International Journal of Innovative
Computing, Information and Control
Volume **7**, Number **6**, June **2011**

ICIC International ©2011 ISSN 1349-4198
pp. **3059–3071**

AN INTELLIGENT TCP VEGAS TO SUPPORT LONG-DISTANCE HIGH SPEED NETWORKS

Tien-Szu Pan, Mong-Fong Horng*, Chien-Chou Shih and Bei-Hao Su

Department of Electronic Engineering
National Kaohsiung University of Applied Sciences
Kaohsiung, Taiwan
*Corresponding author: mfhorng@ieee.org

Received May 2010; revised September 2010

Abstract. *In this paper, an Intelligent TCP Vegas (iTCP-Vegas) is proposed to improve the throughput of long-distance high-speed TCP connections. Large Round Tip Time (RTT) of packets in 1ong-distance networks is a problem to cause the performance degradation of original TCP connections. Although the RTT-based TCP has been recognized to benefit the response performance of TCP connections, the previous conservative schemes do not offers a feasible and reliably approach. For efficient response, the alternative approach to adapt the congestion window is proposed based on a four-stage adjustment, rather than the conventional three-stage approach. Additionally, the original TCP Vegas with constant-interval is improved by the usage of dynamic intervals. In the proposed iTCP-Vegas, the main advantages includes (1) dynamic adjustment of the parameters, α and β, to control the congestion window size values to upgrade the throughput of TCP connections on long-distance networks; (2) Simulation results show that in typical conditions of bandwidth and distance, the proposed mechanism effectively improves the throughput of standard TCP Vegas. Besides, the analysis of fairness among homogenous and heterogeneous TCP connections is also presented.*
Keywords: Congestion control, High-speed transmission, Long-distance networks, TCP Vegas, Quality of services

1. **Introduction.** The exponential growth of Internet population and facility greatly contribute globalization of information. There are so many kinds of Internet services including is the convergence of Internet services in past decades. On the Internet of globalization, more and more service contents are delivery across continents with large delivery delay, typically more than 5000 KM with a delay of 150 ms. Thus, Long transmission Distance and High-speed transmission Network (LDHN) is a trend of Internet in near future. For the decades of rapid and rich development, Internet services demonstrate the diversity of content. In the early stage of Internet, text-style services are popular for archive, file access, hypertext and email. As the development of networking technology, people can enjoy more and more access technology, such as wireless access, optical network and mobile access. In addition, more and more available bandwidth and users terminals are conducted to deliver and present more interesting contents. For an example, Youtube is one of the emerging content providers on Internet. The revenue of Youtube in 2009 is close to USD 1 billion. Users upload their media clips to Youtube through wireless access to share with their friends in worldwide. YouTube has to provide a reliable streaming service for its users in countries. How to make worldwide users access the content smoothly, sufficient bandwidth and long-distance flow control are necessary to promise a trustable service quality.

Another example is VoIP. VoIP, Voice over IP networks, is to emulate the phone service on Internet. According to the report from In-Stat the population of VoIP services has been 50 million in worldwide. In traditional VoIP connections, voice packets are delivered by UDP protocol operating on IP protocol. However, the features of UDP, including no flow control, no congestion control and no altering of packet loss, degrade the quality of service (QoS) in VoIP applications. Although TCP Reno offers the mechanisms of acknowledgement and sliding windows to ensure the transmission reliability and to control end-to-end flow rates, the two well-developed do not fit the needs of VoIP application. First, to guarantee the timeliness of voice packets, retransmissions of lost packets are not necessary, even slight loss is tolerable. Second, because the sliding window size is updated upon the previous packet is acknowledged, the updating of sliding windows of long-distance voice connections is too slow to response to varying network dynamics including congestion and channel error. Third, the continuing development of physical layer technology contributes a boost of network hardware performance, however, the software of layer 2 and 3 protocol oppress the hardware contribution. However, for VoIP applications, the slow start feature of TCP Reno causes the insufficient throughput at initial stage of VoIP connections, even when the available bandwidth is sufficient. Thus, the voice quality is corrupted at initial and this corruption becomes more serious when round trip time, RTT, of connections become larger. As a result, the LDHN users suffer from the poor quality of Internet services due to the large transportation delay.

Thus, for LDHN users, quality of services (QoS) technology is an essential to support satisfactory services. There are many approaches to enhance the Internet QoS. Ali et al. [1] presented a stability analysis to clarify the issues of congestion control on nonlinear links. Lee et al. [2] proposed a contention-aware QoS routing algorithm to improve the network quality on wireless networks. Chen et al. [3] presents an adaptive end-to-end delay equalization for TCP (Transport Control Protocol) virtual transmission in Internet Environment. Obviously in such a situation, the current TCP (Transport Control Protocol) congestion control mechanism can not fully utilize the physical bandwidth of optical networks. Even, the energy efficiency has already become one of QoS issues in networking [4]. The current approach to control congestion is TCP New Reno [5] operating in a manner of Additive Increase Multiplicative Decrease (AIMD) according to packet loss. AIMD can quickly speed up connection throughput upon the reception of packet acknowledgements from receivers. When a possible congestion is detected by the timeout of packet acknowledgement, the connection throughput is multiplicative decreased to release the congestion. AIMD had worked from decades to solve congestion and control flow of end-to-end connections well. However, the AIMD approach can not ensure a smooth throughput. Because of the natures of AIMD approach, the throughput will be increased and decreased in cases of successful packet reception and loss, respectively. Thus the throughput is not stable enough for certain Internet services. In addition, large RTT of packets delivering on long-distance network will degrade the agileness of congestion control. A. Brakmo et al. [6] proposed to estimate the network bandwidth condition by RTT measurement. Depending on RTT, the packet transmission rate can be adjusted by using congestion control mechanism. This mechanism is called TCP Vegas. TCP Vegas decides to increase or decrease the cwnd value according to comparison of difference (*diff*) of expected throughput and actual throughput. There are many researches [7-10] revealing that TCP Vegas has better bandwidth usage efficiency than TCP New Reno. Therefore, in this research, we propose an iTCP-Vegas based on the mechanism of TCP Vegas and then the network bandwidth usage can be improved efficiently. In this we focus on how to enhance the management scheme of congestion window of TCP Vegas and improve the throughput. We adopt dynamic adjustment of α and β values to take the place of fixed α

and β values in original TCP Vegas. In this work, we would like to explore the congestion control schemes of TCP Vegas and propose a better solution to meet the goals of (1) improvement of the throughput of connections on long-distance high-speed networks; (2) higher network bandwidth efficiency in the long-distance high-speed networks. Thus, we design a congestion control scheme based on TCP Vegas to adapt intelligently the flow for various traffic types. The proposed new approach will be more suitable for the services in the future Internet.

