

RESEARCH

Open Access



Formal modelling of TCP congestion control mechanisms ECN/RED and SAP-LAW in the presence of UDP traffic

Mazhar H. Malik^{1*}, Mohsin Jamil², Muhammad N. Khan³ and Mubasher H. Malik¹

Abstract

Congestion avoidance mechanisms are used to prevent the saturation of the gateway which represents a bottleneck of the system. Recently, a new method for the congestion avoidance has been proposed: the smart access point with limited advertised window (SAP-LAW). The main idea is to hijack at the gateway the acknowledge packets in the transmission control protocol (TCP) connections in order to artificially reduce the advertised destination window according to some bandwidth allocation policy. Therefore, the flux control mechanism is artificially exploited to control the congestion at the bottleneck. The advantage of this approach is that it does not drop any packet and does not require any modification in the TCP implementations at the clients. While the most popular congestion avoidance mechanisms are the explicit congestion notification (ECN) and the random early detection (RED). In this paper, we propose stochastic models for the ECN/RED and SAP-LAW mechanisms in order to compare their performances under different scenarios. The models are studied in mean field regime, i.e. under a great number of TCP connections and UDP-based transmissions. Different from previous work for the ECN/RED, in this paper, we consider the presence of UDP traffic with bursts and the case of not greedy TCP connections. The models for SAP-LAW are totally new. The comparison is performed in terms of different performance indices including average queue length, system throughput, and expected waiting time.

Keywords: Congestion control, TCP/UDP, RED/ECN, Mean field regime, Formal methods

1 Introduction

The wide use of Internet connectivity in modern communication networks has made the analysis of protocols designed to share one gateway among many devices more and more important. Different from the early usage of the Internet, now, the requirements in terms of bandwidth and responsiveness of many applications are very high and it is often the case that if these requirements are not met, then the application usability is not acceptable. A popular and remarkable example is given by online games whose requirements in terms of response time are very strict. When several devices share the same gateway, we have to address the problem of allocating the resources, i.e. the bandwidth. The mechanisms that are implemented to this end and that we consider in this paper are based

on the transport protocols transmission control protocol (TCP) and user datagram protocol (UDP). According to many authors [1, 2], TCP is used mainly for reliable data transfer (including web browsing) and UDP in multimedia traffic or for applications with high requirements on responsiveness such as online gaming (obviously, there are exceptions). The reason why real-time applications usually prefer UDP is that it is a connectionless protocol, and it does not manage the retransmission of lost packets, guaranteeing minimum delays in data arrivals. On the contrary, TCP is a connection-oriented transport service: it provides reliable delivery and high throughput. For this reason, it is particularly adapted for applications whose predominant activity is downloading or uploading of files, as their main requirement is the complete and correct arrival of data.

TCP and UDP do not have an intrinsic mechanism to regulate the bandwidth when a gateway is shared, but the TCP (in its various versions) admits congestion and

*Correspondence: mazhar.hussain@isp.edu.pk

¹ Department of Computer Science and IT, Institute of Southern Punjab, 9-Km Bosan Road, Multan, Pakistan

Full list of author information is available at the end of the article

flux controls, while the UDP does not implement any of them. Every TCP connection keeps track of two window sizes: its own sending window and the destination window size. Congestion control is obtained by allowing the sender window size to increase when packets are properly sent and to decrease when the resources are not enough to satisfy the bandwidth request. Flux control is obtained by choosing a sending rate which is given by the minimum between the sender window size and the destination window size.

Popular approaches that use the built-in congestion mechanism to regulate the TCP transmission rates across a gateway are the random early detection (RED) [3] and the explicit congestion notification (ECN) [4, 5], while the smart access point with limited advertised window (SAP-LAW) [6] is based on the flux control mechanism.

The SAP-LAW has been presented in the context of integrated home server with shared access point [6]. In fact, due to the rapid evolution in the world of mobile devices, as well as the development of more and more advanced home entertainment systems, ensuring good performances to the applications sharing data with the World Wide Web has become a hard challenge.

In this paper, we study what happens when TCP and UDP are sharing the same link whose congestion is regulated by one of the mechanisms mentioned above: the ECN/RED and the SAP-LAW. On one side, the UDP-based flows seem to be aggressive towards the TCP-based ones since UDP does not provide any flux or congestion control mechanism. However, it can be also demonstrated that it is the TCP's congestion control mechanism which causes serious problems to the critical deterioration of UDP-based applications [6]. This is due to the fact that a queueing waiting time at the gateway is not a problem for TCP connections, whereas it is for UDP whose requirements for a fast response is usually of pivotal importance (e.g. for online gaming). Even if larger and larger bandwidths are offered today also to home consumers, the coexistence of TCP and UDP continues to present problems: due to its congestion control functionality, TCP continuously probes higher transfer rates, queuing packets on the buffer associated with the bottleneck of the connection, and consequently, the packets can be delayed in queue, causing heavy delays for real-time applications. In the literature, we can find different solutions to overcome the problem, and, in particular, different variations of TCP has been proposed, since the UDP is simpler and there are no many variable aspects [7, 8]. A large number of papers have studied the performance of TCP and ECN, see [9–12] as a not exhaustive list, but the main focus is on the behaviour of a single TCP connection under various scenarios. More similar to the analysis carried out here is that proposed in [13] where the authors consider the overall system performance in the mean field limit

for ECN/RED. The theoretical results proposed here are then extended in [14]. 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. Mean field theory (see, e.g. [14–16]) is an important approach to study large stochastic models with a deterministic approximation thus overcoming the well-known problem of the state space explosion.

The contributions of this paper are the extension of the mean field model presented in [14] in order to encompass the presence of UDP traffic and the introduction of a totally new model to analyse the SAP-LAW congestion mechanism. Another novelty is the relaxation of the common assumption [6, 13, 14] that TCP connections are greedy, i.e. they always have packets to send so that the sender window is always completely used at each time slot. We will study models that allows TCP connections to send a geometrically distributed random number of packets and then finish their life cycle. Finally, we use these models to compare the performances of ECN/RED and SAP-LAW with presence of UDP traffic. We consider both the cases of bursty and smooth UDP traffic.

The paper is structured as follows. Section 2 explains the ECN/RED and SAP-LAW congestion mechanism, and Section 3 gives some background about mean field theory. In Section 4, we introduce our models and give the iterative schemes for the computation of the performance indices. Section 5 briefly discusses the validation of the models, and in Section 6, we compare the performance indices of the two congestion mechanisms under different scenarios. Finally, Section 7 concludes the paper and gives some final remarks.

2 ECN/RED and SAP-LAW congestion control mechanisms

In this section, we review the congestion control mechanism that we study in this paper: the ECN/RED and the SAP-LAW.

2.1 ECN/RED

ECN [5] is an extension of TCP, which avoids the packet drops due to the channel congestion. This scheme uses random early detection (RED) gateways and marks the packets instead of dropping them, when the queue size is inside a given interval. In particular, it fixes a minimum and a maximum queue threshold: when the number of packets in the queue is less than the minimum, the packets are not marked, when it is between the minimum and the maximum threshold, it marks the packets with a probability that depends on the queue length. Finally, if the number of the packets is larger than the maximum, all arriving packets are marked or dropped. In this paper, we approximate this behaviour by using an exponential law to decide the packet marking probability [17]. The weak

point of this strategy is that it does not take in consideration the different requirements of real-time applications, for whom packets arrival delays may have catastrophic effects on the usability (e.g. for video streaming or online gaming).

2.2 SAP-LAW

SAP-LAW is a solution proposed in [6] with the specific aim of reducing the problems caused by the simultaneous presence of both TCP-based and UDP-based applications. The basic idea is to find the optimal trade-off between throughput and time delays by working on a dynamic modification of the TCP flow sending rate, in order to avoid the utilization of the buffer, but at the same time, to guarantee a full utilization of the bandwidth. The key factor is to determine the upper bound for the TCP flow sending rate as a function of the amount of UDP traffic. In this way, we are always sure to reserve enough bandwidth to the real-time applications activity, without limiting too much of the TCP flows. The general formula proposed by Palazzi et al. is

$$\max \text{TCP}_{\text{traffic}}(t) = \frac{(C - \text{UDP}_{\text{traffic}}(t))}{\# \text{TCP}_{\text{flows}}(t)}, \quad (1)$$

where $\text{UDP}_{\text{traffic}}(t)$ is the amount of bandwidth occupied by the real-time applications at time t , C is the capacity of the bottleneck link, and $\# \text{TCP}_{\text{flows}}(t)$ is the number of TCP connections at time t .

