Master's Thesis

Title

Increasing robustness to environmental changes for congestion control mechanisms

Supervisor

Professor Hirotaka Nakano

Author

Mizuho Kodama

February 16th, 2009

Department of Information Networking

Graduate School of Information Science and Technology

Osaka University

Master's Thesis

Increasing robustness to environmental changes for congestion control mechanisms

Mizuho Kodama

Abstract

With explosive growth of the Internet, the performance of data transmission in the Internet depends highly on the transport-layer congestion control mechanisms. For end-to-end transport-layer protocols, since the network is regarded as a black box, indications for network congestion are necessary for effective congestion control. Though the traditional TCP utilizes only packet loss events for the congestion indication, many TCP variants that utilize the delay and bandwidth information of the network paths have been proposed to accommodate recent high-speed, large-bandwidth, and wired-wireless integrated networks. Since the measurement of delay and bandwidth of the network path generally includes the noises and errors, we should deal with them appropriately to improve the performance of such TCP variants.

In this thesis, we focus on the additional mechanisms to Internet congestion control to diminish and intentionally utilize measurement noises and errors due to environmental changes. First, we propose to control the degree of dependence on measured values of congestion indicators. We then introduce the dynamic parameter setting mechanism to control the sensitivity of protocol behavior to the measurement results of the congestion indicators. Second, we consider adding self-induced oscillation to the data transmission rate to absorb the ill-effect of the environmental changes. In general, it is difficult to recognize the sudden changes of the network environment quickly, because we cannot avoid the delay in the network measurement. Against this problem, we propose to add some randomness in determining the increase/decrease slope of data transmission rate.

We confirm the effectiveness of the proposed mechanisms by applying them to TCP Symbiosis, one of the bandwidth-based TCP modifications. Through the simulation experiments, we show that the former approach can keep the link utilization to be close to 100 %

regardless of the degree of measurement noises, and that the latter approach improves the link utilization from 75 % to 99 % by absorbing the ill-effect of the environmental changes.

Keywords

TCP
congestion control
network measurement
available bandwidth
measurement error
self-induced oscillation

Contents

List of Figures

1 Introduction

Today, the Internet environment has spread widely and it became certainly the most valuable information platform with over 800 million people using it everyday. It is expected that a billion people will be clicking away on the Internet by around 2010 [1]. With explosive growth of the Internet, the performance of data transmission in the Internet depends highly on the transport-layer congestion control mechanisms [2, 3]. For end-to-end transport-layer protocols, since the network is regarded as a black box, effective indications for detecting network congestion are necessary for the congestion control in the Internet.

Regarding TCP [4], which is the standard transport layer protocol in the current Internet, it usually utilizes the packet loss events for detecting network congestion [5-10]. For example, in TCP Reno, a TCP sender continues increasing its window size additively while no packet loss occurs, and decreases it multiplicatively when packet losses are detected. This simple mechanism is called as *loss-based* approaches.

Recently, many approaches which utilize the delay information of the network paths for the congestion indication have been proposed for recent high-speed, large-bandwidth, and wired-wireless integrated networks [11-14]. The method in [11, 12], which are called *delay-based* approaches, utilizes Round Trip Time (RTT) values for the congestion indication to deal with the incipient network congestion. When the network congestion occurs, the RTT values for a TCP connection increase due to the queuing delay at the bottleneck link. Therefore, the delay-based approaches can detect the earlier stage of the network congestion than *loss-based* approaches. The authors in [13, 14] proposed *hybrid* approaches which combine *loss-based* and *delay-based* approaches; namely, they utilize both packet loss events and RTT values for the congestion detection. They increase the congestion window size by using *delay-based* mechanisms when the network is not congested. Additionally, when they compete *loss-based* TCP connections, they switch their behavior to a *loss-based* mechanism (identical to the traditional TCP) to achieve fair share with the competing connections.

On the other hand, we have proposed TCP Symbiosis, which is one of the *bandwidth-based* approaches that utilizes the bandwidth information for the congestion indication [15,

16]. TCP Symbiosis regulates the window size so that the transmission rate meets the available bandwidth obtained from inline network measurement [17, 18], which estimates the physical capacity and available bandwidth of the network path in an inline fashion by using data and ACK packets transmitted by an active TCP connection.

Generally, the measurement of delay and bandwidth of the network path includes the noises and errors since the measurement is executed in an end-to-end fashion. The performance of above-mentioned protocols which depends highly on measurement of delay and bandwidth, are sensitive to such measurement noises and errors, although they behave ideally when the measurement results reflect the congestion level of the network path accurately. TCP Symbiosis [15, 16] is the typical instance which has this characteristics. Although TCP Symbiosis has the effectiveness in terms of average throughput, stability, convergence time, fairness among connections, and scalability to the bandwidth-delay product, the performance highly depends on the measurement results of the available bandwidth of the network path. For example, when the measurement result is larger than true value, the congestion window size becomes too large and it causes the buffer overflow at the bottleneck link. On the other hand, when the measurement result is smaller than true value, the congestion window size of TCP Symbiosis does not reach the enough value to fully utilize the available bandwidth. That is, the performance of TCP Symbiosis would degrade when the measurement results of the bandwidth information have noises and errors. Therefore, we should deal with them appropriately to improve the performance of such measurement-based TCP variants.

In this thesis, we focus on the additional mechanisms to Internet congestion control to diminish and intentionally utilize measurement noises and errors due to environmental changes. First, we propose to control the degree of dependence on measured values of congestion indicators. The basic idea under this proposal is that the reliability of the congestion indicators changes according to the accuracy of measurement. We then introduce the dynamic parameter setting mechanism to control the sensitivity of protocol behavior to the variance of the measurement results of the congestion indicators. We give mathematical explanations on the effect of noises and errors on the congestion indicators, and reveal how they affect the protocol performance. By applying the proposed mechanisms to TCP Symbiosis and confirming its performance through ns-2 [19] simulation experi-

ments, we present that it can keep the link utilization to be close to 100 % regardless of the degree of the measurement noises. Second, we consider adding self-induced oscillation to the data transmission rate to absorb the ill-effect of the environmental changes. In general, it is difficult to recognize the sudden changes of the network environment quickly, because we cannot avoid the delay in the network measurement. Against this problem, we propose to add some randomness in determining the increase/decrease slope of data transmission rate. Through simulation results, we confirm that the proposed method is effective to avoid occurring the retransmission timeout when the available bandwidth decreases suddenly due to some environmental changes.

The rest of this thesis is organized as follows. In Section 2, we introduce research background on measurement-based congestion control mechanisms, and explain the behavior of TCP Symbiosis briefly. Then, we introduce two methods and demonstrate them by the simulation results in Sections 3 and 4, respectively. We finally conclude this thesis with future work in Section 5.

2 Related work

2.1 Measurement-based congestion control mechanisms

The traditional TCP usually utilizes only packet loss events for detecting network congestion. For example, a TCP Reno sender continues increasing its window size additively while no packet loss occurs, and decreases it multiplicatively when packet losses are detected. That is, it is reactive congestion control because it cannot recognize the congestion without packet losses. Contrary to the reactive congestion control based on packet loss detection, the proactive congestion control based on the delay and bandwidth is recently investigated in many researches.

Delay-based approaches utilize RTT values for the congestion indication, based on the implication that when the network congestion occurs, the RTT values for a TCP connections increase due to the queuing delay at the bottleneck link. Therefore, the delay-based approaches can detect the earlier stage of the network congestion than *loss-based* approaches. TCP Vegas [11] mechanism is one of delay-based approaches and whose mechanism is the basis for many following protocols. The main idea is to control the congestion window size based on the estimated amount of buffered packets in the network. When the amount is large, the TCP connection is sending too much extra data and it will cause congestion. On the contrary, when the amount is small, a connection is sending too little extra data and it cannot transmit data enough to fully utilize the available bandwidth. According to these estimations of the network congestion level, TCP Vegas regulates the congestion window size. Furthermore, there are some TCP variants which employ similar delay-based mechanisms [12-14].

On the other hand, we have proposed TCP Symbiosis, which is one of the *bandwidth-based* approaches that utilizes the bandwidth information for the congestion indication [15, 16]. Since the window size of a TCP connection indicates the maximum amount of packets that TCP can transmit for one RTT, the window size for a TCP connection should be equal to the product of the available bandwidth and the round-trip propagation delay of the network path between the sender and receiver hosts. TCP measures RTT by checking the departure times of data packets and the arrival times of the corresponding ACK packets. In the next subsection, we explain TCP Symbiosis briefly and show some

problems regarding the measurement noises and errors.

2.2 TCP Symbiosis

TCP Symbiosis utilizes the bandwidth information obtained from an inline measurement technique [17, 18], and the window updating algorithm is borrowed from biophysics models; the logistic growth and Lotka-Volterra competition models [20]. In this subsection, we introduce the inline measurement technique and the window updating algorithm of TCP Symbiosis. In addition, we present the performance characteristics and problems through simulation and implementation experiments.

2.2.1 Inline bandwidth measurement

Inline bandwidth measurement is the technique to estimate the physical capacity and available bandwidth of an end-to-end network path in an inline fashion, by using data and ACK packets transmitted by an active TCP connection. Previously, numerous measurement tools have been proposed in the literature [21-25]. However, we cannot directly employ those methods in TCP mechanisms, primarily because these methods utilize a lot of test probe packets. Moreover, these methods require too much time to obtain one measurement result. On the other hand, we can avoid these problems by using inline measurement technique. It can continuously measure bandwidth by using data and ACK packets of a TCP connection under data transmission. Figure 1 depicts the mechanism of inline network measurement. Since it performs the measurement without transmitting additional probe packets over the network, the effect on other network traffic is negligible. It can also quickly update according to the latest changes in bandwidths by frequently performing measurements (one result per 3-4 RTTs) as long as TCP transmits data packets.

ImTCP [17, 18] is a TCP modification which execute the inline network measurement. The algorithm of ImTCP is briefly summarized as follows.

(1) For searching the value of the available bandwidth, we introduced a search range which is expected to include the current value of the available bandwidth. Because the frequency of sending packets in high-rate reduces by introducing the search range,

serious affects to other traffic can be avoided.

