PAPER

Measurement-Based Evaluation of TCP Throughput

Mika ISHIZUKA[†a)], **Masaki AIDA**[†], ***and*** **Shin-ichi KURIBAYASHI**[††], ***Members***

SUMMARY Since the TCP is the transport protocol for most Internet applications, evaluation of TCP throughput is important. In this paper, we establish a framework of evaluating TCP throughput by simple measurement. TCP throughput is generally measured by sending TCP traffic and monitoring its arrival or using data from captured packets, neither of which suits our proposal because of heavy loads and lack of scalability. While there has been much research into the analytical modeling of TCP behavior, this has not been concerned with the relationship between modeling and measurement. We thus propose a lightweight method for the evaluation of TCP throughput by associating measurement with TCP modeling. Our proposal is free from the defects of conventional methods, since measurement is performed to obtain the input parameters required to calculate TCP throughput. Numerical examples show the proposed framework's effectiveness.

key words: *TCP, throughput, active measurement*

1. Introduction

Since the TCP is the transport protocol for most Internet applications, evaluation of TCP throughput is important to estimate application-level performance. For network providers in particular, knowledge of TCP throughput is necessary for service-level management and bandwidth dimensioning. Users naturally want to know the performance of services. A way of estimating TCP throughput by simple measurement will thus be valuable.

The conventional methods of measuring TCP throughput are active measurement, based on the precise imitation of TCP behavior e.g. Treno [1] and passive measurement, based on the analysis of data from captured packets. The approaches of the former type impose heavy loads on the network, while those of the latter type require a lot of computational power for analysis of the huge amount of data from captured packets. Therefore, a new approach is required to evaluate TCP throughput by simple measurement.

Much research has been concerned with the analytical modeling of the TCP throughput [2]–[15]. In these approaches, TCP throughput is calculated with the aid of some information about the network. Some may be suitable as bases for evaluation of TCP throughput from simple measurement. However, the point of such studies is to understand TCP behavior rather than to find ways of estimating input parameters from measured values. Some are based on input parameters obtained from an analysis of data from captured packets and others involve simulation in which loss was generated by a stochastic model. We consider neither approach suitable, because the former lacks scalability and the models of the latter do not always reflect reality.

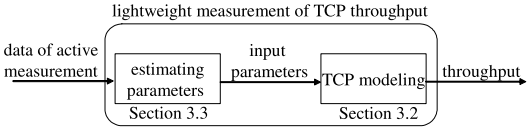

Fig. 1 Framework of our research.

We then propose a framework of realizing lightweight measurement of TCP throughput by using TCP modeling.

We explain our framework by using Fig. 1. Our framework is divided into two parts; "estimating parameters" and "TCP modeling." In the part of "estimating parameters," data obtained from active measurement are used to estimate the input parameters of TCP modeling. In the part of "TCP modeling," TCP throughput is calculated through analytical model of TCP using the input parameters obtained in the part of "estimating parameters."

Specifically, we

- select from the above-cited work, the most suitable model of TCP for the measurement-based evaluation of TCP throughput, and
- propose a lightweight traffic probe to estimate the input parameters for the model.

In addition, we modify the selected model for improved accuracy by taking measurement into account.

Note that, TCP in this paper means TCP-Reno, which is one of the most common versions of TCP. Our framework can be made applicable to TCP-Tahoe and TCP-Sack by modifying the model. On the other hand, both the model and traffic probe have to be changed for other versions of TCP, such as Vegas.

The rest of this paper is organized as follows. Section 2 is a survey of research into TCP modeling and descriptions of how we selected the most suitable model for the measurement-based evaluation of TCP throughput, and of the selected model. Section 3 introduces issues we face in making the framework and presents solutions. Section 4 discusses how we used simulation to demonstrate effectiveness of our proposal. In Sect. 5, we conclude with a brief summary.

Manuscript received December 25, 2003.
Manuscript revised April 23, 2004.

[†]The authors are with NTT Information Sharing Platform Laboratories, NTT Corporation, Musashino-shi, 180-8585 Japan.
[††]The author is with Department of Applied Physics, Faculty of Engineering, Seikei University, Musashino-shi, 180-8633 Japan.
a) E-mail: ishizuka.mika@lab.ntt.co.jp

2. TCP Modeling

In our measurement-based method for the evaluation of TCP throughput, we need to estimate the input parameters required to calculate TCP throughput. In addition, these input parameters should be obtainable through simple measurement. In this section, we start by summarizing requirements for the model and then select the most suitable model on this basis.

2.1 Requirements for TCP Modeling

- Our starting point is to focus on steady-state behavior, and in this case we can draw on much previous work [2]–[7], [9], [11], [14], [15]. Performance in the steady-state is most useful from the viewpoint of network management, since the results are not affected by short-term variations in round-trip-time and loss ratio. From the viewpoint of users, steady-state analysis lets us estimate the throughput of bulk transfer according to the FTP, which is one of the most typical TCP applications.
- Since we need to estimate the input parameters from measurable information, we have to use a model that does not require information on the network configuration [2], [3], [5]–[8], [10], [12], [13]. Similarly, we have to choose a model that does not require details of correlation or the higher-order moments of the loss generation-process [2], [3], [5], [8], [10], [12], [13], since it is not possible to obtain these values by simple measurement.
- Packet loss are correlated when drop-tail routers are used [16], [17]. In earlier work [2], [3], the effect of correlated losses was considered by using the loss-event ratio instead of the packet-loss ratio as an input parameter, where the loss-event ratio is calculated by dividing the total number of loss indications by the total number of packets sent. We think this is a feasible approach and employ it here.
- We selected an independent [2], [3], [8], [10], [15] rather than Markovian model of the loss-generation process, because the former provides more intuitive insight along with reasonable accuracy.
- As control mechanisms, the model needs to include both a fast-retransmit mechanism and a timeout mechanism [2], [5], since timeout events occur more frequently than fast-retransmit events, when the drop-tail policy is used in routers.

Since Padhye et al.'s model [2] meets the above requirements, we decided to use this as the model for measurement-based evaluation of TCP throughput.

2.2 Overview of Padhye et al.'s TCP Model [2]

In this subsection, we briefly explain the TCP model we use.

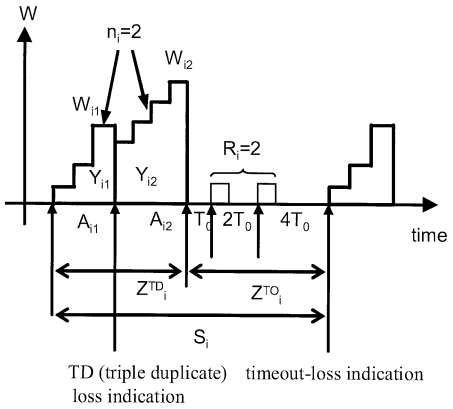

Fig. 2 Evolution of window size.

For details, please refer to Padhye et al.'s original paper [2]. Assumptions of Padhye et al.'s model [2] are summarized in Appendix A.

An "epoch" is defined as a period that starts after, or ends in, a timeout-loss indication, where a timeout-loss indication means the last timeout of a sequence of timeouts (see Fig. 2).

