International Journal of Control and Automation
Vol. 9, No. 8 (2016), pp.109-126
http//dx.doi.org/10.14257/ijca.2016.9.8.12

Invs: A New Congestion Control Algorithm for Heterogeneous Networks

Zhiming Wang, Xiaoping Zeng, Man Xu, Xue Liu and Li Chen

College of Communication Engineering, Chongqing University, Chongqing 400030, China.
jamewzm@163.com

Abstract

The available bandwidth, round trip time (*RTT*) *and packet loss rate in the Internet can vary over many orders of magnitude, which is known as the heterogeneity of the Internet. To cope with the heterogeneity in Internet Congestion Control, we propose a new TCP protocol, namely Invs, that has three key components. First, it employs a novel increase function of congestion window with an adaptive increase factor that adjusts through estimation of path* bandwidth-delay product (*BDP*) *and minimum RTT. The increase function can modulate the window growth rate to fit with the path condition. Second, Invs adopts a novel scheme that adaptively adjusts the queue threshold in loss classification, which distinguishes non-congestion loss from congestion loss. Third, Invs introduces a new back-off scheme to enable quick convergence of congestion window. The performance analysis and evaluation show that Invs achieves good fairness, friendliness, RTT fairness and utilization in heterogeneous networks.*

Keywords*: Transmission control protocol (TCP), congestion control, fairness, bandwidth delay product (BDP), high speed*

1. Introduction

With the widely deployment of wireless technologies and the popularity of mobile Internet, the traditional communication network has evolved into a seamless global Internet [1]. It encompasses a variety of heterogeneous IP networks (wired, wireless and satellite networks). Nowadays, about 90% of Internet traffic is carried by TCP. In heterogeneous networks, TCP must cope with the different transmission media crossed by Internet traffic to provide reliable end-to-end data delivery.

The Internet Congestion Control Research Group (ICCRG) has outlined a variety of distinguishing link and path characteristics of the Internet [2]. (1) Available bandwidth can be either scarce over radio links or abundant over high-speed optical links. (2) Round trip time (RTT) ranges from much less than a millisecond (local interconnects) to very large up to or over 500 milliseconds (satellite links). (3) Packet loss rate ranges from very low over optical fiber links (less than 10^{-6}) to very high over certain wireless links (higher than 1%). Consequently, the available bandwidth, RTT and packet loss rate in the Internet can vary over many orders of magnitude. In fact, these characteristics describe the heterogeneity, which is one of the global challenges that are known today in Internet Congestion Control [2]. In order to achieve good performance in heterogeneous networks, TCP must be designed to cope with the challenge of the Internet heterogeneity.

Many works have tried to improve the performance of TCP in heterogeneous networks and a lot of TCP variants have been proposed [4-8]. Some proposals adopt aggressive growth functions to utilize the available bandwidth efficiently [3], such as HS-TCP, Hybla [4], STCP, TCP Compound (CTCP) [5], *etc*. Some proposals modify both the increase function of congestion window and the back-off scheme [3], such as Cubic [6], TCP Illinois [7], TCP-Cherry, and ER-TCP [8], *etc*. However, these proposals exhibit

ISSN: 2005-4297 IJCA

throughput degradation over wireless links, where packet loss is no longer the indication of congestion. Some proposals use packet loss classification schemes as well as different back-off schemes depending on the kind of the packet loss [3, 9 ,10], such as Veno, JTCP, TCP Westwood and its variants, *etc*. However, these proposals are designed only for the wire/wireless mixed network. Certain other proposals use particular congestion control algorithms. PEPsal [3, 9], indirect-TCP [9] split the connection into several portions to shield wireless network from wired network. TCP-Jersey [3] adopts a router-aided explicit congestion warning scheme to determine the cause of loss. Network Coded TCP [11] incorporates network coding into TCP to provide significant throughput gains in lossy networks. Most of the above proposals focus on a particular environment (traditional networks, satellite networks, wired/wireless networks, *etc*.), and few of them can perform well over lossy, high-speed and long-delay links.

As mobile broadband access widely deployed all over the world, the quality of wireless networks play an increasing important role in the quality of service we experienced. With a multitude of wireless links, packets may be dropt or lost due to high bit error, collision or link failure. Because packet loss is no longer the indication of network congestion in wireless networks, most of TCP enhancements rely on queue-delays or queue-lengths to distinguish non-congestion losses from congestion losses. With the increasing variety of wireless networks, the fixed queue threshold schemes are no longer accurate. Thus, an adaptive queue threshold is introduced in the loss classification.

In heterogeneous networks, TCP traffic also encounters high-speed fiber links, long-delay satellite links and other wireless networks (cellular and Ad Hoc networks) [1, 2]. The high-speed links and long-delay links of the network result in large Bandwidth-delay product (BDP) [12], which indicates a large amount of data allowed in-flight. In large BDP networks, standard TCP need a long time to reach the steady state that results in the underutilization of the available bandwidth. The existing high-speed TCP variants increase the congestion window with the same growth rate in large BDP networks regardless of high-speed links and long-delay links. In fact, long-delay link results in long RTT that leads to the increase of the delay required to receive congestion feedback and the delay of congestion feedback might lead to more packet losses when congestion occurs. An adaptive increase factor that incorporates RTT and BDP is introduced. The BDP represents path capacity while the RTT reflects the conservation of RTT influence in path BDP. The increase factor is used to adaptive accommodate TCP parameters to the scenario. Finally, a new growth function of congestion window with the increase factor is proposed.

In this paper, concerning the challenge of heterogeneity and fairness, we propose a new congestion control algorithm for heterogeneous networks, namely Invs. It distinguishes itself from other proposals in three aspects: 1) to achieve well efficiency and fairness in a whole range of bandwidth and RTT dynamics, Invs introduces a new growth function of congestion window with an adaptive increase factor which adjusts to fit the path condition through estimation of path BDP and minimum RTT; 2) it adopts a scheme that adaptively adjusts the queue threshold in loss classification; 3) it introduces a new back-off scheme to enable quick convergence of congestion window. In addition, Invs preserves the end-to-end semantics and only need modification of TCP sender.

The rest of the paper is organized as follows. Section 2 describes the proposed algorithm in detail. Section 3 provides the performance analysis, and Section 4 presents the performance evaluation. The deployments of various TCPs result in that the Internet TCP traffic composed of mixed TCP flows, and the performance with mixed TCPs in heterogeneous networks is investigated. Section 5 concludes the paper.

2. Proposed Congestion Control Algorithm: Invs

In heterogeneous networks, TCP must use the available bandwidth efficiently over large BDP path and be fair with other TCP flows in low BDP path. Thus, Invs introduces a growth function of congestion window with adaptive increase factor. To deal with the performance degradation of non-congestion loss, a new loss classification scheme and a corresponding back-off scheme are proposed. The notations used are listed in Table. I.

