Complete Stability Region Characterization for PI-AQM

Ahmad T. Al-Hammouri, Vincenzo Liberatore, Michael S. Branicky
Department of Electrical Engineering and Computer Science
Case Western Reserve University
Cleveland, Ohio 44106 USA
E-mail: {ata5, vl, mb}@case.edu
URL: http://vincenzo.liberatore.org/NetBots/

Stephen M. Phillips
Department of Electrical Engineering
Arizona State University
Tempe, Arizona 85287 USA
E-mail: stephen.phillips@asu.edu

***Abstract*— This paper describes a method for finding the stability regions of the PI and PIP controllers for TCP AQM with delays. The method is applied on several representative examples, showing that stable controllers can exhibit widely different performance. Thus, the results highlight the importance of optimizing the design of PI AQM. Furthermore, the paper shows that the previously proposed PIP controller can be unstable in the presence of delays even for the control parameters given in the literature.**

I. Introduction

Control-theoretical methods lead to stable, effective, and robust congestion control, but the limits of control performance are still largely unknown. Congestion control regulates the rate at which traffic sources inject packets into a network to ensure high bandwidth utilization while avoiding network congestion. End-point congestion control can be helped by Active Queue Management (AQM), whereby intermediate routers mark or drop packets prior to the inception of congestion. AQM has been extensively addressed by control-theoretical methods (see for example [1] and the references therein). However, it is still unclear whether existing AQM controllers achieve "optimal" performance. In particular, previous work lacks a complete characterization of the stability region, a definition of network-relevant control performance, and the design of provably optimal AQM controllers.

A long-term goal in congestion control is to understand the limits of control performance. In this paper, we describe the stability region of the Proportional-Integral (PI) AQM controllers. The stability region describes the set of feasible design points. Stable designs can be subsequently considered within an optimization framework. Therefore, the characterization of a stability region is the first essential step toward the design of optimal AQM controllers. The derivation of a stability region is involved due to time delays that arise from the non-negligible network latencies between sources, sinks, and routers. The paper exploits recent results on PI control theory for time-delay systems to obtain the PI stability region. We find that the controller performance varies significantly across the stability region and, in particular, there are stable controllers that have significantly better performance than previously proposed ones. Since stable PI controllers differed widely in performance, the results support the importance of finding optimal PI controllers. Furthermore, the paper shows that the previously proposed PIP controller [2] can be unstable in the presence of delays, even for the control parameters given in the literature.

AQM is one of the most mature areas in network control, but previous work has neglected the investigation of the stability region. The original RED controller has been analyzed in control-theoretical terms, and shown to be outperformed by PI [3]. The PI controller is a natural choice due to its robustness and its ability to eliminate the steady-state error. The original 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 [3]. However, there are other $(k_p,\ k_i)$ pairs that stabilize the closed-loop system *and* result in better performance. The PIP controller is a variant of PI [2]. Although PIP is stable in the absence of time delays, we show in this paper that PIP becomes unstable with time delays even in the exact scenarios considered by previous work.

This paper is organized as follows. In Section II, we introduce the linearized TCP-AQM model with PI and PIP controllers, and we present the method we used to obtain the complete stabilizing region. In Section III, we compute the complete set, $\mathcal{S}_R$, of stabilizing PI parameters. Simulations that stress the importance of extracting a complete stabilizing region are presented in Section IV. In Section V, we study the sensitivity of the stability region to network parameters. Directions for future work are given in Section VI, and conclusions in Section VII.

II. Background

A. Linearized TCP Model with the PI Controller

A fluid-based linearized model for TCP congestion control, delays, and queues is expressed by the transfer function [4]:

$$P(s) = \frac{B}{(s+\alpha)(s+\beta)} e^{-sd}\ , \tag{1}$$

where d is the round-trip delay (seconds), $\alpha = 2N/(d^2C)$, $\beta = 1/d$, $B = C^2/(2N)$, C is the bottleneck link capacity (packets/second), and N is the number of TCP flows traversing

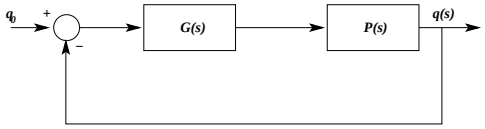

Fig. 1. The closed-loop system of TCP-AQM linearized model $P(s)$, with the PI controller, $G(s)$.

the link. The introduction of PI AQM results in the feedback control shown in Fig. 1 [3], 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 Laplace transform 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 context.

