# A Duality Model of TCP and Queue Management Algorithms

Steven H. Low
CS and EE Departments
California Institute of Technology
Pasadena, CA 91125
slow@caltech.edu

May 14, 2003

**Abstract**

We propose a duality model of end-to-end congestion control and apply it to understand the equilibrium properties of TCP and active queue management schemes. The basic idea is to regard source rates as primal variables and congestion measures as dual variables, and congestion control as a distributed primal-dual algorithm over the Internet to maximize aggregate utility subject to capacity constraints. The primal iteration is carried out by TCP algorithms such as Reno or Vegas, and the dual iteration is carried out by queue management algorithms such as DropTail, RED or REM. We present these algorithms and their generalizations, derive their utility functions, and study their interaction.

## I. Introduction

Congestion control is a distributed algorithm to share network resources (called 'links' in this paper) among competing sources. It consists of two components: a source algorithm that dynamically adjusts rate (or window size) in response to congestion in its path, and a link algorithm that updates, implicitly or explicitly, a congestion measure and sends it back, implicitly or explicitly, to sources that use that link. On the current Internet, the source algorithm is carried out by TCP, and the link algorithm is carried out by (active) queue management (AQM) schemes such as DropTail or RED [6]. Different protocols use different metrics to measure congestion, e.g., TCP Reno [10], [25] and its variants, use loss probability as congestion measure, and TCP Vegas [4], it turns out, uses queueing delay as congestion measure [18]. Both are implicitly updated at the links and implicitly fed back to sources through end-to-end loss or delay, respectively. In this paper, we present a general model of end-to-end congestion control and apply it to understand the equilibrium properties of the closed-loop systems specified by various TCP/AQM protocols.

The basic idea is to regard the process of congestion control as carrying out a distributed computation by sources and links over a network in real time to solve a global optimization problem formulated in [11]. The objective is to maximize aggregate source utility subject to capacity constraints. We will interpret source rates as primal variables, congestion measures as dual variables, and TCP/AQM protocols as distributed primal-dual algorithms to solve this optimization problem and its associated dual problem (Section II). Different protocols, such as Reno, Vegas, RED, and REM [1], all solve the same prototypical problem with different utility functions, and we derive these functions explicitly (Sections III and IV). Moreover all these protocols generate congestion measures (Lagrange multipliers) that solve the dual problem in equilibrium.

The model implies that the equilibrium properties of a large network under TCP/AQM control, such as throughput, delay, queue lengths, loss probabilities, and fairness, can be readily understood by studying the underlying optimization problem (see later sections and [18]). Moreover, since the problem is a concave program, these properties can be efficiently computed numerically.

It is possible to go between utility maximization and TCP/AQM algorithms in both directions. We can start with general utility functions, e.g., tailored to our applications, and then derive TCP/AQM algorithms to maximize aggregate utility, as done in, e.g., [11], [16], [19], [21], [12]. Conversely, and historically, we can design TCP/AQM algorithms and then reverse-engineer the algorithms to determine the underlying utility functions they implicitly optimize and the associated dual problem, as we do here and in [18]. This is the consequence of end-to-end control: as long as the end-to-end congestion measure to which the TCP algorithm reacts is the *sum* of the constituent link congestion measures, such an interpretation is valid.[1].

In Section V, we discuss the interaction of generalized Reno algorithms, and that of Reno and Vegas.

`To appear in IEEE/ACM Trans. on Networking, October 2003. Partial and preliminary results appear in [15].`
This work is suppported by NSF through grants ANI-0113425 and ANI-0230967, and ARO through grant DAAD19-02-1-0283.

[1] Under some mild assumptions on the TCP and AQM algorithms that are typically satisfied (assumptions C1–C3 in Section II)

It will become clear that fairness of TCP algorithms should not be defined solely in terms whether they receive the same equilibrium rates, as commonly done in the literature, because the equilibrium bandwidth allocation generally also depends on AQM, network topology, and routing, etc. We will conclude in Section VI with some insights from the duality model and limitations of this work.

## II. Duality model of TCP/AQM

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[2]

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

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 [23], let $y_l(t) = \sum_s R_{ls} x_s(t)$ be the aggregate source rate at link $l$ and let $q_s(t) = \sum_l R_{ls} p_l(t)$ be the end-to-end congestion measure for source $s$. In vector notation, we have ($\cdot^T$ denotes transpose)

$$y(t) \;=\; Rx(t) \quad \text{and} \quad q(t) \;=\; R^T p(t)$$

Here, $x(t) = (x_s(t), s \in S)$ and $q(t) = (q_s(t), s \in S)$ are in $\Re_+^{|S|}$, and $y(t) = (y_l(t), l \in L)$ and $p(t) = (p_l(t), l \in L)$ are in $\Re_+^{|L|}$ ($\Re_+$ denotes non-negative real). Source $s$ can observe its own rate $x_s(t)$ and the end-to-end congestion measure $q_s(t)$ of its path, but not the vector $x(t)$ or $p(t)$, nor other components of $q(t)$. Similarly, link $l$ can observe just local congestion $p_l(t)$ and flow rate $y_l(t)$.

The source rate $x_s(t)$ is adjusted in each period according to a function $F_s$ based only on $x_s(t)$ and $q_s(t)$: for all $s$,

$$x_s(t+1) \;=\; F_s(x_s(t), q_s(t)) \tag{1}$$

The link congestion measure $p_l(t)$ is adjusted in each period based only on $p_l(t)$ and $y_l(t)$, and possibly some internal (vector) variable $v_l(t)$, such as the queue length at link $l$. This can be modeled by some functions $(G_l, H_l)$: for all $l$,

$$p_l(t+1) \;=\; G_l(y_l(t), p_l(t), v_l(t)) \tag{2}$$
$$v_l(t+1) \;=\; H_l(y_l(t), p_l(t), v_l(t)) \tag{3}$$

where $G_l$ is non-negative so that $p_l(t) \geq 0$. Here, $F_s$ models TCP algorithms (e.g., Reno or Vegas) and $(G_l, H_l)$ model AQM's (e.g., RED, REM); see the next section. We will often refer to AQM's by $G_l$, without explicit reference to the internal variable $v_l(t)$ or its adaptation $H_l$.

We assume that (1)–(3) has a set of equilibria $(x, p)$. The fixed point of (1) defines an implicit relation between equilibrium rate $x_s$ and end-to-end congestion measure $q_s$:

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

Assume $F_s$ is continuously differentiable and $\partial F_s / \partial q_s \neq 0$ in the open set $A := \{(x_s, q_s) | x_s > 0, q_s > 0\}$. Then, by the implicit function theorem, there exists a unique continuously differentiable function $f_s$ from $\{x_s > 0\}$ to $\{q_s > 0\}$ such that

$$q_s \;=\; f_s(x_s) \;>\; 0 \tag{4}$$

To extend the mapping between $x_s$ and $q_s$ to the closure of $A$, define

$$f_s(0) \;=\; \inf \{q_s \geq 0 \mid F_s(0, q_s) = 0\} \tag{5}$$

possibly $\infty$. If $(x_s, 0)$ is an equilibrium point, $F_s(x_s, 0) = x_s$, then define

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

Define the utility function of each source $s$ as

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

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 implies a greedy source – a larger rate yields a higher utility – and concavity implies diminishing return.

Now consider the problem of maximizing aggregate utility formulated in [11]:

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

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. As the sources are coupled through the shared links (the capacity constraint), solving for $x^*$ directly, however, may require coordination among possibly all sources, and

[2]We abuse notation to use $L$ and $S$ to denote sets and their cardinality.

hence is infeasible in a large network. The key to understanding the equilibrium of (1)–(3) 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 [16]):

$$\min_{p \geq 0} \quad \sum_s \max_{x_s \geq 0} (U_s(x_s) - x_s q_s) + \sum_l p_l c_l \tag{9}$$

Hence, the dual variable is a precise measure of congestion in the network. The dual problem has an optimal solution since the primal problem is feasible. We will interpret the equilibria $(x^*, p^*)$ of (1)–(3) as 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 nonnegative functions. Moreover, equilibrium points of (1)–(3) exist.
C2: For all $s \in S$, $F_s$ are continuously differentiable and $\partial F_s / \partial q_s \neq 0$ in $\{(x_s, q_s) | x_s > 0, q_s > 0\}$; moreover, $f_s$ in (4) 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:* Suppose assumptions C1 and C2 hold. Let $(x^*, p^*)$ be an equilibrium of (1)–(3). Then $(x^*, p^*)$ solves the primal problem (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.