Table I. Notations

Notations	Definition
BDP	Estimated bandwidth-delay product.
cwnd	The size of congestion window
RTT_{min}	Estimated minimum round trip time.
$cwnd_{sp}$	The *cwnd* that network bandwidth is fully used.
ssthresh	Slow start threshold.
β	Multiplicative decrease factor.
k	Increase factor of Invs.
$buffer_{est}$	The estimated number of packets queued in the network
maxbuffer	The maximum number of packets can be queued in the network
W_{max}	Window size just before last window reduction.

2.1. Growth Function of Congestion Window

In order to operate efficiently over large BDP path and be fair with standard TCP in low BDP path, Invs introduces a concave growth function of congestion window with a dynamic factor *k*. The update scheme of congestion window is given as follows.

$$cwnd += \begin{cases} \dfrac{cwnd_{sp} - cwnd}{k \cdot cwnd_{sp}}, & cwnd < cwnd_{sp} \\ \dfrac{cwnd - cwnd_{sp}}{k \cdot cwnd}, & cwnd > cwnd_{sp} \\ \dfrac{1}{cwnd}, & else \end{cases} . \quad (1)$$

where *cwnd* is the size of congestion window. *k* is the adaptive increase factor, which depends on the estimated network conditions and is determined in equation (2). $cwnd_{sp}$, represents *saturation point*, equals the estimated *cwnd* when available resources of the path is fully used.

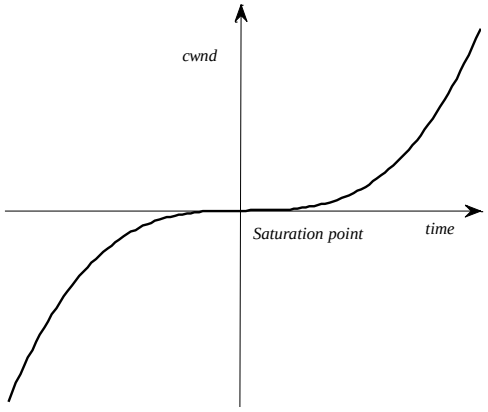


Figure 1. The Window Growth Function of Invs

Figure 1 illustrates *cwnd* curve as the window increases in congestion avoidance phase. From Figure 1, we can see that Invs increases *cwnd* quickly at the beginning of a new CA phase and slow down when *cwnd* approaches *saturation point* ($cwnd_{sp}$). When *cwnd* grows beyond *saturation point*, it slowly accelerate to probe the available resources of the path. The region around $cwnd_{sp}$ (*saturation point*) indicates fully use of network bandwidth. In other words, if TCP sender continues increasing its transmission rate when $cwnd>cwnd_{sp}$, congestion will occur. If TCP sender stops increasing the transmission rate when $cwnd<cwnd_{sp}$, network will be underutilized.

2.2. Increase Factor *k*

In Invs, factor *k* is a parameter that relates path characteristics with the increment of *cwnd*. The value of factor *k* is significant to the efficiency of Invs and fairness to other TCP variants. In Invs, factor *k* is given as a function of path BDP and minimum RTT as follows.

$$k = c\log_2\left(\frac{RTT_{ref}}{RTT_{\min}}\right)\log_2\left(\frac{BDP_{ref}}{BDP_{est}}\right) \tag{2}$$

where c is a constant scale factor; RTT_{ref} is the reference RTT; RTT_{min} is the instantaneous minimum RTT; BDP_{ref} is the reference BDP with a default value of 500 Mbit (which is equivalent to a link with a bandwidth of 1 Gbps and an RTT of 500 ms); *BDP* represents the path BDP_{est}. RTT_{ref} is the reference RTT and the default value is 1 s. In order to guarantee that *k* be meaningful and useful, *k* is set to 1 when its value determined by (2) is less than 1. Since the increment of *cwnd* on each ACK would be large enough in these conditions, the growth rate of cwnd is not increased any more.

In Invs, factor *k* decreases as the path BDP increases. The increase of the path BDP is due to the increase of RTT or bottleneck bandwidth, which indicates that more packets are allowed in-flight to fill the path pipe. In order to fill the path pipe, the increment of *cwnd* per ACK is increased as the path BDP increases. Therefore, in formula (2), factor *k* is designed to decrease when BDP increases.

High-speed TCP variants increase *cwnd* aggressively in large BDP networks to achieve better performance. The increase of path BDP can be attributed to two reasons. (1) Increase of RTT. 2) Increase of bottleneck bandwidth. The increase of RTT results in the increase of time to receive the network feedback. The delay of congestion feedback may lead to more packet losses. If this drawback is ignored when RTT increases, the congestion may occur more frequently and more packets may be lost in each congestion. High-speed TCP like Cubic, Illinois, *etc*. increases *cwnd* with the same increment when the path BDP is fixed regardless of what the RTT is. However, Invs adopts RTT_{min} in factor *k* to take the RTT influence into consideration.

2.3. Action upon Packet Loss

Upon receipt of triple duplicate ACKs, Invs determines new *cwnd*, new *ssthresh* and new $cwnd_{sp}$ as follows.

$$ssthresh = \begin{cases} \beta \cdot cwnd, & buffer_{est} \geq \min(\delta, maxbuffer) \\ \min(BDP_{est}, cwnd), & else \end{cases} \tag{3}$$

$$cwnd = \begin{cases} \beta \cdot cwnd, & buffer_{est} \geq \min(\delta, maxbuffer) \\ \min(BDP_{est}, cwnd), & else \end{cases} \quad (4)$$

where β is the multiplicative decrease factor and $buffer_{est}$ is the estimated number of packets currently queued in the networks. δ is a predefined threshold of queue length for large buffer links. *maxbuffer* maintains the maximum number of packets that can be queued in the network.

In heterogeneous networks, packet loss is no longer the indication of congestion. As in formula (3) and (4), when three duplicate ACKs have been received, Invs takes different back-off schemes depending on whether the estimated queue length is greater than the minimum value of δ and *maxbuffer*. If the current queue length exceeds the threshold, the loss is considered caused by congestion and the traditional back-off process is called. Otherwise, the loss is considered as a non-congestion loss. The *cwnd* and *ssthresh* are set to the minimum value of the estimated BDP and *cwnd*.

Traditional loss classification scheme determines whether a loss is congestion loss by comparing the queue length or queue delay with a fixed threshold. In heterogeneous networks, the queue length might vary with a large magnitude. Fixed threshold can lead to misjudgment. Invs uses the minimum of δ and *maxbuffer* as the new queue threshold to differentiate non-congestion loss and congestion loss. The incorporation of *maxbuffer* ensures that the loss classification scheme can work proper over small buffer links.