Let n_i be the number of loss indications in the i-th epoch. Let Z_i^{TO} be the duration of a sequence of timeouts, and Z_i^{TD} be the time interval between two consecutive timeout sequences. We then define S_i as

$$S_i = Z_i^{TD} + Z_i^{TO}.$$

Let M_i be the number of packets sent during S_i. Throughput B is then written as

$$B = \frac{\mathrm{E}[M]}{\mathrm{E}[S]}.$$

Next, let Y_{ij} be the number of packets sent in the j-th TD period of interval Z_i^{TD} (TDP_{ij}) and A_{ij} be the duration of this period. Furthermore, R_i denotes the number of packets sent during the timeout sequence Z_i^{TO}.

Then, we have

$$\mathrm{E}[M] = \mathrm{E}\left[\sum_{j=1}^{n_i} Y_{ij}\right] + \mathrm{E}[R],$$

$$\mathrm{E}[S] = \mathrm{E}\left[\sum_{j=1}^{n_i} A_{ij}\right] + \mathrm{E}[Z^{TO}].$$

Assuming that $\{n_i\}$ is an i.i.d. sequence of random variables that is independent of $\{Y_{ij}\}$ and $\{A_{ij}\}$, we have

$$\mathrm{E}[M] = \mathrm{E}[Y]\,\mathrm{E}[n] + \mathrm{E}[R],$$
$$\mathrm{E}[S] = \mathrm{E}[A]\,\mathrm{E}[n] + \mathrm{E}[Z^{TO}].$$

If Q denotes the probability that a loss indication is a timeout (timeout-probability), we have $Q = 1/\mathrm{E}[n]$. TCP throughput is then written as

$$B = \frac{\mathrm{E}[Y] + Q\mathrm{E}[R]}{\mathrm{E}[A] + Q\mathrm{E}[Z^{TO}]}. \quad (1)$$

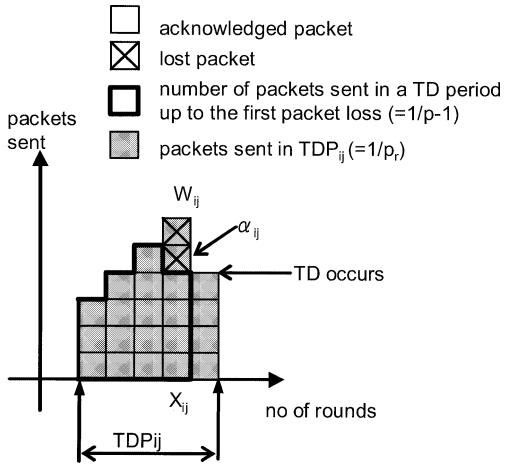

Fig. 3 Packets sent during a TD period.

Here, consider the TD period in Fig. 3. The "round" used on one axis of Fig. 3 is defined as a period starting with the back-to-back transmission of W packets, where W is the current congestion-window size. Let X_{ij} be the number of rounds in the period, W_{ij} be the window size at the end of the period, and α_{ij} be the first packet lost in the period. In addition, we define p as the probability that a packet is lost, given that it is either the first packet in its round or the preceding packets in the same round were not lost; rtt is the average round-trip-time; and T_0 is the average retransmission timeout when a timeout is the first timeout of a timeout sequence (initial retransmission timeout).

Using these variables, the parameters required to calculate (1) are expressed as follows.

$$\mathrm{E}[Y] = \mathrm{E}[\alpha] + \mathrm{E}[W] - 1 = \frac{1-p}{p} + \mathrm{E}[W], \tag{2}$$

$$Q \approx \min(1, 3/\mathrm{E}[W]), \tag{3}$$

$$\mathrm{E}[A] = (\mathrm{E}[X] + 1)rtt, \tag{4}$$

$$\mathrm{E}[X] = \frac{b\mathrm{E}[W]}{2}, \tag{5}$$

$$\mathrm{E}[Z^{TO}] = T_0 \frac{1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6}{1-p}, \tag{6}$$

$$\mathrm{E}[R] = \frac{1}{1-p}, \tag{7}$$

$$\mathrm{E}[W] = \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2} \approx \sqrt{\frac{8}{3bp}}. \tag{8}$$

In (8), b is the inverse of ack frequency, that is, the TCP sends one acknowledgement for every b consecutive packets received. Note that most implementations of TCP use two as the value of b.

3. Proposal for Measurement-Based Evaluation of TCP Throughput

3.1 Issues for Measurement-Based Evaluation of TCP Throughput

In the previous section, we employed Padhye et al.'s model [2] to construct formulae for the measurement-based evaluation of TCP throughput. However, the original model is not optimal for this purpose, since its aim is the modeling of the TCP for better understanding of TCP characteristics. In this subsection, we describe issues for our framework of measurement-based evaluation of TCP throughput.

The first issue is to do with the model. We might improve the model's accuracy by taking measurement into account. In fact, we found that Q and $\mathrm{E}[Y]$ should be modified to raise the accuracy of the model. In Sect. 3.2, we describe how to modify the calculation of these parameters.

The second issue is about a way to estimate the input parameters. In [2], the input parameters for the calculation of TCP throughput are obtained by using data from captured packets. This approach is not suitable for the measurement-based evaluation of TCP throughput, because it lacks scalability. That is, the volume of captured data is huge and its analysis requires too much computational power. Therefore, we need a way to estimate the input parameters based on simple measurement. We describe how this is done in Sect. 3.3.

3.2 Modifying the Original Model

In this subsection, we describe our modifications to Q and $\mathrm{E}[Y]$ in the original model [2].

In the original model, p is the only information available for the calculation of Q. As a result, derivation of Q requires the following assumption: if a packet is lost, all of the following packets are lost until the end of the current round. However, we found in many studies through simulation that this assumption is not always true and that Q can be better estimated by utilizing measurable information other than p.

We propose a new formula for calculating Q.

Packets are retransmitted by timeout events when fast retransmits cannot recover all the packet losses in a window. Fast retransmit cannot recover packet losses in a window when two or more packets in the window are discarded.

In this paper, we consider two typical cases when two or more packets in a window are discarded.

- two or more packets in a window are consecutively discarded, or
- discarding of consecutive packets does not occur but the next packet-loss occurs in the same window as the first packet-loss when the window size is less than ten ($\mathrm{E}[W] < 10$).

  Note that the condition of $\mathrm{E}[W] < 10$ is added to take it into account the case that a timeout event does not occur even when two packets in a window are lost (details

are given by pp.491–492 in Kumar [5] and Appendix B).

Considering these conditions, we define a new form of timeout-probability, Q', calculated as

$$Q' = P_1 + 1_{\{\mathrm{E}[W]<10\}}(1 - P_1)P\{s < \mathrm{E}[W] - 1\}. \quad (9)$$

Here,

P_1
$= P\{(n+1)$-th packet is lost$|n$-th packet is lost$\}$,
$s :=$ the number of consecutive packets not being lost
$P\{s \geq j\}$,
$= P\{j$ or more consecutive packets not being lost$\}$.

The first term of equation (9) corresponds to the first case and the second term to the second.

In addition, we found the accuracy of $\mathrm{E}[Y]$ is reduced by use of the loss-event ratio p_r instead of p. However, since p_r is far easier to measure than p, it is convenient to use p_r as an approximation for p. In fact, p was replaced by p_r in Padhye et al.'s experimental study [2].

Therefore, $Y_e = \mathrm{E}[Y]$ should be changed to Y'_e in order to mitigate the effect of this approximation. Dividing the total number of loss indications by the total number of packets sent yields the loss-event ratio p_r (see Fig. 3). By definition,

$$Y'_e = \frac{1}{p_r}. \quad (10)$$

Substituting (9) and (10) for (3) and (2) respectively, we define the following new formula for the calculation of TCP throughput:

$$B = \frac{Y'_e + Q'\mathrm{E}[R]}{\mathrm{E}[A] + Q'\mathrm{E}[Z^{TO}]}. \quad (11)$$

In the following discussion and evaluation, (11) is used instead of (1).

