Engineering Applications of Artificial Intelligence 24 (2011) 77–86

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Managing Internet routers congested links with a Kohonen-RED queue ☆

Emmanuel Lochin [a,]*, Bruno Talavera [b]

[a] CNRS-LAAS - ISAE, Université de Toulouse, ISAE, 1, Place Emile Blouin, BP 75064, 31033 TOULOUSE Cedex 5, France
[b] Université Pierre et Marie Curie, Polytech' Paris-UPMC, France

A R T I C L E I N F O

Article history:
Received 23 November 2009
Received in revised form
28 July 2010
Accepted 15 October 2010
Available online 3 November 2010

Keywords:
Active queue management
Computer networks
Congestion control
Kohonen neural networks
TCP

A B S T R A C T

The behaviour of the TCP AIMD algorithm is known to cause queue length oscillations when congestion occurs at a router output link. Indeed, due to these queueing variations, end-to-end applications experience large delay jitter. Many studies have proposed efficient active queue management (AQM) mechanisms in order to reduce queue oscillations and stabilize the queue length. These AQM attempt to improve the random early detection (RED) model. Unfortunately, these enhancements do not react in a similar manner for various network conditions and are strongly sensitive to their initial setting parameters. Although this paper proposes a solution to overcome the difficulties of configuring the RED parameters by using a Kohonen neural network model; another goal of this study is to investigate whether cognitive intelligence could be placed in the core network to solve such stability problem. In our context, we use results from the neural network area to demonstrate that our proposal, named Kohonen-RED (KRED), enables a stable queue length without complex parameters setting or passive measurements to obtain a correct configuration.

1. Introduction

TCP, the predominant Internet transport protocol, has the capability to adapt its sending throughput to the changing bandwidth available following the principle described in Jacobson (1988). TCP considers a loss of packets as a congestion inside the network and reacts to this congestion signal by halving its current emission window of packets. A congestion event (or loss event) corresponds to one or several losses occurring in one TCP window during one current round trip time period. When a drop tail queue overflows, several TCP sources can observe a bunch of losses which can lead to a decrease of the throughput involving a brutal decrease of the buffer occupancy. Then, as TCP always operates opportunistically, it restarts to increase exponentially (or linearly following the current mode is operating) in order to occupy the most capacity possible. As a result, this effect results in an accordion phenomenon where queues are constantly oscillating. This behaviour is harmful for the end-hosts which observe a constant oscillation of their current throughput that can be problematic for applications which need a relatively stable rate over the time.

More than 10 years ago, the random early detection (RED) was proposed to avoid congested Internet links (Floyd and Jacobson, 1993). The goal was to replace the current deployed drop tail queues known to provoke large queue oscillations when router's buffers overflow. The main idea of the RED algorithm is to drop packets before the queue is full. As a consequence, when a TCP source gets such preventive drops, it decreases the emitted throughput according to the AIMD (additive increase multiplicative decrease) algorithm inherent to the TCP protocol. RED drops packets with an increasing probability (max_p) when the occupancy of the queue lies between two thresholds (min_{th}, max_{th}). The goal of RED is to maintain a small buffer occupancy and avoid casual bursts of packet losses.

In Ranjan et al. (2004), the authors illustrate that the instantaneous queue oscillations lead the system to a chaotic state. They show that the root of the problem comes from the estimated average queue size which is computed with inaccurate initial values. Then, during an experiment, the queue dynamics slide from a stable fixed point to an oscillatory behaviour and finally to a chaotic state. As a direct consequence, the network operators cannot provide any Quality of Service (QoS) guarantee to their customers as they would need to continuously adapt their parameters for different traffic conditions.

Several other studies emphasized these issues, and in particular the authors in May et al. (1999) and Ziegler et al. (2001) have weighted up the disadvantages for deploying such a mechanism over the Internet. In certain cases, increasing the number of dropped packets can have unexpected effects on the overall performance (Ziegler et al., 2001). This has motivated the use of preventive marking instead of preventive dropping with the use of the explicit congestion notification (ECN) flag (Ramakrishnan et al., 2001) of IP packets. In this case, instead of dropping packets, the RED queue marks the IP packet's ECN flag to notify senders that they are crossing a congested link and that they should decrease

☆ Part of the results have been presented at IEEE ICC 2008.
* Corresponding author.
E-mail addresses: emmanuel.lochin@isae.fr (E. Lochin), bruno.talavera@upmc.fr (B. Talavera).

0952-1976/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2010.10.012

their sending rate. Although this flag is currently implemented both in end-hosts (GNU/Linux, Mac OSX and Windows Vista and newer) and inside the core network (Cisco IOS implements a RED/ECN variant called WRED/ECN), ECN still remains disabled by default for all these systems and the following study (Medina et al., 2005) published in 2004 precises that ECN is only used by 2.1% of computers.

Despite of all these issues, the RED algorithm is recommended by the Internet Engineering Task Force (IETF) with the expectation that the providers make their own effort to select suitable RED control parameters for their network (Braden et al., 1998). Unfortunately, some past work have already suggested that RED is fundamentally hard to tune (Low et al., 2003) while others claim that tuning its parameters is an inexact science (May et al., 1999).

All these configuration problems partly explain the reason why network operators do not enable this algorithm. Thus, it seems obvious and necessary to find a method to easily and automatically tune the RED parameters.