Invs maintains $cwnd_{sp}$ that estimates *cwnd* when the network resources are fully used. The available network resources change due to the starting of new flows and the termination of transmitting flows. When packet loss occurs, $cwnd < cwnd_{sp}$ indicates that congestion occurs earlier this time, which means that the available network resources are diminishing. To enable fast convergence of competing flows, Invs sets $cwnd_{sp}$ to a value less than *cwnd* as in (5). If $cwnd \geq cwnd_{sp}$, Invs set $cwnd_{sp}$ to *cwnd* to track the new *saturation point*.

$$cwnd_{sp} = \begin{cases} \frac{1+\beta}{2} \cdot cwnd, & cwnd < cwnd_{sp} \\ cwnd, & else \end{cases} \quad (5)$$

2.4. Parameter Estimation

Invs uses four parameters in congestion control, including BDP, minimum RTT, queue length and *maxbuffer*. This part presents the estimation of these parameters.

2.4.1. Estimation of BDP

Invs uses BDP in calculation of increase factor *k* and resetting *ssthresh* and *cwnd* after packet loss. Estimation of BDP incorporates available bandwidth estimation (ABE) and RTT estimation. The method in [12] estimates available bandwidth upon each ACK. The intention of adopting BDP in factor *k* is to embody the path characteristics, and an average bandwidth-delay product would be enough. Thus, Invs estimates available bandwidth once every RTT, which equals to the average throughput. Furthermore, a minimum estimation period and a low pass filter are used to remove the spike of estimated bandwidth. Figure 2 introduces the pseudocode of the ABE algorithm.

```
//initiation
#define TCP_INVS_RTT_MIN    (HZ/50)          /* 20 ms */
Start=now;
BWest =0;

if ( ACK is received ) then
      delta = now – measure_start;
      if (data ACKed)
            data_acked_this_epoch += ack_seq - snd_una; // data acked by this ACK
      endif
      if RTT> 0 and delta > max(RTT, TCP_INVS_RTT_MIN) then
            /* low pass filter with coefficient a1 and a2 */
            BWest=a1*BWest+a2* data_acked_this_epoch / delta;
            data_acked_this_epoch = 0;
            measure_start = now;
      endif
endif
```

Figure 2. Pseudocode of the ABE algorithm

2.4.2. Tracking of the Minimum RTT

The minimum RTT only changes when network condition or flow route changes. The update scheme of the minimum RTT can track it accurately only when new minimum RTT is less than the former one. When the minimum RTT of the connection increases due to rerouting or else, the update scheme of the minimum RTT fails. To accurately track the minimum RTT when link or routes changes, the minimum RTT is re-initiated when the difference between RTT and minimum RTT is larger than a threshold in Invs.

2.4.3. Estimation of the Queue Length and *maxbuffer*

The queue length estimates the number of backlog packets along the path while *maxbuffer* estimates the maximum number of packets that can be queued along the path. In order to distinguish non-congestion loss from congestion loss, the queue length is compared with the threshold in formula (3) upon packet loss. The queue length is estimated as follows.

$$buffer_{est} = \left(RTT - RTT_{\min} \right) BW_{est} \tag{6}$$

where BW_{est} is the estimated available bandwidth.

Finally, *maxbuffer* records the maximum queue length. In order to adapt to route change, *maxbuffer* is re-initiated upon timeout.

3. Performance Analysis

This section models the steady state of Invs, and presents the analysis of friendliness, fairness and RTT-fairness of Invs.

3.1. Steady State Throughput of Invs

In steady state, Invs increase *cwnd* from βW_{max} to W_{max} periodically, where W_{max} is the maximum *cwnd* in a CA phase. The increase factor *k* is deemed constant for a certain path. From formula (1), *cwnd* at the beginning of *i*-th round can be expressed as follows.

$$cwnd(i) = W_{max}\left(1-\left(\frac{\beta(k-\beta)}{k}\right)^{i-1}(1-\beta)\right). \quad (7)$$

where i counts the number of RTT rounds from the beginning of a CA phase

From formula (7), we note that the number of packets increased in each RTT obeys an exponential function. Therefore, the number of RTT rounds in the CA phase can be obtained as follows.

$$n = \frac{\ln(1-\beta)W_{max}}{\ln\frac{k}{\beta(k-\beta)}}. \quad (8)$$

Then the expected total number of packets in the CA phase can be obtained as follows.

$$Y = \sum_{i=1}^{n} cwnd(i) = W_{max}\left(n - \frac{k\left((1-\beta)W_{max}-1\right)}{\left(k-k\beta+\beta^2\right)\left((1-\beta)W_{max}\right)}\right). \quad (9)$$

In steady state, the expected total number of packets in the CA can also be expressed by packet loss rate p as follows.

$$Y = \frac{1}{p}. \quad (10)$$

By equaling the right hand side of equation (10) and (11), we can get.

$$W_{max}\left(n - \frac{k\left((1-\beta)W_{max}-1\right)}{(1-\beta)W_{max}\left(k-k\beta+\beta^2\right)}\right) = \frac{1}{p}. \quad (11)$$

Then we have

$$W_{max} = \left(n - \frac{k\left((1-\beta)W_{max}-1\right)}{(1-\beta)W_{max}\left(k-k\beta+\beta^2\right)}\right) = \frac{k(1-\beta-p)+\beta^2}{p\left(\left(k-k\beta+\beta^2\right)n - k(1-\beta)\right)} \quad (12)$$

Substituting formula (8) into equation (12)and solving using Lambert ***W***-function, which is the inverse function of $f(W)=We^W$, the closed-form expression of W_{max} can be obtained as follows.

$$W_{max} = \frac{\left(k(1-\beta-p)+\beta^2\right)\ln\left(\frac{k}{\beta(k-\beta)}\right)}{\left(k-k\beta+\beta^2\right)lambertW(f)}. \quad (13)$$

where $f = \left(\frac{k(1-\beta-p)+\beta^2}{p\left(k-k\beta+\beta^2\right)}(1-\beta)\ln\left(\frac{k}{\beta(k-\beta)}\right)e^{-\frac{k(1-\beta)}{k-k\beta+\beta^2}\ln\left(\frac{k}{\beta(k-\beta)}\right)}\right)$

Then the average throughput can be expressed as follows.

$$Th = \frac{Y}{n\cdot RTT} = \frac{\ln\frac{k}{\beta(k-\beta)}}{p\cdot RTT\cdot\ln(1-\beta)W_{sp}}. \quad (14)$$

3.2. TCP Friendliness, Fairness and RTT-Fairness

TCP fairness defines the equality in sharing bandwidth among flows that experience equivalent path congestion[2]. TCP friendliness defines the fairness of bandwidth sharing between a given flow and other flows using different TCP congestion control algorithms [5]. RTT-fairness defines the fairness of bandwidth sharing among same flows with different RTTs.

