Can high-speed transport protocols be deployed on the Internet? : Evaluation through experiments on JGNII

Kazumi Kumazoe
National Institute of Information and Communications Technology
3-8-1 Asano, Kokurakita-ku,
Kitakyusyu-city, Fukuoka, Japan,
kuma@kyushu.jgn2.jp

Katsushi Kouyama
Kyushu Electric Power Co.,Inc.,
kouyama@kyushu.jgn2.jp

Yoshiaki Hori
Dept. of Computer Science and Communication Engineering,
Kyushu University
hori@csce.kyushu-u.ac.jp

Masato Tsuru Yuji Oie
Dept. of Computer Science and Electronics,
Kyushu Institute of Technology
{tsuru, oie}@cse.kyutech.ac.jp

***Abstract*— While a variety of high-speed transport protocols have been proposed to meet the requirement of high throughput data transfer over fast long-distance networks, less attention has been paid to the problems involved when those transport protocols are deployed in heterogeneously shared network environments like the global Internet. We are investigating several high-speed transport protocols: HighSpeed TCP, Scalable TCP, FAST, CUBIC, HTCP and UDT, through experiments over the Japan Gigabit Network (JGN)II, an open 10Gbps-class high-speed network testbed in Japan, mainly focusing on the following question - what will happen if these protocols are run on the Internet? In this paper, some results of ongoing experiments are reported: characteristics of these transport protocols in cases with realistic conditions, e.g., a variety of receiver-side OSs, coexistence of short-lived TCP flows, or coexistence of constant bit-rate UDP flows. Our preliminary results indicate that these protocols are neither effective nor efficient in terms of network resource sharing if they are running with Internet application traffic such as a number of short-term web browsing flows or long-term video streaming flows.**

I. Introduction

In response to the emerging requirements for high throughput data transfer on fast long-distance networks in distributed data processing and data sharing such as Grid, a variety of high-speed transport protocols have been proposed. Among them, there are two practical end-to-end approaches: one is modification of the congestion control mechanism in TCP at the sender-side, and the other is implementation of new transport protocols over UDP. Their performance (throughput characteristics) in dedicated and/or homogeneous networks have already been studied and reported through various experiments on worldwide testbeds (e.g., [1],[2]).

On the other hand, the bandwidth of the Internet has been increasing more and more in both access networks and core networks. In Japan, for example, 1 Gbps broadband access (FTTH) services have recently become available at a considerably low price, and 10 Gbps services might become reality in the near future. In such situations, Internet users including large application service providers (ASPs) may like to use high-speed transport protocols on the Internet to transfer a larger amount of data, regardless of the intention and/or scenario of the original developers of those protocols. However, little attention has been paid to the problems involved when those transport protocols are deployed on shared and heterogeneous networks such as the global Internet.

Therefore, we started investigating several promising high-speed transport protocols: HighSpeed TCP (HSTCP)[3], Scalable TCP[4], FAST[5], CUBIC[6], HTCP[7] and UDT[8], through experiments over the Japan Gigabit Network (JGN) II[9], an open 10 Gbps/1 Gbps Ethernet-based network testbed in Japan, mainly focusing on the following question – what will happen if those protocols are running on the Internet? In this paper, following the previous experiments[2] in which we mainly examined how a change in network conditions (e.g., amount of background traffic, bandwidth, propagation delay, packet loss, and packet misordering) affects the throughput characteristics, we try to examine what happens in cases with more realistic conditions such as, a variety of receiver-side OSs (Linux, Windows XP, and FreeBSD), coexistence of flows with different RTTs and different protocols, coexistence of short-lived TCP flows (e.g., web browsing flows), or coexistence of constant bit-rate UDP flows (e.g., video stream flows).

This paper is organized as follows. Section 2 explains the configuration of the experimental environments on JGNII. The experimental results are explained in Section 3, which is followed by some closing remarks in Section 4.

II. Configuration of our Experiments on JGNII

Figure 1 shows the network configurations in our experiments: (a) the network emulator path (by Hurricane II from PacketStorm), (b) the JGNII domestic loopback path, (c) the network emulator and the JGNII domestic loopback path, and (d) the JGNII path between the US and Japan. The properties of each path are listed in Table I for JGNII Domestic Line (Kitakyushu-Tokyo-Kitakyushu), JGNII International Line (Kitakyushu-Chicago), and Network Emulator in which both RTT and bandwidth are configurable. While both JGNII domestic and international lines provide a maximum of

10[Gbps] in its bandwidth, in this paper, we only show the results where the bottleneck bandwidth is limited to 1[Gbps] at the links (from end-hosts and to upstream) of the edge routers. The default value of 127 was mainly used as the output buffer size of the edge routers, GS4000 in Fig. 1, because a dedicated performance tuning of network nodes does not match our intention. A larger buffer size of 512, however, was also used to investigate the impact of buffer size of the edge routers.

