

# A Control-Theoretic Analysis of Low-Priority Congestion Control Reprioritization under AQM

LUCA DE CICCIO, Politecnico di Bari, Italy

YIXI GONG, Telecom ParisTech, France

DARIO ROSSI, LTCI UMR5141, Telecom ParisTech, Université Paris Saclay

EMILIO LEONARDI, Politecnico di Torino, Italy

Recently, a negative interplay has been shown to arise when scheduling/Active Queue Management (AQM) techniques and low-priority congestion control protocols are used together; namely, AQM resets the relative level of priority among congestion control protocols. This work explores this issue by carrying out a control-theoretic analysis of the dynamical system to prove some fundamental properties that fully characterize the reprioritization phenomenon. In particular, (i) we provide the closed-form solution of the equilibrium in the open loop (i.e., fixing a target loss probability  $p$ ); (ii) we provide a stability analysis and a characterization of the reprioritization phenomenon when closing the loop with AQM (i.e., that dynamically adjusts the system loss probability). Our results are important as the characterization of the reprioritization phenomenon is not only quantitatively accurate for the specific protocols and AQM considered but also qualitatively accurate for a broader range of congestion control protocol and AQM combinations. Finally, while we find a sufficient condition to avoid the reprioritization phenomenon, we also show, at the same time, such conditions to be likely impractical. Therefore, we propose a simple and practical system-level solution that is able to reinstate priorities among protocols.

CCS Concepts: • **Networks** → **Transport protocols**; **Network performance modeling**; *Network protocol design*

Additional Key Words and Phrases: Bufferbloat, AQM, scavenger protocol, simulation

## ACM Reference Format:

Luca De Ciccio, Yixi Gong, Dario Rossi, and Emilio Leonardi. 2016. A control-theoretic analysis of low-priority congestion control reprioritization under AQM. *ACM Trans. Model. Perform. Eval. Comput. Syst.* 1, 4, Article 17 (August 2016), 33 pages.

DOI: <http://dx.doi.org/10.1145/2934652>

## 1. INTRODUCTION

The Internet is a rather complex ecosystem in which different hardware, speaking a babel of languages (i.e., protocols) with numerous dialects (i.e., software implementations), coexists. Equipment vendors, normalization fora, and software developers let all these protocols evolve, ensuring their interoperability. Of course, within each category, a finer-grained taxonomy is possible. For instance, taking the Internet “protocols” category, the layering principle allows us to break down the protocol ecosystem into,

---

The research leading to these results has received funding from the European Union under the FP7 Grant Agreement No. 608533 (NECOMA). This work also benefited from support of NewNet@Paris, Ciscos Chair “NETWORKS FOR THE FUTURE” at Telecom ParisTech (<http://newnet.telecom-paristech.fr>).

Authors’ addresses: L. De Ciccio, Politecnico di Bari, Via Orabona n.4, 70126, Bari, Italy; email: [luca.deciccio@poliba.it](mailto:luca.deciccio@poliba.it); Y. Gong and D. Rossi, Telecom ParisTech, 23 Avenue d’Italie, 75013, Paris, France; emails: {[name.surname@telecom-paristech.fr](mailto:name.surname@telecom-paristech.fr)}; E. Lombardi, Politecnico di Torino, Corso Duca degli Abruzzi, 24, 10129, Torino, Italy; email: [leonardi@polito.it](mailto:leonardi@polito.it).

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org).

© 2016 ACM 2376-3639/2016/08-ART17 \$15.00

DOI: <http://dx.doi.org/10.1145/2934652>

according the Internet (or Open Systems Interconnection (OSI)) terminology, an application level (or L7), a transport level (L4) (usually referred to as the “upper layers”) vs. an inter-networking (L3) and a host-to-network (L2) layer (referred to as the “lower layers”).

Recent evolutions in the “upper layer” have confirmed TCP to be still the largely most commonly adopted congestion control protocol over the Internet for the support of either elastic data transfer or transfer of data with weak real-time constraints such as video streaming. Nevertheless, over the past few years, an ensemble of application-specific layer-7 congestion control algorithms, running over UDP, have been defined. With few exceptions,<sup>1</sup> these protocols typically support applications executed in the background, such as Peer to Peer (P2P) file transfers, file synchronization, and so on, and are targeted to low-priority congestion control (LPCC) services, so background applications do not subtract bandwidth from interactive applications.

Notable examples of LPCC protocols are represented by Microsoft Background Intelligent Transfer Service,<sup>2</sup> TCP-LP [Kuzmanovic and Knightly 2003], TCP-NICE [Venkataramani et al. 2002], and BitTorrent low extra delay background transport (LEBDAT) [Shalunov et al. 2012], which is especially relevant as BitTorrent is still credited as a primary contributor for uplink bandwidth [Sandvine 2014]. Differently from standard TCP, which reduces the source sending rate only in the occurrence of packet loss events, LPCC congestion control schemes decrease the source sending rate as soon as the estimated packet delay grows beyond a given target. As a result of the previous design choices, the LPCC schemes exhibit a less aggressive profile than TCP when they compete for the bottleneck bandwidth, which is typically placed at the access link (especially in upstream, in today scenarios), thus leaving to the TCP flows most of the bottleneck bandwidth.

However, this is true only when TCP flows are able to fill the bottleneck buffer, inducing a significant increase of the packet delay, that is, when buffers are sized sufficiently large and DropTail packet discarding policies are adopted. In this regard, we recall that Internet buffers are traditionally sized sufficiently large to store up 250–500 ms of packets at the nominal linespeed of the link. This simple rule-of-thumb approach, however, may be cause of the “bufferbloat” phenomenon, that is, the unexpected increase of the packet round-trip time that may reach *several seconds*, in which case the real access bandwidth is significantly smaller than the nominal bandwidth under which the buffer has been designed (this may happen over heavily congested 3G/4G [Jiang et al. 2012] and WiFi networks [Nichols and Jacobson 2012] or low-quality Asymmetric Digital Subscriber Line (ADSL) or cable lines [Kreibich et al. 2010]).

To cope with bufferbloat, recent evolutions of the “lower layer” have been specifically designed and especially have started to be deployed in both the EU and U.S.; indeed, infrastructural solutions to the bufferbloat such as scheduling (SFQ [McKenney 1990], DRR [Shreedhar and Varghese 1995]) and Active Queue Management (RED [Floyd and Jacobson 1993], Controlled Delay (CoDel) [Nichols and Jacobson 2012]) techniques, whose adoption has been so far limited, are now becoming commonplace for operators worldwide to improve the quality of experience for their ADSL/cable line customers (e.g., the French ISP Free has employed SFQ since 2005 [Bizon 2005] and U.S. followed suit with CoDel and the Proportional Integral controller Enhanced (PIE) under active development in modems adhering to the Data Over Cable Service Interface Specification (DOCSIS) [White 2014; Hong et al. 2015]). For the sake of simplicity, in what follows, we will refer to both Active Queue Management (AQM) and scheduling disciplines simply as AQM.

<sup>1</sup>The most notable of which is represented by Google’s QUIC, an application-layer protocol over UDP, targeted for TCP replacement in the context of SPDY/HTTP.

<sup>2</sup><https://msdn.microsoft.com/en-us/library/aa363167.aspx>.

While these trends on LPCC and AQM deployment are known facts, the impact on the dynamics of upper-layer LPCC schemes and lower-layer changes, such as the replacement of simple DropTail buffer management policies with smarter adaptive policies, is, however, not completely clear. Our previous simulative and experimental work [Gong et al. 2013b, 2014] has shown that *under AQM, a reprioritization effect may occur with LPCC flows getting a bandwidth share comparable with the TCP share*. The goal of this article is to shed light on this issue by providing a systematic analysis from a *control-theoretic* perspective of the dynamics of TCP and LEBDAT (taken as representative of the whole class of LPCC schemes) sharing an access bottleneck whose buffer adopts an AQM-RED scheme (taken as representative of the whole class of adaptive buffer management schemes [Kuhn et al. 2014]).

The main contributions of this article are as follow.

- We propose a mathematical model in the form of a time-delay non-linear dynamical system that we study both as an open loop (Section 3) and as a closed loop (Section 4) system, significantly extending our preliminary work [Gong et al. 2013a].
- Using the open-loop model, where we fix an external loss probability  $p$  as if imposed by a generic AQM, we fully characterize the reprioritization regions (Section 4.1) and show that, depending on how the parameters of the LEBDAT and AQM-RED schemes are set, different behaviors are possible. Specifically, we derive a simple sufficient condition to avoid the reprioritization phenomenon; unfortunately, we show these settings to be impractical in real network scenarios.
- Using the closed-loop model, where we use RED to drive the queue size to a specific value, we then characterize the stability of the linearized system around the equilibrium (Section 4.2). Specifically, after deriving the boundaries of the stability region, we show that in the unstable region the system exhibits Hopf supercritical bifurcation [Hassard et al. 1981], that is, an oscillatory behavior around an equilibrium is displayed.
- Using packet-level simulation (Section 5) we validate results of our model: We not only show qualitative agreement but also introduce model refinements that reinforce the quantitative agreement, at the price of a tractable complexity (i.e., introducing a non-linearity by capping the minimum amount of LEBDAT packets in a Round Trip Time (RTT) to 1, which happens in practice to avoid indefinite starvation).
- Finally, we discuss (Section 6) a very simple yet effective way to avoid such reprioritization phenomenon, which would already be deployable in practice, without requiring any further development but a minimum amount of explicit information exchange between the upper and lower layers.

From a protocol design perspective, our work shows that the independent evolution of upper and lower layers is a potential source of problems—namely, the above-mentioned reprioritization effect. We additionally demonstrate that, while operational points not requiring an interaction of upper and lower layers to avoid such problems exist, the are of scarce practical interest. We therefore argue that a minimalistic amount of information needs to be exchanged between layers for a clean and practical solution. We are certainly not the first in pointing out problems rooted in layering [Crowcroft et al. 1992], yet we argue that the solution does not mandate a complex cross-layer solution, as the reprioritization can be solved with the exchange of a minimum amount of information (i.e., a single bit [Podlesny and Gorinsky 2011] carried in lower-layer packet headers and set on request of the upper layers).

Overall, the problem we address is not only timely (in reason of the above trends) and broadly applicable (as the effect we outline here holds for several combination of specimens), but our findings are fairly general (given to both the rigor in our methodology and the extent of the reprioritization) and accurate (as per validation with packet

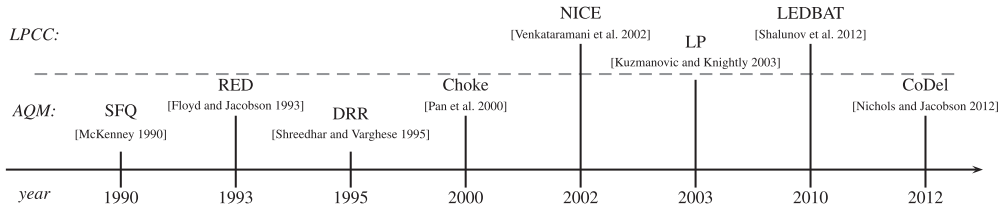


Fig. 1. Timeline of Active Queue Management (AQM) and low-priority congestion control (LPCC) algorithms.

level simulation), and our solution is of practical interest (due to its simplicity and the lack of heavy cross-layering).

## 2. BACKGROUND

### 2.1. Related Work

It would be extremely cumbersome to comprehensively retrace over 20 years of Internet research in these few pages. A historical viewpoint is sketched in Gettys and Nichols [2012]: We extend this viewpoint by reporting in Figure 1 a timeline of research in scheduling/AQM<sup>3</sup> algorithms and LPCC protocols.

Of course, because of space limitations, it was not possible to include all the relevant literature in Figure 1. As such, the classical fair-queueing predecessors, for example, Nagle's round-robin scheduling algorithm [Nagle 1985, 1987] and Weighted Fair Queueing (WFQ) or Generalized Processor Sharing (GPS) [Parekh and Gallagher 1994; Demers et al. 1989] are absent from the picture.

Also prominent AQM schemes that were developed on solid control-theoretic and optimization foundations, such as the Proportional Integral (PI) controller proposed in [Holot et al. 2001] and the Random Exponential Marking (REM) algorithm [Lapsley and Low 1999; Athuraliya et al. 2001] are absent. Similarly, important milestones of the (low-priority) congestion control evolution, such as the Competitive and considerate congestion control protocol (4CP), are not shown for lack of space [Liu et al. 2006].

Still, we believe that the picture is complete enough so the most important takeaway can be correctly gathered; that is, the timeline clearly shows a *temporal separation* of these two research topics, which in our opinion helps to understand why the AQM vs. LPCC interaction assessed in this article was only barely exposed in the past. In this section, we overview the related work separately considering (i) LEDBAT and other low-priority protocols, (ii) the AQM vs. LPCC interaction, and (iii) fluid modeling of TCP and AQM.

**LPCC Protocols.** Protocols such as NICE [Venkataramani et al. 2002], LP [Kuzmanovic and Knightly 2003], 4CP [Liu et al. 2006], and those considered in [Key et al. 2004] share the same low-priority spirit of LEDBAT. We carried out a simulation-based comparison of NICE, LP, and LEDBAT in Carofoglio et al. [2010a], showing that LEDBAT has the lowest level of priority. Some important differences among the above protocols are worth stressing. NICE [Venkataramani et al. 2002] extends the delay-based behavior of TCP Vegas with a multiplicative decrease reaction to early congestion (detected when the number of packets experiencing a large delay in an RTT exceeds a given threshold). Differently from LEDBAT, which reacts to instantaneous one-way delay (OWD) variations, NICE instead reacts to RTT variations, thus possibly reducing the congestion window due to growing delay in the reverse path similar to TCP Vegas

<sup>3</sup>By abuse of language, in the following we refer to both scheduling/AQM algorithms simply as AQM.

[Grieco and Mascolo 2004]. LP [Kuzmanovic and Knightly 2003] modifies the loss-based behavior of NewReno with an early congestion detection based on the distance of the OWD from a weighted moving average calculated on all observations. In case of congestion, the protocol halves the rate and enters an inference phase, during which, if further congestion is detected, the congestion window is set to 1 and normal NewReno behavior is restarted. This differs from LEDBAT, which aims at explicitly bounding the maximum delay introduced in the bottleneck queue, which is particularly important for Voice over IP, video conferencing, gaming, and all other interactive delay-sensitive applications. Finally, a simulation-based analysis of the impact of LEDBAT parameters on its behavior has been recently carried out in Trang et al. [2014]. Trang et al. [2014] focuses on a DropTail bottleneck link shared by LEDBAT and TCP New Reno flows, proposing a set of LEDBAT parameters minimizing the overall LEDBAT bandwidth share.

