A control theoretical analysis of a window-based flow control mechanism in TCP/IP networks

Hiroyuki Ohsaki, Masayuki Murata, Toshimitsu Ushio, and Hideo Miyahara

Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan

(Phone) +81-6-6850-6588
(Fax) +81-6-6850-6589
(E-mail) oosaki@ics.es.osaka-u.ac.jp

Abstract

A window-based flow control mechanism is a sort of feedback-based congestion control mechanisms, and has been widely used in current TCP/IP networks. Recently proposed TCP Vegas is another version of the TCP mechanism and has potential to achieve much better performance than current TCP Tahoe and Reno. However, it has not been fully investigated how to determine control parameters of TCP Vegas. In this paper, we focus on a window-based flow control mechanism based on a congestion avoidance mechanism of TCP Vegas, and analyze its stability and transient behavior using a control theoretic approach. We show that the system stability is quite sensitive to the control parameter of the window-based flow control mechanism, and that its optimal value is particularly dependent on the end-to-end propagation delay. We also demonstrate that its performance can be dramatically improved by dynamically adjusting its control parameters based on our analytic results. We further investigate a buffer dimensioning issue at the router, and derive the condition for achieving full link utilization and preventing packet loss.

1 Introduction

In a packet-switched network, a feedback-based congestion control mechanism is essential to provide data transfer services efficiently. Its main objective is to prevent packet losses in the network, and to utilize network resources effectively. The current Internet uses a window-based flow control mechanism in its TCP (Transmission Control Protocol), as the feedback-based congestion control mechanism. As an example, a version of TCP mechanism called *TCP Reno* uses packet losses in the network as feedback information since packet losses implies congestion occurrence in the network [1, 2]. Until packet loss occurs, TCP Reno gradually increases its window size that limits the number of in-flight packets in the network. As the window size is over its available bandwidth, excess packets are queued at the buffer of intermediate routers for some period. If the window size increases further, packets at the buffer of the router overflows, leading packet losses. The source host detects occurrence of packet losses in the network from, for example, the timeout mechanism, and reduces its window size to one packet. TCP Reno has another mechanism called *fast retransmit* to detect packet losses, which is triggered by receipt of duplicate ACK packets. After reduction of the window size, congestion in the network is remedied so that congestion is relieved. The source host then increases its window size again. In short, the congestion control mechanism of TCP Reno first increases its window size, and as soon as it detects packet losses in the network, it reduces its window size. TCP Reno repeats this process indefinitely during the connection.

Another version of TCP called *TCP Vegas* has been proposed by Brakmo *et al.*, which can achieve better performance than TCP Reno [3, 4]. TCP Vegas has following advantages over TCP Reno: (1) a new retransmission mechanism, (2) an improved congestion avoidance mechanism that controls buffer occupancy, and (3) a modified slow-start mechanism. With these features, it has been reported in [4] that total throughput of TCP Vegas becomes 37–71 % better than TCP Reno, and that the number of retransmitted packets of TCP Vegas can be reduced to about 1/5–1/2 of TCP Reno. In [5], it has been reported that TCP Vegas keeps less data in the network than TCP Reno, resulting in shorter RTT average and variances. The performance improvement is mainly achieved by the congestion avoidance mechanism of TCP Vegas, which uses a measured round-trip time of the packet — i.e., duration between the source host sends a packet and it receives its corresponding ACK (acknowledgment) packet. More specifically, TCP Vegas measures a round-trip time of a packet, and estimates the number of queued packets in the router's buffer. It then controls its window size to make it constant. There is no need for the source host to wait for packet losses to know occurrence of congestion in the network. The window size of TCP Vegas becomes stabilized when the network is in steady state, and therefore it can achieve much better throughput than TCP Reno.

However, most of past studies regarding window-based flow control mechanisms have primarily focused on their performance evaluation. That is, ad hoc control parameters (e.g., an amount of window size increase and/or decrease) are used in those studies, and their performance metrics are through simulation experiments. Therefore, an approach to determine adequate control parameters is of great necessity to utilize network resources efficiently. Since window-based congestion control is essentially a feedback-based control mechanism, we hope that a control theoretical approach is quite worthwhile for building a theoretical framework of packet-switched networks and the Internet. So far, the Markovian queueing model has been a powerful tool for modeling and evaluating the target system. However, the above-mentioned data transmission protocol is essentially a feedback system and the Markovian model (including the queueing network model) has no means to provide an effective way to evaluate the control mechanism of TCP. The model treated in the current paper is our first one and its rather simple, but we believe that our approach shown gives a framework for treating the congestion control mechanism of TCP.

The primary goal of this paper is to investigate a window-based flow control mechanism based on the congestion avoidance mechanism of TCP Vegas by applying a control theoretic approach. TCP Vegas always changes its window size by one. Namely, TCP Vegas increases or decreases its window size by one packet every one round-trip time. Our analytic model, however, differs from TCP Vegas in a point that the amount of window size increase/decrease is specified by a control parameter. In [6], fairness and stability of TCP Vegas have been studied by an analytic approach, and an improvement in TCP Vegas has been proposed. The key idea of their improvement is to get rid of a condition in the TCP Vegas' congestion avoidance algorithm — a condition that window sizes of source hosts are unfairly stabilized. In this paper, we adopt this idea, and the undesirable operation of TCP Vegas is removed in our analytic model.

We first derive a condition that window sizes of TCP connections and a queue length (i.e., the number of packets waiting in the router's buffer) are stabilized in steady state, which we call *a stability condition*. We then investigate an effect of system parameters (e.g., the number of active connections and the propagation delay) on system stability through numerical examples. We next focus on a transient behavior of the system, and investigate the ideal value of control parameters to achieve reasonable transient performance while keeping system stable. We also demonstrate that its performance can be dramatically improved by dynamically adjusting its control parameters based on our analytic results.

In [7-10], control theoretical approaches have been taken to analyze various types of feedback-based congestion control mechanisms. However, these studies have targeted not window-based but rate-based congestion control mechanisms. The fundamental difference between rate-based and window-based congestion control mechanisms is in their packet transmission methods. Namely, in window-based approaches, packet transmission is suspended whenever the number of unacknowledged packets runs up to the current window size. On the contrary, in rate-based ones, packet transmission is performed without discontinuation. Moreover, most congestion control mechanisms in the literature are defined only by linear equations [7, 9, 11-14]. In this paper, we analyze more realistic and complicated congestion control mechanism; our analysis takes account of the number of packets queued in the router's buffer that affects the round-trip time. It is an intrinsic feature of the window-based congestion control mechanism adopted in TCP Vegas, and should not be neglected. Then, our analytic approach leads to non-linear equations as shown below.

Organization of this paper is as follows. In Section 2, we explain our window-based congestion control mechanism based on the congestion avoidance mechanism of TCP Vegas followed by introduction of our analytic model. In Sections 3 and 4, stability and transient analyses are performed, and relation between control parameters and its dynamics is investigated. In Section 5, several simulation results are presented to validate our analysis. Section 6 discusses applicability of our analysis to real networks. In Section 7, we propose an adaptive control mechanism for determining a control parameter, and show that performance can be dramatically improved with this mechanism. We discuss issues on the queue length control and the buffer dimensioning at the bottleneck router in Section 8. In Section 9, we conclude this paper with few remarks on buffer dimensioning of the bottleneck router, follwed by discussion on future works.

2 Analytic Model

We first explain the congestion avoidance mechanism of TCP Vegas. For detailed explanation, refer to [4]. In TCP Vegas, each source host maintains τ, which is a minimum round-trip time obtained when the network is not congested. That is, the minimum round-trip time τ corresponds to the sum of all propagation delays and processing delays at the routers. Hereafter, we call the minimum round-trip time, τ, the *propagation delay* for brevity. The source host is allowed to emit packets of its current window-size (denoted by w) per round-trip time. Therefore, its effective throughput would be w/τ if there is no congestion in the network. Each source host obtains

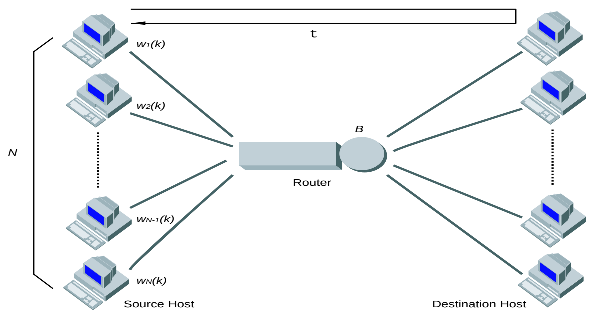


Figure 1: Analytic model.

the actual round-trip time by measuring time duration between a transmission time of a packet and arrival of its corresponding ACK packet. Let r be the actual round-trip time measured at the source host, and $\overline{w}$ be the number of packets the source host sent in the previous round-trip time. Its actual throughput is given by $\overline{w}/r$. TCP Vegas then computes the difference between expected throughput and actual throughput as

$$d = \frac{w}{\tau} - \frac{\overline{w}}{r}.$$

TCP Vegas changes its window size, w, according to relations among d and two threshold values, α and β. If d is less than α, the window size is linearly increased by one packet in the next round-trip time. If d is greater than β, the window size is linearly decreased by one packet in the next round-trip time. Otherwise, the window size is unchanged.