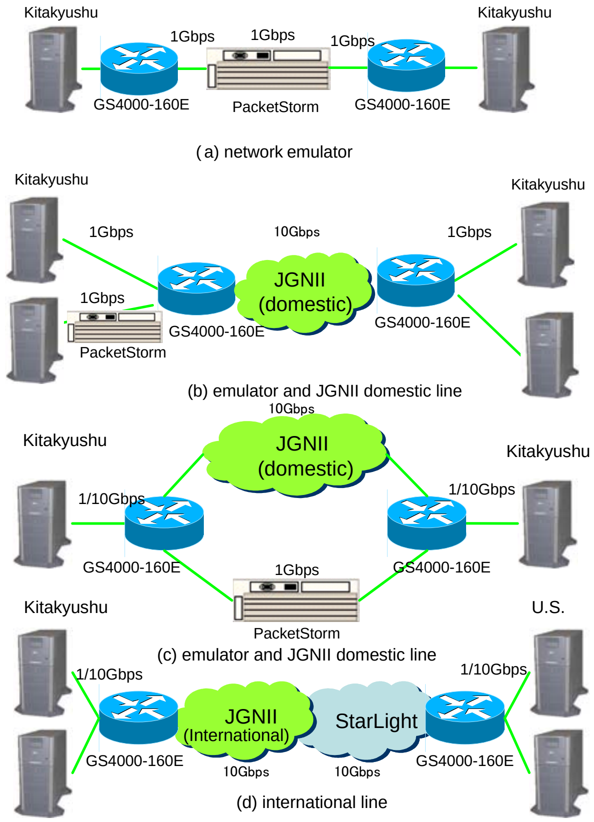

Fig. 1. Network Configuration

The output of the traceroute command on the international lines is as follows.

Kitakyushu to Starlight :
traceroute to 206.220.241.15, 30 hops max, 38 byte packets
1 202.180.37.33 1.967 ms 1.833 ms 1.868 ms
2 203.181.249.126 134.043 ms 121.542 ms 117.437 ms
3 203.181.248.217 199.874 ms 200.385 ms 199.753 ms
4 192.203.116.10 181.276 ms 181.257 ms 181.269 ms
5 206.220.241.15 181.392 ms 181.270 ms 181.263 ms

Starlight to Kitakyushu:
traceroute to 202.180.37.34 (202.180.37.34), 30 hops max, 38 byte packets
1 206.220.241.244 5.202 ms 0.224 ms 0.219 ms
2 192.203.116.9 19.350 ms 18.306 ms 18.305 ms
3 203.181.248.218 180.502 ms 181.464 ms 195.447 ms
4 202.180.34.157 162.540 ms 162.526 ms 162.574 ms
5 202.180.34.154 183.132 ms 196.023 ms 182.979 ms
6 202.180.37.34 181.228 ms 181.279 ms 181.367 ms

TABLE I
PATH CHARACTERISTICS

	Bandwidth[Mbps]	RTT[msec]
Network Emulator	0–1000	0.1–10000
JGNII International Line	10000	approximate 180
JGNII Domestic Line	10000	approximate 38

TABLE II
TARGETED PROTOCOLS

Protocol	Version
HSTCP	patch for Linux 2.4.19
Scalable TCP	patch for Linux 2.4.19
FAST	running on Linux 2.4.20
CUBIC	patch for Linux 2.4.25
HTCP	running on Linux 2.4.20
UDT	version 2.0

For all paths in our experiments, when the bottleneck bandwidth was 1[Gbps], a UDP flow by Iperf could achieve the throughput of 940[Mbps] without packet losses.

We targeted seven high-speed transport protocols and used their implementations provided by the researchers as show Table II. The changes from our previous work ([2]) were adding HTCP and replacing BIC to CUBIC as its successor. We investigated throughput characteristics for TCP flows, and packet loss and delay jitter for UDP flows, which were measured by Iperf as performance measures.

Table III shows the sender and receiver equipment specifications. We adopted Linux (RedHat 9.0 Kernel 2.4.20) as the sender-side OS in our experiments mainly because, expect for UDT, the function codes are provided as patches for the source code of Linux. We tuned various parameters, e.g., Linux txqueulen, system memory, and RxDescriptors of the NIC(e1000) driver, based on the technical information provided in the web pages of each of the targeted protocols.

III. EXPERIMENTAL RESULTS

In this section, we show the results in several scenarios, some of which are similar to the scenarios of our previous paper ([2]). Compared with the configuration in [2], however, in the new configuration of Fig. 1, the network emulator and the edge routers accommodating end-hosts were replaced, and the routes of paths between Japan and the US were changed. The results in those scenarios would allow us to confirm the generality if the old results and new ones are consistent.

Note that the JGNII paths accommodate other constant or temporary traffic in general. To avoid the influence of

TABLE III
EQUIPMENT SPECIFICATIONS

	Endhost in Chicago	Endhost in Kitakyushu
OS	Debian Linux	Red Hat Linux 9.0 Kernel 2.4.20
CPU	Xeons 2.4[GHz],opteron	Xeons 3.2[GHz]
Memory	1[GByte]	2[GByte]
PCI BUS	64[bit]	
NIC (1[Gbps])	Intel Pro(e1000)	Intel Pro(e1000)

unexpected large background traffic, we conducted several trials in each experiment with the same configuration, and then discarded irregular cases. Unless otherwise noted, the results shown hereafter are for the cases exhibiting a relatively good throughput performance.

In addition, when we observed similar tendencies on HSTCP and Scalable TCP, we only showed the characteristics of one of them.

