Shareable Bandwidth Estimation on Satellite Links

Claudio E. Palazzi
Dipartimento di Matematica Pura e Applicata,
Università di Padova,
Via Trieste 63,
35131 Padova, Italy
cpalazzi@math.unipd.it

Giovanni Pau, Cesar Marcondes, Mario Gerla
Department of Computer Science,
University of California, Los Angeles
3803A Boelter Hall, 420 Westwood Plaza
Los Angeles, CA, USA 90095-1596
*{gpau, cesar, gerla}@*cs.ucla.edu

***Abstract*—Satellite networks have the potentiality to guarantee ubiquitous access to the Internet. Yet, their transmission delay and errors represent still a challenge for researchers and practitioners, especially when considering TCP-based flows. Having efficient means to estimate important channel features such as, for instance, the available bandwidth and the total capacity has been proven to be proficient in improving the performance in this context. To this aim we discuss SBE (Shareable Bandwidth Estimator), a solution able to provide a simple and effective estimation of the bottleneck link capacity minus the simultaneously present uniformly distributed traffic. Simulative experiments prove that our scheme is effective since the very beginning of a connection. Furthermore, SBE can be easily implemented as it requires only sender-side modifications and is based on regular TCP functioning.**

Keywords- Capacity Estimation, Satellite, TCP, Wireless.

I. INTRODUCTION

Satellite networks represent a research topic that has recently gained a tremendous interest. The main reason is in the need for anytime and anywhere pervasive access to the Internet that can be satisfied via satellite communication.

A satellite network can serve as part of the Internet backbone, as a high-speed access network, or as both; moreover, it can be based on diverse orbit types (GEO, MEO, LEO). However, all these different satellite networks share similar challenging research issues. For instance, a lot of effort is currently devoted to improve the performance of TCP (Transmission Control Protocol) in this context [1][2].

Indeed, due to its legacy, TCP will probably not be discarded in favor of a completely new transport protocol, even considering new scenarios that were clearly not imagined forty years ago, when TCP was designed. Therefore, even the satellite-based Internet is expected to continue to serve many applications based on TCP. However, current TCP protocols have low performances in satellite networks due to long round trip time (RTT), high link error rates, and asymmetric link bandwidth [3][4][5].

To address the aforementioned issues, several solutions have been proposed, which span from defining a set of window scaling options to estimating the eligible bandwidth or the total capacity of the link [6][7][8][9]. Exploiting information about the available bandwidth/capacity is definitely an interesting approach as it allows to discriminate between error and congestion losses, thus halving the congestion window (i.e., the data sending rate) only when necessary.

A tool to provide information about the available bandwidth or the total capacity on a connection needs to possess three main properties to be really proficient. First, it has to be accurate enough to provide information that can be factually exploited to improve the performance of the system. Second, it has to be fast in generating a useful estimate within few RTTs, otherwise benefits will be very limited. Indeed, it is more important to have a fairly accurate information in a very short time than to have a very precise one after few seconds. Finally, an estimator has to be unobtrusive; if resorting to probing packets, this transmission load should be kept as little as possible.

To this aim, we discuss here a new estimator, named *Shareable Bandwidth Estimator (SBE)*, which possesses all the aforementioned properties. Moreover, SBE is different from previously proposed bandwidth/capacity estimators in that it provides a different kind of information. This new information is the capacity of the bottleneck link detracted the traffic that is continuously present on that link. SBE is a very simple algorithm that recalls the packet train mechanism, but it is perfectly embedded within the normal functionalities of traditional TCPs; in fact, it operates by just observing the regular flow of TCP packets and ACKs (acknowledgments), without generating any transmission overhead. Furthermore, SBE provides accurate information very quickly, even at the very beginning of a TCP session; whereas other estimators proposed in literature needs considerably longer working periods [6][10].

The information produced by SBE could be used to appropriately set the congestion window of a TCP session running on a satellite link to prolong the slow start phase or to discriminate between congestion and error losses. Moreover, SBE could be coupled with other existing bandwidth/capacity estimators to refine their estimations, especially at the beginning of a TCP session.

The rest of the paper is organized as follows. In Section II, we discuss scientific background related to the bandwidth/capacity estimation. Section III is devoted to the explanation of SBE. Section IV presents achieved results from our simulative testbed. Finally, in Section V, conclusions are drawn.