(2) We divided the search range into sub-ranges and send a packet stream for each sub-range. The transmission rate of the streaming packets varies to cover the bandwidth range of the sub-range.

(3) When the corresponding ACK packets arrive at the sender host, we check if an increasing trend exists in the arrival intervals of the ACK packets compared with the sending intervals of the data packets. The increasing trend of transmission delay in a stream indicates that the transmission rate of the stream is larger than the current available bandwidth of the network.

(4) We choose the sub-range which includes the current value of the available bandwidth and estimates the value of available bandwidth. Physical capacity is also calculated by using of the available bandwidth values [18].

(5) The search range is set to 95% of confidence interval of the measurement results and ImTCP continues measurements till the TCP connection is terminated.

Because the search range is set by using the past measurement results, the available bandwidth is sometimes not included in the search range when the value of available bandwidth suddenly changes with the change of network condition. In such occasions, ImTCP can find the new value of available bandwidth by a few measurements. See [17, 18] for more detail.

The authors have also proposed an implementation design of ImTCP [17], in which the measurement program is located at the bottom of the TCP layer. The proposed implementation design maintains the transmission and arrival intervals of TCP data and ACK packets by introducing a FIFO buffer between TCP and IP layers. Note that the measurement algorithm has limited effect on TCP's congestion control algorithm [17], meaning that the measurement algorithm can be applied to any TCP variants including TCP Symbiosis.

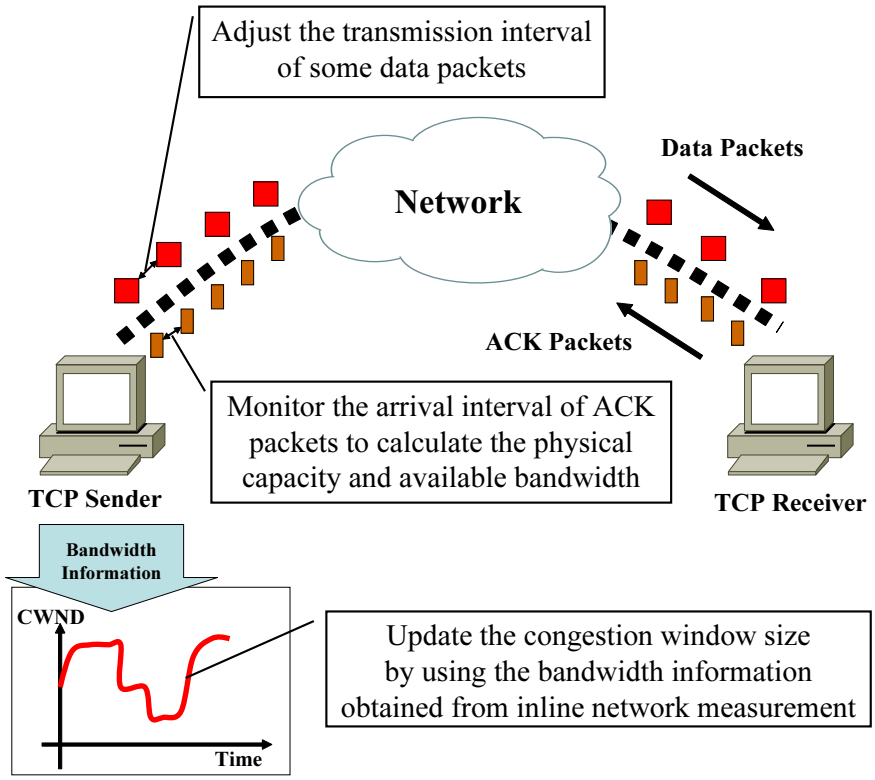


Figure 1: Inline network measurement

2.2.2 Bio-inspired window updating algorithm

TCP Symbiosis utilizes the information of physical capacity and available bandwidth obtained from an inline measurement technique described above, and the window updating algorithm is borrowed from biophysics models; the logistic growth and Lotka-Volterra competition models [20]. In many biological systems, the actions of the entity (e.g., living organism) are not determined based on the results of direct interactions among entities, but rather on information obtained through the environment, which is a fundamental necessary condition for the system to be robust. Furthermore, the species which have the same characteristic (e.g., the carrying capacity of the environment, the intrinsic growth and the ratio of the competition coefficient of other species) can keep fairness. Therefore, we expect TCP Symbiosis can achieve stability, robustness and fairness.

The logistic equation is a formula that represents the evolution of the population of a single species over time. Generally, the per capita birth rate of a species increases as the

population of the species becomes larger. However, since there are various restrictions on living environments, the environment has a carrying capacity (upper limit of population size), which is usually determined by the available sustaining resources. The logistic equation describes such changes in the population of a species as follows [20].

$$\frac{d}{dt}N = \epsilon\left(1 - \frac{N}{K}\right)N \tag{1}$$

where t is time, N is the population of the species, K is the carrying capacity of the environment, and ϵ is the intrinsic growth of the species ($0 < \epsilon$).

The Lotka-Volterra competition model is a well known model for examining the population growth of two or more species that are engaged in inter-specific competition. In the model, Equation (1) is extended to include the effects of both inter-specific competition and intra-specific competition. The basic two species Lotka-Volterra competition model with both species N_1 and N_2 having logistic growth in the absence of the other is comprised of the following equations [20].

$$\frac{d}{dt}N_1 = \epsilon_1\left(1 - \frac{N_1 + \gamma_{12}N_2}{K_1}\right)N_1, \tag{2}$$

$$\frac{d}{dt}N_2 = \epsilon_2\left(1 - \frac{N_2 + \gamma_{21}N_1}{K_2}\right)N_2 \tag{3}$$

where N_i, K_i, and ϵ_i are the population of the species, the carrying capacity of the environment, and the intrinsic growth rate of the species i, respectively. γ_{ij} is the ratio of the competition coefficient of species i with respect of species j.

In this model, the population of species N_1 and N_2 does not always converge to a value larger than 0, and in some cases one species becomes extinct, depending on the values of γ_{12} and γ_{21}. Commonly, the following equations are sufficient conditions for the two species to survive in the environment [20]:

$$\gamma_{12} < \frac{K_1}{K_2}, \quad \gamma_{21} < \frac{K_2}{K_1}. \tag{4}$$

Assuming that the two species have the same characteristics, they have the same values: $K = K_1 = K_2$, $\epsilon = \epsilon_1 = \epsilon_2$, and $\gamma = \gamma_{12} = \gamma_{21}$. Then, Equations (2) and (3) can be

written as follows:

$$\frac{d}{dt}N_1 = \epsilon\left(1 - \frac{N_1 + \gamma N_2}{K}\right)N_1, \tag{5}$$

$$\frac{d}{dt}N_2 = \epsilon\left(1 - \frac{N_2 + \gamma N_1}{K}\right)N_2. \tag{6}$$

In addition, Equation (4) can be written as $\gamma < 1$. Furthermore, we can easily extend Equations (5) and (6) for n species as follows:

$$\frac{d}{dt}N_i = \epsilon\left(1 - \frac{N_i + \gamma\sum_{j=1,i\neq j}^{n} N_j}{K}\right)N_i. \tag{7}$$

Note that survival and convergence conditions are identical, i.e., $\gamma < 1$. Even when two or more species exist, each independently utilizes Equation (7) to obtain N_i, and the population of the species can converge to the value equally shared among competing species. We consider that the changing population trends of species are ideal for controlling the transmission speed of TCP. That is, by using Equation (7) for the congestion control algorithm of TCP, rapid and stable link utilization can be realized, whereas each TCP connection can behave independently as an autonomous distributed system.

To convert Equation (7) to a window increase/decrease algorithm, we consider N_i as the transmission rate of TCP sender i and K as the physical capacity of the bottleneck link. Furthermore, it is necessary for connection i to know the data transmission rates of all other connections that share the same bottleneck link. This assumption is quite unrealistic with respect to the current Internet. Therefore, we estimate the sum of the data transmission rates of all of the other connections using the physical capacity and available bandwidth as follows:

$$\sum_{j=1,i\neq j}^{n} N_j = K - A_i$$

where A_i is the available bandwidth for connections i. Thus, Equation (7) becomes:

$$\frac{d}{dt}N_i = \epsilon\left(1 - \frac{N_i + \gamma(K - A_i)}{K}\right)N_i. \tag{8}$$

The proposed mechanism requires modifications only with respect to sender-side TCP, and no change in receiver-side TCP is required. A TCP sender controls its data transmission rate by changing its window size. To retain the essential characteristics of TCP and

decrease the implementation overhead, we employ window-based congestion control in the proposed TCP by converting Equation (8) into an algorithm of window size in TCP. The window size of connection i, w_i, is calculated from N_i, the transmission rate, using the following equation:

$$w_i = N_i \tau_i.$$

where τ_i is the minimum value of the RTTs of connection i, which is assumed to equal the propagation delay without a queuing delay in the intermediate routers between sender and receiver hosts. Then, Equation (8) can be rewritten as follows.

$$\frac{d}{dt} w_i = \epsilon \left(1 - \frac{w_i + \gamma(K - A_i)\tau_i}{K \tau_i} \right) w_i. \tag{9}$$

Finally, we integrate Equation (9) as follows.

$$w_i(t) = \frac{w_i(0)\tau_i f_i(t) \{K - \gamma(K - A_i)\}}{w_i(0) (f_i(t) - 1) + \tau_i \{K - \gamma(K - A_i)\}}, \tag{10}$$

and

$$f_i(t) = e^{\epsilon t \left\{ 1 - \gamma \left(1 - \frac{A_i}{K} \right) \right\}}.$$

In Equation (10), when we set the initial value of the window size ($w_i(0)$) and the current time to 0 ($t = 0$), we can directly obtain window size $w_i(t)$ for any time t. We use the above equation for the control algorithm of the window size of TCP connections.