A. Basic Characteristics of each Protocol

Figure 2 illustrates the throughput characteristics on a single flow of each targeted protocol in the configuration shown in Fig. 1 (a), which indicate a stable (and thus a basic) throughput characteristic in an ideal network with no (or very few) packet loss. We configured the RTT to 180[msec] in network emulator, which emulates the configuration shown in Fig. 1(d).

In Fig. 2, we can observed that, while every high-speed transport protocol flow achieved a high throughput near to 1 [Gbps], the behaviors of these flows in increasing and decreasing throughput are different.

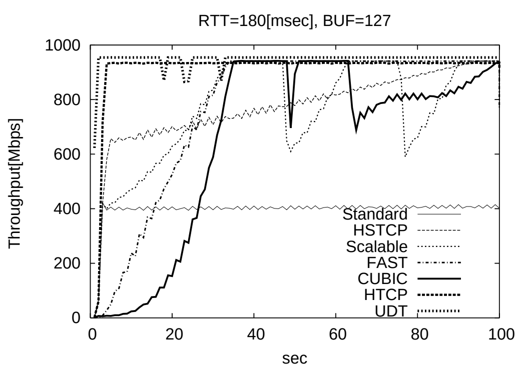

Fig. 2. Throughput of a single flow(Emulator)

Figure 3 presents the throughput characteristics of a single flow of each targeted protocol in the configuration shown in Fig. 1(d). The output buffer size at the edge routers were set to 127 or 512. Compared with Fig. 2, Fig. 3 shows slightly unstable throughput characteristics for all protocols, which might come from the difference between emulated and actual networks even though the RTTs in both cases were nearly identical. For example, the number of intermediate nodes and the existence of cross-traffic are different. Therefore, we run our experiments in both environments if possible: using the network emulator to investigate the basic characteristics and using the JGNII lines to investigate more realistic characteristics.

The upper graph in Fig. 3 shows cases with the buffer size of 127 at the edge routers, while the lower one shows cases with the buffer size of 512. A large difference in their throughput characteristics of FAST and Scalable TCP flows was observed. The throughput of FAST flow was greatly improved setting buffer size at the edge routers larger. When the buffer size was set to 127, packet losses were observed on FAST flow, while no packet losses on the FAST flow when the buffer size was 512. On the other hand, the throughput of Scalable TCP flow seemed unstable in case with a larger buffer size of the edge routers. In fact, the averaged throughput of a Scalable TCP flow for the buffer size of 127 is 450 [Mbps], while 300 [Mbps] for the buffer size of 512.

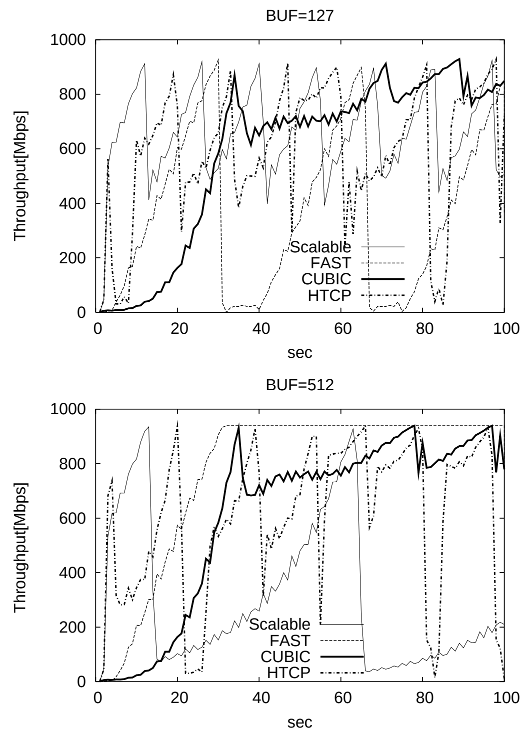

Fig. 3. Throughput of a single flow (International Line)

B. Variety of receiver-side OSs

We adopted Linux as the sender and receiver side OS basically. In this subsection, the performance are shown when the FreeBSD (ver.5.3) and Windows XP (SP2) were adopted as a receiver side OS in the configuration shown in Fig. 1(a), First we tuned the parameters of each OS such as the socket buffer size according to the technical information in [10].

Figure 4 illustrates the throughput characteristics of a single flow for FreeBSD and Windows XP of the receiver side OS. Basically we observed that all protocol flows (except for standard TCP) could achieve high throughput regardless of the kinds of receiver side OS. That is, the majority of users on the Internet are ready to fill up the bandwidth up to 1[Gbps] as receivers if the sender employs such the high-speed transport protocols.

Note that, while every high-speed transport protocol flow eventually achieved a high throughput near to 1 [Gbps], the throughput in the case of XP receiver increased more slowly compared with the cases of Linux and FreeBSD receivers. In addition, Standard TCP flows in the case of XP achieve higher throughput than that in the cases of Linux and FreeBSD.

