Control Theoretic Analysis and Evaluation for Congestion Control Mechanisms in the Internet

Hiroyuki Hisamatsu

Department of Information Networking

Graduate School of Information Science and Technology

Osaka University

January 2006

Preface

Congestion control is required to transfer packets efficiently in an IP (Internet Protocol) network. The IP protocol merely handles packets from the source host to the destination host, but performs no congestion control. Since no congestion control is performed in an IP network, the performance of the network will seriously deteriorate, if the number of packets transferred in the network increases. Therefore, to transmit packets efficiently in the packet-switching network, the congestion avoidance mechanism of TCP (Transmission Control Protocol) was designed. In the current Internet, most of the traffic is transmitted using TCP. Due to the congestion control mechanism of TCP, the Internet have been prevented from congestion collapse.

The design of TCP has not been a straightforward process, involving many modifications and extensions, and it is still continuing. A large number of modifications and the addition of functionality have been carried out on TCP in unstructured way. Therefore, there is no theoretical background of the performance and operation of TCP. Therefore, the performance evaluation of TCP in a mathematical analysis is required.

For the first issue, we focus on the TCP feedback-based congestion control mechanism. We explicitly model the interaction between the congestion control mechanism of TCP and the network as a feedback system for investigating the transient state behavior of TCP. We then analyze the steady state and the transient state behavior of TCP. The main contribution of this issue is to allow steady state analysis of TCP using the analytic model, and more importantly, to analyze the transient state behavior of TCP using a rigorous manner based on control theory. We show that the bandwidth–delay product mostly determines the stability and the transient behavior of TCP. Our studies indicate that the network becomes stable as the number of TCP connections or the amount of the background traffic increases.

For the second issue, we focus on the congestion control mechanism of DCCP (Datagram Congestion Control Protocol), which is proposed as a new transport layer protocol for real-time applications. We investigate the DCCP congestion control mechanism and RED (Random Early Detection) as independent discrete-time systems. We then model the interaction between the congestion control mechanism of DCCP and the RED router as a feedback system, and analyze the steady state performance and the transient state performance of DCCP/RED. Our focus lies on the ramp-up time, overshoot, and settling time in the evaluating of the transient state performance of DCCP/RED. Consequently, we show that the stability and the transient state performance of DCCP/RED degrade when the weight of the exponential weighted moving average is small. By adding changes to the function with which RED determines the packet loss probability, we propose RED-IQI (RED with Immediate Queue Information). We show that RED-IQI significantly improves the transient state performance such as overshoot, ramp-up time, and settling time compared with RED.

For the third issue, we propose a novel analysis method for large-scale networks. One of the important factors for determining the performance of the Internet is the congestion control mechanisms of TCP. One reason is that TCP traffic occupies a great portion of the current Internet traffic. Nevertheless, in the design and performance analysis issues of a large-scale network, the TCP congestion control mechanism, which is based on a feedback control, has not been taken into consideration. We propose a novel analysis method for such large-scale networks with consideration of the behavior of the congestion control mechanism of TCP. In the analysis, we model each network component (TCP end-hosts and network link) as an independent system, and interconnect them into one system for analyzing the entire network. By the analysis, we will derive the utilization of the network link, packet loss ratio of the link buffer, the round-trip time and throughput of TCP connections, and the location and the degree of the network congestion. We show that our analysis method can adequately model the behavior of TCP connections in a large-scale network.

Acknowledgements

I would like to express my most appreciation to my adviser, Prof. Masayuki Murata for his invaluable help and continuous support. I am also heartily grateful to Prof. Makoto Imase and Prof. Hirotaka Nakano for being readers of my thesis committee. Their expertise and insightful comments have been helpful.

I am most grateful to Prof. Masayuki Murata for his enthusiasm in teaching and pursuing research on congestion control with me. His active interest and encouragement have been of great help in pursuing my efforts in this area and his standards of excellence will continuously stay with me throughout my whole life.

Furthermore, I would like to extend my appreciation to Associate Prof. Hiroyuki Ohsaki and Associate Prof. Go Hasegawa. They have been encouraged and given advices to me through my studies and the preparation of this manuscript. I am also indebted to Associate Prof. Naoki Wakamiya, Dr. Shingo Ata, and Dr. Shin'ichi Arakawa for giving me helpful comments and feedbacks.

I would like to thank many friends and colleagues at the Department of Information Networking of the Graduate School of Information Science and Technology at Osaka University for their support — special thanks to Mr. Yukinobu Fukushima and Mr. Masahiro Sasabe for their expert suggestions as well as their kindness.

Last, but not least, I am deeply grateful to my parents. They have always given the endless love and support to me.

List of Publications

Journal papers

1. H. Hisamatsu, H. Ohsaki, and M. Murata, “Steady State and Transient State Behaviors Analyses of TCP Connections considering Interactions between TCP Connections and Network,” *International Journal of Communication Systems*, vol. 18, pp. 619–637, Sept. 2005.

2. H. Hisamatsu, G. Hasegawa, and M. Murata, “Performance Analysis of Large-Scale IP Networks considering TCP Traffic,” submitted to *IEICE Transactions on Communications*, Nov. 2005.

3. H. Hisamatsu, H. Ohsaki, and M. Murata, “On Modeling Datagram Congestion Control Protocol and Random Early Detection using Fluid-Flow Approximation,” submitted to *WSEAS Transactions on Communications*, Dec. 2005.

Refereed Conference papers

1. H. Hisamatsu, H. Ohsaki, and M. Murata, “On modeling feedback congestion control mechanism of TCP using fluid flow approximation and queueing theory” *Proceedings of 4th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT2001)*, pp. 218–222, Jan. 2001.

2. H. Hisamatsu, H. Ohsaki, and M. Murata, “Steady state and transient behavior analyses of TCP connections considering interactions between TCP connections and network,” *Proceedings of International Symposium on Applications and the Internet (SAINT 2003)*, pp. 309–316, Jan. 2003.

3. H. Hisamatsu, H. Ohsaki, and M. Murata, “Modeling a heterogeneous network with TCP connections using fluid flow approximation and queuing theory,” in *Proceedings of SPIE’s International Symposium on the Convergence of Information Technologies and Communications (ITCom 2003)*, pp. 1–6, Sept. 2003.

4. H. Hisamatsu, H. Ohsaki, and M. Murata, “Steady state and transient state analyses of TCP and TCP-friendly rate control mechanism using a control theoretic approach,” in *Proceedings of SPIE’s International Symposium on the Convergence of Information Technologies and Communications (ITCom 2004)*, pp. 1–6, Oct. 2004.

5. H. Hisamatsu, H. Ohsaki, and M. Murata, “Fluid-based analysis of a network with DCCP connections and RED routers,” *Proceedings of International Symposium on Applications and the Internet (SAINT 2006)*, pp. 22–29, Jan. 2006.

Non-Refereed Technical papers

1. H. Hisamatsu, H. Ohsaki, and M. Murata, “Steady state and transient behavior analyses of TCP Coneections considering interactions between TCP connections and network,” *Technical Report of IEICE* (IN2001-149) *(in Japanese)*, pp. 85–90, Jan. 2002.

2. H. Hisamatsu, H. Ohsaki, and M. Murata, “Steady State Analysis of TCP Connections with Different Propagation Delays,” *Technical Report of IEICE* (IN2002-97) *(in Japanese)*, pp. 41–46, Jan. 2002.

3. H. Hisamatsu, H. Ohsaki, and M. Murata, “Steady State Analysis of TCP Connections with Different Propagation Delays,” *IEICE Society Conference*, Sept. 2002.

4. H. Hisamatsu, H. Ohsaki, and M. Murata, “Steady state and transient state analysis of TCP and TCP-friendly rate control mechanism” *Technical Report of IEICE* (IN2003-46) *(in Japanese)*, pp. 25–30, July. 2003.

5. H. Hisamatsu, H. Ohsaki, and M. Murata, “Fluid-based Analysis of Network with DCCP Connections and RED Routers,” *Technical Report of IEICE* (IN2005-75) *(in Japanese)*, pp. 85–90, Sept. 2005.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Congestion control is required to transfer packets efficiently in an IP (Internet Protocol) network. The IP protocol merely handles packets from the source host to the destination host, but performs no congestion control. Since no congestion control is performed in an IP network, the performance of the network will seriously deteriorate, if the number of packets transferred in the network increases.

The congestion avoidance mechanism of TCP (Transmission Control Protocol) was designed [1]. to transmit packets efficiently in the packet-switching network, in the current Internet, most of the traffic is transmitted by TCP. Due to congestion control mechanism of TCP, the Internet has been prevented from suffering a congestion collapse. TCP consists of two mechanisms called *packet retransmission mechanism* and *congestion control mechanism*. The packet retransmission mechanism of TCP realizes reliable data transfer between source and destination hosts by keeping track of lost packets in the network. The congestion control mechanism of TCP, on the other hand, realizes efficient utilization of network resources by dynamically adjusting the number of in-flight packets. TCP uses packet loss as feedback information from the network since a packet loss implies congestion occurrence in the network [1].

The fundamental operation of TCP is summarized as follows. Until a packet loss occurs in the network, TCP gradually increases the window size of the source host. As soon as the window size exceeds the bandwidth-delay product (i.e., the available bandwidth × the round-trip delay), excess packets are queued at the buffer of an intermediate router. When the window size increases further, the buffer of the router overflows, resulting in a packet loss. At the source host, TCP conjectures the packet loss by receiving more than three duplicate ACKs. TCP then decreases the window size for resolving congestion. After the reduction of the window size, congestion in the network is relieved, and TCP increases the window size of the source host again. By repeating this control indefinitely, TCP tries to efficiently utilize network resources as well as to prevent congestion in the network.

It is thought that TCP is designed carefully, however, it has evolved over time to its current version. The design of TCP has not been a straightforward process, involving many modifications and extensions, and it is still continuing. The first widely available version of TCP was created in 1983. This primitive TCP only ensures reliable data transfer between source and destination hosts, but was unable to achieve acceptable performance in a large network. TCP Tahoe introduced three new features, Slow Start, Congestion Avoidance, and Fast Retransmission, in order to overcome the primitive TCP shortcomings. Fast Recovery was introduced by TCP Reno. If two or more packets are lost in one round-trip time, TCP Reno can not recovery from a packet loss without a retransmission time out occurring. In TCP NewReno, the Fast Retransmission and the Fast Recovery algorithms was modified in order to overcome such a situation. A large number of modifications and the addition of functionality have been carried out on TCP in unstructured way.

The number of Internet hosts are increasing exponentially. For example, the number of the computers connected to the Internet was 562 in August 1983 [2], which is the year primitive TCP was born, and it was increased to about 350 million in January 2005 [2]. Although the scale of a network increased about 600,000 times, currently the network is operating without failing. It is thought as a result that TCP was designed successfully. However, there is no theoretical background of the performance and its operation of TCP. Therefore, the performance evaluation of TCP in a mathematical analysis is required.

In the literature, there have been a great number of analytical studies on TCP. In those of studies, the average window size and the throughput of TCP by assuming a constant packet loss probability in the network have been derived. However, in reality, the packet loss probability changes according to packet transmission rates from TCP connections. Conversely, the window size of a TCP connection is dependent on the packet loss probability in the network. In this thesis, we focus on the interaction between the congestion control mechanism of TCP and the network as a feedback system for investigating the transient state behavior of TCP.

In recent years, real-time applications, such as video streaming, IP telephony, TV conferencing, and network gaming, become rapidly popular due to the increasing speed of the network, or the rising demand for multimedia applications [3]. DCCP (Datagram Congestion Control Protocol) is therefore proposed as a new transport layer protocol for real-time applications [4]. DCCP performs congestion control between source and destination hosts, and an application using DCCP can choose the type of congestion control mechanism. Currently, "TCP-like congestion control profile" [5] that performs congestion control similar to TCP, and "TFRC congestion control profile" [6] that performs congestion control similar to TFRC (TCP Friendly Rate Control) are proposed.

In the TCP-like congestion control profile, an AIMD (Additive Increase Multiplicative Decrease) window control is performed as with TCP [5]. The AIMD window control additively increases the window size (i.e., the number of packets that can be transmitted in a round-trip time) until a source host detects network congestion. If a congestion in the network is detected, the source host multiplicatively decreases the window size. On the other hand, in the TFRC congestion control profile, a variation of the packet transmission rate caused by the TCP-like congestion control profile is prevented, and congestion control is performed so that the network bandwidth is fairly shared with other competing TCP connections [6]. In DCCP with the TFRC congestion control profile, a destination host primarily performs congestion control. Namely, in the TFRC congestion control profile, the destination host detects network congestion and notifies the source host it. The source host adjusts its packet transmission rate based on the congestion information (e.g., *packet loss event rate*) notified by the destination host.

Whereas TCP and DCCP performs congestion control between source and destination hosts, AQM (Active Queue Management) mechanisms that perform congestion control at routers in the network have been recently capturing the spotlight. A representative AQM mechanism is RED (Random Early Detection) [7], which probabilistically discard arriving packets. With RED, as compared with the conventional DropTail, the average queue length (i.e., the average number of packets in the buffer) of the router can be kept small, and high throughput can be achieved [7, 8]. In particular, keeping the average queue length small is effective in decreasing the end-to-end transmission delay. Hence, it is expected that AQM mechanisms are effective for real-time applications.

Many studies on the congestion control mechanism of TCP, which is adopted in the TCP-like congestion control profile of DCCP, have been extensively performed. Although characteristics of the mixed environment of the TCP congestion control mechanism and RED have been sufficiently investigated, characteristics of the mixed environment of congestion control mechanism of TFRC, which is adopted in the TCP-like congestion control profile of DCCP, and RED have not been sufficiently studied. Specifically, the effect of the interaction between TFRC connections and RED routers has not been fully investigated. In this thesis, we focus on the interaction between the congestion control mechanism of DCCP and the RED router as a feedback system for analyzing the steady state performance and the transient state performance of DCCP/RED.

The number of nodes connected to the Internet and the number of users using the Internet have increased with exponentially. Along with this, demands with respect to design techniques and performance analysis techniques for large-scale networks have been rising. However, the current situation is one where performance analysis techniques for large-scale networks are not adequately provided.

One of the important factors for determining the performance of the Internet is the congestion control mechanisms of TCP. One reason is that TCP traffic occupies a great portion of the current Internet traffic. Nevertheless, in the design and performance analysis issues of a large-scale network, the TCP congestion control mechanism, which is based on a feedback control, has not been taken into consideration. Most of the previous works regarding large-scale network design assumed the constant-rate UDP flows as traffic demand. In this thesis, we propose a novel analysis method for such large-scale networks with consideration of the behavior of the congestion control mechanism of TCP.

1.2 Outline of Thesis

Steady and Transient State Behaviours Analyses of TCP Connections [9-13]

In this thesis, we explicitly model the interaction between the congestion control mechanism of TCP and the network as a feedback system. Namely, we model the congestion control mechanism of TCP as a dynamic system, where the input to the system is the packet loss probability and the output is the window size. Inversely, we model the network as a dynamic system, where the input is the window size and the output is the packet loss probability. The network is modeled by a $M/M/1/m$ queueing system by assuming an existence of a single bottleneck link. Using our analytic model, the transient state behavior of TCP connections is quantitatively evaluated with several numerical examples. We then analyze the steady state behavior and the transient state behavior of TCP by using our model. We derive the throughput and the packet loss probability of TCP, and the number of packets queued in the bottleneck router. We then analyze the transient state behavior of TCP using a control theoretic approach, showing the influence of the number of TCP connections and the propagation delay on the transient state behavior of TCP. Through numerical examples, it is shown that the bandwidth–delay product of a TCP connection significantly affects its stability and transient state behavior. It is also shown that, contrary to one's intuition, the network becomes more stable as the number of TCP connections and/or the amount of background traffic increases.

Fluid-Based Analysis of a Network with DCCP Connections and RED Routers [14-18]

We model the DCCP congestion control mechanism and RED as independent discrete-time systems using fluid-flow approximation. By interconnecting DCCP congestion control mechanisms and RED routers, we model the entire network as a feedback system called *DCCP/RED*. We then analyze the steady state performance and the transient state performance of DCCP/RED. Specifically, we derive the packet transmission rate of DCCP connections, the packet transmission rate, the packet loss probability, and the average queue length of the RED router in steady state. Moreover, we investigate the parameter region where DCCP/RED operates stably by linearizing DCCP/RED around its equilibrium point. We also evaluate the transient state performance of DCCP/RED in terms of ramp-up time, overshoot, and settling time. Consequently, we show that the stability and the transient state performance of DCCP/RED degrade when the weight of the exponential weighted moving average, which is one of RED control parameters, is small. To solve this problem, we propose RED-IQI (RED with Immediate Queue Information) by adding changes to the function with which RED determines the packet loss probability, as an applications of our analytic result. We analyze the transient state performance of the feedback system DCCP/RED-IQI where DCCP connections and RED-IQI routers are interconnected. Consequently, we show that DCCP/RED-IQI has significantly better transient state performance than DCCP/RED.

Performance Analysis of Large-Scale IP Networks considering TCP Traffic [19]

As another goal of this thesis, we propose a novel analysis method for such large-scale networks with consideration of the behavior of the congestion control mechanism of TCP. In the analysis, we model each network component (end-host's TCP and network link) as a independent system, and interconnect them into one system for analyzing the entire network. We answer to the following questions by using our analysis results: When the network traffic injected to the network increases, which link will become congested? Which of the access networks and the core networks is the bottleneck of the entire network? We then derive the utilization of the network link, packet loss ratio of the output link buffer, the round-trip time and throughput of TCP connections, and the location and the degree of the network congestion. Through numerical examples, it is shown that the capacity of the core network gets low by increasing the access link bandwidth. It is also shown that our analysis method can treat the behavior of TCP connections in the large-scale network appropriately.

Chapter 2

Steady and Transient State Behaviours Analyses of TCP Connections

We model the interaction between the congestion control mechanism of TCP and the network as a feedback system; that is, both the congestion control mechanism of TCP running on a source host and the network seen by TCP are modeled by dynamic systems. We model the congestion control mechanism of TCP as a dynamic system, where the input to the system is the packet loss probability in the network and the output from the system is the window size of TCP. Then, we model the network seen by TCP as a dynamic system, where the input to the system is the window size and the output from the system is the packet loss probability. We analyze the steady state and the transient behavior of TCP. We first derive the throughput of each TCP connection, the packet loss probability at the bottleneck router, and the average queue length at the bottleneck router. by utilizing the control theory, which has been developed in the control engineering, we analyze the transient behavior of TCP. We then show quantitatively how the stability and the transient behavior of TCP are affected by several system parameters: the number of TCP connections, the propagation delay, the bottleneck link capacity, and the buffer size of the bottleneck router.

2.1 Background

A feedback-based congestion control mechanism is essential to realize an efficient data transfer services in a packed-switched network. In the current Internet, a sort of feedback-based congestion control mechanisms called TCP (Transmission Control Protocol) has been used. TCP has two mechanisms called *packet retransmission mechanism* and *congestion control mechanism*. The packet retransmission mechanism of TCP realizes reliable data transfer between source and destination hosts by keeping track of lost packets in the network. The congestion control mechanism of TCP, on the contrary, realizes efficient utilization of network resources by dynamically adjusting the number of in-flight packets.

The most-widely deployed implementation of TCP called *TCP Reno* uses a packet loss in the network as feedback information from the network since a packet loss implies congestion occurrence in the network [1]. The fundamental operation of TCP Reno is summarized as follows. Until a packet loss occurs in the network, TCP Reno gradually increases the window size of a source host. As soon as the window size exceeds the bandwidth-delay product (i.e., the available bandwidth × the round-trip delay), excess packets are queued at the buffer of an intermediate router. When the window size increases further, the buffer of the router overflows, resulting in a packet loss. At the source host, TCP Reno concludes there is packet loss after receiving more than three duplicate ACKs. TCP Reno then decreases the window size for resolving congestion. After the reduction of the window size, congestion in the network is relieved, and TCP Reno increases the window size of the source host again. By repeating this control indefinitely, TCP Reno tries to efficiently utilize network resources as well as to prevent congestion in the network.

In the literature, there have been a great number of analytical studies on TCP (e.g., [20-30]). Most of those studies assume a constant packet loss probability in the network, and derive the throughput of TCP connections [20, 26, 28, 30] or the distribution of window sizes of TCP connections [24, 25, 29]. However, the packet loss probability, in reality, changes according to packet transmission rates from TCP connections. Conversely, the window size of a TCP connection is dependent on the packet loss probability in the network. In this chapter, we explicitly model the interaction between the congestion control mechanism of TCP and the network as a feedback system for investigating the steady state and the transient state behaviors of TCP. For modeling the congestion control mechanism of TCP, we use four different analytic models presented in [31-33]. As a network model, we use a $M/M/1/m$ queueing system, where the input traffic is mixture of TCP traffic and background traffic (i.e., non-TCP traffic).

In [30], the authors have analyzed the performance of TCP by modeling the network as a $M/D/1/m$ queuing system. However, the authors have focused only on the steady state behavior of TCP; that is, the transient state behavior of TCP has not been evaluated. In addition, their analytic model is not TCP Reno but TCP Tahoe, which does not have several important mechanisms found in TCP Reno. For instance, the effect of the fast retransmit mechanism in TCP Reno has not been investigated. In [22, 33], analytic models for TCP Reno and the RED (Random Early Detection) router have been presented, and the performance of TCP with the RED router has been analyzed. In [22], the primary focus of the analysis is on the steady state behavior of TCP. Only a qualitative discussion on the transient state behavior has been presented. In [33], a control theoretic approach has been taken to analyze the stability and the transient state behavior of TCP, where the RED router is modeled by a non-linear discrete-time system. On the other hand, the main objective of this chapter is to analyze the transient state behavior of TCP with the Drop-Tail router, since most existing routers in the current Internet are Drop-Tail routers. We take a different approach of modeling the Drop-Tail router using a queuing theory.

In [21], the authors have derived the average file transfer time without assuming a constant packet loss probability in the network. However, the stability and the transient state behavior of TCP have not been analyzed. In [27], the authors have modeled the interactions of a set of TCP flows and AQM (Active Queue Management) gateways, and they have showed a transient state behavior of TCP. Since their methodology is based on the solution of the differential equations, they have not showed the transient state behavior of TCP, rigorously. In [23], the authors have analyzed a combined TCP and AQM (Active

Queue Management) gateways model from a control theoretic standpoint. However, they have only focused on the stability of the queue length of RED gateways.

In our analytic model, both TCP traffic and background traffic are taken account of. We model the interaction between the congestion control mechanism of TCP and the network as a feedback system; that is, both the congestion control mechanism of TCP running on a source host and the network seen by TCP are modeled by dynamic systems (Fig. 2.1).

The congestion control mechanism of TCP is a window-based flow control mechanism, and it dynamically changes the window size according to occurrence of packet losses in the network. Hence, there exists a tendency that when the packet loss probability is small, the window size becomes large. On the contrary, when the packet loss probability is large, the window size tends to become small. We model the congestion control mechanism of TCP as a dynamic system, where the input to the system is the packet loss probability in the network and the output from the system is the window size of TCP.

On the other hand, the network seen by TCP behaves such that when the number of packets entering the network increases, some packets are waited at the buffer of the router destined for the bottleneck link. This sometimes causes buffer overflow, resulting in a packet loss. So the packet loss probability becomes large when the number of packets entering the network increases. Thus, the network seen by TCP can be modeled by a dynamic system, where the input to the system is the window size and the output from the system is the packet loss probability.

