Vegas Plus: Improving the Service Fairness

Jin-Ru Chen and Yaw-Chung Chen, *Member, IEEE*

***Abstract*—TCP Vegas is a congestion avoidance scheme designed to prevent the periodic packet loss which occurs in traditional schemes. Since Vegas successfully avoids such packet loss, it achieves much higher throughput than TCP Reno. However, it does not concern the fairness among source-destination pairs with different round-trip times (RTT's). In this letter, we propose a different mechanism to adjust the window size, this allows TCP to provide much better fairness regardless the large variation of RTT's.**

***Index Terms*—Delay-control, fairness, Vegas.**

I. Introduction

TRANSMISSION Control Protocol (TCP) is the most widely used end-to-end transport protocol with congestion avoidance capability. A variety of congestion avoidance schemes have been proposed to improve the performance of TCP. These schemes developed after TCP Tahoe can be categorized into three approaches, to avoid the bandwidth waste caused by inefficient loss recovery, to avoid the retransmission of successfully transmitted packets, and to avoid the periodic packet loss caused by self-generated congestion. Reno [6] and New-Reno [2] belong to the first, SACK [4] and FACK [5] belong to the second, and Vegas [1] belong to the third approach.

The aforementioned approaches treat the throughput as the only performance factor without addressing the fairness, which nevertheless should be concerned. Consider the first and the second approaches, in which the control process only uses the window size as the indicator of the bandwidth utilization. It roughly estimates the proper window size without any information regarding the buffer occupancy at intermediate nodes. Therefore, it is hard for these two approaches to accommodate the fairness. The third approach uses the round-trip time (RTT) in its control process, thus it provides the capability to improve the fairness.

Consider the relationship between the buffer occupancy and the bandwidth sharing under the common FIFO environment. According to the first-in first-out policy, the ratio of buffer occupancy in the FIFO reflects the ratio of bandwidth sharing. Therefore, if sources keep their buffer occupancies constant, their bandwidth sharing would be consistent even when they experience different RTT's. Therefore, if Vegas keeps its buffer occupancy at a constant level, the fairness among sources with different RTT's can be achieved.

This letter is constructed as follows: Section II describes how Vegas Plus improves the fairness as well as achieves the high throughput. In Section III, we evaluate the performance of the proposed scheme through simulation. Section IV concludes the work.

Manuscript received May 19, 1999. The associate editor coordinating the review of thisl letter and approving it for publication was Prof. N. B. Shroff.

The authors are with the Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan 30056, R.O.C.

Publisher Item Identifier S 1089-7798(00)03850-3.

II. Fairness Improvement on Vegas Plus

TCP Vegas proposed three improvements, early timeout detection, window size adjustment, and safe slow-start. Since early timeout detection and safe slow-start does not affect the bandwidth utilization, the fairness improvement only applies to the window size adjustment in the aspect of bandwidth sharing. The idea is to adjust the window-size by keeping the average queue occupancy at a fixed level, so that the fairness of bandwidth sharing can be achieved.

Traditional congestion avoidance schemes adjust the window size based on the difference between the threshold and the current window, but Vegas adjusts the window based on the difference between the expected and the actual transmission rates. The former is defined as the window size divided by the minimum RTT, while the latter is defined as the window size divided by the measured RTT. Besides setting the target of window size adjustment, traditional schemes increase their window size unrestrictedly, while Vegas not only increases its window size but also ceases window-size increment or even decreases the window size. The design of Vegas provides a quite flexible control as well as selects a proper control target, say, the measured RTT. However, scheme in Vegas is not fair because the expected transmission rate is a relative value, which is unable to represent the fair bandwidth sharing. Therefore, the sooner the source obtains a larger window size, the higher bandwidth utilization it will achieve. On the other hand, if a source has a longer RTT, it is hard to represent the fair bandwidth utilization through window size.

Instead of using the rate difference, the RTT may be more suitable for the control target. Since the acceptable variance in RTT is hard to decide, we convert the delay variable into a so-called virtual queue occupancy (VQO) as the control target. Virtual queue occupancy is derived from

$$N = T\mu \tag{1}$$

where N is the number of packets stored at the queue, μ represents the packet service rate, and T represents the queuing delay of the packet considered. The RTT is available through sending and monitoring the detection packet. The queuing delay of the RTT detection packet is the total sum of queuing delays throughout the end-to-end path. Since VQO is only used to convert the queuing delay to a real controllable variable, the end-to-end path can be treated as a single virtual queue. The bottleneck switch determines the service rate of this queue. If the detection packet experiences queuing delay T and the queue service rate is μ, the queue occupancy can be calculated using (1). The queue service rate can be obtained by measuring the rate of acknowledgment packet. While the queuing delay can be derived from the measured RTT, which is also available with Vegas control mechanism. The measured RTT includes two

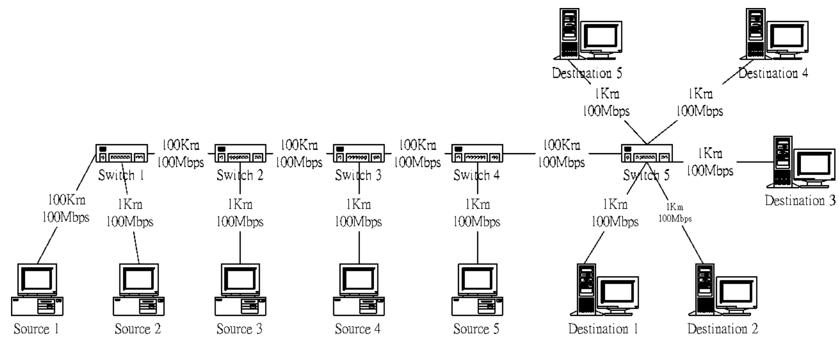

Fig. 1. Network configuration for simulation.

