Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computer Networks 49 (2005) 269–294

Computer Networks

www.elsevier.com/locate/comnet

Analysis and performance evaluation of the EFCM common congestion controller for TCP connections

Michael Savorić [*], Holger Karl, Morten Schläger, Tobias Poschwatta, Adam Wolisz

Faculty of Electrical Engineering and Computer Science, Technical University of Berlin, Einsteinufer 25, 10587 Berlin, Germany

Received 2 June 2004; received in revised form 7 January 2005; accepted 12 January 2005
Available online 9 March 2005

Responsible Editor: G. Morabito

Abstract

Today's standard TCP implementations perform congestion control separately for each connection of an end system. It is advantageous in terms of overall performance, e.g., throughput and (short-term) fairness, to share network information between several TCP connections of an end system by establishing a common congestion control.

In this paper, we present a new common congestion-control approach called "ensemble flow congestion management" (EFCM). First, we describe the design constraints of EFCM. Then, we explain and analyze the EFCM control algorithms. After that, we investigate the performance improvement of EFCM compared to standard TCP congestion control under different network conditions. Simulations with EFCM show a considerable increase in throughput and (short-term) fairness without increasing the aggressiveness of a set of TCP connections—we also show this analytically using a fairness argument. The proposed EFCM controller algorithms are easy to implement and have a low additional complexity.
© 2005 Elsevier B.V. All rights reserved.

Keywords: TCP; Congestion control; Network-information reuse; Common congestion control

1. Introduction

In today's Internet, most of the data flows apply a reliable, congestion-controlled transmission based on the TCP [1] transport protocol. A fundamental design principle of the TCP congestion

[*] Corresponding author. Tel.: +49 30 314 23840.
E-mail addresses: savoric@tkn.tu-berlin.de (M. Savorić), hkarl@ieee.org (H. Karl), morten@ieee.org (M. Schläger), posch@tkn.tu-berlin.de (T. Poschwatta), awo@ieee.org (A. Wolisz).

1389-1286/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2005.01.012

control is its self-clocking property, based on received or overdue acknowledgments: After the reception of an acknowledgment, a TCP sender additively increases its sending window. The sending window is multiplicatively decreased if an acknowledgment is overdue, i.e., the retransmission timer expires or the TCP receiver has requested a missing segment by sending duplicated acknowledgments. TCP uses this scheme to probe the network's capacity trying to adapt to it and not to overload the network; it *implicitly* gathers information about the network's capacity or congestion state.

All TCP instances of an end system perform this information gathering separately and independently of each other. But consider the case when there are multiple connections from a given end system to one communicating peer—say, a user downloading several web pages and files from a large web server, a server providing several downloads, or a file-sharing application with multiple downloads. Such multiple connections can exist concurrently, or they can exist back-to-back, a new one starting when the previous one has terminated. In either case, it is likely that all these connections would, eventually, find out the same information about the current network capacity on their own, but this process takes time that is wasted for more efficient communication. Obviously, such independent information gathering is suboptimal and does not fully exploit all information that is present in the end system's protocol stack.

Consequently, exploiting and sharing network information between multiple TCP connections should improve overall performance. Information sharing can be performed once at the start of a new TCP connection to initialize its congestion-control variables with values more adequate than the defaults. We call this approach "one-time network-information reuse". As an extension, information can be shared continuously among several TCP connections during their whole lifetime in order to *jointly* control them; this second approach is called in the Internet community "joint" or "common congestion control". This paper focuses on common congestion control for TCP connections.

Sharing network information between TCP connections should only occur among TCP connections that use the same network path. Otherwise, for example, (too) optimistic data about one path's capacity could be incorrectly applied to TCP senders loading an already overloaded path. Hence, a common congestion-control approach is only reasonable for TCP connections which typically have the same receiver or at least receivers in the same part of the network. These TCP connections form an *ensemble*. The algorithms that determine which, how, and when network information is shared among different TCP connections of an ensemble form the actual *controller* of a common congestion-control approach.

It should be mentioned that common congestion-control approaches are not restricted to the transport layer, they can also be established in the application layer of the Internet protocol stack. For example, a well-known specific type of network-information sharing approach located in the application layer of the Internet protocol stack is the Hyptertext Transfer Protocol (HTTP) in its current version 1.1 [2,3]. It is used in many WWW servers and provides a network-information sharing mechanism for multiple HTTP transactions to one destination. But compared to this, a common congestion control located in the transport layer features much more functionalities. It is able to share network information not only between multiple HTTP transactions to one destination but also between multiple TCP-based transmissions to one destination or to many destinations in the same subnetwork. These are the reasons why we only consider approaches that are located in the transport layer of the Internet protocol stack.

A common congestion controller's job for TCP connections can be divided into two main tasks: First, the common congestion controller has to manage the one-time network-information exchange between *existing* (or even *recently closed*) TCP connections of an ensemble and a *new* TCP connection joining this particular ensemble. This task is similar to the controller's job in existing pure network-information reuse approaches like the ensemble or temporal (one-time) TCP control