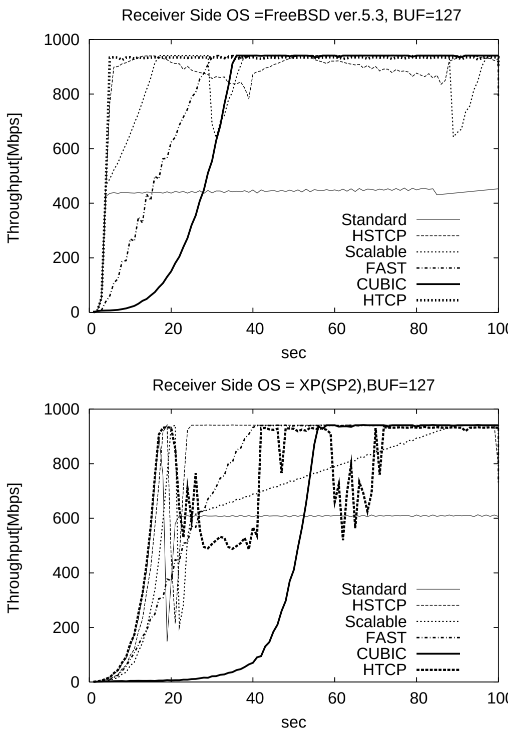

Fig. 4. Throughput characteristics observed with different receiver side OSs

C. Rapid change in network conditions

Route change is still and will be likely to happen in the Internet due to the operational errors and the need for global traffic engineering. Thus, we investigate how rapid changes of network conditions (RTT and bottleneck bandwidth) affect the throughput characteristics of each of the high-speed transport protocols in switching the paths between two routers, the emulator-path (very stable) and the JGNII domestic region path (somewhat unstable) on the network as illustrated in Fig. 1(c).

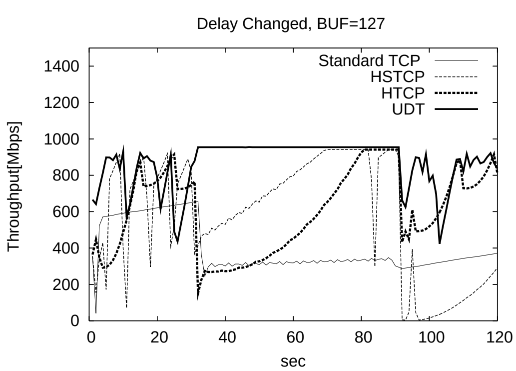

Fig. 5. Path switching(RTT[msec]=38 → 80 → 38)

In Fig. 5, we observed the throughput behavior of a single flow in case that the path was switched from the original path with 38 [msec] of RTT (JGNII domestic path) to the alternative path (the emulator path) with 80 [msec] of RTT just after 30

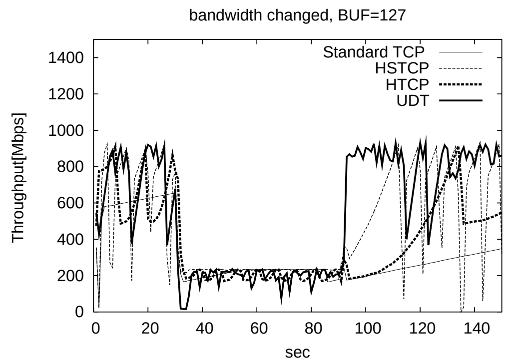

Fig. 6. Path switching(Bandwidth[Mbps]=1000 → 250 → 1000)

seconds had passed since the start of the flow, and then, the path was switched back to the original one after 60 seconds had passed since the first switching. The socket buffer size was set to the maximum bandwidth-delay product, i.e., the longer RTT (80 [ms]) $\times$ the bandwidth (1 [Gbps]).

At the moment of the change to the alternative path with a longer RTT, the throughput of each of the TCP-based protocols decreased near to half (which might be due to a few packet losses) and gradually recovered its original high throughput, while that of UDT seemed insensitive to this change. On the other hand, at the moment of return to the original path with a shorter RTT, the throughput of each of UDT and the TCP-based protocols decreased and then quickly recovered its original behavior.

In Figure 6, we also observed the throughput behavior in case that the path was switched from the original path with 1 [Gbps] of the bottleneck bandwidth to the alternative path with 250 [Mbps] of that just after 30 seconds had passed since the start of the flow, and then, the path was switched back to the original one after 60 seconds had passed since the first switching. The socket buffer size was set to the maximum bandwidth-delay product, i.e., the RTT (38 [ms]) $\times$ the maximum bottleneck bandwidth (1 [Gbps]).

At the moment of the change to the alternative path with a smaller bottleneck bandwidth, the throughput of every protocols became considerably unstable, which was likely due to a too large congestion window. On the other hand, at the moment of return to the original path with a larger bottleneck bandwidth, the throughput of every protocols recovered its original behavior, where no packet loss seemed to happen.

D. coexisting flows which have different RTT values

In the configuration shown in Fig. 1(b), we started two high-speed transport protocol flows (flow1 and flow2) simultaneously, where the ratio of RTT of flow2 to that of flow1 was set to one, two, or four by using the network emulator. Figure 7 shows the time-averaged throughput of these two coexisting flows with different RTTs where RTT of flow 1 is 38 [ms] and that of flow 2 is 38, 76, or 152 [ms]. Note that similar experiments were already reported in the literature [11] in which the bottleneck bandwidth was much lower than that in our case.

It can be observed that the UDT is insensitive to RTT and the TCP-based transport protocols suffer from severe inefficiency as well as unfairness in throughput performance if two flows have different RTTs. In fact, when the RTT of flow 2 is set to four times larger than that of flow 1, the achievable throughput of flow 2 significantly decreases, while that of flow 1 increases just a little.

