Testing TCP Westwood+ over Transatlantic Links at 10 Gigabit/Second rate

Saverio Mascolo and Giuseppe Racanelli

Abstract— Recent introduction of 10 Gigabit Routers and 10 Gigabit Ethernet cards makes of great interest the issue of designing and testing new protocols capable of efficient utilization of 10 gigabit Internet paths. In this work we report a first investigation of the performance of Westwood+ TCP over large delay paths at 10 gigabit/second rates. The investigation has been conducted at the CERN–IT division using the implementation of Westwood+ we have recently made available in the Linux 2.6 stack. The main feature of Westwood+ TCP is its adaptive setting of the control windows after a congestion episode obtained by estimating the bandwidth available along the connection path. The "Westwood+ feature" has been proven to be particularly effective over lossy channels, where packet losses are not due to congestion, and to improve fairness. Experiment results reported in this work show that remarkable throughput and fairness improvements are also present in the presence of very high bandwidth delay product paths. In the future, we plan to combine the way Westwood+ sets the control window after congestion with modifications of standard TCP probing phase. A first try in this direction is also reported in this paper.

***Index Terms*—10 Gigabit/second rates; TCP/IP congestion control, TCP Westwood+**

I. Introduction

The TCP/IP congestion control used in current Operating Systems [1], from now on we call it Standard TCP, provides a long-term throughput T that can be approximated using the following formula [2],[3]:

$$T = \frac{w}{R} = \frac{1.2}{R\sqrt{p}} \qquad (1)$$

where w is the average congestion window in packets, R is the average round trip time in seconds, p is the average packet loss rate.

The formula (1) sets a fundamental limitation for the TCP. In fact, Eq. 1 states that, given a round trip time R, the throughput that a TCP flow can achieve is proportional to w, which in turn is proportional to 1/sqrt(p). This means that to fill a high speed path with bandwidth B it is necessary to open a congestion window

$$w = B \cdot R = \frac{1.2}{\sqrt{p}}$$

which requires a packet loss probability

$$p = \left(\frac{1.2}{B \cdot R}\right)^2 \qquad (2)$$

Eq. 2 states that, in order to obtain full utilization of bandwidth B, a lower and lower p is required with increasing B. To give a further insight into Eq. 2, Sally Floyd points out in [4] that a Standard TCP connection with 1500-byte packets and a 100*ms* round-trip time would require an average congestion window of 83,333 segments to achieve a steady-state throughput of 10 Gbps in the presence of a packet drop rate of at most one loss event every 5,000,000 packets. The average packet drop rate of at most $2 \cdot 10^{-10}$, which is needed for full link utilization in this scenario, corresponds to a bit error rate of at most $2 \cdot 10^{-14}$, which is unrealistic for current networks [4].

For these considerations, the main concept of Westwood+, which consists of shrinking the control windows after congestion by taking into account an estimate of the available bandwidth, is valuable of investigation in the context of very high speed networks.

II. Brief description of Westwood+ TCP

The main idea of Westwood+ TCP is to set the control windows after congestion such that the bandwidth available at the time of congestion is exactly matched [5],[6].

The available bandwidth is estimated by properly counting and averaging the stream of returning ACK packets. In particular, when three DUPACKs are received, both the congestion window (*cwnd*) and the slow start threshold (*ssthresh*) are set equal to the estimated bandwidth (*BWE*) times the minimum measured round trip time ($RTT_{\min}$); when a coarse timeout expires the *ssthresh* is set as before while

This work was supported by the Datatag Project and the IT division of CERN (Switzerland).

Saverio Mascolo, DEE Politecnico di Bari, 70125 Bari, Italy ,Tel. +39-080-5963621, Fax +39-080-5963410, mascolo@poliba.it.

Giuseppe Racanelli (gracanelli@aliceposta.it), summer student at CERN IT division when doing experiments.

the *cwnd* is set equal to one.

The pseudo code of the Westwood+ algorithm is as simple as reported below:

```
a) On ACK reception:
    Increase cwnd accordingly to the Reno algorithm;
    Estimate the available bandwidth (BWE);

b) When 3 DUPACKs are received:
    ssthresh =max(2, (BWE* $RTT_{min}$) / seg_size);
    cwnd = ssthresh;

c) When coarse timeout expires:
    ssthresh = max(2,(BWE* $RTT_{min}$) / seg_size);
    cwnd = 1;
```

In words, when ACKs are received, Westwood+ additively increases the cwnd as standard Reno or New Reno does; when a congestion episode happens, Westwood+ employs an adaptive setting of *cwnd* and *ssthresh* that takes into account the available bandwidth instead of implementing the “blind” by half window reduction of standard TCP.

