

INTERNET CONGESTION CONTROL:
COMPLETE STABILITY REGION FOR PI AQM AND
BANDWIDTH ALLOCATION IN NETWORKED CONTROL

by

AHMAD TAWFIQ AL-HAMMOURI

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

Thesis Advisor: Prof. Vincenzo Liberatore

Department of Electrical Engineering and Computer Science
CASE WESTERN RESERVE UNIVERSITY

January, 2008

CASE WESTERN RESERVE UNIVERSITY
SCHOOL OF GRADUATE STUDIES

We hereby approve the dissertation of

Ahmad Tawfiq Al-Hammouri

candidate for the Ph.D. degree *.

(signed) Prof. Vincenzo Liberatore
(chair of the committee)

Prof. Wei Lin

Prof. Michael Rabinovich

Prof. Michael Branicky

Prof. Stephen Phillips

(date) August 31, 2007

*We also certify that written approval has been obtained for any proprietary material contained therein.

To my loving parents, brothers, and sisters.

Contents

List of Tables	vii
List of Figures	viii
Acknowledgments	xi
Chapter 1 Introduction	1
1.1 PI AQM	2
1.2 CPSs and Networked Control	4
Chapter 2 Analytic Derivation of the PI-AQM Stability Region	6
2.1 Introduction	6
2.2 Background	9
2.2.1 The TCP Model	9
2.2.2 Related AQM Schemes	11
2.2.3 The TCP-PI Feedback Loop	12
2.2.4 Determination of k_p and k_i	13
2.2.5 Stability Region for Time-Delay Systems	14
2.3 Computing $\mathcal{S}_R$ for TCP-AQM PI Controllers	16
2.3.1 Computing $\mathcal{S}_0$	16
2.3.2 Computing $\mathcal{S}_N$	17
2.3.3 Computing $\mathcal{S}_L$ and $\mathcal{S}_R$	18

2.4	Simulations	31
2.4.1	Simulation Methodology	31
2.4.2	Theoretically Stable and Unstable Controllers	31
2.4.3	$\mathcal{S}_R$ versus Δ_{PI} and Ξ_{PI}	34
2.4.4	Discussion	35
2.5	Conclusions	38

Chapter 3 Decentralized and Dynamic Bandwidth Allocation in Cyber-Physical

Systems	39
3.1	Introduction 39
3.2	Related Work 43
3.2.1	Congestion Control in IP Networks 43
3.2.2	Bandwidth Allocation in CPSs 44
3.3	Problem Formulation 45
3.3.1	On the Wire 45
3.3.2	Optimization Formulation 47
3.3.3	Distributed Implementation 48
3.4	Link Queue Controllers 50
3.4.1	Modeling CPS-Queue Interaction 50
3.4.2	Linearized Model 51
3.4.3	P and PI Controllers 54
3.5	Steady-state Error and Queueing Delays 55
3.6	Stability Analysis 56
3.6.1	The P Controller 57
3.6.2	The PI Controller 59
3.7	Robust Stability 61
3.8	Simulations 66
3.8.1	Simulation Software 66

3.8.2	Network Topology	67
3.8.3	Plants and Controllers	68
3.8.4	Experiments	69
3.8.5	Computing Controller Parameters	69
3.8.6	Results	70
3.9	Conclusions	75
Chapter 4	Conclusions and Future Work	76
4.1	Conclusions	76
4.2	Future Work	77
	References	79

List of Tables

3.1	The values of B , p_0 , and $k_{p,\max}^P$ during each interval of time where the number of CPSs is constant.	70
-----	---	----

List of Figures

2.1	A simple experiment showing that RED exhibits steady-state error that is dependent on the number of TCP flows whereas PI does not. The flows, which share a single bottleneck link, are increased from 50 flows to 100 flows at simulation time 50 sec. and then to 300 at time 100 sec.	12
2.2	The closed-loop system of TCP-AQM linearized model $P(s)$, with the PI controller, $G(s)$ [30].	13
2.3	The stabilizing region of k_p and k_i gains for the delay-free closed-loop system, i.e., $\mathcal{S}_0$. The area is under the line with a slope of $(\alpha + \beta)$ and a y -intercept of $\alpha\beta(\alpha + \beta)/B$. Moreover, since $\mathcal{S}_N = \emptyset$, $\mathcal{S}_1 = \mathcal{S}_0$	18
2.4	$L(\omega)$ versus ω (upper) and $\sqrt{M(\omega)}$ versus ω (lower): (a) when $\hat{k}_p \leq \gamma$; (b) when $\hat{k}_p > \gamma$ and $L(\omega_{\min}) > d$; (c) when $\hat{k}_p > \gamma$ and $L(\omega_{\min}) \leq d$	28
2.5	The two stability regions corresponding to $d_s = 60$ msec and $d_l = 167$ msec along with the points p_1 , inside $\mathcal{S}_R(d_l)$, and p_2 , outside $\mathcal{S}_R(d_s)$	32
2.6	The queue size when using the two sets of (k_p, k_i) parameters corresponding to the points p_1 and p_2	33
2.7	The spectral plots of $q(t)$ of Figure 2.6 when using p_1 (left) and p_2 (right).	34
2.8	$\mathcal{S}_R$ based on the analysis in this chapter, the line segment Ξ_{PI} [32], the point Δ_{PI} [30], and seven other points, p_2 – p_8 , to be used in simulations.	34
2.9	Comparing the performance of different points on Ξ_{PI}	36

2.10	Extending the experiment's execution time up to 3000 sec to observe the long-term trend of $q(t)$ when using $p_1 = \Delta_{PI}$	36
2.11	Comparing the performance of different points inside $\mathcal{S}_R$ but off Ξ_{PI} with p_4	37
3.1	CPSs integrate sensing, processing, and actuation tasks that enable for remote monitoring and control of the physical world. Figure is adopted from [6, 42].	40
3.2	A Cyber-Physical System with one controlled system (a.k.a. <i>plant</i>) and one controller. Both the sensor and the actuator are collocated at the plant site.	45
3.3	Examples of two generic utility functions.	47
3.4	Forward and backward delays. Figure is adapted from [69].	51
3.5	Linearized Model of CPS-Queue interaction with the controller $G(s)$	54
3.6	The Nyquist Plot of $H(j\omega) = \frac{Bk_p}{j\omega} e^{-j\omega(d_0+p_0/(k_p C))}$ when $k_p < 0$ (left) and when $k_p > 0$ (right) along with the critical point $-1 + j0$	58
3.7	A single bottleneck topology for experimental simulation.	67
3.8	Six sets of CPSs. Each set uses the network for the duration defined by the extent of the bold horizontal line. For example, the set comprising <code>plant(0)</code> and <code>plant(1)</code> acquires the network at $t_s = 0$ sec and releases it at $t_e = 300$ sec.	69
3.9	The stabilizing range of the P controller and the stabilizing region of the PI controller, $\mathcal{S}_R$, for $B = 1.3319 \times 10^7$, $p_0 = 4.6917 \times 10^{-5}$ and $d_{\max} = 0.15$ sec; and the two points $k_p^P = 4.0 \times 10^{-7}$ and $(k_p^{PI}, k_i^{PI}) = (4.75 \times 10^{-7}, 9.0 \times 10^{-7})$ used in simulations.	71
3.10	Transmission rates when using the P and the PI controllers.	72
3.11	Queue length when using the P and the PI controllers.	72

3.12	Round-trip delays when using the P and the PI controllers. Only delays for <code>plant(0)</code> and <code>plant(1)</code> are highlighted with solid lines. Delays for plants <code>plant(2)</code> through <code>plant(11)</code> are shown in dotted lines, which fall between those of <code>plant(0)</code> and <code>plant(1)</code>	73
3.13	The input signal, $R(t)$, plants are instructed to follow.	73
3.14	Plant state, $x(t)$, for <code>plant(0)</code> and <code>plant(1)</code> while tracking the input signal, $R(t)$, of Figure 3.13 when the PI controller is used as a queue controller.	74
3.15	Plant state, $x(t)$, for <code>plant(0)</code> and <code>plant(1)</code> while tracking the input signal, $R(t)$, of Figure 3.13 when the P controller is used as a queue controller. Only the time interval $[0, 150]$ sec is shown, which is divided into two separate figures (left for $[0, 100]$ sec and right for $[100, 150]$ sec) to highlight differences in CPS control performance.	74

Acknowledgments

First, I would like to express my gratitude to my thesis advisor, Professor Vincenzo Liberatore, for his endless support, encouragements, and advice during the course of my graduate studies at Case Western Reserve University. I was honored to be his apprentice and it was a privilege to work with him.

I am also indebted to my other dissertation committee members, Professors Michael Branicky, Michael Rabinovich, Stephen Phillips, and Wei Lin. Professor Branicky has been as an honorary advisor during the work on this thesis. I truly appreciate him for the invaluable assistance and comments on research ideas and for closely reading and commenting on all my manuscripts. Professor Stephen Phillips was the third professor in our research group (in addition to Profs. Branicky and Liberatore). He offered helpful suggestions on both research and write-up. Professors Rabinovich and Lin provided valuable feedback during both the proposal and the final defenses.

Further, I am grateful to the warm-hearted EECS staff members who offered their hands to help me throughout the years. Special thanks to Cheryl Lange, Elizabethanne Fuller-Murray, Marla Radvansky, and Paul Schneider.

I would like to thank my previous and current colleagues in the Networks research Lab, especially, Deepak Agrawal (my summer-intern advisee), Graham Alldredge, Nathan Wedge, Qingbo Cai, and Wenhui Zhang.

At last but not least, I would like to thank my friends, especially those in USA, for being close to me and for making me know that there is an enjoyable life outside the Lab! Thanks Abdullah Jordan, Ahmed Abdel-Nabi Al-Hammouri, Hussein Alzoubi, Huthaifa Al-Omari, Ibrahim Salami, Issa Abdulmateen, Mohammad Ghanamah, Mohammad Saleh

Alhammouri, Mohammed Aloqlah, Mohammed Yakoub, Muhammad Darawad, Osama Al-Khaleel, Saleem Bani-Hani, Sulieman Bani-Ahmad, and Zakaria Al-Qudah.

I wish everyone good luck and happiness. Once again, thanks everyone!

Ahmad Tawfiq Al-Hammouri

Cleveland, Ohio USA

September 6, 2007

Internet Congestion Control: Complete Stability Region for PI AQM and Bandwidth Allocation in Networked Control

Abstract

by

Ahmad Tawfiq Al-Hammouri

The Internet represents a shared resource, wherein users contend for the finite network bandwidth. Contention among independent user demands can result in congestion, which, in turn, leads to long queueing delays, packet losses or both. Congestion control regulates the rate at which traffic sources inject packets into a network to ensure high bandwidth utilization while avoiding network congestion. In this thesis, we present contributions pertaining to two specific areas in the Internet congestion control: PI AQM and bandwidth allocation in Cyber-Physical Systems (CPSs). In the area of PI AQM, we present an analytic derivation of the complete stability region. The stability region represents the entire set of the feasible design parameters that stabilize the closed-loop TCP-AQM system. Utilizing the complete stability region, we show that the PI parameters used in the literature can be excessively conservative. We also show that provably stable controller parameters can exhibit widely different levels of performance. Furthermore, we present examples of PI controllers that are stable and have significantly better performance than previously proposed ones. These facts explain the previous observation about PI sluggish responsiveness and stress the importance of obtaining the complete stability region for the PI AQM. As for CPSs bandwidth allocation, we devise a bandwidth allocation scheme for Cyber-Physical Systems that have their control loops closed over a distributed network. We

formulate the bandwidth allocation as a convex optimization problem. We then present an allocation scheme that solves this optimization problem in a fully distributed manner. In addition to being fully distributed, the proposed scheme is asynchronous, scalable, dynamic and flexible. Furthermore, we design robust and resilient queue controllers to enhance the performance of the bandwidth allocation scheme to better fulfill the requirements of the CPSs control loops. Throughout the thesis, we present analytical results and we validate them with packet-level simulations via ns-2.

Chapter 1

Introduction

The Internet is a network of individual networks that interconnect multiple users and allow them to communicate data. Each individual network is composed of physical communication links with each having a finite bandwidth capacity. As it is true with any finite resource shared among multiple independent users, the network can become congested. Congestion, which occurs when the offered packet load exceeds network capacity, has severe impacts as bad as total collapses and complete service blackouts. Such collapses had actually struck the early Internet and then they triggered amendments to the well-known TCP to include a congestion control mechanism [34]. In principle, congestion control regulates the rate at which traffic sources inject packets into a network to ensure high bandwidth utilization while avoiding network congestion. Congestion control was arguably one of the reasons that the Internet scaled up to its size today.

From control theory perspectives, congestion control can be viewed as a feedback closed-loop regime whereby end-systems regulate their sending rates based on explicit or implicit congestion signals fed back from the network. As a result, control-theoretical methods have been extensively utilized to analyze congestion control algorithms (see for example [47, 61, 69] and the references therein). Since it ties in different network and protocol parameters, control-theoretical analysis explained most of the observations about

operational poor performance of congestion control algorithms in some particular environments, and it also motivates alternative designs. In a nutshell, control-theoretical approaches lead to stable, effective, and robust congestion control design and operation.

In this thesis, we present novel results on the Internet congestion control pertinent to two different domains: Proportional-Integral (PI) Active Queue Management (AQM) and Cyber-Physical Systems (CPSs), in Chapters 2 and 3, respectively. We summarize the scope of this thesis in the next sections. Chapters 2 and 3 are constructed such that each one is an independent part, is self contained, and thus requires no knowledge about the other part. For example, each chapter reviews the related previous research work, and includes packet-level simulations using ns-2 [1] that validate the theoretical analysis presented in the respective chapter. The order of these chapters in this thesis is thus arbitrary. The terminology we use in this thesis is that when we refer to networks, we usually mean IP networks in the Wide Area Networks (WAN) domain, such as the Internet.

1.1 PI AQM

Congestion control was introduced into TCP to address the problem of congestion collapses that were occurring during the 1980s. Due to the original philosophy of the Internet—the end-to-end principle—end-systems implemented most of the congestion control’s functionality. However, Active Queue Management (AQM) [17] can facilitate end-point congestion control by proactively marking or dropping packets prior to the inception of congestion. AQM’s early feedback provides the opportunity to improve over drop-tail queues [17], which drop packets only when buffers overflow. First, AQM would allow sources to throttle early their transmission rates in an attempt to avoid congestion before its inception. Second, AQM would leave enough space in routers buffers to absorb traffic bursts. Consequently, AQM should lead to low packet losses, short queueing delays, and high bandwidth utilization [4]. Random Early Detection (RED) [22] was one of the earliest

AQM algorithms proposed to accomplish these goals. However, theory and simulations revealed some shortcomings intrinsic to RED, such as slow responsiveness and steady-state backlogs [31]. To address these drawbacks, the Proportional Integral (PI) controller was proposed as an alternative AQM [30]. The PI controller is a more natural choice due to its robustness and its ability to eliminate steady-state error in the queue length. That is, PI stabilizes the queue length around a controllable target level. Both theory and simulations showed that PI outperforms RED [30]. However, previous work lacks a complete characterization of the stability region of the PI controller parameters. The original paper on PI AQM gives a single pair of the *proportional gain*, k_p , and the *integral gain*, k_i , that guarantees the stability of the closed-loop system as a function of the network parameters [30]. However, there are other (k_p, k_i) pairs that stabilize the closed-loop system and result in better performance.

In Chapter 2 of this thesis, we present an analytic derivation of the complete stability region of the PI AQM. The stability region represents the entire set of the feasible design parameters, i.e., k_p and k_i , that stabilize the closed-loop TCP-AQM system. The stability region thus facilitates the selection of stable controllers that lead to better AQM performance. From the control theory point of view, our contribution is major because we analytically characterize the complete stability region for a second-order system and a PI controller. There has been no such analytical characterization thus far [58]. Our contribution is also important from an application perspective because of the following reasons. First, we show that the pair of k_p and k_i given by [30] can be excessively conservative when compared with other stable gains. This conservativeness, in turn, explains the sluggish responsiveness of PI observed in the literature. Second, we show that some provably stable controller parameters enjoy better performance than others obtained according to [30] or [32]. Finally, the same stability analysis of PI can be utilized to obtain the complete stability region for other AQM algorithms, such as REM [15] and PIP [29], by straightforward change of variables. One can then use the stability region to prove mathematically—not

just through simulations as in [55]—that PIP becomes unstable according to the original design [29].

1.2 CPSs and Networked Control

We are witnessing major advances in VLSI, in MEMS, and in communication networks technologies that have brought devices with sensing, processing, actuation, and communication capabilities. These devices have facilitated the convergence of the cyber- and physical-worlds, and have thus contributed to the formation of *Cyber-Physical Systems* (CPSs). CPSs allow humans to monitor, affect, control and interact with remote physical environments, thus extending human’s reach beyond spatial barriers [2, 7]. *Sensors* sense the physical quantities, generate a stream of sampled data, and communicate this data over a network to *controllers*. Controllers process the samples of the sensed data and generate appropriate control signals to be delivered over the network to *actuators*. Actuators transform control signals into actions that affect the physical world [8].

If networked control is to be deployed ubiquitously over IP networks, there arises a critical need for a bandwidth management to allocate the network bandwidth between different CPSs [8]. Addressing this issue in Chapter 3 of this thesis, we devise a bandwidth allocation scheme for CPSs that *fairly* allocates the bandwidth to control congestion and to meet each system’s requirement as best as possible. In the proposed scheme, CPSs adapt their sampling intervals based on the congestion level fed back from the network. We also discuss designing robust and resilient controllers that enhance the performance of the bandwidth allocation scheme to better serve the CPSs control loops requirements. Our proposed scheme has the following features:

- It allocates the bandwidth in a way to ensure stability of all control systems, if feasible.
- It allocates the bandwidth in a way to attain the maximum *aggregate* performance of

all control systems.

- It makes use of network bandwidth efficiently; controls congestion, thus minimizes delays and losses; and achieves fairness by fulfilling performance objectives of different control loops.
- It provides a *fully distributed, asynchronous, and scalable* solution. Each node executes an independent algorithm using local information with no central managing entity. The approach scales up as the number of controlled systems and/or the size of the network increase.
- It is *dynamic, adaptable, and flexible*. It dynamically reallocates the bandwidth as different control systems acquire and relinquish network resources.

Chapter 2

Analytic Derivation of the PI-AQM Stability Region

In this chapter, we derive an analytical characterization of the complete stability region of the PI controller for TCP AQM and we validate it with ns-2 simulations. The analytical challenge is the presence of time-delays in the TCP-AQM feedback loop. The complete stability region provides an in-depth understanding of the performance of PI AQM under different network parameters, especially round-trip delays and bandwidth. Having in hand the complete stability region, we show that the PI parameters used in the literature can be excessively conservative. This characterization explains the previous observation about PI sluggish responsiveness and stresses the importance of obtaining the complete stability region for the PI AQM.

2.1 Introduction

Active Queue Management (AQM) controls congestion by proactively marking or dropping packets before the inception of congestion. AQM's early feedback provides the opportunity to improve over drop-tail queues, which drop packets only when buffers overflow. First, AQM would allow sources to throttle early their transmission rates in an attempt to

avoid congestion before its inception. Second, AQM would leave enough space in routers' buffers to absorb traffic bursts. Consequently, AQM should lead to low packet losses, short queueing delays, and high bandwidth utilization [4]. Several AQM schemes have been proposed to accomplish these goals, with Random Early Detection (RED) [22] being one of the earliest algorithms. To address RED's shortcomings, e.g., slow responsiveness and steady-state backlogs, the Proportional Integral (PI) controller was proposed as an alternative AQM [30]. The PI controller is a more natural choice due to its robustness and its ability to eliminate the steady-state error in the queue length. That is, PI stabilizes the queue length around a controllable target level. Both theory and simulations showed that PI outperforms RED [30]. The advent of PI has spurred the deployment of several related AQM controllers—such as PIP [29], PD [60], PID [57], and P²I [70]—all of which were proposed for a single goal: to speed up the responsiveness of the PI controller. However, these AQM controllers lack a characterization of the complete stability region. Without the complete stability region, designers usually resort to conservative parameter values to ensure the stability and convergence of the congestion control algorithms. In turn, conservative parameters can yield poor performance.

In this chapter, we present an analytic derivation of the complete stability region of the PI AQM. The stability region represents the entire set of the feasible design parameters, i.e., the *proportional* and the *integral* gains, that stabilize the closed-loop TCP-AQM system. The stability region thus facilitates the selection of controllers that lead to better AQM performance.

A closed-form derivation of the complete stability region for PI AQM is complicated by the combination of various factors. Some factors are intrinsic to the PI and TCP-AQM loop: the linearized TCP-AQM model is a second-order plant, includes time delays in the control loop, and is cascaded with PI, a relatively high-order controller. The resultant characteristic equation is a high-order quasi-polynomial whose stability analysis poses great theoretical challenges. Another source of complications pertains to the objective of attain-

ing stability that is robust to approximate values of system parameters. In particular, the TCP-AQM closed-loop system should be stabilized not only for a single value of the delay but for a delay interval.

All these factors render the stability analysis techniques of elementary Control Theory unusable to obtain the complete stability region for the PI AQM. Previous work sidestepped the problem through assumptions and simplifications, ending up with only a subset of the whole space of stabilizing controller parameters. The original paper on PI AQM gave guidelines to choose only a single pair of the proportional gain, k_p , and the integral gain, k_i , that guarantees the stability of the closed-loop system [30]. The stability set was later expanded to a line segment in the k_i – k_p plane [32]. However, all previous work, including [32], used the guidelines prescribed by [30].

In this chapter, we exploit recent results on robust PI control theory for time-delay systems to obtain a complete stability region for the PI controller with the TCP-AQM model. We then validate the theoretical analysis by conducting packet-level simulations using the PI implementation in ns-2. From the control theory point of view, our contribution is major because there has been no analytical characterization of the stability region for a second-order system with a PI controller thus far [58]. Our contribution is also important from an application perspective because of the following reasons. First, we show that the pair of k_p and k_i given by [30] can be excessively conservative when compared with other stable gains. This conservativeness, in turn, explains the sluggish responsiveness of PI observed in the literature. Second, we show that some provably stable controller parameters enjoy better performance than others obtained according to [30] and [32]. Finally, the same stability analysis of PI can be utilized to obtain the complete stability region for other AQM algorithms, such as REM [15] and PIP [29], by straightforward change of variables. One can then use the stability region to prove mathematically—not just through simulations as in [55]—that PIP becomes unstable according to the original design [29]; see [9, 10].

