Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks

Dah-Ming CHIU and Raj JAIN
Digital Equipment Corporation, 550 King Street (LKG1-2/A19), Littleton, MA 01460-1289, U.S.A.

Abstract. Congestion avoidance mechanisms allow a network to operate in the optimal region of low delay and high throughput, thereby, preventing the network from becoming congested. This is different from the traditional congestion control mechanisms that allow the network to recover from the congested state of high delay and low throughput. Both congestion avoidance and congestion control mechanisms are basically resource management problems. They can be formulated as system control problems in which the system senses its state and feeds this back to its users who adjust their controls.

The key component of any congestion avoidance scheme is the algorithm (or control function) used by the users to increase or decrease their load (window or rate). We abstractly characterize a wide class of such increase/decrease algorithms and compare them using several different performance metrics. They key metrics are efficiency, fairness, convergence time, and size of oscillations.

It is shown that a simple additive increase and multiplicative decrease algorithm satisfies the sufficient conditions for convergence to an efficient and fair state regardless of the starting state of the network. This is the algorithm finally chosen for implementation in the congestion avoidance scheme recommended for Digital Networking Architecture and OSI Transport Class 4 Networks.

Keywords. Computer Network, Network Performance, Resource Management, Congestion Control, Congestion Avoidance, Flow Control, Fairness.

1. Introduction

1.1. Background

Congestion in computer networks is becoming an important issue due to the increasing mismatch in link speeds caused by intermixing of old and new technology. Recent technological advances

Dah-Ming Chiu received the B.Sc. degree with first class honours from Imperial College of Science and Technology, London University, in 1975, and the M.S. and Ph.D. degrees from Harvard University, Cambridge, MA, in 1976 and 1980 respectively.

From 1979 to 1980, he was with AT&T Bell Laboratories, where he worked on applying queuing theory to network modeling. Since 1980, he has been with Digital Equipment Corporation, where he has worked on performance modeling and analysis of computer systems and networks. Currently, he is a member of the Distributed Systems Architecture group, where he is working on the name service architecture and analyzing various distributed algorithms. His research interests include operation systems, distributed systems, computer networks, performance analysis, and optimization theories.

Dr. Chiu is a member of the ACM and the IEEE.

Raj Jain received the B.E. degree from A.P.S. University, Rewa, India, the M.E. degree from Indian Institute of Science, Bangalore, India, and the Ph.D. degree from Harvard University, Cambridge, MA, in 1972, 1974, and 1978, respectively.

His Ph.D. dissertation, entitled "Control Theoretic Formulation of Operating Systems Resource Management Policies," was published by Garland Publishing, Inc. of New York in their "Outstanding Dissertations in the Computer Sciences" series. Since 1978, he has been with Digital Equipment Corporation, where he has been involved in performance modeling and analysis of a number of computer systems and networks including VAX Clusters, DECnet, and Ethernet. Currently, he is a Consulting Engineer in the Distributed Systems Architecture and Performance Group. He spent the 1983–1984 academic year on a sabbatical at the Massachusetts Institute of Technology doing research on the performance of networks and local area systems. For three years he also taught a graduate course on computer systems performance techniques at MIT and is writing a textbook on this subject, to be published by Wiley-Interscience.

Dr. Jain is a member of the Association for Computing Machinery, and a senior member of IEEE.

North-Holland
Computer Networks and ISDN Systems 17 (1989) 1–14

0169-7552/89/$3.50 © 1989, Elsevier Science Publishers B.V. (North-Holland)

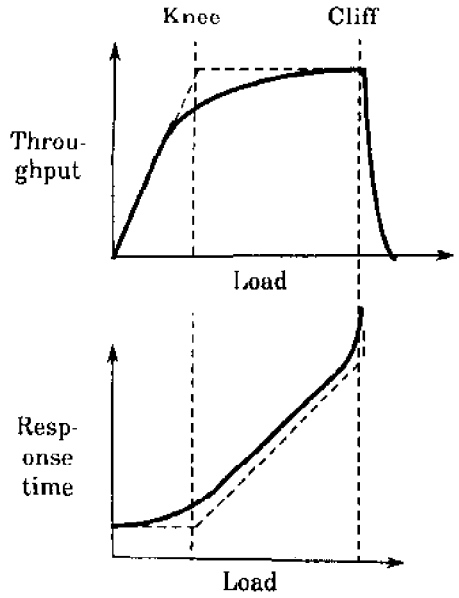

Fig. 1. Network performance as a function of the load. Broken curves indicate performance with deterministic service and interarrival times.

such as local area networks (LANs) and fiber optic LANs have resulted in a significant increase in the bandwidths of computer network links. However, these new technologies must coexist with the old low bandwidth media such as the twisted pair. This heterogeneity has resulted in a mismatch of arrival and service rates in the intermediate nodes in the network causing increased queuing and congestion.

Traditional congestion control schemes help improve performance after congestion has occurred. Figure 1 shows general patterns of response time and throughput of a network as the network load increases. If the load is small, throughput generally keeps up with the load. As the load increases, throughput increases. After the load reaches the network capacity, throughput stops increasing. If the load is increased any further, the queues start building, potentially resulting in packets being dropped. Throughput may suddenly drop when the load increases beyond this point and the network is said to be *congested*. The response-time curve follows a similar pattern. At first the response time increases little with load. When the queues start building up, the response time increases linearly until finally, as the queues start overflowing, the response time increases drastically.