B. The PIP Controller

To enhance the speed of response of the PI controller, a position feedback compensation technique was proposed as an inner loop to the system in Fig. 1 [2]. The new arrangement, which is called a PIP controller, is shown in Fig. 2.

It can be shown with some mathematical manipulation that the characteristic equation of the closed-loop transfer function with PIP control is equal to that with PI control, except that the proportional gain, k_p, of the PI controller is increased by the value of k_h. Therefore, the stability region in the (k_p,k_i) plane for the system with the PIP scheme is simply the same stability region as with a PI controller shifted to the left by k_h. Therefore, the stability analysis can be restricted, without loss of generality, to the PI system in Fig. 1.

The original PIP stability argument disregards the delay term e^{-sd}. This simplification is a significant weakness because delays in feedback loops are known to reduce stability margins drastically. The experiments in Section IV will show the danger of neglecting delays in control design.

C. Analysis of Time-Delay Systems

Given a network topology with specific C and N, our goal is to determine all values of the (k_p,k_i) gains so that the feedback closed-loop system in Fig. 1 is stable for all values of delay less than d. Delays in the feedback loop are captured in (1) in the exponential term e^{-sd}, which in turn greatly complicates the stability analysis beyond the traditional textbook techniques of Control Theory [5]. Previous work sidestepped the problem through assumptions and by constraining the PI gains [3]. An alternative approach is to exploit recent results on time-delay systems [6]. This section reviews one such recent method for time-delay PI control, and the rest of the paper will apply this method for the stability analysis of TCP AQM.

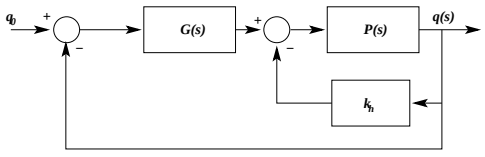

Fig. 2. The PIP controller: an inner loop with constant gain is introduced to provide position feedback compensation.

The stability region $\mathcal{S}_R$ is the complete set of points (k_p,k_i) for which the closed-loop system in Fig. 1 is stable for all delays L between 0 and d. The stability region $\mathcal{S}_R$ can be expressed as $\mathcal{S}_R = \mathcal{S}_1 \setminus \mathcal{S}_L$ [6, 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. Formally, $\mathcal{S}_N$ is

$$\mathcal{S}_N = \left\{ (k_p,k_i) : \lim_{s\to\infty} |G(s;k_p,k_i)P_0(s)| \geq 1 \right\}. \quad (2)$$

- $\mathcal{S}_L$ 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

$$\begin{aligned} \mathcal{S}_L \quad = \quad & \{(k_p,k_i) \notin \mathcal{S}_N : \exists L \in [0,d], \omega \in \mathbb{R} \text{ s.t.} \\ & G(j\omega;k_p,k_i)P_0(j\omega)e^{-jL\omega} = -1\}. \end{aligned} \quad (3)$$

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 $\hat{k}_p$:

$$\mathcal{S}_R = \bigcup_{\hat{k}_p} \mathcal{S}_{R,\hat{k}_p}. \quad (4)$$

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

$$\begin{aligned} \mathcal{S}_{1,\hat{k}_p} \quad &= \quad \{(k_p,k_i) \in \mathcal{S}_1 : k_p = \hat{k}_p\}, \\ \mathcal{S}_{N,\hat{k}_p} \quad &= \quad \{(k_p,k_i) \in \mathcal{S}_N : k_p = \hat{k}_p\}, \\ \mathcal{S}_{L,\hat{k}_p} \quad &= \quad \{(k_p,k_i) \in \mathcal{S}_L : k_p = \hat{k}_p\}. \end{aligned}$$

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 (3) 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:

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

where

$$\mathcal{S}^{+}_{L,\hat{k}_p} = \left\{ (\hat{k}_p, k_i) \notin \mathcal{S}_{N,\hat{k}_p} : \exists \omega \in \Omega^{+} . k_i = \sqrt{M(\omega)} \right\}, \quad (5)$$

$$\mathcal{S}^{-}_{L,\hat{k}_p} = \left\{ (\hat{k}_p, k_i) \notin \mathcal{S}_{N,\hat{k}_p} : \exists \omega \in \Omega^{-} . k_i = -\sqrt{M(\omega)} \right\}, \quad (6)$$

$$\Omega^{+} = \left\{ \omega : \omega > 0, M(\omega) \geq 0, \right.$$
$$\left. L(\omega) = \frac{\pi + \angle[(\sqrt{M(\omega)} + j\hat{k}_p\omega) R_0(j\omega)]}{\omega} \leq d \right\}, \quad (7)$$

$$\Omega^{-} = \left\{ \omega : \omega > 0, M(\omega) \geq 0, \right.$$
$$\left. \frac{\pi + \angle[(-\sqrt{M(\omega)} + j\hat{k}_p\omega) R_0(j\omega)]}{\omega} \leq d \right\}, \quad (8)$$

$$M(\omega) = \frac{1}{|R_0(j\omega)|^2} - \hat{k}_p^2 \omega^2 , \quad (9)$$

$$R_0(s) = \frac{P_0(s)}{s} . \quad (10)$$

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

In this section, we compute $\mathcal{S}_R$ for the PI controller of Fig. 1. 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.

A. 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 Fig. 1 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

$$s^3 + (\alpha+\beta)s^2 + (\alpha\beta + Bk_p)s + Bk_i = 0. \quad (11)$$

To compute $\mathcal{S}_0$, we construct the Routh array [5] as follows:

s^3	:	1	$\alpha\beta + Bk_p$
s^2	:	$\alpha+\beta$	Bk_i
s^1	:	$[(\alpha+\beta)(\alpha\beta+Bk_p) - Bk_i] / (\alpha+\beta)$	0
s^0	:	Bk_i	

A necessary and sufficient condition for stability is that all entries in the first column (after the colon) are positive [5, 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) $[(\alpha+\beta)(\alpha\beta+Bk_p) - Bk_i] / (\alpha+\beta) > 0$, 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

$$k_{i,\max} = \frac{(\alpha+\beta)(\alpha\beta+Bk_p)}{B}. \quad (12)$$

Moreover, for the range of k_i to be feasible, $(\alpha+\beta)(\alpha\beta + Bk_p)/B$ 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.

The shaded area in Fig. 3 is the permissible region that satisfies the stability conditions of 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 $\alpha\beta(\alpha+\beta)/B$.

B. Computing $\mathcal{S}_N$

Since

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

we have that $\mathcal{S}_N = \emptyset$ by definition (2) of $\mathcal{S}_N$. Thus, $\mathcal{S}_1 = \mathcal{S}_0$.

C. Computing $\mathcal{S}_L$ and $\mathcal{S}_R$

The stability region $\mathcal{S}_R$ will be plotted by following the analysis in Section II-C and using the results of Section III-A and III-B. Sweep through the values of k_p and for each $k_p = \hat{k}_p$:

- Compute the set Ω^{+} as in (7).
- Compute the set $\mathcal{S}^{+}_{L,\hat{k}_p}$ as in (5).
- Compute $\mathcal{S}_{R,\hat{k}_p} = \mathcal{S}_{1,\hat{k}_p} \setminus \mathcal{S}^{+}_{L,\hat{k}_p}$.

Then, the stability region $\mathcal{S}_R$ is obtained as in (4). Because we consider only positive gain values, we ignore the two cases of (8) and (6).

IV. Simulations

In this section, we use the analysis of Section III to compute the stability regions for examples with PI and PIP controllers. Furthermore, we use Simulink® [7] to simulate the step-input response of the continuous-time fluid-based control systems in Fig. 1 and in Fig. 2. In the simulations, a non-linear saturation element is added to prevent the queue from growing

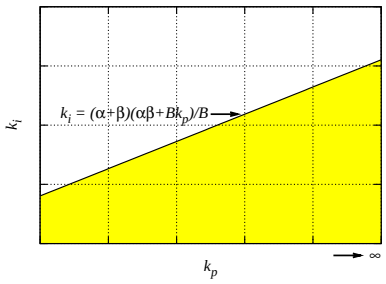

Fig. 3. Stabilizing region of k_p and k_i gains for the non-delay closed loop system.

negative. Although such a saturation element is easily added in simulation, it has been disregarded in the analysis above and in previous work because its non-linearity makes the system analysis intractable.