In this paper, we model the above-mentioned congestion control mechanism of TCP Vegas as follows. Figure 1 depicts our analytic model used throughout this paper. The number N of source hosts are connected to corresponding destination hosts through a single bottleneck router. TCP Vegas changes its window size once every round-trip time. We therefore consider the system as a discrete-time model, where each time slot corresponds to the round-trip time. Note that since the round-trip time changes as the network status changes, the length of one slot is not fixed in our model.

Let $w_n(k)$ be the window size of the source host n $(1 \leq n \leq N)$ at slot k. This indicates that the source host n can inject $w_n(k)$ packets into the network during slot k. We assume that each source host always has packets to transmit so that the number $w_n(k)$ of packets are sent at slot k. Let $q(k)$ be the number of packets queued in the router's buffer at slot k, and L be the buffer size of the router. At the router, all packets coming from source hosts are processed in a FIFO (First-In First-Out) manner; that is, all packets are first queued in the single buffer, and then transmitted onto the output link in order. We denote the bandwidth of the router (i.e., the processing speed of the router or the bandwidth of the output link) by B. Note that $w_n(k)$ (the window size), $q(k)$ (the number of packets in the router's buffer), and L (the buffer size) are represented in units of packets.

During a round-trip time, TCP Vegas allows the source host consume the bandwidth being worth of its given window size. Provided that round-trip times of all connections are equal, the number of packets in the buffer at slot $k+1$, $q(k+1)$, is given by the following equation.

$$q(k+1) = \min(\max(q(k) + \sum_{n=1}^{N} w_n(k) - B\, r(k), 0), L)$$

where $r(k)$ denotes the round-trip time at slot k.

TCP Vegas changes its window size based on the measured round-trip time. The difference between the expected throughput and the actual throughput $d(k)$ is computed as

$$d(k) = \frac{w_i(k)}{\tau} - \frac{w_i(k)}{r(k)} \qquad (1)$$

where τ is the round-trip time when there is no waiting packets in the router's buffer. The round-trip time $r(k)$ is determined by τ and the number of packets in the buffer; Namely,

$$r(k) = \tau + \frac{q(k)}{B}$$

TCP Vegas linearly increases or decreases its window size based on $d(k)$. The window size of the source host n at slot $k+1$ is determined as

$$w_n(k+1) = \begin{cases} w_n(k)+1 & \text{if } d(k) < \alpha \\ w_n(k)-1 & \text{if } d(k) > \beta \\ w_n(k) & \text{otherwise} \end{cases} \qquad (2)$$

In the above equation, two threshold values (α and β) are control parameters at the source host, which specify the amount of excess packets the source host is permitted to send in a round-trip time. However, we modify Eq. (2) as follows.

$$w_i(k+1) = \max(w_n(k) + \delta(\gamma - d(k)), 0) \qquad (3)$$

where δ is a control parameter that determines the amount of increase/decrease of the window size in a round-trip time. The purpose of introducing δ is not only for enabling application of a control theory, but also for improving transient performance as will be demonstrated in Section 4. In [6], it has been reported that fairness among connections cannot be satisfied when $d(k)$ lies in $[\alpha, \beta]$. In our analytic model, we therefore unify α and β in Eq. (2) into γ as in Eq. (3). With this modification, fairness among connections can be improved [6]. Intuitively, γ controls the number of in-flight packets in the network for each connection.

When packet loss occurs in the network, the congestion control mechanism of TCP Vegas halves its window size. In our analysis, this behavior of TCP Vegas is not considered since packet loss never occurs as long as control parameters satisfy the stability condition, which will be derived in Section 3.

3 Stability Analysis

For simplicity, we assume that the initial window sizes of all source hosts are equal, and that all source hosts change their window sizes according to Eq. (3). The number of packets in the router's buffer at slot $k+1$ is given by

$$\begin{aligned} q(k+1) &= \min(\max(q(k) + N\,w(k) - B\,r(k), 0), L) \\ &= \min(\max(N\,w(k) - B\,\tau, 0), L) \qquad (4) \end{aligned}$$

where $w(k) \equiv w_n(k)$ for all n's. Note that $q(k+1)$ is not dependent on $q(k)$. This is because each time slot is defined as a round-trip time in our analytic model. Namely, in a window-based flow control mechanism, all packets sent in slot k must be successfully transmitted at the beginning of slot $k+1$.

Let w^*, q^*, and d^* be the fixed points of $w(k)$, $q(k)$, and $d(k)$ when $k \to \infty$. By using Eqs. (1), (3), and (4), w^*, q^*, and d^* can be obtained as follows.

$$\begin{aligned} w^* &= \tau\left(\frac{B+\gamma N}{N}\right) && (5) \\ q^* &= \gamma N \tau && (6) \\ d^* &= \gamma && (7) \end{aligned}$$

Since $w(k)$ is given by a non-linear equation, we linearize it around the fixed point. Let $\mathbf{x}(k)$ be the difference from the fixed point, defined by

$$\mathbf{x}(k) = \left[\begin{array}{c} w(k) - w^* \\ q(k) - q^* \end{array}\right]$$

$\mathbf{x}(k+1)$ is given by

$$\mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) \qquad (8)$$

where

$$\mathbf{A} = \left[\begin{array}{cc} 1 - \frac{\delta}{\tau} + \frac{B\delta}{(B+\gamma N)\tau} & -\frac{B\delta}{N(B+\gamma N)\tau} \\ N & 0 \end{array}\right]$$

In the system defined by Eqs. (1), (3), and (4), the fixed point (w^*, q^*) is locally exponentially stable when the roots of the characteristic equation s_i $(i = 1, 2)$ satisfy $|s_i| < 1$. Note that the characteristic equation is given by

$$D(s) \equiv |s\mathbf{I} - \mathbf{A}| = 0 \qquad (9)$$

Since the characteristic equation $D(s)$ is quadratic, this condition is equivalent to the following inequalities [15].

$$\begin{aligned} D(1) &> 0 \\ D(-1) &> 0 \\ |D(0)| &< 1 \end{aligned}$$

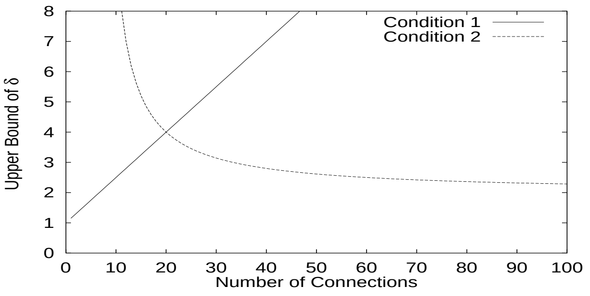


Figure 2: Stability region in the δ–N plane (B = 20 packet/ms, $\tau = 1$ ms, $\gamma = 3$ packet)

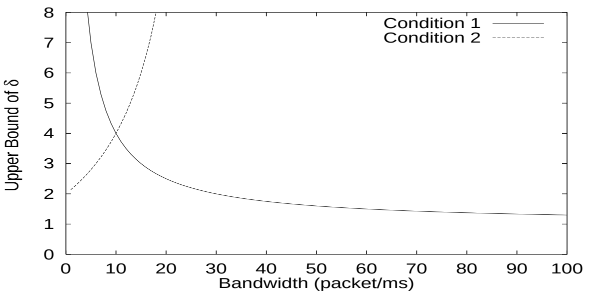


Figure 3: Stability region in the δ–N plane ($N = 10$, $\tau = 1$ ms, $\gamma = 3$ packet)

Hence, the fixed point of the system (w^*, q^*) is locally exponentially stable if and only if the following inequalities hold.

$$\begin{aligned} \delta &> 0 && (10) \\ \frac{\delta(B-\gamma N)}{(B+\gamma N)\tau} + 2 &> 0 && (11) \\ \frac{B\delta}{(B+\gamma N)\tau} &< 1 && (12) \end{aligned}$$