In this chapter, we have two main contributions. One of the main contributions of this chapter is to propose the model considering interaction between the congestion control mechanism of TCP and the network as a feedback system for investigating the transient state behaviors of TCP. The other main contributions is to analyze the transient state behavior of TCP using a rigorous manner based on control theory. We derive the throughput of each TCP connection, the packet loss probability at the bottleneck router, and the average queue length (i.e., the number of packets awaited in the buffer) at the bottleneck router. By utilizing the control theory, which has been developed in the control engineering, we analyze the transient state behavior of TCP. We show quantitatively how the stability and

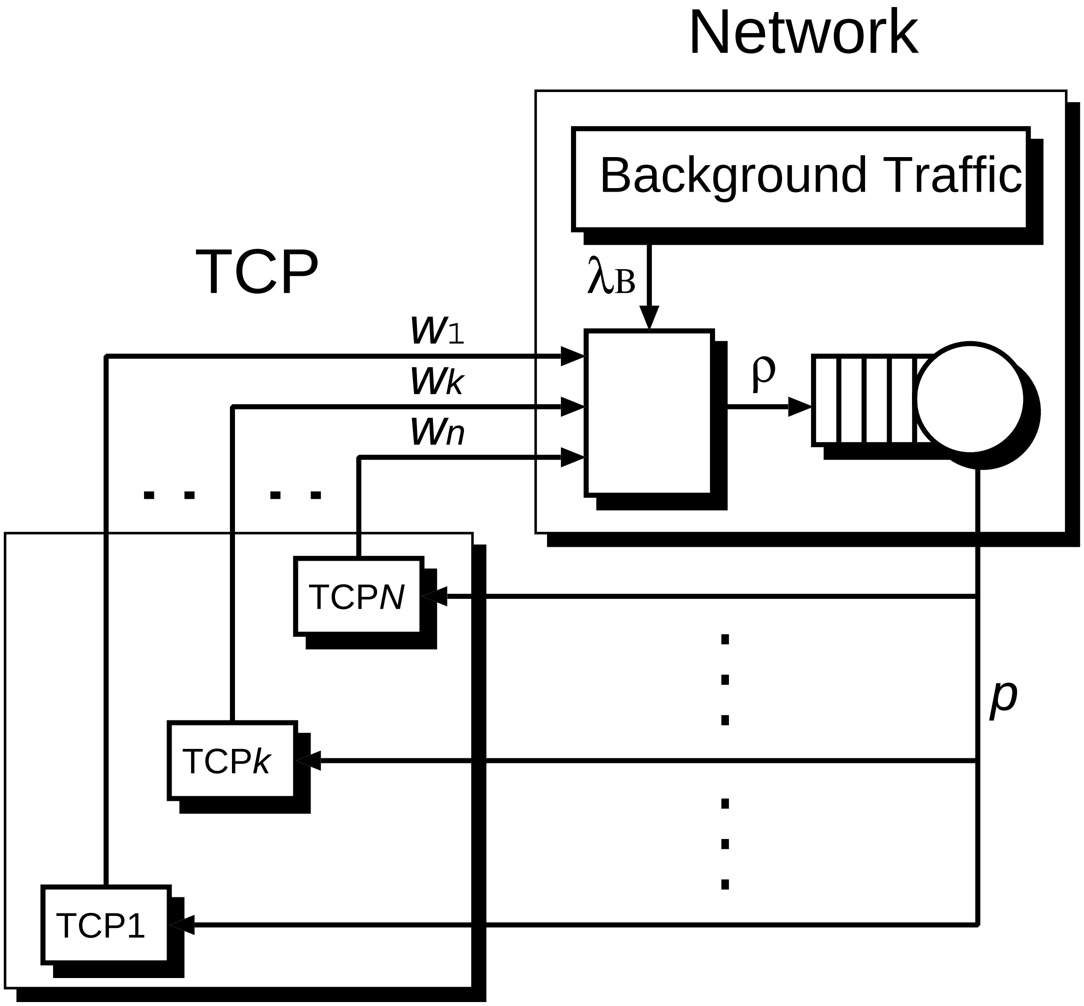


Figure 2.1: Analytic model as a feedback system consisting of TCP connections and network

the transient state behavior of TCP are affected by several system parameters: the number of TCP connections, the propagation delay, the bottleneck link capacity, and the buffer size of the bottleneck router.

2.2 Analytic Model

In this section, we describe how the interaction between TCP and the network can be modeled as a feedback system. We first model the network using queueing theory, and then present four analytic models of the congestion control mechanism of TCP.

2.2.1 Modeling Network using Queuing Theory

We assume that there exists only a single bottleneck link in the network. In the followings, the router just before the bottleneck link is called *bottleneck router*. We also assume that the bottleneck router adopts a Drop-Tail discipline. Provided that the network is stationary, the bottleneck router can be modeled by a single queue. Thus, once the packet arrival rate and the capacity of the bottleneck router are known, the packet loss probability and the average waiting time can be obtained from the queuing theory. Since the packet departure process from a source host is oscillatory, in reality, the network is not stationary. However, as we will show in Section 2.3, the network seen by TCP can be well modeled by a queueing system at a relatively large time scale (e.g., the round-trip time). In the rest of this subsection, we formally describe how the network seen by TCP can be modeled by a queueing system.

Let N be the number of TCP connections, and w_i and r_i be the window size and the round-trip time of ith ($1 \leq i \leq N$) TCP connection. Assuming that each TCP connection continuously sends packets, the transmission rate from ith TCP connection can be approximated by w_i/r_i. The average packet arrival rate at the bottleneck router, λ, is therefore given by $\sum_{i=1}^{N} w_i/r_i + \lambda_B$, where λ_B is the average arrival rate of the background traffic at the bottleneck router. Let μ be the capacity of the bottleneck link, the offered traffic load at the bottleneck router ρ is given by $\rho = \lambda/\mu$. Depending on the packet arrival process, the distribution of the packet processing time, and the buffer capacity, there can be several queuing systems suitable for modeling the network seen by TCP. As a network model, we use a finite buffer queuing system, $M/M/1/m$, where m represents the buffer size of the bottleneck router. Namely, the packet loss probability at the bottleneck router is given by

$$p = \frac{(1-\rho)\rho^m}{1-\rho^{m+1}} \tag{2.1}$$

where ρ and r are given by

$$\begin{aligned} \rho &= \frac{1}{\mu}\left(\frac{\sum_{i=1}^{N} w_i}{r_i} + \lambda_B\right) \\ r_i &= 2\tau_i + \frac{\rho\,(1 - m\rho^m + m\rho^{m+1})}{\mu(1-\rho^{m+2})\,(1-\rho)} \end{aligned}$$

Use of the queuing model to analyze the steady state behavior of TCP is straightforward and promising. However, the application of the queueing model to analyze the transients state behavior requires the careful treatment since the queueing model was originally developed for analyzing not the dynamic behavior but the statistical behavior. However, we believe that the queueing model can give some insight on dynamic systems. The applicability of our approximate analysis will be validated in the latter sections by comparing analytic results with simulations ones.

2.2.2 Modeling TCP using Different Approaches

The congestion control mechanism of TCP is quite complicated since it performs several control mechanisms such as detecting packet losses in the network and retransmitting lost packets. It is therefore impossible to build an exact analytic model of TCP. In this section, we model only the main part of the congestion control mechanism of TCP, and ignore the rest; that is, we model the essential behavior of TCP (i.e., the window-based flow control mechanism and the loss recovery mechanism including the fast retransmit mechanism of TCP Reno) in its congestion avoidance phase. In our analytic model, several TCP mechanisms, such as the slow-start phase, the Nagle algorithm, and the delayed ACK, are not modeled. Since our main focus is in long-lived TCP connections, we assume that the TCP traffic is persistent. Under this assumption of persistent traffic, effect of these unmodeled factors could be negligible.

In [32-34], several analytic models for the congestion avoidance phase of TCP have been presented, describing the relation between the packet loss probability in the network and the resulting window size of TCP. In what follows, we introduce four analytic models called A, A', B, and C, which are derived from different modeling approaches. In Section 2.3, we will discuss which model is suitable for analyzing the transient state behavior of TCP.

- **Model A**

In [32], by assuming a constant packet loss probability in the network (denoted by p), the authors have presented an analytic model describing the window size of a TCP connection in steady state. The authors have derived the average throughput of a TCP connection. In this model, the authors assume that the initial window size at the beginning of a congestion avoidance phase is equal to that at the beginning of the next congestion avoidance phase, and that TCP sends the number $1/p$ of packets in each congestion avoidance phase. In summary, the average throughput of a TCP connection, λ_T, is derived as

$$\lambda_T = \frac{\frac{1-p}{p} + E[W] + \hat{Q}(E[W])\frac{1}{1-p}}{r\left(\frac{b}{2}E[W] + 1\right) + \hat{Q}(E[W])T_o\frac{f(p)}{1-p}}$$

where

$$\begin{aligned} E[W] &= \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2} \\ \hat{Q}(w) &= \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{(1-(1-p)^w)} \\ f(p) &= 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6 \end{aligned}$$

and r is the average round-trip time of the TCP connection, and b is a parameter of delayed ACKs (i.e., a destination host returns an ACK packet for every b data packets). T_O is the length of TCP's retransmission timer. $\hat{Q}(w)$ is a probability that when the window size is w, the source host fails to detect a packet loss from duplicate ACKs. From these equations, the window size of TCP in steady state, w_A, is given by

$$w = \lambda_T\, r \tag{2.2}$$

- **Model A$'$**

When the packet loss probability is very small ($p << 1$), Eq. (2.2) is approximated as [32]

$$w \simeq \sqrt{\frac{3}{2bp}} \tag{2.3}$$

• **Model B** In [34], the authors have analyzed a congestion control mechanism using ECN (Explicit Congestion Notification). ECN is a mechanism to explicitly notify source hosts of congestion occurrence in the network. When a router experiences congestion, by setting the CE bit of arriving packets, it informs destination hosts of the congestion occurrence. Then the destination host informs the source host of the congestion occurrence by setting the ECN echo bit of ACK packets In [34], the authors assume that the ECN echo bit of an ACK packet is set with a probability of p_E, and have derived a state transition equation for the window size. Let $w(k)$ be the window size at slot k (i.e., the time when kth ACK packet is received). Their analytic model is different from TCP; that is, when the ECN echo bit is set, the source host linearly increases the window size by $I(w(k))$. Otherwise, it multiplicatively decreases the window size by $D(w(k))$. By calculating the expected value of the window size at each receipt of an ACK packet, the evolution of the window size is given by

$$w(k) = w(k-1) + (1-p_E)\, I(w(k-1)) - p_E\, D(w(k-1)) \tag{2.4}$$

The analytic model presented in [34] is not for TCP, but can be easily applied. Namely, an ACK packet with the ECN echo bit not set corresponds to a non-duplicate ACK in TCP (i.e., indication of no congestion). Similarly, an ACK packet with the ECN bit set corresponds to duplicate ACKs (i.e., indication of congestion). Thus, when the packet loss probability is p, the state transition equation for the window size, w_B, is given by

$$\begin{aligned} w(k) &= w(k-1) + (1-p)\,\frac{1}{w(k-1)} \\ &\quad -p\,(1-\hat{Q}(w(k-1)))\,\frac{w(k-1)}{2} - p\,\hat{Q}(w(k-1))\,(w(k-1)-1) \end{aligned} \tag{2.5}$$

Note that we modify and extend Eq. (2.4) to include the timeout mechanism of TCP.

• Model C

In [33], the authors have derived the state transition equation for the window size in the congestion avoidance phase of TCP. This analytic approach uses a discrete-time model, where a time slot corresponds to the duration between two succeeding packet losses. However, their analytic model is not for the Drop-Tail router but for the RED router, where the router randomly discards arriving packets. In what follows, we describe a modification to the analytic model presented in [33] for analyzing TCP with the Drop-Tail router.

In [33], the authors have derived $\overline{X}(k)$, the expected number of packets passing through the RED router at slot k as

$$\overline{X}(k) = \frac{1/p_b(k) + 1}{2}$$

where $p_b(k)$ is the packet dropping probability of the RED router at slot k. Let p be the packet loss probability of the Drop-Tail router, $\overline{X}(k)$ is changed to

$$\overline{X}(k) = \sum_{n=1}^{\infty} n(1-p)^{n-1} p = \frac{1}{p}$$

Thus, when the packet loss probability is p, the window size w at the beginning of slot k is obtained as [33]

$$w(k) = \frac{1}{4}\left\{-1 + \sqrt{(1 - 2w(k-1))^2 + \frac{8}{p}}\right\} \tag{2.6}$$

Note that Eq. (2.6) is derived by assuming that a packet loss probability is constant in a slot. Since the packet loss probability is, in reality, increased as the window size increases, this analytic model might overestimate the window size.

We note that models A and A' are built based on the window size in steady state. It is therefore expected that these models are not suitable for analyzing the transient state behavior of TCP. On the contrary, models B and C describe the dynamic behavior of the

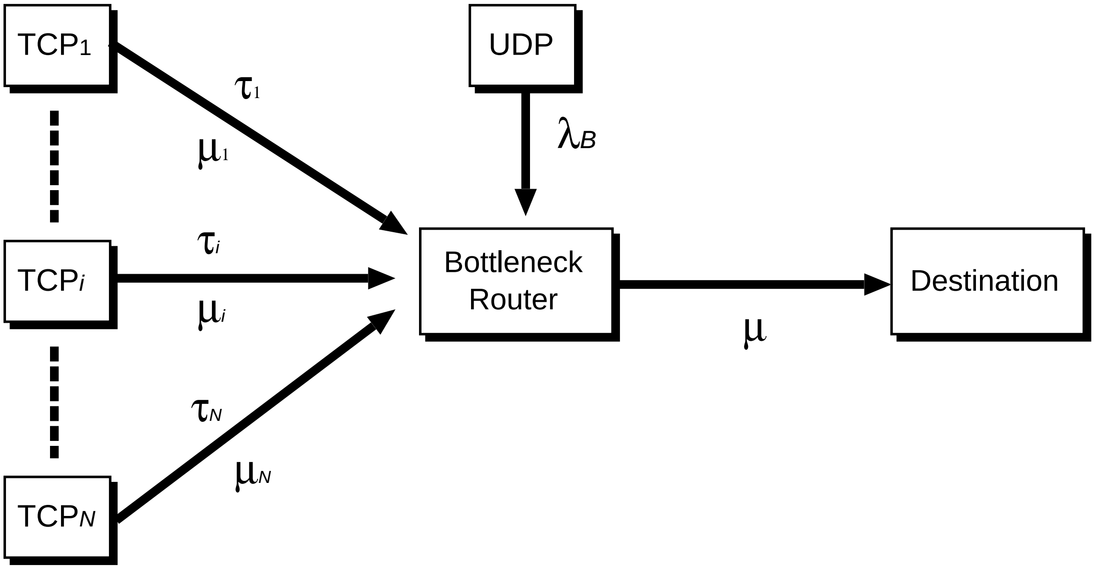


Figure 2.2: Simulation model of 10 TCP connections and a single bottleneck link

window size in the congestion avoidance phase. Thus, it is expected that models B and C are suitable for analyzing the transient state behavior of TCP. In the next section, we compare these four analytic models using numerical and simulation results.

2.3 Model Validation with Simulation

2.3.1 Simulation Model

The simulation model is shown in Fig. 2.2. In this model, 10 TCP connections share the bottleneck link. The propagation delay of ith TCP connection is $5 + i$ [ms], and the link capacity from the ith source host to the router is $5 + 0.5i$ [packet/ms]. We model the background traffic as UDP packets, where the packet arrival of UDP packets is modeled by a Poisson process with the average arrival rate of $\lambda_B = 2$ [packet/ms]. Unless explicitly noted, we use the following parameters in all simulations: both TCP and UDP packet sizes are fixed at 1000 [byte], the capacity of the bottleneck link μ is 5 [packet/ms], and the propagation delay of the bottleneck link τ is 5 [ms]. Note that with these simulation parameters, 1 [packet/ms] corresponds to about 8 [Mbit/s]. Also note that performance evaluation for fixed-sized TCP packets would be sufficient since we assume persistent TCP traffic in our analysis. We run every simulation for 30 seconds using ns-2 [35].

2.3.2 Network Models

Figure 2.3 shows the relation between the offered traffic load and the packet loss probability. These values are measured at the bottleneck router for every 10 [ms]. Namely, these values are rough estimation of the *instantaneous* offered traffic load and the *instantaneous* packet loss probability. In the figure, the packet loss probabilities obtained from well-known results of $M/M/1/m$ and $M/M/1$ are also plotted. This figure shows that the dynamics of the network at a relatively small time scale can be well modeled by the $M/M/1/m$ model. Note that the queuing theory is for analyzing the statistical behavior, not the dynamical behavior. Note also that UDP and TCP packet sizes are fixed at 1000 [byte]. This figure indicates that $M/M/1/m$ could be usable for analyzing the transient state behavior of TCP. However, simulation results are scattered around the result of $M/M/1/m$. This means that the packet loss probability has a variability even when the offered traffic load at the bottleneck router is fixed.

2.3.3 TCP Models

By comparing with simulation results, we discuss how accurately four analytic models of TCP capture the relation between the window size and the packet loss probability. Figure 2.4 shows the relation between the packet loss probability and the window size obtained using models A, A', B and C, respectively. In this figure, the window size for a given packet loss probability is obtained using Eqs. (2.2), (2.3), (2.5), and (2.6). Note that in the model A, the analytic result is calculated by assuming no timeout (i.e., $\hat{Q}(w) = 0$). Also note that in the model C, Eq. (2.6) gives the window size at the beginning of a slot, and not the average window size. For comparison purposes, the average window size is calculated and plotted in the figure. Refer to [33] for more detail. We also plot simulation results; that is, points corresponding to the average window size and the packet loss probability.

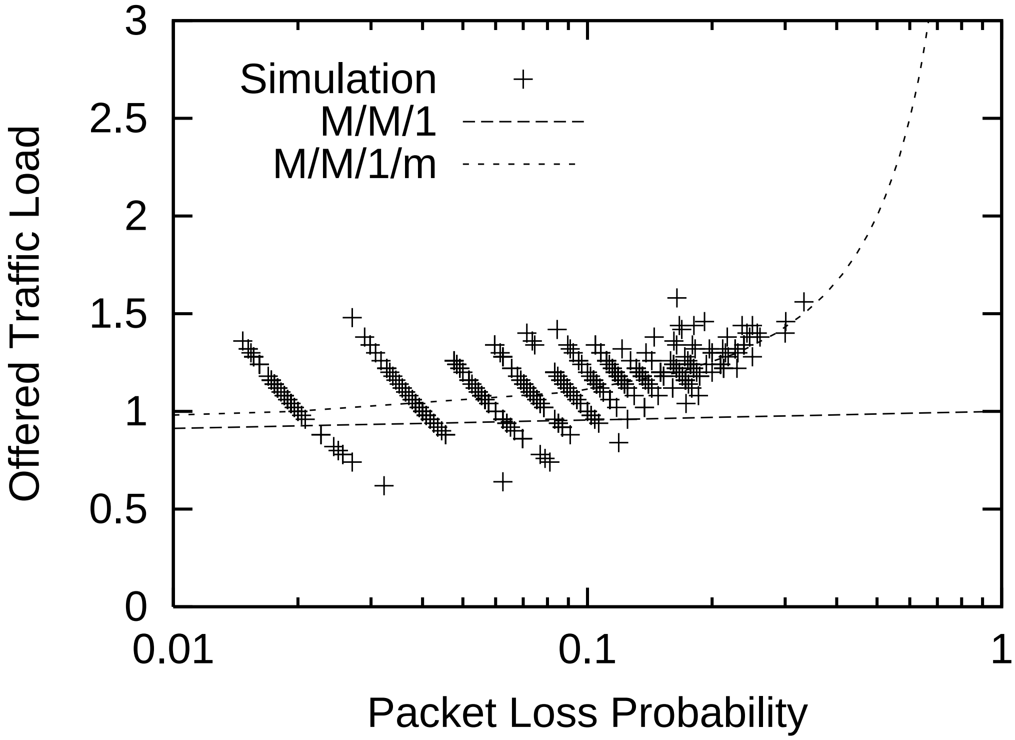


Figure 2.3: Comparison of $M/M/1/m$ queuing system with simulation result

As with Fig. 2.3, these values are *instantaneous values* of the average window size and the packet loss probability, which are measured at the bottleneck router for every 1 [s]. This figure shows that when the packet loss probability is less than 0.02, analytic results obtained from models A, A', and B show good agreement with simulation results. On the other hand, when the packet loss probability is more than 0.03, analytic results obtained from models B and C show good agreement.

From these observations and discussions, we choose the model B for analyzing the steady state and the transient state behaviors of TCP. Regarding the steady state behavior, the model B shows good agreement with simulation results in a certain range of packet loss probabilities. Although models A and A' give close analytic results with those with the model B, models A and A' should not be appropriate for analyzing the transient state behavior of TCP since these models are based on the steady state analysis of TCP. In the following sections, with using the model B, we will derive the TCP throughput, the packet loss probability, and the queue length of the bottleneck router. We will also analyze the

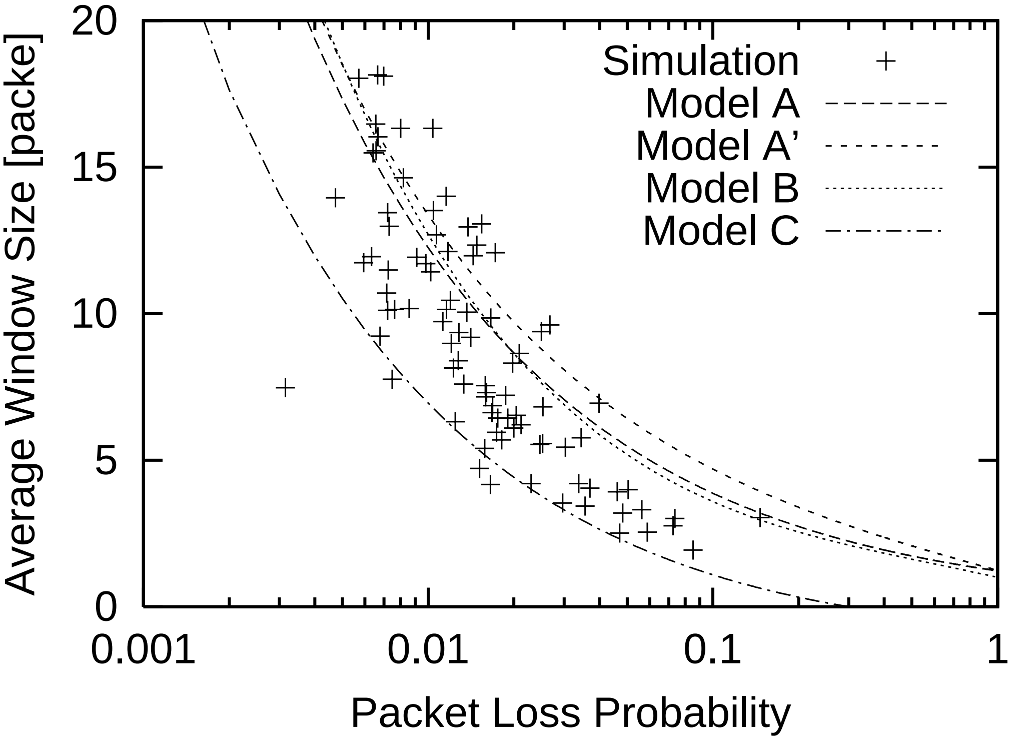


Figure 2.4: Comparison of four TCP models with simulation results

transient state behavior of TCP using a control theoretic approach.

2.4 Steady State Analysis

In what follows, we present steady state analysis for the combination of the model B for TCP and a $M/M/1/m$ queuing system for the network. As have been explained in Section 2.2.2, the model B describes the change of the window size every receipt of an ACK packet. Hence, in the following analysis, the duration between two succeeding ACK packets corresponds to a unit time. In this section, we derive the TCP throughput, the packet loss probability, and the average queue length in steady state. We then validate our approximate analysis by comparing analytic results with simulation ones.