II. BACKGROUND

Estimating the available bandwidth or the factual capacity of a connection represents a topic that has been discussed for several years; yet, it still attracts researchers all over the world for the many implications with current research issues. For instance, one of the main research fields that employs bandwidth/capacity estimators is represented by wireless networking (e.g., satellite networks). Indeed, efficient communications in this context, especially when considering TCP-based transmissions, are challenged by technical obstacles that could be overcome with the employment of effective estimators of channel features such as the available bandwidth and the total capacity [4]. Equipped with these information, protocols become able to set the most appropriate data sending parameters decoupled from the wireless losses.

Most of the proposed solutions are either based on packet trains or on packet pairs. In the first case, a group of packets, i.e., a packet train, is sent one packet after the other between two nodes. [11][12][13][14][15]. The classic approach of this kind of solution is that of self-inducing congestion; this is accomplished through sending packets at an increasing rate until this rate surpasses the available bandwidth along the path. At this point, the receiving rate at destination will be inferior to the sending one and this receiving rate actually corresponds to the available bandwidth. Clearly, this approach is not unobtrusive. Furthermore, solutions belonging to this class have generally been designed for wired networks and fail when employed in a wireless scenario [16][17][18].

Instead, the second class of solutions employs a pair of packets that is sent between two nodes and their inter-arrival time at destination is used as a metric for measuring the capacity of the bottleneck link. Yet, it is well known that if one of the two packets encounters along the path a different queue length with respect to the other packet in the pair, then the resulting estimation will be wrong [7][19][20].

In the following, we discuss some representative examples of practical use of bandwidth and capacity estimators.

An end-to-end transport protocol making use of a channel estimator is proposed in [21], where the high number of errors and the variable latency of a wireless environment is faced through using periodic SACK (Selective Acknowledgment) packets to understand when a retransmission is appropriate [22] and through estimating the channel's bandwidth to set the data sending rate. In the proposed protocol, the departure time is included in each packet, thus the receiver can use this information and the packet inter-arrival time to measure the bandwidth; the most appropriate sending rate is then communicated back to the sender. Unfortunately, the modifications required on both sender and receiver side limits the development of this protocol (and of its estimator).

To support TCP's performance in wireless environments, [6] suggests to use a sender side, end-to-end estimation of the available bandwidth. The idea is to use the rate of the returning ACKs to measure the effective link availability. This bandwidth estimation is computed by sampling and filtering methods that have been progressively refined in order to be effective and fair at the same time and then used, after a loss or during slow start to appropriately set the slow start threshold [23][24]. This technique has shown great results on big pipes affected by error losses such as, for instance, satellite links. However, by simply using the returning ACKs of regular TCP flows, the developed estimator computes the available bandwidth as the latest effective data transmission rate; in other words, only after the transmission rate has reached the actual available bandwidth, the estimator is able to determine the its value.

Focusing on capacity estimators, [25] proposes the use of special pairs of probe packets after each loss to measure this value; when one of these couples reaches the receiver, the measured delay of the probing packets is used to determine whether the link was congested when the loss happened, or was experiencing disconnection/fading. In the former case it lowers the data sending rate, whereas in the latter case it temporarily freezes transmissions to avoid further losses and erroneous restrictions of the congestion window. Unfortunately, this protocol requires modification at both sender and receiver side in order to handle probing packets: this lack of compatibility with the current implementation of the TCP seriously affects its real deployment possibility.

Similar, [26] makes use of packet pairs to measure the capacity of the bottleneck link. In case of multimodal distribution, the number of packets used is increased until having a cardinality of the packet train that produces a unimodal distribution. This scheme results to be accurate but computationally slow thus limiting its utilization in scenarios involving satellite links.

Finally, a very interesting way to employ packet pair techniques to estimate the capacity of a bottleneck link is suggested by [7]. Specifically, various packet pairs are sent between two nodes; then, among all these pairs, one is chosen that has the "minimum delay sum" (the sum of the two transmission delays for the two packets composing the pair). Indeed, this value, is minimum as it has not suffered by cross traffic, thus maximizing the accuracy of the packet pair technique. The provided estimation results hence very close to the real channel capacity but, on the other hand, packet pair techniques suffer the bulk nature of TCP transmissions and the use of delayed ACKs if embedded within TCP's operations.