3.2.1. TCP Fairness

TCP Fairness is an important index for TCP variants. In order to show fairness of Invs, an informal vector-based illustration of two Invs flows sharing the same path is showed in Figure 3 similar to that in [14]. Suppose that flow 1 starts first and has achieved a high *cwnd* in CA phase. Flow 2 starts later and at the end of slow start phase, the bottleneck buffer overflows and congestion occurs. Both flow 1 and flow 2 back off their congestion windows with the decrease factor β to point A in Figure 3. According to formula (6) in steady state, flow 1 set its $cwnd_{sp}$ to a lower value of the maximum *cwnd* in last CA phase, while flow 2 set its $cwnd_{sp}$ to the maximum *cwnd* in the last CA phase. Then flow 2 gets more bandwidth than flow 1 before overflow of bottleneck buffer. Because flow 1 increases its *cwnd* to a higher value in this CA than in the last CA when congestion occurs at point B. Both of them back off their congestion windows to point C. Through congestion and back-off again and again, flow 1 gradually releases bandwidth to flow 2 until an equal bandwidth share is reached.

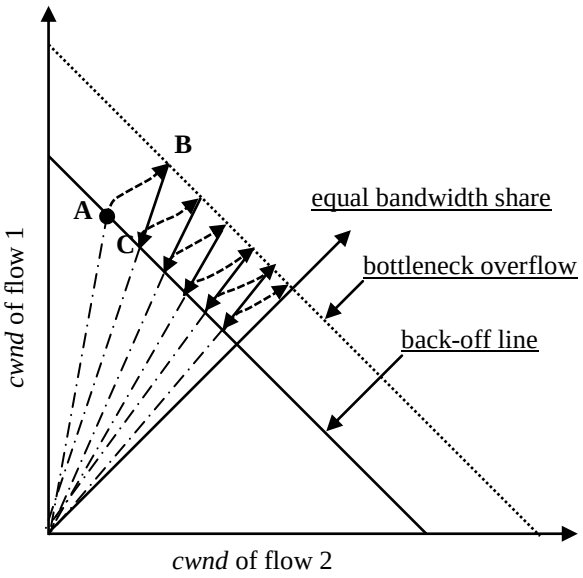


Figure 3. Convergence toward Fair Bandwidth Share

Figure 4. shows the congestion windows of two Invs flows that share the same path. It is obtained by running NS2 simulation on a dumbbell network with a bottleneck capacity of 100 Mbps and an RTT of 80 ms. From the results, we can see that the congestion windows of the two Invs flows converge. It also demonstrates that the high *cwnd* flow sets its $cwnd_{sp}$ to a lower value than the *cwnd* when packet loss occurs, which is in accordance with the fast convergence in formula (6).

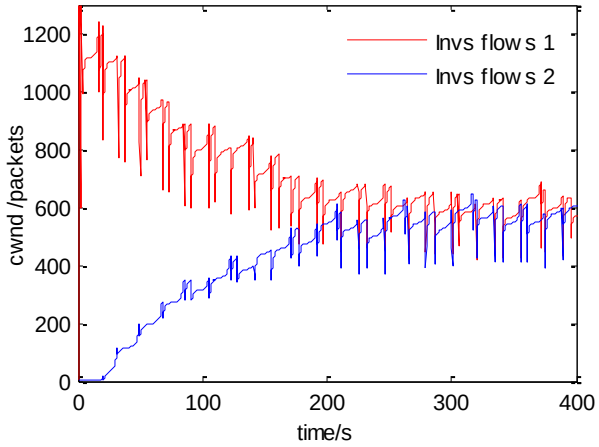


Figure 4. Congestion Windows of 2 Competing Invs Flows

3.2.2. TCP Friendliness

Invs is friendly to TCP Reno/NewReno/CTCP in small BDP networks. Because in small BDP networks, the factor *k* would be very large. A large *k* makes the increase rate of congestion window is close to or even less than that of TCP Reno. When the congestion window of Invs is lower than that of TCP Reno, Invs uses the window of TCP Reno. The congestion window of TCP Reno in Invs is estimated as follows [6].

$$W_{TCP}(t) = 3\frac{1+\beta}{1-\beta}\frac{t}{RTT} + \beta W_{\max} \tag{15}$$

where ***t*** is the time from the beginning of the CA phase.

Figure 5 shows the congestion windows of an Invs flow and a Reno flow sharing the same bottleneck over a small BDP network with a bottleneck bandwidth (BW) of 10 Mbps and an RTT of 10 ms. The results show that Invs is fair with Reno in a small BDP network. Figure 6 shows the congestion windows of an Invs flow and a Reno flow over a high BDP network (100 Mbps Bottleneck bandwidth and 100 ms RTT). Invs flow takes about 80% of the total bandwidth and Reno flow takes about 20% bandwidth.