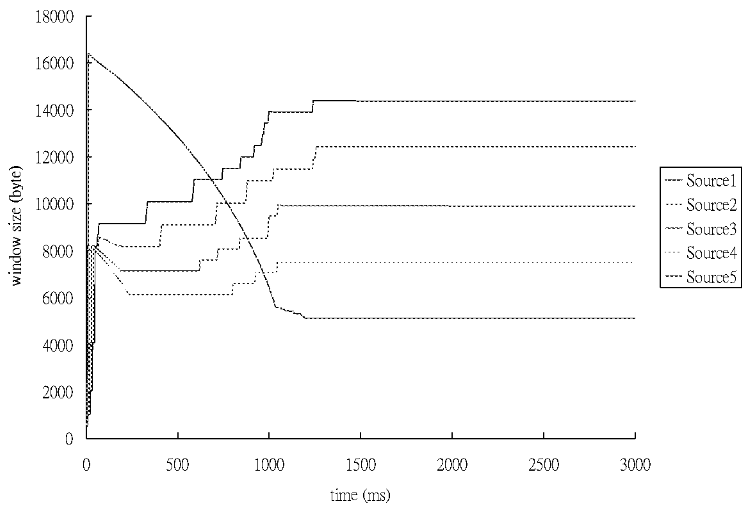

Fig. 2. Window size change in Vegas Plus.

parts, the fixed processing delay and the variable queuing delay. The former can be set to the smallest RTT ever measured, as Vegas does. Then, the queuing delay can be calculated through the measured RTT minus the smallest RTT. According to (1), the VQO is the product of the queue service rate and the queuing delay.

TCP Vegas defines two parameters, α and β, to avoid the window-size fluctuation, the window size will not change when the rate difference falls between these two parameters. Nevertheless, they imply the unfair uses of bandwidth sharing. The queue occupancy setting for α and β in [1] are 2 and 4, respectively, this means that Vegas sources are allowed to keep a queue occupancy from 2 to 4 at an intermediate node, and the variation of bandwidth sharing under the ideal case may be as high as 200%. Since our purpose is to improve the fairness of TCP Vegas, we use only one parameter, α', as the control target. Doing this, the window size may be not as stable as using two parameters, but the fairness can be improved.

Our algorithm for congestion avoidance is described as follows.

1) Define BaseRTT as the smallest RTT ever measured.
2) The Vegas Plus calculates the actual service rate. The measured RTT is defined as the period from sending a segment until its acknowledgment. The amount of data acknowledged during this period is also recorded. The actual service rate is calculated by the amount of data acknowledged dividing the measured RTT.
3) Vegas Plus uses VQO $= (Measured\_RTT - BaseRTT \times Actual\_Service\_Rate$ to calculate the VQO.
4) At first, when the VQO is less than one segment, the window is increased exponentially. On the other situation, Vegas Plus compares VQO to α', and adjusts the window accordingly.
   4.1) If VQO $> \alpha'$, Vegas Plus will decrease the window size linearly to reduce the queue occupancy.
   4.2) When VQO $< \alpha'$, the source needs to allocate more bandwidth to achieve fair bandwidth sharing, and Vegas Plus will increase the window size linearly.
   4.3) It leaves the congestion window unchanged when VQO is equal to α'.

III. Simulation Result and Numerical Results

We evaluate the fairness and throughput of our Vegas Plus through the simulation. A parking-lot configuration is used to examine the performance among sources with different RTT's. Besides, both the throughput and Jain's fairness index [3] are used to compare the fairness before and after the improvement.

Fig. 1 demonstrates the simulation configuration, the simulation period is set to 3 s. The parameter setting for Vegas is same as in [1] and α' in Vegas Plus is set to 3, which is just in the middle of two parameters α and β in Vegas.

The change in window size of Vegas Plus is presented in Fig. 2. It shows that, although the source with the shortest RTT may reach the highest window size initially, but Vegas Plus will force these sources with improper window size to adjust their window to the proper size, thus the window size will converge to the fair sharing value. The performance of these two congestion avoidance schemes is addressed as follows. Vegas sources features the total throughput of 99.7% and the fairness index of 0.543. While the total throughput of Vegas Plus sources is 99.7%, which is as good as Vegas. The advantage of Vegas Plus is in its fairness index, 0.957, which is much higher than Vegas.

IV. Conclusions

In this article, an improvement on TCP based on Vegas congestion control mechanism is proposed. Our mechanism improves the fairness of TCP Vegas among sources with different RTT's without sacrificing its throughput. The simulation is performed to evaluate the fairness in well-known index. From the numerical results, it is obvious that the proposed mechanism features much better performance in the aspect of fairness.

References

[1] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End to end congestion avoidance on a global Internet," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1465–1480, Oct. 1995.

[2] S. Floyd and T. Henderson. (1999, Apr.) The NewReno modification to TCP's fast recovery algorithm. Internet Engineering Task Force. [Online] Internet Draft, draft-ietf-tcpimpl-newreno-00.txt

[3] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Reading, MA: Addison-Wesley, 1991.

[4] M. Mathis, J. Mahdavi, PSC, S. Floyd, LBNL, A. Romanow, and Sun Microsystems, "TCP selective acknowledgment options," Internet Engineering Task Force, RFC 2018, Oct. 1996.

[5] M. Mathis and J. Mahdavi, "Forward acknowledgment: Refining TCP congestion control," *Comput. Commun. Rev.*, vol. 26, no. 4, pp. 281–291, 1996.

[6] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithm," Internet Engineering Task Force, RFC 2001, Jan. 1997.