3 Theoretical background

In order to analyse the performances of the different solutions examined in this paper, we use a model for interacting objects, with individual states and global memory, introduced by Le Boudec et al. in [14].

In order to present a self-contained paper, we summarise the results proposed in [14]. We consider a heterogeneous population of N objects whose states are in the form (c, i) , where $c \in \Gamma$ is the class of the object and $i \in \{1, \dots, S_c\}$ is its internal state. At each time epoch, the objects interact with the environment and they can undergo a state transition (in this paper, we do not consider the possibility that an object changes its class). $X_n^N(t)$ indicates the state of the n th object at time slot t , while $K_{(c,i),(c,j)}^N(t)$ describes the probability that an object of class c goes from state i to state j during time slot t . The object state transitions depend on the current state of the objects and on the memory $\vec{R}^N(t) \in \mathbb{R}^P$ where P is the dimension of the vector. $\vec{M}^N(t)$ is the *occupancy measure*, i.e. the row vector where each component $M_{c,i}^N(t)$ is the proportion of objects that are in state (c, i) at time t , i.e.

$$M_{c,i}^N(t) = \frac{1}{N} \sum_{n=1}^N 1_{\{X_n^N(t)=(c,i)\}},$$

where $1_{x=(c,i)} = 1$ when $x = (c, i)$ and 0 otherwise. If we want to consider the single classes of objects, the proportion of class c objects in state i at time t is $\frac{1}{p_c} M_{c,i}^N(t)$,

where $p_c = \sum_{i=1}^{S_c} M_{c,i}(t)$ at some time slot t . Function g is a deterministic continuous function $g : \mathbb{R}^d \times \mathbb{P}(\varepsilon) \mapsto \mathbb{R}$ such that, for $t \geq 0$:

$$\vec{R}^N(t+1) = g(\vec{R}^N(t), \vec{M}^N(t+1)). \quad (2)$$

The role of function g is to govern the evolution of the model's memory. Hence, $\forall n$ we have

$$\begin{aligned} \mathbb{P}\{X_n^N(t+1) = (c, j) \mid \vec{R}^N(t) = \vec{r}, X_n^N(t) = (c, i)\} \\ = K_{c,i,c,j}^N(\vec{r}). \end{aligned} \quad (3)$$

For each class c , the $S_c \times S_c$ matrix $K_c^N(\vec{r})$ is the transition matrix for an individual object of class c and may depend on the population N and on the value \vec{r} of the global memory. $K_{c,i,c,j}^N$ is the transition probability from state i to state j ; therefore, we assume $K_{c,i,c,j}^N(\vec{r}) \geq 0$ and $\sum_{j=1}^{S_c} K_{c,i,c,j}^N(\vec{r}) = 1$ for all $1 \leq i \leq S_c$. We assume that

Assumption 1 $\forall c \in \Gamma, \forall i, j \in \varepsilon_c$, for $N \rightarrow \infty$, $K_{c,i,c,j}^N(\vec{r})$ converges uniformly in \vec{r} to some $K_{c,i,c,j}(\vec{r})$, which is a continuous function of \vec{r} .

A sufficient condition for this assumption to be satisfied is that transition matrix $K^N(\vec{r})$ does not depend on the number of objects N and that is continuous on \vec{r} , which is always the case in our examples.

Theorem 1 has been introduced and proved in [14]. Roughly speaking, it states that as $N \rightarrow \infty$, $\vec{M}^N(t)$ and $\vec{R}^N(t)$ can be replaced by deterministic approximations if the initial state is known. The proof of the correctness of the models that we propose in this paper widely depends on this theorem.

Theorem 1 Assume that the initial occupancy measure $\vec{M}^N(0)$ and memory $\vec{R}^N(0)$ converge almost surely to deterministic limits $\vec{\mu}(0)$, $\vec{\rho}(0)$ and for $t \geq 0$

$$\vec{\mu}(t+1) = \vec{\mu}(t)K(\vec{\rho}(t)) \quad (4)$$

$$\vec{\rho}(t+1) = g(\vec{\rho}(t), \vec{\mu}(t+1)). \quad (5)$$

Then, for any fixed time t , almost surely

$$\lim_{N \rightarrow \infty} \vec{M}^N(t) = \vec{\mu}(t) \text{ and } \lim_{N \rightarrow \infty} \vec{R}^N(t) = \vec{\rho}(t).$$

4 Models for ECN/RED and SAP/LAW congestion mechanisms

In this section, we propose different models for the ECN/RED and SAP/LAW congestion mechanisms. Different from other approaches previously addressed by the literature, we consider the following:

- The presence of UDP traffic as well as TCP. Specifically, we will be able to model both the case in which UDP traffic is smooth and the case in which it is bursty.
- The case in which the TCP connections are not greedy, i.e. in case of an infinite capacity gateway, their windows would not tend to be at their maximum size. In practice, this is a frequent scenario which approximates the situation in which TCP connections are closed and new ones are newly opened maintaining the total number of active connections approximatively constant.

The study is done under the mean field regime, where large numbers of users are interacting with each other. Figure 1 gives a conceptual overview of the topology where S_1, S_2, S_3, \dots are the stations which are connected with the access point. We assume that there are large number of access points which are handling stations requests and AP is managing congestion issues via the proposed techniques which are modelled.

4.1 Modelling ECN/RED with UDP traffic and greedy TCP

In this section, we propose a mean field model for TCP and UDP connections sharing the same gateway. We

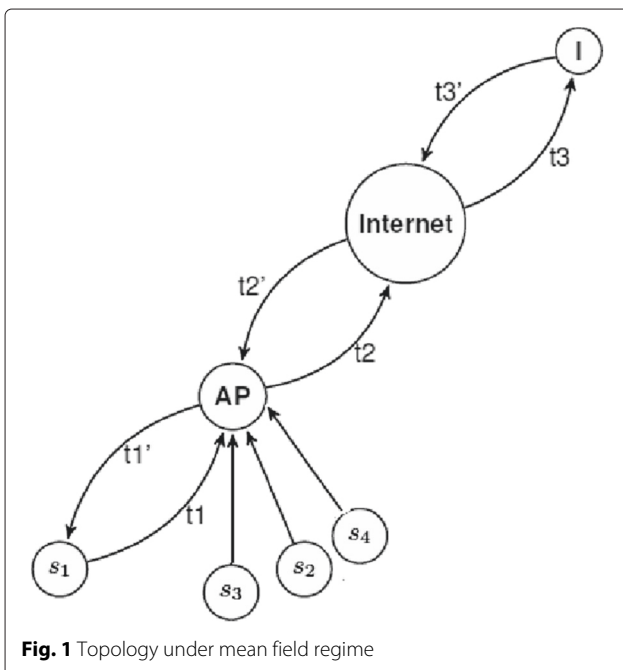


Fig. 1 Topology under mean field regime

assume that TCP connections are greedy, i.e. the process associated with the connection fills the entire window with data to be sent at each time slot. We assume the time slot to be a round trip time as in [13, 14].

According to the methodology described in Section 3, we annotate each object with a class $c \in \{p, u\}$ where p denotes the TCP connections and u the UDP agents. Analogous to [14], each TCP connection has a state $i \in \mathcal{S}_p$, where $\mathcal{S}_p = \{1, \dots, I_p\}$. At state i , the sending rate is $s_p(i) \in \mathbb{N}$ and, without loss of generality, we assume $i' > i'' \implies s_p(i') > s_p(i'')$. Although UDP transmissions are stateless, we model them by using a probabilistic automaton whose transition probabilities are independent of the rest of the model (in particular of the queue length). This choice allows us to use the automata to model different behaviours of the transmission of a UDP agent, e.g. with smooth or bursty traffic. In practice, the transition probabilities may be decided with a model-fitting approach such as those proposed in [16]. The set of states associated with a process using a UDP transmission is denoted by $\mathcal{S}_u = \{1, \dots, I_u\}$, and the sending rate at state $i \in \mathcal{S}_u$ is $s_u(i)$. Notice that the flexibility of this modelling choice is very high: it allows us to represent synchronised or partially synchronised UDP transmissions (such as those occurring during online gaming), the autocorrelation and the periodicity in the transmission bursts, as well as a simple smooth transmission of packets.

