(t) + u_{2h}(t))$$

$$\begin{bmatrix} u_{1h}(t) \\ u_{2h}(t) \end{bmatrix} = \int_{-\tau}^{0} \begin{bmatrix} e^{-(\sigma+h)a_1} - e^{-(\sigma+h)a_2} \\ a_1 e^{-(\sigma+h)a_1} - a_2 e^{-(\sigma+h)a_2} \end{bmatrix} \frac{B_1}{a_1 - a_2} u(\sigma + t) d\sigma. \qquad (25)$$

Consider the following optimization problem:

$$\min_{u(\cdot)} J(u(\cdot)) = \int_0^\infty [s^T(t) Q s(t) + u(t)u(t)] dt \qquad (26)$$

subject to (24) where $Q = \text{diag}(Q_1, Q_2) \geq 0$ and the pair $(A, Q^{\frac{1}{2}})$ is observable.

Theorem 5 *The optimal AQM (RHA) that solves (26) subject to (24) is given by:*

$$u^*(t) = k_1[\delta b(t) + u_{1h}(t)] + k_2[\delta \dot{b}(t) + u_{2h}(t)] \qquad (27)$$

Moreover, the closed-loop system (20) with delayed state feedback u^* is asymptotically stable.

The control gains k_1 and k_2 can be explicitly computed; see [11]. Note that the extra terms $u_{1h}(t)$ and $u_{2h}(t)$ in (27) represent a correction to the control action in the previous delay period.

The next two results show that a second order AQM solves (26) with appropriate weighting matrix Q. Their proofs can be found in [11].

Theorem 6 *Given AQM $u(t) = [k_1,\ k_2]s(t)$ that satisfies $A_1+\hat{B}_1 k_1$ and $A_2+\hat{B}_1 k_2$ are negative, it solves the receding horizon problem (26) with weights*

$$Q_1 = \frac{k_1^2\hat{B}_1^2 + 2k_1 A_1 \hat{B}_1}{\hat{B}_1^2}$$
$$Q_2 = \frac{k_2^2\hat{B}_1^2 + 2k_2 A_2 \hat{B}_1 + 2k_1\hat{B}_1}{\hat{B}_1^2}.$$

Alternatively, an AQM can be specified by the eigenvalues of the desired closed-loop system.

Theorem 7 *Given the eigenvalues λ_1 and λ_2 of the closed-loop system of the transformed variable $s(t)$, they solves the receding horizon problem (26) with weights*

$$Q_1 = \frac{(\lambda_1\lambda_2)^2 - A_1^2}{\hat{B}_1^2}$$
$$Q_2 = \frac{\lambda_1^2 + \lambda_2^2 - A_2^2 - 2A_1}{\hat{B}_1^2}.$$

We make several remarks on the RHA with delay compensation and interpret RED as an approximation of RHA.

Theorem 5 shows that, at second order, under optimal AQM, the probability $p(t)$ should depend on both queue length and aggregate rate. Moreover, because of the delay, the control should correct the error in input over the previous delay period, as represented by $u_{ih}(t)$.

As before we model RED by (note the input is $\delta p(t)$ not $\delta\dot{p}(t)$ as in the third order system):

$$\delta p(t) = k_1^r \delta b(t)$$

Hence RED sets both $k_2 = 0$ and the history of past inputs $u_{ih}(t)$ to zero. This reduces the decay rate of RED; see [11].

6 Simulation Examples

Here, we illustrate the delay effect of AQM algorithms via NS simulation for the nonlinear model.

In NS simulation, we set N and c as 100 and 4000, respectively. For implementation of RHA, we set a target queue length as $b^* = 175$ packets/sec.

We set $Q_1 = \frac{\alpha}{\hat{B}_1^2}$, $Q_2 = \frac{2\sqrt{A_1^2+\hat{B}_1^2 * Q_1} - A_2^2 - 2A_1}{\hat{B}_1^2}$, and $T_s = 0.04$ sec, respectively where T_s is a sampling time.

We compare RHA (27) with RHA (27) (with $u_{ih} = 0$) and RED ($u(t) = \rho\delta q(t)$, $\rho \approx 0.001$). Fig. 3 shows queue length trajectories ($q(t)$) when $\tau = 0.25$ sec. Fig. 4 shows queue length trajectories ($q(t)$) when $\tau = 0.15$ sec. Figures show that queue length $q(t)$ of RHA (27) goes to the target queue length 175 pkts, almost two or three times faster than RED, while that of RHA (27) (with $u_{ih}(t) = 0$) oscillate. This result illustrates that we should consider the delay term when we design AQM algorithms.

