IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661 Volume 3, Issue 1 (July-Aug. 2012), PP 13-19
www.iosrjournals.org

Performance Evaluation of High Speed Congestion Control Protocols

Mohamed Ali Mani and Rachid Mbarek

ABSTRACT: *Computer networks are facing several technological and scientific challenges. Among these challenges is congestion control. It limits the quantity of information input at a rate less important than that of the transmission one to ensure good performance as well as protect against overload and blocking of the network. Researchers have done a great deal of work on improving congestion control protocols, especially on high speed networks.*

In this paper, we will be studying the congestion control alongside low and high speed congestion control protocols. We will be also simulating, evaluating, and comparing seven of high speed congestion control protocols : Bic TCP, Cubic TCP, Hamilton TCP, HighSpeed TCP, Illinois TCP, Scalable TCP and YeAH TCP, with multiple flows.

KEYWORDS: *HIGH-SPEED, TCP PROTOCOLS, CONGESTION CONTROL, PERFORMANCE, MULTIPLE FLOWS.*

I. INTRODUCTION

Due to the ease of finding information, data access and utilization by a simple mouse move in multimedia pages, Internet has become necessary in our daily lives. In fact, the number of Internet users is growing rapidly, and according to the ITU (International Telecommunication Union) it reached only 500 million users in the beginning of 2000s while it was 1.6 billion in 2008 to rise to the number of 2 billions in 2011.

This increase must undergo a parallel development of both of hardware and software at the same time. Early, throughput was about 56 kbps, however, now we talk about Gigabit Internet aiming to meet the users' needs mainly in astronomy and video conference fields.

In Internet, there are some machines responsible for transmitting data packets, and other that take off these packets from the queue. If the sender machine sends a data packet rate much more important than the rate of the receiver, network congestion is produced, hence, a congestion control will limit the quantity of input information with a lower rate than the transmission one to guarantee a good performance as well as a network protection against overloading and blocking.

The high number of high speed congestion control protocols led us to prepare this research work, which focuses on evaluating and comparing high speed congestion control protocols.

This paper is organized as follows:

In the second section we study the state of the art. In the third one, we present the architectures used as well as the curves and the performances evaluation for different high speed congestion control protocols.

STATE OF THE ART

Researchers have worked on the enhancement of high speed congestion control protocols. Practically every year, one or two protocols are implemented having for each one of them its own specific strengths and weaknesses.

Recently, the research works are interested in evaluating the performance of these protocols by comparing among 2 to 5 protocols with 1 to 12 flows [1] [2] [3] [4] [5].

HS-TCP [6]: If the congestion window is low, HS-TCP behaves just like the standard TCP, once the congestion window exceeds the max_ssthresh, it rises in an aggressive way. The algorithm of HS-TCP is an AIMD Additive Increment cwnd = cwnd + a(cwnd) Multiplicative Decrement cwnd = (1- b(cwnd)) x cwnd, the values used for HSTCP are a(cwnd) in the interval of [1,72], b(cwnd) in the interval of [0.1,0.5] and low_window equals 38 packets. when cwnd is lower than low_window a(cwnd) = 1 and b(cwnd) = 0.5 thus the algorithm acts just like a standard TCP, in the case of a packet loss, the cwnd is reduced to 50%, when the cwnd reaches the high_window a(cwnd) = 72 and b(cwnd)=1.

If there is an acknowledgment:

$$\begin{cases} w_1 = \dfrac{1+\beta}{2} \times \text{cwnd} & \text{cwnd} < w_1 \\ w_1 = \text{cwnd} & \text{cwnd} \geq w_1 \end{cases}$$

w_2 = cwnd

Scalable TCP [7]: For this protocol the cwnd increases with 0.001, if a congestion is detected the cwnd reduces with 0.125 independently from the throughput and the packet size. This algorithm is a MIMD; Multiplicative

Increase since the cwnd increases with 0.01 per acknowledgment, Therefore there is a total increase of 0.01 cwnd per RTT, and Multiplicative Decrease since the cwnd decreases with 0.125 per packet loss.