3.3 Using a Pseudo-TCP Probe in Parameter Estimation

In this subsection, as a solution to the second issue in Sect. 3.1, we propose a way to estimate the input parameters for (11) by simple measurement.

Inspection of (4)–(10) indicates that we need to estimate the following parameters:

(a) P_1 and $P\{s \geq j\}$ to calculate timeout-probability (Q'),
(b) loss-event ratio (p_r),
(c) average round-trip-time (rtt), and
(d) average initial retransmission timeout (T_0).

for calculation of the throughput.

As for (a) and (b), P_1, $P\{s \geq j\}$, and p_r evidently depend on the packet-sending rate. On the other hand, it is well known that the sending rate of the TCP is strongly affected by the ack-receiving rate. We need to take this characteristic into account.

As for (c) and (d), we have to take account of the facts that round-trip-time measurement in TCP is done once per window and that T_0 is calculated by using the measured round-trip-time.

We meet the above requirements by using a pseudo-TCP probe to estimate these parameters.

3.3.1 Behavior of the Pseudo-TCP Probe

Here, we briefly describe the pseudo-TCP probe. The pseudo-TCP probe has a congestion-control mechanism similar to that of the TCP but requires far less traffic, since the pseudo-TCP probe captures only those TCP characteristics that are essential to estimation of the input parameters.

Details of the pseudo-TCP probe are explained below. Note that the pseudo-TCP probe can be implemented by extending the ping or TCP protocol.

- A source-end system sends probe packets with sequence numbers (SNs).
- When a destination-end system receives a probe packet, it sends an ack-probe back to the source-end system. The ack-probe has an ack number, which is same as the SN of the probe packet it acknowledges.
  To take the delayed ack mechanism of the TCP into account, the destination-end system sends ack probes every b probe-packets (b is the inverse of the ack frequency of the TCP). The destination-end system, however, sends delayed ack probes immediately when delayed-ack timer is expired or when it receives out-of-order probe-packets.
- Let r_n be the ack number that is expected next. Here, the ack-probe with this ack number is called the expected ack-probe. If the source-end system receives this ack-probe, it adds one to r_n. If the source-end system receives an ack-probe whose ack-number is not r_n, it discards the ack-probe.
- The source-end system only sends probe packets with SN less than $r_n + w$, where w is the congestion-window size for the pseudo-TCP probe. When the source-end system receives the expected ack-probe, the congestion-window size of the pseudo-TCP probe is increased as follows:
  $$w := \min(w + 1/w, W). \quad (12)$$
  In (12), W is the maximum window size for the pseudo-TCP probe.
- If the source-end system does not receive the expected ack-probe within the retransmission timeout period after receiving the previous expected ack-probe, then the packet whose SN is r_n is considered lost. The source-end system then sets the congestion-window size to 1 and sends one new packet. This simulates the timeout mechanism of the TCP, although the retransmission timeout, T, for the pseudo-TCP probe is constant.

The pseudo-TCP probe corresponds to TCP without a fast-retransmit mechanism, with a slow-start threshold of 1 and

with constant retransmission timeout.

We can reduce the volume of probe traffic by appropriate W, T, and packet size. In the remainder of this section, we describe how we estimate the input parameters by using the pseudo-TCP probe.

3.3.2 Estimation of Timeout-Probability

By using the history of received ack-probes, we can obtain $P_1 = P\{(n+1)\text{-th packet is lost}|n\text{-th packet is lost}\}$, and $P\{s \geq j\} = P\{j \text{ or more consecutive packets are not lost}\}$.

However, we use $\hat{P}_1$ instead of P_1 to calculate (9), since whether or not the first packet after timeout is discarded is independent of whether or not the last packet before timeout has been discarded. This is because of the long retransmission timeout of the pseudo-TCP probe.

$\hat{P}_1$ is calculated as follows. Let N be the number of probe packets and t_n be the time at which sending the n-th probe packet. Then

$$\begin{aligned}\hat{P}_1 &= P\{(n+1)\text{-th packet is lost} \\ &\qquad |n\text{-th packet is lost},\ t_{n+1} - t_n < T\} \\ &= \frac{\sum_{n=1}^{N} 1_{\{(n+1)\text{-th is lost},\ n\text{-th is lost},\ t_{n+1}-t_n<T\}}}{\sum_{n=1}^{N} 1_{\{n\text{-th packet is lost},\ t_{n+1}-t_n<T, n<N\}}}.\end{aligned} \tag{13}$$

3.3.3 Estimation of Loss-Event Ratio

To estimate the loss-event ratio, we need to know the average number of packets lost per loss-event.

The estimated loss-event ratio $\hat{p}_r$ can be expressed as follows:

$$\hat{p}_r = \frac{p_s}{\mathrm{E}[L]}. \tag{14}$$

Here, p_s is the packet loss ratio for the pseudo-TCP probe, and $\mathrm{E}[L]$ is the estimated number of packets lost in one loss-event.

Since $\mathrm{E}[L]$ can be approximated by the number of packets lost in a window, we estimate $\mathrm{E}[L]$ as follows:

- Firstly, the epoch number i and packet-loss counter n_l are set to 0.
- When the loss of a packet is detected, one is added to the packet-loss counter n_l.
- When the interval from the last expected ack-probe is greater than the retransmission-timeout period, the number of packets lost in the i-th epoch l_i is set to n_l and n_l is reset to 0. One is then added to the epoch number i.
- $E[L]$ is given as the average of l_i.

Note that the estimated value of p_r is strongly dependent on the maximum window size for the pseudo-TCP probe.

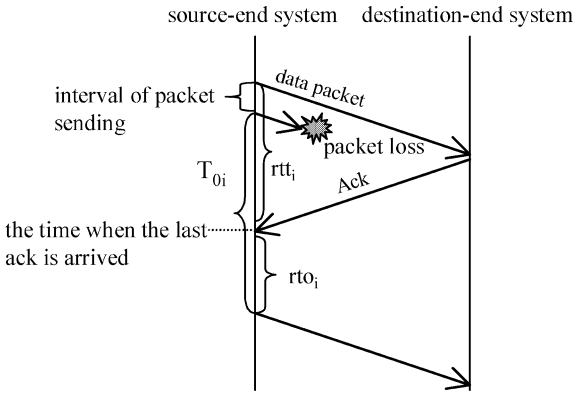

Fig. 4 Calculation of retransmission timeout.

The setting of this maximum window size is examined in Sect. 4.3.

3.3.4 Estimation of Round-Trip-Time and Initial Retransmission Timeout

Since TCP stipulates measurement of the round-trip-time once per window, we measure the round-trip-time of the pseudo-TCP probe once per window. Let rtt_i be the i-th value of the measured round-trip-time. We then take the average of rtt_i as the estimated average round-trip-time $\widehat{rtt}$.