**Proof.** The discussion after the definition (7) of $U_s$ proves the second claim when C4 holds, so we only prove the first claim.

By duality theory (e.g., [3, Proposition 5.1.5]), $(x^*, p^*)$ is primal-dual optimal if and only if $x^*$ is primal feasible, $p^*$ is dual feasible, complementary slackness holds, and

$$x^* \quad = \quad \arg \max_{x \geq 0} L(x, p^*) \tag{10}$$

where $L$ is the Lagrangian of (8) defined as:

$$L(x, p) \quad = \quad \sum_s U_s(x_s) + \sum_l p_l (c_l - \sum_s R_{ls} x_s)$$

Hence, to prove the first claim, we only need to establish (10). Now

$$\begin{aligned} & \max_{x \geq 0} L(x, p^*) \\ = & \max_{x \geq 0} \; \sum_s U_s(x_s) + \sum_l p_l^* \left( c_l - \sum_s R_{ls} x_s \right) \\ = & \sum_s \max_{x_s \geq 0} \left( U_s(x_s) - x_s \sum_l R_{ls} p_l^* \right) + \sum_l p_l^* c_l \end{aligned}$$

By construction of $U_s$, we have from (7) and (4) that, for any equilibrium at which $x_s^* > 0$, $(x^*, p^*)$,

$$U_s'(x_s^*) = f_s(x_s^*) \quad = \quad q_s^* = \sum_l R_{ls} p_l^* \tag{11}$$

Note that if $q_s^* = 0$, then (11) holds by (6). If $x_s^* = 0$, we have from (5)

$$U_s'(0) \; = f_s(0) \quad \leq \quad q_s^* \tag{12}$$

But, (11)–(12) implies that

$$\frac{\partial L}{\partial x_s}(x^*, p^*) \quad \leq \quad 0$$

with equality if $x_s^* > 0$. Since $L(x, p^*)$ is concave in $x$, this is the necessary and sufficient Karush-Kuhn-Tucker condition for $x^*$ to maximize $L(x, p^*)$ over $x \geq 0$. Hence the proof is complete. ■

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 interpretation is the consequence of end-to-end control: it holds as long as the end-to-end congestion measure to which the TCP algorithm reacts is the *sum* of the constituent link congestion measures, under some mild assumptions on the TCP and AQM algorithms that are typically satisfied (assumptions C1–C3 in Section II).

Note that the definition of utility function $U_s$ depends only on TCP algorithm $F_s$. The role of AQM $(G, H)$ is to ensure that the complementary slackness condition of problem (1)–(3) is satisfied (condition C3). The complementary slackness has a simple interpretation: AQM should match input rate to capacity to maximize utilization at every bottleneck link. Any AQM that stabilizes queues possesses this property (see (16) below) and generates a Lagrange multiplier $p^*$ that solves the dual problem.

In the following sections, we apply Theorem 1 to interpret TCP Reno with RED and with REM, and TCP Vegas with DropTail. We first derive an algorithm model $(F, G, H)$ from protocol description, and then use (7) to derive the utility function $U_s$ which the protocol implicitly optimizes. The results are summarized in Table I.

| TCP | | |
|---|---|---|
| Reno-1 | $F_s(x_s(t), q_s(t))$ | $\left[x_s(t) + \frac{1-q_s(t)}{D_s^2} - \frac{2}{3}q_s(t)x_s^2(t)\right]^+$ |
| | Utility | $\frac{\sqrt{3/2}}{D_s} \tan^{-1}\left(\sqrt{\frac{2}{3}}\, x_s D_s\right)$ |
| Reno-2 | $F_s(x_s(t), q_s(t))$ | $\left[x_s(t) + \frac{1-x_s(t)D_s q_s(t)}{D_s^2} - \frac{2}{3}q_s(t)x_s^2(t)\right]^+$ |
| | Utility | $\frac{1}{D_s} \log \frac{x_s D_s}{2x_s D_s + 3}$ |
| Vegas | $F_s(x_s(t), q_s(t))$ | $\begin{cases} x_s(t) + \frac{1}{D_s^2} & \text{if } x_s(t) < \overline{x}_s(t) \\ x_s(t) - \frac{1}{D_s^2} & \text{if } x_s(t) > \overline{x}_s(t) \\ x_s(t) & \text{otherwise} \end{cases}$ |
| | Utility | $\alpha_s d_s \log x_s$ |
| **AQM** | | |
| RED | $G_l(y_l(t), p_l(t), v_l(t))$ | $\begin{cases} 0 & r_l(t+1) \le \underline{b}_l \\ \rho_1(r_l(t+1) - \underline{b}_l) & \underline{b}_l \le r_l(t+1) \le \overline{b}_l \\ \rho_2(r_l(t+1) - \overline{b}_l) + m_l & \overline{b}_l \le r_l(t+1) \le 2\overline{b}_l \\ 1 & r_l(t+1) \ge 2\overline{b}_l \end{cases}$ |
| | $H_l(y_l(t), p_l(t), v_l(t))$ | $b_l(t+1) = [b_l(t) + y_l(t) - c_l]^+$ <br> $r_l(t+1) = (1-\alpha_l)r_l(t) + \alpha_l b_l(t)$ |
| REM | $G_l(y_l(t), p_l(t), v_l(t))$ | $1 - \phi^{r_l(t+1)}$ |
| | $H_l(y_l(t), p_l(t), v_l(t))$ | $b_l(t+1) = [b_l(t) + y_l(t) - c_l]^+$ <br> $r_l(t+1) = [r_l(t) + \gamma(\alpha_l b_l(t) + y_l(t) - c_l)]^+$ |
| Delay | $G_l(y_l(t), p_l(t), v_l(t))$ | $p_l(t+1) = [p_l(t) + \frac{y_l(t)}{c_l} - 1]^+$ |

TABLE I
Summary: duality model of TCP/AQM algorithms. Notations are explained in Sections III and IV.

## III. Reno/AQM

For TCP, we only model the congestion avoidance phase and ignore other (important) aspects such as slow-start and fast retransmit/fast recovery. For AQM, it is useful to distinguish between measure of congestion and feedback of congestion measure. TCP Reno, for instance, uses loss probability as a measure of congestion. The value of this congestion measure can be fed back to sources either by dropping packets or setting an ECN bit with this probability. In this paper, we are concerned with the design of congestion measure and its equilibrium properties, and our AQM models do not capture the feedback mechanism. We will henceforth use 'marking' to refer to either dropping a packet or setting an ECN bit.

### *A. $(F, G, H)$ model*

In this subsection, we present models of TCP Reno, RED and REM. The implications of these models will be given in the following subsection and in the Conclusion section.

We only model the average behavior of AIMD and does not differentiate between TCP Reno [25] and its variants such as NewReno, SACK, etc. All these protocols (henceforth referred to as 'Reno') increase the window by one every round trip time if there is no mark in the round trip time, and halves the window otherwise. There are two versions of multiplicative decrease. Older variants of Reno halves the window every time a mark is detected, whereas new versions of Reno halves the window only once if there is one or more marks in the round trip time. We will call the former version Reno-1 and the latter Reno-2; as we will see below, they have slightly different utility functions and fairness property. For both versions, we interpret packet marking probability as a measure of congestion.

Under DropTail, a packet that arrives to a full buffer is dropped. We do not know a convenient expression for the dynamics of marking probability. A model of loss rate that has been used, e.g., in [7], [12], is that for a bufferless queue, $p(t+1) = [1 - c/\sum_s x_s(t)]^+$. This model is suitable for the penalty function approach to solving (8), but not the duality approach because of the feasibility constraint. Hence, we only present models for RED and REM.

Let $w_s(t)$ be the window size. Let $D_s$ be the equilibrium round trip time (propagation plus equilibrium queueing delay), which we assume is constant, as customary in the literature, e.g., [13], [20]. Let $x_s(t)$ defined by $x_s(t) = w_s(t)/D_s$ be the source rate at time $t$. The time unit is on the order of several round trip times and source rate $x_s(t)$ should be interpreted as the average rate over this timescale. Dynamics smaller than the timescale of a round trip time is not captured by the fluid model.