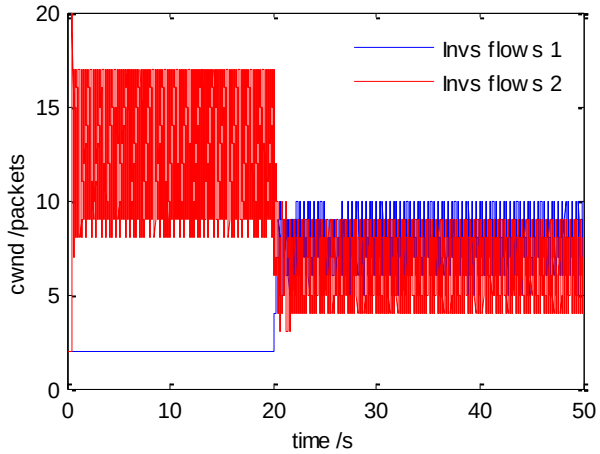


Figure 5. Congestion Windows of Invs and Reno in a Small BDP Network

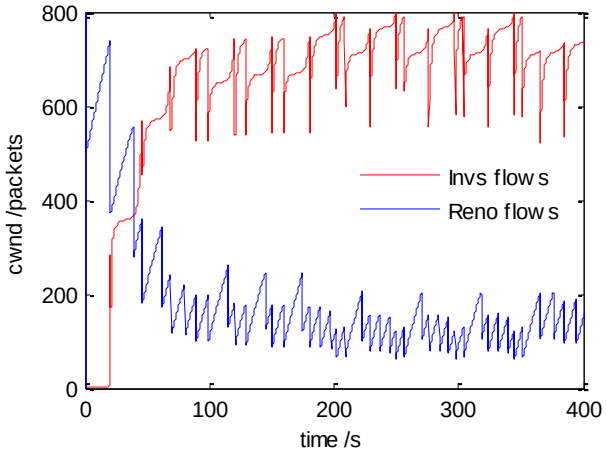


Figure 6. Congestion Windows of INVS and Reno in a High BDP Network

3.2.3. RTT-Fairness

Most of TCP variants, similar to TCP Reno, have the RTT unfairness problem, especially over satellite links. Some TCP variants, such as Hybla, shows well RTT-fairness. However, TCP variants with perfect RTT-fairness ignore the fact that in long RTT environment, the sender waits long to be informed about the network condition. The delay of congestion notification can cause severer congestion when using aggressive growth function of *cwnd*. Invs aims to achieve a compromised RTT-fairness between perfect RTT-fairness (Hybla) and RTT unfairness of Reno using an adaptive fact *k*. If factor *k* is fixed, Invs flows, similar to BICTCP flows, would be unfair with flows that have different RTTs. In Invs, factor *k* is a function of path BDP and RTT, and decreases as RTT increases, which results in the increase of the aggressiveness of Invs. On the other hand, the logarithm function reduces the influence of RTT variants on the aggressiveness of Invs. If a linear function were used, it would have similar RTT fairness as Cubic. Therefore, a compromised RTT-fairness is achieved.

4. Performance Evaluation

This section presents the NS2 performance evaluation results of Invs. The fairness, delivery ratio and average throughput of Invs are evaluated and compared under various network scenarios with Reno, CTCP, Cubic, Illinois, Westwood and ER-TCP. There are two topologies, namely dumbbell and hybrid, used in our evaluation. The dumbbell topology is configured with a bottleneck of 100 Mbps and a side bandwidth of 1000 Mbps. The link delay varies in each scenario. The dumbbell topology is used in fairness evaluation under traditional environments and RTT fairness evaluation. The hybrid topology and configurations are shown in Figure 7. The hybrid topology is used to evaluate fairness between satellite flows and wired flows and TCP performance over wireless links. The queue sizes in all the simulation, without specification, are set to the path BDP by default. Furthermore, in order to evaluate fairness of Invs, the Jain's fairness index is used as in (16).

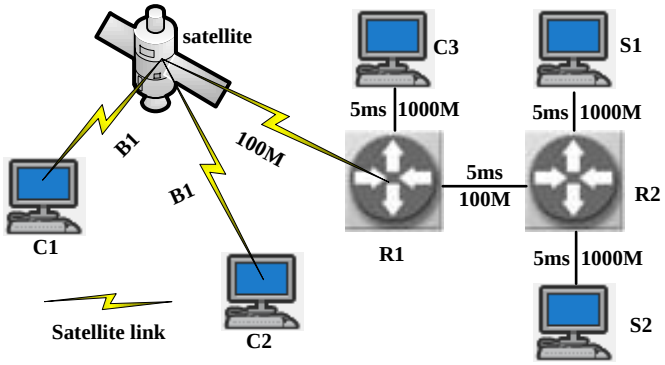


Figure 7. Hybrid Topology

$$F_{index} = \frac{\left(\sum_{i=1}^{m} x_i\right)^2}{m\sum_{i=1}^{m} x_i^2} \tag{16}$$

here m is for the number of TCP flows and x_i is the throughput of the i-th flow.

4.1. TCP Fairness with 50 TCP Flows

This part evaluates the performance of Invs under traditional environments. We investigate fairness and link utilization of Invs with 50 compound flows under the dumbbell topology. The 50 TCP flows are composed of 80% background flows and 20% test flows. The 80% background flows consist of 20% Reno flows, 30% Cubic flows, 30% CTCP flows. The background flows are used to emulate the traditional Internet TCP traffic which is dominated by Reno flows, Cubic flows and CTCP flows [15]. Test flows are flows that use one of the test TCP variant (Illinois, Invs, Westwood, Cubic, CTCP, ER-TCP). All TCP flows are randomly generated according to the predefined percentage and uniformly started between 0s and 5s. The 50 flows scenarios are used to compare the influences and performance when some flows, which use the test TCP variants, are competing with the dominate background flows.

Figure 8 shows the fairness indexes of the 50 flows scenarios when RTT varies from 50 ms to 200 ms. Figure 9 shows the corresponding link utilizations under the same scenarios. From Figure 8, we note that ER-TCP has the worst fairness index than other TCP variants when RTT is 50 ms. This is because ER-TCP is designed for satellite links. It behaves too aggressive in low RTT environment, which leads to the unfair problem when RTT is 50 ms. When the RTT increases from 100 ms to 200 ms, the fairness indexes of all the tested TCP variants (Cubic, Invs Westwood, Illinois and CTCP) drop sharply. This is attributed to that Reno flows are unable to compete with other high-speed TCP variants in long-delay environments. From Figure 9, we note that the link utilizations of all the scenarios decrease a little as the RTT increases. Based on both the fairness and the link utilization, we can deduce that Invs shows a comparable fairness and link utilization with TCP variants, such as Cubic, Westwood, Illinois, and CTCP, in traditional environments.

