Computer Networks 53 (2009) 668–679

Contents lists available at ScienceDirect

Computer Networks

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

Analysis of burstiness monitoring and detection in an adaptive Web system

Katja Gilly [a,*], Salvador Alcaraz [a], Carlos Juiz [b], Ramon Puigjaner [b]

[a] Miguel Hernández University, Departamento de Física y Arquitectura de Computadores, Avda. de la Universidad, 03202 Elche, Spain
[b] University of Balearic Islands, Departament de Ciències Matemàtiques i Informàtica, Carretera de Valldemossa, km 7.5, 07071 Palma de Mallorca, Spain

ARTICLE INFO

Article history:
Received 19 June 2008
Received in revised form 24 October 2008
Accepted 28 October 2008
Available online 21 November 2008

Responsible Editor: J. Neuman de Souza

Keywords:
Internet
Network monitoring
Performance measures
Simulation

ABSTRACT

Due to the heavy tailed pattern of Internet traffic, it is crucial to monitor the incoming arrival rate in a Web system to preserve its performance. In this study, we focus on the arrival rate processing mechanism as part of the design of an adaptive load balancing Web algorithm. The arrival rate is one of the most important metrics to be monitored in a Web site to avoid the possible congestion of Web servers. Six methods are analysed to detect the burstiness of incoming traffic in a Web system. We define six burstiness factors to be individually included in an adaptive load balancing algorithm, which also needs to monitor some Web servers' parameters continuously, such as the arrival rate at the server or its CPU utilization in order to avoid an unexpected overload situation.

We also define adaptive time slot scheduling based on the burstiness factor, which reduces considerably the overhead of the monitoring process by increasing the monitoring frequency when bursty traffic arrives at the system and by decreasing the frequency when no bursts are detected in the arrival rate. Simulation results of the behaviour of the six burstiness factors and adaptive time slot scheduling when sudden changes are detected in the arrival rate are presented and discussed. We have considered a scenario made up of a locally distributed cluster-based Web information system for simulations.

1. Introduction

The fact that Internet traffic flows exhibit heavy tailed probability distributions has been widely discussed in the Internet literature [1,2]. As Web traffic inter-arrival times normally follow a heavy tailed distribution, maintaining a good performance of the Web system is normally more complicated than if this distribution were easily predictable. Hence, it is possible that in a few seconds a Web server that is not overloaded may receive an increase in the number of connections, which produces a situation of congestion [3,4]. This occurs when the server reaches the connection number limit it can handle. Even, without reaching this limit, if the client's request for a connection is accepted, the response time for that request may experience a long delay because of the long queue of requests waiting to be served by the Web system [4]. Internet service providers often offer different Quality of Service (QoS) levels to provide different priority to different users. When the congestion situation is severe, admission policies are normally applied. This leads to the challenge of satisfying the performance requirements for different types of requests at all times.

Our main concern in the design of a load balancing Web system is how to monitor some Web servers' parameters in a very adaptive way in order to reduce the algorithm overhead. Some of the Web servers' parameters likely to be monitored are the arrival rate, the CPU/disk utilization, I/O performance, etc. The performance of the nodes that compound the Web system have to be monitored continuously in order to know their status and make the appropriate decisions in case of overload to avoid a possible congestion situation. This can be done in several ways: (i) each time a request arrives at the front-end of the

* Corresponding author. Address: Department of Physics and Computer Architecture, Miguel Hernández University, Avda. de la Universidad, s/n, 03202 Elche (Alicante) Spain. Tel.: +34 966658565; fax: +34 966658814.
E-mail addresses: katya@umh.es (K. Gilly), salcaraz@umh.es (S. Alcaraz), cjuiz@uib.es (C. Juiz), putxi@uib.es (R. Puigjaner).

1389-1286/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2008.10.026

Web system; (ii) at fixed times by using static time slot scheduling; or (iii) at non-fixed times by using dynamic time slot scheduling. The overhead introduced by option (i) is the biggest because each time a request arrives at the Web system, Web node parameters are monitored. While option (ii) introduces a constant overhead, option (iii) monitors the system at non-fixed intervals, hence, its overhead will depend on the frequency of those intervals. The drawback of defining monitoring in a constant duration interval schedule (option (ii)) is the choice of monitoring time interval. It is very difficult to set a duration interval that fits with all possible Internet arrival rates at the Web system due to its heavy tailed pattern.

