Detection of Change in One-Way Delay for Analyzing the Path Status

Yoshito Tobe, Yosuke Tamura and Hiroto Aida

Abstract—
A new scheme for detecting a change in one-way delay is presented in this paper. The objective of detecting the change in one-way delay is to capture a congestion signal that should be reflected to a rate control of an RTP/RTCP-type unicast stream communication. Although a packet of the RTP flow may not be lost, the delay of the flow will increase due to an increase in queuing delay at the link. Therefore, only a change in delay instead of the absolute value of delay should be obtained for the rate control. Measuring the change in delay is also significant for a path that includes an error-prone wireless link. In such a wireless link, some packet losses may be irrelevant to congestion. Therefore, congestion control depending on packet loss ratio alone results in an erroneously low rate. In this paper, we describe the design and the implementation of Estimation of Skew with Reduced Samples (ESRS). We have measured changes in delay using the scheme over a path between Keio University (Japan) and Carnegie Mellon University (U.S.), and a path between Keio University and CESAT (Spain). The variation in one-way delay obtained with ESRS is shown to include the path status of the flow.

I. Introduction

End-to-end delay as well as loss of packets affects the performance of applications over the Internet. Thus, measurement of delay has an important bearing on expectations respecting the performance of communications. Our research group was confronted with the necessity of measuring the delay, in particular, the one-way delay, when we were investigating TCP-friendly schemes. In the process of evaluating TCP-friendly schemes [4], [5], [7], [10], [12], [14], [15] for RTP [11]/UDP flows, it was found that reaction only to packet losses was insufficient for the UDP flows. A conventional scheme relying merely on information about packet loss was insufficient to achieve a TCP-friendly rate control on a small network; when an RTP flow competes with a TCP flow on a link and traffic increases on that link, the RTP flow may not undergo a packet loss because the TCP flow is much quicker in reducing its rate in many cases. Although a packet of the RTP flow may not be lost, the delay of the flow will increase due to an increase in queuing delay at the link. In the Internet, UDP and TCP flows do not encounter packet losses at the same probability possibly due to drop-tail routers.

As an alternative way of detecting congestion for UDP flows, the analysis of one-way delay, or one-way transit time (OTT) [8], is explored. In the variation of the difference between a sender's timestamp and a receiver's one, an abrupt increase in OTT appears when the rate of TCP flow drops. This can be considered due to temporary increase of traffic at an intermediate router. Thus, such a difference can be a good indication of congestion.

However, since the difference contains a skew and an offset between the sender's clock and the receiver's one, a precise OTT is only obtained after the skew and the offset are removed. Since a variation is our target, a scheme for removing only the skew is explored by allowing for inaccuracy in the offset. Although several algorithms have been proposed for removing the skew [8], [6], they are not suitable for on-line calculation of the skew; they are intended to calculate the skew after samples over a long period are collected. In this paper, we propose a simple scheme for calculating the skew. When the sender transmits packets at a regular interval, inter-arrival time of packets is expected to be in a certain range. Therefore, packets outside the range are eliminated from the calculation, thus reducing the number of samples for calculation. The proposed scheme is referred to as Estimation of Skew with Reduced Samples (ESRS).

In this paper, ESRS and some examples of the application of ESRS are described. We also show how variation in OTT obtained by ESRS is related to the path status of flow using a phase plot. We have measured changes in delay using the scheme over a path between Keio University (Japan) and Carnegie Mellon University (U.S.), and a path between Keio University and CESAT (Spain). The results show that distribution of two adjacent values of difference in OTT indicates the path status of the flow.

The remainder of this paper is organized as follows. In section 2, basic observations of TCP and UDP flows are presented. In sections 3 and 4, the details of ESRS and the analysis of variation in OTT are described, respectively. Related work is described in section 5. Section 6 contains the conclusion.

II. Basic Observations of TCP and UDP Flows

In this section, we present some experimental results for one-way delay.

Many TCP-friendly algorithms to some extent rely on the indication of packet losses; the transmission rate is decreased when a receiver reports that some packets have been lost. We examine the validity of dependence on the packet loss through an experiment. In Figure 1, all hosts are installed to FreeBSD and equipped with a Pentium counter for measurement. The timestamp contained in the header uses a value converted from the Pentium counter.