The rest of the chapter is structured as follows. Section 2.2 covers the related back-

ground, introduces the linearized TCP-AQM model with PI controller, and presents the method we used to obtain the complete stabilizing region. In Section 2.3, we compute the complete set, $\mathcal{S}_R$, of the stabilizing PI parameters. Simulations that stress the importance of obtaining the complete stabilizing region are presented in Section 2.4. Finally, Section 2.5 concludes the chapter.

2.2 Background

2.2.1 The TCP Model

Congestion control was introduced into TCP to address the problem of congestion collapses that were occurring during the 1980s. Congestion control was arguably one of the reasons that the Internet scaled up to its size today. Due to the original philosophy of the Internet—the end-to-end principle—end-systems had to implement most of the congestion control’s functionality. However, AQM can facilitate the end-point congestion control by marking or dropping packets prior to the inception of congestion. Because the interaction between end-systems and AQM routers gives rise to a feedback loop, AQM has been extensively analyzed using control-theoretical methods (see for example [32] and the references therein). Control-theoretical approaches lead to stable, effective, and robust congestion control operation.

A nonlinear fluid-based model was developed to describe the dynamics of multiple TCP flows with AQM routers [51]. The model consists of a system of nonlinear differential equations. To facilitate further analysis, the model was then simplified by ignoring the time-out component. The resultant system of equations is given by [32]:

$$\begin{cases} \dot{W}(t) &= \frac{1}{d(t)} - \frac{W(t)W(t-d(t))}{2d(t-d(t))}p(t-d(t)) \\ \dot{q}(t) &= \frac{W(t)}{d(t)}N(t) - C, \end{cases} \quad (2.1)$$

where $W(t)$ is the window size, $d(t)$ is the round-trip delay, $p(t)$ is the probability of packet marking, $q(t)$ is the queue length, $N(t)$ is the number of TCP flows, and C is the link capacity. The system of equations (2.1) was derived independently in other papers following different approaches [44, 47]. According to simulations, Eq. (2.1) closely models the TCP dynamics [32].

To carry out linear system analysis, we use the linearized version of (2.1) around the equilibrium point (W_0, q_0, p_0) defined by $[\dot{W}(t) \quad \dot{q}(t)]^T = [0 \quad 0]^T$ [32]. Further, since we use frequency-response analysis, we consider the Laplace transform of the linearized system. The linearization and its Laplace transformation are straightforward and the resultant transfer function is given by

$$P(s) = \frac{B}{(s + \alpha)(s + \beta)} e^{-sd}, \quad (2.2)$$

where $B = C^2/(2N)$, $\alpha = 2N/(d^2C)$, $\beta = 1/d$, and d is the round-trip delay at equilibrium [32, 44, 47]. The variables $B > 0$, $\alpha > 0$, and $\beta > 0$ are introduced for convenience.

Notes on the Model

The TCP models in (2.1) and (2.2) assume the following:

- All flows are TCP. Although real networks carry different types of traffic (e.g., UDP), non-TCP traffic can be modeled and incorporated in (2.2) as in [33]. For stability analysis purposes, the resulting model has the same form as of (2.2) but with slightly different values of B , α , and β . Since our analysis considers B , α , and β as primary and independent variables, our results still hold for the model that is accounting for unresponsive traffic as long as the results are expressed in terms of B , α , and β but not in terms of N , C , and d .
- The number of flows, N , and the link capacity, C , are stationary or change slowly

over time compared to other dynamics, e.g., $W(t)$ and $q(t)$.

- The model equations are parametrized by N , C , and d . None of N , C , or d is restricted to be higher or lower than any specific value except that they are all > 0 .
- The topology is a single bottleneck link topology, which allows for deeper insight and serves as a starting point to more general topologies as future research.

As mentioned earlier, there is a general consensus on the validity of the TCP models (2.1) and (2.2) in the literature. However, if a new study were to develop a more accurate model for the TCP congestion control, the new model could still be approximated by a second-order system as in (2.2), and hence its stability analysis would follow directly from the analysis presented in this chapter.

2.2.2 Related AQM Schemes

RED was one of the earliest AQM mechanisms proposed to replace drop-tail queues [22]. It enjoyed the most attention from researchers, and is the only one that made its way into an IETF standard. RED monitors congestion on the outgoing link by maintaining an EWMA (exponentially weighted moving average) of the queue length. Packets are marked when the average queue length exceeds some threshold. RED has then undergone several refinements to improve its performance; see for example [20] and the RED implementation in ns-2 [1]. However, these refinements increased the number of parameters that need to be tuned, thus amplifying its operational complexity. The original RED was analyzed using control theory techniques and was shown to exhibit slow responsiveness and large steady-state backlogs, especially when the number of flows is large [30]; see Figure 2.1. PI was then proposed as an alternative controller. PI uses an integration action to eliminate steady-state backlogs and stabilizes the queue around a fixed target regardless of the increase in the number of flows; see Figure 2.1. PI is also a fundamental component for other AQM and congestion control schemes. For example, several AQM schemes can be considered

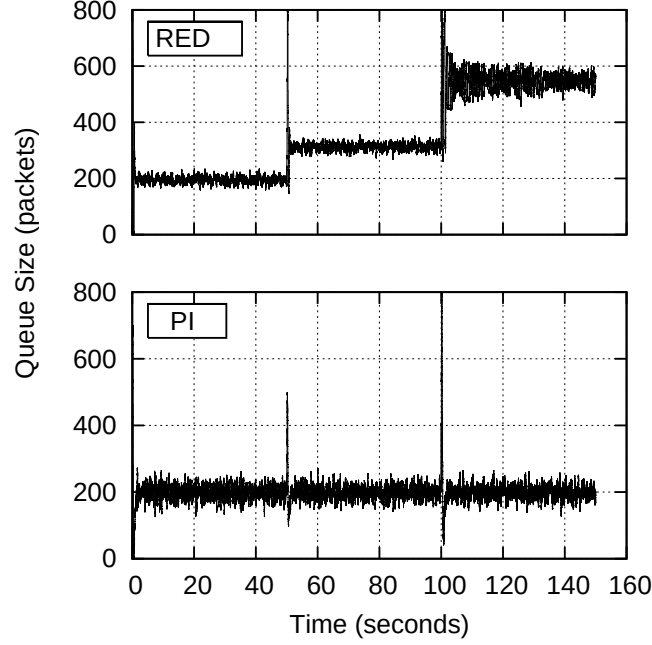


Figure 2.1: A simple experiment showing that RED exhibits steady-state error that is dependent on the number of TCP flows whereas PI does not. The flows, which share a single bottleneck link, are increased from 50 flows to 100 flows at simulation time 50 sec. and then to 300 at time 100 sec.

variants and extensions of PI, such as, REM [15], PIP [29] and PID [57]. In general, all these AQM schemes can use the results and the analysis presented in this chapter as will be highlighted in the next subsection.

2.2.3 The TCP-PI Feedback Loop

The introduction of PI AQM results in the feedback control loop shown in Figure 2.2, where $q(s)$ is the Laplace transform of the instantaneous queue length $q(t)$, q_0 is the desired queue length around which the controller should stabilize $q(t)$, and

$$G(s; k_p, k_i) = k_p + \frac{k_i}{s} = \frac{k_p s + k_i}{s}$$

is the transfer function of the PI controller. The controller $G(s; k_p, k_i)$ will be denoted simply as $G(s)$ when the proportional gain k_p and the integral gain k_i are clear from the

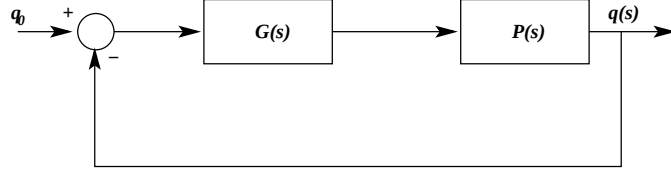


Figure 2.2: The closed-loop system of TCP-AQM linearized model $P(s)$, with the PI controller, $G(s)$ [30].

context.

Although we restrict the analysis to the PI controller and the feedback loop in Figure 2.2, the results can be applied to related AQM schemes by simple changes of variables, such as in the case of PIP (see [9, 10] for details) and REM, or the same analysis can be easily extended to other schemes, such as AVQ [39].

2.2.4 Determination of k_p and k_i

Given a network topology with specific C , N and d , the objective is to determine the values of the parameters k_p and k_i that stabilize the TCP-AQM closed-loop system in Figure 2.2 for delays less than or equal to d .

In [30], the stability analysis was oversimplified and resulted in guidelines to choose only a single pair $\Delta_{PI} = (k_p, k_i)$ of stabilizing PI gains. Since these gains are conservative, the PI controller showed sluggish responsiveness [21, 29]. In [32], a less conservative analysis used the pole-zero cancellation technique [38] to achieve model order reduction. The idea is to cancel the $(s + \alpha)$ pole in (2.2) by mandating the condition $k_i = \alpha k_p$. However, this confines the space of stabilizing PI gains to a line segment Ξ_{PI} that is a portion of the infinite line $k_i = \alpha k_p$ in the k_i - k_p plane.

In this chapter, we exploit recent results on time-delay systems to characterize the complete set of stabilizing k_p and k_i gains. To be self-contained, this chapter reviews one such recent method for time-delay PI control [58] in the next subsection. In the rest of the chapter, we will apply this method for the stability analysis of TCP AQM.

2.2.5 Stability Region for Time-Delay Systems

The stability region $\mathcal{S}_R$ is the complete set of points (k_p, k_i) for which the closed-loop system in Figure 2.2 is stable for all delays L between 0 and d . The stability region $\mathcal{S}_R$ can be expressed as $\mathcal{S}_R = \mathcal{S}_1 \setminus \mathcal{S}_L$ [58, p. 249], where

- $\mathcal{S}_1 = \mathcal{S}_0 \setminus \mathcal{S}_N$.
- $\mathcal{S}_0$ is the set of k_p and k_i values that stabilize the delay-free system $P_0(s)$.
- $\mathcal{S}_N$ is the set of k_p and k_i values such that $G(s; k_p, k_i)P_0(s)$ is an improper transfer function. (Also, $\mathcal{S}_N$ is the set that destabilizes the closed-loop system when the delay is introduced.) Formally, $\mathcal{S}_N$ is

$$\mathcal{S}_N = \left\{ (k_p, k_i) : \lim_{s \rightarrow \infty} |G(s; k_p, k_i)P_0(s)| \geq 1 \right\}. \quad (2.3)$$

- $\mathcal{S}_L$ is the set of (k_p, k_i) values such that $G(s; k_p, k_i)P(s)$ has a minimal destabilizing delay that is less than or equal to d . Formally, $\mathcal{S}_L$ is

$$\begin{aligned} \mathcal{S}_L = \{ (k_p, k_i) \notin \mathcal{S}_N : \exists L \in [0, d], \omega \in \mathbb{R} \text{ s.t.} \\ G(j\omega; k_p, k_i)P_0(j\omega)e^{-jL\omega} = -1 \}. \end{aligned} \quad (2.4)$$

To compute $\mathcal{S}_R$, first define the projection of the stability region $\mathcal{S}_R$ on the line $k_p = \hat{k}_p$ as:

$$\mathcal{S}_{R, \hat{k}_p} = \{ (k_p, k_i) \in \mathcal{S}_R : k_p = \hat{k}_p \},$$

so that the stability region can be calculated for each value of the proportional gain $\hat{k}_p$:

$$\mathcal{S}_R = \bigcup_{\hat{k}_p} \mathcal{S}_{R, \hat{k}_p}. \quad (2.5)$$

To compute $\mathcal{S}_{R,\hat{k}_p}$, define the projections

$$\mathcal{S}_{1,\hat{k}_p} = \{(k_p, k_i) \in \mathcal{S}_1 : k_p = \hat{k}_p\} ,$$

$$\mathcal{S}_{N,\hat{k}_p} = \{(k_p, k_i) \in \mathcal{S}_N : k_p = \hat{k}_p\} ,$$

$$\mathcal{S}_{L,\hat{k}_p} = \{(k_p, k_i) \in \mathcal{S}_L : k_p = \hat{k}_p\} .$$

Then, $\mathcal{S}_{R,\hat{k}_p} = \mathcal{S}_{1,\hat{k}_p} \setminus \mathcal{S}_{L,\hat{k}_p}$. It remains to compute $\mathcal{S}_{L,\hat{k}_p}$ by evaluating the condition in (2.4) that $G(j\omega; k_p, k_i)P_0(j\omega)e^{-jL\omega} = -1$. The set $\mathcal{S}_{L,\hat{k}_p}$ can be further decomposed and computed as:

$$\mathcal{S}_{L,\hat{k}_p} = \mathcal{S}_{L,\hat{k}_p}^+ \cup \mathcal{S}_{L,\hat{k}_p}^- ,$$

where

$$\mathcal{S}_{L,\hat{k}_p}^+ = \{(\hat{k}_p, k_i) \notin \mathcal{S}_{N,\hat{k}_p} : \exists \omega \in \Omega^+ . k_i = \sqrt{M(\omega)}\} , \quad (2.6)$$

$$\mathcal{S}_{L,\hat{k}_p}^- = \{(\hat{k}_p, k_i) \notin \mathcal{S}_{N,\hat{k}_p} : \exists \omega \in \Omega^- . k_i = -\sqrt{M(\omega)}\} , \quad (2.7)$$

$$\begin{aligned} \Omega^+ &= \left\{ \omega : \omega > 0, M(\omega) \geq 0, \right. \\ &\quad \left. L(\omega) = \frac{\pi + \angle[(\sqrt{M(\omega)} + j\hat{k}_p\omega)R_0(j\omega)]}{\omega} \leq d \right\} , \end{aligned} \quad (2.8)$$

$$\begin{aligned} \Omega^- &= \left\{ \omega : \omega > 0, M(\omega) \geq 0, \right. \\ &\quad \left. \frac{\pi + \angle[(-\sqrt{M(\omega)} + j\hat{k}_p\omega)R_0(j\omega)]}{\omega} \leq d \right\} , \end{aligned} \quad (2.9)$$

$$M(\omega) = \frac{1}{|R_0(j\omega)|^2} - \hat{k}_p^2 \omega^2 , \quad (2.10)$$

$$R_0(s) = \frac{P_0(s)}{s} . \quad (2.11)$$

By convention, we restrict the *phase* function, $\angle[z]$, of a complex number, z , in the interval $[-\pi, \pi)$.

2.3 Computing $\mathcal{S}_R$ for TCP-AQM PI Controllers

In this section, we compute $\mathcal{S}_R$ for the PI controller of Figure 2.2. Henceforth, the analysis assumes that $k_p, k_i \geq 0$: negative gains are counterintuitive in operational terms because they lead to a decrease in the sending rate when the queue length is less than the target value. Although negative gains are disregarded as operationally meaningless, they can formally stabilize the closed-loop system because the open-loop is stable and can tolerate a slightly destabilizing controller.

2.3.1 Computing $\mathcal{S}_0$

By dropping the delay term, e^{-sd} , from $P(s)$, we obtain that

$$P_0(s) = \frac{B}{(s + \alpha)(s + \beta)}.$$

The characteristic equation of the closed loop-system in Figure 2.2 becomes:

$$1 + G(s) \cdot P_0(s) = 1 + \frac{k_p s + k_i}{s} \cdot \frac{B}{(s + \alpha)(s + \beta)} = 0,$$

which is equivalent to

$$s^3 + (\alpha + \beta)s^2 + (\alpha\beta + Bk_p)s + Bk_i = 0. \quad (2.12)$$

To compute $\mathcal{S}_0$, we construct the Routh array [23] as follows:

$$\begin{array}{lcl} s^3 & : & 1 \qquad \qquad \alpha\beta + Bk_p \\ s^2 & : & \alpha + \beta \qquad \qquad Bk_i \\ s^1 & : & (\alpha\beta + Bk_p) - Bk_i / (\alpha + \beta) \qquad 0 \\ s^0 & : & Bk_i \end{array}$$

A necessary and sufficient condition for stability is that all entries in the first column (after the colon) are positive [23, p. 215]. This condition reduces to the following

inequalities:

1. $\alpha + \beta > 0$, which is always true (the network parameters, N , C , and d , cannot be negative).
2. $Bk_i > 0$, which yields $k_i > 0$ since B is always positive (the network parameters, N and C , cannot be negative).
3. $(\alpha\beta + Bk_p) - Bk_i/(\alpha + \beta) > 0$, which reduces to $k_i < (\alpha + \beta)(\alpha\beta + Bk_p)/B$

Combining the last two conditions defines the following range of stabilizing k_i values with the upper boundary being a function of k_p : $0 < k_i < k_{i,\max}$, where

$$k_{i,\max} = \frac{(\alpha + \beta)(\alpha\beta + Bk_p)}{B}. \quad (2.13)$$

Moreover, for a feasible solution $(\alpha + \beta)(\alpha\beta + Bk_p)/B$ must be positive. This gives the range of stabilizing k_p values, i.e., $k_p > -\alpha\beta/B$, which is always satisfied since only non-negative gains are considered in this analysis. Consequently,

$$\mathcal{S}_0 = \{(k_p, k_i) : k_p > 0, 0 < k_i < k_{i,\max}\},$$

which is depicted in Figure 2.3.

2.3.2 Computing $\mathcal{S}_N$

Since

$$\lim_{s \rightarrow \infty} \left| \frac{(k_p s + k_i)P_0(s)}{s} \right| = \lim_{s \rightarrow \infty} \left| \frac{(k_p s + k_i)B}{s(s + \alpha)(s + \beta)} \right| = 0 < 1,$$

we have that $\mathcal{S}_N = \emptyset$ by definition (2.3) of $\mathcal{S}_N$. Thus, $\mathcal{S}_1 = \mathcal{S}_0$; see Figure 2.3.

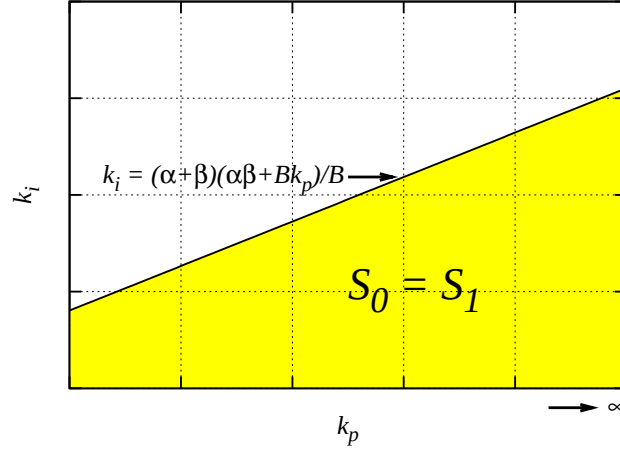


Figure 2.3: The stabilizing region of k_p and k_i gains for the delay-free closed-loop system, i.e., $\mathcal{S}_0$. The area is under the line with a slope of $(\alpha + \beta)$ and a y -intercept of $\alpha\beta(\alpha + \beta)/B$. Moreover, since $\mathcal{S}_N = \emptyset$, $\mathcal{S}_1 = \mathcal{S}_0$.

2.3.3 Computing $\mathcal{S}_L$ and $\mathcal{S}_R$

First, we give an outline of the proof and then we turn to the formal derivations. The stability region $\mathcal{S}_R$ will be derived by using some of the tools in Section 2.2.5. Because we consider only positive gain values, we ignore the two cases of (2.7) and (2.9). Using the fact that $\mathcal{S}_{R, \hat{k}_p} = \mathcal{S}_{1, \hat{k}_p} \setminus \mathcal{S}_{L, \hat{k}_p}^+ = \mathcal{S}_{1, \hat{k}_p} \setminus (\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p})$, we can facilitate the derivation of $\mathcal{S}_R$ by considering $\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p}$ instead of $\mathcal{S}_{L, \hat{k}_p}^+$. In turn, $\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p}$ can be obtained by using the restriction of Ω^+ to the case in which $\sqrt{M(\omega)} \in \mathcal{S}_{1, \hat{k}_p}$ (see (2.6)), and we use this restriction in the derivation of $\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p}$. After computing $\mathcal{S}_{1, \hat{k}_p} \setminus (\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p})$, the stability region $\mathcal{S}_R$ is obtained as in (2.5).

We now turn to the formal derivation of $\mathcal{S}_L$ and $\mathcal{S}_R$. Throughout the chapter, simple proofs and arguments will be omitted tacitly.

For the transfer function in (2.2), $R_0(s)$ and $M(\omega)$ are given as follows:

$$\begin{aligned} R_0(s) &= \frac{B}{s(s + \alpha)(s + \beta)} \\ &= \frac{B}{s^3 + (\alpha + \beta)s^2 + \alpha\beta s}, \end{aligned} \quad (2.14)$$

$$\begin{aligned} M(\omega) &= \frac{1}{|R_0(j\omega)|^2} - \hat{k}_p^2 \omega^2 \\ &= \frac{(\alpha + \beta)^2 \omega^4 + \omega^2(\omega^2 - \alpha\beta)^2}{B^2} - \hat{k}_p^2 \omega^2 \\ &= \frac{\omega^2 \cdot Q(\omega)}{B^2}, \end{aligned} \quad (2.15)$$

where

$$Q(\omega) = \omega^4 + (\alpha^2 + \beta^2)\omega^2 + (\alpha^2\beta^2 - B^2\hat{k}_p^2). \quad (2.16)$$

We start by computing the conditions in (2.8). First, the following sequence of lemmas 2.3.1, 2.3.3, and 2.3.5 gives a lower bound on $\omega \in \Omega^+$.