The rest of this paper is organized as follows. Related work is explored in Section 2 to point the issues investigated in this work. An intelligent TCP Vegas to improve the throughput of TCP connections on LDHNs is presented and analysis in Section 3. In Section 4, a series of simulations in various conditions based on NS2 is presented. Finally, we conclude this work in Section 5.

2. **Related Work.** Protocols of transport layer are the key to create connections across Internet. The most popular transport-layer protocol is TCP-family. Internet usually uses TCP which is a traditional transmission control protocol to transmit information. TCP provides (1) reliable delivery, (2) connection-oriented service and (3) flow control. Flow control is accomplished by controlling the congestion window (cwnd) to adjust the transmission rate. Congestion window is a sliding window to control the number of sent packets that have not been acknowledged by the receivers. The size of congestion window indicates the maximum number of packets could be on the fly. The more the *cwnd* is, the higher the throughput is. In the typical TCP such as TCP Reno, *cwnd* will be increased when a successful packet delivery is acknowledged to raise the throughput. However, with no proper management, an unlimited *cwnd* definitely causes packet loss because the resulted flow rate exceeds the available bandwidth. Reno TCP is the most popular WAN transmission protocol to offer the flow control service. There are three different phases in the flow control mechanism of Reno TCP, including, (1) slow start (SS) phase, (2) congestion avoidance (CA) phase and (3) fast retransmission (FR) phase. In SS phase, the *cwnd* grows at exponential rate to help the connection gather more bandwidth. Once a cumulative ACK packet is received, the *cwnd* will increase by one. This action will repeat constantly until the *cwnd* reaches a slow start threshold, *threshold*, and the connection of Reno TCP will enter CA phase.

In CA phase, the connection is regarded as its approaching to the desired throughput. The desired throughput is specified by either the most available bandwidth or the upper limit of bandwidth caused by applications. The *cwnd* value is updated according to Equation (1) when the sender receives a cumulative ACK packet. However, if there are three duplicate ACK packets received, no matter in SS or CA phase, Reno TCP will enter FR phase. In FR phase, Reno TCP updates *threshold* first and the *cwnd* is also reset to *threshold*. Then, Reno TCP retransmits the lost packet and enters CA phase again. However, in the other situation, if a packet loss is encountered, *cwnd* would also be updated according to Equation (2), and the cwnd would be reset to 1. Reno TCP enters SS phase again.

$$cwnd(t+1) = cwnd(t) + \frac{a((cwnd(t))}{cwnd(t)} \tag{1}$$

$$cwnd(t+1) = cwnd(t) \times (1 - b(cwnd(t))) \tag{2}$$

where

$t \in [0, \infty)$ is the iteration number of *cwnd* updating,

$a(cwnd(t))$ is the increase function and always equals to 1 in Reno TCP,

$b(cwnd(t))$ is the decrease function and always equals to 1/2 in Reno TCP.

Both Equations (1) and (2) are called as response functions to govern the *cwnd* with respective to the occurrence of congestion along the link paths.

However, the control scheme can not meet the requirements of new Internet services. First, to improve TCP, lots of researchers propose new approaches to adapt TCP parameters. These proposed approached are categorized according to the criteria of adapting the congestion window as (1) packet loss-based measurement, (2) RTT-based measurement. For examples, HighSpeed TCP [11], FAST TCP [12], Scalable TCP [13] and H-TCP [14]. In order to improve the network bandwidth efficiently, the Enhanced Vegas [15] was presented to improve TCP Vegas throughput based on some alternative measurements. Ada Vegas is another example. Ada Vegas [16] is similar to TCP Vegas in using the difference between expected throughput and actual throughput to adjust *cwnd* size. In Ada Vegas mechanism, a variable *inc* is added to associate with the legacy variables, α and β, of original TCP Vegas to dynamically control congestion window. Through the variable *inc*, the modification step of *cwnd* is magnified by *inc* $\times$ MSS to prompt the throughput. Quick Vegas is another approach based on dynamic *cwnd*. Quick Vegas increases or reduces the *cwnd* size according to *diff* value based on current throughput. It really improves the problem of lack *cwnd* growth rate in TCP Vegas. However, the growth rate of *cwnd* size of Quick Vegas is still conservative. It is insufficient for applying in long-distance high-speed networks.

From the analysis above, we conclude that (1) adaptive sliding window control is the key issue of the next generation Internet; (2) the control approaches based on packet losses have not been suitable for long-distance connections due to its oscillation characteristics and conservative throughput (3) RTT- based approaches have the potential to offer connections with stable and high throughput. Although there are still some weakness of slow response and rough throughput guarantee, RTT-based approaches have been recognized as the controllability based on certain measurement and estimation. In this work, we utilize the RTT measurement to evaluate and adjust congestion window of connections using TCP Vegas. According to the analyses of above TCP Vegas, we find that there are two methods to improve the throughput of TCP Vegas in long-distance high-speed networks: (1) raise the growth rate of *cwnd* size, (2) adjust α and β to increase TCP Vegas performance efficiently. Therefore, we propose a congestion avoidance mechanism based on methods (1) and (2) in this research.

