Active Queue Management Supporting TCP Flows Using Dynamically Controlled Target Queue Length

Ryosuke Hotchi and Ryogo Kubo
Department of Electronics and Electrical Engineering, Keio University, Japan

***Abstract*—Active queue management (AQM) is a congestion control scheme for transmission control protocol (TCP) flows. However, the appropriate adjustment of a target queue length has not been studied and discussed for proportional-integral-derivative (PID)-based AQM controllers. This paper proposes an AQM technique using a PID-based controller and an algorithm that dynamically controls the target queue length in order to more efficiently utilize the buffer capacity of the bottleneck router. Simulation results show that the proposed AQM using a dynamically controlled target queue length outperforms the conventional AQM using a constant target queue length in terms of buffer utilization efficiency.**

I. Introduction

Recently, the number of devices connected to the Internet and the capacity of communication contents are increasing at an accelerating rate. This implies that the amount of Internet traffic needed for communication has been increasing rapidly, which will result in serious traffic congestion. The majority of Internet communication is performed using transmission control protocol (TCP) as the communication protocol. TCP has many functions such as data error detection using checksum and retransmission of lost packets, and is known for its high reliability.

In the router, a method called DropTail is used as a default congestion control method. However, because of its characteristics, the DropTail method combined with TCP has several problems. In order to deal with these problems, a method called active queue management (AQM) has been proposed. As a method to realize a stable congestion control system, an AQM based on control theory has been proposed [1]. Additionally, as an extension to the study of this control-theory-based AQM, a congestion control system using a proportional-integral-differential (PID) controller has been proposed [2].

AQM based on control theory utilizes a parameter called target queue length, which is a value that the system attempts to keep the actual queue length close to. Traditionally, this value has been set to half the buffering capacity, thus creating a vacant space in the router's buffer. In this paper, we propose an algorithm that dynamically controls the target queue length in order to effectively utilize the buffering capacity of the bottleneck router.

II. Congestion Control

A. DropTail

DropTail is the default congestion control method utilized in the bottleneck router. This method drops packets only after the buffering capacity of the bottleneck router is full, which means TCP flows can detect congestion only after buffer overflow. This characteristic of the DropTail method is known to have various problems, such as inducing mass packet disposal, leading to global synchronization, thus making bursty traffic more likely to be affected by congestions.

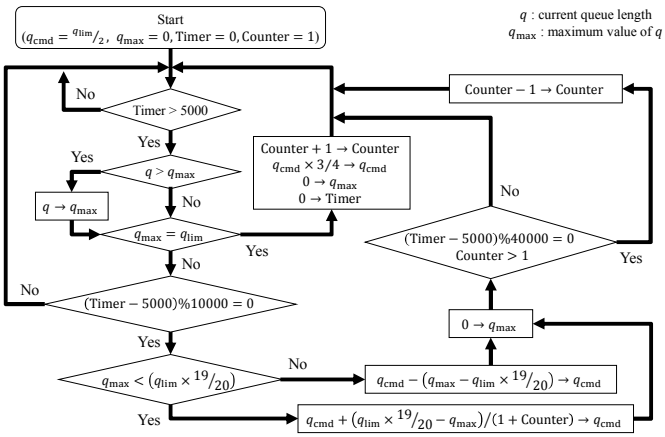

Fig. 1. Proposed algorithm to determine the target queue length

B. Active Queue Management

To deal with the problems of the DropTail method, AQM has been proposed. AQM is a method that discards packets actively and randomly before a serious congestion occurs and attempts to maintain the queue length (the number of packets buffered in the router's buffer) stable. With the AQM method, the queue length is kept lower than the buffering limit of the router, which resolved various problems of the DropTail method.

There are several types of AQM schemes. In this paper, we consider an AQM based on control theory, which utilizes a PID controller proposed by Hollot et al. AQM using PID controller utilizes a parameter called "target queue length," which is a value that the system attempts to keep the queue length close to.

III. Dynamically Controlled Target Queue Length

Traditionally, the target queue length of AQM using PID controller was set to half the buffering limit of the bottleneck router, in order to maintain enough vacant space for queue length over shoot and under shoot. However, in most circumstances, this value would leave unused buffering space in the router. In order to fully utilize the buffering limit of the bottleneck router, we proposed an algorithm that attempts to increase the target queue length.

While the algorithm attempts to increase the target queue length, it also needs to avoid the queue length reaching the buffering limit. If the queue length reaches the buffering limit, a situation similar to that encountered with the DropTail method results. This means that the AQM fails to detect congestion before buffer overflow, thus inducing the same problems encountered when using the DropTail method.

When the algorithm attempts to increase the target queue length while avoiding buffer overflow, the effect of queue

length overshoot must be considered. The oscillation of queue length is an inevitable aspect of AQM, even with a very well designed PID controller. Thus, a rapid increment in target queue length will lead to a massive overshoot of queue length, which may trigger buffer overflow. The proposed algorithm considers this fact, and increase the target queue length gradually to avoid overshoot.

Figure 1 shows the flow chart of the proposed algorithm. q,$q_{\max}$,$q_{\lim}$, and q_{cmd} denotes the current queue length, the maximum value of queue length in certain period (10 seconds in this paper), the buffering limit of the bottleneck router, and the target queue length, respectively. The "Timer" shown in the figure records time in milliseconds.

