Realtime Burstiness Measurement

Ryousei Takano
National Institute of Advanced Industrial Science and Technology (AIST), Japan and *AXE, Inc.*
Yuetsu Kodama, Tomohiro Kudoh, Motohiko Matsuda, Fumihiro Okazaki
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{takano-ryousei, y-kodama, t.kudoh, m-matsuda, f-okazaki}@aist.go.jp
and Yutaka Ishikawa
University of Tokyo and *National Institute of Advanced Industrial Science and Technology (AIST), Japan*
ishikawa@is.s.u-tokyo.ac.jp

Abstract

Bursty traffic causes excessive queuing delay and packet losses. Many researchers have proposed the modeling of burstiness and burstiness mitigation schemes. However, there is no consensus on a quantitative definition of burstiness. In this paper, burstiness is defined as the queue size of a virtual bottleneck buffer. This definition is based on the fact that packets are queued when there is a bottleneck, and then the difference between target and service rates is reflected in queue length, i.e. the amount of data stored at the buffer before the bottleneck. We design and implement a realtime burstiness measurement method on a fully programmable network testbed, GtrcNET-1. Furthermore, we measure actual traffic on two networks, and we present the relationship between burstiness and the packet loss rate, and the behavior of flow aggregation. The results show the effectiveness of both the definition and our measurement method.

1. Introduction

On the Internet, bursty traffic occurs due to several causes, including TCP's slow start, idle restarts, ACK compression, packet reordering, and segmentation of application messages into multiple UDP packets [8]. Bursty traffic is characterized as traffic in which short term bandwidth exceeds the average bandwidth. If the bursty traffic exceeds the available bandwidth of a path, packets are queued at the router or the switch at the entry point of the path, and may cause excessive queuing delay and packet losses. Especially in TCP/IP communication over fast long-distance networks, burstiness tends to increase, and communication performance can be markedly degraded [1][2]. In order to improve the performance, it is important to detect the relationship between burstiness and performance factors, and apply this information in traffic engineering and network planning.

We have developed a traffic shaping software program called PSPacer [2], which precisely regulates bandwidth and minimizes the burstiness of outgoing traffic on the sender side. Figure 1 shows the observed outgoing packets from a PC, when each of PSPacer and Linux's Token Bucket Filter (TBF) are used to regulate the bandwidth of outgoing traffic. PSPacer accurately adjusts transmission intervals between adjacent packets, i.e. the Inter Packet Gap (IPG), in which case IPG is adjusted to 288 μs. On the other hand, TBF causes burst transmission, whereby 33 packets are transmitted in bursts and the transmission stops over a period of 9.5 ms. Obviously, there is always a certain amount of burstiness when TBF is used. When the traffic is transferred at 40 Mbps, on average, through a FTTH subscriber line (B Flet's)[1], with PSPacer, the packet loss rate is 0.35%. On the other hand, with TBF, the packet loss rate is 1.6%. TBF output, which has more burstiness, causes a larger number of losses.

To understand this problem, the burstiness of traffic should be quantitatively defined. However, there is no established *quantitative definition of burstiness*. Some definitions of burstiness have been proposed [4], especially during the ATM era. Most of them are based on statistical approaches, and not directly related to the queuing delay or packet losses caused by the bursty traffic. We believe the definition of the burstiness should be directly related to the performance impact, and the type of definition above is not sufficient. In this paper, we propose a quantitative definition of burstiness, which is directly related to the possible queuing problem, i.e. delay or packet loss, caused by the traffic involved.

[1] The experimental setting is shown in Section 4.

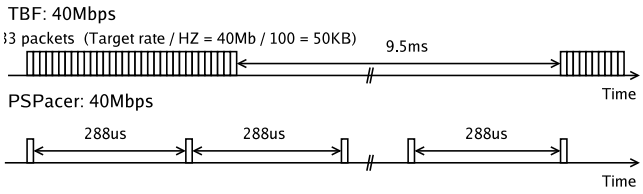

Figure 1. Two traffic shaping methods: Token Bucket Filter and PSPacer (Target rate=40Mbps, HZ=100, MTU=1500B).