Figure 2 shows a stability region in the δ–N plane, where the bandwidth of the router B is set to 20 packet/ms, the propagation delay τ is to 1 ms, and the desired number of in-flight packets γ is to 3. The number of connections N is changed from 1 to 100 in this figure. The dotted line (labeled by "Condition 1") and solid line (labeled by "Condition 2") correspond to Eqs. (11) and (12), respectively. The system is stable if δ is lower than both of these lines. As the number of connections N reaches 20, δ can take a large value. However, after it gets beyond 20, δ should be smaller as N increases. It is shown in the figure that the network is always stable for any $N > 0$ if δ is less than 1. However, from Eq. (3), the congestion control mechanism is regarded as a static feedback system with feedback gain δ from a control theoretical viewpoint. It means that if δ is set to be larger

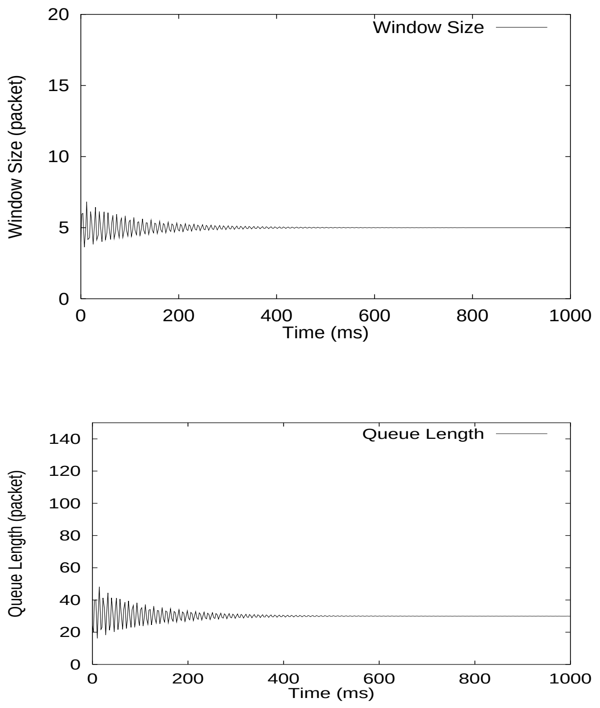


Figure 4: Stable behavior ($\delta = 2.4$, $B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\gamma = 3$ packet)

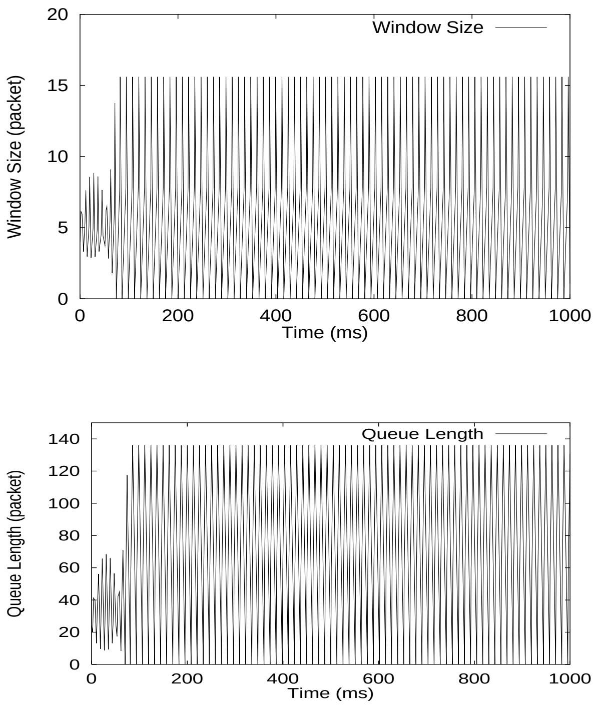


Figure 5: Unstable behavior ($\delta = 2.6$, $B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\gamma = 3$ packet)

(i.e., we choose high gain feedback), transient behaviors of the system would be better while the stability would be lost to some extent. Figure 2 shows such a tendency.

Another important factor for deciding δ is the router's bandwidth (or the router's packet processing capability in our modeling). Figure 3 shows a stability region when the number of connections N is fixed at 10 while the router's bandwidth is changed from 1 to 100 packet/ms. In this figure, other parameters are equal to those of Fig. 2. One can find that an appropriate value of δ depends not only on the number of connections but also on the router's bandwidth. Namely, δ must be carefully chosen where each router might have a different capability. One suggestion is that for safety, δ should be decided according to the router with least bandwidth.

To see how the value of δ affects the system stability, we next investigate dynamical behaviors of the system when δ is either inside or outside of the stability region. From Eqs. (10)–(12), the system is stable if $0 < \delta < 2.5$. In Figs. 4 and 5, we show behaviors of the window size $w(k)$ and the number of packets $q(k)$ when the network is either stable ($\delta = 2.4$) and unstable ($\delta = 2.6$) for $N = 10$. Note that initial values of the window size $w(0)$ and the number of packets in the router's buffer $q(0)$ are set to 80% of w^* and q^*, respectively. Evolutions of $w(k)$ and $q(k)$ are obtained by numerical computation using Eqs. (3) and (4). When δ is in the stability region as in Fig. 4, both the window size and the number of packets in the router's buffer converge to the fixed point within 500 ms, indicating stable operation of the system. However, if δ is out of the stability region, they oscillate with large amplitude as shown in Fig. 5, resulting in an unstable operation. The maximum number of packets in the router's buffer in Fig. 5 is about 130 packets, whereas 50 packets in Fig. 4. This would cause increase of packet transmission delays due to longer waiting time in the router's buffer. Moreover, it can be found that the queue length drastically oscillates in Fig. 5, and the router's buffer periodically becomes empty.

This tendency becomes apparent as the propagation delay increases. Shown in Figs. 6 and 7 are dynamical behaviors of the window size and the number of packets when the propagation delay τ is set to 5 ms. The stability condition (Eqs. (10)–(12)) suggests that in this case the system becomes stable if $0 < \delta < 12.5$. We thus set $\delta = 12$ (stable) and $\delta = 13$ (unstable) in these cases. Figure 6 shows that the system slowly converges to the fixed point since δ satisfies the stability condition. Slower convergence than the case of $\tau = 1$ ms (Fig. 4) is resulted from less update frequency of the window size. That is, since the window size is changed once per a round-trip time, changes of the window size become slow as the

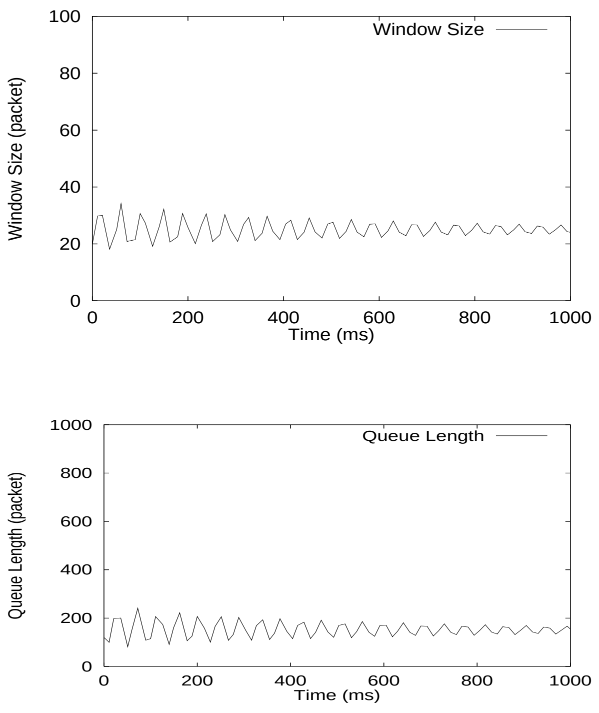


Figure 6: Stable behavior ($\delta = 12$, $B = 20$ packet/ms, $N = 10$, $\tau = 5$ ms, $\gamma = 3$ packet)

