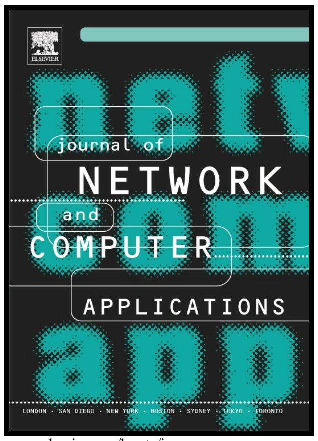
Author's Accepted Manuscript


journal of
NETWORK
and
COMPUTER
APPLICATIONS
LONDON • SAN DIEGO • NEW YORK • BOSTON • SYDNEY • TOKYO • TORONTO

www.elsevier.com/locate/jnca

TCP Congestion Control Algorithm for Heterogeneous Internet

Zhiming Wang, Xiaoping Zeng, Xue Liu, Man Xu, Ya Wen, Li Chen

PII: S1084-8045(16)30032-7
DOI: http://dx.doi.org/10.1016/j.jnca.2016.03.018
Reference: YJNCA1608

To appear in: *Journal of Network and Computer Applications*

Received date: 29 August 2015
Revised date: 28 October 2015
Accepted date: 28 March 2016

Cite this article as: Zhiming Wang, Xiaoping Zeng, Xue Liu, Man Xu, Ya Wen and Li Chen, TCP Congestion Control Algorithm for Heterogeneous Internet *Journal of Network and Computer Applications* http://dx.doi.org/10.1016/j.jnca.2016.03.018

This is a PDF file of an unedited manuscript that has been accepted fo publication. As a service to our customers we are providing this early version o the manuscript. The manuscript will undergo copyediting, typesetting, an review of the resulting galley proof before it is published in its final citable forn Please note that during the production process errors may be discovered whic could affect the content, and all legal disclaimers that apply to the journal pertain

ACCEPTED MANUSCRIPT

TCP Congestion Control Algorithm for Heterogeneous Internet

Zhiming Wang[*], Xiaoping Zeng, Xue Liu, Man Xu, Ya Wen, Li Chen

College of Communication Engineering, Chongqing University, Chongqing 400030, China.

Abstract

The available bandwidth, round trip time (RTT) and packet loss rate can vary over many orders of magnitude, which characterizes the heterogeneity of the Internet. To cope with the heterogeneity in Internet congestion control, we propose a new TCP protocol known as INVS that contains three key components: 1) INVS employs an exponential-function-based growth function of the congestion window, which is more efficient than the cubic function in CUBIC; 2) INVS introduces an adaptive increase factor into the growth function to ensure that the window growth rate matches the path condition, and this increase factor measures the path condition using a custom function of the available bandwidth and minimum RTT; 3) INVS adopts an adaptive queue threshold in the loss classification scheme to improve the performance of TCP over lossy links. In addition, INVS requires modification only on the TCP sender and traces the path capacity to enable quick convergence of the congestion window. The performance analysis and evaluation show that INVS achieves good throughput, fairness, RTT-fairness and utilization in heterogeneous networks.

Keywords:
Transmission control protocol (TCP), congestion control, fairness, bandwidth delay product (BDP), heterogeneity

Accepted manuscript

[*]Corresponding author.

Email addresses: jamewzm@163.com (Zhiming Wang), zxp@cqu.edu.cn (Xiaoping Zeng), maojiu2005@qq.com (Xue Liu), 323109402@qq.com (Man Xu), 327704920@qq.com (Ya Wen), chenli@cqu.edu.cn (Li Chen)

ACCEPTED MANUSCRIPT

1. Introduction

The traditional communication network has evolved into a seamless global Internet that encompasses a variety of heterogeneous IP networks (wired, wireless and satellite networks) [1, 2]. The Internet Congestion Control Research Group has outlined a variety of distinguishing link and path characteristics for the Internet [3]: (1) the available bandwidth can be either scarce over radio links or abundant over high-speed optical links; (2) the round trip time (RTT) ranges from much less than a millisecond (local interconnects) to notably large, i.e., equal or grater than 500 milliseconds (satellite links); (3) the packet loss rate (PLR) ranges from notably low over optical fiber links (less than 10^{-6}) to notably high over certain wireless links (higher than 1%). Consequently, the available bandwidth, RTT and PLR of the Internet can vary over many orders of magnitude, which characterizes the heterogeneity of the Internet. The Transmission Control Protocol (TCP), which provides reliable end-to-end data delivery across the Internet, must cope with the different transmission media that travel as Internet traffic over the best-effort IP networks [4]. With the increasing heterogeneity of the Internet, this mission of TCP has become more difficult, and the heterogeneity of the Internet has been identified as one of the global challenges in Internet Congestion Control [3].

Many publications have addressed congestion control algorithm in terms of the heterogeneity of the Internet. Most of the literature focuses on one aspect of the heterogeneity or need for network aid and can be grouped into several classifications.

First, Hybla [5] is proposed for satisfying RTT-fairness to cope with RTT difference, which is often considered as an RTT-unfairness problem. However, when passing through lossy links in which the packet loss is no longer the indication of congestion, Hybla flows back off upon packet loss, which is often caused by link error. As a result, Hybla flows cannot use the available bandwidth effectively, which is treated as performance degradation [6].

Second, HS-TCP [6], CUBIC [7], Compound TCP (CTCP) [8], ER-TCP [9] and Illinois [10] have been proposed to cope with the difference of bandwidth and RTT, especially the poor bandwidth utilization of traditional TCP in high-speed networks and long-delay networks [11], and CUBIC and CTCP have been widely deployed in the Internet. However, these proposals are not compatible with lossy wireless links and also suffer from unfairness problems [12, 13]. For example, when sharing bandwidth with TCP Reno or CTCP,

Accepted manuscript

ACCEPTED MANUSCRIPT

CUBIC, STCP and HS-TCP (which employ highly aggressive growth functions of the congestion window regardless of high-speed and long-delay networks or traditional networks) take up most of the total bandwidth, which is treated as a fairness problem [12]. Due to the high random packet loss over wireless links, especially for third generation (3G) communication links, mobile devices (smartphones, tablets, etc., most of which use CUBIC as the default TCP congestion control algorithm) suffer from significant TCP performance degradation [14, 15].

