Computer Communications 35 (2012) 2281–2293

Contents lists available at SciVerse ScienceDirect

Computer Communications

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

# IAPI: An intelligent adaptive PI active queue management scheme

Jinsheng Sun [a], Sammy Chan [b,*], Moshe Zukerman [b]

[a] Department of Automation, Nanjing University of Science and Technology, Nanjing 210094, PR China
[b] Department of Electronic Engineering, City University of Hong Kong, Hong Kong SAR, PR China

ARTICLE INFO

*Article history:*
Received 20 December 2011
Received in revised form 6 June 2012
Accepted 23 July 2012
Available online 1 August 2012

*Keywords:*
Congestion control
TCP
Active queue management
Adaptive
Intelligent

ABSTRACT

Internet traffic is bursty and unpredictable. It is therefore important to control congestion to protect users from traffic surges in a way that is efficient, stable and robust. Research has shown that active queue management (AQM) has the potential to control congestion such that these requirements are met. However, despite much work and many published proposals of AQM algorithms, a satisfactory solution is still unavailable. In this paper, an intelligent adaptive PI controller (IAPI) is proposed as a new AQM scheme, and its design details are presented. Setting of parameter values for IAPI is provided based on previous studies on PI and additional empirical studies. The performance of IAPI is evaluated by simulations and the results demonstrate that it is stable and robust under various scenarios including cases involving different types of background traffic (UDP and HTTP) and a case involving a multiple bottlenecks. Its superiority over other AQMs is also demonstrated.

© 2012 Elsevier B.V. All rights reserved.

## 1. Introduction

Congestion control is important to the stability, performance, and robustness of the Internet. It has been realized that TCP network congestion control that relies only on actual packet loss due to buffer overflow to adapt the sending rate to congestion level is inefficient. Consequently, active queue management (AQM) schemes, which signal congestion early before the buffer is completely full, have been proposed to complement the TCP network congestion control. AQM is a router based congestion control mechanism that uses various methods to convey congestion notification to the end-hosts, enabling the sources to reduce their sending rates prior to buffer overflow.

AQM schemes are driven by the following goals: low buffer occupancy resulting in small queueing delays, low queue length jitter (stability), low packets losses and high link utilization. Preferably, an AQM scheme would make a good choice of packet dropping/marking at each congestion stage (measured by, e.g., queue length or link occupancy) so that the above mentioned goals are realized. The second desirable feature of a good AQM scheme is its robustness. A robust AQM scheme would require little tuning by a network operator and would remain inherently stable to traffic fluctuations [15].

In the past two decades, many new AQM algorithms or enhancements [1,2,6–9,11,12,14,20–22,24,25,32] have been proposed. Although the benefit of AQM feedback seems obvious, its widespread use in the Internet has not materialized because of the difficulty of configuring AQM algorithms in a dynamic networking environment to achieve a stable operation. The main reason is that the network congestion control system is a nonlinear, time-varying, time-delay system. Thus, a fixed-parameters AQM cannot obtain satisfactory performance in all possible situations.

In order to improve the performance of TCP/AQM system, many AQM with adaptive parameters were proposed. In [6], a self-configuring RED mechanism was proposed, which varies the maximum drop probability $max_p$ using two different constant factors based on observed average queue length. Floyd et al. [9] improved this proposal using the so-called Additive Increase Multiplicative Decrease (AIMD) approach. Hao et al. [10] proposed an adaptive PID called improved expert intelligent PID algorithm (IEI-PID), which tunes the PID control parameters according to queue length error. However, this method did not improve the steady-state performance. Sun et al. [23] introduced a new adaptive REM (AREM) algorithm by increasing the control parameter $\alpha$ when queue length error is larger than a preset threshold, and adjusting the drop probability according to instantaneous queue length. Wang et al. [27] proposed a self-tuning priced-based congestion control (SPC) scheme based on REM. SPC uses a PID-type price and its key parameter is adjusted dynamically to optimize the selected criteria for satisfactory performance. Xiong et al. [30] proposed an improved PD [22] called NPD-RED, which calculated the control parameters of PD online on the basis of network parameters. They also extended their method to develop an AQM called self-tuning proportional and integral [29].

* Corresponding author. Tel. +852 34424243.
*E-mail addresses:* jssun67@yahoo.com.cn (J. Sun), eeschan@cityu.edu.hk (S. Chan), m.zu@cityu.edu.hk (M. Zukerman).

0140-3664/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comcom.2012.07.013

This paper presents a new AQM scheme called intelligent adaptive proportional integral controller (IAPI). IAPI adaptively adjusts the integral gain according to the queue length error to achieve faster convergence of queue length, and adjusts the drop probability according to instantaneous queue length to decrease the queue length jitter. We demonstrate by simulations that IAPI performs well independently of traffic loading, round trip propagation delay, and bottleneck capacity. We also demonstrate that IAPI is robust to different background traffic such as UDP traffic and HTTP traffic, and maintains its performance in the presence of multiple bottlenecks.

We first compare IAPI with proportional-integral (PI) AQM [11] to demonstrate the value of its added intelligent adaptive features and demonstrate the superiority of IAPI in achieving faster convergence to queue length target and smaller queue length jitter. Then we compare IAPI with REM [1], PI [11] and Q-SAPI [4] to further demonstrate the advantages of IAPI in lower packet loss and higher link utilization.

The rest of the paper is organized as follows. Section 2 reviews related research work. Section 3 describes IAPI in detail and provides parameter setting for IAPI based on previous studies on PI and additional empirical studies. Section 4 presents simulation results for performance evaluation of IAPI for a single bottleneck network, multiple bottleneck network, and for comparison of IAPI with other AQM schemes. Finally, in Section 5 we present our conclusions.