Since Equation (10) requires the measurements of the physical capacity and available bandwidth of an end-to-end network path, we use the same algorithm as TCP Reno for window updating algorithm until the measurement results are obtained through ImTCP. In case of detecting packet losses by receiving three duplicated ACKs, a window size is halved in an identical way to TCP Reno. Similarly, when a timeout occurs, the sender TCP discards all measurement results, the window size is reset to 1, and the slow-start phase begins. With obtained bandwidth information of the network path, we can calculate the window size from Equation (10) by regarding the arrival interval between the latest two ACK packets as t.

TCP Symbiosis has two parameters, γ and ϵ. γ determines how the connection is affected by other connections which share the same bottleneck link. It has to satisfy $(0 < \gamma < 1)$ so that the window size converges to positive value, regardless the physical

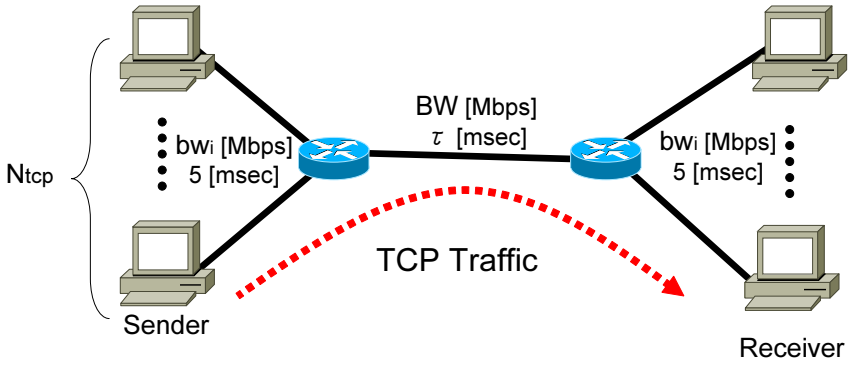


Figure 2: Network environment for simulation experiments in Section 2

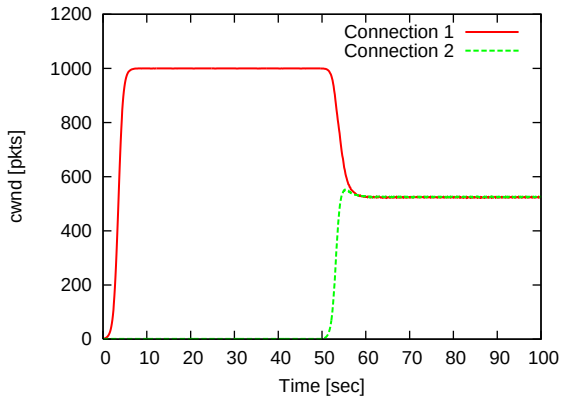


Figure 3: Basic behavior of TCP Symbiosis

bandwidth of all connections. We have to consider of the trade-off between convergence speed and the number of packets stored in the buffer of bottleneck link. ϵ determines the convergence speed. The more ϵ increases, the faster the converge speed is. However, in high-speed and long-delay networks, the large ϵ adversely affects the networks, because a TCP Symbiosis connection transmits many packet in a burst fashion when the increment of the window size is large when it receives ACK packets.

2.2.3 Performance characteristics and problems

We show the fundamental behavior of TCP Symbiosis by using the results of the simulation and implementation experiments.

We first show the typical behavior of TCP Symbiosis by the results of the simple

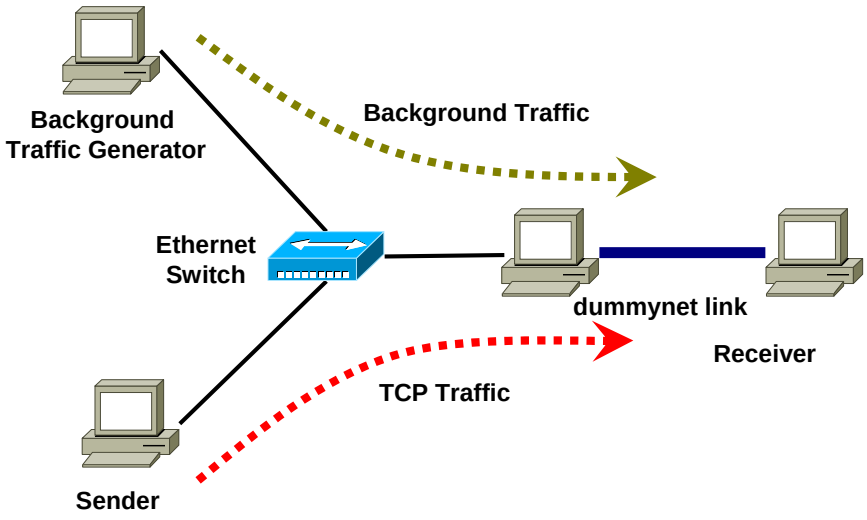
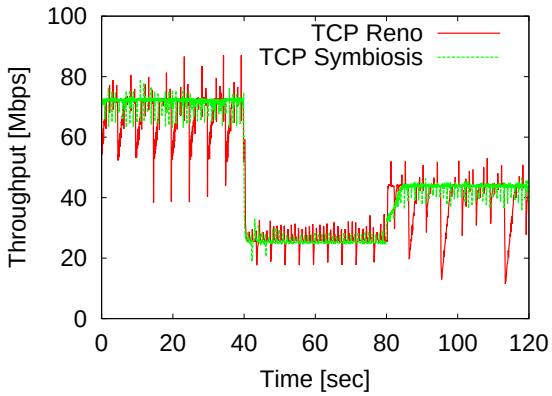


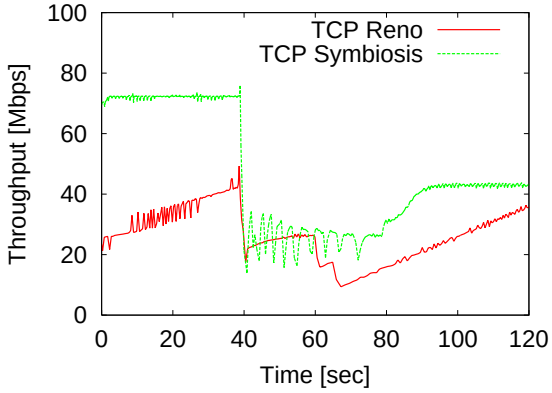
Figure 4: Network environment for implementation experiments

simulation. The simulation environment is shown in Figure 2, where $bw = 200$ Mbps, $BW = 100$ Mbps, and $\tau = 50$ msec. The bottleneck link bandwidth is 100 Mbps and the minimum RTT is 120 msec. We run 2 TCP Symbiosis connections and the second connection joins the network at 50 sec. Figure 3 illustrates the change in the congestion window size of the two TCP Symbiosis connections. The first connection rapidly increases at the beginning of the simulation and converges the congestion window size without packet losses. The converged congestion window size is close to 1000 packets, which means that the transmission rate is equal to 100 Mbps. That is, TCP Symbiosis connection can fully utilize the available bandwidth quickly. When the second connection joins the network, the first connection quickly recognizes the environmental changes and decreases the congestion window size. As a result, both connections converge their congestion window sizes to the same value. It means that the two connections fairly share the bottleneck bandwidth. In addition, the congestion window size is stable while the environment is unchanged.

We next show the performance of TCP Symbiosis by implementation experiments.



(a) RTT 30 msec



(b) RTT 150 msec

Figure 5: Basic behaviors of TCP Symbiosis and TCP Reno in different RTT

The network environment used in the experiments is depicted in Figure 4. The network environment consists of a PC router in which Dummynet [26] is installed, two sender hosts which are a TCP test-traffic generator and a UDP background traffic generator, and a receiver host which receives packets from each sender hosts. The test traffic is generated by using iperf [27]. The bottleneck link bandwidth and propagation delay of the network environment between the sender and receiver hosts can change by Dummynet. In this experiment, we set the bottleneck link bandwidth to 100 Mbps and the propagation delay is set so that the minimum RTT is 30 msec or 150 msec. UDP background traffic generator generates UDP traffic to the receiver so that the available bandwidth of the bottleneck link is 70 Mbps from 0 to 40 sec, 30 Mbps from 40 to 80 sec, and 50 Mbps from 80 to 120 sec.

We establish a TCP Symbiosis connection between Sender and Receiver. In addition, we conduct the same experiments using a TCP Reno connection for comparison purposes. Figure 5 shows changes in the throughput of each TCP connections as a function of the experiment time. When RTT is set to 30 msec, both protocols can effectively utilize the available bandwidth for TCP connection, while TCP Reno oscillates the throughput due to its well-known behavior [28]. On the other hand, when RTT is set to 150 msec, TCP Reno cannot achieve enough throughput. This is because the increment rate of congestion window size is too small for such a network with large bandwidth-delay-product. In contrast, the throughput of TCP Symbiosis is almost as same as in a network with small bandwidth-delay-product. From these results, we can conclude that TCP Symbiosis has scalability to the bandwidth-delay-product. We next focus on the packet losses during 40-80 sec and the delayed increase in the congestion window size after 80 sec of the TCP Symbiosis connection. These phenomena are caused by the low accuracy of measured available bandwidth. Because the measurement frequency is low as the RTT increases, the measurement accuracy of ImTCP degrades. As the result, the performance of TCP Symbiosis also degrades.

From above results, we presented that TCP Symbiosis has the effectiveness in terms of average throughput, convergence time, fairness among connections, stability, and scalability to the bandwidth-delay-product. At the same time, we revealed that TCP Symbiosis highly depends on the measurement results of available bandwidth. Refer to [15, 16] for more performance evaluation results.

3 Measurement noise-aware parameter settings

In the previous section, we have revealed that the measurement noises and errors of the available bandwidth degrade the throughput of TCP Symbiosis. In this section, we give the mathematical explanation of the effect of the measurement noises and propose the dynamic parameter setting algorithm to alleviate the problem.