*Interaction of AQM and LPCC.* To the best of our knowledge, aside from our previous experimental [Gong et al. 2013b] and simulation-based [Gong et al. 2014] work, only Schneider et al. [2010] mentions AQM and a LPCC (namely, LEDBAT) in the same article. In one of the tests, the authors experiment with a home gateway that implements some (non-specified) AQM policy other than DropTail. When LEDBAT and TCP are both marked in the same “background class,” the “TCP upstream traffic achieves a higher throughput than the LEDBAT flows but significantly lower than” under DropTail [Schneider et al. 2010]. This is explicitly mentioned in the LEDBAT RFC, stating that under AQM it is possible that “LEDBAT reverts to standard TCP behavior, rather than yield to other TCP flows” [Shalunov et al. 2012]. In our previous work [Gong et al. 2013b, 2014], we further show that this behavior is general and can arise from the interaction of any scheduling/AQM discipline and LPCC protocol shown in Figure 1, using a twofold methodology including ns2 simulation and experiments from both controlled testbed and wild Internet. The present work differs from Gong et al. [2013b, 2014] in both its depth and methodology: Indeed, we adopt a narrower but profound scope, selecting LEDBAT and RED as representative examples of the LPCC and scheduling/AQM design space that we then mathematically model.

*Fluid Modeling.* Fluid models have been extensively used in the literature to investigate the dynamics of TCP flows [Misra et al. 2000; Hollot et al. 2001; Liu et al. 2003; Marsan et al. 2005; Michiels et al. 2006; Mascolo 1999; De Cicco et al. 2011]. In particular, one line of research (see, for example, Mascolo [1999] and De Cicco et al. [2011]) models the network as a time-delay linear system and TCP congestion control algorithms as Smith predictor controllers whose aim is to track an exogenous reference signal that models the TCP congestion window. The mainstream approach, closer to ours, is to model TCP window dynamics by means of either Delay Differential Equations (DDE) or Partial Differential Equations (PDE) [Misra et al. 2000; Hollot et al. 2001; Liu et al. 2003; Marsan et al. 2005]. We point out that since, generally, a single dominant TCP flavor is modeled [Misra et al. 2000; Hollot et al. 2001; Mascolo 1999] (optionally including unresponsive background traffic [Liu et al. 2003] or short-lived connections [Marsan et al. 2005]), the novelty in this context lies in the definition of a fluid model of *heterogeneous* responsive sources, notably including LEDBAT. As our main innovation is not on the technique per se but on its application to the study of a particular problem, we resort to classic models for TCP [Liu et al. 2003] and RED [Misra et al. 2000], which we extend to incorporate novel popular protocols such as LEDBAT. While the model presented in this work builds on our previous work [Gong et al. 2013a], this work introduces a significant amount of new control-theoretic material: Notably, in Section 3, we provide closed-form solution for the equilibrium in open loop and in Section 4 we provide a stability analysis and a characterization of the reprioritization phenomenon when closing the loop with RED.



*Fairness.* Our main focus in this article concerns *fairness* of the capacity share among heterogeneous control protocols on a bottleneck governed by AQM. While fairness is a long-studied subject, its investigation generally considered settings that rather differed from ours: Indeed, generally speaking, fairness is considered to be a desirable goal, as opposed to an unwanted (reprioritization) effect.

Often, fairness has been tackled in the *intra-protocol* case [Floyd 1991; Cigno and Gerla 1999; Rossi et al. 2010b; Carofiglio et al. 2010b, 2013]; that is, heterogeneous settings of a single protocol flavor. For instance, Floyd [1991] studies RTT unfairness of TCP Reno, while we pointed out the existence of a LEDBAT latecomer unfairness issue [Rossi et al. 2010b]—that we show to be less relevant in the case of short-lived flows and solve for backlogged connections in Carofiglio et al. [2013]. Fairness in the *inter-protocol* case, thus closer to ours heterogeneous control protocols settings, has long been studied as well [Ahn et al. 1995; Alessio et al. 2001; Eshete et al. 2012; Eshete and Jiang 2011]. Old works especially focused on the undesirable side effect of delay-based congestion control of Vegas, which makes it back off in the presence of TCP NewReno [Ahn et al. 1995]. Other work instead focused on the study of different flavors such as TCP Tahoe and TCP NewReno [Alessio et al. 2001]. Even more recent work on the topic has studied issues that differ from ours. Eshete et al. [2012] and Eshete and Jiang [2011] focus on several high-speed variants of TCP; in their case, fairness between the different protocols is thus desirable, while in our settings *unfairness would be desirable* (as it would imply that low-priority property is maintained). Complementary to this work, Eshete and Jiang [2011] design and analyze an AQM scheme (named Approximate Fairness through Partial Finish Tag or AFpFT) that they show, via ns2 simulations, to reinstate fairness in the heterogeneous protocols case [Eshete et al. 2012].

*Our Previous Work.* To better highlight the contributions of this work, we finally contrast them to our own previous work on the topic. First, most of our work on the topic focuses on LEDBAT under a DropTail FIFO discipline [Rossi et al. 2010a, 2010b; Carofiglio et al. 2010a, 2013], while we previously studied the interaction of LEDBAT and AQM in Gong et al. [2013a, 2013b, 2014] (which are thus closer to this work in terms of topic). In terms of methodology, we either perform experiments [Rossi et al. 2010a; Gong et al. 2014], simulations (in Rossi et al. [2010b], Carofiglio et al. [2010a], and Gong et al. [2013b]) or apply fluid modeling techniques (in Carofiglio et al. [2013] and Gong et al. [2013a]) (which are thus closer to this work in terms of methodology).

We now discuss only the closest related work in terms of methodology or topic. In terms of aim, our previous simulative and experimental work [Gong et al. 2013b, 2014] has shown that a reprioritization effect may occur with LPCC flows getting a bandwidth share comparable with the TCP share. While the main aim of Gong et al. [2013b, 2014] is to gather quantitatively accurate results of the reprioritization, the main aim of this article is to study its root cause with a fluid modeling approach.

In terms of methodology, fluid modeling techniques are used in Carofiglio et al. [2013] to propose a sound modification to LEDBAT to relieve the “latecomer unfairness” issue [Rossi et al. 2010b] that happens in DropTail First In First Out (FIFO) settings, so it is the reprioritization investigated in Gong et al. [2013a] that this work extends. However, whereas Gong et al. [2013a] only perform numerical solutions of the proposed fluid model, in this article we adopt a more control-theoretic approach and gather several new insights on the reprioritization issue: notably, (i) we demonstrate, on an open-loop model, a simple sufficient condition to avoid the reprioritization phenomenon; (ii) we investigate the stability properties of the closed-loop model around the equilibrium, deriving the boundaries of the stability region, and show that in the unstable region the system exhibits Hopf supercritical bifurcation; (iii) we validate our model against an extended set of simulative results; and (iv) we discuss and implement a proof-of-concept system that solves the reprioritization issue.

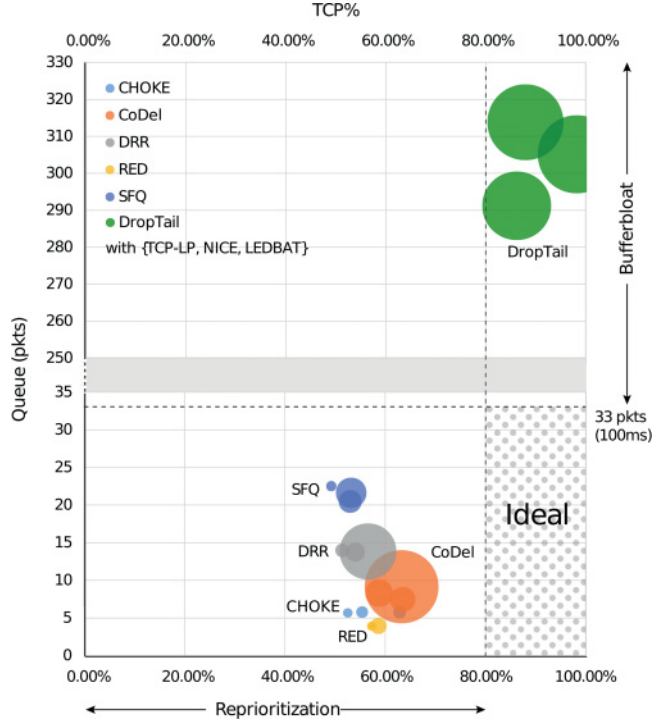


Fig. 2. Interaction of scheduling/AQM disciplines and LPCC protocols.

## 2.2. Motivations

We can now better understand the relevance of the present work. First, it could be objected that, given that the focus of this work is admittedly narrower than that in Gong et al. [2013b, 2014], where we instead consider a larger basket of LPCC and AQM techniques, the value of the present work is similarly narrower. Yet, it is worth pointing out that the negative interplay of AQM and LPCC techniques is a general one, so results gathered for any AQM+LPCC combination are fairly representative of all other AQM+LPCC pairs. We depict in Figure 2 an original representation of the simulations performed in Gong et al. [2013b, 2014]: The results are arranged as a bubble chart, where the center of the bubble represents the average (TCP share%, queue occupancy) for a specific AQM+LPCC pair, and the radius of the bubble represents the standard deviation over the different network parameters considered.

It can be seen that, as expected, under DropTail, TCP monopolizes the bottleneck (TCP% close to 1), which also causes bufferbloat (“bufferbloat” zone in the  $y$ -axis). Conversely, for any combination of AQM+LPCC with AQM  $\in$  {RED, CHOKE, DRR, SFQ, CoDel} and LPCC  $\in$  {LEDBAT, NICE, TCP-LP}, experiments exhibit a reduction of the queue length below the threshold considered to be harmful for interactive communications (100ms or 33 packets (pkt) for the capacity considered in this example) but also a dramatically low TCP share (“reprioritization” zone in the  $x$ -axis). It is worth stressing that, even though the actual values of TCP share or queue size vary *quantitatively* across combination, reprioritization is a *qualitatively* general phenomenon. Additionally, notice that the horizontal distance of the center of all bubbles fall in a very thin band: Approximately, the furthest-apart TCP shares (under SFQ vs. CoDel) are just 10 percentage points apart.

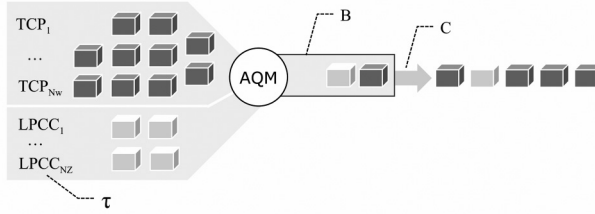


Fig. 3. Network scenario.

Second, we stress that combinations of AQM and LPCC algorithms are already commonplace in the current Internet landscape [Hong et al. 2015]. On the one hand, it has to be observed that LEDBAT is used by BitTorrent, the most prominent P2P application; whereas downlink traffic is dominated by video streaming applications and portals (e.g., Netflix and YouTube), BitTorrent is the top application on uplink traffic and can be credited for over one third of all upload traffic in North America, Latin America, and Asia Pacific, as the latest report [Sandvine 2014] from the Canadian broadband management company Sandvine testifies. As claimed by Brahm Cohen, “LEDBAT is now the bulk of all BitTorrent traffic, [...] most consumer Internet Service Providers (ISPs) have seen the majority of their upload traffic switching to a UDP-based protocol” [Cohen 2011], which is also confirmed by our own independent measurement in customer ISPs [Carofiglio et al. 2013]. On the other hand, it has also to be stressed that AQM adoption worldwide has started to change, with operators implementing scheduling policies in the upstream of the ADSL modem to improve the quality of user experience. For instance, in France, Free has implemented Stochastic Fair Queuing (SFQ) since 2005 [Bizon 2005] and Orange started the deployment of Shortest Queue First (SQF) in 2010 [Benamer et al. 2013]. Similarly, the U.S. followed suit with new promising AQM techniques, such as CoDel [Nichols and Jacobson 2012] and PIE [Pan et al. 2013], under active development in DOCSIS modems [White 2014].

Based on the above observations, we argue that the RED+LEDBAT combination is a reasonable modeling target. Indeed, although there is no agreement on a single specific AQM technique, due to the *quantitatively similar* behavior of any AQM+LPCC combinations (since from Figure 2, TCP shares for any combination fall in a 10% band), results will still be *qualitatively relevant* across combinations. To simplify the analysis, we therefore decide to propose a new model for LEDBAT (which is by far the most-used LPCC protocol on the Internet at present) and resort to known models of RED (which is by far the most popular AQM technique in the literature).

### 3. OPEN-LOOP MODEL

This section provides an open-loop model of the system composed of the LEDBAT and TCP sources that access a bottleneck whose queue is governed by a generic AQM controller.

In particular, the network scenario analyzed in this article is shown in Figure 3.  $N_T$  TCP flows and  $N_L$  LEDBAT flows share a bottleneck with a fixed capacity  $C$ , whose access queue is of size  $B$  and implements a generic AQM scheme.

In Section 3.1 we derive a mathematical model of the *open-loop* system, meaning that we consider the packet dropping probability  $p(t)$  set by the AQM controller as given, that is, it is assumed to be an input parameter of the model. This assumption is not fully realistic, since, in practice, packet loss probability  $p(t)$  arises from queuing dynamics, which in turn are induced by the behavior of the sources. However, the analysis of this simplified system allows us to gather important preliminary information on the behavior of the system. In particular, our study of the open-loop system focuses



Table I. Table of Symbols

| Symbol     | Definition                             |
|------------|----------------------------------------|
| $W$        | TCP window size                        |
| $Z$        | LEDBAT window size                     |
| $q$        | Queue length                           |
| $\tau$     | LEDBAT delay target                    |
| $N_W$      | Number of TCP flows                    |
| $N_Z$      | Number of LEDBAT flows                 |
| $C$        | Bottleneck capacity                    |
| $B$        | Bottleneck queue size                  |
| $T$        | Propagation RTT                        |
| $R$        | RTT ( $T + q/C$ )                      |
| $p$        | Packet drop probability set by the AQM |
| $q_{\min}$ | RED minimum threshold                  |
| $q_{\max}$ | REM maximum threshold                  |

mainly on its equilibrium points and the analysis of their properties (see Section 3.2). Furthermore, we complement our analysis in Section 3.3 by providing a clear physical interpretation of our findings. Finally, in Section 4 we will close the loop considering the effect of queuing dynamics on the loss probability for the specific case of RED.