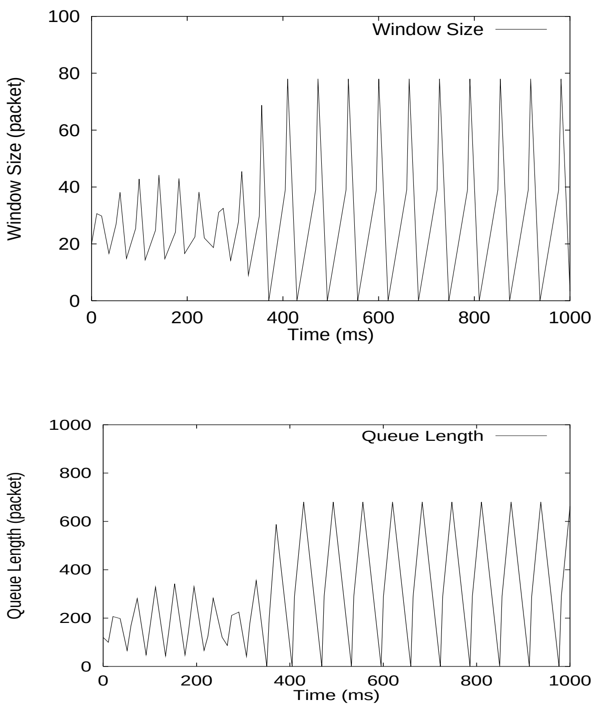


Figure 7: Unstable behavior ($\delta = 13$, $B = 20$ packet/ms, $N = 10$, $\tau = 5$ ms, $\gamma = 3$ packet)

propagation delay increases. When δ is out of the stability region as in Fig. 7, both of the window size and the number of packets in the buffer oscillate excessively. Because of the system instability, the window size of the source host happens to become zero for some time. In such a situation, packet transmission from the source host must be kept waiting in the source host's buffer. In the figure, it can also be found that the maximum number of packets in the buffer becomes quite large (i.e., 670 packets) compared with the previous cases. This indicates that the great amount of buffer must be provided at the router to prevent packet losses.

From these numerical results, one can find that stability of the system is heavily affected by choice of δ. In particular, when the propagation delay is large, inappropriate configuration of δ easily causes unstable operation of the network, and leads to terrible performance degradation. Moreover, convergence speed becomes slow as the propagation delay increases. The system would be asymptotically stable as long as δ satisfies the stability condition (Eqs. (10)–(12)). However, for realization of efficient networks, δ should be chosen with enough consideration on its transient behavior. In Section 4, we further investigate a selection method of control parameters by taking account of tradeoff between system stability and convergence speed.

4 Transient Behavior

Transient behavior of the system defined by Eqs. (1), (3), and (4) are determined by roots of its characteristic equation. Let s_i be the roots of the characteristic equation. Then, s_i is given by

$$\begin{aligned} s_i &= \frac{1}{2(B+\gamma N)\tau}\left(-\delta\gamma N + B\tau + \gamma N\tau\right. \\ &\qquad \left.\pm\sqrt{-4B\delta\tau(B+\gamma N) + (\delta\gamma N - B\tau - \gamma N\tau)^2}\right) \end{aligned}$$

Convergence speed to the fixed point is determined by $|s_i|$ (i.e., vector length in Euclid space). For instance, the system converges to the fixed point faster as $|s_i|$ is closer to zero. Therefore, δ should be chosen to minimize $|s|$ defined as

$$|s| \equiv max(|s_1|, |s_2|) \tag{13}$$

which determines convergence speed in transient state. Recall that for system stability, δ should satisfy the stability conditions given by Eqs. (10)–(12).

Figure 8 shows the relation between δ and $|s|$. The bandwidth of the router B is set to 20 packet/ms, the control parameter γ is to 3 packet, and the propagation delay τ is changed as 1, 3, 5, and 10 ms. It can be found from the figure that there exists an ideal value of δ that minimizes

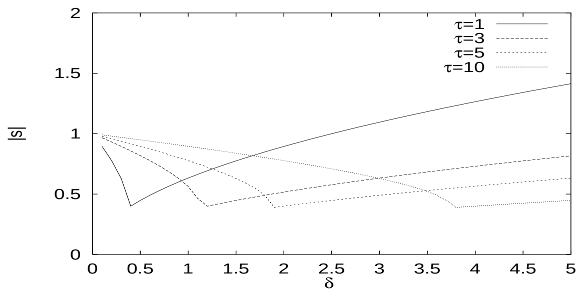


Figure 8: Relation between δ and $|s|$ ($B = 20$ packet/ms, $N = 10$, $\gamma = 3$ packet)

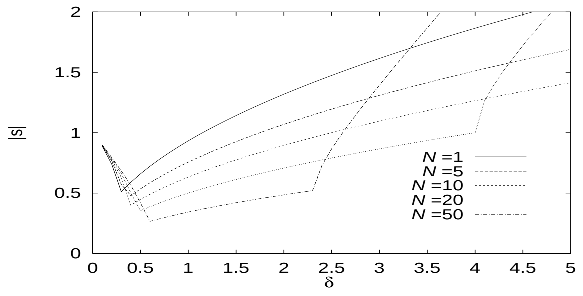


Figure 9: Relation between δ and $|s|$ ($B = 20$ packet/ms, $\tau = 1$ ms, $\gamma = 3$ packet)

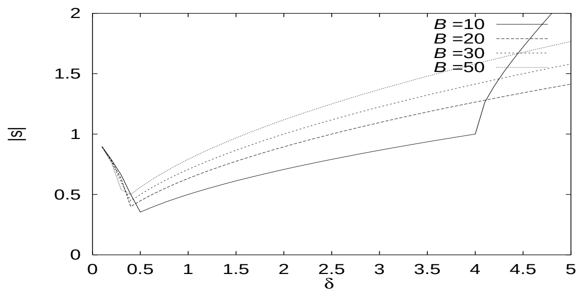


Figure 10: Relation between δ and $|s|$ ($N = 10$, $\tau = 1$ ms, $\gamma = 3$)

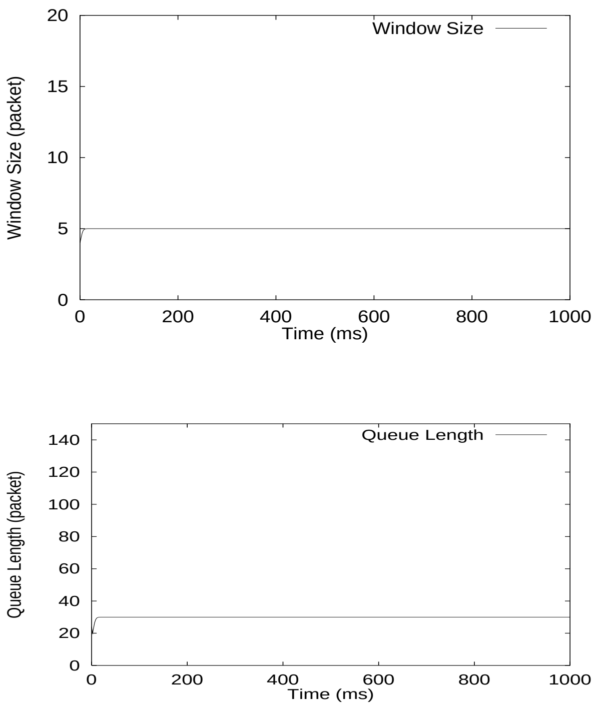


Figure 11: Case of appropriate δ ($B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\gamma = 3$ packet, $\delta = 0.4$)

$|s|$ for given τ, but it can also be found that the ideal value of δ changes considerably as τ changes. For example, the optimal value can be numerically computed as $\delta = 0.397$ for $\tau = 1$ ms, while $\delta = 1.877$ for $\tau = 5$ ms. Figure 9 shows relation between δ and $|s|$ for different numbers of connections, i.e., N = 1, 5, 10, 20, and 50. In addition, Fig. 10 shows relation between δ and $|s|$ for different values of the router's bandwidth, i.e., B = 10, 20, 30, and 50 packet/ms. These figures indicates the optimal value of δ is not significantly affected by the number of connections or the router's bandwidth.

In Figs. 11 and 12, we demonstrate how system performance is improved by setting δ appropriately. Figure 11 corresponds to Fig. 4 except that δ is changed from 2.4 to 0.4. Similarly, Fig. 12 corresponds to Fig. 6 except δ being 1.9 not 1.2. Note that the stability condition is satisfied in every case. These figures clearly exhibit importance of choosing δ appropriately; that is, transient performance of the system is dramatically improved while preserving system stability. Comparison between Figs. 6 and 12 shows a major reduction in the maximum number of packets (i.e., from 250 packets to 150 packets). Moreover, Fig. 6 shows the system never reaches the fixed point within 1000 ms, but the system is stabilized just in 50 ms by appropriate δ as shown in Fig. 12.

