Application of control theory to a window-based flow control mechanism with ECN routers

Hiroyuki Ohsaki, Masayuki Murata, 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, use of an ECN (Explicit Congestion Notification) mechanism as congestion indication from the network to source hosts has been actively discussed in the IETF (Internet Engineering Task Force). In this paper, we focus on a window-based flow control mechanism, which cooperates with routers supporting the ECN mechanism. The first part of this paper discusses how the ECN mechanism can be incorporated into the TCP/IP network when all source hosts respond to ECN messages. The second part of this paper gives a control theoretic approach to the window-based flow control mechanism, which cooperates with ECN routers. We derive a stability condition of the window-based flow control mechanism, and show that system stability is affected by the router's buffer size as well as the bandwidth of the bottleneck router. We also show that the number of TCP connections is unrelated to the system stability. We further design a regulator for the window-based flow control mechanism, which utilizes the current window size and the estimated number of packets at the router's buffer as a feedback input. We show that the transient performance is significantly improved by applying the regulator. Several practical issues are also discussed.

1 Introduction

In a packet-switched network, a feedback-based congestion control mechanism is essential for providing efficient data transfer services. The current Internet uses a window-based flow control mechanism in the TCP (Transmission Control Protocol), as the feedback-based congestion control mechanism. For example, a version of the TCP mechanism called *TCP Reno* uses packet loss in the network as feedback information since packet loss implies congestion occurrence in the network [1, 2]. Until packet loss occurs in the network, TCP Reno gradually increases its window size. As the window size is over its available bandwidth, excess packets are queued at the buffer of the bottleneck router for some period. If the window size increases further, packets at the buffer of the router overflows, leading to packet loss. The source host detects occurrence of packet loss in the network from, for example, its timeout mechanism, and reduces its window size to one. TCP Reno has another mechanism called *fast retransmit* to detect packet loss, 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. Since the congestion control mechanism of TCP Reno relies on packet loss in the network, packet loss cannot be prevented. It is necessary expenses for TCP Reno to work correctly since the congestion control mechanism of TCP Reno only utilizes information on *occurrence* of packet loss. However, it is desirable for the congestion control mechanism to prevent packet losses in the network.

Accordingly, the use of an ECN (Explicit Congestion Notification) mechanism has been actively discussed in the IETF (Intermediate Engineering Task Force). ECN is a mechanism to explicitly notify source hosts of congestion occurrence in the network. The ECN mechanism can be implemented in TCP/IP networks in several ways [3]. In [4], *ICMP Source Quench message* is defined for conveying congestion information from the congested router to source hosts. One-bit use of the DS-byte in the differentiated service architecture has been proposed in [5]. According to [5], an example implementation of the ECN mechanism in TCP/IP networks is as follows. One-bit in the header of the data packet is reserved for the ECN bit. The router in the network uses the ECN bit for notifying source hosts of its incipient congestion. The router computes the average number of packets in the buffer. If it exceeds a threshold value (e.g., p % of the buffer capacity), the router sets the ECN bits of all arriving packets.

This information is then carried to source hosts via corresponding destination hosts by the ACK packet with the ECN bit set. The source host responds to the ECN message by, for example, reducing its window size as in the case of packet loss [3]. The advantage of the ECN mechanism is that unnecessary packet loss can be prevented if source hosts respond to the ECN message appropriately. In [3], it has been reported that the ECN mechanism can avoid unnecessary packet delays for low-bandwidth and delay-sensitive TCP connections. It has also been reported that another advantage of the ECN mechanism is that the source host can detect congestion rapidly regardless of coarse granularity of the TCP's timer.

In [6], Ramkrishnan *et al.* have proposed a more feasible application of the ECN mechanism to TCP/IP networks. The basic concept of their proposal is that the congestion control mechanism of TCP should respond to the ECN message as the same manner to packet loss. One reason for this is to allow incremental deployment of the ECN mechanism in both the source host and the router. If TCP Reno's response to receipt of the ECN message is different from that of packet loss, it causes unfairness between ECN-capable and non-ECN-capable connections. They have also proposed that the source host responds to the ECN message at most once per round-trip time. This idea comes from the fact that a single packet loss is sufficient for the congestion control mechanism of TCP Reno to throttle its window size. However, this is not the case for other versions of TCP such as TCP Vegas [7, 8].

In [9], the authors have proposed an application of the ECN mechanism to TCP Vegas' congestion control mechanism to solve several drawbacks of TCP Vegas. The fundamental idea of the proposed mechanism is to use the ECN message only when the congestion control mechanism of TCP Vegas fails controlling congestion in th network. Namely, the router sends the ECN message to source hosts only when the TCP Vegas' congestion control mechanism cannot control congestion in the network by itself. Through simulation experiments, the authors have shown that the use of the ECN mechanism is helpful for improving fairness among TCP connections. In this paper, we proceed one step further to more actively use the ECN message in the context of a window-based flow control paradigm. Namely, the ECN message can inform source hosts of more detailed information on congestion of the network and the source host can take a proper action against the congestion level of the network by the ECN message. In other words, we propose and demonstrate that if the congestion control mechanism of TCP cooperates with the router's ECN setting mechanism, and if the router informs source hosts of its congestion status more accurately, a more efficient congestion control mechanism is realized.

Organization of this paper is as follows. In Section 2, we discuss how the ECN mechanism is used for realizing an efficient window-based flow control mechanism, and propose a window-based flow control mechanism, which cooperates with the router supporting the ECN mechanism. In Section 3, the stability analysis of the window-based flow control mechanism is performed, and the relation between control parameters and the system stability is investigated. In Section 4, we design the regulator that improves transient performance by applying control theory. Finally we conclude this paper and discuss future works in Section 5.