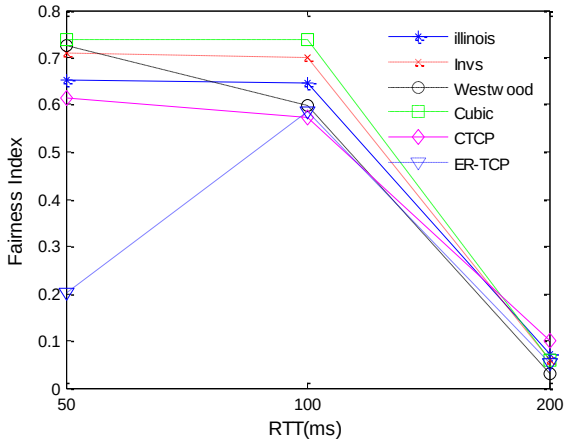


Figure 8. Fairness with 50 Flows

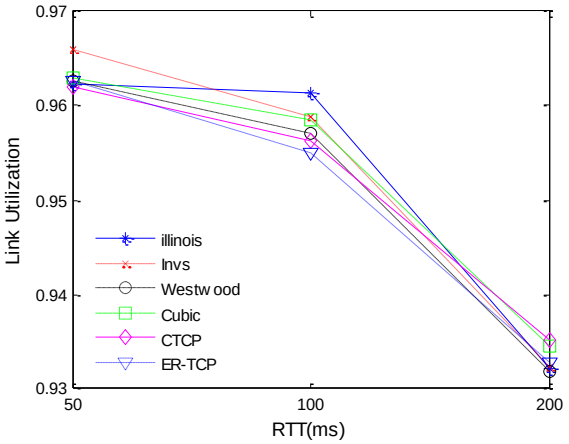


Figure 9. Link Utilization with 50 Flows

4.2. TCP Fairness in Long RTT/ High BDP Networks

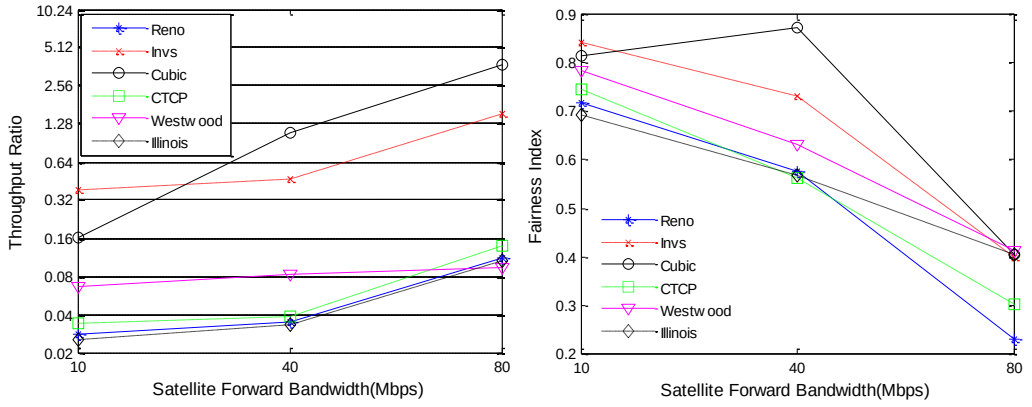
In order to evaluate the performance of Invs in the hybrid satellite and wired networks (Figure 7). Five flows of the tested protocol are set up between client C1 and server S1 and 20 background flows are set up between C3 and S2. Cubic, CTCP, Reno, Invs, Westwood, Illinois are tested in this section. The background flows are composed of 20% Reno flows, 50% Cubic flows and 30% CTCP flows. All flows are randomly started between 0s and 5s. Each simulation lasts 500s. The bandwidth (BW) of satellite forward link (SFL) varies from 10 Mbps to 80 Mbps.

Figure 10(a) shows the ratio of the average throughput of satellite flows to the average throughput of wired flows. The results show that the throughput ratios of all tested protocols increase as the bandwidth of the SFL increases. The results also shows that the average throughput of satellite flows is lower than the average throughput of wired flows when the BW of SFL is 10 Mbps, which indicates that the average BW of the SFL per flow (2 Mbps) is less than the average wired bandwidth per flow (4 Mbps). On the other hand, when the BW of the SFL is 80 Mbps, the throughput ratio of Cubic is 5 and Invs is 1.5, while the ratios of other TCP variants (Reno, CTCP, Westwood and Illinois) are less than 0.3. This indicates that the satellite Cubic flows take much more bandwidth than the wired flows in sharing same bottleneck bandwidth when the satellite bandwidth is sufficient. The satellite Invs flows have approximately the same throughputs as the wired flows while the

satellite flows of other TCP variants are unable to compete with the wired flows in these cases.

Figure 10(b) shows the fairness index. Figure 10(c) and Figure 10(d) show the utilization ratio of SFL and utilization ratio of wired link, respectively. The results show that Cubic has the highest fairness index when the BW of the SFL is 40 Mbps. The satellite Cubic flows achieves comparable bandwidths as the wired flows when the BW of the SFL is 40 Mbps. The aggressiveness of Cubic improves the utilization ratio of the SFL as show in Figure 10 (c). On the contrary, it impairs the utilization ratio of wired link as shown in Figure 10 (d). The satellite Cubic flows achieves good utilization ratio of the SFL when the BW of the SFL is 80 Mbps. However, from the results of throughput ratio, fairness index and utilization ratio of wired link, we note that the satellite Cubic flows are too aggressive that they have occupied more bandwidth than that of the wired links. Because fairness requires that all the flows passing the bottleneck have approximately equal throughputs. In Figure 10(d), Invs reduces the utilization ratio of the wired link by 5%-20% when the BW of the SFL is 10 Mbps. However, from Figure 10 (a), (b) and (c), we note that Invs greatly improves the utilization ratio of the SFL when the BW of the SFL is 10 Mbps. Invs also achieves good throughput ratio when the BW of the SFL is 40 Mbps and 80 Mbps. From Figure 10(a) and (c), the results also show that other satellite TCP (Reno, CTCP, Westwood and Illinois) flows are capable to compete with the wired flows. As a result, almost 90% of the satellite bandwidth, which is very costly, is wasted. On the other hand, Cubic flows only utilize 50% of the wired bandwidth, which is often very high. Therefore, Invs is designed and achieves a compromising performance between the utilization of the satellite bandwidth and the wired bandwidth.

In addition, Figure 10(e) show the total delivery ratio. The results of Figure 10(e) show that the delivery ratio of all the TCP protocols are very high, only with a slight drop when the BW of the SFL increases. This indicates that the higher the throughput of the satellite flows, the severer the congestion on the wired flows. In fact, the higher throughput of the satellite flows, the longer time the congestion lasts, which might cause more packet losses to wired flows.



(a) (b)