Lemma 2.3.1 *For the biquadratic polynomial $Q(\omega)$ defined in (2.16), $Q'(0) = 0$ and $Q'(\omega) > 0$ for $\omega > 0$, i.e., $Q(\omega)$ is strictly increasing for $\omega > 0$.*

Proof Since $Q'(\omega) = 4\omega^3 + 2(\alpha^2 + \beta^2)\omega = 2\omega(2\omega^2 + \alpha^2 + \beta^2)$ and $\omega > 0$, $Q'(\omega) > 0$, i.e., $Q(\omega)$ is strictly increasing for $\omega > 0$. ■

Definition 2.3.2 *Define $\gamma = \alpha\beta/B$ —we will refer to γ as the critical value of the proportional gain. If $\hat{k}_p > \gamma$, then the function $\omega_{\min}(\hat{k}_p)$ is*

$$\omega_{\min}(\hat{k}_p) = \sqrt{\frac{-(\alpha^2 + \beta^2) + \sqrt{(\alpha^2 - \beta^2)^2 + 4B^2\hat{k}_p^2}}{2}}.$$

For simplicity, the argument $\hat{k}_p$ will be omitted when it is clear from the context, i.e., we use $\omega_{\min}$.

Lemma 2.3.3 *As defined above, $\omega_{\min}$ has the following properties:*

- $\omega_{\min}$ is real and positive,
- $\omega_{\min}$ is strictly increasing,
- $\lim_{\hat{k}_p \rightarrow \gamma^+} \omega_{\min}(\hat{k}_p) = 0$, and
- $Q(\omega_{\min}) = 0$.

Proof First, note that $(\alpha^2 - \beta^2)^2 + 4B^2\hat{k}_p^2 > 0$. Since

$$\begin{aligned} (\alpha^2 - \beta^2)^2 + 4\hat{k}_p^2 B^2 &> \alpha^4 + \beta^4 - 2\alpha^2\beta^2 + 4\frac{\alpha^2\beta^2}{B^2}B^2 \\ &= (\alpha^2 + \beta^2)^2, \end{aligned}$$

we have that $\omega_{\min} \in \mathbb{R}$ and $\omega_{\min} > 0$. Since the square root function is increasing and when $\hat{k}_p > \gamma$, $4B^2\hat{k}_p^2$ is strictly increasing, it follows that $\omega_{\min}(\hat{k}_p)$ is strictly increasing as well. The last two properties can be proven by substitution, i.e., $\omega_{\min}(\gamma^+) = 0$ and $Q(\omega_{\min}) = 0$. ■

Definition 2.3.4 *We define the set Ω as follows*

$$\Omega = \begin{cases} (0, \infty) & \text{if } \hat{k}_p \leq \gamma, \\ [\omega_{\min}, \infty) & \text{otherwise.} \end{cases}$$

Lemma 2.3.5 *For $\omega > 0$, the inequality $Q(\omega) \geq 0$ holds if and only if $\omega \in \Omega$.*

Proof Since $\Omega \subseteq (0, \infty)$, it is enough to show that $Q(\omega) \geq 0$ if $\omega \in \Omega$ and $Q(\omega) < 0$ if $\omega \in (0, \infty) \setminus \Omega$. From Lemma 2.3.1, $Q(\omega) > Q(0) = \alpha^2\beta^2 - B^2\hat{k}_p^2$. Now, there are two cases

- If $\hat{k}_p \leq \gamma$, $Q(\omega) > \alpha^2\beta^2 - B^2\hat{k}_p^2 \geq 0$ for $\omega > 0$.

- If $\hat{k}_p > \gamma$, $Q(0) = -|\alpha^2\beta^2 - B^2\hat{k}_p^2| < 0$. Because of the monotonicity of $Q(\omega)$, there exists one and only one $\omega_0 > 0$ such that $Q(\omega_0) = 0$. From Lemma 2.3.3, $\omega_0 = \omega_{\min}$. Now, for $\omega \geq \omega_{\min}$, $Q(\omega) \geq Q(\omega_{\min}) = 0$. For $0 < \omega < \omega_{\min}$, $Q(\omega) < Q(\omega_{\min}) = 0$. ■

The following corollary is a consequence of Lemma 2.3.5 and (2.15).

Corollary 2.3.6 *The two conditions $\omega > 0$ and $M(\omega) \geq 0$ hold if and only if $\omega \in \Omega$.*

Before proceeding to compute the third condition $L(\omega) \leq d$ in (2.8), we give an upper bound on ω . This bound will greatly simplify the evaluation of $L(\omega)$. The upper bound on ω is derived from looking ahead to (2.6) and excluding values of ω that would give $k_i = \sqrt{M(\omega)} \notin \mathcal{S}_{1,\hat{k}_p}$. The bound will be derived utilizing the following two lemmas 2.3.7 and 2.3.8.

Lemma 2.3.7 *The function $k_i(\omega) = \sqrt{M(\omega)}$ is strictly increasing in the interval Ω .*

Proof Since $k_i(\omega) = (1/B)\sqrt{\omega^2 Q(\omega)}$, it is enough to show that for $\omega \in \Omega$, $Y(\omega) = \omega^2 Q(\omega)$ is strictly increasing. For $\omega \in \Omega$, $Y'(\omega) = 2\omega Q(\omega) + \omega^2 Q'(\omega) > 0$. ■

Lemma 2.3.8 *If $\hat{k}_p \leq \gamma$, $\lim_{\omega \rightarrow 0^+} [\sqrt{M(\omega)}] = 0$. Otherwise, i.e., $\hat{k}_p > \gamma$, $\sqrt{M(\omega_{\min})} = 0$. In both cases, $\lim_{\omega \rightarrow \infty} \sqrt{M(\omega)} = +\infty$.*

Proof If $\hat{k}_p \leq \gamma$,

$$\begin{aligned} \lim_{\omega \rightarrow 0^+} [\sqrt{M(\omega)}] &= \lim_{\omega \rightarrow 0^+} [(1/B)\sqrt{\omega^2 Q(\omega)}] \\ &= (1/B)\sqrt{(0)Q(0)} = 0. \end{aligned}$$

If $\hat{k}_p > \gamma$, $\sqrt{M(\omega_{\min})} = (1/B)\sqrt{\omega_{\min}^2 Q(\omega_{\min})} = 0$. In both cases, $\lim_{\omega \rightarrow \infty} \sqrt{M(\omega)} = +\infty$. ■

Since $k_i(\omega) = \sqrt{M(\omega)}$ increases monotonically with ω and because of Lemma 2.3.8, there exists one and only one $\omega_{\max}$ such that $\sqrt{M(\omega_{\max})} = k_{i,\max}$ (see (2.13)). Moreover, for $\omega \in \Omega$, $k_i(\omega) < k_{i,\max}$ if and only if $\omega < \omega_{\max}$. Therefore, for $\omega \in \Omega$, $(\hat{k}_p, k_i(\omega)) \in \mathcal{S}_{1,\hat{k}_p}$ if and only if $\omega < \omega_{\max}$.

The value of $\omega_{\max}$ is given by the following lemma.

Lemma 2.3.9 *The value of*

$$\omega_{\max} = \sqrt{\alpha\beta + B\hat{k}_p}$$

is the only positive real value that solves

$$k_{i,\max} = \sqrt{M(\omega_{\max})} = \frac{(\alpha + \beta)(\alpha\beta + B\hat{k}_p)}{B}. \quad (2.17)$$

Proof First, note that $\omega_{\max} \in \mathbb{R}$ and that $\omega_{\max} > 0$. Also, $\omega_{\max}$ satisfies (2.17). The uniqueness of such value follows directly from the monotonicity of $\sqrt{M(\omega)}$ in the interval $\omega \in \Omega$. ■

In the sequel, we confine our analysis to the sets Ω_u and Ω_u^+ that use $\omega_{\max}$ and are given in the following definition.

Definition 2.3.10 *The sets Ω_u and Ω_u^+ are defined as*

$$\begin{aligned} \Omega_u &= \Omega \cap (0, \omega_{\max}), \\ \Omega_u^+ &= \Omega^+ \cap (0, \omega_{\max}). \end{aligned}$$

We next study the third condition in (2.8)

$$L(\omega) = \frac{\pi + \angle[(\sqrt{M(\omega)} + j\hat{k}_p\omega)R_0(j\omega)]}{\omega} \leq d. \quad (2.18)$$

The following two lemmas 2.3.11 and 2.3.13 will simplify the study of $L(\omega)$. By conven-

tion, we consider $\arctan(\pm\infty) = \pm\pi/2$.

Lemma 2.3.11 *Let $\theta_1(\omega)$ and $\theta_2(\omega)$ be*

$$\begin{aligned}\theta_1(\omega) &= \arctan \left[\frac{B\hat{k}_p}{\sqrt{Q(\omega)}} \right], \\ \theta_2(\omega) &= \pi - \arctan \left[\frac{\omega^2 - \alpha\beta}{(\alpha + \beta)\omega} \right].\end{aligned}$$

Then, if $\sqrt{\alpha\beta} < \omega < \omega_{\max}$, $\theta_1(\omega) + \theta_2(\omega) > \pi$.

Proof Since the $\arctan$ is a monotonically increasing function, it is enough to show that

$$B\hat{k}_p(\alpha + \beta)\omega > (\omega^2 - \alpha\beta)\sqrt{Q(\omega)}.$$

Define $V(\omega)$ as

$$V(\omega) = \omega^2(\omega^2 - 2\alpha\beta) + \alpha^2\beta^2 - B^2\hat{k}_p^2.$$

Note that if $\omega > \sqrt{\alpha\beta}$, $V(\omega)$ is a strictly increasing function, and thus for $\sqrt{\alpha\beta} < \omega < \omega_{\max}$,

$$\begin{aligned}V(\omega) &< V(\omega_{\max}) \\ &< \omega_{\max}^2(\omega_{\max}^2 - 2\alpha\beta) + \alpha^2\beta^2 - B^2\hat{k}_p^2 \\ &= 0.\end{aligned}$$

Therefore, $\omega^4 + \alpha^2\beta^2 - B^2\hat{k}_p^2 < 2\alpha\beta\omega^2$.

Now,

$$\begin{aligned}
& (\omega^2 - \alpha\beta)\sqrt{Q(\omega)} \\
&= (\omega^2 - \alpha\beta)\sqrt{\omega^4 + (\alpha^2 + \beta^2)\omega^2 + \alpha^2\beta^2 - B^2\hat{k}_p^2} \\
&< (\omega^2 - \alpha\beta)\sqrt{(\alpha^2 + \beta^2)\omega^2 + 2\alpha\beta\omega^2} \\
&= (\omega^2 - \alpha\beta)(\alpha + \beta)\omega \\
&< (\omega_{\max}^2 - \alpha\beta)(\alpha + \beta)\omega \\
&= B\hat{k}_p(\alpha + \beta)\omega,
\end{aligned}$$

which proves the lemma. ■

Definition 2.3.12 We define $\phi(\omega)$, $\phi_1(\omega)$, and $\phi_2(\omega)$ as follows

$$\begin{aligned}
\phi(\omega) &= \pi + \angle[(\sqrt{M(\omega)} + j\hat{k}_p\omega)R_0(j\omega)], \\
\phi_1(\omega) &= \theta_1(\omega) \text{ and } \phi_2(\omega) = \theta_2(\omega) - \pi.
\end{aligned}$$

Lemma 2.3.13 If $\omega \in \Omega_u$, then $\phi(\omega) = \phi_1(\omega) + \phi_2(\omega)$.

Proof Define

$$\begin{aligned}
\tilde{\phi}(\omega) &= \phi(\omega) - \pi = \angle[(\sqrt{M(\omega)} + j\hat{k}_p\omega)R_0(j\omega)], \\
z &= -(\alpha + \beta)\omega^2 + j\omega(\omega^2 - \alpha\beta).
\end{aligned}$$

Also, define $\tilde{\phi}_2(\omega)$ as

$$\tilde{\phi}_2(\omega) = \angle[R_0(j\omega)] = \angle[z].$$

Note that $\phi_1(\omega) = \angle[\sqrt{M(\omega)} + j\hat{k}_p\omega] \in (0, \pi/2]$. Now, $\tilde{\phi}(\omega) = \phi_1(\omega) + \tilde{\phi}_2(\omega) + 2\pi n$, where $n \in \mathbb{Z}$ ensures that $\tilde{\phi}(\omega) \in [-\pi, \pi)$. There are two cases:

- When $\omega \leq \sqrt{\alpha\beta}$, z belongs to the third quadrant in the complex plane, and so

$$\tilde{\phi}_2(\omega) = -\pi - \arctan \left[\frac{\omega^2 - \alpha\beta}{(\alpha + \beta)\omega} \right] = \phi_2(\omega) - \pi .$$

Since $\phi_1(\omega) \in (0, \pi/2]$ and $\tilde{\phi}_2(\omega) \in [-\pi, -\pi/2)$, $\phi_1(\omega) + \tilde{\phi}_2(\omega) \in (-\pi, 0) \subset [-\pi, \pi)$, and thus $n = 0$. Consequently, $\phi(\omega) = \pi + \tilde{\phi}(\omega) = \pi + \phi_1(\omega) + \tilde{\phi}_2(\omega) = \phi_1(\omega) + \phi_2(\omega)$.

- When $\omega > \sqrt{\alpha\beta}$, z belongs to the second quadrant in the complex plane, and so

$$\tilde{\phi}_2(\omega) = \pi - \arctan \left[\frac{\omega^2 - \alpha\beta}{(\alpha + \beta)\omega} \right] = \theta_2(\omega) = \phi_2(\omega) + \pi .$$

Since $\phi_1(\omega) \in (0, \pi/2]$, $\tilde{\phi}_2(\omega) \in (\pi/2, \pi)$, and $\phi_1(\omega) + \tilde{\phi}_2(\omega) > \pi$ (Lemma 2.3.11), $\phi_1(\omega) + \tilde{\phi}_2(\omega) \in (\pi, 3\pi/2)$. Thus, $n = -1$. Consequently, $\phi(\omega) = \pi + \tilde{\phi}(\omega) = \pi + \phi_1(\omega) + \tilde{\phi}_2(\omega) - 2\pi = \phi_1(\omega) + \phi_2(\omega)$. ■

Utilizing Lemma 2.3.13, we express $L(\omega)$ as

$$L(\omega) = \frac{\phi(\omega)}{\omega} = \frac{\phi_1(\omega) + \phi_2(\omega)}{\omega} .$$

We then state the following result.

Lemma 2.3.14 *The function $L(\omega)$ is strictly decreasing for $\omega \in \Omega_u$.*

Proof First, note that $\phi \in [0, 2\pi) \geq 0$. Next, for $\phi_1(\omega)$,

$$\phi'_1(\omega) = -\frac{B\hat{k}_p[2\omega^3 + (\alpha^2 + \beta^2)\omega]}{[Q(\omega) + B^2\hat{k}_p^2]\sqrt{Q(\omega)}} < 0$$

if $\omega \in \Omega_u$. As for $\phi_2(\omega)$,

$$\phi'_2(\omega) = -\frac{(\alpha + \beta)(\omega^2 + \alpha\beta)}{(\omega^2 + \alpha^2)(\omega^2 + \beta^2)} < 0$$

if $\omega \in \Omega_u$. Since $\phi(\omega) = \phi_1(\omega) + \phi_2(\omega)$ for $\omega \in \Omega_u$, $\phi'(\omega) < 0$ for $\omega \in \Omega_u$.

Since $L(\omega) = \phi(\omega)/\omega$, taking the derivative of both sides yields

$$L'(\omega) = \frac{\omega \cdot \phi'(\omega) - \phi(\omega)}{\omega^2} < 0$$

for $\omega \in \Omega_u$ (note that $\phi(\omega) \geq 0$ and $\omega \cdot \phi'(\omega) < 0$). Hence, $L(\omega)$ is strictly decreasing function of ω for $\omega \in \Omega_u$. ■

Now, we are ready to determine the sets Ω_u^+ , $\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p}$, and $\mathcal{S}_{R, \hat{k}_p}$. Keep in mind, as reading through the proofs of Lemma 2.3.15 and Corollary 2.3.16, to consult Figure 2.4 that illustrates the computation of Ω_u^+ and $\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p}$.

Lemma 2.3.15 *The set Ω_u^+ is given by $\Omega_u^+ = [\omega^+, \omega_{\max})$, where*

$$\omega^+ = \begin{cases} \omega_{\min} & \text{if } \hat{k}_p > \gamma \text{ and } L(\omega_{\min}) \leq d \\ \omega_d & \text{otherwise,} \end{cases}$$

and ω_d is the unique value such that $\omega_d > 0$, $\omega_d > \omega_{\min}$ if $\hat{k}_p > \gamma$ and $L(\omega_{\min}) > d$, and $L(\omega_d) = d$.

Proof By definition, $\Omega_u^+ = \{\omega \in \Omega_u : L(\omega) \leq d\}$. First, note that $L(\omega)$ is continuous and strictly decreasing of $\omega \in \Omega_u$ and that $L(\omega_{\max}) = 0$. Consider the following cases:

- If $\hat{k}_p \leq \gamma$, $\Omega_u = (0, \omega_{\max})$. Since $L(\omega) = \phi(\omega)/\omega$ and $\phi(0) = \phi_1(0) + \phi_2(0) > 0$, $\lim_{\omega \rightarrow 0^+} L(\omega) = +\infty > d$. Then, there exists one and only one ω_d such that $0 < \omega_d < \omega_{\max}$ and $L(\omega_d) = d$. Therefore, $L(\omega) \leq d$ if and only if $\omega_d \leq \omega < \omega_{\max}$. Hence, $\Omega_u^+ = [\omega_d, \omega_{\max}) = [\omega^+, \omega_{\max})$; see Figure 2.4(a).
- If $\hat{k}_p > \gamma$ and $L(\omega_{\min}) > d$, there exists one and only one ω_d such that $\omega_{\min} < \omega_d < \omega_{\max}$ and $L(\omega_d) = d$. Then, the analysis proceeds similarly to the previous case; see Figure 2.4(b).

- If $\hat{k}_p > \gamma$ and $L(\omega_{\min}) \leq d$, $\Omega_u^+ = [\omega_{\min}, \omega_{\max}) = [\omega^+, \omega_{\max})$; see Figure 2.4(c). ■

Corollary 2.3.16 *The set $\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p}$ is given by*

$$\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p} = \{(\hat{k}_p, k_i) : k_i \in [\sqrt{M(\omega^+)}, k_{i, \max})\}.$$

Proof

$$\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p} = \{(\hat{k}_p, \sqrt{M(\omega)}) : \omega \in \Omega_u^+\}.$$

Using lemmas 2.3.7–2.3.9, we obtain that

$$\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p} = \{(\hat{k}_p, k_i) : k_i \in [\sqrt{M(\omega^+)}, k_{i, \max})\}.$$

■

Corollary 2.3.17 *The set $\mathcal{S}_{R, \hat{k}_p}$ is given by*

$$\mathcal{S}_{R, \hat{k}_p} = \{(\hat{k}_p, k_i) : k_i \in (0, \sqrt{M(\omega^+)})\}.$$

Proof From the definition of $\mathcal{S}_{R, \hat{k}_p} = \mathcal{S}_{1, \hat{k}_p} \setminus \mathcal{S}_{L, \hat{k}_p}^+$, we obtain that

$$\mathcal{S}_{R, \hat{k}_p} = \{(\hat{k}_p, k_i) : k_i \in (0, k_{i, \max}) \setminus [\sqrt{M(\omega^+)}, k_{i, \max})\},$$

which proves the corollary. ■

Remark 2.3.18 *Note that when $\hat{k}_p > \gamma$ and $L(\omega_{\min}) \leq d$, $\Omega_u^+ = [\omega_{\min}, \omega_{\max})$, $\mathcal{S}_{L, \hat{k}_p}^+ \cap \mathcal{S}_{1, \hat{k}_p} = \{(\hat{k}_p, k_i) : k_i \in [0, k_{i, \max})\}$, and $\mathcal{S}_{R, \hat{k}_p} = \emptyset$; see Figure 2.4(c).*

Remark 2.3.18 enables us to obtain an exact upper bound, $k_{p, \max}$, on the values of stabilizing k_p .

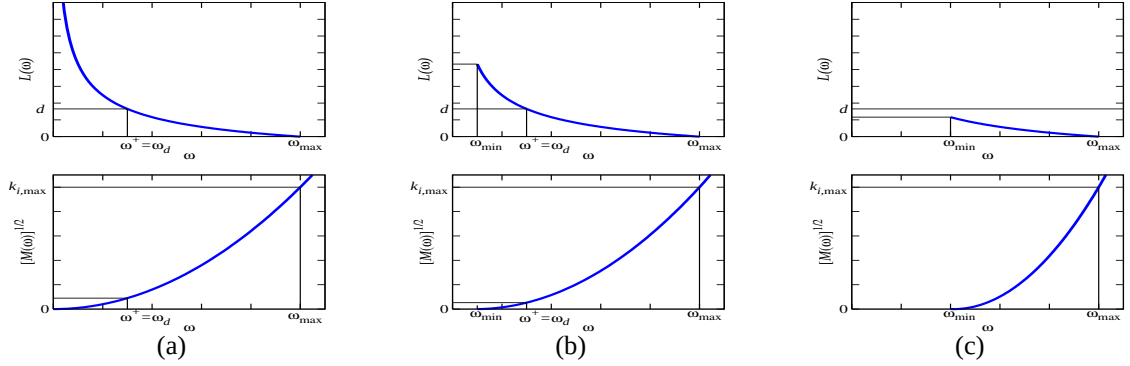


Figure 2.4: $L(\omega)$ versus ω (upper) and $\sqrt{M(\omega)}$ versus ω (lower): (a) when $\hat{k}_p \leq \gamma$; (b) when $\hat{k}_p > \gamma$ and $L(\omega_{\min}) > d$; (c) when $\hat{k}_p > \gamma$ and $L(\omega_{\min}) \leq d$.