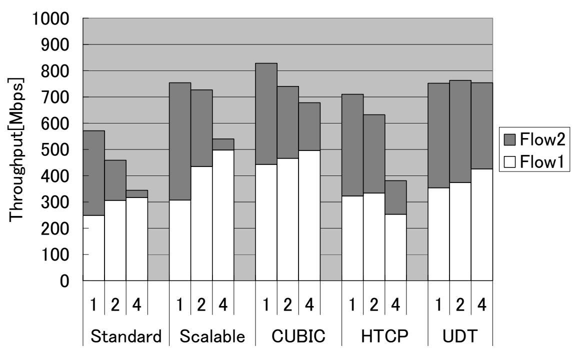

Fig. 7. Throughput characteristics of flows with different RTTs

E. Coexistence of long-lived TCP flows

We first examined the scenario in which a long-lived Standard TCP and a high-speed transport protocol flow coexist. We observed the well-known unfairness problem: that is, a high-speed transport protocol flow starved the long-lived Standard TCP flow for bandwidth, and the performance of the high-speed transport protocol merely degraded.

We also examined that throughput characteristics when the two high-speed transport protocol flows coexist in the path. We performed the simultaneous runs of two flows. All the combinations of two protocols taken among the high-speed transport protocols were examined, and their sum of average throughput of two flows over 300 seconds is shown in Fig. 8(a), where a flow by a protocol indicated in X-axis and a flow indicated by an indicator coexist. We found that the sum of the throughput of different kind of high-speed transport protocol flows were smaller than those of coexisting identical high-speed transport protocol flows, i.e., the link utilization degrades when different kinds of high-speed transport protocol flows coexist.

Figure 8(b) shows average throughput of each flow in case of coexisting a HSTCP flow and an other high-speed transport protocol flow. It is clearly observed that there are unfairness in throughput. The unfairness problem were found in all cases of coexisting different kind of high-speed transport protocol flows. In particular, UDT protocol flow significantly affected the performance of coexisting TCP-based protocol flows.

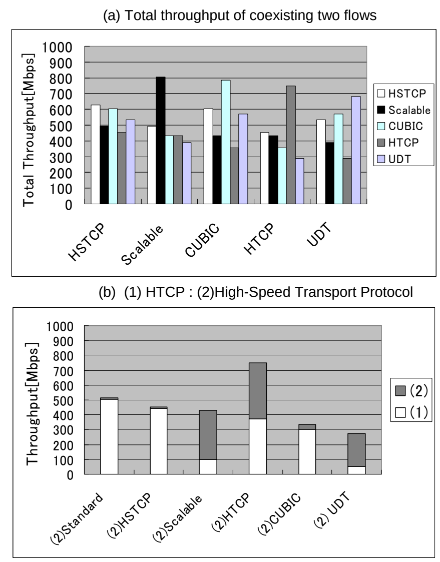

Fig. 8. Coexisting two kinds of high-speed transport protocol flows

F. Coexistence of short-lived TCP flows

We examined the throughput performance in case that short-lived TCP flows and a high-speed transport protocol flow coexist. In [2], we mainly focused on the performance of short-lived TCP flows in similar scenarios and showed that the degree of the degradation of performance of short-lived TCP flows could be mitigated by setting socket buffer size for a high-speed transport protocol flow to appropriate values. In the following, we investigated not only the performances of short-lived TCP flows, but also those of the high-speed transport protocol flows.

We performed simultaneous runs of a single long-lived flow using one of the high-speed transport protocols and 3000 short-lived flows using Standard TCP over the Japan-US international line as illustrated in Fig. 1(d). The size of each short-lived flow followed a Pareto distribution with the mean of 100, 300 or 500 [KB], with the shape parameter set to 1.3. The starting time of each transfer was randomly selected within the duration of 300 seconds.

Figure 9(a) shows the averaged throughput over 300[s] of the high-speed transport protocol flow coexisting with short-lived TCP flows, in case that the output buffer size of the edge routers is 127. The leftmost side bars in each group on the bar graph show the throughput when there was only a single high-speed transport protocol flow on the path. It is clear that the performance of the high-speed transport protocol flow was considerably affected by the coexisting short-lived TCP flows, even through the amount of these flows is small. The larger the averaged file size of the short-lived TCP flow was, the larger the observed damage in the high-speed transport protocol flow became, except for UDT flow. The UDT flow was also affected by coexisting short-lived TCP flow, but its degree of degradation was relatively small.

Figure 9(b) shows the averaged throughput of the short-lived TCP flows defined by $(\sum S_i)/(\sum t_i)$, where S_i denotes the file size of flow i and t_i denotes the transfer time of flow i, respectively. The larger the averaged file size became, the higher the averaged throughput of short-lived TCP flows achieved. However, in case of averaged file size of 500[KB], the averaged throughput of short-lived TCP flows coexisting with a high-speed transport protocol flow was smaller compared when the short-lived TCP flows run without coexisting a high-speed transport protocol flow. Figure 10 plots the time-series of throughput of high-speed transport protocol flows in case that the short-lived TCP flows with an average file size of 500[KB] were randomly generated during 50[s] and 350[s] periods.