3.1 Ill-effect of measurement noises

In general, the measured value of the available bandwidth, which is denoted as $A'(t)$ where t is time, includes the measurement noises. On the other hand, we assume that the physical capacity, K, remains unchanged and can be accurately measured, since the measurement noises of the physical capacity is relatively small compared with that of the available bandwidth. Equation (9) can be written as follows:

$$\frac{d}{dt}w(t) = \epsilon \left(1 - \frac{w(t) + \gamma(K - A'(t))\tau}{K\tau}\right) w(t). \tag{11}$$

The size of measurement noises is affected by that of measured bandwidth, because the general measurement algorithms of available bandwidth are based on the intervals of sending and receiving packets. Given that the noise ratio of $(K - A'(t))$ is $n(t)$. Then,

$$K - A'(t) = (1 + n(t))(K - A(t)), \tag{12}$$

where $A(t)$ is the true value of the available bandwidth at time t. Hence Equation (11) can be written as follows:

$$\frac{d}{dt}w(t) = \epsilon \left(1 - \frac{w(t) + (1 + n(t))\gamma(K - A(t))\tau}{K\tau}\right) w(t).$$

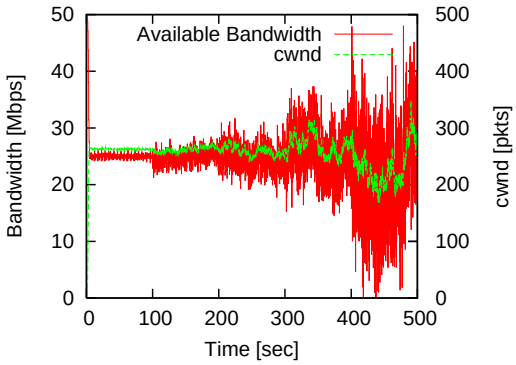
Now, Equation (12) can be written as follows:

$$\gamma'(t) = (1 + n(t))\gamma, \tag{13}$$

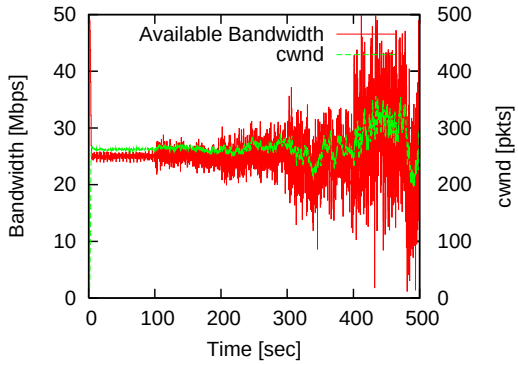
$$\frac{d}{dt}w(t) = \epsilon \left(1 - \frac{w(t) + \gamma'(t)(K - A(t))\tau}{K\tau}\right) w(t). \tag{14}$$

By comparing Equations (11) and (14), we can observe that the measurement noises in the available bandwidth can be regarded as the noises in the control parameter γ.

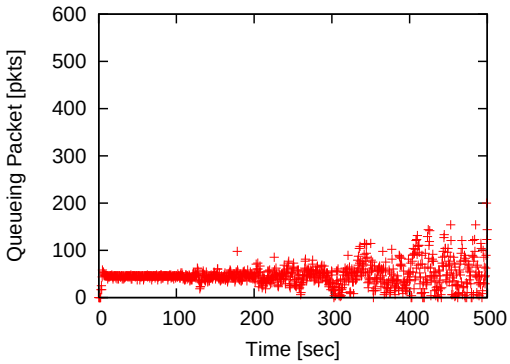
As shown in Subsection 2.2.2, γ has to satisfy $0 < \gamma < 1$ for the stable behavior in Lotka-Volterra competition model. When $\gamma > 1$, some species become extinct, meaning



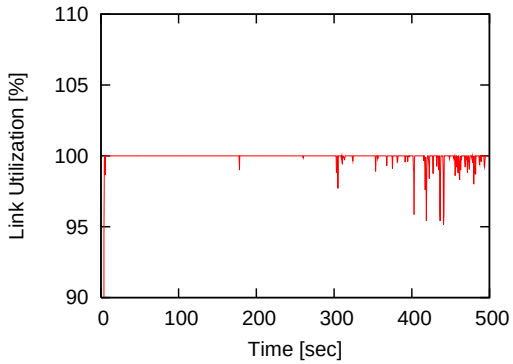
(a) Available bandwidth and cwnd of connection 1



(b) Available bandwidth and cwnd of connection 2



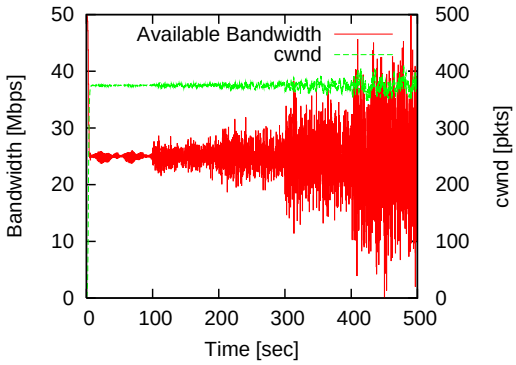
(c) Queue length of bottleneck link



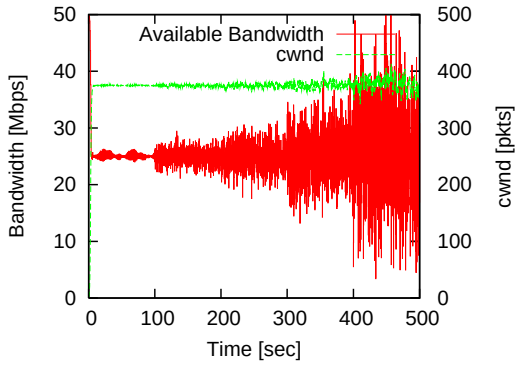
(d) Bottleneck link utilization

Figure 6: Simulation results with $\gamma = 0.95$

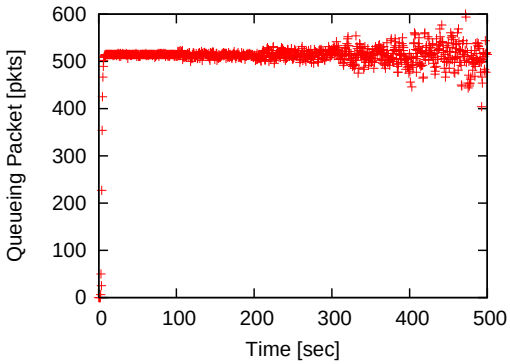
that some TCP connections suffer from very low throughput. Therefore, the noises in γ degrade the performance of TCP Symbiosis because it sometimes makes γ be larger than 1. Setting γ to the safer side (far smaller than 1) enables to avoid the extinct situation. However, because the converged window size, denoted as w^*, is $w^* = \{\gamma A + (1-\gamma)K\}\tau$ from Equation (9), the small value of γ would increase the queue length at the bottleneck link at the converged situation. We exhibit the above properties by simulation results. The simulation topology is the same as in Figure 3. The bottleneck link bandwidth is 50 Mbps and the minimum RTT is 120 msec. We ran 2 TCP Symbiosis flows at time 0. We intentionally add white noises to the measurement results of the available bandwidth to investigate the effect of the measurement noises. We start the simulation with the small



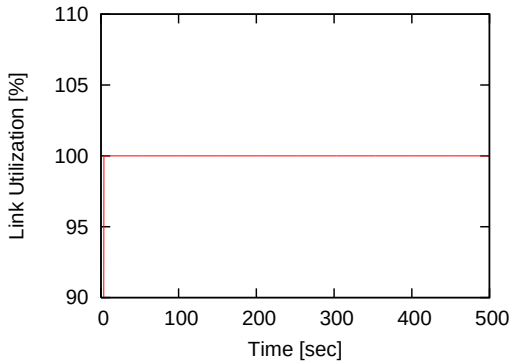
(a) Available bandwidth and cwnd of connection 1



(b) Available bandwidth and cwnd of connection 2



(c) Queue length of bottleneck link



(d) Bottleneck link utilization

Figure 7: Simulation results with $\gamma = 0.5$

noise, and double the noise size at every 100 sec. The results are summarized in Figure 6 for $\gamma = 0.95$ and Figure 7 for $\gamma = 0.5$, respectively. When γ is close to 1 ($\gamma = 0.95$), we can observe that the congestion window sizes of the two connections become unstable with larger fluctuations. This is because the larger noise in the available bandwidth makes $\gamma'(t)$ in Equation (14) to be larger than 1 frequently. This results in the unfairness between the two connections, unstable queue length (Figure 6(c)), and lower utilization of the bottleneck link bandwidth (Figure 6(d)). On the other hand, when γ is far smaller than 1 ($\gamma = 0.5$), the congestion window sizes of the two connections are stable regardless of the noise ratio (Figures 7(a) and 7(b)), and the bottleneck link utilization remains 100% (Figure 7(d)). However, as shown in Figure 7(c), the queue length of the bottleneck link

is kept at larger value as compared with Figure 6(c) with γ=0.95. This large queue length would increase the queueing delay at the bottleneck link buffer and the probability of buffer overflow also increases with sudden change of the background traffic.

From the above results, we conclude that the constant value of γ can not perform well in noise-prone environments.

3.2 Dynamic parameter setting

Against the above problem, we propose the dynamic setting algorithm of γ according to the measurement results of the available bandwidth. We determine γ based on the variance of the measured bandwidth values over time.

From Equation (13), we can dynamically determine γ as follows:

$$\gamma(t) = \frac{\gamma_t}{1 + lC_v}, \tag{15}$$

where γ_t is the target value for γ. Since $n(t)$ in Equation (13) is the noise ratio of $(K - A'(t))$, we replace $n(t)$ with C_v, which is the coefficient of variation of the measurement values of $(K - A'(t))$. In addition, we introduce the parameter l to control the degree of noise awareness.