Definition 2.3.19 Define $\tilde{L}(\hat{k}_p)$ as

$$\tilde{L}(\hat{k}_p) = L\left(\omega_{\min}(\hat{k}_p)\right) .$$

Lemma 2.3.20 The function $\tilde{L}(\hat{k}_p)$ is given by

$$\tilde{L}(\hat{k}_p) = \frac{1}{\omega_{\min}} \left(\frac{\pi}{2} + \phi_2(\omega_{\min}) \right) .$$

Moreover, $\tilde{L}(\hat{k}_p)$ is a strictly decreasing function of $\hat{k}_p$,

$$\lim_{\hat{k}_p \rightarrow \gamma^+} \tilde{L}(\hat{k}_p) = +\infty , \text{ and } \lim_{\hat{k}_p \rightarrow \infty} \tilde{L}(\hat{k}_p) = 0 .$$

Proof By Lemma 2.3.3, $\sqrt{Q(\omega_{\min})} = 0$. Thus,

$$\tilde{L}(\hat{k}_p) = L(\omega_{\min}) = \frac{1}{\omega_{\min}} \left(\frac{\pi}{2} + \phi_2(\omega_{\min}) \right) .$$

Define $\psi(\omega_{\min}) = \pi/2 + \phi_2(\omega_{\min})$. First, note that since $\omega_{\min} > 0$,

$$\phi_2(\omega_{\min}) = -\arctan\{[\omega_{\min}^2 - \alpha\beta]/[(\alpha + \beta)\omega_{\min}]\} \in (-\pi/2, \pi/2) .$$

Since for $\omega_{\min} > 0$,

$$\frac{d\psi(\omega_{\min})}{d\omega_{\min}} = -\frac{(\alpha + \beta)(\omega_{\min}^2 + \alpha\beta)}{(\omega_{\min}^2 + \alpha^2)(\omega_{\min}^2 + \beta^2)} < 0$$

and $\psi(\omega_{\min}) \in (0, \pi) > 0$, then

$$\frac{dL(\omega_{\min})}{d\omega_{\min}} = \frac{\omega_{\min} \cdot \psi'(\omega_{\min}) - \psi(\omega_{\min})}{\omega_{\min}^2} < 0 .$$

By the chain rule and from Lemma 2.3.3,

$$\frac{d\tilde{L}(\hat{k}_p)}{d\hat{k}_p} = \frac{dL(\omega_{\min})}{d\omega_{\min}} \frac{d\omega_{\min}}{d\hat{k}_p} < 0 .$$

As for the limits part of the Lemma,

$$\begin{aligned} \lim_{\hat{k}_p \rightarrow \gamma^+} \tilde{L}(\hat{k}_p) &= \lim_{\omega_{\min} \rightarrow 0^+} L(\omega_{\min}) = +\infty , \\ \lim_{\hat{k}_p \rightarrow \infty} \tilde{L}(\hat{k}_p) &= \lim_{\omega_{\min} \rightarrow \infty} L(\omega_{\min}) = 0 . \end{aligned}$$

■

Lemma 2.3.21 *The set $\mathcal{S}_{R, \hat{k}_p} \neq \emptyset$ if and only if $\hat{k}_p < k_{p, \max}$, where $k_{p, \max}$ is the unique value such that*

$$\tilde{L}(k_{p, \max}) = d .$$

Proof First, we prove the uniqueness of $k_{p, \max}$. Note that $\tilde{L}(\hat{k}_p)$ is defined only for $\hat{k}_p > \gamma$.

From Lemma 2.3.20, there exists one and one $k_{p, \max} > \gamma$ such that $\tilde{L}(k_{p, \max}) = d$.

We next prove that if $\hat{k}_p \geq k_{p, \max}$, then $\mathcal{S}_{R, \hat{k}_p} = \emptyset$.

First, $\hat{k}_p \geq k_{p, \max} > \gamma$. Since $\tilde{L}(\hat{k}_p)$ is strictly decreasing (Lemma 2.3.20),

$$L(\omega_{\min}(\hat{k}_p)) = \tilde{L}(\hat{k}_p) \leq \tilde{L}(k_{p, \max}) = d .$$

Therefore, from Remark 2.3.18, $\mathcal{S}_{R,\hat{k}_p} = \emptyset$.

Now, it remains to prove the last step, if $\hat{k}_p < k_{p,\max}$, then $\mathcal{S}_{R,\hat{k}_p} \neq \emptyset$. When $\hat{k}_p < k_{p,\max}$, there are two cases: $\hat{k}_p \leq \gamma$ or $\gamma < \hat{k}_p < k_{p,\max}$. If $\hat{k}_p \leq \gamma$, then since $\omega_d > 0$, $\sqrt{M(\omega_d)} = \sqrt{M(\omega^+)} > 0$ by lemmas 2.3.7 and 2.3.8. Thus from Corollary 2.3.17, $\mathcal{S}_{R,\hat{k}_p} \neq \emptyset$. If $\gamma < \hat{k}_p < k_{p,\max}$, then from Lemma 2.3.20,

$$\tilde{L}(\hat{k}_p) > \tilde{L}(k_{p,\max}) = d .$$

Since $\omega_d > \omega_{\min}$, $\sqrt{M(\omega_d)} = \sqrt{M(\omega^+)} > 0$. Thus from Corollary 2.3.17, $\mathcal{S}_{R,\hat{k}_p} \neq \emptyset$, which proves the lemma. ■

We now state the main result in this chapter.

Theorem 2.3.22 *The complete region of stabilizing k_p and k_i gain values is $\mathcal{S}_R$, where*

$$\mathcal{S}_R = \{(k_p, k_i) : k_p \in (0, k_{p,\max}), k_i \in (0, \sqrt{M(\omega^+)})\} ,$$

and $k_{p,\max}$ and ω^+ were defined in lemmas 2.3.21 and 2.3.15, respectively.

Proof It follows from Lemmas 2.3.1–2.3.21. ■

Theorem 2.3.22 allows us to calculate the complete stability region of a PI controller for given network parameters, N , C , and d . Usually, C is constant and known by the network administrator unlike d and N . The following remark facilitates the computation of $\mathcal{S}_R$ when the values of d and N are unknown.

Remark 2.3.23 *As in [10], it can be shown that $\mathcal{S}_R$ shrinks as d increases and that $\mathcal{S}_R$ expands as N increases. Therefore, given a minimum number of TCP sessions N_0 , a maximum round-trip delay d_0 , and a link capacity C , $\mathcal{S}_R$ obtained for N_0 , d_0 and C will stabilize the system for all $N \geq N_0$ and $d \leq d_0$. Consequently, precise values of d and N are not required. An overestimated value of d and an underestimated value of N lead to increased stability robustness.*

2.4 Simulations

In this section, we use experiments to

- Validate the theoretical analysis of the previous section by comparing results for controller parameters that are theoretically stable with others that are theoretically unstable.
- Stress the importance of obtaining the complete stability region, $\mathcal{S}_R$, for the PI AQM by showing that some theoretically stable controllers can outperform other stable ones based on [30] and [32].

2.4.1 Simulation Methodology

We consider a simulation environment similar to the one in [30]. There are N TCP sources that share a single bottleneck link with $C = 3750$ packets/second (corresponding to 15 Mbps with average packet size of 500 Bytes). Moreover, $q_0 = 200$ packets and the queue limit is 1000 packets. In our experiments, we use long-lived FTP sessions.

We use Theorem 2.3.22 to plot $\mathcal{S}_R$ for $C = 3750$ packets/second, N and d , where the two parameters N and d will be varied across experiments.

To compare the performance of different controller parameters, we employ the queue's speed of convergence as a comparison criterion. That is, if one controller, $g_1 = (k_p^1, k_i^1)$, causes the queue length, $q(t)$, to converge to q_0 faster than $g_2 = (k_p^2, k_i^2)$, then g_1 is *preferable* to g_2 .

2.4.2 Theoretically Stable and Unstable Controllers

The stability analysis is validated by comparing a controller that is provably stable with one that is provably unstable. Since the stability region depends on various parameters, a provably stable controller is one that falls within the stability region corresponding to the

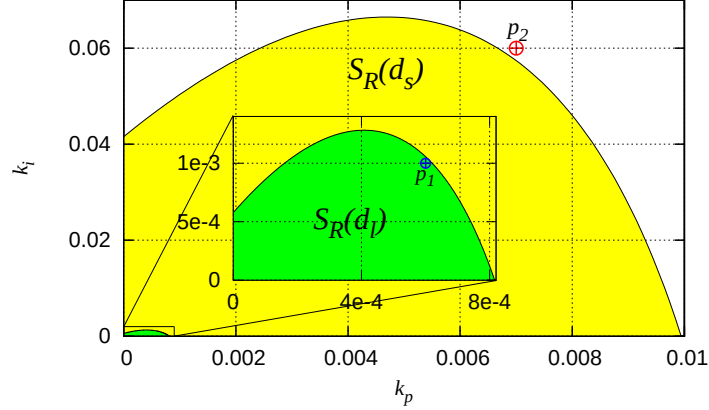


Figure 2.5: The two stability regions corresponding to $d_s = 60$ msec and $d_l = 167$ msec along with the points p_1 , inside $\mathcal{S}_R(d_l)$, and p_2 , outside $\mathcal{S}_R(d_s)$.

most conservative estimate of the parameters. Analogously, a provably unstable controller is one that falls outside the stability region obtained with the most optimistic estimate of the parameters. For example, the stability region depends on the maximum RTT d , which in turn depends on queueing delays. The most conservative estimate of d assumes the longest possible queueing delays; the most optimistic estimate of d assumes that there are no queueing delays. The stability analysis is validated by comparing the behavior of a theoretically stable controller corresponding to the longest RTT with a theoretically unstable controller that violates stability even for the shortest RTT.

Consider the network parameters stated in Section 2.4.1. We assume that there are $N = 100$ FTP sessions having homogeneous end-to-end fixed delays of 60 msec. The shortest expected d is $d_s = 60$ msec (assuming no queueing delays) and the longest expected d is $d_l = 167$ msec (assuming a maximum queue length of 400 packets). Using Theorem 2.3.22, we plot the two $\mathcal{S}_R$'s: $\mathcal{S}_R(d_s)$ and $\mathcal{S}_R(d_l)$ in Figure 2.5. In this pedagogical example, $\mathcal{S}_R(d_s)$ is considered merely for the validation of the theoretical analysis and it ought not to be considered for selecting k_p and k_i for a network with similar parameters. The reason is that to attain stability that is robust to time delay uncertainties, control design should take into account the largest expected d [58]. From Figure 2.5, one can note that

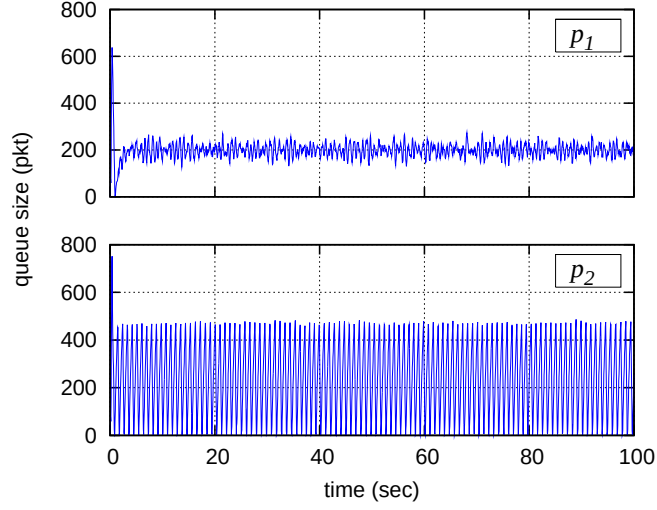


Figure 2.6: The queue size when using the two sets of (k_p, k_i) parameters corresponding to the points p_1 and p_2 .

$\mathcal{S}_R(d_l) \subset \mathcal{S}_R(d_s)$ as implied by the definition of $\mathcal{S}_R$ in Section 2.2.5. Also, one can note that $\mathcal{S}_R(d_s)$ is much larger than $\mathcal{S}_R(d_l)$.

We choose a provably stable controller $p_1 \in \mathcal{S}_R(d_l)$, i.e., inside $\mathcal{S}_R(d_l)$, and a provably unstable controller $p_2 \notin \mathcal{S}_R(d_s)$, i.e., outside $\mathcal{S}_R(d_s)$; see Figure 2.5. Figure 2.6 compares the instantaneous queue length, $q(t)$, for the two sets of (k_p, k_i) parameters defined by the points p_1 and p_2 . With p_1 , the queue exhibits small variations around q_0 . On the contrary, with p_2 , the queue hits zero very frequently and its oscillations are large and severe. To further analyze the nature of $q(t)$ oscillations in Figure 2.6, we use the discrete Fourier transform (DFT) technique [56] to plot the Frequency Spectrum of $q(t)$, which is shown in Figure 2.7. Figure 2.7 confirms that when using p_1 , the queue oscillations are but random fluctuations around q_0 . In contrast, when using p_2 , the oscillations show conspicuously repetitive and persistent pattern. Consistent with the terminology of [27, 44, 48], p_1 , a theoretically stable controller, yields a *stable* regime whereas p_2 , a theoretically unstable controller, yields an *unstable* regime.

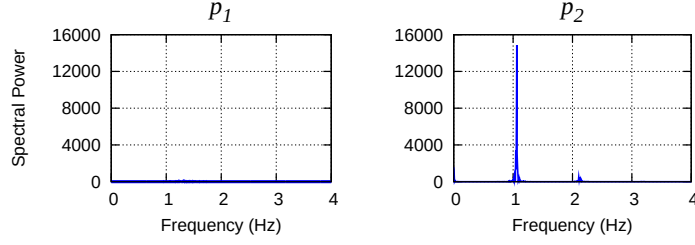


Figure 2.7: The spectral plots of $q(t)$ of Figure 2.6 when using p_1 (left) and p_2 (right).

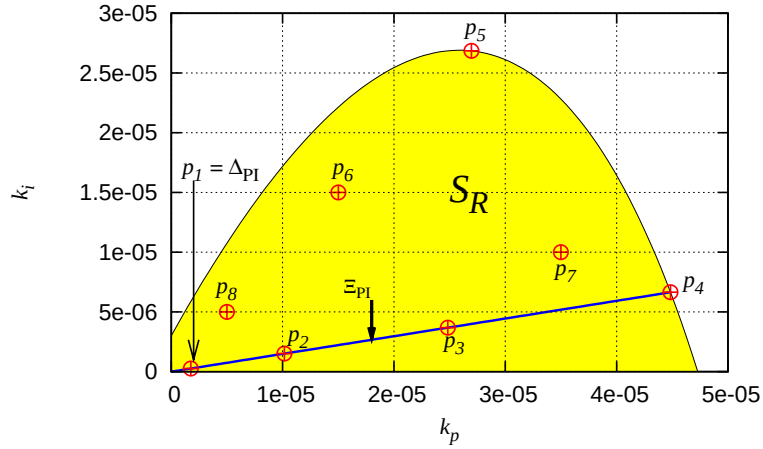


Figure 2.8: $\mathcal{S}_R$ based on the analysis in this chapter, the line segment Ξ_{PI} [32], the point Δ_{PI} [30], and seven other points, p_2 – p_8 , to be used in simulations.

2.4.3 $\mathcal{S}_R$ versus Δ_{PI} and Ξ_{PI}

In this section, we compare the performance of different control parameters chosen inside $\mathcal{S}_R$. The simulation environment uses $N_1 = 25$ FTP sessions that start at $t = 0$ sec, and another $N_2 = 175$ sessions that start at $t = 50$ sec. Propagation delays for all flows are chosen randomly from a uniform distribution in $[60, 200]$ msec. The most conservative estimate of d assumes the largest expected d and the most conservative value of N assumes the least expected N (see [10] for details). Consequently, we obtain $\mathcal{S}_R$ for $C = 3750$ pkt/sec, $N = 25$, and $d = 0.3$ sec; see Figure 2.8. The line segment Ξ_{PI} and the point Δ_{PI} (see Section 2.2.4) are superimposed on the same figure. Note that Ξ_{PI} starts from the origin and terminates exactly at the boundary of $\mathcal{S}_R$ and that $\Delta_{PI} \in \Xi_{PI}$.

First, we study the performance of PI parameters proposed by previous work [30, 32]. We select four points on Ξ_{PI} (Figure 2.8):

- $p_1 = \Delta_{\text{PI}}$, the point prescribed in [30].
- p_2 , an arbitrary point between p_1 and p_3 .
- p_3 , a point that is approximately in the middle of the k_p range.
- p_4 , the point that terminates Ξ_{PI} and gives the largest possible values of both k_p and k_i on that line.

As clearly seen in Figure 2.9, the performance improves by moving the point on Ξ_{PI} toward the right, i.e., increasing the PI gains. Conservative gains of Δ_{PI} cause a large overshoot in $q(t)$ and yield a drastically slow convergence; see Figure 2.10. Given that previous research experiments used only Δ_{PI} in the PI simulations, Figs. 2.8 and 2.10 explain clearly the sluggish responsiveness previously observed about PI. Among the points on Ξ_{PI} , p_4 achieves the best performance.

To explore the performance of the PI controller for other points inside $\mathcal{S}_R$, four points— p_5 , p_6 , p_7 and p_8 —are chosen inside $\mathcal{S}_R$ but off Ξ_{PI} ; see Figure 2.8. When using these points, the corresponding queue response is shown in Figure 2.11. The same figure also shows the response of p_4 . Clearly, p_5 , p_6 , and p_7 show superior performance over p_4 . Among them, p_5 gives the best results.

2.4.4 Discussion

Sections 2.4.2 and 2.4.3 highlight the distinction between stability and performance. Informally, stability assesses whether a system's state converges toward a target trajectory. That is, a stable system is a system whose state converges eventually toward a target trajectory whereas an unstable system is a system whose state never converges toward a target trajectory. On the other hand, performance gauges the quality of a stable system as its

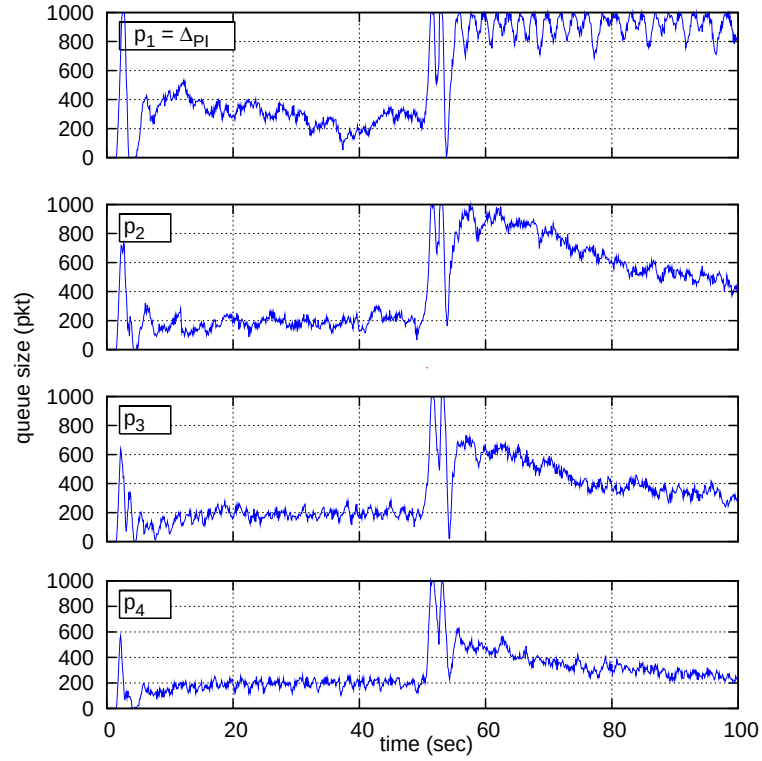


Figure 2.9: Comparing the performance of different points on Ξ_{PI} .

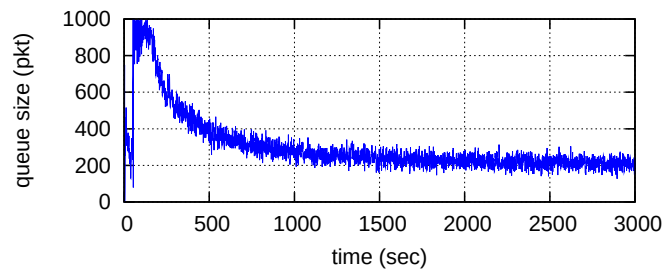


Figure 2.10: Extending the experiment's execution time up to 3000 sec to observe the long-term trend of $q(t)$ when using $p_1 = \Delta_{PI}$.

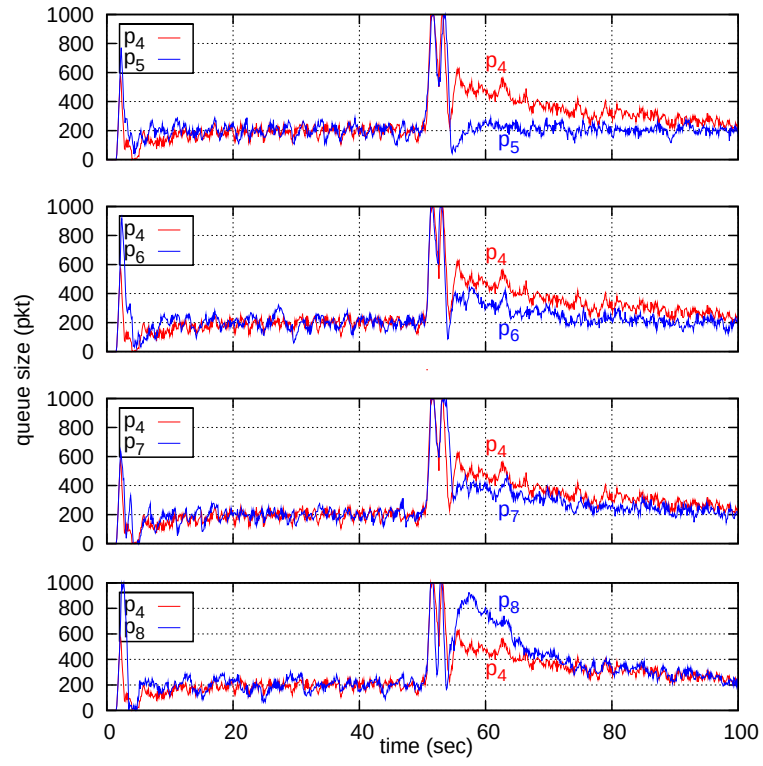


Figure 2.11: Comparing the performance of different points inside $\mathcal{S}_R$ but off Ξ_{PI} with p_4 .

state converges toward a target point, for instance, how fast or slow the convergence is. As we have seen in Section 2.4.2, p_1 results in a stable regime while p_2 results in an unstable regime. In comparison, in Section 2.4.3, p_1 – p_8 are all stable controllers but they exhibit widely different levels of performance. The advantage of $\mathcal{S}_R$ is that it gives the region of all stabilizing controllers, and thus it avoids choosing conservative parameters, e.g., $p_1 = \Delta_{\text{PI}}$, or even more radically conservative ones, e.g., $(10^{-20}, 10^{-20})$. This advantage of $\mathcal{S}_R$ is manifested most when designing controller gains to guarantee stability that is robust to network parameter uncertainties, e.g., to account for large d and small N , because $\mathcal{S}_R$ admits stable controller gains that are more aggressive, i.e., achieve improved convergence properties, than Δ_{PI} and Ξ_{PI} .

