Performance Evaluation of Competing High-Speed TCP Protocols.

Rachid Mbarek, Mohamed Tahar Ben Othman*, Salem Nasri

**Senior Member, IEEE*

College of Computer, Qassim University, kingdom of Saudi Arabia.

Abstract: Fairness and efficiency were, and still are, the center of the most of researches conducted to evaluate the performance of high-speed TCP protocols. The majority of these researches were carried out using two flows. Most of them dealt with Fairness vs round trip time RTT. Some others used several flows but covered only two to three protocols. It may not give a precise idea about Fairness of different protocols and impact performance results. In this paper we propose an evaluation method of Fairness and efficiency for different high-speed protocols: HTCP, HS, Scalable TCP, BIC and CUBIC protocols using multiple flows starting at the same time in the first step, each flow starts at different time in the second one.

Keywords: High-Speed TCP Protocols, Fairness, Efficiency, Performance.

1. INTRODUCTION

Several researches showed that Standard TCP, which handles most Internet traffic, has limitations when a connection attempts to send data at high speed. To solve this problem several high-speed protocols have been developed. These new high-speed variants of TCP were designed to solve the limitations with high-speed transfers while maintaining efficiency and fairness to Standard TCP flows [7]. Then researches were conducted to evaluate the performance of these new protocols. Most of these researches used a well known topology with 2 sources, 2 routers and 2 destinations [7-12]. Only few considered a topology where several flows are used in source and destination [5, 13]. These researches focused only on one or two protocols evaluation. Some of them focused on fairness without taking into account the efficiency. This may not give correct performance evaluation.

As we believe that these new high-speed TCP implementations should be tested in an environment as close as possible to their likely real-world deployment, we consider multi-flows in source and destination in the present paper.

The rest of this paper will be as follow. In the second section we introduce the related previous work that dealt with high-speed TCP protocol evaluation and mainly those considered fairness and efficiency. The third section presents the background where we define the different studied protocols. In the section 4, we describe our topology and background traffic. Section 5 deals with the simulation and results. Finally the conclusions and future work are presented.

2. RELATED WORK

Let us briefly introduce the main studies on performance evaluation of high-speed protocols relevant to our paper. Yee-Ting Li et al give experimental results evaluating the performance of Scalable-TCP, HS-TCP, BIC-TCP, FAST-TCP and H-TCP [8]. Four main criteria were used to evaluate the performance: Fairness, Backward compatibility, Efficiency and Responsiveness. A dumbbell topology was used with only two competing flows. When RTT Fairness was measured, FAST and Scalable exhibit very substantial unfairness. Lisong et al. proposed a new protocol BI-TCP and studied the performance under the following criteria: Scalability, RTT fairness, TCP friendliness and Fairness, and convergence [14]. 2 to 10 flows were used but comparison was done only with AIMD, Scalable and HS. A more recent study, presents experimental results for evaluating fairness of several protocols. It focuses on four different views of fairness: TCP-friendliness, RTT-fairness, intra- and inter-protocol fairness [12]. The experimental tested setup is also only two flows and efficiency was not studied so it may impact the found results.

3. BACKGROUND

In this section, we introduce the different high-speed protocols, for which we are evaluating the performance: H-TCP, high-speed TCP (HSTCP), Scalable-TCP, BIC-TCP,

---

Manuscript received June 5, 2008.
Manuscript revised June 20, 2008.

and CUBIC-TCP. All These protocols have been recently the subject of intensive research, with patches implementing each of them on the Linux operating system publicly available [1, 2, 3, 4, 5]. All of the high-speed protocols that we evaluate attempt to be fair to Standard TCP flows that might be sharing the link. These protocols use Standard TCP when the TCP window *cwnd* is less than a threshold value and only use the high-speed version when *cwnd* is above the threshold [7].

HS-TCP: When an acknowledgment (ACK) is received, HS-TCP increases *cwnd* by α(*cwnd*)/ *cwnd*. When one or more losses are detected in an RTT, HS-TCP sets *cwnd* to (1-β(*cwnd*))**cwnd* [1]. The goal is for a more aggressive increase and less aggressive decrease than Standard TCP in low-loss environments (i.e, environments where w is allowed to grow past the threshold, LowWindow). Current implementations of HS-TCP use a lookup table to determine the values of α(cwnd) and β(cwnd). Recommended settings allow for α(cwnd) in the range of [1, 72] and β(cwnd) in the range [0.1, 0.5] segments.