Note that, we have made following assumptions in Section 2.2: (1) to focus on the steady state behavior of TCP, all TCP connections are assumed to operate in the congestion avoidance phase, (2) the bottleneck router is a Drop-Tail router with a single FIFO queue for all TCP connections, (3) the TCP packet size is fixed, (4) all TCP connections have infinite data to transfer, (5) the packet loss only occurs at the bottleneck router due to its buffer overflow, and (6) the maximum window size of TCP is sufficiently larger than the bandwidth–delay product of the network.

The congestion control mechanism of TCP is an AIMD (Additive Increase and Multiplicative Decrease) based feedback control. When the propagation delay is non-negligible, the window size oscillates and never converges to a constant value. Note that the symbol $w(k)$ represents not the instant value of the oscillating TCP window size but the expected value of the TCP window size after a long period.

Let equilibrium values of the TCP window size and the packet loss probability in the network be w^* and p^*, respectively; i.e.,

$$
\begin{aligned}
w^* &\equiv \lim_{k\to\infty} w(k) && (2.7)\\
p^* &\equiv \lim_{k\to\infty} p(k) && (2.8)
\end{aligned}
$$

These values can be numerically obtained by solving Eqs. (2.1) and (2.5) with equating $w(k+1) \equiv w(k)$ and $p(k+1) \equiv p(k)$. Using these equilibrium values, the TCP throughput T and the average queue length of the bottleneck router L are given by

$$
\begin{aligned}
T &= \frac{w^*}{r^*} && (2.9)\\
L &= \rho^* \mu (r^* - 2\tau)\\
&= \frac{\rho^{*2}\,(1 - m\rho^{*m} + m\rho^{*(m+1)})}{(1-\rho^{*(m+2)})\,(1-\rho^*)} && (2.10)
\end{aligned}
$$

where r^* and ρ^* are the equilibrium values of $r(k)$ and $\rho(k)$, respectively and decided by w^* and p^*. In the above equations, the TCP throughput T is approximated by the number of packet per a unit time emitted by source host, and the average queue length L is obtained from the number of customers waiting to be served of the $M/M/1/m$ queue.

We next compare analytic results with simulation ones for validating our approximate analysis. In the following analytic results, we calculate the TCP throughput T, the packet

Table 2.1: Parameter values

$N = 10$	$\mu = 2$ [packet/ms]
$\tau = 30$ [ms]	$\lambda_B = 0.2$ [packet/ms]
$m = 50$ [packet]	packet size = 1000 [byte]

loss probability p^*, and the average queue length L using Eqs. (2.9), (2.8), and (2.10), respectively. Using ns-2 simulator, we run several simulation experiments at a packet level for the same network model with Fig. 2.1. Each simulation experiment is continued for 24 seconds, and the last 20 seconds are used for calculating simulation results — the TCP throughput, the packet loss probability, and the average queue length. Each simulation experiment is repeated 50 times, and 95 % confidence intervals of all performance measures are calculated. Note that our analytic model uses fluidflow approximation whereas simulation results are obtained using a packet-level network simulation.

In obtaining the analytic and simulation results, we use the following parameters: the number of TCP connections $N = 10$, the bottleneck link capacity $\mu = 2$ [packet/ms], the propagation delay $\tau = 30$ [ms], the average arrival rate of the background traffic $\lambda_B = 0.2$ [packet/ms], and the buffer size of the bottleneck router $m = 50$ [packet]. In simulation experiments, we model the background traffic by UDP traffic. The packet size of TCP and UDP packets is fixed at 1000 [byte]. The maximum window size of all TCP connections is fixed at a sufficiently large value, 10,000 [packets]. We use TCP version Reno on all source hosts. Table 2.1 summarizes parameters used in obtaining the analytic and simulation results.

Figure 2.5 shows the TCP throughput, the packet loss probability, and the average queue length for the different bottleneck link capacities. For comparison purposes, another analytic result of the TCP throughput from [32] is shown in Fig. 2.5(a). In [32], the TCP throughput is derived as a function of the round-trip time and the packet loss probability for a TCP connection. More specifically, the TCP throughput T' derived in [32] is given by

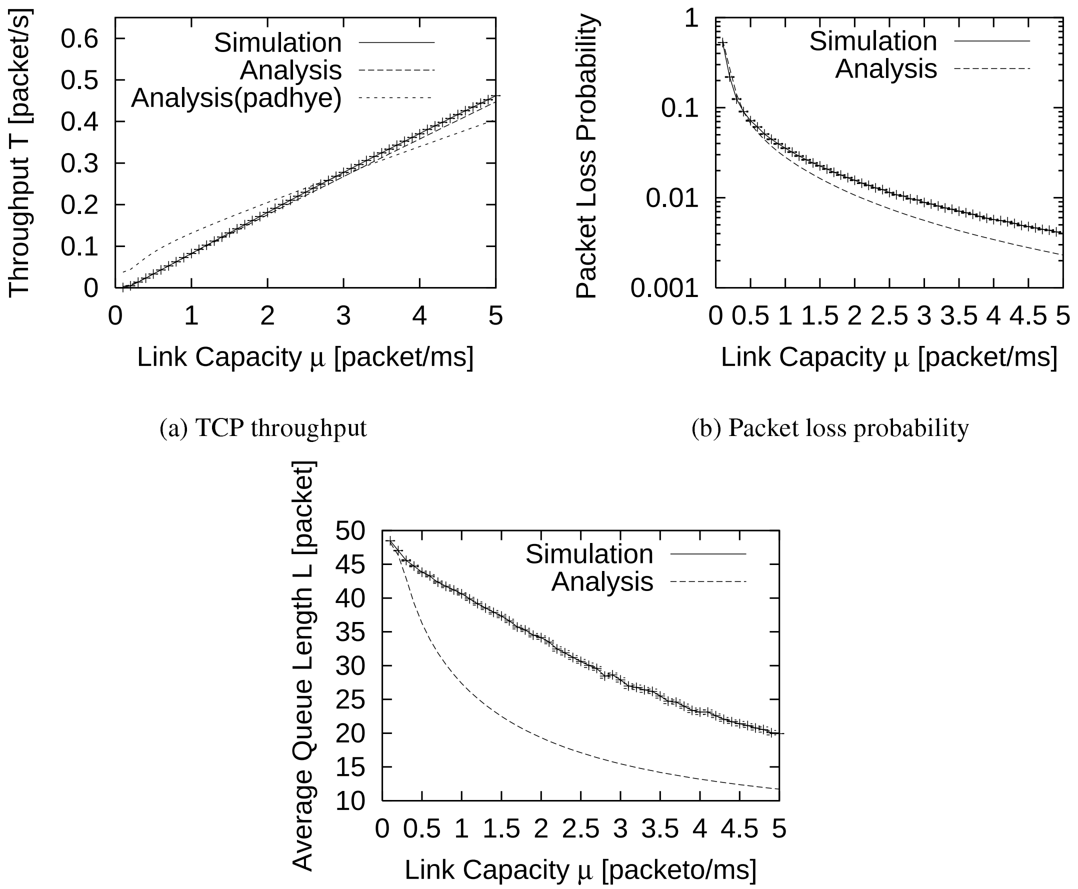
$$T' = \frac{\frac{1-p}{p} + E[W] + \hat{Q}(E[W])\frac{1}{1-p}}{r\left(\frac{b}{2}E[W] + 1\right) + \hat{Q}(E[W])T_o\frac{f(p)}{1-p}}$$

where

$$\begin{aligned} E[W] &= \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2} \\ \hat{Q}(w) &= \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{(1-(1-p)^w)} \\ f(p) &= 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6 \end{aligned}$$

In this section, we calculate the TCP throughput from the above equation using the packet loss probability and the round-trip time obtained from the simulation. It can be found, in terms of the TCP throughput and the packet loss probability, both analytic and simulation results show a good agreement. In particular, in respect to the TCP throughput, it can be found that our analytic results show better agreement with the simulation results than the value obtained from the expression in [32]. However, in terms of the average queue length, it can be found that our analytic results are much smaller than simulation results. Such a disagreement between analytic and simulation results is probably caused by our assumption that the packet arrival at the bottleneck router follows a Poisson process. In running the simulation, the average arrival rate of the background traffic is fixed at $\lambda_B = 0.2$ [packet/ms]. Hence, the amount of the TCP traffic becomes relatively larger than the amounts of the background traffic as the bottleneck link capacity becomes large. As a result, the packet arrival process at the bottleneck router cannot be modeled by a Poisson process.

In Fig. 2.6, both analytic and simulation results are shown for different propagation delays. Similarly to the previous case, it can be found that both analytic and simulation results show a good agreement in terms of the TCP throughput. However, as the propagation delay increases, the packet loss probability obtained from our analysis deviates from the



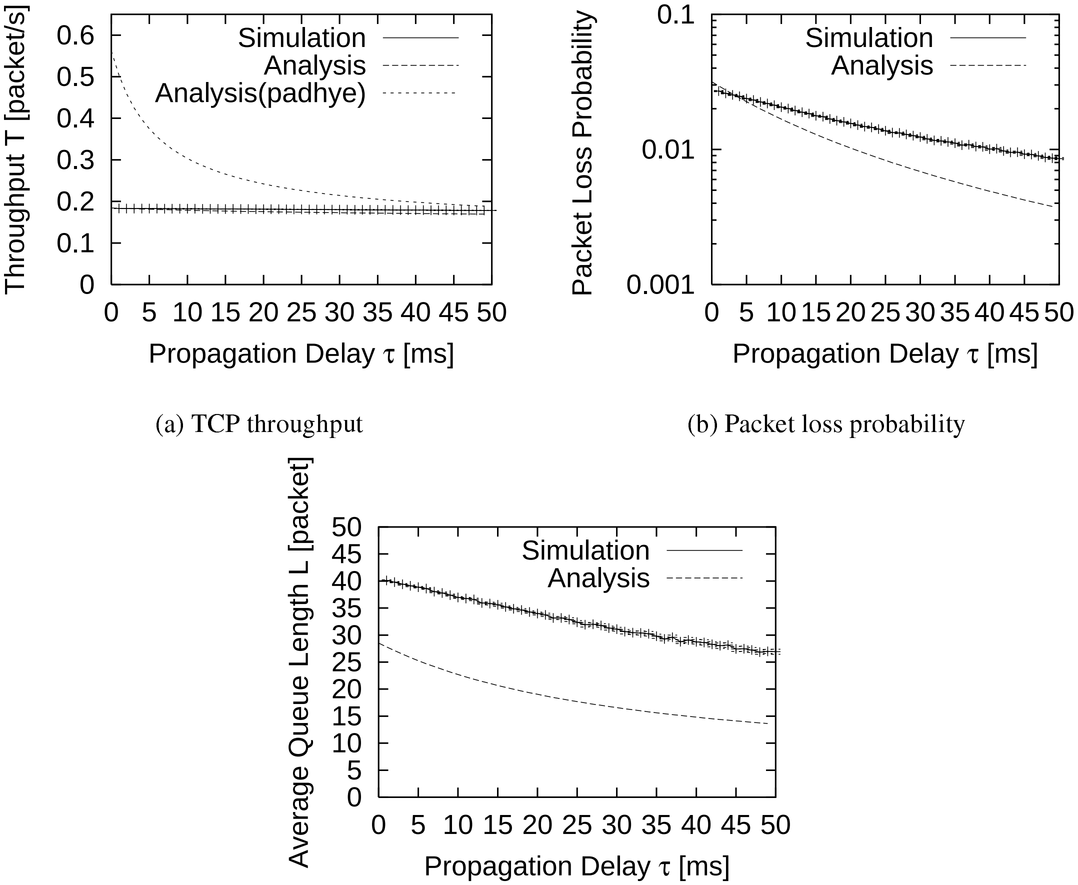
(a) TCP throughput

(b) Packet loss probability

(c) Average queue length

Figure 2.5: Analytic and simulation results for different bottleneck link capacities

corresponding simulation result. It can also be found that, in respect to the average queue length of the bottleneck router, our analytic results are much smaller than simulation ones. Such a disagreement between analytic and simulation results is probably caused by our assumption that the packet arrival at the bottleneck router follows a Poisson process. Since TCP uses a window-based flow control mechanism, the packet emission process from the source host becomes more bursty as the propagation delay becomes large. Hence, as the propagation delay becomes large, the Poisson process becomes insufficient for modeling the arrival process of the background traffic at the bottleneck router.



(a) TCP throughput

(b) Packet loss probability

(c) Average queue length

Figure 2.6: Analytic and simulation results for different propagation delays

2.5 Transient State Behavior Analysis

Using the analytic model presented in Section 2.2, we analyze the transient state behavior of TCP in the congestion avoidance phase. By the word *transient state behavior*, we mean the dynamics of the window size from its initial value to its equilibrium value. TCP changes the window size according to the occurrence of a packet loss in the network. Since a packet loss occurs probabilistically, the window size can be thought of as a random variable. By focusing on the *average behavior* of TCP, we analyze the transient state behavior of TCP. More specifically, we analyze the transient state behavior of TCP by investigating how the expected value of the window size changes.

The state of the network at slot k is then fully described by the window size $w(k)$ and the packet loss probability $p(k)$. For given initial values of the window size and the packet loss probability, the evolution of the window size and the packet loss probability can be numerically obtained. Recall that $w(k)$ is not the instant value of the window size, but the average value of the window size. Using these equations and calculating the evolutions of $w(k)$ and $p(k)$, the transient state behavior of TCP can be analyzed. We next present several numerical examples, showing how the amount of background traffic λ_B and the propagation delay τ of the bottleneck link affect the transient state behavior of TCP. In the following numerical examples, unless explicitly noted, the initial window size is 1 [packet], the initial packet loss probability is 0, the number of TCP connections N is 10, the capacity of the bottleneck link μ is 5 [packet/ms], the propagation delay τ is 15 [ms], and the buffer size of the bottleneck router m is 50 [packet].

Figure 2.7 shows the evolution of the window size in the congestion avoidance phase for the amount of background traffic λ_B of 0, 2.0, and 4.5 [packet/ms]. From this figure, one can find that the window size in steady state becomes small as the amount of background traffic increases, indicating that TCP suffers less throughput. One can also find that the convergence speed (i.e., in this case, the increase rate of the window size) of the window size is independent of the amount of background traffic. This is because, in the congestion avoidance phase, TCP increases the window size by one packet per a round-trip time, which

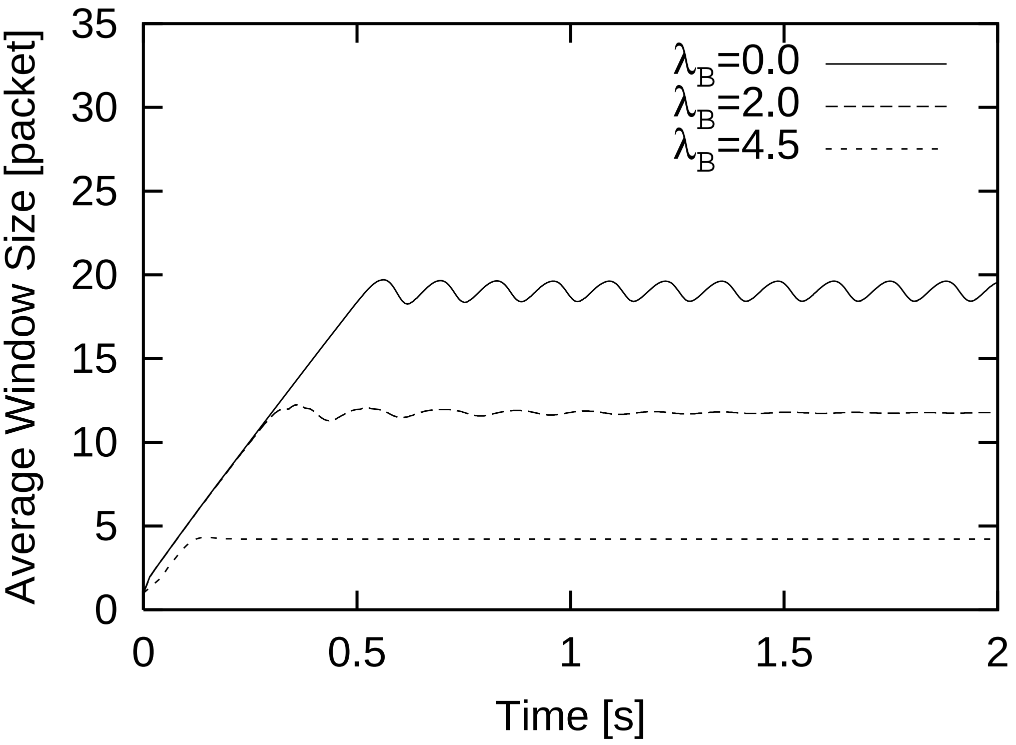


Figure 2.7: Transient state behavior of TCP for different amount of background traffic

is essentially irrelevant to the TCP throughput.

Figure 2.8 shows the evolution of the window size in the congestion avoidance phase for the propagation delay τ of 10, 30, and 50 [ms]. One can find that the window size becomes large as the propagation delay increases. This can be intuitively understood from the increased bandwidth-delay product. In addition, as the propagation delay becomes large, one can find that the convergence speed of the window size becomes slow, and that the ramp-up time of the window size becomes short. In general, as the feedback delay becomes large, the transient state behavior is degraded and the system becomes less stable. However, the contrary occurs when the propagation delay τ is small (e.g.,10 [ms]), the window size oscillates for long (e.g., more than 1.5 [s]). This is because, from a control theoretical viewpoint, the feedback gain in the congestion avoidance phase of TCP is changed according to the round-trip time. Namely, in the congestion avoidance phase of TCP, the window size is incremented by one packet for every round-trip time. Thus, increasing the propagation delay implies decreasing the feedback gain.

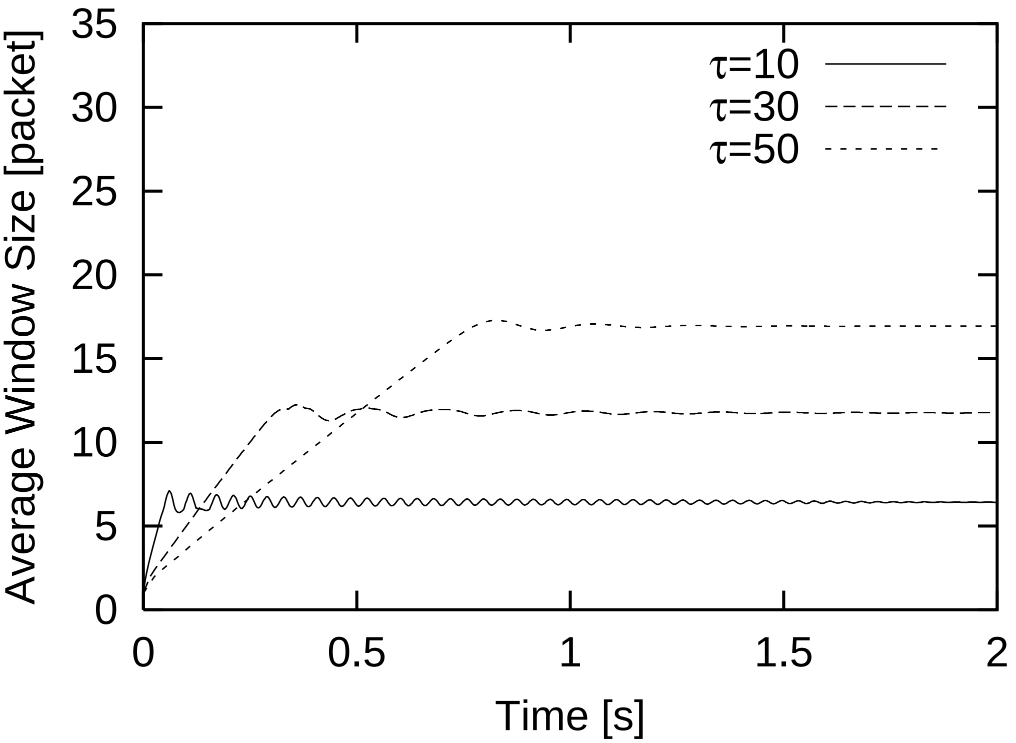


Figure 2.8: Transient state behavior of TCP for different propagation delay of bottleneck link

We then analyze the TCP behavior in transient state using state transition equations. Specifically, by applying the control theory, we show how the TCP window size and the packet loss probability converge to their equilibrium points.

Let $\mathbf{x}(k)$ be the difference between $(w(k), p(k))$ and (w^*, p^*).

$$\mathbf{x}(k) \equiv \begin{bmatrix} w(k) & - & w^* \\ p(k) & - & p^* \end{bmatrix}$$

Since Eqs. (2.1) and (2.5) have non-linearity, we linearize them around their equilibrium points and write them in a matrix form

$$\mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) \tag{2.11}$$

where $\mathbf{A}$ is a state transition matrix. Eigenvalues of the state transition matrix determine the stability and the transient state behavior of the feedback system around the equilibrium point [36]. It is known that the system is stable if the maximum modulus is less than one. It is also known that the smaller the maximum modulus is, the better the transient state behavior becomes. In the followings, we show several numerical examples to reveal how the stability and the TCP transient state behavior are affected by several system parameters — the number of TCP connections, the propagation delay, the bottleneck link capacity, and the buffer size of the bottleneck router.

Figure 2.9 and 2.10 shows the contour plot of the maximum modulus of the eigenvalues. The figure means that the maximum modulus of the eigenvalues becomes large as the filled color becomes dark. A white area is an area where the maximum modulus of eigenvalues is more than 1.0 (i.e., unstable area).

Figure 2.9 shows the maximum modulus of the eigenvalues for different numbers of TCP connections of N = 5, 10, and 15. In this figure, we plot the maximum modulus of eigenvalues of the state transient matrix $\mathbf{A}$ for different bottleneck link capacities of μ = 0–5 [packet/ms] and propagation delays of τ = 0–5 [ms]. The buffer size of the bottleneck router m is fixed at 50 [packet] and the average arrival rate of the background traffic λ_B is fixed at 0.2 [packet/ms].

From Fig. 2.9, one can find that the maximum modulus of the eigenvalues is mostly determined by $\mu \times \tau$. This indicates that the stability and the transient state behavior of TCP are determined by the bandwidth–delay product. This is because the congestion control mechanism of TCP is a window-based mechanism, and it changes the window size at every receipt of an ACK packet. Provided that the packet size is fixed, the number of ACK packets in the network during a round-trip time is proportional to the bandwidth–delay product. In the control engineer's view, the increase of the propagation delay means the decrease of the feedback gain or the feedback delay. Hence, the stability and the transient state behavior of TCP are determined by the bandwidth–delay product.