III. SBE: RATIONALE OF THE IDEA AND ALGORITHM

A packet train is a set of packets which depart from the sender, one close to the other. Their leaving time is beaten by the current transmission rate and depends on the outgoing capacity. Along the path toward the destination, a packet train can encounter links with smaller bandwidth; consequently, packets will require more time to be transmitted and the train becomes longer. This dispersion could be caused not only by narrow links, but also by the time spent in queue due to other traffic sharing part of the same connection. The corresponding ACKs go back to the sender, triggering new transmissions at a rate that depends on all these factors and causes packets coming out separated by gaps of idle time.

In Fig. 1, the time is divided into slots; using these slots as the measurement units, we call X the part used to transmit the packets back to back, and Y the time needed to contain the dispersion of the corresponding ACKs. As it is evident, the maximum portion of the slot that is usable to send packets

corresponds to X / Y. The rate at which the sender could transmit is given by

$$Bits\_transmitted / X; \quad (1)$$

whereas the maximum rate usable, given the bottleneck, is

$$Bits\_transmitted / Y. \quad (2)$$

In a situation with a single flow and having the packets leaving the sender back to back, (2) easily determines the capacity of the connection. If we introduce in this scenario other traffic, we also have to take into account the possibility that packets could leave the source distributed over a wider portion of the slot (see Fig. 2), or even over the whole slot. In fact, the effective transmitting time of the sender will probably be divided into several chunks of time, having gaps in between them that remain unused. Moreover, other time gaps are inserted just by the fact of having a TCP congestion window smaller than the actual pipe size. In those pauses, the sender just waits for returning ACKs in order to be allowed to transmit new data. Consequently, the corresponding ACK distribution is characterized by gaps that does not depend on the bottleneck size. In this situation, (2) cannot be used anymore to measure features of the connection such as the capacity.

Instead, we designed SBE to eliminate the wasted time due to a low sending window maintaining, at the same time, the dispersion of the packets due to links with different total bandwidth along the path. Details on the algorithm follow.

Time is divided into slots: we set packets-slots as large as an RTO [27] and count packets leaving in that period, we then wait for corresponding returning ACKs, these will form an ACKs-slot. Since the number of packets-slots and of ACKs-slots is the same, on average they will have the same length.

When the ACKs corresponding to sent packets come back, we compute the following formula

$$SBE = Bits\_ACKed / (ACKs\_slot\_time - Wasted\_time). \quad (3)$$

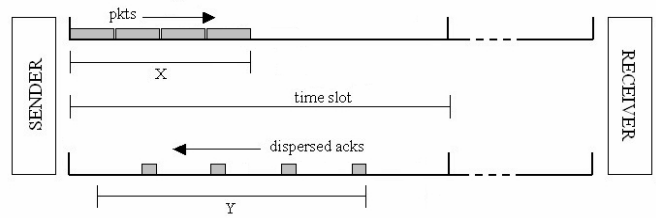

Figure 1. ACK dispersion due to the bottleneck along the path.

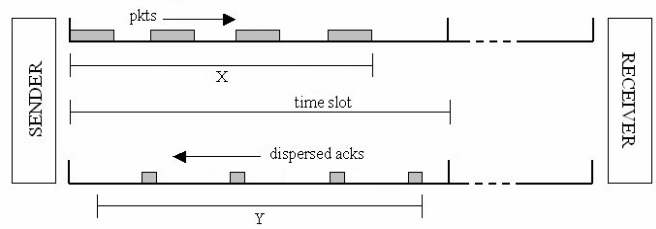

Figure 2. Packet and ACK dispersion due to other traffic along the path.

With SBE, we utilize the dispersion of the ACKs, the source ratio and the information regarding the time wasted with no data sent toward the destination due to a low congestion window. In particular, the last element is the enhancement introduced by our scheme that allows to have an accurate estimation since the very first period of a connection. Instead, this sender-side wasted time, potentially affects the effectiveness of all the bandwidth or capacity estimators that relies on returning ACKs.

Following the aim of considering just the wasted time due to a little sending window while maintaining the dispersion induced by the narrower links, we have to calculate the *Wasted_time* from the ACKs-slot standpoint. Our proposed scheme measures the average of the inter-arrival time between any two successive ACKs in the slot (the first one is counted from the beginning of the slot and the slot ends receiving the last ACK belonging to it); the *Wasted_time* is then computed as the sum of the time exceeding this average in each ACK inter-arrival time of the ACKs-slot. This formula is justified by the fact that all packets within the same slot will generally experience similar channel conditions in terms of transmission time; consequently, any gap between ACKs that exceeds the average inter-arrival time is most likely a result of having experienced a period of no transmission due to a small sending window size.