The rest of the paper is organised as follows. Section 2 describes a proposed definition of burstiness. Section 3 presents the design and implementation of a realtime burstiness measurement tool using fully programmable network testbed hardware. Section 4 shows the experimental results on two domestic networks in order to evaluate the effectiveness of both the definition and the measurement method. In Section 5, we discuss related work on prior definitions of burstiness and the relationship between burstiness and TCP congestion events. Finally, Section 6 summarizes the paper.

2. Definition of Burstiness

To define burstiness, we assume a measurement point as a virtual bottleneck (VB) with a given bandwidth (BW_{vb}) and an infinite input buffer (VBQ), as shown in Figure 2. If unidirectional IP traffic flows from sources to destinations went through a measurement point, and the bandwidth of the traffic exceeded BW_{vb}, the excess traffic would be stored in VB. We define the burstiness as the amount of data queued on this input buffer, i.e. queue size (Q).

Furthermore, we propose two definitions of burstiness, i.e. a static burstiness and a dynamic burstiness, according to the purpose of use. In order to evaluate whether the traffic can be accommodated by a known and fixed bandwidth network path, BW_{vb} set to a given target rate is useful, in which case we call Q a *static burstiness*. On the other hand, if the available bandwidth is unknown or varied, BW_{vb} set to the average bandwidth of the observed traffic is useful, in which case we call Q a *dynamic burstiness*. To measure dynamic burstiness, we first have to know the average bandwidth (ABW_T) of the observed traffic over a certain period of time T. Now, according to the definition, $BW_{vb} = ABW_T$. If the incoming traffic exceeds ABW_T, Q increases, while if it is below ABW_T, Q decreases. The maximum burstiness $max(Q)_T$ is the maximum value of Q during the period T.

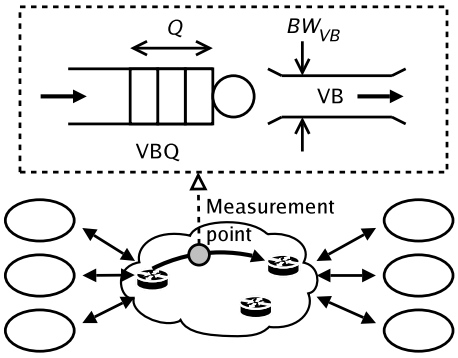

Figure 2. Burstiness: Virtual Bottleneck (VB) and VB Queue (VBQ)

3. Burstiness Measurement Method

We developed a realtime burstiness measurement tool using fully programmable network testbed hardware. The requirements were as follows:

- It had to be able to measure a static burstiness and a dynamic burstiness
- It had to be able to measure simultaneously both the burstiness and the average bandwidth
- It had to be able to measure in realtime without any probe effect

3.1. Burstiness Measurement Logic

In order to measure dynamic burstiness accurately, the value of $max(Q)_T$ for packets observed over a period of time T should be obtained by using ABW_T measured from the packets. Therefore, the proposed burstiness measurement method is separated into two stages: an ABW stage and a VBQ stage, as shown in Figure 3. The observed traffic is divided into traffic segments (S_i) at each time of T, where T is a measurement resolution. In the ABW stage, ABW_T of S_i is measured, and then in the VBQ stage, $max(Q)_T$ of S_i is measured. That is, ABW_T of S_i and $max(Q)_T$ of S_{i-1} are measured simultaneously. Thus, the burstiness $max(Q)_T$ and the average bandwidth ABW_T can be obtained in tandem.

Figure 4 shows a block diagram of this burstiness measurement logic. There are two measurement modes: a static burstiness mode and a dynamic burstiness mode, where each mode measures static burstiness or dynamic burstiness. The logic is equipped with 2 FIFOs: FIFO A is used to measure the average bandwidth of incoming traffic, and to store packets for a period of time T, FIFO B is used as a VBQ buffer.

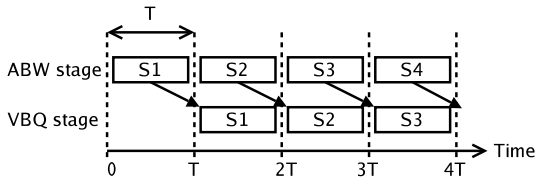

Figure 3. ABW stage and VBQ stage: In the ABW stage, average bandwidth (ABW_T) of traffic segment (S_i) is measured, and then in the VBQ stage, burstiness ($max(Q)_T$) of S_i is measured.

