Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Communications 27 (2004) 1851–1867
computer communications
www.elsevier.com/locate/comcom

Proportional loss rate differentiation in a FIFO queue

James Aweya*, Michel Ouellette, Delfin Y. Montuno

Nortel Networks, P.O. Box 3511, Ottawa, Ont. Canada K1Y 4H7

Received 6 October 2003; revised 6 July 2004; accepted 7 July 2004
Available online 29 July 2004

Abstract

The relative differentiated service model is one of several models proposed for service differentiation in networks [IEEE Network Sept/Oct (1999) 26]. In this model, an assurance is given that 'higher classes will be better, or at least no worse than lower classes.' This paper describes a relative loss rate differentiation scheme based on RED. The scheme is used for differentially dropping packets in a FIFO queue during times of congestion. The main idea is, if packet losses are unavoidable in the FIFO queuing system, then they should be distributed among the different service classes in the queue in inverse proportion to the service price or weight assigned to each class. The simulation studies using TCP traffic show that the scheme is very effective in ensuring relative loss rate differentiation between service classes.

Keywords: Relative differentiated services; Relative loss rate differentiation; Congestion control; Active queue management; TCP congestion control

1. Introduction

In the relative differentiated service model, all traffic is grouped into N classes of service. For each class i, the service provided to class i (in terms of local (per-hop) performance measures of queuing delays and packet losses) will be better (or at least no worse) than the service provided to class $(i-1)$, where $1<i\leq N$ [1–4]. The 'or at least no worse' clause is included for levels of low network activity where all classes experience the same quality of service. The level of service in a class is relative to the other classes in the network and is not an absolute guarantee (in terms of end-to-end delay bound, bandwidth, etc.) since there is no admission control and resource reservation.

The proportional differentiation model [1–4] provides a way to control the quality spacing between classes locally at each hop, independent of the class loads. In this model, certain forwarding performance metrics are ratioed proportionally to the class differentiation parameters that the network operator chooses. Queuing delay and packet loss are two performance measures that can be used for proportional service differentiation.

Let $\bar{l}_i$ be the average loss rate for class i. Using this measure, the proportional differentiation model requires that the class loss rates be spaced as

$$\frac{\bar{l}_i}{\bar{l}_j}=\frac{\sigma_i}{\sigma_j},\quad 1\leq i,\ j\leq N. \tag{1}$$

The parameters σ_i are the loss rate differentiation parameters and are ordered as $\sigma_1>\sigma_2>\ldots\sigma_N>0$. In this particular definition, higher classes have better performance in terms of loss rates. The loss rate differentiation parameters $\sigma_1>\sigma_2>\ldots\sigma_N>0$ provide the network operator with tuning knobs for adjusting the quality spacing between classes, independent of the class loads.

Unlike other relative loss rate differentiation mechanisms [1–4], this paper describes a relative loss rate differentiation mechanism designed around an active queue management (AQM) scheme. AQM schemes such as random early detection (RED) [5,6] provide network devices with some means to detect incipient congestion early and to convey congestion notification to the end-systems, allowing them to reduce their transmission rates before queues in the network

* Corresponding author. Tel.: +1-613-763-6491; fax: +1-613-765-3370.
E-mail addresses: aweyaj@nortelnetworks.com (J. Aweya), ouellett@nortelnetworks.com (M. Ouellette), delfin@nortelnetworks.com (D.Y. Montuno).

0140-3664/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2004.07.003

overflow and packets are dropped. The basic RED scheme (and its newer variants) maintains an average of the queue length which it uses together with a number of queue thresholds to detect congestion. RED schemes drop incoming packets in a random probabilistic manner where the probability is a function of recent buffer fill history. The objective is to provide a more equitable distribution of packet loss, avoid the synchronization of flows [5,6], and at the same time improve the utilization of the network.

The mechanism described in this paper is for relative loss rate differentiation in a FIFO queue only and does not cover delay differentiation or the coupling of the two. FIFO queuing has always been attractive because it is very simple to implement (packets are stored and served in the order in which they arrive). The limitation, however, is FIFO queuing provides little protection from high-bandwidth flows that consume a lot of bandwidth at the expense of other flows at the queue. A workaround to this problem is to rate-limit flows (e.g. using token buckets) at the network edge before the FIFO queuing point.