2 A Window-Based Flow Control Mechanism and the ECN Mechanism

This section discusses how the ECN mechanism is incorporated into a window-based flow control mechanism; that is, how the ECN message is used effectively as feedback information from the network. We first explain principles of the window-based flow control mechanism cooperating the router generating the ECN message. We then propose a window-based flow control mechanism, which consists of two parts: (1) the window-based flow control mechanism at the source host and (2) the ECN setting algorithm at the router.

2.1 Principles

The fundamental idea of a window-based flow control mechanism actively utilizing the ECN mechanism should be that both of the source host and the router cooperate as a single mechanism. In the current TCP/IP networks, the congestion control mechanism of TCP assumes nothing about the router's operation. It is because neither packet scheduling discipline (e.g., FIFO (First-In First-Out) or fair queueing) nor packet discarding algorithm (e.g., drop-tail or RED (Random Early Detection)) is known by the source host in real networks. The congestion control mechanism of TCP was designed to work without any knowledge on the router's algorithm. Actually, separation of the TCP's congestion control mechanism from the router's algorithm is desirable when several types of congestion control mechanisms and router's algorithms coexist in the network as in the current Internet.

However, such a generality of the congestion control mechanism of TCP significantly limits the network performance. For designing a truly efficient congestion control mechanism, both the source host and the router should be designed simultaneously. It is also desirable to split functionality of a window-based flow control mechanism into two parts: (1) detection of the congestion by the router, and (2) control of the congestion by the source host. The router is the best place to detect congestion in the network since congestion does occur at the router. Similarly, it is

natural for the source host to control its traffic flow because no other than the source host causes congestion.

In this paper, we assume that the router is equipped with a single FIFO buffer shared by all TCP connections, and performs no per-connection accounting. We also assume that only one-bit information is used as the ECN message. These assumptions are for implementation simplicity, and for investigating the possibility of the ECN mechanism in such a simple network environment.

We first consider a desirable functionality of the router, which is responsible for setting the ECN bit of the arriving packet. Since the congestion control is performed at the source host, the role of the router should be simple; it should send its congestion information to all source hosts as accurate as possible. Namely, the router should do nothing other than conveying its congestion status to source hosts. The router uses the ECN bit of the packet header to carry congestion information to source hosts. Since the ECN message is only one-bit information, it can have only two meanings; that is, the network is *congested*, or *not congested*.

One algorithm for the router to set the ECN bit of the arriving packet is to use a single threshold value, as explained in Section 1. Namely, if the number of packets in the buffer exceeds the threshold value, ECN bits of all packets are marked. Otherwise, the router does not change the ECN bit of arriving packets. This algorithm is easy to implement, and would work effectively when the propagation delay is negligible. However, as the propagation delay increases, window sizes of TCP connections and the number of packets in the router's buffer oscillate excessively, and the network performance is degraded.

One-bit information of the ECN message is apparently insufficient for the fine control of the network. It is possible that the router uses more bits to indicate the congestion status more accurately. However, it implies complicated processing at the router, which is not a desirable feature. We therefore propose to use a probabilistic number of ECN messages; that is, the router notifies degree of congestion by setting the ECN bit with a certain probability. More specifically, the router sets the ECN bit in the header of the arriving packet with a probability, which is proportional to the current number of packets waiting in its buffer. Otherwise, it does not change the ECN bit of the arriving packet. When each router independently sets the ECN bit of the arriving packet, the router sets the ECN bit of the arriving packet with a certain probability, regardless of the former status of the ECN bit. Namely, the ECN bit of the packet is OR-ed at all congested routers. It is however uncommon that several routers are congested at the same time. And if several routers would get congested, the ratio of ECN messages were higher than the case of a single congested router. This means that the source host receives more conservative feedback from the network, so that safer operation can be expected.

We next consider a desirable functionality of the source host, which is responsible for controlling its traffic flow. The source host, being located at the edge of the network, has to throttle its window size once the network falls into congestion. On the contrary, the source host should increase its window size when network resources are not fully utilized. The desirable operating point of the congestion control mechanism is therefore that the network is always lightly but not heavily congested. The control objective of the window-based flow control mechanism should be to stabilize the number of packets in the buffer of the bottleneck router to a certain level. If the number of packets in the bottleneck router is greater than zero, it implies full utilization of the bottleneck bandwidth. If the number of packets in the bottleneck router is below its buffer size, it means that packet loss is not likely to happen.

The above-mentioned control objective is inspired by that of TCP Vegas [7, 8]. However, it substantially differs in the following point. The control objective of TCP Vegas' congestion control mechanism is to stabilize the number of packets from *each* connection at the bottleneck router's buffer. Namely, TCP Vegas allows every connection to have a several extra packets in the network. So the total number of packets at the router's buffer is proportional to the number of TCP connections. In [9], the authors have shown that the control objective of TCP Vegas' congestion control mechanism causes a scalability problem as the number of connection increases. On the other hand, the control objective of our congestion control mechanism is to stabilize the total number of packets at the bottleneck router's buffer, which avoids such a scalability problem. That is, in steady state, the number of packets at the router's buffer is therefore independent of the number of connections.