To cope with the PLR difference, especially the performance degradation over lossy links, many proposals attempt to distinguish non-congestion losses from congestion losses by comparing the queue delay or queue length with a fixed threshold and apply different multiplicative decrease schemes accordingly [6, 12, 13, 16–20], e.g., Veno [16], TCP Westwood [17], etc. With the increasing variety of wireless networks, the accuracy of the fixed threshold approaches in loss classification are decreasing. Certain other proposals use particular congestion control algorithms. PEPsal [18] and indirect-TCP [19] split the connection into several portions to shield wireless links from wired links. TCP-Jersey [6] adopts a router-aided explicit congestion warning scheme to determine the cause of the loss. Network Coded TCP [20] incorporates network coding into TCP to provide significant throughput gains in lossy networks. However, these particular algorithms were not widely accepted and deployed due to their special requirements.

Finally, to cope with differences in bandwidth, RTT and PLR simultaneously, the authors in [21] proposed a congestion control scheme based on RTT, and PLR, but taking PLR as the congestion degree is not suitable for paths with lossy links. In [22], a link-type-based congestion control scheme is proposed that uses Vegas over satellite links, Westwood over wireless links, HS-TCP over high-speed links and NewReno over traditional links. Due to the demand of adding link type information in IP packets by routers, this method could not be deployed in the Internet. In [23], TCP BRJ grades the congestion into five grades based on delay jitter to differentiate random losses from congestion losses. Because the delay jitter in the Internet varies widely, the reliability of grading according to the fixed delay jitter threshold must be further investigated.

As the diversity of communication links increases, the TCP congestion control algorithm must match the path condition for each connection. Although TCP has been extensively studied in recent past years, most of the proposals focus on particular environments, and the proposals for heteroge-

Accepted manuscript

ACCEPTED MANUSCRIPT

neous networks cannot be deployed in or not suitable for the heterogeneous Internet. Therefore, in this paper, we propose a new TCP congestion control algorithm, i.e., INVS, to cope with the multiple differences that exist in the heterogeneous Internet. This method incorporates the merits of CUBIC and Westwood and distinguishes itself from other proposals in three key aspects: (1) It employs an exponential-function-based growth function for the congestion window that is more efficient than the cubic function in CUBIC; (2) It introduces an adaptive increase factor in the growth function to scale the window growth rate to match the path condition. This increase factor measures the path condition using a custom function of the available bandwidth and the minimum RTT in which a parameter γ is also introduced to reflect the conservation of RTT influence; 3) It adopts an adaptive queue threshold in the loss classification scheme to improve the performance of TCP over lossy links. In addition, INVS requires modification of TCP sender only and traces the path capacity to enable quick convergence of the congestion window.

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

2. Proposed Congestion Control Algorithm: INVS

To cope with the heterogeneity of the Internet, we propose an exponential-function-based congestion window growth function to ensure efficiency, introduce an adaptive increase factor to match the window growth rate with the path condition, and adopt an adaptive queue-threshold scheme to improve the performance of loss classification.

2.1. Growth Function of the Congestion Window

For efficiency in large BDP networks, INVS uses an exponential-function-based convex window growth function at the front of each congestion avoidance (CA) phase and a concave function for exploring available resources. With the convex function, the TCP sender can increase the congestion window quickly at the beginning of the CA phase to fully utilize the available bandwidth and provide slow increases thereafter. The size of the congestion

Accepted manuscript

ACCEPTED MANUSCRIPT

window at time t in INVS at the front of each CA phase is given as follows.

$$cwnd\,(t) = cwnd_{sp}\left(1 - (1-\beta)\,\alpha^{t}\right), 0 < \alpha < 1 \tag{1}$$

where t is the time from the beginning of the congestion avoidance phase, $cwnd_{sp}$ is the congestion window at *saturationpoint* (which represents the state in which the available resources of the path is fully used), β is the multiplication decrease factor, and α is a parameter that controls the window growth rate.

INVS uses the convex growth and the concave probing through a congestion window update scheme as shown in formula (2). Fig. 1 shows the exponential-function-based congestion window growth functions and the cubic function of CUBIC. INVS1, INVS2 and INVS3 are the exponential-function-based growth functions when parameter k increases. Comparing INVS1 with CUBIC, we note that the exponential-function-based growth function is more efficient when the duration of the CA phase is the same. The results also show that the duration of the CA phase increases as k increases, and *cwnd* remains high most of the time in a CA phase. By setting a proper parameter k, the exponential growth function could have a moderate increase rate without sending long bursts at the front of each CA phase while ensuring the efficiency. Therefore, k is chosen to be adaptive according to the path condition.

$$cwnd{+} = \begin{cases} \dfrac{cwnd_{sp} - cwnd}{k \cdot cwnd_{\text{sp}}}, & cwnd \le cwnd_{sp} - 1 \\[2ex] \dfrac{cwnd - cwnd_{sp}}{k \cdot cwnd}, & cwnd \ge cwnd_{sp} + 1 \\[2ex] \dfrac{1}{cwnd}, & cwnd_{sp} - 1 < cwnd < cwnd_{sp} + 1 \end{cases} \tag{2}$$

where *cwnd* is the size of congestion window, and k is an adaptive increase factor that relates to α as $\alpha = \frac{\beta(k-\beta)}{k}$.