2.5 Conclusions

This chapter makes an indispensable contribution to the understanding of PI AQM by providing an analytical characterization of its complete stability region. The chapter has demonstrated the importance of obtaining the complete stability region by presenting examples of PI controllers that are stable and have significantly better performance than previously proposed ones. While we have focused the analysis on PI, other AQM schemes can benefit from the results and the analysis discussed in this chapter.

Chapter 3

Decentralized and Dynamic Bandwidth Allocation in Cyber-Physical Systems

In this chapter, we propose a bandwidth allocation scheme for Cyber-Physical Systems that have their control loops closed over a distributed network, such as the Internet. We first formulate the bandwidth allocation as a convex optimization problem. We then present an allocation scheme that solves this optimization problem in a fully distributed manner. In addition to being fully distributed, the proposed scheme is asynchronous, scalable, dynamic and flexible. We further discuss mechanisms to enhance the performance of the allocation scheme. We present analytical and simulation results.

3.1 Introduction

Networked embedded devices are becoming increasingly ubiquitous in our physical environments and will lead to the formation of *Cyber-Physical Systems (CPSs)* [41]. CPSs integrate sensing, processing, and actuation tasks that enable remote monitoring and control of the physical world; see Figure 3.1. Representative applications include industrial automation, distributed instrumentation, unmanned vehicles, home robotics, distributed virtual environments, power distribution, and building structure control [5, 7, 42, 43]. Since

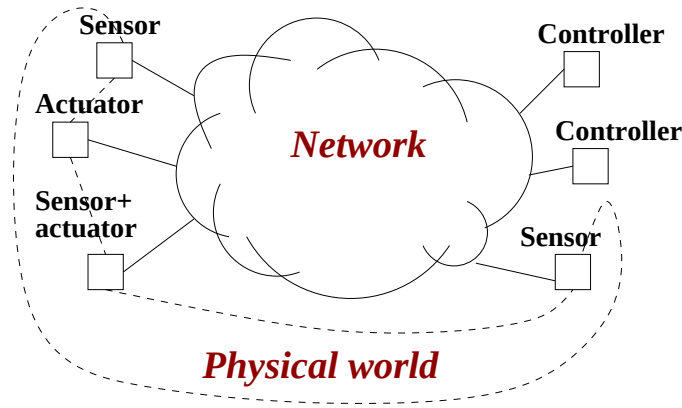


Figure 3.1: CPSs integrate sensing, processing, and actuation tasks that enable for remote monitoring and control of the physical world. Figure is adopted from [6, 42].

CPSs are concerned with real-time actuation and control, they differ significantly from sensor networks, whose primary scope is data acquisition in wireless, energy-constrained environments. CPSs' strength stems from the integration of the physical world with the cyber world; however, this integration poses fundamental challenges to the methods and protocols of communication networks. In general, existing communication methods may no longer be applicable when interconnected devices both sense and operate on a physical environment. For example, CPSs nodes should communicate with each other at a rate that is appropriate for the physical environment being controlled. Since CPSs involve control and actuation on the physical world, CPSs rate control is likely to differ significantly from congestion control for bulk data transfer, for multimedia traffic, or for sensor networks. Although existing congestion control techniques may prove to be an important source of inspiration, their applicability to CPSs is an open problem.

In this chapter, we propose an adaptive transmission-rate scheme to allocate the bandwidth among several CPSs. The bandwidth allocation problem is an inherent and a crucial issue because bandwidth is a finite resource to be shared among several CPSs. Without bandwidth management, congestion becomes a common consequence. Congestion is undesirable because it leads to long queueing delays, packet losses or both. In turn, long delays and packet losses deteriorate CPSs performance and jeopardize the stability

of the underlying physical system. Therefore, the objective is to allocate the bandwidth among CPSs to control congestion and to meet each system's requirement as best as possible. To achieve this objective, our proposed scheme mandates CPSs to adapt their sampling intervals based on the congestion level fed back from the network.

Summary of Contributions

The main contribution of this chapter is that we propose a bandwidth allocation scheme for CPSs that has the following features:

- It allocates the bandwidth in a way to ensure stability of all control systems, if feasible.
- It allocates the bandwidth in a way to attain the maximum *aggregate* performance of all control systems.
- It makes use of network bandwidth efficiently; controls congestion, thus minimizes delays and losses; and achieves fairness by fulfilling performance objectives of different control loops.
- It provides a *fully distributed, asynchronous, and scalable* solution. Each node executes an independent algorithm using local information with no central managing entity. The approach scales up as the number of controlled systems and/or the size of the network increase.
- It is *dynamic, adaptable, and flexible*. It dynamically reallocates the bandwidth as different control systems acquire and relinquish network resources.

Along with the above main contribution, this chapter has several specific contributions, which are summarized as follows. First, we formulate the bandwidth allocation problem as a mathematical optimization problem (Section 3.3). An optimization approach

is both natural and necessary because it determines the optimal transmission rates of the individual CPSs such that the overall CPSs performance is maximized, subject to network and stability constraints. Since our optimization formulation has similarities with mathematical formulations used for bulk-data congestion control, the solution of our optimization problem and thus our approach can borrow some of techniques used for bulk-data congestion control (e.g., [45]). However, the central concerns are different. CPSs usually have stringent real-time requirements necessary to ensure the stability and safety of CPSs whereas bulk-data flows can often tolerate transient poor levels of QoS. To make certain that the allocation scheme fulfills CPSs requirements, we develop a dynamical system model that describes the interaction between CPSs and the network. We then use this model to analyze and to enhance the dynamic properties of the allocation scheme (Section 3.4). For example, using this model, we show that a gradient-based approach (as in [45]) exhibits steady-state queue backlogs (Section 3.5). Although this side effect, i.e., the steady-state error, was previously observed in the literature when conducting simulations, it has never been precisely quantified. In contrast, we derive closed-form expressions for the steady-state error and show its dependency on the number of CPSs and on other network parameters (Section 3.5). To remedy the steady-state error in the queue length, we use a proportional-integral (PI) controller, which effectively stabilizes the queue length around a small reference value (Sections 3.4 and 3.5). Moreover, we characterize the *complete robust-stability region* of the PI controller (Sections 3.6 and 3.7). The complete robust-stability region gives the complete space of the controller parameters that ensure the stability of the allocation scheme even when round-trip delays and model parameters are unknown. Finally, we present packet-level simulations to evaluate the proposed allocation scheme and to validate the theoretical results (Section 3.8).

3.2 Related Work

3.2.1 Congestion Control in IP Networks

Addressing the bandwidth allocation problem in data communication networks using theoretical approaches is not a new subject. For example, [37, 45] formulated the bandwidth allocation problem as an optimization problem, and proposed distributed solutions for the problem. Although both papers started with exactly the same optimization formulation, they pursued different approaches to solve the problem, resulting in two different control algorithms: *primal* and *dual*. In the primal algorithm as in [37], sources adjust their transmission rate using *dynamic* control laws, while links compute congestion level using *static* laws. Conversely, in the dual algorithm as in [45], sources' algorithms are static and links' algorithms are dynamic [47]. Much research has then targeted stability analysis of each algorithm, especially the algorithm of [37] (see for example [35, 40, 50, 64, 69]).

Since the primal algorithm solves a relaxation rather than the exact version of the original optimization problem [37], we adopt the approach in [45], which indeed solves the original optimization problem exactly [46]. However, [45] did not address the issue of steady-state backlog in queue lengths. Also, it did not detail how to ascertain the algorithm's step size to maintain stability or to achieve faster convergence to steady state (it only suggests that the algorithm's step size must be chosen sufficiently small). To eliminate the problem of steady-state backlog, a new link control algorithm, called REM, was then proposed in [15], but again no details were given to determine the controller gains. The stability of the dual algorithm in the presence of delays was analyzed in both [53, 54]. However, [53] dealt with the first-order controller, i.e., the proportional controller, that exhibits steady-state error and [54] dealt only with a special family of utility functions.

In this chapter, we use a Proportional-Integral (PI) controller for optimization-based congestion control. PI control has the ability to eliminate the steady-state error [23] and to stabilize the queue length around a reference level. PI was first proposed as an AQM in [32]

but for TCP flows. In [32], only a subset of the stable gains was given. This subset forms a line segment in the space of control parameters. In contrast, we derive the complete *region* of stabilizing control parameters. Furthermore, unlike [61], our analysis provides robust stability conditions in that it ensures the allocation scheme’s stability for uncertain values of delays and utility functions, and does not require exact values of them.

3.2.2 Bandwidth Allocation in CPSs

The issue of bandwidth management in networked sensing, actuation and control systems (known as *Networked Control Systems* [72]) has gained considerable research attention; see for example [13, 63, 65] and the references contained therein. However, all such research efforts have focused on bandwidth scheduling in limited (local) area networks, e.g., in a car, in an airplane, or in a factory. Several factors hinder the extension of such bandwidth scheduling schemes to the domain of Wide Area Networks (WANs), such as the Internet [66]. These schemes usually require time synchronization among the different devices in the network (such as in TDMA-based schemes), or constrain physical distances over which the scheme can operate (such as in CSMA-based schemes, e.g., CAN protocol [65]). Moreover, the allocation schemes are either static or dynamic. Static schemes, where allocation is determined pre-run, lack flexibility and adaptability to dynamic changes. Dynamic schemes, on the other hand, required centralized implementations. Reference [13] discussed various examples and differences between static and dynamic approaches, and between TDMA- and CSMA-based approaches.

In this chapter, we propose a bandwidth allocation scheme for CPSs that is asynchronous, dynamic and flexible, and fully distributed. To the best of our knowledge, there has been no prior research into bandwidth allocation for CPSs. In our scheme, CPSs adapt their bandwidth usage by varying their sampling intervals based on a feedback from the network so as to avoid network congestion, and to preserve high control performance levels. It is worthwhile here to evaluate the ideas presented in [25] and in [63]. In [25], the

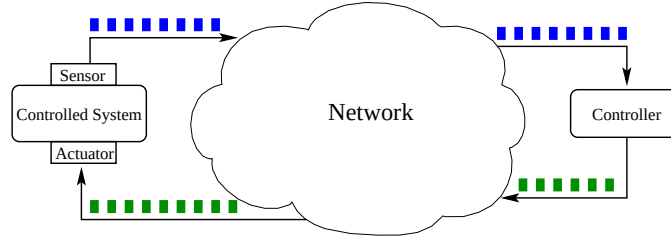


Figure 3.2: A Cyber-Physical System with one controlled system (a.k.a. *plant*) and one controller. Both the sensor and the actuator are collocated at the plant site.

authors proposed an algorithm to adapt the sampling interval of controlled systems implemented over a CAN bus based on two factors, network load and stability threshold. The algorithm per se is special to CAN in the way it determines the network load. Moreover, the heuristic of increasing and decreasing the sampling interval has no mathematical justification. The algorithm proposed in [63] uses the network’s available bandwidth and the error in each system’s state to adapt the sampling interval. However, the paper fails to discuss an important implementation issue: measuring the occupied bandwidth (to be used along with the network’s capacity to obtain the amount of unused bandwidth). In this chapter, we introduce an approach that relies on solid mathematical foundations, and we discuss its implementation details over IP networks. We also present results from a network simulator that was extended to simulate control systems [19].

3.3 Problem Formulation

3.3.1 On the Wire

Figure 3.2 shows a configuration of a single CPS in which the feedback loop is closed over a network. In general, the sensor samples the values of physical quantities, writes them in a packet, and sends the packet to the controller. The controller examines the received sample to generate a control signal that is then sent to the actuator.

The time interval between two sample packets is called the *sampling interval*, and it

is denoted by h . In other words, the sensor sends one packet containing sample data every h seconds. The reciprocal of the sampling interval,

$$r = \frac{1}{h}, \quad (3.1)$$

is the *rate* of transmission from the plant to the controller. The rate can be similarly defined in the reverse path from the controller to the plant. Although, in principle, the rates in the two paths could differ, in most CPS applications, the two rates are identical. The transmission rate is the amount of bandwidth resources that a particular plant-controller pair consumes. If the rate exceeds the end-to-end available bandwidth, the network is *congested*, and the communication is then characterized by packet losses, delays, and jitter. In principle, the rate should be small enough to avoid congestion. However, a CPS typically benefits from higher sampling rates. For example, the physical behavior usually tracks more closely the intended reference behavior if the sampling rate is higher. In some extreme circumstances, a very low sampling rate may cause the physical system to become *unstable*, in which case even small perturbations can cause massive breakdowns. Hence, the sampling rate r must strike a balance between network utilization and intended physical behaviors. The sampling rate is thus a critical tuning factor in CPSs.

The effect of the transmission rate r on the physical system dynamics is often captured by a *utility function*, $U(r)$. The utility value $U(r)$ expresses the degree to which a particular system can benefit from sampling rate r . In general, the utility function is a monotonically increasing function of the rate r , which reflects the fact that higher sampling rates lead to better control performance. In practice, the utility function is also often a strictly concave function of r , which reflects a law of diminishing returns as the rate increases. Finally, the utility function is defined only for $r \geq r_{\min}$, where $r_{\min}$ is the minimum rate below which the system becomes unstable or has unacceptable behavior. To carry out mathematical analysis easily, we pose an extra condition on $U(r)$ in which we

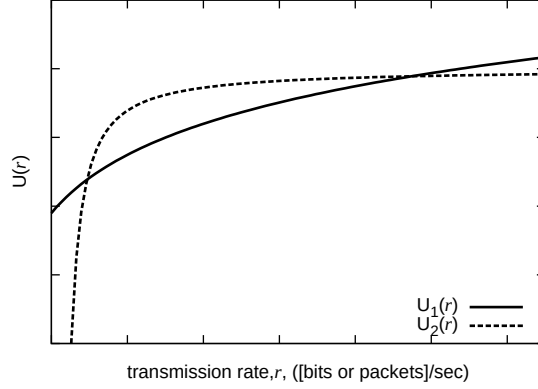


Figure 3.3: Examples of two generic utility functions.

require $U(r)$ to be doubly differentiable. Figure 3.3 shows two generic examples of utility functions associated with different applications. In the networked control systems literature, quadratic and exponential utility (performance) functions are commonly used for optimization purposes of CPU- and bandwidth- scheduling algorithms [18].

3.3.2 Optimization Formulation

One of the major pillars of the bandwidth allocation scheme is to achieve *fairness* among individual CPSs [8]. As in [59], we define a *fair* allocation to be the one that *maximizes* the sum of the utility functions of individual CPSs, i.e., the aggregate benefit of all CPSs. Then, consider a set S of CPSs using a set L of network links, where each link $l \in L$ has a capacity C_l . For each CPS $i \in S$, the objective is to determine its transmission rate r_i so as to maximize the sum of utilities $\sum_{i \in S} U_i(r_i)$, subject to (a) each CPS i 's stability constraint $r_i \geq r_{\min,i}$, and (b) each link l 's capacity constraint $\sum_{i \in S_l} r_i \leq C_l$, where S_l is the set of CPSs whose communication loops use link l . We state our objective formally as [8]:

$$\begin{aligned}
 & \max \quad \sum_{i \in S} U_i(r_i), \\
 & \text{s. t.} \quad \sum_{i \in S_l} r_i \leq C_l, \forall l \in L, \\
 & \text{and} \quad r_i \geq r_{\min,i}, \forall i \in S.
 \end{aligned} \tag{3.2}$$

In this formulation, we assume that the communication loop for each CPS can use link l only once. This assumption is always valid if all links are full duplex (in which case, forward and backward traffic do not interfere).

Due to the concavity characteristic of $U(r)$, Equation (3.2) is a convex optimization problem, which means it can be solved quickly and efficiently to yield a global, optimal solution [24]. However, the objective is to solve this program with a distributed approach with no centralized coordination.

3.3.3 Distributed Implementation

Due to its convenient structure, Equation (3.2) can be decomposed into separable sub-problems [49]. The solution can then be implemented in a distributed fashion, whereby individual controlled systems and links execute independent algorithms. This solution is achieved by considering a dual version of (3.2) that incorporates the Lagrange multipliers for link capacity constraints [45]. We summarize next the distributed algorithm and the protocol based on [45].

The algorithm works in an iterative manner until the optimal solution is achieved. Each link l computes a congestion level, p_l , based on local information, such as the aggregate incoming traffic, the queue length or both. The computation of p_l according to [45] is as follows:

$$p_l(t+1) = \max \left\{ 0, p_l(t) + \gamma \left(\sum_{i \in S_l} r_i(t) - C_l \right) \right\}, \quad (3.3)$$

where $p_l(t+1)$ and $p_l(t)$ are the congestion levels at the next and current steps, respectively; $\gamma > 0$ is the step size; $\sum_{i \in S_l} r_i(t)$ is the aggregate incoming rates at link l ; and C_l is link l 's capacity [45]. To carry congestion information from links back to plants, a special header field is introduced in the sensor and the controller packets. When the sensor generates a packet to carry the sampled data, the plant initializes the value of this field to zero. As the

packet traverses network links in the directed path from the sensor to the controller and back to the actuator, each link adds its current value of p_l to whatever value has accumulated in the field. Thus, when the control packet arrives at the plant, this special field would contain the total sum of p_l values of all individual links along the directed path from the sensor to the controller and back to the actuator. Upon receiving the controller packet, the actuator applies the control signal and the sensor regulates its sampling (sending) rate r based on the fed-back congestion information as follows:

$$r(p_t) = \min\{\max\{U'^{-1}(p_t), r_{\min}\}, r_{\max}\}, \quad (3.4)$$

where

- p_t is the value of p in the received controller packet, which is the sum of p_l values of all the links along the path from the plant to the controller and back to the plant;
- U'^{-1} is the inverse of the derivative of the utility function;
- $r_{\min}$ is the minimum transmission rate that satisfies the stability condition of the plant; and
- $r_{\max}$ is the maximum sampling rate and/or the maximum transmission rate a plant can use, which may stem from inherent hardware limitations of the sensor.

Based on the newly computed $r(p_t)$, the value of h is then calculated according to (3.1), which defines the wait time before generating the next sample.

Introducing the header field in the sensor and controller packets to carry the value of p_l , we assumed that routers are aware of and can manipulate this header. Also, we assumed that the overhead for this field is negligible (at most 64 bits for a double-precision floating-point number) compared to the size of each packet. Such assumptions are often implied for new congestion control protocols; see for example [36]. However, if practical implementations dictate otherwise, the p_l value in our protocol can be quantized and encoded by the

two ECN bits that already exist in transport protocols as in [68].

3.4 Link Queue Controllers

In this section, we model the interaction between controlled systems and network links in the proposed allocation scheme as a time-delay dynamical system. To facilitate the analysis, we focus on fluid-based modeling that approximates the actual packet dynamics. Utilizing the developed model, we then design controllers for link queues to enhance the performance of the scheme. This approach gives rise to two types of feedback loops. The first type of loop is for CPSs with distributed sensors, actuators, and controllers. The second is the loop of the developed model that captures the interaction between CPSs and links. Our focus in this chapter is on the second loop, where we design controllers to enhance the performance of the bandwidth allocation scheme to better meet the requirements of the CPSs loops.

3.4.1 Modeling CPS-Queue Interaction

Routers connect two or more network links. Thus, the link algorithm is actually executed at the router deployed at the link's input. Routers use buffers to hold incoming packets while servicing others. Congestion at a link causes the buffer to fill and possibly to overflow. Thus, congestion results in long delays, jitter, and packet losses. The aim is to stabilize the buffer's queue around a controllable small length greater than zero, which has a two-fold advantage. First, a stable, small queue length eliminates excessive delays, jitter, and losses. Second, a queue length greater than zero avoids network underutilization because the queue will always have packets to transmit. From this discussion, we regard the queue length as the primary response variable to be controlled around a setpoint.

We model the interaction between N CPSs and a single bottleneck link. We denote the forward delay from plant j to the queue by $\vec{d}_j$, and the backward delay from the queue

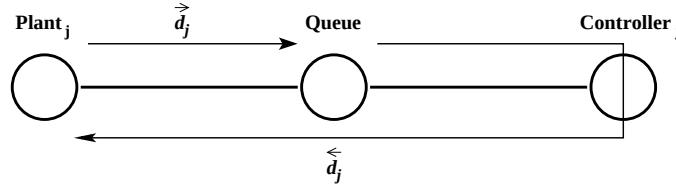


Figure 3.4: Forward and backward delays. Figure is adapted from [69].

to the controller and back to plant j by $\overleftarrow{d_j}$; see Figure 3.4. At time instant t , plant j transmits packets at rate $r_j(t)$. These packets start arriving at the queue and thus affecting the queue length, $q(t)$, after a delay $\overrightarrow{d_j}$, whereupon the queue computes a new value of $p(t)$. This new value of $p(t)$ reaches plant j after a delay $\overleftarrow{d_j}$. The evolution of the queue length, $q(t)$, in time can be modeled as:

$$\dot{q}(t) = \begin{cases} \sum_{i=1}^N r_i(t - \overrightarrow{d_i}) - C & \text{if } q(t) > 0 \\ \max \left\{ \sum_{i=1}^N r_i(t - \overrightarrow{d_i}) - C, 0 \right\} & \text{if } q(t) = 0, \end{cases} \quad (3.5)$$

where $\sum_{i=1}^N r_i(t - \overrightarrow{d_i})$ is the aggregate incoming traffic from all plants, and C is the bandwidth capacity of the outgoing link. Each plant j computes its sending rate as follows:

$$r_j(t) = \left[U_j'^{-1} \left(p(t - \overleftarrow{d_j}) \right) \right]_{r_{\min}}^{r_{\max}}, j = 1, \dots, N, \quad (3.6)$$

where $[x]_m^M = \min \{ \max \{ x, m \}, M \}$. (Note that (3.6) is a more complete version of (3.4) because it incorporates delays.)