First, the algorithm sets q_{cmd} to half of $q_{\lim}$ and all other variables to 0 as an initialization. Then, after the timer records over 5000 ms, it starts recording $q_{\max}$. If at any moment $q_{\max}$ reaches $q_{\lim}$, q_{cmd} is dropped to its 3/4 of the value, the counter is incremented by 1, and both $q_{\max}$ and the timer are reset to 0. Using this function, the algorithm avoids the problems induced by buffer overflow such as those of the DropTail method.

After every 10 s after starting the recording of $q_{\max}$, which has never reached $q_{\lim}$, the algorithm starts updating q_{cmd}. It compares the current q_{cmd} with the value of $q_{\lim} \times 19/20$, and if the current q_{cmd} is larger than that, it is incremented by $(q_{lim} \times 19/20 - q_{\max})/(1 + \mathrm{Counter})$. If it is smaller, it is decremented by $q_{\max} - q_{lim} \times 19/20$. After this procedure, $q_{\max}$ is reset to 0. Also, after every 3 updates, the algorithm checks the value of the counter, and if its value is larger than 1, it will be decremented by 1. Through these procedures, q_{cmd} will be increased to a higher value, while avoiding serious buffer overflow.

IV. Simulation

In order to validate the utility of the proposed algorithm, the network simulation software ns-2 was utilized.

A. Simulation conditions

In the simulation, the network topology shown in Figure 2 was used. As shown in Figure 2, there are 100 nominal TCP sessions, and the link capacity of the sending side, receiving side, and bottleneck link are all 10 Mbps. The round trip time (RTT) of the network is 30 ms. The packet size is 1000 bytes and the buffering capacity of the bottleneck router is 200 packets. The simulation was performed at a control frequency of 0.001 s. As mentioned before, we used a PID controller as the AQM controller. The parameters of the PID controller are as follows: proportional gain $K_{\mathrm{p}} = 900$, integral gain $K_{\mathrm{i}} = 700$, and differential gain $K_{\mathrm{d}} = 55$. The cutoff frequency used for pseudo-differential calculation was set to 50 rad/s. All of these parameters can be referred to in [3].

B. Simulation results

Figures 3 and 4 show the graphs of the simulation results. Figure 3 shows the simulation result with the conventional constant target queue length, and Figure 4 shows the simulation results obtained using the proposed dynamical target queue length control algorithm. These graphs show that the proposed algorithm successfully raises the queue length, more efficiently utilizing the buffering capacity, while avoiding the queue length reaching the buffering limit.

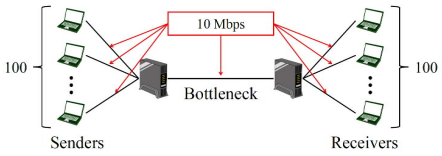

Fig. 2. Network topology

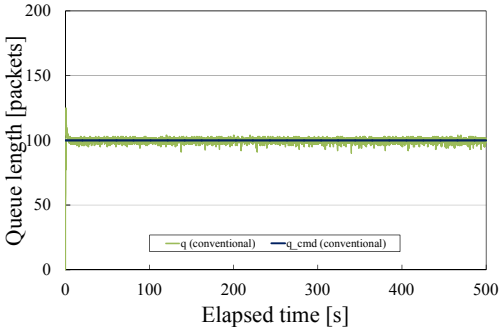

Fig. 3. Simulation result (constant target queue length)

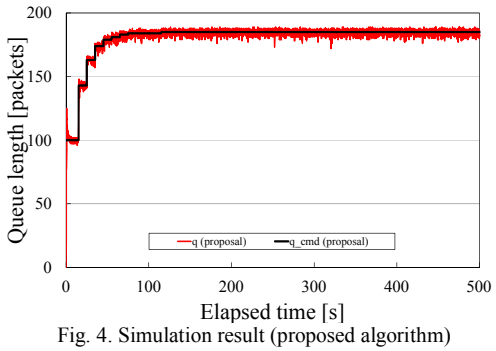

Fig. 4. Simulation result (proposed algorithm)

This efficient usage of the buffering capacity is due to the function of the proposed algorithm where q_{cmd} is raised every 10 s. Under this simulation setup where the RTT is relatively low, the TCP/AQM system was stable regardless of the current q_{cmd}, and buffer overflow did not occur.

V. Conclusion

In this paper, we proposed a TCP/AQM control scheme using a dynamically controlled target queue length. The simulation results showed that the proposed algorithm could effectively increase the target queue length, making use of the buffering capacity of the bottleneck router more efficiently, while avoiding the queue length reaching the buffering limit.

Acknowledgement

This research was supported in part by JSPS KAKENHI Grant Number 16K16049 and Kenjiro Takayanagi Foundation.

References

[1] C.V. Hollot, V. Misra, D. Towsley, and W. Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Trans. Autom. Control*, Vol. 47, No. 6, pp. 945–959, June 2002.

[2] A. Haider, H. Sirisena, and K. Pawlikowwski, "PID based congestion control algorithms for AQM routers supporting TCP/IP flows," *IEICE Trans. Commun.*, Vol. E87-B, No. 3, pp. 548–555, Mar. 2004.

[3] R. Kubo, J. Kani and Y. Fujimoto, "Advanced Internet congestion control using a disturbance observer," in *Proc. IEEE Global Communications Conf. (GLOBECOM 2008)*, pp. 1-5, Nov. 2008.