The first expression in (2) realizes the exponential growth in a CA phase, as in (1). The second expression in (2) attempts to probe additional resources, and the last expression in (2) addresses the transition. In the implementation, *cwnd* is defined as an integer, and the last expression ensures that *cwnd* can reach and surpass $cwnd_{sp}$. According to formula (2), *cwnd* increases quickly at the beginning of a new CA phase and slows when *cwnd*

Accepted manuscript

ACCEPTED MANUSCRIPT

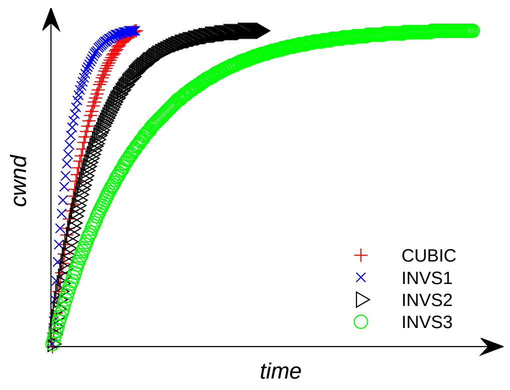


Figure 1: Convex growth function

approaches $saturationpoint$. When $cwnd$ grows beyond $saturationpoint$, it slowly accelerates to probe the available resources of the path. The region surrounding $cwnd_{sp}$ indicates the full use of network bandwidth.

Formula (2) ensures the efficiency of TCP under a given path condition. In the heterogeneous Internet, TCP traffic might encounter high-speed fiber links, low-speed wireless links, or even long-delay satellite links. The increase of RTT or bandwidth results in the increase of the path BDP, which indicates that more packets are allowed in-flight to fill the path pipe. To fill the path pipe efficiently, the increment of $cwnd$ per ACK should be increased as the path BDP increases. For efficiency in large BDP networks and fairness with other TCP variants (especially Reno) in traditional networks, k is designed to decrease as the path BDP increases. In INVS, k is expressed as shown in formula (3). The logarithm function is used to map the infinite bandwidth and RTT to a relative small range. If growth rate of $cwnd$ were sufficiently large, k would not decrease further as the bandwidth or the RTT increases beyond the reference value.

$$k = c\left(\log_2\left(r_{bw} r_{rtt}^{\gamma}\right) + 1\right), 0 < \gamma < 1 \tag{3}$$

where $r_{bw} = \max\left[\frac{BW_{ref}}{BW_{est}}, 1\right], r_{rtt} = \max\left[\frac{RTT_{ref}}{RTT_{\min}}, 1\right]$, c is a constant scale factor (2 by default), γ is the power factor that reflects the conversation of RTT influence, BW_{ref} is the reference bandwidth with a default value of 500 Mbps, BW_{est} is the estimated bandwidth, RTT_{ref} is the reference RTT with a default value of 0.5 s and RTT_{min} is the minimum RTT. When $BW_{est} > BW_{ref}$ and $RTT_{min} > RTT_{ref}$, k is equal to c.

Accepted manuscript

ACCEPTED MANUSCRIPT

In addition, the increase of RTT also results in the increase of the delay required to receive network feedback. Delayed congestion feedback might lead to additional packet losses when congestion occurs. If this drawback is ignored when the RTT increases, congestion can occur frequently and additional packets might be lost in each congestion event. Therefore, INVS introduces a power factor γ in determination of the increase factor k to take the conservation of this RTT influence into account. The larger the factor γ, the more negative the RTT influence is considered. In INVS, γ is 0.75 by default.

2.2. Action Upon Packet Loss and Loss Classification

Upon receipt of triple duplicate ACKs, INVS determines a new *cwnd* and new *ssthresh* as follows.

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

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

In the heterogeneous Internet, packet loss is no longer an indication of congestion. The 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 of the path can vary over a large magnitude, and a fixed threshold could lead to misjudgment. INVS uses the maximum of δ and *maxbuffer* as the new queue threshold to differentiate non-congestion loss and congestion loss. The incorporation of *maxbuffer* makes the loss classification scheme scalable for variety of buffer conditions, and the limitation by δ avoids misjudgment upon rerouting. In INVS, δ is 5 packets by default.

As in formula (4), upon receipt of the third duplicate ACK, INVS takes on different schemes depending on whether the estimated queue length is greater than the minimum value of δ and *maxbuffer*. If the current queue length has exceeded the threshold, the loss is considered as 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*.

Accepted manuscript

ACCEPTED MANUSCRIPT

Furthermore, INVS maintains $cwnd_{sp}$ to estimate $cwnd$ when the network resources are fully used. The available network resources change due to the start of new flows and termination of transmitting flows. When packet loss occurs, $cwnd < cwnd_{sp}$ shows that congestion occurs earlier in this CA phase, which indicates that the available network resources are diminishing. To enable fast convergence of competing flows, $cwnd_{sp}$ is set to a value less than $cwnd$ as in formula (5). If $cwnd \geq cwnd_{sp}$, $cwnd_{sp}$ is set to $cwnd$ to track the new $saturationpoint$.

$$cwnd_{sp} = \begin{cases} \frac{1+\beta}{2} \cdot cwnd, \; cwnd < cwnd_{sp} \\ \\ cwnd \,, \qquad else \end{cases} \tag{5}$$

2.3. Parameter Estimation

2.3.1. Estimation of Available Bandwidth

INVS uses the available bandwidth in the calculation of the increase factor k and the path BDP, which is used in resetting *ssthresh* and *cwnd* after packet loss. Because 1) the available bandwidth does not always change widely for each connection; 2) the factor k and path BDP do not rely on instant bandwidth, the available bandwidth is estimated once every RTT instead of every ACK to avoid estimating the available bandwidth too frequently.

Algorithm 1 presents the pseudo-code of the available bandwidth estimation (ABE) algorithm for each ACK. In algorithm 1, now and *measure_start* record the kernel time of the algorithm execution and the start time of the measurement, respectively, and *ack_seq* and *snd_una* record the sequence number of packets cumulatively acknowledged (ACKed) and unacknowledged, respectively. *RTT* is the round trip time estimated from the latest ACK. *INVS_RTT_MIN* is the minimum estimation duration and is set to $HZ/50$, which is equal to 20 ms. Line 1 calculates the time from the start of this estimation epoch. Lines 2-4 count the accumulated data that have been ACKed in this epoch. Line 5 checks whether the elapsed time is at least a RTT or a minimum duration from the start of this epoch. A minimum duration prevents large variation of the estimated available bandwidth if the RTT is quite small. Line 6 realizes ABE with a low-pass filter, which can provide a robust estimation of the average bandwidth.

