Selective-TCP for Wired/Wireless Networks

Rajashree Paul and Ljiljana Trajković
{rpaul2, ljilja}@cs.sfu.ca

Communication Networks Laboratory
http://www.ensc.sfu.ca/cnl
School of Engineering Science
Simon Fraser University, Vancouver, Canada

Roadmap

- Motivation
- Background and related work
- Selective-TCP:
  - algorithm
  - implementation
  - performance evaluation
- Performance comparison: Selective-TCP vs. TCP variants
- Conclusions and references

Motivation

- TCP is a widely used Internet transport protocol: 95% of IP traffic in 2004
- TCP performance degrades in wireless and in mixed wired/wireless (cellular) networks
- Main reasons for TCP's poor performance in wireless networks:
  - TCP assumes that all packet losses are due to network congestion
  - it cannot distinguish losses due to wireless link errors from losses due to congestion
  - packet loss in wireless networks is due to: high bit error rate (burst error), random loss, and link failure

Roadmap

- Motivation
- Background and related work
- Selective-TCP:
  - algorithm
  - implementation
  - performance evaluation
- Performance comparison: Selective-TCP vs. TCP variants
- Conclusions and references

Background

- Approaches to improve TCP performance in wired/wireless (heterogeneous) networks:
  - end-to-end: TCP Reno, TCP NewReno, TCP Westwood
  - split connection: I-TCP, MTCP, M-TCP
  - link layer: Snoop, TCP packet control
- End-to-end schemes:
  - significant performance gain without any modification in the intermediate routers
  - simpler to implement
  - not as effective as link layer based approaches in handling wireless losses

H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *Computer Communication Review*, vol. 26, no. 4, pp. 256–269, Aug. 1996.

Related work

- Improving TCP performance in wired/wireless networks by detecting the type of packet losses
- End-to-end approach:
  - TCP Veno
    - a combination of TCP Vegas and TCP Reno
    - differentiates congested and non-congested states of the network using proactive congestion control of TCP Vegas
  - TCP Real
    - receiver oriented congestion control mechanism
    - measures available bandwidth at the receiver and informs the sender

C. P. Fu and S.C. Liew, "TCP Veno: TCP enhancement for transmission over wireless access networks," *IEEE J. Select. Areas Commun.*, vol. 21, no. 2, pp. 216–228, Feb. 2003.
V. Tsaoussidis and Cz. Zhang, "TCP-Real: receiver-oriented congestion control," *Computer Networks*, vol. 40, no. 4, pp. 477–497, Nov. 2002.

Related work

- Link Layer based approach:
  - TCP SNACK-Snoop
    - combines TCP-Snoop with SNACK (selective negative ACK)
    - distinguishes congestion losses from wireless link losses at the intermediate routers by explicit loss notification
  - TCP-Jersey
    - estimates available bandwidth at the sender
    - distinguishes congestion losses from wireless link losses at the intermediate routers

F. Sun, V. O. K. Li, and S. C. Liew, "Design of SNACK mechanism for wireless TCP with new snoop," in *Proc. IEEE Wireless Communications and Networking Conference*, Atlanta, GA, Mar. 2004, vol. 2, pp. 1051–1056.
K. Xu, Y. Tian, and N. Ansari, "TCP-Jersey for wireless IP communications," *IEEE J. Select. Areas Commun.,* vol. 22, no. 4, pp. 747–756, May 2004.

Roadmap

- Motivation
- Background and related work
- Selective-TCP:
  - algorithm
  - implementation
  - performance evaluation
- Performance comparison: Selective-TCP vs. TCP variants
- Conclusions and references

Selective-TCP algorithm: overview

- An end-to-end solution to improving TCP performance in heterogeneous networks
- Detects type of packet losses at the receiver
- Corrective measures:
  - wireless loss: the receiver sends Selective Negative Acknowledgement (SNACK) to the sender
  - congestion loss: the sender's congestion window size is set according to the bandwidth measured at the receiver

    bandwidth =

    (*no. of received packets* × *size of packets in bits*)/ (*inter-arrival time between last in-sequence packet received and most recent packet* × 1,000) kbps
- Implemented in ns-2 v. 2.27