Focusing on the queuing time, we have to notice that, since the bulk nature of TCP transmission, this element will not be endured equally by all packets in a slot. This difference in time spent in queue will produce ACKs with higher inter-arrival time than the average. Consequently, the queuing time is removed by our mechanism in case of contemporary presence of other TCP flows and the final result is the capacity of the bottleneck. Conversely, if the considered TCP connection competes for the channel with other CBR (Constant Bit Rate) flows, the outcome of SBE has a different meaning. Indeed, the distribution shape of a CBR transmission uniformly occupies a portion of space both in the channel and in the queues. Since all the inter-arrival times of the ACKs are in this case equally affected by delays caused by concurrent CBR traffic, our mechanism leaves these delays in the average of the inter-arrival time. Therefore, the final estimation will count also inter-packet delays caused by the CBR traffic computing the shared bandwidth. Summarizing, we can say that SBE returns the bottleneck capacity, detracted the portion of channel continuously occupied by some traffic.

IV. Simulative Results

We have verified the correctness of SBE by embedding its algorithm within the functionalities of a TCP variant and running several simulations with NS-2 [28]. To this aim, we have chosen to utilize TCP Westwood because this transport protocol generates an estimation of the eligible rate and is specifically designed for satellite links [6][23][24]. This choice gives us the possibility to compare our estimator with TCP Westwood's ones so as to make emerge the differences among them and the possible combinations in their employment. The general simulative scenario is depicted in Fig. 3; it includes one or more TCP Westwood senders that transmit packets in a certain direction and a CBR flow that can transmit along the

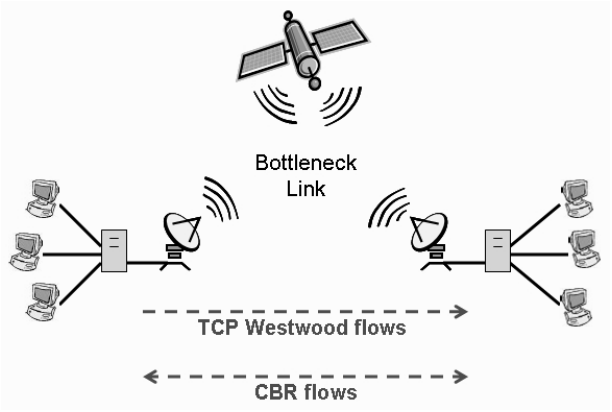

Figure 3. Simulated scenario.

same path of the TCP flows or on the reverse path. Different satellite link capacity and minimum RTT values have been tested; however, the satellite link always represents the bottleneck of the connection. Finally, the buffer size at routers is set as the pipe size (bandwidth-RTT product) of the connection.

A. Evaluation of SBE

First, we have considered a simple scenario involving a LEO satellite link of 5 Mbps. In essence, a single TCP Westwood connection with a minimum RTT of 70 ms runs alone or together with some CBR traffic. Furthermore, transmissions can be affected by errors.

The outcome for the case with a single TCP Westwood flow and no wireless errors is reported in Fig. 4. In particular, the congestion window (*cwnd*), the eligible rate determined by TCP Westwood (*w-ere*), and the computed SBE value (*sbe*) are shown in the chart. Since the presence of a single connection on the channel, TCP Westwood's eligible rate, the total capacity of the link, and our SBE have to coincide. Indeed, this is confirmed by results shown in Fig. 4; in particular the value provided by our SBE oscillates around 43 packets, while the actual pipe size is 44 packets. Moreover, our SBE reaches the correct value almost immediately, anticipating TCP Westwood's estimation of the eligible rate.

We have also evaluated a more complex scenario where the TCP Westwood flow shares the link with some CBR traffic traveling in the same direction with respect to TCP packets' one. Specifically, the scenario involves also an UDP-based CBR flow transmitting 125 B packets every 1 ms. To better appreciate the impact of this CBR traffic on the estimator, we have configured the simulation so as to have it running only for a limited time. In particular, whereas the TCP Westwood flow operates in our simulation from 0 s to 30 s, the CBR flow starts at 8 s and ends at 18 s. Furthermore, we have also added a PER (Packet Error Rate) of 0.1 % to simulate the error prone conditions of a typical wireless environment.

The outcome is shown in Fig. 5 and, as it is evident, the capacity of the channel and the arrival of a uniformly distributed traffic are perfectly detected by our SBE, which decrements its value for the whole duration of the CBR traffic. Results also demonstrate that the presence of errors on the channel does not affect the accuracy of SBE.