3. An Intelligent TCP Vegas to Support Long-distance High Speed Networks.

In general, the throughput of single link is derived from Equation (3). We find that the throughput will be low if (1) higher RTT of packets in a LDHN connection, (2) the small *cwnd* due to the conservative management of *cwnd*. Unfortunately, in LDHNs, not only the RTT value is very high, but the *cwnd* is low. In this research, the proposed intelligent TCP Vegas is based on TCP Vegas. Depending on the measured RTT value and comparing with *diff* value, and according to which area the *diff* value laid on. Thus, the growth rate of *cwnd* size will be revised and the values of α and β will be adjusted dynamically. Using this method, the situation of falling into fixed *cwnd* size in TCP Vegas will be improved and the throughput will also be raised.

$$Throughput = \frac{cwnd \times packet\_size \times 8}{RTT} \tag{3}$$

The original TCP Vegas has three operational phases: slow-start, congestion avoidance (CA) and fast retransmission. In this research, the phase of congestion avoidance will be focused, the other two phases will not be considered. The pseudo-code of TCP Vegas CA phase is shown in Figure 1.

```
/* CA phase */
IF diff < α
      cwnd(k + 1) = cwnd(k) + 1/cwnd(k);
else if α < diff < β
      cwnd(k + 1) = cwnd(k);
else if β < diff
      cwnd(k + 1) = cwnd(k)??1/cwnd(k);
End if
```

FIGURE 1. Pseudo code of TCP Vegas

For each connection of TCP Vegas, the desired throughput is given when a connection is initiated. Upon a reception of packet acknowledgement, first the actual throughput is derived from the RTT and packet size of the previous packet. Then, the difference denoted as *diff*, between the actual throughput and the desired throughput is obtained from Equation (3) as given by

$$\begin{aligned} diff(k) &= (Expected - Actual)^* BaseRTT \\ &= \left(\frac{cwnd(k) \times packet\_size}{BaseRTT} - \frac{cwnd(k) \times packet\_size}{RTT(k)} \right) \times BaseRTT \\ &= cwnd(k) \times packet\_size \times \left(1 - \frac{BaseRTT}{RTT(k)} \right) \end{aligned} \tag{4}$$

where $BaseRTT$ and $RTT(k)$ is the minimum RTT and the RTT measured at time k. Typically $BaseRTT$ is the RTT of packets delivered in the Expected rate. In the initial stage of connection establishment, if the measured RTT is close to $BaseRTT$, the condition of Area 1 is held to fast increase the *cwnd* and throughput. The measured RTT is varied with the congestion of links. The more links the congestion occurs, the more the RTT will be. There are two important operational parameters named as α and β to determine the adjustment of *cwnd*. As shown in Figure 1, there are three conditions for any connection of TCP Vegas. When the condition of $diff < \alpha$ is held, the throughput is low. The throughput is updated as the original TCP Vegas. When *diff* is between α and β, the throughput is moderate and is kept steady. The throughput will be reduced if the *diff* is far larger than β. According to the pseudo-code above, obviously TCP Vegas derives the new *cwnd* value from the relationship among *diff*, α and β. However, this approach is not agile enough for the connections with large delay, for an example of LDHNs. The main reason is that with large RTT, the growth of *cwnd* value can not be update in time. Therefore, we propose an iTCP-Vegas to improve TCP Vegas performance. First of all, we propose a new decision layout segmented by the point set of (α, $(\alpha + \beta)/2$, β), rather than the original TCP Vegas with (α, β) as shown in Figure 2.

In Figure 2, the main reason of four areas is expected to further distinguish the current situation of network congestion. Through four areas of iTCP-Vegas, the current network situation can be adjusted efficiently. We define these four areas as: the Area 1 indicates that the network bandwidth is highly available. In Area 1 the *cwnd* size will be updated and increased rapidly. In the situation of Area 2, it means that network bandwidth will be exhausted nearly. Thus the increment of *cwnd* will be slowed down. A stable connection is obtained when the condition of Area 3 is held. In such a condition, *cwnd* is also kept stable. When the condition of Area 4 is held, network condition is going into be in congestive region. As a result, *cwnd* size will be reduced to avoid the network congestion and packet loss. The four areas of iTCP-Vegas are the operational model proposed to

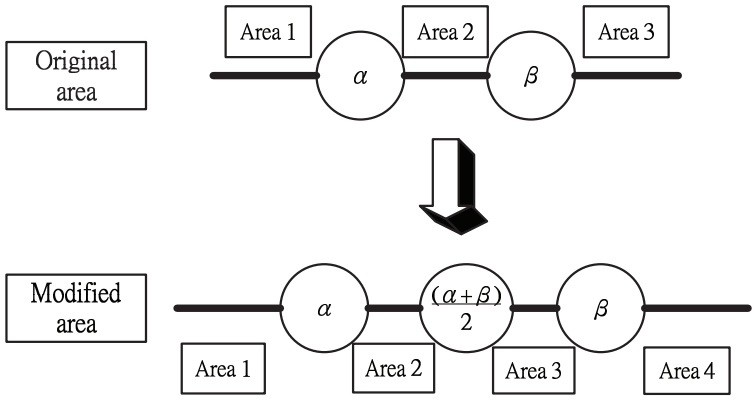

FIGURE 2. A new decision layout segmented by the point set of (α, $(\alpha + \beta)/2$, β)

ensure the avoidance of connection congestion. The flow diagram is shown in Figure 3. When the *diff* value is equal to the partition point of α, β or $(\alpha + \beta)/2$, the *cwnd* size and α and β values will not be revised; only P value will be set to zero.

In the past approaches, packet loss is the main index to detect the link congestion and to decide how to slow down the packet rate at senders. However, how to prevent packet losses and slow down the packet in prior to link congestion are two typical issues discussed in the previous works. Instead, round trip time of delivered packets is suggested to evaluate the link congestion. However, how to derive the congestion information from the RTT measurement and to adjust the packet rates accordingly have become significant issues from 2003 [19-21,23]. Due to the high dynamic of network traffic, the congestion window adjustment according to RTT measurement is a challenging work. In comparison to the previous work, including Reno and Vegas, the proposed approach features some advantages in flow control of long distant Internet connections, such as The proposed iTCP-Vegas features (1) dynamic adjustment of *cwnd* in various connection conditions; (2) dynamics adjustments of operational parameters, α and β; (3) dynamic scaling of the increment and decrease of *cwnd* to speed the performance response. In future work, the proposed iTCP-Vegas will be implemented and integrated into Linux-based kernel as well as deficiencies of proposed iTCP-Vegas will also be reformed at that time. There are four main parts in this flow chart, labeled in A, B, C, D and E. In part A, the *diff* is used as the measurement to choose one from four strategies to decide how to update the *cwnd* and the operational parameter of α and β. In part B, the throughput is exhaustedly prompted by increasing the scaling variable P, and the operational parameters. When the throughput is valid for the criteria of Area 2, the flow of Part C will slow down the increment of *cwnd* and reduce the gap between α and β to make the throughput more stable. In Part D, the throughput is not changed but the gap between α and β will adapt closely. The flow depicted in Part E is to rapidly reduce, even reset, the throughput as a possible congestion is detected. Regarding to the computation issue, as shown in Figure 3, the proposed iTCP-Vegas approach is composed of three stages; including RTT measurement, congestion window (*cwnd*) adjustment and parameter (α and β) adjustment. The flows of the three stages are composed of conditional instructions such as if-command, rather than for-loop. Thus, the computation complexity is constant, as denoted as O(1). Besides, due to the stateful property of TCP schemes, the proposed iTCP-Vegas has to maintain the extra parameter P used in dynamic adjustment of congestion windows. This slight overhead should be a neglectful effect to real implementation.