Accepted manuscript

ACCEPTED MANUSCRIPT

Algorithm 1: Pseudocode of the ABE algorithm upon ACK.

```
1 $delta = now - measure\_start$;
2 if $data\_ACKed$ then
3     $data\_acked\_this\_epoch += ack\_seq - snd\_una$;
4 end
5 if $((RTT > 0) and (delta > max(RTT, INVS\_RTT\_MIN))$ then
6     $BW_{est} = a_1 * BW_{est} + a_2 * data\_acked\_this\_epoch/delta$;
7     $data\_acked\_this\_epoch = 0$;
8     $measure\_start = now$;
9 end
```

2.3.2. Tracking of the Minimum RTT

In practice, the minimum RTT is updated only when the new RTT is less. The minimum RTT only changes when the network condition or flow route changes. When the minimum RTT of the connection increases due to rerouting, the minimum RTT update scheme will fail. To accurately track the minimum RTT when the route changes, the minimum RTT is re-initiated when the difference between RTT and the minimum RTT is larger than a threshold (2 times the minimum RTT by default).

2.3.3. Estimation of the Queue Length and maxbuffer

The queue length estimates the number of backlog packets along the path, and *maxbuffer* estimates the maximum number of packets that can be queued along the path. To distinguish non-congestion loss from congestion loss, the queue length is compared with the threshold in formula (4) upon packet loss. The queue length is estimated as follows.

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

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

Accepted manuscript

3. Performance Analysis

3.1. Steady State Throughput of INVS

Given a certain path, the increase factor k is deemed constant. In steady state, *cwnd* of INVS increases from $\beta cwnd_{sp}$ to $cwnd_{sp}$ periodically. From

ACCEPTED MANUSCRIPT

formula (2), the size of congestion window at the beginning of i-th round can be obtained as follows.

$$cwnd\left(i\right) = cwnd_{sp}\left(1 - \left(\frac{\beta\left(k-\beta\right)}{k}\right)^{i-1}\left(1-\beta\right)\right) \tag{7}$$

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

Let n be the number of RTT rounds for $cwnd$ to reach $cwnd_{sp}$ from the beginning of a CA phase. According to formula (2), $cwnd$ is equal to $cwnd_{sp}$ at the end of the i-th round when $cwnd(i) \geqslant cwnd_{sp} - 1$. From formula (7), we note that the number of packets increased in each RTT obeys an exponential function. Therefore, we can obtain n as follows.

$$n = \log_{\frac{k}{\beta(k-\beta)}}\left(cwnd_{sp}\left(1-\beta\right)\right) \tag{8}$$

The total number of packets in the CA phase can be obtained.

$$\begin{aligned} Y &= \sum\nolimits_{i=1}^{n} cwnd\left(i\right) \\ &= cwnd_{sp}\left(n - \frac{k\left(\left(1-\beta\right)cwnd_{sp}-1\right)}{\left(k-k\beta+\beta^2\right)cwnd_{sp}}\right) \end{aligned} \tag{9}$$

In the steady state, the total number of packets in the CA phase also can be expressed by the congestion event rate p.

$$Y = \frac{1}{p} \tag{10}$$

By equating the right-hand sides of formulas (9) and (10), we obtain

$$cwnd_{sp}\left(n - \frac{k\left(\left(1-\beta\right)cwnd_{sp}-1\right)}{\left(k-k\beta+\beta^2\right)cwnd_{sp}}\right) = \frac{1}{p} \tag{11}$$

Additionally, we obtain

$$cwnd_{sp} = \frac{k\left(1-\beta-p\right)+\beta^2}{p\left(\left(k-k\beta+\beta^2\right)n - k\left(1-\beta\right)\right)} \tag{12}$$

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

$$cwnd_{sp} = \frac{\left(k\left(1-\beta-p\right)+\beta^2\right)\ln\left(\frac{k}{\beta(k-\beta)}\right)}{p\left(k-k\beta+\beta^2\right)lambertW\left(f\right)} \tag{13}$$

Accepted manuscript

ACCEPTED MANUSCRIPT

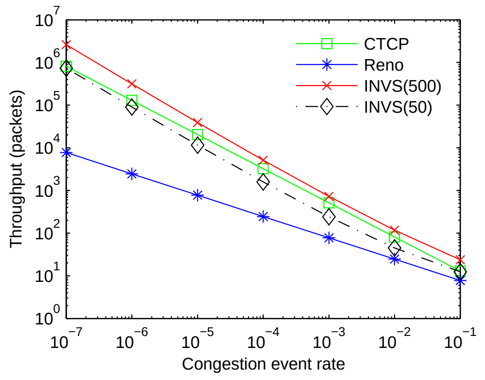


Figure 2: Response functions of various protocols

where

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

Therefore, the average throughput can be expressed as follows.

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

Fig. 2 shows the response function of INVS under various loss rate. For comparison, the response functions of CTCP and Reno are also given. It is regretable that we have not found proper response functions for CUBIC, Illinois, Hybla and Westwood, which are compared with INVS in Section 4. INVS(500) and INVS(50) represent the throughputs under bandwidths of 500 Mbps and 50 Mbps, respectively. From the Figure, we observe that INVS achieves higher throughputs than Reno and CTCP in large bandwidth networks (bandwidth=500 Mbps) and moderate throughputs if the bandwidth is 50 Mbps.

3.2. TCP Friendliness and Fairness

TCP fairness defines the equality in shared bandwidth among flows that experience equivalent path congestion [3]. TCP friendliness defines the fairness of bandwidth sharing between a given flow and other flows using different TCP congestion control algorithms [7, 8].