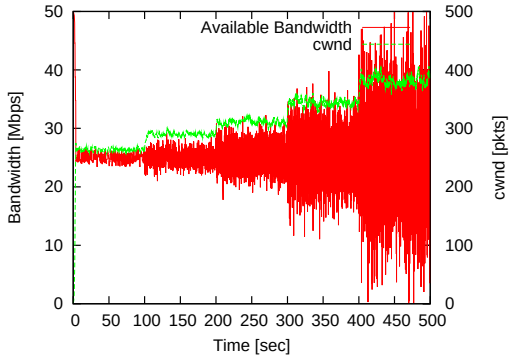
We obtain the deviation of the measurement results of the available bandwidth as RTO calculation in TCP [2]:

$$\begin{aligned} Err &= x - \mu, \\ \mu &\leftarrow \mu + gErr, \\ \sigma &\leftarrow \sigma + h(|Err| - \sigma), \end{aligned}$$

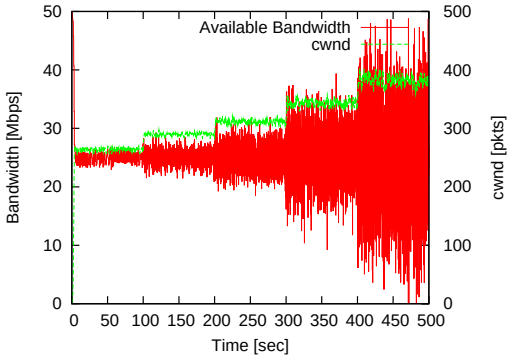
where x, μ and σ are the latest measured value, the moving average value and the average deviation. g and h are the weighting factor. Then we can calculate $C_v = \sigma/\mu$.

3.3 Performance evaluation

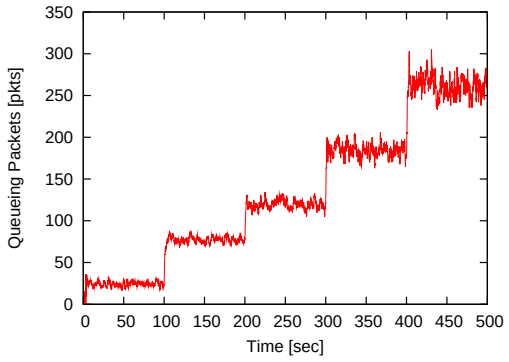
In Figure 8, we show the simulation results with the same simulation setting as in Figures 6 and 7. We set $g = 0.125$, $h = 0.25$ and $l = 4$ by reference to [2]. We also set $\gamma_t = 0.95$ because the performance of TCP Symbiosis with $\gamma = 0.95$ is reasonable when the



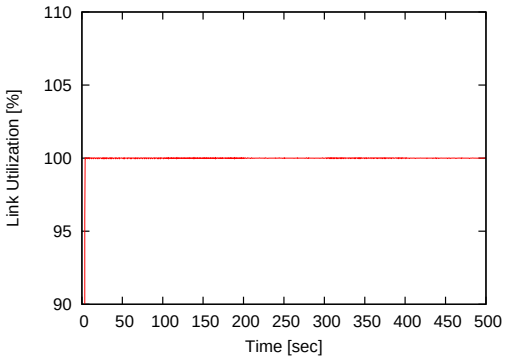
(a) Available bandwidth and cwnd of connection 1



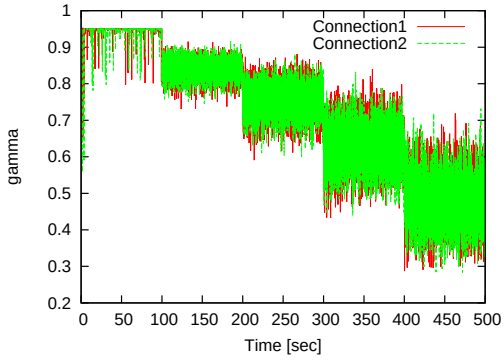
(b) Available bandwidth and cwnd of connection 2



(c) Queue length of bottleneck link



(d) Bottleneck link utilization



(e) Change in the value of γ

Figure 8: Simulation results with proposed method

measurement results are accurate. Figure 8(e) exhibits the changes in $\gamma(t)$ for the two connections in the proposed mechanism. From the figure, $\gamma(t)$ is varied with the variance of the measurement results of the available bandwidth. The change means that the degree of dependence on available bandwidth is varied based on the noises in the measurement results of the available bandwidth. As a result, we can see from Figure 8 that the congestion window sizes of the two connections remain stable regardless of the degree of the noise ratio (Figures 8(a) and 8(b)), and the utilization of the bottleneck link bandwidth keeps 100% (Figure 8(d)). In addition, the queue length of the bottleneck link is regulated as short as possible with keeping high link utilization and smaller than in Figure 7(c) especially when the degree of the noise ratio is small.

4 Self-induced oscillation for absorbing environmental changes

TCP Symbiosis highly depends on the measurement results of available bandwidth of a network path. In Section 3, we discussed the dynamic parameter setting according to the degree of the measurement noises, while the true value of the available bandwidth remains unchanged. In this section, we focus on the behavior of TCP Symbiosis when the true value of the available bandwidth suddenly changes due to some environmental changes.

4.1 Ill-effects of environmental changes

Generally, the end-to-end available bandwidth is measured by repeating the exchange of probe packets between endhosts. For example, in ImTCP, the sender TCP transmits TCP data packets with pre-defined transmission intervals, and calculates the available bandwidth from arrival intervals of the corresponding ACK packets. As a result, ImTCP gives estimation results every 3-4 RTTs. This means that when the true value of the available bandwidth changes, it takes several RTTs for the sender-side TCP to recognize the change. Furthermore, TCP symbiosis requires additional RTTs to converge its congestion window size according to the new value of the available bandwidth. These delayed behaviors of TCP Symbiosis affect the performance against the environmental changes.

We show an example of the performance degradation by simulation experiments. The simulation environment is shown in Figure 9, where $bw = 200$ Mbps, $BW = 100$ Mbps, and $\tau = 50$ msec. The bottleneck link bandwidth is 100 Mbps and the minimum RTT is 120 msec. Note that the buffer size of the bottleneck link is set to 1000 packets, the background traffic is composed by an UDP flow, and there is one TCP Symbiosis connection. The bandwidth of UDP traffic is 10 Mbps per flow and N_{udp} is set to 2 in 0-20 sec, 8 in 20-40 sec, and 2 in 40-60 sec so that the available bandwidth changes as 80 Mbps in 0-20 sec, 20 Mbps in 20-40 sec, and 80 Mbps in 40-60 sec. Figure 10 shows the change in the congestion window size. When the available bandwidth suddenly decreases at 20 sec, TCP's retransmission timeout occurs and the congestion window size is reset to one packet, which significantly degrades the throughput. This is caused by the packet losses at the buffer of the bottleneck link. On the other hand, when the available bandwidth suddenly increases at 40 sec, TCP Symbiosis requires some delay to fill-up the

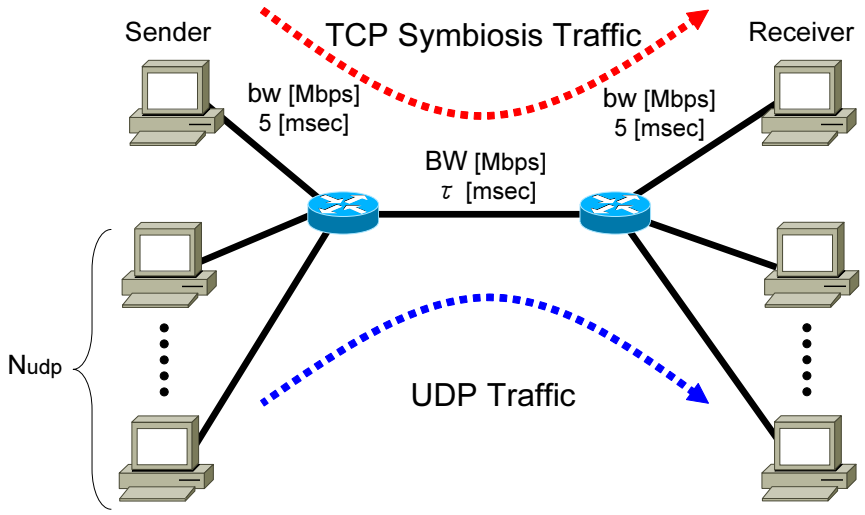


Figure 9: Network environment for simulation experiments in Section 4

increased bandwidth.

This delayed behavior of TCP Symbiosis is caused since it relies on the measurement results of the available bandwidth. When the network environment is stable, this mechanism gives good results in terms of stable and high throughput. However, when facing environmental changes, it degrades the performance due to its delayed behavior.

We also show another kind of ill-effect on fairness among connections when the number of concurrent connections increases. The simulation environment is shown in Figure 2, where $bw = 200$ Mbps, $BW = 100$ Mbps, and $\tau = 50$ msec. The bottleneck link bandwidth is 100 Mbps and the minimum RTT is 120 msec. We run 10 TCP Symbiosis connections and the number of concurrent connections increases from one to ten every 200 sec and it decreases from ten to one every 200 sec. Figure 11 illustrates the changes in the congestion window size and the minimum RTT τ of each connection. From Figure 11(a), the connections do not fairly share the bandwidth after the fourth connection joined the network, although the existing three connections remain fair. This phenomena are caused by the difference of τ among connections, as shown in Figure 11(b). Note that TCP Symbiosis utilizes the minimum RTT value for its control of the congestion window size (Equation (10)). Because the TCP Symbiosis connections can fully utilize the available bandwidth with small number of concurrent connections, the queue length remains almost

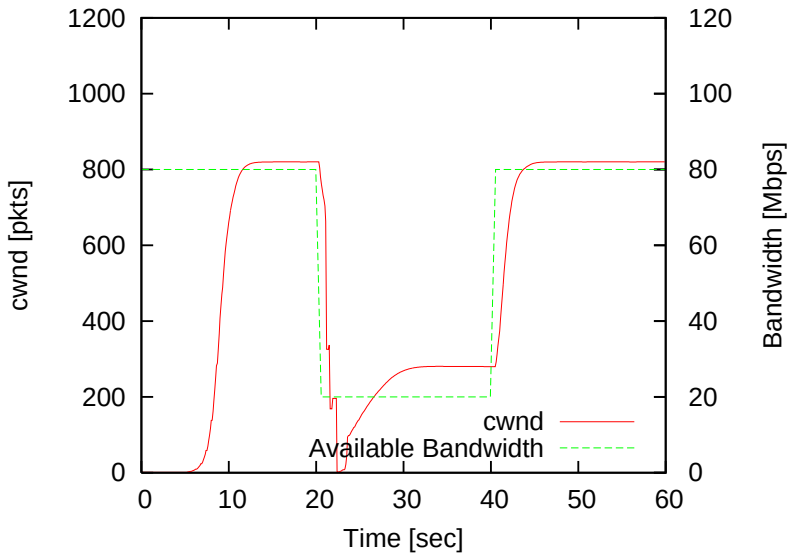


Figure 10: Change in the congestion window size without self-induced oscillation

unchanged. Then the additional connection joins the network, the connection can not obtain the correct value of the minimum RTT value. This leads the wrong convergence of the congestion window size.