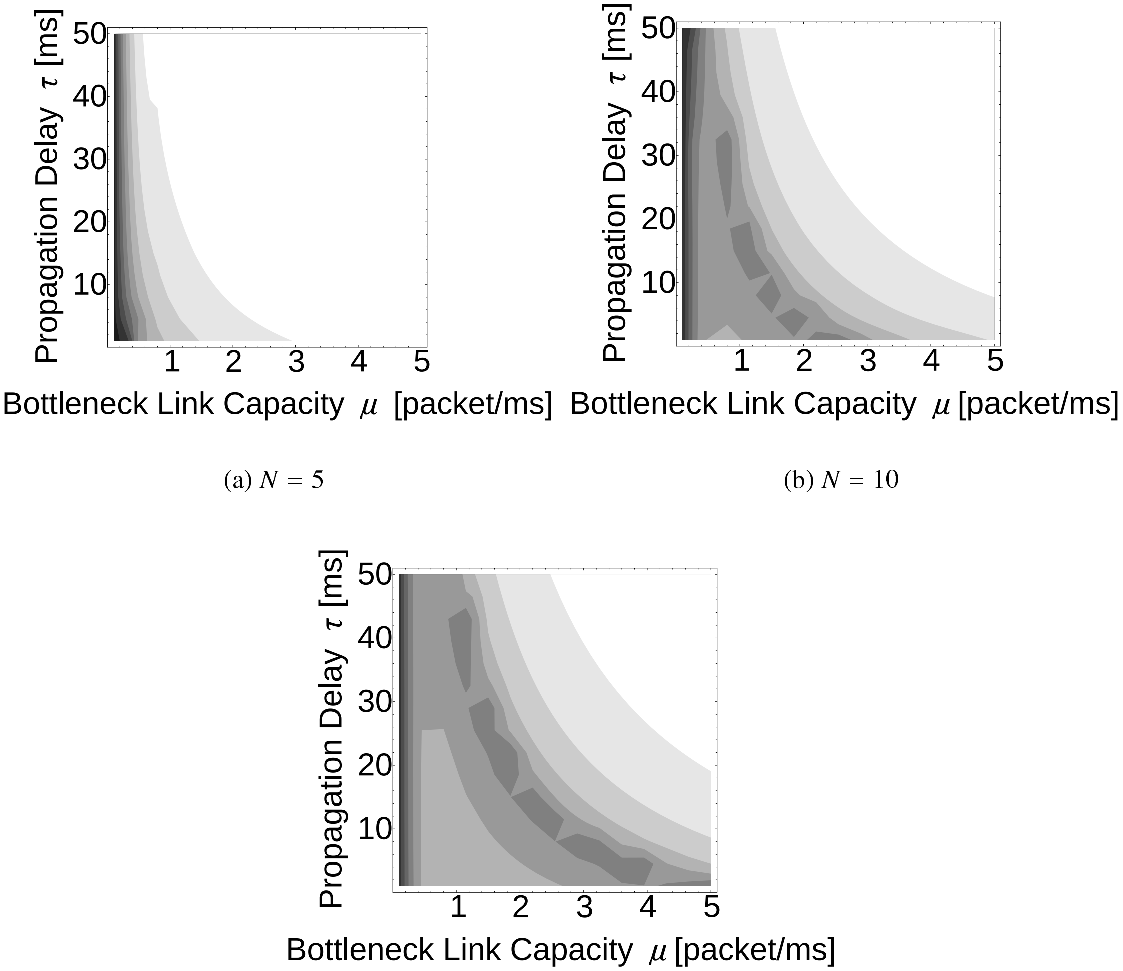
From Fig. 2.9, one can find that the larger the bottleneck link capacity is, the worse the transient state behavior becomes. We need to take the TCP transient state behavior into consideration, when we design a network of which bandwidth is large. In addition, it is clear that the bandwidth of the future network will be larger than that of the current network. It becomes more important than currently to take the TCP transient state behavior into consideration in the future.

By comparing Figs. 2.9(a)–(c), one can find that as the number of TCP connections increases, the stability region becomes large. This is because the larger the number of TCP connections becomes, the smaller the bandwidth–delay product of each TCP connection becomes. The small bandwidth–delay product means that a source host receives a small number of ACK packets which carry feedback information. As a result, the increase of the number of TCP connections has the same effect with the decrease of the feedback delay and/or the feedback gain.

Figure 2.10 shows the maximum modulus of eigenvalues for the number of TCP connections $N = 10$ and different arrival rate of the background traffic, $\lambda_B = 0$, 0.2 and 0.5 [packet/ms]. By comparing Fig. 2.10(a)–(c), one can find that the stability region becomes slightly larger, as λ_B becomes large. This is because the increase of the background traffic corresponds to the decrease of the available bandwidth to TCP connections. Namely, the decrease of the available bandwidth to TCP connections results in the smaller bandwidth–delay product, which means a little feedback gain. Because the little feedback gain makes system sensitivity to the changes of the environment low, the reduction of the available bandwidth would bring the larger stability region.

To validate our transient state behavior analysis, we next show how the TCP transient state behavior changes for different maximum moduli using simulation experiments. Figure 2.11 shows the window size and the average queue length obtained from our simulation experiments for different bottleneck link capacities $\mu = 0.5$, 2.0, and 5.0 [packet/ms]. Note that when the bottleneck link capacity μ is 0.5, 2.0, and 5.0, the maximum modulus of eigenvalues of the state transient matrix is 0.619, 0.780, and 0.923, respectively. We use the same values with Tab. 2.1 for all parameters except the bottleneck link capacity.

Using the ns-2 simulator, we run simulations 50 times at a packet level for the same network model shown in Fig. 2.1, and investigate the evolution of the average TCP window size and the average queue length. More specifically, we calculate the average TCP window

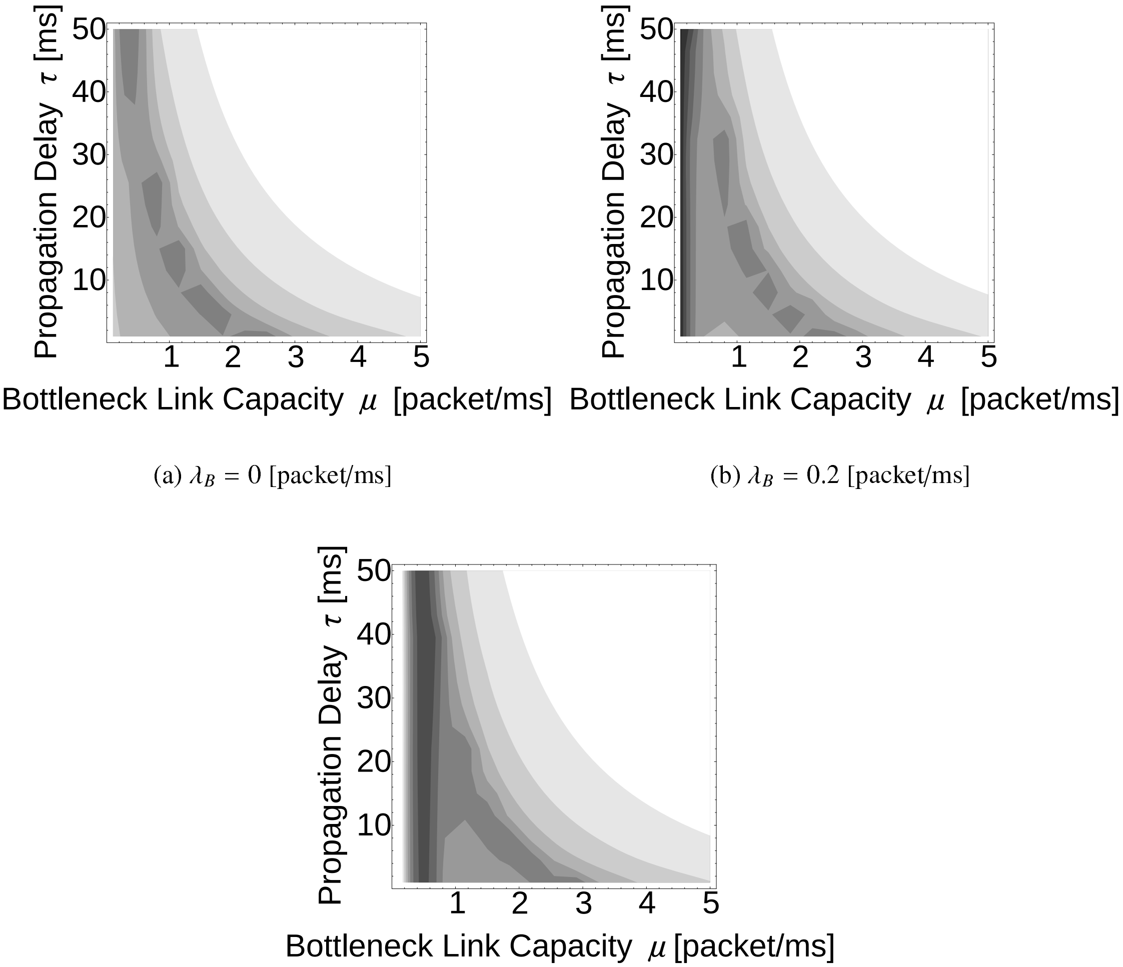


(a) $N = 5$

(b) $N = 10$

(c) $N = 15$

Figure 2.9: Maximum modulus of eigenvalues for different number of TCP connections

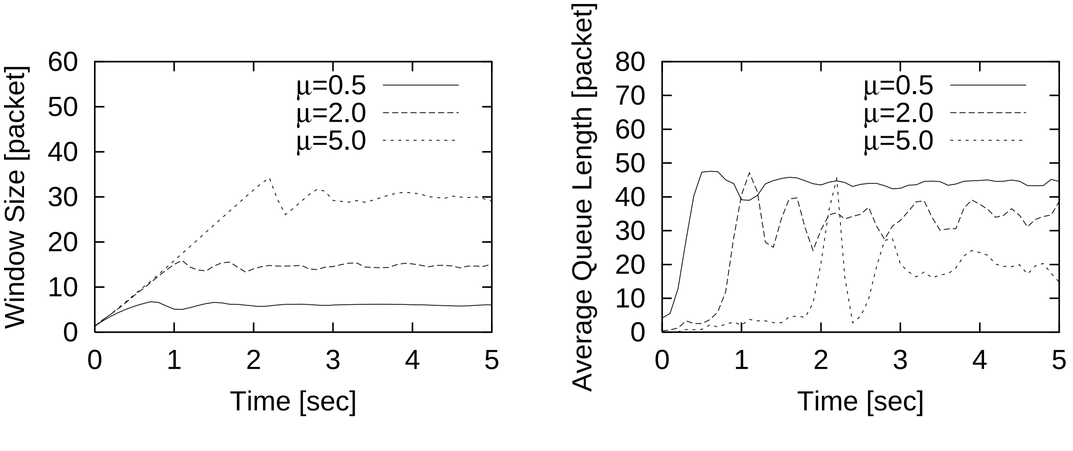


(a) $\lambda_B = 0$ [packet/ms]

(b) $\lambda_B = 0.2$ [packet/ms]

(c) $\lambda_B = 0.5$ [packet/ms]

Figure 2.10: Maximum modulus of eigenvalues for different arrival rate of background traffic



(a) TCP window size

(b) Average queue length

Figure 2.11: Simulation results for different maximum moduli

size and the average queue length every 100 ms. From this figure, one can find that the smaller the maximum modulus is (the smaller the bottleneck link capacity is), the better the transient state behavior becomes. From these observations, we conclude that our transient state behavior analysis using the control theory accurately captures the dynamics of TCP.

2.6 Conclusion

In this chapter, we have modeled both the congestion control mechanism of TCP and the network as a feedback system, and have analyzed the steady state and the transient state behaviors of TCP. We have derived the throughput of each TCP connection, the packet loss probability, and the average queue length at the bottleneck router. We have also analyzed the TCP transient state behavior by using the control theory. As a result, we have found that the bandwidth–delay product mostly determines the stability and the transient state behavior of TCP. We have also found that the network becomes stable as the number of TCP connections or the amounts of the background traffic increases. We have shown that the transient state behavior is heavily dependent on the propagation delay of the bottleneck link, but is almost independent of the amount of background traffic.

Chapter 3

Fluid-Based Analysis of a Network with DCCP Connections and RED Routers

In this chapter, we model DCCP congestion control mechanism and RED as independent discrete-time systems by using the modeling approach in [33, 37, 38]. We then analyze the steady state performance and the transient state performance of DCCP/RED. Consequently, we shown that the stability and the transient state performance of DCCP/RED degrade when the weight of the exponential weighted moving average, which is one of RED control parameters, is small. To solve this problem, by adding changes to the function with which RED determines the packet loss probability, we propose RED-IQI (RED with Immediate Queue Information), as an applications of our analytic result.

3.1 Background

In recent years, real-time applications, such as video streaming, IP telephone, TV conference, and network game, become popular rapidly by increasing speed of the network, or the rising demand for multimedia applications [3]. Generally, either UDP (User Datagram Protocol) [39] or TCP (Transmission Control Protocol) [40] has been used as a transport layer protocol for real-time applications. Since the Internet is a best-effort network where multiple users share the network bandwidth, all network applications need to have a mechanism for adapting to congestion status of the network. However, UDP is simply a protocol for datagram transfer, and does not have a mechanism for controlling network congestion. Hence, when a real-time application uses UDP as a transport layer protocol, it is necessary for the application to implement a certain congestion control mechanism at an application layer for preventing congestion collapse of the network [41].

On the contrary, TCP has a mechanism for adjusting the packet transmission rate according to the available bandwidth of the network by performing congestion control between source and destination hosts. However, TCP is a transport layer protocol originally designed for data transfer applications that can tolerate a certain amount of transmission delay [42]. Since the congestion control mechanism of TCP is the AIMD (Additive Increase and Multiplicative Decrease) window flow control, the packet transmission rate from a source host fluctuates at the time scale of approximately round-trip time. Although such fluctuation is not a problem when using TCP with non-realtime applications such as data transfer applications, it becomes a serious problem in real-time applications such as video streaming [42].

DCCP (Datagram Congestion Control Protocol) is therefore proposed as a new transport layer protocol for real-time applications [4]. DCCP performs congestion control between source and destination hosts, and an application using DCCP can choose the type of congestion control mechanisms. Currently, “TCP-like congestion control profile” [5] that performs congestion control similar to TCP, and “TFRC congestion control profile” [6] that performs congestion control similar to TFRC (TCP Friendly Rate Control) are proposed.

In the TCP-like congestion control profile, an AIMD window control is performed as with TCP [5]. The AIMD window control additively increases the window size (i.e., the number of packets that can be transmitted in a round-trip time) until a source host detects network congestion. If congestion in the network is detected, a source host multiplicatively decreases the window size. Therefore, the packet transmission rate of DCCP using the TCP-like congestion control profile fluctuates at the time scale of approximately round-trip time. Hence, for instance, DCCP with the TCP-like congestion control profile is suitable for a streaming application that buffers a large amount of data at a destination host [5].

On the contrary, in the TFRC congestion control profile, variation of the packet transmission rate caused by the TCP-like congestion control profile is prevented, and congestion control is performed so that the network bandwidth is fairly shared with other competing TCP connections [6]. In DCCP with the TFRC congestion control profile, a destination host primarily performs congestion control. Namely, in the TFRC congestion control profile, the destination host detects network congestion and notifies it of a source host. The source host adjusts the packet transmission rate from a source host based on the congestion information (e.g., *packet loss event rate*) notified from the destination host. For instance, DCCP with the TFRC congestion control profile is suitable for a streaming application that buffers a small amount of data at a destination host [6].

Whereas DCCP performs congestion control between source and destination hosts, AQM (Active Queue Management) mechanisms that perform congestion control at routers in the network have been capturing the spotlight in recent years [41, 43]. A representative AQM mechanism is RED (Random Early Detection) [7], which probabilistically discards an arriving packet. With RED, as compared with the conventional DropTail, the average queue length (i.e., the average number of packets in the buffer) of the router can be kept small, and high throughput can be achieved [7, 8]. In particular, keeping the average queue length small is effective in decreasing the end-to-end transmission delay. Hence, it is expected that an AQM mechanism is effective for real-time applications.

In the literature, many studies on the congestion control mechanism of TCP, which is adopted in the TCP-like congestion control profile of DCCP, have extensively performed [22, 23, 44, 38, 45]. In particular, characteristics of the mixed environment of TCP connections and RED routers have been extensively studied. For instance, in [33, 37, 38], the congestion control mechanism of TCP and RED are modeled as independent discrete-time systems. The entire network is then modeled as a feedback system where TCP connections and the RED router are interconnected. By applying control theory, the steady state performance and the transient state performance of the TCP congestion control mechanism and RED are analyzed. Moreover, in [22, 23, 44], the TCP congestion control mechanism and RED are modeled as independent continuous-time systems, and the steady state performance of RED is analyzed. In [45], it is shown that the transient state performance and the robustness of RED improve, when the function with which RED determines the packet loss probability is changed to a concave function to the average queue length.

Although characteristics of the mixed environment of TCP congestion control mechanism and RED have been sufficiently investigated, characteristics of the mixed environment of TFRC congestion control mechanism and RED have not been sufficiently studied [46, 42, 47, 48]. In [47], fairness between TCP-friendly rate control mechanism and TCP in steady state is evaluated with simulations and traffic measurements of the Internet. Moreover, in [42], fairness between TFRC and TCP is evaluated by simulation. The transient state performance of a TCP-friendly rate control mechanism is also evaluated. However, these studies assume that all routers are DropTail and the effect of the interaction between TFRC connections and RED routers has not been fully investigated [46, 48].

In this chapter, we therefore model DCCP congestion control mechanism and RED as independent discrete-time systems by using the modeling approach in [33, 37, 38]. We then analyze the steady state performance and the transient state performance of DCCP/RED. Specifically, we derive the packet transmission rate of DCCP connections, the packet transmission rate, the packet loss probability, and the average queue length of the RED router in steady state. Moreover, we investigate the parameter region where DCCP/RED operates stably by linearizing DCCP/RED around its equilibrium point. We also evaluate the transient state performance of DCCP/RED in terms of ramp-up time, overshoot, and settling time. Consequently, we shown that the stability and the transient state performance of DCCP/RED degrade when the weight of the exponential weighted moving average, which is one of RED control parameters, is small. To solve this problem, by adding changes to the function with which RED determines the packet loss probability, we propose RED-IQI (RED with Immediate Queue Information), as an applications of our analytic result. We analyze the transient state performance of the feedback system DCCP/RED-IQI, where DCCP connections and RED-IQI routers are interconnected. Consequently, we show that DCCP/RED-IQI has significantly better transient state performance than DCCP/RED.

3.2 DCCP (Datagram Congestion Control Protocol)

DCCP is a transport layer protocol designed for real-time applications [4]. The reliable data transfer is not guaranteed in DCCP. Namely, even if a packet is discarded in the network, a source host does not retransmit a lost packet.

In DCCP, applications using DCCP can choose a congestion control mechanism by specifying the congestion control profile. The identifier called CCID (Congestion Control IDentifier) is assigned to each congestion control profile supported by DCCP. At the time of connection establishment, source and destination hosts of DCCP exchange information on supported CCIDs, and negotiate the congestion control profile used during the data transfer. Moreover, DCCP supports ECN [49] and ECN Nonce [50], which are mechanisms by which the router explicitly notifies the congestion occurrence of the source host. Currently, CCID2 (TCP-like congestion control profile) and CCID3 (TFRC congestion control profile) are supported as congestion control profiles [5, 6].

In the TCP-like congestion control profile, the AIMD window control is performed similarly to TCP [5]. In the AIMD window control, a source host additively increases the window size (i.e., the number of packets that can be transmitted in a round-trip time) until the source host detects network congestion. If the network congestion is detected, the source host multiplicatively decreases the window size. However, the TCP-like congestion control profile of DCCP differs from TCP congestion control in the following four points.

First, congestion control of DCCP is performed also to ACK packets from a destination host to a source host using the *ACK Ratio mechanism* [5]. The transmission rate of ACK packets that a destination host returns to a source host is determined by the ACK Ratio. Specifically, when the ACK Ratio is R, the destination host of DCCP will send one ACK packet back to the source host per R data packets received from a source host.

Second, since DCCP is an unreliable transport layer protocol, a source host of DCCP does not retransmit a packet [5]. In the congestion control mechanism of TCP, when a packet is discarded, the source host identifies whether it is a retransmission packet. However, such procedure is not performed in DCCP.

Third, a destination host of DCCP can notify the cause of a packet loss of a source host [5]. This is realized by the *Data Dropped option* contained in ACK packets from a destination host to a source host. For instance, the destination host can notify it of the source host whether the packet loss is resulted from bit error of the transmission link or buffer overflow at the destination host.

Fourth, the TCP-like congestion control profile of DCCP does not perform the flow control; i.e., only the AIMD window control is performed. The buffer management of a destination host, which is performed by TCP congestion control mechanism using the advertising window, is not performed in DCCP.

On the contrary, in the TFRC congestion control profile, TCP-friendly congestion control that can fairly share bandwidth with competing TCP congestion control is performed, avoiding variation of the packet transmission rate [6]. In DCCP with the TFRC congestion control profile, congestion control is primarily performed at a destination host. Namely, in DCCP with the TFRC congestion control profile, a destination host detects network congestion, and it is notified of a source host. The source host adjusts the packet transmission rate from a source host based on the congestion information (e.g., packet loss event rate) notified from the destination host. The TFRC congestion control profile of DCCP differs from TFRC congestion control in the following point.

In the TFRC congestion control profile, a destination host can notify the cause of a packet loss of a source host [6]. This is realized similarly to the TCP-like congestion control profile using the Data Dropped option contained in ACK packets.

3.3 Modeling DCCP and RED

In this section, we model DCCP congestion control mechanism and RED as independent discrete-time systems with a time slot of Δ. We model the entire network as a single feedback system where DCCP connections and RED routers are interconnected. First, we model the congestion control mechanism of DCCP as a discrete-time system, where the input is the packet arrival rate at a destination host and the output is the packet transmission

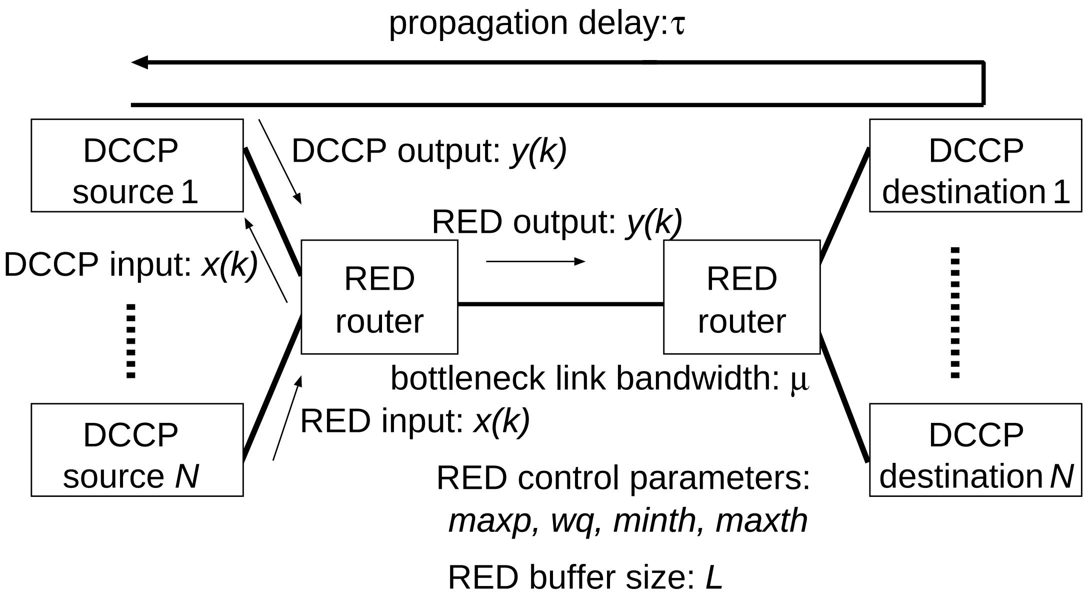


Figure 3.1: Analytic model as a feedback system consisting of DCCP connections and RED routers

rate from a source host. Next, we model RED as a discrete-time system, where the input is the packet arrival rate and the output is the packet transmission rate.

Figure 3.1 shows the analytic model used in this section. N DCCP connections share the single bottleneck link. All DCCP connections' two-way propagation delays are equal, which are denoted by τ. The bottleneck link bandwidth is denoted by μ. We denote four control parameters of RED by max_p (maximum packet loss probability), max_{th} (maximum threshold), min_{th} (minimum threshold), and w_q (weight of exponential weighted moving average). Furthermore, RED buffer size is denoted by L. Table 3.1 shows the definition of symbols used in this analysis.

