Hindawi
Wireless Communications and Mobile Computing
Volume 2019, Article ID 6941248, 10 pages
https://doi.org/10.1155/2019/6941248

Research Article

Smart-RED: A Novel Congestion Control Mechanism for High Throughput and Low Queuing Delay

Armir Bujari,[1] **Andrea Marin**,[2] **Claudio E. Palazzi**,[1] **and Sabina Rossi**[2]

[1] *Università di Padova, Italy*
[2] *Università Ca' Foscari Venezia, Italy*

Correspondence should be addressed to Sabina Rossi; sabina.rossi@unive.it

Received 28 June 2018; Accepted 20 March 2019; Published 4 April 2019

Academic Editor: Pierre-Martin Tardif

We consider the scenario in which several TCP connections share the same access point (AP) and a congestion avoidance/control mechanism is adopted with the aim of assigning the available bandwidth to the clients with a certain fairness. When UDP traffic with real-time requirements is present, the problem becomes even more challenging. Very well-known congestion avoidance mechanisms are the Random Early Detection (RED) and the Explicit Congestion Notification (ECN). More recently, the Smart Access Point with Limited Advertised Window (SAP-LAW) has been proposed. Its main idea is that of computing the maximum TCP rate for each connection at the bottleneck, taking into account the UDP traffic to keep a low queue size combined with a reasonable bandwidth utilization. In this paper, we propose a new congestion control mechanism, namely, Smart-RED, inspired by SAP-LAW heuristic formula. We study its performance by using mean field models and compare the behaviours of ECN/RED, SAP-LAW, and Smart-RED under different scenarios. We show that while Smart-RED maintains some of the desirable properties of the SAP-LAW, it solves the problems it may have in case of bursty UDP traffic or TCP connections with very different needs of bandwidth.

1. Introduction

Nowadays, the use of a single access point (AP) to retrieve Internet contents both with and without real-time requirements is a widespread scenario. In this context, the AP bandwidth must be assigned to the set of clients according to some principles which are not always easy to satisfy given the well-known problems connected with priority labeling of the Internet traffic (see, e.g., [1]). The TCP congestion control has been used to regulate the connection sending rates in presence of a bottleneck such as an AP [2–5]. The goal is to provide a fair assignment of the bandwidth to each TCP flow sharing the bottleneck. This can be achieved at the AP by implementing classic algorithms such as the Random Early Detection (RED) [6] and the Explicit Congestion Notification (ECN) [7]. These approaches are based on a random selection of the packets enqueued at the AP which are marked and discarded (RED) or an explicit notification with a request to slow down that is sent to the sender (ECN). In both cases, the TCP reacts by reducing its congestion window size (assuming a packet loss in case of RED). ECN/RED has been widely studied in the literature (see, e.g., [8–11]) and shown effective in regulating the TCP traffic. Nevertheless, ECN/RED takes into account only the TCP traffic and ignores the UDP since the latter, at the transport layer, does not provide any mechanism for regulating the transmission rate. In [12] the authors show that, in some cases, greedy TCP connections may occupy great part of the AP bandwidth and this directly penalizes the performance of UDP traffic whose packets have to wait in the queue. Unfortunately, if queueing time is usually not a big issue for TCP traffic (e.g., file downloading) it represents a serious problem for UDP communications which are often not delay tolerant. This problem is further exacerbated in absence of buffer management techniques in routers [13–15]. At the access tier, the IEEE 802.11e standard was proposed to allow different classes of traffic to provide different QoS levels especially in terms of achieved throughput. Unfortunately, it does not guarantee low per-packet delays and the problem of having the traffic properly marked without risk of errors or cheating still remains along with several other problems jeopardizing its deployability [16, 17].

Instead, in [12] the authors proposed the Smart Access Point with Limited Advertising Window (SAP-LAW) which exploits the flux control mechanism of regular TCP implementations. The algorithm was then further refined in [18] to be used also in cloud systems. In practice, the AP hijacks the acknowledgement packets by changing the advertising window size to a computed size which allows a queue length near to zero. The size is simply computed as the AP's TCP capacity, i.e., the difference between its total capacity and an estimation of the instantaneous UDP traffic, divided by the number of active TCP connections. This approach has been shown to be effective under two conditions: (1) when TCP connections are greedy, i.e., all of them need the highest available bandwidth, and (2) when the UDP traffic is constant or changes slowly in time. When condition (1) is violated, SAP-LAW unnecessarily allocates portions of bandwidth to TCP connections causing a reduction of the AP utilization, while when condition (2) is violated the protocol may fail to keep a short queue length. Indeed, since SAP-LAW never considers the queue length in its decision-making algorithm (it is based only on the estimation of the instantaneous UDP traffic), a bursty UDP traffic may cause the congestion of the AP as we discuss later on in Section 2.

In this paper, we combine ECN/RED and SAP-LAW in a new protocol called Smart-RED which aims at solving the drawbacks of SAP-LAW. In Smart-RED the SAP-LAW algorithm is activated only when a threshold (min_{th}) is exceeded. The introduction of the minimum threshold reduces the effects of the violation of condition (1). The application of the ECN/RED when the queue length is greater than another threshold $max_{th} > min_{th}$ ensures that the AP is never congested (and hence the problems of the violation of condition (2) are overcome). In order to compare the performance of the protocols, we assume a large number of TCP connections and agents employing UDP transmissions, all going through a common AP in order to access the Internet. We show that the model converges to a mean field regime, allowing us to perform an efficient deterministic simulation of the system and a consequent computation of the performance indices.

A large number of papers have studied the performance of TCP and ECN; see [8, 19–21] as a not exhaustive list, but the main focus is on the behavior of a single TCP connection under various scenarios. Other works focused on the specific TCP versions installed by default in the Linux and Windows operating systems and developed models studying their performance [22–24]. More similar to the analysis carried out here are those proposed in [9, 25, 26] where the authors consider the overall system performance in the mean field limit for ECN/RED. The theoretical results proposed in [9] are then extended in [10]. Starting from these two papers we extend the considered model in several directions, as discussed later on, and we propose new ones for SAP-LAW and Smart-RED. Mean field theory (see, e.g., [10, 27, 28]) is an important approach used to study large stochastic models with a deterministic approximation thus overcoming the well-known problem of the state space explosion.

