TCP Vegas-L An Adaptive End-to-End Congestion Control Algorithm over Satellite Communications

Lianqiang Li, Jie Zhu and Ningyu He
Department of Electronic Engineering, Shanghai Jiao Tong University (SJTU), Shanghai 200240, China
Emails: {sjtu_llq, zhujie, NingyuHe_ruby}@sjtu.edu.cn

***Abstract*—Satellite communications is one of the wireless communication technologies, which is widely spread all over the world. Vegas is a kind of the Transmission Control Protocols (TCPs) over satellite communications. However, there are some shortcomings with this protocol, such as being less aggressive and unfair. These would penalize its performance over satellite communications to a certain extent by taking into account the long round trip time and high packet error rates. In this paper, an adaptive end-to-end congestion control algorithm (Vegas-L) is proposed to overcome these issues. Vegas-L can take full advantage of the historical information of network to divide network conditions into more detailed states, then carry out some adaptive and reasonable adjustments according to the specific network state. Simulation results show that Vegas-L is not only able to improve the aggressiveness to gain high throughput but also has the ability to enhance its fairness over satellite networks.**

Keywords–satellite communications; wireless communications; TCP Vegas; aggressiveness; throughput; fairness

I. INTRODUCTION

With the continuous development of satellite communications, their advantages, such as global coverage, bandwidth flexibility and access convenience have been gradually reflected [1]–[3]. However, satellite environments have some intrinsic characteristics including long round trip time (RTT) and high packet error rate (PER) [4], which would degrade the performance of traditional transport layer protocol heavily [5], such as TCP Reno [6]. Some studies have demonstrated that TCP Vegas [7] outperforms TCP Reno with respect to the overall network utilization, throughput and packet loss [8]–[11]. As a result, Consultative Committee for Space Data Systems (CCSDS) has set TCP Vegas as a part of the Space Communications Protocol Standard Transport Protocol (SCPS-TP).

However, there are two main issues with original Vegas over satellite communications. One is that Vegas is too conservative to increase its congestion window size ($Cwnd$) to gain satisfying throughput. The other is unfairness when it competes with Reno. To cope with these shortcomings, several enhanced variants of Vegas have been proposed like Vegas-A [12] and Veno [13]; they increase throughput and fairness by enhancing Vegas's aggressiveness. The strategies employed by them may work in some cases, but they are not general for different PER satellite environments.

In this paper, inspired by the variation of throughput used in Vegas-A, we propose an adaptive end-to-end congestion control algorithm called TCP Vegas-L. Vegas-L will employ more historical information of network to adjust $Cwnd$ and other key variables according to the specific network state. Also, Vegas-L does not add any new parameters and only needs some modifications on the sending ends. Simulation results prove that the proposed Vegas is competitive and even better than Vegas, Vegas-A and Veno over satellite communications under different PER environments.

The rest of this paper is organized as follows. Section II provides a brief description of related works. Section III introduces the proposed Vegas-L. Section IV presents the performance of evaluation under different PER environments over satellite networks. Finally, we summarize the conclusions and present our future research direction in section V.

II. RELATED WORKS

There are some variants of Vegas that have been proposed to solve the above issues over satellite communications. We will review Vegas, Vegas-A and Veno. They would be studied in this paper. Among them, Vegas-A is the referenced version of Vegas-L.

A. Vegas

Vegas is a delay based congestion control algorithm. It is not bias against connections with long $RTTs$, which is crucial to satellite communications. There are three key variables $Cwnd$, α and β. The latter two are the thresholds of extra data to be kept in the network. The pivotal mechanism of it is that Vegas uses measured RTT to calculate the difference ($Diff$) between expected throughput ($Expected\_Th$) and actual throughput ($Actual\_Th$) to estimate the congestion level in the early stage. The core algorithm behind it can be expressed as follows:

$$Expected\_Th = Cwnd/Base \tag{1}$$

$$Actual\_Th = Cwnd/RTT \tag{2}$$

$$Diff = (Expected\_Th - Actual\_Th)Base \tag{3}$$

$$Cwnd = \begin{cases} Cwnd + 1, & \text{if } Diff < \alpha \\ Cwnd, & \text{if } \alpha < Diff < \beta \\ Cwnd - 1, & \text{if } Diff > \beta \end{cases} \tag{4}$$

where $Base$ is the minimum RTT of observation. The thresholds α and β employed by Vegas are fixed, with default values to be 1 and 3.