Accepted manuscript

ACCEPTED MANUSCRIPT

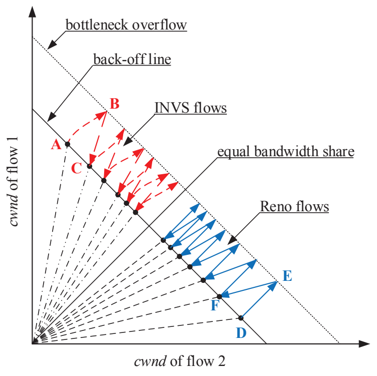


Figure 3: Convergence toward fair bandwidth share

3.2.1. TCP Fairness

To demonstrate the fairness, an informal vector-based illustration is presented in Fig. 3, which is similar to that in [24]. The red dashed curve illustrates the *cwnd* convergence process of two INVS flows sharing the same path.

Suppose flow 1 begins first, flow 2 begins later and flow 1 has attained a higher *cwnd* than flow 2. Congestion occurs, and both flows back off. Flow 1 sets its $cwnd_{sp}$ to $0.5(1+\beta)cwnd$ according to the first expression in formula (5), and flow 2 sets its $cwnd_{sp}$ to the maximum *cwnd*. Flow 1 and flow 2 begin a new CA phase at point A. The increase rates of both flows would not necessarily be equal even though they use nearly the same amount of time to increase *cwnd* to $cwnd_{sp}$, according to formula (2). Because flow 1 uses a lower $cwnd_{sp}$, both flow 1 and flow 2 can reach *saturationpoint* before congestion in this CA phase. As a result, flow 1 releases an amount of bandwidth to flow 2 before congestion occurs at point B. Both flows back off their congestion windows to point C. The abovementioned process repeats, and flow 1 gradually releases bandwidth to flow 2 until an equal bandwidth share is reached.

For comparison, the blue curve illustrates the same process for Reno flows. Because Reno uses a linear increase function, the *cwnd*s of both flows have the same increment in each CA phase. In addition, it is obvious that the multiplicative decrease factor β is significant to the convergence rate of both INVS and Reno. The larger β, the quicker the convergence will be. Because β is also important to the utilization, in INVS, β equals 0.75, which is the

Accepted manuscript

ACCEPTED MANUSCRIPT

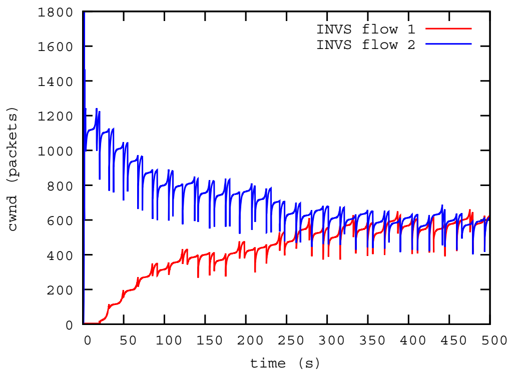


Figure 4: Congestion windows of 2 competing INVS flows

same as in CUBIC.

Fig. 4 shows the congestion windows of two INVS flows that share the same path and is obtained by running NS2 simulation over a dumbbell network with bottleneck capacity of 100 Mbps and RTT of 80 ms. The results show that the congestion windows of the two INVS flows converge near the same value.

3.2.2. TCP Friendliness

INVS is friendly to TCP Reno in small BDP networks. In small BDP networks, the factor k could be notably large, and as a result, the congestion window increase rate of INVS is close to or even less than that of TCP Reno. When the *cwnd* of INVS is lower than that of TCP Reno, INVS uses the *cwnd* of TCP Reno. The congestion window of TCP Reno is obtained as follows[7]

$$W_{TCP}\left(t\right) = 3\frac{1+\beta}{1-\beta}\frac{t}{RTT} + \beta cwnd_{sp} \tag{15}$$

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

Fig. 5 shows the congestion windows of an INVS flow and a Reno flow sharing the same bottleneck over a small BDP dumbbell network. The results show that INVS is fair with Reno in the small BDP network. Fig. 6 shows the congestion windows of an INVS flow and a Reno flow over a high BDP dumbbell network with the bottleneck bandwidth of 100 Mbps and RTT of 100 ms. Comparing with the result when the two flows are all of Reno (Fair share), Fig. 6 shows that INVS flow deprives some bandwidth from

Accepted manuscript

ACCEPTED MANUSCRIPT

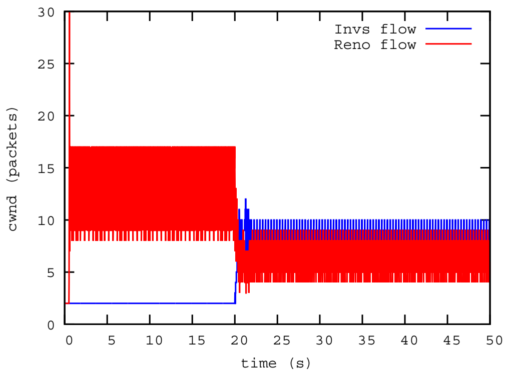


Figure 5: Congestion windows of INVS and Reno in a small BDP network (10 Mbps bottleneck BW and 10 ms RTT)

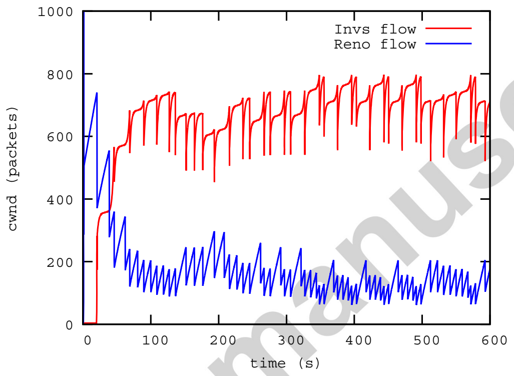


Figure 6: Congestion windows of INVS and Reno in a high BDP network

Reno flow. However, after evaluating the total utilization, we find that using INVS improves the utilization from 75% to more than 90%. This reveals the contradiction between fairness and utilization in high BDP network.