Yoshito Tobe is with the Keio Research Institute at SFC, Keio University Email: tobe@mkg.sfc.keio.ac.jp. Yosuke Tamura is with the Graduate School of Media and Governance, Keio University. Email: tamura@ht.sfc.keio.ac.jp. Hiroto Aida and Hideyuki Tokuda are with the Faculty of Environmental Information, Keio University, 5322 Endo, Fujisawa, Kanagawa 252-8520, Japan Email: {haru,hxt}@ht.sfc.keio.ac.jp

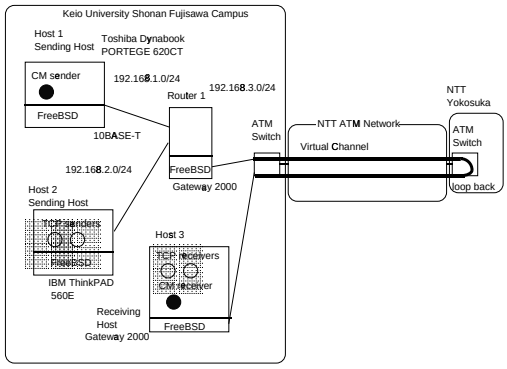

Fig. 1. Topology 1

```
 1  htgw.ht.sfc.keio.ac.jp (133.27.171.1)  0.926 ms  0.762 ms  0.753 ms
 2  gw14-v2.sfc.keio.ac.jp (133.27.16.1)  0.909 ms  0.852 ms  0.799 ms
 3  133.27.3.2 (133.27.3.2)  0.906 ms  0.912 ms  0.920 ms
 4  fw1.sfc.keio.ac.jp (133.27.1.1)  1.193 ms  1.098 ms  0.922 ms
 5  wide-keio-p2p.sfc.keio.ac.jp (133.27.1.254)  1.194 ms  1.112 ms  1.061 ms
 6  cisco12.fujisawa.wide.ad.jp (203.178.138.125)  2.037 ms  2.052 ms  1.907 ms
 7  cisco2.otemachi.wide.ad.jp (203.178.141.81)  126.576 ms  204.540 ms  7.285 ms
 8  cisco5.otemachi.wide.ad.jp (203.178.137.37)  11.104 ms  22.782 ms  6.332 ms
 9  tpr-loopback0.jp.apan.net (203.181.248.207)  8.765 ms  7.288 ms  8.010 ms
10  startap-tpr.jp.apan.net (203.181.248.241)  145.197 ms  145.241 ms  148.234 ms
11  vbns-st.startap.net (206.220.240.194)  147.148 ms  146.490 ms  147.562 ms
12  jn1-at1-0-0-12.dng.vbns.net (204.147.136.1)  157.360 ms  151.709 ms  147.225 ms
13  jn1-so4-0-0-0.nor.vbns.net (204.147.136.147)  153.924 ms  155.335 ms  153.462 ms
14  cs-atm0-0-0-15.psc.vbns.net (204.147.131.177)  157.685 ms  160.409 ms  158.865 ms
15  cmu-vbns.psc.net (198.32.224.64)  165.104 ms  158.245 ms  158.685 ms
16  MODERN.ART.CS.CMU.EDU (128.2.206.187)  179.676 ms  162.794 ms  163.723 ms
```

Fig. 2. Result of *traceroute* from Keio to CMU

As a result, it does not mean the absolute time. Host 1 and 2 send a UDP flow and two TCP flows, respectively. Let us define four flows. Flows 1, 2, and 4 are sent from Host 2 to 3, while Flow 3 is sent from Host 1 to 3. Flows 1-3 begin at time 0 s. Flow 4 begins transmission after 20 s. Thus, the number of TCP flows increases from 2 to 3 at time 20 s.