A. PI Simulations

Example 4.1: Consider a network with the following parameters: $N = 60$, $C = 3750$ pkt/sec, $d = 0.246$ sec, and $q_0 = 50$ (the same scenario as in [3, Example 2]). The region of stabilizing k_p and k_i is shown in Fig. 4. The black dot represents the point $(k_p^\star, k_i^\star) = (1.8497 \cdot 10^{-5}, 9.7811 \cdot 10^{-6})$ prescribed in [3]. The system response (the queue length) with these values is shown in Fig. 5. Other points inside the stability region did not give better performance than the one chosen by [3]. For example, Fig. 6 shows the response when the PI parameters are in the middle of the region, i.e., $(k_p, k_i) = (10^{-4}, 6 \cdot 10^{-5})$.

Example 4.2: Let $N = 60$, $C = 1250$ pkt/sec, $d = 0.22$ sec, and $q_0 = 50$ (as in [2]). Now, $k_p^\star$ and $k_i^\star$ according to [3] are $7.5546 \cdot 10^{-4}$ and $1.4984 \cdot 10^{-3}$, respectively. The complete region of stabilizing k_p and k_i is shown in Fig. 7 along with the point $(k_p^\star, k_i^\star)$. The system response (the queue length) with these values is shown in Fig. 8. Fig. 9 is the output response when using another set of parameters, $(k_p, k_i) = (3 \cdot 10^{-4}, 5.9 \cdot 10^{-4})$, which shows improved performance over the set of $(k_p^\star, k_i^\star)$.

Example 4.3: As a third example, assume the network parameters are $N = 75$, $C = 1250$ pkt/sec, $d = 0.15$ sec, and $q_0 = 50$. Fig. 10 shows the stabilizing region along with the point $(k_p^\star, k_i^\star) = (0.0044, 0.0233)$ that results from applying the method in [3]. Fig. 11 shows the system response using $k_p^\star$, and $k_i^\star$. The response exhibits oscillations, an overshoot of about 100%, and a relatively long settling time. On the other hand, when using another set of stable controller gains, such as $(k_p, k_i) = (0.002, 0.005)$, the response settles at 50 packets in a much shorter time, without oscillations and without overshoot; see Fig. 12.

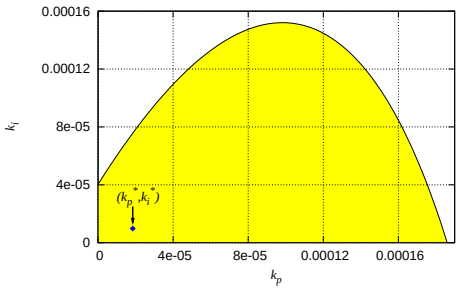

Fig. 4. Stabilizing (k_p, k_i) region for Example 4.1.

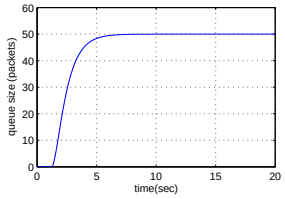

Fig. 5. The output response for Example 4.1 using $(k_p^\star, k_i^\star) = (1.8497 \cdot 10^{-5}, 9.7811 \cdot 10^{-6})$.

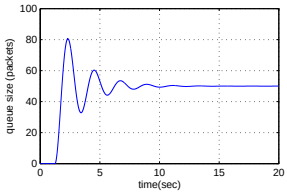

Fig. 6. The output response for Example 4.1 using $(k_p, k_i) = (1.0 \cdot 10^{-4}, 6.0 \cdot 10^{-5})$.

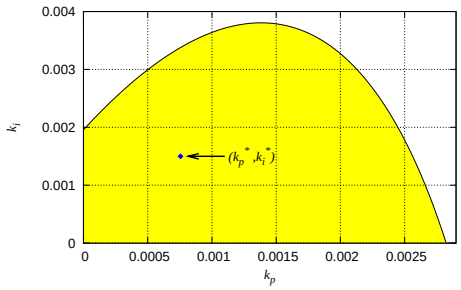

Fig. 7. Stabilizing (k_p, k_i) region for Example 4.2.

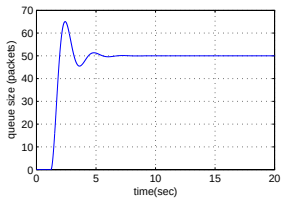

Fig. 8. The output response for Example 4.2 using $(k_p^\star, k_i^\star) = (7.5546 \cdot 10^{-4}, 1.4984 \cdot 10^{-3})$.