#### A.1 Reno-1

Let $p_l(t)$ be the marking probability at link $l$ at time $t$. We make the key assumption that the end-to-end marking probability $q_s(t)$ to which source algorithm reacts is the sum of link marking probabilities:

$$q_s(t) \quad = \quad \sum_l R_{ls} p_l(t) \qquad (13)$$

This is reasonable when $p_l(t)$ are small, in which case $q_s(t) = 1 - \prod_{l \in L_s}(1 - p_l(t)) \simeq \sum_{l \in L_s} p_l(t)$. In period $t$, it transmits at rate $x_s(t)$ *packets* per unit time, and receives (positive and negative) acknowledgments at approximately the same rate, assuming every packet is acknowledged. On average, source $s$ receives $x_s(t)(1 - q_s(t))$ number of positive acknowledgments per unit time and each positive acknowledgment increases the window $w_s(t)$ by $1/w_s(t)$. It receives, on average, $x_s(t)q_s(t)$ negative acknowledgments (marks) per unit time and each halves the window. Hence, in period $t$, the net change to the window is roughly[3]

$$x_s(t)(1 - q_s(t)) \cdot \frac{1}{w_s(t)} - x_s(t)q_s(t) \cdot \frac{1}{2} \cdot \frac{4w_s(t)}{3}$$

Then the source algorithm $F_s(x_s(t), q_s(t))$ of Reno-1 is given by:

$$x_s(t+1) \quad = \quad \left[x_s(t) + \frac{1 - q_s(t)}{D_s^2} - \frac{2}{3} q_s(t) x_s^2(t)\right]^+ (14)$$

The quadratic term signifies the property that, if rate doubles, the multiplicative decrease occurs at twice the frequency with twice the amplitude.

#### A.2 Reno-2

Reno-2 increments the window by 1 per round trip time $D_s$ if there is no mark, and halves the window once in each round trip time if there is one or more marks. We model this as follows: in each period $t$ (which is on the order of a few round trip times), the window increases by $1/D_s$ with probability $1 - \hat{q}_s(t)$ and decreases by $2w_s(t)/3D_s$ with probability $\hat{q}_s(t)$, where $\hat{q}_s(t)$ is the end-to-end probability that *at least* one packet is marked in period $t$ in the path of $s$. Again, let $p_l(t)$ denote the probability that a packet is marked at link $l$ in period $t$, and $q_s(t)$ be the end-to-end packet marking probability given by (13). We model $\hat{q}_s(t)$ as

$$\hat{q}_s(t) \quad = \quad w_s(t) q_s(t)$$

[3]The factor $\frac{4}{3}$ is motivated by considering a single Reno flow, where in *smaller* timescale than that of the fluid model, the window oscillates between $\frac{4}{3}w_s(t)$ and $\frac{2}{3}w_s(t)$ with an average of $w_s(t)$. It is more customary to replace the factor $\frac{4}{3}$ by 1 in the literature, as we will do in numerical examples in Section V.

where $w_s(t)$ is the window size. This would be justified if packets in the same window are marked independently of each other and the packet marking probability $q_s(t)$ is small, in which case $\hat{q}_s(t) = 1-(1-q_s(t))^{w_s(t)} \simeq w_s(t)q_s(t)$.

Then, the average change in window size in period $t$ is

$$\begin{aligned} & \frac{1}{D_s}(1-\hat{q}_s(t)) - \frac{2w_s(t)}{3D_s}\hat{q}_s(t) \\ = \quad & \frac{1-w_s(t)q_s(t)}{D_s} - \frac{2}{3}q_s(t)x_s(t)w_s(t) \end{aligned}$$

Hence the source algorithm $F_s(x_s(t), q_s(t))$ for Reno-2 is given by:

$$\begin{aligned} & x_s(t+1) \\ = \quad & \left[x_s(t) + \frac{1-x_s(t)q_s(t)D_s}{D_s^2} - \frac{2}{3}q_s(t)x_s^2(t)\right]^+ \end{aligned} \quad (15)$$

A.3 RED

RED [6] maintains two internal variables, the instantaneous queue length $b_l(t)$ and average queue length $r_l(t)$. They are updated according to

$$b_l(t+1) = [b_l(t) + y_l(t) - c_l]^+ \quad (16)$$
$$r_l(t+1) = (1-\alpha_l)r_l(t) + \alpha_l b_l(t) \quad (17)$$

where $\alpha_l \in (0,1)$. Then, (the 'gentle' version of) RED marks a packet with a probability $p_l(t)$ that is a piecewise linear increasing function of $r_l(t)$:

$$p_l(t) = \begin{cases} 0 & r_l(t) \le \underline{b}_l \\ \rho_1(r_l(t) - \underline{b}_l) & \underline{b}_l \le r_l(t) \le \overline{b}_l \\ \rho_2(r_l(t) - \overline{b}_l) + m_l & \overline{b}_l \le r_l(t) \le 2\overline{b}_l \\ 1 & r_l(t) \ge 2\overline{b}_l \end{cases} \quad (18)$$

where

$$\rho_1 = \frac{m_l}{\overline{b}_l - \underline{b}_l} \quad \text{and} \quad \rho_2 = \frac{1-m_l}{\overline{b}_l}$$

The equations (16)–(18) define the model $(G, H)$ for RED.

A.4 REM

REM [1] also maintains two internal variables, instantaneous queue length $b_l(t)$ and a quantity called 'price' $r_l(t)$. As in RED, $b_l(t)$ is modeled by (16); the price $r_l(t)$ is updated according to

$$r_l(t+1) = [r_l(t) + \gamma(\alpha_l b_l(t) + y_l(t) - c_l)]^+ \quad (19)$$

where $\gamma > 0$ and $\alpha_l > 0$ are constants. It marks packets with a probability that is exponential in price $r_l(t)$:

$$p_l(t) = 1 - \phi^{-r_l(t)} \quad (20)$$

where $\phi > 1$ is an REM parameter. In practice, (19) can be replaced by

$$r_l(t+1) = \left[r_l(t) + \gamma(\alpha_l(b_l(t) - \hat{b}_l) + y_l(t) - c_l)\right]^+$$

where $\hat{b}_l \ge 0$ is a target equilibrium backlog. A larger $\hat{b}_l$ generally yields a higher utilization especially when the queue oscillates widely [1]. With this version, the equilibrium queue length in Theorem 3 below is $b_l^* = \hat{b}_l$. (19) corresponds to setting $\hat{b}_l = 0$.

Exponential marking probability (20) is useful for estimating end-to-end price $\sum_{l \in L_s} r_l(t)$ at the source $s$. Since this is not used by Reno, other increasing function can also be used, as explained in [1]. For instance, the marking probability can be linear in price $r_l(t)$:

$$p_l(t) = \min\{\rho r_l(t), 1\} \quad (21)$$

for some constant $\rho > 0$. The version with nonzero target queue length $\hat{b}_l$ and linear marking probability is equivalent to the PI controller of [9]. Other proposed AQM's such as Adaptive Virtual Queue of [12] can also be modeled in the form of (2)–(3).

The equations (16), (19), and (20) or (21) define the model $(G, H)$ for REM.

## B. *Utility functions of Reno*

In this subsection, we derive the utility functions of Reno-1 and Reno-2 and show that, with RED or REM, they solve both the primal and dual problem. Note that all results of this subsection apply to a network that contains *both* Reno-1 and Reno-2 sources and *both* RED and REM links.

*Lemma 2:* The functions $(F, G, H)$ that model Reno-1, Reno-2, RED and REM (equations (14)–(21)) satisfy conditions C1, C2, C4.

**Proof.** Clearly, condition C1 is satisfied. For both Reno-1 and Reno-2, when $x_s > 0$, $F_s$ is continuously differentiable and $\partial F_s / \partial q_s \neq 0$. For Reno-1,

$$q_s = \frac{3}{2x_s^2 D_s^2 + 3} =: f_s(x_s) \quad (22)$$

For Reno-2,

$$q_s = \frac{3}{x_s D_s(2x_s D_s + 3)} =: f_s(x_s) \quad (23)$$

Hence $f_s(x_s)$ is strictly decreasing for both Reno-1 and Reno-2, implying strict concavity of their utility functions. Hence conditions C2 and C4 are both satisfied. ■

Combining (22) and (7), the utility function of Reno-1 (14) is

$$U_s(x_s) = \frac{\sqrt{3/2}}{D_s} \tan^{-1}\left(\sqrt{\frac{2}{3}} x_s D_s\right) \quad (24)$$

Similarly, from (23), the utility function of Reno-2 (15) is:

$$U_s(x_s) \;=\; \frac{1}{D_s}\log\frac{x_sD_s}{2x_sD_s+3} \qquad (25)$$

Note that the utility functions of Reno-1 and Reno-2 imply that, unlike Vegas, it is possible for a source that traverses many bottleneck links to receive zero bandwidth (when its end-to-end price is 1 unit).