2. Proportional loss rate differentiation mechanism

Here, we describe the proportional loss rate (PLR) differentiation mechanism based on random early detection. An AQM module would monitor the aggregate load or queue size for the N classes in the FIFO queue, and then based on a queue control law, would determine a packet drop probability p_{d}. This probability would then be used to drop packets to provide feedback to the TCP data sources to throttle their transmission rates. The idea here is, if packet losses are avoidable in the FIFO queue, then they should be distributed among the different service classes in proportion to their loss rate differentiation parameters σ_i, $i=1,2,\ldots,N$. That is, each class i should have an average drop rate of

$$p_{\mathrm{d},i} = \frac{\sigma_i}{\sum_{j=1}^{N}\sigma_j} \cdot p_{\mathrm{d}}, \quad i = 1,2,\ldots,N, \tag{2}$$

which is equivalent to having

$$\frac{p_{\mathrm{d},i}}{p_{\mathrm{d},j}} = \frac{\sigma_i}{\sigma_j}, \quad 1 \le i,\ j \le N. \tag{3}$$

We assume that a single shared buffer is used to store the packets of N classes with different loss rates. We assume also that each class i, ($i=1,2,\ldots,N$) has a different service price (or weight) g_i, $i=1,2,\ldots,N$, determined based on pricing or other network policies.

Now, considering a single bottleneck link, the throughput of a TCP connection belonging to class i can be approximated by [7]

$$B_i = \frac{0.93 S_i}{\bar{d}_i\sqrt{p_{\mathrm{d},i}}}, \tag{4}$$

where S_i is the connection's maximum segment size (MSS), and $\bar{d}_i$ is the round-trip delay (which consists of a fixed propagation delay and average queuing delay) in the connection's path. The ratio of the throughputs of connections in classes i and j is given as

$$\frac{B_i}{B_j} = \frac{\bar{d}_j}{\bar{d}_i}\sqrt{\frac{p_{\mathrm{d},j}}{p_{\mathrm{d},i}}} = \frac{\bar{d}_j}{\bar{d}_i}\sqrt{\frac{\sigma_j}{\sigma_i}}. \tag{5}$$

We see that the ratio of the TCP throughputs between classes are related in an inverse square-root sense to the PLR differentiation parameters. This relationship shows that the TCP throughput ratios between classes can be controlled by appropriately manipulating the loss rates and/or queuing delays (since propagation delay is fixed) in each class. The mechanism described in this paper manipulates only the loss rates since we are operating in FIFO queue. We get the following expression when the connections in the classes have equal round-trip delays

$$\frac{B_i}{B_j} = \sqrt{\frac{p_{\mathrm{d},j}}{p_{\mathrm{d},i}}} = \sqrt{\frac{\sigma_j}{\sigma_i}}. \tag{6}$$

2.1. Packet drop probability computations

In this section, we give a brief overview of the packet drop probability computations required for the PLR differentiation scheme. The random packet drop algorithm [8,9] for AQM is a mechanism for congestion control in a network device. The algorithm controls congestion by randomly dropping packets with a probability p_d, which constitutes a signal to applications (TCP sources) to reduce their sending rate. The algorithm takes its decision of dropping or accepting an incoming packet so that the queue occupancy level is kept at a given target level, thereby eliminating buffer underflow and overflow as much as possible.

The actual queue size in the switch or router is assumed to be sampled every Δt units of time (s), and the packet drop controller provides a new value of the drop probability p_{d} every Δt units of time. Let $q(n)$ denote the actual queue size and $T(n)$ the target buffer occupancy at discrete time n, where $n=1\Delta t,\ 2\Delta t,\ 3\Delta t,\ldots$. We determine a drop probability p_{d}, which will drive the queue size to this target buffer occupancy.

Due to the bursty nature of the network traffic and other perturbations in the network, the queue size is highly fluctuating, so we perform some low pass filtering first. We use a discrete-time first-order low-pass filter (an exponentially weighted moving average filter, EWMA) with gain $0<\beta<1$. The filtered queue size is given by

$$\hat{q}(n) = (1-\beta)\hat{q}(n-1) + \beta q(n). \tag{7}$$

The goal of the controller is therefore to adapt p_{d} so that the magnitude of the error signal

$$e(n) = \hat{q}(n) - T(n). \tag{8}$$

is kept as small as possible.