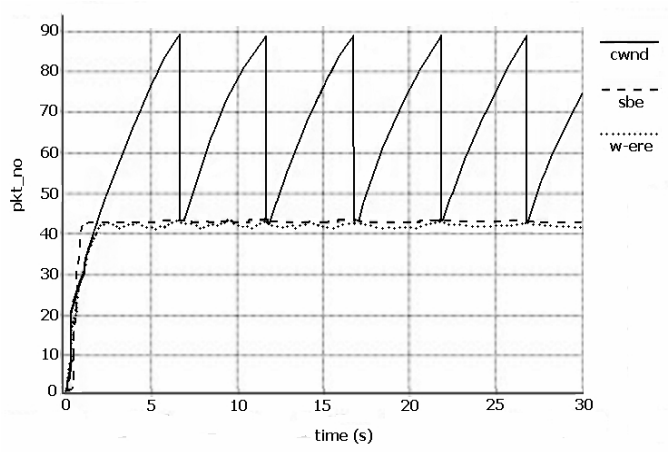

Figure 4. Single TCP Westwood flow on a bottleneck link of 5 Mbps.

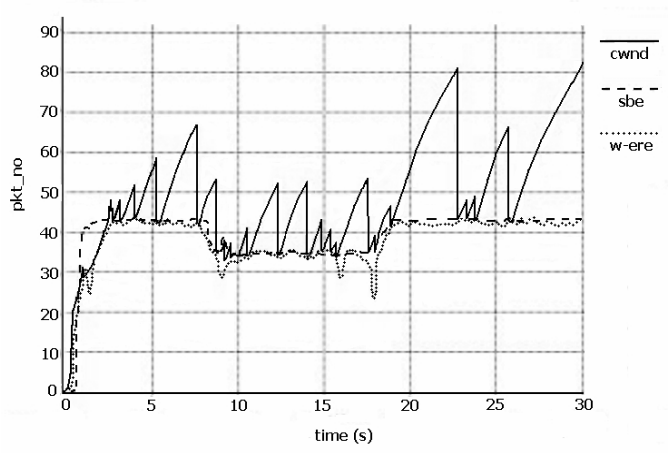

Figure 5. Single TCP Westwood flow on a bottleneck link of 5 Mbps, PER of 0.1 % and a period of CBR traffic (from 8 s to 18 s).

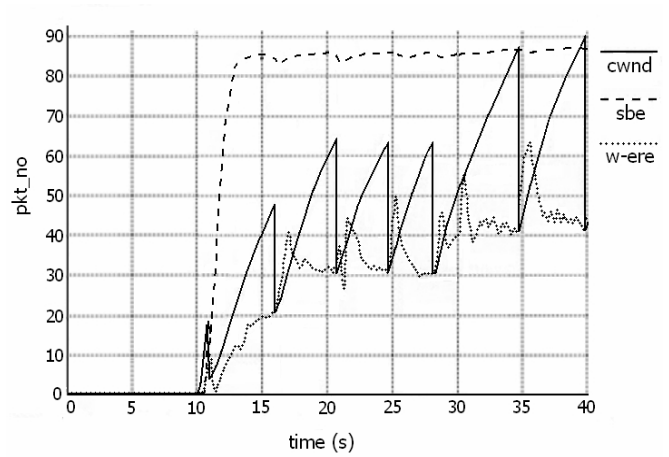

Figure 6. TCP Westwood flow starting at 10 s and ending at 40 s, coexisting with other two TCP Westwood flows (from 0 s to30 s and from 20 s to 40 s, respectively); bottleneck link of 10 Mbps.

Finally, we have also simulated a scenario in which three TCP Westwood flows compete for a common bottleneck of

10 Mbps over a total period of time of 40 s; the RTT is 70 ms, thus the pipe size results to be 88 packets. To study the impact of reverse traffic we have inserted a CBR flow sharing the same bottleneck but transiting in the opposite direction. In this configuration, the first source starts sending packets at 0 s and ends at 30 s, the second one transmits from 10 s to 40 s, and the third one starts at 20 s and finishes at 40 s.