### 3.1. The Mathematical Model

In this Section we propose a fluid flow model of the open loop system. The notation used throughout the rest of the paper is shown in Table I. We denote the average TCP and LEDBAT window at time  $t$  as  $W(t)$  and  $Z(t)$ , respectively. For TCP, we neglect the slow-start phase, which is instead only optional in LEDBAT. As such, we limitedly model the TCP congestion window dynamics in the congestion avoidance phase. At the reception of the  $(n + 1)$ -th ACK at time  $t_{n+1}$ , the TCP congestion window is updated as follows:

$$W(t_{n+1}) = \begin{cases} \frac{1}{2} W(t_n) & \text{on packet loss,} \\ W(t_n) + \frac{1}{W(t_n)} & \text{otherwise.} \end{cases} \quad (1)$$

Similarly to TCP, LEDBAT reacts to losses by halving the congestion window, but it increases its congestion window at a rate that is proportional to the distance of the queuing delay  $q(t)$  from the delay target  $\tau$  and is always bounded by the TCP ramp-up in congestion avoidance:

$$Z(t_{n+1}) = \begin{cases} \frac{1}{2} Z(t_n) & \text{on packet loss,} \\ Z(t_n) + \frac{1}{Z(t_n)} \frac{\tau - q_d(t_n)}{\tau} & \text{otherwise,} \end{cases} \quad (2)$$

with the current queuing delay  $q_d(t)$  measured as:

$$q_d(t_n) = D(t_n) - D_{\min}, \quad (3)$$

$$D_{\min} = \min_n D(t_n), \quad (4)$$

where  $D(n)$  represents the instantaneous OWD estimate, while the base delay  $D_{\min}$  is the minimum observed OWD. The rationale is that, over a sufficiently large number of observations,  $D_{\min}$  accurately represents the fixed component of the delay (i.e., propagation delay plus negligible transmission delay, which should be the one found when queues are empty) so the  $D(t_n) - D_{\min}$  difference represents the variable component of the delay (i.e., queuing delay plus negligible processing delay).

Notice that host synchronization over the Internet is known to be hard. As such, it is worth stressing that the OWD estimate  $D(t_n)$  is affected by an unknown clock offset

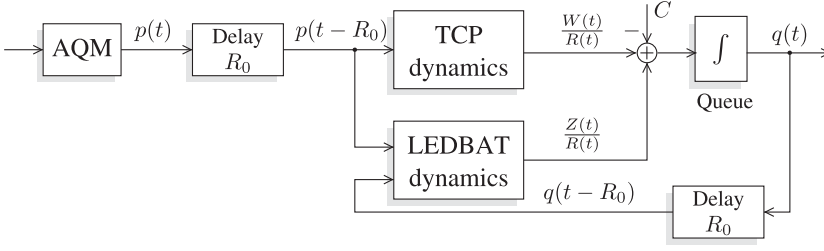


Fig. 4. Block diagram of the proposed model.

between the two endpoints and is thus of no practical use. Conversely, the offset cancels in the difference operation in 3, which is only affected by clock drift, which is of much smaller magnitude and also easier to correct [Cohen and Norberg 2010].

From Equation (2), we gather that ramp-up is as fast as TCP only whenever the queue is empty (Equation (3)), that is,  $\lim_{q_d \rightarrow 0} \frac{\tau - q_d}{\tau} = 1$ . Furthermore, whenever the queuing delay hits the target  $\tau$ , the congestion window settles since, when  $q_d = \tau$  holds, it results in  $Z(t_{n+1}) = Z(t_n)$  for all  $n$ .

To analyze the interactions between sources and queue dynamics, we adopt a fluid flow modeling approach [Misra et al. 2000; Hollot et al. 2001; Liu et al. 2003; Marsan et al. 2005] in which the average dynamics of both sources and queues are described by nonlinear time-delay differential equations. In the following, we denote by  $W_i(t)$  the instantaneous congestion window at time  $t$  for connection  $i$  in the fluid system, by  $R_i(t)$  the Round Trip Time (RTT), and by  $p(t)$  the packet dropping probability set by the AQM algorithm. We consider the case of  $N_W$  TCP and  $N_Z$  LEDBAT connections sharing the same bottleneck, where the flow-level congestion window evolution for the TCP case is adopted from Misra et al. [2000]:

$$\frac{dW_i(t)}{dt} = \frac{1}{R_i(t)} - \frac{W_i(t)W(t - R_i(t))}{2R_i(t - R_i(t))} p(t - R_i(t)), \quad (5)$$

$$\frac{dZ_i(t)}{dt} = \frac{\tau - q(t - R_i(t))/C}{\tau} \frac{1}{R_i(t)} - \frac{Z_i(t)Z(t - R_i(t))}{2R_i(t - R_i(t))} p(t - R_i(t)), \quad (6)$$

$$\frac{dq(t)}{dt} = \sum_{i=1}^{N_W} \frac{W_i(t)}{R_i(t)} + \sum_{i=1}^{N_Z} \frac{Z_i(t)}{R_i(t)} - C 1_{q(t) \geq 0}, \quad (7)$$

where the queuing delay is modeled as  $q_d(t) = q(t)/C$  as suggested in Hollot et al. [2001]. The RTT  $R_i(t)$  is the superposition of two components: one constant component, that is, the propagation delay  $T_i$ , and one time varying, that is, the queuing delay  $q_d(t)$ , which gives  $R_i(t) = T_i + q(t)/C$ . For this reason the RTT cannot be considered to be constant since the component due to queuing delay can be predominant over the propagation delay—which is especially true in case of bufferbloat due to FIFO buffering. Conversely, in case AQM is used to avoid bufferbloat, it could be reasonable to assume the reverse  $q(t)/C \ll T_i$  to hold.

We conclude this section by analyzing Figure 4 that provides a representation of the dynamical system of Equations (5), (6), and (7) in the form of a block diagram. The figure shows that two control components interact to form the overall control system: (i) end-to-end control algorithms, namely TCP and LEDBAT, and (ii) the AQM controller.

The AQM control component sets a packet drop probability  $p(t)$  that is first delayed and then fed in parallel to the TCP and LEDBAT blocks. Thus, we can argue that this control action acts equivalently on both the dynamics of the TCP and LEDBAT flows.

Concerning the end-to-end control algorithms, it is easy to check that the only difference stems from the fact that LEDBAT dynamics is explicitly affected by the instantaneous queue length. With this regard it is interesting to notice that, from a control-theoretic point of view, the term  $-q(t - R_i(t))/(\tau C)$  in Equation (6) plays the role of a “stabilizing” control signal since its sign is always negative or zero ( $q(t) \geq 0$ ). Moreover, it is intuitive to anticipate that the value of the constant positive gain  $1/(\tau C)$  plays an important role: The larger this constant, the larger will be the effect of the control signal  $-q(t - R_i(t))/(\tau C)$  on the dynamics of the LEDBAT flow. The next sections will give a theoretical ground to this statement and show some important properties of the system.

### 3.2. Equilibrium and Properties

In this section we compute the equilibrium of the proposed mathematical model, which allows us to derive some fundamental properties of the system. For the sake of simplicity we consider a bottleneck shared by an equal number of TCP and LEDBAT flows, that is,  $N_W = N_Z = N$ . Moreover, we consider a homogeneous case, in which all the connections exhibit the same RTT (i.e., for both LEDBAT and TCP sources  $T_i = T \forall i$ ). It is also worth noticing that the results presented below are, to a good extent, valid regardless of the AQM control algorithm employed.<sup>4</sup>

**PROPOSITION 3.1.** *The unique equilibrium  $(w, z, q)$  of Equations (5), (6), and (7) when  $N = N_W = N_Z$ , in the case of a generic AQM algorithm setting a steady-state packet drop probability  $p \in [0, 1]$ , is given by:*

$$w = \sqrt{\frac{2}{p}}, \quad (8)$$

$$z = \begin{cases} w & \sqrt{2} < w \leq \frac{C}{2N}T \\ \frac{w}{2} \left[ \sqrt{\left(\frac{Nw}{\tau C} - 2\right)^2 + 4\frac{T}{\tau}} - \frac{Nw}{\tau C} \right] & \frac{C}{2N}T < w \leq (\tau + T)\frac{C}{N}, \\ 0 & w > (\tau + T)\frac{C}{N} \end{cases}, \quad (9)$$

$$q = \begin{cases} 0 & \sqrt{2} < w \leq \frac{C}{2N}T \\ \tau C \left[ 1 - \frac{1}{4} \left( \sqrt{\left(\frac{Nw}{\tau C} - 2\right)^2 + 4\frac{T}{\tau}} - \frac{Nw}{\tau C} \right)^2 \right] & \frac{C}{2N}T < w \leq (\tau + T)\frac{C}{N}. \\ Nw - TC & w > (\tau + T)\frac{C}{N} \end{cases}. \quad (10)$$

The proof of this proposition can be found in the appendix.

**Remark 3.2.** At steady state,  $w$ ,  $z$ , and  $q$  vary as a function of  $p$  according to the three different operating zones<sup>5</sup> that are shown in Figure 5:

- (1) *No reprioritization*  $0 \leq p < 2N^2/(C^2(\tau + T)^2)$ : In this zone,  $z = 0$  and thus the dynamics of  $N$  TCP flows accessing a bottleneck described in Hollot et al. [2001] is recovered.

<sup>4</sup>One case where discrepancies may arise is tied to AQM techniques, such as CHOCe, where also packets in the middle of the queue can be dropped: This can alter the queuing delay seen by packets in the buffer and is not considered in our model. At the same time, despite the approximation the results of the model are valid to a certain extent, since as shown in Figure 2, the reprioritization phenomenon is largely similar across many LPCC and AQM combinations.

<sup>5</sup>In the following, we consider that  $CT/(2N) > \sqrt{2}$ ; otherwise, the first zone would collapse to a point.

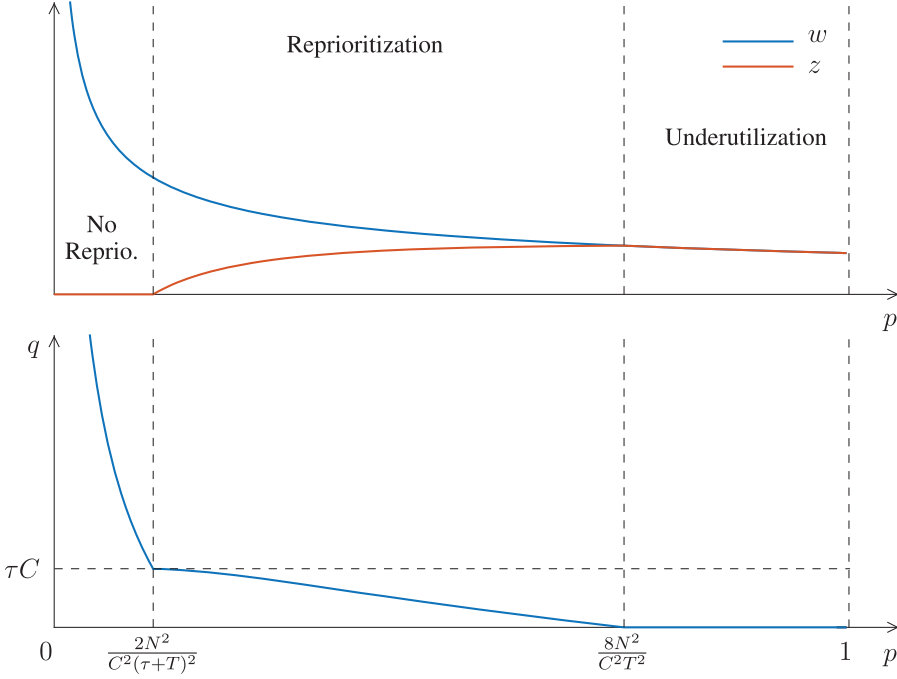


Fig. 5. LEDBAT window size  $z$ , TCP window size  $w$ , and queue size  $q$  at the equilibrium as function of  $p \in [0, 1]$ .

- (2) *Reprioritization*  $\frac{2N^2}{C^2(\tau + T)^2} \leq p < \frac{8N^2}{(CT)^2}$ : In this zone,  $0 < z < w$  and  $q > 0$ , meaning that reprioritization is occurring since the LEDBAT flows are getting a non-negligible bandwidth share;
- (3) *Underutilization*  $\frac{8N^2}{(CT)^2} \leq p \leq 1$ : In this zone,  $w = z = \sqrt{2/p}$  and  $q = 0$ , and in this case the link is not fully utilized since  $\sum w/R + \sum z/R = 2Nw/R < C$ ;

**PROPOSITION 3.3.** *A sufficient condition to avoid reprioritization in the case an equal number  $N$  of TCP and LEDBAT flows access a bottleneck link of capacity  $C$ , round-trip propagation delay  $T$ , with a queue governed by a generic AQM algorithm is that:*

$$\frac{C}{N} \frac{T + \tau}{\sqrt{2}} < 1. \quad (11)$$

**PROOF.** To prove this proposition we need to show that, for any  $p \in [0, 1]$  set by a generic AQM algorithm, the LEDBAT congestion window  $z$  is zero at the equilibrium. From Remark 3.2 we have that  $z = 0$ , that is, no reprioritization occurs, when the following condition holds:

$$0 < p < \frac{2N^2}{(\tau + T)^2 C^2}. \quad (12)$$

To conclude the proof, we observe that when Equation (11) holds, the right extreme  $\frac{2N^2}{(\tau + T)^2 C^2}$  of Equation (12) is always greater than 1, meaning that the only possible equilibrium is  $z = 0$  regardless of the loss probability  $p$  set by the employed AQM control algorithm.  $\square$

*Remark 3.4.* Since Equation (11) is a sufficient condition, when Equation (11) does not hold, reprioritization could be still avoided by a particular AQM control law that sets a steady state  $p$  such that Equation (12) holds.

### 3.3. Discussion

In this section, we analyze the tension existing between the two main components of the overall system: (i) *end-to-end* control algorithms, implemented at the transport or application layer in the hosts and (ii) AQM controllers that are executed in the routers, that is, in the network. Ideally, one could aim at independently tuning the two control algorithms while obtaining the following two properties at steady state: (i) the LPCC flows only get a negligible share of the bottleneck when competing with TCP flows (reprioritization is avoided) and (ii) the queuing delays are minimized (bufferbloat is avoided).

In this section, we investigate whether, by only tuning the LEDBAT delay target  $\tau$ , it is possible to satisfy the two properties mentioned above. Towards this end we consider Proposition 3.3, which provides a sufficient condition to avoid reprioritization. Thus, if we can show that Equation (11) is general enough in practical scenarios and, when it holds, is able to also avoid bufferbloat, we would have satisfied both the properties by only tuning  $\tau$ .

With this purpose in mind, we rewrite Equation (11) as follows:

$$0 < \tau < \sqrt{2} \frac{N}{C} - T, \quad (13)$$

and we observe that if the inequality (13) is verified for some positive  $\tau$ , then reprioritization is avoided regardless of the specific AQM employed. Of course, satisfying Equation (13) would need *a priori* knowledge, or a worst-case estimate, of the number of concurrent TCP and LEDBAT flows  $N$ , the capacity of the bottleneck  $C$ , and the round-trip propagation delay of connections  $T$ . By considering a typical example, we show that using Equation (13) might be unfeasible because it could require setting a delay target  $\tau$  that is either negative or too small.

To begin with, notice that for a typical ADSL whose upload data rate is 500kbps, the transmission of a full Maximum Transmission Unit (MTU) packet takes about 24ms: initial versions of LEDBAT used to set  $\tau = 25$ ms, that is, a packet worth of queuing. However, due to practical limitations (including timestamp precision in Windows OS, clock drift of several ppm in off-the-shelf PCs, etc.), this setting did not allow us to fully exploit the link capacity, which is why the target was later increased to  $\tau = 100$ ms. Thus, even though, in principle, one should set  $\tau$  as low as possible, for practical issues  $\tau$  has to be set to a value that is lower bounded at least by  $\tau_{\min} = 25$ ms (for which we already know that underutilization of the bottleneck occurs).

