

TCPW with Bulk Repeat in Next Generation Wireless Networks

Guang Yang, Ren Wang, M. Y. Sanadidi, Mario Gerla

Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90095, USA
{yangg, renwang, medy, gerla}@cs.ucla.edu

Abstract — Wireless links are error-prone. In Next Generation wireless networks, link capacities will still grow. It is known that in such configurations the performance of current TCP variants degrades severely. In this paper we propose a new retransmission strategy for TCP in high error situations – Bulk Repeat (BR). We apply BR to TCP Westwood (TCPW). BR has only three sender-side modifications to TCP, i.e. Bulk Retransmission, Fixed Retransmission Timeout and Intelligent Window Adjustment. BR permits efficient recovery from multiple losses in the same congestion window. To discriminate error from congestion loss, a Loss Discrimination Algorithm (LDA), based on Spike and Rate Gap Threshold, is used. Simulation results in wireless network scenarios show that TCPW BR improves throughput performance up to an order of magnitude over TCPW and NewReno when the error rate is high (>5%). The results also show that TCPW BR has satisfactory fairness and friendliness to TCP NewReno.

Keywords — Wireless networks, Bit error, TCP retransmission and recovery, TCP Westwood

I. INTRODUCTION

TCP is the protocol in charge of end-to-end, reliable, and congestion controlled data transport over the Internet [Jaco88]. The performance of the widely used TCP NewReno, however, is known to deteriorate severely over wireless links when errors grow above a few percent. The major cause lies at the heart of TCP congestion control mechanism: a NewReno sender regards each packet loss as an indication of congestion and reduces the congestion window (*cwnd*).

Losses in wireless networks may be bursty, resulting in multiple losses in one window. TCP NewReno retransmits only one lost packet per RTT in fast recovery; it takes multiple RTTs to recover from losses. More often this will cause a timeout and severely degrade performance, especially when RTT is long as in satellite links and inter-continental paths.

A number of research efforts have been undertaken to address long delay and error-prone wireless environments [Rizz96] [BPSK97] [RFC2883] [AMP01]. Often the focus is on hiding losses from the transport layer by link-layer retransmissions, or engaging routers to provide explicit feedback. Our research focuses on designing strictly end-to-end schemes. The proposed scheme, called Bulk Repeat (BR), includes only three sender-side modifications: (1) Bulk

Retransmission: retransmitting all unacknowledged packets immediately when more than one loss is detected in a window; (2) Fixed Retransmission Timeout: using a fixed timeout value instead of exponential backoff when consecutive losses occur; and (3) Intelligent Window Adjustment: keeping *cwnd* fixed in spite of losses. The sender enables the BR modifications only when it suspects the losses to be caused by errors rather than congestion. A Loss Discrimination Algorithm (LDA) is used to distinguish between congestion and errors.

We describe and evaluate BR in the context of TCP Westwood (TCPW) [CGMSW01] [WVSG02]. However, the BR technique is independent of the specific TCP version and can be applied with other variants. One advantage of using TCPW is that a LDA implementation is readily available. TCPW BE is known to perform better than NewReno in light loss environments. However TCPW BE handles multiple losses the same way as NewReno, suffering the same degradation in heavy loss scenarios. Simulations show that TCPW with Bulk Repeat (TCPW BR) has much improved performance in heavy loss environments. Throughput can be up to an order of magnitude higher than TCPW BE, with satisfactory fairness and friendliness to NewReno.

The rest of the paper is organized as follows. In Section II we give a TCPW overview. In Section III we explain the three sender-side modifications of BR. Section IV describes our current LDA implementation. Section V presents simulation results. Related work is introduced in Section VI. Finally, Section VII discusses future work and concludes the paper.