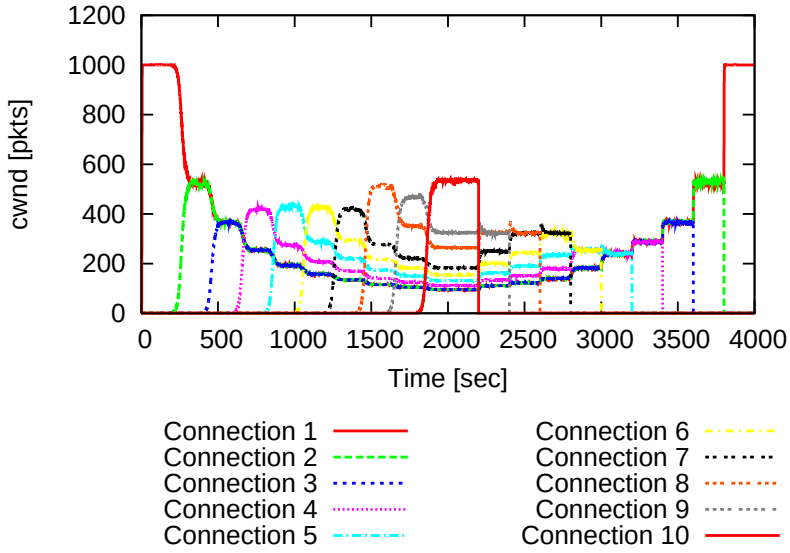
4.2 Self-induced oscillation

To solve the above problems, we propose to add self-induced oscillation to the congestion window size. This means that the congestion window size remains fluctuating even when the available bandwidth remains unchanged. Note that the average throughput does not change since the oscillation is equally injected in both of increase and decrease directions. By this mechanism, we expect that the ill-effect can be absorbed probabilistically.

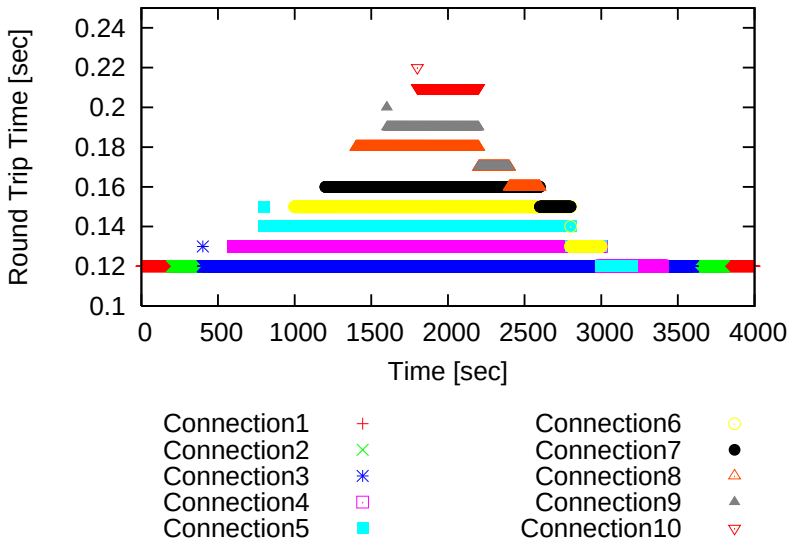
In [29], the authors define the two-species Lotka-Volterra model in the presence of noises as follows:

$$\begin{aligned} \frac{dx}{dt} &= \mu_1 x(\alpha_1 - x - \beta_1(t)y) + x\xi_x(t), \\ \frac{dy}{dt} &= \mu_2 y(\alpha_2 - y - \beta_2(t)x) + y\xi_y(t), \end{aligned} \tag{16}$$

where x and y are the population of the species, α_1 and α_2 are the intrinsic growth rate, $\beta_1(t)$ and $\beta_2(t)$ are the coefficient of intra-specific competition. $\xi_x(t)$ and $\xi_y(t)$ denotes Gaussian white noise. These equations represent the change in the population of



(a) Change in congestion window size



(b) Change in τ

Figure 11: Ill-effect on fairness among connections

two species with the presence of the noises and some other deterministic periodical drive force present in the ecosystems such as the temperature. We refer to these equations for adding self-induced oscillation to TCP Symbiosis mechanisms, by converting the noises in Equation (16) into the self-induced oscillation. We then obtain the following equation to control the congestion window size as follows:

$$\frac{d}{dt}w(t) = \epsilon\left(1 - \frac{w(t) + \gamma(K - A(t))\tau}{K\tau}\right)w(t) + mw(t)\xi(t), \tag{17}$$

where $\xi(t)$ denotes Gaussian white noise and m is a parameter to determine the degree of the self-induced oscillation. We use normal random number as white noise and set $\xi(t) = N(0, 1)$.

Since TCP Symbiosis calculates the congestion window size every time it receives ACK packets, we can see that $w(t)$ is unchanged during the interval between the latest two ACK packets. Then, we can regard $w(t)$ in $w(t)\xi(t)$ as constant. So, we have only to think the integral of $\xi(t)$ to derive the window updating mechanism based on Equation (17).

According to [30, 31], the integral of white noise is equivalent to Brownian noise. Therefore, we can regard the integral of $\xi(t)$ as $B(t)$ which satisfies below conditions.

(1) $B(0) = 0$.

(2) $B(t)$ is a real random function with independent Gaussian increments such that $B(t + h) - B(t)$ has mean zero and variance of $|h|$, and such that $B(t_2) - B(t_1)$ is independent of $B(t_4) - B(t_3)$ if the intervals $(t_1,\ t_2)$ and $(t_3,\ t_4)$ do not overlap.

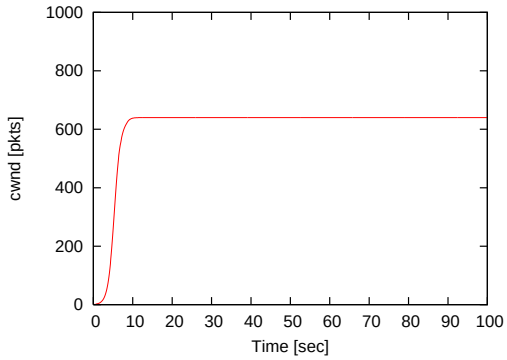
Therefore, TCP Symbiosis calculates the congestion window size according to the following equation by instead of the Equation (10).

$$w(t) = \frac{w(t)\tau f(t)\left\{K - \gamma(K - A(t))\right\}}{w(t)\left(f(t) - 1\right) + \tau\left\{K - \gamma(K - A(t))\right\}} + mw(t)B(t), \tag{18}$$

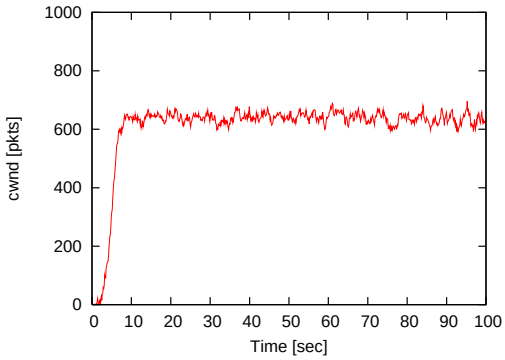
where $B(0) = 0$, $B(t_n) = B(t_{n-1}) + r_n$ at $t_n \geq t_{n-1}$, and r_n is random number which has normal distribution, $N(0, 1)$.

4.3 Performance evaluation

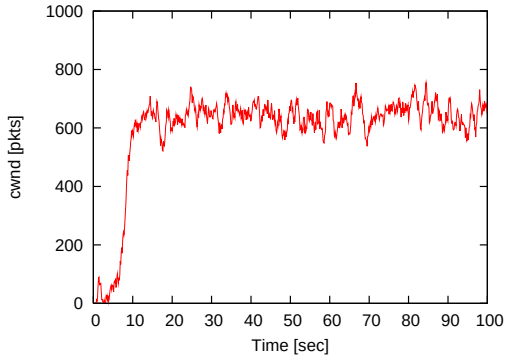
We first show the fundamental behavior of TCP Symbiosis with self-induced oscillation by generating fixed background traffic. The simulation environment is the same as in



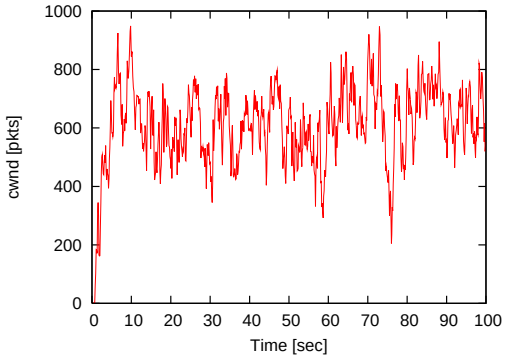
(a) m=0 (without self-induced oscillation)