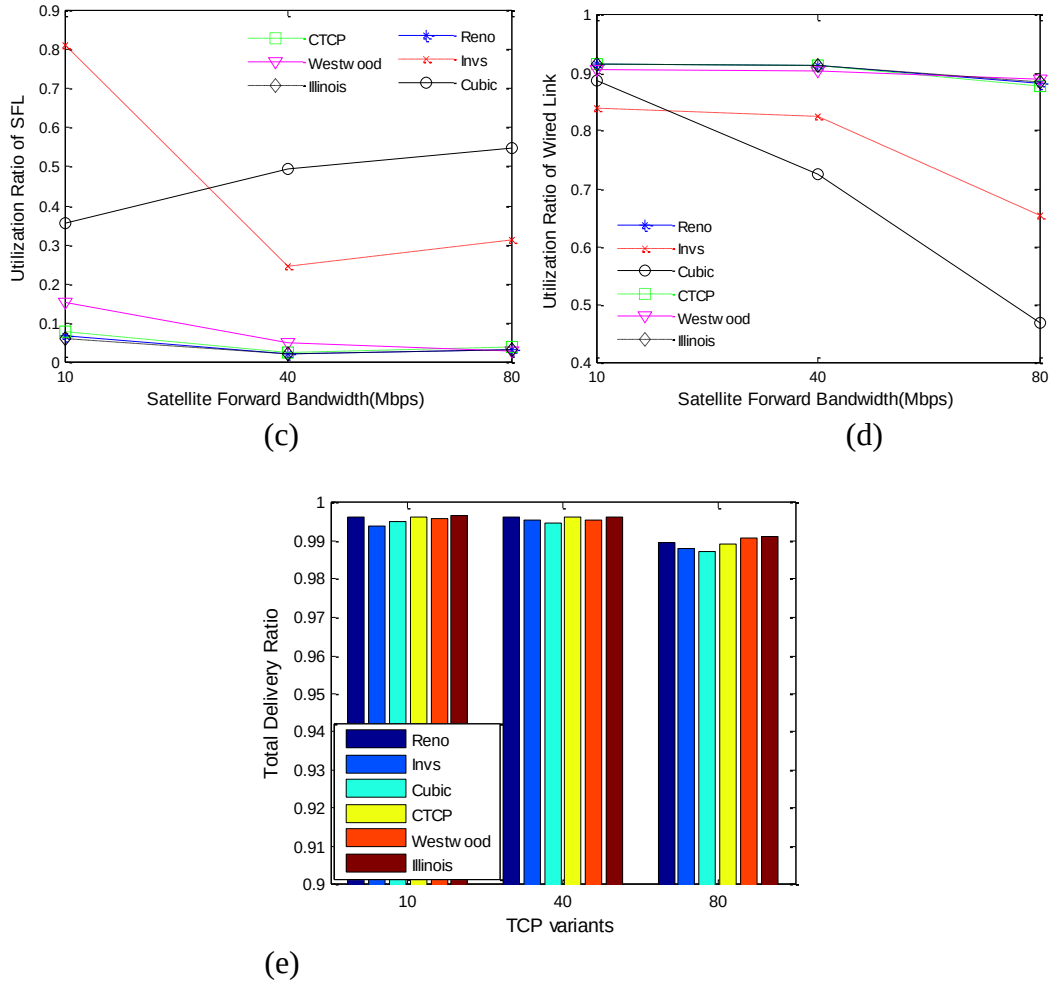


Figure 10. Hybrid satellite and Wired Networks Evaluation. (a) Throughput Ratio. (b) Fairness Index. (c) Utilization Ratio of SFL. (d) Utilization Ratio of Wired Link. (e) Total Delivery Ratio

4.3. RTT Fairness

In this part, we evaluate the RTT fairness of Invs with the dumbbell topology. The bottleneck bandwidth is 100 Mbps and the side bandwidth is 1000 Mbps. The bottleneck buffer is set to 250packets (3 Mbit) and the maximum segment size (MSS) is set to 1500byte. Each simulation lasts 600s and the results are obtained by averaging the throughput over the running time. Two flows that use the same protocol share the bottleneck. One flow (flow 1) has a fixed RTT of 80 ms while the RTT of the other flow (flow 2) varies from 20 ms to 320 ms. Flow 1 starts at 1 s and flow 2 starts at 20 s later.

Figure 11 shows fairness evaluation results. The throughput ratio represents the ratio of the higher throughput of the two flows to the lower one. In Figure 11, Reno, CTCP, Illinois, Westwood and Invs achieve good throughput ratios when the RTT of flow 2 is 80 ms and 160 ms. However, Reno and Illinois suffer the worst throughput ratios when the RTT of the flow 2 decreases below 80 ms. Invs and Westwood achieve the best throughput ratios when the RTT of the flow 2 is 40 ms and 20 ms. When the RTT of the flow 2 is 320 ms, Invs achieves the best throughput ratio.

In fact, Reno flows is designed to be fair when two flows share the same path (the same RTT). On the other hand, Reno flows cannot achieve fair share of bottleneck bandwidth when the RTT of the flows are different. CTCP improves the RTT-fairness through aggressive growth function which shorts the cwnd increase duration to the state that the bandwidth is fully used after back-off for flows with long delay. Westwood improves the RTT fairness of Reno by setting *cwnd* to the estimated value when packet loss occurs. The back-off scheme of Westwood increases the capability of long-delay flow in full utilization of the available bandwidth after packet loss. In all, Reno and CTCP have very good RTT fairness performance when the RTT of flow 2 is 80 ms and 160 ms. Invs has the best RTT fairness when the RTT of flow2 is 20 ms, 40 ms and 320 ms. Overall, Invs achieves very good RTT fairness, especially when the RTT difference of the two flows increases.

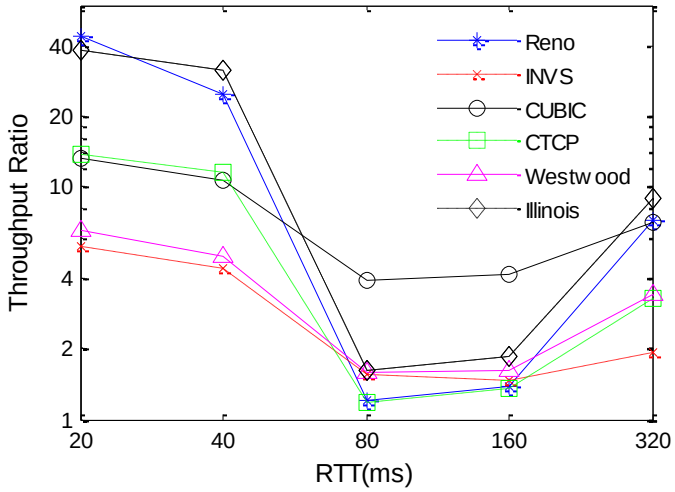


Figure 11. RTT Fairness

4.4. Invs Performance in Mixed Wired/Wireless Networks

Wireless networks plays a significant role in heterogeneous networks. The performance of Invs is tested under mixed wired and wireless networks. Since wireless link is characterized with high error rate, a simple random error model is added to the SFL in the hybrid topology. The SFL is 4 Mbps. Each simulation lasts 500s.

Figure 12 shows the goodputs of CTCP, Cubic, Illinois, Invs, Reno and Westwood under long-delay and lossy network environments. The results show that Reno has the lowest goodputs when the packet loss rate is less than 10^{-3}. The goodputs of Reno approximates the goodputs of CTCP and Illinois when the packet loss rate is greater than 10^{-3}. This is attributed to that CTCP and Illinois adopt aggressive increase function of *cwnd*. However, these improvements decay quickly when the packet loss rate is beyond 10^{-4}. The results also show that the goodputs of Cubic, Invs and Westwood are close when the packet loss rate is less than 10^{-4}. Invs achieves the best goodputs than Cubic and Westwood when the packet loss rate is 10^{-3}. Cubic adopts very aggressive *cwnd* increase scheme and a new back-off scheme to achieve high goodput in high BDP networks. Westwood uses a delay-based back-off scheme when packet loss occurs. Invs adopt the loss classification scheme and an adaptive increase function of *cwnd*. Therefore, Invs achieves the best performance in mixed wired and wireless networks. In addition, the goodputs of all the protocols seem to tend towards the same as the packet loss rate is high enough (>0.01).

