Proposal of Novel MPTCP Congestion Control to Suppress QoS Fluctuation for WebQoE Improvement

Kensuke Noda
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya, Japan
noda@en.nitech.ac.jp

Yoshihiro Ito
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya, Japan
yoshi@nitech.ac.jp

***Abstract*—This paper proposes a new congestion control of Multi–Path TCP (MPTCP) for improvement of Quality of Experience (QoE). This method controls the congestion window for each subflow so as to suppress the fluctuation of Quality of Service (QoS). The fluctuation of QoS is estimated from that of the Round Trip Time, and the congestion window is controlled according to the estimated fluctuation. The authors implemented the proposed method and evaluate QoS by experiment. The experimental results show that the proposed method can suppress the fluctuation of QoS as compared with the existing congestion controls of MPTCP under congestion.**

***Index Terms*—MPTCP, Congestion Control, QoS**

I. Introduction

Today, we can select an access network of a mobile terminal from multiple networks, such as Wi-Fi, LTE, and so on. By using plural networks simultaneously, we expect to disperse the traffic of the mobile networks that keeps increasing rapidly. This dispersion will produce the suppression of congestion and then the improvement of the communication quality.

However, TCP[1], which is the main stream of the current transport layer protocol of the Internet, can handle only one path per connection. Consequently, even if we can utilize multiple networks, TCP cannot handle them simultaneously.

In order to solve the above–mentioned problem, Multi–Path TCP (MPTCP)[2] has been standardized as a next–generation transport layer protocol; it can treat multiple paths at once. Therefore, it is expected to provide stable and fast communication.

On the other hand, similarly to TCP, MPTCP performs window–based congestion control using a congestion window[3]. Since a congestion control generally affects Quality of Service (QoS) of MPTCP, many congestion controls for MPTCP have been proposed to improve QoS[4][5][6]. Moreover, in a hierarchical network model, QoS of a layer affects that of its upper layers; it finally influences Quality of Experience (QoE). That is, the congestion control of MPTCP affects QoS of the layers higher than transport layer and QoE.

Shibata, et.al investigated the influence of QoS fluctuation due to the difference between TCP congestion controls on QoE for a Web service (WebQoE) in TCP[7]; they confirmed that a congestion control which suppresses QoS fluctuation can provide higher WebQoE. Therefore, in the same way, we expect MPTCP to provide higher WebQoE by adopting a congestion control that can reduce fluctuation of QoS. However, almost all the existing congestion controls for MPTCP are aimed only at improving the mean of the throughput.

This paper proposes a novel MPTCP congestion control which suppresses the fluctuation of QoS rather than improving the throughput for WebQoE improvement; it implements the proposed congestion control and evaluates the effectiveness by experiment.

The rest of this paper is organized as follows. Section II gives an overview of MPTCP and its congestion control. We explain proposed scheme in Sect. III. Section IV and Sect. V show our experiment and its results, respectively. Finally, we conclude our research in Sect. VI.

II. MPTCP

MPTCP has been standardized as a next–generation transport layer protocol[2]; it can use multiple TCP flows, which are called *subflows*, over multiple paths.

MPTCP has a window–based congestion control. A congestion control is generally classified into the following two types according to its algorithm which determines the congestion window size. One is the loss–based control and the other is the delay–based one. For example, *LIA*[4] and *OLIA*[5] are loss–based control while *WVEGAS*[6] is a delay–based one.

III. PROPOSAL

We propose a new congestion control for MPTCP to suppress the fluctuation of QoS rather than improving the mean of throughput as follows. Note that we consider the round trip time (RTT) as an index of QoS.