BIC TCP [8]: uses a function that allows to have a rapid increase of cwnd when its value is far from the maximal threshold Smax and to slowly decrease when having a value close to w_1.

cwnd = (1 – β) × cwnd

With :

$$f_\alpha(\delta, \text{cwnd}) = \begin{cases} \frac{\beta}{\delta} & \delta \leq 1 \text{ and cwnd} < w_1 \\ & \text{or } w_1 \leq \text{cwnd} < w_1 + \beta \\ \delta & 1 < \delta \leq S_{max} \text{ and cwnd} < w_1 \\ \frac{w_1}{\beta - 1} & \beta \leq \text{cwnd} - w_1 < S_{max}(1-\beta) \\ S_{max} & \text{otherwise} \end{cases}$$

cwnd : Current congestion window
S_{max} : Maximal threshold
S_{min} : Minimal threshold
B : Decrease factor usually equals 0.875

CUBIC TCP [9]: The name CUBIC refers to the function of the congestion window increase which is cubic, when receiving an ACK:

$$\text{cwnd} = C(T - K)^3 + Max_{cwnd}$$

When having a packet loss:

$$\text{cwnd} = \beta\, Max_{cwnd}$$

With :

Cwnd*:*Current congestion window
C*:*Constant
T*:*Duration of time since the last congestion
Max_{cwnd} *:*Size of the last congestion window
K*:*$\sqrt[3]{Max_{cwnd}\ \beta/C}$

HTCP [10]: uses Δ the duration since the last congestion rather than cwnd as information about the product delay bandwidth BDP. The increase parameter of AIMD (additive-increase/multiplicative-decrease) varies depending on Δ. It also depends on the value of RTT which has provoked the unfairness among the competitor flows with different RTTs.

In case of receiving an ACK:

$$\text{cwnd} = \text{cwnd} + \frac{2(1-\beta)\, f_\alpha(\Delta)}{\text{cwnd}}$$

In case of packet loss:

$$\text{cwnd} = g_\beta(B) \times \text{cwnd}$$

With :

$$f_\alpha(\Delta) = \begin{cases} 1 & \Delta \leq \Delta_L \\ \max\left(\bar{f}_\alpha(\Delta)\, T_{min}, 1\right) & \Delta > \Delta_L \end{cases}$$

$$g_\beta(B) = \begin{cases} 0.5 & \left|\frac{B(K+1) - B(k)}{B(K)}\right| > \Delta_B \\ \min\left(\frac{T_{min}}{T_{max}}, 0.8\right) & \text{otherwise} \end{cases}$$

cwnd*:*Current congestion window
α*:*Increase parameter
β*:*Decrease parameter
Δ*:*Duration of time since the last congestion
Δ_L *:*Threshold to change the mode low speed to highspeed
T_{min} *:*Minimal duration of sending and receiving packets
T_{max} *:*Maximal duration of sending and receiving packets
B(K + 1)*:*Maximal throughput reached before a congestion
$\bar{f}_\alpha(\Delta)$*=*1 +10 (Δ - Δ_L) + 0,25 (Δ - Δ_L)²

Illinois TCP [11] : Similar to standard TCP, Illinois-TCP increases the congestion window by using two parameters α and β, it is based on packet loss to define the congestion window value, however the values of α and β are not constant by using the delays.

When receiving an ACK:

$$cwnd = cwnd + \frac{\alpha}{cwnd}$$

When losing packets:

$$cwnd = (1- \beta) \times cwnd$$

With :

$$\alpha = f_1(d_a) = \begin{cases} \alpha_{max} & d_a \leq d_1 \\ \frac{k_1}{k_2 + d_a} & otherwise \end{cases}$$

$$\beta = f_2(d_a) = \begin{cases} \beta_{min} & d_a \leq d_2 \\ k_3 + k_4 d_a & d_2 < d_a < d_3 \\ \beta_{max} & otherwise \end{cases}$$