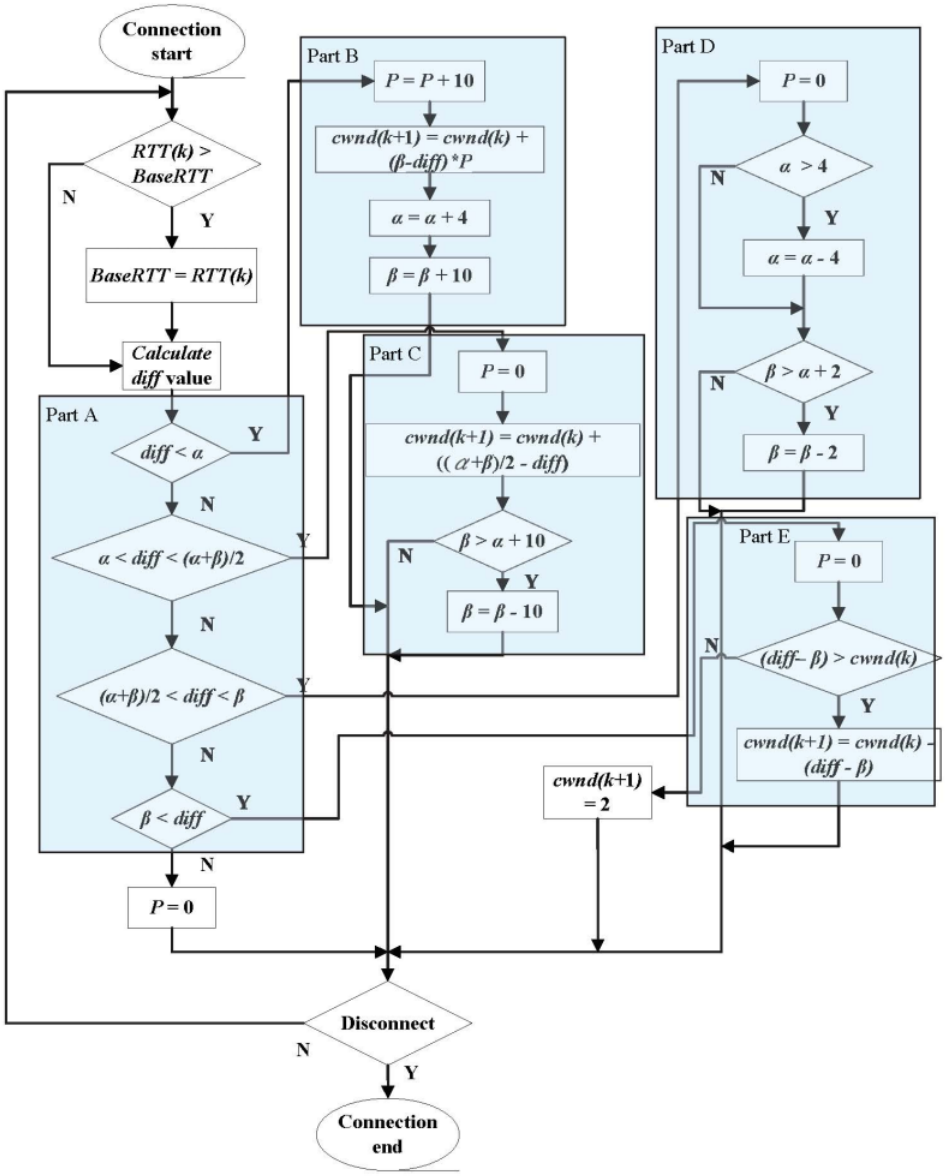

FIGURE 3. Flow chart of the proposed iTCP-Vegas

4. **Performance Evaluation and Analysis.**

4.1. **Test scenarios.** To verify the performance of the proposed iTCP-Vegas, there are two stages composed of software simulation and true implementation deployment. So far, we have finished the software simulation. We plan to release a Linux kernel with the proposed iTCP-Vegas module in 2011 for test. In software simulation, an iTCP-Vegas module has been developed for performance evaluation. NS2 [22] is the most popular and well-recognized simulation tools for network engineering. The test scenarios presented in this paper include (1) single connection throughput, (2) throughput comparison of connections with various TCPs, (3) throughput comparison of connection with various link delays. Typical network topology [21] used in the simulations is shown in Figure 4. In the topology, there is a backbone connecting two remote local area networks. This backbone is a modelled with various bandwidths and delays to stand for the real conditions. In the LANs, the typical delay between hosts and LAN router is less than 10 ms, depending on traffic dynamics. And the link bandwidth is specified as a Gigabit