II. TCP WESTWOOD OVERVIEW

In TCPW, the sender continuously monitors ACKs from the receiver and computes its current Eligible Rate Estimate (ERE). ERE is based on the rate at which ACKs are received and on their payload. In essence, ERE is the rate that the TCP connection is currently achieving. Thus, it must be a “feasible” rate by definition. DUPACKs and delayed ACKs are properly counted in ERE computation. Upon a packet loss (3 DUPACKs or a timeout) the sender sets *cwnd* and *ssthresh* based on the ERE. When an ERE is obtained, TCPW uses the following algorithm to set *cwnd* and *ssthresh*. For a complete description of TCPW please see [CGMSW01] [WVSG02].

if (3 DUPACKs are received)
 $ssthresh = (ERE * RTT_{min}) / seg_size;$

```

if (cwnd > ssthresh) /*congestion avoid*/
    cwnd = ssthresh;
endif
endif

if (coarse timeout expires)
    cwnd = 1;
    ssthresh = (ERE * RTTmin) / seg_size;
    if (ssthresh < 2)
        ssthresh = 2;
    endif
endif
endif

```

III. BULK REPEAT

In this section we briefly introduce the three sender-side modifications of BR, namely Bulk Retransmission, Fixed Retransmission Timeout and Intelligent Window Adjustment. More discussion on this subject can be found in [YWSG02]. Please note that all these modifications are enabled only when the path is not congested and losses are due to errors.

A. Bulk Retransmission

In fast retransmit a NewReno sender retransmits a single packet upon a partial ACK. In wireless environments where losses are largely due to errors, this strategy is inefficient since it requires multiple RTTs to recover from multiple losses. With Bulk Retransmission, if the sender gets only a partial ACK after fast retransmit, which probably indicates additional losses, it retransmits ALL outstanding, unacknowledged packets in the current congestion window. By doing this we avoid the overhead of recovering from each loss individually.

Several questions are raised here. The first is how many times should the sender attempt Bulk Retransmission in one window? Our decision is not to limit this number if there is no congestion. A maximum number may also be set. The second question is whether we advance the window progressively as the lost packets are recovered, or we work on the initial window only, retransmitting in bulk the portion that has not yet been acknowledged. In this study we use the latter option. Recently, we have also experimented, with success, with BR sliding window models. The algorithm in TCPW is described below:

```

if (sender receives 3 duplicate ACKs or times out)
    Retransmit the lost packet;
endif

if (sender receives a partial new ACK)
    if (discriminator indicates no congestion)
        Retransmit all outstanding and unacknowledged
        packets within the congestion window;
        Stay in fast recovery;
    endif
else
    Retransmit the lost packet;
    Stays in fast recovery;
endif
endif

```

B. Fixed Retransmission Timeout

TCP NewReno uses Karn's clamped retransmit backoff [Jaco88] [KP87], which has been proved stable and robust in the Internet. However, in heavy loss environments this strategy

could lead to disastrous TCP throughput collapse [VH99]. With high error rate, it is likely that the same packet is retransmitted many times before success. Karn's algorithm would cause the sender to wait idly for a long period of time. Simulations show at an error rate of 5% both NewReno and TCPW BE collapse because of this reason.

In order to improve TCP performance in heavy loss environments, we freeze the retransmission timeout to Fixed Retransmission Timeout. When a timeout occurs due to errors, the TCP sender does not double the timeout value if there is no congestion. This scheme is similar to what has been proposed in mobile wireless ad hoc networks to overcome large TCP timeouts following path breakage [VH99].

The algorithm in TCPW is described below:

```

if (sender times out)
    Retransmit the lost packet;
    if (discriminator indicates no congestion)
        Next RTO = fixed timeout value;
    else
        if (backoff_factor < 64)
            backoff_factor *= 2;
        endif
        Next RTO = fixed timeout value * backoff_factor;
    endif
endif

if (sender receives a new ACK)
    if (discriminator indicates no congestion)
        backoff_factor = 1;
    endif
endif
endif

```

C. Intelligent Window Adjustment

When a TCP sender adjusts *ssthresh* after a loss is detected, it also sets *cwnd* to *ssthresh* if *cwnd* is currently greater than *ssthresh*. In a normal situation this reduction is reasonable, but in case of heavy error losses the sender could use a more aggressive sending rate since many packets will be lost on the way. We change the *cwnd* adjustment strategy in TCP after an error loss has occurred. The sender does not cut *cwnd* when it is larger than *ssthresh*. This keeps congestion window relatively large so more packets will be transmitted during each RTT. The algorithm is described in TCPW context as:

```

If (a loss has been detected)
    Set ssthresh = ERE; /* TCPW */
    if (discriminator indicates no congestion)
        Do nothing, leave cwnd untouched;
    else if (congestion window > ssthresh)
        Set congestion window to ssthresh;
    endif
endif
endif

```

IV. LOSS DISCRIMINATION ALGORITHM

LDA is the most delicate concept in BR. The TCP sender must first determine whether losses are due to congestion or errors. Only when losses are due to errors, and there is no congestion, can the sender use BR; otherwise BR would lead to more congestion and worse throughput. Different LDAs have been proposed recently [Sama99] [CCV02]. We are using two

schemes, Spike and Rate Gap Threshold (RGT), as our current LDA in TCPW BR. Better LDAs are a future research focus.

