Adaptive Bandwidth Share Estimation in TCP Westwood

Ren Wang, Massimo Valla, M.Y. Sanadidi, and Mario Gerla
UCLA Computer Science Department, Los Angeles, CA 90095, USA
{renwang,mvalla,medy,gerla}@cs.ucla.edu

***Abstract*--TCP Westwood (TCPW) is a recently proposed sender side modification of TCP congestion control. TCPW relies on bandwidth share estimation techniques to enhance congestion control over high speed and/or wireless networks. The bandwidth share estimation methods turn out to be critical to guarantee both throughput improvement and friendliness towards widely used TCP protocols such as NewReno. In this paper we propose a new bandwidth share estimation technique, called "Adaptive Bandwidth Share Estimation", or ABSE. ABSE adapts to changing network congestion level, round trip times, and other relevant network conditions, as well as to the rate at which such changes occur. We compare the throughput gain and friendliness of ABSE with that of NewReno and previous estimation methods used for TCPW. We test the new technique using different simulation scenarios, including RED, to show the benefit of our proposed ABSE estimation method.**

***Index terms*-- TCP Westwood, Adaptive Bandwidth Share Estimation, Friendliness and Fairness, Wireless Networks**

I. INTRODUCTION

The Transmission Control Protocol (TCP), which has evolved over time into a number of versions, from TCP Tahoe to the currently widely used TCP NewReno, provides end-to-end, reliable, congestion controlled connections over the Internet[Jaco88,90]. Originally designed for the "wired" environment of the Internet, where congestion accounts for most packet losses, it is well known that the current TCP throughput deteriorates in high-speed heterogeneous networks including wired and wireless links. Unlike wired networks, in wireless networks, many of the packet losses are due to noise and external interference over wireless links. Congestion control schemes in current TCP assume that a packet loss is invariably due to congestion and reduce their congestion window by half, thus the performance deterioration mentioned above.

Many research efforts have been made to adapt TCP to the new high-speed wired/wireless environment [GMLW99] [KVM99][GMPG00][BPSK97][BK98]. Such work can be classified, according to the protocol level the schemes are operating on, into three main categories: (1) Schemes relying on link layer enhancements proposing ARQ schemes at the wireless link layer, exemplified by SNOOP [BSAK95]; (2) Network layer features beneficial to TCP performance, or providing to TCP explicit congestion information, exemplified by Explicit Congestion Notification (ECN) [Floy94]; (3) End-To-End schemes at the Transport layer, requiring no support form lower layers; examples here include TCP Vegas[BP95], Packet Pair[Kesh91], and TCP Westwood.

To handle wireless losses in heterogeneous networks, in [CGMSW01] we have proposed TCP Westwood (TCPW, for short). TCPW design adheres to the end-to-end transparency guidelines set forth in [Clar88] and requires slight modifications, only at the sender side. A TCPW sender continuously estimates the packet rate of the connection by properly averaging the rate of returning acknowledgements using a discrete low-pass filter. The *cwin* and slow start threshold *(ssthresh)* after a packet loss are set based on the estimated packet rate of the connection.

TCPW estimation has evolved starting with a method we call Bandwidth Estimation, or BE for short. TCPW BE strategy provides significant throughput gains when error loss is as likely as congestion loss [CGMSW01]. However, when co-existing with TCPW BE, under certain conditions TCP NewReno may experience some performance degradation since TCPW "learns" more about connection performance and thus can take better advantage of available bandwidth.

To manage the efficiency/friendliness tradeoffs, we have proposed in [WVNMG02] to combine the original TCPW Bandwidth Estimation (BE) strategy with a new Rate Estimation (RE) strategy, introducing the Combined Rate and Bandwidth estimation (CRB) scheme. One finds that BE provides significantly higher utilization, but may, under certain conditions, overestimate a connection fair share. RE, on the other hand, tends to be closer to the achieved rate of a connection, but it may underestimate the connection fair share. We have shown that RE works best when packet loss is mostly due to congestion. If, on the other hand, packet loss is mostly due to link errors, BE gives better performance. In CRB, a connection first infers the predominant cause of packet loss (buffer congestion or random error) and then uses the most appropriate estimation method. Simulation shows that the adaptive CRB provides an effective compromise between efficiency and friendliness. CRB is friendlier to NewReno. It suffers, however, efficiency degradation in lossy environments.