To estimate the average initial retransmission timeout T_0, we use the measured round-trip-time rtt_i to evaluate the formula for retransmission timeout [18]. In this paper, we assume the TCP implementation of BSD 4.4 [19]. According to Stevens [18], the i-th value of retransmission timeout rto_i is given as follows:

$$rto_i = srtt_i + 4v_i, \text{ where}$$

$$srtt_i = \frac{rtt_i + 7srtt_{i-1}}{8} \text{ and} \tag{15}$$

$$v_i = \frac{|rtt_i - srtt_{i-1}| + 3v_i}{4}. \tag{16}$$

Moreover, in BSD 4.4, timeout occurs when a new ack is not received within the retransmission-timeout period after reception of the last ack (see Fig. 4). Therefore, the i-th value of the estimated retransmission timeout $\widehat{T}_{0i}$ is

$$\begin{aligned}\widehat{T}_{0i} &= rto_i + rtt_i - \text{interval of packet sending} \\ &\approx rto_i + rtt_i.\end{aligned}$$

The estimated value of average initial retransmission timeout, $\widehat{T}_0$, is given as the average of $\widehat{T}_{0i}$.

4. Simulation-Based Study

In this section, we examine the validity of our proposal by simulating its operation on ns2 [20].

4.1 Simulation Settings

The network configuration is shown in Figs. 5 and 6. Fig-

ure 5 has a single bottleneck link and Fig. 6 has two. Considering that the number of bottleneck links may have effect on the behavior of the pseudo-TCP probe, we use these two models.

In both figures, the TCP connection for measurement (TCP sender) has infinite data to transmit, and sends from left to right. The pseudo-TCP probe sender also sends probe-packets from left to right.

In Fig. 5, background traffic is offered by 160 end systems, which transmit on-off TCP traffic. Half of them (bg#1–80) send data from left to right and the other half (bg#81–160) send from right to left.

In Fig. 6, there are four groups of end systems. Groups #1 and #4 consist of 20 end systems and Groups #2 and #3 consist of 10 end systems. All the source end systems transmit on-off TCP traffic. Source end systems from Group #4 send data from right to left and the other source end systems send from left to right. TCP connections form Groups #1 and #4 pass through the link #1 and #2, those from Group #2 pass through only the link #1, and those from Group #3 pass through only the link #2.

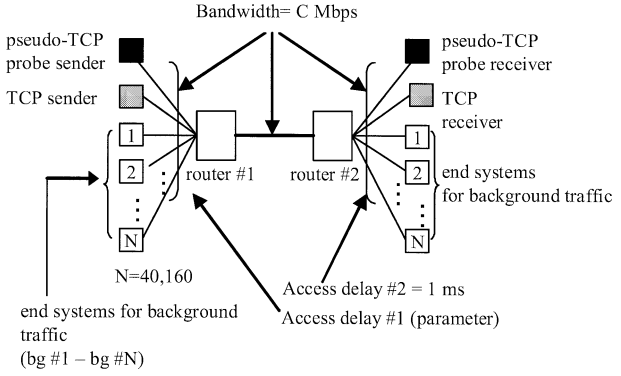

Fig. 5 Network configuration (Scenario #1–3).

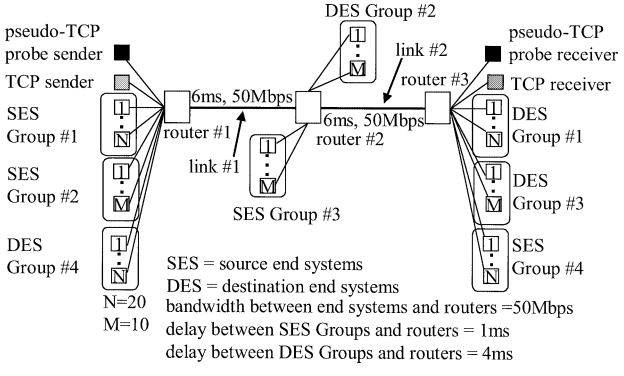

Fig. 6 Network configuration (Scenario #4).

Since TCP throughput is affected by round-trip-time and loss-event ratio, we chose simulation settings to realize various values for these parameters. To change the delay setting, we use three simulation scenarios, with the settings shown in Table 1. Note that we change the setting for bandwidth (C in Fig. 5) in response to changes to the delay setting, in order to meet Padhye et al.'s assumption [2] that the throughput of a TCP connection does not often reach the access bandwidth. Since the loss-event ratio is affected by the load through background traffic, we adjust this parameter by changing the mean burst interval of the background traffic. The settings for the background traffic are shown in Tables 2 and 3. For each scenario, we choose five values of α for the expressions in Tables 2 and 3, i.e. 0.45, 0.65, 0.8, 1.0 and 10.

The output queue length from router #1 to router #2 is set to 100 (packets); for the other queues, lengths great enough that packet loss does not occur are selected.

Packet size is set to 1500 (bytes) for all connections. Since one basis for the original model [2] was the assumption that TCP throughput is not limited by the advertised window size, we use a large enough advertised-window size so that this assumption is met. Minimum retransmission timeout of TCP is set to 0.05 (s). As for the pseudo-TCP probe, as long as there is no notice, the maximum window

Table 2 Background-traffic patterns (Scenario #1–3).

bg no.	Mean burst length [dist.]	Mean burst interval [dist.]	Shape parameter
#1–20	1 (Mbytes) [exp.]	$380/(\alpha C)$ (s) [exp.]	not defined
#21–40	50 (kbytes) [exp.]	$110/(\alpha C)$ (s) [exp.]	not defined
#41–60	1 (Mbytes) [Pareto]	$380/(\alpha C)$ (s) [Pareto]	1.5
#61–80	100 (kbytes) [Pareto]	$110/(\alpha C)$ (s) [Pareto]	1.5
#81–100	1.3 (Mbytes) [exp.]	$120/(\alpha C)$ (s) [exp.]	not defined
#101–120	2 (Mbytes) [exp.]	$120/(\alpha C)$ (s) [exp.]	not defined
#121–140	1 (Mbytes) [Pareto]	$160/C$ (s) [Pareto]	1.5
#141–160	200 (kbytes) [Pareto]	$160/C$ (s) [Pareto]	1.5

Table 1 Simulation setting (Secnario #1–3).

item		Scenario #1 (short delay)	Scenario #2 (middle delay)	Scenario #3 (long delay)
bandwidth C (Mbps)		10	50	100
delay between routers #1 and #2		30	6	2
access delay #1	TCP, psuedo TCP probe	25	4	1
	#bg1-40	25	4	1
	#bg41-120	1	40	40
	#bg121-160	1	1	1

Table 3 Background-traffic patterns (Scenario #4).

Group no.	Mean burst length [dist.]	Mean burst interval [dist.]	Shape parameter
#1	400 (kbytes) [exp.]	$4.0/\alpha$ (s) [exp.]	not defined
#2	300 (kbytes) [Pareto]	$2.7/\alpha$ (s) [Pareto]	1.5
#3	500 (kbytes) [exp.]	$3.0/\alpha$ (s) [exp.]	not defined
#4	1 (Mbytes) [pareto]	5 (s) [exp.]	not defined

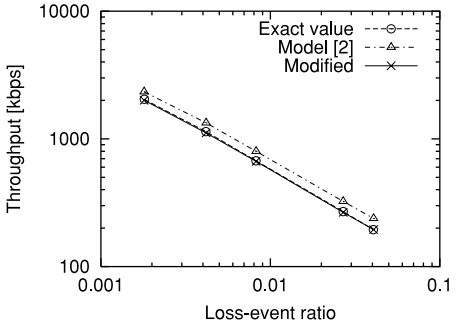

Fig. 7 Effect of modification (scenario #1).

size is set to 9 (packets) and the timeout value to 6 (s). Validity of this choice is examined by simulation study in 4.3 and 4.4.

The inverse of ack frequency b is set to 1 for all TCP connections and the pseudo-TCP probe, since we consider that evaluation when $b = 1$ is enough to show the validity of the pseudo-TCP probe and that other cases are extensions of the case where $b = 1$.