In [29] we propose a mean field model for TCP and UDP traffic handled by ECN/RED or SAP-LAW. With respect to this work, in this paper we introduce a new congestion control mechanism that combines the two previous protocols and compare their performance under different scenarios. While the modelling technique that we use in this paper is close to that of [29], the model that we derive is a nontrivial generalisation of the previous ones. As a consequence, the models for ECN/RED and SAP-LAW presented in [29] can be obtained as different instantiations of the parameters of the model presented in this paper. The results presented in Section 6 are related to the comparison of the performance of the new protocol Smart-RED with ECN/RED and SAP-LAW.

In Section 2, we review the salient features of the congestion control mechanism of TCP protocols and those of ECN, RED, and SAP-LAW algorithms, respectively. In the same section, we discuss some issues associated with the latter protocol. In Section 3 we explain the theoretical background, introduce the notation, and enunciate the mean field theorem that is used to prove the convergence of the models. Next, in Section 4, we describe the Smart-RED novel approach. In Section 5, we develop the models for RED, SAP-LAW, and Smart-RED used in Section 6 to compare the protocols under different scenarios. Finally, in Section 7 we give some final remarks and conclude our paper.

2. TCP and Congestion Control

In this section, we briefly review the congestion control mechanism implemented by the ECN/RED and SAP-LAW algorithms.

(a) ECN/RED. Let us consider a set of clients equipped with a TCP implementation sharing an Internet connection via a wireless access point. ECN and RED are used in conjunction as congestion avoidance mechanism working at the access point in order to regulate the TCP window sizes with the aim of preventing congestion at the AP. In this paper we consider TCP connections over a ECN/RED AP as done in [9, 10]. ECN/RED works with two thresholds: min_{th} and max_{th}. It accepts all the packets when the queue length ql at the access point is less than the minimum threshold min_{th}. When ql is between the minimum and the maximum thresholds, each arriving packet is marked with a probability p, which is a function depending on ql. If ql is greater than or equal to the maximum threshold max_{th}, all the incoming packets will be marked. Either all the marked packets are dropped or a notification of congestion is sent to the client depending on whether the RED is used in conjunction with ECN. Notice that ECN/RED exploits only the TCP congestion control protocol and it does not regulate the UDP traffic.

(b) SAP-LAW. SAP-LAW is a solution proposed in [12] with the specific aim of addressing the problems caused by the simultaneous presence of both TCP and UDP-based applications. The basic idea is to find a trade-off between throughput and time delay. This is achieved by dynamically modifying the TCP sending rate in order to avoid congestion while guaranteeing a high utilization of the bandwidth. The key factor is to determine the upper bound for the TCP

sending rate as a function of the amount of UDP traffic. In this way, we are always sure to reserve enough bandwidth for the real-time application activities. The general formula proposed by Palazzi et al. [12] is

$$maxTCP_{\text{traffic}}(t) = \frac{\left(C - UDP_{\text{traffic}}(t)\right)}{\#TCP_{\text{flows}}(t)}, \tag{1}$$

where $UDP_{\text{traffic}}(t)$ is the amount of bandwidth occupied by the real-time applications at time t, C is the capacity of the bottleneck link, and $\#TCP_{\text{flows}}(t)$ is the number of TCP connections at time t. SAP-LAW regulates the TCP sending rate by taking into account both the TCP and the UDP traffic. The approach provides an optimal solution when TCP connections are trying to utilize the highest possible bandwidth and when UDP traffic is smooth. However, two major issues arise: first, when UDP traffic is bursty, SAP-LAW may not be able to avoid congestion (see Proposition 1) and second, it allocates the same bandwidth to TCP connections with low and high bandwidth requirements, which might lead to a low utilization of the channel.

Proposition 1. *When the peaks of the sending rates of UDP agents are periodically higher than the AP capacity, then SAP-LAW fails in preventing congestion.*

Intuitively, the proof is based on the fact that when SAP-LAW realizes the presence of a peak of UDP traffic, it reduces the TCP sending rate only for the duration of the burst. As a consequence, immediately after, it allows an aggressive behaviour of the TCP connections. If the sending rate during the peak is higher than the AP capacity, the consequence is that the queue which is left at that time slot is not consumed.

3. Theoretical Background

Mean field theory is widely used in many research domains, including Physics, Epidemiology, and Computer Science. The main idea behind this analysis technique is that the interaction of many objects whose behaviour is stochastic tends to become deterministic, at least from the prospective of an observer that does not distinguish the identity of a single object but knows the fraction of them that is in a certain state at a certain time. In this paper, we refer to the mean field results proposed by Le Boudec et al. in [10].

Let us consider a system consisting of N objects belonging to distinct classes. The state of an object is represented by a pair (c, i), where $c \in \Gamma$ is the class of the object and $i \in \mathscr{E}_c = \{1, \ldots, S_c\}$ is its internal state. Objects can interact with the environment and change their state.

Let $X_n^N(t)$ be the state of the n-th object at time slot t, and let $K^N_{(c,i),(c,j)}(t)$ be the probability that an object of class c goes from state i to state j during the time slot t.

The peculiarity of this approach to mean field is that the model has a memory which may affect the behaviour of the objects. We represent this memory by $\overrightarrow{R}^N(t) \in \mathbb{R}^d$, where d is some fixed integer. As stated before, in the mean field analysis, the observer cannot see the state of a single object, but it is able to see the fraction of objects that are in certain state. For this reason, we introduce the *occupancy measure*, $\overrightarrow{M}^N(t)$, a vector whose components $M^N_{c,i}(t)$ represent the proportion of objects that are in state (c, i) at time t, i.e.,

$$M^N_{c,i}(t) = \frac{1}{N}\sum_{n=1}^{N} 1_{\{X_n^N(t)=(c,i)\}}, \tag{2}$$

where $1_{x=(c,i)}$ is the indicator function. If we condition on the object class, we have that the proportion of class c objects in state i at time t is $(1/p_c)M^N_{c,i}(t)$, where $p_c = \sum_{i=1}^{S_c} M_{c,i}(t)$.