The following result applies Theorem 1 to Reno with RED or with REM. It implies that the equilibrium queue length with RED depends on the problem instance (network topology, routing, number of sources, etc.) and RED parameters, and hence inevitably grows as load increases. RED parameters can be tuned, statically or dynamically, to reduce equilibrium queue length, but only at the expense of potential instability; see Example 1 below. In contrast, the equilibrium queue length with REM is zero regardless of load.

*Theorem 3:* 1. Let $(x^*, p^*)$ be an equilibrium of a network that contains both Reno-1 and Reno-2 sources and both RED and REM links. Then $(x^*, p^*)$ solves the primal (8) and the dual problem (9), with utility functions given by (24) for Reno-1 and (25) for Reno-2 sources. Moreover, the equilibrium rate vector $x^*$ is unique.
2. If link $l$ implements RED, then the equilibrium queue length $b_l^*$ satisfies $b_l^* > \underline{b}_l$ at links with $p_l^* > 0$. If link $l$ implements REM, then $b_l^* = 0$.
**Proof.** By Lemma 2, C1, C2, C4 are satisfied by combinations of (14)–(21). Given an equilibrium $(x^*, p^*)$, to show that it is primal-dual optimal, we need to check that C3 is also satisfied. From (16), $y_l^* \le c_l$ with both RED and REM and hence primal feasibility is satisfied. Suppose $p_l^* > 0$. If link $l$ implements RED, then from (17) and (18),

$$b_l^* \;=\; r_l^* \;>\; \underline{b}_l \;\ge\; 0 \qquad (26)$$

but $b_l^* > 0$ implies that $y_l^* = c_l$. If link $l$ implements REM, then $p_l^* > 0$ implies $r_l^* > 0$ and hence, from (19),

$$\alpha_l b_l^* + y_l^* - c_l \;=\; 0 \qquad (27)$$

We know $y_l^* \le c_l$. If $y_l^* < c_l$, then (16) implies that $b_l^* = 0$; but this contradicts (27). Hence $y_l^* = c_l$ (and $b_l^* = 0$). We have thus shown that complementary slackness is satisfied with both RED and REM, and hence C3 is satisfied and $(x^*, p^*)$ is primal-dual optimal.

Moreover, (26) also shows that $b_l^* > \underline{b}_l$ when $p_l^* > 0$ with RED. With REM, the preceding argument shows that $b_l^* = 0$. This completes the proof. ■

**Example 1: Reno/RED at a single link**

Consider a single link with capacity $c$ shared by a set of Reno-1 sources with round trip delays $D_s$. From (22) and $y_l^* = c_l$, the equilibrium rates are

$$x_s^* \;=\; \frac{\overline{D}}{D_s}\,c,$$

and the equilibrium marking probability is

$$p^* \;=\; \frac{3}{2c^2\overline{D}^2+3} \;>\; 0$$

where $\overline{D} = \left(\sum_s D_s^{-1}\right)^{-1}$. If the sources are Reno-2 instead, then the equilibrium rates are the same (use (23)), but the marking probability

$$p^* \;=\; \frac{3}{2c^2\overline{D}^2+3c\overline{D}} \;>\; 0$$

is typically lower since $c\overline{D}$ is usually greater than 1 packet.

If RED is used, the equilibrium probability $p^*$ determines the equilibrium queue length through the marking probability function. Inverting (18), we have (since $b_l^* = r_l^*$ and $p^* > 0$)

$$b_l^* \;=\; \begin{cases} \underline{b}_l + \rho_1^{-1}p_l^* & \text{if } 0 < p_l^* \le m_l \\ \overline{b}_l + \rho_2^{-1}(p_l^* - m_l) & \text{if } m_l < p_l^* \le 1 \end{cases}$$

In particular, as the number of sources increases, $\overline{D}$ decreases, and hence both $p_l^*$ and $b_l^*$ increase. Indeed, $b_l^*$ under RED grows toward twice the maximum threshold as load increases:

$$\lim_{\overline{D}\to 0} b_l^* \;=\; \lim_{p^*\to 1} b_l^* \;=\; 2\overline{b}_l$$

To reduce equilibrium queue length $b_l^*$, a large $m_l$ (`max_p`) and a small $\overline{b}_l$ (`max_th`) should be used. But this increases the slope $\rho_1$ and compromises stability; see [14]. Hence, RED parameters can be tuned either to maintain stability *or* reduce equilibrium queueing delay. ■

**Remarks:**
1. The relations (22) and (23) imply that Reno-1 and Reno-2 discriminate against sources with large $D_s$, as well known in many previous studies, e.g., [5], [6], [13], [20]. Moreover, (22) for Reno-1 can be rewritten as

$$x_s \;=\; \frac{\sqrt{3/2}}{D_s}\sqrt{\frac{1-q_s}{q_s}} \;\simeq\; \frac{\sqrt{3/2}}{D_s}\frac{1}{\sqrt{q_s}}$$

in packets per unit time, when probability $q_s$ is small, a relation widely observed previously. Some authors, e.g., [9], [2], assume that Reno increases its window by 1 every round trip time deterministically. This corresponds to replacing $(1-q_s(t))$ by 1 in (14), which holds when the marking probability is small. This model gives $x_s = \sqrt{3/2}/D_s\sqrt{q_s}$, with a corresponding utility function

$$U_s(x_s) \quad = \quad -\frac{3/2}{x_s D_s^2} \tag{28}$$

as used in [19], [12] (ignoring a constant term). For Reno-2, (23) can be approximated by

$$q_s \quad = \quad \frac{3}{x_s D_s(2x_s D_s + 3)} \quad \simeq \quad \frac{3}{2x_s^2 D_s^2}$$

when $2x_sD_s \gg 3$, or when $q_s$ is small. Then Reno-2 has the same utility function as Reno-1 given by (28).
2. By duality theory, given a dual optimal $p$, the rate vector $x$ given by

$$x_s \quad = \quad U_s'^{-1}(q_s) \tag{29}$$

is the (unique) optimal rate vector, where $q_s = \sum_l R_{ls}p_l$. The rate adjustment process of Reno, (14) or (15), can be regarded as a *smoothed* version of this strategy, in the following sense. Let $\overline{x}_s(t) = U_s'^{-1}(q_s(t))$ be the target rate determined by (29), given $p(t)$, using the utility function of Reno-1 or Reno-2. Then using (24) for Reno-1, we have

$$\overline{x}_s(t) \quad = \quad U_s^{'-1}(q_s(t)) \quad = \quad \frac{1}{D_s}\sqrt{\frac{3}{2}\frac{1-q_s(t)}{q_s(t)}}$$

We can then rewrite the rate adjustment (14) in terms of the target rate $\overline{x}_s(t)$ as:

$$x_s(t+1) \quad = \quad \left[x_s(t) + \frac{2q_s(t)}{3}\left(\overline{x}_s^2(t) - x_s^2(t)\right)\right]^+$$

Hence, instead of setting the rate $x_s(t+1)$ directly to the target rate $\overline{x}_s(t)$ in one step, Reno-1 moves the current rate $x_s(t)$ toward the target rate $\overline{x}_s(t)$ by adding an amount proportional to the difference of their squares, $2q_s(t)(\overline{x}_s^2(t) - x_s^2(t))/3$.
For Reno-2, from (23), the target rate $\overline{x}_s(t)$ must satisfy

$$q_s(t) \quad = \quad \frac{3}{\overline{x}_s(t)D_s(2\overline{x}_s(t)D_s + 3)}$$

Hence $F_s$ in (15) can be rewritten in terms of the target rate $\overline{x}_s(t)$ as

$$\begin{aligned} & x_s(t+1) \\ = \quad & \left[x_s(t) + \frac{1}{D_s^2}\left(1 - \frac{x_s(t)(2x_s(t)D_s+3)}{\overline{x}_s(t)(2\overline{x}_s(t)D_s+3)}\right)\right]^+ \end{aligned}$$

i.e., increase rate if $x_s(t) < \overline{x}_s(t)$ and decrease otherwise.
3. The approach taken here follows that in [17] where queue management mechanisms are modeled entirely by $(G_l, H_l)$. In contrast, the model in [15] includes the marking probability function as a part of $F_s$, making utility function dependent on AQM as well as TCP algorithms.