Scalable-TCP: The basic idea in Scalable-TCP is to make the recovery time after a congestion event independent of window size [2]. Specifically, Scalable-TCP proposes that the TCP cwnd be updated as follows:

Ack: *cwnd* ← *cwnd* + α

Loss: *cwnd* ← β * *cwnd*

Suggested values for the parameters α and β are 0.01 and 0.875, respectively. A mode switch is used whereby the standard TCP *cwnd* update rules are used when *cwnd* is less than a threshold, *Low_Window*, and the Scalable-TCP update rules are used for larger *cwnd* values.

H-TCP: uses the elapsed time Δ since the last congestion event, rather than *cwnd*, to indicate path bandwidth-delay product and the AIMD increase parameter is varied as a function of Δ [2]. The AIMD increase parameter is also scaled with path round-trip time to mitigate unfairness between competing flows with different round-trip times. The AIMD decrease factor is adjusted to improve link utilization based on an estimate of the queue provisioning on a path. In more details, H-TCP proposes that *cwnd* be updated as follows

$$\text{Ack:}\quad cwnd \leftarrow cwnd + \frac{2(1-\beta)f_\alpha(\Delta)}{cwnd} \qquad (1)$$

$$\text{Loss:}\quad cwnd \leftarrow g_\beta(B) \times cwnd \qquad (2)$$

with

$$f_\alpha(\Delta) = \begin{cases} 1 & \Delta \le \Delta_L \\ \max(\bar{f}_\alpha(\Delta) T_{\min}, 1) & \Delta > \Delta_L \end{cases} \qquad (3)$$

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

Where Δ_L is a specified threshold such that the standard TCP update algorithm is used while $\Delta \le \Delta_L$. A quadratic increase function f_α is suggested in [18], namely $f_\alpha(\Delta) = 1+10(\Delta - \Delta_L) + 0.25(\Delta - \Delta_L)^2$. T_{min} and T_{max} are measurements of the minimum and maximum round-trip time experienced by a flow. B(k+1) is a measurement of the maximum achieved throughput during the last congestion epoch.

BIC-TCP: employs a form of binary search algorithm to update *cwnd* [10]. Briefly, a variable w1 is maintained that holds a value halfway between the values of *cwnd* just before and just after the last loss event. The *cwnd* update rule seeks to rapidly increase *cwnd* when it is beyond a specified distance *Smax* from w_1, and update *cwnd* more slowly when its value is close to w_1. Multiplicative backoff of *cwnd* is used on detecting packet loss, with a suggested backoff factor of 0.8. In more detail, the BIC-TCP update algorithm is as follows.

$$\text{Ack:}\quad \begin{cases} \delta = \dfrac{(w_1 - cwnd)}{B} \\ cwnd \leftarrow cwnd + \dfrac{f_\alpha(\delta - cwnd)}{cwnd} \end{cases} \qquad (4)$$

$$\text{Loss:}\quad \begin{cases} w_1 = \begin{cases} \dfrac{1+\beta}{2} \times cwnd & cwnd < w_1 \\ cwnd & otherwise \end{cases} \\ w_2 = cwnd \\ cwnd \leftarrow \beta \times cwnd \end{cases} \qquad (5)$$

With

$$f_\alpha(\delta, cwnd) = \begin{cases} B/\delta & (\delta \le 1, cwnd < w_1) \\ & or(w_1 \le cwnd < w_1 + B) \\ \delta & 1 < \delta \le S_{\max}, cwnd < w_1 \\ w_1/(B-1) & B \le cwnd - w_1 < S_{\max}(B-1) \\ S_{\max} & otherwise \end{cases} \qquad (6)$$

CUBIC: CUBIC is a modification of BIC-TCP with the goal of improving on BICTCP's fairness [1]. In CUBIC, the window increase is determined by a cubic function:

$cwnd = C(t - K)^3 + Wmax$, where C is a constant used for scaling, t is the time since the window was last reduced, *Wmax* is the size of the window just before the window was last reduced, and K= $(W_{max}.\beta/C)^{1/3}$, where β is a constant decrease factor. When a loss occurs, the window is reduced to β.*Wmax*, with β= 0.8.