In the following simulation, throughput means the number of bytes transmitted per unit time. In addition, when the loss of consecutive packets within a window leads to both a fast retransmit and a timeout, this is counted as a single timeout (the justification for this is given in Appendix C).

4.2 Simulation Results

To show validity of our proposal, we examine both

- the accuracy of our TCP model and
- the accuracy of the estimated input parameters

in this section (see Fig. 1).

We started by examining the effect of the modification proposed in Sect. 3.2. The throughput calculated by the new formula (11) is shown in Figs. 7–10. For comparison, the throughput as calculated by the original formula (1) is also shown in these figures. The input parameters used to produce Figs. 7–10 were obtained by using data on captured packets from the TCP sender (not from the pseudo-TCP probe), since we want to see only the effect of modification.

Modified modeling yields better estimates than the original formula in all regions.

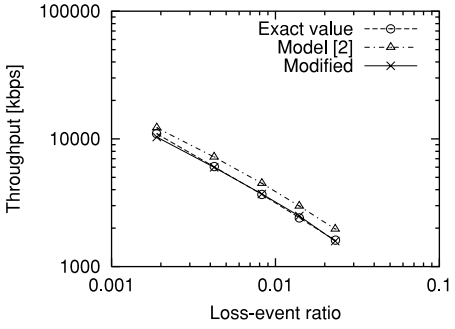

Fig. 8 Effect of modification (scenario #2).

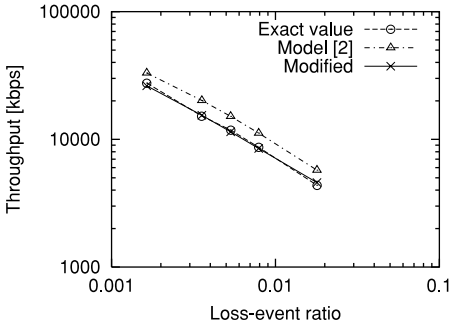

Fig. 9 Effect of modification (scenario #3).

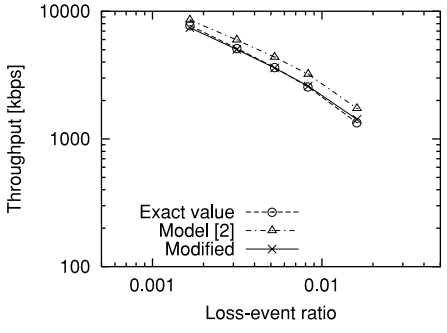

Fig. 10 Effect of modification (scenario #4).

Next, we examine the validity of throughput estimates obtained by using the pseudo-TCP probe. The calculated throughput when all of the input parameters are estimated from results for the pseudo-TCP probe is shown in Figs. 11–14. The estimations of throughput are reasonably accurate for all scenarios. However, estimated throughput is a little lower than the exact value when the loss-event ratio is below 0.005.

To examine why this happens, the throughput as calculated with only p_r replaced by the estimated value, $\hat{p}_r$, is shown in Figs. 15–18 (note that the other input parameters are obtained by using data on captured packets from the TCP sender). A small discrepancy between the estimated and exact throughput is again seen, when the loss-event ratio is below 0.005. This means that inaccuracy of the estimated loss-event ratio is the main reason for the discrepancy in Figs. 11–14.

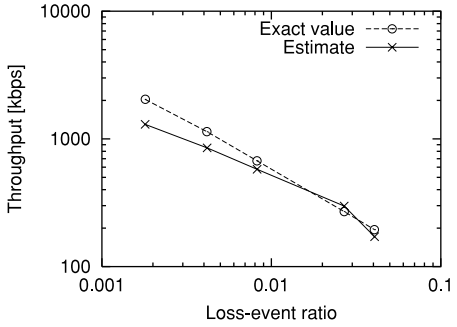

Fig. 11 Throughput as estimated by using the pseudo-TCP probe (scenario #1).

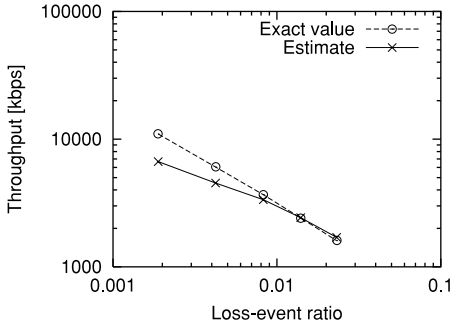

Fig. 12 Throughput as estimated by using the pseudo-TCP probe (scenario #2).

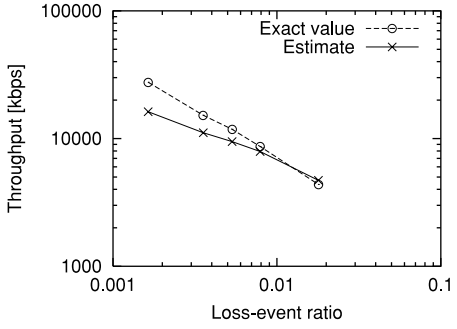

Fig. 13 Throughput as estimated by using the pseudo-TCP probe (scenario #3).

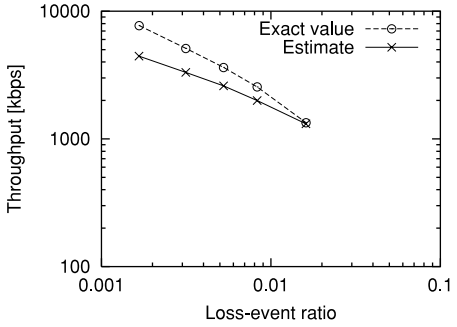

Fig. 14 Throughput as estimated by using the pseudo-TCP probe (scenario #4).

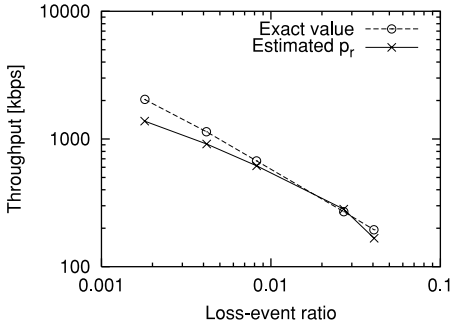

Fig. 15 Throughput as calculated by using the estimated value of p_r (scenario #1).

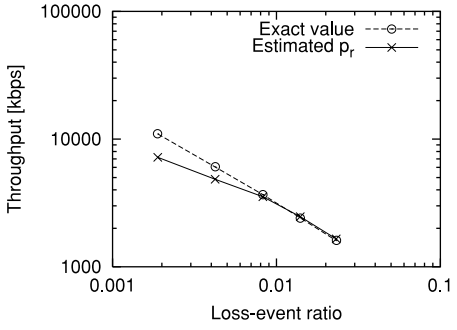

Fig. 16 Throughput as calculated by using the estimated value of p_r (scenario #2).

The reason that the accuracy of the estimate becomes worse in the region of lower loss-event ratio is explained as follows:

When loss-event ratio is low, difference between the expectation of congestion-window size when loss-event occurs (E[W]) and the maximum window size for the pseudo-TCP probe becomes large. This large difference lowers the accuracy, because this large difference means that pseudo-TCP probe cannot send enough volume of traffic to imitate characteristics of TCP traffic. Since E[W] depends only on loss-event ratio, we can say that loss-event ratio and the maximum window size for the pseudo-TCP probe determine the accuracy of estimated p_r.