## IV. Vegas/DropTail

A duality model of Vegas has been developed and validated in [18]. In this section, we summarize the main results. We consider the situation where the buffer size is large enough to accommodate the equilibrium queue length so that Vegas sources can converge to the unique equilibrium. In this case, there is no packet loss in equilibrium.

It is shown in [18] that Vegas uses queueing delay as congestion measure, $p_l(t) = b_l(t)/c_l$, where $b_l(t)$ is the queue length in period $t$. The update rule is therefore $G_l(y_l(t), p_l(t))$ given by (dividing both sides of (16) by $c_l$):

$$p_l(t+1) \quad = \quad \left[p_l(t) + \frac{y_l(t)}{c_l} - 1\right]^+ \tag{30}$$

Hence, AQM for Vegas does not involve any internal variable.

Given $p(t)$, or $q_s(t)$, let $\overline{x}_s(t)$ given by:

$$\overline{x}_s(t) \quad = \quad \frac{\alpha_s d_s}{q_s(t)} \tag{31}$$

be the target rate, where $\alpha_s$ is a parameter of Vegas, and $d_s$ is the round trip propagation delay of source $s$, assumed known by $s$. The update rule for source rate is then $F_s(x_s(t), q_s(t))$ given by:

$$x_s(t+1) \quad = \quad x_s(t) + \frac{1}{D_s^2}\,\mathbf{1}(\overline{x}_s(t) - x_s(t)) \tag{32}$$

where $\mathbf{1}(z) = 1$ if $z > 0$, $-1$ if $z < 0$, and 0 if $z = 0$. In equilibrium, we have $x_s = \overline{x}_s = \alpha_s d_s/q_s$. Hence $U_s'(x_s) = \alpha_s d_s/x_s$ or $U_s(x_s) = \alpha_s d_s \log x_s$. The following result is proved in [18]. It implies, in particular, that we can compute the queue length at each link by solving a simple concave program.

*Theorem 4* ([18]) 1. An equilibrium $(x^*, p^*)$ of Vegas/DropTail as modeled by (30)–(32) solves the primal (8) and the dual problem (9), with utility functions $U_s$ given by

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

Moreover, $x^*$ is unique and weighted proportionally fair.
2. The equilibrium queue lengths at links $l$ are $c_l p_l^*$.
Again the rate adjustment of Vegas (32) can be interpreted as a smoothed version of (29) with the utility function given in the theorem. Instead of setting the rate $x_s(t+1)$ in one step to the target rate $\overline{x}_s(t)$ determined by (29), Vegas moves the current rate $x_s(t)$ closer to the target rate $\overline{x}_s(t)$ by $1/D_s^2$ in each step.

## V. Generalization and TCP-friendliness

In this section we derive the utility function of Reno-like algorithms, and consider the interaction of different TCP algorithms.

### A. Reno-like algorithms

Consider algorithms that increase the rate $x_s(t)$ by $\alpha_s(x_s(t))$ on each positive acknowledgment, and decrease it by $\beta_s(x_s(t))$ on each mark. Then $F_s$ in (14) and (15) are generalized to

$$\begin{aligned} x_s(t+1) &= [x_s(t) + (1-q_s(t))x_s(t)\alpha_s(x_s(t)) \\ &\qquad -q_s(t)x_s(t)\beta_s(x_s(t))]^+ \end{aligned} \quad (33)$$

Reno-1 is a special case with $\alpha_s(x_s) = 1/x_s D_s^2$ and $\beta_s(x_s) = x_s/2$, and Reno-2 with $\alpha_s(x_s) = 1/x_s D_s^2$ and $\beta_s(x_s) = 1/2D_s$. We will index these algorithms by their increase-decrease functions $(\alpha_s, \beta_s)$.

From (33) we have in equilibrium

$$q_s = \frac{\alpha_s(x_s)}{\alpha_s(x_s)+\beta_s(x_s)} := f_s(x_s) \quad (34)$$

and hence the utility function is

$$U_s(x_s) = \int \frac{\alpha_s(x_s)}{\alpha_s(x_s)+\beta_s(x_s)} dx_s \quad (35)$$

A source algorithm $(\alpha_s, \beta_s)$ is said to be *TCP-friendly* if its equilibrium rate coincides with Reno's. In the following, we will use Reno-1 in the definition of TCP-friendliness; however an analogous analysis applies to Reno-2.

Equating (14) for Reno-1 and (34), we see that an algorithm $(\alpha_s, \beta_s)$ is TCP-friendly if and only if

$$\frac{\alpha_s(x_s)}{\alpha_s(x_s)+\beta_s(x_s)} = \frac{2}{2+x_s^2 D_s^2}$$

or

$$\frac{\alpha_s(x_s)}{\beta_s(x_s)} = \frac{2}{x_s^2 D_s^2} \quad (36)$$

Hence, TCP-friendliness of a Reno-like algorithm depends on the increase-decrease functions only through their ratio.

As an illustration, we consider a class of Reno-like algorithms called binomial algorithms in [2]. These algorithms are indexed by a pair $(k, l)$ and corresponds to

$$\alpha_s(x_s) = \alpha/x_s^{k+1} D_s^{k+2} \quad (37)$$
$$\beta_s(x_s) = \beta x_s^l D_s^{l-1} \quad (38)$$

for some constants $\alpha, \beta > 0$ (Reno corresponds to $(k,l) = (0,1)$). Substituting (37)–(38) into the condition (36) for TCP-friendliness yields

$$\frac{\alpha_s(x_s)}{\beta_s(x_s)} = \frac{\alpha}{\beta} \frac{1}{(x_s D_s)^{k+l+1}}$$

which implies the $k+l$ rule of [2]: a binomial algorithm is TCP-friendly if and only if $k+l=1$ and $\alpha/\beta = 2$.

The utility functions of binomial algorithms can be derived from (35) and (37)–(38) to be

$$U_s(x_s) = \frac{1}{D_s}\left(\frac{\alpha}{\beta}\right)^{\frac{1}{n}} \int \frac{dy}{1+y^n}\bigg|_{y=x_s D_s (\beta/\alpha)^{\frac{1}{n}}}$$

where $n = k+l+1$. The class of $n=1$ includes the AIAD algorithm and has a utility function

$$U_s(x_s) = \frac{1}{D_s}\frac{\alpha}{\beta} \log\left(1 + x_s D_s \frac{\beta}{\alpha}\right)$$

The class of $n=2$ is TCP-friendly when $\alpha/\beta = 2$ and has a utility function

$$U_s(x_s) = \frac{1}{D_s}\sqrt{\frac{\alpha}{\beta}} \tan^{-1}\left(x_s D_s \sqrt{\frac{\beta}{\alpha}}\right)$$

For $n=3$, the utility function is [24, pp. 60]

$$\begin{aligned} U_s(x_s) &= \frac{1}{3D_s}\left(\frac{\alpha}{\beta}\right)^{\frac{1}{n}} \\ &\left(\log \frac{1 + x_s D_s(\beta/\alpha)^{\frac{1}{n}}}{\sqrt{1 - x_s D_s(\beta/\alpha)^{\frac{1}{n}} + x_s^2 D_s^2 (\beta/\alpha)^{\frac{2}{n}}}}\right. \\ &\left. + \sqrt{3}\tan^{-1} \frac{x_s D_s (\beta/\alpha)^{\frac{1}{n}}\sqrt{3}}{2 - x_s D_s(\beta/\alpha)^{\frac{1}{n}}}\right) \end{aligned}$$

### B. Interaction: binomial algorithms

The duality model provides a convenient framework in which to study the interaction of different TCP/AQM schemes, provided all TCP algorithms use the same congestion measure. Once the schemes under study are characterized by $(F, G, H)$ and their utility functions, the equilibrium rates and performance such as loss, delay and queue length can be obtained by solving the concave program (8). Close-form solutions are usually unavailable for general network topology, but numerical solutions can be efficiently computed to provide insight on the equilibrium properties, such as throughput and fairness. For single-link network, closed-form solutions can be easily obtained, as we now illustrate. We first consider the interaction of binomial algorithms $(k, l)$, which use the same congestion measure, marking probability. In the next subsection, we consider the interaction of Reno and Vegas, which use different congestion measures.