7 Conclusion

In this paper, we propose a unified mathematical framework based on receding horizon control for analyzing and designing AQM (Active Queue Management) algorithms in stabilizing TCP

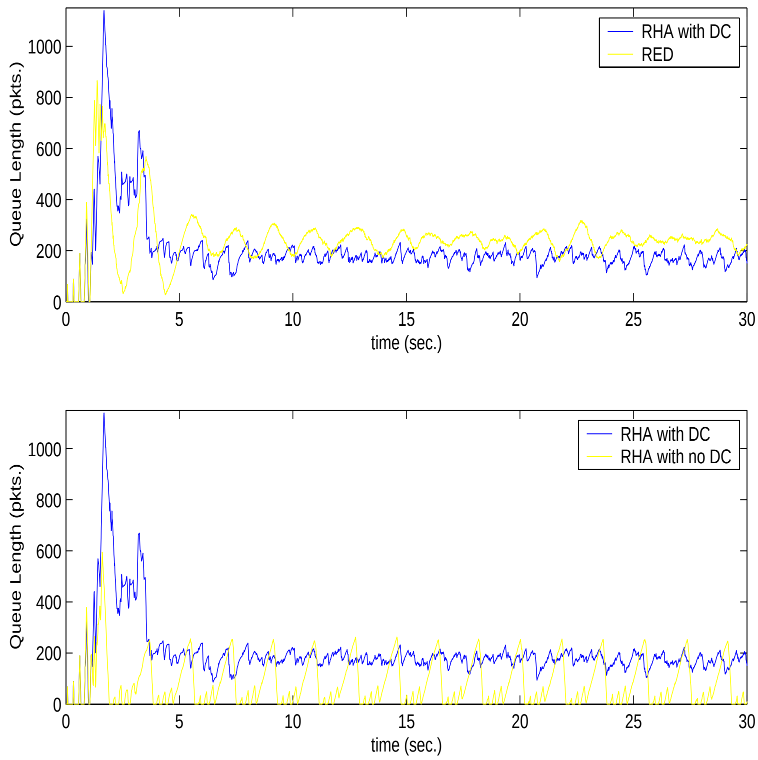

Figure 3: Queue length (q) trajectory

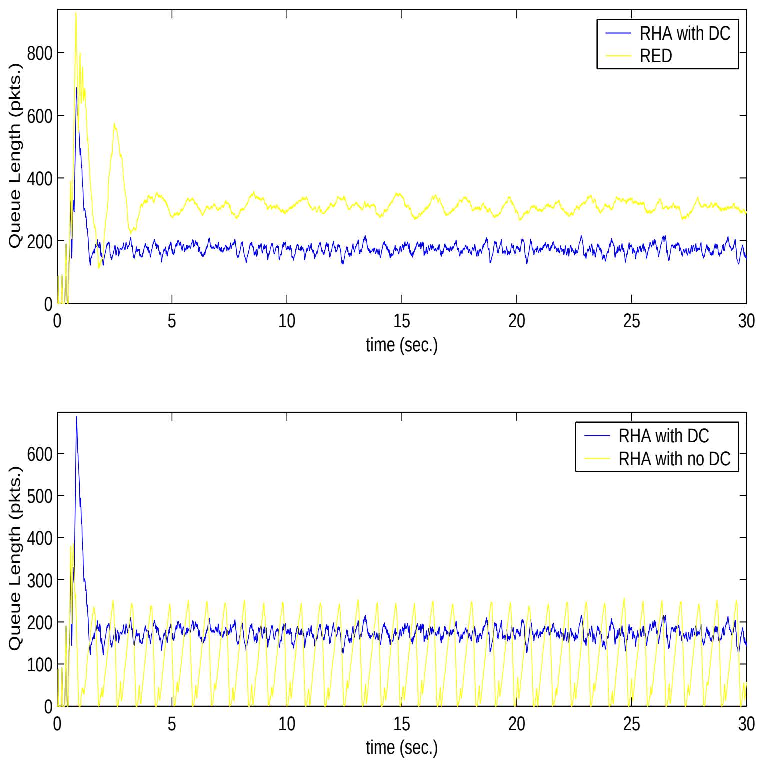

Figure 4: Queue length (q) trajectory

(Transfer Control Protocol). The proposed framework is based on a dynamical system of the given TCP and a linear quadratic cost on transients in queue length and flow rates. We derive the optimal receding horizon AQMs (RHAs) that stabilizes the linearized dynamical system with the minimum cost. Conversely, we show that any AQM with an appropriate structure solves the same optimal control problem with appropriate weighting matrix. We interpret existing AQM's such as RED, REM, PI and AVQ as different approximations of the optimal AQM, and discuss the impact of these approximations on performance.