While it is true that increasing the maximum window size for the pseudo-TCP probe improves the accuracy, it also increases the load on the network. Therefore, we need to identify appropriate settings of maximum window size for the pseudo-TCP probe.

4.3 Settings of the Maximum Window Size

In this section, we examine how to set the maximum window size for the pseudo-TCP probe.

To see the effect of a large maximum window, we obtained results for a maximum window size of 16; these results are given in Figs. 19–22. When we compare Figs. 11–14 and Figs. 19–22, we see that the accuracy when the loss-event ratio is below 0.005 is improved by increasing the maximum window size.

To further examine how to set the maximum window

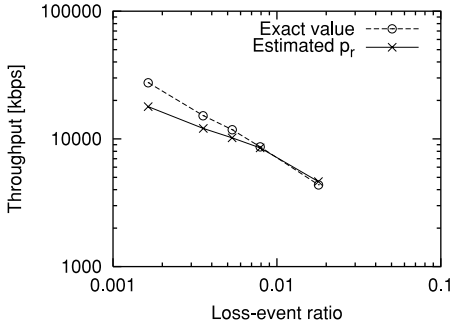

Fig. 17 Throughput as calculated by using the estimated value of p_r (scenario #3).

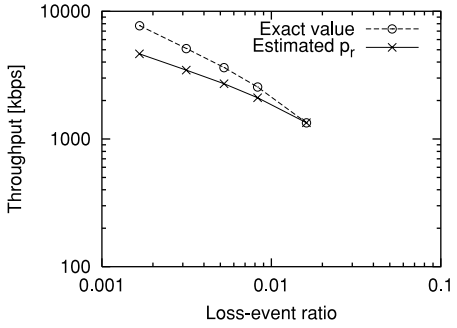

Fig. 18 Throughput as calculated by using the estimated value of p_r (scenario #4).

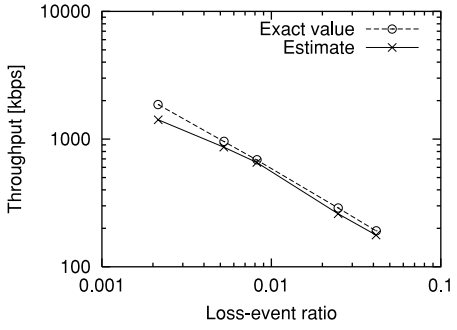

Fig. 19 Throughput as estimated by using the pseudo-TCP probe (W=16, scenario #1).

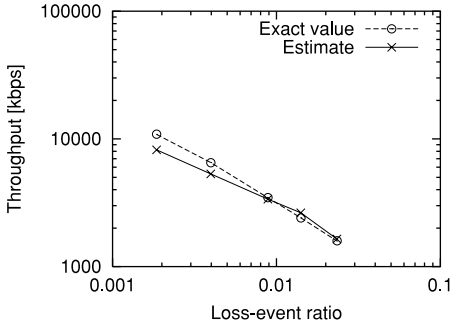

Fig. 20 Throughput as estimated by using the pseudo-TCP probe (W=16, scenario #2).

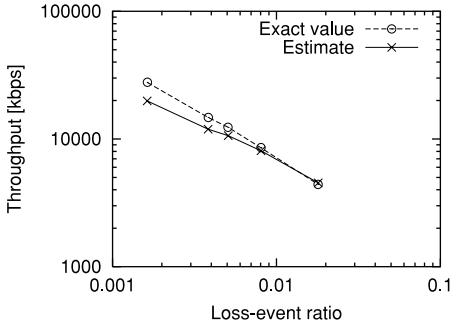

Fig. 21 Throughput as estimated by using the pseudo-TCP probe (W=16, scenario #3).

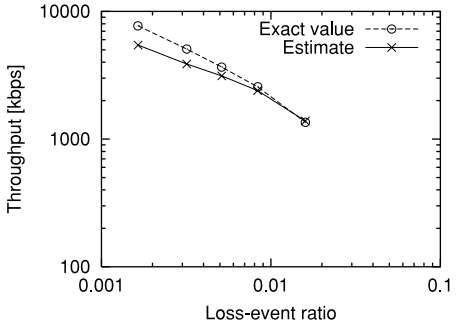

Fig. 22 Throughput as estimated by using the pseudo-TCP probe (W=16, scenario #4).

size, we obtained the relationships between the estimated loss-event ratio and the exact values in Figs. 23–26. These figures indicate that results are improved by changing the maximum window size from 9 to 16 when the estimated loss-event ratio is below 0.007.

4.4 Settings of the Retransmission Timeout

In this section, we examine settings of the retransmission timeout T for the pseudo-TCP probe.

Figures 27–30 show the throughput as estimated when $T = 6$, 18, and 24 (s). In these figures, settings of the maximum window size for the pseudo-TCP probe follow the way described in Sect. 4.3.

In these figures, the retransmission timeout T does not affect the accuracy in all the scenarios. This is because stationary environment is assumed.

Therefore, we can say that any retransmission timeout for the pseudo-TCP probe is applicable in stationary environment.

4.5 Load Imposed by the Pseudo TCP-Probe

We went on to examine the number of packets being sent by the pseudo-TCP probe in this section. The ratio of the number of packets sent by the pseudo-TCP probe to the number of packets sent by the TCP sender is shown in Figs. 31 and 32. In these figures, $T = 6$ (s) is used.

In scenario #1, the number of packets sent by the pseudo-TCP probe is about a fifth of the number of packets

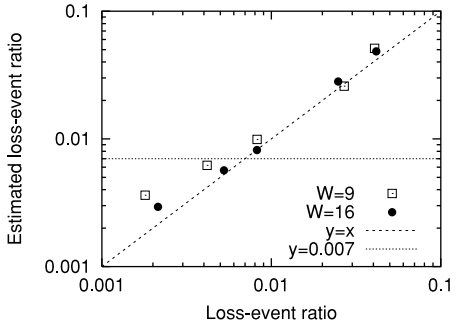

Fig. 23 Accuracy of the estimated p_r (scenario #1).

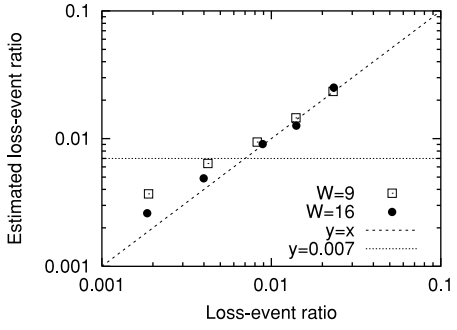

Fig. 24 Accuracy of the estimated p_r (scenario #2).

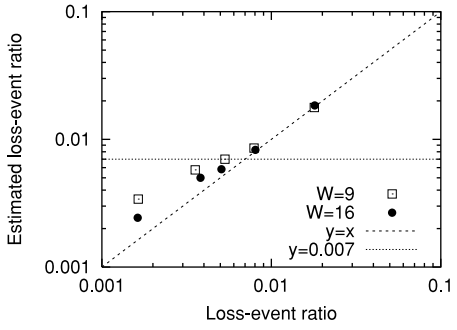

Fig. 25 Accuracy of the estimated p_r (scenario #3).