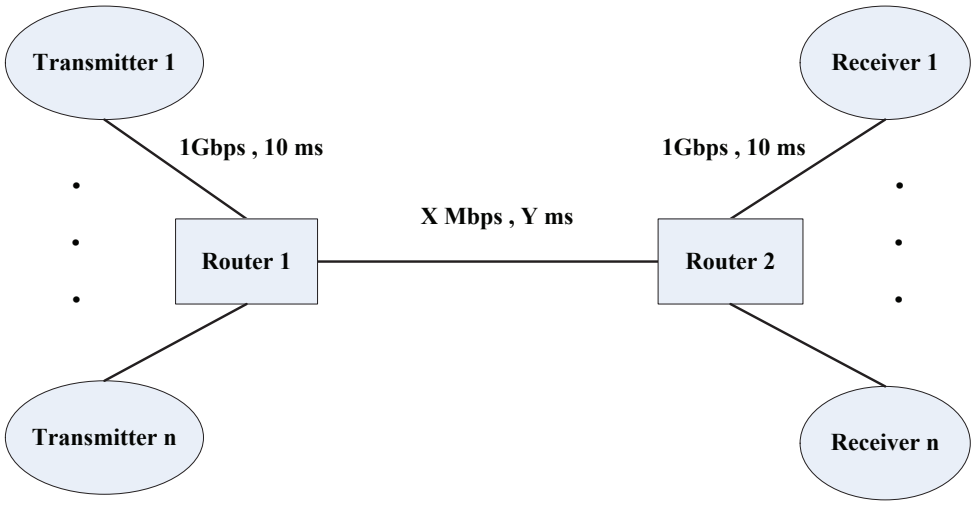

FIGURE 4. Network topology

Ethernet (IEEE 802.3u) due to its popularity in real deployments. The transmission delay on the link between edge router1 and 2 is denoted as Y ms. The bandwidth of long-haul connection is denoted as X Mbps. In the following simulations, the X and Y are used as controlled variables to explore the effects of RTT and link bandwidth on connection throughput. There are three parts of performance evaluation to illustrate the effectiveness of the results. The three parts are designed to evaluate the throughput performance of the proposed iTCP in comparison with the Vegas and its variations; (2) evaluate the effect of link bandwidths such as 155/622 Mbps ATM networks due to its popularity of worldwide deployment. (3) evaluate the effect of link delays caused by long-distant links.

4.2. **Evaluation and analysis.** The first scenario is to specify a single connection of FTP service from Transmitter 1 to Receiver 1. FTP traffic is an elastic traffic because TCP Vegas will adapt the flow rate according to the available bandwidth to avoid packet loss. In this scenario, the backbone bandwidth is supposed to 622 Mbps complied with a long-distance SONET link and the link delay is 110 ms to emulate a distance approximating to 10000 Km. The scenario will verify how fast the proposed approach can boost up the throughput when no other traffic exists. The curves of resulted *cwnd* and RTT, with respect to time, are the key values indicating the performance. A RTT comparison of the proposed iTCP-Vegas and the original TCP Vegas is shown in Figures 5 and 6. The proposed iTCP-Vegas drives the throughput to 250 Mbps in 55 seconds and keeps the throughput steady. In contrast, the *cwnd* of the connection using the original TCP-Vegas grows very slowly.

The RTT variation of TCP Vegas and iTCP-Vegas is shown in Figure 6. Both TCP Vegas and iTCP-Vegas switch the phase from slow-start phase to congestion avoidance phase as $t = 4$ seconds. Afterward, due to the large RTT of 260 ms, the connections of TCP Vegas suffer from slow growth. The slow growth is caused by the insufficient increment of the congestion windows as depict in Figure 1. In contrast, the proposed iTCP-Vegas dynamically adjust the *cwnd* increment, α and β according to the traffic condition, the throughput as shown in Figure 5 is improved by iTCP-Vegas. Particularly, at the initial stage of connection establishment, if the measured RTT is close to $BaseRTT$, the condition of Area 1 is held to fast increase the *cwnd* and throughput. Therefore, the number of packet is increased slightly in queue so that the RTT value is also increased. Simulation time in 54th second the RTT value is increased around 10 ms abruptly and

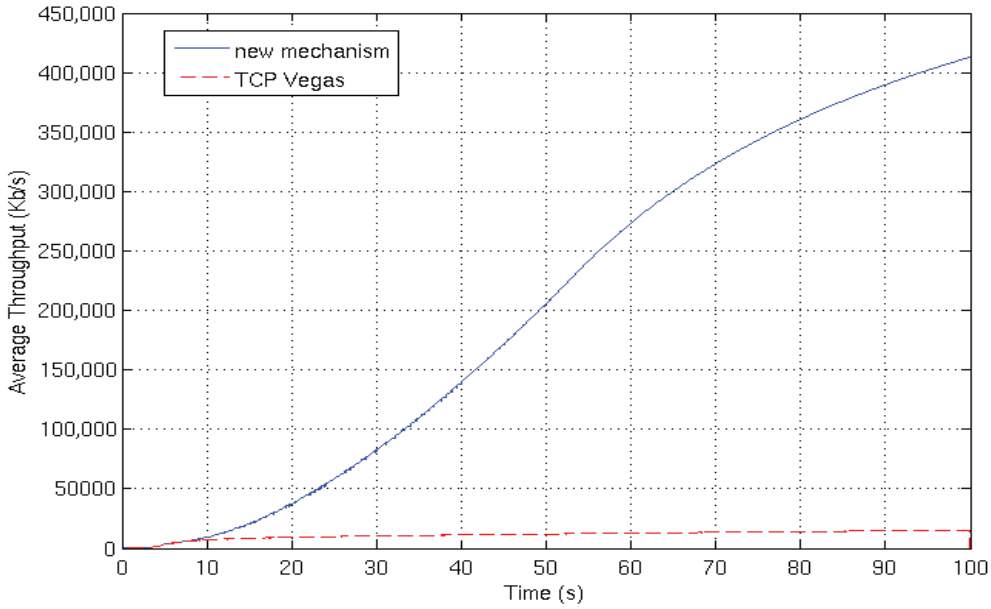

FIGURE 5. Comparison of throughput caused by iTCP-Vegas and TCP Vegas

TABLE 1. Average throughput of iTCP-Vegas and TCP Vegas

Mechanism	**Average throughput (in 100 seconds)**
TCP Vegas	15.33 Mbps
iTCP-Vegas	412.32 Mbps

then kept in 270 ms. Thus the most available bandwidth has almost been utilized. The *diff* value still lays on Area 1. When the bandwidth limitation has been reached, the *diff* value is raised and gone into Areas 2 and 3 in order. In Area 2, the *cwnd* growth will be slowed down and in Area 3 the *cwnd* size will not be revised, only the α and β values will be fine tuned. Due to the mechanism of Areas 2 and 3 above, the RTT value will stay steady. Then, the RTT value will reduce slightly.