2.2 Algorithm

Based on the above discussion, we propose a window-based flow control mechanism using the ECN mechanism. The router is equipped with a single FIFO (First-In First-Out) buffer, which is shared by all connections destined for the same output port. The router does not perform any per-connection accounting; It only maintains the total number of packets queued in the buffer. The ECN bit in the packet header is used to convey congestion information from the router to every source host via the corresponding destination host. The router's algorithm of setting the ECN bit in the packet header is similar to that of RED (Random Early Detection) router with the ECN marking [10]. Our algorithm is however simpler than the RED router.

We first describe the operation algorithm of the router. It sets the ECN bit of a portion of all arriving packets. The router has two control parameters T_{min} and T_{max} ($0 < T_{min} < T_{max}$). These parameters are lower- and upper-thresholds to calculate the probability for setting the ECN

bit of the arriving packet, being denoted by p_a, which is calculated as

$$p_a = \frac{q - T_{min}}{T_{max} - T_{min}}, \quad (1)$$

where q is the number of packets queued at the router's buffer. The router's algorithm described above is different from that of the RED router; in our algorithm, (1) ECN bits of almost all packets are marked when the number of packets is close to T_{max}, and (2) p_a is calculated from an instantaneous value of the current number of packets in the buffer. The RED router only marks the ECN bit for a small fragment of packets even when the number of packets is close to T_{max}. It is because the RED router's algorithm only targets the congestion control mechanism of TCP Reno. A single packet with its ECN bit set is sufficient for TCP Reno to respond to congestion in the network. When both the source host and the router cooperate, such an assumption is unnecessary. Additionally, the RED router maintains the average number of packets in the buffer, and calculates the marking probability based on this value. As we have discussed earlier, when the congestion control is performed at the source host, the router should notify all source hosts of its congestion information as accurate as possible. Hence, in our window-based flow control mechanism, the probability p_a is calculated from an instantaneous value of the queue length. It should be noted that averaging or filtering of feedback information can be performed at the source host, if necessary.

The source host adjusts its window size based on the feedback information returned as a series of ECN bits by the bottleneck router. Once per round-trip time, the source host calculates e which is a ratio of ECN messages: i.e., the ratio of the number of ACK packets with ECN bit set to the number of all received ACK packets. The source host counts the number of ACK packets with ECN bit set N_e and the total number of ACK packets N_a in a round-trip time. It then calculates the ratio of ECN messages r_e as

$$r_e = \frac{N_e}{N_a}, \qquad 0 \leq e \leq 1. \quad (2)$$

If e is close to 1, it implies that the network is heavily congested so that the window size should be reduced quickly. On the contrary, if e is close to 0, the network is not congested so that the window size should be increased. The control objective of our window-based congestion control mechanism is therefore to converge the observed ratio of ECN messages e to a control target ϵ ($0 \leq \epsilon \leq 1$). The algorithm of the source host to change its window size is described by

$$cwnd \leftarrow \max(cwnd + \delta \times cwnd \times (\epsilon - e), 1), \quad (3)$$

where δ is a control parameter, which determines the amount of increase/decrease of the window size per a round-trip time.

In the window-based flow control mechanism described above, the ratio of ECN messages e is computed every round-trip time. The number of ACK packets that the source host receives is limited by its window size. So the estimated value of e contains a quantization error; the granularity of e is directly determined by the current window size. Namely, the quantization error is the order of $1/cwnd$.

3 Analysis

In this section, we analyze the window-based flow control mechanism described in Section 2 by applying control theory. We explain an analytic model, followed by its stability analysis. We also show several numerical examples.

3.1 Analytic Model

Figure 1 depicts the analytic model. The number N of source hosts are connected to corresponding destination hosts through a single bottleneck router. The window-based flow control mechanism 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.

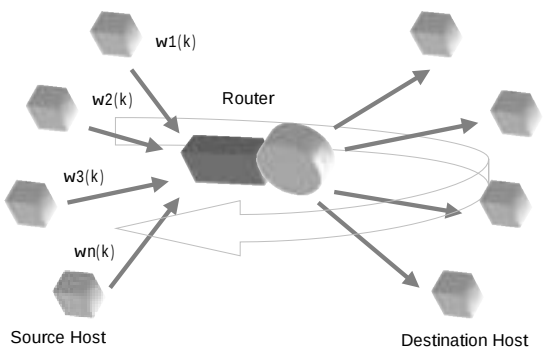


Figure 1: Analytic model.

Let $w_n(k)$ be the window size of the source host n ($1 \leq n \leq N$) at slot k. That is, 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. 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. The

round-trip delay (i.e., the sum of the source–destination delay and the destination–source delay) is denoted by τ, which includes all propagation delays and processing delays. Note that τ does not include a queueing time at the router.

During a round-trip time, the source host is allowed to 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.

$$q(k+1) = \max(\sum_{n=1}^{N} w_n(k) - B\, r(k), 0), \tag{4}$$

where $r(k)$ denotes the round-trip time at slot k, and is given by

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

Note that $r(k)$ corresponds to the length of the slot k.

Let $e_n(k)$ be the ratio of ECN messages observed by the source host n at slot k. Since $e_n(k)$ can be approximated by the router's probability of setting the ECN bit p_a, $e_n(k)$ is given by

$$e_n(k) \quad \simeq \quad \frac{q(k) - T_{min}}{T_{max} - T_{min}}. \tag{5}$$

The source host changes its window size based on the difference between the observed ratio of ECN messages $e_n(k)$ and the control target ϵ. From Eq. (3), the window size of the source host n at slot $k+1$, $w_n(k+1)$, is determined as

$$w_n(k+1) = \max\left(w_n(k) + \delta\, w_n(k)\,(\epsilon - e_n(k)), 1\right). \tag{6}$$

3.2 Stability Analysis

For simplicity, we assume that the initial window sizes of all source hosts are identical, and that all source hosts change their window sizes according to Eq. (6). The number of packets in the router's buffer at slot $k+1$, $q(k+1)$, is given by

$$q(k+1) \quad = \quad \max\left(N\, w(k) - B\, r(k), 0\right), \tag{7}$$

where $w(k) \equiv w_n(k)$, $1 \le n \le N$.