Consider a single link shared by $N_n$ type-$n$ sources with equilibrium round trip delay $D_n$, where $k+l+1 = n$, $n = 1, 2, \cdots$. Let $x_n$ be the common equilibrium rate of all type-$n$ sources. Let $p$ be the common equilibrium marking probability. Since, in equilibrium, $p = U_s'(x_s)$ for all sources $s$, we have from (34), $p = (1 + \lambda x_n^n D_n^n)^{-1}$ and

$$x_n = \frac{1}{D_n}\left(\frac{1-p}{\lambda p}\right)^{\frac{1}{n}} \quad (39)$$

where $\lambda = \beta/\alpha$. Hence $x_1 D_1 = (x_n D_n)^n = (1-p)/\lambda p$ for all $n$. Since $(1-p)/\lambda p$ is greater than 1 if and only if (40) below holds, we have

*Theorem 5:* Consider classes of binomial algorithms $(k, l)$ indexed by $n$ with $n = k + l + 1$ that share the same link. The window $w_n = x_n D_n$ of type-$n$ sources is related to that of type-1 sources by

$$w_1 \quad = \quad w_n^n, \qquad n = 1, 2, \ldots$$

If

$$p \quad < \quad \frac{\alpha}{\alpha + \beta} \tag{40}$$

then

$$w_1 \ > \ w_2 \ > \ \ldots$$

Otherwise

$$w_1 \ \le \ w_2 \ \le \ \ldots$$

with equalities if and only if equality holds in (40).
Reno sources are of type $n = 2$ sources with $\alpha = 1$, $\beta = 1/2$ and $\lambda = 1/2$.[4] Then (40) becomes $p < 2/3$, which usually holds in practice. The theorem then implies that the window size of a type-$n$ binomial source is no larger than that of a Reno source if and only if $n \ge 2$.

We close by presenting a numerical example.

**Example 2: Binomial algorithms**
Consider a link of capacity $c$ shared by $N_1$ type-1 sources, $N_2$ type-2 sources, and $N_3$ type-3 sources, all with the same round trip delay of $D = 200$ ms. From (39) we have

$$x_1 = \frac{\eta^6}{D}, \qquad x_2 = \frac{\eta^3}{D}, \qquad x_3 = \frac{\eta^2}{D} \tag{41}$$

where

$$\eta \quad := \quad \left( \frac{1-p}{\lambda p} \right)^{\frac{1}{6}} \tag{42}$$

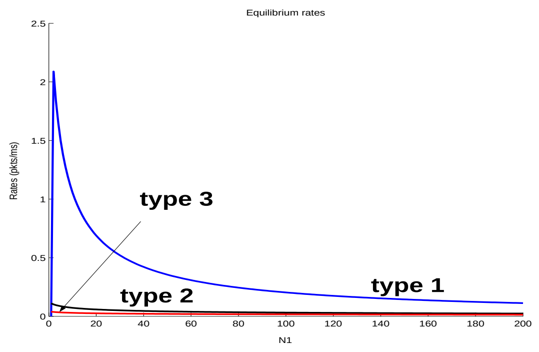
Since $N_1 x_1 + N_2 x_2 + N_3 x_3 = c$ the link capacity, we have

$$N_1 \eta^6 + N_2 \eta^3 + N_3 \eta^2 \quad = \quad cD \tag{43}$$

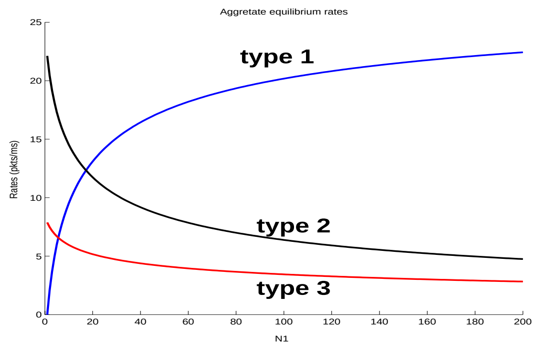
Hence we can solve the polynomial in (43), and then obtain marking probability $p$ from (42) and equilibrium rates from (41). We compute the case for $\alpha = 1$, $\beta = 1/2$ and $\lambda = 1/2$, under which type $n = 2$ sources are Reno. We fix the number of Reno sources, $N_2 = 200$, and vary the numbers $N_1$ or $N_3$ to observe the effect of unfriendly sources on equilibrium rates. The link capacity is $c = 30$ packets/ms.

[4]We ignore the factor $\frac{4}{3}$ in this section; see footnote 3.

Figure 1 shows the equilibrium rates $x_1, x_2, x_3$ when $N_2 = N_3 = 200$ and the number $N_1$ of aggressive sources varies from 0 to 200. Figure 1(a) shows the rates of individual sources whereas Figure 1(b) shows the aggregate rates, summed over all sources of the same type. As observed in [2], type-1 sources are more



(a) Individual rates

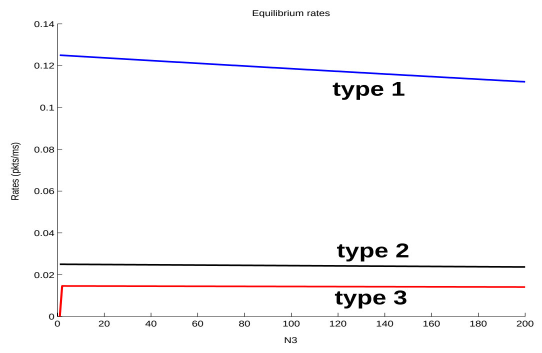


(b) Aggregate rates

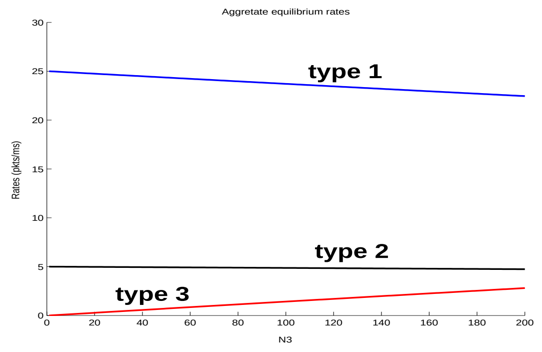
Fig. 1. Equilibrium rates as type 1 sources varies. $N_1 = 0, \ldots 200$, $N_2 = N_3 = 200$, $D = 200$ ms, $c = 30$ pkts/ms.

aggressive than Reno, while type-3 sources are less aggressive. Moreover, the presence of type 1 sources can seize a disproportionally large amount of bandwidth: when there is just one type-1 source, $x_1 = 2.088$ pkts/ms while $x_2 = 0.102$ pkts/ms and $x_3 = 0.037$ pkts/ms (when there are no type-1 sources, $x_2 = 0.111$ pkts/ms and $x_3 = 0.039$ pkts/ms). As $N_1$ increases, while individual rate $x_1$ drops, the aggregate rate of all type-1 sources rises sharply.

Figure 2 shows the individual and aggregate rates when $N_3$ varies from 0 to 200, while keeping $N_1 = N_2 = 200$. The effect of polite sources is much less dramatic than that of aggressive sources. The aggregate share of all type-1 sources ranges from 83% to 75% as $N_3$ varies from 0 to 200. ■



(a) Individual rates



(b) Aggregate rates

Fig. 2. Equilibrium rates as type-3 sources varies. $N_1 = N_2 = 200$, $N_3 = 0, \ldots 200$, $D = 200$ ms, $c = 30$ pkts/ms.

## C. *Interaction: Reno and Vegas*

Suppose Reno and Vegas sources share the same network. Under what condition will they receive the same equilibrium rate? This is not as straightforward as for binomial algorithms because Reno and Vegas use different congestion measures, marking probability for Reno and queueing delay for Vegas. A Reno-like source is TCP-friendly as long as its increase-decrease ratio satisfies (36). Note that this means that if such a source is friendly under any condition (network topology, routing, etc.), then it is friendly under *all* conditions.