It is worth noting that the *adaptive decrease* mechanism employed by Westwood+ TCP improves the stability and the utilization of the standard TCP multiplicative decrease algorithm. In fact, the adaptive window setting provides a congestion window that is decreased more in the presence of heavy congestion and less in the presence of light congestion or losses that are not due to congestion, such as in the case of losses due to unreliable links. In another way, the setting $cwnd = B*RTT_{\min}$ sustains a transmission rate $(cwnd/RTT) = (B*RTT_{\min})/\mathrm{RTT}$ that is smaller than the bandwidth *B* estimated at the time of congestion: as a consequence, the Westwood+ TCP flow clears out its path backlog after the setting thus leaving buffer room available to coexisting flows, which improves statistical multiplexing and fairness [5].

III. EXPERIMENTAL RESULTS

The experimental scenario consists of the very high speed path from CERN (Geneve, Switzerland) to CalTech (Los Angeles, California) that is shown in Fig. 1. We have considered two sender stations located at the CERN side, and two receiver stations at the Caltech side. The path consists of the following links: Geneva-Chicago (LHCnet/Datatag, 7067 km); Chicago-Indianapolis (Abilene, 263 km); Indianapolis-Kansas City (Abilene, 727 km); Kansas City-Sunnyvale (Abilene, 2403 km); Sunnyvale-Los Angeles (Abilene, 489 km) for a total of 10949 km (distances were measured by Virtual GPS). In all tests we use jumbo frames, i.e. the packet size is set to 9000 bytes.

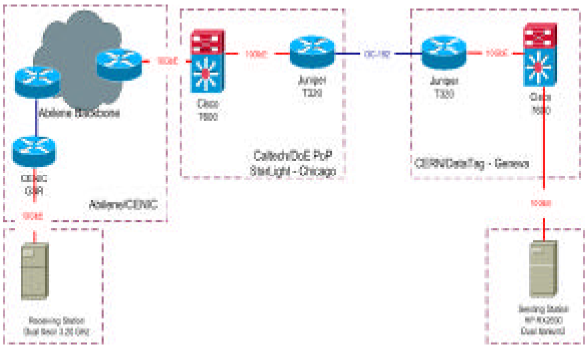

Fig. 1: Experimental testbed

3.1 Single Stream Tests

We start by considering a single TCP persistent stream going from the sender station at CERN to the receiver station at Caltech. Fig. 2 shows the congestion window and the slow-start threshold dynamics of a New Reno TCP connection. The measured round trip time was 265ms. Fig. 2 shows that at t=180s the *cwnd* reduces from 2.5*10^8 bytes to 2.7*10^7 bytes and the TCP enters the congestion avoidance phase. To increase the congestion window from 2.7*10^7 bytes to 2.5 *10^8 bytes, that corresponds to increase the rate from 820 Mbps to 7.5 Gbps, the TCP would take almost 2 hours under the unrealistic assumption of no losses during two hours!

Fig. 3 shows the throughput along with its average: in particular the average throughput is around 1.8Gbps, which is less than one fifth of the channel capacity.