In this analysis, we introduce a concept of *the packet arrival rate at a destination host* notified of a source host by ACK packets, to unify the input and the output of the models to the packet arrival/transmission rate. Since information on the arrival status of packets at a destination host is included in ACK packets, a source host can estimate the packet arrival rate at a destination host.

Note that, we assume the followings; (1) since DCCP is mainly used for real-time applications, it is assumed that a source host always has data to transfer. (2) When the packet

Table 3.1: Definition of symbols

network parameters	
N	number of DCCP connections
τ	two-way propagation delay of DCCP connection
μ	bottleneck link bandwidth
L	buffer size of RED router
Δ	time slot
DCCP parameters	
$w(k)$	window size of CCID2
t_{RTO}	retransmission timer of CCID2
$p_e(k)$	packet loss event rate of CCID3
$R(k)$	round-trip time
RED parameters	
max_p	maximum packet loss probability
min_{th}	minimum threshold
max_{th}	maximum threshold
w_q	weight of exponential weighted moving average
$q(k)$	current queue length
$\overline{q}(k)$	average queue length
$p(k)$	packet loss probability

loss probability of the network is small and DCCP congestion control works appropriately, DCCP operates in the congestion avoidance phase. Therefore, DCCP with the TCP-like congestion control profile is assumed to operate in the congestion avoidance phase.

First, we model change of the DCCP window size. The packet loss probability in the network is denoted by p, and the DCCP window size is denoted by w. Change of the DCCP window size is given by [51]

$$w \leftarrow w + (1-p)\frac{1}{w} - p\,(1 - p_{TO}(w,p))\,\frac{1}{2}\frac{4\,w}{3} - p\,p_{TO}(w,p)\left(\frac{4\,w(k)}{3} - 1\right),$$

where $p_{TO}(w,p)$ is the probability that DCCP detects the packet loss by the timeout mechanism when the window size is w and the packet loss probability is p [32]:

$$p_{TO}(w,p) \;\; = \;\; \frac{(1-(1-p)^3)\,(1+(1-p)^3\,(1-(1-p)^{w-3}))}{(1-(1-p)^w)}.$$

$p(k)$ is defined as the packet loss probability at slot k in the network, $R(k)$ the DCCP round-trip time, and $w(k)$ the DCCP window size. The packet loss probability of the network that a source host detects at slot k is given by $p(k - \frac{R(k)}{\Delta})$. Suppose that ACK packets are not discarded due to congestion on the path from a destination host to a source host, the ACK Ratio value converges to 1 [5]. Hence, the DCCP window size $w(k+1)$ at slot $k+1$ is approximately given by

$$\begin{aligned} w(k+1) \quad \simeq \quad & w(k) + \frac{w(k - \frac{R(k)}{\Delta})}{R(k)} \Delta \Big\{ (1 - p(k - \frac{R(k)}{\Delta})) \frac{1}{w(k)} \\ & - p(k - \frac{R(k)}{\Delta})(1 - p_{TO}(w(k - \frac{R(k)}{\Delta}), p(k - \frac{R(k)}{\Delta}))) \frac{2\, w(k)}{3} \\ & - p(k - \frac{R(k)}{\Delta})\, p_{TO}(w(k - \frac{R(k)}{\Delta}), p(k - \frac{R(k)}{\Delta}))(\frac{4\, w(k)}{3} - 1) \Big\}. \end{aligned} \tag{3.1}$$

The packet arrival rate at a destination host $x(k)$ is determined by the past packet transmission rate of a source host and the past packet loss probability in the network, $y(k - \frac{R(k)}{\Delta})$ and $p(k - \frac{R(k)}{\Delta})$.

$$x(k) \quad = \quad (1 - p(k - \frac{R(k)}{\Delta}))y(k - \frac{R(k)}{\Delta}) \tag{3.2}$$

Thus, the DCCP packet transmission rate is given by the following equation from change of the DCCP window size given by Eq. (3.1).

$$\begin{aligned} y(k+1) \quad \simeq \quad & f(x(k), y(k), R(k)) \\ = \quad & y(k) + \Delta \tfrac{x(k)}{y(k) R(k)^2} - \tfrac{2}{3} \Delta y(k) z(k) \{1 - p_{TO}(k)\} \\ & - \left\{ \tfrac{4}{3} y(k) - \tfrac{1}{R} \right\} \Delta z(k) p_{TO}(k), \end{aligned} \tag{3.3}$$

where $z(k) \equiv y(k - \frac{R(k)}{\Delta}) - x(k)$.

Next, we model the congestion control mechanism of DCCP with the TFRC congestion control profile as a discrete-time system. The input $x(k)$ of DCCP with the TFRC congestion control profile is the packet arrival rate at the destination host notified of the source host at slot k. Moreover, the output $y(k)$ is the packet transmission rate from a source host at slot k.

The packet loss event rate at slot k is defined by $p_e(k)$, and the DCCP connection's round-trip time $R(k)$. Suppose that the source host receives an ACK packet at slot k. In this case, the DCCP source host changes the transmission rate $y(k+1)$ at slot $k+1$ as [52]

$$y(k+1) \quad = \quad \min\left(X(p_e(k), R(k)), 2\, x(k)\right), \tag{3.4}$$

where $X(p_e(k), R(k))$ is given by

$$X(p_e(k), R(k)) \quad = \quad \frac{1}{R(k)\sqrt{\frac{2p_e(k)}{3}} + t_{RTO}\left(3\sqrt{\frac{3p_e(k)}{8}}p_e(k)(1+32p_e(k)^2)\right)},$$

where t_{RTO} is the TCP retransmission timer, and is can be approximated by $4R(k)$ [52].

Supposing that a RED router discards a packet randomly with the probability p, the packet loss event rate p_e measured by DCCP and the packet loss probability p at a RED router satisfy the following relation:

$$\frac{1}{p(k)} \quad = \quad 1 \times \sum_{i=1}^{M}\left((1-p_e(k))^{i-1}\, p_e(k)\right) + \sum_{i=M+1}^{\infty}\left(i\,(1-p_e(k))^{i-1}\, p_e(k)\right),$$

where M is the number of packets $M(= R(k)\, y(k))$ that arrive at the RED router during a round-trip time.

Finally, we model the RED router as a discrete-time system. The input $x(k)$ is the packet arrival rate at the RED router at slot k. Moreover, the output $y(k)$ is the packet transmission rate from the RED router at slot k.

We define μ as the bottleneck link bandwidth and $p(k)$ as the probability that the RED router discards packets. Since the packet arrival rate at the RED router is $x(k)$, the packet transmission rate from the RED router is given by $(1-p(k))\, x(k)$. Furthermore, since the maximum packet transmission rate from the RED router is limited by the output link bandwidth, the maximum of $y(k)$ is limited by the bottleneck link bandwidth μ. Hence, the output $y(k)$ of RED is given by [51]

$$y(k) = \min((1-p(k))\, x(k), \mu). \tag{3.5}$$

The current queue length of RED at slot k is denoted by $q(k)$, and the average queue length is denoted by $\overline{q}(k)$. When the buffer size of the RED router is L, the current queue length $q(k+1)$ at slot $k+1$ is given by [51]

$$q(k+1) \quad = \quad \min\left[\max\left\{q(k) + (x(k) - \mu)\,\Delta, 0\right\}, L\right]. \tag{3.6}$$

Let q be the current queue length of RED, and $\overline{q}$ be the average queue length of RED. RED updates the average queue length $\overline{q}$ for every packet receipt as [7]

$$\overline{q} \leftarrow (1 - w_q)\,\overline{q} + w_q\, q. \tag{3.7}$$

Since the packet arrival rate at slot k is $x(k)$, the average queue length $\overline{q}(k)$ at slot $k+1$ is approximately given by [51]

$$\overline{q}(k+1) \quad \simeq \quad \overline{q}(k) + x(k)\,\Delta\, w_q(q(k) - \overline{q}(k)). \tag{3.8}$$

RED determines the packet loss probability $p_b(k)$ from its average queue length $\overline{q}(k)$ [7] as

$$p_b(k) \quad = \quad \begin{cases} 0 & \text{if } \overline{q}(k) < min_{th} \\ \dfrac{max_p}{max_{th} - min_{th}}(\overline{q}(k) - min_{th}) & \text{if } min_{th} \leq \overline{q}(k) < max_{th} \\ 1 & \text{if } \overline{q}(k) \geq max_{th}. \end{cases} \tag{3.9}$$

Finally, the RED router discards arriving packets with the probability $p_a(k)$ determined by

$$p_a(k) = \frac{p_b(k)}{1 - count \times p_b(k)}, \tag{3.10}$$

where *count* is the number of packets arrived at the router since the last packet discarded. Since the packet loss probability $p(k)$ in the RED router is the average of $p_a(k)$, it is given by [7]

$$p(k) = \frac{2p_b(k)}{1 + p_b(k)}. \tag{3.11}$$

Note that using the current queue length $q(k)$ of RED, a DCCP connection's round-trip time at slot k is given by

$$R(k) = \frac{q(k)}{\mu} + \tau.$$

3.4 Steady State Analysis

In what follows, we analyze the steady state performance of DCCP/RED utilizing analytic models constructed in Section 3.3. Specifically, we derive the packet transmission rate of DCCP connections, the packet transmission rate, the packet loss probability, and the average queue length of RED in steady state. In Section 3.6, we will validate our approximate analysis by comparing numerical examples with simulation ones.

Since the congestion control mechanism of DCCP with the TCP-like congestion control profile is the AIMD window control, the window size oscillates when the feedback delay is not negligible. Consequently, the packet transmission rate never converges to a fixed value. Note that the output from our DCCP model with the TCP-like congestion control profile represents not an instantaneous value of the oscillating packet transmission rate, but the expected value of the packet transmission rate.

The packet transmission rate of DCCP and RED in steady state ($k \to \infty$) are denoted by y^*_D and y^*_R, respectively. Let N be the number of DCCP connections. We can numerically obtain y^*_D and y^*_R by solving equations $y(k+1) = y(k) = y^*_R$, $x(k) = \frac{y^*_R}{N}$ (Eq. (3.3)), $y(k+1) = y(k) = y^*_R$, and $x(k) = N\, y^*_D$ (Eq. (3.5)). Focusing on the input x^*_R and the output y^*_R of a RED router, we have the following relation

$$y^*_R = (1 - p^*)\, x^*_R, \tag{3.12}$$

where p^* is the packet loss probability at the RED router in steady state. We can obtain p^* by solving Eq. (3.12) for p^*. Furthermore, from Eqs. (3.9) and (3.11), we can easily obtain the average queue length $\overline{q}^*$ of the RED router.

3.5 Transient State Analysis

We then analyze the transient state performance of DCCP/RED by linearizing the discrete-time model around its equilibrium point.

First, we focus on the feedback system where DCCP connections with the TCP-like congestion control profile and RED routers are interconnected. The state of DCCP and RED is determined by the packet arrival rate $x_D(k)$ at the destination host (notified by a destination host via ACK packets) at slot k, the packet transmission rates $y_D(k) \cdots y_D(k - \frac{R(k)}{\Delta})$ from the source host, the packet arrival/transmission rate of the RED router at slot k, $x_R(k)$ and $y_R(k)$. We introduce a state vector $\mathbf{x}(k)$ that are composed of differences between each state variable at slot k and its equilibrium value:

$$\mathbf{x}(k) \equiv \begin{bmatrix} x_D(k) & - & x_D^* \\ y_D(k) & - & y_D^* \\ & \vdots & \\ y_D(k - \frac{R(k)}{\Delta}) & - & y_D^* \\ x_R(k) & - & x_R^* \\ y_R(k) & - & y_R^* \end{bmatrix}$$

We focus on state transition between slot k and slot $k+1$. Although all discrete models (Eqs. (3.1)–(3.3), (3.5)–(3.11)) in our analysis are nonlinear, they can be written in the following matrix form by linearizing them around their equilibrium values x_D^*, y_D^*, x_R^*, and y_R^*.

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k), \tag{3.13}$$

where $\mathbf{A}$ is the state transition matrix of the state vector from $\mathbf{x}(k)$ to $\mathbf{x}(k+1)$. The eigenvalues of the state transition matrix $\mathbf{A}$ determine the transient state performance (i.e., convergence performance to the equilibrium point) of the discrete-time systems given by Eqs. (3.1)–(3.3), (3.5)–(3.11). Let $\lambda_i(1 \leq i \leq \frac{R(k)}{\Delta} + 3)$ be the eigenvalues of the state transition matrix $\mathbf{A}$. The maximum absolute value of eigenvalues (*maximum modulus*) determines the stability and the transient state performance of the feedback system around its equilibrium point [53]. It is known that the smaller the maximum modulus is, the better the transient state performance becomes. It is also known that the system is stable if the maximum modulus is less than 1.0.

Next, we focus on the feedback system where DCCP connections with the TFRC congestion control profile and RED routers are interconnected. The state of DCCP with the TFRC congestion control profile and RED are determined by the packet arrival rate $x_D(k)$ at the destination host at slot k , the packet transmission rates $y_D(k) \cdots y_D(k - \frac{R(k)}{\Delta})$ from the source host, and the packet arrival/transmission rate of RED,at slot k, $x_R(k)$ and $y_R(k)$. Hence, the state vector $\mathbf{x}(k)$ that are composed of differences between each state variable at slot k and its equilibrium value is given by (3.14).

We assume that the DCCP destination host sends an ACK packet to its source host every n slots. We focus on state transition between slot k and slot $k+n$. Although all discrete models in our analysis (Eqs. (3.4)–(3.11)) are nonlinear, they can be written in the following matrix form by linearizing them around their equilibrium values x_D^*, y_D^*, x_R^*, and y_R^*.

$$\mathbf{x}(k+n) \quad = \quad \mathbf{A}\,\mathbf{B}^{n-1}\mathbf{x}(k), \tag{3.14}$$

where $\mathbf{A}$ is the state transition matrix of the state vector from $\mathbf{x}(k)$ to $\mathbf{x}(k+1)$ when the DCCP source host receives an ACK packet (Eq. (3.4)). Moreover, $\mathbf{B}$ is the state transition matrix of the state vector from $\mathbf{x}(k)$ to $\mathbf{x}(k+1)$ when the DCCP source host does not receive any ACK packet (i.e., $x(k+1) = x(k)$). $\mathbf{A}\,\mathbf{B}^{n-1}$ is the state transition matrix of the state vector from $\mathbf{x}(k)$ to $\mathbf{x}(k+n)$. The eigenvalues of the state transition matrix determine the transient state performance (i.e., the convergence performance to the equilibrium point) of the discrete-time system given by Eqs. (3.4)–(3.11).

3.6 Numerical Examples

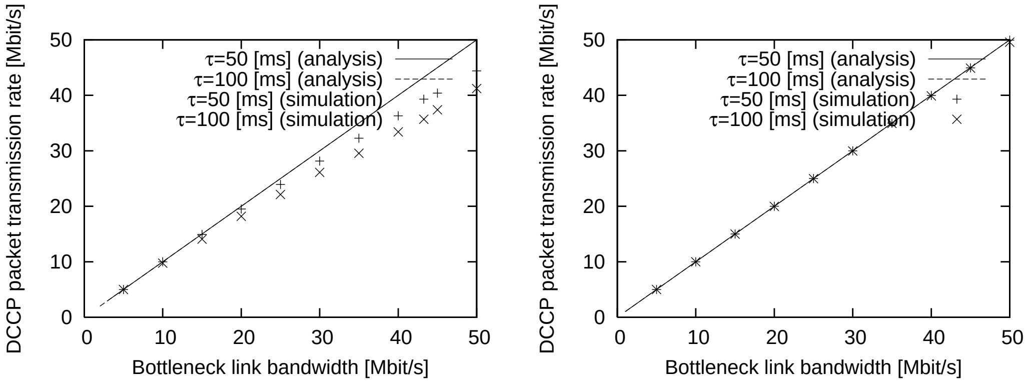
In this section, by presenting some numerical examples, we show quantitatively how the steady state performance and the transient state performance of DCCP/RED change according to the bottleneck link bandwidth and the propagation delay of the network. Furthermore, we validate our approximate analysis by comparing analytic results with simulation ones.

Unless explicitly stated, in the following numerical examples and simulations, values shown in Tab. 3.2 are used as control parameters and system parameters. We performed simulation using ns-2 for the network topology shown in Fig. 3.1. In this network, the link between two RED routers is the bottleneck, so that we focus on the packet loss probability and the average queue length of the upstream RED router. We run simulation for 150 [s] and used simulation result of the last 100 [s] for measuring DCCP connections' packet transmission rates and the packet loss probability of the RED router. We repeated simulation 10 times and measured averages of the DCCP connections' packet transmission rates and the packet loss probability of the RED router.

Table 3.2: Parameters used in numerical example and simulation

network parameters		
number of DCCP connections	N	10
two-way propagation delay of DCCP connection	τ	50, 100 [ms]
access link bandwidth		10 μ [Mbit/s]
packet length of DCCP connection		1000 [byte]
RED parameters		
maximum packet loss probability	max_p	0.1
minimum threshold	min_{th}	20 [packet]
maximum threshold	max_{th}	100 [packet]
weight of exponential weighted moving average	w_q	0.002

First, we focus on the steady state performance of DCCP/RED. We show the DCCP packet transmission rate for different settings of the bottleneck link bandwidth in Fig. 3.2.



(a) DCCP with TCP-like congestion control profile

(b) DCCP with TFRC congestion control profile

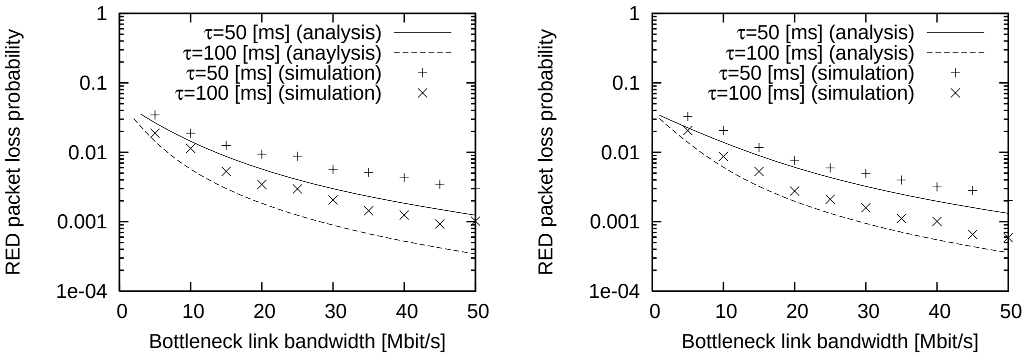
Figure 3.2: DCCP/RED steady state performance (DCCP packet transmission rate)

Here, we configure the DCCP connection's two-way propagation delay to $\tau = 50$ and $\tau = 100$ [ms]. Figure 3.2(a) shows results for DCCP with the TCP-like congestion control profile. Figure 3.2(b) shows for DCCP with the TFRC congestion control profile.

These figures indicate that the DCCP packet transmission rate increases as the bottleneck link bandwidth increases. Moreover, we compare analytic results with simulation ones. In DCCP with the TCP-like congestion control profile, some errors are observed between analytic results and simulation ones in the region where bottleneck link bandwidth is large. In other region, analytic results and simulation ones coincide closely.

We show the packet loss probability of the RED router for different settings of the bottleneck link bandwidth in Fig. 3.3. The DCCP connection's two-way propagation delay is configured to $\tau = 50$ and $\tau = 100$ [ms]. Figure 3.3(a) shows results for DCCP with the TCP-like congestion control profile. Figure 3.3(b) shows for DCCP with the TFRC congestion control profile. These figures show that the packet loss probability of RED decreases rapidly as the bottleneck link bandwidth increases. Moreover, it indicates that analytic results and simulation ones coincide with sufficient accuracy.

Next, we focus on the transient state performance of DCCP/RED. Figure 3.4 shows the maximum modulus of the state transition matrix ($\mathbf{A}$ or $\mathbf{A}\,\mathbf{B}^{n-1}$) of DCCP/RED for different settings of the bottleneck link bandwidth. Figure 3.4(a) shows results for DCCP with the



(a) DCCP with TCP-like congestion control profile

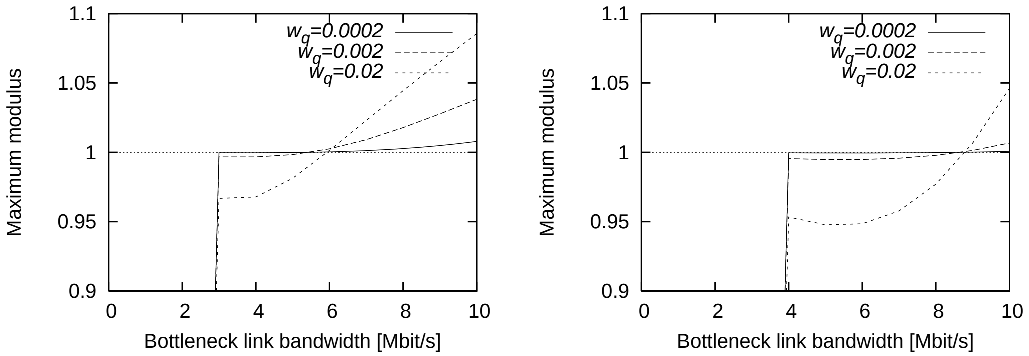
(b) DCCP with TFRC congestion control profile

Figure 3.3: DCCP/RED steady state performance (RED packet loss probability)

TCP-like congestion control profile (Eq. (3.3)). Figure 3.4(b) shows results for DCCP with the TFRC congestion control profile (Eq. (3.4)). In these figures, the weight w_q of the exponential weighted moving average of RED is configured to 0.0002, 0.002 and 0.02. Moreover, the number of DCCP connections is $N = 1$, and the two-way propagation delay of DCCP connection is $\tau = 10$ [ms].

These figures show that the maximum modulus increases as the bottleneck link bandwidth increases. This means that the transient state performance of DCCP/RED degrades as the bottleneck link bandwidth increases. Moreover, it can be found that the maximum modulus increases as the weight w_q of the exponential weighted moving average of RED becomes small. This can be explained as follows. The time for the average queue length of RED following change of the network state increases as the weight w_q of the exponential weighted moving average becomes small. Hence, it becomes slow that the packet loss probability of RED follows change of the network state. Namely, setting w_q to be a small value has the same effect with increasing the feedback delay of the entire network.

Finally, we investigate how the maximum modulus of the state transition matrix of DCCP/RED affects the transient state performance of DCCP/RED. Figure 3.5 shows the evolution of the average queue length $\overline{q}(k)$ of RED. Figure 3.5(a) shows results for DCCP with the TCP-like congestion control profile. Figure 3.5(b) shows for DCCP with the



(a) DCCP with TCP-like congestion control profile

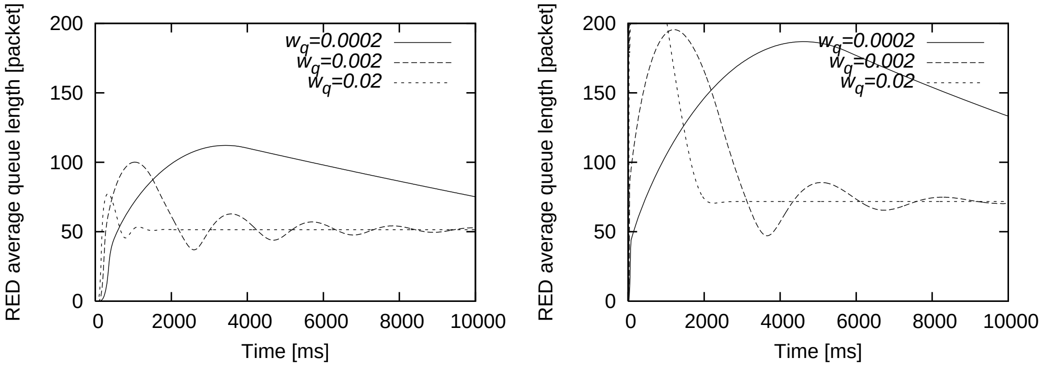
(b) DCCP with TFRC congestion control profile

Figure 3.4: DCCP/RED transient state performance (maximum modulus of the state transition matrix)

TFRC congestion control profile.