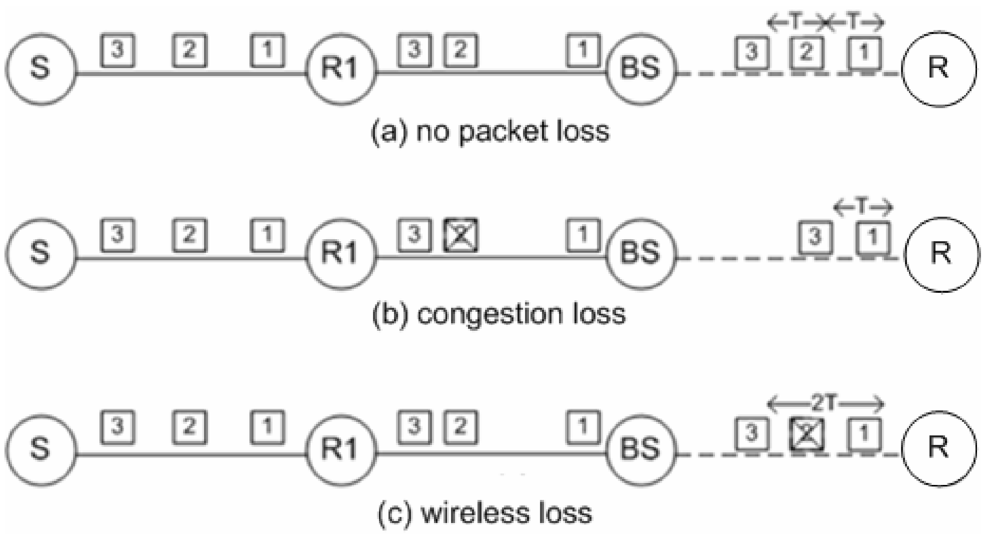
Loss detection scheme

- Detects the type of loss based on the packet inter-arrival times at the receiver
- Assumptions:
  - wireless link is the bottleneck
  - sender performs bulk data transfers
  - on the connection path, only the last link is wireless
- These assumptions ensure high accuracy of loss detection

S. Biaz and N. H. Vaidya, "Discriminating congestion losses from wireless losses using inter-arrival times at the receiver," in *Proc. ASSET'99*, Mar. 1999, pp. 10-17.
S. Cen, P. C. Cosman, and G. M. Voelker, "End-to-end differentiation of congestion and wireless losses," *Proc. of the SPIE - the International Society for Optical Engineering,* vol. 4673, pp. 1-15, 2001.

Loss detection scheme

- Wireless link is the bottleneck: packets will queue at the base station



S: sender
R1: router
BS: base station
R: receiver
T: minimum packet inter-arrival time at the receiver

- For a single packet loss:

  if (T < inter-arrival time ≤ 2T) {wireless loss}

  else {congestion loss}

Selective negative acknowledgement: SNACK

- TCP-SNACK is an option of the Satellite Communication Protocol Stack-Transport Protocol (SCPS-TP)
- Widely used for satellite links
- Negative: receiver informs sender about the segments not received
- Selective: can send information for multiple lost segments
  - good for long delay networks
- When a sender receives SNACK:
  - aggressively retransmits lost packets, preventing unnecessary retransmission time-outs

Consultative Committee for Space Data Systems, *Space Communications Protocol Specification—Transport Protocol (SCPS-TP)*, Blue Book, issue 1, May 1999.

Selective-TCP algorithm: description

- Detects type of packet losses
- In case of wireless loss, the receiver sends SNACK
  - sender retransmits the lost packets immediately without waiting for retransmission timer to expire:
    - no duplicate ACKs sent
    - TCP's congestion control is not invoked
    - congestion window is not reduced
  - SNACK is selective
    - helps in presence of packet reordering
  - result: increased bandwidth utilization and higher goodput