## 2. Related work

Based on the linearized fluid-based TCP model, Hollot et al. [11,12] developed an AQM based on a PI controller. In order to maintain a stable queue length over a wide range of network dynamics, the parameters of the fixed-gain PI controller are set for the worst case, namely, they are set conservatively. The result is that it cannot achieve both satisfactory transient response and small deviation for steady-state behavior over a wide range of network dynamics. This gives rise to new research on how to introduce adaptivity to further improve the performance of PI controller.

Xu et al. [31] proposed a nonlinear PI AQM algorithm, which uses nonlinear functions of queue length error to adjust the control coefficients. Hong et al. [13] proposed a self-tuning PI controller, that adjusts the controller parameters when gain or phase margins falls outside a specified interval. The method requires knowledge of network parameters, namely, round trip time (RTT) $R_0$, bottleneck link capacity $C$, and number of active long-lived TCP connections $N$, which are unknown in routers and difficult to estimate accurately. Zhang et al. [32] proposed a self-tuning structure to adaptively adjust control parameters according to the estimated $C$ and $\frac{N}{R_0C}$, which is applicable to any AQM algorithm that was parameterizable in terms of link capacity and TCP load. However, as can be seen from the adaptive parameters equations in [32], the $R_0$ value is also required to be estimated. Chang et al. [3] proposed an adaptive PI, that increases the queue length error when it is larger than a certain threshold. Then, they improved this method and proposed another adaptive PI called Q-SAPI, which used nonlinear functions to adjust the control parameters when queue length error is larger than the threshold [4]. Wang et al. [26] proposed to use both average queue length and loss rate to adjust the drop probability, thus increasing the response rate of the PI controller. Chen and Yang [5] proposed to adjust the control parameters using pole-placement method. However, again, calculation of the control parameters requires the knowledge of $N, C$, and $R_0$, some of which are unknown in routers. Qing et al. [17] designed a neuron-based adaptive PI algorithm, which uses a single neuron to adaptively adjust the control parameters.

In summary, though there are many adaptive PI that have been proposed, their performance are unclear. Some of which depend on network parameters ($N, C$ and $R_0$), which may be difficult to estimate, such as $R_0$. Some of them use queue length error to adjust the control parameters, which cannot guarantee both satisfactory transient response and stability over a wide range of network dynamics.

## 3. IAPI AQM

### 3.1. Motivation

Hollot et al. [11,12] proposed to use PI controller as an AQM algorithm, which has the form

$$p(t) = k_pe(t) + k_i\int_0^t e(t), \tag{1}$$

where $p(t)$ is the packet dropping/marking probability,

$$e(t) = q(t) - q_{ref} \tag{2}$$

is queue length error, $q(t)$ is the instantaneous queue length, $q_{ref}$ is the target queue length, $k_p$ and $k_i$ are proportional coefficient and integral coefficient, respectively. In the implementation of the algorithm, drop probability is calculated periodically, namely drop probability is recalculated every fixed-length sampling interval of size $\Delta T$. So, the PI AQM algorithm can be expressed in a discrete form as follows

$$p(k) = p(k-1) + k_p(e(k) - e(k-1)) + k_ie(k), \tag{3}$$

where $k$ is the time index representing the $k$th sampling time.

Ref. [12] has provided a procedure to design the control parameters for PI controller. Using this procedure, we can obtain suitable control parameters provided that we know the network parameters – $C$, $N$ and $R_0$. Normally, for a router, $C$ is known. If $C$ is unknown, it is easy to estimate it by keeping track of the outgoing packets. However, $N$ and $R_0$ are unknown. In order to guarantee the stability of the TCP/AQM system, the PI controller is designed under conservative situation – small $N^-$ and large $R^+$, and a fixed-gain PI controller can maintain the TCP/AQM system stable when $N \geqslant N^-$ and $R_0 \leqslant R^+$ [4]. However, this causes sluggish response when the network traffic conditions change. At the same time, PI has another drawback – its queue length fluctuations in steady state are large.

In order to improve the performance of the PI controller, in this paper, we propose a new method to adaptively adjust the PI controller parameters according to queue length dynamics, so we can guarantee that the TCP/AQM system is stable, has a faster response and small queue length fluctuation in steady state.

### 3.2. Description of IAPI

Our method to improve the performance of the PI controller is based on three ideas. First, we make the response rate of IAPI to be a function of the queue length error $e(k)$. Recall that a key objective of AQM is to control $q(k)$ to be as close to $q_{ref}$ as possible. Therefore, as $e(k)$ increases, IAPI should have a higher response rate so that $q(k)$ approaches $q_{ref}$ faster. By introducing this adaptive response rate, the sluggish response of PI controller caused by conservatively designed parameters is avoided. Since the response rate in the PI controller is determined by $k_i$, we modify the fixed-valued $k_i$ in [12] as follows. Let us define a preset error threshold $e^*$, which represents how much queue length error the system is willing to tolerate. When $|e(k)| > e^*$, a simple adaptive function is to make $k_i$ to be directly proportional to the error magnitude exceeding $e^*$,

$$k_i = k_{i0}\left(1 + \frac{|e(k)| - e^*}{2e^*}\right), \tag{4}$$

where $k_{i0}$ is the basic integral coefficient designed in [12].

Download English Version:

https://daneshyari.com/en/article/448404

Download Persian Version:

https://daneshyari.com/article/448404

Daneshyari.com