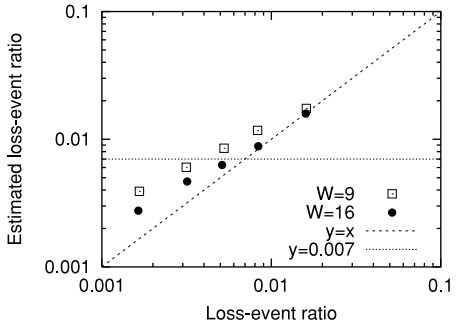

Fig. 26 Accuracy of the estimated p_r (scenario #4).

sent by the TCP sender. In the other cases, the number of packets sent by the pseudo-TCP probe is less than a tenth of the number of packets sent by the TCP sender. These results confirm that the volume of the pseudo-TCP probe traffic is

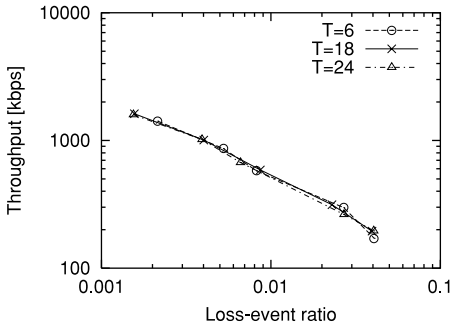

Fig. 27 Throughput as estimated by using the pseudo-TCP probe (T=6,18.24 (s), scenario #1).

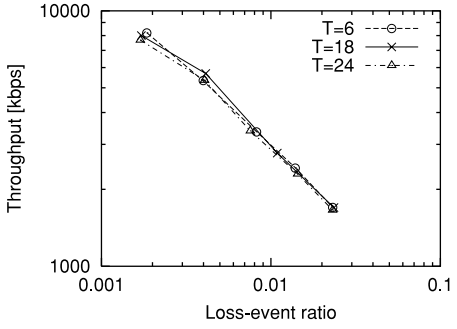

Fig. 28 Throughput as estimated by using the pseudo-TCP probe (T=6,18,24 (s), scenario #2).

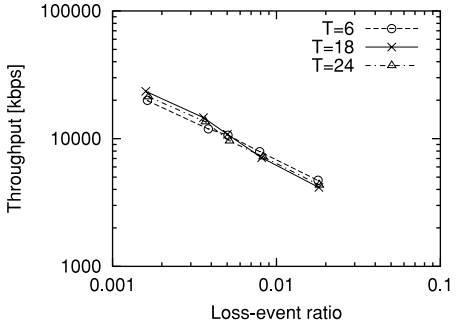

Fig. 29 Throughput as estimated by using the pseudo-TCP probe (T=6,18,24 (s), scenario #3).

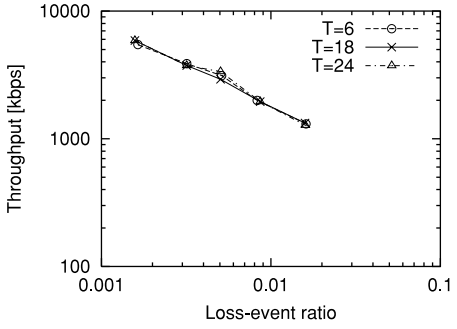

Fig. 30 Throughput as estimated by using the pseudo-TCP probe (T=6,18,24 (s), scenario #4).

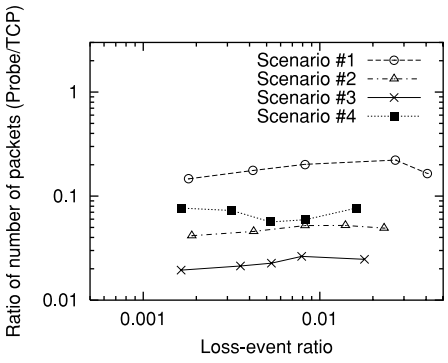

Fig. 31 Load imposed by the pseudo-TCP probe (W=9).

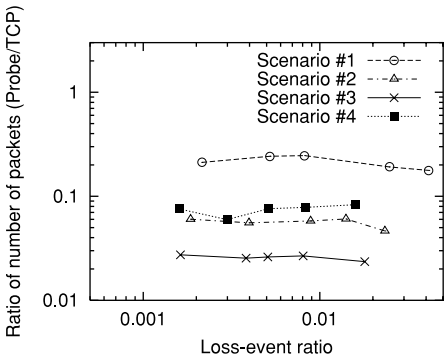

Fig. 32 Load imposed by the pseudo-TCP probe (W=16).

much smaller than that of the TCP sender.

From another point of view, the almost equal ratio of load over the whole range of loss-event ratio means that the actual number of packets sent by the pseudo-TCP probe decreases with loss-event ratio, since that by the TCP sender also decreases with loss-event ratio.

Through simulation-based study in this section, we confirm that both the accuracy of our TCP model and that of estimated input parameters are reasonable. Note that our TCP model imposes slightly more overhead of measurement than Padhye et al.'s model [2] does, though the accuracy of our TCP model is better than that of Padhye et al.'s model [2]. The more overhead in our TCP model is because our model needs more information than Padhye et al.'s model [2].

The pseudo-TCP probe is applicable even when Padhye et al.'s model [2] is used instead of our TCP model. Therefore, we can choose which TCP model to use in our framework of measurement by taking a trade-off between accuracy and overhead into account.

5. Conclusion

In this paper, we have proposed a measurement-based method for the evaluation of TCP throughput, that is, a way to evaluate TCP throughput by lightweight measurement. In our proposed method, measurement is used to obtain the input parameters required to calculate TCP throughput. Therefore, the method is free from the defects of the conventional methods.

Simulation showed that the method has good accuracy and imposed relatively low load.

We consider that the pseudo-TCP probe is applicable in other scenarios than those in this paper, since simulation results in Sect. 4.2 indicated that the accuracy of estimation does not depend on network topology but loss-event ratio and the maximum window size for the pseudo-TCP probe. Therefore, we conclude the proposed method of estimation provides sufficient accuracy under situations where Padhye et al.'s model [2] is applicable.

In addition, simulation showed that the actual load imposed by the pseudo-TCP probe decreases with loss-event ratio. This characteristic of the pseudo-TCP probe is desirable when it is compared to other ways of active measurement that emit traffic periodically.

Finally, we show some situations where TCP throughput needs to be estimated and the pseudo-TCP probe is effective.

- an alarm might be triggered when the estimated throughput falls below a threshold value,
- the pseudo-TCP probe packets are sent to each relay node to investigate the reason for degraded performance,
- the pseudo-TCP probe packets are sent to perform benchmark testing on TCP throughput, and
- the estimated throughput is used for traffic engineering (e.g. load balancing) in a MPLS network.

Acknowledgement

We would like to thank Ms. Kyoko Ashitagawa for her help with the simulations.

References

[1] M. Mathis, "TReno bulk transfer capacity," http://www.ietf.org/proceedings/99nov/I-D/draft-ietf-ippm-treno-btc-03.txt.
[2] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," ACM SIGCOMM'98, pp.303–314, 1998.
[3] M. Mathis, J. Semke, and T. Ott, "The macroscopic behavior of TCP congestion avoidance algorithm," Comput. Commun. Rev., vol.27, no.3, pp.20–26, 1997.
[4] T.V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," IEEE/ACM Trans. Netw., vol.5, no.3, pp.336–350, 1997.
[5] A. Kumar, "Comparative performance analysis of versions of TCP in a local network with a lossy link," IEEE/ACM Trans. Netw., vol.6, no.4, pp.485–498, 1998.
[6] E. Altman, K. Avrachenkov, and C. Barakat, "TCP in presence of bursty losses," Perform. Eval., vol.42, no.2-3, pp.129–147, 2000.
[7] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of TCP/IP with stationary random losses," ACM SIGCOMM'00, pp.231–242, 2000.
[8] M. Misara, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," ACM SIGCOMM'00, pp.151–160, 2000.