In contrast, Vegas sources can receive more, equal, or less bandwidth than Reno sources depending on the network condition. Specifically, let $q_v$ be the end-to-end queueing delay of a Vegas source $v$, in equilibrium, and let $p_n$ be the end-to-end marking probability of a Reno source $n$ sharing the same network, among other sources. Then, the equilibrium rate of the Vegas source $v$ is $x_v = \alpha_v d_v / q_v$, where $\alpha_v > 0$ is a protocol parameter and $d_v$ is the round trip propagation delay of $v$. The equilibrium rate of the Reno-1 source $n$ is $x_n = \sqrt{2(1-p_n)/p_n}$. Hence they receive the same equilibrium rate if and only if

$$\left(\frac{\alpha_v d_v}{q_v}\right)^2 = \frac{2(1-p_n)}{p_n}$$

Hence, whether $v$ is TCP-friendly or not depends on the network condition through equilibrium queueing delay $q_v$ and marking probability $p_n$. But these equilibrium properties depend not only on TCP Reno and Vegas algorithms $F_s$, but also on AQM algorithm $(G_l, H_l)$ and its parameter setting, as well as network topology, routing, and link capacity. Hence, TCP-friendliness of a scheme that uses a different congestion measure should not be defined simply in terms of its equilibrium bandwidth share, because one can generally find scenarios where the scheme receives higher bandwidth share than TCP Reno and scenarios where the reverse is true.

To be concrete, consider Reno (again, we consider only Reno-1 sources though the analysis applies to Reno-2 sources as well) and Vegas sources sharing a single link employing RED or REM. Reno sources react to RED or REM marks by halving its rate. If Vegas reacts to marks in the same way, then its behavior would be similar to Reno. Hence we study the case where Vegas source ignores RED marks and only reacts to delay in its path as it does under DropTail. Under REM, we use the Vegas/REM algorithm in [18] in which a Vegas source estimates the price and uses it to replace queueing delay in setting its rate.

Notice that RED uses queue length $b(t)$ as an internal variable that determines both the marking probability for Reno and queueing delay for Vegas. Hence we can regard $b(t)$ as a *common* congestion measure to which Reno and Vegas react under RED.[5] For REM, the common congestion measure can be taken to be the price variable. The following examples show that AQM can have a big effect on the equilibrium rate allocation when sources react to different congestion signals.

**Example 3: Reno and Vegas under RED**
Suppose there are $N_1$ Reno sources and $N_2$ Vegas sources. The round trip propagation delay for type-$i$ sources is $d_i$, $i = 1, 2$, so that the round trip time is $D_i = d_i + b/c$ in equilibrium, where $b$ is the queue length, $c$ is the link capacity, and $b/c$ is the queueing delay. Vegas sources all have parameter $\alpha_2$ so that each keeps $\alpha_2 d_2$ packets in the buffer in equilibrium.

Consider the case where the link uses RED with marking probability that depends on queue length $b$:

$$p(b) = \frac{b - \underline{b}}{\overline{b} - \underline{b}}, \qquad \underline{b} \le b \le \overline{b} \tag{44}$$

[5]The utility function of Reno is different under this formulation; see [15].

i.e., the marking probability rises from 0 to 1 over the interval $[\underline{b}, \overline{b}]$:

From (22), Reno's equilibrium rate satisfies

$$p(b) \quad = \quad \frac{2}{2 + x_1^2(d_1 + b/c)^2}$$

Combining with (44), we have

$$x_1 \quad = \quad \left( \frac{2(\overline{b} - b)}{b - \underline{b}} \right)^{\frac{1}{2}} \frac{c}{b + cd_1} \qquad (45)$$

From (31), the equilibrium rate of Vegas sources are

$$x_2 \quad = \quad \frac{\alpha_2 d_2}{b/c} \qquad (46)$$

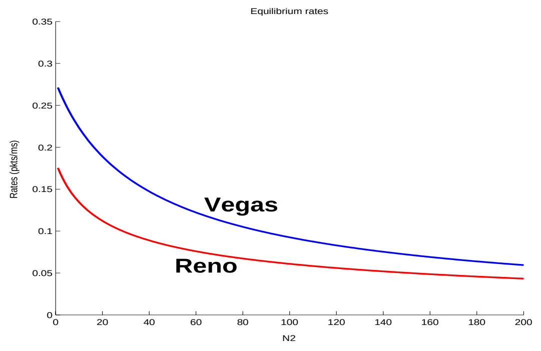
Since $N_1 x_1 + N_2 x_2 = c$, we have

$$\left( \frac{2(\overline{b} - b)}{b - \underline{b}} \right)^{\frac{1}{2}} \frac{N_1}{b + cd_1} + \frac{\alpha_2 d_2 N_2}{b} \quad = \quad 1 \qquad (47)$$

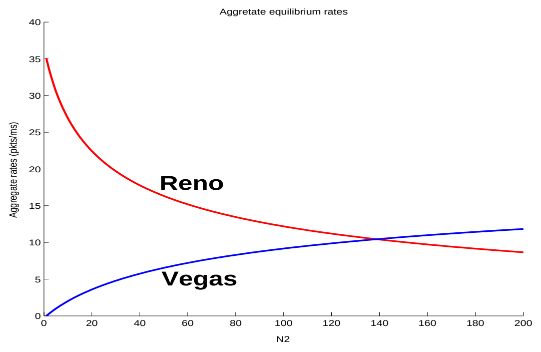
Hence we can obtain equilibrium queue length $b$ by solving (47), and then equilibrium rates using (45)–(46).

All sources have a round trip propagation delay of $d_1 = d_2 = 100$ ms. We fix the number of Reno sources, $N_1 = 200$, and vary the number $N_2$ of Vegas sources from 0 to 200. Each Vegas source has $\alpha_2 = 0.01$ pkts/ms so that it keeps $\alpha_2 d_2 = 1$ pkts in its path in equilibrium. RED parameters are $\underline{b} = 50$ pkts and $\overline{b} = 4000$ pkts. The link capacity is $c = 20$ packets/ms. Figure 3 shows the individual and aggregate rates of Reno and Vegas in equilibrium as the number of Vegas sources increases from 0 to 200. The behavior is qualitatively similar to the interaction of Reno with aggressive binomial sources shown in Figure 1, with individual Vegas sources seizing a larger proportion of bandwidth than individual Reno sources. The aggregate share of all Vegas sources rises as the number of Vegas sources increases.

Each Vegas source keeps $\alpha_2 d_2 = 1$ packet in the link. The equilibrium backlog determines the marking probability $p(b)$ which then determines the source rate of Reno through (45). The rates of Vegas sources are proportional to their shares of the buffer occupancy (see (46)). Figure 4 shows the number of Reno and Vegas packets in the queue. As the number of Vegas sources increases, Vegas packets in the queue exceed Reno packets and they receive a larger aggregate bandwidth. ■



(a) Individual rates



(b) Aggregate rates

Fig. 3. Equilibrium rates under RED as Vegas sources varies from 0 to 200. $N_1 = 200$, $N_2 = 0, \ldots, 200$, $D = 100$ ms, $c = 20$ pkts/ms.

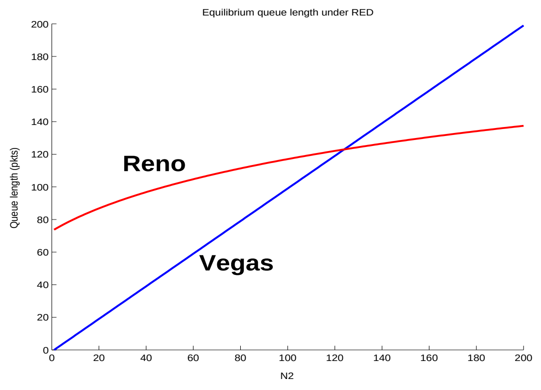


Fig. 4. Equilibrium queue and Vegas share under RED. $N_1 = 200$, $N_2 = 0, \ldots, 200$, $D = 100$ ms, $c = 20$ pkts/ms.