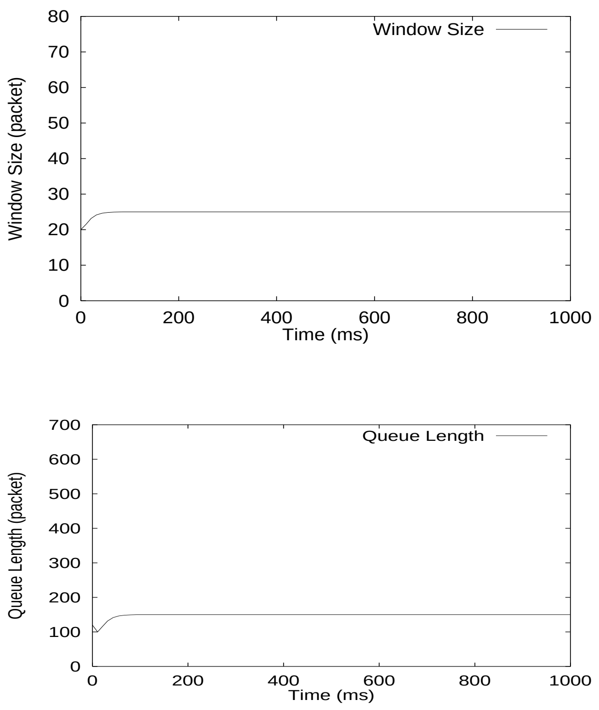


Figure 12: Case of appropriate δ ($B = 20$ packet/ms, $N = 10$, $\tau = 5$ ms, $\gamma = 3$ packet, $\delta = 1.9$)

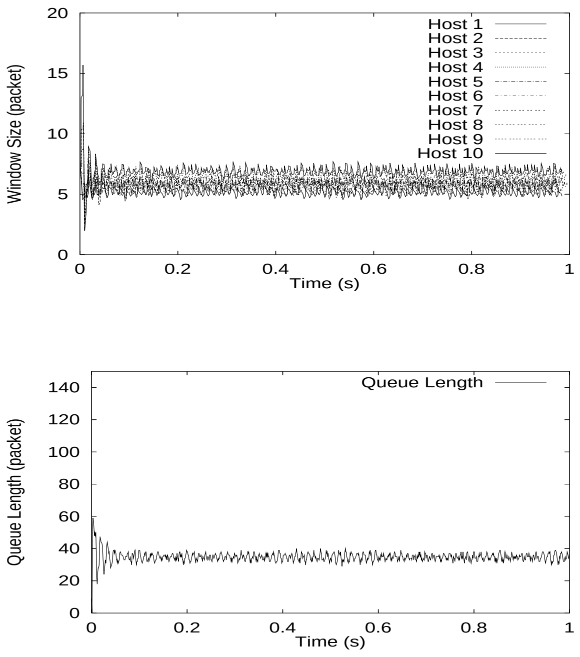


Figure 13: Stable behavior ($\delta = 2.0$, $B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\gamma = 3$ packet).

5 Simulation

In this section, we present several simulation results to validate our analysis. The simulation model is same with the analytic model shown in Fig. 1. We have implemented the window-based flow control mechanism based on TCP Vegas on the ns (Network Simulator) package [16]. The packet size is fixed at 1,000 bytes, and the control parameter δ is changed; 0.4 (optimal), 2.0 (stable), and 3.0 (unstable). For other control parameters, those of the numerical example in Fig. 4 are used: i.e., the router's bandwidth B is set to 20 packet/ms, the number of connections N is 10, the propagation delay τ is 1 ms, and the control parameter γ is 3 packet.

In Figs. 13 and 14, we first show simulation results for $\delta = 2.0$ and $\delta = 3.0$, each of which presents stable and unstable behavior of the window-based flow control mechanism based on TCP Vegas. These figures show dynamical behaviors of the window size of each connection and the number of packets in the router's buffer. One can find that Fig. 13 (stable behavior) shows the slight oscillation of both the window size and the number of packets in the router's buffer. This is caused by the disturbance in measurement of the round-trip time (e.g., variation in processing delays at both TCP and IP layers and timer granularity of the source host). Thus, a smaller value of δ

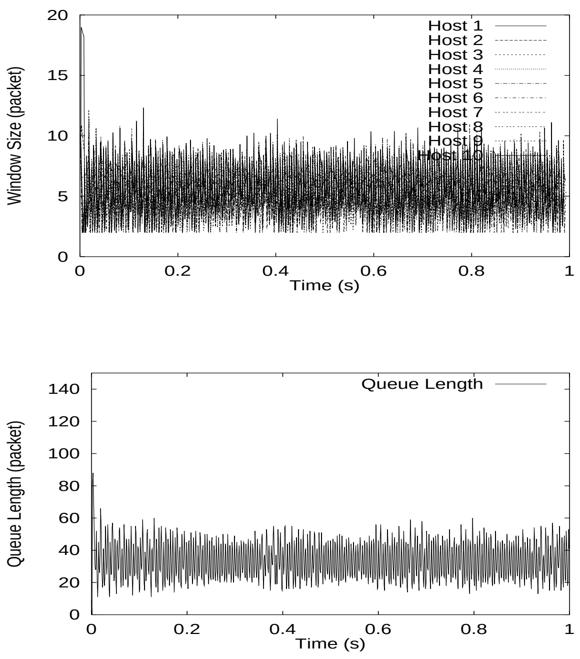


Figure 14: Unstable behavior ($\delta = 3.0$, $B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\gamma = 3$ packet).

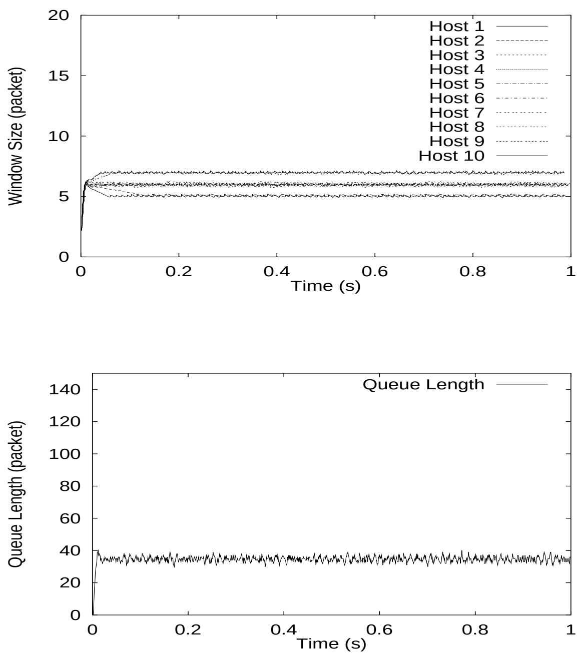


Figure 15: Case of appropriate δ ($\delta = 0.4$, $B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\gamma = 3$ packet).

would be appropriate for achieving a better stability. Figure 14 (unstable behavior) shows the drastic oscillation of the window size and the number of packets in the buffer. These figures suggest the validity of our stability analysis presented in Section 3.

We next show the case of an appropriate value of δ in Fig. 15, where δ is set to 0.4 as in Fig. 11. This figure shows the more stable operation than Fig. 13. However, it should be noted that fairness among connections is not fully satisfied. This problem is caused by a drawback of the window-based flow control mechanism of TCP Vegas [17]. It is incapability to measure the propagation delay exactly. The congestion avoidance mechanism of TCP Vegas relies on the assumption that the propagation delay (i.e., the round-trip time without any queueing delay at the router) is known in advance. This assumption is invalid if a router has a shared output buffer for all connections, or if the offered traffic load is not very low. Therefore, the inaccurate measurement of the propagation delay causes unfairness among connections in most real networks. Refer to [17] for more detailed discussion on this topic and a solution using ECN (Explicit Congestion Notification) mechanism.

6 Discussion on Applicability

We presented a control theoretic approach to the window-based congestion control mechanism. However, our approach has several limitations as well as those of previous researches. That is, the analysis itself relies on several assumptions such as an existence of the single bottleneck router, and identical propagation delays/initial window sizes of all connections. Also, the active number of connections should be known a priori. Once those conditions are satisfied, our approach exhibits the excellent performance. However, we cannot expect it in the actual situation.

In real networks, the number of active connections and the actual propagation delay would change as time changes. It sometimes causes inappropriate settings of δ even if δ was once configured properly. Figure 8 showed that the ideal value of δ mostly depends on the propagation delay. It is therefore possible to avoid degradation in transient performance to a certain degree by determining δ according to the largest propagation delay predicted in advance. However, if δ is chosen in this way, the system might be in an unstable operation as the propagation delay unexpectedly becomes small.

One conservative approach is to choose δ for the smallest propagation delay. In this case, the system never becomes unstable even when the propagation delay is heavily changed (see Fig. 8). However, the transient performance is considerably degraded as the propagation delay increases. In the next section, we therefore propose an adaptive control of δ by keeping track of changes in network status, and show that such an adaptive control dramatically improves system performance.