$$k_1 = \frac{(d_m - d_1)\,\alpha_{min}\,\alpha_{max}}{\alpha_{max} - \alpha_{min}} *$$

$$k_2 = \frac{(d_m - d_1)\,\alpha_{min}}{\alpha_{max} - \alpha_{min}} - d_1$$

$$k_3 = \frac{\beta_{min}\,d_3 - \beta_{max}\,d_2}{d_3 - d_2} * k_4 = \frac{\beta_{max} - \beta_{min}}{d_3 - d_2}$$

$$\alpha_{min} = \frac{k_1}{k_2 + d_{moy}}, \alpha_{max} = \frac{k_1}{k_2 + d_1}$$

$$\beta_{min} = k_3 + k_4 d_2 \ , \beta_{max} = k_3 + k_4 d_3$$

YeAH TCP [12] the algorithm starts with the Slow Start if cwnd < ssthresh and Scalable TCP once cwnd reaches the ssthresh. This protocol uses two modes, fast mode and slow mode. In the case of the fast one, the congestion window increases aggressively, for the second mode, the protocol behaves similarly to Reno TCP. The mode is chosen according to the queue status. Having RTT_{BASE} the minimal RTT measured by the sender and RTT_{MIN} the minimal RTT estimated from the current window. The estimated delay is:

$$RTT_{queue} = RTT_{min} - RTT_{base}$$

Thanks to this value, the number of packet on hold in the queue can be defined as follows:

$$Q = RTT_{queue} \times G = RTT_{queue} \times \frac{cwnd}{RTT_{min}}$$

And G is the throughput.

II. SIMULATION AND PERFORMANCES EVALUATION

2.1. Architecture

In order to simulate the high speed congestion control protocols, we have chosen a topology (as shown in figure 1) composed of a sender and a receiver linked together with two routers by a line of 1Gbps of bandwidth, the delay is 1ms. The routers are linked to each other with a line having a bandwidth of 200Mbps, the delay is equal to 92 ms and the queue capacity is exactly 100 packets [13]. The MSS size equals 1460 bytes. The differences among the bandwidths capacities provoke congestion.

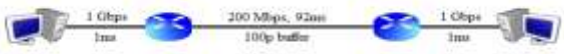


Figure 1. Basic topology

To evaluate the performances, we have used 2, 4, 6, 8, 10, 12, 16 and 24 identical flows (as shown in figure 2) with a link capacity of 1 Gbps and a delay of 1 ms.

Figure 2. Topology with multiple flows

2.2. TCP-Friendly

To test the TCP Friendly of different protocols, we have used a multiple flows topology with 4 input flows triggered at the same time. The results are shown in the following graphs:

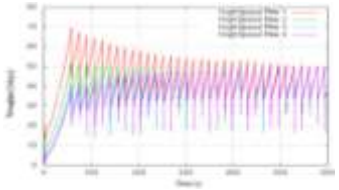
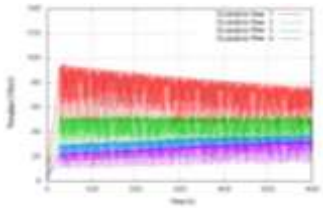


Figure 3.a. HS TCP : Throughput variation for 4 flows



*

Figure 3.b. Scalable TCP : Throughput variation for 4 flows

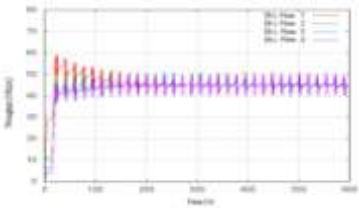
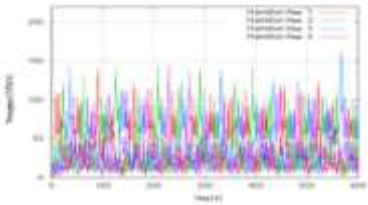