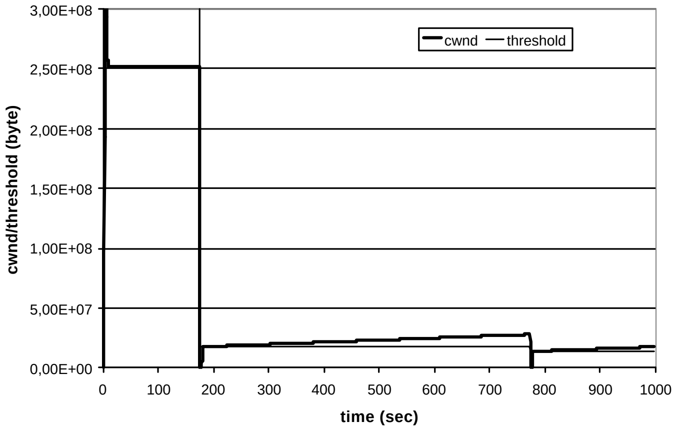

Fig. 2: Cwnd and ssthresh behavior of New Reno TCP

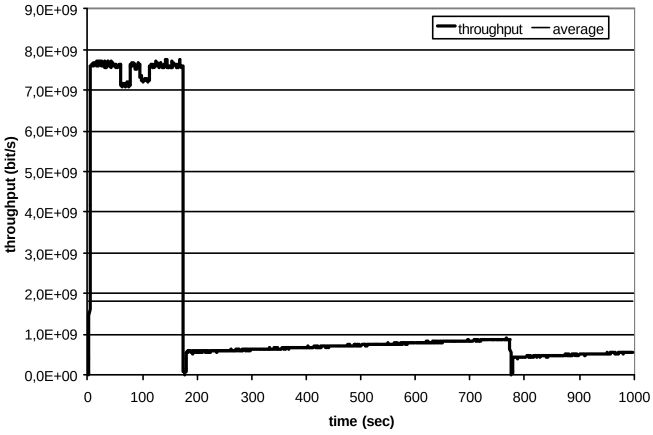

Fig. 3: Instantaneous and mean throughput of NewReno TCP

Fig. 4 shows the *cwnd* and *ssthresh* dynamics obtained in the same scenario using Westwood+ TCP. In this case, the *cwnd* after congestion reduces from 2.5*10^8 bytes to 2.3*10^8 bytes, which is remarkably larger than the corresponding value obtained using New Reno. Fig. 5 shows that the achieved throughput is now around 7 Gbps.

Now we investigate what happens when congestion is provoked by turning on, for few seconds, an UDP flow at 5Gbps going from the second sender station at CERN to the second receiver station at Caltech. In this case, Fig. 6 shows that the slow start threshold is set to 3.5*10^7 bytes after congestion and, again, it takes a long time for the TCP in congestion phase to grab all the bandwidth available after the UDP is turned off. Fig. 7 shows that around one tenth of the available bandwidth (i.e. 1.2 Gbps) is achieved.