Spike [CCV02] is a scheme based on RTT measurement. Our version of Spike is slightly different from the original one in that we use RTT instead of Relative One-way Trip Time (ROTT). Discussions in [YWSG02] validate this change. Spike keeps track of the minimum and maximum RTT values, RTT_{min} and RTT_{max} , and computes thresholds $Bspike_{start}$ and $Bspike_{end}$ with parameters a and b .

$$Bspike_{start} = RTT_{min} + a * (RTT_{max} - RTT_{min})$$

$$Bspike_{end} = RTT_{min} + b * (RTT_{max} - RTT_{min})$$

Spike enters congestion mode from error mode when RTT is greater than $Bspike_{start}$, and vice versa when RTT is less than $Bspike_{end}$. A typical setting for a and b is 0.4 and 0.05, respectively.

RGT is focused on the gap between the expected rate ($cwnd/RTT_{min}$) and the achieved rate, which we estimate by ERE now. If the gap is large then the sender is trying to send more than the network can handle, and losses are likely to be caused by congestion, otherwise losses are attributed to random errors. We observe that RGT works well when error rate is low, while Spike becomes more accurate when error rate grows higher. Our LDA combines the two and works as follows (RGT_{thresh} is a tunable parameter of RGT):

```

if ((ERE < RGT-thresh*(cwnd/RTTmin)) || (Spike in error mode))
    Loss is due to error;
else
    Loss is due to congestion;
endif

```

V. SIMULATION RESULTS

We evaluate the throughput performance of TCPW BR including fairness and friendliness. We first compare the performance of TCPW BR with TCPW BE and NewReno under different uniform error rates, round trip propagation times and bottleneck capacities. Then we study the fairness and friendliness of TCPW BR. Finally we evaluate the performance of TCPW BR, TCPW BE and NewReno over bursty error links. The results reveal that TCPW BR improves TCP end-to-end throughput significantly in heavy loss environments while retains good fairness and friendliness towards NewReno.

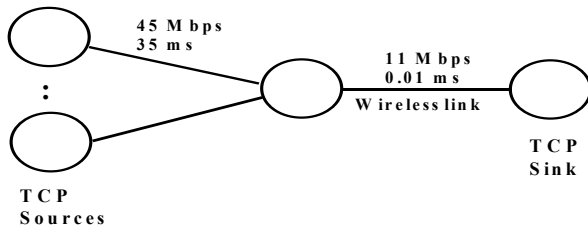


Figure 1. Simulation configuration

All results in this paper are obtained using ns-2 simulator. Topology of the experiments is shown in Figure 1. Wired links have a capacity of 45 Mbps and one-way propagation time of 35 ms (roughly the delay from West to East coast in the US). The wireless link is an 11Mbps point-to-point link connecting a mobile host to the base station. The wireless link is affected by

noise and error-prone. No link level retransmissions are attempted. This assumption is required in order to avoid the complex interplay between MAC and TCP retransmissions – a research topic of its own deserving separate investigation. The pipe size (bandwidth-delay product) is about 70 packets for packet size of 1000 bytes.

A. Uniform Channel Error

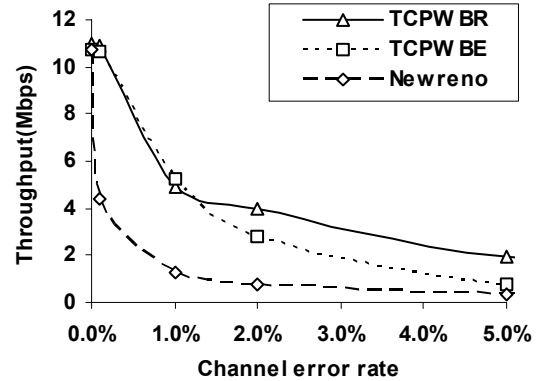


Figure 2. Throughput vs. error rate (linear scale)