B. Vegas-A

The main idea of Vegas-A is that rather than fix key variables, they could be adjusted dynamically [12]. Vegas-A uses the difference between present round trip throughput ($Th\_(i)$) and the throughput of last round trip ($Th\_(i-1)$) to adjust its congestion control mechanism more flexibly. The core algorithm of Vegas-A is shown in Figure 1.

```
Input: $Th\_(i)$ and $Th\_(i-1)$
Output: $Cwnd(i+1)$, $\alpha$ and $\beta$
 1: if $\alpha$<$Diff$<$\beta$ then
 2:   if $Th\_(i)$>Th_(i-1) then
 3:     $Cwnd(i+1)$=$Cwnd(i)$+1, $\alpha$=$\alpha$+1, $\beta$=$\beta$+1
 4:   else
 5:     No update of $Cwnd(i+1)$, $\alpha$ and $\beta$
 6:   end if
 7: else if $Diff$<$\alpha$ then
 8:   if $\alpha$>1 then
 9:     if $Th\_(i)$>Th_(i-1) then
10:       $Cwnd(i+1)$=$Cwnd(i)$+1
11:     else
12:       $Cwnd(i+1)$=$Cwnd(i)$-1, $\alpha$=$\alpha$-1, $\beta$=$\beta$-1
13:     end if
14:   else if $\alpha$=1 then
15:     $Cwnd(i+1)$=$Cwnd(i)$+1
16:   end if
17: else if $Diff$>$\beta$ then
18:   if $\alpha$>1 then
19:     $Cwnd(i+1)$=$Cwnd(i)$-1, $\alpha$=$\alpha$-1, $\beta$=$\beta$-1
20:   else
21:     No update of $Cwnd(i+1)$, $\alpha$ and $\beta$
22:   end if
23: end if
```

Figure 1. Vegas-A Congestion Control Algorithm

Where $Cwnd(i)$ is the congestion window size under the current round trip, $Cwnd(i+1)$ is the congestion window size of the next round trip.

C. Veno

Veno is a combination of Vegas and Reno. It uses original Vegas's estimation algorithm to carry out early detection of network congestion. But unlike Vegas, the estimation algorithm is only used for adjusting the increase/decrease coefficient of the Reno congestion control algorithm [14].

Compared with original Vegas, there are two enhancements adopted by Veno. On one hand, the sender will probe network resources very conservatively when the estimation algorithm indicates a congestion state. On the other hand, Veno will have the ability to determine whether the loss of data is due to network congestion or channel errors to a certain extent. The second enhancement is very important for satellite communications as Veno would not ascribe all the losses to congestion under a high PER environment.

III. TCP VEGAS-L

As we were studying the above algorithms, we realized that none of them has enough network status information. In other words, the utilization of historical information of the three algorithms is not sufficient. Taking into account their deficiencies, Vegas-L brings the network probing capability into its congestion avoidance phase by employing the recent network status and the far time network status simultaneously.

First, let us declare a number of variables. In Vegas-L, we use $Th\_(i-2)$ to denote the throughput of two $RTTs$ before. The difference between $Th\_(i)$ and $Th\_(i-1)$ is $Diff\_Now$ while the difference between $Th\_(i-1)$ and $Th\_(i-2)$ is $Diff\_Before$. They are shown as follows:

$$Diff\_Now = Th\_(i) - Th\_(i-1) \tag{5}$$

$$Diff\_Before = Th\_(i-1) - Th\_(i-2) \tag{6}$$

Vegas-L employes the variables to classify network circumstances into 6 cases and carries out some adaptive adjustments. The adjustments are shown in Figure 2.

```
Input: $Diff\_Now$ and $Diff\_Before$
Output: $Cwnd(i+1)$, $\alpha$ and $\beta$
 1: if $Diff\_Now$>0 then
 2:   if $Diff\_Before$>0 then
 3:     if $Diff\_Now$>$Diff\_Before$ then
 4:       Case 1
 5:     else
 6:       Case 2
 7:     end if
 8:   else
 9:     Case 3
10:   end if
11: else
12:   if $Diff\_Before$>0 then
13:     Case 4
14:   else
15:     if $Diff\_Now$>$Diff\_Before$ then
16:       Case 5
17:     else
18:       Case 6
19:     end if
20:   end if
21: end if
```

Figure 2. Vegas-L Congestion Control Algorithm

Case 1: The network has just experienced the growth of throughput for two consecutive $RTTs$. Furthermore, the value of increased in the second RTT is bigger than that in the first one. We can judge that the network is experiencing a rapid growth phase and the remaining bandwidth is sufficient, so we adjust the growth rate of $Cwnd$, α and β as twice as the condition of $Th\_(i) > Th\_(i-1)$ in Vegas-A.