Both UDP and TCP flows transmit 512-byte packets. Router 1 schedules outgoing packets in a FIFO fashion. The allocated bandwidth to shared ATM link is 1.2 Mbps. The transmission rate of Flow 3 is set to 400 kbps. Before Flow 4 joins, Flows 1 - 3 share the bandwidth almost equally. But after Flow 4 joins, three TCP flows share (1200 - 400) kbps, which results in higher rate for Flow 3. While Flow 4 increases its rate, the packets of Flow 3 were not lost although the transmission rate was not reduced. This suggests that congestion cannot be detected by packet losses alone in such a small network. Therefore, we were motivated to examine the delay instead of loss.

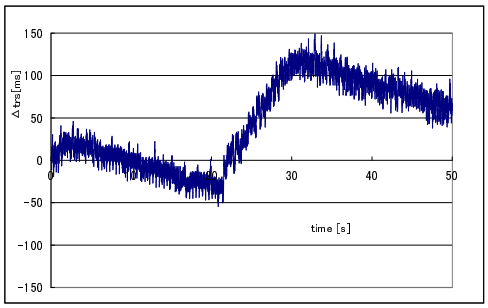

Fig. 3. Delay change (Topology 1)

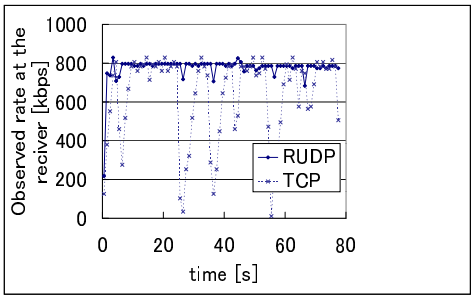

Fig. 4. Rates of TCP and UDP flows (Keio-CMU)

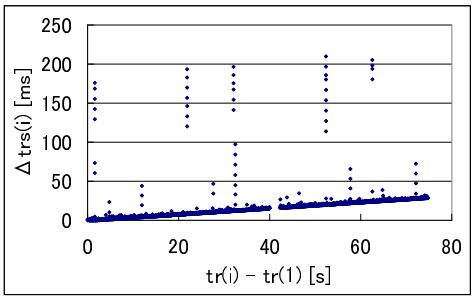

Fig. 5. Delay of UDP flow (Keio-CMU)

Let $t_s(i)$ $(i = 1, 2, 3, ...)$ and $t_r(i)$ denote the time that i-th packet of the CM flow is sent and received, respectively. Then we define a *relative delay* as follows:

$$\Delta t_{rs}(i) \equiv t_r(i) - t_s(i) - (t_r(1) - t_s(1)).$$

$\Delta t_{rs}(i)$ is referred to as a relative delay on a path because it is not an absolute delay between the sender and the receiver. Note that $t_s(i)$ and $t_r(i)$ are measured with the clock of the sender and the receiver, respectively. Hence $\Delta t_{rs}(i)$ monotonically increases or decreases depending on the difference between the precise tick of these two clocks.

Figure 3 shows $\Delta t_{rs}(i)$ of Flow 3 with respect to $t_r(i)$. Although a decrease due to the skew of clocks is observed, a change in $\Delta t_{rs}(i)$ is evident after Flow 4 begins transmission. It is expected that the change in traffic can be easily detected by monitoring $t_r(i)$ if the skew is properly removed.

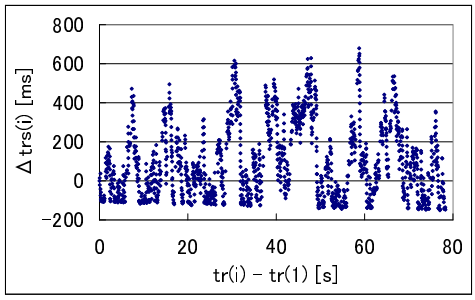

Fig. 6. Delay of UDP flow (CESAT-Keio)

A similar experiment was conducted on the path between Keio University in Japan and CMU in the U.S., for which the result of *traceroute* is shown in Figure 2. In this experiment, we created one flow of TCP and another of Rate-probing UDP (RUDP) simultaneously. The size of TCP or UDP payload was set to 1440 bytes, which is a default TCP maximum segment size over Ethernet in FreeBSD. In RUDP, rate probing using TCP is intermittently inserted [13]. Figures 4 and 5 show rates of RUDP and TCP flows measured at the receiving host and $\Delta t_{rs}(i)$ of the RUDP flow. In Figure 5, TCP probing packets were omitted and time 0 corresponds to the time at which RUDP flow begins transmitting UDP packets. Comparing these figures, the TCP rate drops when $\Delta t_{rs}(i)$ increases. This suggests that packets are likely to be dropped when $\Delta t_{rs}(i)$ increases.