Let (w^*, q^*) be the fixed point of $(w(k), q(k))$. By using Eqs. (5)–(7), (w^*, q^*) is obtained as follows.

$$\left[\begin{array}{c} w^* \\ q^* \end{array}\right] = \left[\begin{array}{c} \left(2\epsilon(T_{max} - T_{min}) + 2T_{min} + B\,\tau\right)/N \\ \epsilon(T_{max} - T_{min}) + T_{min} \end{array}\right] \tag{8}$$

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

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

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

$$\mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) \tag{9}$$

where

$$\mathbf{A} \quad = \quad \left[\begin{array}{cc} 1 & -\frac{\delta\, w^*}{T_{max} - T_{min}} \\ N & -1 \end{array}\right].$$

In the system defined by Eqs. (5), (6), and (7), the fixed point (w^*, q^*) is locally asymptotically stable when all roots of the characteristic equation lie in the unit circle. Note that the characteristic equation is given by

$$D(s) \equiv |s\mathbf{I} - \mathbf{A}| = 0. \tag{10}$$

Since the characteristic equation $D(s)$ is quadratic, Eq. (10) is equivalent to the following inequalities [11].

$$D(1) > 0; \quad D(-1) > 0; \quad |D(0)| < 1$$

The fixed point of the system (w^*, q^*) is locally asymptotically stable if and only if the following inequalities (i.e., *stability condition*) hold.

$$0 < \frac{\delta\,\left\{2\,\epsilon\,(T_{max} - T_{min}) + 2T_{min} + B\,\tau\right\}}{T_{max} - T_{min}} < 2 \tag{11}$$

The stability condition given by Eq. (11) suggests a distinctive feature of the window-based flow control mechanism described in Section 2; stability of the system is independent of the number of connections N. In other words, the number of connections has no relation with system stability. This characteristic of the window-based flow control mechanism is desirable since the number of TCP connections usually varies according to time, and it is one of the most difficult system parameters for the source host to estimate.

The reason that the number of connections is unrelated to system stability can be explained as follows. A mathematical explanation is because the eigenvalues of the system transition matrix $\mathbf{A}$ is independent of the number of connections N [12]. This indicates that the convergence speed of the system defined by Eq. (9) is not affected by N. Also, an intuitive explanation is inter-dependency between the window size $w(k)$ and the number of packets at the buffer $q(k)$ around the fixed point. Namely, the number of packets depends proportionally on the number of connection (Eq. (7)). Also, the window size depends inverse proportionally on the number of packets at the buffer (see Eqs. (8) and (9)). We note that this inter-dependency is resulted from the factor $w_n(k)$ in the second term of the right-hand side in Eq. (6).

Equation (11) indicates that the control parameter, δ, should be positive for stability because ϵ, T_{min}, T_{max}, B, τ are all positive and T_{min} is less than T_{max}. From a control theoretic point of view, δ can be thought as a feedback gain of the system. It is therefore natural that

the feedback gain δ should be positive and small for system stability. Another interesting observation is that the system becomes stable as the control target ϵ decreases. This indicates that the system becomes less robust as the number of packets at the router's buffer grows.

3.3 Numerical Examples

By using Eq. (11), a stability region in the δ–τ plane is plotted in Fig. 2, where the router's bandwidth B, is set to 2 packet/ms, lower- and upper-thresholds, T_{min} and T_{max}, are set to 0 and 100 packets, respectively. Remind that the stability condition is independent of the number of connections N. In the figure, the control target ϵ, which controls the amount of packets queued in steady state, is changed from 0.1 to 0.9. Each line in the figure is an upper-bound of the stability region, so that the system becomes stable if δ is chosen below the boundary.

One can find that the stability region becomes narrow as the propagation delay τ increases. This tendency is more noticeable when the control target ϵ is set to be small. This implies that the window size can be changed aggressively when the control target ϵ is low. In other words, the window-based flow control mechanism is more robust in terms of stability when the control target ϵ is small. This indicates that it is desirable not to queue many packets at the router's buffer for system stability.

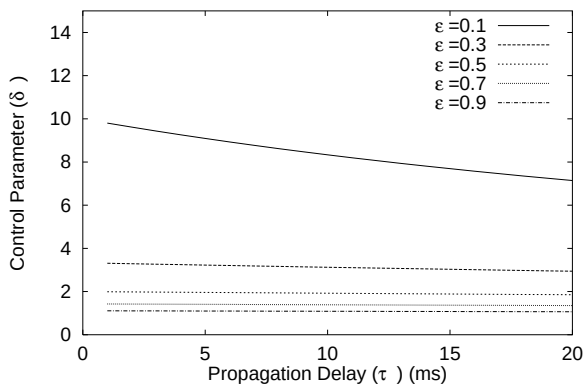


Figure 2: Stability region in the δ–τ plane ($B = 2$ packet/ms, $T_{min} = 0$ packet, $T_{max} = 500$ packet)

In Fig. 3, a stability region in the δ–τ plane with a different parameter set is plotted. Compared with the previous case (Fig. 2), only the router's bandwidth is changed to 10 times larger (i.e., $B = 20$ packet/ms). By examining these two figures, one can find that the upper-bound of δ is almost unchanged whereas the router's bandwidth becomes 10 times larger. In the window-based flow control mechanism, the amount of packets the source host can emit is determined by the router's bandwidth and the router's buffer capacity. Namely, in the round-trip time, the source host is allowed to send as many packets as its share of the router' bandwidth plus its share of the buffer space. Therefore, allowable aggressiveness of the window-based flow control mechanism, which directly affects the upper-bound of δ, is determined not only by the router's bandwidth but also by the buffer capacity. The stability condition clearly suggests this tendency (see the numerator of Eq. (11)).