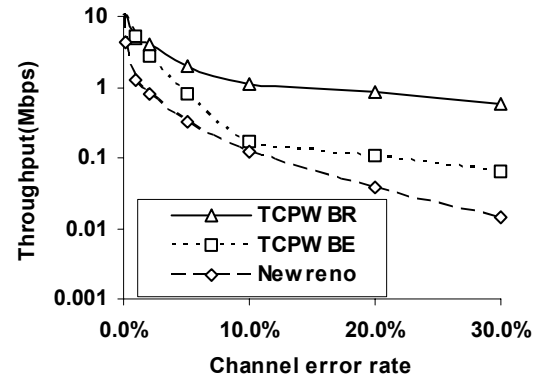


Figure 3. Throughput vs. error rate (log scale)

Figures 2 and 3 show the throughput results of a single flow. TCPW BR and TCPW BE are always better than NewReno. When the error rate goes beyond 1%, TCPW BR performs better than the other two. As an extreme case when the error rate is 30%, TCPW BR manages a remarkable 0.9Mbps throughput, about 10 times better than TCPW BE and 100 times better than NewReno.

Figures 4 and 5 assess the relationship of throughput to the propagation time and pipe size, respectively. In Figure 4 the two-way propagation time varies from 10 to 200ms with the wireless link error rate fixed at 5%. In Figure 5 the wireless link capacity varies from 2Mbps to 45Mbps with the error rate fixed at 5%. Figure 4 shows that as propagation time increases, the throughput of TCPW BR degrades less severely than TCPW BE or NewReno. Figure 5 shows that as pipe size increases, TCPW BR can utilize more capacity than the other

two.

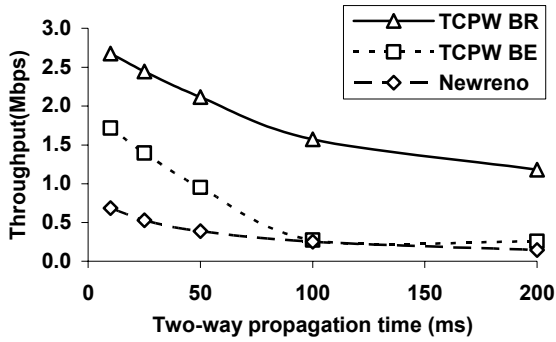


Figure 4. Throughput vs. two-way propagation time

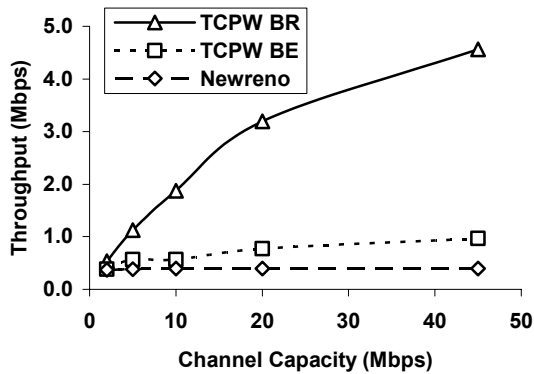


Figure 5. Throughput vs. Channel Capacity

B. Fairness and Friendliness

Fairness refers to the performance of a set of flows of the same TCP variant. To evaluate the fairness of TCPW BR we have run simulations with two simultaneous TCPW BR flows. Figures 6 and 7 show the result with error rates of 0% and 5%, respectively. TCPW BR is very fair to its own.

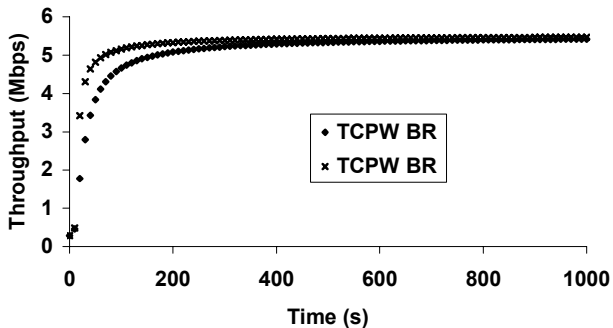


Figure 6. Two TCPW BR flows (error rate=0%)

Next, we evaluate the friendliness of TCPW BR toward NewReno and show the result in Figures 8 and 9. The wireless channel error rates are 0% and 5%, respectively. Channel capacity is almost equally shared at 0% error rate. At 5% error

rate we note (by comparison with Figure 2) that the individual throughputs of TCPW BR and NewReno are the same as in the single flow experiments. The two flows do not interfere and thus are friendly to each other.