Finally, a UDP flow was created on the path between CESAT in Spain and Keio. In this experiment, 64-byte UDP payload was transmitted at a regular interval with 10 kbps. As can be seen in Figure 6, phenomena of peaky increase in $\Delta t_{rs}(i)$ occur more frequently than for the Keio-CMU path and are overlapped.

From these observations, it is expected that not only an absolute value but also even a change of one-way delay has useful information about the path that a flow traverses, if the clock skew is properly removed. In the following section, we explore a way of removing the clock skew.

III. Skew Estimation

In this section, we present the details of ESRS after considering how to reduce the number of samples for calculating the skew.

A. Reducing Samples

In the following discussion, we refer to samples obtained for the Keio-CMU and CESAT-Keio paths: Sample 1 from the Keio-CMU path and Sample 2 from the CESAT-Keio path.

Since only plots near the base delay line are related to the calculation of skews in the ($t_r(i)$ - $t_r(1)$, $\Delta t_{rs}(i)$) plane, those far from the line can be removed from the calculation of the skew. In Figure 5, there are several sequences of peaky packets of the sent flow. These phenomena correspond to "probe compression" explained in the literature [1]. This compression occurs when packets of the flow accumulate behind a large number of packets of other flows at a router. As seen in Figure 7, when the flow is transmitted at a constant interval, the inter-arrival time of packets at the receiver is widened at the beginning of the compression and narrowed while in the compression. Therefore, if a transmission interval is known to the receiver, packets outside the expected range can be neglected in the calculation of the skew. With this reduction process, the reduction factor is only 1.5 % for Sample 1. However, the number of samples for Sample 2 is reduced from 1548 to 213; 86 % of samples are removed. This property of a higher reduction factor for a more unstable state is desirable.

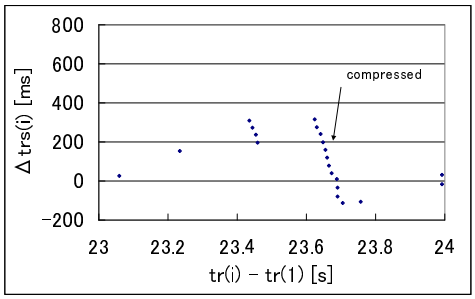

Fig. 7. Enlarged part of increase

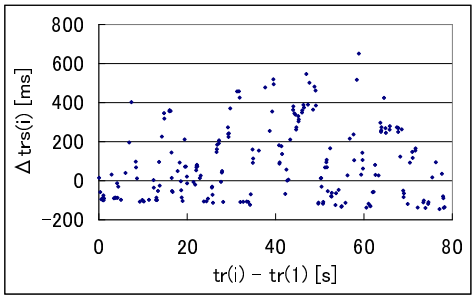

Fig. 8. Reduced samples

B. MST Scheme

Before we explain our scheme, we make some observations on the previous work. Moon *et al.* proposed the application of linear programming to remove clock skew [6]. We call their scheme the "MST scheme." Although different notation is used in the MST scheme, the basic idea of the MST scheme can be expressed in our notation and we translate the MST scheme into our terms.

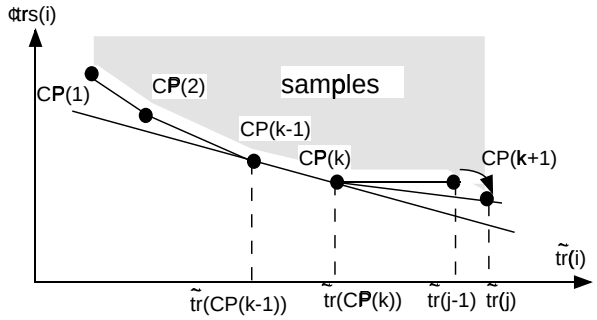

Fig. 9. Seeking and connecting convex points

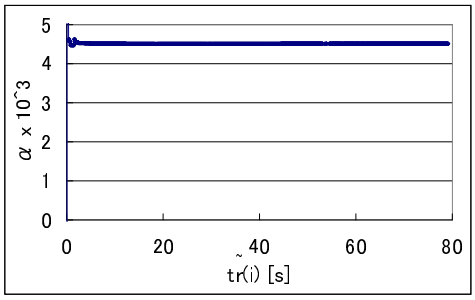

Fig. 11. Calculated slope (Keio-CMU)