Therefore, if we require now that  $\tau > \tau_{\min}$ , for Equation (13) to be feasible we have to satisfy the following condition:

$$N > \frac{C}{\sqrt{2}}(T + \tau_{\min}). \quad (14)$$

Now, we turn our attention to the case in which an ADSL whose uplink datarate is 1Mbps, which corresponds to  $C = 100$ pkt/s for packets of fixed size  $P = 1250$ B, is shared by flows whose round-trip propagation delay  $T$  is 50ms. In this case, Equation (14) would require at least  $N = N_Z = N_W = 6$ , that is, a total of 12 flows sharing the bottleneck, a condition that might not hold in general. From these arguments, we can say that it is in general not possible to tune  $\tau$  independently from the AQM control law in order to avoid both reprioritization and bufferbloat.



We now take a different point of view, and we analyze the formula of the TCP bandwidth share  $\rho$  at steady state that is able to both quantitatively explain the interplay between AQM and LEDBAT algorithms and the reprioritization issue shown in Figure 2. In particular, we have:

$$\rho = \frac{w}{w + z}. \quad (15)$$

By plugging the steady state  $w$  and  $z$  in the form of Equation (28) we readily obtain:

$$\rho = \begin{cases} \frac{1}{1 + \sqrt{1 - \frac{q/C}{\tau}}} & 0 \leq q/C \leq \tau \\ 1 & q/C > \tau \end{cases}. \quad (16)$$

It is straightforward to notice that  $\rho \in [1/2, 1]$ :  $\rho = 1$  corresponds to the desirable case in which reprioritization is avoided, and that is obtained when the steady-state queuing delay  $q/C$  is greater than the LEDBAT target delay  $\tau$ ;  $\rho = 1/2$  represents the non-desirable situation where LEDBAT and TCP get the same bandwidth share, which is obtained when  $q/C \rightarrow 0$ . This discussion confirms what we have anticipated at the end of Section 3.1, that is, that the control term  $-q/(\tau C)$  that appears in the LEDBAT differential equation (6) is the main component driving the interplay between the two control algorithms acting on the system.

Interestingly, Equation (16) is only function of the two parameters used by the AQM and the LEDBAT algorithms. The main parameter that can be used to tune LEDBAT is  $\tau$ , whereas AQM algorithms trying to avoid bufferbloat strive to make the queuing delay  $q_d = q/C$  small.

While a 100ms target may be reasonable for an end-to-end protocol, such as in the case of LEDBAT [Shalunov et al. 2012], an AQM may be more precise in measuring the queue size and in adopting more aggressive dropping policy. This is in fact the case for two recently proposed AQM algorithms, namely CoDel and PIE: CoDel employs a target sojourn time of only 5ms, whereas the default queuing delay target used by PIE is 20ms. It is then reasonable to assume that in practical scenarios  $q_d < \tau$  holds and, consequently,  $\rho < 1$ . This explains why the ideal interplay of AQM and LPCC that avoids bufferbloat and reprioritization shown in Figure 2 is unfeasible.

Another important observation can be drawn by analyzing the absolute sensitivity of  $\rho$  with respect to  $\tau$  or  $q_d = q/C$ , that is,  $\partial\rho/\partial q_d$  and  $\partial\rho/\partial\tau$ , in the region  $q_d < \tau$  where reprioritization occurs:

$$\frac{\partial\rho}{\partial q_d}(\tau, q_d) = \frac{1}{2\tau(1 + \sqrt{1 - q_d/\tau})^2 \sqrt{1 - q_d/\tau}}, \quad (17)$$

$$\frac{\partial\rho}{\partial\tau}(\tau, q_d) = \frac{q_d/\tau}{2\tau(1 + \sqrt{1 - q_d/\tau})^2 \sqrt{1 - q_d/\tau}}. \quad (18)$$

Both the sensitivity functions show an asymptote at  $q_d = \tau$ , meaning that a sharp transition phase occurs when moving from the no-reprioritization ( $q_d > \tau$ ) to the reprioritization zone ( $q_d < \tau$ )—yet another reason that makes it impractical to rely on a single parameter selection to drive the overall system performance.

#### 4. CLOSED-LOOP MODEL

In this section, we focus on the properties of the closed-loop system that is obtained when a particular AQM controller, namely RED, is used. As already mentioned, RED has been chosen as a representative AQM algorithm mainly due to its popularity in the literature. It is worth noting that, even though the *quantitative* results derived in the following are specific to RED, the *qualitative* behavior of the system obtained by closing the loop with another AQM algorithm would be similar (see Figure 2).

We start our analysis by deriving the model of the closed-loop system. For this purpose, we take the model described by Equation (5), (6), and (7), and we plug the RED control law that sets the packet dropping probability  $p(t)$  according to a static function  $f : \mathbb{R}_+ \rightarrow [0, 1] \subset \mathbb{R}$  of the queue size  $q(t)$  defined as follows:

$$f(q) = \begin{cases} 0 & q < q_{\min} \\ \max_p \frac{q - q_{\min}}{q_{\max} - q_{\min}} & q_{\min} \leq q \leq q_{\max} \\ 1 & q > q_{\max} \end{cases} \quad (19)$$

where  $\max_p \in [0, 1]$  is the maximum dropping probability when  $q \in [q_{\min}, q_{\max}]$ . In Section 3, we showed that reprioritization is driven by two parameters, that is, the steady-state queuing delay  $q/C$  and the LEDBAT target  $\tau$  (see Equation (16)). For this reason, we consider the system parameters space to be  $q_{\min}$ , which relates to  $q/C$ , and  $\tau$ . In the following, we analyze the closed-loop system to characterize (i) the region of the system parameters where the reprioritization phenomenon is avoided and (ii) the stability region of the equilibrium.

#### 4.1. Characterizing Reprioritization

In order to characterize the reprioritization phenomenon, we have to compute the window size of the LEDBAT and the TCP flows at the equilibrium. Towards this end, we use Proposition 3.1 (Section 3), which characterizes the equilibrium of the open-loop system in a closed form as a function of the packet drop probability  $p$  set by a generic AQM controller.

We start by observing that the cases  $q < q_{\min}$  and  $q > q_{\max}$  are trivial and can be treated as follows. In the case  $q < q_{\min}$ , RED sets a packet dropping probability equal to 0. In this situation, according to Equation (8),  $w \rightarrow \infty$ , and thus, since  $w > (\tau + T)C/N$ , LEDBAT obtains a window size equal to zero at steady state. Moreover, when such a condition holds, it is easy to show by considering Equation (10) that  $q/C > \tau$ , that is, the queuing delay at the equilibrium is larger than the LEDBAT delay target.

The other limit case occurs when  $q > q_{\max}$  and RED would compute a packet dropping probability of 1, meaning that all packets are dropped, a situation of limited practical interest.

From the consideration made above, we limitedly focus on the case  $q \in [q_{\min}, q_{\max}]$ , where  $p$  is set proportional to the error  $q - q_{\min}$ :

$$p(q) = k(q - q_{\min}) \quad (20)$$

with the gain  $k$  defined as  $k = \max_p / (q_{\max} - q_{\min})$ . Thus, substitution of Equation (20) into Equation (10) yields the following equation, which has to be solved to get the steady-state queue length  $q \in [q_{\min}, q_{\max}]$ :

$$\sqrt{1 - \frac{q}{\tau C}} = \frac{TC + q}{N} \sqrt{\frac{k}{2}(q - q_{\min})} - 1. \quad (21)$$

In order to get  $p$  we substitute the steady-state queue length  $q$  solution of Equation (21) into Equation (19). Finally, plugging  $p$  into Equations (8) and (9), we get the steady-state window size of the TCP and LEDBAT flows, respectively.

We now turn our attention to the characterization of the (local) stability properties of the equilibrium. Towards this end we linearize the system around the equilibrium  $(w, z, q)$  to get the following linear time-delay system:

$$\dot{x}(t) = A_0 x(t) + A_R x(t - R), \quad (22)$$

where the state of the linearized system is  $x(t) = [W(t) - w, Z(t) - z, q(t) - q]^\top$ ,  $R = T + q/C$ , and the matrices  $A_0$  and  $A_R$  are defined as follows:

$$A_0 = \begin{bmatrix} -\frac{\sqrt{p}}{\sqrt{2}R} & 0 & -\frac{1}{R^2C} \\ 0 & -\frac{\sqrt{p(1-\frac{q}{\tau C})}}{\sqrt{2}R} & -\frac{\tau+T}{\tau R^2C} \\ \frac{N_W}{R} & \frac{N_Z}{R} & -\frac{N_W w + N_Z z}{R^2C} \end{bmatrix}; A_R = \begin{bmatrix} -\frac{\sqrt{p}}{\sqrt{2}R} & 0 & \frac{1}{R^2C} - \frac{k}{pR} \\ 0 & -\frac{\sqrt{p(1-\frac{q}{\tau C})}}{\sqrt{2}R} & \frac{\tau-q/C}{\tau R^2C} - k\frac{1-\frac{q}{\tau C}}{pR} \\ 0 & 0 & 0 \end{bmatrix}. \quad (23)$$

We are interested in characterizing the (local) stability properties of the unique equilibrium of the closed-loop system as a function of the control loop delay  $T$  and the LEDBAT target delay  $\tau$ . It is well known that the equilibrium of a non-linear time-delay system is locally stable if all the eigenvalues of the characteristic equation associated to Equation (22) are in the left half-plane [Niculescu and Michiels 2007]. Since the eigenvalues associated to the equilibrium  $(w, z, q)$  cannot be found in closed form, we resort to numerically finding the rightmost eigenvalue  $\lambda_{\max}$  of Equation (22) by using the tools described in Michiels [2011]. Now, if the real part of  $\lambda_{\max}$  is negative, then the equilibrium is locally asymptotically stable. By using such a procedure, it is easy to numerically find the *stability region*  $\mathcal{S}$ , that is, the subset of the parameters space  $(T, \tau)$  such that the equilibrium is stable:

$$\mathcal{S} = \{(T, \tau) \in \mathbb{R}_+^2 \mid \text{Re}(\lambda_{\max}(T, \tau)) < 0\}. \quad (24)$$

The boundary of such set is defined as the *stability boundary*  $\partial\mathcal{S}$ .

Let us now characterize another interesting region of the parameters space, namely the *reprioritization region*  $\mathcal{R}$ . As discussed above, reprioritization occurs when the LEDBAT flows obtain a non-negligible bandwidth share or, equivalently, when the TCP bandwidth share  $\rho = w/(w + z)$  is less than 1:

$$\mathcal{R} = \{(T, \tau) \in \mathbb{R}_+^2 \mid \rho(T, \tau) < 1\}. \quad (25)$$

Such region can be easily computed by considering Equation (16). In fact, according to Equation (16)  $\rho(T, \tau) < 1$  holds true whenever  $q/C < \tau$ . Hence, we can write:

$$\mathcal{R} = \{(T, \tau) \in \mathbb{R}_+^2 \mid q < \tau C\}, \quad (26)$$

where  $q$  is the queue at steady state, which can be computed by solving Equation (21). We can then define the *reprioritization boundary*  $\partial\mathcal{R}$  as the boundary of the set  $\mathcal{R}$ , which can be obtained by imposing  $q = \tau C$  in Equation (21). Thus, by plugging  $q = \tau C$  into Equation (21), we get after a little algebra in the following parameterization of the reprioritization boundary:

$$\partial\mathcal{R} = \left\{ (T, \tau) \in \mathbb{R}_+^2 \mid T = \max \left( \frac{N}{C} \sqrt{2 \frac{q_{\max} - q_{\min}}{\tau C - q_{\min}}} - \tau, 0 \right) \right\}. \quad (27)$$

Figure 6 shows both reprioritization and stability regions in the parameter space  $(T, \tau)$  in the case of a bottleneck with a capacity  $C = 100\text{pkt/s}$ , propagation round-trip time  $T = 50\text{ms}$ , and maximum queue length  $q_{\max} = 100\text{pkt}$ , which corresponds to a maximum queuing delay of 1s. It is worth noting that such values are commonplace today, with modem buffers able to hold up to 4s' worth of data [Kreibich et al. 2010]. We have considered  $q_{\min} \in \{1, 2, \dots, 10\}$  pkt to cover both AQM algorithms specifically designed to contain the queuing delay (CoDEL and PIE) and AQMs designed to contain queue length (e.g., RED and Choke [Pan et al. 2000]). Finally, for simplicity and without loss of generality, Figure 6 shows the case of one LEDBAT with one concurrent TCP flow, that is,  $N = 1$ .

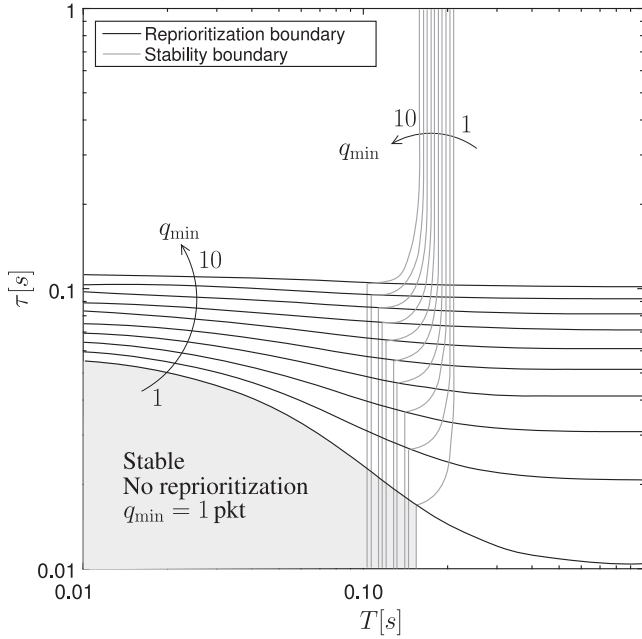


Fig. 6. Reprioritization and stability regions in the case of  $N = 1$ ,  $C = 1\text{Mbps}$ ,  $q_{\max} = 100\text{pkt}$ .

Let us consider a pair  $(T^*, \tau^*)$  and  $q_{\min} = q_{\min}^*$ : The equilibrium is stable if  $(T^*, \tau^*)$  is at the left of stability boundary (gray line) associated to  $q_{\min}^*$ ; similarly, it is simple to show that no reprioritization occurs, that is, the LEDBAT bandwidth share at steady state is zero ( $z = 0$ ) if  $(T^*, \tau^*)$  is below the *reprioritization boundary* (black line) associated to  $q_{\min}^*$ . To give an example, Figure 6 shows where both the stability of the equilibrium and reprioritization avoidance are obtained in the case of  $q_{\min} = 1\text{pkt}$  in the gray the region.