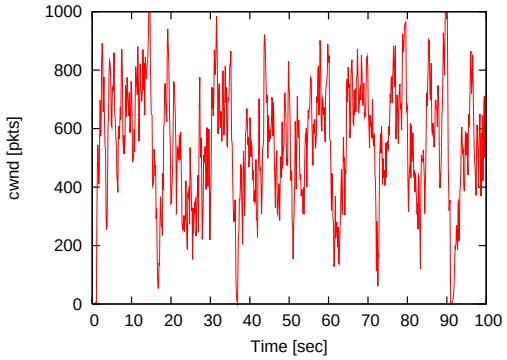
(b) m=0.05



(c) m=0.1

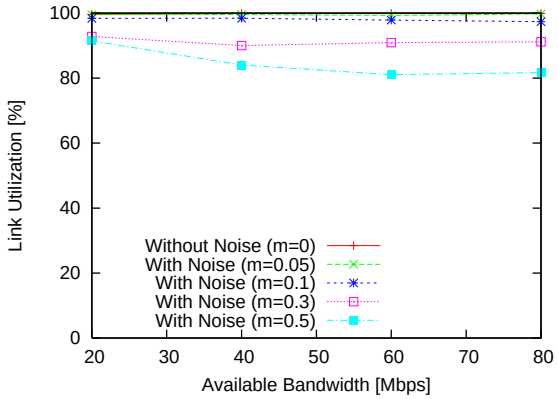


(d) m=0.3

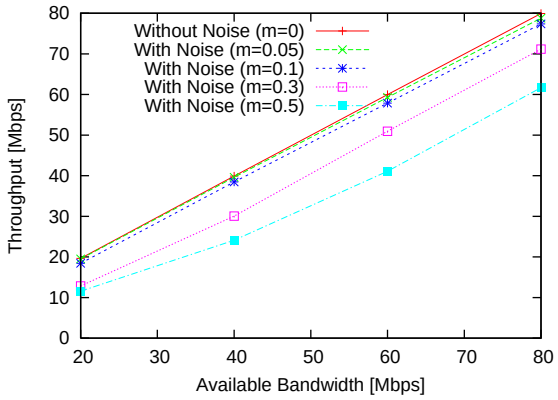


(e) m=0.5

Figure 12: Effect of the degree of self-induced oscillation



(a) Link utilization



(b) Throughput

Figure 13: Link utilization and throughput of TCP Symbiosis with self-induced oscillation

Subsection 4.1, except that N_{udp} is fixed to 4 and the available bandwidth of TCP Symbiosis connection is 60 Mbps at all time. Figure 12 plots the change in congestion window size and Figure 13 plots the link utilization and throughput at $m = 0, 0.05, 0.1, 0.3, 0.5$, respectively. The result labeled by $m = 0$ corresponds to TCP Symbiosis without self-induced oscillation as compared Figure 12(a) and Figures 12(b)-12(e), the average of the congestion window size with self-induced oscillation is close to that without the oscillation. However, from Figure 13, the link utilization and throughput are lower as the oscillation is larger. This is because the period in which the congestion window size is lower than the size without self-induced oscillation is long as the degree of the oscillation increases.

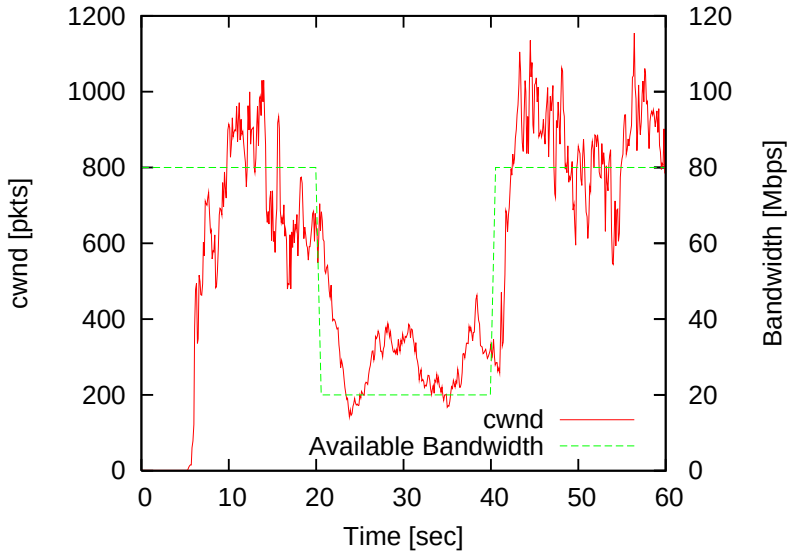
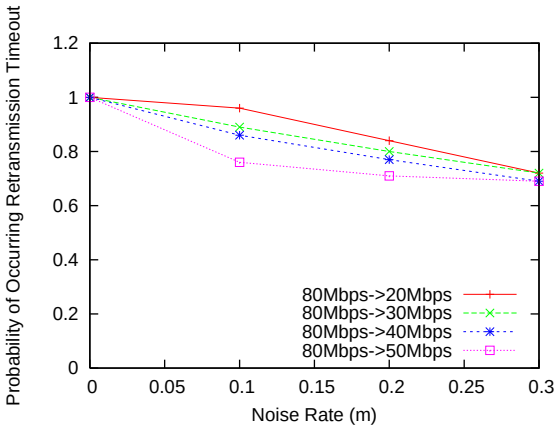


Figure 14: Change in the congestion window size with self-induced oscillation

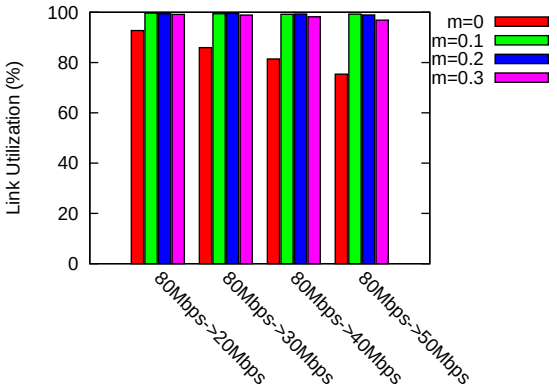
In addition, when m=0.5, the oscillation is so large that packet losses occur. This means that adding oversized oscillation to congestion window size degrades the link utilization and throughput. Then, $m < 0.5$ is reasonable for this environment.

Figure 14 shows the simulation results of the proposed method with $m = 0.3$ with the same simulation settings as in Figure 10. We can observe that the congestion window size continues fluctuating even when the available bandwidth remains stable. Note that at 20 sec, the proposed method experiences no packet loss and retransmission timeout. After the sudden increase of the available bandwidth at 40 sec, the congestion window size increases in shorter time than in Figure 10.

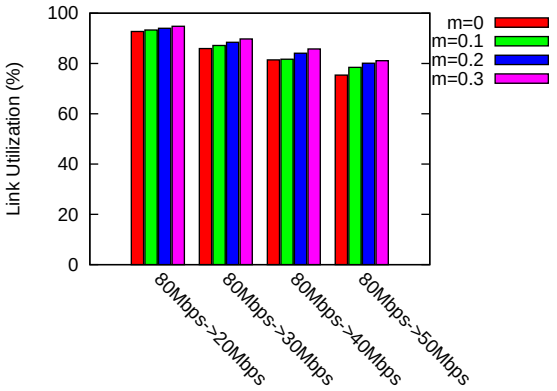
We show the effect of self-oscillation on the probability of occurring retransmission timeout. Figure 15 plots the probability of occurring retransmission timeout when available bandwidth suddenly decreases, as a function of m. In the graph, we show the results of the four cases of the decrease degree of the available bandwidth. We can observe from Figure 15(a) that by adding the oscillation to the congestion window size, we can avoid the retransmission timeout with significant probability, regardless of the degree of the bandwidth decrease. In Figure 15(b), for comparison purposes, we plots the link utilization with $m = 0$ when retransmission timeout occurs. From this figure, the avoidance of retransmission timeout largely improves the link utilization, and the improvement degree