4. Performance evaluation criteria

4.1. Fairness

Several definitions of fairness exist among which: The Chiu and Jain fairness index defined in [6] and asymmetry metric from Bullot et al. [17]. We use the fairness index to evaluate fairness between different flows.

$$\textbf{Fairness} = \frac{\left(\sum_{i=1}^{n} \frac{\tilde{Q}_i}{Q_{opt}}\right)^2}{n * \sum_{i=1}^{n}\left(\frac{\tilde{Q}_i}{Q_{opt}}\right)^2} \qquad (7)$$

Where $\tilde{Q}_i$ is the average throughput of the flow *i*, n is the number of flows and Q_{opt} is the optimal throughput for each flow which is here equal to: $\frac{B_L}{n}$ (B_L is the link bandwidth).

To evaluate the fairness, we consider multiple TCP flows for each protocol defined above and propose the following tests:

(i) Fairness vs multiple flows starting at the same time: Measure the average throughput of each flow when each flow operates the same High-Speed protocol, has the same propagation delay and has a shared bottleneck link and determine the optimal throughput Q_{opt} for all flows.

(ii) Fairness vs multiple flows starting at different times: Fairness is calculated here in each interval where the number of flows is constant. In different periods Q_{opt} is different as the number of flows is different.

4.2. Efficiency

Fairness may not have significance without efficiency. Efficiency is calculated as the total average throughput of all flows over the link bandwidth:

$$Efficiency = \frac{\sum_{i=0}^{n} \tilde{Q}_i}{B_L} \qquad (8)$$

Where B_L is the link bandwidth and $\tilde{Q}_i$ is the average throughput of the flow i.

4.3. Trade-off between Fairness and Efficiency

As we are dealing with Multi-Objective Optimization it is difficult to compare different protocols by only comparing their fairness and efficiency separately.

A standard technique for Multi-Objective Optimization is to maximize a positively weighted convex sum of the objectives [14-15], that is,

$$\sum_{i=1}^{n} \alpha_i f_i(x), \qquad \alpha_i > 0, \ \text{i=1,2,...n.} \qquad (9)$$

To calculate a trade-off between fairness and efficiency the convex combination of two points is used:

$$Performance = \alpha \times E + (1-\alpha) \times F \qquad (10)$$

Where $0 < \alpha < 1$, E, and F stand for Efficiency and Fairness respectively. As we believe that maximizing the fairness does only have a meaning when the efficiency is high we took $\alpha = 0.2$.

5. Topology and Background Traffic

As a baseline topology, we consider many flows sharing a single congested link as shown in Fig. 1. The bandwidth of this link is 622 Mbps and it has a propagation delay of 48 ms. The access links have a capacity of 1000 Mbps and a delay of 1ms, so that the minimum round-trip time for all flows is approximately 100 ms. The queue size is set to 20% BDP.

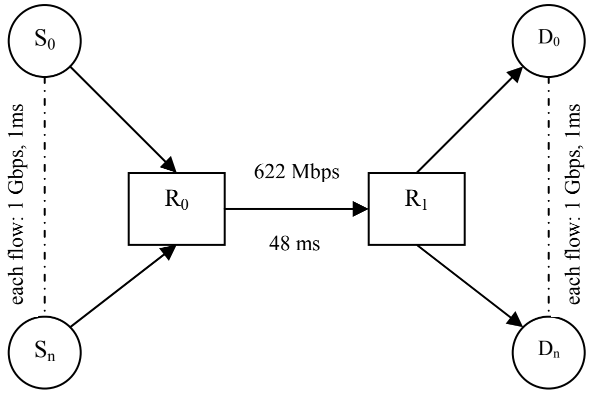


Fig. 1: Dumbbell topology with multiple flows

To evaluate the fairness and efficiency of the different protocols we used 2, 4, 6 , 8, 10 and 12 flows where each flow uses a link capacity of 1Gbps and a delay of 1ms. In the first set of simulations, all flows start at the same time (at 0.1s) and in the second set the first flow starts at 0.1s and after each 50s a new flow from the remaining ones starts. Each simulation run lasts 500s in the first set and 600s in the second set.