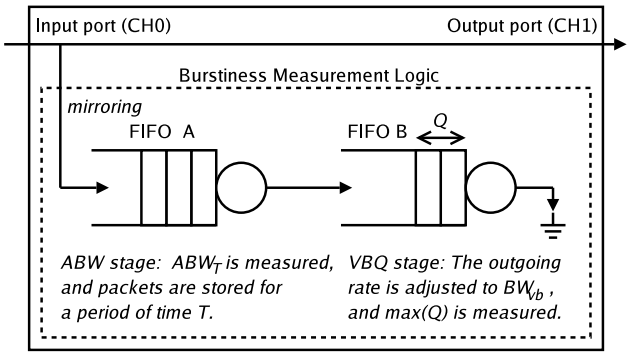

Figure 4. Burstiness Measurement Logic: It is equipped with 2 FIFOs: FIFO A and FIFO B are used for the ABW stage and the VBQ stage, respectively.

The incoming packets from the input port (CH0) are first mirrored (copied), and the original packets are sent out from the output port (CH1). Because the burstiness is measured using the mirrored packets, the burstiness can be measured without any probe effect. The mirrored packets are enqueued in FIFO A. Based on the amount of incoming traffic during a given period of time T, ABW_T is measured. The packets are stored in FIFO A for this period, and then these are forwarded to FIFO B. All packets are retained for the same period time in FIFO A, and thus jitter is preserved when the packets exit FIFO A.

Packets stored in FIFO B are dequeued at the rate of BW_{vb}. In the static burstiness mode, the bottleneck bandwidth of VB BW_{vb} is statically set to the given bandwidth. In the dynamic burstiness mode, BW_{vb} is dynamically set to the value of ABW_T. Now, the burstiness Q can be monitored as the queue size of FIFO B. All packets at FIFO B are discarded at the end of each measurement period T.

3.2. Implementation on GtrcNET-1

We developed a realtime burstiness measurement tool using a fully programmable network testbed, GtrcNET-1 [3], as shown in Figure 4. GtrcNET-1 is comprised of a Field Programmable Gate Array (FPGA), four 144 Mbit SRAM blocks and four gigabit Ethernet ports. The FPGA is a Xilinx XC2V6000, which includes a 76K logic cells, 2.5 Mbit of memory, and 824 user I/O pins. By configuring the FPGA, functions such as traffic monitoring, traffic shaping, and WAN emulation can be provided at gigabit wire speed. GtrcNET-1 also has a USB 1.1 interface to connect to the control PC, which sets and gets the parameters and measurement results.

Our tool consists of a burstiness measurement function on GtrcNET-1 and a monitor program on the control PC. Before a measurement attempt is started, T and a measurement mode are set from the monitor program. In the static burstiness mode, BW_{vb} must also be set to the target bandwidth. And then, the monitor program acquires a tuple $\{ABW_T, max(Q)_T\}$ at each time of T.

GtrcNET-1 has four gigabit Ethernet ports. In this tool, two ports are paired, i.e. CH0 - CH1 and CH2 - CH3 to emulate two gigabit links. One link (CH0 - CH1) is used as the path through the original packets, the other link (CH2 - CH3) is used as the VB link. The transmission delay between CH0 and CH1 is about 1.1 μs. We think that this overhead of measurement is small enough and can be ignored. FIFO A and FIFO B are assigned to individual SRAM blocks. In FIFO B, ABW_T is calculated from the amount of incoming packets and the hardware clock (32 bit increasing counter of each $2^{-24}second \simeq 59.6ns$). In FIFO B, the outgoing rate is shaped by controlling the IPG precisely.

The measurement resolution T can be specified up to 134 ms with 32 ns resolution. It is limited by the capacity of FIFO A. For example, 6.25 MB of memory is required for measurement in the case of $ABW_{100ms} = 500Mbps$.

4. Experiment

We measured actual traffic on two domestic networks: B Flet's and SuperSINET, using the proposed realtime burstiness measurement tool.