[9] M. Garreto, R.L. Cigno, M. Meo, and M.A. Marasan, "A detailed and accurate closed queueing network model of many interacting TCP flows," INFOCOM, pp.1706–1715, 2001.
[10] C.V. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "A control theoretic analysis of RED," INFOCOM, pp.22–26, 2001.
[11] C. Casetti and M. Meo, "A new approach to model the stationary behavior of TCP connections," INFOCOM, pp.367–375, 2000.
[12] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," INFOCOM, pp.1742–1751, 2000.
[13] T. Bonald and M. May, "Analytic evaluation of RED performance," INFOCOM, pp.1415–1424, 2000.
[14] A. Misra and T.J. Ott, "The window distribution of idealized TCP congestion avoidance with variable packet loss," INFOCOM, pp.1564–1572, 1999.
[15] E. Altman, J. Bolot, P. Nain, D. Elouadghiri, M. Erramdani, P. Brown, and D. Collange, "Performance modeling of TCP/IP in a wide-area network," Technical Report 3142, INRIA, 1997.
[16] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno, and SACK TCP," Comput. Commun. Rev., vol.26, no.3, pp.5–21, 1996.
[17] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," IEEE/ACM Trans. Netw., vol.1, no.4, pp.397–413, 1993.
[18] W.R. Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Addison-Wesley, Reading, MA, 1994.
[19] G.R. Wright and W.R. Stevens, TCP/IP Illustrated, Volume 2: The Implementation, Addison-Wesley, Reading, MA, 1995.
[20] The VINT Project, "UCB/LBNL/VINT network simulator—ns (version 2)," http://www.isi.edu/nsnam/ns

Appendix A: Assumptions of Padhye et al.'s Model [2]

We summarize assumptions that Padhye et al.'s model made.

- each epoch is independent,
- round-trip-time and p are independent,
- congestion window size does not reach the maximum widow size, and
- throughput does not reach access bandwidth.

Appendix B: TCP Behavior When There are Two Packet Losses in a Window

Figure A· 1 shows an example of the case when a timeout event does not occur even though two packets are lost in the same window. Let W be the congestion window size when packet loss occurs. Here we assume TCP already has sent W packets until the first fast-retransmit occurs. Thus, TCP receives $W-2$ duplicate acks for the first lost packet, congestion window size becomes $W' = W/2+W-2 = 3W/2-2$ when the last duplicate ack is received. Three duplicate acks for the second lost packet is needed to cause fast-retransmit for that packet. That means three more packets have to be transmitted at least after the first fast-retransmit. Since TCP has already sent W packets before the first fast-retransmit, TCP can send additional $W' - W = W/2 - 2$ packets after the first fast-retransmit. These packets cause duplicate acks for the second lost packet. Therefore, when $W' - W = W/2 - 2 \geq 3$ (that is, $W \geq 10$), the fast-retransmit for the second lost packet can occur.

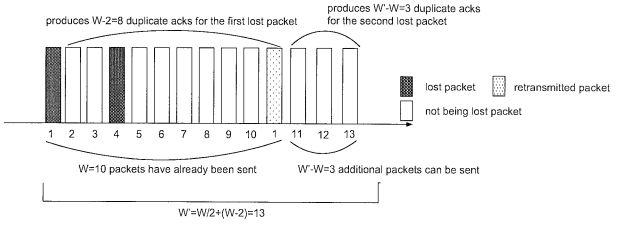

Fig. A· 1 An example of two fast-retransmits in a window.

Appendix C: TCP Performance When Consecutive Packets within a Window are Lost

This appendix describes the case when consecutive packets loss in a window leads to the combination of a fast retransmit and a timeout. When three duplicate acks for a first lost packet are received, fast retransmit of that packet is initiated. However, other lost packets are not retransmitted by this first fast retransmit. Therefore, timeout may occur for these lost packets even though fast retransmit has already been performed for the first lost packet in the same window.

In the original model [2], p means the probability that a packet is lost, given that it is either the first packet in its round or the preceding packets in the same round were not lost. The loss of consecutive packets in one window is counted as a single loss event. It is thus natural to count a fast retransmit and timeout are counted as a single timeout when this is initiated by the loss of consecutive packets.

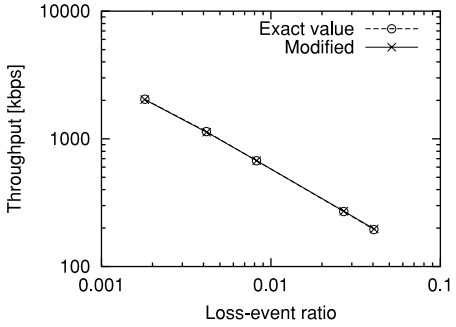

Fig. A· 2 Throughput when two loss-events in a window are counted as one (scenario #1).

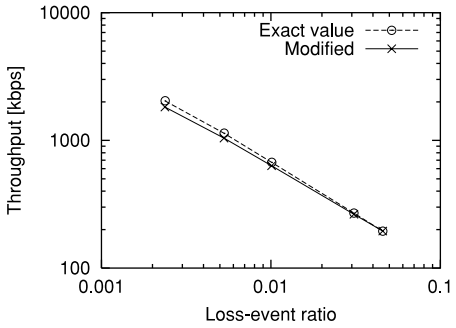

Fig. A· 3 Throughput when two loss-events in a window are counted separately (scenario #1).

For reference, we compare the throughput when such loss-events are counted as one and two loss-events in Figs. A· 2 and A· 3, using scenario #1 described in Sect. 4.1. Counting them as single loss-events leads to slightly better accuracy.

Mika Ishizuka graduated from Keio University, Tokyo, with B.E. and M.E. degrees in Control Engineering in 1992 and 1994, respectively. In 1994, she joined NTT and has been engaged in research on traffic control in computer networks. She received the IEICE Information Network Research Award in 2003. Ms. Ishizuka is a member of the Operations Research Society of Japan.

Masaki Aida received his B.S. and M.S. in Theoretical Physics from St. Paul's University, Tokyo, Japan, in 1987 and 1989, and received the Ph.D. in Telecommunication Engineering from the University of Tokyo, Japan, in 1999. Since joining NTT Laboratories in 1989, he has been mainly engaged in research on traffic issues in ATM networks and computer communication networks. From March 1998 to March 2001, he was a manager at Traffic Research Center, NTT Advanced Technology Corporation (NTT-AT). He is currently a Senior Research Engineer at NTT Information Sharing Platform Laboratories. His current interests include traffic issues in communication systems. He received IEICE's Young Investigators Award in 1996, and Information Network Research Awards in 2001, 2002, 2003, and 2004. Dr. Aida is a member of the IEEE and the Operations Research Society of Japan.

Shin-ichi Kuribayashi received his B.S., M.S., and Ph.D. in Engineering from Tohoku University, Sendai, in 1978, 1980, and 1988 respectively. He joined NTT Electrical Communications Labs in 1980. He has been engaged in the design and development of DDX and ISDN packet switching, ATM, PHS, and IMT2000 and IP-VPN systems. He researched distributed communication systems at Stanford University from December 1988 through December 1989. He participated in international standardization on ATM signaling and IMT2000 signaling protocols at the ITU-T SG11 from 1990 through 2000. Since April 2004, he has been a Professor in Department of Applied Physics, Faculty of Engineering, Seikei University.