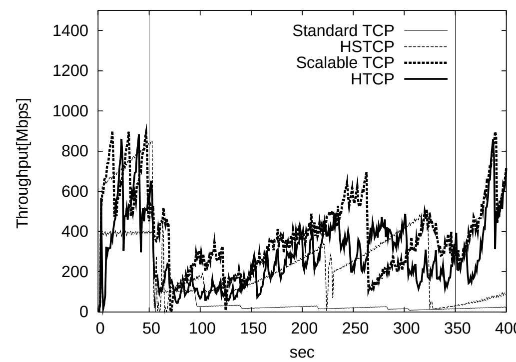

Fig. 10. Throughput characteristics of high-speed transport protocol flows coexisting with short-lived TCP flows during 50-350[s]

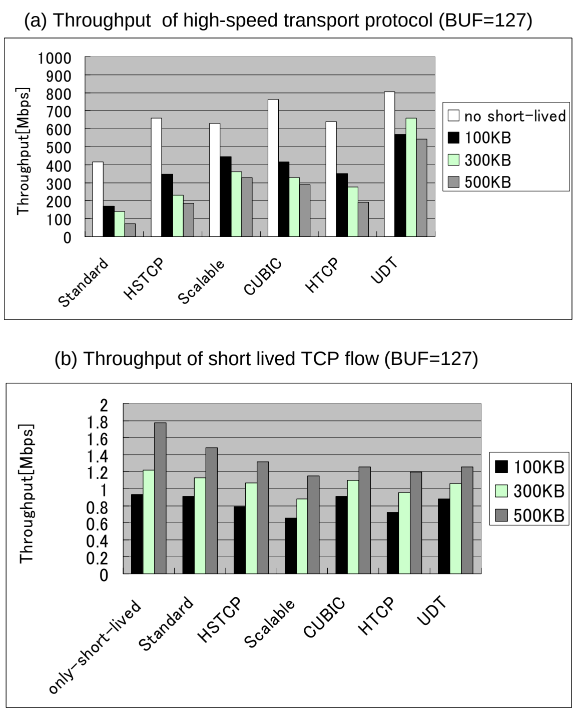

Fig. 9. Throughput of high-speed flows and short-lived TCP flows

Figure 11 shows the average throughput of a high-speed transport protocol flow (shown in (a)) and short-lived TCP flows (shown in (b)) when the buffer size at the edge routers is 127 or 512 in case that averaged file size of short-lived TCP flows is 500[KB]. It is observed that by setting the buffer size large at the edge routers, the achievable throughput of coexisting flows were improved to some extent.

These results indicated that coexisting short-lived standard TCP flows could considerably damage the performance of high-speed TCP-based transport protocol flows although co-existing long-lived standard TCP flows could not so. This may be because the slow-start phases of short-lived TCP flows randomly change the available bandwidth. The performance of short-lived standard TCP flows with relatively large file size was also adversely affected by the coexisting high-speed transport protocol flows. Setting the buffer size at the bottleneck nodes larger could mitigate the degradation of throughput of both high-speed TCP and short-lived TCP flows.

G. Coexistence of CBR UDP flows

We performed simultaneous runs of a single long-lived flow by one of the high-speed transport protocols and two CBR (constant bit rate) streams by the UDP protocol on the path shown in Fig. 1. Each stream consisted of 200 [byte] UDP packets sent at the rate of 1.6, 3.2 and 8 [Mbps] (representing 64 [Kbps] $\times$ n flows, corresponding to the cases where n = 25, 50 and 125). In this subsection, we present the results of two scenarios as follows: (1) two UDP flows start first, followed by a single high-speed transport protocol flow starting at 30 [s] (case 1); and (2) a single high-speed transport protocol starts first, and two UDP flows are injected at 30 [s] (case 2).

Figure 12 illustrates the packet loss rate observed in 8[Mbps] UDP flows when they coexist with a high-speed transport protocol flow in case 2. When there were only UDP flows in the path or they coexist with a Standard TCP flow, no packet loss occurred. However, the UDP flows coexisting with a high-speed transport protocol flow seen few packet losses.

Figure 13(a) shows the jitter characteristics of two 8[Mbps] UDP flows observed in case 1 when the buffer size at the edge routers is 127 or 512. Comparing the averaged jitter of two CBR flows with a coexisting high-speed transport protocol flow and without that (indicated by "only UDP flows"), it is obvious that the jitter of the CBR flows is affected by the coexisting high-speed flow.

Figure 13(b) shows the average throughput over 300 [s] of high-speed transport protocol flows when the buffer size at the edge routers is 127 or 512. The each grouped bar graph, the bar labeled "no UDP" shows the throughput characteristics of each high-speed transport protocol flow without any coexisting UDP flow. Throughput of high-speed transport protocol flows