Furthermore, the average queue length $\overline{q}^*$, the maximum modulus λ of the state transition matrix of DCCP/RED, ramp-up time, overshoot and settling time are shown in Tab. 3.3. In our experiments, ramp-up time is defined as the time required for the average queue length of RED to reach 95% of the equilibrium value. Overshoot is defined as the maximum difference of the average queue length of RED from the equilibrium value. Settling time is defined as the time required for the average queue length of RED to be settled within 5% of the equilibrium value. The weight w_q of the exponential weighted moving average of RED is configured to 0.0002, 0.002 and 0.02. Moreover, the number of DCCP connections is $N = 1$, the bottleneck link bandwidth is $\mu = 4$ [Mbit/s], and the two-way propagation delay of DCCP is $\tau = 10$ [ms].

These results show that the ramp-up time and the settling time become small as the weight w_q of the exponential weighted moving average of RED becomes large. Moreover, comparison of DCCP with the TCP-like congestion control profile and DCCP with the TFRC congestion control profile indicates that each congestion control profile shows different characteristics regarding the overshoot. Namely, the overshoot of DCCP with the TCP-like congestion control profile becomes small as the weight w_q of the exponential



(a) DCCP with TCP-like congestion control profile

(b) DCCP with TFRC congestion control profile

Figure 3.5: DCCP/RED transient state performance (evolution of RED average queue length)

weighted moving average becomes large. On the contrary, overshoot of DCCP with the TFRC congestion control profile becomes large as w_q becomes large.

3.7 RED-IQI (RED with Immediate Queue Information)

In Section 3.6, in the system where DCCP connections and RED routers are interconnected, we have shown that the settling time becomes large as the weight w_q of the exponential weighted moving average becomes small.

The packet loss probability p_b of the RED router is determined by the liner function of $(\overline{q} - min_{th})/(max_{th} - min_{th})$ (Eq.3.9). We call $(\overline{q} - min_{th})/(max_{th} - min_{th})$ *queue occupancy*. Use of this function is determined without sufficiently taking account of the steady state performance and the transient state performance of RED. It is known that when the concave function is used as the function that determines the packet loss probability p_b of the RED router, the transient state performance and the robustness of RED improve [45].

Therefore, in this section, to improve the stability and transient state performance of the system where DCCP connections and RED routers are interconnected, we propose a RED-IQI (RED with Immediate Queue Information) by adding the following changes to

Table 3.3: DCCP/RED transient state performance indices

wq	profile	$\overline{q}^*$	λ	ramp-up time [ms]
0.0002	CCID2	51.443	0.9996	560
0.002	CCID2	51.443	0.9967	270
0.02	CCID2	51.443	0.9678	180
0.0002	CCID3	71.724	0.9995	380
0.002	CCID3	71.724	0.9954	40
0.02	CCID3	71.724	0.9533	30

wq	profile	overshoot [packet]	settling time [ms]
0.0002	CCID2	27.846	36140
0.002	CCID2	24.996	7960
0.02	CCID2	17.217	920
0.0002	CCID3	44.189	35320
0.002	CCID3	45.408	7200
0.02	CCID3	54.653	1960

RED.

First, we change the calculation method of the average queue length of RED. In RED-IQI, the weight of the exponential weighted moving average is set to $w_q = 1$. Thereby, the feedback delay of DCCP/RED-IQI becomes small, and the stability and the transient state performance are expected to improve. However, by configuring to $w_q = 1$, the packet loss probability of RED-IQI may sensitively fluctuate according to temporary variation of the network state. However, since the AIMD congestion control is used in the TCP-like congestion control profile, it is thought that the variation of the packet loss probability causes little performance degradation. On the other hand, since the TFRC congestion control profile smooths the packet loss event rate [52], it is thought that the variation of the packet loss probability is also causes little performance degradation.

Next, we change the function that determines the packet loss probability of RED. RED determines the packet loss probability using the linear function to the queue occupancy. In RED-IQI, we change this function to a concave function. Specifically, we change the function that determines the packet loss probability p_b to

$$p_b \quad = \quad max_p \, \mathcal{G}_\phi\left(\frac{\overline{q} - min_{th}}{max_{th} - min_{th}}\right), \tag{3.15}$$

where $\mathcal{G}_\phi(x)$ is defined as

$$\mathcal{G}_\phi(x) = \left(1 - \sqrt{1 - x^2}\right)^{\phi}. \tag{3.16}$$

$\phi(> 0)$ is a parameter determining the concavity. In order for $\mathcal{G}_\phi$ to be concave,

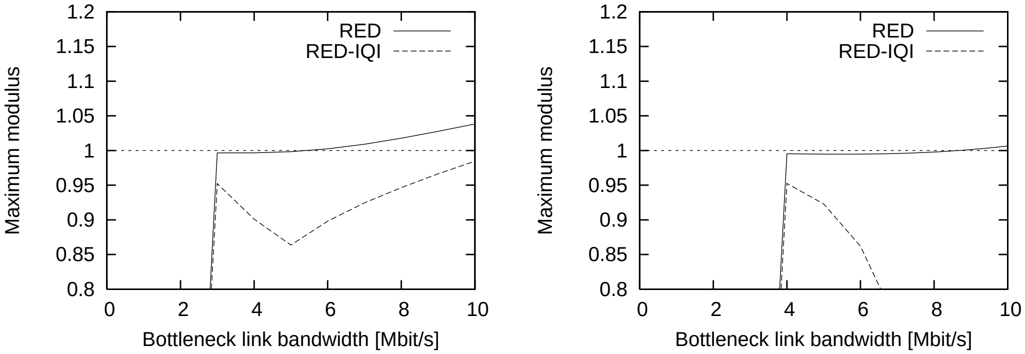
$$\begin{aligned} \frac{d^2\mathcal{G}_\phi(x)}{dx^2} &= \frac{1}{(1 - x^2)\frac{3}{2}} \left\{ \phi \left(1 - \sqrt{1 - x^2}\right)^{\phi - 2} \right. \\ &\quad \left. \times \left(1 + \sqrt{1 - x^2}\left((\phi - 1)\, x^2 - 1\right)\right) \right\} \geq 0 \end{aligned}$$

must be satisfied. By solving the above inequality for ϕ, we have

$$\phi \geq \lim_{x \to 0} \frac{-1 + \sqrt{1 - x^2} + x^2\sqrt{1 - x^2}}{x^2\sqrt{1 - x^2}} = \frac{1}{2}. \tag{3.17}$$

In what follows, by presenting several numerical examples, we show quantitatively how the transient state performance of DCCP/RED-IQI changes with the bandwidth and the propagation delay of the network. First, we focus on the transient state performance of DCCP/RED-IQI. Figure 3.6 shows the maximum modulus of the state transition matrix ($\mathbf{A}$ or $\mathbf{A}\,\mathbf{B}^{n-1}$) of DCCP/RED-IQI for different settings of the bottleneck link bandwidth. Figure. 3.6(a) shows results for DCCP with the TCP-like congestion control profile (Eq. (3.3)). Figure. 3.6(b) shows results for DCCP with the TFRC congestion control profile (Eq. (3.4)). For comparison purposes, the maximum modulus of the state transition matrix of DCCP/RED is also shown in the figure. Here, the weight w_q of the exponential weighted moving average of RED is configured to 0.002. Moreover, the number of DCCP connections is $N = 1$, and the two-way propagation delay of DCCP connection is $\tau = 10$ [ms].

It can be found that the maximum modulus of DCCP/RED-IQI increases as the bottleneck link bandwidth increases from this figure. Moreover, by comparing the maximum modulus of DCCP/RED-IQI with that of DCCP/RED, it can be found that the value of



(a) DCCP with TCP-like congestion control profile

(b) DCCP with TFRC congestion control profile

Figure 3.6: DCCP/RED-IQI transient state performance (maximum modulus of the state transition matrix)

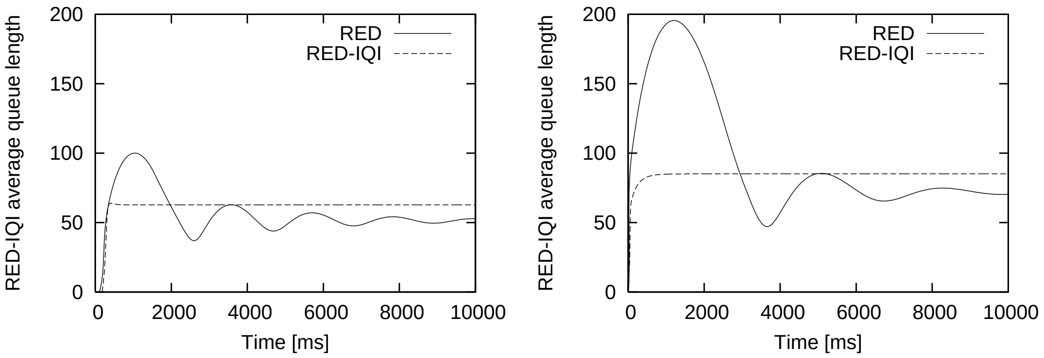
DCCP/RED-IQI is smaller than that of DCCP/RED. This means that DCCP/RED-IQI operates more stably than DCCP/RED.

Next, we show the evolution of the average queue length $\overline{q}(k)$ of RED-IQI in Fig. 3.7. Furthermore, the average queue length $\overline{q}^*$, the maximum modulus λ of the state transition matrix, ramp-up time, overshoot and settling time of DCCP/RED-IQI are shown in Tab. 3.4. For comparison purposes, the average queue length $\overline{q}^*$, maximum modulus λ of the state transition matrix, ramp-up time, and overshoot and settling time of DCCP/RED are also shown in Tab. 3.4. Here, the weight w_q of the exponential weighted moving average of RED is configured to 0.002. The number of DCCP connections is $N = 1$, the bottleneck link bandwidth is $\mu = 4$ [Mbit/s], and the two-way propagation delay of DCCP is $\tau = 10$ [ms]. Figure 3.7(a) shows results for DCCP with the TCP-like congestion control profile. Figure 3.7(b) shows results for DCCP with the TFRC congestion control profile. These results show that the overshoot and the settling time of DCCP/RED-IQI become smaller and the ramp-up time of DCCP/RED-IQI becomes larger than those of DCCP/RED.

Table 3.4: DCCP/RED and DCCP/RED-IQI transient state performance indices

	profile	$\overline{q}^*$	λ	ramp-up time [ms]
RED	CCID2	51.443	0.9996	260
RED-IQI	CCID2	62.715	0.9011	340
RED	CCID3	71.724	0.9995	40
RED-IQI	CCID3	85.057	0.9525	380

	profile	overshoot [packet]	settling time [ms]
RED	CCID2	25.00	36140
RED-IQI	CCID2	1.31	340
RED	CCID3	45.41	35320
RED-IQI	CCID3	0	380



(a) DCCP with TCP-like congestion control profile

(b) DCCP with TFRC congestion control profile

Figure 3.7: DCCP/RED-IQI transient state performance (average queue evolution of RED-IQI)

3.8 Conclusion

In this chapter, we have modeled DCCP congestion control mechanism and RED as independent discrete-time systems, and have modeled the entire network as a feedback system by interconnecting DCCP connections and RED routers. We have analyzed the steady state and transient state performance of DCCP/RED. We have derived the packet transmission rate of DCCP connections, the packet transmission rate, the packet loss probability, and the average queue length of the RED router in steady state. We have also derived the parameter region where DCCP/RED operates stably by linearizing DCCP/RED model around its equilibrium point. Furthermore, we have evaluated the transient state performance of

DCCP/RED in terms of ramp-up time, overshoot, and settling time. Consequently, we have shown that the stability and the transient state performance of DCCP/RED degrade when the weight of the exponential weighted moving average is small. By adding changes to the function with which RED determines the packet loss probability, we propose RED-IQI. We have shown that RED-IQI significantly improves the transient state performance such as the maximum modulus, the overshoot and the settling time compared with RED.

Chapter 4

Performance Analysis of Large-Scale IP Networks considering TCP Traffic

In this chapter, we propose a novel analysis method for such large-scale networks with consideration of the behavior of the congestion control mechanism of TCP. In the analysis, we model each network component (TCP end-host's and network link) as an independent system, and interconnect them into one system for analyzing the entire network. Note that we assume many TCP flows on each network link and utilize appropriate modeling methods according to the assumption. By the analysis, we derive the utilization of the network link, packet loss ratio of the link buffer, the round-trip time (RTT) and throughput of TCP connections, and the location and the degree of the network congestion.

4.1 Introduction

In recent years, the numbers of internet nodes/hosts and internet users have been increasing exponentially. For example, the number of computers connected to the Internet was about 250 million in February 2004, whereas by January 2005 it had increased to about 350 million. This means that the number of internet hosts has increased by about 40% in only 11 months [2]. Consequently, the importance of design and performance analysis techniques for large-scale networks is increasing. However, currently, there are no effective methods for analyzing such large-scale networks.

One important factor for determining the performance of the Internet is the congestion control mechanisms of TCP. One reason for this is that TCP traffic accounts for a large proportion of current internet traffic [54]. However, when considering the design and performance analysis issues of a large-scale network, the TCP congestion control mechanism, which is based on a feedback control, has been neglected. Most previous studies on large-scale network design assume that the constant-rate UDP flows as traffic demand [55-57]. For example, in [57], the authors revealed that the router-level topology of the current Internet follows a power-law distribution, as a consequence of seeking to maximize the throughput of the network subject to technological constraints on link capacities, packet processing speeds, and the number of input/output links. However, in [57], only the UDP flow with constant bit rate is considered as network traffic, while the packet loss in a network is ignored. That is, it does not include the effect of the behavior of TCP, which uses packet loss as feedback information from the network and regulates the packet transmission rate.

On the other hand, there have been some studies done on the relationship between the congestion control mechanisms of TCP and network performance [58, 59, 13]. In these studies, the authors utilized TCP traffic as network traffic and revealed in detail various characteristics on the interaction between TCP behavior and the underlying networks. However, in most of these studies, the number of TCP connections which can be treated is limited to thousands, and very simple network topologies, such as a dumbbell-type network, are used. One of the reasons for this may be the limitations of network simulators such as ns-2 [35].

There have also been many studies on methods for analyzing a large-scale network and many flows modeled using a fluid-flow approximation [60, 51, 61]. For example, in [60], a performance evaluation technique for large-scale networks using a fluid approximation model has been proposed. In [60], the congestion control mechanisms of TCP and the active queue management mechanism are modeled. In addition, the effect of the routers' packet processing speed is also modeled through explicit modeling of the order of the routers in which each TCP connection traverses. However, to the best of our knowledge, these studies are currently in the establishment phase in terms of creating analysis methods. As such, there is no means of finding out the interaction effect of a large number of TCP connections, i.e. with over 10,000 connections, and a large-scale network with over 100/1,000/1,000 routers/hosts/links.

In this chapter, we propose a novel analysis method for such large-scale networks that takes into consideration the behavior of the congestion control mechanism of TCP. In the analysis, we model each network component (end-host's TCP and network link) as an independent system, and then combine them into one system in order to analyze the entire network. Note that we assume many TCP flows on each network link and utilize appropriate modeling methods based on this assumption. Using this analysis, we can analysis a large-scale network, i.e. with over 100/1,000/10,000 routers/hosts/links and 100,000 TCP connections in substantially short time. Especially, a calculation time of our analysis, it is different from that of ns-2, is independent of a network bandwidth and/or propagation delay. Specifically, we derive the utilization of the network link, packet loss ratio of the link buffer, the round-trip time (RTT) and throughput of TCP connections, and the location and the degree of the network congestion. Consequently, we are then in a position to answer the following questions based on the analysis results: When the network traffic increases, which link will become congested? Which access networks and core networks are bottlenecks for the entire network? Which part of the network should be upgraded when we want to increase the network performance? If networking technologies in access/core networks, such as the link bandwidth and the number of input/output ports of routers, are improved, how will the congestion points of the network move (or will they remain unchanged)? Furthermore, will the end-to-end TCP throughput increase as we expect? By answering the above questions, we can use the proposed analysis method to design future high-speed and large scale networks.

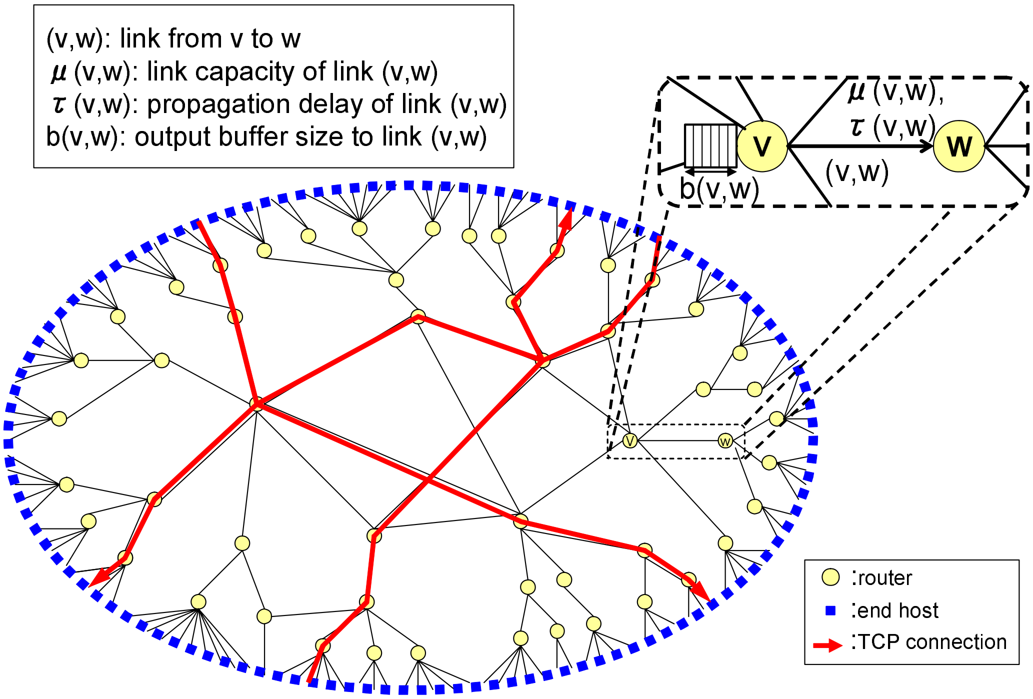

(v,w): link from v to w
μ (v,w): link capacity of link (v,w)
τ (v,w): propagation delay of link (v,w)
b(v,w): output buffer size to link (v,w)
μ (v,w),
τ (v,w)
(v,w)
v
w
b(v,w)
:router
:end host
:TCP connection

Figure 4.1: Network model in the analysis

4.2 Network and Traffic Models

In this section, we introduce the models of network and traffic used in this chapter. In the analysis, we analyze the average behavior of the entire network when there are many TCP connections present.

4.2.1 Network Model

Figure 4.1 shows the network model used in the analysis. The model consists of nodes and links, where the nodes correspond to a host or a router, and the links to links between routers and hosts. Let v and w ($v, w \in \mathcal{R}$) be nodes, where $\mathcal{R}$ is a set of nodes in a network. The ordered pair (v, w) refers to the unidirectional link from node v to node w. Note that, in this analysis, link (v, w) differs from link (w, v). Let $\mathcal{L}$ be a set of links in a network and $\mathcal{L}(\chi)$ be a set of links that the TCP connection χ traverses. The link capacity and

Table 4.1: Notations for network model

$\mathcal{R}$	set of source and destination hosts and routers
$\mathcal{L}$	set of links
$\mathcal{L}(\chi)$	set of links that TCP connection χ traverses
$\mu_{(v,w)}$	capacity of link (v, w)
$\tau_{(v,w)}$	propagation delay of link
$b_{(v,w)}$	output link buffer size to (v, w) at node v
C	set of TCP connections
$C(v, w)$	set of TCP connections traversing link (v, w)

propagation delay of link (v, w) are denoted by $\mu_{(v,w)}$ and $\tau_{(v,w)}$, respectively. In this analysis, each router is assumed to have separate output buffers for each outgoing link. The buffer size of the output link buffer to link (v, w) at node v is denoted by $b_{(v,w)}$.

TCP connections are established between end hosts according to the amount of traffic defined in Section 4.2.2. C is a set of TCP connections. After determining the route which each TCP connection traverses, we can determine $C(v, w)$, which is a set of TCP connections that traverse link (v, w). In the numerical example in Section 4.4 we use Dijkstra's shortest path algorithm for determining the route which each TCP connection traverses. Note that we could apply any kind of routing algorithm. For example, we could evaluate the effect of the overlay routing algorithm by applying the algorithm to the TCP connections which join the overlay network. We summarize the notations employed in the network model in Table 4.1.

In this chapter, we use a Drop-Tail discipline at a router buffer, and focus on the average behavior of queue occupancy at the router buffer. Note that we can apply other kinds of queuing disciplines, such as Random Early Detection (RED) and a mixture of multiple disciplines, by applying the appropriate model to the router buffer. For example, for RED discipline, we can use the existing model in [13].

4.2.2 Traffic Model

The amount of network traffic is determined using the "gravity model" [62]. By applying the basic gravity model, we assume that the amount of traffic from router v to router w

is proportional to the product of the amount of traffic that enters the network at router v and the amount of traffic that leaves the network at router w. In this analysis, we assume that the network traffic is generated from the router to which the end host is connected. In what follows, we call the router an "edge router". We also assume that the amount of traffic injected into/leaving from the edge router is proportional to the number of end hosts connected to the edge router. Finally, the number of TCP connections between edge routers is taken to be proportional to the amount of traffic between the edge routers. The number of TCP connections that traverse from edge router v to w is defined as

$$N_{(v,w)} = \alpha \times E_v \cdot E_w, \tag{4.1}$$

where E_v and E_w are the numbers of end hosts connected to the edge routers v and w, respectively, and α is a parameter for determining the overall amount of network traffic.

In this chapter, for the sake of simplicity, we employ the TCP Reno version for TCP traffic. Note that we can easily treat other versions of TCP by using appropriate models for TCP throughput. Moreover, we can also analyze a network which has different TCP versions in the same network. Hereafter, TCP Reno is simply denoted as TCP unless noted otherwise.

4.3 Analysis

In the analysis, we first model a TCP and a network link as independent systems. We then combine them into an entire network system and create simultaneous equations. By solving the equations, we can derive various network characteristics, such as the window size and throughput of TCP connections, the buffer occupancy and the packet loss ratio of network links. We also propose a method for decreasing the complexity of the simultaneous equations by removing links which do not cause congestion.