In the proposed method, when a sender receives an ACK segment, it calculates the queueing delay of the subflow as shown in Eq. (1) and use it to evaluate the degree of network congestion.

$$Queueing_k = RecentRTT_k - BaseRTT_k \quad (1 \leq k \leq n) \tag{1}$$

Here, n represents the number of subflows which are established in the MPTCP connection. Also, $Queueing_k$ is the queueing delay of the k-th subflow. $RecentRTT_k$ means the minimum value of RTT obtained by the latest observation in the k-th subflow, and $BaseRTT_k$ indicates the minimum value of RTT obtained by the observation after the connection establishment in the k-th subflow. Then, for the k-th subflow, we calculate the mean of RTT ($MeanRTT_k$) and that of the queueing delay ($MeanQueueing_k$) by Eq. (2) and Eq. (3), respectively.

$$MeanRTT_k = 0.8 \times MeanRTT_k + 0.2 \times RecentRTT_k \quad (2)$$

$$MeanQueueing_k = MeanRTT_k - BaseRTT_k \quad (3)$$

Finally, by comparing $Queueing_k$ with $MeanQueueing_k$, the sender updates the congestion window size of the k-th subflow ($cwnd_k$) as shown in Eq. (4). Here, the unit of $cwnd_k$ is the maximum segment size (MSS) of the k-th subflow.

$$cwnd_k = \begin{cases} cwnd_k + \frac{1}{2 \times cwnd_k} & (Queueing_k < MeanQueueing_k) \\ cwnd_k - \frac{1}{cwnd_k} & (Queueing_k \geq MeanQueueing_k) \end{cases} \quad (4)$$

By this method, it is possible to control the $cwnd_k$ so as to suppress the fluctuation of RTT of each subflow. The pseudo–code of the above–mentioned proposed is presented in the following.

Algorithm 1 The pseudo–code of proposed algorithm

```
if ACK is received then
    $Queueing_k = RecentRTT_k - BaseRTT_k$
    $MeanRTT_k = 0.8 \times MeanRTT_k + 0.2 \times RecentRTT_k$
    $MeanQueueing_k = MeanRTT_k - BaseRTT_k$

    if $Queueing_k < MeanQueueing_k$ then
        $cwnd_k \Leftarrow cwnd_k + \frac{1}{2 \times cwnd_k}$

    else $\{Queueing_k \geq MeanQueueing_k\}$
        $cwnd_k \Leftarrow cwnd_k - \frac{1}{cwnd_k}$

    end if
end if
```

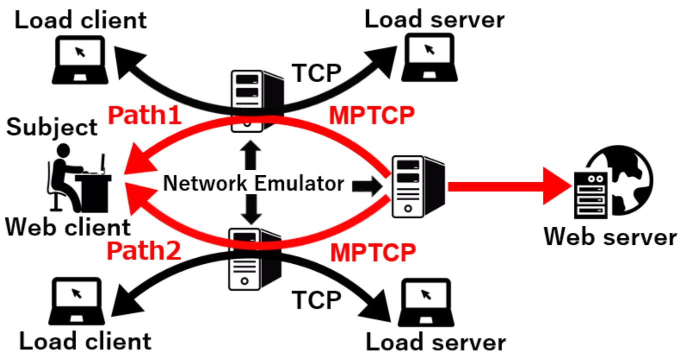

Fig. 1: Experimental Environment

TABLE I: Parameter Values of Network Emulators

Env.1, Env.2, Env.3			
	bandwidth	packet loss rate	delay
Path1	10Mbps	3%	50ms
Path2	10Mbps	3%	50ms

Env.4, Env.5, Env.6			
	bandwidth	packet loss rate	delay
Path1	10Mbps	1%	100ms
Path2	10Mbps	3%	50ms

TABLE II: Number of TCP Connections

Env.1	Env.2	Env.3	Env.4	Env.5	Env.6
10	20	From 10 to 20	10	20	From 10 to 20

IV. EXPERIMENT

We evaluate QoS of the proposed method by experiment. Figure 1 shows our actual experimental environment.

In Fig. 1, the subject accesses the Web server from the Web client which is connected to the Web server via the network emulators; he/she acquires specified Web pages. The Web client has two network interfaces. The network emulators operate as routers and set two paths (Path1 and Path2) between the Web client and the Web server. These two paths are used by MPTCP communication.