In this paper we address the efficiency degradation in TCPW CRB, introducing a method that achieves both efficiency and friendliness. The new method, the Adaptive Bandwidth Share Estimation (ABSE) provides continuous adaptivity to congestion level. In this paper we also consider automated tuning of estimator parameters so that the estimator adapts to relevant network conditions and the rate at which they change. Relevant network conditions include Round Trip Time, bottleneck and its bandwidth, traffic loading on the connection path, and error rate. As a result, the estimator is agile enough to react to persistent changes, while tolerating transient noise.

It is worth noting that all TCPW variants (including TCPW ABSE) rely only on information readily available at the sender, using the current TCP header, and do not require any support from receivers or any network component.

This research was supported by NSF under grant ANI-9983138

In Section II, we briefly describe the TCPW ABSE protocol. Then in section III, we present our adaptive bandwidth share estimation algorithm. In Section IV, we provide a performance study under different network environments using ns-2 simulator. Finally, Section V concludes the paper.

II. TCPW ABSE PROTOCOL

TCPW ABSE is a sender-only modification of TCP NewReno. The TCP sender Adaptively determines a Bandwidth Share Estimate (TCP-ABSE). The estimate is based on information in the ACKs, and the rate at which the ACKs are received. After a packet loss indication, which could be due to either congestion or link errors, the sender uses the estimated bandwidth to properly set the congestion window and the slow start threshold.

Further details regarding bandwidth estimation are provided in following Sections. For now, let us assume that a sender has determined the connection bandwidth estimate as mentioned above, and let us describe how the estimate is used to properly set *cwin* and *ssthresh* after a packet loss indication.

First, we note that in TCPW, congestion window dynamics during slow start and congestion avoidance are unchanged; that is, they increase exponentially and linearly, respectively, as in current TCP NewReno.

A packet loss is indicated by (a) the reception of 3 DUPACKs, or (b) a coarse timeout expiration. In case (a), TCPW sets *cwin* and *ssthresh* as follows:

```
if (3 DUPACKs are received)
   ssthresh =  (ABSE * RTTmin) / seg_size;
   if (cwin > ssthresh) /* congestion avoid. */
      cwin = ssthresh;
   endif
endif
```

In case a packet loss is indicated by a timeout expiration, *cwin* and *ssthresh* are set as follows:

```
if (coarse timeout expires)
   cwin = 1;
   ssthresh = (ABSE * RTTmin) / seg_size;
   if (ssthresh < 2)
      ssthresh = 2;
   endif;
endif
```

The rationale of the algorithm above is that after a timeout, *cwin* and the *ssthresh* are set equal to 1 and ABSE, respectively. Thus, the basic Reno behavior is still captured, while a reasonably speedy recovery is ensured by setting *ssthresh* to the value of ABSE.

III. ADAPTIVE BANDWIDTH SHARE ESTIMATION (ABSE)

In this Section, we present our Adaptive Bandwidth Share Estimation (ABSE) algorithm in detail. ABSE adapts to the congestion level in performing its bandwidth sampling, and employs a filter that adapts to the round trip time and to the rate of change of network conditions.

The bandwidth share estimation is computed using a time varying coefficient, exponentially-weighted moving average (EWMA) filter, which has both adaptive gain and adaptive sampling. Let t_k be the time instant at which the k_{th} ACK is received at the sender. Let s_k be the bandwidth share sample, and $\hat{s}_k$ the filtered estimate of the bandwidth share at time t_k. Let α_k be the time-varying coefficient at t_k. The ABSE filter is then given by:

$$\hat{s}_k = \alpha_k \hat{s}_{k-1} + (1-\alpha_k) s_k \qquad (1)$$

where $\alpha_k = \frac{2\tau_k - \Delta t_k}{2\tau_k + \Delta t_k}$, and τ_k is a filter parameter which determines the filter gain, and varies over time adapting to path conditions.

In the filter formula above, the bandwidth sample at time k is $s_k = \frac{\sum_{t_j > t_k - T_k} d_j}{T_k}$, where d_j is the number of bytes that have been reported delivered by the j_{th} ACK, and T_k is an interval over which the bandwidth sample is calculated.