Figure 3.c. Bic TCP : Throughput variation for 4 flows



*

Figure 3.d. Hamilton TCP : Throughput variation for 4 flows

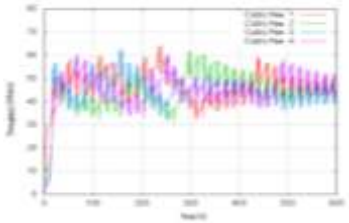
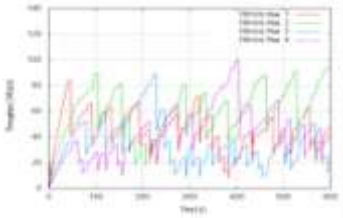


Figure 3.e. Cubic TCP : Throughput variation for 4 flows



*

Figure 3.f. Illinois TCP : Throughput variation for 4 flows

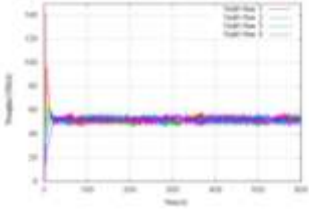


Figure 3.g. YeAH TCP : Throughput variation for 4 flows

2.3. Efficiency

The efficiency (rate of utilization or performance) of a network is the percentage of utilization [14]. Mathematically, it is the division of the average throughput by the optimal throughput.
An efficient network uses the maximum of capacity.
The following variable q_i is used as the average throughput of a flow i.
The throughput of a source with n flows is:

$$Q = \frac{1}{n} \sum_{i=1}^{n} q_i$$

We calculate the ratio between the average throughput and the optimal one which is the result of an ideal network performance:

$$E = \frac{Q}{Q_{opt}}$$

We have used these two topologies to simulate the seven protocols for 1, 2, 4, 6, 8, 10, 12, 16 and 24 flows.
The efficiency calculus results are as mentioned in the following graph:

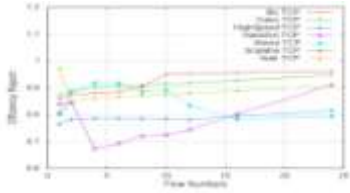


Figure 4. Efficiency for different flow numbers

2.4. Fairness

The fairness is the attempt of sharing the network capacities among users in a fair way.
For the purpose of measuring the fairness, one method is used in the networks field which is called the Maximin law proposed by Raj Jain [15]. Here is the procedure that allows us to calculate the fairness of a proposed algorithm:
Having an algorithm that provides the distribution $v_i = [x_1, x_2, \ldots, x_n]$ instead of the optimal distribution $v_{opt} = [x_{1,opt}, x_{2,opt}, \ldots, x_{n,opt}]$. We calculate the standardized distribution for every source as follows:

$$X_i = \frac{x_i}{x_{opt}}$$

Thus, the fairness index F equals the sum of distributions squared and divided by the square of sums:

$$F = \frac{(\sum_{i=1}^{n} X_i)^2}{n \sum_{i=1}^{n} X_i^2}$$

We have used these two topologies to simulate the seven protocols for 1, 2, 4, 6, 8, 10, 12, 16 and 24 flows.
The fairness calculus results are as mentioned in the following graph:

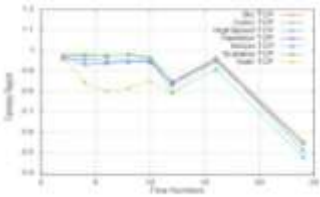


Figure 5. Fairness for different flow numbers

2.5. Performance

The performance of a congestion control algorithm is the relation between efficiency and fairness:
Performance = $\alpha \times E + (1 - \alpha) \times F$
With $\alpha = [0{,}1 .. 0{,}9]$, E the efficiency and F the fairness.
For the different algorithms, we have calculated the performances for a network rather efficient with $\alpha = 0{,}8$, a network rather fair with $\alpha = 0{,}2$ and a balanced network when $\alpha = 0{,}5$. The results for 1, 2, 4, 6, 8, 10, 12, 16 and 24 flows are as follows:

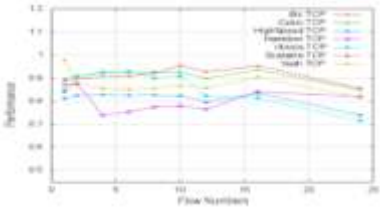


Figure 6.a. Performance for different flow numbers (α = 0,8)

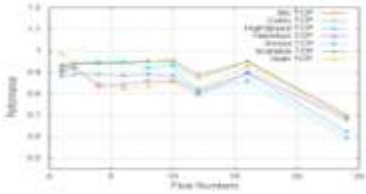


Figure 6.b. Performance for different flow numbers (α = 0,5)

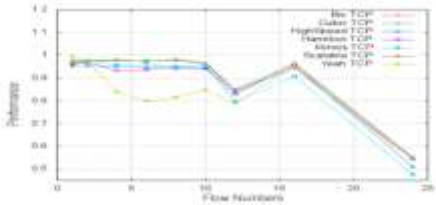


Figure 6.c. Performance for different flow numbers (α = 0,2)

2.6. Link Utilisation per RTT

We varied the RTT for different congestion control algorithms to calculate the link utilization. The used RTT values are: 20 ms, 60 ms, 100 ms, 140 ms, 180 ms and 220 ms.

We used the basic topology with only one flow; the results are described in the following graph:

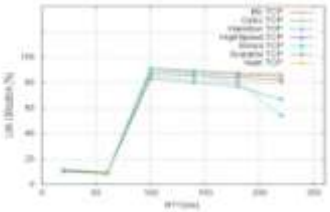


Figure 7. Link utilization for different RTTs

2.7. Link utilization for different queue sizes

Certainly the queue size has an important impact on the efficiency of the network, but which algorithm benefits the more of this change? We varied the queue size for different congestion control algorithms to calculate the link utilization. The used sizes are: 50, 100, 150, 200, 250 and 300 packets.

We used the basic topology with one flow, the results are as follows:

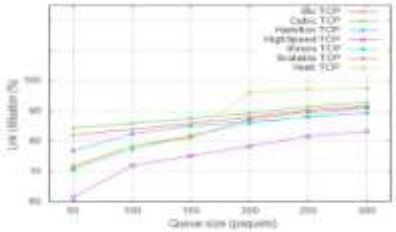


Figure 8. Link utilization for different queue sizes

III. CONCLUSIONS

The high number of congestion control algorithms never satisfied the researchers. Since this field is always having changes and enhancements due to the users' needs and the evolution of both of hardware and software in Telecommunications. Indeed, it is difficult to conceive an algorithm that gives satisfying results for all the architectures.

During this research work, we simulated seven high speed congestion control protocols for 1, 2, 4, 6, 8, 10, 12, 16 and 24 flows. We evaluated their performances by calculating the efficiency, the fairness as well as the performance while varying the RTT and the queue size.

Basing on the simulation and the high speed protocols performance evaluation, we can conclude that:

- Some protocols perform well in some defined cases, but weak in others.
- The network architecture has got an important impact on the protocols performance.
- For 24 flows, all protocols are unfair.
- For 24 flows, the protocols: Bic TCP, Cubic TCP, Hamilton TCP, Scalable TCP and YeAH TCP are efficient.
- Scalable is not only TCP-friendly, but also selfish, since the first flow gets the most important throughput, and doesn't leave its place for any other flow such as the case of Cubic TCP, Hamilton TCP and Illinois TCP.
- The protocols Bic TCP, HighSpeed TCP and YeAH TCP are TCP-friendly
- For low values of RTT, all the protocols use the network capacities in a bad way.
- For low values of the queue size, Cubic TCP is the most efficient.
- YeAH TCP and Illinois TCP give the best results while increasing the queue size.
- Although his aggressiveness, YeAH TCP is quite TCP-friendly. It is also quite performant, but it doesn't suit all the architectures.