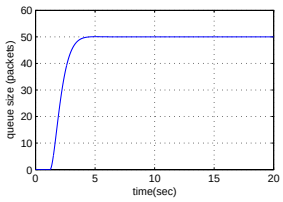

Fig. 9. The output response for Example 4.2 using $(k_p, k_i) = (3.0 \cdot 10^{-4}, 5.9 \cdot 10^{-4})$.

B. PIP Simulations

Example 4.4: For the PIP controller, we show the original PIP experiment [2]. The network parameters are $C = 1250$ pkt/sec, $d = 0.22$ sec, and $q_0 = 50$. As in [2], we let $k_h = 0.0014$, $k_p = 2.0 \cdot 10^{-3}$, and $k_i = 5.0 \cdot 10^{-3}$.
The stability region is same shape as that of Example 4.2 shifted to the left by $k_h = 0.0014$ (see Sec. II-B), and it is shown in Fig. 13. The chosen gains correspond to the point in the figure on the upper right, and are outside the stability region. Correspondingly, the step input response of the linearized systems grows unbounded (see Fig. 14). The system is then simulated with the addition of a non-linear element to bound $q(t) \geq 0$ (see Fig. 16) and the output response is shown in Fig. 15.

The saturation element is keeping the output bounded, but it cannot prevent severe queue oscillations.

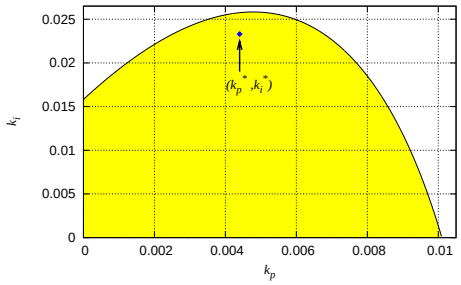

Fig. 10. Stabilizing (k_p, k_i) region for Example 4.3.

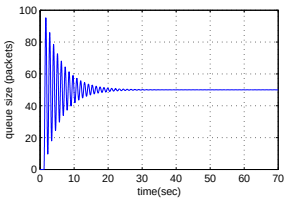

Fig. 11. The output response for Example 4.3 using $(k_p^\star, k_i^\star) = (0.0044, 0.0233)$.

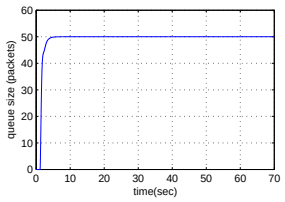

Fig. 12. The output response for Example 4.3 using $(k_p, k_i) = (0.002, 0.005)$.

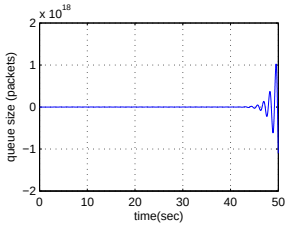

Fig. 14. The output response for the PIP without adding the saturation element that bounds $q(t) \geq 0$ (Example 4.4).

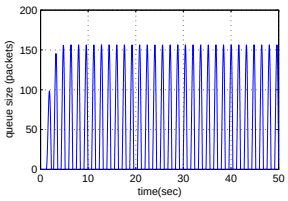

Fig. 15. The output response for the PIP when adding the saturation element to bound $q(t) \geq 0$ (Example 4.4).

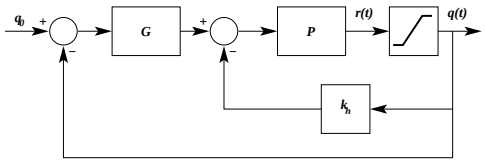

Fig. 16. The PIP controller: the inclusion of the saturation element, $\max\{0, r(t)\}$, to bound $q(t) \geq 0$ in simulations.

The flawed design of the PIP controller in [2] was also pointed out in [8].