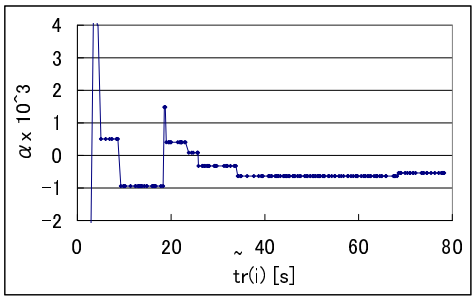

Fig. 12. Calculated slope (CESAT-Keio)

==========

Initialization:
$CP(1) = 1$; $CP(2) = 2$; $k = 2$;

On arrival of i-th ($i \geq 3$) packet:
if ($\tilde{t}_r(i)$ - $\tilde{t}_r(i-1)$ is not within an
expected range)
ignore the packet;
if ($\Delta t_{rs}(i) < f(CP(k-1), CP(k), \tilde{t}_r(i))$) {
find l such that $s(CP(l-1), CP(l)) \leq$
$s(CP(l-1), i) < s(CP(l), CP(l+1))$;
$CP(l) = i$; $k = l$;
}
else if ($CP(k) == i-1$) $CP(k+1) = i$;
else if ($\Delta t_{rs}(i) < f(CP(k), CP(k+1), \tilde{t}_r(i))$)
$CP(k+1) = i$;
if ($\tilde{t}_r(CP(k)) < \tilde{t}_r(i) - \tilde{t}_r(CP(k))$) {
$k = k + 1$;
$cp[k+1] = i$;
}
==========

Fig. 10. Pseudo-code for ESRS

Let us reformulate the skew problem. Let $\tilde{t}_s(i)$ and $\tilde{t}_r(i)$ be as follows:

$$\tilde{t}_s(i) = t_s(i) - t_s(1)$$

$$\tilde{t}_r(i) = t_r(i) - t_r(1)$$

Then, $\Delta t_{rs}(i) = \tilde{t}_r(i) - \tilde{t}_s(i)$. Let $\Delta T_{rs}(i)$ represent the OTT without the clock skew. $\Delta T_{rs}(i)$ can be expressed as follows:

$$\Delta T_{rs}(i) = \Delta t_{rs}(i) - \alpha \tilde{t}_s(i) + \beta,$$

where α and β are a coefficient related to the skew and an offset value, respectively.

Let N denote the total number of samples. If we apply the MST scheme to the above reformulated representation, the problem is described as follows.

(1) **Obtain** values of α and β
(2) such that they **minimize**

$$\sum_{i=1}^{N} (\Delta t_{rs}(i) - \alpha \tilde{t}_s(i) + \beta)$$

(3) **subject to**

$$\Delta t_{rs}(i) - \alpha \tilde{t}_s(i) + \beta \geq 0$$

C. ESRS

The objective of ESRS is to obtain the following $\Delta T_{rs}(i)$:

$$\Delta T_{rs}(i) = \Delta t_{rs}(i) - \alpha \tilde{t}_r(i) + \beta$$

$\tilde{t}_r(i)$ instead of $\tilde{t}_s(i)$ is used since it is the measured at the receiver.

We make the MST scheme the starting points of ESRS. However, we modify the scheme. First, as explained above, a packet that has an inter-arrival time outside the expected range is not taken into consideration. Second, ESRS enhances the estimate of the base delay incrementally on arrival of packets rather than calculating the skew after a certain period. Finally, ESRS is simplified such that convex points that are under $(\tilde{t}_r(i), \Delta t_{rs}(i))$ plots are searched. The resulting line of base delay is a line that traverses two

convex points. The two convex points are chosen such that the distance on the $\tilde{t}_r(i)$ axis becomes the largest.