Case 2: Even if the network has just experienced the growth of throughput for two consecutive $RTTs$, the value of increased throughput in the second RTT is smaller than that in the first one. We could infer that the network is experiencing a slow growth phase and the remaining bandwidth is not abundant, so we just follow the strategy of Vegas-A as $Th\_(i) > Th\_(i-1)$.

Case 3: In this case, the throughput has just come from the reduction phase to the growth phase. There is a slight improvement in the network condition, but the remaining bandwidth would not be too much. So we set the growth rate of $Cwnd$, α and β more moderately. As a result, they are half of Vegas-A's strategy as $Th\_(i) > Th\_(i-1)$.

Case 4: In this case, the throughput has just come from the growth phase to the reduction phase. There is a slight deterioration in the network condition, but the degree of congestion level would not be very serious. So we set the reduction rate of $Cwnd$, α and β to be half of Vegas-A's strategy as $Th\_(i) < Th\_(i-1)$.

Case 5: In spite of network has just experienced the reduction of throughput for two consecutive $RTTs$, the throughput reduced in the second RTT is smaller than that in the first one. It indicates that network begins to relieve its congestion level. We just keep the strategy of Vegas-A as $Th\_(i) < Th\_(i-1)$.

Case 6: The network has just experienced the reduction of throughput for two consecutive $RTTs$. Moreover, the throughput reduced in the second RTT is bigger than that in the first one. We can assume that the network is experiencing a very congested phase. As a result, the reduction rate of $Cwnd$, α and β are set to be as twice as Vegas-A's strategy when $Th\_(i) < Th\_(i-1)$.

These more particular network states and adaptive adjustments will enable Vegas-L to gain better performance over satellite communications.

IV. Performance Evaluation

We use Network Simulator 2 (ns2.35) [15] to validate the effectiveness of Vegas-L. The simulation settings and simulation results are shown as follows.

A. Simulation Settings

The employed satellite network topology is expressed in Figure 3. The first ground launching node is placed in Beijing (39.4°N,116.4°E), corresponding receiver is in New York (40.7°N,74°W). The first data flow is attached with original Vegas, Vegas-A, Veno and Vegas-L respectively. The second ground launching node is collocated at Shanghai (31.2°N,121.5°E) and receiver is collocated at Washington (28.9°N,77°E). The second data flow is attached with the enhanced Reno, NewReno [16].

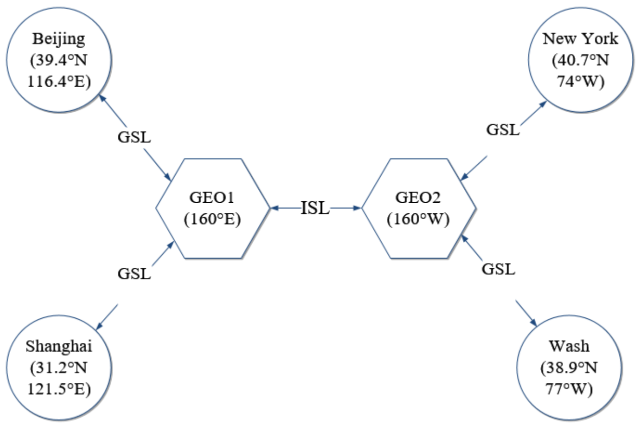
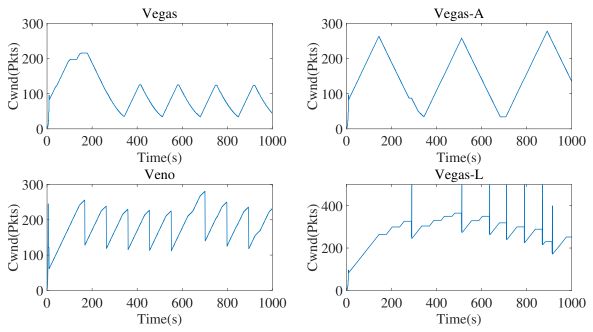