A. ABSE adaptive sampling

Recall that in TCPW BE estimation, the bandwidth sample is obtained from the most recent pair of ACKs. Since TCP traffic tends to be bursty, BE may overestimate the connection bandwidth share. In Rate Estimation (RE) the bandwidth sample is computed based onthe amount of data acknowledged during the latest interval of time T. The resulting bandwidth estimate is generally lower, since the calculation in RE has the effect of equally spacing the ACKs over the interval T. The CRB scheme switches between RE and BE, after identifying whether the predominant cause of packet loss is error or congestion.

One conclusion we have drawn from the analysis and simulations in [WVSNG02] is that, the longer the time interval T, the more conservative the RE estimation becomes. Consequently, the more severe the congestion, the longer T should be. This adaptation idea of the sampling interval is illustrated in Fig. 1. The sampling interval ranges between two extremes T_{min} and T_{max}. T_{min} is the ACK interarrival time, while T_{max} has been set to a fixed value T in CRB.

ABSE provides an adaptive sampling scheme, in which the time interval T_k associated with the k_{th} received ACK is appropriately chosen between the two extremes (as illustrated in Fig. 1), depending on the network congestion level.

To determine the network congestion level, a simple throughput filter is proposed to estimate the recent throughput achieved. By comparing this estimate with the instantaneous sending rate obtained from *cwin*, we get a measure of the path congestion level. The difference between the instantaneous sending rate and the achieved rate (as measured from the throughput filter) clearly feeds the bottleneck queue, thus revealing that the path is becoming congested. The larger the difference, the more severe the congestion, and the larger the new value of T_k should be.

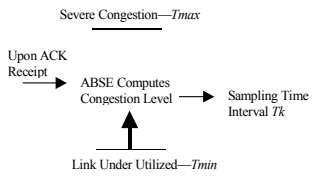


Fig. 1. Illustration of Sampling Time Interval T_k Adaptation

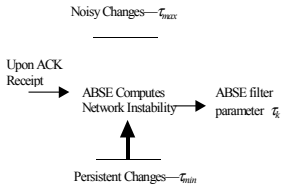


Fig. 2. Illustration of ABSE Filter τk Adaptation

When the k_{th} ACK arrives, a sample of throughput during the previous RTT is calculated as:

$Th_k = \frac{\sum_{t_j > t_k - RTT} d_j}{RTT}$, where d_j is the amount of data reported by ACK j. We employ a constant-gain (ε=0.6) filter to calculate the recent throughput as:

$$\hat{Th}_k = \varepsilon \hat{Th}_{k-1} + (1-\varepsilon) Th_k \quad (2)$$

When $\hat{Th}_k * RTT$ is larger than the current *cwin* value, indicating a path without congestion, T_k is set to T_{min}. Otherwise, T_k is set to:

$$T_k = RTT * \frac{cwin - (\hat{Th}_k * RTT_{\min})}{cwin} \quad (3)$$

Or upon rearrangement:

$$T_k = RTT * (\frac{cwin}{RTT_{\min}} - \hat{Th}_k) / \frac{cwin}{RTT_{\min}} \quad (4)$$

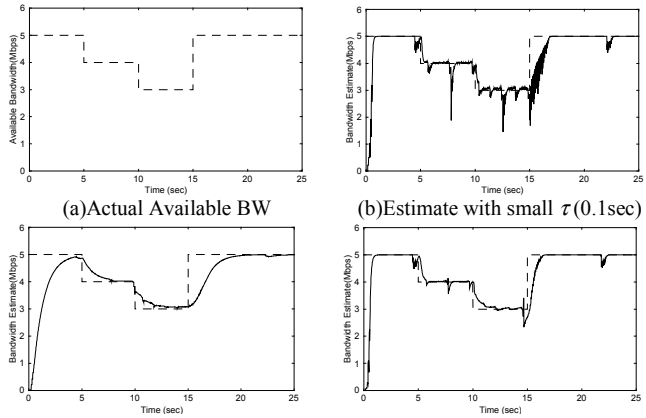
In equation (4), $cwin / RTT_{\min}$ is the expected throughput assuming there is no congestion in the network, while $\hat{Th}_k$ is the actual throughput the network allowed. The farther away the actual throughput gets from the expected throughput, the more congestion there is in the network, thus the larger time interval we need to space the packets uniformly to obtain the fair bandwidth share. After T_k is chosen, the ABSE sample associated with k_{th} received ACK is then expressed by:

$$s_k = \frac{\sum_{t_j > t_k - T_k} d_j}{T_k} \quad (5)$$

Note that the adaptive sampling could also be performed, using similar strategy, by choosing the length of ACK stream, i.e., the number of ACKs, and computing the bandwidth share sample. In fact, in the steady state, suppose there is no delayed ACKs, we can roughly convert the time interval T_k into the ACK stream length N by $N=T_k * cwin/RTT$.

B. Filter Gain Adaptation

The EWMA (Exponentially Weighted Moving Average) filter $\hat{s}_k = \alpha_k \hat{s}_{k-1} + (1-\alpha_k) s_k$ (1), used in ABSE, places more importance to more recent data by discounting older data in an exponential manner. The value of the filter parameter α_k, dictates the degree of filtering. Small α_k corresponds to agile filter while large α_k to stable filter. To relate the filter behavior to the parameter τ_k, we need to look at equation $\alpha_k = \frac{2\tau_k - \Delta t_k}{2\tau_k + \Delta t_k}$ (6). When τ_k is larger, α_k will be larger and the filter tends to be more stable and less agile. After a certain point, α_k basically stays unchanged as the value of τ_k increases.



(a)Actual Available BW (b)Estimate with small τ (0.1sec)

(c)Estimate with Large τ (1sec) (d)Estimate with Adaptive τ_k

Fig. 3. Bandwidth Estimation with Different τ

The parameter τ_k adapts to network conditions to dampen estimates when the network exhibits very unstable behavior, and react quickly to persistent changes. A stability detection filter can be used to dynamically change the value of τ_k. We measure the network instability U with a time-constant EWMA filter [KN01]:

$$U_k = \beta U_{k-1} + (1-\beta)|s_k - s_{k-1}| \quad (7)$$

In (7), s_k is the k_{th} sample, and β is the gain of this filter, which is set to be 0.6 in our experiments. When the network exhibits high instability, the consecutive observations diverge from each other, as a result, U_k increases. Under this condition, increasing the value of τ_k makes the ABSE filter (1) more stable. The adaptation of parameter τ_k is illustrated in Fig. 2.

When a TCP connection is operating normally, the interval between the consecutive acknowledgements are likely to vary between the smallest the bottleneck capacity allows, and one RTT. Therefore, τ_k should be larger than one RTT, thus $\tau_{min} = RTT$. We set the τ_k to be:

$$\tau_k = RTT + N * RTT \frac{U_k}{U_{\max}} \quad (8)$$

The value of RTT can be obtained from the smoothed RTT estimated in TCP. The factor N is set to be 10 in our experiments, which gives good performance under various scenarios. U_{max} is the largest instability in the ten most recent observations as in [KN01].

The τ_k adaptation algorithm described above is able to achieve agility to persistent changes while retaining stability against noise. Fig. 3 shows the bandwidth share estimated by applying a small τ, a large τ, and the adaptive τ_k respectively. The sampling time interval T_k is fixed to the interval of the last two ACKs in this set of experiments. The simulation configuration features a 5Mbps bottleneck link with a one-way propagation time of 35ms. The bottleneck router is FCFS Drop-tail with buffer capacity equal to the pipe size (i.e. the bandwidth-delay product). The TCPW connection shares the bottleneck with on-off non-adaptive UDP traffic with time varying intensity. The actual available bandwidth for the TCP connection is shown in Fig. 3(a).

The results show that with all three cases of τ, the estimator can essentially track the available bandwidth. However, in case (b) with small τ, the amplitude of the oscillation is large. In case (c) with large τ, the oscillation is eliminated because the

filtering is stronger, but the response to the actual bandwidth variations is very slow. In case (d) with adaptive τ_k, the estimator can achieve fast response with less oscillations due to noise.