3.4.2 Linearized Model

In this chapter, we conduct a linear systems analysis to study the stability and the performance of the system modeled by (3.5) and (3.6). To use linear analysis, we linearize (3.5) and (3.6) in the neighborhood of the operating point (q_0, r_{j0}, p_0) , where $q_0 > 0$ is the desired steady-state queue length, r_{j0} is the steady-state transmission rate of plant j , and p_0

is the steady-state p . According to (3.6), p_0 and r_{j0} are related to each other according to

$$p_0 = U'_j(r_{j0}) . \quad (3.7)$$

We assume (q_0, r_{j0}, p_0) is away from the boundary conditions in (3.5) and (3.6). Therefore, (3.5) and (3.6) reduce to

$$\begin{aligned} \dot{q}(t) &= \sum_{i=1}^N r_i(t - \overrightarrow{d_i}) - C , \\ r_j(t) &= U_j'^{-1} \left(p(t - \overleftarrow{d_j}) \right) , j = 1, \dots, N . \end{aligned}$$

Combining these equations yields

$$\dot{q}(t) = \sum_{i=1}^N U_i'^{-1} (p(t - d_i)) - C , \quad (3.8)$$

where $d_i = \overrightarrow{d_i} + \overleftarrow{d_i}$.

To linearize (3.8) about (q_0, r_{j0}, p_0) , we proceed as follows. First, define

$$f(p) = \dot{q}(t) = \sum_{i=1}^N U_i'^{-1} (p(t - d_i)) - C, \quad (3.9)$$

and $p = p(t - d_i)$. At the operating point (q_0, r_{j0}, p_0) , $\dot{q}(t) = 0$. Thus, $f(p_0) = 0$. Expanding the right-hand side of (3.9) using Taylor series [67] about (q_0, r_{j0}, p_0) , and ignoring

second- and higher-order terms yield

$$\begin{aligned}
f(p) &= f(p_0) + \left. \frac{df(p)}{dp} \right|_{p=p_0} (p - p_0) \\
&= \sum_{i=1}^N \left. \frac{d}{dp} U_i'^{-1}(p) \right|_{p=p_0} (p - p_0) \\
&= \sum_{i=1}^N \frac{1}{\left. \frac{d}{dr_i} U_i'(r_i) \right|_{r_i=r_{i0}}} (p - p_0) \\
&= \sum_{i=1}^N \frac{1}{U_i''(r_{i0})} (p - p_0) .
\end{aligned}$$

(The derivation proceeded from line 2 to line 3 by using the two facts: $dx/dy = 1/(dy/dx)$, i.e., the derivative identity of the inverse function, and $p_0 = U_j'(r_{j0})$, see (3.7).) Next, define $\delta q(t) = q(t) - q_0$ and $\delta p(t) = p(t) - p_0$. Therefore,

$$\dot{q}(t) = \frac{d(\delta q(t))}{dt} = \sum_{i=1}^N \frac{1}{U_i''(r_{i0})} \delta p(t - d_i) .$$

Thus, we obtain

$$\frac{d}{dt} \delta q(t) = \sum_{i=1}^N \beta_i \cdot \delta p(t - d_i) , \quad (3.10)$$

where $\delta q(t) = q(t) - q_0$ and $\delta p(t) = p(t) - p_0$ are the perturbations of q and p around q_0 and p_0 , respectively, and $\beta_i = 1/U_i''(r_{i0})$.

To simplify the analysis, we further assume that delays among CPSs are homogeneous, i.e., $d_i = d, i = 1, \dots, N$; however, our simulations in Section 3.8 validate our scheme using heterogeneous delays. Consequently, (3.10) becomes

$$\frac{d}{dt} \delta q(t) = -B \cdot \delta p(t - d) , \quad (3.11)$$

where $B = -\sum_{i=1}^N \beta_i$. Because the utility function is concave, $\beta_i < 0, i = 1, \dots, N$, and

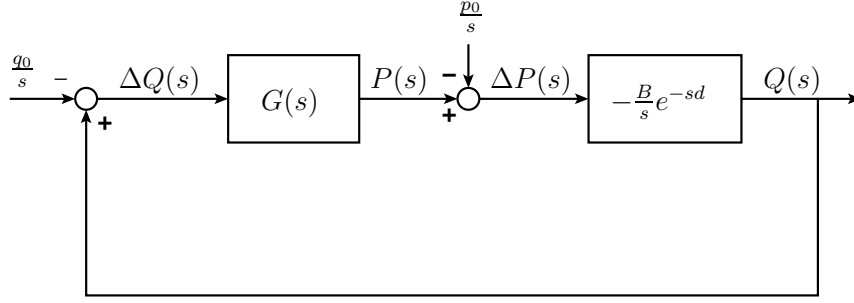


Figure 3.5: Linearized Model of CPS-Queue interaction with the controller $G(s)$.

hence $B > 0$.

We analyze the linearized model (3.11) in the frequency domain [23]. Since $q(0^-) = 0$, $\delta q(0^-) = q(0^-) - q_0 = -q_0$. Then, the Laplace transform of (3.11) is

$$s\Delta Q(s) - \delta q(0^-) = sQ(s) = -B \cdot \Delta P(s) \cdot e^{-sd}. \quad (3.12)$$

Figure 3.5 shows the block diagram of (3.12), where $G(s)$ is the Laplace transform of the function that relates $p(t)$ and $\delta q(t)$ and is called the *queue controller*.

3.4.3 P and PI Controllers

In this section, we design the controller $G(s)$ to stabilize and to improve the response of the closed-loop feedback system in Figure 3.5. Among different controllers, the simplest are the Proportional (P) and the Proportional-Integral (PI) controllers. Choosing such a simple controller algorithm allows the router to process large amounts of traffic efficiently.

The transfer function of a P controller is $G_P(s) = k_p$, and that of a PI is $G_I(s) = k_p + k_i/s$, where k_p and k_i are the proportional gain and the integral gain constants, respectively. Setting k_i to zero in the PI controller, $G_I(s)$, results in a pure P controller, $G_P(s)$.

Remark 3.4.1 Note that the direct solution of the optimization problem (3.2) using the gradient method yields a P controller, $G_P(s)$, as the queue controller, see (3.3) and [45, 53]

for details. As we show in Section 3.5, the P controller exhibits a steady-state error in the queue length whereas the PI controller does not. The steady-state error has negative effects that will be discussed in the next two sections.

Remark 3.4.2 Although solving the optimization problem (3.2) does not yield a PI controller, the PI controller still achieves an optimal solution for (3.2) because it stabilizes the queue [47].

3.5 Steady-state Error and Queueing Delays

The output, $Q(s)$, in Figure 3.5 is

$$Q(s) = \frac{Be^{-sd}}{s^2 + sBG(s)e^{-sd}} [p_0 + q_0G(s)] . \quad (3.13)$$

We find the steady state of $q(t)$, q_{ss} , by applying the final value theorem [52]. To apply the final value theorem, we assume that the controller's design parameters (k_p for the P controller, and k_p and k_i for the PI controller) are chosen such that the closed-loop system is stable (see Section 3.6).

When $G(s) = k_p$,

$$q_{ss}^P = \lim_{t \rightarrow \infty} q(t) = \lim_{s \rightarrow 0} sQ(s) = q_0 + p_0/k_p .$$

Similarly, when $G(s) = k_p + k_i/s$,

$$q_{ss}^{PI} = \lim_{t \rightarrow \infty} q(t) = \lim_{s \rightarrow 0} sQ(s) = q_0 .$$

Therefore, the steady-state error, e_{ss} , for P and PI is $e_{ss}^P = p_0/k_p$ and $e_{ss}^{PI} = 0$, respectively.

Moreover, the respective steady-state queueing delays, d_q , are

$$d_q^P = \frac{q_0}{C} + \frac{p_0}{k_p C}, \text{ and} \quad (3.14)$$

$$d_q^{PI} = \frac{q_0}{C}. \quad (3.15)$$

When using PI, queueing delays are predictable and independent of the characteristics of CPSs (i.e., utilities or number of CPSs) because both q_0 and C are constant. On the other hand, d_q^P depends on p_0 . In dynamic environments where different CPSs go on- and off-line at different times, p_0 changes over time. Unless k_p is adapted dynamically to track p_0 variations, d_q^P will vary as the environment changes. Therefore, in these dynamic environments, d_q^P is unpredictable. Although designing an allocation scheme in which k_p is dynamically adapted is a powerful mechanism, this is not this chapter's intent and we leave this idea for future research.

3.6 Stability Analysis

The characteristic function, $\psi(s)$, of the system of Figure 3.5 is

$$\psi(s) = s + BG(s)e^{-sd}. \quad (3.16)$$

Closed-loop stability requires the roots of (3.16) to lie in the open left-half of the complex plane [23]. That is, the closed-loop in Figure 3.5 is stable iff $\forall s_0$ such that $\psi(s_0) = 0$, $\Re\{s_0\} < 0$. When the later condition holds, we equivalently say that the characteristic function, $\psi(s)$, is stable (consistent with [26, p. 32]).

Because of the exponential term e^{-sd} , which originates from the delay in the feedback loop, (3.16) is called a *quasi-polynomial* and it has an infinite number of roots. In Sections 3.6.1 and 3.6.2, we analyze the stability of (3.16) for the two cases when $G_P(s) = k_p$ and $G_I(s) = k_p + k_i/s$.

3.6.1 The P Controller

Substituting $G(s) = k_p$ in (3.16) yields

$$\psi(s) = s + Bk_p e^{-sd} . \quad (3.17)$$

The round-trip delay d comprises two components: a fixed-delay (e.g., propagation-delay) component and a queueing-delay component that depends on k_p (see (3.14)). Therefore, we decompose d as $d = d_g + d_q = d_g + q_0/C + p_0/(k_p C)$, where d_g is the end-to-end fixed-delay component. Let $d_0 = d_g + q_0/C$. Then, $d = d_0 + p_0/(k_p C)$. The following Lemma determines the range of the stabilizing k_p .

Lemma 3.6.1 *The quasi-polynomial (3.17) is stable for all $(d_0 + p_0/(k_p C)) \geq 0$ and $B > 0$ iff*

$$0 < k_p < \frac{1}{Bd_0} \left(\frac{\pi}{2} - \frac{Bp_0}{C} \right) .$$

Proof Denote by $H(j\omega)$ the open-loop transfer function of Figure 3.5, corresponding to (3.17) and evaluated at $s = j\omega$. $H(j\omega)$ is given by

$$H(j\omega) = \frac{Bk_p}{j\omega} e^{-j\omega(d_0 + p_0/(k_p C))} .$$

Because $H(j\omega)$ does not have any open-loop poles in the right-half of the complex plane, the Nyquist stability criterion requires that the Nyquist plot of $H(j\omega)$ not encircle the point $-1 + j0$ [23]. Now, there are two cases: $k_p < 0$ and $k_p > 0$. When $k_p < 0$, the Nyquist plot of $H(j\omega)$ is shown in Figure 3.6 (left). The big arc at ∞ , which is due to the $1/\omega$ term in $H(j\omega)$, encircles the whole left-half plane. Therefore, $k_p < 0$ can never stabilize (3.17). When $k_p > 0$, the Nyquist plot is shown in Figure 3.6 (right). The first crossing of the Nyquist plot to the negative real-axis should occur to right of $-1 + j0$. This happens

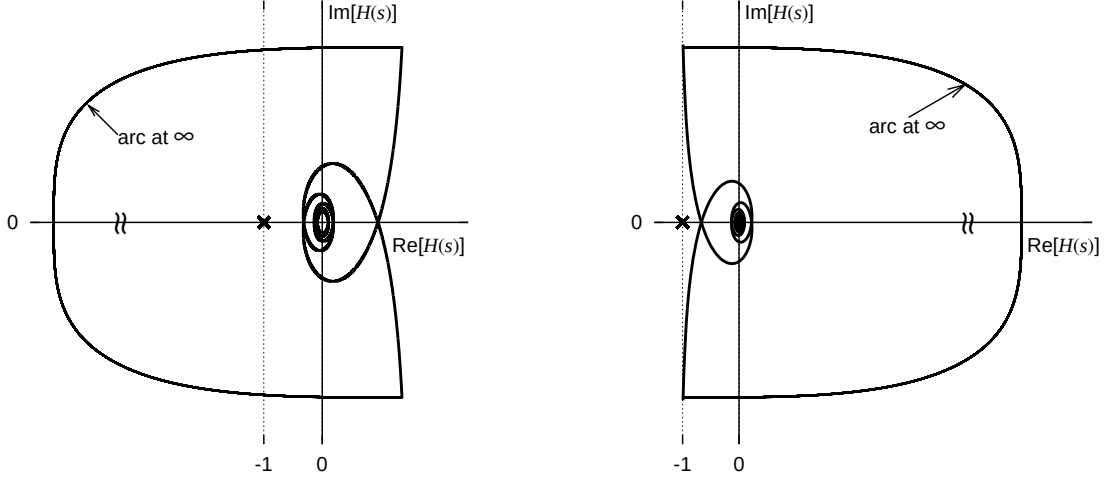


Figure 3.6: The Nyquist Plot of $H(j\omega) = \frac{Bk_p}{j\omega} e^{-j\omega(d_0+p_0/(k_p C))}$ when $k_p < 0$ (left) and when $k_p > 0$ (right) along with the critical point $-1 + j0$.

iff for $\omega_0 = \inf\{\omega_c > 0 : \arg\{H(j\omega_c)\} = -\pi\}$, $|H(j\omega_0)| < 1$. Solving for ω_0 yields

$$\omega_0 = \frac{\pi}{2} \frac{k_p C}{d_0 k_p C + p_0}.$$

After some manipulations, the condition $|H(j\omega_0)| < 1$ yields

$$k_p < \frac{1}{Bd_0} \left(\frac{\pi}{2} - \frac{Bp_0}{C} \right).$$

This completes the proof. ■

Remark 3.6.2 For a feasible range of a stabilizing k_p , we require

$$\frac{Bp_0}{C} < \frac{\pi}{2}. \tag{3.18}$$

If (3.18) does not hold, there will be no k_p that stabilizes the allocation scheme, no matter how small k_p is chosen. (Note that this situation was not predicted by the discrete-time analysis of [45].)

3.6.2 The PI Controller

With $G(s) = k_p + k_i/s$, the characteristic quasi-polynomial becomes

$$\psi(s) = s^2 + B(k_p s + k_i)e^{-sd} . \quad (3.19)$$

Unlike the case with the P controller, d in (3.19) is independent of k_p and k_i , see (3.15). Thus, we proceed by assuming d is constant. The following Lemma determines the region $\mathcal{S}_R(d, B)$ in the k_p - k_i plane that stabilizes (3.19) for given values of $d \geq 0$ and $B > 0$. First, we present the following known result that we use in proving the Lemma.

Theorem 3.6.3 (Theorem 12.13, [16]) *Let $H(z) \equiv z^2 e^z + pz + q$, where p and q are real. Denote by a_p the root of the equation (there is such a root if (a) below holds)*

$$\sin a = p/a ,$$

which lies on the open interval $(0, \pi/2)$. A necessary and sufficient condition that all the roots of $H(z) = 0$ lie to the left of the imaginary axis is that

$$(a) \quad 0 < p < \pi/2,$$

$$(b) \quad 0 < q < a_p^2 \cos a_p.$$

Lemma 3.6.4 *For given $d \geq 0$ and $B > 0$, (3.19) is stable if and only if $(k_p, k_i) \in \mathcal{S}_R(d, B)$, where*

$$\mathcal{S}_R(d, B) = \{(k_p, k_i) : 0 < k_p < k_{p,\max}, 0 < k_i < k_{i,\max}\} ,$$

$$k_{p,\max} = \begin{cases} \infty & \text{if } d = 0 \\ \frac{\pi}{2Bd} & \text{if } d \neq 0, \end{cases} \quad (3.20)$$

$$k_{i,\max} = \begin{cases} \infty & \text{if } d = 0 \\ \frac{\alpha_0^2 \cos \alpha_0}{Bd^2} & \text{if } d \neq 0, \end{cases} \quad (3.21)$$

and α_0 is the solution of

$$Bk_p d - \alpha \sin \alpha = 0 \quad (3.22)$$

in the interval $(0, \pi/2)$.

Proof We prove the Lemma for the two separate cases: when $d = 0$ and when $d \neq 0$.

When $d = 0$, (3.19) becomes

$$s^2 + Bk_p s + Bk_i = 0 .$$

The two roots of this quadratic equations are

$$s_{1,2} = \frac{-Bk_p \mp \sqrt{B^2 k_p^2 - 4Bk_i}}{2} .$$

Since $B > 0$, the closed-loop system is stable iff $k_p > 0$ and $k_i > 0$.

When $d \neq 0$, we need to analyze the roots of $\psi(s) = 0$ (see (3.19)). First, consider the quasi-polynomial

$$\tilde{\psi}(s) = d^2 e^{sd} \psi(s) = Bk_p d^2 s + Bk_i d^2 + d^2 s^2 e^{sd} .$$

Because $d^2 e^{sd} \neq 0$, the roots of $\tilde{\psi}(s)$ are exactly same as of those of $\psi(s)$. Now, let $z = sd$ and rewrite $\tilde{\psi}(s)$ as

$$\tilde{\psi}(z) = Bk_p dz + Bk_i d^2 + z^2 e^z .$$

Invoking Theorem 3.6.3 with $p = Bk_p d$ and $q = Bk_i d^2$ completes the proof. ■

3.7 Robust Stability

The stability region, $\mathcal{S}_R(d, B)$, obtained according to Lemma 3.6.4 assumes both d and B are fixed and known to the router that is implementing the control algorithm. However, this assumption is far from reality. Even if delays can be made known to routers en route (for example, using a special header in packets), delays can differ among flows and for each flow they usually vary over time. As for B , the situation is even harder because B depends on the utility functions and on the steady-state transmission rates of the individual CPSs, thus making such information available to the router is impractical in a fully distributed environment. In this section, we present a theorem that facilitates the design of the k_p and k_i gains to ensure the stability of the allocation scheme when only upper bounds for B and d are available. First, we analyze the stability of (3.19) when the values of d and B are uncertain but known to belong to an interval. Precisely, the objective is to compute the complete set of k_p and k_i gains that stabilize the closed-loop system $\forall d \in [0, d_{\max}]$ and $\forall B \in [B_1, B_2]$, where $d_{\max} > 0$ and $B_2 > B_1 > 0$.

The following two Lemmas 3.7.1 and 3.7.2 will be used in the proof of the main stability theorem.

Lemma 3.7.1 *For a given B , $k_{p,\max}$ defined in Lemma 3.6.4 is a strictly decreasing function of $d > 0$. Also, for given B and k_p , $k_{i,\max}$ defined in Lemma 3.6.4 is a strictly decreasing function of $d > 0$.*

Proof Since $Bd^2 > 0$,

$$\frac{\partial k_{p,\max}}{\partial d} = -\frac{\pi}{2Bd^2} < 0 .$$

Therefore, $k_{p,\max}$ is a strictly decreasing function of d .

As for $k_{i,\max}$,

$$\begin{aligned}\frac{\partial k_{i,\max}}{\partial d} &= \frac{1}{B^2 d^4} \left[B d^2 \frac{\partial}{\partial d} (\alpha_0^2 \cos \alpha_0) - 2 B d \alpha_0^2 \cos \alpha_0 \right] \\ &= \frac{1}{B d^3} \phi_1(d),\end{aligned}$$

where

$$\phi_1(d) = d \frac{\partial}{\partial d} (\alpha_0^2 \cos \alpha_0) - 2 \alpha_0^2 \cos \alpha_0 .$$

Since $B d^3 > 0$, it is enough to show that $\phi_1(d) < 0$. Now,

$$\frac{\partial}{\partial d} (\alpha_0^2 \cos \alpha_0) = \frac{\partial \alpha_0}{\partial d} [-\alpha_0^2 \sin \alpha_0 + 2 \alpha_0 \cos \alpha_0] .$$

From (3.22),

$$B k_p d - \alpha_0 \sin \alpha_0 = 0 . \tag{3.23}$$

Taking the derivative of both sides of (3.23) with respect to d yields

$$B k_p - \frac{\partial \alpha_0}{\partial d} [\alpha_0 \cos \alpha_0 + \sin \alpha_0] = 0 .$$

Or,

$$\frac{\partial \alpha_0}{\partial d} = \frac{B k_p}{\alpha_0 \cos \alpha_0 + \sin \alpha_0} .$$

Expanding $\phi_1(d)$ gives

$$\phi_1(d) = \frac{\alpha_0}{\sin \alpha_0 + \alpha_0 \cos \alpha_0} \phi_2(d) ,$$

where

$$\phi_2(d) = (2 \cos \alpha_0 - \alpha_0 \sin \alpha_0) B k_p d - 2 \alpha_0 \sin \alpha_0 \cos \alpha_0 - 2 \alpha_0^2 \cos^2 \alpha_0 .$$

Since $\alpha_0 \in (0, \pi/2)$ (see Lemma 3.6.4), $\alpha_0 / [\sin \alpha_0 + \alpha_0 \cos \alpha_0] > 0$. Therefore, it is enough to show that $\phi_2(d) < 0$. Substituting $\alpha_0 \sin \alpha_0$ for $B k_p d$ (see (3.23)) in $\phi_2(d)$ yields

$$\phi_2(d) = -\alpha_0^2 (1 + \cos^2 \alpha_0) .$$

Hence, $\phi_2(d) < 0$, which proves the Lemma. ■

Lemma 3.7.2 *For a given d , $k_{p,\max}$ defined in Lemma 3.6.4 is a strictly decreasing function of $B > 0$. Also, for given d and k_p , $k_{i,\max}$ defined in Lemma 3.6.4 is a strictly decreasing function of $B > 0$.*

The proof of this Lemma follows along the same lines of that of Lemma 3.7.1 and is presented below for completeness.

Proof Since $B^2 d > 0$,

$$\frac{\partial k_{p,\max}}{\partial B} = -\frac{\pi}{2B^2 d} < 0 .$$

Therefore, $k_{p,\max}$ is a strictly decreasing function of B .