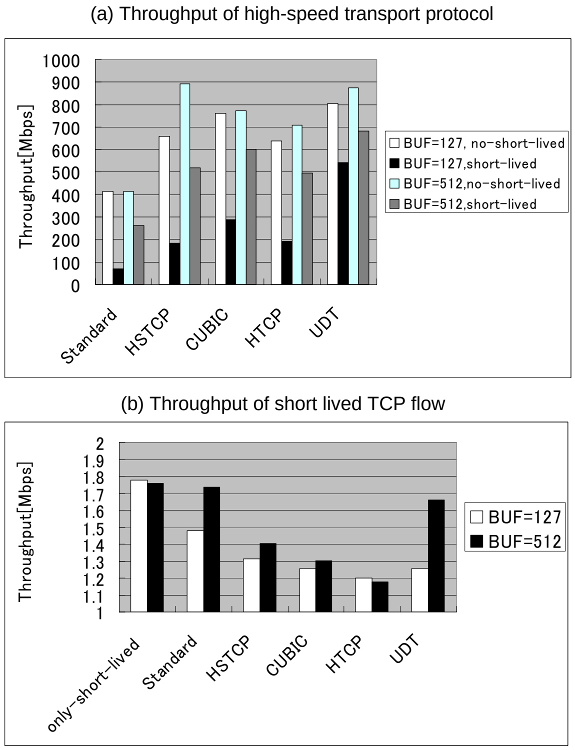

Fig. 11. Effect of buffer size at edge routers

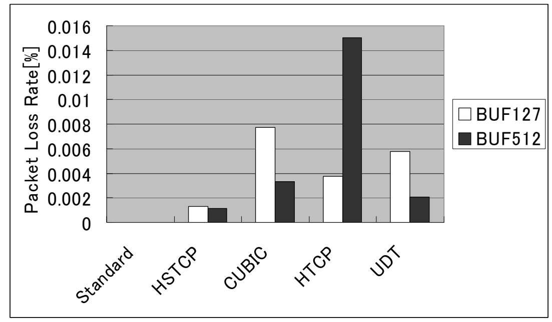

Fig. 12. Packet loss rate observed on UDP flows

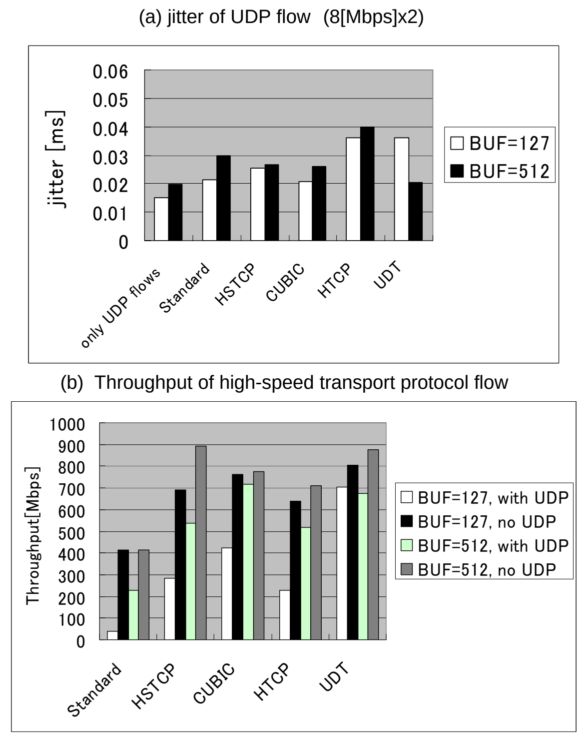

Fig. 13. Jitter of UDP flow and throughput of high-speed transport protocol flow in case 1

were more or less affected by coexisting UDP flows. However, compared with TCP based high-speed transport protocols, the degree of the degradation observed in the UDT flow is smaller.

Setting the buffer size at the edge routers larger, adverse influence of UDP flow to the throughput of coexisting high-speed transport protocol flow were mitigated.

Figure 14 depicts the performance characteristics observed in case 2. The similar tendencies were observed in the jitter of UDP flows and in the throughput characteristics of the high-speed transport protocol flow, as in case 1, except for the throughput of Standard TCP flow. Since Standard TCP flow does not increase its window size aggressively during its congestion avoidance phase, it might not be affected by coexisting UDP flows in case 2. On the other hand, in case 1 where the UDP started first, the Standard TCP flow competed the bandwidth with UDP flows during its slow start phase.

Figure 15 shows throughput characteristics of a single flow by each of two high-speed transport protocols (HSTCP and CUBIC) in cases of no UDP flows coexisting, of UDP flows starting beforehand (case 1), and of UDP flows starting afterward (case 2), respectively, in case that the buffer size at the edge routers is 127. For both TCP protocols, in case 1, the throughput of the TCP flow could not increase rapidly at the slow start phase, which may be due to the packet losses caused by the coexisting UDP flows. In case 2, although the UDP flows were injected after the TCP flow entered the congestion avoidance phase, occasional packet losses occurred when the flow throughput increased near to the maximum bandwidth. This prevented the TCP flow from achieving a good average throughput. We observed similar tendencies in case that the buffer size at the edge routers is 512.