The figure shows that, as previously observed, the reprioritization phenomenon vanishes when  $\tau < q/C$ : In other words, when this condition holds, all LEDBAT flows will by design yield to TCP and the system will behave as in Misra et al. [2000] and Holot et al. [2001]. At the same time, this scenario is unlikely to hold in practice. In fact, as previously observed, end-to-end congestion control protocols such as LEDBAT rely on noisy measures of queuing delay, so they will not be able to guarantee protocol efficiency when  $\tau \rightarrow 0$ .

In the following, we shall characterize reprioritization using a more fine-grained approach as opposed to the binary information that Figure 6 conveys. In particular, we employ the TCP bandwidth share  $\rho$  given by Equation (16) to measure the level of reprioritization, recalling that, when  $\rho$  is close to 1, reprioritization is avoided.

We depict in Figure 7 the TCP share ratio  $\rho$  at the equilibrium for varying user scenarios (i.e., number of TCP and LEDBAT flows  $N$ , LEDBAT target settings  $\tau$ , and RED settings  $q_{\min}$ ).

Figure 7 shows that under AQM the TCP share exhibits a sharp transition phase as soon as  $\tau C$  exceeds  $q_{\min}$ , quickly dropping with a hyperbolic slope from a monopoly situation ( $\rho \rightarrow 1$ ) to a fair share ( $\rho^* \approx 0.58$  for  $\tau C = 2q_{\min}$ ). Interestingly, Carofiglio et al. [2010a] shows that in the DropTail case, which can essentially be recovered from ours by letting  $q_{\min} \rightarrow q_{\max} = B$ , a sharp transition phase from TCP monopoly to a fair share happens whenever  $\tau C \rightarrow B$ . This difference is rooted on the fact that RED

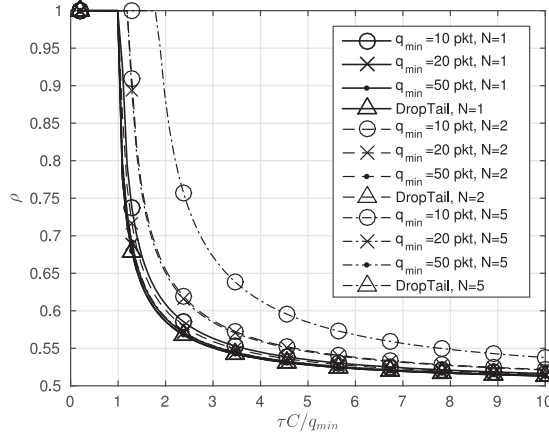


Fig. 7. TCP share ratio  $\rho$  at the equilibrium as a function of  $\tau C/q_{\min}$  for various flow number  $N$ , LEDBAT  $\tau$ , and RED  $q_{\min}$  settings ( $q_{\max} = 100$ ). The lines labeled as DropTail were obtained by letting  $q_{\min} \rightarrow q_{\max}$ .

dropping rates are strictly positive as soon as the queue size exceeds  $q_{\min}$ , whereas DropTail decisions have to wait until the queue exceeds  $B$ .

From Figure 7 we also gather that different RED settings only minimally affect the reprioritization phenomenon. Trivially, since no dropping happens for  $q < q_{\min}$ ,  $q_{\min}$  plays the biggest role in determining the queue size at the equilibrium. Next comes the load factor, that is, the number of flows converging on the bottleneck.

The impact of the LEDBAT target  $\tau$  on the queue size has almost a steplike behavior, which can be explained taking into account the discussion about the absolute sensitivity of  $\rho$  with respect to  $\tau$  (see Equation (18)) or  $q_d = q/C$  (see Equation (17)) provided in Section 3.3.

#### 4.2. Characterizing System Dynamics

In this section, we explore the parameter space with the aim of characterizing the dynamical behavior of the system. For this purpose, let us consider again Figure 6: The figure shows that the main parameter affecting the stability of the equilibrium is  $T$ , whereas the influence of  $q_{\min}$  and  $\tau$  is less important. Indeed, the adverse effect of time delays in control loops is well known and it affects a wide class of dynamical systems [Niculescu and Michiels 2007]. In particular, many time-delay systems display Hopf bifurcations when the time delay  $T$  is varied: The linearized system is stable, that is, all the eigenvalues are in the left-half plane, until  $T$  is equal to a critical value  $T_c$  for which a couple of purely imaginary eigenvalues appear; then, for  $T > T_c$ , the real-part of those eigenvalues increases. When a Hopf bifurcation takes place, a limit cycle, that is, an isolated periodic orbit, branches from the fixed point and a self-sustained oscillatory dynamics is established for  $T \geq T_c$  [Hassard et al. 1981].

Even though it is not the main focus of this work, in the following we shall characterize the properties of the dynamics of the system when  $T$  and  $\tau$  are varied. In particular, we will show that, when the local stability of the equilibrium is lost, that is, when passing from the stable to the unstable region shown in Figure 6, a Hopf bifurcation occurs, and the behavior of the solutions at steady state will change from being stationary to being periodic. With this purpose, we make  $T$  and  $\tau$  vary, and we numerically solve the time-delay differential equations to measure the amplitude of the oscillations as a function of these two parameters. Figure 8 shows a contour plot of the oscillation amplitude function of  $T$  and  $\tau$  obtained in the case of a bottleneck with



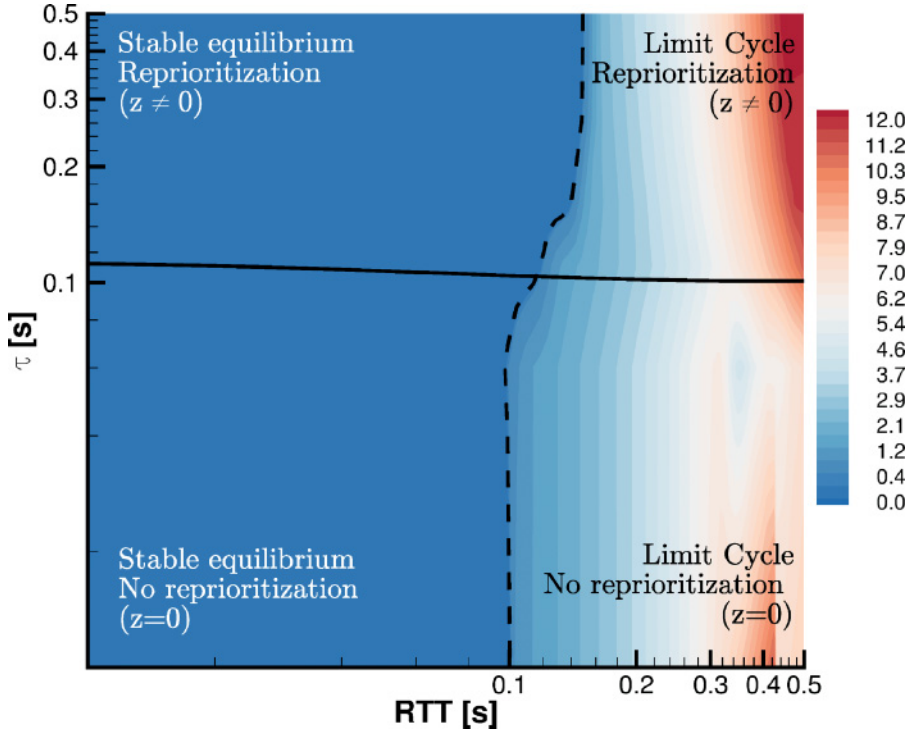


Fig. 8. Bifurcation analysis. Contour plot of the TCP window size oscillation amplitude in the case of  $q_{\min} = 10\text{pkt}$ ,  $q_{\max} = 100\text{pkt}$ ,  $C = 1\text{Mbps}$ ,  $N = 1$ .

capacity  $C = 1\text{Mbps}$  shared by one LEDBAT flow and one TCP flow and governed with RED configured with parameters  $q_{\min} = 10\text{pkt}$ ,  $q_{\max} = 100\text{pkt}$ . It is important to notice that the stability (black dashed line) and reprioritization (solid black line) boundaries are computed using the same procedure employed in Section 4.1 (see Figure 6).

As expected, in the stable region—the one at the left of the stability boundary—the amplitude of the oscillations is zero, meaning that a stationary behavior is obtained; as soon as we leave the stable region, self-sustained oscillations appear whose amplitudes get larger and larger as we move away from the stable region. Figure 9 shows the phase plots in the case of  $T = 0.1\text{s}$  and  $T = 0.2\text{s}$  when  $\tau$  is fixed to  $0.2\text{s}$ : In the case of  $T = 0.1\text{s}$ , since we are in the stable region, the equilibrium is reached asymptotically; on the other hand, when  $T = 0.2\text{s}$ , the trajectory describes a closed orbit not converging to the equilibrium. Let us now consider Figure 10, which shows us the dynamics of the system in each of the four regions shown in Figure 8. The two figures at the left are representative of the region where the system is stable and exhibits a stable response: The figure at the top-left corner shows the case in which reprioritization is obtained and the LEDBAT flow gets a non-negligible share of the bottleneck bandwidth; on the other hand, the figure at the bottom-left corner shows the desirable case where the LEDBAT flow is starved by the TCP flow. The two figures at the right of Figure 10 show the corresponding behavior in the case where the propagation delay  $T$  is larger than the critical value and, as expected, considering the above analysis, a periodic response is obtained.

We conclude this section by showing the time evolution of the system equations when  $N_W = N_Z = 1$  in Figure 11 under either the DropTail (Figure 11(a)) or RED

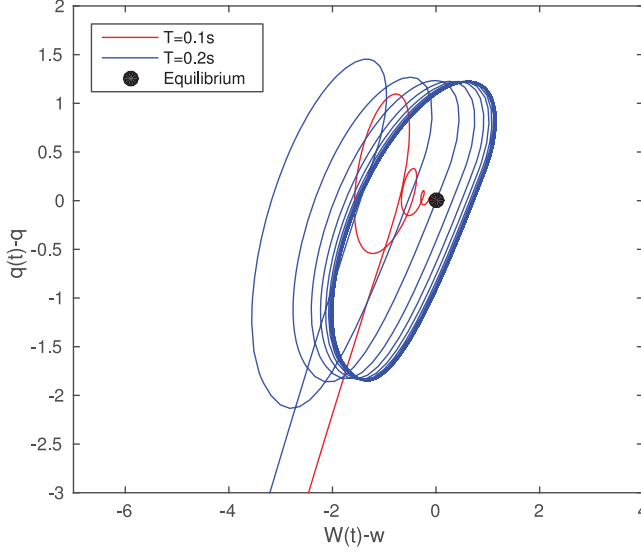


Fig. 9. Phase plots showing a stationary dynamics ( $T = 0.1s$ ) and a periodic dynamics ( $T = 0.2s$ ).  $\tau = 0.2s$ ,  $q_{\min} = 10\text{pkt}$ ,  $q_{\max} = 100\text{pkt}$ ,  $C = 1\text{Mbps}$ ,  $N = 1$ .

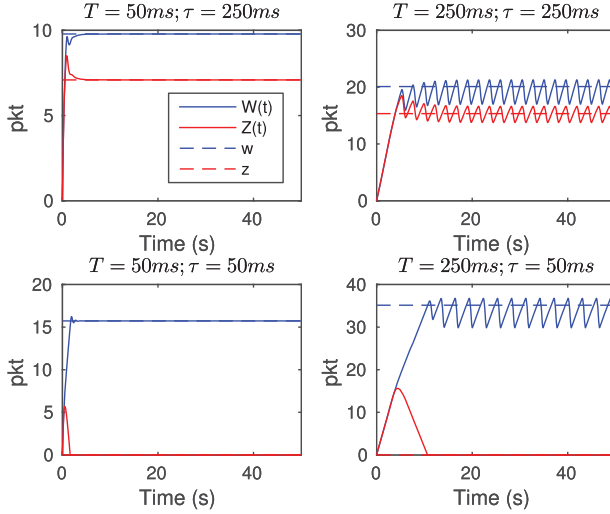


Fig. 10. LEDBAT and TCP window dynamics comparison in the four regions ( $q_{\min} = 10\text{pkt}$ ,  $q_{\max} = 100\text{pkt}$ ,  $C = 1\text{Mbps}$ ,  $N = 1$ ).

(Figure 11(b)–(d)) disciplines. The top plots show the LEDBAT and TCP window congestion windows evolution, while queue  $q(t)$  is reported in bottom plots.

In the DropTail case, we set  $\tau = 0.1s$  and observe the same behavior shown via ns2 simulation in Rossi et al. [2010b]; that is, LEDBAT yields to TCP as expected under DropTail. In the RED case, we set  $q_{\max} = B = 100\text{pkt}$ ,  $q_{\min} = 10\text{pkt}$  for the sake of illustration and let  $\tau$  grow from  $0.1s$  (Figure 11(b)) to  $0.2s$  (Figure 11(c)) and  $0.4s$  (Figure 11(d)). Notice that in Figure 11(b), RED drastically reduces the queue size and lets the TCP rate match the capacity after a short transient. Yet, when the target  $\tau$

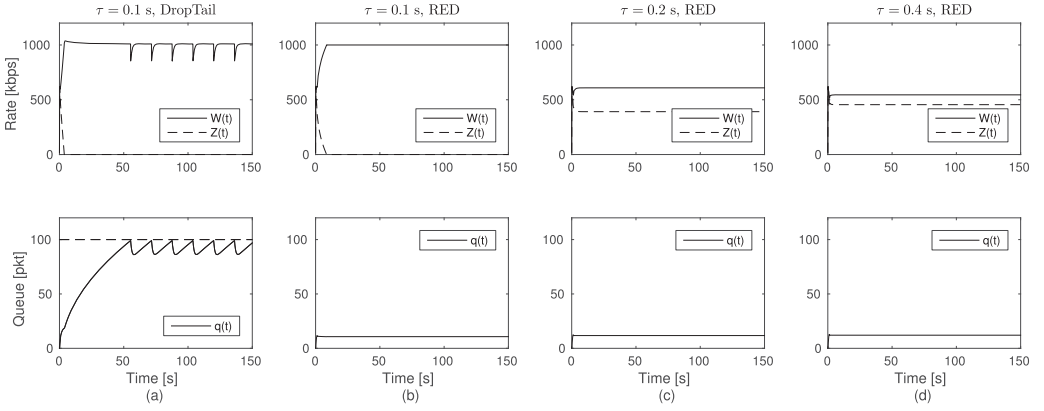


Fig. 11. Reprioritization phenomenon: Time evolution of  $W(t)$ ,  $Z(t)$ , and  $Q(t)$  under DropTail (a) and RED ((b)–(d)) for different values of  $\tau$ .

increases in Figure 11(c) and Figure 11(d), LEDBAT becomes increasingly aggressive under RED and competes more fairly against TCP.

To avoid cluttering the pictures, we do not report here the behavior of LEDBAT for increasing target  $\tau$  under DropTail: From the simulation-based sensitivity analysis reported in Carofiglio et al. [2010a], it emerges that LEDBAT yields to TCP for a large range of  $\tau < B/C$  values, and only whenever  $\tau$  approaches (or exceeds)  $B/C$  LEDBAT behavior does it become as loss-based as TCP.