4.1. Experimental Setting

Figure 5 shows the network setting, and Table 1 shows the specifications of the sender and receiver PCs. The senders are in Tsukuba, and the receiver is in Akihabara, which are about 60 kilometers apart. There are two routes between Tsukuba and Akihabara. The first one is a FTTH subscriber line (B Flet's), whose maximum physical bandwidth and Round Trip Time (RTT) are 100 Mbps and about 6.0 ms, respectively. The available bandwidth is 40 Mbps, on average. The observed traffic is much affected by cross

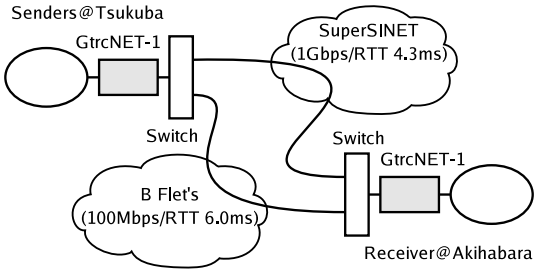

Figure 5. Experimental Setting.

Table 1. PC specifications.

	Sender	Receiver
CPU	Opteron 2.0 GHz dual	Xeon 2.4 GHz dual
M/B	IBM eServer 325	SuperMicro X5DAE
Memory	6GB (PC2700)	2GB (PC2100)
NIC	Broadcom BCM5704	Intel 82545EM
I/O Bus	PCI-X 133MHz/64bit	PCI-X 133MHz/64bit
OS	SUSE SLES 8.1	FedoraCore 3
Kernel	Linux 2.4.21-251	Linux 2.6.13.3

traffic. The second one is an academic high speed network (SuperSINET), whose bandwidth and RTT are 1 Gbps and about 4.3 ms, respectively. While the access link is 1 Gbps, the backbone network bandwidth is 10 Gbps, and we can use it almost as a dedicated 1 Gbps network. We deploy GtrcNET-1 on each side to measure both the outgoing and incoming traffic.

We used the iperf benchmark, and measured constant bit rate (CBR) UDP flows in order to avoid the effect of TCP congestion control. In this experiment, generating CBR flow at the target rate was achieved by two different traffic shaping methods. The first one was TBF, which is included in the Linux kernel. The second one was PSPacer [2], which achieves accurate network bandwidth control and smoothing of bursty traffic precisely. PSPacer provides both fixed-rate pacing and rate-based TCP pacing. In this experiment, the latter is used.

4.2. Results

We measured the single 100Mbps CBR UDP flow and the five individual 100Mbps CBR flows aggregated into one 500Mbps flow generated by iperf over a period of 1 minute. We used a dynamic measurement mode, and T is set to 100 ms. Table 2 summarizes the results of the experiments, showing average and maximum of ABW_{100ms} and $max(Q)_{100ms}$. In the rest of the paper, the subscript $100ms$ is omitted. We measured traffic under test as the combination of the measurement point (or path), the target rate and the number of flows. In the case of 'O', it is measured on the sender side, in the cases of 'B' and 'S', it is measured on the receiver side. 'B' and 'S' indicate incoming traffic from a B Flet's line and SuperSINET line, respectively. 'T' and 'P' indicate the use of traffic shaping method, representing TBF and PSPacer, respectively. 'n/m' means the target rate (n Mbps) and the number of flows (m flows).

Table 2. ABW and $max(Q)$ over a period of 1 minute.

TUT	ABW (Mbps)		$max(Q)$ (KB)	
	ave	max	ave	max
O: outgoing traffic				
OT40/1	40.0	44.1	49.9	93.2
OP40/1	39.9	39.9	0.004	0.004
OT100/1	100.3	110.3	113	189
OP100/1	99.7	99.8	0.457	0.528
B: incoming traffic from B Flet's				
BT40/1	40.0	52.4	2.21	63.2
BP40/1	39.9	46.5	0.977	22.5
S: incoming traffic from SuperSINET				
ST100/1	100	110	112	189
SP100/1	99.8	100	0.483	3.01
ST100/5	502	522	322	595
SP100/5	499	500	2.69	8.64

TUT (Traffic Under Test):
'O', 'B' and 'S' indicate outgoing traffic, incoming traffic from B Flet's, and SuperSINET, respectively.
'T' and 'P' indicate TBF and PSPacer, respectively.
'n/m' means the target rate (n Mbps) and the number of flows (m flows).

In this paper, we focus on two comparisons of the results: the relationship between burstiness and the packet loss rate, and the behavior of flow aggregation.