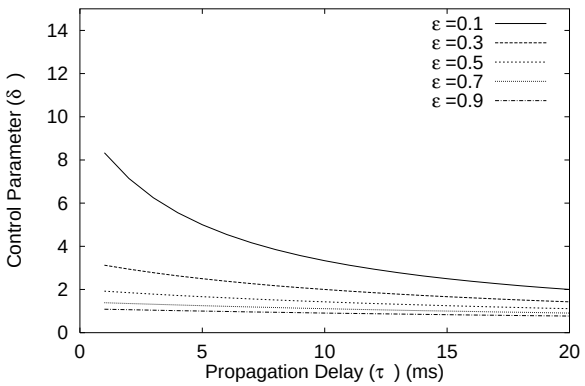
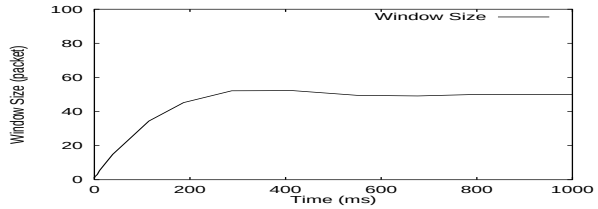


Figure 3: Stability region in the δ–τ plane ($B = 20$ packet/ms, $T_{min} = 0$ packet, $T_{max} = 500$ packet)

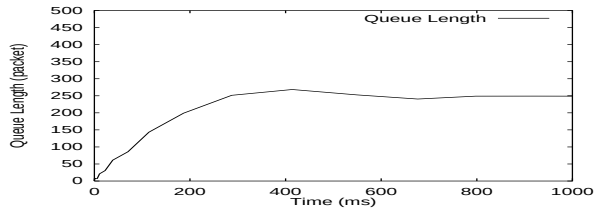
Next, we show dynamical behaviors of the window-based flow control mechanism in Figs. 4 though 7. We numerically obtained dynamics of the window size $w(k)$ and the number of packets at the router $q(k)$ from Eqs. (5)–(7). The initial values of $w(k)$ and $q(k)$ are set to 1 and 0, respectively. In these figures, following parameters are used: the number of connections $N = 10$, the propagation delay $\tau = 1$ ms, two thresholds $T_{min} = 0$ and $T_{max} = 500$ packets, and the control target $\epsilon = 0.5$. The router's bandwidth B is set to 2 packet/ms in Figs. 4 and 5, and to 20 packet/ms in Figs. 6 and 7. The control parameter δ is set to 1.5 in Figs. 4 and 6 (stable case), and to 2.5 in Figs. 5 and 7 (unstable case).

By comparing these figures, one can find that the system exhibits stable operation when the stability condition is satisfied. However, the system never reaches steady state in Fig. 6 even though stability condition is satisfied. Both the window size and the number of packets at the router's buffer slightly oscillate around the fixed point. It is because of the quantization error of the ratio of ECN messages as discussed in Section 2. In this case, since the fixed point of the window size is 52 packets, about 1% of the observation error is unavoidable. One solution for this would be to use averaging or filtering at the source host. However, it should be noted that if such a mechanism is applied, the averaging interval or the cut-off frequency of the low-pass filter must be chosen according to a trade-off between stability and transient performance.

Note that the quantization error of the ratio of ECN

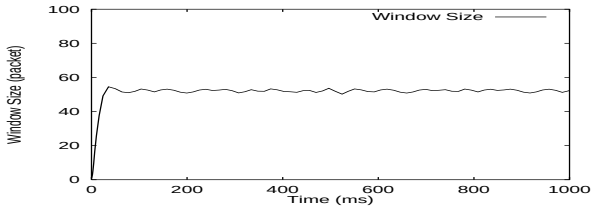


(a) Window size

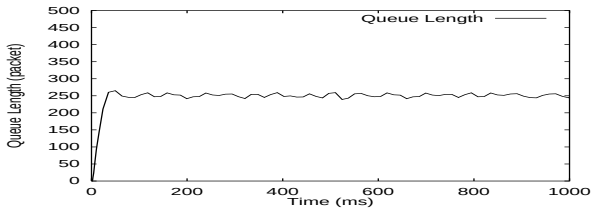


(b) Queue length

Figure 4: Stable behavior ($\delta = 1.5$, $B = 2$ packet/ms, $N = 10$, $\tau = 1$ ms, $\epsilon = 0.5$)