The bulky nature of TCP flows and the reverse CBR flow generate a very little portion of continuously present traffic. Therefore, we expect that SBE will estimate a value very close to the capacity of the bottleneck. Indeed, this happens for all the three flows. For instance, Fig. 6 reports the outcome for the second one. As it is evident, whereas TCP Westwood's eligible rate estimation slowly reaches the correct value and experiences high variations, SBE reaches its value very quickly and is minimally impacted by variations in the packet transmission rate or in the number of flows simultaneously active on the channel. After the initial growth, SBE oscillates between 84.24 and 86.92 packets, which is coherent with the 88 packets of pipe size and the very little traffic that is continuously present.

B. Employment of SBE

Having a fast and accurate means to infer the channel conditions can be exploited in several ways to bring a positive contribution to the achieved performance. We show here just one of the many possible ways to employ our SBE and we evaluate the performance improvement.

As a proof of concept, we have employed SBE to set the initial value of TCP's slow start threshold; this way, the data sending rate of the TCP flow will be free to grow faster on large links. Clearly, the advantages of this solution are more evident when considering the download of small files over links with long transmission delays. Furthermore, we have also used SBE similar to what TCP Westwood does with its eligible rate; practically, SBE value is used to set the slow start threshold after a packet loss. The attempt is that of making TCP performance resilient to wireless error losses.

To test this practical employment of SBE, Fig. 7 reports the time required to a single regular TCP (*TCP_reg*) or to a single TCP exploiting SBE (*TCP_sbe*) to download a file of 1 MB, 2 MB, or 3 MB. The considered scenario involves a LEO satellite link of 5 Mbps and 70 ms of minimum RTT. The chart shows that a regular TCP flow needs from ~29 % (for the 3 MB file) to ~40 % (for the 1 MB file) more time to download those files with respect to a TCP that exploits SBE.

In Fig. 8, the experiment has been replicated but considering a GEO satellite link of 10 Mbps and 500 ms of minimum RTT. As expected, the performance improvement ensured by exploiting SBE are here even more evident as the presence of a very long RTT makes crucial to be able to extend the slow start when channel conditions allow it. The chart shows that a regular TCP flow needs from ~86 % (for the 3 MB file) to ~100 % (for the 1 MB file) more time to download those files with respect to a TCP that exploits SBE.

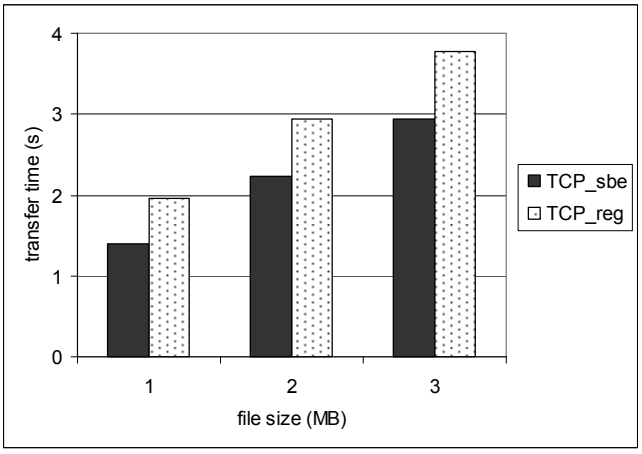

Figure 7. Transmission time for small files; minimum RTT of 70 ms, bottleneck capacity of 5 Mbps.

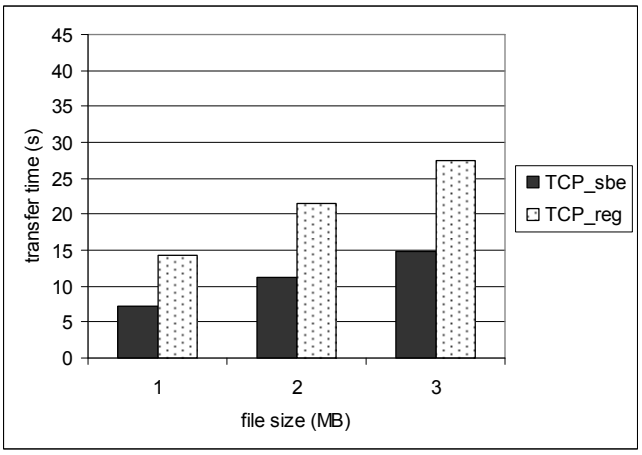

Figure 8. Transmission time for small files; minimum RTT of 500 ms, bottleneck capacity of 10 Mbps.