To govern the system dynamics, we assume that there is a deterministic continuous function g that, given the states of the memory and of the occupancy vector, updates the state of the memory:

$$\overrightarrow{R}^N(t+1) = g\left(\overrightarrow{R}^N(t), \overrightarrow{M}^N(t+1)\right). \tag{3}$$

In conclusion, $\forall n$ we have

$$\begin{aligned}&\mathbb{P}\left\{X_n^N(t+1) = (c,j) \mid \overrightarrow{R}^N(t) = \overrightarrow{r}, X_n^N(t) = (c,i)\right\}\\ &\quad = K^N_{(c,i),(c,j)}\left(\overrightarrow{r}\right).\end{aligned} \tag{4}$$

$K^N_{(c,i),(c,j)}$ is the transition probability from state (c, i) to state (c, j) and it depends on the memory state. Therefore we assume $K^N_{(c,i),(c,j)}(\overrightarrow{r}) \geq 0$ and $\sum_{j=1}^{S_c} K^N_{(c,i),(c,j)}(\overrightarrow{r}) = 1$ for all $1 \leq i \leq S_c$.

According to [10], we assume the following.

Assumption 2. $\forall c \in \Gamma, \forall i, j \in \mathscr{E}_c$, and for $N \longrightarrow \infty$, $K^N_{(c,i),(c,j)}(\overrightarrow{r})$ converges uniformly in $\overrightarrow{r}$ to some $K_{(c,i),(c,j)}(\overrightarrow{r})$, which is a continuous function of $\overrightarrow{r}$.

A sufficient condition for this assumption to be satisfied is that the transition matrix $K^N(\overrightarrow{r})$ does not depend on the number of objects N and that is continuous on $\overrightarrow{r}$, which is always the case in our examples. We conclude this section by presenting Theorem 1 of [10]. The theorem will be widely used in the modelling sections of this paper and is a formalisation of what has been informally described at the beginning of this section.

Theorem 3. *Assume that the initial occupancy measure* $\overrightarrow{M}^N(0)$ *and memory* $\overrightarrow{R}^N(0)$ *converge almost surely to deterministic limits* $\overrightarrow{\mu}(0)$, $\overrightarrow{\rho}(0)$ *and for* $t \geq 0$,

$$\overrightarrow{\mu}(t+1) = \overrightarrow{\mu}(t) K\left(\overrightarrow{\rho}(t)\right) \tag{5}$$

$$\overrightarrow{\rho}(t+1) = g\left(\overrightarrow{\rho}(t), \overrightarrow{\mu}(t+1)\right). \tag{6}$$

Then for any fixed time t, *almost surely*

$$\begin{aligned}\lim_{N\longrightarrow\infty} &= \overrightarrow{\mu}(t),\\ \lim_{N\longrightarrow\infty} \overrightarrow{R}^N(t) &= \overrightarrow{\rho}(t).\end{aligned} \tag{7}$$

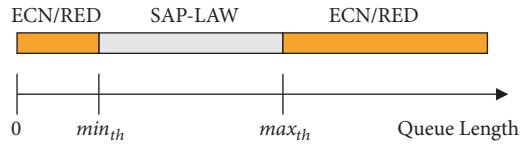

Figure 1: Smart-RED: regions of application of ECN/RED and SAP-LAW control policies.

(1) **for** each arriving packet **do**
(2) **if** $ql < min_{th}$ **then**
(3) accept that packet
(4) **else if** $min_{th} \leq ql < max_{th}$ **then**
(5) **set** $rwnd = (C - UDP_{traffic})/\#TCP_{\text{flows}}$
(6) **else if** $max_{th} \leq ql$ **then**
(7) mark that packet
(8) **end if**
(9) **end for**

Algorithm 1: Smart-RED Algorithm.

4. Smart-RED: A Novel Approach

In this section, we describe the algorithm that we propose for the congestion control problem at the AP: Smart-RED. This is based on a combination of ECN/RED and SAP-LAW. In contrast with the SAP-LAW, but similarly to ECN/RED, Smart-RED is unconditionally stable. However, it resorts to the SAP-LAW flux control approach only when the queue length is greater than a given threshold. Smart-RED works with two thresholds min_{th} and max_{th} and behaves as SAP-LAW when the queue length is between min_{th} and max_{th}; i.e., the advertised window is computed using (1). When the queue length is lower than min_{th} or greater than max_{th}, the Smart-RED algorithm behaves as RED (see Figure 1).

Smart-RED works as described in Algorithm 1.

5. Models for Smart-RED, ECN/RED, and SAP-LAW

Smart-RED combines the ECN/RED and the SAP-LAW techniques to overcome the drawbacks of these two approaches to the problem of congestion control. In this section we propose three different models for ECN/RED, SAP-LAW, and Smart-RED, respectively, in order to study their performance in the following scenarios.

(i) We consider the UDP traffic which may be either bursty or smooth.

(ii) We regulate the life cycle of TCP connections in order to be able to study the behaviour of the three protocols according to the bandwidth needs of the TCP connections.

The proposed models are showed to converge to a mean field regime and then they are studied; i.e., we assume that the number of TCP connections and UDP transmissions are very high (tend to be infinite). Henceforth, we will simply talk of a mean field model to refer to a modelling technique that converges to a mean field regime.

5.1. Modelling Smart-RED with UDP Traffic and TCP Connections. The Smart-RED mean field model is derived from that of ECN/RED presented in [29] by applying the SAP-LAW formula to update the memory of the system. We work in a discrete time setting [9, 10, 29] and consider two classes of objects: the TCP flows which will be denoted by p and the UDP sources denoted by u. We need to distinguish these two types of objects since TCP transmissions have a mechanism to control the transmission rate, while UDP do not. Let N_p and N_u denote the number of objects associated with TCP and UDP flows, respectively, and $N = N_p + N_u$, while $p_p = N_p/N$ and $p_u = N_u/n$ denote the fraction of TCP and UDP objects, respectively.

(a) Modelling of TCP State. TCP flows use the window mechanisms to control their transmission speed. In particular, each transmitting agent maintains a sending window to store the packets that have not been acknowledged by the receiver yet. The size of these windows determines the transmission speed. In our model, the window of TCP flows can have I_p states, ranging from 1 to I_p. When the window is in state $i \in [1, \ldots, I_p]$, then the sender can transmit $s_p(i) \in \mathbb{N}$ packets in a time slot. s_p is a monotonically increasing function and, henceforth, we assume that it has the form $s_p(i) = \alpha i$ for some $\alpha \in \mathbb{N}^+$.

Each TCP agent state encodes the state of its sending window and the number of packets that have still to be sent denoted by $s \in [1, \ldots, s_p(I_p)+1]$, where I_p+1 denotes the fact that the sender cannot fit all its queue of packets in one time slot. This compact way of storing the remaining packets to be sent takes advantage of the assumption that the TCP flows have geometric distributed sizes. For TCP objects, we have that the occupancy vector components are $M^N_{p,i,s}(t)$, where p denotes the TCP class, i the state of the window, and s the backlog of the packets. The proportion of TCP connections in state (i, s) at time t is then $p_p^{-1} M^N_{p,i,s}(t)$.