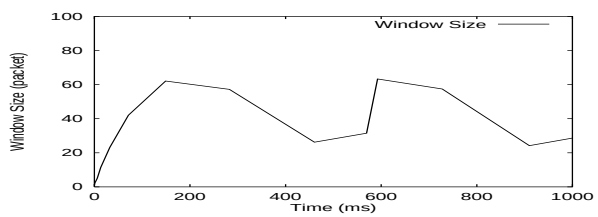


(a) Window size

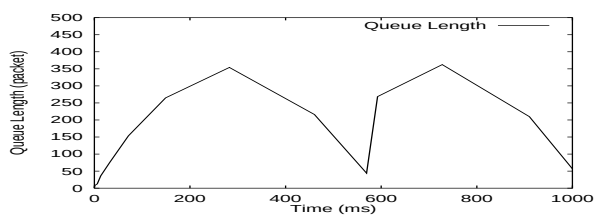


(b) Queue length

Figure 6: Stable behavior ($\delta = 1.0$, $B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\epsilon = 0.5$)

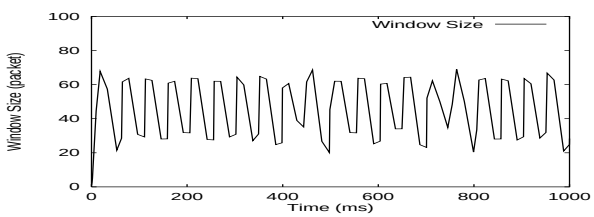


(a) Window size

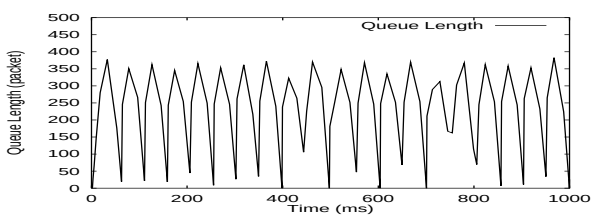


(b) Queue length

Figure 5: Unstable behavior ($\delta = 2.5$, $B = 2$ packet/ms, $N = 10$, $\tau = 1$ ms, $\epsilon = 0.5$)



(a) Window size



(b) Queue length

Figure 7: Unstable behavior ($\delta = 2.0$, $B = 20$ packet/ms, $N = 10$, $\tau = 1$ ms, $\epsilon = 0.5$)

messages is inversely proportional to the window size. Equation (8) indicates that the quantization error becomes small as one of the upper-threshold T_{max}, the router's bandwidth B, or the propagation delay τ increases. It also indicates that the larger value of the control target ϵ is helpful to reduce the quantization error. Hence, the quantization error of the ratio of ECN messages would become negligible as the network becomes faster in its speed and/or larger in its scale.

4 Regulator Design

In this section, we design a regulator for the window-based flow control mechanism. By using the regulator, a difficult problem of configuring the control parameter, δ, becomes unnecessary. Moreover, one can explicitly specify convergence speed of the system regardless of system parameters. In what follows, we first describe a design method of the regulator for the window-based congestion control mechanism. We then provide several numerical examples, showing that the transient performance can be considerably improved by using the regulator. This section is finished with discussion on several practical issues in real networks.

4.1 Implementation

As have been discussed in Section 3, the performance of the window-based flow control mechanism mostly depends on choice of the control parameter δ. If δ is configured to an inappropriate value, the network would take quite a long time to converge, or would never reach the steady state. The most desirable characteristic of the window-based flow control mechanism described in Section 2 is that the stability condition is independent of the number of connections N. Therefore, δ can be chosen without paying attention to the number of connections. However, the feedback gain δ must be chosen according to other parameters such as the router's bandwidth B, the propagation delay τ, the router's thresholds T_{min} and T_{max}, and the control target ϵ.

A regulator is a state feedback controller to improve stability and convergence speed of a given system by using the current system state as an input to the system [11]. Letting $\mathbf{u}(k)$ be the input to the system at slot k, the window-based flow control mechanism defined by Eq. (9) is rewritten as

$$\mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\,\mathbf{u}(k). \tag{12}$$

It should be noted that the window size $w(k)$ can be controlled freely at the source host, but the number of packets at the buffer $q(k)$ cannot. The window size at the source host is controlled by TCP's congestion control mechanism. However, the number of packets in the router's buffer is changed as a result of subsequent packet arrivals and departures. It is therefore impossible for the router to directly control the queue length. For a practical use, the matrix $\mathbf{B}$ should be of the form:

$$\mathbf{B} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \tag{13}$$

The basic idea of the regulator is to use $\mathbf{x}(k)$ as a feedback input to the system in order to improve stability and transient performance. When the system state $\mathbf{x}(k)$ is observable, $\mathbf{x}(k)$ can be used as the input to the system; the input, $\mathbf{u}(k)$, is replaced by

$$\mathbf{u}(k) = -\mathbf{F}\,\mathbf{x}(k), \tag{14}$$

where $\mathbf{F}$ is a feedback gain matrix. By combining Eqs. (12) and (14), the window-based flow control mechanism is given by the following equation.

$$\mathbf{x}(k+1) \;=\; \mathbf{A}\,\mathbf{x}(k) - \mathbf{B}\,\mathbf{F}\mathbf{x}(k) = (\mathbf{A} - \mathbf{B}\,\mathbf{F})\,\mathbf{x}(k)$$

It is known that if and only if $(\mathbf{A}, \mathbf{B})$ is controllable, the eigenvalues of $\mathbf{A} - \mathbf{B}\,\mathbf{F}$ can be set arbitrary by choosing $\mathbf{F}$ appropriately [11]. In the current case, the controllability matrix, $\mathbf{U_c}$, becomes

$$\mathbf{U_c} \;=\; (\mathbf{B}, \mathbf{A}\,\mathbf{B}) = \begin{bmatrix} 1 & 1 \\ 0 & N \end{bmatrix},$$

and

$$|\mathbf{U_c}| = N \neq 0.$$

Therefore, the window-based flow control mechanism is controllable, so that eigenvalues of $\mathbf{A} - \mathbf{B}\,\mathbf{F}$ can be set arbitrary. This means that one can specify convergence speed of the system by using the regulator. Note that the eigenvalues can be chosen freely regardless of system parameters.