As for $k_{i,\max}$,

$$\begin{aligned} \frac{\partial k_{i,\max}}{\partial B} &= \frac{1}{B^2 d^4} \left[B d^2 \frac{\partial}{\partial B} (\alpha_0^2 \cos \alpha_0) - d^2 \alpha_0^2 \cos \alpha_0 \right] \\ &= \frac{1}{B^2 d^2} \phi_1(B), \end{aligned}$$

where

$$\phi_1(B) = B \frac{\partial}{\partial B} (\alpha_0^2 \cos \alpha_0) - \alpha_0^2 \cos \alpha_0 .$$

Since $B^2 d^2 > 0$, it is enough to show that $\phi_1(B) < 0$. Now,

$$\frac{\partial}{\partial B} (\alpha_0^2 \cos \alpha_0) = \frac{\partial \alpha_0}{\partial B} [-\alpha_0^2 \sin \alpha_0 + 2\alpha_0 \cos \alpha_0] .$$

Taking the derivative of both sides of (3.23) with respect to B yields

$$k_p d - \frac{\partial \alpha_0}{\partial B} [\alpha_0 \cos \alpha_0 + \sin \alpha_0] = 0 .$$

Or,

$$\frac{\partial \alpha_0}{\partial B} = \frac{k_p d}{\alpha_0 \cos \alpha_0 + \sin \alpha_0} .$$

Expanding $\phi_1(B)$ gives

$$\phi_1(B) = \frac{\alpha_0}{\sin \alpha_0 + \alpha_0 \cos \alpha_0} \phi_2(B) ,$$

where

$$\phi_2(B) = (2 \cos \alpha_0 - \alpha_0 \sin \alpha_0) B k_p d - \alpha_0 \sin \alpha_0 \cos \alpha_0 - \alpha_0^2 \cos^2 \alpha_0 .$$

Since $\alpha_0 \in (0, \pi/2)$ (see Lemma 3.6.4), $\alpha_0 / [\sin \alpha_0 + \alpha_0 \cos \alpha_0] > 0$. Therefore, it is enough to show that $\phi_2(B) < 0$. Substituting $\alpha_0 \sin \alpha_0$ for $B k_p d$ (see (3.23)) in $\phi_2(B)$ yields

$$\phi_2(B) = \alpha_0 (\sin \alpha_0 \cos \alpha_0 - \alpha_0) .$$

Since $\alpha_0 \in (0, \pi/2)$,

$$\alpha_0 > \sin \alpha_0 > \sin \alpha_0 \cos \alpha_0 > 0 .$$

Therefore, $\phi_2(B) < 0$, which proves the Lemma. ■

Lemmas 3.7.1 and 3.7.2 lead to Corollaries 3.7.3 and 3.7.4.

Corollary 3.7.3 *Let $\mathcal{H}$ be a family of closed-loop systems with each having a characteristic equation given by (3.19) where $d \in [0, d_{\max}]$ and $d_{\max} > 0$. If there is a PI controller (k_p^*, k_i^*) that stabilizes (3.19) for $d = d_{\max}$, then (k_p^*, k_i^*) stabilizes the entire family $\mathcal{H}$.*

Proof Clearly, (k_p^*, k_i^*) stabilizes (3.19) for $d = 0$ (see Lemma 3.6.4). Let $0 < \hat{d} < d_{\max}$. According to Lemma 3.6.4,

$$\begin{aligned}\mathcal{S}_R(\hat{d}, B) &= \left\{ (k_p, k_i) : 0 < k_p < k_{p,\max}(\hat{d}) \text{ and} \right. \\ &\quad \left. 0 < k_i < k_{i,\max}(\hat{d}) \right\}, \\ \mathcal{S}_R(d_{\max}, B) &= \left\{ (k_p, k_i) : 0 < k_p < k_{p,\max}(d_{\max}) \text{ and} \right. \\ &\quad \left. 0 < k_i < k_{i,\max}(d_{\max}) \right\}.\end{aligned}$$

From Lemma 3.7.1, $k_{p,\max}(d_{\max}) < k_{p,\max}(\hat{d})$ and $\forall k_p \in (0, k_{p,\max}(d_{\max}))$, $k_{i,\max}(d_{\max}) < k_{i,\max}(\hat{d})$. Therefore, $\mathcal{S}_R(d_{\max}, B) \subset \mathcal{S}_R(\hat{d}, B)$. ■

Corollary 3.7.4 *Let $\mathcal{Q}$ be a family of closed-loop systems with each having a characteristic equation given by (3.19) where $B \in [B_1, B_2]$ and $B_2 > B_1 > 0$. If there is a PI controller (k_p^*, k_i^*) that stabilizes (3.19) for $B = B_2$, then (k_p^*, k_i^*) stabilizes the entire family $\mathcal{Q}$.*

Proof Let $0 < \hat{B} < B_2$. According to Lemma 3.6.4,

$$\begin{aligned}\mathcal{S}_R(d, \hat{B}) &= \left\{ (k_p, k_i) : 0 < k_p < k_{p,\max}(\hat{B}) \text{ and} \right. \\ &\quad \left. 0 < k_i < k_{i,\max}(\hat{B}) \right\}, \\ \mathcal{S}_R(d, B_2) &= \left\{ (k_p, k_i) : 0 < k_p < k_{p,\max}(B_2) \text{ and} \right. \\ &\quad \left. 0 < k_i < k_{i,\max}(B_2) \right\}.\end{aligned}$$

From Lemma 3.7.2, $k_{p,\max}(B_2) < k_{p,\max}(\hat{B})$ and $\forall k_p \in (0, k_{p,\max}(B_2))$, $k_{i,\max}(B_2) < k_{i,\max}(\hat{B})$. Therefore, $\mathcal{S}_R(d, B_2) \subset \mathcal{S}_R(d, \hat{B})$. ■

Now, we state our main result regarding robust stability.

Theorem 3.7.5 *Let $\mathcal{P}$ be a family of closed-loop systems with each having a characteristic equation given by (3.19) where $d \in [0, d_{\max}]$, $d_{\max} > 0$, $B \in [B_1, B_2]$ and $B_2 > B_1 > 0$. If there is a PI controller (k_p^*, k_i^*) that stabilizes (3.19) for $d = d_{\max}$ and $B = B_2$, then (k_p^*, k_i^*) stabilizes the entire family $\mathcal{P}$.*

Proof It follows directly from Corollaries 3.7.3 and 3.7.4. ■

Therefore, when designing k_p and k_i gains, we must consider the maximum expected d and the maximum expected B among all CPSs. Precise values of $d_{\max}$ and B_2 are not necessarily required. An overestimated value of $d_{\max}$ and an overestimated value of B_2 lead to increased stability robustness. However, such overestimated values diminish $\mathcal{S}_R$ and constrain the possible values of k_p and k_i , which in turn affect the protocol's speed of convergence adversely. Therefore, it is advisable to obtain reasonably tight bounds on both $d_{\max}$ and B_2 .

Remark 3.7.6 *In this section, we focused on robust-stability analysis for the PI controller only. However, the same analysis can be easily carried out to the P controller. That is, if k_p^* stabilizes (3.17) for $d_0 = d_{0,\max} > 0$, $B = B_2 > 0$, and $p_0 = p_{0,\max} > 0$, then k_p^* can stabilize the whole family of (3.17) with $d_0 \in [0, d_{0,\max}]$, $B \in (0, B_2]$, and $p_0 \in (0, p_{0,\max}]$.*

3.8 Simulations

In this section, we explain the experimental setup, and we present simulation results that evaluate the proposed allocation scheme and validate the theoretical results.

3.8.1 Simulation Software

We have extended ns-2 [1] by adding two new agents: *NSCSPlant* and *NSCSController*, which stand for networked-sensing-and-control-systems plant and controller, respectively.

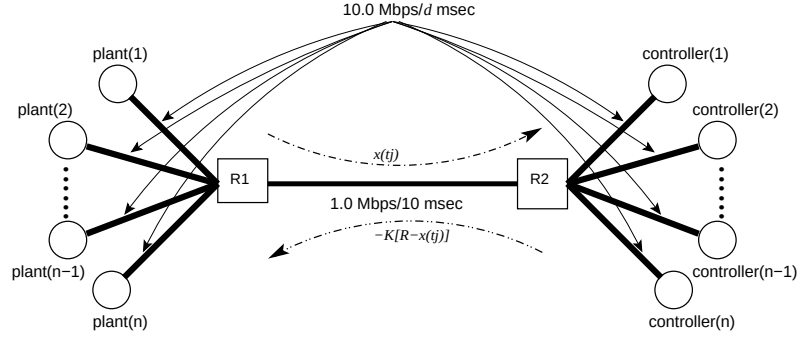


Figure 3.7: A single bottleneck topology for experimental simulation.

NSCSPlant is an abstract agent class, which can be used to instantiate several controlled systems, each of which simulates a physical system. NSCSController can be used to instantiate a controller to control a plant. With these two `ns-2` agents, we can then simulate the dynamics of physical systems combined with the dynamics of a communication network. NSCSPlant and NSCSController are based on an earlier `ns-2` implementation, `Agent/Plant`, to simulate networked control systems, see [19, 28]. (Recently, we have also developed a co-simulation platform for CPSs [12] that we will use in future experiments. This platform combines `ns-2` with Modelica, a modeling language for large-scale physical systems [62].)

3.8.2 Network Topology

Our experiments are based on the dumbbell topology shown in Figure 3.7. There, all CPSs share the single bottleneck link that connects the two routers, R1 and R2. Several plants are connected to R1; and their corresponding controllers to R2. Each link's bandwidth and propagation delay is shown in the figure. In Figure 3.7, d will be varied across plants.

3.8.3 Plants and Controllers

In this chapter, we confine our focus on linear scalar plants and proportional controllers.

Each plant's state, $x(t)$, evolves according to the following differential equation:

$$\dot{x}(t) = ax(t) + bu(t),$$

where a and b are constants, and $u(t)$ is the input from the controller. The sensor samples $x(t)$ at discrete time instances, generating $x(t_0), x(t_1), \dots, x(t_j)$. For each received plant's sample $x(t_j)$, the controller calculates $u(t_j)$ as follows:

$$u(t_j) = -K(R(t) - x(t_j)),$$

where K is the constant controller gain, and $R(t)$ is the reference signal the plant is required to follow.

Reference [18] proposed a performance measure for linear scalar networked control systems that is a function of the sampling interval, h . Substituting $1/r$ in place of h , we obtain the following utility function for plant i :

$$U_i(r_i) = \frac{a_i - b_i K_i}{a_i} e^{a_i/r_i}. \quad (3.24)$$

Such a utility function satisfies all required conditions mentioned at the end of Subsection 3.3.1. Moreover, $r_{\min}$ is derived in [71] for the same family of linear scalar networked control systems, and is given by

$$r_{\min,i} = \frac{a_i}{\ln \left(\frac{b_i K_i + a_i}{b_i K_i - a_i} \right)}.$$

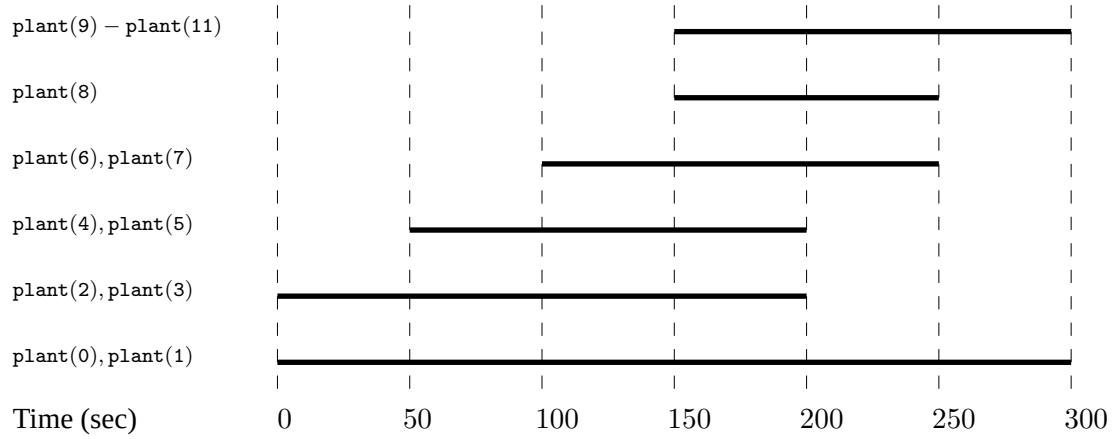


Figure 3.8: Six sets of CPSs. Each set uses the network for the duration defined by the extent of the bold horizontal line. For example, the set comprising plant(0) and plant(1) acquires the network at $t_s = 0$ sec and releases it at $t_e = 300$ sec.

3.8.4 Experiments

We have a dynamic environment where CPSs acquire and release the network at different times. There are six sets of CPSs based on when each set acquires or releases the network as in Figure 3.8.

We assume that all plants have identical physical dynamics, $a_i = 0.12$ and $b_i = 1.0$, and all have the same corresponding controllers, $K_i = 4.7$. Propagation delays for links connecting individual plants and controllers (i.e., d in Figure 3.7) are as follows. For plant(0), $d = 0$ msec; for plant(1), $d = 10$ msec; and for plant(2) through plant(11), the fixed delay, d , is drawn from a uniform distribution on the interval $[0, 10]$ msec. (See [3, 8] for different sets of simulations and results.)

3.8.5 Computing Controller Parameters

To calculate B and p_0 , we assume that the plants' and controllers' packets have size of 100 bytes. Therefore, $C = 1250$ pkts/sec. Based on the number of plants using the network during each time interval, B and p_0 values are summarized in Table 3.1.

To estimate $d_{\max}$, we assume that $q_0 = 50$ pkts. Based on Figure 3.7 and the afore-

Table 3.1: The values of B , p_0 , and $k_{p,\max}^P$ during each interval of time where the number of CPSs is constant.

Time interval (sec)	Number of plants	B	p_0	$k_{p,\max}^P$
$[0, 50]$	4	1.3319×10^7	4.6917×10^{-5}	5.3603×10^{-7}
$[50, 100]$	6	5.9178×10^6	1.0558×10^{-4}	1.2065×10^{-6}
$[100, 150]$	8	3.3278×10^6	1.8774×10^{-4}	2.1456×10^{-6}
$[150, 200]$	12	1.4782×10^6	4.2258×10^{-4}	4.8307×10^{-6}
$[200, 250]$	8	3.3278×10^6	1.8774×10^{-4}	2.1456×10^{-6}
$[250, 300]$	5	8.5228×10^6	7.3315×10^{-5}	8.3769×10^{-7}

mentioned assumptions, we choose an overestimate $d_{\max}$, such as $d_{\max} = 0.15$ sec. (This value of $d_{\max}$ is almost 150% the expected amount of delays at steady state, i.e., d_0 .)

For the P controller, the upper bound, $k_{p,\max}^P = [\pi/2 - Bp_0/C]/(Bd_0)$ (see Lemma 3.6.1), of the stabilizing k_p range is given in Table 3.1 during each time interval. The intersection of the six stabilizing k_p intervals where each defined by $(0, k_{p,\max}^P)$ is $(0, 5.3603 \times 10^{-7})$. Thus, for the P controller, we choose $k_p^P = 4.0 \times 10^{-7}$, i.e., about 75% of 5.3603×10^{-7} . As for PI, the k_p - k_i stabilizing region, $\mathcal{S}_R$, corresponding to $B = 1.3319 \times 10^7$ and $d_{\max} = 0.15$ sec is shown in Figure 3.9. We choose the PI controller gains as $(k_p^{PI}, k_i^{PI}) = (4.75 \times 10^{-7}, 9.0 \times 10^{-7})$, i.e., in the middle of $\mathcal{S}_R$. According to the analysis in Section 3.7, $(k_p^{PI}, k_i^{PI}) = (4.75 \times 10^{-7}, 9.0 \times 10^{-7})$ stabilizes the allocation scheme during all time intervals.

3.8.6 Results

Under both the P and the PI queue controllers, Figure 3.10 shows how CPSs that are using the network adapt their transmission rates by reducing their sampling rates when new CPSs start operating or by increasing their sampling rates when some CPSs stop operating. All CPSs share the bottleneck bandwidth equally since all have the same physical dynamics and thus the same utility functions. Moreover, the allocation scheme retains 100% network

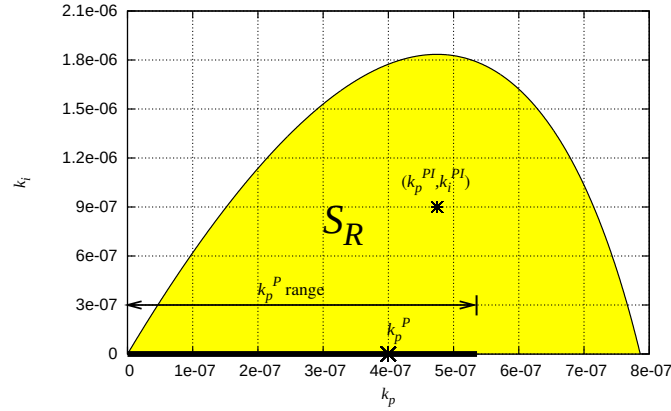


Figure 3.9: The stabilizing range of the P controller and the stabilizing region of the PI controller, $\mathcal{S}_R$, for $B = 1.3319 \times 10^7$, $p_0 = 4.6917 \times 10^{-5}$ and $d_{\max} = 0.15$ sec; and the two points $k_p^P = 4.0 \times 10^{-7}$ and $(k_p^{PI}, k_i^{PI}) = (4.75 \times 10^{-7}, 9.0 \times 10^{-7})$ used in simulations.

utilization when CPSs acquire or release the network bandwidth (this can be inferred by adding transmission rates of all plants during each time interval).

Although Figure 3.10 does not reveal a significant difference between using the P and PI as queue controllers, the difference is pronounced when comparing $q(t)$ behaviors under the two controllers; see Figure 3.11. With the P controller, the queue exhibits a steady-state error. As more CPSs acquire the network, the deviation of $q(t)$ from $q_0 = 50$ pkts widens. On the contrary, $q(t)$ under the PI control settles to $q_0 = 50$ pkts after short periods of transient behavior regardless of the number of CPSs using the network. Thus, the results in Figure 3.11 confirm the analysis in Section 3.5.

Long queue backlogs increase round-trip delays (see Figure 3.12), which in turn affect the control performance adversely. Figure 3.14 and Figure 3.15 depict the response of plant(0) and plant(1) when using PI and P as queue controllers, respectively. The CPS control objective is to follow the square-wave input in Figure 3.13.

With PI, the CPSs stay stable and track the input signal accurately (Figure 3.14). On the other hand, with the P controller, long queueing delays degrade the performance of the CPSs and cause instability (Figure 3.15). These results, which confirm the theoretical analysis, show the superiority of PI as a queue controller and the inadequacy of the P

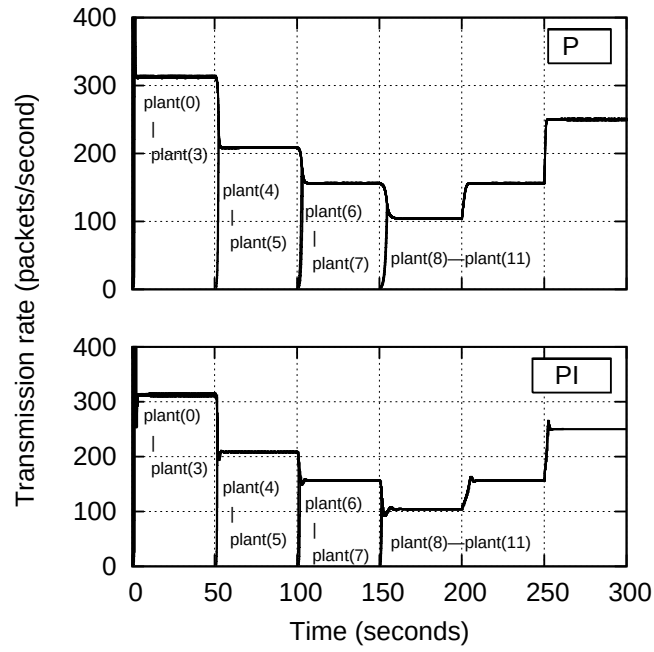


Figure 3.10: Transmission rates when using the P and the PI controllers.

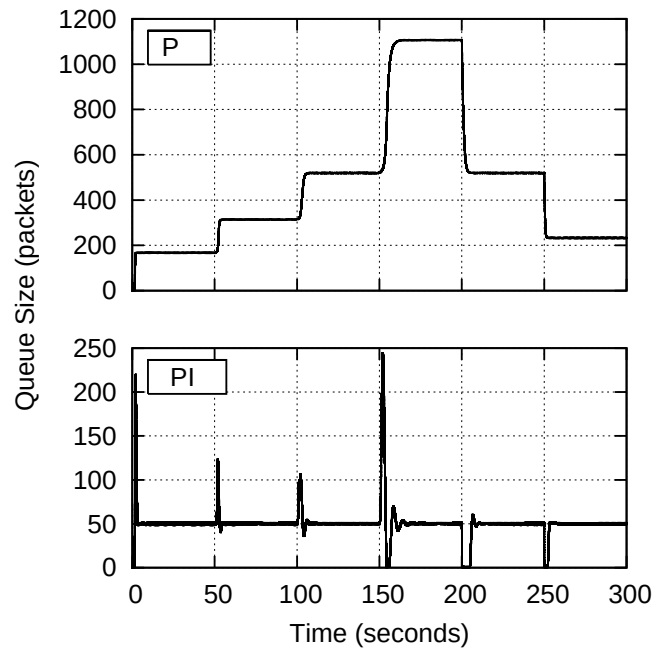


Figure 3.11: Queue length when using the P and the PI controllers.