Let $CP(k)$ represent i at the k-th convex point. The following two variables are defined.

$$s(i_1, i_2) = \frac{\Delta t_{rs}(i_2) - \Delta t_{rs}(i_1)}{\tilde{t}_r(i_2) - \tilde{t}_r(i_1)}$$

$$f(i_1, i_2, t) = s(i_1, i_2)(t - \tilde{t}_r(i_1)) + \Delta t_{rs}(i_1)$$

With these variables, a pseudo-code for ESRS is shown in Figure 10. Note that there is no variable for the total number of received packets.

With this ESRS, $\Delta T_{rs}(i)$ is obtained by approximation as follows:

$$\Delta T_{rs}(i) = \Delta t_{rs}(i) - f(CP(k-1), CP(k), \tilde{t}_r(i))$$

The value of s eventually converges to α. However, even before it converges to a certain value, $\Delta T_{rs}(i)$ is calculated. Figures 11 and 12 show how s converges for Samples 1 and 2, respectively.

MST is also applied to Samples 1 and 2 with a modification; α is a coefficient for $\tilde{t}_r(i)$. The resulting slopes for all the samples are the same as those obtained with ESRS.

IV. Path Status Analysis

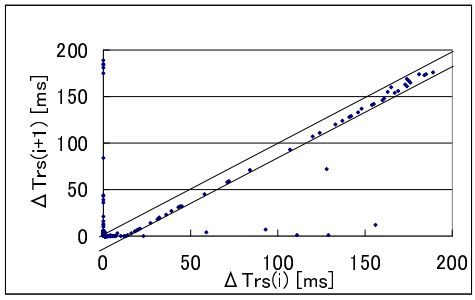

Fig. 13. Phase plot of variation in OTT for Sample 1

In this section, the patterns of OTT obtained from the samples in the previous sections are analyzed with phase plot [1]. The phase plot was originally used to produce plots of two adjacent values of RTTs. We apply the phase plot to variation in OTTs, $\Delta T_{rs}(i)$. Although $\Delta T_{rs}(i)$ is not correctly calculated when $\tilde{t}_r(i)$ is small, we here assume that the skew is properly calculated when $i = 0$ to simplify the analysis.

Figures 13 and 14 show phase plots for Samples 1 and 2. In Figure 13, sequences of plots both beside the vertical line and along the line

$$\Delta T_{rs}(i+1) = \Delta T_{rs}(i) - \Delta H(\text{constant})$$

are evident. In contrast, there are no such evident plots when peaky plots overlap one another as in Figure 14.

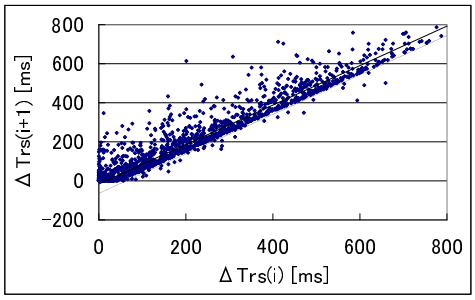

Fig. 14. Phase plot of variation in OTT for Sample 2

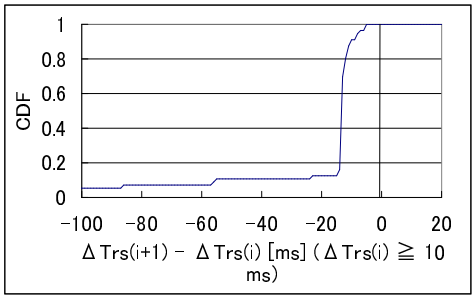

Fig. 15. CDF of $(\Delta T_{rs}(i+1) - \Delta T_{rs}(i))$ for Sample 1

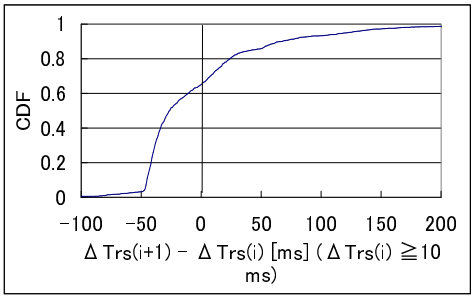

Fig. 16. CDF of $(\Delta T_{rs}(i+1) - \Delta T_{rs}(i))$ for Sample 2

According to [1], ΔH is related to an interval of transmission δ;