Thus, the state of an object is denoted by a pair (c, i) where $c \in \{p, u\}$ and $i \in \{1, \dots, I_p\}$ if $c = p$ and $i \in \{1, \dots, I_u\}$ otherwise. To avoid trivialities, we assume $I_p > 1$ and $I_u \geq 1$.

Let $N \in \mathbb{N}$ be the number of objects and $C \in \mathbb{R}$ the bottleneck capacity for every connection.

Since we do not admit class switching during the system activity (TCP connections cannot turn into UDP connections, and vice versa), we define N_p as the number of TCP connections and N_u the number of UDP transmissions, i.e. for all $t \geq 0$

$$N_p = \sum_{n=1}^N 1_{X_n^N(t)=(p,i)} \text{ for some } i \in \mathcal{S}_p \quad (6)$$

$$N_u = \sum_{n=1}^N 1_{X_n^N(t)=(u,i)} \text{ for some } i \in \mathcal{S}_u, \quad (7)$$

where $\bar{X}^N(t)$ is the stochastic process underlying the model and $X_n^N(t)$ is the state of object n at time t . Clearly, $N = N_p + N_u$. Let p_p (p_u respectively) be the proportion of TCP (respectively UDP) connections, then we have

$$p_p = \frac{N_p}{N} = \sum_{i=1}^{I_p} M_{p,i}^N(t) \quad \forall t \geq 0 \quad (8)$$

$$p_u = \frac{N_u}{N} = \sum_{i=1}^{I_u} M_{u,i}^N(t) \quad \forall t \geq 0, \quad (9)$$

where $M_{c,i}$ is the occupancy vector defined in Section 3. The proportion of TCP connections in state i at time t is then $p_p^{-1}M_{p,i}^N(t)$ and, for UDP, we have $p_u^{-1}M_{u,i}^N(t)$. The average normalised sending rate of TCP connections at time t is hence $S_p^N(t) = \sum_{i=1}^{I_p} s_p(i)M_{p,i}^N(t)$, while for UDP, connection it is $S_u^N(t) = \sum_{i=1}^{I_u} s_u(i)M_{u,i}^N(t)$.

We now define the memory of the system, $\vec{R}^N(t) = (R_c^N(t), R_p^N(t))$, that consists of a pair of real numbers that represents the normalised population of packets enqueued at the bottleneck at the end of time slot t and $t - 1$, respectively. The update equation is

$$\begin{aligned} R_c^N(t+1) &= \max\left(R_c^N(t) + S_p^N(t+1) \right. \\ &\quad \left. + S_u^N(t+1) - C, 0\right) \\ R_p^N(t+1) &= R_c^N(t) \end{aligned} \quad (10)$$

We approximate the behaviour of the ECN/RED mechanisms as proposed in [13]: as soon as a packet arrives at the bottleneck, it is marked for being discarded or sent. The marking of each packet is independent of the others arriving during the same time slot and cannot be changed. The marking of packets at time slot t depends only on the queue length seen by the packets immediately before their arrival, i.e. on the memory at time t via function $q(\vec{R}^N(t))$. According to [4], we define function q as follows:

$$q(R_p^N(t)) = 1 - \exp(-\gamma R_p^N(t)) \quad (11)$$

i.e. the probability of discarding a packet grows exponentially with the total queue length. Observe that the marking policy is coherent with that proposed in [16] where the packets are marked according to the queue length that they find at their arrival epoch, while in [14], the queue length after their departure is used. Parameter γ is a positive real number whose value will be specified in the following sections. Finally, the transition matrix for each object is

$$\begin{aligned} K_{(p,i),(p,i+1)}(\vec{r}) &= (1 - q(r_p))^{s_p(i)} \mathbf{1}_{i < I_p} \\ K_{(p,I_p),(p,I_p)}(\vec{r}) &= (1 - q(r_p))^{s_p(I_p)} \\ K_{(p,i),(p,d(i))}(\vec{r}) &= 1 - (1 - q(r_p))^{s_p(i)} \\ K_{(u,i),(u,j)}(\vec{r}) &= \kappa_{(u,i),(u,j)}, \quad 1 \leq i, j \leq I_u, \end{aligned} \quad (12)$$

where $\vec{r} = (r_c, r_p)$ is a value of $\vec{R}^N(t)$, $\kappa_{(u,i),(u,j)} \in [0, 1]$ and for all $i \in [1, I_u]$, we have

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

Function $d(i)$ is used to model the destination state in case of a marked packet. We may have $d(i) = 1$ for all i or as in [15] $d(i) = \lfloor i/2 \rfloor$ (and the number of packets sent $s_p(i)$ is proportional to i). Generally speaking, we require that $d(i) = i$ only for $i = 1$, while for $i > 1$, we have that $d(i) < i$.

Remark 1 We observe that the proposed model for ECN/RED is novel with respect to the one proposed in [15] since it takes into account different classes of traffic, not only TCP. In [14], the authors propose a solution for different types of TCP connection. However, in their model, the ECN/RED mechanism is modelled in a simpler and less accurate way than what is proposed here: in fact, in [14], the probability of a TCP connection to reduce its window size depends only on the queue length, i.e. is the same for connections with a large and already small windows. We consider our approach more accurate since it takes into account that a TCP connection with a large window sends more packets than one with a small window and hence has higher probability to be selected for being slowed down.

Proposition 1 If $\vec{M}^N(0)$ and $\vec{R}^N(0)$ converge almost surely to $\vec{\mu}(0)$ and $\vec{\rho}(0)$, respectively, as $N \rightarrow \infty$, then for any finite horizon t , we have that

$$\lim_{N \rightarrow \infty} \vec{M}^N(t) = \vec{\mu}(t) \text{ and } \lim_{N \rightarrow \infty} \vec{R}^N(t) = \vec{\rho}(t)$$

almost surely, where $\vec{\mu}(t)$ and $\vec{\rho}(t)$ are defined by the iterative scheme of Table 1.

Proof The proof follows from the fact that Assumption 1 is satisfied and from Theorem 1. \square

Table 1 Iteration system for ECN/RED model with greedy TCP connections

$$\begin{aligned} \mu_{p,i}(t+1) &= (1 - q(\rho_p(t)))^{s_p(i-1)} \mu_{p,i-1}(t) \mathbf{1}_{i > 1} \\ &\quad + \sum_{j:d(j)=i} (1 - (1 - q(\rho_p(t)))^{s_p(j)}) \mu_{p,j}(t) \quad 1 < i < I_p \\ \mu_{p,I_p}(t+1) &= (1 - q(\rho_p(t)))^{s_p(I_p-1)} \mu_{p,I_p-1}(t) + (1 - q(\rho_p(t)))^{s_p(I_p)} \mu_{p,I_p}(t) \\ \mu_{u,i}(t+1) &= \sum_{j=1}^{I_u} \kappa_{(u,j),(u,i)} \mu_{u,j}(t) \\ \sigma_p(t+1) &= \sum_{i=1}^{I_p} \mu_{p,i}(t+1) s_p(i) \quad \sigma_u(t+1) = \sum_{i=1}^{I_u} \mu_{u,i}(t+1) s_u(i) \\ \rho_c(t+1) &= \max(\rho_p(t) + \sigma_p(t+1) + \sigma_u(t+1) - C, 0) \quad \rho_p(t+1) = \rho_c(t) \end{aligned}$$