## 5. VALIDATION

In this section we validate a subset of the numerical results obtained by integrating the time-delay differential equations modelling the system against those obtained from the ns2 simulator. We point out that we take care in the fidelity of our simulation environment: As such, we use the ns2 Agent/TCP/Linux module, which employs the Linux kernel code (both for TCP and LEDBAT, for which we use our own ns2 implementation of LEDBAT, available as open source<sup>6</sup>). The realism of the simulation is further confirmed by matching with experimental results as confirmed in Gong et al. [2014].

In what follows, validation is performed on the most challenging (in terms of matching the simulation vs. fluid model results to get conservative estimate of model accuracy) and relevant scenarios (in terms of practical relevance). Focusing only on a single pair of LPCC (i.e., LEDBAT) and AQM (i.e., RED) allows us to explore a significant fraction of the parameter space (over 40,000 simulation settings are explored).

We additionally point out that we released all scripts and scenarios<sup>7</sup> to reproduce the simulation (as well as the experimental) results of Gong et al. [2014], where the generality of the reprioritization is confirmed over five AQM techniques (DropTail, RED, CHOCe, DRR, SFQ, FQCoDel) and three LPCC protocols (LEDBAT, NICE, TCP-LP), over an extended set of scenarios (about 3,000 varying simulation settings, covering, however, a larger fraction of the LPCC and AQM share).

### 5.1. Scenario

We argue that the most challenging scenario, in terms of matching results gathered via simulation and fluid model, is the one with a small number of flows. This is intuitive, since in the case of multiple backlogged connections, statistical multiplexing will

<sup>6</sup><http://www.enst.fr/~drossi/ledbat>.

<sup>7</sup><http://www.enst.fr/~drossi/dataset/ledbat+aqm>.

smooth out the impact of events, such as TCP retransmission time out, that would otherwise cause discontinuities in the case of few connections. At the same time, we also argue that the most practically relevant scenario is precisely one with a relatively small number of flows. Indeed, since the bottleneck is the user access, the number of concurrent connections will be likely small, even considering multiple applications/users in the household.

We consider both the Cloud and the P2P cases. In the Cloud case, it is easy to see that a small number of connections will be opened, at any given time, for a specific service. While considering a single user, even the server contacted will evolve over time (e.g., due to load balancing); this likely happens over timescales that are much larger with respect to the short timeframes that we consider as “backlogged” data transfers (i.e., from tens of seconds to minutes) in this article. Hence, the number of backlogged connections is upper bounded by the number of Cloud services the user subscribes to, such as DropBox for data, GoogleMusic for music, and Picasa for pictures/videos. Additionally, the number of simultaneous connections also depends on the on/off synchronization pattern toward the Cloud. As users are not continuously generating all kind of data at the same time, it is reasonable to envision only a moderate number of concurrent backlogged connections per household, some of which may be lower priorities (e.g., pictures) over others (e.g., critical data, backup).

Consider next the P2P case, where it makes sense to consider file-sharing applications such as BitTorrent due to its popularity, and since it introduced LEDBAT in the first place precisely due to the bufferbloat problem. In BitTorrent, pipelining of piece requests at the application level can cause multiple chunks to be transmitted consecutively over the same connection at the transport level. Since BitTorrent limits the number of concurrent slots to about<sup>8</sup> 4 per torrent, the number of concurrent connections will be again small. Moreover, BitTorrent peers periodically evaluate the throughput toward other peers every tens of seconds, and connections are maintained in case of good end-to-end throughput: Coupled to pipelining, this entails that over the tens of seconds to minute timescale, connections can be considered backlogged.

## 5.2. Model Validation against ns2 Simulations

From the above discussion, in the following we will start considering an equal number  $N = N_W = N_Z$  of flows and let the total number of flows vary in the  $2N \in [1, 20]$  range. We next consider a fixed population of  $N$  flows and let the number  $N_W$  of TCP flows vary in  $[1, N - 1]$ .

Unless otherwise stated, we consider homogeneous RTT delay settings with propagation delay  $T = 50\text{ms}$  (to which we add a jittering component of  $1\text{ms}$  to avoid synchronization of the congestion window dynamics). To precisely characterize system equilibrium properties, we will let the LEDBAT target  $\tau$  vary; as for that in the uTorrent implementation of LEDBAT, this can be easily done by adjusting the `net.utp_target_delay` settings.

Without loss of generality, we consider a single access bottleneck and fix the capacity to  $C = 1\text{Mbps}$ , typical range for ADSL/Cable access. The bottleneck buffer can accommodate up to  $B = q_{\max} = 100$  packets that, considering  $P = 1250\text{B}$ -sized packets for simplicity, corresponds to a maximum queuing delay of  $1000\text{ms}$ . Notice that these values are commonplace nowadays, with modem buffers able to hold up to  $4\text{s'}$  worth of traffic [Kreibich et al. 2010].

We point out that our goal is not to provide an exhaustive coverage of all scenarios considered in Gong et al. [2014] (as scripts to reproduce experiments and simulations are available as pointed out before). Rather, our main aim here is to validate the most

<sup>8</sup>The limit actually increases with the square root of the uplink capacity.

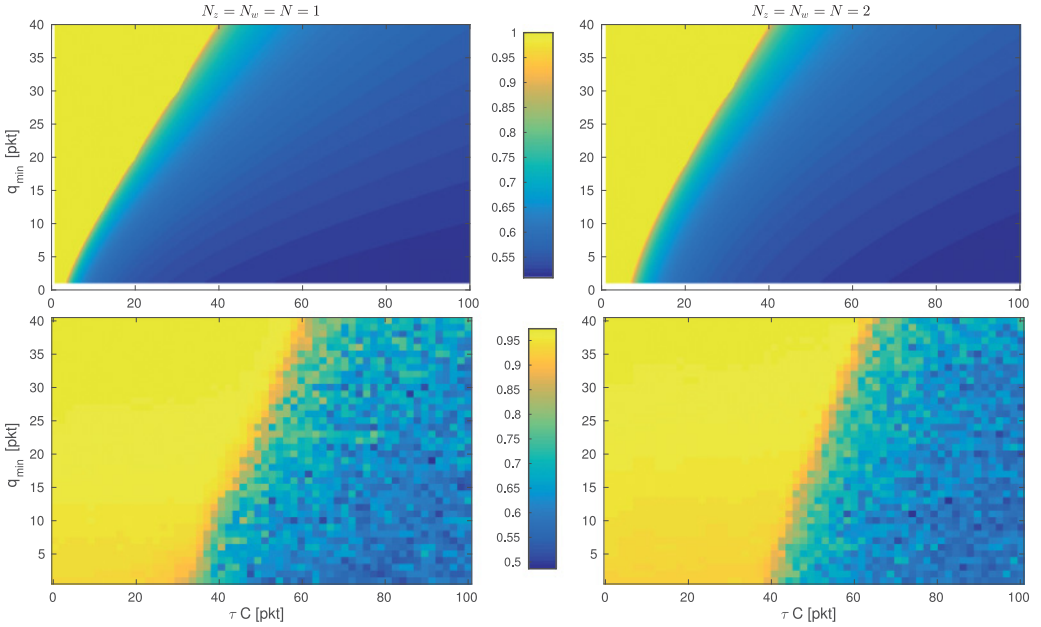


Fig. 12. Heatmaps of  $\rho(\tau, q_{\min})$  in the case of either  $N = 1$  (left) or  $N = 2$  (right) obtained using the proposed model (top) or ns2 simulations (bottom).

representative instance of our results, which is clearly represented by the TCP share ratio  $\rho$  that precisely quantifies the reprioritization.

As we have previously seen,  $q_{\min}$  and  $\tau$  have by far the biggest role in determining the TCP share curve, followed by the number  $N$  of flows in the bottleneck. Hence, in Figure 12 we compare the TCP share ratio  $\rho = w/(z + w)$  as a function of  $\tau C \in [5, 100]$ pkt and the RED minimum threshold  $q_{\min} \in [5, 40]$ pkt obtained either numerically (top figures) or via ns2 simulations (bottom figures) in the case of  $N = 1$  (left) or  $N = 2$  (right). Each point in the heatmaps reports simulation results averaged over 10 runs, corresponding to about 40,000 runs overall.

The figure shows that the proposed model is able to qualitatively capture the reprioritization phenomenon that is observed with ns2 simulations (and experiments) across the considered parameter space  $(\tau C, q_{\min})$ . In particular, by fixing  $q_{\min} = q_{\min}^*$  and moving from the left to the right on the line parallel to the  $x$ -axis connecting  $(0, q_{\min}^*)$  to  $(\tau_{\max} C, q_{\min}^*)$ , an abrupt decrease of the TCP bandwidth share  $\rho$  occurs for any  $q_{\min}^*$ . However, Figure 12 also shows that the model overestimates the reprioritization: In fact, ns2 simulations provide a zone where  $\rho \simeq 1$  (no reprioritization), which is larger than the one obtained by using the model. We shall return shortly to this matter and propose a refinement of the model to address this issue. Finally, Figure 12 shows that the effect of increasing the number  $N$  of concurrent flows on reprioritization is qualitatively the same, that is, with an increased number of flow reprioritization is mitigated. This result also confirms the theoretical findings conveyed by Proposition 3.3.

### 5.3. Model Refinement

We now return on the issue of the model underestimation of the TCP share that has been observed in Figure 12 to provide a refined model. Towards this end, we fix  $q_{\min} = 10$ pkt,  $q_{\max} = B = 100$ pkt, and  $\max_p = 0.1$  and consider three traffic scenarios  $N_W = N_Z = \{1, 5, 10\}$ . Because of the user access bottleneck, this number of flows



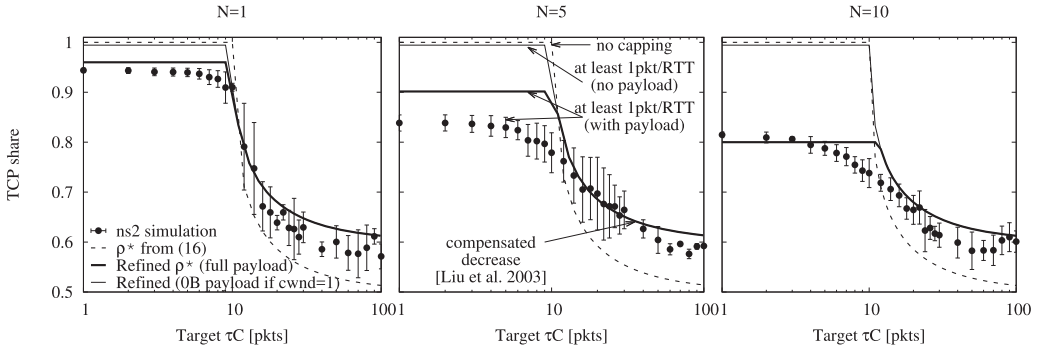


Fig. 13. Simulation validation of TCP share ratio  $\rho$  at steady state as a function of  $\tau C$  for various traffic scenarios  $N_W = N_Z = \{1, 5, 10\}$  and  $q_{\min} = 10\text{pkt}$ .

is reasonable (consider that BitTorrent typically uses four LEDBAT connections in parallel per torrent, whereas the push toward HTTP2 will keep the number of TCP connection limited for Web traffic). Figure 13 compares average simulation results (solid point, with standard deviation bars over multiple runs) against the equilibrium given by Equation (16), discussed previously (dotted line), and a slightly more accurate version (solid black line) that compensates two simplifications of the fluid model that we discuss in the following. Notice indeed that Equation (16) captures reasonably well the essence of the reprioritization phenomenon. Still, two quantitative discrepancies arise.

First, it can be seen that for values of  $\tau C > q_{\min}$  the model underestimates the TCP share. This results from a known problem of the TCP model presented in Liu et al. [2003] that this work extends; that is, Liu et al. [2003] is known to underestimate the TCP congestion window with respect to simulation, which can be easily compensated by taking into account a multiplicative decrease factor of 1.5 (instead of 2) as in Liu et al. [2003]. The refined equilibrium takes into account this correction and is significantly more accurate when  $\tau C > q_{\min}$ .

Second, recall that when  $\tau C < q$ , the model degenerates into a simpler one in which only TCP flows compete on the bottleneck, hence  $\rho = 1$ . In practice, however, we know that LEDBAT will keep sending a minimum of 1 packet per RTT: This is done to continuously measure the queuing delay, at very low frequency and intrusiveness. LEDBAT does this in order to promptly react to queuing delay reduction and effectively utilize the spare capacity as soon as the link becomes free again. Hence, in case  $\tau C < q$ , our model of LEDBAT window dynamics can be, in principle, easily refined to account for this effect by introducing a non-linearity (i.e, capping from below window) at  $z = 1$ . As a result, the capacity available for TCP is reduced proportionally to the number of LEDBAT flows, that is,

$$\rho < 1 - \frac{N}{\frac{CT_p}{p} + q}.$$

Observe that the new equilibrium point taking into account this second correction is significantly more accurate with respect to Equation (16) when  $\tau C < q_{\min}$ , though one should notice that accuracy is obtained at the price of having to handle the additional complexity brought by the additional non-linearity. At the same time, we point out that a scavenging service such as LEDBAT could also favor the use of non-homogeneous packet sizes. Indeed, in case of congestion, LEDBAT sends at the minimum rate of 1 packet per RTT only to continuously measure the queuing delay with low intrusiveness. It follows that LEDBAT could prefer to use small packets (e.g., 0 bytes payload) when

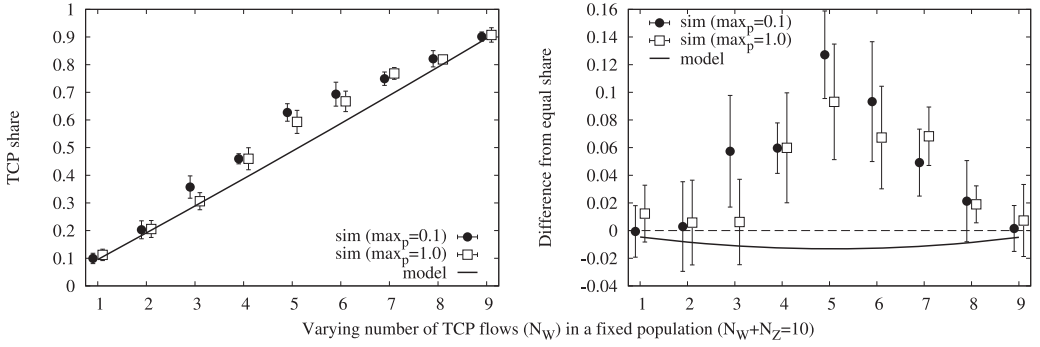


Fig. 14. Simulation validation of TCP share ratio  $\rho$  at steady state (left) and difference with respect to an equal share (right) as a function of the number  $N_W$  of TCP flows for various AQM configurations.

the congestion window is  $z = 1$  (at most, with the exception of the very first packet of the connection when  $z = 1$  is not a signal of congestion). In this case, the refinement due to the window size capping could become of negligible interest (solid line denoted with “OB payload” in Figure 13), as the refined model is very close to the original one in this case (notice that simulation do consider only homogeneous packet sizes for the sake of simplicity and to avoid cluttering the picture). Hence, the simplest model may be enough for a qualitatively sound analysis of reprioritization.