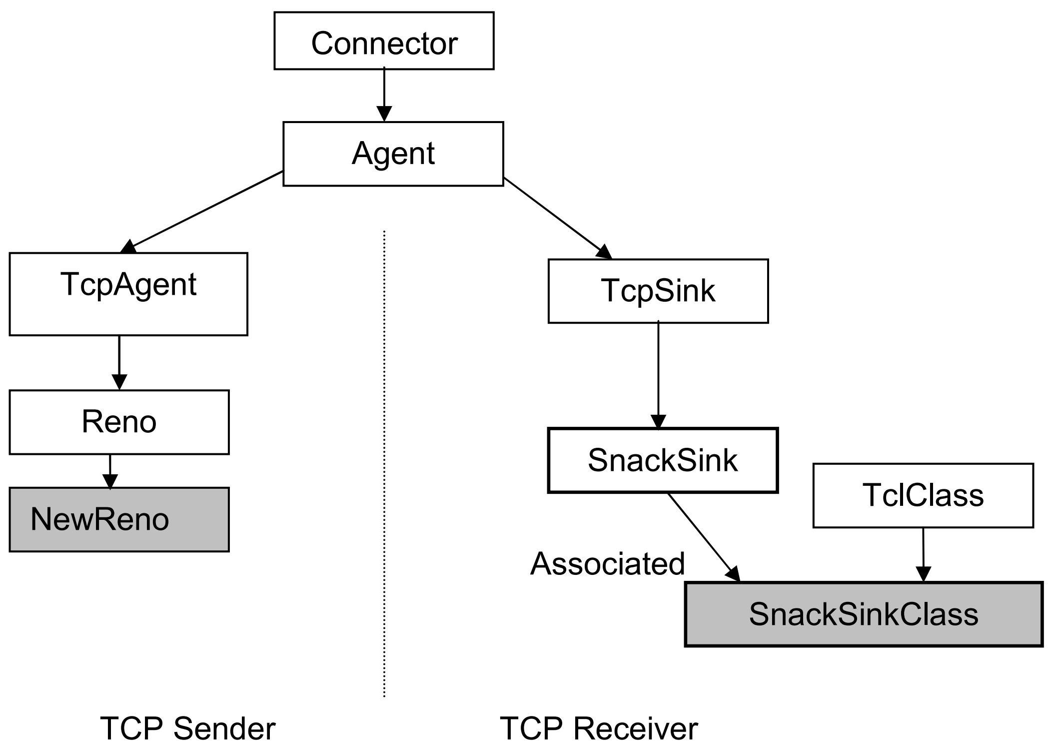
Selective-TCP algorithm: description

- In case of congestion loss:
  - sender's congestion window size is set according to the bandwidth measured at receiver
  - TCP's AIMD algorithm is prevented from setting congestion window lower than necessary
  - result: increased bandwidth utilization and higher goodput

AIMD: additive increase multiplicative decrease

goodput: the maximum sequence number of packets received by the destination host

Selective-TCP: implementation in ns-2



Extensions made:

tcp-newreno.cc
tcp-newreno.h } TCP sender

tcp-sink.cc
tcp.h } TCP receiver

Scenarios generated:

Tcl script files

D. Anantharaman, "Performance analysis of SNACK in satellite networks through simulation," M.S. Thesis, Lamar University, Lamar, TX, 2004.

Pseudo-code of Selective-TCP algorithm at the receiver

```
if (out-of-order packet received) {
        // check type of loss
        if ( wireless loss) {
                if (snack_delay = 0) // snack_delay is 50 ms
                        send SNACK
                else
                        do nothing
        }
        else { // congestion loss
                1) set congestion_count = congestion_count + 1
                2) set congestion_info = current bandwidth  measured
                at the TCP receiver
                        if (congestion_count = k) {
                        1) send congestion_info to the TCP sender
                                2) reset congestion_count
                        }
                        else
                        send ACK  // as in the case of TCP sink
        }
else // in-sequence packet received
        send ACK (same as TCP-sink)
```

Selective-TCP: module at receiver

- tcp-sink.cc:

```
void SnackSink::recv(Packet* pkt, Handler*)
{     ………………
  // code inserted by Rajashree
     prev_pkt_ts = present_pkt_ts;
     present_pkt_ts = Scheduler::instance().clock();
     tmin = present_pkt_ts - prev_pkt_ts;
     // T_min is the minimum packet inter-arrival time seen so far
     if (T_min > tmin)
            T_min = tmin;
   …………….
}
```