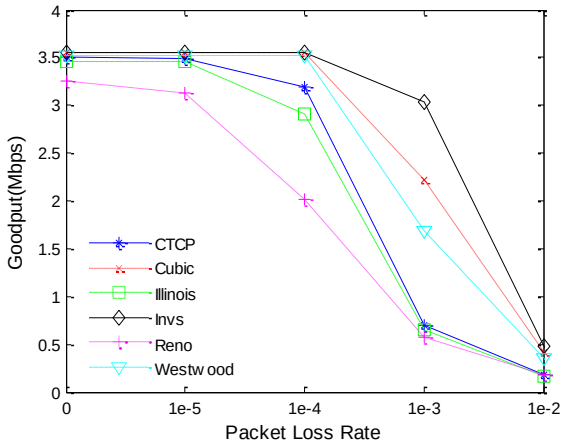


Figure 12. Throughput versus Packet Loss Rate.

5. Conclusion

To cope with the heterogeneity in Internet Congestion Control, we proposed a new TCP protocol, called Invs. Invs employs a novel increase function of the congestion window with an adaptive increase factor. The increase factor adjusts through estimation of the path BDP and the minimum RTT. Thus Invs can modulate the window growth rate to fit with the path condition. To improve the performance of TCP with wireless links or mobile node, Invs adopts a novel scheme that adaptively adjusts the queue threshold in loss classification to distinguish non-congestion loss from congestion loss. Furthermore, Invs introduces a new back-off scheme to enable quick convergence of congestion window. Finally, throughput model, illustrative analysis are developed and extensive simulations have been run to evaluate the fairness, friendliness, RTT-fairness and the efficiency of Invs.

Our analysis show that Invs achieves good fairness. The simulation results under traditional environment show that Invs has comparable performance with other TCP protocols. The simulation results under high BDP networks show that Invs achieves a compromised performance between aggressiveness and fairness with the satellite links. The RTT fairness evaluation results show that Invs achieves very good RTT fairness, especially when the RTT difference of the two flows increases. The simulation results under mixed wired/wireless networks show that Invs is capable to distinguish non-congestion loss from congestion loss and achieves the best throughputs in mixed wired/wireless networks. Overall, Invs not only performs well as other TCP variants in wired environments, but also achieves better fairness, RTT fairness and utilization than other TCP variants in heterogeneous networks. In addition, Invs preserves the end-to-end semantics and only need modification on TCP sender.

Invs distinguishes non-congestion loss from congestion losses. However, in wireless networks, non-congestion losses may be attributed to some other causes (mobile handoff, link failure) other than random losses. These causes of loss need to be further researched and actions upon these causes of loss need to be developed.

Acknowledgements

This work is sponsored by the National Natural Science Foundation of China under grant No. 61171089 and No.61302054, and the Training Program of the Major Research Plan of the National Natural Science Foundation of China under grant No. 91438104.

References

[1] A. Damnjanovic, J. Montojo, W. Yongbin, J. Tingfang, L. Tao, M. Vajapeyam, Y. Taesang, S. Osok and D. Malladi, "A survey on 3GPP heterogeneous networks", Wireless Communications, IEEE, vol. 18, no. 3, **(2011)**, pp. 10-21.
[2] D. Papadimitriou, M. Welzl, M. Scharf and B. Briscoe, "Open research issues in Internet congestion control", Internet Research Task Force (IRTF), **(2011)**.
[3] A. Afanasyev, N. Tilley, P. Reiher and L. Kleinrock, "Host-to-Host Congestion Control for TCP", Communications Surveys & Tutorials, IEEE, vol. 12, no. 3, **(2010)**, pp. 304-342.
[4] C. Caini and R. Firrincieli, "TCP Hybla: a TCP enhancement for heterogeneous networks", International journal of satellite communications and networking, vol. 22, no. 5, **(2004)**, pp. 547-566.
[5] K.T.J. Song, M. Sridharan and C.-Y. Ho, "CTCP: Improving TCP-Friendliness Over Low-Buffered Network Links", Proc. Proc. 6th Int. Workshop on Protocols for FAST Long-Distance Networks (PFLDnet 2008), **(2008)**.
[6] S. Ha, I. Rhee and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant", ACM SIGOPS Operating Systems Review, vol. 42, no. 5, **(2008)**, pp. 64-74.
[7] S. Liu, T. Başar and R. Srikant, "TCP-Illinois: A loss- and delay-based congestion control algorithm for high-speed networks", Performance Evaluation, vol. 65, no. 6–7, **(2008)**, pp. 417-440
[8] M. Park, M. Shin, D. Oh, D. Ahn, B. Kim and J. Lee, "ER-TCP (exponential recovery-TCP): High-performance TCP for satellite networks", IEICE Transactions on Communications, vol. E95-B, no. 5, **(2012)**, pp. 1679-1688.
[9] D. Ros and M. Welzl, "Less-than-Best-Effort Service: A Survey of End-to-End Approaches, Communications Surveys & Tutorials", IEEE, vol. 15, no. 2, **(2013)**, pp. 898-908.
[10] K.-C. Leung and V.O. Li, "Transmission control protocol (TCP) in wireless networks: issues", approaches, and challenges, IEEE Communications Surveys and Tutorials, **(2006)**.
[11] J.K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher and J. Barros," Network Coding Meets TCP: Theory and Implementation", Proceedings of the IEEE, vol. 99, no. 3, **(2011)**, pp. 490-512.
[12] M. A. Alrshah, M. Othman and B. Ali, "Comparative study of high-speed Linux TCP variants over high-BDP networks", Journal of Network and Computer Applications, vol. 43, **(2014)**, pp. 66-75.
[13] C. Casetti, M. Gerla, S. Mascolo, M. Sanadidi and R. Wang, "TCP Westwood: end-to-end congestion control for wired/wireless networks", Wireless Networks, vol. 8, no. 5, **(2002)**, pp. 467-479.
[14] J.F. Kurose and K.W. Ross, "Computer Networking: A Top-Down Approach", Pearson Education, Limited, **(2010)**.
[15] Y. Peng, L. Wen, X. Lisong, J. Deogun and Y. Lu, "TCP Congestion Avoidance Algorithm Identification", Book TCP Congestion Avoidance Algorithm Identification, Series TCP Congestion Avoidance Algorithm Identification, **(2011)**, pp. 310-321.

Authors

Zhiming Wang, He was born in 1986. He received the B.E degree in Electronic Information Engineering from the Chongqing University, Chongqing, China, in 2009. He is a Ph.D. degree candidate with the College of Communication Engineering, Chongqing University. His research interests include TCP modeling, congestion control, and aeronautical Ad hoc networks.