Figure 3. The satellite network topology

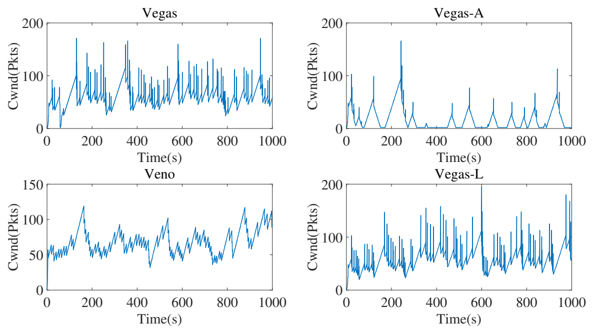
The bandwidth of links are set as $5Mbps$. The link type is LL/Sat while the queue management type of node buffer is DropTail. TCP packet size is 1024 bytes. Simulation lasts for 1000 seconds to get steady results. We evaluate the performance by considering two circumstances. In the first one, PER is low, just 10^{-6}, and the PER is as high as 10^{-3} in the second one.

B. Simulation Results

The dynamics of $Cwnd$ is the basic metric for evaluating a congestion control algorithm. It could reflect real-time network state during the simulation. The evolution of $Cwnds$ of Vegas, Vegas-A, Veno and Vegas-L under different PER environments over the satellite network is shown in Figure 4.



(a) The dynamics of Cwnd under low PER environment



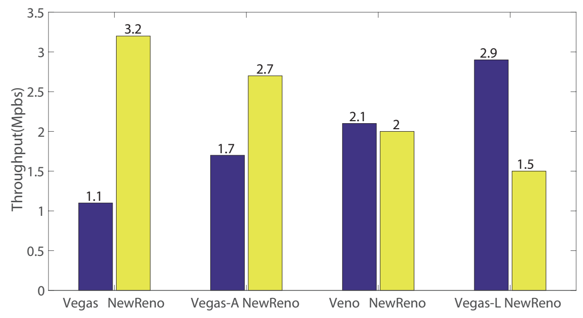
(b) The dynamics of Cwnd under high PER environment

Figure 4. The dynamics of Cwnds under different PERs environments

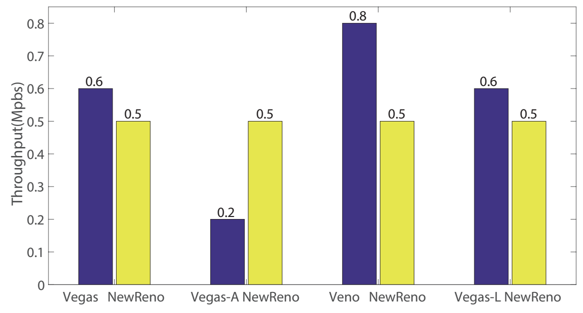
By observing Figure 4(a), we can see that the performance of $Cwnd$ of Vegas-L under the low PER environment is the best one. More concretely, the $Cwnds$ of these algorithms are similar to each other in the first 150 seconds. Then there are some fluctuations in original Vegas, Vegas-A and Veno. On the contrary, Vegas-L begins to enter a steady growth phase. There is little fluctuation in its $Cwnd$. Furthermore, the steady value of Vegas-L is 350 packets, which is not only 200% larger than that in original Vegas but also larger than those in Vegas-A and Veno. Next, we can also see from Figure 4(b) that the performance of all four algorithms has declined under the high PER environment. There are more dense fluctuations. The values of $Cwnd$ are much smaller than those in low PER environment. However, from the whole point of view, the $Cwnd$ value of Vegas-L is still similar to Vegas and Veno. Their performance is tied for the best one, which is better than Vegas-A.

The performance of $Cwnd$ is related to strategies used by the protocols. As far as Vegas and Vegas-A are concerned, their congestion control algorithms are relatively conservative. As a result, their $Cwnds$ change gradually in low PER environment. Veno is the combination of Vegas and Reno. It also has the intrinsic of loss-based algorithms to gain network resources aggressively. Vegas-L is not as sensitive and dull as Vegas, it could be more active to adjust its $Cwnd$ according to specific network state. Consequently, Veno and Vegas-L will have some sharp fluctuations in $Cwnds$ when the PER is low. The performance of $Cwnd$ which degrades with the increases of PER over satellite network is unavoidable. But Vegas-L is always gaining the best performance either under low PER environment or under high PER environment. The good performance of Vegas-L is due to that Vegas-L not only always monitors the satellite network status but also improves its $Cwnd$ and other key variables adaptively. Vegas-L inherits the stability from Vegas. As contrast, the performance of Vegas-A is the worst one in the high PER environment. Although Vegas-A also inherits the stability from Vegas, the congestion control algorithm employed by Vegas-A is too fragile to ensure a good performance in a high PER environment over satellite communications.