## Example 4: Reno and Vegas under REM

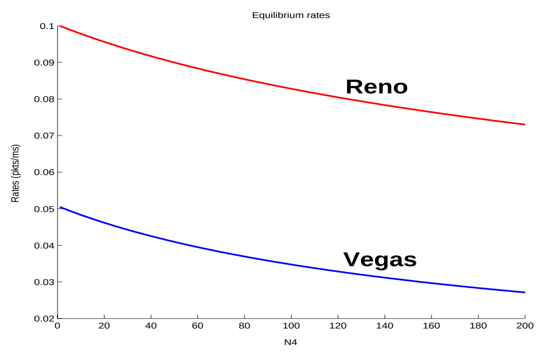
We repeat Example 3 with REM. In this case, the price $r$ can be regarded as the common congestion measure to which Reno and Vegas react. The marking probability is given by (20). We use $\phi = 1.001$ (other REM parameters do not affect equilibrium). Combining (20) with (22) and noting that $D_1 = d_1$ since backlog is zero (Theorem 3), we obtain Reno source rate as:

$$x_1 = \frac{1}{d_1} \frac{1}{\sqrt{(\phi^r - 1)/2}} \qquad (48)$$

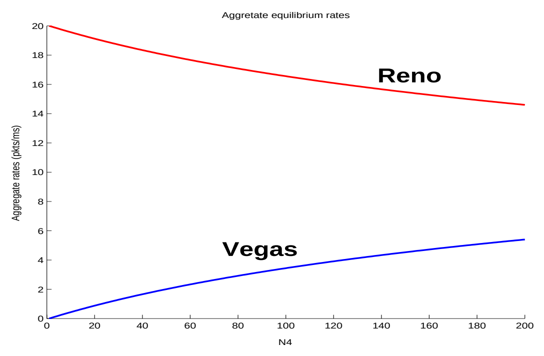
With the Vegas/REM algorithm of [18], Vegas source rates are given by (46) with queueing delay $b/c$ replaced by price $r$. Since $N_1 x_1 + N_2 x_2 = c$, we have

$$\frac{N_1}{d_1} \frac{1}{\sqrt{(\phi^r - 1)/2}} + \frac{\alpha_2 d_2 N_2}{r} = c \qquad (49)$$

Hence we can obtain the equilibrium price $r$ by solving (49) and then rates from (48) and modified (46). The results are shown in Figure 5. For this example,



(a) Individual rates



(b) Aggregate rates

Fig. 5. Equilibrium rates under REM as Vegas sources varies from 0 to 200. $N_1 = 200$, $D = 100$ ms, $c = 20$ pkts/ms.

Vegas receives much less bandwidth than Reno, both individual rates and the aggregate.

## VI. Conclusions

We have presented a duality model of several TCP/AQM protocols. It interprets these protocols as distributed primal-dual algorithms carried out over the Internet in real time to maximize aggregate utility subject to capacity constraints. Different TCP algorithms have different utility functions and we have derived the utility of Reno and Vegas; see Table I. This model can be used to analyze equilibrium properties, such as throughput, loss, delay and fairness, of a network that contains different TCP sources and different AQM links, as long as they use a common measure of congestion.

The duality model has several interesting implications. First, it is well-known that a bottleneck queue can fluctuate about the buffer capacity under Reno or DropTail, generating packet losses. What is more intriguing is that increasing the buffer size does not reduce loss rate significantly, but only grows the queueing delay. According to the duality model, loss probability under Reno is the Lagrange multiplier, and hence its *equilibrium* value is determined solely by the network topology and the utility functions of the sources, *independent of link algorithms and buffer size.* Increasing the buffer size with everything else unchanged does not change the equilibrium loss probability, and hence a larger backlog must be maintained to generate the same loss probability. This means that with DropTail, the buffer at a bottleneck link is always close to full regardless of buffer size. With RED, since loss probability (Lagrange multiplier) is increasing in average queue length, the average queue length must increase steadily as the number of sources grows. Second, it is well-known that TCP Reno discriminates against connections with large propagation delays. This is borne out by the duality model, as discussed in Remark 1 of Section III. TCP Vegas achieves proportional fairness as it has a log utility function (so does Reno-2 approximately). Third, when Reno and Vegas sources share a common network, Vegas sources may receive more, equal, or less bandwidth than Reno sources, depending on the network topology and AQM algorithm at the links. In general, the 'friendliness' of TCP algorithms that adopt different congestion measures depends not only on themselves, but also on their environment such as AQM algorithm and network parameters. This suggests that TCP-friendliness that is defined solely in terms of source algorithm is too restrictive.

We have only studied the equilibrium properties and have ignored the stability and dynamics of these protocols. The global stability of REM in the absence of delay is established in [22] using a Lyapunov argument. Local stability of Reno/RED has been studied in [8], [14]. It would be interesting to investigate delayed global stability of various TCP/AQM protocols. Here we derive the utility functions $U_s$ from rate adjustment algorithms $F_s$. One can turn the question around and tailor utility functions $U_s$ to applications, and then design a TCP algorithm $F_s$ to optimize it.

**Acknowledgment:** We gratefully acknowledge insightful discussions with John Doyle (Caltech), Fernando Paganini (UCLA), and Li Zhu (NJIT) on an earlier version of the paper.

## References

[1] Sanjeewa Athuraliya, Victor H. Li, Steven H. Low, and Qinghe Yin. REM: active queue management. *IEEE Network*, 15(3):48–53, May/June 2001. Extended version in *Proceedings of ITC17*, Salvador, Brazil, September 2001. http://netlab.caltech.edu.

[2] Deepak Bansal and Hari Balakrishnan. Binomial congestion control algorithms. In *Proceedings of IEEE Infocom*, April 2001.

[3] D. Bertsekas. *Nonlinear Programming.* Athena Scientific, 1995.

[4] Lawrence S. Brakmo and Larry L. Peterson. TCP Vegas: end-to-end congestion avoidance on a global Internet. *IEEE Journal on Selected Areas in Communications*, 13(8):1465–80, October 1995. http://cs.princeton.edu/nsg/papers/jsac-vegas.ps.

[5] S. Floyd. Connections with multiple congested gateways in packet–switched networks, Part I: one–way traffic. *Computer Communications Review*, 21(5), October 1991.

[6] 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.

[7] R. J. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 35:1969–1985, 1999.

[8] Chris Hollot, Vishal Misra, Don Towsley, and Wei-Bo Gong. A control theoretic analysis of RED. In *Proceedings of IEEE Infocom*, April 2001. http://www-net.cs.umass.edu/papers/papers.html.

[9] Chris Hollot, Vishal Misra, Don Towsley, and Wei-Bo 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.

[10] V. Jacobson. Congestion avoidance and control. *Proceedings of SIGCOMM'88, ACM*, August 1988. An updated version is available via ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z.

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

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

[13] T. V. Lakshman and Upamanyu Madhow. The performance of TCP/IP for networks with high bandwidth–delay products and random loss. *IEEE/ACM Transactions on Networking*, 5(3):336–350, June 1997. http://www.ece.ucsb.edu/Faculty/Madhow/Publications/ton97.ps.

[14] S. H. Low, F. Paganini, J. Wang and J. C. Doyle. Linear stability of TCP/RED and a scalable control. *Computer Networks Journal*, to appear 2003. http://netlab.caltech.edu.

[15] Steven 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.

[16] Steven H. Low and David 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.

[17] Steven H. Low, Fernando Paganini, and John C. Doyle. Internet congestion control. *IEEE Control Systems Magazine*, 22(1):28–43, February 2002.

[18] Steven H. Low, Larry Peterson, and Limin Wang. Understanding Vegas: a duality model. *J. of ACM*, 49(2):207–235, March 2002. http://netlab.caltech.edu.

[19] L. Massoulie and J. Roberts. Bandwidth sharing: objectives and algorithms. In *Infocom'99*, March 1999. http://www.dmi.ens.fr/\%7Emistral/tcpworkshop.html.

[20] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott. The macroscopic behavior of the TCP congestion avoidance algorithm. *ACM Computer Communication Review*, 27(3), July 1997. http://www.psc.edu/networking/papers/model_ccr97.ps.

[21] Jeonghoon Mo and Jean Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on Networking*, 8(5):556–567, October 2000.

[22] Fernando Paganini. A global stability result in network flow control. *System & Control Letters*, 46(3):153–163, 2002.

[23] Fernando Paganini, John C. Doyle, and Steven 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.

[24] I. M. Ryshik and I. S. Gradstein. *Tables of series, products, and integrals.* VEB Deutscher Verlag der Wissenschaften, Berlin, 1963.

[25] W. Stevens. *TCP/IP illustrated: the protocols*, volume 1. Addison–Wesley, 1999. 15th printing.

**Steven. H. Low** (M 92, SM 99) received his B.S. degree from Cornell University and PhD from the University of California – Berkeley, both in electrical engineering. He was with AT&T Bell Laboratories, Murray Hill, from 1992 to 1996 and with the University of Melbourne, Australia, from 1996 to 2000. He is now an Associate Professor at the California Institute of Technology, Pasadena. He has held visiting academic positions in the US and Hong Kong, and has consulted with companies and governments in the US and Australia. He was a co-recipient of the IEEE Bennett Prize Paper Award in 1997 and the 1996 R&D 100 Award. He is on the editorial board of *IEEE/ACM Transactions on Networking* and *Computer Networks Journal*. His research interests are in the control and optimization of networks and protocols. His home is netlab.caltech.edu and email is slow@caltech.edu.