The throughput comparison of iTCP-Vegas and TCP Vegas is shown in Figure 5. When $t = 10$ seconds, the throughput of iTCP-Vegas is similar to TCP Vegas due to the same algorithm of slow-start. When $t = 10$ seconds, the iTCP-Vegas increases the *cwnd* size rapidly to gather network bandwidth as much as possible.

Therefore, the throughput growth of iTCP-Vegas is faster than TCP Vegas, as shown in Table 1. Average throughput of iTCP-Vegas approaches to 412.32 Mbps. In this case, we demonstrate that the performance of iTCP-Vegas in long-distance high-speed networks is better than TCP Vegas.

The second scenario is designed to evaluate the performance of the proposed iTCP-Vegas approach in comparison with some variations of TCP-Vegas such as Quick Vegas, Vegas-A and Ada Vegas mechanisms. The network environment is as the same as the previous scenario. First, the throughput performance is verified. The average throughput of iTCP-Vegas comparison with Quick Vegas, Vegas-A and Ada Vegas is shown in Figure 7.

The average throughput of iTCP-Vegas is similar to the other three mechanisms in 10 seconds. The reason is that the slow-start algorithm of iTCP-Vegas is the same as the other three mechanisms. After 10 seconds, due to the iTCP-Vegas raises the *cwnd*

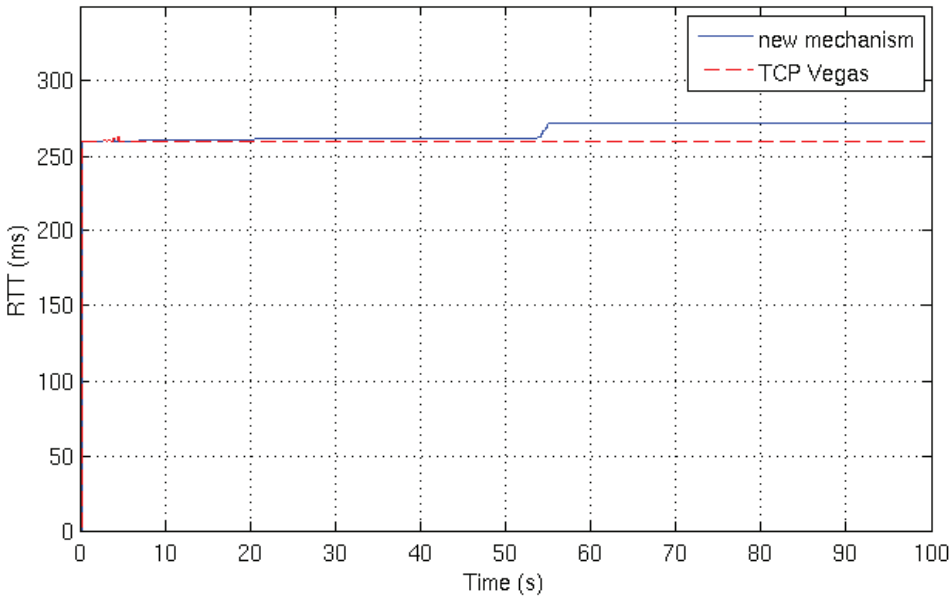

FIGURE 6. Comparison of *RTT* caused by iTCP-Vegas and TCP Vegas

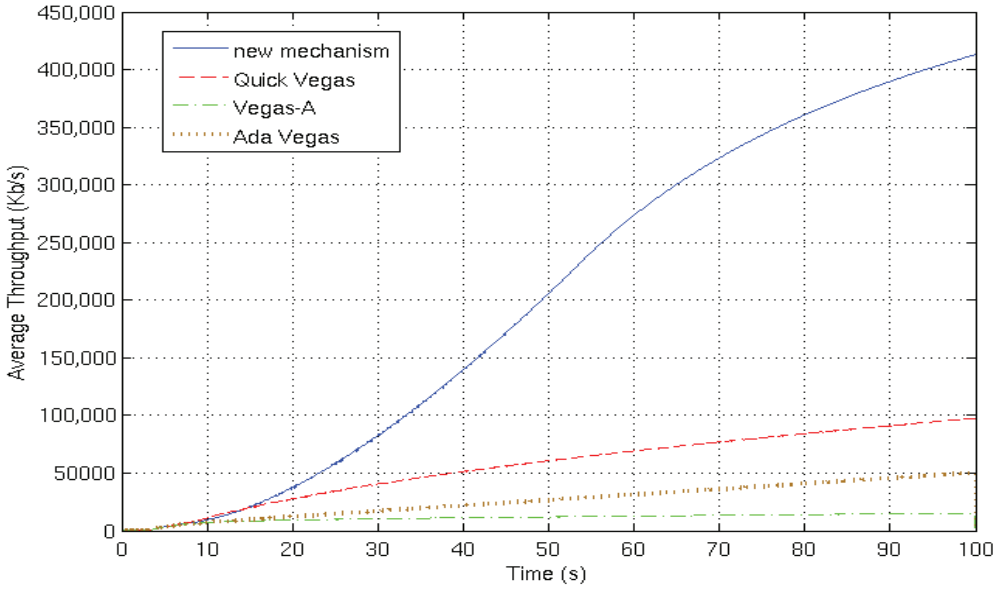

FIGURE 7. Average throughput of various TCP-Vegas variations

size rapidly to gather the whole network bandwidth. Therefore, the *cwnd* growth rate of iTCP-Vegas is faster than the other three mechanisms. The RTT value of the proposed iTCP-Vegas comparison with Quick Vegas, Vegas-A and Ada Vegas is shown in Figure 8. The results confirms that the proposed iTCP does not introduce too much RTT to the controlled connections. Obviously, the others can not fully utilize the bandwidth resource. That will lead to the worse resource utilization.