(b) Modelling the UDP State. UDP flows do not have any window to regulate their transmission rate. We imagine that each application that interacts with our bottleneck is in a state that demands some transmission rate. For each UDP agent we introduce I_u states ranging from 1 to I_u. When an UDP agent is in state $i \in [1, \ldots, I_u]$ it sends $s_u(i) \in \mathbb{N}$ packets in a time slot. The occupancy measure for UDP agents is denoted by $M^N_{u,i}(t)$ and the proportion of UDP agents in state i at time t is then $p_u^{-1} M^N_{u,i}(t)$.

(c) Modelling the State of the Memory. We assume the bottleneck to have a capacity of C packets for time slot for each object (TCP or UDP) that interacts with it. The memory must allow us to estimate the UDP traffic intensity in the latest interval of time. Therefore, we assume the time interval that we use to estimate the instantaneous arrival rate for the UDP packets to be equal to a multiple Y of the round trip time; i.e., $\xi(t) = y(S_u^N(t-Y), \ldots S_u^N(t-1))$, where $\xi(t)$ denotes the intensity of the UDP traffic estimated at time t. Thus, the memory $\overrightarrow{R}^N(t) = (R_c^N(t), R_p^N(t), \overrightarrow{R}_u^N(t))$ is a pair of real numbers (r_c, r_p) followed by a vector of reals $\overrightarrow{r}_u$ where r_c denotes the normalised queue length at the bottleneck (counting both TCP and UDP packets) at a given time slot, while r_p is the normalised queue length at the previous time slot and $\overrightarrow{r}_u = (r_{u0}, \ldots, r_{uY})$ denotes the normalized counting of the arrived UDP packets at the latest $(Y+1) > 1$ time slots. We will denote by $\overrightarrow{r}$ a possible element of $\overrightarrow{R}^N(t)$. The dynamic of the memory is specified by the following equations:

$$\begin{aligned} R_c^N(t+1) &= \max\left(R_c^N(t) + S_p^N(t+1) + S_u^N(t+1) - C, 0\right), \\ R_p^N(t+1) &= R_c^N(t), \\ R_{ui}^N(t+1) &= R_{u(i-1)}^N(t), \quad 1 \le i \le Y, \\ R_{u0}^N(t+1) &= S_u^N(t+1). \end{aligned} \tag{8}$$

$S_p^N(t+1)$ and $S_u^N(t+1)$ are the total traffic generated by TCP and UDP agents, respectively. The traffic generated by UDP agents is rather simple and can be derived as

$$S_u^N(t) = \sum_i^{I_u} s_u(i)\, M_{u,i}^N(t). \tag{9}$$

On the other hand, the traffic generated by the TCP agents is more complicated and follows the rules specified in Section 4:

$$\begin{aligned} S_p^N(t) = \sum_{i=1}^{I_p} \sum_{s=1}^{s_p(I_p)+1} & M_{p,i,s}(t+1) \\ & \cdot \min\left(s, s_p\left(h\left(\overrightarrow{R}_u^N(t)\right)\right)\right) \cdot 1_{R_u^N(t) \ge min_{th}} \\ & + M_{p,i,s}(t+1) \min\left(s_p(i), s\right) \cdot 1_{R_u^N(t) < min_{th}} \end{aligned} \tag{10}$$

We note that we use the memory to know the UDP traffic at the previous slot and hence to decide the bandwidth available within the minimum and maximum thresholds. Function h is defined as follows:

$$h\left(\overrightarrow{r}_u\right) = \arg\max_j \left(s_p(j), s_p(j) \le \left(C - \tilde{y}\left(\overrightarrow{r}_u\right)\right) \frac{1}{p_p} \vee j = 1 \right), \tag{11}$$

and $\tilde{y}(\overrightarrow{r}_u) = y(r_{u1}, \ldots, r_{uY})$.

(d) Modelling the Change of State of TCP Agents. The state transition matrix for each TCP object in the Smart-RED model is defined as follows:

$$\begin{aligned} & \tau\left(s'\right) \\ & = w(1-w)^{s'-1} 1_{s' \le s_p(I_p)} + (1-w)^{s_p(I_p)} 1_{s' = S_p(I_p)+1} \end{aligned} \tag{12}$$

$$\begin{aligned} & K_{(p,i,s),(p,i+1,s')}\left(\overrightarrow{r}\right) \\ & = \left(1 - q\left(r_p\right)\right)^{s_p(i)} \cdot 1_{i < I_p, s > s_p(i), r_p < min_{th}} \tau\left(s'\right) \\ & \quad + \left(1 - q\left(r_p\right)\right)^{\min(s, s_p(h(\overrightarrow{R}_u^N(t))))} \\ & \quad \cdot 1_{i < I_p, s > \min(s, s_p(h(\overrightarrow{R}_u^N(t)))), r_p \ge min_{th}} \tau\left(s'\right) \end{aligned} \tag{13}$$

$$K_{(p,i,s),(p,1,s')}\left(\overrightarrow{r}\right) = 1_{s \le s_p(i)} \tau\left(s'\right) \tag{14}$$

$$\begin{aligned} & K_{(p,I_p,s),(p,I_p,s')} \\ & = \left(1 - q\left(r_p\right)\right)^{s_p(I_p)} 1_{s > s_p(i), r_p < min_{th}} \tau\left(s'\right) \\ & \quad + \left(1 - q\left(r_p\right)\right)^{\min(s, s_p(h(\overrightarrow{R}_u^N(t))))} \\ & \quad \cdot 1_{s > \min(s, s_p(h(\overrightarrow{R}_u^N(t)))), r_p \ge min_{th}} \tau\left(s'\right) \end{aligned} \tag{15}$$

$$\begin{aligned} & K_{(p,i,s),(p,d(i),s')} \\ & = \left(1 - \left(1 - q\left(r_p\right)\right)^{s_p(i)}\right) \cdot 1_{s > s_p(i), r_p < min_{th}} \tau\left(s'\right) \\ & \quad + \left(1 - \left(1 - q\left(r_p\right)\right)^{\min(s, s_p(h(\overrightarrow{R}_u^N(t))))}\right) \\ & \quad \cdot 1_{s > \min(s, s_p(h(\overrightarrow{R}_u^N(t)))), r_p \ge min_{th}} \tau\left(s'\right) \end{aligned} \tag{16}$$