7 Adaptive Control of δ

Our stability analysis in Section 3 and investigation on transient behavior in Section 4 have shown that system performance is significantly affected by the control parameter δ. For an efficient operation of the system, δ must be chosen according to Eqs. (10)–(12) and Eq. (13). Application of these equations, however, requires knowledge on system parameters such as the number of active connections and the propagation delay, which are rarely known a priori in real networks. For instance, the number of active connections varies as time goes, and the propagation delay would be changed as packet routing changes. Hence, it would be necessary to adapt the control parameter δ as the network status changes. Note that another control parameter γ should be chosen to prevent buffer overflow and underflow since γ is directly related to the queue length at the router, which will be extensively discussed in Section 8.

In this section, we focus on the adaptive control of δ, which determines the amount of increase/decrease of the window size in a round-trip time. In what follows, we propose an adaptive control mechanism of the control parameter by keeping track of changes in the network status, followed by its performance evaluation. Our objective here is to improve the window-based congestion control mechanism based on TCP Vegas with the least modification. An adaptive control theory could be used for dynamically adjusting δ, but it beyonds the scope of this papper. The key idea of the mechanism is to change δ at the source host as network status changes. From the stability condition (Eqs. (10)–(12)) and the ideal value of δ (Eq. (13)), it is shown that δ must be determined from B (the bandwidth of the router), N (the number of active connections), τ (the propagation delay), and γ (control parameter). The source host already knows γ and τ; γ is a control parameter, and τ is always maintained for its congestion avoidance mechanism in TCP Vegas. Therefore, the bandwidth of the router B and the number of connections N should be estimated to compute the ideal value of δ at the source host. One approach to estimate the bandwidth of the bottleneck router at the source host is to use a “packet pair” method [1, 7, 18]. In this method, two adjacent packets are used to probe the bottleneck bandwidth. More specifically, the source host sends two packets into the network leaving no space between them. When ACK packets for them are returned, it then measures time duration between these two ACK packets. Since the minimum duration of two packets is determined by the bandwidth of the bottleneck router, the packet pair method conjectures the bottleneck bandwidth from this duration as noted earlier. The packet pair method is applicable to our congestion control mechanism. It is known that the packet pair method suffers from a certain amount of estimation errors when the bottleneck router adopts other than fair queueing discipline, or when the network topology is asymmetric [19]. However, since the ideal value of δ is not sensitive to the router’s bandwidth as shown in Fig. 10, rough estimation of the router’s bandwidth is sufficient for the adaptive control of δ.

The number of active connections N can be estimated from the window size $w_n(k)$ if the system is in steady state. That is, the relation between the window size in steady state and the number of connections is given by Eq. (5). The estimated number of connections N^* is therefore determined as

$$N^* = \max(\frac{B^*\,\tau}{w^* - \gamma\tau}, 1) \tag{14}$$

where B^* is the estimated bandwidth of the bottleneck router.

Our adaptive control mechanism recomputes the ideal value of δ based on these estimated values (i.e., the number of connections N^* and the bandwidth of the router B^*) and our analytic results. More specifically, δ is recomputed from the stability condition (Eqs. (10)–(12)) and the transient behavior (Eq.(13)) as soon as either the bandwidth of the router B^*, the number of connections N^*, or the propagation delay τ, changes. For obtaining the ideal value of δ, Eq. (13) must be minimized while satisfying Eqs. (10)–(12). This can be numerically computed by, for example, Brent’s method [20]. However, such computation is rather expensive for the source host to perform in real-time. Thus, frequency of parameter recomputation may be limited by the processing power of the source host. As we will show later, the performance is greatly improved by applying our adaptive control mechanism particularly when the propagation delay is changed. Therefore, for practical use, the control parameter δ could be updated only for change in the propagation delay.

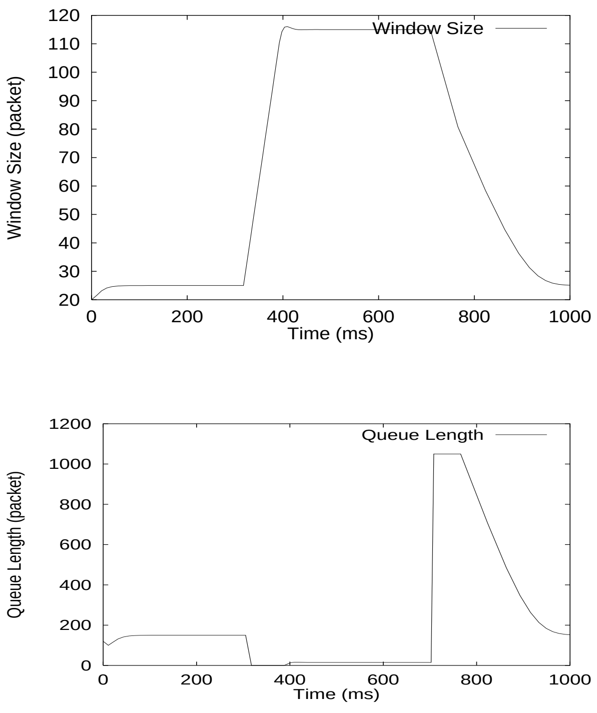


Figure 16: Effect of number of connections variation ($B = 20$ packet/ms, $\tau = 5$ ms, $\gamma = 3$ packet, $\delta = 1.9$)

We first investigate the influence of the number of connections N. Note that the propagation delay τ is fixed. We assume that the bandwidth of the router B is known by source hosts. Figures 16 and 17 show dynamical behaviors of the window size and the number of packets in the router’s buffer for $\tau = 5$ ms when the adaptive control of δ is either not used and used, respectively. In these figures, system parameters are equivalent to those of Fig. 12, but the number of connections N is changed to 1 between $t = 300$ ms and $t = 700$ ms. In case of no adaptive control of δ (Fig. 16), we set $\delta = 1.9$ that is the ideal value for $N = 10$. In case of the adaptive control in Fig. 17, δ is numerically recomputed every 10 ms at the source host

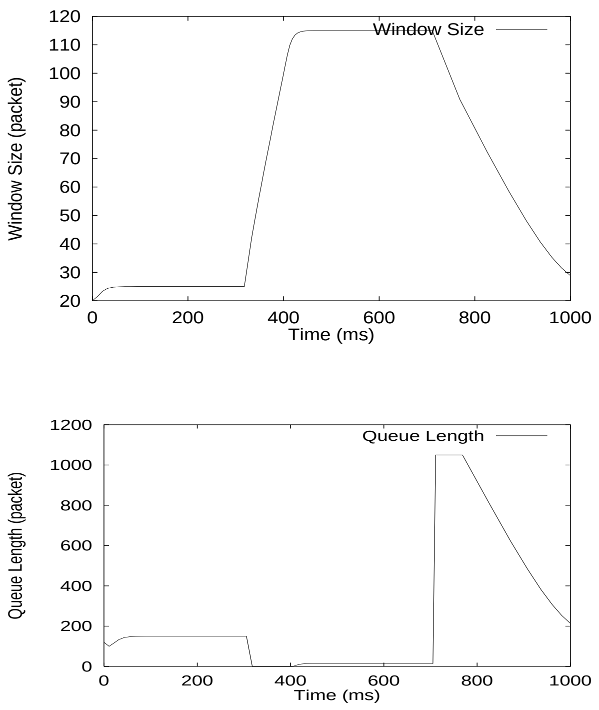


Figure 17: Effect of number of connections variation with adaptive control ($B = 20$ packet/ms, $\tau = 5$ ms, $\gamma = 3$ packet)

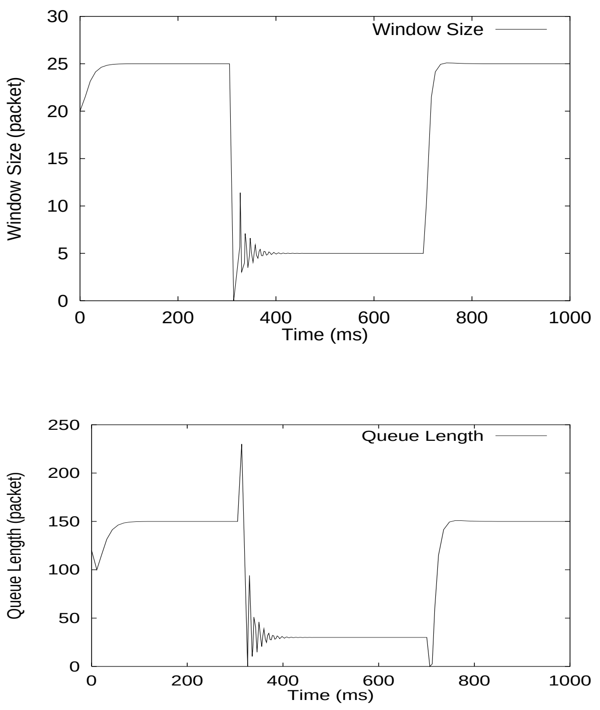


Figure 18: Effect of propagation delay variation ($B = 20$ packet/ms, $N = 10$, $\gamma = 3$ packet, $\delta = 1.9$)

based on the estimated number of active connections (see Eq. (14)). By comparing these figures, it can be found that the performance gain resulted from the adaptive control of δ is minor. In other words, the dynamical behavior of the system is almost identical even with the adaptive control of δ. This is because the ideal value of δ is not greatly affected by change in the number of connections as seen in Fig. 9.