6. Simulation results

In this section, we present the different measurements conducted using different number of flows. All simulations are performed under ns-2 [16]. In the First set of simulations, all flows have same start time (SST). For the second set, different flows have different start time (DST). We provide both measurements over total run time (TRT) and over only the second half time (SHT).

6.1. First set:

For each proposed TCP protocol we run several simulations using 2, 4, 6, 8, 10 and 12 flows as sources and destinations with the same features. The flow capacity is 1Gbps and the delay is 1ms. Fig. 2, 3 and 4 present the efficiency, fairness and performance respectively for all protocols measured over all simulation period (500s). Fig. 5, 6 and 7 depict the efficiency, fairness and performance measured at the second half time.

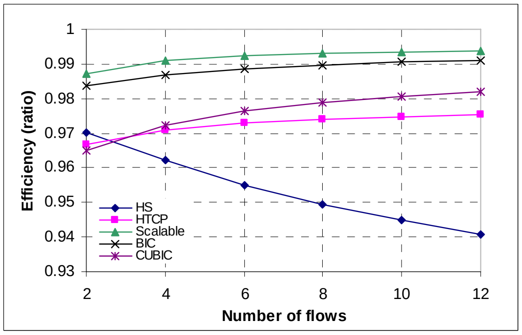


Fig. 2: Efficiency (SST-TRT)

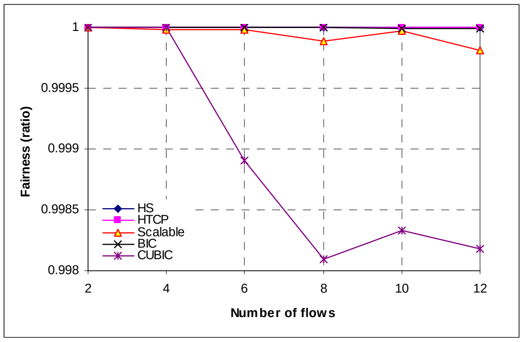


Fig. 3: Fairness (SST-TRT)

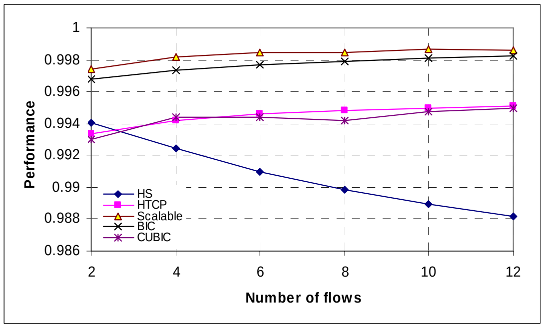


Fig. 4: Performance (SST-TRT)

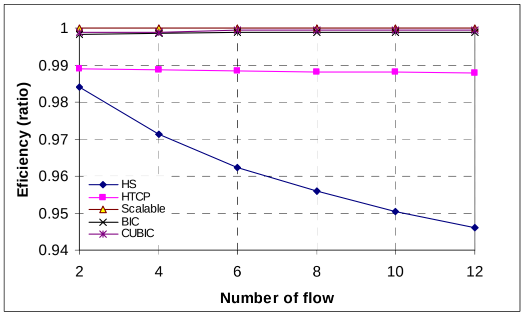


Fig. 5: Efficiency (SST-SHT)

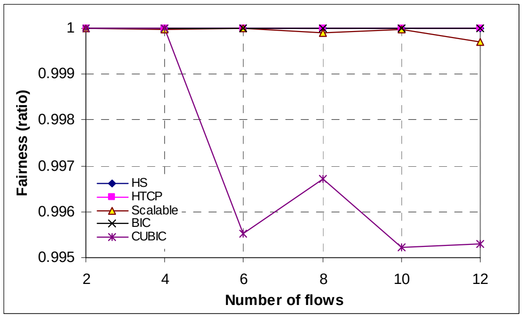


Fig. 6: Fairness (SST-SHT)

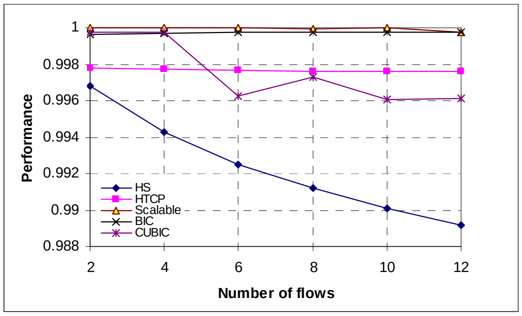


Fig. 7: Performance (SST-SHT)

6.2. Second set:

We run the same simulation for each proposed TCP protocol were each flow starts at different time. The first flow starts at 0.1s and after each 50s a new flow from the remaining ones starts. Efficiency has to be considered during all the simulation time. Fig. 8 shows the efficiency, within each period of 50s, of different protocols using 12 flows.