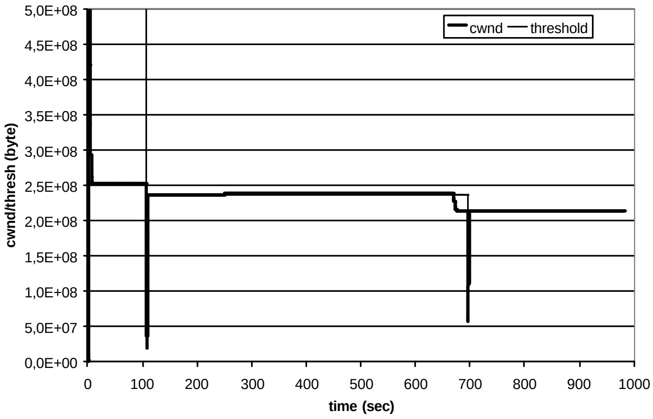

Fig. 4: Cwnd and ssthresh behavior of Westwood+ TCP

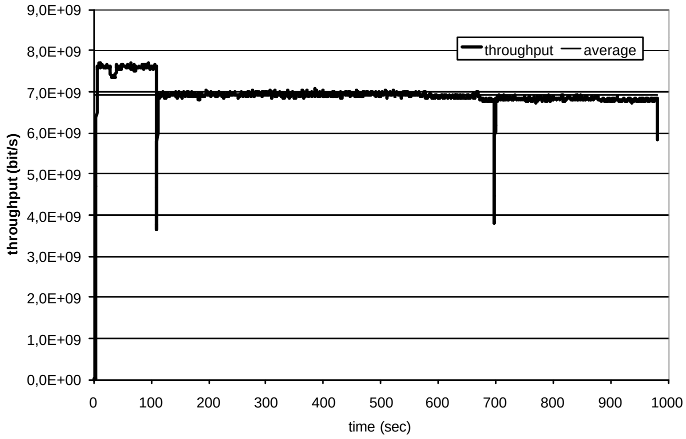

Fig. 5: Instantaneous and mean throughput of Westwood+ TCP

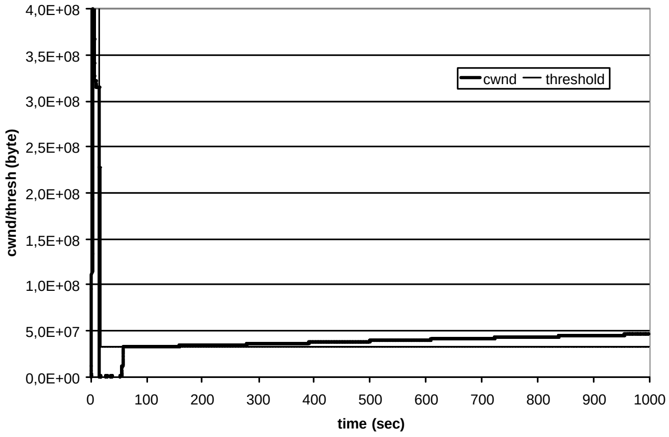

Figure 6: Cwnd and ssthresh of Westwood+ TCP when injecting an UDP stream at 5Gbps

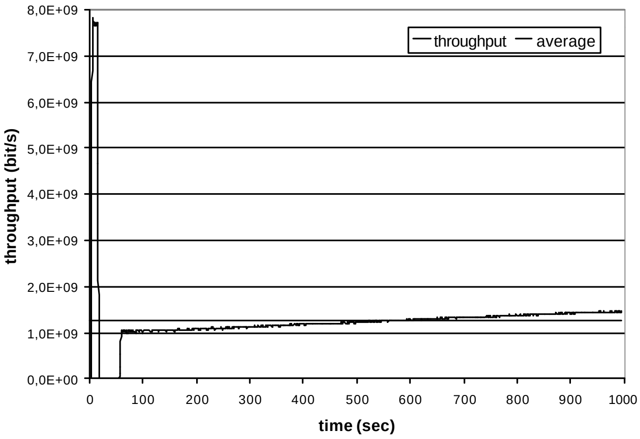

Figure 7: Throughput of Westwood+ in the case of heavy congestion

In order to overcome the problem that the congestion avoidance phase is too slow in very high-speed networks, we make a first attempt to modify the congestion avoidance phase of Westwood+ using, for instance, a slightly modified version of Scalable TCP [15].

The idea of Scalable TCP is to introduce a multiplicative increase phase instead of the standard TCP additive increase phase. In this paper we implement an increasing phase that is the same as the one used by standard TCP up to reach a congestion window equal to *window_threshold*. Here, we use a *window_threshold* of 100 packets that corresponds to sustain a rate equal to 27Mbps over an RTT of 260ms. When the congestion window reaches the *window_threshold*, we implement a multiplicative increase phase *à la Scalable* TCP with coefficient 1.04. The pseudo code is as follows:

```
• on ACK reception;
    If  ssthresh < = cwnd < window_threshold
        cwnd=cwnd+1/cwnd;

    If  cwnd> window_threshold
        cwnd=cwnd+0.04
```

By increasing *cwnd* of 0.04 on every ack reception, we get a congestion window that increases of one twenty-fifth per *RTT*, i.e., the growth is greater with larger windows. For values of *cwnd* less than 100 segments, the standard TCP probing phase is used.

Fig. 8 shows the behavior of the TCP in the same scenario of Fig. 6 and 7 using Westwood+ TCP with the modified probing phase above described. In this case, even though the setting of the threshold is below the network capacity, the congestion window quickly increases and provides good results in terms of average throughput, which jumps from 1.3 Gbps (see Fig.7) to 6.2Gbps (see Fig. 9).

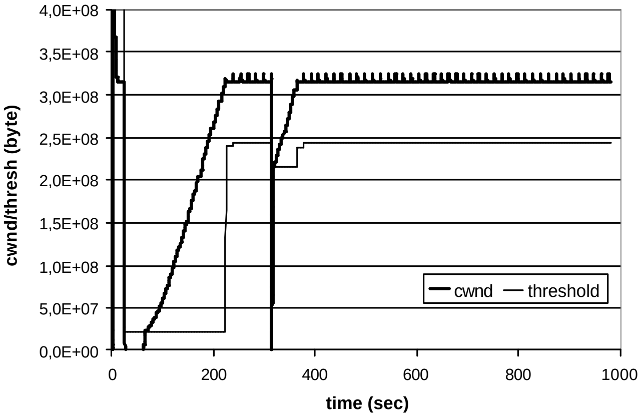

Figure 8: cwnd and ssthresh of Westwood+ TCP using a modified congestion avoidance phase