In this experiment, in order to compare the performance of the proposed method and those of existing methods, we consider the four MPTCP congestion controls; the proposed method, *LIA*, *OLIA* and *WVEGAS*. The proposed method is implemented on the Web server. The experimenter changes the congestion control of the Web server. Note that the Web client uses *LIA* which is standard in MPTCP as congestion control.

Each network emulator gives delay and packet losses to packets which pass through it to change the communication quality of each path. We utilize Dummynet[8] as network emulators. The network emulators are also connected between the load client and the load server. They generate load traffic of TCP between them to make Path1 and Path2 congested. We treat six different environments which are indicated in Table I; we change the number of TCP connections as shown in Table II. It should be noted that, in Table I, Env.1, Env.2 and Env.3 are uniform environments whose communication quality of multiple paths are equivalent to each other while Env.4, Env.5 and Env.6 are heterogeneous ones.

V. RESULT

We plot our results in Fig. 2 through Fig. 5. In these figures, the abscissa means the experimental environments and confidence interval of 95% is also indicated. Here, *Proposal* in these figures means the proposed method.

First, we show the variance of RTT in Fig. 2 and Fig. 3. From these, we see that the variance of RTT of *Proposal*

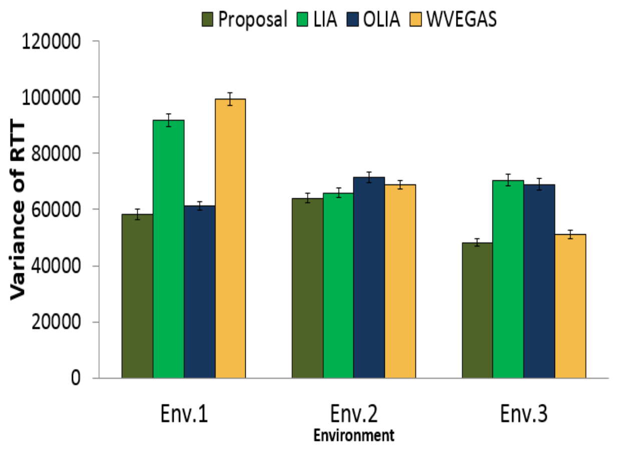

Fig. 2: Variance of RTT from Env.1 to Env.3

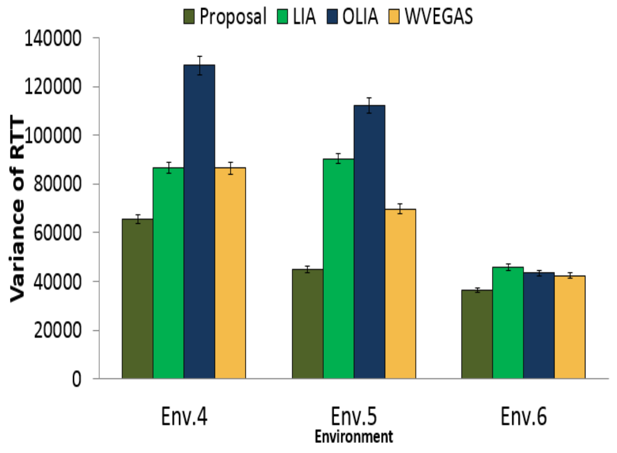

Fig. 3: Variance of RTT from Env.4 to Env.6

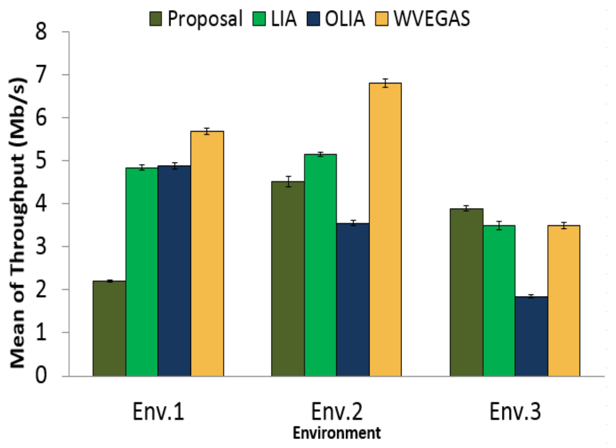

Fig. 4: Mean of Throughput from Env.1 to Env.3

is smaller than those of other congestion controls under all the experimental environment. These results indicate that the proposed method can suppress the fluctuation of RTT than the existing methods under not only uniform environments but also heterogeneous environments.