Several studies attempted to enhance the basic RED algorithm (Feng et al., 1999; Floyd et al., 2001; Hollot et al., 2001; Athuraliya et al., 2001; Low et al., 2003; Suthaharan, 2007; Kim and Yeom, 2008). As a non-exhaustive list, fair RED (FRED) (Feng et al., 1999) and adaptive RED (ARED) (Floyd et al., 2001) introduced the notion of adaptive AQM. These adaptive strategies recompute the max_p probability value following an AIMD algorithm. However, the parameters that weight this AIMD process still remain difficult to estimate and far from being generic as we will see in our simulation. In Low et al. (2003), the authors show that RED parameters can be tuned to improve stability, but only at the cost of large queues even when they are dynamically adjusted. In Suthaharan (2007), the author introduces an interesting approach where the algorithm dynamically adjusts the moving exponential weighted computation parameter which is normally static, in order to impact on the accuracy of the rate estimation. Even if other different queueing approaches have been proposed to improve the efficiency of RED-like algorithms in various network conditions, the parameters used to set these new AQM are sometimes more complex to determine than RED ones. In particular, this is the case for the PI controller (Hollot et al., 2001). Nowadays, general parameters able to stabilize the queue do not yet exist whatever the AQM used and we could discuss whether the problem is in fact solvable.

Although the validity of RED concept is still debated, we claim that the parameters' settings are one of the main barrier to its acceptance and that this problem is a perfect candidate for the neural networking area. In this paper, we seek at evaluating whether neural network theory can help by simply acting on the standard RED parameters. As a result, this paper does not attempt to extend the prolific models previously proposed. We do not attempt to design another queueing mechanism or propose to enhance the core mechanism itself. We only focus on the optimal estimation of the probability parameter denoted max_p. Thus, this paper aims at illustrating the impact of the role of learning mechanisms on core network Internet problems with similar motivation than the one presented in Beverly and Sollins (2007). The purpose is to illustrate the perfect capability of a class of neural networks to solve this stability issue without requesting complex tuning from network engineers.

This paper is structured as follows. Section 2 presents the motivation of this work. Section 3 gives pointers related to the implementation of the core mechanism. Then, Section 4 evaluates the proposal and finally Section 6 gives the perspectives of this work.

2. Motivation of using a Kohonen neural network

Kohonen networks (Kohonen, 2001) are a class of neural networks known to solve the pole balancing problem (Makarovic, 1991). Pole balancing is a control benchmark historically used in mechanical engineering. It involves a pole placed on a cart via a joint allowing movement along a single axis. The cart is able to move along a track with a fixed length as represented in Fig. 2(a). The aim of the problem is to keep this pole balanced by applying forces to the cart.

The main idea of our contribution is based on the analogy existing between this balancing problem and the RED queueing problem. In RED, we can compare the pole balancing to the evolution of the queue occupancy which oscillates between both thresholds (min_{th}, max_{th}).

As shown in Fig. 1, the probability to drop (or to mark) packets increases when the buffer occupancy increases following the probability scheme configured (i.e. following both thresholds set and the slope of the probability curve). The physical forces resulting on the pole have a similar role to the packets arrival rate in the queue. Fig. 2 illustrates this view.

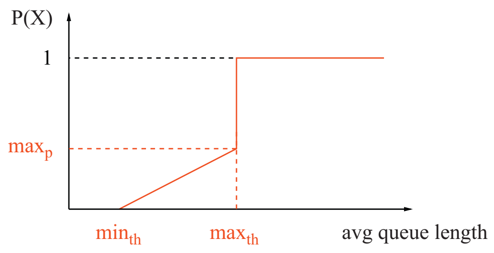

Fig. 1. The random early detection (RED) queue management scheme.

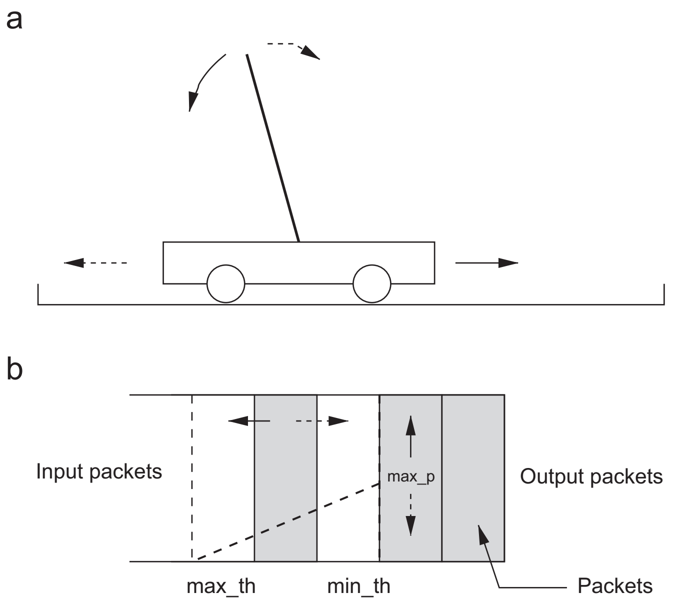

Fig. 2. Analogy between the single pole balancing problem and RED AQM: (a) pole balancing and (b) adaptive RED.

Table 1
Input and output values used.

	Pole	RED
input_value[1]	Previous position	Previous queue length
input_value[2]	New position	Current queue length
output_value[1]	Force to apply in Newton	max_p