By letting λ_i (i = 1, 2) be the eigenvalues of $(\mathbf{A} - \mathbf{B}\,\mathbf{F})$, the following equation holds.

$$|s\mathbf{I} - (\mathbf{A} - \mathbf{B}\,\mathbf{F})| = (s - \lambda_1)(s - \lambda_2)$$

The feedback gain matrix $\mathbf{F}$ is obtained by solving the above equation.

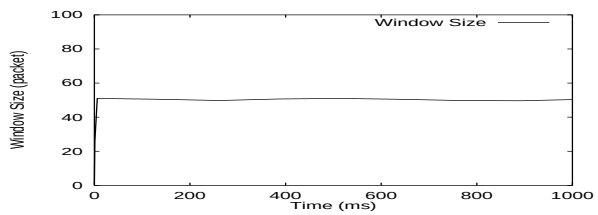
$$\mathbf{F} = \begin{bmatrix} -(\lambda_1 + \lambda_2) & \frac{(1+\lambda_1)(1+\lambda_2)}{N} - \frac{\delta\, N\, w^*}{T_{max} - T_{min}} \end{bmatrix} \tag{15}$$

Finally, $(\mathbf{A} - \mathbf{B}\,\mathbf{F})$ is given by

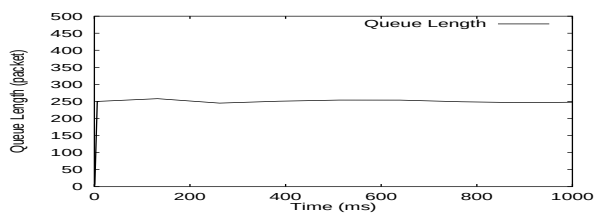
$$\mathbf{A} - \mathbf{B}\,\mathbf{F} = \begin{bmatrix} 1 + \lambda_1 + \lambda_2 & -\frac{(1+\lambda_1)(1+\lambda_2)}{N} \\ N & -1 \end{bmatrix}. \tag{16}$$

4.2 Numerical Examples

In Fig. 8, dynamical behaviors of the window size and the number of packets at the router's buffer when the regulator given by Eq. (15) is used at the source host. The regulator requires the current number of packets in the router's buffer $q(k)$. It is estimated from the ratio of ECN messages $e_n(k)$ using Eq. (5). In this case, the source host controls its window size using Eq. (16), where the roots, λ_1 and λ_2, are set to zero, meaning fastest convergence to the fixed point as we will explain later. All system parameters and control parameters are equivalent to those of Fig. 4. One can find that the transient performance is dramatically improved by introducing the regulator at the source host. For example, the window size is increased to the neighborhood of the fixed point immediately after the source host begins its packet transmission. It only takes less than 10 ms for the source host to open its window size to 50 packets in Fig. 8, while it takes 280 ms in Fig. 4. It can also be found that both the window size and the number of packets slightly oscillate around the fixed point as with the case of Fig. 4. This oscillation is also caused by the quantization error of the ratio of ECN messages.



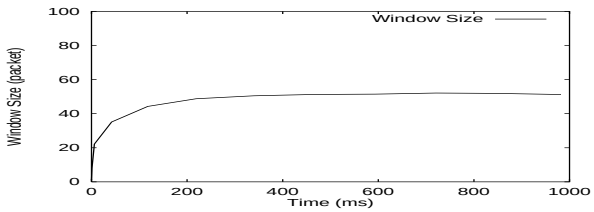
(a) Window size



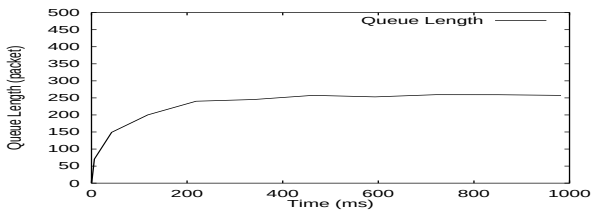
(b) Queue length

Figure 8: Case of regulator ($B = 2$ packet/ms, $N = 10$, $\tau = 1$ ms, $\lambda_1 = \lambda_2 = 0$)

Another desirable feature of using the regulator is that the convergence speed can be explicitly specified by choosing the roots, λ_1 and λ_2, appropriately. In theory, if both λ_1 and λ_2 are close to the origin of the unit circle, the system defined by Eq. (16) converges to its fixed point quickly. As the roots, λ_1 and λ_2, increases, the system takes more time to converge. And if one of the roots is out of the unit circle, the system becomes unstable. However, in general, there is a trade-off between convergence speed and robustness. When both roots are located close to the origin, the system is quite sensitive to the external noise, for example, the quantization error. Shown in Fig. 9 are dynamical behaviors of the window size and the number of packets at the buffer for $\lambda_1 = \lambda_2 = 0.5$. This figure shows that the convergence speed is slower than the case of Fig. 8.