4.2 Modelling SAP-LAW with UDP traffic and greedy TCP

The SAP-LAW is a solution to the problem of congestion at a bottleneck proposed in [6] with the specific aim of reducing the problems caused by the simultaneous presence of both TCP-based and UDP-based applications. The key aspect of this scheme is a manipulation of the TCP acknowledge packets forwarded by the gateway. The idea is to artificially reduce the value of the receiving window size of the destination in the acknowledge packets. Basically, if the ECN/RED uses the TCP congestion detection and resolution algorithm to handle the bottleneck, the SAP-LAW uses the flux control algorithm that imposes the sender to use a window which is the minimum between its sending window and the destination's receiving window (see, e.g. [2]). According to [6], the gateway counts the arrived UDP packets in a time interval and hence estimates the total instantaneous UDP arrival rate at time t , $\xi(t)$. Let NC be the total capacity of the gateway and $N_p(t)$ the number of TCP connections at time t , then the maximum window size which is sent back to the TCP transmitters is the same for all the TCP connections and equal to

$$\max\text{TCP}(t) = \frac{NC - \xi(t)}{N_p(t)}. \tag{13}$$

In our setting, we assume $N_p(t) = N_p$, $N = N_p + N_u$, and the time interval that we use to estimate the instantaneous arrival rate for the UDP packets to be equal to a multiple Y s of the round trip time, i.e. $\xi(t) = y(S_u^N(t - Y), \dots, S_u^N(t - 1))$. The notation, unless differently specified, is that of Section 4.1. The packets sent by each TCP connection at each time slot depends on its state and on Eq. 13 and the granularity of the model. For the sake of simplicity, we assume that $s_p(i) = \alpha i$ for some $\alpha \in \mathbb{N}$, i.e. the maximum number of packets sent in a time slot by a TCP connection grows linearly with the state numbering. The memory $\vec{R}^N(t) = (R_c^N(t), R_p^N(t), \vec{R}_u^N(t))$ is a pair of real numbers (r_c, r_p) followed by a vector of reals \vec{r}_u where r_c denotes the normalised queue length at the bottleneck (counting both TCP and UDP packets) at a given time slot, r_p is the normalised queue length at the previous time slot, and $\vec{r}_u = (r_{u0}, \dots, r_{uY})$ denotes the normalised counting of the arrived UDP packets at the latest $(Y + 1) > 1$ time slots.

The transitions for UDP objects are the same as those shown in Section 4.1. Each TCP connection has a state represented by a pair of numbers $(i, j) \in \mathbb{N}^2$, where i denotes the state corresponding to the sender window size $s_p(i)$, while j takes into account the number of received acknowledges. Indeed, the SAP-LAW mechanism does not allow the sender to use its full window size and consequently the growth of the sender window size is in generally slower than what would be observed with a

gateway with infinite capacity. Formally, the state of each object in our model is (ℓ, \vec{x}) where $\ell \in \{p, u\}$ (where p stands for TCP and u for UDP) and

$$\vec{x} = \begin{cases} (i, j) : 1 \leq i \leq I_p \wedge 0 \leq j < i & \text{if } \ell = p \\ i : 1 \leq i \leq I_u & \text{if } \ell = u \end{cases}.$$

The dynamic of the memory is specified by the following equation:

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

Function $S_u^N(t)$ is specified as shown in Section 4.1, while S_p^N is defined as

$$S_p^N(t) = \sum_{i=1}^{I_p} \sum_{j=0}^{i-1} M_{p,ij}(t) \cdot \min \left(s_p(i), s_p(h(\vec{R}_u^N(t))) \right),$$

where

$$h(\vec{r}_u) = j \left(s_p(j), s_p(j) \leq (C - \tilde{y}(\vec{r}_u)) \frac{1}{p_p} \right) \quad \forall j = 1, \tag{14}$$

and $\tilde{y}(\vec{r}_u) = y(r_{u1}, \dots, r_{uY})$. Given a state (i, j) of a TCP connection, and the state z corresponding to the maximum sending window obtained by Eq. 14, the following state of the connection is $f(i, j, z)$ defined as follows:

$$f(i, j, z) = \begin{cases} (i, j + \min(z, i)) & \text{if } j + \min(z, i) < i \\ (i + 1, j + \min(z, i) - i) & \text{if } j + \min(z, i) \geq i \\ & \wedge i < I_p \\ (I_p, I_p - 1) & \text{otherwise} \end{cases} \tag{15}$$

We can now give the mean field limit for $N \rightarrow \infty$.

Proposition 2 *If $\vec{M}^N(0)$ and $R^N(0)$ converge almost surely to $\vec{\mu}(0)$ and $\vec{\rho}(0)$, respectively, as $N \rightarrow \infty$ then for any finite horizon t we have that*

$$\lim_{N \rightarrow \infty} \vec{M}^N(t) = \vec{\mu}(t) \text{ and } \lim_{N \rightarrow \infty} \vec{R}^N(t) = \vec{\rho}(t)$$

almost surely, where $\vec{\mu}(t)$ and $\vec{\rho}(t)$ are defined by the iterative scheme of Table 2.

Proof The proof follows straightforwardly from Theorem 1 after verifying Assumption 1. \square

Table 2 Iteration system for SAP-LAW model with greedy TCP connections

$$\begin{aligned}
\mu_{p,ij}(t+1) &= \sum_{i'=1}^{I_p} \sum_{j'=0}^{i-1} \mu_{p,i'j'}(t) \mathbf{1}_{f(i',j',h(\bar{\rho}_u(t)))=(ij)} \\
\mu_{u,j}(t+1) &= \sum_{j=1}^{I_u} \kappa_{(u,j),(u,j)} \mu_{u,j}(t) \\
\sigma_p(t+1) &= \sum_{i=1}^{I_p} \sum_{j=1}^{i-1} \mu_{p,ij}(t+1) \min(s_p(i), s_p(h(\bar{\rho}_u(t)))) \\
&\quad \times \sigma_u(t+1) = \sum_{i=1}^{I_u} \mu_{u,j}(t+1) s_u(i) \\
\rho_c(t+1) &= \max(\rho_c(t) + \sigma_p(t+1) + \sigma_u(t+1) - C, 0) \\
\rho_p(t+1) &= \rho_c(t) \rho_{ui}(t+1) = \rho_{u(i-1)}(t), 1 \leq i \leq Y \quad \rho_{u0}(t+1) = \sigma_u(t+1)
\end{aligned}$$

4.3 Modelling ECN/RED with UDP and temporary TCP connections

When comparing the SAP-LAW and the ECN/RED congestion mechanisms, we should take into account that the former tends to work better when the TCP sending windows are very large. Intuitively, if all the TCP connections are willing to send an infinite queue of packets, then the SAP-LAW allows them to constantly increase their sending window (possibly slower than what happens in ECN/RED). In the long run, all the windows will be at their maximum size and the SAP-LAW decides which actual rate they can send according to the estimated UDP traffic. However, the SAP-LAW is fair with respect to the distribution of the bandwidth to each TCP connection as shown by Eq. 13; therefore, it may happen that part of the bandwidth is allocated to a TCP connection whose sending window is still small causing a waste of the resources.

In this and in the following section, we modify the previous models in order to take into account that a TCP connection has a random amount of packets to send. Once they are sent, we assume that the connection is closed and a new one restarts from the minimum window size.

We assume that once the connection is established, it has to send an amount of packets which is geometrically distributed with parameter $w \in (0, 1)$, i.e. the probability of sending exactly ξ packets is $w(1-w)^{\xi-1}$, with an expected value of $1/w$. We modify the model presented in Section 4.1 in order to take into account this aspect as follows (when not differently specified, the notation is inherited by the previous sections). Let $L_f(t)$ for $1 \leq f \leq N_p$ and $t \geq 0$ be a set of independent and identically distributed geometric random variables with parameter $w \in (0, 1)$ that represent the number of packets that TCP connection i has still to send at time t . The event of closure of TCP connection f occurs at time t_0 if the number of packets that it sends at t_0 is equal to $L_f(t_0)$. Notice that due to the memoryless property of the geometric random variable, the distribution of the number of packets sent by

a TCP connection from its opening to its closing epoch is still geometric.