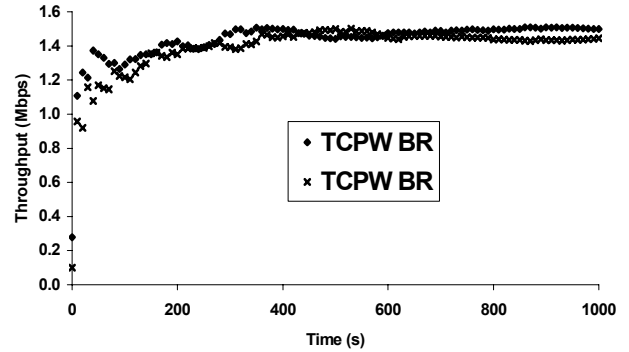


Figure 7. Two TCPW BR flows (error rate=5%)

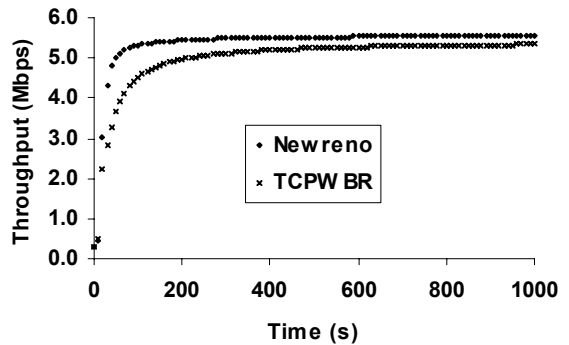


Figure 8. TCPW BR and NewReno (error rate=0%)

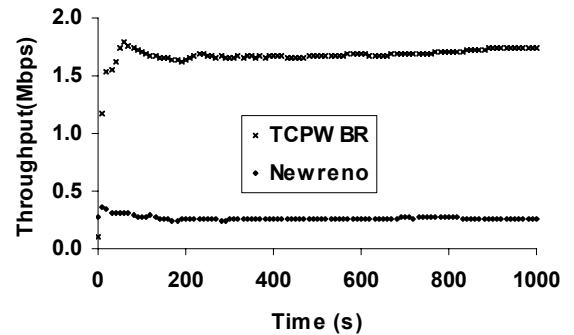


Figure 9. TCPW BR and NewReno (error rate=5%)

C. Bursty Error Channels

In this scenario we investigate the bursty error channels which are common in high error episodes. In satellite environments, for example, weather and line-of-sight blockage (to mobile receivers) tend to cause bursts of errors.

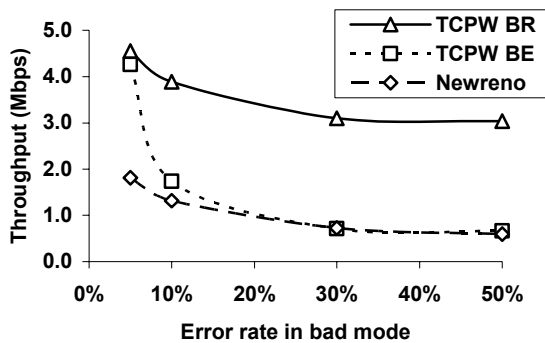


Figure 10. Performance comparison in bursty error channels

To capture bursty errors we use the two state error channel model [CGMSW01]. Error rate in the good state is 0.1% and 5% - 50% in the bad state. The result is shown in Figure 10. NewReno has a very low throughput for any bad state error rate. TCPW BE throughput drops rapidly beyond 5%. TCPW BR throughput drops only slightly all the way up to 50%.

VI. RELATED WORK

Several methods have been proposed to improve TCP performance over error-prone channels. However, many of them make use of link layer enhancements or explicit feedback. Examples include ARQ, Snoop [BPSK97] and ELN [BK98].

Far more challenging are the end-to-end schemes that rely on transport layer mechanisms. When the error rate is low, TCP SACK [RFC2883] is able to recover from multiple losses within one RTT, but it does not work well when error rate is high. It also requires modifications at both sender and receiver.