V. Sensitivity of $\mathcal{S}_R$ to N, d, and C

In this section, we study the impact of the individual network parameters d, N, and C on the stability region $\mathcal{S}_R$. In each case, we fix two of the three parameters and we plot $\mathcal{S}_R$ for different values of the third investigated parameter. Here, we show the plots of $\mathcal{S}_R$ as the investigated parameter varies between *four* different levels (or values). There are two reasons why we show the plots of $\mathcal{S}_R$ for only four levels of the investigated parameter. First, the same conclusions carry over whether the investigated parameter varies between four or more number of levels. Second, as the number of the investigated parameter's levels increase, $\mathcal{S}_R$ plots become crowded and cannot be shown conveniently on the same scale.

A. The Impact of d on $\mathcal{S}_R$

Let $N = 60$ and $C = 3750$ (as in Example 4.1). We plot $\mathcal{S}_R$ for the following values of d: 0.1, 0.2, 0.3, and 0.4 seconds, see Fig. 17. It is clear that $\mathcal{S}_R$ is very sensitive to d and that $\mathcal{S}_R$ shrinks in size as d increases. Also, since $\mathcal{S}_R$ for a large value of d is contained within $\mathcal{S}_R$ for a smaller value of d, then the PI controller that stabilizes a closed-loop system for some $d = d_0$ still stabilizes a closed-loop system with $d < d_0$. This can also be implied from the definition of $\mathcal{S}_R$ in Section II-C. Therefore, it is advisable to always design a controller based on the maximum expected d.

B. The Impact of N on $\mathcal{S}_R$

Let d and C be 0.1 second and 3750 pkt/sec, respectively. We plot $\mathcal{S}_R$ for the following values of N: 25, 60, 100, and 150, see Fig. 18. Now, $\mathcal{S}_R$ expands in size as N increases. In addition, the PI controller that stabilize the system for some $N = N_0$, would stabilize a system with $N > N_0$. In contrast to the case of d, one should design the controller for the minimum expected value of N.

C. The Impact of C on $\mathcal{S}_R$

Let d and N be 0.1 second and 60, respectively. We plot $\mathcal{S}_R$ for the following values of C (assuming average packet size is 1000 Bytes): 30 (3750), 45 (5625), 60 (7500), and 100 (12500) Mbps (pkt/sec), see Fig. 19. From Fig. 19, the increase of C causes $\mathcal{S}_R$ to shrink.

The parameter C is different from both d and N. Usually, C is constant and known by the network administrator unlike d and N. Figure 19 is not then intended to advise that the minimum or the maximum expected value of C should be used in determination of k_p and k_i; rather, it shows that as the bandwidth of a link increases, $\mathcal{S}_R$ decreases in size.

In conclusion, given a minimum number of TCP sessions N_0, a maximum round-trip delay d_0, and link capacity C, the stability region $\mathcal{S}_R$ obtained for N_0, d_0 and C will stabilize the system for all $N \geq N_0$ and $d \leq d_0$.

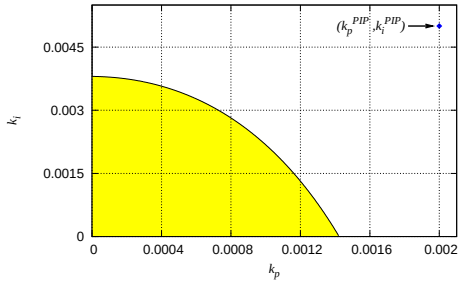

Fig. 13. Stabilizing (k_p, k_i) region for PIP controller with the parameters of Example 4.4.

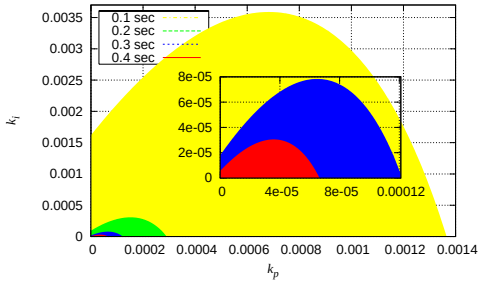

Fig. 17. As d increases, $\mathcal{S}_R$ shrinks in size.

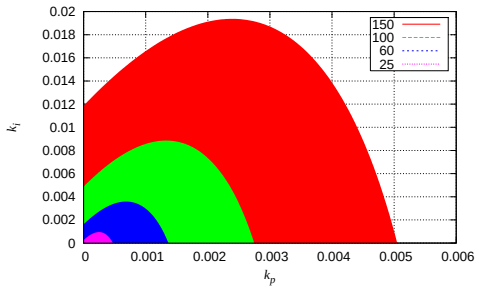

Fig. 18. As N increases, $\mathcal{S}_R$ increases in size.