$$\Delta H = \delta - P/\mu,$$

where P and μ are the length of packets and the service rate at the bottleneck, respectively. Let P_p [1] and u denote the size of packet payloads and the transmission rate, respectively. Since $\delta = P_p/u$,

$$\mu = P/(P_p/u - \Delta H).$$

[1] $P = P_p$ + size of UDP and IP headers.

This is converted to payload-level bottleneck bandwidth μ';

$$\mu' = P_p/P \times \mu = P_p/(P_p/u - \Delta H).$$

Thus, peaky plots contain information about the bottleneck bandwidth. Some plots are observed under the above line. These plots correspond to the packets of which the following packets are lost.

Rather than an absolute value of ΔH, the sign of ΔH provides more useful information. It indicates whether the transmission rate exceeds the bottleneck bandwidth. When

$$u > P_p/(P/\mu + \Delta H),$$

$\Delta H > 0$. In Figure 14, there are many plots where $\Delta H > 0$; the transmission rate should have been set to be below 10 kbps. To observe the distribution of $(\Delta T_{rs}(i+1) - \Delta T_{rs}(i))$, its cumulative distribution functions (CDFs) are shown in Figures 15 and 16. We focus on the area where $\Delta T_{rs}(i) \geq 10$ ms because the behaviors only when $\Delta T_{rs}(i)$ increases are concerned with congestion. As can be seen in the graphs, most values of $(\Delta T_{rs}(i+1) - \Delta T_{rs}(i))$ are distributed in the negative area for the Keio-CMU path, and thus the CDF indicates the stability of the transmission rate.

This observation motivates us to use the distribution to identify the path status of flow. When most $(\Delta T_{rs}(i+1) - \Delta T_{rs}(i))$ data are collected in the negative area, the path is stable although there may be packet losses. Note that the path status is also related to the transmission rate. If the transmission rate is sufficiently reduced, $(\Delta T_{rs}(i+1) - \Delta T_{rs}(i))$ may not become positive except for the first packet in the compression period.

V. Related Work

Paxson [8] examined the skew removal with a robust line fitting technique. In his algorithm, $\Delta t_{rs}(i)$'s are partitioned into several segments depending on the total number of samples. The minimum delay that is picked from each segment can be a sample for calculation of the skew. The skew is calculated from the slope of all possible pairs of samples. The detail is described in [8]. Unlike Paxson's approach, MST scheme [6] applies linear programming. MST scheme is more robust than Paxson's scheme in that the calculation seeks the minimal points. Our ESRS scheme enhances MST scheme by reducing the number of samples and it can be applied to on-line calculation at a receiver.

Our ultimate goal is not the analysis of OTT but applying information about change in OTT to congestion control. There have been several efforts to introduce delay into congestion control. Jain [3] advocated congestion avoidance based on a change in round-trip time (RTT). This work provides a good starting point for a consideration of delay. In TCP Vegas [2], RTT is used not only to adjust timeout values, but also to estimate an expected throughput. The difference between an actual throughput and an expected one provides a base for rate control. However issues of stability and coexistence among other TCP variants are still being studied. LDA [12] uses RTT for its control, but LDA only obtains samples of RTT by exchanging RTCP messages and the sampled RTT values do not contain dynamic behaviors of the delay. Bolot, in the literature [1], studied properties of RTT on the Internet with UDP probing packets at a regular interval. This work describes several significant findings: the relationship between the interval and two consecutive RTT values, and existence of probe compression. Our study does not use packets only for probing and focuses on variation in OTT instead of RTT.

In the Internet, congestion has been notified implicitly. Explicit Congestion Notification (ECN) [9] is an alternative mechanism for notifying congestion. In the presence of ECN, detection of change in OTT can be substituted by monitoring packets with a congestion-experienced mark. Even with ECN, a rate control based on congestion is still required and a scheme of detecting a change in OTT and ECN are complementary.