Function $h(\overrightarrow{r}_u)$ is defined in (11), while Function τ is the density function of a truncated geometric random variable, where the probability mass of outcomes greater than $s_p(I_p)$ is concentrated in the last state, $s_p(I_p)+1$ (see [29]). Function $d(i)$ is used to model the destination state in case of a marked packet. As in [9] we have $d(i) = \lfloor i/2 \rfloor$ (and the number of packets sent $s_p(i)$ is proportional to i). Equation (13) controls the transition from state i to $i+1$ of a TCP window. Equation (14) defines the transition to the window size 1 when a TCP connection ends its transmission and a new one is begun. Equation (15) considers the case in which the window size is at its maximum and remains there (but with a possibly different amount of remaining packets to send). Finally, (16) models the case in which a window reduces its size because a packet waiting for acknowledgement goes in time out due to the marking of ECN/RED. Function $q(R_p^N(t))$ is specified in the following equation:

$$q\left(R_p^N(t)\right) = \begin{cases} 0 & \text{if } R_p^N(t) < max_{th} \\ 1 & \text{if } R_p^N(t) \ge max_{th} \end{cases} \tag{17}$$

(e) Modelling the Change of State of UDP Agents. The probabilistic description of the behaviour of the agents modelling UDP flows is rather simple since it does not depend on external factors encoded in the memory. Recall that UDP does not have an internal congestion or flow control mechanisms. Thus, we have

$$K_{(u,i),(u,j)}\left(\overrightarrow{r}\right) = \kappa_{(u,i),(u,j)}, \quad 1 \le i, j \le I_u \tag{18}$$

$\kappa_{(u,i),(u,j)} \in [0,1]$ and for all $i \in [1, I_u]$ we have

$$\sum_{j=1}^{I_u} \kappa_{(u,i),(u,j)} = 1. \tag{19}$$

We are now in position to give the most important result of this section, i.e., the mean field model corresponding to the limit for $N \longrightarrow \infty$ of the described probabilistic model.

Proposition 4. *If $\overrightarrow{M}^N(0)$ and $R^N(0)$ converge almost surely to $\overrightarrow{\mu}(0)$ and $\overrightarrow{\rho}(0)$, respectively, and $\forall i = 1, \ldots, I_p$ it holds that $M^N_{p,i,s}(0)/\sum_{s'=1}^{s_p(I_p)+1} M_{p,i,s'}$ converge almost surely to $\tau(s)$, as $N \longrightarrow \infty$, then for any finite horizon t we have that*

$$\begin{aligned}
\lim_{N\longrightarrow\infty} \sum_{s=1}^{s_p(I_p)+1} M^N_{p,i,s}(t) &= \widetilde{\mu}_{p,i}(t) \\
\lim_{N\longrightarrow\infty} M^N_{u,i}(t) &= \mu_{u,i}(t) \\
\lim_{N\longrightarrow\infty} \overrightarrow{R}^N(t) &= \overrightarrow{\rho}(t)
\end{aligned} \tag{20}$$

almost surely, where $\widetilde{\mu}_{p,i}(t)$, $\mu_{u,i}(t)$, and $\rho(t)$ are defined by the Iteration system for Smart-RED model with TCP connections as follows:

$$\begin{aligned}
&\widetilde{\mu}_{p,1}(t+1) \\
&= \sum_{j:d(j)=1} \left(1 - \left(1 - q\left(\rho_p(t)\right)\right)^{\min(s_p(j), s_p(h(\overrightarrow{\rho}_u(t))))}\right) \\
&\cdot \widetilde{\mu}_j(t) (1-w)^{\min(s_p(j), s_p(h(\overrightarrow{\rho}_u(t))))} .1_{\rho_p(t) \ge min_{th}} \\
&+ \sum_{j:d(j)=1} \left(1 - \left(1 - q\left(\rho_p(t)\right)\right)^{s_p(j)}\right) \widetilde{\mu}_j(t) \\
&\cdot (1-w)^{s_p(j)} .1_{\rho_p(t) < min_{th}} + \sum_{j=1}^{I_p} \widetilde{\mu}_{p,j}(t) \\
&\cdot \left(1 - (1-w)^{\min(s_p(j), s_p(h(\overrightarrow{\rho}_u(t))))}\right) .1_{\rho_p(t) \ge min_{th}} \\
&+ \widetilde{\mu}_{p,j}(t) \left(1 - (1-w)^{s_p(j)}\right) .1_{\rho_p(t) < min_{th}}
\end{aligned}$$

$$\begin{aligned}
&\widetilde{\mu}_{p,i}(t+1) \\
&= \sum_{j:d(j)=i} \left(1 - \left(1 - q\left(\rho_p(t)\right)\right)^{\min(s_p(j), s_p(h(\overrightarrow{\rho}_u(t))))}\right) \\
&\cdot \widetilde{\mu}_j(t) (1-w)^{\min(s_p(j), s_p(h(\overrightarrow{\rho}_u(t))))} .1_{\rho_p(t) \ge min_{th}} \\
&+ \sum_{j:d(j)=i} \left(1 - \left(1 - q\left(\rho_p(t)\right)\right)^{s_p(j)}\right) \widetilde{\mu}_j(t) \\
&\cdot (1-w)^{s_p(j)} .1_{\rho_p(t) < min_{th}} + \Big(1 \\
&- q\left(\rho_p(t)\right)\Big)^{\min(s_p(i-1), s_p(h(\overrightarrow{\rho}_u(t))))} \widetilde{\mu}_{p,i-1}(t) (1 \\
&- w)^{\min(s_p(i-1), s_p(h(\overrightarrow{\rho}_u(t))))} .1_{\rho_p(t) \ge min_{th}} + \Big(1 \\
&- q\left(\rho_p(t)\right)\Big)^{s_p(i-1)} \widetilde{\mu}_{p,i-1}(t) (1-w)^{s_p(i-1)} .1_{\rho_p(t) < min_{th}} \\
&\qquad\qquad 1 < i < I_p
\end{aligned}$$