The state (i, s) of each TCP object consists of an integer $i \in [1, I_p]$ that denotes the connection state and an integer $s \in [1, s_p(I_p) + 1]$ that denotes the number of packets that still have to be sent. So the transition probabilities are given by the following equations:

$$\begin{aligned}
\tau(s') &= w(1-w)^{s'-1} \mathbf{1}_{s' \leq s_p(I_p)} \\
&\quad + (1-w)^{s_p(I_p)} \mathbf{1}_{s'=s_p(I_p)+1}
\end{aligned} \tag{16}$$

$$\begin{aligned}
K_{(p,i,s),(p,i+1,s')}(\vec{r}) &= (1-q(r_p))^{s_p(i)} \\
&\quad \cdot \mathbf{1}_{i < I_p, s > s_p(i)} \tau(s') \\
K_{(p,i,s),(p,1,s')}(\vec{r}) &= \mathbf{1}_{s \leq s_p(i)} \tau(s')
\end{aligned} \tag{17}$$

$$K_{(p,I_p,s),(p,I_p,s')} = (1-q(r_p))^{s_p(I_p)} \mathbf{1}_{s > s_p(I_p)} \tau(s')$$

$$\begin{aligned}
K_{(p,i,s),(p,d(i),s')} &= (1 - (1-q(r_p))^{s_p(i)}) \\
&\quad \cdot \mathbf{1}_{s > s_p(i)} \tau(s').
\end{aligned}$$

Notice that Eq. 17 is the key point to decide when a TCP connection finishes its transmission and a new one is created. Function τ is the density function of a truncated geometric random variable (r.v.), where the probability mass of outcomes greater than $s_p(I_p)$ is concentrated in the last state, $s_p(I_p) + 1$.

Proposition 3 *The number of packets which are sent from two consecutive observations of the transition given by Eq. 17 is geometrically distributed with mean $1/w$.*

Proof The proof follows from the memoryless property of the geometric random variable and by the observation that the transition described by Eq. 17 occurs with probability 1 from each state (i, s) such that $s \leq s_p(i)$. \square

The memory of the system is still a pair of reals $\vec{R}^N(t) = (R_c(t), R_p(t))$ that denotes the normalised queue length at the gateway at epoch t and $t-1$, respectively. The update rule for the memory is

$$\begin{aligned}
R_c^N(t+1) &= \max\left(R_c^N(t) + S_p^N(t+1) \right. \\
&\quad \left. + S_u^N(t+1) - C, 0\right) \\
R_p^N(t+1) &= R_c^N(t),
\end{aligned}$$

where

$$S_p^N(t+1) = \sum_{i=1}^{I_p} \sum_{s=1}^{s_p(I_p)+1} M_{p,i,s}(t+1) \min(s_p(i), s).$$

Proposition 4 states the mean field approximation for this model. Notice that the state space of each TCP object

is rather large but we are interested only in the normalised population of a connection state, i.e., $\tilde{\mu}_{p,i}(t) = \sum_{s=1}^{s_p(I_p)+1} \mu_{p,j,s}(t)$.

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

$$\begin{aligned} \lim_{N \rightarrow \infty} \sum_{s=1}^{s_p(I_p)+1} M_{p,i,s}^N(t) &= \tilde{\mu}_{p,i}(t) \\ \lim_{N \rightarrow \infty} M_{u,i}^N(t) &= \mu_{u,i}(t) \\ \lim_{N \rightarrow \infty} \vec{R}^N(t) &= \vec{\rho}(t) \end{aligned}$$

almost surely, where $\tilde{\mu}_{p,i}(t)$, $\mu_{u,i}(t)$, and $\rho(t)$ are defined by the iterative scheme of Table 3.

Notice that Theorem 1 is applicable to obtain the mean field approximation for $\vec{\mu}$; however, the number of states of each component may grow very quickly and make the computation of the iterative model inefficient. In order to derive the interesting performance indices, we can use the iterative scheme of Table 3 which aggregates the TCP object states that share the same window size. We now prove its correctness.

Proof Let $X_i^p(t)$ be the r.v. describing the state of TCP object i at time slot t . The support of $X_i^p(t)$ is $[1, I_p] \times [1, S_p(I_p) + 1]$. Let $X_{i1}^p(t) = j$ if and only if $X_i^p(t) =$

Table 3 Iteration system for ECN/RED model with temporary TCP connections

$$\begin{aligned} \tilde{\mu}_{p,1}(t+1) &= \sum_{j:d(j)=1} \left(1 - (1 - q(\rho_p(t)))^{s_p(j)}\right) \tilde{\mu}_j(t)(1-w)^{s_p(j)} \\ &\quad + \sum_{j=1}^{l_p} \tilde{\mu}_{p,j}(t)(1 - (1-w)^{s_p(j)}) \quad (18) \\ \tilde{\mu}_{p,i}(t+1) &= \sum_{j:d(j)=i} \left(1 - (1 - q(\rho_p(t)))^{s_p(j)}\right) \tilde{\mu}_j(t)(1-w)^{s_p(j)} \\ &\quad + (1 - q(\rho_p(t)))^{s_p(i-1)} \tilde{\mu}_{p,i-1}(t)(1-w)^{s_p(i-1)} \quad 1 < i < l_p \\ \tilde{\mu}_{p,l_p}(t+1) &= (1 - q(\rho_p(t)))^{s_p(l_p-1)} \tilde{\mu}_{p,l_p-1}(t)(1-w)^{s_p(l_p-1)} \\ &\quad + (1 - q(\rho_p(t)))^{s_p(l_p)} \tilde{\mu}_{p,l_p}(t)(1-w)^{s_p(l_p)} \\ \mu_{u,i}(t+1) &= \sum_{j=1}^{l_u} \kappa_{(u,j),(u,i)} \mu_{u,j}(t) \\ \sigma_p(t+1) &= \sum_{i=1}^{l_p} \tilde{\mu}_{p,i}(t+1) \frac{1 - (1-w)^{s_p(i)}}{w} \quad \sigma_u(t+1) = \sum_{i=1}^{l_u} \mu_{u,i}(t+1) s_u(i) \\ \rho_c(t+1) &= \max(\rho_c(t) + \sigma_p(t+1) + \sigma_u(t+1) - C, 0) \\ \rho_p(t+1) &= \rho_c(t) \quad \rho_{up}(t+1) = \rho_{uc}(t), i > 0 \quad \rho_{uc}(t+1) = \sigma_u(t+1) \end{aligned}$$

(j, s) for $s \in [1, S_p(I_p) + 1]$ and $X_{i2}^p(t) = s$ if and only if $X_i^p(t) = (j, s)$ for some $j \in [1, I_p]$. By definition, we have

$$\mu_{p,i,s}(t) = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{j=1}^{N_p} \mathbf{1}_{X_j^p(t)=(i,s)}.$$

Notice that r.v. $X_j^p(t)$ are asymptotically independent (the proof is analogue to that of [13, Theorem 1]) and are identically distributed; therefore, for the law of large numbers, we have that for $N \rightarrow \infty$, $\mu_{p,i,s}$ almost surely converges to $p_p E[\mathbf{1}_{X_j^p(t)=(i,s)}] = p_p \mathbb{P}\{X_j^p(t) = (i, s)\}$ for some j . Noting that by model definition $X_{j1}^p(t)$ and $X_{j2}^p(t)$ are independent, we have

$$\begin{aligned} \mu_{p,i,s}(t) &= p_p \mathbb{P}\{X_{j1}^p(t) = i\} \mathbb{P}\{X_{j2}^p(t) = s\} \\ &= \tilde{\mu}_{p,i}(t) \mathbb{P}\{X_{j2}^p(t) = s\}, \text{ a.s.} \end{aligned} \quad (19)$$