We now confirm that the settings we consider are conservative for what concerns model accuracy by considering a scenario with a heterogeneous number of flows. We now fix  $\tau = 100\text{ms}$ ,  $q_{\min} = 10\text{pkt}$ , and  $q_{\max} = B = 100\text{pkt}$ ; let  $\max_p \in \{0.1, 1\}$ ; and consider a varying number of  $N_W = [1, 9]$  flows out of a total population of  $N = N_Z + N_W = 10$  flows. Figure 14 shows the TCP share ratio  $\rho$  at steady state (left) and the difference with respect to an equal share  $\rho - 1/N$  (right) as a function of the number  $N_W$  of TCP flows for various AQM configurations. From the picture, it can easily be seen that (i) the model closely follows the simulation results; that (ii) the largest discrepancy is observed under the homogeneous traffic model; (iii) that the model is conservative with respect to an equal share, slightly conservatively estimating the actual TCP share; and (iv) finally confirms that, as anticipated, the homogeneous traffic scenarios of Figures 12 and 13 provide rather conservative results of the model accuracy.

Overall, results in this section confirm that the model is able to both qualitatively and quantitatively (with refinements) assess the extent of the reprioritization phenomenon. On the one hand, the simplest model allows us to gather fundamental insights of qualitative relevance about the root cause of reprioritization. On the other hand, the refined model enables a quantitatively accurate assessment of the expected system performance. Clearly, a greater level of accuracy requires a greater level of detail (e.g., different TCP congestion avoidance protocols, slow start, fast recovery, etc.), which can only be captured with simulations or experiments. At the same time, we also argue that that quantifying the *exact* level of reprioritization is less relevant for practical purposes; that is, as users will likely be interested in knowing whether their non-critical bulky transfers are indeed lower priority with respect to critical continuous backups or if they compete on a roughly equal basis. In light of this, we have that our model of LEDBAT vs. RED interaction also captures the TCP share in a qualitatively accurate fashion for all the  $\{\text{RED, CHOCe, DRR, SFQ, FQCoDel}\} \times \{\text{LEDBAT, NICE, TCP-LP}\}$  cases (recall from Figure 2 in Section 2.2 that TCP shares for any of the above combination fall in a 10% band). Still, it is worth pointing out that while our model qualitatively captures reprioritization for many system combinations, this is

more a consequence of the quantitative similarity across systems rather than because the model truly encompasses all dynamics of such systems.

To summarize, we formulated the system as a set of DDE governing the TCP and LEDBAT window dynamics in Equations (5)–(7). We then show in Section 3 that, when the system is open loop, three regions exist, and especially one of these regions avoids reprioritization (see Proposition 3.3). Furthermore, we have characterized the stability of the equilibrium and the reprioritization properties in the parameters space  $(T, \tau)$  in the case where the loop is closed with RED (see Section 4). Finally, this section provides a validation of the model and proposes a refined model to improve accuracy in assessing the expected system performance.

## 6. SOLUTION

Recent evolutions on the upper (i.e., applications) and lower layers (i.e., infrastructure) seem to suggest that AQM and LPCC protocols will have to coexist: Indeed, popular applications are developing delay-based congestion control protocols such as BitTorrent/LEDBAT, on the one hand, whereas operators are starting to deploy AQM/scheduling on the user access uplink, on the other hand. As such, it is imperative to find solutions to the negative AQM/LPCC interplay that we have shown in this article. While a general solution is hard to find, as is evident by the current standpoint after over 20 years of research, a patch to this specific problem may be within reach.

Some might argue that small buffers would be enough to solve bufferbloat altogether. Yet, there are several reasons why this simple solution is not sufficient. First, in the presence of too-small buffers, it would be difficult for TCP and other congestion control to fully saturate the capacity, causing an undesirable efficiency loss. Second, deciding a buffer size is a matter of concern per se: Consider WiFi links, where the capacity may fluctuate widely over time, so no single buffer size can at the same time (i) be large enough to support TCP congestion control and (ii) rule out bufferbloat in a fast-to-slow transition from 54Mbps to 2Mbps. Finally, jeopardization of relative priorities are not solved by small buffers, as we show in Gong et al. [2013b, 2014].

An ideal solution should achieve two goals: (i) meet quality-of-service constraints while (ii) respecting relative levels of priorities among protocols. Quality-of-service constraints clearly translate into upper bounding the queuing delay, which we know is used by protocols to enforce their relative priorities. Since even a single TCP flow may bufferbloat the others, the solution *needs AQM*, as otherwise the quality of-service constraints would be violated. At the same time, to avoid the LPCC reprioritization phenomenon, we argue that classification capabilities will be needed in AQM to account for flows' *explicitly advertised* level of priority.

Although, in the more general case, classification has failed to be adopted (IP Type Of Service (TOS) field, DiffServ, etc.), and the ability to claim *higher* priority could be easily gamed, in a hybrid AQM vs. LPCC scenario, it makes sense for flows to claim a *lower* priority; indeed, generally malicious sources would tend to exploit resources by claiming high priority, so if the source is marking its own packets as low priority, for the scheduler there is no reason not to follow the application desire by prioritizing dropping of low-priority packets. We believe that this subtle difference can make an important practical difference in terms of deployability.

A simple way to express low priority would be to let applications exploit the IP TOS field.<sup>9</sup> While the overloading of the IP TOS can be troublesome within an operator network, this is not an issue in the home. Indeed, the usefulness of the IP TOS is not end to end but merely meant as a low-priority signal to the box in the user home, where the

<sup>9</sup>In this section, we describe the proposed solution in the case of IPv4. Nevertheless, in the case of IPv6 the solution can be deployed by using the Traffic Class field in place of the TOS field.

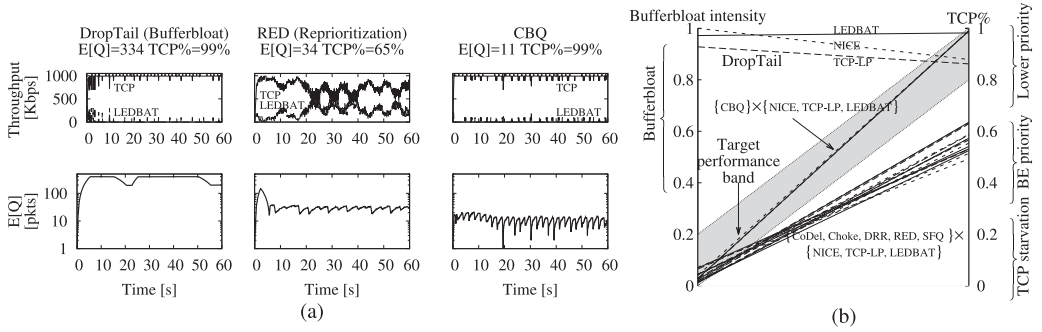


Fig. 15. Assessment of proof-of-concept solution: (a) temporal evolutions of system dynamics and (b) performance comparison via a parallel coordinate plot. Notice that the proposed solution falls in the ideal performance band (gray shaded zone) and holds irrespectively of the LPCC protocol used.

bottleneck and contention arise. Hence, IP TOS could be leveraged by the ISP Customer Premise Equipment (CPE) in the user's home to apply differential treatment to best-effort vs. low-priority traffic (e.g., different AQM loss profiles, different scheduling weights), after which the end-user IP TOS value is no longer useful and can be rewritten by the CPE (or at the DSLAM or BRAS, etc.) in the network of an operator using DiffServ if needed.

The advertised TOS value could be exploited by the CPE to either prioritize or shape the traffic, similarly to Podlesny and Gorinsky [2011]. We demonstrate the soundness of this latter solution in Figure 15. We resort to Class Based Queueing (CBQ) [Floyd and Jacobson 1995] and set up two classes: a Best Effort (BE) for TCP and a LPCC for LEDBAT that are served by a Weighted Round Robin (WRR) scheduler, with the “formal” CBQ algorithm [Floyd and Jacobson 1995]. We set weight of the CBQ/BE class to 0.99, relegating by design the CBQ/LPCC to starvation in presence of TCP traffic. At the same time, when the TCP traffic is absent, flow in the LPCC class can compete and exploit all the available bandwidth. CBQ/BE and CBQ/LPCC queues are further managed by an AQM/scheduling mechanism (SFQ in the example) to ensure that any of these queues does not grow unbounded.

The plots in Figure 15(a) show the temporal evolution of the instantaneous throughput of a TCP and a LEDBAT flows sharing the same 1Mbps bottleneck. While DropTail yields to bufferbloat and AQM (RED) to reprioritization, our solution does manage to reinstate the relative levels of priorities without causing bufferbloat. The figure is further annotated with the average buffer occupancy  $E[Q]$  and TCP% share computed over 10 simulations runs. Of course, we are not advocating that the weights we have used in this example are of general use (for instance, shares greater than 1% may be reasonable in some practical cases). As such, we stress that our main contribution here is not on the fine-tuning of the proposed solution but rather to show the soundness of our design in the most challenging case; that is, avoiding reprioritization (i.e., link monopolized by TCP) and bufferbloat (i.e., keeping the queue occupancy as low as possible) at the same time.

To better contrast solutions, we offer in Figure 15(b) a complementary visualization to the bubble plot of Figure 2. Specifically Figure 15(b) illustrates average results as a *parallel coordinate* plot, where the two main metrics of interest are reported on a pair of vertical axes; namely, the bufferbloat intensity  $E[Q]/Q_{\max}$  (left y-axis) and TCP% share (right y-axis). Each (LPCC, AQM) pair is represented as a line in Figure 15(b), and a light-gray strip represents the ideal case where the queue is short but the relative levels of priorities are unaltered. For instance, we see that the three (LPCC,

DropTail) combinations appear as horizontal lines on the top of the figure, since under DropTail bufferbloat has maximal intensity and TCP% monopolizes the bottleneck (the order of the curves reflect the order of priority in Carofiglio et al. [2010a]). Conversely, all (LPCC,AQM) combinations excluding DropTail are very close, as AQM implies low bufferbloat but jeopardizes CC priorities (specifically, all TCP and LPCC flows have roughly the same priority). Finally, we see that our proposed CBQ solution falls precisely in the desired regime. Additionally, we see that the reprioritization issue holds for several LPCC and AQM combination: Similarly, we see that our proposed solution holds for several LPCC algorithms (LEDBAT, NICE, and TCP-LP). As such, while the model we present in this article is limited to the (LEDBAT,RED) case, however, the insights we gather are much broader.

We also make ns2 scripts to gather and reproduce these results available to the community.<sup>10</sup> It is also worth stressing that the solution should be fairly easily deployable in practice. Indeed, the firmware governing home-routers and WiFi APs is generally based on some variants of the Linux kernel, possibly open source, as in the OpenWrt or CeroWrt cases. Now, the Linux traffic control (tc) suite supports the CBQ queuing discipline (CBQ qdisc) that offers the very same WRR shaping and fine-grained prioritization capabilities we used to simulate the proof-of-concept solution we propose. Alternatively, we point out that Linux tc also supports priority queuing (PRIO qdisc), that is, non-shaping containers for a configurable number of classes that are dequeued in order. This simpler solution allows for easy prioritization of traffic, where lower classes are only able to send if higher ones have no packets available (but would not allow for a finer tuning as opposed to our CBQ-based proposal, which retains freedom to guarantee a minimum amount of bandwidth to LPCC flows if this is deemed important).

Overall, our proposed solution is simple, as it requires a minimum amount of information exchange precisely at the interface between layers, with the upper layers requiring a marking in the lower-layer packet header. This also means that evolution of protocols at each layer can still happen independently, without requiring a heavy cross-layer or hybrid solution. This simplicity is, we believe, the root of its appeal, as this also increases the chances of its deployment, making it of high practical relevance for the problem under consideration.

## 7. CONCLUSIONS

In this work, we analyzed the interdependency phenomenon that occurs when heterogeneous protocols, namely best-effort and LPCC protocols, share a bottleneck governed by an AQM algorithm.

In previous works we have shown by means of simulations and Internet experiments [Gong et al. 2014] that a negative interplay exists: In particular, the relative level of priority of the congestion control protocols is reset, a phenomenon that we call *reprioritization*, meaning that the protocols compete on a roughly equal basis.

With the purpose of analyzing this phenomenon, we have proposed a mathematical model that captures the dynamics of an overall system composed of (i) TCP flows, which are representative of best-effort congestion control algorithm; (ii) LEDBAT flows, which is a prominent example of LPCC algorithms; and (iii) an AQM control algorithm, namely RED, that is executed at the bottleneck queue.

By analyzing dynamical properties of the system around its equilibrium points, we have been able to provide an explanation to the reprioritization phenomenon and confirmed the generality of this issue. In fact, even though we provide a sufficient condition that allows us to independently tune the LEDBAT parameter  $\tau$  to avoid

<sup>10</sup><http://www.telecom-paristech.fr/~drossi/dataset/ledbat+aqm/PoC-Solution.tgz>.



reprioritization regardless of the AQM employed, we show that such a condition is of scarce practical interest due to the difficulty of accurately measuring the delays at the end systems. Moreover, model predictions have been validated against ns2 simulations when RED is used as AQM algorithm.

Due to the increasing deployment of both LPCC and AQM techniques that today are employed to fight bufferbloat, the problem discussed in this article may be of significant practical relevance. As we believe that it may be desirable for end-users (or end-user applications) to autonomously and coarsely set their relative level of priorities, we have proposed simple yet effective system-level design and practices that solve the issue that we have analyzed in this work.