$$\begin{aligned}
&\widetilde{\mu}_{p,I_p}(t+1) = \left(1 - q\left(\rho_p(t)\right)\right)^{\min(s_p(I_p-1), s_p(h(\overrightarrow{\rho}_u(t))))} \\
&\cdot \widetilde{\mu}_{p,I_p-1}(t) (1-w)^{\min(s_p(I_p-1), s_p(h(\overrightarrow{\rho}_u(t))))} .1_{\rho_p(t) \ge min_{th}} \\
&+ \left(1 - q\left(\rho_p(t)\right)\right)^{s_p(I_p-1)} \widetilde{\mu}_{p,I_p-1}(t) (1-w)^{s_p(I_p-1)} . \\
&1_{\rho_p(t) < min_{th}} + \left(1 - q\left(\rho_p(t)\right)\right)^{\min(s_p(I_p), s_p(h(\overrightarrow{\rho}_u(t))))} \\
&\cdot \widetilde{\mu}_{p,I_p}(t) (1-w)^{\min(s_p(I_p), s_p(h(\overrightarrow{\rho}_u(t))))} .1_{\rho_p(t) \ge min_{th}} \\
&+ \left(1 - q\left(\rho_p(t)\right)\right)^{s_p(I_p)} \widetilde{\mu}_{p,I_p}(t) (1-w)^{s_p(I_p)} . \\
&1_{\rho_p(t) < min_{th}}
\end{aligned}$$

$$\begin{aligned}
&\mu_{u,i}(t+1) = \sum_{j=1}^{I_u} \kappa_{(u,j),(u,i)} \mu_{u,j}(t) \\
&\sigma_p(t+1) = \sum_{i=1}^{I_p} \widetilde{\mu}_{p,i}(t+1) \frac{1-(1-w)^{s_p(i)}}{w} .1_{\rho_p(t) < min_{th}} \\
&\quad + \widetilde{\mu}_{p,i}(t+1) \frac{1-(1-w)^{min(s_p(i), s_p(h(\overrightarrow{\rho}_c(t))))}}{w} . \\
&\quad 1_{\rho_p(t) \ge min_{th}} \\
&\sigma_u(t+1) = \sum_{i=1}^{I_u} \mu_{u,i}(t+1) s_u(i) \\
&\rho_c(t+1) = \max\left(\rho_c(t) + \sigma_p(t+1) + \sigma_u(t+1)\right. \\
&\quad \left. - C, 0\right) \\
&\rho_p(t+1) = \rho_c(t) \\
&\rho_{u(c+1)}(t+1) = \rho_{uc}(t), \\
&\qquad\qquad 0 \le c < Y \\
&\rho_{u0}(t+1) = \sigma_u(t+1)
\end{aligned} \tag{21}$$

5.2. Model for ECN/RED. We previously defined a model for ECN/RED following the principles described in this paper

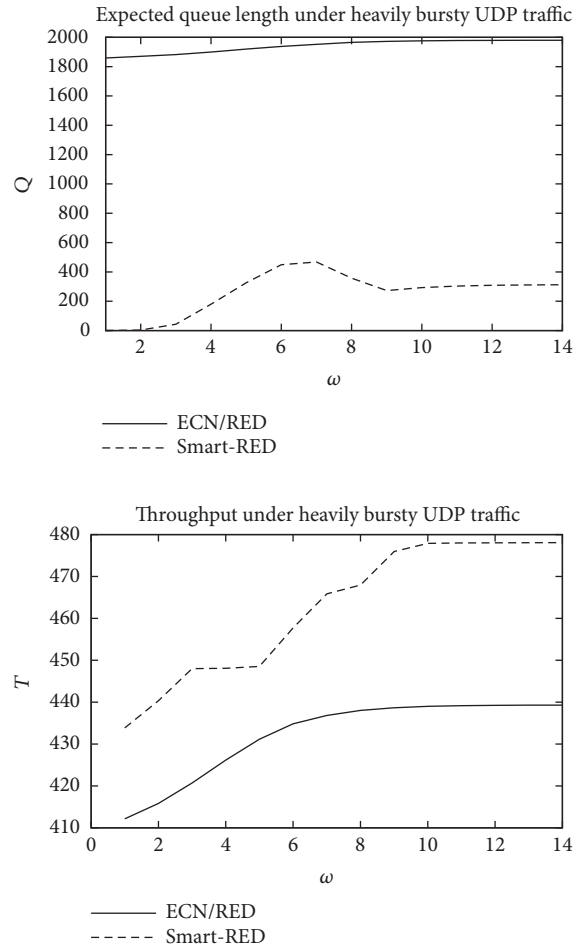

Figure 2: Comparison of ECN/RED and Smart-RED under heavily bursty UDP traffic.

in [29]. In order to keep the paper self-contained we briefly discuss how it is possible to derive the model of [29] from the one proposed here. First, we desire our Smart-RED to work only in the ECN/RED policy. We can achieve this by setting $min_{th} = max_{th} = \infty$. Moreover, since we do not observe the history of UDP packets we have to evict the vector $\overrightarrow{R}_u^N(t)$ from the memory state $\overrightarrow{R}^N(t)$. Finally, we need to update the packet marking function as follows [6, 7, 30]:

$$q\left(R_p^N(t)\right) = 1 - \exp\left(-\gamma R_p^N(t)\right) \tag{22}$$

5.3. Model for SAP-LAW. Analogously of what we have done for the ECN/RED, we can derive the model for SAP-LAW as special case of that proposed for Smart-RED. In fact, we can just set $min_{th} = 0$ and $max_{th} = \infty$ in order to force Smart-RED to work exactly as if it were a SAP-LAW protocol.

6. Performance Evaluation

In this section, we evaluate the performance of Smart-RED with ECN/RED and SAP-LAW. A mean field model is implemented to compare the three different protocols with respect to the following performance indices: expected queue length Q (expressed in normalized number of packets at the AP) and throughput T (expressed in normalized sent packets per time slot). To this end, we consider different scenarios. All the mean field simulations performed in the following sections share the following set of parameters unless differently specified: $I_p = 100$, $I_u = 15$, $C = 500$, $s_p(i) = 10i$, $min_{th} = 30$, $max_{th} = 1500$, $p_u = 0.3$, $p_p = 0.7$, and $\gamma = 5E-6$.

We point out that these modelling and simulation approaches have been previously validated by means of simulations performed in NS2 [29].