Then, we would like to explore the performance of the proposed iTCP-Vegas in the conditions of various distances. The link delay of backbone is specified as four conditions, including 240 ms, 440 ms, 640 ms and 840 ms, and the link bandwidth is specified to a typical 155-Mbps link. The four conditions of various RTTs representing the various

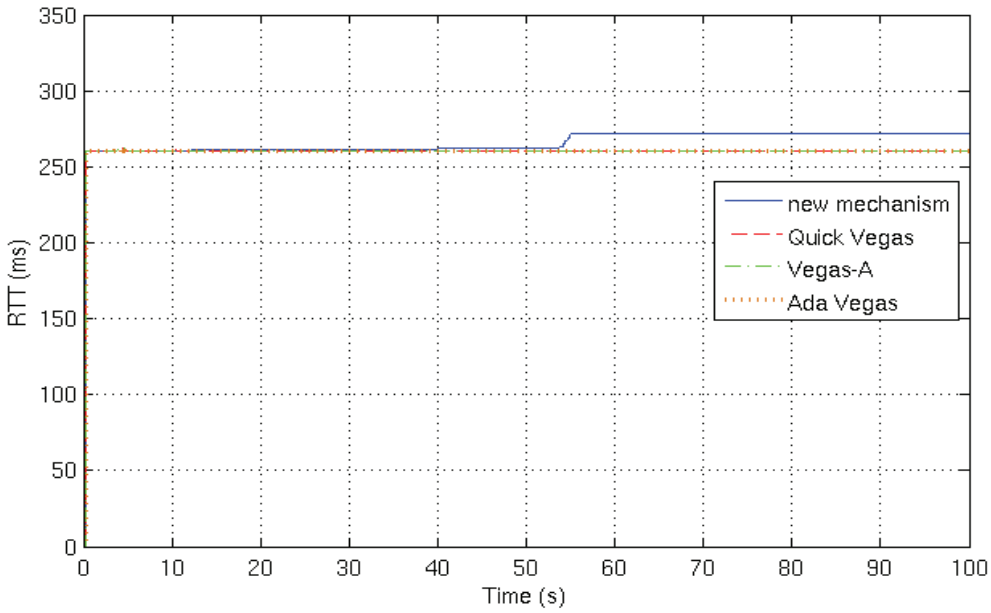

FIGURE 8. *RTT* dynamics of various TCP-Vegas approaches

long-distance links. We find that the higher *RTT* value, the longer time to reaches the bandwidth limitation. It is a nature that a long-distance connection has to take more time to estimate the actual throughput due to the longer time to have an acknowledgement from the receiver. Certainly, in Figure 9, we find that the lower *RTT* value, the higher throughput growth rate. The reason is that the shorter the receiving ACK time, the faster the throughput growth rate, vice versa. In the slow-start phase, the throughput growth rate is rise slowly. Once the *diff* value goes into congestion avoidance phase, that is, the iTCP-Vegas is activated. The throughput grows rapidly. Finally, the throughput growth rate goes slowly into convergence.

In Table 2, we show that the smaller the *RTT* value, the faster the average throughput can be reached to the most available bandwidth in the same time, vice versa. The main reason is that the *cwnd* growth rate is affected by *RTT* value. When the link delay is large, the transmitter needs more time to measure the packet RTT. Although, under the network conditions of these four different *RTT* values, the iTCP-Vegas reaches the half bandwidth in 80 seconds when the *RTT* is less 500 ms. According to the measurement by Shakkottai et al. [22] during 2001-2003, 80% traffic in global has the RTT less than 400 ms. Thus, in most cases, the proposed approach enables the connection with the throughput of the half of link bandwidth. Moreover, the smaller the RTT is, the shorter the time is to reach to the maximum bandwidth. Therefore, the iTCP-Vegas is applicable in common *RTT* values. In other words, the iTCP-Vegas is more feasible for long-distance networks due to its fast convergence.

5. **Conclusion.** In this paper, the new TCP Vegas, called iTCP-Vegas, is designed for the connections on long-distance high-speed networks. Although the RTT-based TCP, like TCP Vegas, has been recognized to benefit the response performance of TCP connections, the previous conservative schemes do not offers a feasible and reliably approach. The original TCP Vegas utilize the constants of α and β to evaluate the congestion. In the proposed iTCP-Vegas, the parameters of α and β is adjustable according to the congestion. Thus, the proposed approach has a more efficient response. In the performance evaluation,

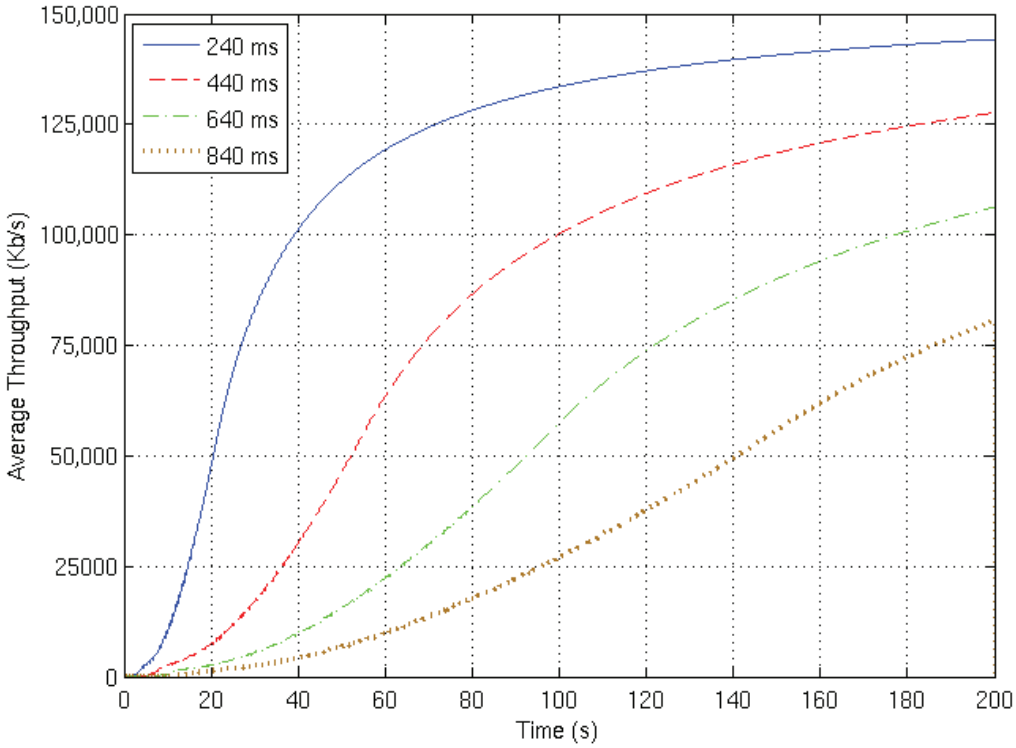

FIGURE 9. Average throughput variation of iTCP-Vegas in different conditions of RTT values