4.2.1. Relationship between burstiness and packet loss

The preciseness of PSPacer is shown quantitatively. When PSPacer is used (OP40/1 and OP100/1), the burstiness Q of outgoing traffic is quite small compared with the cases where TBF is used (OT40/1 and OT100/1). TBF controls the transmission rate by using a kernel timer, and bursty transmission can not be avoided [2]. In theory, $max(Q)$ of TBF may be as large as $target\ rate/HZ$, where HZ is the frequency of timer interrupt. This means that burstiness increases proportional to the target rate and inversely proportional to the resolution of a kernel timer. When the target rate is 40 Mbps and HZ is 100 Hz, $target\ rate/HZ$ is $50KB$. The experimental result (OT40/1) confirms this. These differences of burstiness are

[2] In the Linux kernel 2.4.x, HZ (kernel timer frequency) is set to 100. That is, the timer resolution is 10 ms. If HZ is set to 1000, the burstiness becomes 1/10 times.

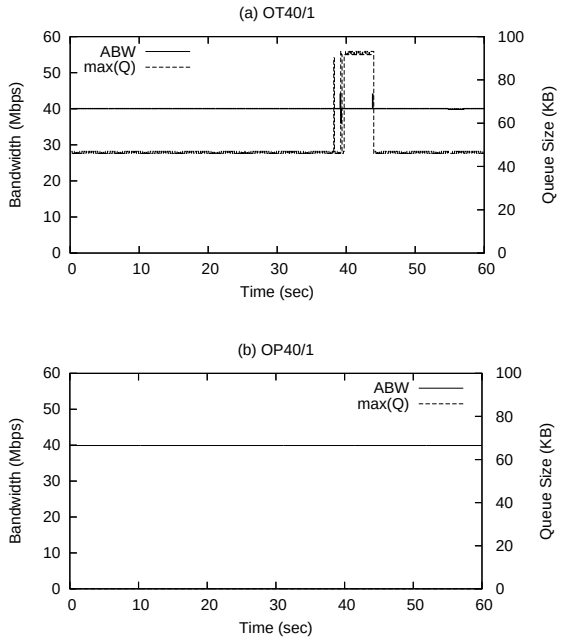

Figure 6. A single 40 Mbps CBR flow (outgoing traffic).

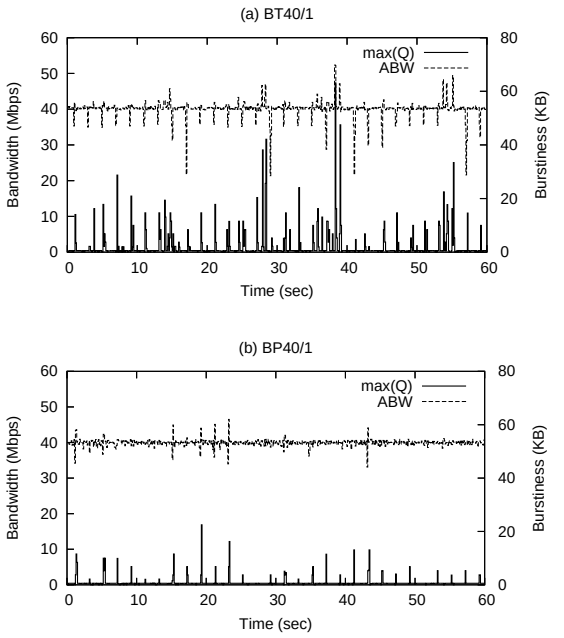

Figure 7. A single 40 Mbps CBR flow over B Flet's (100 Mbps, RTT 6.0 ms).

directly reflected to the loss rate. The packet loss rate with PSPacer and with TBF are 0.35% and 1.6%, respectively. These results show that larger burstiness increases the probability of packet losses.

Figure 6 (a) and (b) show the results of sender's outgoing traffic, where the target rate is set at 40 Mbps (i.e. OT40/1 and OP40/1). Figure 7 (a) and (b) show the results over B Flet's, i.e. BT40/1 and BP40/1. ABW is almost the same as the target rate in both BT40/1 and BP40/1. We note that PSPacer smooths variation of ABW in comparison with TBF. Average $max(Q)$ of TBF is 2.3 times larger than that of PSPacer. With TBF, $max(Q)$ on the receiver side (BT40/1) is smaller than on the sender side (OT40/1). With PSPacer, $max(Q)$ on the receiver side (BP40/1) is larger than on the sender side (OP40/1). On the other hand, $max(Q)$ over SuperSINET is almost same for both the sender side and the receiver side. These results show traffic is moderated due to cross traffic or queuing at switches and routers on a shared link.