6.1. Instability of SAP-LAW. We assume that each TCP connection sends an expected number of packets $w(\omega)^{-1} = 10^3 2^{\omega}$. UDP traffic is bursty; i.e., all the UDP transmissions are synchronised and they have a periodicity of 15 time slots, with a peak traffic of $5E3$, $13E3$, and $5E3$ in consecutive time slots. Under this scenario, for all the values of $\omega > 0$, the SAP-LAW is instable so we compare the performances of Smart-RED with ECN/RED which are shown in Figure 2. We observe that Smart-RED, similarly to ECN/RED, is stable although the queue lengths for both the protocols tend to be

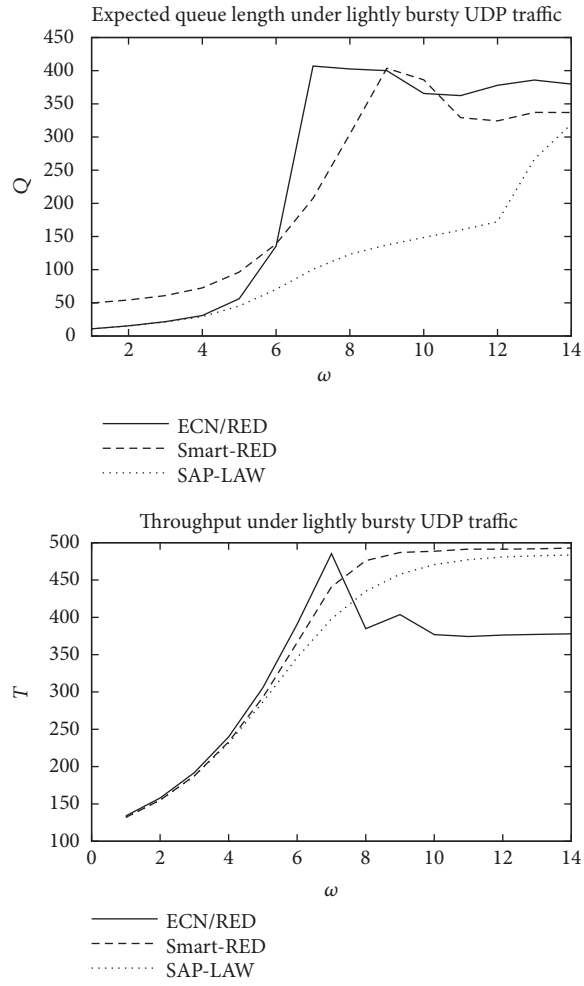

Figure 3: Comparison of ECN/RED and Smart-RED under lightly bursty UDP traffic.

high. Smart-RED has also a better throughput and a better average queue length under greedy TCP connections.

6.2. Comparison of the Protocols under Slightly Bursty UDP Traffic. In this experiment we have the same setting of Section 6.1 but the UDP is less bursty; i.e., the peak traffic is $1E3$, $3E3$, and $1E3$ in consecutive time slots. The results of the simulations are shown in Figure 3. Notice that Smart-RED improves the throughput of the SAP-LAW for both greedy and nongreedy TCP connections. In case of nongreedy TCP connections it mitigates the issues of SAP-LAW while maintaining a better performance than the ECN/RED in case of greedy TCP. On the other hand, the improvement of the throughput with respect to the SAP-LAW is paid with a greater expected queue length.

6.3. Comparison of the Protocols under Smooth UDP Traffic. Smooth UDP traffic is the ideal scenario for SAP-LAW. In this case we consider the same definition of a UDP agent given in the previous section, but the agents are not synchronised. In mean field regime, this leads to a smooth UDP traffic. The simulation results are shown in Figure 4. In this case, Smart-RED shows the highest throughput. With respect to ECN/RED it also maintains a lower queue length, while it pays the gain in terms of throughput with respect to the SAP-LAW with a longer expected queue.

7. Conclusion

In this paper we have proposed a novel algorithm, named Smart-RED, employed to avoid congestion at a shared AP. To this aim, we combined the features of the well-known ECN/RED with those of the more recent SAP-LAW. The latter shows the best performances when UDP traffic varies slowly in time and the TCP connections need the maximum available bandwidth. We showed that when these conditions are violated, SAP-LAW may be unable to prevent congestion occurring at the AP. Similarly to ECN/RED, Smart-RED is unconditionally stable provided that the UDP traffic does not flood the AP.

We showed that in case of heavy burstiness of the UDP traffic, Smart-RED reacts better than ECN/RED (while SAP-LAW is not applicable). On the other hand, in case of greedy TCP and smooth UDP traffic, SAP-LAW remains the best

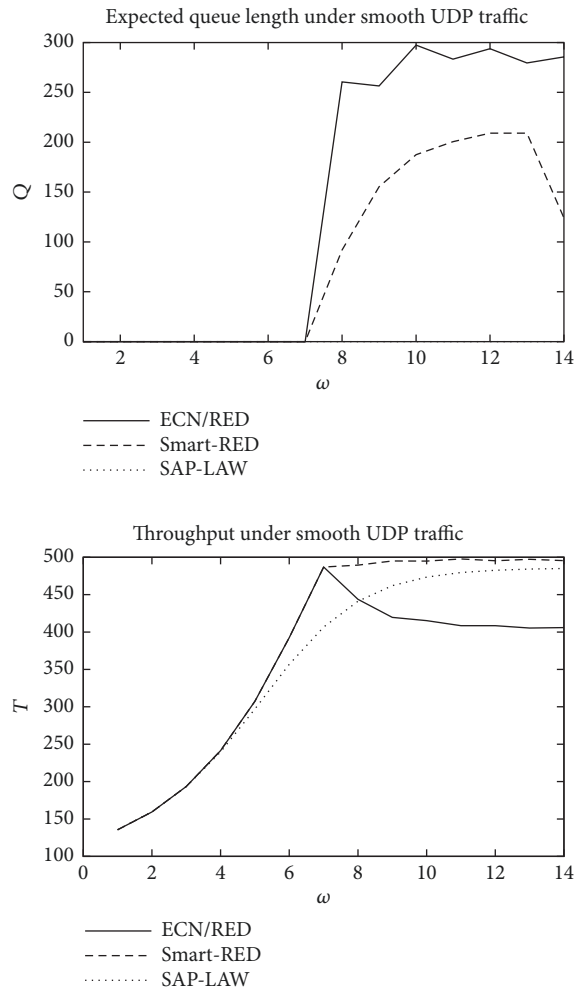

Figure 4: Comparison of ECN/RED under smooth UDP traffic.

choice even if Smart-RED performs better than ECN/RED. Notice that Smart-RED works exactly as SAP-LAW if the lowest threshold is set to 0 and the higher is set to ∞. Therefore, a self-adjusting policy of the thresholds based on the traffic measurements could be studied.