VI. Future Work

This paper makes an indispensable contribution to the understanding of PI controllers by providing a complete characterization of their stability regions. This characterization paves the way to several avenues of future work.

In the first place, the Simulink® simulations describe the continuous-time dynamics of the flows and queues. However, continuous-time systems are but an abstraction and a simplification of the actual packet dynamics. Thus, the Simulink® evaluation should be complemented with packet-level simulations and emulations, e.g., via `ns-2` [9]. More fundamentally, continuous-time controllers must be translated into discrete-time to be implemented in AQM routers. The translation from continuous- to discrete-time is called discretization and it includes the choice of a discretization method and the choice of a sampling frequency. The discretization may then affect the performance of the packet-level system.

The design of an optimal PI controller is the problem of finding the k_p and k_i parameters that maximize or minimize a certain objective function and that lead to a stable controller. Hence, the stability region is the set of feasible solution to an optimization problem that is yet to be addressed. In the first place, the optimization problem requires that an objective function be defined so as to express convincingly the congestion control goals that are relevant to networks. Further, the optimization problem must be solved to obtain the optimal controller.

Another subject of future research is the analysis of the non-linear element that bounds $q(t) \geq 0$. Since this non-linear element prevented the queue length from growing unbounded (see Example 4.4), there stems a need to explore how beneficial it can be, for example, for choosing relatively larger controller gains to speed up the convergence time.

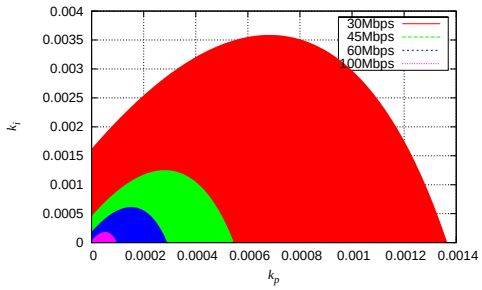

Fig. 19. As C increases, $\mathcal{S}_R$ shrinks in size.

VII. Conclusions

In this paper, we have characterized the stability region of PI AQM controllers. The derivations are involved due to the presence of time delays in the control loops. The paper has shown a test to determine whether a control design is stable or not. The understanding of the PI AQM stability region is an essential step in the design of optimal PI controllers. For example, the paper presented examples of PI controllers that are stable and have significantly better performance than previously proposed ones.

The paper has highlighted the importance of giving a complete characterization of the stability region to evaluate alternative designs. In general, the paper is the first step toward the application of optimal control methods to congestion control. We speculate that similar procedures could be useful in other areas of feedback control of computer systems. Another lesson learned is that it is hard to deal with time delays, but delays are unavoidable in networks systems. In particular, we have highlighted the danger of aggressive controllers, such as PIP, that become unstable in the presence of time delays.

Acknowledgment

This work was supported in part under NSF CCR-0329910, Department of Commerce TOP 39-60-04003, NASA NNC04AA12A, and an OhioICE Training grant.

References

[1] C. Hollot, V. Misra, D. Towsley, and W. Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Transactions on Automatic Control*, vol. 47, no. 6, pp. 945–959, 2002.

[2] Z. Heying, L. Baohong, and D. Wenhua, "Design of a robust active queue management algorithm based on feedback compensation," in *Proceedings of ACM SIGCOMM*, 2002.

[3] C. Hollot, V. Misra, D. Towsley, and W. Gong, "On designing improved controllers for AQM routers supporting TCP flows," in *Proceedings of IEEE INFOCOM*, 2001.

[4] ——, "A control theoretic analysis of RED," in *Proceedings of IEEE INFOCOM*, 2001.

[5] G. Franklin, J. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*, 3rd ed. Prentice Hall, 1999.

[6] G. Silva, A. Datta, and S. Bhattacharyya, *PID Controllers for Time-Delay Systems*. Birkhäuser, 2005.

[7] Simulink® version 6.1. The MathWorks Inc., 2004.

[8] A. Papachristodoulou, L. Li, and J. Doyle, "Methodological frameworks for largescale network analysis and design," *Computer Communication Review*, vol. 34, no. 3, pp. 7–20, 2004.

[9] "The network simulator—`ns-2`," [Online]. Available: http://www.isi.edu/nsnam/ns/.