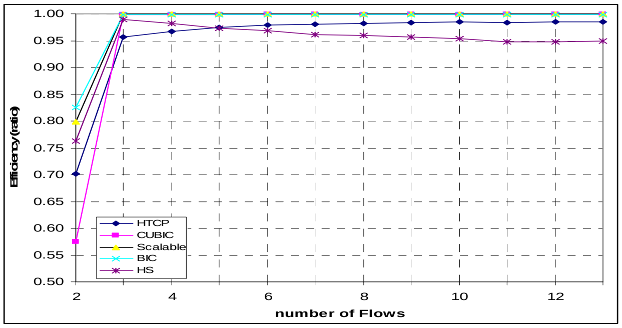


Fig. 8: Efficiency (DST-SHT)

As shown in Fig 8, HS and HTCP have the lowest efficiency. The HS efficiency drops every time a new flow is added to the system.

The fairness is measured only for 2, 4, 6 and 8 flows in the second half time as there is no stability in the system each time a new flow is in transition state. The protocol Scalable and then CUBIC showed the worst fairness. The first Scalable flow is more aggressive than any other flow starting later as shown in the Fig. 9, which is aligned with the results found in [7].

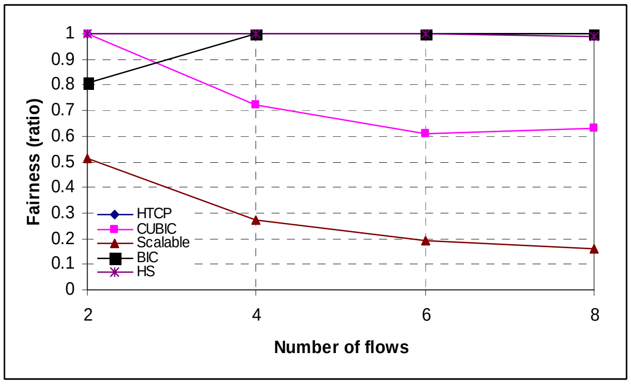


Fig. 9: Fairness (DST-SHT)

7. Conclusions and future work

In this paper we present two different ways to evaluate performance of competing high speed protocols HS, HTCP, Scalable, BIC and CUBIC: using multiple flows (instead of only two used in most related works) starting at the same time, and starting at different times.

As we think that performance cannot be evaluated by only measuring fairness factor without taking into account efficiency and that both factors cannot be compared separately, we consider this as Multi-Objective Optimization problem and make a tradeoff between fairness and efficiency using a convex combination of two points.

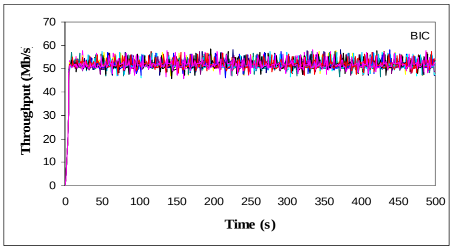
Although CUBIC is a modification of BIC with the goal of improving on BIC's fairness [1] simulations showed that CUBIC has a bad fairness when dealing with flows starting at different time.

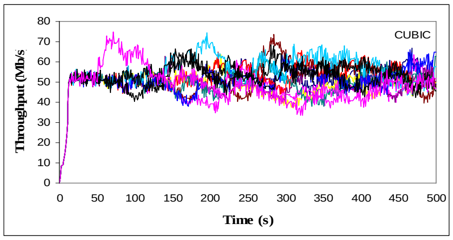
Overall, BIC followed by Scalable performed well.