Network throughput is another basic metric to show the effectiveness of a congestion control algorithm. The final average throughput of the four algorithms under different PER environments over the satellite network is shown in Figure 5.



(a) The average throughput under low PER environment



(b) The average throughput under high PER environment

Figure 5. The average throughput under different PER environments

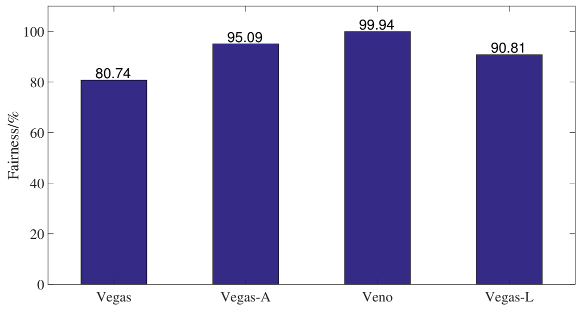
As illustrated in Figure 5(a), the throughput of Vegas is $1.1Mbps$ under low PER environment over the satellite network. The throughput is improved in Vegas-A and Veno, but the highest one is Vegas-L with $2.9Mbps$. We could also find that the throughput of NewReno varies widely under low PER environment. It reaches $3.2Mbps$ when competes with original Vegas while only gains $1.5Mbps$ when competes with Vegas-L. As shown in Figure 5(b), all of these algorithms' performance becomes poor with the increases of PER. Among them, Vegas-A decreases rapidly, its throughput is the smallest one. Conversely, Veno is the largest one, followed by original Vegas and Vegas-L. The throughput of NewReno is always $0.5Mbps$ when it competes with variants of Vegas under high PER environment.

The performance of throughput is corresponding to the values of their $Cwnds$. Vegas-L has more historical information, detailed network states and dynamical strategies. It would not revise its $Cwnd$ recklessly. In consequence, Vegas-L has a better $Cwnd$, which leads to better throughput under different PERs over the satellite network. These phenomena also demonstrate that Vegas-L has much better aggressiveness.

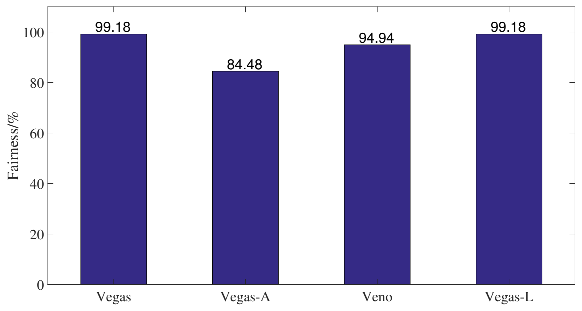
The fairness index is closely related to throughput. In this paper, we use Jain's fairness index [17], which is defined as Eq.7:

$$f(x_1, x_2, \ldots, x_n) = (\sum_{i=1}^{n} x_i)^2 / (n \sum_{i=1}^{n} x_i^2) \tag{7}$$

where n represents the number of data flows and x_i is the throughput of the ith data flow. The results are shown in Figure 6.



(a) The fairness under low PER environment



(b) The fairness under high PER environment

Figure 6. The fairness under different PER environments

It is clear from Figure 6(a) that the fairness index of original Vegas is just 80.74%, which is the smallest one. The fairness performance is enhanced in Vegas-A, Veno and Vegas-L. Particularly, the fairness index of Veno is as high as 99.94%. As shown in Figure 6(b), the fairness performance of original Vegas and Vegas-L are tied for the best one under high PER environment over the satellite network, followed by Veno. Compared with them, Vegas-A owns the worst performance.

We should point out that the throughput of Vegas-L is larger than the throughput of corresponding NewReno under low PER circumstance. Vegas-L not only has the ability to share bandwidth equally with NewReno but also can occupy a leading role, which results in the ordinary fairness performance. Similarly, Veno has the ability to distinguish whether packet loss is due to network congestion or channel errors.