We propose using adaptive time slot scheduling (option (iii)) where the frequency of monitoring depends on a burstiness factor that will increase its value when bursty arrivals reach the system, and decrease it, if no burstiness is detected. The adaptive time interval we define depends directly on this burstiness factor. Therefore, the monitoring task's overhead is related to the burstiness of the arrivals in the Web system and time slot scheduling is completely adaptive to the burstiness detected in the arrival rate that reaches the system.

We have included six burstiness factors in a content-aware load balancing model designed with OPNET Modeler [5] to compare the effect of including different burstiness factors in the Web system performance. A previous study of burstiness in a Web system was presented in a conference paper [6]. The load balancing algorithm used is beyond the scope of this article and is fully described in [7].

The following sections of this paper are organised as follows: Section 2 describes related studies on burstiness analysis in Internet traffic. Section 3 details the definition of monitoring slots. The burstiness factors we have considered for our experiments are detailed in Section 4 and the adaptive time slot scheduling mechanism is described in Section 5. Section 6 details the simulation scenario and shows the results obtained. Finally, we discuss some concluding points and the open problems.

2. Related work

In this section, burstiness modelling and detection related research is introduced.

2.1. Burstiness detection based on traffic

The pioneers in modelling burstiness are Wang et al. [8]. They analyse the relation between jitter and burstiness in real-time communications. In this paper, the burstiness detection mechanism is defined for individual packets. The authors define the burstiness of the *m*th packet as a measure of time that expresses the distance between the actual arrival time and the right edge of the *m*th packet arrival interval because they consider the servers usually process packets one by one at a constant rate. An implementation of this mechanism has been defined in our simulation scenario. More details and results are described in the following sections.

Burstiness detection based on the traffic rate and independent of individual packets is defined in other papers [9–12]. Menascé and Almeida [9] are the first to introduce a burstiness factor. They define it as the fraction of time during the time slot arrival rate that exceeds the average arrival. In this case, burstiness monitoring is carried out following fixed time slot scheduling. In Section 4, we describe in detail how this burstiness factor is defined and works and introduce some modifications to it. Baryshnikov et al. [10] study how traffic predictions can be very useful to reduce latencies and performance degradation in Web servers. They use linear extrapolation as a prediction technique and state that this technique for burstiness detection is not a good predictor. In general, however, they conclude that even simple prediction algorithms have a significant prediction power. We also consider their mechanism in the burstiness factors we define in Section 4. In [11], van de Meent et al. detect burstiness through the average traffic rate and the peak rate each second. They define a non-linear relation between these two variables to model the variation in the traffic rate that shows burstiness. Li et al. [12] detect burstiness in their model by defining thresholds. If the arrival rate exceeds the thresholds in a number of successive slots, then sustained burstiness occurs.

2.2. Burstiness detection based on TCP protocol

Burstiness has also been modeled for TCP traffic in other sudies such as like [13,14]. In [13], the authors detect bursts of TCP acknowledgment packet transmissions to increase the size of the congestion window. A burstiness model is defined in [14] that assigns a burstiness value to each TCP packet based on the RTT in order to control the actual sending rate.

2.3. Burstiness detection in databases

With regards to databases, Vlachos et al. [15] detect short-term and long-term bursts in online search queries by comparing the moving average (of 7 days and 30 days for short-term and long-term bursts, respectively) of user demands with its mean value.

2.4. Burstiness detection based on Internet traffic characterisation

Other papers deal with the characteristics of Internet traffic by focusing on the burstiness implicit to it. Sarvotham et al. [16] define an alpha/beta traffic model, considering the traffic bursts as the alfa-traffic and analyse why the bursts occur in network traffic. Lan et al. [17] define a burst as a train of packets with a lower inter-arrival time than a threshold and study the correlations between size, rate and burstiness.

As indicated above, we have included some ideas from previous papers [8–10] in our burstiness definition, which are detailed in Section 4. We have also used the non-linear relation defined in [11] to analyse some of the results obtained in Section 6.