4.3.1 Modeling of TCP Behavior

We focus on the average behavior of a TCP connection, which varies the average window size depending on the packet loss ratio. That is, we model a TCP connection as a system with one input (packet loss ratio) and one output (average window size). Given a packet loss ratio d_χ and a RTT r_χ of a TCP connection χ, λ_χ, the average throughput of a TCP connection, can be calculated using the following result [32];

$$\lambda_\chi = \frac{1}{r_\chi\left(\sqrt{\frac{2d_\chi}{3}} + 6\sqrt{\frac{3b\,d_\chi}{2}}d_\chi(1+32d_\chi^2)\right)}, \tag{4.2}$$

where b is the number of required data packets for a TCP receiver to generate one ACK packet, and T_o is the initial value of the TCP retransmission timeout. By applying $b = 1$ and $T_o = 4\,r_\chi$ [52], the average size of the congestion window of TCP connection χ, denoted by w_χ, can be given by;

$$w_\chi = \frac{1}{\sqrt{\frac{2p_\chi}{3}} + 6\sqrt{\frac{3b\,p_\chi}{2}}p_\chi(1+32p_\chi^2)}. \tag{4.3}$$

Let $q_{(v,w)}$ and $d_{(v,w)}$ be the number of packets in the output link buffer and the packet loss ratio at link (v, w), respectively. Then, we can derive the packet loss ratio for TCP connection χ, denoted by d_χ, as follows;

$$d_\chi = 1 - \prod_{(v,w)\in\mathcal{L}(\chi)} (1 - d_{(v,w)}), \tag{4.4}$$

We can also derive r_χ and τ_χ, which are the RTT of the TCP connection χ and the round-trip propagation delay of the TCP connection r_χ which does not include the queuing delay at traversing links, respectively, as follows;

$$r_\chi = \tau_\chi + \sum_{(v,w)\in\mathcal{L}(\chi)} \frac{q_{(v,w)}}{\mu_{(v,w)}} \tag{4.5}$$

Table 4.2: Notations for TCP model

w_χ	congestion window size of TCP connection χ
τ_χ	round trip propagation delay of TCP connection χ
r_χ	RTT of TCP connection χ
d_χ	packet loss ratio of TCP connection χ
λ_χ	throughput of TCP connection χ
$d_{(v,w)}$	packet loss ratio at output buffer of link (v, w)
$q_{(v,w)}$	number of packets in output link buffer of link (v, w)

$$\tau_\chi = \sum_{(v,w)\in\mathcal{L}(\chi)} \tau_{(v,w)} \tag{4.6}$$

We summarize the notations used in this subsection in Table 4.2.

4.3.2 Modeling of Network Link

We focus on the behavior of a network link when TCP connections, which have certain values of congestion window size, traverse the link. Therefore, the network link is modeled as a system with one input (window sizes of TCP connections) and one output (packet loss ratio).

In [63], the authors have revealed the following characteristic on TCP connections traversing a link: when the number of TCP connections is sufficiently large and the TCP connections do not behave in a synchronized fashion, the sum of the congestion window size of the TCP connections follows a normal distribution. Since we are interested in large-scale networks having a large number of TCP connections, we utilize the above characteristics. Then, we can calculate $d_{(v,w)}$, the packet loss ratio at the buffer of link (v, w), as follows;

$$\begin{aligned} d_{(v,w)} &= Prob[q_{(v,w)} > b_{(v,w)}] \\ &= 1 - \frac{1}{2} Erf\left(\frac{b_{(v,w)} - q_{(v,w)}}{\sigma(q_{(v,w)})}\right), \end{aligned} \tag{4.7}$$

where $\sigma(q_{(v,w)})$ is the standard deviation of the distribution of the number of packets in the output link buffer of link (v, w), and $Erf()$ is the error function. The analysis in [63]

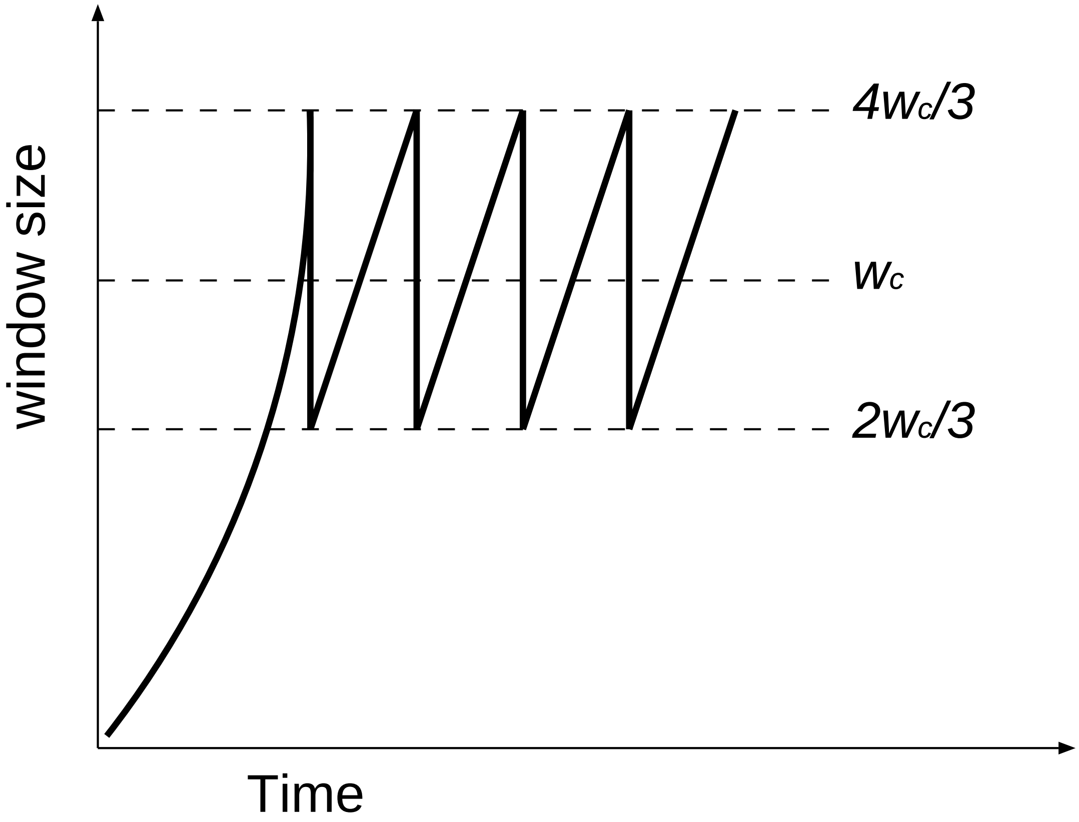


Figure 4.2: Evolution of TCP congestion window

assumes that the standard deviation of the distribution of the number of packets in the output link buffer is identical to that of the sum of the congestion window size of the TCP connections traversing the link. We therefore derive $d_{(v,w)}$ based on this assumption as follows.

Figure 4.2 depicts the typical change in the congestion window size of a TCP connection. By assuming that the TCP connection is always in the congestion avoidance phase (this assumption is reasonable when the packet loss ratio is small), we can regard the variation of the congestion window size as a uniform distribution with a lower limit of $2w_\chi/3$ and an upper limit of $4w_\chi/3$, where w_χ is the average size of the congestion window of the TCP connection. Consequently, we can obtain the standard deviation of the window size of the TCP connection as follows;

$$\sigma(w_\chi) \quad = \quad \frac{w_\chi}{3\sqrt{3}}.$$

By assuming that the distributions of the window size of all TCP connections are independent and identical, we can determine the standard deviation of the distribution of the sum of the window size of the TCP connections traversing link (v, w) by using the following equation;

$$\sigma\left(\sum_{\chi \in C(l_{v,w})} w_\chi\right) = \sigma\left(\sqrt{\sum_{\chi \in C(l_{v,w})} \sigma(w_\chi)^2}\right). \tag{4.8}$$

In addition, we utilize the assumption that when link (v, w) is congested, the sum of the throughput of TCP connections traversing link (v, w) becomes the link capacity $\mu_{(v,w)}$;

$$\mu_{(v,w)} = \sum_{\chi \in C(v,w)} \lambda_\chi. \tag{4.9}$$

4.3.3 Connecting Systems and Analysis

We regard Equations (4.2) – (4.6) and ((4.8) – (4.9)) as simultaneous equations, and solve them for w_χ, $d_{(v,w)}$, and $q_{(v,w)}$. We then obtain the window size and throughput of each TCP connection, the number of packets in the output link buffer and the packet loss ratio at each network link. The straightforward nature of the analysis is one of the advantages of our analysis method.

4.3.4 Reduction of Analysis Model

In the actual network, the number of congested links is not as large as the total number of links. The number of packets in the buffer and the packet loss ratio of uncongested links become zero. By removing such uncongested links from the analysis calculation, we can reduce the calculation time. In our analysis, we utilize the following method for reducing the number of links from the analysis. Note that the following method is based on the similar method in [60], but we have extended the method to accommodate TCP traffic.

1. Calculate the “maximum” throughput of each TCP connection.

A maximum throughput $max\lambda_{\chi}$ of TCP connection χ traverses link (v, w), where v is the source node of TCP connection χ and is defined as follows;

$$max\lambda_{\chi} = \mu_{(v,w)} \times \frac{\frac{1}{\tau_{\chi}}}{\sum_{\psi \in C(v,w)} \frac{1}{\tau_{\psi}}}.$$

2. Calculate the "maximum" amount of traffic of each network link.

   We take the maximum amount of traffic $maxT_{(v,w)}$ of link (v, w) to be the sum of the maximum throughput $\lambda_{\psi}(\psi \in C(v, w))$ of TCP connections traversing the link,

$$maxT_{(v,w)} = \sum_{\psi \in C(v,w)} max\lambda_{\psi}$$

3. Compare the maximum amount of traffic with the bandwidth at each link.

   case 1 $\forall(v, w) maxT_{(v,w)} \leq \mu_{(v,w)}$.

   There is no congestion at links which satisfy $maxT_{(v,w)} < \mu_{(v,w)}$. We remove these uncongested links from the analysis model and reduce the complexity of the model.

   case 2 $\exists(v, w) maxT_{(v,w)} > \mu_{(v,w)}$.

   We change the maximum throughput of the TCP connections. We focus on link (v, w), which has the maximum difference between the maximum amount of traffic $maxT_{(v,w)}$ and the bandwidth $\mu_{(v,w)}$ of link (v, w). Then, we change the maximum throughput of TCP connections as follows. Let $\alpha_{\chi}(\chi \in C(v, w))$ be the maximum throughput of TCP connections. We divide the link capacity $\mu_{(v,w)}$ by the inverse ratio of propagation delays of TCP connections traversing (v, w). We denote them as $\beta_{\chi}(\chi \in C(v, w))$. In the case of $\alpha_{\chi} \leq \beta_{\chi}$, we assign β_{χ} to α_{χ}. Then, we repeat Step 2.

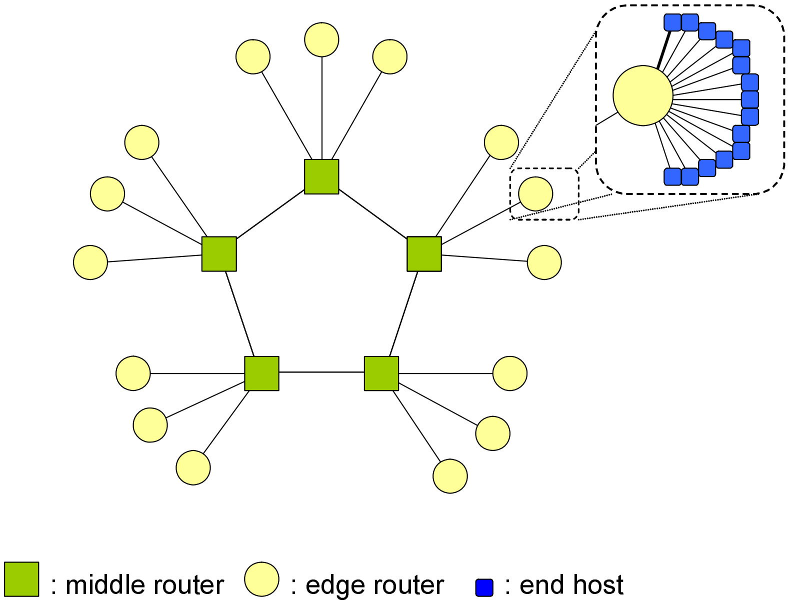


Figure 4.3: Network model for testing the accuracy of the analysis method

4.4 Numerical Examples

In this section, we verify the accuracy of the analysis method by comparing the analysis and simulation results. We then show analytic results for large-scale networks and demonstrate the ability of the proposed analysis method.

4.4.1 Accuracy of Analysis Method

We use the network model depicted in Fig. 4.3 for assessing the accuracy of the analysis method. The network topology consists of "middle routers", "edge routers" and "end hosts". For simplicity, we denote links between middle routers as l_{mm}, those between middle routers and edge routers as l_{me}, and those between edge routers and end hosts as l_{ee}.The bandwidth, propagation delay, and output link buffer size were set to the values shown in Tab. 4.3. We set α in Eq. 4.1 to $2/45$. In this setting, the total number of TCP connections

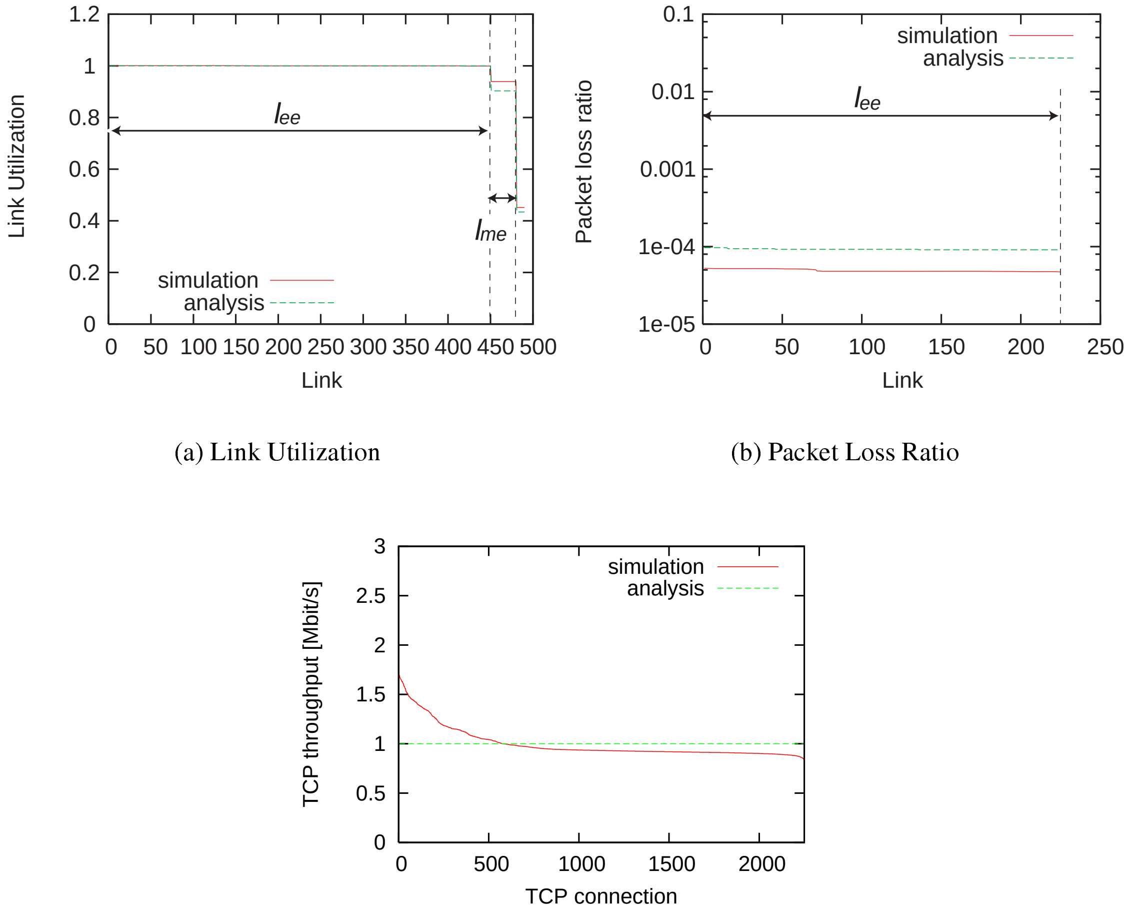
Table 4.3: Parameter settings (1)

Link	Bandwidth	Prop. Delay	Buffer Size
l_{mm}	622 [Mbit/s] (OC12)	5 [ms]	1,043 [packet]
l_{me}	155 [Mbit/s] (OC3)	5 [ms]	850 [packet]
l_{ee}	10 [Mbit/s]	10 [ms]	1,500 [packet]

in the network becomes $2,250$. The number of TCP connections between edge routers is determined by the gravity-model introduced in Section 4.2.2. To obtain the simulation results, we utilized an ns-2 simulator and conducted the simulation using the same network model as of the analysis. The simulation time was 1,050 [s], and we omit the results for the initial 50 [s] to avoid the effect of unstable behavior at the beginning of the simulation. The packet size was set to 1,000 [bytes]. Our experiment is carried on a Dell powerEdge 1850, which has two Intel Xeon processors (3.80GHz) and 4GB memory. Note that we can not conduct the ns-2 simulation for the larger scale networks than in Fig. 4.3.

Figures 4.4 and 4.5 show the analytic and simulation results when the bandwidth of the access links is set to 10 and 20 [Mbit/s], respectively. Figures 4.4(a) and 4.5(a) plot the utilization of the links. Figures 4.4(b) and 4.5(b) show the packet loss ratio of links having non-zero packet loss ratio and Figs. 4.4(c) and 4.5(c) show the throughput of TCP connections in the network. In the simulation results of these figures, we plot the link utilization and the packet loss ratio and the TCP throughput in decreasing order of their value. In the analytic results of these figures, we plot them in the same order as those of the the analytic results to compare the analytic result with the simulation ones. We also add which type of the link gives the corresponding results in Figs. 4.4(a),(b), and 4.5(a),(b).

It can be found that the analytic results results give the close estimation of the simulation results. In particular, in respect to the utilization of the links, it can be found that our analytic results show very good agreement with the simulation results. However, in terms of the TCP throughput, it can be found that our analytic results are different from simulation results especially when the TCP throughput of the simulation results is comparatively large (Figure 4.4(c)). This is because of the deviation of packet loss occurrences among TCP connections in simulation. That is, packets of “lucky TCP connections” are seldom



(a) Link Utilization

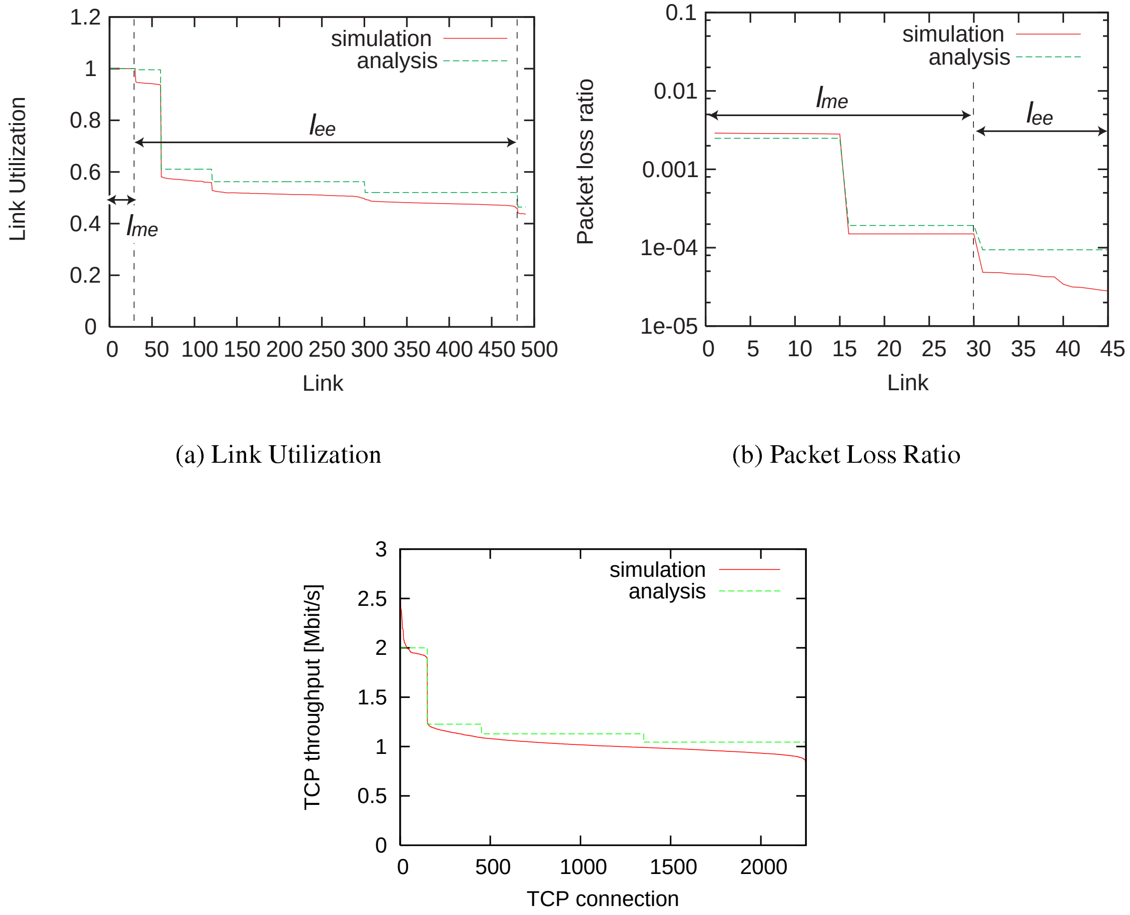
(b) Packet Loss Ratio

(c) TCP Throughput

Figure 4.4: Access link bandwidth = 10 [Mbit/s]

lost, and many packets of "unlucky TCP connections" are lost, which brings the fluctuation of the throughput of TCP connections. In other words, the 1,050 [s] of the simulation time is small to obtain the average throughput of TCP connections in this situation. This is one of the shortcomings of the simulation, whereas the analysis method in this chapter directly gives the result of the average TCP throughput.

From Fig. 4.4, we can see that when the bandwidth of the l_{ee} is 10 [Mbit/s], the bottleneck point in the network is the link l_{ee}. Note that our analytic results and simulation ones show that the same links are bottlenecks. From Figs. 4.5, we can see that when the bandwidth of the l_{ee} is 20 [Mbit/s], the bottleneck point in the network moves to the link l_{me}, Note that our analytic and simulation results find the same links as bottleneck links. From



(a) Link Utilization

(b) Packet Loss Ratio

(c) TCP Throughput

Figure 4.5: Access link bandwidth = 20 [Mbit/s]

the above results, we conclude that our analysis can precisely determine the bottleneck point precisely.