4. Performance Evaluation

This section presents the NS2 evaluation results of INVS. Two topologies, namely, dumbbell and hybrid, are used. The dumbbell topology is configured with a side bandwidth (BW) of 1000 Mbps. The delay of links and the bottleneck BW might vary in each scenario. The hybrid topology and configuration are shown in Fig. 7. The queue or buffer sizes in all of the simulations (without specification) are set to the path BDP by default. Fur-

Accepted manuscript

ACCEPTED MANUSCRIPT

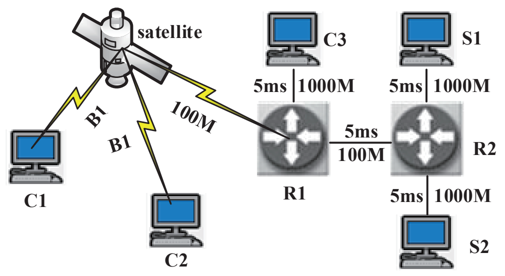


Figure 7: Hybrid topology

thermore, to emulate realistic Internet background traffic, background TCP flows are used, which consist of 20% Reno flows, 50% CUBIC (BIC is ignored in this case) flows and 30% CTCP flows. The ratio of background flows is chosen with reference to the statistical percentage range of the TCP algorithm deployed in the Internet in [25] and considers the use of Windows 7 and its later versions, which use CTCP as the default TCP, as well as smart phones (most of which are based on Linux kernel with CUBIC as the default TCP algorithm). In addition, the Jain's fairness index is used to evaluate fairness as follows.

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

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

4.1. Single Flow Scenario

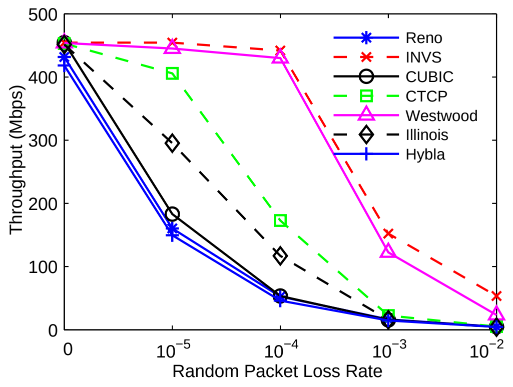
To evaluate the performance of INVS over lossy links, we simulate and measure the goodputs[1] of a single flow under two topologies. The dumbbell topology is used to emulate high-speed networks. The bottleneck bandwidth is 500 Mbps, the RTT is 20 ms and the buffer size of bottleneck link is 4 Mb. The hybrid topology, as shown in Fig. 7, is used to emulate long-delay (satellite) networks, in which the BW of satellite forward link (B1) is 4 Mbps.

Fig. 8a and Fig. 8b show the average goodputs of a single flow under high-speed networks and long-delay networks, respectively. From inspection of the figures, the following observations can be made:

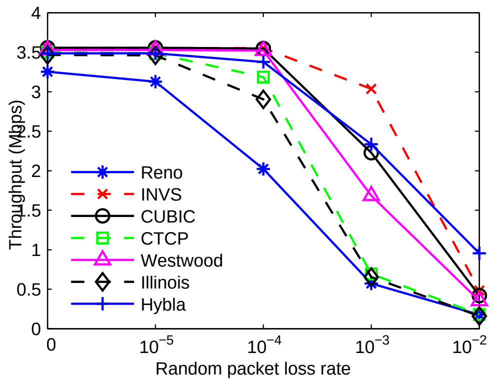
[1]Goodput denotes the throughput of successfully delivered packets.

Accepted manuscript

ACCEPTED MANUSCRIPT



(a)



(b)

Figure 8: Single Flow with Packet Loss. (a) Goodput under high-speed networks. (b) Goodput under long-delay (satellite) networks.

- The higher the packet loss rate, the worse the performance of the TCP variants will be.
- Reno is unable to efficiently use the available bandwidth compared with other TCP variants even when the link is error-free (congestion only).
- Hybla achieves notably good performance in long delay networks, but performs similar to Reno in high-speed networks with a 20 ms RTT. This observation can be attributed to the fact that Hybla is designed to have the same performance as Reno with a reference RTT of 25 ms.
- CUBIC achieves notably good performance as the packet loss rate increases in a long-delay network, but performs only slightly better than Reno in high-speed networks. This observation can be attributed to the fact that CUBIC designs its initial parameter with a 100 ms RTT reference network and determines K independent of RTT.
- INVS achieves the best performance in high-speed networks and long-delay networks when the packet loss rate is less than 10^{-2}. Overall, INVS improves the performance of TCP over lossy links.

4.2. Two Flow Scenario: RTT Fairness

In this section, we evaluate the RTT-fairness using the dumbbell topology. The RTT-fairness measures the fairness of bandwidth sharing among

Accepted manuscript

ACCEPTED MANUSCRIPT

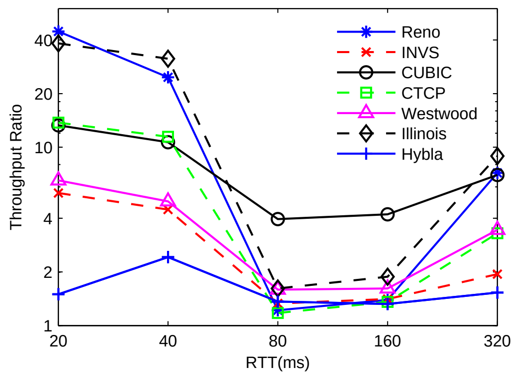


Figure 9: RTT Fairness

flows using the same TCP variants but different RTT, under the dumbbell topology. The bottleneck buffer is set to 250 packets (3 Mbit). Each simulation lasts 600 s and the results are obtained by averaging the goodput 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, and 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 20 s later.

Fig. 9 shows the goodput ratio of the higher goodput of the two flows to the lower one. In Fig. 9, the two flows will have a fair share of bottleneck bandwidth if their goodput ratio equals to 1. The results show that Reno and CTCP achieves the best RTT-fairness when the RTT of flow 1 and flow 2 are equal. CTCP improves the RTT-fairness of Reno when the RTT difference of the two flows increases. Due to the use of an RTT-independent increase rate, Hybla achieves the best RTT-fairness. INVS achieves better RTT-fairness than other variants (Reno, CTCP, Westwood, Illinois and CUBIC) when the RTT difference of the two flows increases. This result can be attributed to the fact that INVS adopts the advantage of Westwood and the adaptive increase factor.