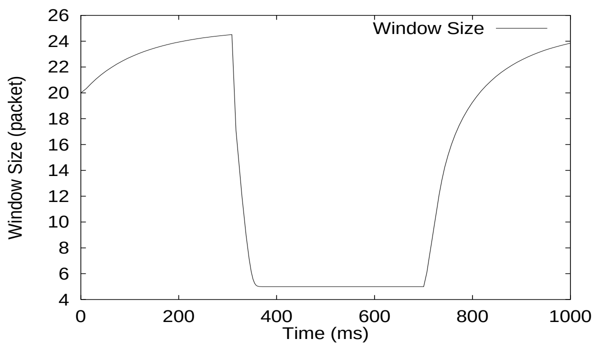
When the propagation delay changes, however, system performance is heavily degraded without our adaptive control of δ. Figure 18 shows this case, where all parameters are identical with those of Fig. 12, but the propagation delay is changed to $\tau = 1$ ms during $t = 300$–700 ms. This figure indicates that when the propagation delay is changed at $t = 300$ ms, it takes some time for the system to become stabilized (about 100 ms in this case). This is because $\delta = 1.9$ is the ideal value for $\tau = 5$ ms, but not for $\tau = 1$ ms. Namely, $\delta = 1.9$ is inappropriate for $\tau = 1$ ms so that transient performance is not good. On the other hand, when the propagation delay is changed to $\tau = 5$ ms again at $t = 700$ ms, the system immediately converges to the fixed point. It is because $\delta = 1.9$ is the ideal value for $\tau = 5$ ms.

We next show the case where δ is configured according to the case of 1 ms propagation delay. In Fig. 19, only δ is changed to 0.4 from Fig. 18. It can be found from this figure that transient performance is severely degraded when the propagation delay is changed to 5 ms at $t = 700$ ms. This phenomenon is apparent by comparing with Fig. 18. This can be explained as follows. The window-based flow control mechanism changes its window size every one round-trip time. Thus, the window size is changed more slowly as the propagation delay becomes large. If the propagation delay is large and the control parameter δ is inappropriate, convergence speed is quite slow so that transient performance is severely degraded.

Finally, we show how transient performance can be improved with the adaptive control of δ in Fig. 20. One can find from the figure that by adaptively recomputing δ, almost ideal performance can be obtained regardless of the values of the propagation delay. For instance, the system is stabilized within 50 ms when the propagation delay is changed from 5 ms to 1 ms at $t = 300$ ms. Also the system quickly converges to the fixed point when the propagation delay is changed to 5 ms at $t = 700$ ms.

8 Discussion on Buffer Dimensioning

The router's buffer is one of the most important resources



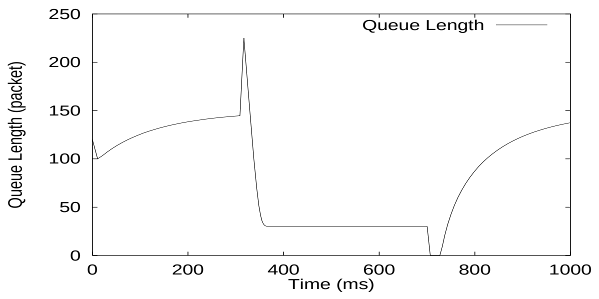
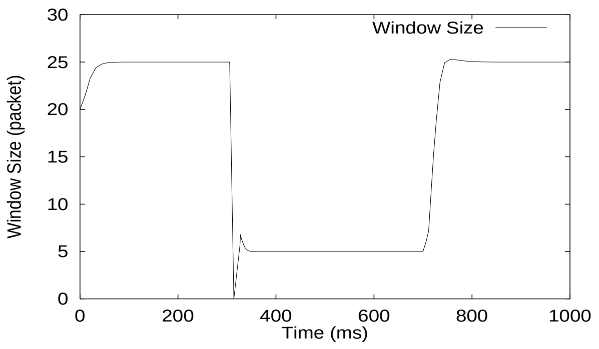


Figure 19: Effect of propagation delay variation ($B = 20$ packet/ms, $N = 10$, $\gamma = 3$ packet, $\delta = 0.4$)

in the network. It is desirable to work the router with a small buffer without compromising its performance. If the buffer size is too large, not all the buffer is utilized. On the contrary, if the buffer size is too small, many packets overflow the router's buffer, leading to significant performance degradation. The main objective of this section is therefore to dimension the buffer size required at the router, which achieves reasonable performance in terms of packet loss probability and link utilization. The queue length (i.e., the number of waiting packets in the buffer) at the router is mainly dependent on the control parameter γ since the objective of γ is to determine how many in-flight packets are allowed in a round-trip time as indicated by Eq. (7). Thus, γ affects the buffer occupancy in steady state as shown in Eq. (6). It is therefore important to properly choose γ to achieve high link utilization of the outgoing link.

We first discuss an appropriate value of γ for achieving high link utilization. For this purpose, we derive the minimum number of packets in the router's buffer. When the number of connections is decreased from N to N' ($N' < N$), the number of packets in the buffer takes its minimum immediately after N is changed (see, e.g., around $t = 300$ in Fig. 16). The minimum number of packets in the router's buffer q_{min} is obtained from



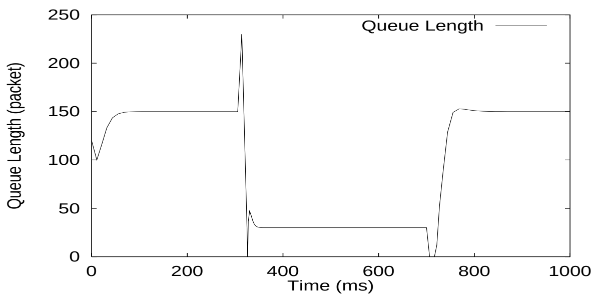


Figure 20: Effect of propagation delay variation with adaptive control ($B = 20$ packet/ms, $N = 10$, $\gamma = 3$ packet)

Eq. (17) as

$$q_{min} = \min\left\{\max\left[N'\left(\tau\frac{B+\gamma N}{N}\right) - B\tau, 0\right], L\right\} \quad (15)$$

As Eq. (15) indicates, the minimum number of packets queued in the buffer linearly increases for γ. To achieve full link utilization, the minimum queue length q_{min} should be greater than zero. Consequently, a condition for achieving full link utilization is given from Eq. (15) by the following inequality.

$$\gamma > B(\frac{1}{N'} - \frac{1}{N}) \quad (16)$$

We next focus on the maximum number of packets queued in the router's buffer. The number of packets in the router's buffer increases when the number of connections is increased (see, e.g., around $t = 700$ ms in Fig. 16). In particular, it takes its maximum immediately after the number of connections is changed. Let us consider a case where the number of connections is suddenly increased from N to N' ($N' > N$) in steady state. Let q_{max} be the maximum number of packets in this case. Using Eqs. (2), (4) and (6), the maximum number of packets in the router's buffer is given by

$$q_{max} = \min\left\{\max\left[N'\left(\tau\frac{B+\gamma N}{N}\right) - B\tau, 0\right], L\right\} \quad (17)$$

From Eq. (17), it can be easily observed that the maximum queue length q_{max} is increased linearly for the propagation delay τ and the number of connections N'. To prevent buffer overflow at the bottleneck router, the control parameter γ should be chosen such that Eq. (17) is less than the buffer size L. In other words, Eq. (17) suggests the required capacity for the buffer. Note that Eq. (17) is for the worst-case; that is, all connections are assumed to behave identically and synchronously. So the maximum number of packets obtained in Eq. (17) is the upper-bound of the actual one. Also note that the queue length of the router increases when the propagation delay is changed as in Fig. 18. The maximum queue length in this case can be easily obtained with the same approach although results are not included in this paper due space limitation.