(a) Probability of retransmission timeout

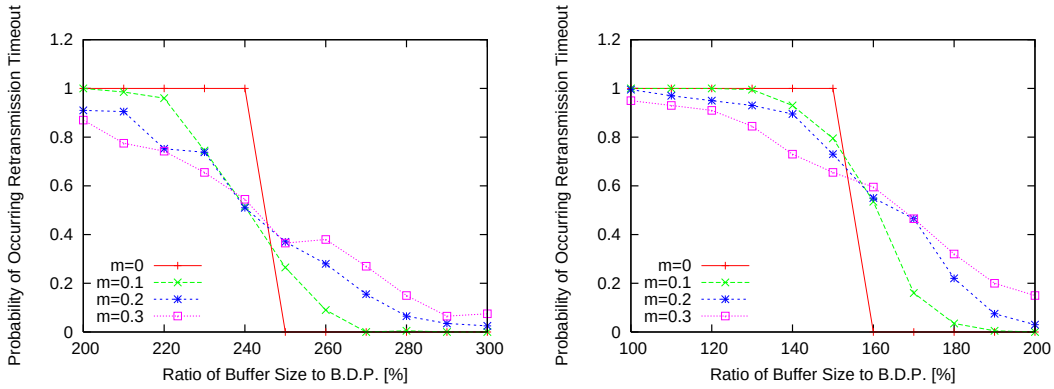


(b) Link utilization when retransmission timeout does not occur



(c) Average link utilization

Figure 15: Probability of occurring retransmission timeout and link utilization



(a) 80 Mbps→20 Mbps

(b) 80 Mbps→30 Mbps

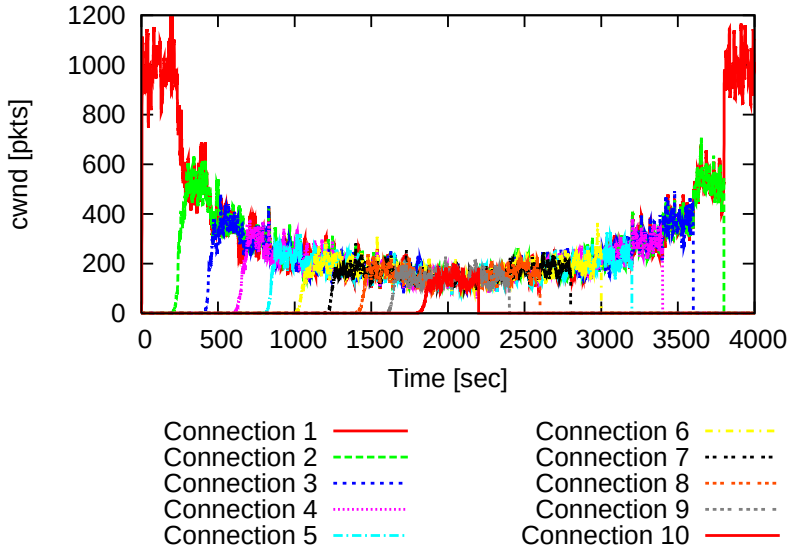
(c) 80 Mbps→40 Mbps

(d) 80 Mbps→50 Mbps

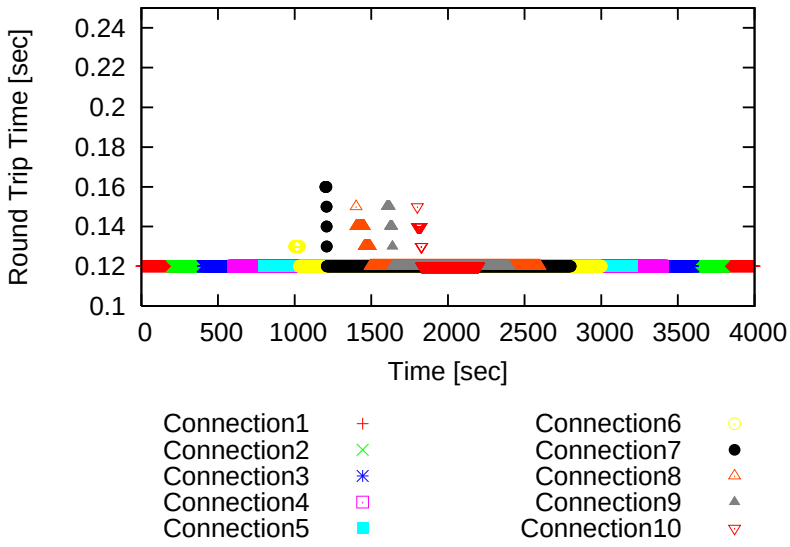
Figure 16: Effect of bottleneck link buffer size

becomes higher as the oscillation range becomes smaller. Figure 15(c) plots the average link utilization of both cases for all simulation experiments. The average link utilization becomes higher as the oscillation becomes larger, because it becomes more frequent to avoid the retransmission timeout. We also investigate of the effect of buffer size at the bottleneck link on probability of occurring retransmission timeout. Figure 16 shows the relationship between the buffer size (represented the ratio to the bandwidth-delay product of the network path) and the probability. The probability of occurring retransmission timeout decreases as the buffer size increase, regardless the degree of the oscillation. When the buffer size is small, large oscillation is more effective in decreasing the probability. On the other hand, the trend is reversed when the buffer size is too large. The reverse of the trend happens at the point where the probability in $m = 0$ changes. That is, when the buffer size is so large that the retransmission timeout does not occur, adding the oscillation works negatively. In the environment which has large bandwidth-delay-product but small buffer size, self-induced oscillation is effective to avoid the retransmission timeout.

We finally show the effect of self-oscillation on unfairness among competing connections shown in Figure 11. By adding the self-induced oscillation, we expect that the queue length continues fluctuating and all connections would obtain the correct value of the minimum RTT. Figure 17 indicates the results with self-induced oscillation through the same experiments as shown in Figure 11. From Figure 17(b), each of new connections can obtain the correct value of the minimum RTT at the beginning of the connection. As the result, the congestion window sizes for all connections keep fairness regardless of the number of connections, as shown in Figure 17(a).



(a) Change in congestion window size



(b) Change in τ

Figure 17: Improvement in fairness among connections with self-induced oscillation

5 Conclusion

In this thesis, we focused on the noises and errors of measurement in network congestion indication for TCP and proposed two methods which dealt with them appropriately. We demonstrated the effectiveness by applying them to one of TCP congestion control mechanisms, TCP Symbiosis, which utilizes the bandwidth information as congestion indication. Through the simulation experiments, we confirmed that the proposed methods can utilize the measurement noises and errors effectively and it can avoid the performance degradation based on environmental changes.

First, we proposed to control the degree of dependence on measured value of congestion indicators based on the degree of the measurement noises. By dynamic setting the dependence degree on the measurement noises, we demonstrated that the link utilization can be kept high, while the queue length remains small.

Second, we proposed adding self-induced oscillation to the data transmission rate to absorb the ill-effect of the environmental changes. We demonstrated that the proposed method can reduce the frequency of retransmission timeout with sudden environmental changes while the fundamental performance is almost unchanged. In addition, we revealed that the proposed method is effective especially in the network environment which has large bandwidth-delay-product but small buffer size.

For future work, we plan to evaluate the proposed methods by applying them to other congestion control mechanisms. Implementation experiments in the actual Internet environment are another interesting issue.

Acknowledgements

I would like to express my deepest gratitude to Professor Hirotaka Nakano of Osaka University. His comments and suggestions were invaluable during the course of my study.

Special thanks also go to Associate Professor Go Hasegawa of Osaka University. All works of this thesis also would not have been possible without his appropriate guidance and empathic advice. I am most grateful to him that he always helped me.

I am deeply grateful to Professor Masayuki Murata of Osaka University, for his excellent guidance and continuous support through my studies of this thesis. His deep insight inspired me to think not only about study but also about my future.

I would like to express my sincere appreciation to Professors Koso Murakami, Makoto Imase, and Teruo Higashino of Osaka University, for their technical guidance and cooperation.

I also appreciate to Research Assistants Masahiro Sasabe and Yoshiaki Taniguchi of Osaka University, who gave me useful comments and supports.

Finally, I want to give thanks to my friends and colleagues in the Department of Information Networking of the Graduate School of Information Science and Technology of Osaka University for their support. Our conversations and work together have greatly influenced this thesis.

References

[1] Internet trends. available at `http://www.internettrends.org/`.

[2] V. Jacobson, "Modified TCP congestion avoidance," *end2end-interest mailing list*, Apr. 1990.

[3] V. Jacobson, "Congestion avoidance and control," *ACM SIGCOMM computer communication review*, vol. 25, pp. 157–187, Jan. 1995.

[4] J. B. Postel, "Transmission control protocol," *Request for Comments 793*, Sept. 1981.

[5] M. Allman, V. Paxxon, and W. Stevens, "TCP congestion control," *Request for Comments 2581*, Apr. 1999.

[6] S. Floyd and T. Henderson, "New Reno modification to TCP's fast recovery," *Request for Comments 2582*, Apr. 1999.

[7] S. Floyd, "HighSpeed TCP for large congestion windows," *Request for Comments 3649*, Dec. 2003.

[8] T. Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," in *Proceedings of PFLDnet 2003*, Feb. 2003.

[9] D. Leith and R. Shorten, "H-TCP protocol for high-speed long distance networks," *PFLDnet 2004*, 2004.

[10] I. Rhee and L. Xu, "CUBIC: A new TCP-friendly high-speed TCP variant," in *Proceedings of PFLDnet 2005*, Feb. 2005.

[11] 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, Mar. 1995.

[12] C. Jin, D. X. Wei, and S. H. Low, "FAST TCP: motivation, architecture, algorithms, performance," in *Proceedings of IEEE INFOCOM 2004*, Mar. 2004.

[13] K. T. J. Song, Q. Zhang, and M. Sridharan, "Compound TCP: A scalable and TCP-friendly congestion control for high-speed networks," in *Proceedings of PFLDnet 2006*, Feb. 2006.

[14] H. Shimonishi, T. Hama, and T. Murase, "TCP-Adaptive Reno: Improving efficiency-friendliness tradeoffs of TCP congestion control algorithm," in *Proceedings of PFLDnet 2006*, Feb. 2006.

[15] G. Hasegawa and M. Murata, "TCP Symbiosis: Congestion control mechanisms of TCP based on Lotka-Volterra competition model," in *Proceedings of Inter-Perf 2006*, Oct. 2006.

[16] M. Kodama, G. Hasegawa, and M. Murata, "Implementation experiments of TCP Symbiosis: bio-inspired mechanisms for Internet congestion control," in *Proceedings of CQR 2008*, Apr. 2008.

[17] C. L. T. Man, G. Hasegawa, and M. Murata, "ImTCP: TCP with an inline measurement mechanism for available bandwidth," *Computer Communications Journal special issue of Monitoring and Measurements of IP Networks*, vol. 29, Issue 10, June 2006.

[18] C. L. T. Man, G. Hasegawa, and M. Murata, "A simultaneous inline measurement mechanism for capacity and available bandwidth of end-to-end network path," *IEICE Transactions on Communications*, vol. E89-B, pp. 2469–2479, Sept. 2006.

[19] The Network Simulation - NS2. available at `http://www.isi.edu/nsnam/ns/`.

[20] J. D. Murray, *Mathematical Biology I: An Introduction*. Springer Verlag Published, 2002.

[21] B. Melander, M. Bjorkman, and P. Gunninberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Proceedings of IEEE GLOBECOM 2000*, Nov. 2000.

[22] M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," in *Proceedings of ACM SIGCOMM 2002*, Aug. 2002.

[23] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell, "pathchirp: Efficient available bandwidth estimation for network paths," in *Proceedings of NLANR PAM 2003*, Apr. 2003.

[24] R. L. Carter and M. E. Crovella, "Measuring bottleneck link speed in packet-switched networks," Tech. Rep. BU-CS-96-006, Boston University Computer Science Department, Mar. 1996.

[25] C. Dovrolis, P. Ramanathan, and D. Moore, "What do packet dispersion techniques measure?," in *Proceedings of IEEE INFOCOM 2001*, Apr. 2001.

[26] L. Rizzo, "Dummynet." available at `http://info.iet.unipi.it/~luigi/ip_dummynet/`.

[27] NLANR/DAST: Iperf 1.7.0 - The TCP/UDP Bandwidth Measurement Tool. available at `http://dast.nlanr.net/Projects/Iperf/`.

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

[29] B. Spagnolo, D.Valenti, and A. Fiasconaro, "Noise in ecosystems: A short review," *Mathematical Biosciences and Engineering*, vol. 1, pp. 185–211, June 2004.

[30] E. Nelson, *Dynamical Theories of Brownian Motion*. Princeton University Press, Feb. 1967.

[31] B. B. Mandelbrot and J. W. V. Ness, "Fractional brownian motions, fractional noises and applications," *SIAM Review*, vol. 10, pp. 422–437, Oct. 1968.