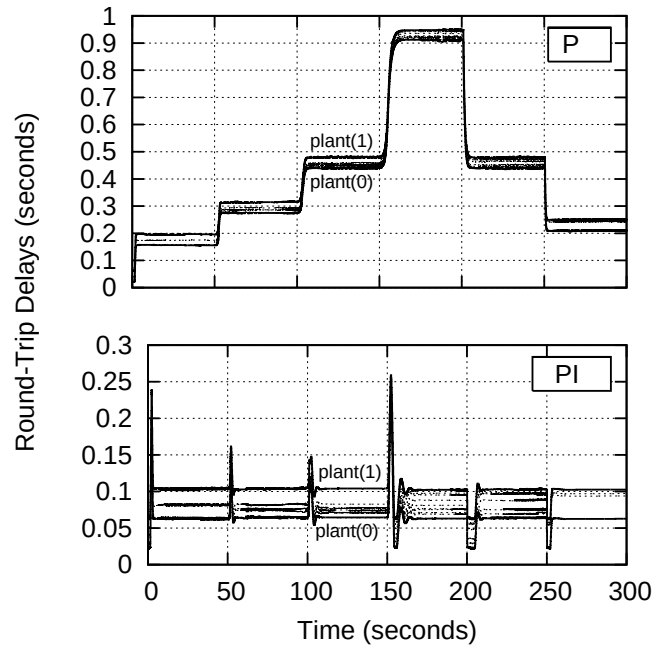


Figure 3.12: Round-trip delays when using the P and the PI controllers. Only delays for plant(0) and plant(1) are highlighted with solid lines. Delays for plants plant(2) through plant(11) are shown in dotted lines, which fall between those of plant(0) and plant(1).

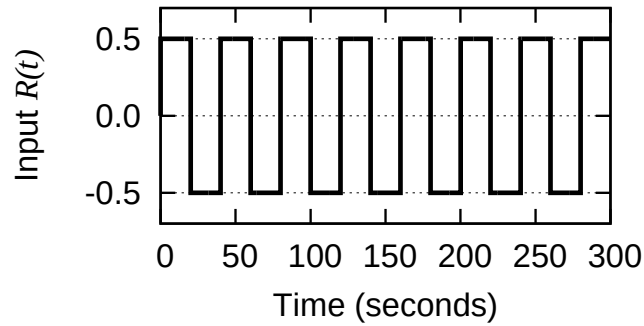


Figure 3.13: The input signal, $R(t)$, plants are instructed to follow.

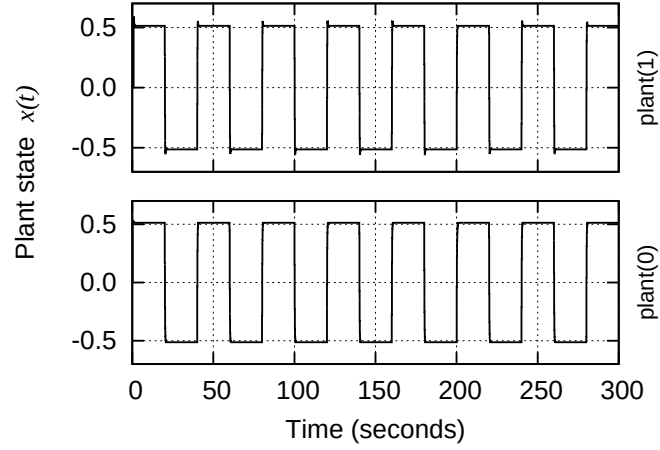


Figure 3.14: Plant state, $x(t)$, for plant(0) and plant(1) while tracking the input signal, $R(t)$, of Figure 3.13 when the PI controller is used as a queue controller.

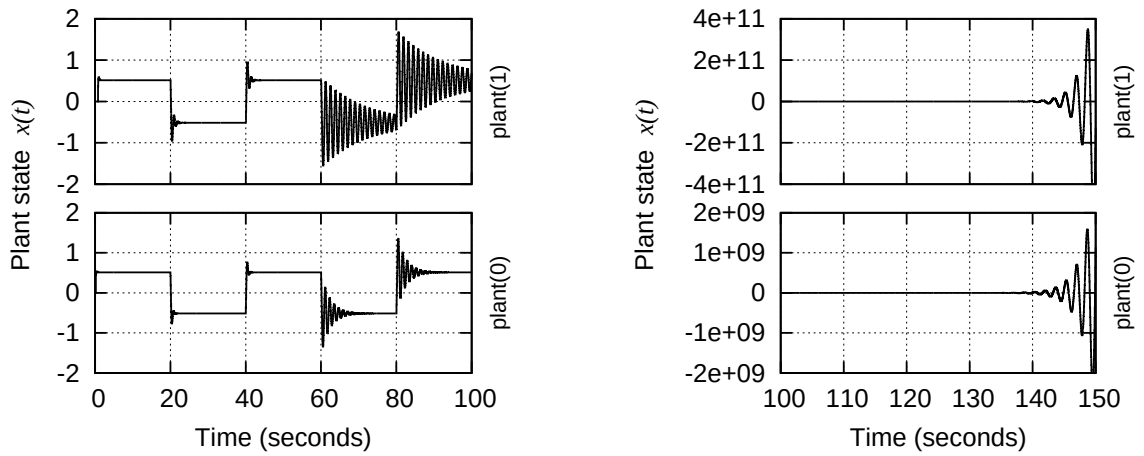


Figure 3.15: Plant state, $x(t)$, for plant(0) and plant(1) while tracking the input signal, $R(t)$, of Figure 3.13 when the P controller is used as a queue controller. Only the time interval $[0, 150]$ sec is shown, which is divided into two separate figures (left for $[0, 100]$ sec and right for $[100, 150]$ sec) to highlight differences in CPS control performance.

controller for use in the proposed allocation scheme.

3.9 Conclusions

In this chapter, we have presented a scheme for bandwidth allocation in Cyber-Physical Systems (CPSs). First, we formulated the problem of bandwidth allocation as a convex optimization problem whereby the objective is to maximize the aggregate performance of all CPSs subject to stability and network constraints. Second, we presented a distributed implementation of the optimization problem. Third, we developed a dynamical model to analyze the performance of the decentralized scheme under two queue control algorithms: P and PI controllers. Fourth, we showed how to design robust and resilient controllers to guarantee the stability of the allocation scheme when communication delays and model parameters are uncertain. Finally, we experimentally validated the theoretical results using ns-2 simulations.

Chapter 4

Conclusions and Future Work

This chapter concludes the thesis by summarizing our contributions and presenting directions for future work.

4.1 Conclusions

This thesis presented contributions in two specific areas of Internet congestion control: PI AQM and bandwidth allocation in cyber-physical systems. These contributions are summarized as follows. In Chapter 2, we derived an analytical characterization of the complete stability region of the PI controller for TCP AQM and we validated it with ns-2 simulations. The analytical derivation was complicated mostly due to time-delays in the TCP-AQM feedback loop and to the relatively high-order TCP-AQM plant model. Utilizing the complete stability region, we showed that the PI parameters used in the literature could be excessively conservative, a fact that explained the previous observation about PI sluggish responsiveness. We also showed that provably stable controller parameters could exhibit widely different levels of performance. Furthermore, we presented examples of PI controllers that are stable and have significantly better performance than previously proposed ones. Therefore, the chapter demonstrated the importance of obtaining the complete stability region for the PI AQM. While we have focused the analysis on PI, other AQM

schemes can benefit from the results and the analysis discussed in this chapter.

In Chapter 3, we proposed a bandwidth allocation scheme for Cyber-Physical Systems (CPSs) that have their control loops closed over a spatially distributed network. We formulated the bandwidth allocation as a convex optimization problem whereby the objective is to maximize the aggregate performance of all CPSs subject to stability and network constraints. We then presented an allocation scheme that solved the optimization problem in a fully distributed manner. In addition to being fully distributed, the proposed scheme had other features, such as being asynchronous, scalable, dynamic and flexible. We further showed how to ascertain the scheme's parameters to achieve robust and resilient operation in face of uncertainties in communication delays and in characteristics of CPSs.

4.2 Future Work

This thesis paves the way to several avenues of future work. The first step is to extend the results, e.g., the complete stability region, to larger and more general topologies than the dumbbell set-up. Another area is to design AQM and queue controllers that use system characterization and gain scheduling techniques to adapt controller parameters on the fly. Gain-scheduling and adaptive queue controllers require the estimation of different network parameters, e.g., delays and number of flows; and the construction of the complete stability region, which represents the whole feasible space from which particular points are to be chosen to attain particular performance metrics. Adaptive controllers [14] can outperform static controllers especially in highly dynamic environments.

Control theoretical procedures similar to ones used in this thesis can be applied to other layers in the Internet stack. In particular, an adaptive and reflective middleware [11] can be implemented in the application layer to ameliorate for the lack of QoS. Such a middleware will provide a common platform for a range of applications by acting as a broker between the applications and the network as follows. It collects online measurements for

network-oriented metrics, such as delay, jitter, packet loss rate, and bandwidth availability, and maps their statistics to application-oriented metrics. It then alters some system configurations and parameters in response to and to adapt to network vagaries [11, 42].

References

- [1] The network simulator—ns - 2. [Online]. Available: <http://www.isi.edu/nsnam/ns/>. 2, 11, 66
- [2] Ahmad T. Al-Hammouri. A distributed framework to facilitate human-robot remote interaction. Master's thesis, Case Western Reserve University, Cleveland, Ohio, January 2004. Advisor: Vincenzo Liberatore. 4
- [3] Ahmad T. Al-Hammouri and Vincenzo Liberatore. Optimization congestion control for Networked Control Systems. In *Proc. of IEEE INFOCOM Student Workshop*, Miami, FL, March 2005. Abstract. 69
- [4] Ahmad T. Al-Hammouri and Vincenzo Liberatore. Analytic derivation of the PI-AQM stability region. In *Proc. of Northeast Ohio Networking Workshop (NEONet 07)*, Baldwin Wallace College, Cleveland, OH, April 2007. Short Abstract. 2, 7
- [5] Ahmad T. Al-Hammouri, Adam Covitch, David Rosas, Murat Kose, Wyatt S. Newman, and Vincenzo Liberatore. Compliant control and software agents for Internet robotics. In *Proc. of IEEE International Workshop on Object-oriented Real-time Dependable Systems (WORDS 03)*, January 2003. 39
- [6] Ahmad T. Al-Hammouri, Vincenzo Liberatore, and Michael S. Branicky. A testbed for Networked Control Systems. In *Proc. of Research ShowCase*, Case Western Reserve University, Cleveland, OH, April 2004. Poster. ix, 40

- [7] Ahmad T. Al-Hammouri, Vincenzo Liberatore, Huthaifa A. Al-Omari, and Stephen M. Phillips. Transversal issues in real-time sense-and-respond systems. In *Proc. of ACM Workshop on End-to-End, Sense-and-Respond Systems, Applications, and Services (EESR 05)*, Seattle, WA, June 2005. 4, 39
- [8] Ahmad T. Al-Hammouri, Michael S. Branicky, Vincenzo Liberatore, and Stephen M. Phillips. Decentralized and dynamic bandwidth allocation in networked control systems. In *Proc. of International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS 06)*, Island of Rhodes, Greece, April 2006. 4, 47, 69
- [9] Ahmad T. Al-Hammouri, Vincenzo Liberatore, Michael S. Branicky, and Stephen M. Phillips. Parameterizing PI congestion controllers. In *Proc. of International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks (FeBID 06)*, Vancouver, Canada, April 2006. 8, 13
- [10] Ahmad T. Al-Hammouri, Vincenzo Liberatore, Michael S. Branicky, and Stephen M. Phillips. Complete stability region characterization for PI-AQM. *SIGBED Review*, 3 (2), April 2006. 8, 13, 30, 34
- [11] Ahmad T. Al-Hammouri, Wenhui Zhang, Robert F. Buchheit, Vincenzo Liberatore, Panos K. Chrysanthis, and Kirk Pruhs. Network awareness and application adaptability. *Information Systems and E-Business Management*, 4(4):399–419, October 2006. 77, 78
- [12] Ahmad T. Al-Hammouri, Vincenzo Liberatore, Huthaifa Al-Omari, Zakaria Al-Qudah, Michael S. Branicky, and Deepak Agrawal. A co-simulation platform for actuator networks. In *Proc. of ACM Conference on Embedded Networked Sensor Systems (SenSys 07)*, Sydney, Australia, November 2007. Demonstration. To appear. 67

- [13] L. Almeida, P. Pedreiras, and J.A. Fonseca. The FTT-CAN protocol: Why and how. *IEEE Transactions on Industrial Electronics*, 49(6):1189–1201, 2002. 44
- [14] K.J. Astrom and B. Wittenmark. *Adaptive Control*. Prentice Hall, 1994. 77
- [15] S. Athuraliya, S. Low, V. Li, and Q. Yin. REM: Active queue management. *IEEE Network*, 15(3):48–53, 2001. 3, 8, 12, 43
- [16] R. Bellman and K.L. Cooke. *Differential-Difference Equations*. Academic Press Inc., 1963. 59
- [17] Christof Brandauer, Gianluca Iannaccone, Christophe Diot, Thomas Ziegler, Serge Fdida, and Martin May. Comparison of tail drop and active queue management performance for bulk-data and Web-like Internet traffic. In *Proc. of IEEE Symposium on Computers and Communications*, Hammamet, Tunisia, 2001. 2
- [18] M.S. Branicky, S.M. Phillips, and W. Zhang. Scheduling and feedback co-design for networked control systems. In *Proc. of IEEE Conference on Decision and Control*, 2002. 47, 68
- [19] M.S. Branicky, V. Liberatore, and S.M. Phillips. Networked control system co-simulation for co-design. In *Proc. of American Control Conference*, 2003. 45, 67
- [20] W.-C. Feng, D.D. Kandlur, D. Saha, and K.G. Shin. A self-configuring RED gateway. In *Proc. of IEEE INFOCOM*, 1999. 11
- [21] R. Fengyuan, L. Chuang, Y. Xunhe, S. Xiuming, and W. Fubao. A robust active queue management algorithm based on sliding mode variable structure control. In *Proc. of IEEE INFOCOM*, 2002. 13
- [22] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, 1993. 2, 7, 11

- [23] G.F. Franklin, J.D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice Hall, third edition, 1999. 16, 43, 54, 56, 57
- [24] Frontline Systems. Convex optimization. [Online]. Available: <http://www.solver.com/probconvex.htm>. 48
- [25] I.A. Gravagne, J.M. Davis, J.J. DaCunha, and R.J. Marks. Bandwidth reduction for controller area networks using adaptive sampling. In *Proc. of International Conference on Robotics and Automation*, 2004. 44
- [26] Keqin Gu, Vladimir L. Kharitonov, and Jie Chen. *Stability of Time-Delay Systems*. Birkhäuser, 2003. 56
- [27] H. Han, C.V. Hollot, Y. Chait, and V. Misra. TCP networks stabilized by buffer-based AQMs. In *Proc. of IEEE INFOCOM*, 2004. 33
- [28] Justin R. Hartman, Michael S. Branicky, and Vincenzo Liberatore. Time-dependent dynamics in networked sensing and control. In *Proc. of American Control Conference*, Portland, OR, 2005. 67
- [29] Z. Heying, L. Baohong, and D. Wenhua. Design of a robust active queue management algorithm based on feedback compensation. In *Proc. of ACM SIGCOMM*, 2002. 3, 4, 7, 8, 12, 13
- [30] C. Hollot, V. Misra, D. Towsley, and W. Gong. On designing improved controllers for AQM routers supporting TCP flows. In *Proc. of IEEE INFOCOM*, 2001. viii, 3, 7, 8, 11, 13, 31, 34, 35
- [31] C.V. Hollot, V. Misra, D. Towsley, and W. Gong. A control theoretic analysis of RED. In *Proc. of IEEE INFOCOM*, 2001. 3
- [32] C.V. Hollot, V. Misra, D. Towsley, and W. Gong. Analysis and design of controllers

- for AQM routers supporting TCP flows. *IEEE Transactions on Automatic Control*, 47(6):945–959, 2002. viii, 3, 8, 9, 10, 13, 31, 34, 35, 43, 44
- [33] C.V. Hollot, Y. Liu, V. Misra, and D. Towsley. Unresponsive flows and AQM performance. In *Proc. of IEEE INFOCOM*, 2003. 10
- [34] Van Jacobson. Congestion avoidance and control. In *Proc. of ACM SIGCOMM*, Stanford, CA, August 1988. 1
- [35] R. Johari and D. Tan. End-to-end congestion control for the Internet: Delays and stability. *IEEE/ACM Transactions on Networking*, 9(6):818–832, 2001. 43
- [36] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *Proc. of ACM SIGCOMM*, 2002. 49
- [37] F. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–252, 1998. 43
- [38] B.Z. Khan and B. Lehman. Setpoint PI controllers for systems with large normalized dead time. *IEEE Transactions on Control Systems Technology*, 4(4):459–466, July 1996. 13
- [39] S. Kunniyur and R. Srikant. Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management. In *Proc. of ACM SIGCOMM*, August 2001. 13
- [40] R. La, P. Ranjan, and E. Abed. Global stability conditions for rate control of discretized model with communication delays. In *Proc. of Global Telecommunications Conference*, 2004. 43
- [41] Edward A. Lee. Cyber-Physical Systems—are computing foundations adequate? In

- Proc. of NSF Workshop on Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, Austin, Texas, October 2006. Position Paper. 39
- [42] V. Liberatore. Integrated play-back, sensing, and networked control. In *Proc. of IEEE INFOCOM*, 2006. ix, 39, 40, 78
- [43] Vincenzo Liberatore, Wyatt Newman, Ahmad T. Al-Hammouri, Adam Covitch, David Rosas, and Murat Kose. Remote interaction with machines. In *Proc. of Wireless Communications and Networking Workshop*, Case Western Reserve University, Cleveland, OH, November 2002. Poster and live demonstration. 39
- [44] S. Liu, T. Basar, and R. Srikant. Pitfalls in the fluid modeling of RTT variations in window-based congestion control. In *Proc. of IEEE INFOCOM*, 2005. 10, 33
- [45] S.H. Low and D.E. Lapsley. Optimization flow control—I: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, 1999. 42, 43, 48, 54, 58
- [46] S.H. Low and R. Srikant. A mathematical framework for designing a low-loss, low-delay Internet. *Network and Spatial Economics*, 4(1):75–102, March 2004. 43
- [47] S.H. Low, F. Paganini, and J.C. Doyle. Internet congestion control. *IEEE Control Systems Magazine*, 22(1):28–43, February 2002. 1, 10, 43, 55
- [48] S.H. Low, F. Paganini, J. Wang, S. Adlakha, and J.C. Doyle. Dynamics of TCP/RED and a scalable control. In *Proc. of IEEE INFOCOM*, 2002. 33
- [49] D.G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley Publishing Company, second edition, 1989. 48
- [50] L. Massoulié. Stability of distributed congestion control with heterogenous feedback delays. *IEEE Transactions on Automatic Control*, 47(6):895–902, 2002. 43

- [51] 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 *Proc. of ACM SIGCOMM*, 2000. 9
- [52] Katsuhiko Ogata. *Modern Control Engineering*. Prentice Hall, Inc., fourth edition, 2002. 55
- [53] F. Paganini. Flow control via pricing: a feedback perspective. In *Proc. of Allerton Conference*, 2000. 43, 54
- [54] F. Paganini, J. Doyle, and S. Low. Scalable laws for stable network congestion control. In *Proc. of IEEE Conference on Decision and Control*, 2001. 43
- [55] A. Papachristodoulou, L. Li, and J.C. Doyle. Methodological frameworks for largescale network analysis and design. *Computer Communication Review*, 34(3): 7–20, 2004. 4, 8
- [56] R.W. Ramirez. *The FFT: Fundamentals and Concepts*. Prentice-Hall, 1985. 33
- [57] S. Ryu, C. Rump, and C. Qiao. A predictive and robust active queue management for internet congestion control. In *Proc. of International Symposium on Computers and Communication*, 2003. 7, 12
- [58] G.J. Silva, A. Datta, and S.P. Bhattacharyya. *PID Controllers for Time-Delay Systems*. Birkhäuser, 2005. 3, 8, 13, 14, 32
- [59] R. Srikant. *Models and Methods for Analyzing Internet Congestion Control Algorithms*. Lecture Notes in Control and Information Sciences (LCNCIS). Springer-Verlag, 2004. 47
- [60] J. Sun, G. Chen, K.-T. Ko, S. Chan, and M. Zukerman. PD-controller: A new active queue management scheme. In *Proc. of IEEE GLOBECOM*, 2003. 7

- [61] Yu-Ping Tian. Stability analysis and design of the second-order congestion control for networks with heterogeneous delays. *IEEE/ACM Transactions on Networking*, 13(5):1082–1093, 2005. 1, 44
- [62] Michael M. Tiller. *Introduction to Physical Modeling with Modelica*. Springer, 2001. 67
- [63] M. Velasco, J.M. Fuertes, C. Lin, P. Marti, and S. Brandt. A control approach to bandwidth management in networked control systems. In *Proc. of IEEE Industrial Electronics Society (IECON04)*, 2004. 44, 45
- [64] G. Vinnicombe. On the stability of end-to-end congestion control for the Internet. [Online]. Available: <http://www.eng.cam.ac.uk/gv>, 2001. 43
- [65] G. Walsh, H. Ye, and L. Bushnell. Stability analysis of networked control systems. *IEEE Transactions on Control Systems Technology*, 10(3):438–4467, 2002. 44
- [66] Nathan A. Wedge, Ahmad T. Al-Hammouri, Michael S. Branicky, and Vincenzo Liberatore. Discussion on: “Development and experimental verification of a mobile client-centric Networked Controlled System”. *European Journal of Control*, 11(2): 243–246, 2005. 44
- [67] Eric W. Weisstein. Taylor series. [Online]. Available: <http://mathworld.wolfram.com/TaylorSeries.html>. 52
- [68] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanarama. One more bit is enough. In *Proc. of ACM SIGCOMM*, 2005. 50
- [69] Lei Ying, G.E. Dullerud, and R.L. Srikant. Global stability of Internet congestion controllers with heterogeneous delays. *IEEE/ACM Transactions on Networking*, 14(3):579–590, June 2006. ix, 1, 43, 51

- [70] M. Zhang, J. Wu, C. Lin, and K. Xu. Rethink the tradeoff between proportional controller and PI controller. In *Proc. of International Symposium on Computers and Communications*, 2002. 7
- [71] W. Zhang and M.S. Branicky. Stability of networked control systems with time-varying transmission period. In *Proc. of Allerton Conference*, 2001. 68
- [72] W. Zhang, M. Branicky, and S. Phillips. Stability of networked control systems. *IEEE Control Systems Magazine*, 21(1):84–99, February 2001. 44