Combining these observations, we summarize buffer dimensioning of the bottleneck router for preventing packet losses while achieving full link utilization. That is, the buffer size of the bottleneck router L should satisfy

$$L > q_{max}$$

where q_{max} is a function of γ as shown in Eq. (17), and γ should also satisfy Eq. (16).

In Figs. 21 and 22, we show dynamical behaviors of the number of packets in the router's buffer for cases of appropriate and inappropriate settings of γ, respectively. These figures use the following parameters: the router's bandwidth $B = 20$ packet/ms, the propagation delay $\tau = 5$, and the control parameter $\delta = 1.9$. The number of connections N is changed to 10 ($t < 300$), 15 ($300 \leq t < 500$), 5 ($500 \leq t < 700$), and 10 ($t > 700$). From Eq. (17), the control parameter γ should be greater than 2.67 for preventing buffer underflow in this case. We therefore set $\gamma = 3$ (such that Eq. (16) is satisfied) in Fig. 21 and $\gamma = 2$ (i.e., Eq. (16) is not satisfied) in Fig. 22. One can find from these figures that the number of packets in the router's buffer suddenly drops when the number of connections is decreased at $t = 500$ ms. It can also be found that the router's buffer in Fig. 22 becomes empty, which results in low throughput. Using Eq. (17), the maximum number of packets in the router's buffer is 275 packets for the case of $\gamma = 3$. Figure 21 shows a good agreement with this value. Note that the average number of packets in Fig. 21 is much larger than that in Fig. 22. This implies that the packet transmission delay becomes large as γ increases. In other words, there is a trade-off between throughput and packet transmission delay.

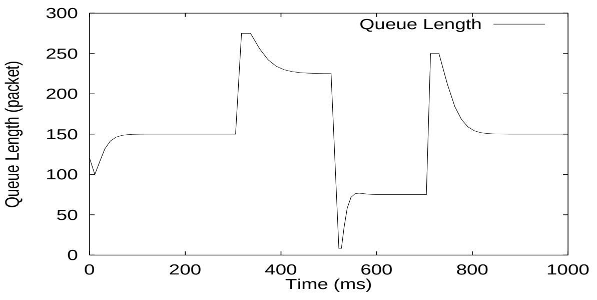


Figure 21: Case of appropriate γ ($B = 20$ packet/ms, $\tau = 5$ ms, $\gamma = 3$ packet, $\delta = 1.9$

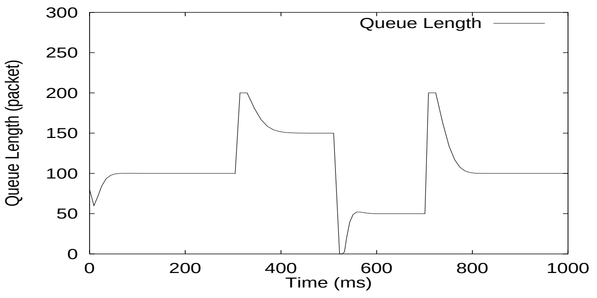


Figure 22: Case of inappropriate γ ($B = 20$ packet/ms, $\tau = 5$ ms, $\gamma = 3$ packet, $\delta = 1.9$

9 Conclusion

In this paper, we have focused on a window-based flow control mechanism based on the congestion avoidance mechanism of TCP Vegas, and have analyzed its stability and transient performance using a control theoretic approach. We have first derived the stability condition, which guarantees convergence of the window size and the number of packets in the router's buffer. Our stability analysis has quantitatively shown that stability region of the system's feedback gain — determining the amount of increase/decrease of the window size — depends on the number of connections and the propagation delay. We have next focused on transient behavior of the system, and derived the ideal value of the feedback gain that minimizes convergence time to the fixed point. These analyses suggest that control parameters such as a feedback gain must be chosen carefully. In particular, fluctuations of the number of active connections and the propagation delay must be considered. We have proposed an adaptive control of the control parameter by keeping track of changes in the network, and have shown that its effectiveness through numerical examples. Our analytic results gave much insight on the congestion control mechanism of TCP. For instance, we have investigated the buffer dimensioning problem of the bottleneck router's buffer. It allows us to choose an appropriate buffer size of the bottleneck router for avoiding buffer overflow and also for preventing link under-utilization.

As a future work, we should extend our analysis to more realistic networks. For example, interference between different types of traffic should be considered. Since both of TCP and UDP traffic co-exist in real TCP/IP networks, performance of a congestion control mechanism for TCP traffic must be affected by existence of UDP traffic. In the current paper, we have assumed that all connections have identical propagation delays and identical initial window sizes. In real TCP/IP networks, propagation delays of all connections are not identical, so that dynamics of their window sizes are not synchronized. In [21], we have extended our analysis to more generic network configurations, where every connection is allowed to have a different propagation delay. The result there was that the analytic results in this paper is still applicable to heterogeneous networks. Also note that the analytic result in [21] is rather complex, so that it is difficult to perform such computation at the source host in real-time. For practical use, the analytic result in this paper is more suitable.

References

[1] V. Jacobson, "Congestion avdoidance and control," in *Proceedings of SIGCOMM '88*, pp. 314–329, August 1988.

[2] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*. New York: Addison-Wesley, 1994.

[3] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," in *Proceedings of ACM SIGCOMM '94*, pp. 24–35, October 1994.

[4] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End to end congestion avoidance on a global Internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 1465–1480, October 1995.

[5] J. S. Ahn, P. B. Danzig, Z. Liu, and L. Yan, "Evaluation with TCP Vegas: emulation and experiment," *ACM SIGCOMM Computer Communication Review*, vol. 25, pp. 185–195, August 1995.

[6] G. Hasegawa, M. Murata, and H. Miyahara, "Fairness and stability of congestion control mechanism of TCP," in *Proceedings of 11th ITC Special Seminar*, pp. 255–262, October 1998.

[7] S. Keshav, "A control-theoretic approach to flow control," in *Proceedings of ACM SIGCOMM '91*, pp. 3–15, September 1991.

[8] H. Zhang and O. W. Yang, "Design of robust congestion controllers for ATM networks," in *Proceedings of IEEE INFOCOM '97*, pp. 302–309, April 1997.

[9] C. E. Rohrs, R. A. Berry, and S. J. O'Halek, "A control engineer's look at ATM congestion avoidance," in *Proceedings of IEEE GLOBECOM '95*, pp. 1089–1094, 1995.

[10] S. Fuhrmann, Y. Kogan, and R. A. Milito, "An adaptive autonomous network congestion controller," in *Proceedings of IEEE CDC '96*, p. WA12 11:40, December 1996.

[11] C. E. Rohrs, "A linear control approach to explicit rate feedback in ATM networks," in *Proceedings of IEEE INFOCOM'97*, vol. 1, pp. 277–282, April 1997.

[12] A. Kolarov and G. Ramamurthy, "A control theoretic approach to the design of closed-loop rate based flow control for high speed ATM networks," in *Proceedings of IEEE INFOCOM '97*, pp. 293–301, April 1997.

[13] Y. Zhao, S. Q. Li, and S. Sigarto, "A linear dynamic model for design of stable explicit-rate ABR control," in *Proceedings of IEEE INFOCOM '97*, pp. 283–292, April 1997.

[14] B.-K. Kim and C. Thompson, "Optimal feedback control of ABR traffic in ATM networks," in *Proceedings of IEEE GLOBECOM '98*, pp. 844–848, 1998.

[15] R. Isermann, *Digital control systems, Volume 1: fundamentails, deterministic control*. Springer-Verlag Berlin Heidelberg, 1989.

[16] "LBNL network simulator (ns)." available at `http://www-nrg.ee.lbl.gov/ns/`.

[17] H. Ohsaki, M. Murata, T. Ushio, and H. Miyahara, "A control theoretical approach to a window-based flow control mechanism with explicit congestion notification," *38th IEEE Conference on Decision and Control*, Dec. 1999.

[18] R. Carter and M. Crovella, "Measuring bottleneck link speed in packet-switched networks," *Technical Report BU-CS-96-006, Computer Science Department, Boston University*, March 1996.

[19] V. Paxon, "Measurements and analysis of end-to-end Internet dynamics," *Ph. D. Thesis, UCB/CSD-97-945, Computer Science Division, University of California, Berkeley*, April 1997.

[20] R. P. Brent, *Algorithms for Minimization without Derivatives*. Englewood Cliffs, N.J.: Prentice-Hall, 1973.

[21] K. Takagaki, H. Ohsaki, and M. Murata, "Stability analysis of a window-based flow control mechanism for TCP connections with different propagation delays," in *Proceedings of INET 2000: The Internet Global Summit*, July 2000.