These results demonstrated that the throughput of high-speed transport protocols were adversely affected by coexisting UDP flows even if the load of the UDP flows was only 16[Mbps]. We also observed the UDP flows with a lower load(3.2 or 6.4[Mbps]) could affect the performance of coexisting high-speed transport protocols in case of 127 buffer size. Setting the buffer size of the edge routers larger

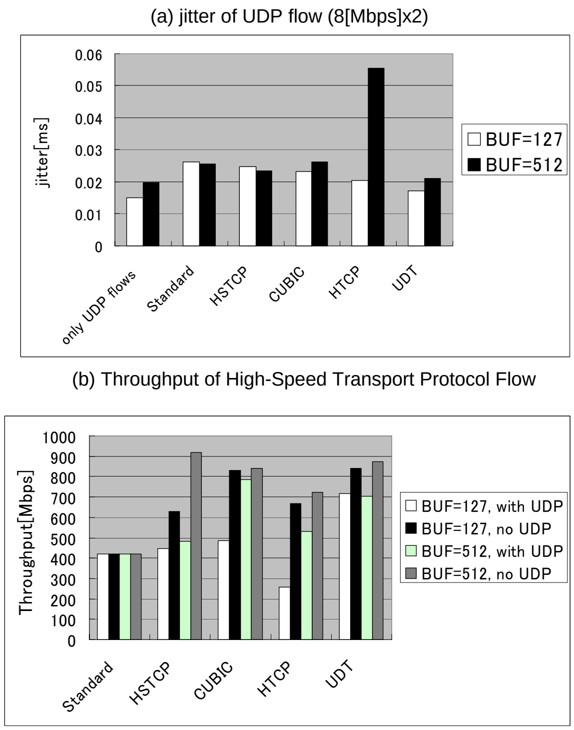

Fig. 14. Jitter of UDP flow and throughput of high-speed transport protocol flow in case 2

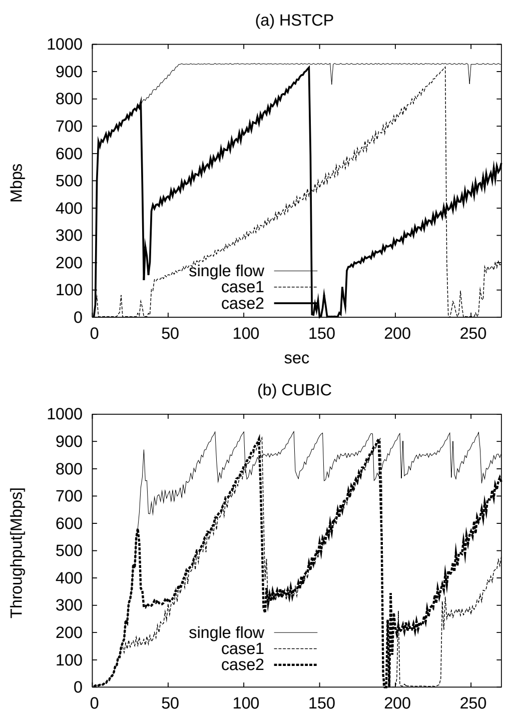

Fig. 15. Throughput characteristics of high-speed transport protocol coexisting UDP flows

could mitigate such throughput degradations of TCP-based high-speed transport protocols, while making the jitter of UDP flows larger in most cases.

IV. Concluding remarks

We are investigating what happens if high-speed transport protocols are used in the global Internet, through experiments on the JGN II, an open 10Gbps-class network testbed in Japan. Our results (in cases with 1 [Gbps] end-hosts) indicated that, when long-lived data transfer flows of high-speed transport protocols run in coexistence with Internet application traffic such as short-term web browsing flows or long-term video streaming flows, not only the performance of web access or video streaming is degraded, but also that of the high-speed transfer flows is considerably degraded even if the amount of this application traffic is low. In other words, such circumstances are neither effective nor efficient in terms of bandwidth sharing.

Our ultimate goal is to identify a feasible and cost-effective goal itself we should pursue for the future Internet where the demanding high throughput data transfer must coexist with various other application traffic over shared heterogeneous networks, and to finally propose ways to realize such a desirable coexistence by an improved high-speed data transport protocol and/or by a new management mechanism in the intermediate nodes.

We thank the NCDM team of UIC and Dr. Injong Rhee and Dr. Lisong Xu for their helpful comments and technical advices. We are also grateful to the APAN/JGN2 NOC members for their assistance in conducting the experiments. This work was supported in part by the Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research (A) (15200005).

References

[1] R.L. Cottrell, S. Ansari, P. Khandpur, R. Gupta,R.Hughes-Jones,M.Chen, L.McInttosh, F.Leers, “Characterization and Evaluation on TCP and UDP-based Transport on Real Networks,” PFLDnet2005, Feb. 2005.
[2] K Kumazoe, K. Kouyama, Y. Hori, M. Tsuru, Y. Oie, “Transport Protocols for Fast Long-Distance Networks:Evaluation of Their Penetration and Robustness on JGNII,“ PFLDnet2005, Feb. 2005.
[3] S. Floyd, “HighSpeed TCP for Large Congestion Windows,” RFC 3649, Experimental, December 2003.
[4] T. Kelly, “Scalable TCP: Improving Performance in Highspeed Wide Area Networks,” Computer Communication Review 32(2), April 2003
[5] http://netlab.caltech.edu/FAST
[6] http://www.csc.ncsu.edu/faculty/rhee/export/bitcp/
[7] Hamilton Institute Net Lab, http://www.hamilton/ie/net/
[8] UDT, http://udt.sourceforge.net
[9] JGNII, http://www.jgn.nict.go.jp/e/index.html
[10] TCP Tuning Guide, http://www-didc.lbl.gov/TCP-tuning/
[11] “Experimental Evaluation of TCP Protocols for High-Speed Networks”, Yee-Ting Li, D. Leith, R. N. Shorten, 2005, http://www.hamilton.mav.ie/net/eval/results_HI2005.pdf