Further work is required to evaluate the performances of these protocols by using Relentless TCP which will be the topic for a multitude of research works in the near future, particularly to define if the protocols that have more stability and take more time before falling another time in a new congestion such as Cubic TCP and Illinois TCP in order to check if they will well exploit the law that forms the basis of Relentless TCP which is the reduction of the congestion window with the number of lost segments. It will be also interesting to create a model that changes dynamically the congestion control protocols used in terms of flow number to better exploit the network capacities as well as studying the QoS.

REFERENCES

[1] R. Mbarek, M. T. BenOthman and S. Nasri. "Fairness of High-Speed TCP Protocols with Different Flow Capacities". Journal Of Networks, Vol. 4, No. 3, pp. 163-169, May 2009.

[2] S. Hemminger. "TCP testing – Preventing global Internet warming". LinuxConf Europe, Sep. 2007.

[3] S. Ha, L. Le, I. Rhee and L. Xu. "Impact of Background Traffic on Performance of High-Speed TCP Variant Protocols". The International Journal of Computer and Telecommunications Networking, Vol.51, May, 2007.

[4] M. Nabeshima and K. Yata. "Performance Evaluation and Comparison of Transport Protocols for Fast Long-Distance Networks". IEICE Trans. Commun., Vol.E89-B, No.4, pp. 1273-1283, Apr. 2006.

[5] M. Weigle, P. Sharma and J. Freeman. "Performance of Competing High-Speed TCP Flows". Networking 2006, LNCS 3976, pp. 476-487, 2006.

[6] S. Floyd. "High-Speed TCP for Large Congestion Windows". RFC 3649, Experimental, Dec. 2003.

[7] T. Kelly. "Scalable TCP : Improving Performance in High-Speed Wide Area Networks". In Proceedings of PFLDnet, Geneva, Switzerland, Feb. 2003.

[8] L. Xu, K. Harfoush, and I. Rhee. "Binary Increase Congestion Control for Fast, Long Distance Networks". In Proceedings of IEEE INFOCOM, Hong Kong, Mar. 2004.

[9] I. Rhee and L. Xu. "CUBIC: A New TCP-Friendly High-Speed TCP Variant". In Proceedings of PFLDnet, Lyon, France, Feb. 2005.

[10] R. N. Shorten and D. J. Leith. "H-TCP : TCP for high-speed and long-distance networks". In Proceedings of PFLDnet, Argonne, Feb. 2004.

[11] S. Liu, T. Basar and R. Srikant.,"TCP-Illinois : A Loss- and Delay-based Congestion Control Algorithm for High-Speed Networks", Performance Evaluation, 65(6-7) : 417-440, Mar. 2008.

[12] A. Baiocchi, A. P. Castellani and F. Vacirca. "YeAH-TCP: Yet Another Highspeed TCP". PFLDnet, Roma, Italy, Feb. 2007.

[13] J. Semke, J. Mahdavi, and M. Mathis. "Automatic TCP Buffer Tuning", Computer Communications Review, a publication of ACM SIGCOMM, Vol. 28, No. 4, Oct. 1998.

[14] R. Mbarek, M. T. BenOthman and S. Nasri. "Performance Evaluation of Competing High-Speed TCP Protocols". IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.6, pp. 99-105, June 2008.

[15] R. Jain. "Congestion Control and Traffic Management in ATM Networks : Recent Advances and A Survey". Computer networks and ISDN Systems, Nov. 1995.

Authors

M. Mohamed Ali Mani received his M.S degree from ISITCOM, Hammam Sousse, Sousse University Tunisia, in 2011. He is currently a lecturer in the Department of Computer Science in ISTLS in the same University. His research interests lie in the general area of computer networks, including high speed transports protocols, flow control, congestion control in high-speed networks and wireless networks.

Dr. Rachid Mbarek received his Ph.D from the College of Engineering (ENIS), Sfax University, Tunisia, received his M.Sc. degree in Computer Engineering in 2003 from the same University. He is currently a Lecturer in the Department of Telecommunication in the ISITCOM at Sousse University, Tunisia. His research interests lie in the general area of computer communication networks, including high-speed transports protocols, flow control and congestion control in high-speed networks, network performance analysis, peer to peer networking.