Next, we show the mean of throughput in Fig. 4 and Fig. 5. These figures show that the mean of the throughput of *Proposal* is lower than those of other congestion controls

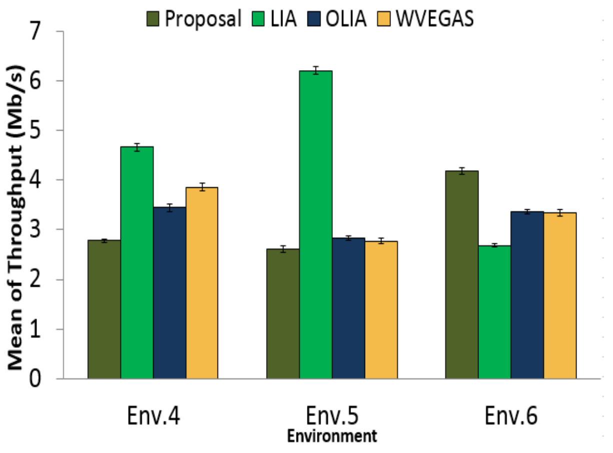

Fig. 5: Mean of throughput from Env.4 to Env.6

under Env.1, Env.4 and Env.5. However, under Env.2, Env.3 and Env.6, *Proposal* did not reduce the mean of throughout than others. The reason for this is that *Proposal* can maintain its congestion window size of all subflows more stably than others in environments where the packet losses and increase of the delay occur frequently due to congestion.

From the above–mentioned issues, we can conclude that our proposed method can suppress the fluctuation of QoS than existing methods.

VI. CONCLUSIONS

This paper proposed a new congestion control of MPTCP which suppresses the fluctuation of QoS for WebQoE improvement. We implemented the proposed method and evaluate its QoS by actual experiments. From our experimental results, we confirmed that the proposed method can suppress the fluctuation of QoS as compared with the existing congestion controls.

As our future work, we will evaluate WebQoE using the proposed method under various environments.

ACKNOWLEDGMENT

This work was supported by MEXT KAKENHI Grant Number JP 16K00122. The authors thank the member of our laboratory for their assistance in our research. They also thank the subjects who joined our experiments.

REFERENCES

[1] J. Postel, “Transmission Control Protocol,” IETF RFC 793（Internet standard）, Sept. 1981.
[2] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, “TCP Extensions for Multipath Operation with Multiple Addresses,” IETF RFC 6824, Jan. 2013.
[3] M. Welzl, “Network Congestion Control： Managing Internet Traffic,” W.S. Kingdom, ed., John Wiley and Sons, 2005.
[4] C. Raiciu and M. Handley and D. Wischik, “Coupled Congestion Control for Multipath Transport Protocols,” IETF RFC 6356, Oct. 2011.
[5] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, and J.Y. Le Boudec, “MPTCP is Not Pareto-optimal: Performance Issues and a Possible Solution,” Proc. CoNEXT ‘12, USA, pp.1-12, Dec. 2012.
[6] Y. Cao, M. Xu, and X. Fu, “Delay-based Congestion Control for Multipath TCP,” Proc. ICNP, USA, pp.1-10, Oct. 2012.
[7] Masaaki Shibata and Yoshihiro Ito, “Effect of TCP congestion control on WebQoE,” Proc. IA workshop, Jan. 2015.
[8] L. Rizzo, “Dummynet,” http://info.iet.unipi.it/luigi/dummynet/.