VI. Conclusion

This paper presents a simple scheme for detecting a change in OTT. Instead of an absolute value of OTT, variation in OTT has been proven to contain information about a path that a flow traverses. First the difference between a sender's timestamp and a receiver's one was collected over a path between Keio University (Japan) and Carnegie Mellon University (U.S.), and a path between Keio University and CESAT (Spain). Then, a scheme for calculating the skew between a sender's clock and a receiver's one, ESRS, is presented. In ESRS, the number of samples for calculation of the skew is reduced by eliminating packets that have unexpected inter-arrival time. ESRS also accommodates incremental calculation of the skew by seeking and changing convex points. We applied ESRS to several samples and it was shown that the skew was effectively calculated. Finally, we analyze the variation in OTT. It is found that cumlative probability functions for phase plots of variation in OTT provide good indication of the path status.

There are several future works. We plan to investivate the extraction of information from the variation in OTT that can be effective when used for congestion control.

Acknowledgments

The authors would like to thank the members of Keio MKG project and Carnegie Mellon University Real-Time and Multimedia Laboratory for their valuable comments. We also wish to thank Jun Murai for useful discussions. Special thanks to Ragunathan Rajkumar and Sourav Ghosh of Carnegie Mellon University and Anastasio Molano of CESAT for their assistance to our experiments.

References

[1] J.-C. Bolot, "End-to-end packet delay and loss behavior in the Internet," *Proc. of ACM SIGCOMM'93,* pp. 289 - 298, Sept. 1993.

[2] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: new techiniques for congestion detection and avoidance," *Proc. of ACM SIGCOMM'94,* pp. 24 - 35, Sept. 1994.

[3] R. Jain, "A delay-based approach for congestion avoidance in interconnected heterogeneous computer networks," *ACM Computer Communication Review,* 19(5), pp. 56-71, Oct. 1989.

[4] J. Mahdavi and S. Floyd, "TCP-friendly unicast rate-based flow control," http://www.psc.edu/networking/ papers/tcp_friendly.html.

[5] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM Computer Communication Review,* 27(3), July 1997.

[6] S. B. Moon, P. Skelly, and D. Towsley, "Estimation and removal of clock skew from network delay measurements," *Proc. of INFOCOM'99,* pp. 227-234, Mar. 1999.

[7] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model based TCP-friendly rate control protocol," *Proc. of the 9th Int. Workshop on Network and Operating Systems Support for Digital Audio and Video,* pp. 137-151, June 1999.

[8] V. Paxson, "Measurements and analysis of end-to-end Internet dynamics," *University of California, Berkeley, Ph.D. thesis,* Apr. 1997.

[9] K. Ramakrishnan and S. Floyd, "A proposal to add explicit congestion notification (ECN) to IP," *RFC 2481,* Jan. 1999.

[10] R. Rejaie, M. Handley, and D. Estrin, "An end-to-end rate-based congestion control mechanism for realtime streams in the Internet," *Proc. of INFOCOM'99,* Mar. 1999.

[11] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," *RFC 1889,* Jan. 1996.

[12] D. Sisalem and H. Schulzrinne, "The loss-delay based adjustment algorithm: a TCP-friendly adaptation scheme," *Proc. of the 8th Int. Workshop on Network and Operating Systems Support for Digital Audio and Video,* pp. 215-226, July 1998.

[13] Y. Tobe, Y. Tamura, H. Nishino, and H. Tokuda, "Rate probing based adaptation at end hosts," *Proc. of IEEE Workshop on QoS Support for Real-Time Internet Applications,* pp. 58 - 65, June 1999.

[14] L. Vicisano, L. Rizzo, and J. Crowcroft, "TCP-like congestion control for layered multicast data transfer," *Proc. of INFOCOM'98,* Apr. 1998.

[15] H. A. Wang and M. Schwartz, "Achieving bounded fairness for multicast and TCP traffic in the Internet," *Proc. of ACM SIGCOMM'98,* pp. 81 - 92, Aug. 1998.