IV. PERFORMANCE EVALUATION

In this Section, we compare the performance of TCPW ABSE, BE, CRB and TCP NewReno. All results are obtained using the ns2 simulator [ns2][TCPW]. We study the fairness and friendliness in Section 6.1. In Section 6.2 and 6.3, we examine the transient dynamics and interaction with RED routers respectively. In Section 6.4, we evaluate the throughput performance of the different protocols over lossy links (wireless environments).

The summary of the results is that ABSE provides a means of adapting to loss causes and network stability, and achieves both efficiency and friendliness to TCP NewReno.

A. Fairness and Friendliness

Fairness relates to the relative performance of a set of connections of the same TCP variant. Friendliness relates to how sets of connections running different TCP flavors affect the performance of each other. The simulation topology consists of a single bottleneck link with a capacity of 10 Mbps, and one-way propagation delay of 35ms. The buffer size at the bottleneck router is equal to the pipe size. The link is loss free except where otherwise stated.

A set of simulations with 10 simultaneous flows was run to investigate fairness of ABSE. The Jain's fairness index [Jain91] of ABSE reached 0.9919, and that of NewReno is 0.9904. Therefore, fairness of ABSE is comparable to that of NewReno.

We evaluate below the friendliness of TCPW BE and ABSE towards NewReno. We ran simulations with total 10 TCP connections of various schemes sharing a 10Mbps bottleneck. In fig. 4 and Fig. 5, the horizontal axis represents the number of competing NewReno connections, the remaining connections being BE (in Fig. 4) or ABSE (in Fig. 5). The vertical axis represents the average throughput of TCP NewReno and TCPW respectively. The results in Fig. 4 show that TCPW BE achieves higher throughput than its fair share, thus it is unfriendly to NewReno. This is because BE over-estimates the connection bandwidth share [WVSNG02].

In Fig. 5, we observe that the throughput achieved by both NewReno and ABSE are very close to the fair share, showing that ABSE achieves friendliness towards NewReno.

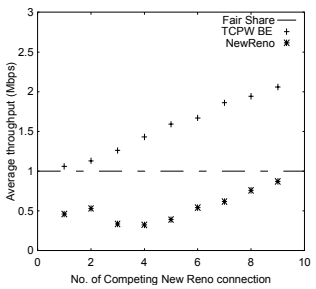


Fig. 4. TCPW BE Friendliness towards NewReno

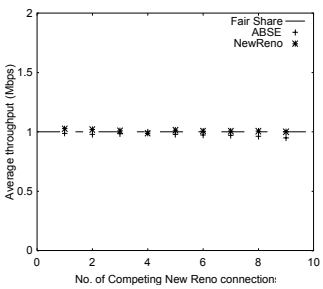
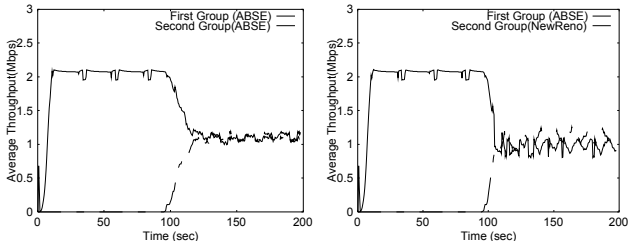


Fig. 5. TCPW ABSE Friendliness towards NewReno

B. Transient Dynamics

To examine the transient behavior of new TCP connections against established connections, we simulate 10 flows sharing a 10Mbps bottleneck. In our experiments, 5 sources are established at the beginning; another 5 new sources become active after 100s. Fig. 6(a) shows the throughput averaged over a set of connections, when the new connections are ABSE, which join established ABSE connections. In Fig. 6(b), the new connections are TCP NewReno. The results show that the new connections, both ABSE and NewReno, readily acquire their fair share against the established connections.

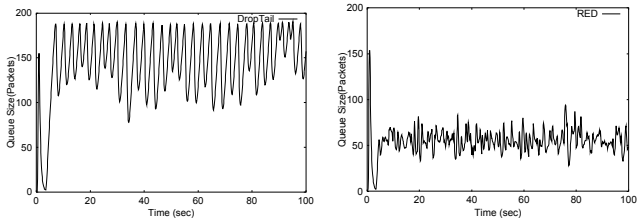


(a) ABSE +ABSE (B) ABSE +NewReno

Fig. 6 Transient Dynamics: Instantaneous Throughput of New and Established connections