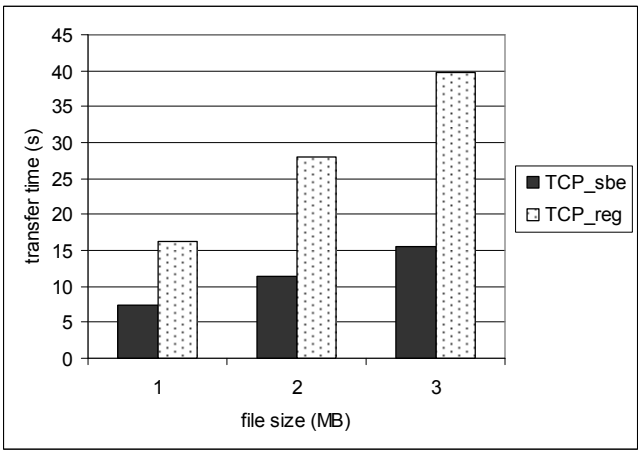

Figure 9. Transmission time for small files; minimum RTT of 500 ms, bottleneck capacity of 10 Mbps, PER of 0.1 %.

Finally, in Fig. 9 we have considered the same scenario of Fig. 8 but we have added a PER of 0.1 %. Results clearly show the positive impact of exploiting SBE to set the slow start threshold after a loss. In this way, the data sending rate is not unnecessarily shrank when the loss is caused by wireless errors. Indeed, the regular TCP flow needs from ~121 % (for the 1 MB file) to ~156 % (for the 3 MB file) more time to download those files with respect to a TCP that exploits SBE. Note that in this case the main advantage is achieved with the largest size of the downloaded file, whereas in the previous two cases (Fig. 7 and Fig. 8) the performance improvement is more evident when downloading the 1 MB file. This is due to the fact that, in the previous case, SBE was mainly used just to set the initial value of the slow start threshold; thereby, the larger the initial phase with respect to the whole download (i.e., the smaller the file), the better the improvement. Instead, in Fig. 9, error losses happen along the whole duration of the download. Downloading a bigger file, more packets will be lost and more times the SBE value will be used, thus enlarging the performance difference among *TCP_reg* and *TCP_sbe*.

V. CONCLUSION

In this work, we discussed SBE, a simple algorithm able to provide an accurate estimation of the channel capacity detracted the continuously present concurrent traffic. Our scheme recalls the packet train mechanism but is perfectly embedded within the regular functionalities of traditional TCP.

Preliminary simulation results are very promising as they demonstrate the accuracy of SBE even at the very beginning of a connection. This is a very important property especially for long delay connections such as satellite links. Certainly, the analysis presented in this paper can be extended through considering more extensive and complex scenarios, e.g., different RTTs, preexisting or successively starting concurrent heterogeneous flows, presence of reverse traffic and delayed ACKs. However, we have here shown a proof of concept on how the information provided by SBE, coupled with its high stability, can be exploited alone or combined with other bandwidth/capacity estimators (e.g., TCP Westwood's eligible rate) to improve TCP's performance on satellite links.

REFERENCES

[1] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links," in Proc. of ACM MOBICOM 2001, Rome, Italy, Jul 2001.

[2] R. Wang, M. Valla, M.Y. Sanadidi, M. Gerla, "Adaptive Bandwidth Share Estimation in TCP Westwood," in Proc. of IEEE Globecom 2002, Taipei, Taiwan, R.O.C., Nov 2002.

[3] R. Wang, G. Pau, K. Yamada, M. Y. Sanadidi, M. Gerla, "TCP Startup Performance in Large Bandwidth Delay Networs," in Proc. of IEEE INFOCOM 2004, Hong Kong, Mar 2004.

[4] W. R. Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Addison Wesley, 1994.

[5] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla, M. Y. Sanadidi, "CapProbe: A Simple and Accurate Capacity Estimation Technique," in Proc. of ACM SIGCOMM 2004, Portland, OR, USA, 2004.

[6] R. Kapoor, L.-J. Chen, M. Y. Sanadidi, M. Gerla, "Accuracy of Link Capacity Estimates Using Passive and Active Approaches with CapProbe," in Proc. of ISCC 2004, Alexandria, Egypt, Jun 2004.

[7] G. Huston, "The future for TCP," The Internet Protocol Journal, vol. 3, no. 3, Sep 2000.

[8] G. Huston, "TCP in a Wireless World," IEEE Internet Computing, Mar–Apr 2001, pp. 82-84.