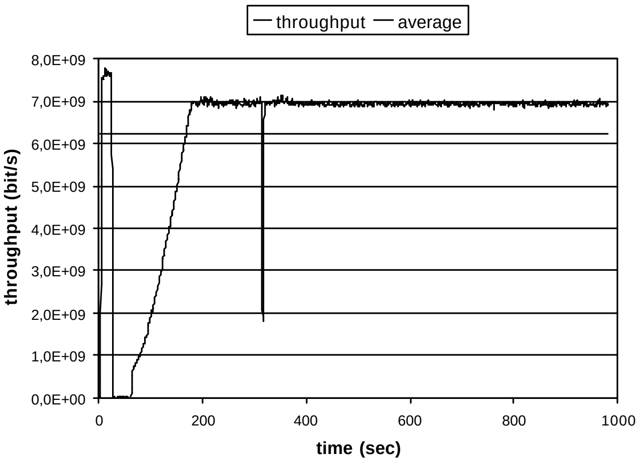

Fig. 9: Throughput of Westwood+ TCP using a modified probing phase

3.2 MULTIPLE STREAM TESTS

When proposing a new protocol it is important not only to investigate its ability to fully utilize the available bandwidth but also to investigate its fairness in bandwidth utilization when different flows share the same bottleneck. To this purpose we investigate how three different streams share a 1 Gbps bottleneck link. We have considered 3 NewReno flows and 3 Westwood+ TCP (i.e. the original Westwood+ TCP flow). Fig. 10 shows the network testbed used in this case. It consists of a 10 Gbps connection going from Geneva to Chicago. The link between the Cisco router 7606 at Geneva and the Extreme router s01gva is a 1 Gbps link.

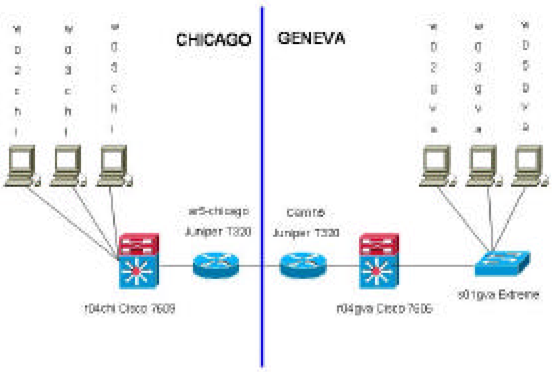

Fig. 10: Network scenario with multiple streams

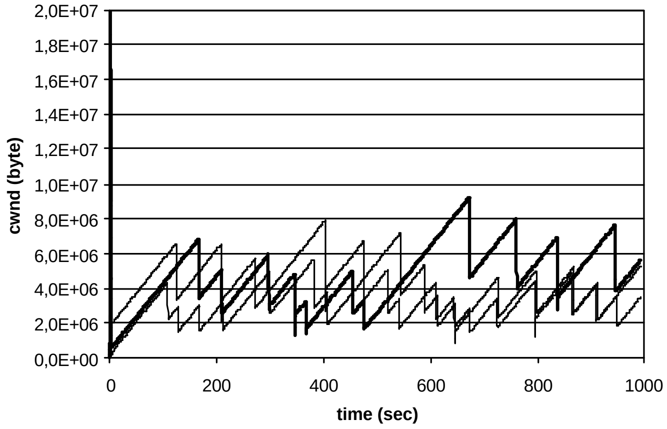

Figure 11: cwnd of the 3 New Reno flows

Fig. 11 shows the *cwnd* behaviour in the case of 3 NewReno flows, whereas Fig. 12 shows the *cwnd* in the case of 3 Westwood+ flows. New Reno flows exhibit the classic “sawtooth” oscillatory behaviour of the *cwnd*, which is due to the by half window reduction. On the other hand, it is very interesting to note that the *cwnd* of Westwood+ exhibits an oscillation free behaviour (the congestion window is kept around the same value of 5*10^06 byte during all the test). Throughputs in the case of New Reno and Westwood are shown in Fig. 13 and 14, respectively. The average per-connection throughput in the case of NewReno is 270 Mbps, whereas the average per-connection throughput in the case of Westwood+ is 320 Mbps.