Acknowledgement: The first author acknowledges the support of the Post-doctoral Fellowship Program of Korean Science & Engineering Foundation (KOSEF). We would like to thank to Sanjeewa Athuraliya for his help of NS-simulation.

References

[1] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin. REM: active queue management. *IEEE Network*, May/June 2001. Extended version in *Proceedings of ITC17*, Salvador, Brazil, September 2001. http://netlab.caltech.edu.

[2] H. Chen and F. Allgower. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34:1205–1217, 1998.

[3] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, August 1993. ftp://ftp.ee.lbl.gov/papers/early.ps.gz.

[4] G. De Nicolao and L. Magni and R. Scattolini. Stabilizing receding horizon control of nonlinear time-varying systems. *IEEE Trans. Automat. Contr.*, 43(7):1030 – 1036, 1998.

[5] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong. A control theoretic analysis of RED. In *Proceedings of IEEE Infocom*, April 2001. http://www-net.cs.umass.edu/papers/papers.html.

[6] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong. On designing improved controllers for AQM routers supporting TCP flows. In *Proceedings of IEEE Infocom*, April 2001. http://www-net.cs.umass.edu/papers/papers.html.

[7] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997. http://www.statslab.cam.ac.uk/~frank/elastic.html.

[8] F. P. Kelly. Mathematical modelling of the Internet. In *Proc. 4th International Congress on Industrial and Applied Mathematics*, July 1999. http://www.statslab.cam.ac.uk/~frank/mmi.html.

[9] F. P. Kelly, A. Maulloo, and D. Tan. Rate control for communication networks: Shadow prices, proportional fairness and stability. *Journal of Operations Research Society*, 49(3):237–252, March 1998.

[10] K. B. Kim. Disturbance attenuation of for input-constrained discrete time-invariant systems via receding horizon control. *To appear in IEEE Transactions on Automatic Control*, 2001.

[11] K. B. Kim and S. H. Low. Analysis and design of AQM based on receding horizon control in stabilizing TCP. *To be submitted for publication*, March, 2002.

[12] M. V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32:1361 – 1379, 1996.

[13] S. Kunniyur and R. Srikant. A time–scale decomposition approach to adaptive ECN marking. In *Proceedings of IEEE Infocom*, April 2001. http://comm.csl.uiuc.edu:80/~srikant/pub.html.

[14] S. Kunniyyr and R. Srikant. End–to–end congestion control schemes: utility functions, random losses and ECN marks. In *Proceedings of IEEE/INFOCOM*, 2000. http://www.ieee-infocom.org/2000/papers/401.ps.

[15] S. Kunniyyr and R. Srikant. Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management. In *Proceedings of ACM/SIGCOMM*, 2001. `http://comm.csl.uiuc.edu/~srikant/pub.html`.

[16] W. H. Kwon and K. B. Kim. On stabilizing receding horizon controls for linear continuous time-invariant systems. *IEEE Transactions on Automatic Control*, 45(8):1329–1334, 2000.

[17] S. H. Low. A duality model of TCP flow controls. In *Proceedings of ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management*, September 18-20 2000. `http://netlab.caltech.edu`.

[18] S. H. Low and D. E. Lapsley. Optimization flow control, I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, December 1999. `http://netlab.caltech.edu`.

[19] S. H. Low, F. Paganini, J. Wang, S. A. Adlakha, and J. C. Doyle. Dynamics of TCP/AQM and a scalable control. In *Proceedings of IEEE Infocom*, June 2002.

[20] S. H. Low, L. Peterson, and L. Wang. Understanding Vegas: a duality model. *J. of ACM, to appear*, 2002. `http://netlab.caltech.edu/pub.html`.

[21] F. Paganini, J. C. Doyle, and S. H. Low. Scalable laws for stable network congestion control. In *Proceedings of Conference on Decision and Control*, December 2001. `http://www.ee.ucla.edu/~paganini`.

[22] J. B. Rawlings and K. R. Muske. The stability of constrained receding horizon control. *IEEE Trans. Automat. Contr.*, 38(10):1512 – 1516, 1993.