In conclusion, Smart-RED exploits the idea of SAP-LAW but eliminates the problem of the possible instability and reduces the throughput sufferance in case of TCP connections with different transmission needs. On the other hand, Smart-RED pays these achievements with longer expected waiting times in the ideal work scenarios of ECN/RED or SAP-LAW.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. S. Tanenbaum, *Computer Networks*, Prentice-Hall, 2003.

[2] G. Marfia and M. Roccetti, "TCP at last: reconsidering TCP's role for wireless entertainment centers at home," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 4, pp. 2233–2240, 2010.

[3] A. Abdelsalam, M. Luglio, C. Roseti, and F. Zampognaro, "TCP Wave: A new reliable transport approach for future internet," *Computer Networks*, vol. 112, pp. 122–143, 2017.

[4] K. Hassine, M. Frikha, and T. Chahed, "Access point backhaul resource aggregation as a many-to-one matching game in wireless local area networks," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 3523868, 11 pages, 2017.

[5] H. Kim, W. Lee, H. Kim, H. Kim, and J. M. Yang, "Protecting download traffic from upload traffic over asymmetric wireless links," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 1283420, 15 pages, 2018.

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

[7] S. Floyd, "TCP and explicit congestion notification," *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 5, pp. 8–23, 1994.

[8] T. Bonald, M. May, and J.-C. Bolot, "Analytic evaluation of RED performance," in *Proceedings of INFOCOM*, vol. 3, pp. 1415–1424, 2000.

[9] P. Tinnakornsrisuphap and A. Makowski, "Limit behavior of ECN/RED gateways under a large number of TCP flows," in *Proceedings of the INFOCOM*, vol. 2, pp. 873–883, 2003.

[10] J. L. Boudec, D. McDonald, and J. Mundinger, "A generic mean field convergence result for systems of interacting objects," in *Proceedings of the IEEE International Conference on the Quantitative Evaluation of Systems*, pp. 3–18, 2007.

[11] M. H. Malik, A. Majeed, M. E. Aydin, and M. H. Malik, "A parametric study for congestion control in queuing networks," in *Proceedings of the International Conference on Future Networks and Distributed Systems, ICFNDS '17*, ACM, New York, NY, USA, 2017.

[12] C. E. Palazzi, S. Ferretti, M. Roccetti, G. Pau, and M. Gerla, "What's in that magic box? The home entertainment center's special protocol potion, revealed," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 4, pp. 1280–1288, 2006.

[13] H. Jiang and C. Dovrolis, "Why is the internet traffic bursty in short time scales?" *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, pp. 241–252, 2005.

[14] J. Gettys and K. Nichols, "Bufferbloat: dark buffers in the internet," *Communications of the ACM*, vol. 55, no. 1, pp. 57–65, 2012.

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

[16] N. Tadayon, S. Zokaei, and E. Askari, "A novel prioritization scheme to improve QoS in IEEE 802.11e networks," *Journal of Computer Systems, Networks, and Communications*, vol. 2010, Article ID 856724, 12 pages, 2010.

[17] N. Chendeb Taher, Y. Ghamri-Doudane, B. El Hassan, and N. Agoulmine, "An accurate analytical model for 802.11e EDCA under different traffic conditions with contention-free bursting," *Journal of Computer Networks and Communications*, vol. 2011, Article ID 136585, 24 pages, 2011.

[18] A. Bujari, M. Massaro, and C. E. Palazzi, "Vegas over access point: making room for thin client game systems in a wireless home," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 12, pp. 2000–2002, 2015.

[19] E. Altman, K. Avrachenkov, and C. Barakat, "TCP in presence of bursty losses," in *Proceedings of the SIGMETRICS*, pp. 124–133, 2000.

[20] W. Kang, F. Kelly, N. Lee, and R. Williams, "Fluid and brownian approximations for an internet congestion control model," in *Proceedings of the 2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, vol. 4, pp. 3938–3943, 2004.

[21] N. Khademi, G. Armitage, M. Welzl, S. Zander, G. Fairhurst, and D. Ros, "Alternative backoff: achieving low latency and high throughput with ECN and AQM," in *Proceedings of the 2017 IFIP Networking Conference and Workshops*, pp. 1–9, 2017.

[22] S. R. Pokhrel and C. Williamson, "Modeling compound TCP over WiFi for IoT," *IEEE/ACM Transactions on Networking*, vol. 26, no. 2, pp. 864–878, 2018.

[23] G. Raina, S. Manjunath, S. Prasad, and K. Giridhar, "Stability and performance analysis of compound TCP with REM and drop-tail queue management," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 1961–1974, 2016.

[24] W. Bao, V. W. Wong, and V. C. Leung, "A model for steady state throughput of TCP cubic," in *Proceedings of the IEEE Global Communications ConferencE (IEEE GLOBECOM '10)*, pp. 1–6, 2010.

[25] F. Baccelli, D. R. McDonald, and J. Reynier, "A mean-field model for multiple TCP connections through a buffer implementing RED," *Performance Evaluation*, vol. 49, no. 1, pp. 77–97, 2002.

[26] D. R. McDonald and J. Reynier, "Mean field convergence of a model of multiple TCP connections through a buffer implementing RED," *The Annals of Applied Probability*, vol. 16, no. 1, pp. 244–294, 2006.

[27] A. Bobbio, M. Gribaudo, and M. Telek, "Analysis of large scale interacting systems by mean field method," in *Proceedings of the 5th International Conference on the Quantitative Evaluation of Systems, QEST '08*, pp. 215–224, 2008.

[28] F. Baccelli, M. Lelarge, and D. McDonald, "Mestable regimes for multiplexed TCP flows," in *Proceedings of the Annual Allerton Conference on Communication Control, and Computing*, Allerton House, University of Illinois at Urbana-Champaign, Monticello, Ill, USA, 2004.

[29] A. Bujari, A. Marin, C. E. Palazzi, and S. Rossi, "Analysis of ECN/RED and SAP-LAW with simultaneous TCP and UDP traffic," *Computer Networks*, vol. 108, pp. 160–170, 2016.

[30] R. Srikant, *The Mathematics of Internet Congestion Control*, Springer, 2004.