V. Conclusions

In this paper, we have proposed an adaptive end-to-end congestion control algorithm called TCP Vegas-L for satellite communications. It can take full advantage of more historical information to adjust its congestion control strategies dynamically. The modifications introduced by Vegas-L are related to specific network state. After a comprehensive comparison of original Vegas, Vegas-A, Veno and Vegas-L, we can draw some conclusions as follows:

1) Vegas-L could gain outstanding performance in terms of $Cwnd$, throughput and fairness under low PER environment over satellite communications.
2) Vegas-L is able to gain satisfying performance with respect to $Cwnd$, throughput and fairness under high PER environment over satellite communications.
3) Vegas-L is competitive and even better than Vegas, Vegas-A and Veno over satellite communications under different PER environments.

However, there is still some room to improve the performance of Vegas-L. On the one hand, adjusting the aggressiveness and fairness of Vegas according to particular state of network is a novel method, but how to optimize its strategies is still worth studying. On the other hand, we could see that the performance of Vegas-L under high PER satellite environment is not the best one; how to modify Vegas-L to make it more suitable for high PER satellite environments would be another challenge.

Acknowledgement

This work is supported by Shanghai Aerospace Science and Technology Innovation Foundation of China (grant No. SAST2015039) and the National Natural Science Foundation of China (grant Nos.11433002 and 61371147).

References

[1] L. Li, H. You, J. Zhu, and Y. Yang, "A novel design on multi-layer satellite constellation," Journal of Shanghai Normal University, vol. 45, no. 2, 2016, pp. 248–252.

[2] L. Li, J. Zhu, Y. Yang, and Z. Hu, "Evaluation of tcp congestion control algorithms on satellite ip networks," Journal of Aerospace Shanghai, vol. 33, no. 6, 2016, pp. 109–114.

[3] Q. Zhao, H. Zhou, and H. Deng, "An enhanced vegas congestion control algorithm based on beidou navigation system," IEICE Communications Express, vol. 1, no. 7, 2012, pp. 275–281.

[4] L. Yang, D. Wei, C. Pan, and K. Wang, "Congestion control algorithm based on dual model control over satellite network," in Wireless Communications & Signal Processing (WCSP), 2015 International Conference on. IEEE, 2015, pp. 1–6.

[5] H. Obata, K. Tamehiro, and K. Ishida, "Experimental evaluation of tcp-star for satellite internet over winds," in 2011 Tenth International Symposium on Autonomous Decentralized Systems. IEEE, 2011, pp. 605–610.

[6] V. Jacobson, "Modified tcp congestion avoidance algorithm," end2end-interest mailing list, 1990.

[7] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "Tcp vegas: New techniques for congestion detection and avoidance," vol. 24, no. 4, 1994, pp. 24–35.

[8] S. A. Nor, A. N. Maulana, F. A. A. Nifa, M. N. M. Nawi, and A. Hussain, "Performance of tcp variants over lte network," in AIP Conference Proceedings, vol. 1761, no. 1. AIP Publishing, 2016, pp. 020–021.

[9] E. Abolfazli and V. Shah-Mansouri, "Dynamic adjustment of queue levels in tcp vegas-based networks," Electronics Letters, vol. 52, no. 5, 2016, pp. 361–363.

[10] J. Qu, "An enhanced tcp vegas algorithm based on route surveillance and bandwidth estimation over geo satellite networks," in 2010 International Conference on Measuring Technology and Mechatronics Automation, vol. 1. IEEE, 2010, pp. 464–467.

[11] M. Nirmala and R. V. Pujeri, "Evaluation of tcp congestion control algorithms on different satellite constellations," in Advanced Computing and Communication Systems (ICACCS), 2013 International Conference on. IEEE, 2013, pp. 1–7.

[12] K. Srijith, L. Jacob, and A. L. Ananda, "Tcp vegas-a: Improving the performance of tcp vegas," Computer communications, vol. 28, no. 4, 2005, pp. 429–440.

[13] C. P. Fu and S. C. Liew, "Tcp veno: Tcp enhancement for transmission over wireless access networks," IEEE Journal on selected areas in communications, vol. 21, no. 2, 2003, pp. 216–228.

[14] A. Afanasyev, N. Tilley, P. Reiher, and L. Kleinrock, "Host-to-host congestion control for tcp," IEEE Communications surveys & tutorials, vol. 12, no. 3, 2010, pp. 304–342.

[15] "The network simulator 2.35," http://www.http://www.isi.edu/nsnam/.

[16] S. Floyd, T. Henderson, and A. Gurtov, "The newreno modification to tcp's fast recovery algorithm," 2004.

[17] R. Jain, A. Durresi, and G. Babic, "Throughput fairness index: An explanation," Tech. rep., Department of CIS, The Ohio State University, Tech. Rep., 1999.