The point at which the packets start getting lost is called a *cliff* due to the fact that the throughput falls off rapidly after this point. We use the term *knee* to describe the point after which the increase in the throughput is small, but when a significant increase in the response time results.

A scheme that allows the network to operate at the knee is called a *congestion avoidance* scheme, as distinguished from a *congestion control* scheme that tries to keep the network operating in the zone to the left of the cliff. A properly designed congestion avoidance scheme will ensure that the users are encouraged to increase their traffic load as long as this does not significantly affect the response time, and are required to decrease them if that happens. Thus, the network load oscillates around the knee.

Both congestion avoidance and congestion control mechanisms are dynamic resource management problems that can be formulated as system control problems in which the system senses its state and feeds this back to its users who adjust their controls. For the congestion problem, the state consists of the load on the network and the control is the number of packets put into the network by the users. Often a window mechanism is used in the transport layer protocols to limit the number of packets put into the network. An alternative mechanism consists of setting a limit on the rate (packets per second or bits per second) that can be sent by a user. In either case, the control (window or rate) can be dynamically adjusted as the total load on the system changes. This control, which we call the increase/decrease algorithm, is at the heart of all congestion avoidance mechanisms.

We have investigated a number of congestion avoidance mechanisms, reported in a series of papers, and this paper is a part of that series [7,8,10,11]. The concept of congestion avoidance and several alternatives are described in [7]. We chose a particular alternative called the "binary feedback scheme" which is described in detail in [11]. This scheme is later extended in [10] to include a "selective feedback" mechanism in which the routers monitor different users and permit some users to increase load while requesting others to decrease load. All of our work on congestion avoidance is summarized in [8].

This paper concentrates on a detailed analysis of the increase/decrease algorithms. This analysis resulted in the selection of the increase/decrease algorithms used in the binary feedback scheme proposed in [11] and [10]. However, the analysis presented here is general and applies to many other applications besides congestion avoidance.

Briefly, the binary feedback scheme for congestion avoidance operates as follows. The resources in the network monitor their usage and determine if they are loaded below or above an optimal load level. Depending upon the load level, the resource sends a binary feedback (1 = overloaded, 0 = underloaded) to the users who then adjust their load using an increase/decrease algorithm. This binary feedback is sent by setting a bit in the packet header. The use of a bit in the packet header as a feedback mechanism has been incorporated into the OSI connectionless networking protocol standards [4]. The bit is called a "congestion experienced bit" and is a part of a field called "quality of service" in the network layer header.

The abstract model assumes that all the users sharing the same bottleneck will receive the *same* feedback. Based on this feedback, the users try to adjust their load so that the bottleneck is efficiently used as well as equally shared by all users. In this abstracted context, we assume that the feedback and control loop for all users is synchronous, that is, all users receive the same feedback and react to it; the next feedback is then generated after all users have reacted to the feedback and so on. Also, we concentrate on one bottleneck resource and the users that share it. Because of these abstractions, we are able to demonstrate some of the subtle behavior of this type of algorithm. The results presented here were verified by detailed simulations of real networks [7,10,11].

1.2. Past Work

The algorithms studied here belong to a class of distributed algorithms for managing distributed

At the other end of the spectrum, we have decentralized decision-making. In this case the decisions are made by the users while the resources feed information regarding current resource usage. Algorithms studied by Jaffe [5] and later extensions by Gafni [2] and Mosely [9] are all good examples of this approach.

In this paper we analyze a class of decentralized decision-making algorithms that are based on a special form of feedback, namely the feedback from the resource is a binary signal. This binary signal indicates whether the resource is currently overloaded or underutilized. A very good reason for considering a binary form of feedback is the motivation of making the controller/manager of the resource as simple and efficient as possible. The requirement of a binary feedback often minimizes the work at the resource in generating the feedback.

1.3. Notations and Definitions
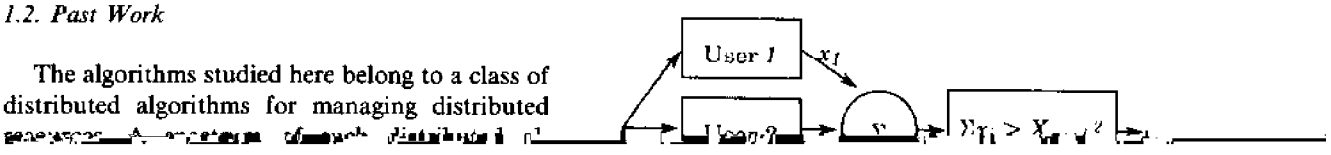
Figure 2 shows the assumed model of the network with n users sharing it. The congestion state of the system is determined by the number of packets in the system. We assume a discrete time operation with time divided into small slots. These slots basically represent intervals at the beginning of which the users set their load level based on the network feedback received during the previous interval. If during time slot t, the ith user's load is $x_i(t)$, then the total load at the bottleneck resource would be $\Sigma x_i(t)$, and the state of the system is denoted by the n-dimensional vector $x(t) = \{x_1(t), x_2(t), \ldots, x_n(t)\}$. Since we are operating at or near the knee, all resources demanded by the users are granted (this is not true at the cliff). Thus, $x_i(t)$ denotes the ith user's