4.4.2 Analysis Results of Large-Scale Network

In this subsection we give examples of the analytic results on the large-scale network. Figure 4.6 shows the network model used in the analysis. This network topology was created according to the characteristics of the actual router-level topology, which was described in [57], where the core routers have a smaller number of links with higher bandwidth, whereas the edge routers have a larger number of links with lower bandwidth. The network

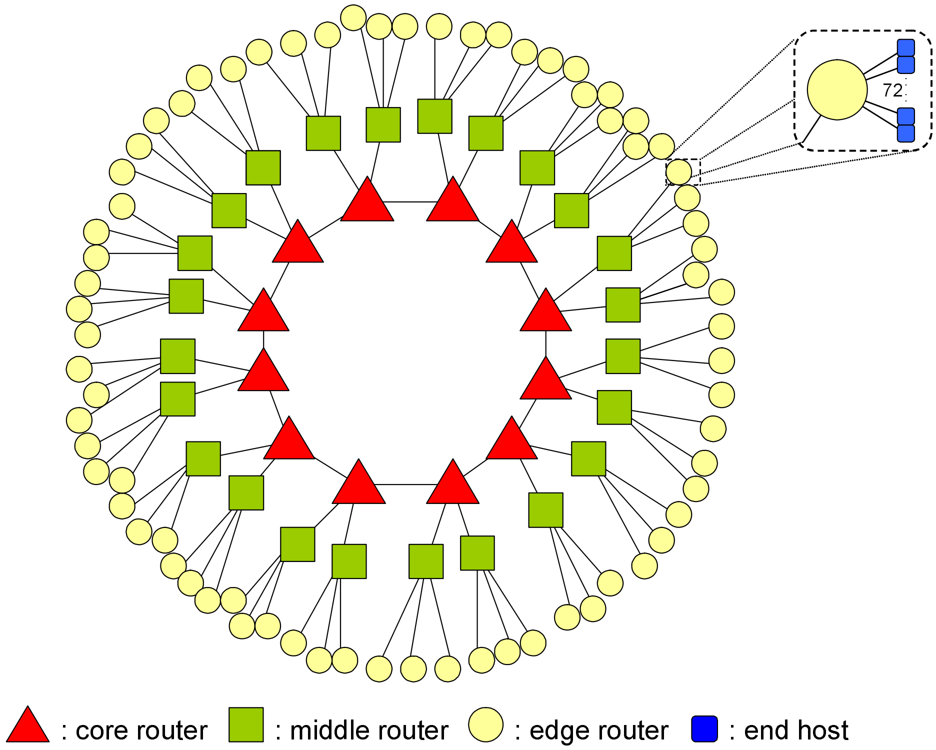


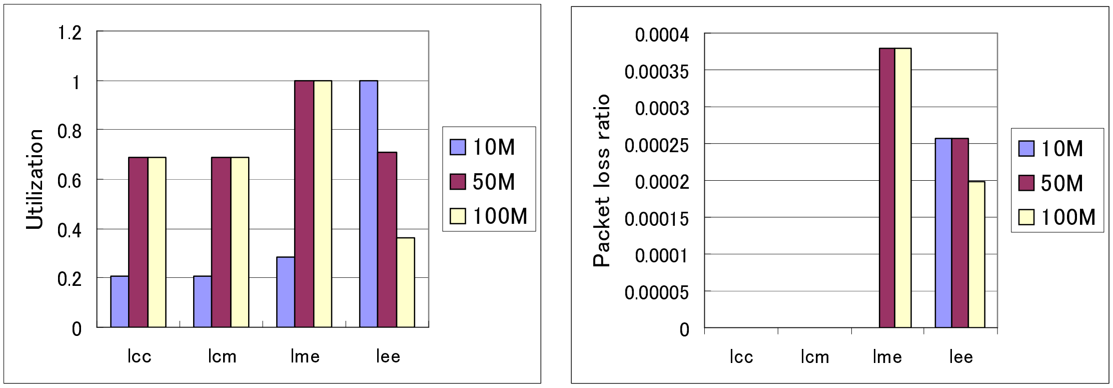
Figure 4.6: Network model for analysis of large-scale network

topology consists of “core routers”, “middle routers”, “edge routers” and “end hosts”. For simplicity, we denote links between core routers as l_{cc}, those between core routers and middle routers as l_{cm}, those between middle routers and edge routers as l_{me}, and those between edge routers and end hosts as l_{ee}. The bandwidth, propagation delay, and output link buffer size were set to the values shown in Tab. 4.4. Note that the buffer sizes of all links except l_{ee} were set according to the guidelines given in [63], where the number of connections at l_{cc}, l_{cm}, and l_{me} was assumed to be 10,000, 1,000, and 1,000, respectively, and the average RTT of TCP connection was assumed to 150 (ms). We set α in Eq. 4.1 to $20/5,184$. In this setting, the total number of TCP connections in the network becomes $103,680$. The number of TCP connections between edge routers is determined by the gravity-model as in the previous subsection. It is unable for the ns-2 simulator to carry out the simulation of this scale of network.

Figure 4.7 shows the analytic results when the bandwidth of l_{ee}, which is the access

Table 4.4: Parameter settings (2)

Link	Bandwidth	Prop. Delay	Buffer Size
l_{cc}	10 [Gbit/s] (OC192)	15 [ms]	3,750 [packet]
l_{cm}	2.5 [Gbit/s] (OC48)	5 [ms]	2,370 [packet]
l_{me}	1 [Gbit/s] (GE)	5 [ms]	1,185 [packet]
l_{ee}	10 [Mbit/s]	10 [ms]	1,500 [packet]



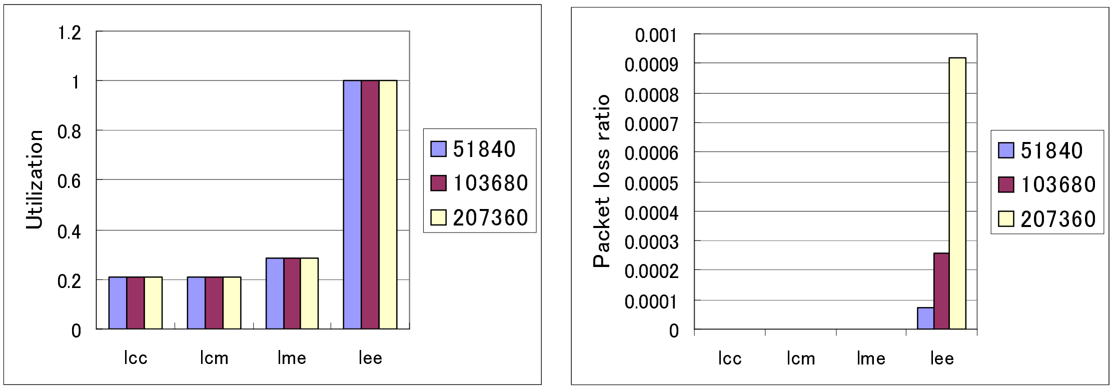
(a) Link Utilization

(b) Packet Loss Ratio

Figure 4.7: Effect of access link bandwidth

link bandwidth, had the values 10, 50, and 100 [Mbit/s]. Figures 4.7(a) and (b) plot the average link utilization and the average packet loss ratio of the links l_{cc}, l_{cm}, l_{me} and l_{ee}, respectively. From Fig. 4.7(a), we can observe that the utilization of the links inside the network (l_{cc}, l_{cm}, and l_{me}) increases as the bandwidth of l_{ee} increases. Furthermore, we can see that when the bandwidth of the link l_{ee} is 10 [Mbit/s], the bottleneck point in the network is the link l_{ee}, but when it is increased to 50 [Mbit/s], the bottleneck point moves to the link l_{me}. This movement of the bottleneck point is confirmed by Fig. 4.7(b), where the packet loss probability of l_{me} is more than that of l_{ee} when the l_{ee} is 50 [Mbit/s]. These results mean that by increasing the access link bandwidth, the capacity of the core network becomes comparatively small. We can see such a situation easily by using the analytic results, without the need for time-consuming simulation experiments.

We next show the results when we vary the number of TCP connections in the network to 51,840, 103,680, and 207,360, by changing α to $10/5,184$, $20/5,184$, and $40/5,184$,



(a) Link Utilization

(b) Packet Loss Ratio

Figure 4.8: Effect of the number of TCP connections

respectively. We set the bandwidth of the link l_{ee} to 10 [Mbit/s]. Figures 4.8(a) and 4.8(b) show the average link utilization and the average packet loss ratio of the links l_{cc}, l_{cm}, l_{me}, and l_{ee}. We can determine the following network characteristics of the networks from the analytic results. The link utilization remains unchanged when the number of TCP connections changes. This clearly shows the greedy nature of the congestion control mechanism of TCP: TCP always tries to fully utilize the link bandwidth when the receive socket buffer size is large enough. The effect of increasing the number of TCP connections is found on the packet loss ratio, shown in Fig. 4.8(b). This again demonstrates the nature of TCP connections. From these results, we have confirmed that our analysis method can describe the behavior of TCP connections in a large-scale network appropriately.

4.5 Conclusion

In this chapter, we have proposed a novel analysis method for such large-scale networks with consideration of the behavior of the congestion control mechanism of TCP. In the analysis, we have modeled each network component (end-host's TCP and network link) as a independent system, and interconnect them into one system for analyzing the entire network. By the analysis, we have derived the utilization of the network link, packet loss ratio of the link buffer, the round-trip time and throughput of TCP connections, and the location and the degree of the network congestion. By showing some numerical examples, we have shown that our analysis method can treat the behavior of TCP connection in the large-scale network appropriately.

In recent years, data transmission between the end-hosts may be carried out via overlay nodes by dividing the end-to-end TCP connection into multiple split TCP connections. It would be capable to apply our proposed analysis method to design a overlay network. For future work, we plant to resolve the “location of overlay nodes problem” and “path between overlay nodes choice problem” by the analysis method in this chapter. We use Dijkstra’s shortest path algorithm for determining the route which each TCP connection traverses. It would be interesting to use other routing algorithms and show how the end-to-end TCP throughput changes. For instance, evaluating TCP throughput when using ETR (Estimated-TCP-throughput Maximization based Routing) algorithm [64] for determining the route in a large-scale network would be interesting. It would be important to analyze a network which has new TCP variants proposed for high-speed and large-delay networks. By using our proposed analysis method, we can investigate the influence of such TCP variants to a large-scale network; how will the congestion points of the network move by introducing such TCP variants ? Our analysis can be easily applied to such a situation. In case of HSTCP [65], for example, we can easily model the throughput of an HSTCP by extending the approach proposed in [32].

Chapter 5

Conclusion

In this thesis, we have foucused on the feedback-based congestion control mechanisms in the Internet, and have analyzed the congestion control mechanisms in the Internet using the control theoretic approach.

In Chapter 2, we have modeled both the congestion control mechanism of TCP and the network as a feedback system, and have analyzed the steady state and the transient state behaviors of TCP. We have derived the throughput of each TCP connection, the packet loss probability, and the average queue length at the bottleneck router. We have also analyzed the TCP transient state behavior by using the control theory. As a result, we have found that the bandwidth–delay product mostly determines the stability and the transient state behavior of TCP. We have also found that the network becomes stable as the number of TCP connections or the amounts of the background traffic increases. We have shown that the transient state behavior is heavily dependent on the propagation delay of the bottleneck link, but is almost independent of the amount of background traffic.

In Chapter 3, we have modeled DCCP congestion control mechanism and RED as independent discrete-time systems, and have modeled the entire network as a feedback system by interconnecting DCCP connections and RED routers. We have analyzed the steady state and transient state performance of DCCP/RED. We have derived the packet transmission rate of DCCP connections, the packet transmission rate, the packet loss probability, and the average queue length of the RED router in steady state. We have also derived the parameter region where DCCP/RED operates stably by linearizing DCCP/RED model around its equilibrium point. Furthermore, we have evaluated the transient state performance of DCCP/RED in terms of ramp-up time, overshoot, and settling time. Consequently, we have shown that the stability and the transient state performance of DCCP/RED degrade when the weight of the exponential weighted moving average is small. By adding changes to the function with which RED determines the packet loss probability, we have proposed RED-IQI. We have shown that RED-IQI significantly improves the transient state performance such as the maximum modulus, the overshoot and the settling time compared with RED.

In Chapter 4, we have proposed a novel analysis method for such large-scale networks with consideration of the behavior of the congestion control mechanism of TCP. In the analysis, we have modeled each network component (end-host's TCP and network link) as a independent system, and interconnect them into one system for analyzing the entire network. By the analysis, we have derived the utilization of the network link, packet loss ratio of the link buffer, the round-trip time and throughput of TCP connections, and the location and the degree of the network congestion. By showing some numerical examples, we have showed that our analysis method can treat the behavior of TCP connection in the large-scale network appropriately.

Bibliography

[1] V. Jacobson and M. J. Karels, “Congestion avoidance and control,” in *Proceedings of SIGCOMM ’88*, pp. 314–329, Nov. 1988.

[2] “Hobbes’ Internet timeline v8.1.” available at `http://www.zakon.org/robert/internet/timeline/`.

[3] E. Kohler, M. Handley, and S. Floyd, “Designing DCCP: Congestion control without reliability,” tech. rep., ICSI CENTER for Internet Research, 2003.

[4] E. Kohler, M. Handley, and S. Floyd, “Datagram congestion control protocol (DCCP),” *Internet Draft* <`draft-ietf-dccp-spec-11.txt`>, Mar. 2005.

[5] E. Kohler, M. Handley, and S. Floyd, “Profile for DCCP congestion control ID 2: TCP-like congestion control,” *Internet Draft* <`draft-ietf-dccp-ccid2-10.txt`>, Mar. 2005.

[6] E. Kohler, M. Handley, and S. Floyd, “Profile for DCCP congestion control ID 3: TFRC congestion control,” *Internet Draft* <`draft-ietf-dccp-ccid3-11.txt`>, Mar. 2005.

[7] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” *IEEE/ACM Transactions on Networking*, vol. 1, pp. 397–413, Aug. 1993.

[8] Y. Zhang and L. Qiu, “Understanding the end-to-end performance impact of RED in a heterogeneous environment,” *Cornell CS Technical Report 2000-1802*, July 2000.

[9] H. Hisamatu, H. Ohsaki, and M. Murata, "On modeling feedback congestion control mechanism of TCP using fluid flow approximation and queueing theory," in *Proceedings of 4th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT2001)*, pp. 218–222, Aug. 2001.

[10] H. Hisamatu, H. Ohsaki, and M. Murata, "Steady state and transient behavior analyses of TCP coneections considering interactions between TCP connections and network," *Technical Report of IEICE* (IN2001-149), pp. 1–6, Jan. 2002.

[11] H. Hisamatu, H. Ohsaki, and M. Murata, "Steady state analysis of TCP connections with different propagation delays," *Technical Report of IEICE* (IN2002-97), pp. 41–46, Oct. 2002.

[12] H. Hisamatu, H. Ohsaki, and M. Murata, "Steady state and transient behavior analyses of TCP connections considering interactions between TCP connections and network," in *Proceedings of International Symposium on Applications and the Internet (SAINT 2003)*, pp. 309–316, Jan. 2003.

[13] H. Hisamatu, H. Ohsaki, and M. Murata, "Steady state and transient state behaviors analyses of TCP connections considering interactions between TCP connections and network," *International Journal of Communication Systems*, vol. 18, pp. 619 – 637, Sept. 2005.

[14] H. Hisamatu, H. Ohsaki, and M. Murata, "Steady state and transient state analysis of TCP and TCP-friendly rate control mechanism," *Technical Report of IEICE* (IN2003-46), pp. 25–30, July 2003.

[15] H. Hisamatu, H. Ohsaki, and M. Murata, "Steady state and transient state analyses of TCP and TCP-friendly rate control mechanism using a control theoretic approach," in *Proceedings of SPIE's International Symposium on the Convergence of Information Technologies and Communications (ITCom 2004)*, pp. 40–47, Oct. 2004.

[16] H. Hisamatu, H. Ohsaki, and M. Murata, "Fluid-based analysis of a network with DCCP connections and RED routers," in *Proceedings of International Symposium on Applications and the Internet (SAINT 2006)*, pp. 22–29, Jan. 2006.

[17] H. Hisamatu, H. Ohsaki, and M. Murata, "Fluid-based analysis of network with DCCP connections and RED routers," *Technical Report of IEICE* (IN2005-75), pp. 85–90, Sept. 2005.

[18] H. Hisamatu, H. Ohsaki, and M. Murata, "On modeling datagram congestion control protocol and random early detection using fluid-flow approximation," submitted to *WSEAS Transactions on Communications*, Dec. 2005.

[19] H. Hisamatsu, G. Hasegawa, and M. Murata, "Performance analysis of large-scale IP networks considering TCP traffic," submitted to *IEICE Transactions on Communications*, Nov. 2005.

[20] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "Analytic models and comparative study of latency and steady-state throughput of TCP Tahoe, Reno nad SACK," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 959–971, Dec. 2003.

[21] E. Alessio, M. Garetto, R. Cigno, M. Meo, and M. Marsan, "Analytical estimation of completion times of mixed NewReno and Tahoe TCP connections over single and multiple bottleneck networks," in *Proceedings of IEEE GLOBECOM2001*, Nov. 2001.

[22] V. Firoiu and M. Borden, "A study of active queue management for congestion control," in *Proceedings of IEEE INFOCOM 2000*, pp. 1435–1444, Mar. 2000.

[23] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "A control theoretic analysis of RED," Tech. Rep. TR 00-41, CMPSCI, July 2000.

[24] A. Kumar, "Comparative performance analysis of versions of TCP in a local network with a lossy link," *IEEE/ACM Transactions on Networking*, vol. 6, no. 4, pp. 485–498, 1998.

[25] M. Mathis, J. Semke, and J. Mahdavi, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM SIGCOMM Communication Review*, vol. 27, July 1997.

[26] A. Misra and T. J. Ott, "The window distribution of idealized TCP congestion avoidance with variable packet loss," in *Proceedings of IEEE INFOCOM '99*, pp. 1564–1572, Mar. 1999.

[27] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proceedings of ACM SIGCOMM 2000*, pp. 151–160, Aug. 2000.

[28] T. J. Ott, J. Kemperman, and M. Mathis, "The stationary behavior of ideal TCP congestion avoidance," 1996.

[29] J. Padhye, V. Firoiu, and D. Towsley, "A stochastic model of TCP Reno congestion avoidance and control," tech. rep., CMPSCI Technical Report 99-02, 1999.

[30] C. Casetti and M. Meo, "A new approach to model the stationary behavior TCP connections," in *Proceedings of IEEE INFOCOM 2000*, pp. 367–375, Mar. 2000.

[31] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Reno performance: a simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, pp. 133–145, Apr. 2000.

[32] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: a simple model and its empirical validation," in *Proceedings of ACM SIGCOMM '98*, pp. 303–314, Sept. 1998.

[33] H. Ohsaki and M. Murata, "Steady state analysis of the RED gateway: stability, transient behavior, and parameter setting," *IEICE Transactions on Communications*, vol. E85-B, pp. 107–115, Jan. 2002.

[34] T. J. Ott, "ECN protocols and the TCP paradigm," June 1999. available at `http://web.njit.edu/~ott/Papers/ECN/ECN.pdf`.

[35] "The network simulator – ns2." available at `http://www.isi.edu/nsnam/ns/`.

[36] H. Ohsaki, M. Murata, T. Ushio, and H. Miyahara, "Stability analysis of window-based flow control mechanism in TCP/IP networks," *1999 IEEE International Conference on Control Applications*, pp. 1603–1606, Aug. 1999.

[37] M. Kisimoto, H. Ohsaki, and M. Murata, "Analyzing the impact of TCP connections variation on transient behavior of RED gateway," in *Proceedings of the 16th International Conference on Information Networking (ICOIN-16)*, pp. 10A2.1–10A2.12, Jan. 2002.

[38] M. Kisimoto, H. Ohsaki, and M. Murata, "On transient behavior analysis of random early detection gateway using a control theoretic approach," in *Proceedings of the IEEE Control Systems Society Conference on Control Applications (CCA/CACSD 2002)*, pp. 1144–1149, Sept. 2002.

[39] J. Postel, "User datagram protocol," *Request for Comments (RFC) 768*, Aug. 1980.

[40] M. Allman, V. Paxson, and W. R. Stevens, "TCP congestion control," *Request for Comments (RFC) 2581*, Apr. 1999.

[41] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE Transactions on Networking*, vol. 7, pp. 458–472, May 1999.

[42] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications: the extended version," tech. rep., International Computer Science Institute, Mar. 2000.

[43] B. Braden *et al.*, "Recommendations on queue management and congestion avoidance in the Internet," *Request for Comments (RFC) 2309*, Apr. 1998.

[44] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "On designing improved controllers for AQM routers supporting TCP flows," in *Proceedings of IEEE INFOCOM 2001*, pp. 1726–1734, 2001.

[45] H. Ohsaki and M. Murata, "On packet marking function of active queue management mechanism: Should it be linear, concave, or convex?," in *Proceedings of SPIE's International Symposium on the Convergence of Information Technologies and Communications (ITCom 2004)*, Oct. 2004.

[46] D. Bansal, H. Balakrishnan, S. Floyd, and S. Shenker, "Dynamic behavior of slowly-responsive congestion control algorithms," in *Proceedings of ACM SIGCOMM*, pp. 263–274, Aug. 2001.

[47] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model based TCP-friendly rate control protocol," *UMass-CMPSCI Technical Report TR 98-04*, 1998.

[48] Y. R. Yang, M. S. Kim, and S. S. Lam, "Transient behaviors of TCP-friendly congestion control protocols," *The International Journal of Computer and Telecommunications Networking*, vol. 41, pp. 193–210, Feb. 2003.

[49] K. Ramakrishnan, S. Floyd, and D. Black, "The addition of explicit congestion notification (ECN) to IP," *Request for Comments (RFC) 3168*, Sept. 2001.

[50] N. Spring, D. Wetherall, and D. Ely, "Robust explicit congestion notification (ECN) signaling with nonces," *Request for Comments (RFC) 3540*, June 2003.

[51] H. Ohsaki, J. Ujiie, and M. Imase, "On scalable modeling of TCP congestion control mechanism for large-scale IP networks," in *Proceedings of IEEE SAINT 2005*, pp. 361–369, Feb. 2005.

[52] M. Handly, S. Floyd, J. Padhye, and J. Widmer, "TCP friendly rate control (TFRC): protocol specification," *Request for Comments (RFC) 3448*, Jan. 2003.

[53] N. S. Nise, *Control Systems Engineering*. New York: John Wiley & Sons, 4th ed., Aug. 2003.

[54] M. Fomenkov, K. Keys, D. Moore, and k claffy, "Longitudinal study of Internet traffic from 1998–2003," in *Winter International Symposium on Information and Communication Technologies (WISICT 2004)*, Jan. 2004.

[55] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, “Fast accurate computation of large-scale IP traffic matrices from link loads,” in *Proceedings of the ACM SIGMETRICS*, pp. 206–217, June 2003.

[56] M. Roughan, M. Thorup, and Y. Zhang, “Traffic engineering with estimated traffic matrices,” in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, pp. 248–258, Oct. 2003.

[57] L. Li, D. Alderson, W. Willinger, and J. Doyle, “First-principles approach to understanding the Internet’s router-level topology,” in *Proceedings of ACM SIGCOMM ’04*, pp. 3–14, Sept. 2004.

[58] S. H. Low, F. Paganini, J. Wang, S. Adlakha, and J. C. Doyle, “Dynamics of TCP/RED and a scalable control,” in *Proceedings of IEEE INFOCOM*, June 2002.

[59] S. H. Low, “A duality model of TCP and queue management algorithms,” *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 525–536, 2003.

[60] Y. Liu, F. L. Presti, V. Misra, D. Towsley, and Y. Gu, “Fluid models and solutions for large-scale IP networks,” in *Proceedings of ACM/SIGMETRICS 2003*, pp. 91–101, June 2003.

[61] M. A. Marsan, M. Garetto, P. Giaccone, E. Leonardi, E. Schiattarella, and A. Tarello, “Using partial differential equations to model TCP mice and elephants in large IP networks,” *IEEE/ACM Transactions on Networking*, vol. 13, dec 2005.

[62] A. Kowalski and B. Warfield, “Modelling traffic demand between nodes in a telecommunications network,” in *Australian Telecommunications and Networks Conference (ATNC)*, Dec. 1995.

[63] G. Appenzeller, I. Keslassy, and N. McKeown, “Sizing router buffers,” in *Proceedings of ACM SIGCOMM ’04*, pp. 281–292, Sept. 2004.

[64] H. Takahashi, M. Saito, H. Aida, Y. Tobe, and H. Tokuda, “Estimated-tcp-throughput maximization based routing,” in *Proceedings of IEEE International Conference on-Local Computer Networks (LCN)*, pp. 120–129, 10 2003.

[65] S. Floyd, “Highspeed TCP for large congestion windows,” *Request for Comments (RFC) 3649*, Dec. 2003.