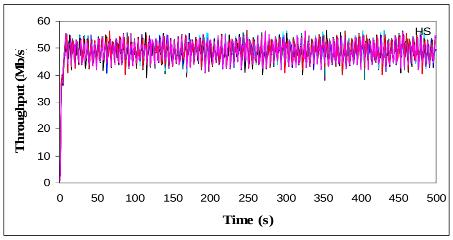
Our future work will be concentrated on finalizing this study by evaluating the performance using different flow capacities and different RTTs.

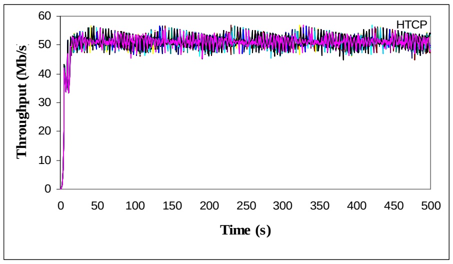
Appendix

In this appendix, we present the throughput of different protocols for 12 flows. In Fig. 10, all flows at the same start time same flow capacity (SST), Fig. 11 shows the throughput at different start times.









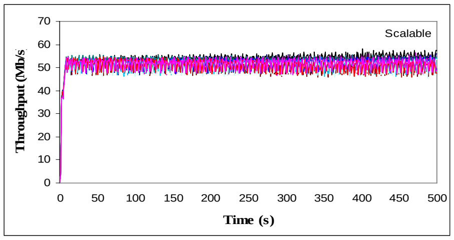


Fig. 10: Throughput (SST)

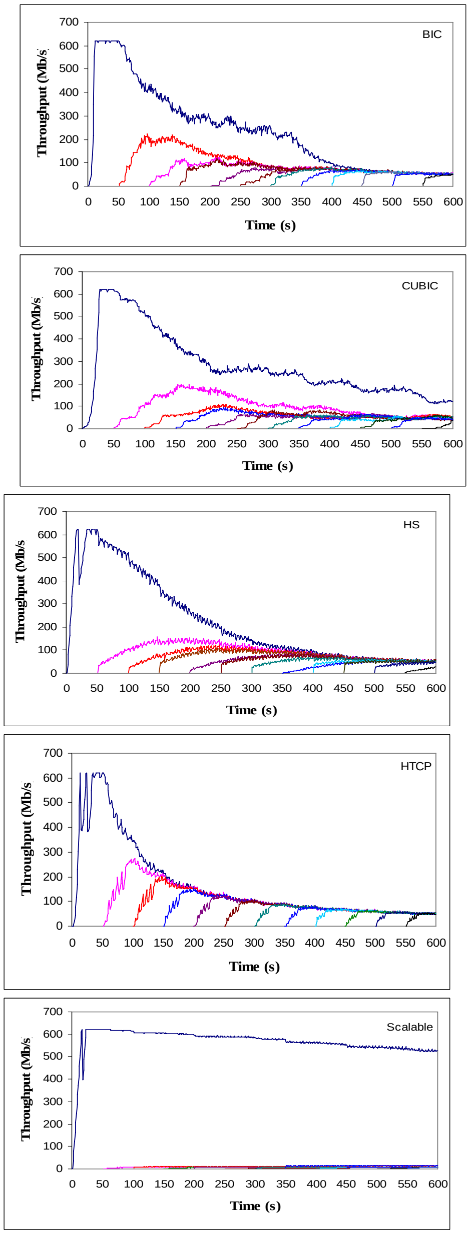


Fig. 11: Throughput (DST)

References

[1] S. Floyd. High Speed TCP for Large Congestion Windows. RFC 3649, Experimental, December 2003.

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

[3] T. Kelly. Scalable TCP: Improving performance in high speed wide area networks. In Proceedings of PFLDnet, Geneva, Switzerland, Feb. 2003.

[4] 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.

[5] I. Rhee and L. Xu. CUBIC: A new TCP-friendly high-speed TCP variant. In Proceedings of PFLDnet, Lyon, France, Feb. 2005.

[6] D. Chiu and R. Jain, “Analysis of the increase and decrease algorithms for congestion avoidance in computer networks,” Compute Networks and ISDN Systems, vol. 17, pp. 1–14, Jun. 1989.