C. Interaction with RED

The RED queue management scheme has been proposed to prevent extreme congestion and "phasing", and to enhance fairness [FJ93]. To evaluate the interaction of TCPW ABSE and RED, we simulate 10 ABSE flows sharing a 10Mbps bottleneck link, with or without RED, and observe the bottleneck queue dynamics. The buffer capacity is 200 packets (about twice the pipe size.) The min_{th}, max_{th} and maximum drop probability p_{max} are set to be 30, 150, 0.3 respectively. The queue length dynamics are shown in Fig. 7. Comparing Fig. 7(a) and Fig. 7(b), the RED router maintains smaller and more stable queue length, thus reducing the jitter and queuing delay. Most importantly, this shows that similar to TCP NewReno, TCPW ABSE interoperates productively with the RED algorithm.



(a) DropTail (b) RED

Fig. 7. Queue Length of the Bottleneck Router

D. Throughput with Lossy Links

In this subsection, we examine a number of different scenarios to compare the performance of TCPW BE, CRB, ABSE, to that of NewReno in the wired/wireless configuration. The wired portion has a capacity of 45 Mbps and one-way propagation time of 35ms (roughly the delay from West to East coast in the USA). The wireless portion is a very short 11-Mbps link with a negligible propagation time (e.g. WaveLAN link). The main lesson learned by this

experiment is that ABSE, while achieving friendliness towards NewReno, does not suffer efficiency deterioration the same way CRB does.

The throughput of ABSE, BE, CRB and NewReno are compared under packet loss rate varying from 0 to 5%. The results in fig. 8 show that the throughput in ABSE, BE and CRB, is higher than that in NewReno. The largest improvement is obtained around 0.1% to 1% loss rate, where ABSE and BE throughput gain over NewReno is about 500%, while CRB gains only 300% over NewReno.

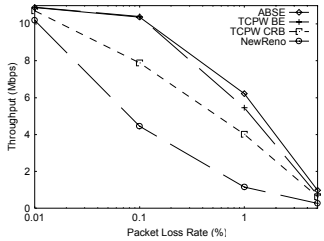


Fig. 8. Throughput vs. packet loss rate

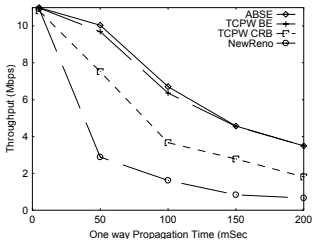


Fig. 9. Throughput vs. one-way propagation time

To assess the relation of the throughput gains to the E2E propagation time, we ran simulations with the wired portion one-way propagation time varying from 0 to 200ms and the wireless link loss rate set to 0.1%. The results in Fig. 9 show a significant gain for ABSE, BE and CRB of up to 528%, 528% and 384% respectively. Notice that ABSE performs slightly better than BE in some range. When the propagation time is small (say, less than 5ms), all protocols are equally effective. This is because a small window is adequate and window optimization is not an issue.

Simulation results in Fig. 10 show that ABSE, BE and CRB gains increase significantly as the bottleneck link transmission speed increases. Thus, they are more effective than TCP NewReno in utilizing the Gbps bandwidth provided by new-generation, high-speed networks. The error rate here is 0.1%, and the E2E propagation time is 70ms.