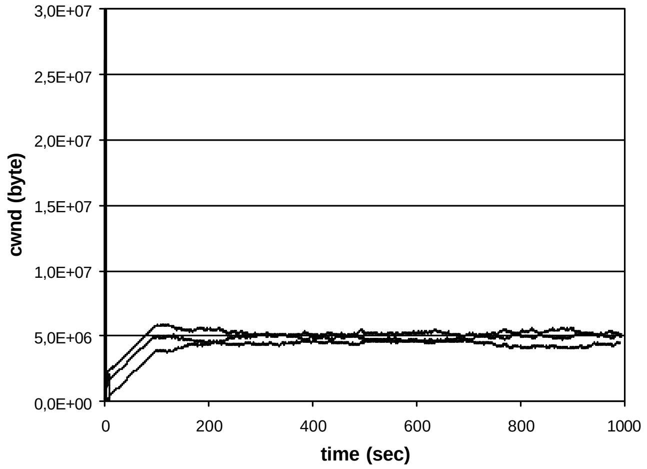

Fig. 12: Cwnd dynamics of 3 Westwood+ streams

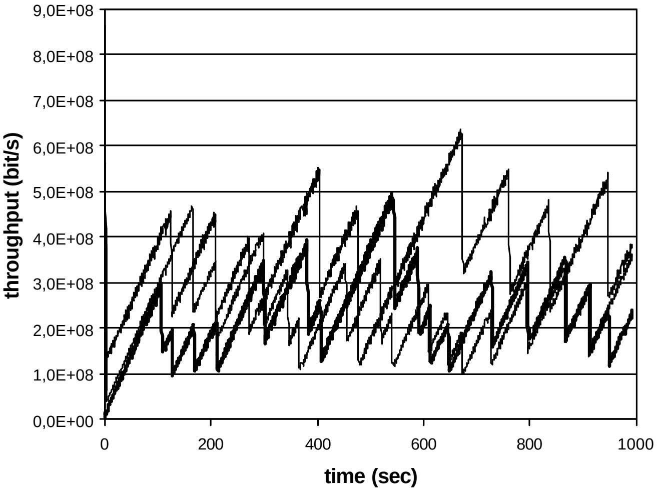

Fig. 13: Throughput in case of 3 new Reno flows

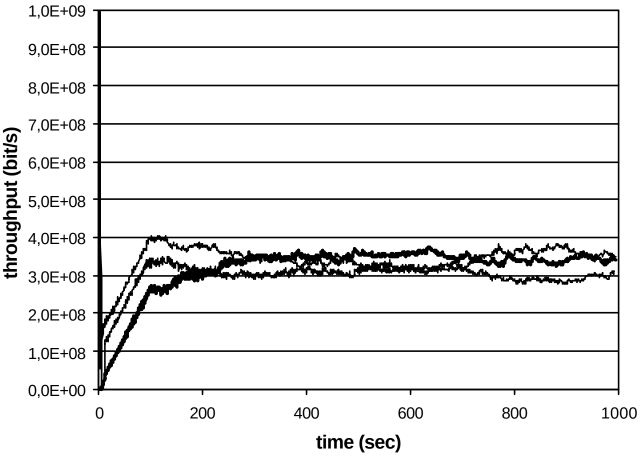

Fig:14 : Throughput in the case of 3 Westwood+ streams

In order to provide a mathematical evaluation of the fairness, we plot the dynamics of the Jain fairness index defined as below:

$$J_{FI}(t) = \frac{\left(\sum_1^M b_i(t)\right)^2}{M\sum_i^M b_i(t)^2}$$

where $b_i(t)$ is the instantaneous throughput of the *ith* connection and M is the number of connections sharing the bottleneck. The Jain fairness index belongs to the interval [0,1] and increases with fairness up to the value of one.

Fig. 15 and 16 shows the dynamics of Jain fairness index obtained in the case of 3 NewReno connections and 3 Westwood+ TCP connections. The fairness index of NewReno TCP oscillates between 1 and 0.8 with an average value around 0.9. On the other hand, the fairness index of the 3 Westwood+ TCP flows reaches a steady state value of 1 after an initial short transient.