We now prove the correctness of Eq. 18, the other ones' proofs follow the same line. By definition of $\tilde{\mu}_{p,i}(t)$ and by Theorem 1 we have

$$\begin{aligned} \tilde{\mu}_{p,1}(t) &= \sum_{s=1}^{s_p(I_p)+1} \mu_{p,1,s}(t) \\ &= \sum_{s=1}^{s_p(I_p)+1} \left(\sum_{j:d(j)=1} \sum_{s'=1}^{s_p(j)+1} \left(1 - (1 - q(\rho_p(t)))^{s_p(j)}\right) \right. \\ &\quad \left. \times \tau(s) \mu_{p,j,s'}(t) \mathbf{1}_{s' > s_p(j)} + \sum_{j=1}^{I_p} \sum_{s'=1}^{s_p(I_p)+1} \tau(s) \mu_{p,j,s'}(t) \mathbf{1}_{s' \leq s_p(j)} \right), \end{aligned}$$

that by Eq. 19 becomes

$$\begin{aligned} \sum_{s=1}^{s_p(I_p)+1} \tau(s) \sum_{j:d(j)=1} \left(1 - (1 - q(\rho_p(t)))^{s_p(j)}\right) \tilde{\mu}_{p,j}(t) \\ \times \sum_{s'=s_p(j)+1}^{s_p(I_p)+1} \tau(s') + \sum_{s=1}^{s_p(I_p)+1} \tau(s) \tilde{\mu}_{p,j}(t) \sum_{s'=1}^{s_p(j)} \tau(s'). \end{aligned}$$

Now, Eq. 18 follows after some algebraic simplifications. \square

4.4 Modelling SAP-LAW with UDP and temporary TCP connections

The last model we propose considers the SAP-LAW mechanism with temporary TCP connections. We start from the basic model of Section 4.2 and maintain the definition of the memory $\vec{R}^N(t)$ and the transition matrix $K_{(u,i),(u,j)}(\vec{r})$ for UDP objects (see the last equation of Eq. 12). The state of an object associated with a TCP connection consists of four components (p, \vec{x}, s) where p marks the TCP object, \vec{x} has the same meaning given in Section 4.2 and s denotes the number of packets that has

to be sent. Similar to what we proposed in Section 4.3, we assume that the number of packets sent by a TCP connection before being closed and restarted is geometrically distributed with mean $1/w$ and hence, by exploiting the memoryless property of geometric random variables (see Proposition 3), we obtain the following transition matrix for TCP objects

$$K_{(p,i,j,s),(p,i',j',s')}(\bar{r}) = \mathbf{1}_{f(i,j,h(\bar{r}_u))=(i',j')} \cdot \mathbf{1}_{s>s_p(h(\bar{r}_u))} \tau(s')$$

$$K_{(p,i,j,s),(p,1,1,s')}(\bar{r}) = \mathbf{1}_{s \leq s_p(h(\bar{r}_u))} \tau(s'),$$

where $\bar{r} = (r_p, r_c, r_{u0}, \bar{r}_u) = (r_p, r_c, r_{u0}, r_{u1}, \dots, r_{uY})$ is a value of the memory at a certain time slot, $\tau(s')$ is defined in Eq. 16, and $f(i, j, z)$ in Eq. 15. We can now state the mean field result that, similar to that proposed by Proposition (4) gives an iterative scheme whose computation is more efficient than that straightforwardly obtained by the application of Theorem 1 since it considers an aggregation of states for the TCP objects.

Proposition 5 *If $\bar{M}^N(0)$ and $R^N(0)$ converge almost surely to $\bar{\mu}(0)$ and $\bar{\rho}(0)$, respectively, and $\forall i = 1, \dots, I_p$, it holds that $M_{p,i,j,s}^N(0) / \sum_{s'=1}^{s_p(I_p)+1} M_{p,i,j,s'}^N$ converge almost surely to $\tau(s)$, as $N \rightarrow \infty$, then for any finite horizon t we have that*

$$\lim_{N \rightarrow \infty} \sum_{s=1}^{s_p(I_p)+1} M_{p,i,j,s}^N(t) = \tilde{\mu}_{p,i,j}(t)$$

$$\lim_{N \rightarrow \infty} M_{u,i}^N(t) = \mu_{u,i}(t)$$

$$\lim_{N \rightarrow \infty} \bar{R}^N(t) = \bar{\rho}(t)$$

almost surely, where $\tilde{\mu}_{p,i,j}(t)$, $\mu_{u,i}(t)$, and $\bar{\rho}(t)$ are defined by the iterative scheme of Table 4 and function h is defined by Eq. 14.

Table 4 Iteration system for ECN/RED model with temporary TCP connections

$$\tilde{\mu}_{p,1,1}(t+1) = \sum_{i'=1}^{I_p} \sum_{j'=1}^i \tilde{\mu}_{p,i',j'}(t) \left(1 - (1-w)^{\min(s_p(i'), s_p(h(\bar{\rho}_u(t))))} \right)$$

$$\tilde{\mu}_{p,i,j}(t+1) = \sum_{i'=1}^{I_p} \sum_{j'=1}^i \tilde{\mu}_{p,i',j'}(t) \mathbf{1}_{f(i',j',h(\bar{\rho}_u(t)))=(i,j)} (1-w)^{\min(s_p(i'), s_p(h(\bar{\rho}_u(t))))}$$

$$\sigma_p(t+1) = \sum_{i=1}^{I_p} \tilde{\mu}_{p,i,j}(t+1) \frac{1 - (1-w)^{\min(s_p(i), s_p(h(\bar{\rho}_u(t))))}}{w} \tag{20}$$

$$\times \sigma_u(t+1) = \sum_{i=1}^{I_u} \mu_{u,i}(t+1) s_u(i)$$

$$\rho_c(t+1) = \max(\rho_c(t) + \sigma_p(t+1) + \sigma_u(t+1) - C, 0)$$

$$\rho_p(t+1) = \rho_c(t) \quad \rho_{up}(t+1) = \rho_{uc}(t), i > 0 \quad \rho_{uc}(t+1) = \sigma_u(t+1)$$

Remark 2 *Notice that there are some equivalence relations among the models specified in this section. Specifically, the ECN/RED, with greedy or temporary TCP connections is equivalent to the corresponding SAP-LAW model when $C \geq \max\{s_p(i), 1 \leq i \leq I_p\} \cup \{s_u(j), 1 \leq j \leq I_u\}$. In fact, in this case, none of the two congestion mechanisms is applied and hence, the gateway is not a bottleneck for the system. Another interesting relation is that the model presented in Section 4.1 (Section 4.2) can be derived as the limit for $w \rightarrow 0^+$ of the model presented in Section 4.3 (Section 4.4). In fact, when w is very small, the amount of data that each TCP connection sends is so high that its behaviour resembles the one of a greedy connection. Nevertheless, we think that having presented and described the four models should have helped the clarity of the exposition.*

5 Validation of the model

The models for ECN/RED have been validated using MATLAB which is proposed in [13] by setting $p_p = 1$ and $p_u = 0$. In Fig. 2, we show the convergence to the mean field of the ECN/RED model with increasing number of objects. The parameters for the TCP connections and the gateway used for the simulation are as follows: $I_p = 100$, $s_p(i) = 10 \cdot i$, $p_p = 0.7$, $C = 600$, and $\gamma = 5 \cdot 10^{-8}$, and function d halves the size of the windows.

We assume a bursty behaviour of UDP traffic. We model the state of each UDP object as a periodic DTMC consisting of $I_u = 15$ such that $\kappa_{(ui),(u,i+1)} = 1$ for $1 \leq i < I_u$, and $\kappa_{(u,I_u),(u,1)} = 1$. Vector $s_u(i) = (0, \dots, 700, 800, 700)$. The initial state of the TCP connections is chosen with uniform probability between 1 and I_p , while all the UDP objects start from state 1. Figure 2 shows a comparison between the mean field limit and a simulation trace obtained with different numbers of objects in terms of

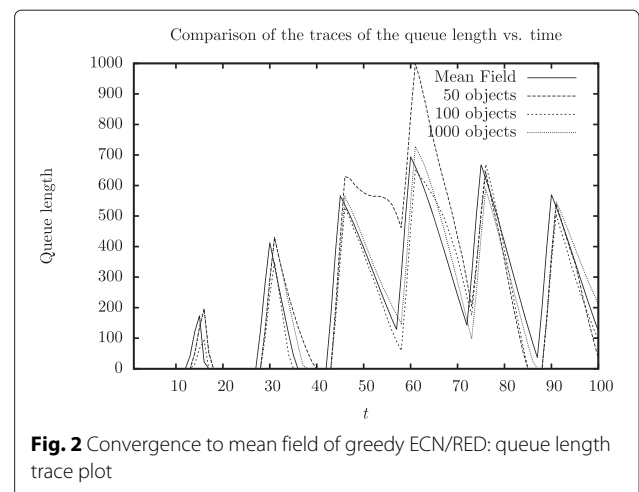


Fig. 2 Convergence to mean field of greedy ECN/RED: queue length trace plot

normalised queue length. Notice the effects of the periodic bursts on the queue length and the reaction of the ECN/RED mechanism to contain the queue length.