4.2.2. Flow aggregation

Comparing the single flow and the aggregate flow in Table 2, with PSPacer, average $max(Q)$ of the aggregate flow (SP100/5) is 5.6 times larger than that of the single flow (SP100/1), in which case the average $max(Q)$ is proportional to the number of streams. With TBF, on the other hand, average $max(Q)$ of the aggregate flow (ST100/5) is 2.9 times larger than that of the single flow (ST100/1). In this case, maximum $max(Q)$ of the aggregate flow is 5.3 times larger than average $max(Q)$ of the single flow. However, the average $max(Q)$ is not proportional to the number of streams, because the variation of the individual flow's burstiness is averaged.

Figure 8 shows the results over SuperSINET in detail. With TBF, Figure 8 (a) and (b) show the results of ST100/1 and ST100/5, respectively. Burstiness of the single flow is 112 KB on average, and stable. On the other hand, that of the aggregate flow is 322 KB on average, and variable. Figure 8 (c) shows the result of SP100/5. The transmission rate is quite stable, because outgoing paced traffic is not disturbed over SuperSINET by cross traffic, which is in contrast with the B Flet's.

5. Related Work

Some burstiness definitions have been proposed during the ATM era. Most of them are based on statistical approaches. Michiel et al [4] discussed prior studies of burstiness characteristics such as peak-to-mean ratio, coefficient of variation (standard deviation to mean ratio), index of dispersion for counts, index of dispersion for intervals, Hurst parameter, and so on, in the B-ISDN context.

Recently, some burstiness definitions based on a model of TCP dynamics have been proposed. Allman et al [5]

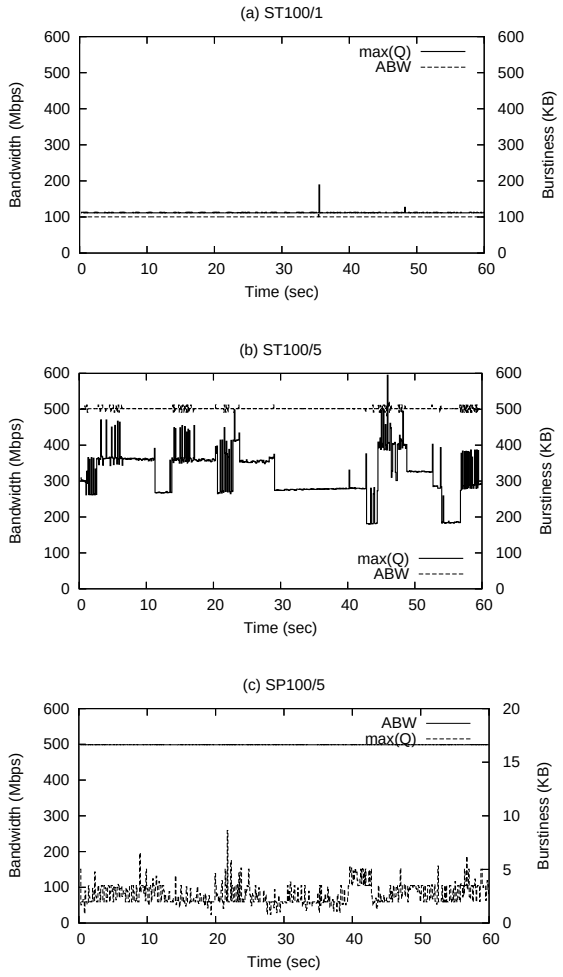

Figure 8. A single 100 Mbps CBR flow and an aggregate five 100Mbps CBR flow over SuperSINET (1 Gbps, RTT 4.3 ms).

defined two kinds of TCP bursty behavior in terms of timescale: macro-bursts and micro-bursts. A micro-burst is defined as a sequence of at least four segments sent between two successive ACKs, and the paper discussed the pros and cons of several burstiness mitigation schemes in its simulation study. Shakkottai et al [7] proposed a simple threshold algorithm to identify a *flight*, which is a sequence of packets in bursts, and held that delayed ACKs and window dynamics can cause flights. Wei et al [10] proposed a two-layer model for TCP congestion control: window control over RTT time scale and burstiness control over packet time scale, and they presented a mechanism to measure and control burstiness based on the amount of the backlog in the interface queue at the sender side.