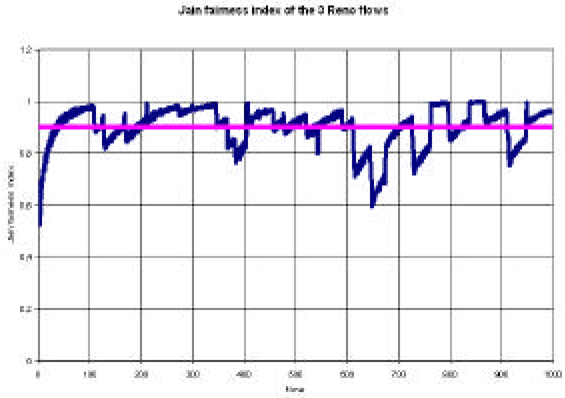

Fig. 15 Jain Fairness Index of the three New Reno flows

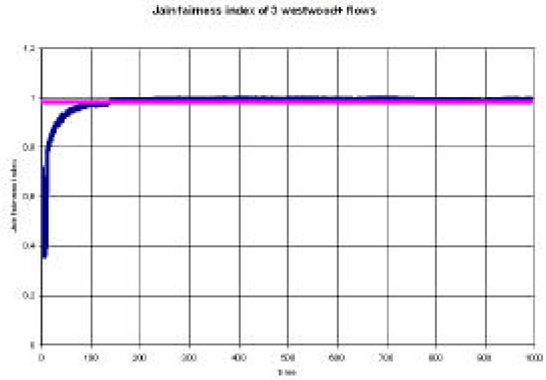

Fig. 16 Jain Fairness Index of the 3 Westwood+ flows

IV. Conclusions and Future Work

Measurements on the Datatag testbed have shown that Weswtood+TCP provides significant throughput and fairness improvement with respect to standard NewReno TCP. They have also shown that further significant improvements can be achieved by modifying the Westwood+ probing phase. We plan to design and investigate new modifications of the TCP probing phase used by Westwood+ TCP and compare this modified version of Westwood+ with Scalable TCP and HS-TCP [4].

Acknowledgments

We would like to thank Olivier Martin at the IT division of CERN and all the CS group, namely: Sylvain Ravot and Dan Nae (from Caltech), Paolo Moroni, Edoardo Martelli for their great support and for allowing us to collect the measurements reported in this paper.

References

[1] V. Jacobson, “Congestion Avoidance and Control,” ACM Computer Communications Review, 18(4): 314 - 329, August 1988.

[2] F. P. Kelly, “Mathematical Modeling of the Internet,” Proc. 4th International Congress on Industrial and Applied Mathematics, July 1999.

[3] J. Padhye and V. Firoiu and D. Towsley and J. Kurose", "Modeling TCP Throughput: A Simple Model and its Empirical Validation", Proc. ACM Sigcomm 1998, pp. 303-314.

[4] S. Floyd, “HighSpeed TCP for Large Congestion Windows”, IETF Internet Draft draft-ietf-tsvwg-highspeed-00.txt, August 2004.

[5] L. A. Grieco, S. Mascolo “Performance evaluation and comparison of Westwood+, Vegas and New Reno TCP congestion control”, ACM Computer Communication Review, April 2004.

[6] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, R. Wang, “TCP Westwood: End-to-End Bandwidth Estimation for Efficient Transport over Wired and Wireless Networks”, ACM Mobicom 2001, July, Rome, Italy.

[7] S. Mascolo, “Congestion control in high-speed communication networks using the Smith principle”, Automatica, vol. 35, no. 12, dec. 1999.

[8] Tom Dunigan, URL at http://www.csm.ornl.gov/~dunigan/net100/

[9] URL at http://netlab.caltech.edu/FAST

[10] The Web100 project. URL "http://www.web100.org/".

[11] The Westwood+ TCP project. URL “http://www-ictserv.poliba.it/mascolo/tcp%20westwood/tcpwestwood.htm

[12] V. Jacobson, R. Braden, D. Borman, “ TCP Extensions for High Performance ”, RFC 1323, May 1992.

[13] Hoe, J., C., “Improving the Start-up Behavior of a Congestion Control Scheme for TCP,” Proc. of ACM Sigcomm'96, pp. 270-280.

[14] Villamizar, C. and Song C. (1995), “High Performance TCP in ANSNET”, ACM Computer Communication Review, vol. 24, no. 5, pp. 45-60.

[15] T. Kelly, “Scalable TCP: Improving Performance in High speed Wide Area Networks”, ACM Computer Communication Review, Vol. 33 No. 2 - April 2003.

[16] Datatag Project. URL http://datatag.web.cern.ch/datatag/.