[7] Michele C. Weigle, Pankaj Sharma, Jesse R. Freeman IV: Performance of Competing High-Speed TCP Flows. Networking 2006: 476-487

[8] Yee-Ting Li, Douglas J. Leith, Robert Shorten: Experimental evaluation of TCP protocols for high-speed networks. IEEE/ACM Trans. Netw. 15(5): 1109-1122 (2007)

[9] B. Even, Y.Li, D.J. Leith, “Evaluating the Performance of TCP Stacksfor High-Speed Networks”, PFLDnet 2006, Nara, Japan.

[10] Yee-Ting Li, Douglas Leith, and Robert Shorten, “Experimental Evaluation of TCP Protocols for High-Speed Networks”, Technial report, Hamilton Institute, 2005.

[11] Martin Bateman, Saleem Bhatti, Greg Bigwood, Devan Rehunathan, Colin Allison, Tristan Henderson. "A Comparison of TCP Behaviour at High Speeds Using ns-2 and Linux.", 11th Communications and Networking Simulation Symposium (CNS 2008) Ottawa, Canada, April 14-17 2008.

[12] Dimitrios Miras, Martin Bateman, Saleem Bhatti. "Fairness of High-Speed TCP Stacks.", The IEEE 22nd International Conference on Advanced Information Networking and Applications. March 25-28, AINA 2008.

[13] L. Xu, K. Harfoush, and I. Rhee, “Binary increase congestion control for Fast, Long Distance Networks", In Proceeding of IEEE, INFOCOM March-2004

[14] J. Rakowska, R. T. Haftka and L. T. Watson. Tracing the Efficient Curve for Multi-Objective Control-Structure Optimization. Computing Systems in Engineering. Vol. 2, No. 6, pp. 461-471, 1991.

[15] J. R. Rao and P. Y. Papalambros. A Non-linear Programming Continuation Strategy for One Parameter Design Optimization Problems. Proceedings of ASME Design Automation Conference, Montreal, Quebec, Canada, Sept. 17-20, 1989, pp. 77-89.

[16] ns-2 http://nsnam.isi.edi/nsnam/

[17] H. Bullot, R. L. Cottrell, and R. Hughes-Jones. Evaluation of advanced TCP stacks on fast long-distance production networks. Journal of Grid Computing, 1(4):345–359, 2003.

[18] D. J.Leith, R.N. Shorten, H-TCP Protocol for High-Speed Long-Distance Networks. Proc. 2nd Workshop on Protocols for Fast Long Distance Networks. Argonne, Canada, 2004.

Rachid Mbarek received his M.Sc. degree in Computer Engineering from the College of Engineering (ENIS), Sfax University, Tunisia, in 2003. He is a Ph.D. student at the same university. He is currently a Lecturer in the Department of Computer Engineering in the College of Computer at Qassim University, Kingdom of Saudi Arabia. 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.

Mohamed Tahar Ben Othman received his Ph.D. in computer science from The National Institute Polytechnic of Grenoble INPG France in 1993, His master degree in computer science in ENSIMAG "École Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble" in 1989. He received a degree of Senior Engineer diploma in computer science from faculty of science of Tunis. This author became a Member (M) of IEEE in 1997, and a Senior Member (SM) in 2007. He worked as post-doc researcher in LGI (Laboratoire de Genie Logiciel) in Grenoble France between 1993 and 1995, dean of the faculty of science and engineering between 1995 and 1997 at the university of science and technology in Sanaa-Yemen, as Senor Software Engineer in Nortel Networks between 1998 and 2001 and Assistant professor in computer college at Qassim University in Saudi Arabia from 2002 until present. His research interests are wireless networks, Adhoc networks, communication protocols, and bioinformatics.

Salem Nasri received his PhD in Automatic Control and Computer Engineering from 'INSA: Institut National des Sciences Appliquées' Toulouse, France, in June 1985. He obtained the diploma of "HDR: Habilitation à Diriger les Recherches" in Computer Engineering, in May 2001 from the 'ENIS: Ecole Nationale d'Ingénieurs de Sfax', Tunisia. He has been working as Associate Professor at 'ENIM: Ecole Nationale d'Ingénieurs de Monastir', Tunisia. Currently, he is Professor at Computer College, Qassim University, Kingdom of Saudi Arabia. His research interests are: Computer Networks, Protocols, Wireless, Communication Systems and Multimedia Applications.