Analogously, the SAP-LAW mean field model has been validated against simulation.

6 Performance evaluation

In this section, we use the mean field models presented in Section 4 to compare the ECN/RED and the SAP-LAW mechanisms in terms of expected queue length Q (expressed in normalised number of packets) and system's throughput T (expressed in normalised sent packets per time slot). Assuming that no packet is discarded, i.e. the control mechanism for ECN/RED is performed by sending notification packets rather than by discarding some random packets, we also indirectly derive the expected waiting time using Little's law $W = Q/T$ (expressed in time slots) and assuming a simulation time horizon long enough to stabilise these values. Notice that W denotes the expected number of time slots that a packet has to wait in the queue *before* being sent. In all the models of our experiments, we use for SAP-LAW $Y = 1$ and function \tilde{y} is the identity, i.e., we use a time window of one round trip time to estimate the UDP traffic. Function $d(i)$ for ECN/RED simply halves the size of the current window. The percentage of TCP objects is 70 % of the total. The finite horizon is fixed at 800 time slots.

6.1 Greedy TCP connections and smooth UDP traffic

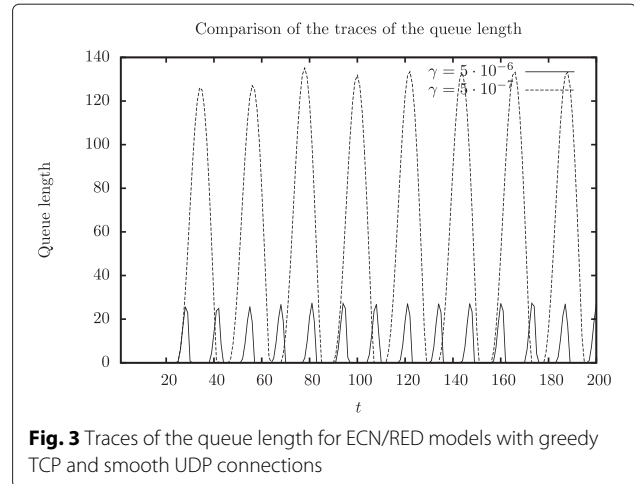
In this section, we study the models for ECN/RED and SAP-LAW under the scenario of greedy TCP connections, i.e. they tend to use all the size of their sending windows, and smooth UDP traffic. The behaviour of each UDP connection is that described in Section 5, but in order to smooth the UDP packet sending, we set the initial state of a connection with uniform probability on the object's state. The results of the experiment and its parameters are shown in Table 5.

Figure 3 shows the queue length trace for the two ECN/RED models obtained with different values of γ .

This scenario is ideal especially for the SAP-LAW mechanism. In fact, the smoothness of the UDP traffic (in the limit, it tends to become constant) allows the gateway to accurately predict the bandwidth to reserve for

Table 5 Comparison of ECN/RED with SAP-LAW under greedy TCP and smooth UDP. Parameters: $l_p = 100, l_u = 10, s_p(i) = 10 * i$, and $C = 600, s_u = (0, \dots, 1000, 3000, 1000)$

	SAP-LAW	ECN $\gamma = 5E - 6$	ECN $\gamma = 5E - 7$
T	591.27	585.69	594.23
Q	0	5.73	53.72
W	0	0.098	0.0901



UDP. On the other hand, the greedy behaviour of TCP connections prevents the SAP-LAW mechanism to overestimate the bandwidth needed by each connection since, after some time, all the sending windows will be opened at their maximum sizes. For this reason, in this ideal scenario, SAP-LAW gives the best performances by obtaining a high throughput with no waiting time.

6.2 Greedy TCP connections and bursty UDP traffic

In this part, we compare the ECN/RED and the SAP-LAW congestion control mechanisms under bursty UDP. We still use the models presented in Sections 4.1 and 4.2. In this case, the SAP-LAW model shows an instable behaviour that proves that this mechanism alone is not sufficient to safely prevent the congestion in any scenario. The issue raises when the UDP traffic is very bursty and the monitor window is small as in our case. The SAP-LAW policy does not take into account the actual queue length but only estimates the instantaneous UDP traffic. Therefore, this parameter must be chosen carefully when implementing this congestion control mechanism. Figure 4

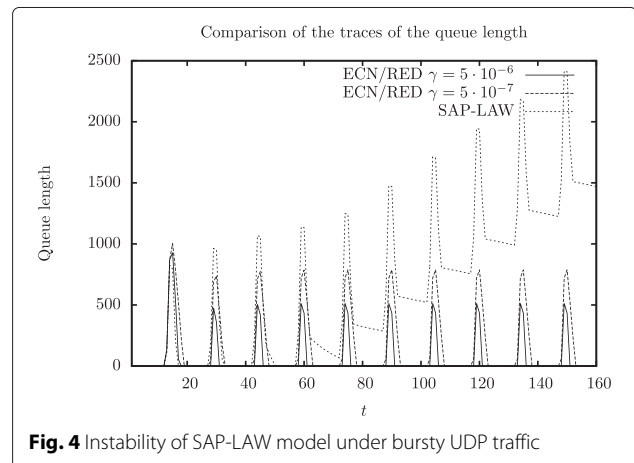


Fig. 4 Instability of SAP-LAW model under bursty UDP traffic

Table 6 Comparison of ECN/RED with SAP-LAW under greedy TCP and bursty UDP. Parameters: $l_p = 100, l_u = 10, s_p(i) = 10 * i,$ and $C = 600 s_u = (0, \dots, 1000, 3000, 1000)$

	SAP-LAW	ECN $\gamma = 5E - 6$	ECN $5E - 7$
T	n/a	276.00	428.85
Q	n/a	64.52	155.34
W	n/a	0.234	0.3622

shows the traces of the queue lengths in the ECN/RED and SAP-LAW. The graphs have been obtained with the same parameters of the experiment run in Section 6.1 but with all identical UDP objects' initial states.

Table 6 shows the performance measures for ECN.

It is worthy of notice that the burstiness of the UDP activity strongly deteriorates the performance measures of TCP traffic. In Table 7, we show the behaviour of the two congestion mechanisms under bursty UDP traffic but in stability.

We can see that although the throughput of SAP-LAW is much higher than that of ECN, also the queue length tends to be higher, so reducing the benefits that we observed in the scenario with smooth UDP.

6.3 Temporary TCP connections

In this section, we assume that each TCP connection has to send a number of packets modelled by an independent geometric random variable with mean w^{-1} . After sending this amount of packets, the connection restarts from state 1. The models that are studied have been presented in Sections 4.3 and 4.4. The results of the analysis are shown in Table 8. The table shows that SAP-LAW pays a reduction of the throughput because the mechanism is fair in dividing the residual bandwidth among the TCP connections not considering that some of them may be in their initial phase. This does not happen for ECN/RED since it tends to reduce the window size of fast connections and allows the new ones to grow.

This is even more evident when we introduce UDP burstiness. Figures 5 and 6 compare the queue length and the bottleneck's throughput for different sizes of w . We assume $w(\omega) = 10^{32\omega}$ and $C = 300, l_p = 100, s_p(i) = 10i, l_u = 15, s_u = (0, 0, \dots, 500, 600, 500)$. Notice that for low values of w , ECN/RED shows its benefits in the queue

Table 7 Comparison of ECN/RED with SAP-LAW under greedy TCP and bursty UDP. Parameters: $l_p = 100, l_u = 10, s_p(i) = 10 * i,$ and $C = 800 s_u = (0, \dots, 1000, 3000, 1000)$

	SAP-LAW	ECN $\gamma = 5 \cdot 10^{-6}$	ECN $\gamma = 5 \cdot 10^{-7}$
T	713.27	333.29	529.4077
Q	153.95	35.20	83.79
W	0.2000	0.1056	0.1583

Table 8 Comparison of ECN/RED with SAP-LAW under greedy TCP and smooth UDP traffic. Parameters: $l_p = 100, l_u = 10, s_p(i) = 10 * i,$ and $C = 800 s_u = (0, \dots, 1000, 3000, 1000), w = 10^4$

	SAP-LAW	ECN $\gamma = 5E - 6$	ECN $\gamma = 5E - 7$
T	219.17	300.06	295.50
Q	0.2160	361.1346	39.335
W	$0.85E - 4$	1.2035	0.133

length control policy as discussed above, whereas when w grows, SAP-LAW takes advantage of being more aggressive in the bandwidth usage and this compensates the assignment of bandwidth to slow connections.