[9] The Network Simulator – ns-2. http://www.isi.edu/nsnam/ns/

[10] D. J. Bem, T. W. Wieckowski, R. J. Zielinski, "Broadband Satellite Systems," IEEE Communincations Surveys, vol. 3, no. 1, 2000.

[11] S. R. Pratt, C. E. Fossa Jr., R. A. Raines, M. A. Temple, "An Operational and Performance Overview of the IRIDIUM Low Earth Orbit Satellite System," IEEE Communications Surveys, vol. 2, no. 2, Apr 1999, pp. 2-10.

[12] N. Celandroni, E. Ferro, "The FODA-TDMA Satellite Access Scheme: Presentation, Study of the System, and Results," IEEE. Transactions on Communications, vol. 39, no. 12, Dec 1991, pp. 1823–1831.

[13] V Jacobson, R Braden, D Borman, "TCP Extensions for High Performance," IETF Request for Comments (RFC) 1323, 1992.

[14] C. E. Palazzi, "Residual Capacity Estimator for TCP on wired/wireless links", in Proc. of the WCC2004 Student Forum IFIP World Computer Congress 2004, Toulouse, France, Aug 2004.

[15] K. Fall, S, Floyd, "Simulation-based Comparisons of Tahoe, Reno, and Sack TCP", ACM Computer Communication Review, Jul 1996.

[16] P. Sinha, N. Venkitaraman, R. Sivakumar, V. Bharghavan, "WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks," in Proc. of ACM Mobicom '99, Seattle, Washington, USA, Aug 1999.

[17] C. Dovrolis, P. Ramathan, D. Moore "What Do Packet Dispersion Techniques Measure?" in Proc. of IEEE Infocom'01, Anchorage, Alaska, USA, Apr 2001.

[18] V. Tsaoussidis, H. Badr, "TCP-Probing: Towards an Error Control Schema with Energy and Throughput Performance Gains," in Proc. of the 8th IEEE Conference on Network Protocols (ICNP 2000), Osaka, Japan, Nov 2000.

[19] J. Strauss, D. Katabi, F. Kaashoek "A Measurement Study of Available Bandwidth Estimation Tools," in Proc. of the 3rd ACM SIGCOMM conference on Internet measurement table of contents, Karlsruhe, Germany, Aug 2003.

[20] L. Lao, C. Dovrolis, M. Y. Sanadidi, "The Probe Gap Model Can Underestimate the Available Bandwidth of Multihop Paths," in ACM SICGOMM Computer Communications Review (CCR), Editorial Section, vol. 36, no. 5, Oct 2006.

[21] M. Jain, C. Dovrolis, "End-to-End Available Bandwidth: Measurement methodology, Dynamics, and Relation with TCP Throughput", in Proc. of ACM SIGCOMM, Pittsburgh, PA, USA, Aug 2002

[22] N. Hu, P. Steenkiste, "Evaluation and Characterization of Available Bandwidth Techniques," IEEE Journal on Selected Areas in Communications, vol. 21, no. 6, Aug 2003.

[23] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, L. Cottrell, "pathChirp: Efficient Available Bandwidth Estimation for Network Paths," in Proc. of the 4th Passive and Active Measurement Workshop (PAM 2003), La Jolla, CA, USA, Apr 2003.

[24] M. Neginhal, K. Harfoush, H. Perros, "Measuring Bandwidth Signatures of Network Paths," IFIP NETWORKING 2007, Atlanta, GA, USA, May 2007

[25] X. Liu, K. Ravindran, D. Loguinov, "A Queuing-Theoretic Foundation of Available Bandwidth Estimation: Single-Hop Analysis," IEEE/ACM Transactions on Networking, vol. 15, no. 6, Dec 2007.

[26] M. Li, M. Claypool, R. Kinicki, "Packet Dispersion in IEEE 802.11 Wireless Networks", in Proc. of the 31st IEEE Conference on Local Computer Networks (LCN 2006), Tampa, FL, USA, Nov 2006.

[27] A. Johnsson, M. Bjorkman, B. Melander, "An Analysis of Active End-to-End Bandwidth Measurements in Wireless Networks," in Proc. of the 4th IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services (E2EMON 2006), Vancouver, Canada, Apr 2006.

[28] K. Lakshminarayana, V. N. Padmanabhan, J. Padhye, "Bandwidth Estimation in Broadband Access Networks," in Proc. of the 4th ACM SIGCOMM Conference on Internet Measurement, Taormina, Italy, Oct 2004.