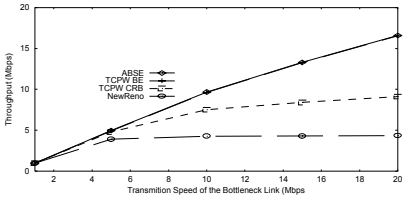


Fig. 10. Throughput vs. bottleneck capacity

V. CONCLUSION

In this paper we have described a new adaptive bandwidth-share estimation technique that, when used with the TCP Westwood protocol, can provide higher throughput over wireless links while maintaining friendliness to TCP NewReno connections. ABSE adapts the size of the observation interval, over which it obtains a bandwidth sample, to the existing network congestion level. Bandwidth samples are further smoothed through a low pass filter, whose agility is also adapted to measured network instability. By exploiting these two levels of adaptability, the proposed ABSE filter produces a more accurate estimation of the connection bandwidth share, which turns out to be a critical factor to obtain efficiency as well as friendliness towards TCP NewReno connections. We have tested the proposed ABSE scheme in single and multiple connections scenarios, with random errors, and with and without RED routers. We have shown that, similar to NewReno, TCPW interoperates positively with RED. In the near future we intend to address the performance of ABSE in short lived sessions, in "many flows" situations and with buffers smaller than "pipe size". We will study the coexistence of ABSE with rate-adaptive real time UDP flows managed by TCP-like, equation driven rate control. Moreover, we will address the efficient implementation of ABSE in popular systems such as Linux and Free BSD. Following that, measurements in laboratory settings, on the Internet, and combinations of both, will be carried out to assess ABSE behavior in actual network settings.

REFERENCES

[AP99] M. Allman and Vern Paxson, "On Estimating End-to-End Network Path Properties", *ACM/Sigcomm 1999.*

[BK98] H. Balakrishnan and R. H. Katz, "Explicit Loss Notification and Wireless Web Performance," In *Proceedings of IEEE GLOBECOM'98 Internet Mini-Conference, Sydney, Australia, November 1998.*

[BP95] L.S. Brakmo and L.L. Perterson. "TCP Vegas: End-to-End Congestion Avoidance on a Global Internet." *IEEE JSAC, Vol. 13, Nov. 8, October 1995.*

[BPSK97] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," *IEEE/ACM Transactions on Networking, December 1997.*

[BSAK95] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz, "Improving TCP/IP Performance Over Wireless Networks," *MOBICOM'95, Berkeley, CA, USA, November 1995.*

[CGMSW01] C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links," *In Proceedings of Mobicom 2001, Rome, Italy, Jul. 2001.*

[Clar88] D. Clark, "The design philosophy of the DARPA Internet protocols," *In Proceedings of Sigcomm'88 in ACM Computer Communication Review, vol. 18, no. 4, pp. 106 - 114, 1988.*

[FF96] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP," *Computer Communication Review, V. 26 N. 3, July 1996, pp. 5-21.*

[FJ93] S. Floyd and V. Jacobson, "Random Early Detection gateways for congestion Avoidance," *IEEE/ACM transactions on Networking, August 1993*

[Floy94] S. Floyd, "TCP and Explicit Congestion Notification," *ACM Computer Communication Review, V. 24 N. 5, pp. 10-23, Oct. 1994.*

[GMPG00] T. Goff, J. Moronski, D. S. Phatak, and V. Gupta, "Freeze-TCP: a True End-to-end TCP Enhancement Mechanism for Mobile Environments," *In Proceedings of IEEE INFOCOM'2000, Tel Aviv, Israel, March 2000.*

[Hoe96] J. C. Hoe, " Improving the Start-up of A Congestion Control Scheme for TCP", *Proc. ACM SIGCOMM '96, pp. 270-280*

[Jaco88] V. Jacobson, "Congestion Avoidance and Control," ACM Computer Communications Review, 18(4) : 314 - 329, August 1988.

[Jaco90] V. Jacobson, "Berkeley TCP evolution from 4.3-Tahoe to 4.3 Reno," *Proceedings of the 18th Internet Engineering Task Force, University of British Colombia, Vancouver, BC, Sept. 1990.*

[Jain91] R. Jain, "The art of computer systems performance analysis," *John Wiley and sons, QA76.9.E94J32, 1991*

[Kesh91] S. Keshav "A Control-Theoretic Approach to Flow Control," *Proceeding of ACM SIGCOMM 1991*

[KN01] Minkyong Kim and Brian Noble, "Mobile Network Estimation," *In Proceedings of Mobicom 2001, Rome, Italy, Jul. 2001.*

[ns2] ns-2 network simulator (ver.2.) LBL, *URL: http://www.mash.cs.berkley.edu/ns.*

[RFC2582] S. Floyd, and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," *RFC 2582, Experimental, April 1999.*

[TCPW] TCP Westwood modules for ns-2. *URL: http://www.cs.ucla.eduNRL/hpi/tcpw*

[WVSNG02] Ren Wang, Massimo Valla, M.Y. Sanadidi, Bryan Ng and Mario Gerla, "Efficiency/Friendliness Tradeoffs in TCP Westwood", *IEEE Symposium on Computers and Communications, Taormina, Italy, July 2002.*