7 Conclusions

In this paper, we have compared the performances of two mechanisms for the congestion control at a shared gateway in a TCP/IP-based network: the ECN/RED and the SAP-LAW. The comparison is performed assuming a large number of TCP and UDP transmissions and considering two different behaviours for the TCP connections (greedy or temporary) and for the UDP traffic (smooth or bursty). We showed that the SAP-LAW gives better performances (in terms of expected queue length and throughput) than ECN/RED in case of greedy TCP connections and smooth UDP traffic. When the first condition is violated, SAP-LAW shows worse performances because it allocates part of the available bandwidth to TCP connections that are in their initial phases. The impact of burstiness on SAP-LAW can be dangerous: we showed that a wrong choice of the time interval used to estimate the intensity of the UDP traffic may lead to instability. As concerns the ECN/RED mechanism, the performances of its congestion control mechanism are not affected by temporary TCP connections since when a potential congestion is detected, the

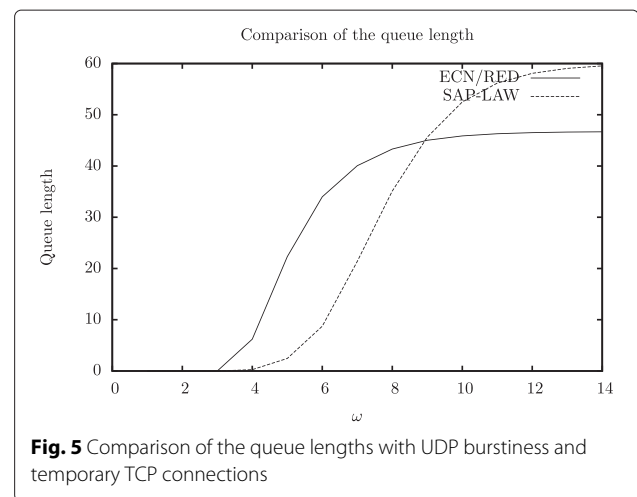
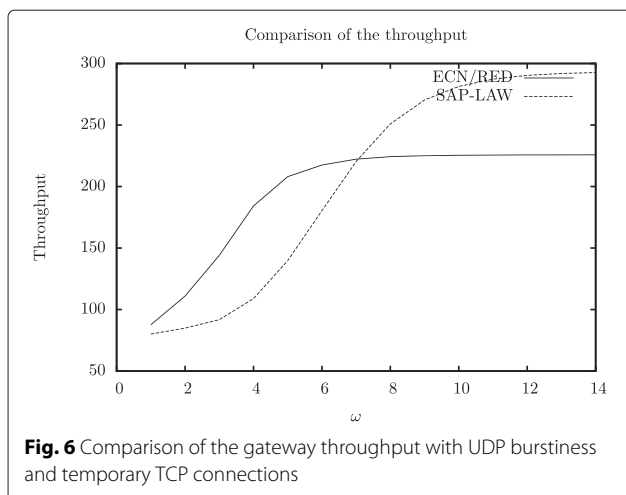


Fig. 5 Comparison of the queue lengths with UDP burstiness and temporary TCP connections



fastest TCP connections are with the highest probability are slowed down. On the other hand, the impact of burstiness on the UDP traffic can strongly reduce the throughput. This happens because during the peak of UDP traffic, the ECN/RED marks the packets with high probability thus reducing the associated connections' window size and does not take into account that the burst is a short-time phenomenon (if there were only TCP connections, it would be correct to expect that a high queue length at time t is followed by a worse situation at time $t + 1$ if countermeasures are not adopted). It is worthy of mentioning that the mean field analysis supports the *fast simulation* that allows to study the detailed performance indices associated with a single object in the mean field regime in a very efficient way.

Competing interest

The authors declare that they have no competing interests.

Author details

¹Department of Computer Science and IT, Institute of Southern Punjab, 9-Km Bosan Road, Multan, Pakistan. ²Department of Robotics and Intelligent Machine Engineering, National University of Sciences and Technology (NUST), Sector H-11, Islamabad, Pakistan. ³Department of Electrical Engineering, The University of Lahore, 1-KM Defence Road, Lahore, Pakistan.

Received: 13 January 2016 Accepted: 3 June 2016

Published online: 02 July 2016

References

1. LL Peterson, BS Davie, *Computer Networks: A Systems Approach*. (Elsevier, Amsterdam, Netherlands, 2007)
2. AS Tanenbaum, *Computer Networks*. (Prentice-Hall, New Jersey, USA, 2003)
3. S Floyd, TCP and Explicit Congestion Notification. *SIGCOMM Comput. Commun. Rev.* **24**(5), 823 (1994)
4. S Floyd, V Jacobson, Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.* **1**(4), 397–413 (1995)
5. I Han, H-S Park, Y-K Jeong, K-R Park, An integrated home server for communication, broadcast reception, and home automation. *IEEE Trans. Consumer Electron.* **52**(1), 104–109 (2006)
6. CE Palazzi, S Ferretti, M Rocchetti Pau, What's in that magic box? The Home Entertainment Center's special protocol potion, revealed. *IEEE Trans. Consumer Electron.* **52**(4), 1280–1288 (2006)

7. M Jeonghoon, R La, V Anantharam Walrand, in *Proc. of INFOCOM 1999*. Analysis and comparison of TCP Reno and Vegas, vol. 3, (New York, USA, 1999), pp. 1556–1563
8. S Liu, T Başar, R Srikant, TCP-Illinois: A loss- and delay-based congestion control algorithm for high-speed networks. *Perform. Eval.* **65**, 417–440 (2008)
9. E Altman, K Avrachenkov, C Barakat, in *Proc. of SIGMETRICS*. TCP in presence of bursty losses, (Santa Clara, CA, USA, 2000), pp. 124–133
10. T Bonald, M May, J-C Bolot, in *Proc. of INFOCOM*. Analytic evaluation of RED performance, vol. 3, (Tel Aviv, Israel, 2000), pp. 1415–1424
11. W Kang, F Kelly, NH Lee, RRJ Williams, in *Proc. of CDC 2004*. Fluid and Brownian approximations for an Internet congestion control model, vol. 4, (Atlantis Paradise Island Bahamas, 2004), pp. 3938–3943
12. MH Malik, M Aydin, Z Shah, S Hussain, in *Industrial Electronics and Applications (ICIEA) 2014 IEEE 9th Conference on*. Stochastic model of TCP and UDP traffic in IEEE 802.11 b/g (IEEE, Hangzhou, China, 2014), pp. 2170–2175
13. P Tinnakornsrisuphap, A Makowski, in *Proc. of INFOCOM 2003*. Limit behavior of ECN/RED gateways under a large number of TCP flows, vol. 2, (San Francisco California, USA, 2003), pp. 873–883
14. J-Y Le Boudec, D McDonald, J Munding, in *Proc. of QEST 2007*. A generic mean field convergence result for systems of interacting objects (IEEE, Edinburgh, Scotland, 2007), pp. 3–18
15. F Baccelli, M Lelarge, D McDonald, in *42nd Annual Allerton Conference on Communication Control, and Computing*. Metastable regimes for multiplexed TCP flows (University of Illinois at Urbana-Champaign, 2004)
16. A Bobbio, M Griboudo, M Telek, in *Proc. of QEST 2008*. Analysis of large scale interacting systems by mean field method, (St Malo, France, 2008), pp. 215–224
17. R Srikant, *The Mathematics of Internet Congestion Control*. (Springer, New Delhi, India, 2004)

Submit your manuscript to a SpringerOpen® journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Immediate publication on acceptance
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com