## APPENDIX

### PROOF OF PROPOSITION 3.1.

PROOF. We denote with  $(w, z, q)$  the equilibrium components of the steady state. By imposing  $dW/dt = 0$  in Equation (5) we readily obtain the following:

$$w = \sqrt{\frac{2}{p}}.$$

It has to be noted that since  $p \in [0, 1]$ , the steady-state value of the TCP window is lower bounded by  $\sqrt{2}$ . By imposing  $dZ/dt = 0$  in Equation (6) we get:

$$\frac{1}{\tau R}(\tau - q/C) - \frac{z^2}{2R}p = 0,$$

which gives:

$$z = \begin{cases} w\sqrt{1 - \frac{q}{\tau C}} & 0 \leq q \leq \tau C \\ 0 & q > \tau C. \end{cases} \quad (28)$$

Let us begin by focusing on the case  $0 \leq q \leq \tau C$ . By letting  $\dot{q} = 0$  in Equation (10) we obtain the following:

$$Nw + Nw\sqrt{1 - \frac{q}{\tau C}} = TC + q. \quad (29)$$

We now make the change of variables:

$$x(q) = \sqrt{1 - \frac{q}{\tau C}}, \quad (30)$$

which is a monotonically decreasing mapping from the set  $[0, \tau C]$  to the set  $[0, 1]$ . By plugging Equation (30) into Equation (29) we get:

$$x^2 + \frac{Nw}{\tau C}x + \frac{Nw}{\tau C} - \frac{T}{\tau} - 1 = 0. \quad (31)$$

By solving Equation (31) we get a unique positive solution:

$$x = \frac{1}{2} \left( \sqrt{\left(\frac{Nw}{\tau C} - 2\right)^2 + 4\frac{T}{\tau} - \frac{Nw}{\tau C}} - \frac{Nw}{\tau C} \right). \quad (32)$$

The solution (Equation (32)) is meaningful only if it belongs to the domain  $[0, 1]$ . Let us start by looking at the upper bound of the solution by plugging  $x = 1$  into Equation (32),



which corresponds to the case  $q = 0$ :

$$\sqrt{\left(\frac{Nw}{\tau C} - 2\right)^2 + 4\frac{T}{\tau} - \frac{Nw}{\tau C}} = 2. \quad (33)$$

After a little algebra, we obtain:

$$w_{min} = \frac{TC}{2N}. \quad (34)$$

To get a physical interpretation of  $w_{min}$ , we have to consider that for this value of  $w$  the queue is zero and  $z(w_{min}) = w$ , that is, the  $N$  TCP and  $N$  LEDBAT flow obtain exactly the same per flow rate equal to  $w_{min}/T$ . It has also be noticed that the sum of the rates of the LEDBAT flows and TCP flows is exactly equal to the capacity of the link:

$$N\frac{w_{min}}{T + \frac{q}{C}} + N\frac{w_{min}}{T + \frac{q}{C}} = \frac{2N}{T}w_{min} = C. \quad (35)$$

Let us now consider the other extreme case  $x = 0$  corresponding to  $q = \tau C$ . In this case, by substituting  $x = 0$  into Equation (32), we get the following:

$$w_{max} = \frac{C}{N}(T + \tau). \quad (36)$$

Thus, we conclude that the interval  $0 < q < \tau C$  maps to the interval  $TC/(2N) < w < C(T + \tau)/N$ . By considering Equations (32) and (30), we obtain the equilibrium for the queue when  $TC/(2N) < w < C(T + \tau)/N$  as follows:

$$q = \tau C \left[ 1 - \frac{1}{4} \left( \sqrt{\left(\frac{Nw}{\tau C} - 2\right)^2 + 4\frac{T}{\tau} - \frac{Nw}{\tau C}} \right)^2 \right]. \quad (37)$$

It is important to notice that  $z = 0$  when  $w = w_{max}$ , meaning that in this case the whole link capacity is allocated to the  $N$  TCP flows. Finally, by plugging Equation (37) into Equation (28), we get the LEDBAT window size at steady state when  $TC/(2N) < w < C(T + \tau)/N$ :

$$z = \frac{w}{2} \left[ \sqrt{\left(\frac{Nw}{\tau C} - 2\right)^2 + 4\frac{T}{\tau} - \frac{Nw}{\tau C}} \right]. \quad (38)$$

Let us now consider the case  $q \geq \tau C$  that occurs when  $w \geq C(T + \tau)/N$ . In this case,  $z = 0$  and, as a consequence,  $q = Nw - TC > 0$ .

To conclude the proof, we have to consider the case  $w < w_{min} = TC/(2N)$ . It is easy to check that  $q = 0$  and thus by plugging this value into Equation (28) we obtain  $z = w$ .  $\square$

## ACKNOWLEDGMENTS

This work was carried out at LINC <http://www.lincs.fr>. Any opinion, findings, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the European Union, or of any of its members, and the partners of the Chair.

## REFERENCES

- J. Ahn, P. Danzig, Z. Liu, and L. Yan. 1995. Evaluation of TCP vegas: Emulation and experiment. In *ACM SIGCOMM Computer Communication Review*, Vol. 25. ACM, New York, NY, 185–195.
- E. Alessio, M. Garetto, R. Lo Cigno, Michela Meo, and M. Ajmone Marsan. 2001. Analytical estimation of completion times of mixed NewReno and Tahoe TCP connections over single and multiple bottleneck networks. In *IEEE GLOBECOM*, Vol. 3. 1788–1793.

- Sanjeeva Athuraliya, S. H. Low, V. H. Li, and Qinghe Yin. 2001. REM: Active queue management. *Network, IEEE* 15, 3 (May 2001), 48–53. DOI: <http://dx.doi.org/10.1109/65.923940>
- Nabil Benameur, Fabrice Guillemin, and Luca Muscariello. 2013. Latency reduction in home access gateways with shortest queue first. In *Proc. ISOC Workshop on Reducing Internet Latency*.
- Maxime Bizon. 2005. *[Nouveaux Firmwares Freebox V3 & V4] ADUF - Historique Firmware, 01/07/2005*. Technical Report. Blog post, available online at <http://88.191.250.12/viewtopic.php?t=164746&view=previous>.
- G. Carofiglio, L. Muscariello, D. Rossi, and C. Testa. 2010a. A hands-on assessment of transport protocols with lower than best effort priority. In *IEEE LCN*.
- G. Carofiglio, L. Muscariello, D. Rossi, C. Testa, and S. Valenti. 2013. Rethinking low extra delay background transport protocols. *Elsevier Computer Networks* (2013), 1838–1852.
- G. Carofiglio, L. Muscariello, D. Rossi, and S. Valenti. 2010b. The quest for LEDBAT fairness. In *IEEE GLOBECOM*. 1–6.
- Renato Lo Cigno and Mario Gerla. 1999. Modeling window based congestion control protocols with many flows. *Perform. Eval.* 36 (1999), 289–306.
- B. Cohen. 2011. How has BitTorrent as a protocol evolved over time. <http://www.quora.com/BitTorrent-protocol-company>. (2011).
- B. Cohen and A. Norberg. 2010. Correcting for clock drift in uTP and LEDBAT. In *9th USENIX International Workshop on Peer-to-Peer Systems (IPTPS'10)*.
- J. Crowcroft, I. Wakeman, Z. Wang, and D. Sirovica. 1992. Is layering, harmful? *IEEE Network* 6, 1 (Jan 1992), 20–24.
- Luca De Cicco, Saverio Mascolo, and Silviu-Iulian Niculescu. 2011. Robust stability analysis of Smith predictor-based congestion control algorithms for computer networks. *Automatica* 47, 8 (2011), 1685–1692.
- A. Demers, S. Keshav, and S. Shenker. 1989. Analysis and simulation of a fair queueing algorithm. In *Symposium Proceedings on Communications Architectures & Protocols (SIGCOMM'89)*. ACM, New York, NY, 1–12. DOI: <http://dx.doi.org/10.1145/75246.75248>
- A. Eshete and Y. Jiang. 2011. Approximate fairness through limited flow list. In *Proceedings of the 23rd International Teletraffic Congress (ITC23)*. 198–205.
- A. Eshete, Y. Jiang, and L. Landmark. 2012. Fairness among high speed and traditional TCP under different queue management mechanisms. In *Proceedings of the ACM Asian Internet Engineering Conference (AINTEC)*. ACM, New York, NY, 39–46.
- Sally Floyd. 1991. Connections with multiple congested gateways in packet-switched networks part 1: One-way traffic. *ACM SIGCOMM Comput. Commun. Rev.* 21, 5 (1991), 30–47.
- S. Floyd and V. Jacobson. 1993. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Network.* 1, 4 (1993), 397–413.
- Sally Floyd and Van Jacobson. 1995. Link-sharing and resource management models for packet networks. *IEEE/ACM Trans. Network.* 3, 4 (Aug. 1995), 365–386.
- J. Gettys and K. Nichols. 2012. Bufferbloat: Dark buffers in the internet. *Commun. ACM* 55, 1 (2012), 57–65.
- YiXi Gong, Dario Rossi, and Emilio Leonardi. 2013a. Modeling the interdependency of low-priority congestion control and active queue management. In *The 25th International Teletraffic Congress (ITC25)*.
- Y. Gong, D. Rossi, C. Testa, S. Valenti, and D. Taht. 2013b. Fighting the bufferbloat: On the coexistence of AQM and low priority congestion control. In *IEEE INFOCOM Workshop on Traffic Measurement and Analysis (TMA'13)*. Turin, Italy.
- Y. Gong, D. Rossi, C. Testa, S. Valenti, and M. D. Taht. 2014. Fighting the bufferbloat: On the coexistence of AQM and low priority congestion control. *Comput. Network.* 65, 0 (2014), 255–267.
- Luigi A. Grieco and Saverio Mascolo. 2004. Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control. *ACM SIGCOMM Comput. Commun. Rev.* 34, 2 (April 2004), 25–38.
- Brian D. Hassard, Nicholas D. Kazarinoff, and Yieh-Hei Wan. 1981. *Theory and Applications of Hopf Bifurcation*. Vol. 41. CUP Archive.
- C. V. Hollot, V. Misra, D. Towsley, and W. Gong. 2001. A control theoretic analysis of RED. In *IEEE INFOCOM'01*, Vol. 3. 1510–1519.
- Gongbing Hong, James Martin, and James M. Westall. 2015. On fairness and application performance of active queue management in broadband cable networks. *Comput. Networks* 91, C (Nov. 2015), 390–406. DOI: <http://dx.doi.org/10.1016/j.comnet.2015.08.018>

- H. Jiang, Y. Wang, K. Lee, and I. Rhee. 2012. Tackling bufferbloat in 3G/4G networks. In *ACM IMC*.
- P. Key, L. Massoulié, and B. Wang. 2004. Emulating low-priority transport at the application layer: A background transfer service. In *ACM SIGMETRICS*. ACM, New York, NY.
- C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson. 2010. Netalyzer: Illuminating the edge network. In *ACM SIGCOMM Internet Measurement Conference (IMC'10)*. 246–259.
- Nicolas Kuhn, Emmanuel Lochin, and Olivier Mehani. 2014. Revisiting old friends: Is CoDel really achieving what RED cannot? In *Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop*. ACM, 3–8.
- A. Kuzmanovic and E. W. Knightly. 2003. TCP-LP: A distributed algorithm for low priority data transfer. In *IEEE INFOCOM*.
- David Lapsley and Steven Low. 1999. Random early marking for internet congestion control. In *Global Telecommunications Conference, 1999. GLOBECOM'99*, Vol. 3. IEEE, 1747–1752.
- S. Liu, M. Vojnovic, and D. Gunawardena. 2006. 4cp: Competitive and considerate congestion control protocol. In *ACM SIGCOMM*.
- Y. Liu, F. Lo Presti, V. Misra, D. Towsley, and Y. Gu. 2003. Fluid models and solutions for large-scale IP networks. In *ACM SIGMETRICS*, Vol. 31. ACM, New York, NY, 91–101.
- M. Ajmone Marsan, Michele Garetto, Paolo Giaccone, Emilio Leonardi, Enrico Schiattarella, and A. Tarello. 2005. Using partial differential equations to model TCP mice and elephants in large IP networks. *IEEE/ACM Transactions on Networking*, 13, 6 (2005), 1289–1301.
- S. Mascolo. 1999. Congestion control in high-speed communication networks using the Smith principle. *Special Issue on "Control methods for communication networks" Automatica* 35, 12 (1999), 1921–1935.
- Paul McKenney. 1990. Stochastic fairness queueing. In *IEEE INFOCOM*.
- Wim Michiels. 2011. Spectrum-based stability analysis and stabilisation of systems described by delay differential algebraic equations. *IET Control Theor. Appl.* 5, 16 (2011), 1829–1842.
- Wim Michiels, D. Melchor-Aguilar, and S.-I. Niculescu. 2006. Stability analysis of some classes of TCP/AQM networks. *Int. J. Control* 79, 9 (2006), 1136–1144.
- V. Misra, W. Gong, and D. Towsley. 2000. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. *ACM SIGCOMM* 30, 4 (2000), 151–160.
- John Nagle. 1985. On packet switches with infinite storage. *RFC 970, IETF* (1985).
- J. Nagle. 1987. On packet switches with infinite storage. *IEEE Trans. Commun.* 35, 4 (Apr 1987), 435–438. DOI: <http://dx.doi.org/10.1109/TCOM.1987.1096782>
- K. Nichols and V. Jacobson. 2012. Controlling queue delay. *Commun. ACM* 55, 7 (July 2012), 42–50.
- S.-I. Niculescu and W. Michiels. 2007. *Stability and Stabilization of Time-Delay Systems. An Eigenvalue-Based Approach*. SIAM: Philadelphia, PA.
- Rong Pan, Preethi Natarajan, Chiara Piglion, Mythili Suryanarayana Prabhu, Vijay Subramanian, Fred Baker, and Bill VerSteeg. 2013. PIE: A lightweight control scheme to address the bufferbloat problem. In *Proc. of IEEE HPSR'13*.
- R. Pan, B. Prabhakar, and K. Psounis. 2000. Choke - A stateless active queue management scheme for approximating fair bandwidth allocation. In *IEEE INFOCOM*.
- Abhay K. Parekh and Robert G. Gallager. 1994. A generalized processor sharing approach to flow control in integrated services networks: The multiple node case. *IEEE/ACM Trans. Netw.* 2, 2 (April 1994), 137–150. DOI: <http://dx.doi.org/10.1109/90.298432>
- Maxim Podlesny and Sergey Gorinsky. 2011. Leveraging the rate-delay trade-off for service differentiation in multi-provider networks. *J. Select. Areas Commun.* 29, 5 (2011), 997–1008.
- D. Rossi, C. Testa, and S. Valenti. 2010a. Yes, we LEDBAT: Playing with the new BitTorrent congestion control algorithm. In *PAM*.
- D. Rossi, C. Testa, S. Valenti, and L. Muscariello. 2010b. LEDBAT: The new BitTorrent congestion control protocol. In *IEEE ICCCN*.
- Sandvine. 2014. Global Internet Phenomena Report (1H 2014). <https://www.sandvine.com/downloads/general/global-internet-phenomena/2014/1h-2014-global-internet-phenomena-report.pdf>. (2014).
- J. Schneider, J. Wagner, R. Winter, and H. J. Kolbe. 2010. Out of my way – Evaluating low extra delay background transport in an ADSL access network. In *ITC22*.
- S. Shalunov, G. Hazel, J. Iyengar, and M. Kuehlew. 2012. Low Extra Delay Background Transport (LEDBAT). IETF RFC 6817. (2012).
- M. Shreedhar and George Varghese. 1995. Efficient fair queueing using deficit round robin. In *ACM SIGCOMM*.

Si Quoc Viet Trang, N. Kuhn, E. Lochin, C. Baudoin, E. Dubois, and P. Gelard. 2014. On the existence of optimal LEDBAT parameters. In *IEEE International Conference on Communications (ICC)*. 1216–1221. DOI : <http://dx.doi.org/10.1109/ICC.2014.6883487>

A. Venkataramani, R. Kokku, and M. Dahlin. 2002. TCP nice: A mechanism for background transfers. In *USENIX OSDI*.

G. White. 2014. *Active Queue Management in DOCSIS 3.x Cable Modems*. Technical Report. CableLabs.

Received April 2015; revised March 2016; accepted May 2016