TABLE 2. Average throughput of iTCP-Vegas in different RTT values

***RTT* value (ms)**	**Average throughput (in 200 seconds)**	**Time to reach the bandwidth limitation (seconds)**
240	144.17 Mbps	22.13
440	127.48 Mbps	57.29
640	106.07 Mbps	102.52
840	80.35 Mbps	156.35

we use the RTT and the throughput to evaluate the performance of the proposed approach in comparison with the previous approaches, such as Quick-Vegas, Vegas-A, Ada Vegas. Through a series of experiments, we confirm that the iTCP-Vegas improves the bandwidth utilization in long-distance high-speed networks, even better than the other TCP Vegas variations. Thus, the proposed iTCP-Vegas benefits the Internet connections in dynamic traffic conditions.

REFERENCES

[1] A. Moarefianpour and V. J. Majd, Input-to-state stability congestion control problem of computer networks with nonlinear links, *International Journal of Innovative Computing, Information and Control*, vol.5, no.8, pp.2091-2105, 2009.

[2] G. Lee, Z.-K. Lee, H. Song and S. H. Park, Contention-aware QoS routing algorithm based on multi-rate service of IEEE 802.11 over mobile ad hoc networks, *International Journal of Innovative Computing, Information and Control*, vol.6, no.3(B), pp.1221-1229, 2010.

[3] R.-C. Chen and C.-M. Lin, Adaptive end-to-end delay equalization for TCP virtual path transmission in internet environment, *International Journal of Innovative Computing, Information and Control*, vol.6, no.3(A), pp.1069-1078, 2010.

[4] Y. Obashi, H. Chen, H. Mineno and T. Mizuno, An energy-efficient routing scheme with node relay willingness in wireless sensor networks, *International Journal of Innovative Computing, Information and Control*, vol.3, no.3, pp.1-10, 2007.
[5] S. Floyd, T. Henderson and A. Gurtov, The new Reno modification to TCP's fast recovery algorithm, *RFC3782*, 2004.
[6] L. Brakmo and L. Peterson, TCP Vegas: End to end congestion avoidance on a global internet, *IEEE Journal on Selected Areas in Communication*, vol.13, no.8, pp.1465-1480, 1995.
[7] J. S. Ahn, P. B. Danzig, Z. Liu and L. Yan, Evaluation of TCP Vegas: Emulation and experiment, *SIGCOMM*, pp.185-205, 1995.
[8] J. Semke, J. Mahdavi and M. Mathis, Automatic TCP buffer tuning, *The ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp.315-323, 1998.
[9] B. L. Tierney, D. Gunter, J. Lee, M. Stoufer and J. B. Evans, Enabling network-aware applications, *The 10th IEEE International Symposium on High Performance Distributed Computing*, pp.281-288, 2001.
[10] T. Dunigan, M. Mathis and B. Tierney, A TCP tuning daemon, *The ACM/IEEE Conference on Supercomputing*, pp.9-24, 2002.
[11] S. Floyd, HighSpeed TCP for large congestion windows, *RFC3649*, 2003.
[12] D. X. Wei, C. Jin, S. H. Lowa and S. Hegde, FAST TCP: Motivation, architecture, algorithms, performance, *IEEE/ACM Trans. on Networking*, vol.14, pp.1246-1259, 2006.
[13] T. Kelly, Scalable TCP: Improving performance in high speed wide area networks, *ACM SIGCOMM Computer Communication Review*, vol.33, pp.83-91, 2003.
[14] R. N. Shorten, D. J. Leith, J. Foy and R. Kilduff, Analysis and design of congestion control in synchronized communication networks, *The 12th Yale Workshop on Adaptive and Learning Systems*, 2003.
[15] Y. C. Chan, C. T. Chan and Y. C. Chen, An enhanced congestion avoidance mechanism for TCP Vegas, *Communications Letters, IEEE*, vol.7, pp.343-345, 2003.
[16] A. Maor and Y. Mansour, Ada Vegas: Adaptive control for TCP Vegas, *Global Telecommunications Conference, IEEE*, vol.7, pp.3647-3651, 2003.
[17] K. N. Srijith, L. Jacob and A. L. Ananda, TCP Vegas-A: Improving the performance of TCP Vegas, *Proc. of SCI Computer Communications*, vol.28, no.4, pp.429-440, 2004.
[18] Y. C. Chan, C. L. Lin, C. T. Chan and C. Y. Ho, Improving performance of TCP Vegas for high bandwidth-delay product networks, *Proc. of IEEE Advanced Communication Technology*, vol.1, pp.987-997, 2006.
[19] Y. Lei, R. Zhu and W. Wang, A survey on TCP protocol and RTT estimation, *Proc. of the 6th World Congress on Intelligent Control and Automation*, Dalian, China, 2006.
[20] D. J. Leith, R. N. Shorten and G. McCullagh, Making available Base-RTT for use in congestion control applications, *IEEE Communications Letters*, vol.12, no.6, pp.429-431, 2008.
[21] Y.-T. Li, D. Leith and R. N. Shorten, Experimental evaluation of TCP protocols for high-speed networks, *IEEE/ACM Trans. on Networking*, vol.15, no.5, pp.1109-1122, 2007.
[22] *NS2, Network Simulator Version 2*, http://www.isi.edu/nsnam/.
[23] S. Shakkottai, R. Srikant, N. Brownlee, A. Broido and K. C. Claffy, *The RTT Distribution of TCP Flows in the Internet and its Impact on TCP based Flow Control*, http://www.caida.org/.