Different from these existing approaches, we proposed a quantitative definition of burstiness, which is protocol independent, and directly related to the performance impact caused by the bursty traffic. In addition, most of existing burstiness measurement algorithms [6][7] assume off-line analysis of packet trace files. One proposed method can measure in realtime without any probe effect.

Many researchers also have proposed burstiness mitigation schemes. In TCP congestion control algorithms, burst packet losses are even more serious than single packet loss. Sometimes these losses are not recovered by NewReno or SACK algorithms and cause serious retransmission timeouts. Blanton et al [6] showed that packet losses caused by micro-bursts are not frequent on the Internet since the size of bursts is modest, while larger bursts increase the probability of packet losses. Jiang et al [9] investigated the fact that burstiness in sub-RTT scales can be significantly reduced by TCP pacing [11], if a kernel timer had sufficiently fine resolution. Thus, a burstiness measurement tool must provide correlation between burstiness and TCP congestion events, which is useful to reduce congestion and improve communication performance. In order to investigate both the correlation and macro-level burstiness, an analysis technique based on time series of ABW_T and $max(Q)_T$ is required.

6. Conclusion

We proposed a quantitative definition of burstiness, which directly relates to the performance impact caused by the bursty traffic. Based on the definition, we implemented a realtime burstiness measurement method on a fully programmable network testbed, GtrcNET-1. This method provides two different burstiness measurements: a static burstiness in which case the average bandwidth is known and fixed, and a dynamic burstiness in which case the average bandwidth varies based on the observed traffic. We evaluated the effectiveness of both the definition and the measurement method. The preliminary measurement results show PSPacer reduces both burstiness and packet losses, and flow aggregation moderates burstiness in the cases where TBF is used. While we focused on CBR flows in this paper, we plan to analyze the behavior of the high-speed TCP variants. Future work is needed to develop an analysis technique of the correlation between burstiness and TCP congestion events, based on time series of ABW_T and $max(Q)_T$.

Acknowledgment

A part of this research was supported by a grant from the Ministry of Education, Sports, Culture, Science and Technology (MEXT) of Japan through the NAREGI (National Research Grid Initiative) Project.

References

[1] S. Floyd, “HighSpeed TCP for Large Congestion Windows,” RFC 3649, December 2003.

[2] R. Takano, T. Kudoh, Y. Kodama, M. Matsuda, H. Tezuka, and Y. Ishikawa, “Design and Evaluation of Precise Software Pacing Mechanisms for Fast Long-Distance Networks,” PFLDnet 2005, February 2005.

[3] Y. Kodama, T. Kudoh, R. Takano, H. Sato, O. Tatebe, and S. Sekiguchi, “GNET-1: Gigabit Ethernet Network Testbed,” IEEE Cluster 2004, September 2004.

[4] H. Michiel and K. Leavens, “Teletraffic engineering in a broad-band era,” Proceedings of the IEEE, vol. 85, no. 12, pp. 2007-2033, December 1997.

[5] M. Allman and E. Blanton, “Notes on Burst Mitigation for Transport Protocols,” ACM Computer Communication Review, April 2005.

[6] E. Blanton and M. Allman, “On the Impact of Bursting on TCP Performance,” Passive and Active Measurement Workshop, March 2005.

[7] S. Shakkottai, N. Brownlee, and K. C. Claffy, “A Study of Burstiness in TCP Flows,” Passive and Active Measurement Workshop, March 2005.

[8] H. Jiang and C. Dovrolis, “Source-Level IP Packet Bursts: Causes and Effects,” ACM Internet Measurements Conference, October 2003.

[9] H. Jiang and C. Dovrolis, “Why is the Internet traffic bursty in short time scales?,” ACM SIGMETRICS, June 2005.

[10] D. X. Wei, S. Hegdesan, and S. H. Low, “A Burstiness Control for TCP,” PFLDnet 2005, February 2005.

[11] A. Aggarwal, S. Savage, and T. Anderson, “Understanding the performance of TCP pacing,” IEEE INFOCOM, pp. 1157-1165, March 2000.