(a) Window size



(b) Queue length

Figure 9: Case of regulator ($B = 2$ packet/ms, $N = 10$, $\tau = 1$ ms, $\lambda_1 = \lambda_2 = 0.5$)

4.3 Practical Issues

One of the hardest constrains for using the regulator in real TCP/IP networks is that the regulator assumes all system states to be observable. Namely, the current state of the system — i.e., the window size and the number of packets at the router's buffer — must be known. Since the regulator is performed at the source host as a part of the window-based flow control mechanism, the window size is already known. The problem is how to obtain the number of packets at the router. In the window-based flow control mechanism described in Section 2, the source host calculates the ratio of ECN messages. So it is easy for the source host to estimate the current number of packets from Eq. (1) if it knows two thresholds, T_{min} and T_{max}. This method was used in numerical examples.

As can be seen from Eq. (16), the regulator must know the number of connections N, which is another constraint of applying the regulator to the window-based flow control mechanism. There exists no general way to esti-

mate the current number of connections at the source host. One solution would be that the router observes the number of connections and informs source hosts of this value. However, this approach complicates the router's algorithm. Another solution would be to estimate the number of connections from the window size in steady state [13]. Namely, the number of connections can be calculated from Eq. (8) once other parameters are obtained in some way. However, the number of connections cannot be estimated by this approach immediately after the source host starts its packet transmission. It is because observation for a certain duration is necessary.

Also, note that the source host has to know the fixed point (w^*, q^*) for controlling its window size according to Eq. (16). The fixed point can be easily obtained for long-lived connections, i.e., TCP connections continuously send packets for a long period. This is the case we have examined in numerical examples. For short-lived connections, a measurement-based approach is difficult to apply. It would be possible for the source host to know the number of packets in steady state, q^*. It is because q^* can be computed from Eq. (8) since T_{min}, T_{max}, and ϵ are all constants. On the other hand, the window size in steady state, w^*, is difficult to obtain. It is because from Eq. (8), w^* depends on both the bandwidth of the bottleneck router B and the number of connections N, which dynamically change in real networks. One possible solution for this problem is that the source host maintains the fixed point (w^*, q^*) in its internal variable, and use this value for short-lived connections.

Although the regulator is difficult to apply to short-lived connections, it brings an advantage for long-lived connections. Namely, once the fixed point of the system is measured at the source host, the source host works quite effectively without estimating the bandwidth of the router B and the propagation delay τ. Namely, the regulator controls the window size according to Eqs. (5) and (16), both of which do not contain any term including B and τ. In short, the window-based flow control mechanism with the regulator enables to freely specify its convergence speed, and requires information on neither the bandwidth of the router nor the propagation delay.

5 Conclusion

In this paper, we have focused on a window-based flow control mechanism, which cooperates with routers supporting the ECN mechanism. We have thoroughly discussed how the ECN mechanism is incorporated into the TCP/IP network when all source hosts respond to the ECN message. By applying control theory, we have derived the stability condition of the window-based flow control mechanism, and have shown that the system stability is significantly affected by the router's buffer size as well as the bandwidth of the bottleneck router. We have also shown that the number of TCP connections is unrelated to the system stability. Also, we have designed a regulator for the window-based flow control mechanism, which utilizes the current window size and the estimated number of packets at the router's buffer as the feedback input. We have shown that the transient performance is considerably improved by using the regulator, and have also discussed several practical issues.

For future work, a compatibility issue of the proposed window-based flow control mechanism with other existing congestion control mechanisms such as TCP Reno and TCP Vegas should be examined. Every congestion control mechanism has a different control objective so that it would cause an unfairness problem between groups of TCP connections obeying different congestion control mechanisms. For instance, the control objective of the proposed window-based flow control mechanism is to settle the buffer occupancy at a certain level. On the contrary, the control objective of TCP Reno is to keep the buffer occupancy almost full at any time. Accordingly, it is expected that connections obeying TCP Reno consumes more bandwidth than connections obeying our proposed mechanism.

Another important issue is fairness among connections. Since our window-based flow control mechanism is a sort of multiplicative increase/decrease algrotihms. This algrotihm does not tend to achieve fairness among connections when every connection has a different initial window size [14]. For achieving better fairness, some modificatino on the proposed algrotihm would be necessary.

Also, we should extend our analysis to more generic network topologies. In the current paper, we have assumed that all connections have identical propagation delays, and there is only a single bottleneck router in the network. In real TCP/IP networks, propagation delays of all connections are not identical, and the bottleneck router may change as time changes. The performance of our window-based flow control mechanism in such generic network configurations should be studied.

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] S. Floyd, "TCP and explicit congestion notification," *ACM Computer Communication Review*, vol. 24, pp. 10–23, October 1994.

[4] B. Braden and J. Postel, "Requirements for internet gateways," *Request for Comments (RFC) 1009*, June 1987.

[5] S. Kalyanaraman, D. Harrison, S. Arora, K. Wanglee, and G. Guarriello, "A one-bit feedback enhanced differentiated services architecture," *Internet Draft* `<draft-shivkuma-ecn-diffserv-01.txt>`, March 1998.

[6] K. Ramakrishnan and S. Floyd, "A proposal to add Explicit Congestion Notification (ECN) to IP," *Request for Comments (RFC) 2481*, January 1999.

[7] 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.

[8] 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.

[9] 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*, December 1999.

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

[11] G. F. Franklin and J. D. Powell, *Digital Control of Dyanamic Systems*. Addision-Wesley Publishing Company, 1980.

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

[13] H. Ohsaki, M. Murata, T. Ushio, and H. Miyahara, "A control theoretic analysis of a window-based flow control mechanism in TCP/IP networks," submitted to *ACM SIGMETRICS 2000*, October 1999.

[14] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithm for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17, pp. 1–14, June 1989.