Accepted manuscript

4.3. Multiple Mixed Flows

In this section, the performance of INVS is evaluated with multiple mixed flows under the hybrid topology as shown in Fig. 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. All flows begin randomly between 0 s and 5 s. The bandwidth of the satellite forward link (SFL) varies from 10 Mbps

ACCEPTED MANUSCRIPT

to 80 Mbps. As a result, the average bandwidth of the satellite flow varies from 2 Mbps to 16 Mbps, and the average BW over the wired bottleneck is 4 Mbps.

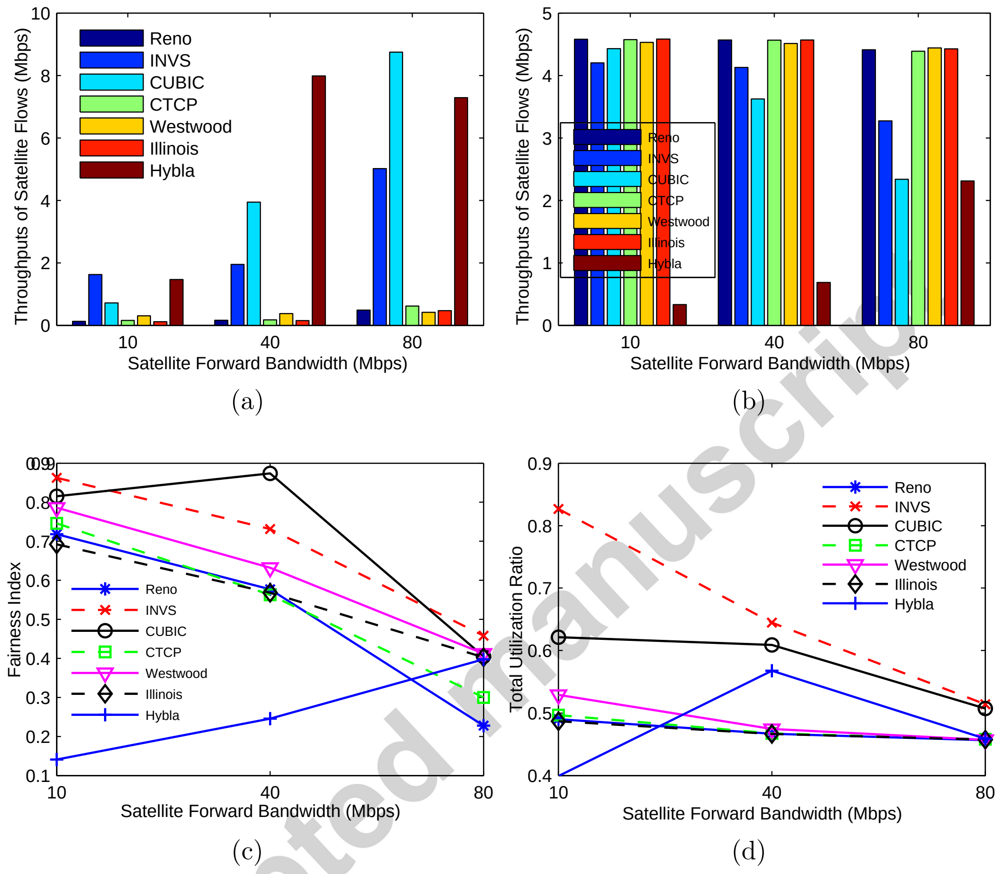


Figure 10: Hybrid satellite and wired networks evaluation. (a) Throughputs of satellite flows. (b) Throughputs of wired flows. (c) Fairness index. (d) Total link utilization ratio.

Fig. 10a and Fig. 10b show the average throughputs of the satellite flows and the wired flows, respectively. The results show that as the BW of SFL increases, which indicates that the SFL is no longer the bottleneck of satellite flows, the satellite INVS flows attain approximately the same average throughputs as the wired flows. The results also show that satellite CUBIC flows and Hybla flows are too aggressive in that they significantly impair the throughputs of the wired flows whereas the satellite flows of other

Accepted manuscript

ACCEPTED MANUSCRIPT

TCP variants (CTCP, Illinois, Reno and Westwood) are unable to take a fair share of bandwidth with wired flows, especially the wired CUBIC flows. Fig. 10c shows the fairness index. The results show that INVS outperforms other TCP variants in fairness when the BW of SFL is 10 Mbps and 80 Mbps, and CUBIC achieves the highest fairness index when the BW of SFL is 40 Mbps. Fig. 10d shows the total utilization ratios, which are the average values of the utilization ratio of SFL and that of the wired bottleneck. From the results, we note that INVS achieves the best total utilization ratio. To summarize the above arguments, we infer that INVS improves the fairness and the total efficiency of TCP in a hybrid topology.

5. Conclusion

A new TCP protocol known as INVS has been proposed to cope with the heterogeneity in Internet congestion control. INVS employs an exponential-function-based growth function for the congestion window to ensure efficiency under certain network scenarios, introduces an adaptive increase factor in the growth function to ensure that the window growth rate matches the path condition, and adopts an adaptive queue threshold in the loss classification scheme to improve the performance of TCP over lossy links. Furthermore, INVS also traces the trend of the path capacity to enable quick convergence of the congestion window and requires modification of the TCP sender only. Finally, a throughput model and illustrative analysis are developed and extensive simulations are conducted to evaluate the throughput, fairness, RTT-fairness and efficiency of INVS.

The simulation results of a single flow scenario with random packet loss show that INVS achieves better performance over lossy links in both high-speed networks and satellite networks. The evaluation results for RTT-fairness show that INVS achieves the best RTT-fairness except for Hybla. The simulation results under a multiple mixed flows scenario show that INVS achieves better efficiency and fairness than other TCP variants. Overall, INVS achieves better fairness, RTT-fairness and utilization than the other TCP variants in heterogeneous networks.