TCP-Peach [AMP01] is an end-to-end scheme that uses low-priority dummy packets to probe available bandwidth and discriminate between random error and congestion. TCP-Peach however requires a priority scheme in routers. Furthermore, TCP-Peach does not address performance degradation due to multiple errors and consecutive timeouts in a single window.

In [KAPSI02] Krishnan et al. propose Explicit Transport Error Notification (ETEN) for error-prone wireless and satellite environments. ETEN notifies TCP sender when packets get lost due to errors so the sender can react differently. ETEN assumes that sufficient information about the corrupted packet, such as IP addresses, is still available to routers and/or the receiver.

NewReno-FF [BM02] is a NewReno variant that uses a loss labeling technique to discriminate error from congestion. It assumes that change in RTT and the nature of losses are correlated; the longer the RTT the more likely the loss is congestion induced. It measures and maintains a history of RTT to distinguish between error and congestion. Simulations show it performs better than TCPW BE in error-prone but heavily congested environments. We believe this is because TCPW BE overestimates ERE and is too aggressive in congested networks. This problem has been pointed out and

fixed in [WVSG02]. NewReno-FF, to our knowledge, does not address slow retransmission or exponential backoff issues.

VII. CONCLUSIONS

In this paper we have proposed TCPW with Bulk Repeat (TCPW BR) to deal with high bandwidth, error-prone wireless links, a configuration which will become pervasive in Next Generation wireless networks. TCPW BR can solve the TCP degradation problem caused by high loss rates (say > 5%), providing order of magnitude improvements over all known methods to-date. Simulation results show that TCPW BR maintains fairness and friendliness to TCP NewReno. Important issues for future work are the loss discrimination algorithm (LDA) and Bulk Retransmission over a sliding window.

REFERENCES

- [AMP01] I.F., Akyildiz, G. Morabito, S. Palazzo, "TCP Peach: a new congestion control scheme for satellite IP networks," IEEE/ACM Transactions on Networking, Vol. 9, No. 3, pp. 307-321, Jun. 2001.
- [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, Nov. 1998.
- [BM02] D. Barman and I. Matta, "Effectiveness of loss labeling in improving TCP performance in wired/wireless networks", Boston University Technical Report BUCS-TR-2002-016, 2002.
- [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, Dec. 1997.
- [CCV02] S. Cen, P. C. Cosman and G. M. Voelker, "End-to-end differentiation of congestion and wireless losses," Proceedings of ACM Multimedia Computing and Networking 2002, San Jose, CA, Jan. 2002.
- [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.
- [HV99] G. Holland, and N. H. Vaidya, "Analysis of TCP performance over mobile adhoc network," In Proceeding of ACM MobiCom'99, Seattle, WA, Aug. 1999.
- [Jac088] V. Jacobson, "Congestion avoidance and control," ACM Computer Communications Review, 18(4) : 314 - 329, Aug. 1988.
- [KAPSI02] R. Krishnan, M. Allman, C. Partridge, J. Sterbenz, and W. Ivancic, "Explicit Transport Error Notification (ETEN) for error-prone wireless and satellite networks," Earth Science Technology Conference - 2002, CA, Jun. 2002.
- [KP87] P. Karn, C. Partridge, "Improving round-trip time estimates in reliable transport protocols", In proceedings of ACM SIGCOMM 87.
- [RFC2883] S. Floyd, J. Mahdavi, M. Mathis and M. Podolsky, "An extension to the Selective Acknowledgement (SACK) Option for TCP," RFC 2883, Jul. 2000.
- [Rizz96] Luigi Rizzo, "TCP retransmissions on very lossy networks", info.iet.unipi.it/~luigi/tcp.ps, Jan. 1996.
- [Sama99] N. Samaraweera, "Non-congestion packet loss detection for TCP error recovery using wireless links", in IEEE Proceedings of Communications, 146(4), pp. 222-230, Aug. 1999.
- [WVSG02] R. Wang, M. Valla, M.Y. Sanadidi and M. Gerla, "Using Adaptive Bandwidth Estimation to provide enhanced and robust transport over heterogeneous networks", 10th IEEE International Conference on Network Protocols (ICNP 2002), Paris, France, Nov. 2002.
- [YWSG02] G. Yang, R. Wang, F. Wang, M.Y. Sanadidi, and M. Gerla, "TCP Westwood with Bulk Repeat for heavy loss environments" UCLA CSD Technical Report #020029.