INVS distinguishes non-congestion loss from congestion loss. However, in wireless networks, non-congestion losses might be attributed to certain other causes (mobile handoff, link failure) rather than random losses. INVS is unable achieve satisfactory performance under these cases. Therefore, these causes of loss must be further researched, and actions for addressing

Accepted manuscript

ACCEPTED MANUSCRIPT

these causes of loss should be developed. Furthermore, to make the proposed algorithm available in Linux kernel and run efficiently, several aspects must be optimized, such as simplifying the calculation of factor k using the same k if the minimum RTT and bandwidth do not change more than the threshold.

Acknowledgments

This work is sponsored by the National Natural Science Foundation of China under grant No. 61171089, the Training Program of the Major Research Plan of the National Natural Science Foundation of China under grant No. 91438104, Pre-research Project of Civil Aircraft Key Technologies under grant MIIT No. Equipment [2010]307, the 2013 Innovative Team Construction Project of Chongqing Universities, and Graduate Students Research Program of Chongqing under grant No.CYS14005.

[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", IEEE Trans. Wireless Commun, vol. 18, no. 3, pp. 10-21, 2011.

[2] L. Pan and F. Yuguang, "On the Throughput Capacity of Heterogeneous Wireless Networks", IEEE Trans. Mobile Comput, vol. 11, no. 12, pp. 2073-2086, 2012.

[3] D. Papadimitriou, M. Welzl, M. Scharf and B. Briscoe, Open research issues in Internet congestion control, Internet Research Task Force (IRTF), 2011.

[4] D. Ros and M. Welzl, "Less-than-Best-Effort Service: A Survey of End-to-End Approaches", Communications Surveys and Tutorials, IEEE, vol. 15, no. 2, pp. 898-908, 2013.

[5] C. Caini and R. Firrincieli, "TCP Hybla: a TCP enhancement for heterogeneous networks", International journal of satellite communications and networking, vol. 22, no. 5, pp. 547-566, 2004.

[6] A. Afanasyev, N. Tilley, P. Reiher and L. Kleinrock, "Host-to-Host Congestion Control for TCP", Commun. Surveys Tuts, vol. 12, no. 3, pp. 304-342, 2010.

Accepted manuscript

ACCEPTED MANUSCRIPT

[7] 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, pp. 64-74, 2008.

[8] K. T. J. Song, M. Sridharan and C.-Y. Ho, "CTCP: Improving TCP-Friendliness Over Low-Buffered Network Links", Proc. 6th Int. Workshop on Protocols for FAST Long-Distance Networks (PFLDnet 2008), 2008.

[9] 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 Trans. Commun., vol. E95-B, no. 5, pp. 1679-1688, 2012.

[10] S. Liu, T. Basar and R. Srikant, "TCP-Illinois: A loss- and delay-based congestion control algorithm for high-speed networks", Performance Evaluation, vol. 65, no. 6-7, pp. 417-440, 2008.

[11] Alrshah M A, Othman M, Ali B, et al. "Comparative study of high-speed Linux TCP variants over high-BDP networks". Journal of Network and Computer Applications, 43: 66-75, 2014.

[12] W. Jingyuan, W. Jiangtao, H. Yuxing, Z. Jun, L. Chao and X. Zhang, "CUBIC-FIT: A High Performance and TCP CUBIC Friendly Congestion Control Algorithm", IEEE Commun. Lett, vol. 17, no. 8, pp. 1664-1667, 2013.

[13] K.-C. Leung and V. O. Li, "Transmission control protocol (TCP) in wireless networks: issues, approaches, and challenges", Commun. Surveys Tuts, 2006.

[14] Wan Y-J, Zhang L, Wu Z-M, Tang Y. "Uplink performance in 3G networks". In: 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications, Ottawa, 2014: 1213-1216.

[15] Chen Y-C, E. M. Nahum, R. J. Gibbens, and D. Towsley. "Measuring cellular networks: Characterizing 3 g, 4 g, and path diversity". Technical report, UMass Amherst Technical Report: UM-CS-2012-022

[16] F. Cheng Peng and S. C. Liew, "TCP Veno: TCP enhancement for transmission over wireless access networks, IEEE J. Sel. Areas Commun", vol. 21, no. 2, pp. 216-228, 2003.

Accepted manuscript

ACCEPTED MANUSCRIPT

[17] 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, pp. 467-479, 2002.

[18] C. Caini, R. Firrincieli and D. Lacamera, "PEPsal: a Performance Enhancing Proxy for TCP satellite connections", IEEE Trans. Aerosp. Electron. Syst, vol. 22, no. 8, pp. 7-16, 2007.

[19] T. Ye, K. Xu and N. Ansari, "TCP in wireless environments: problems and solutions", IEEE Commun. Mag, vol. 43, no. 3, pp. S27-S32, 2005.

[20] J.K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher and J. Barros, "Network Coding Meets TCP: Theory and Implementation", Proc. IEEE, vol. 99, no. 3, pp. 490-512, 2011.

[21] Xie J, Yu L, Jin F L. "Congestion control based on queuing delay and packet loss probability". Journal of Electronics and Information Technology, 2010, 32(9): 2058-2064.

[22] Bi Yuan-mei, A Study of Enhancing TCP Performance over Heterogeneous Networks, master thesis, Chongqing University, 2009

[23] Nan D, Wu R-Q, and Jie H. "TCP BRJ: Enhanced TCP Congestion Control Based on Bandwidth Estimation and RTT Jitter for Heterogeneous Networks". In: Proceedings of the Third International Conference on Communications, Signal Processing, and Systems. Springer International Publishing, 2015: 623-632.

[24] J. F. Kurose and K. W. Ross, "Computer Networking: A Top-Down Approach", Pearson Education, Limited, 2010.

[25] Peng Yang; Juan Shao; Wen Luo; Lisong Xu; Deogun, J.; Ying Lu, "TCP Congestion Avoidance Algorithm Identification". IEEE/ACM Transactions on Networking, vol.22, no.4, pp.1311,1324, Aug. 2014 doi: 10.1109/TNET.2013.2278271

Accepted manuscript