Pseudo-code of Selective-TCP at the TCP sender

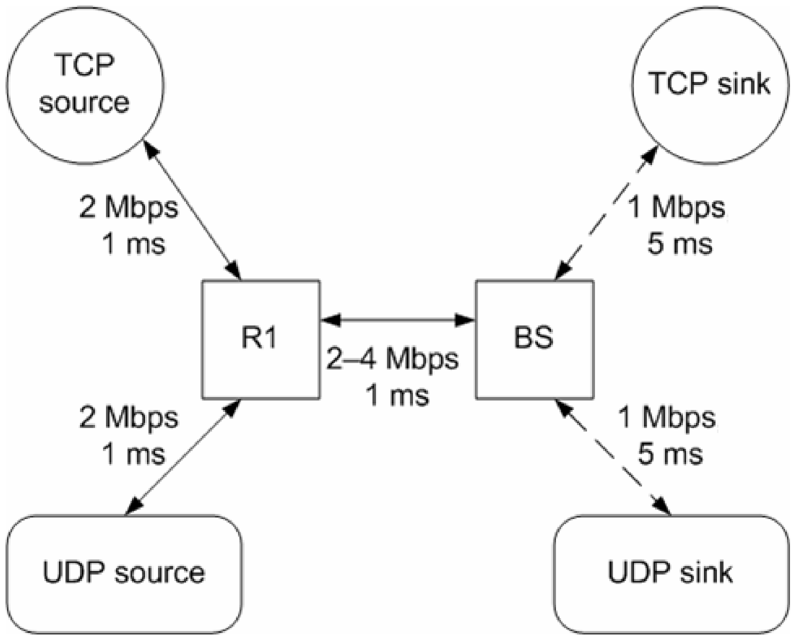
```
if (SNACK received) {
        1) retransmit lost packet(s) as indicated in SNACK
        2) reset retransmission timer
        }
else if (congestion_info ≠ 0) {
        //set size of congestion window as the bandwidth measured at receiver
        1) set cwnd_ = congestion_info * base_rtt
        //cwnd_ denotes congestion window size and
        //base_rtt is the initial round trip time
        2) reset congestion_info
        }
else //standard ACK received
        do as standard TCP NewReno sender
```

Selective-TCP: module at sender

- tcp-newreno.cc:

```
void NewRenoTcpAgent::recv(Packet *pkt, Handler*)
{
      ………….
      if (tcph->snack_option()) // Sending Snack
            processSnack(pkt);
      else if (tcph->congestion_info())  // Setting congestion window size
     {        ……….
              seq_num = tcph->seqno();
              cwnd_= tcph->congestion_info()/base_rtt;
              output(t_seqno_++,0);
              Packet::free(pkt);
      }
       …………….
 }
```

Simulation scenario: network topology



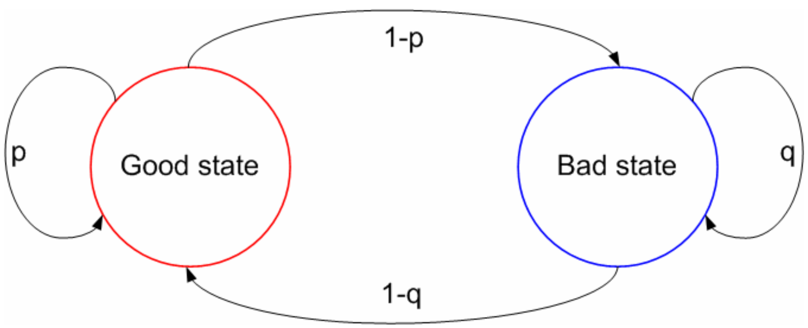
TCP source: sending rate = 2 Mbps
UDP source: sending rate = 512 kbps

R1: router
BS: base station

- wired link: bandwidth 2 or 4 Mbps, propagation delay 1 ms
- wireless link: bandwidth 1 Mbps, propagation delay 5 ms

Burst error model

- Two-state Markov model for modeling burst errors over wireless link



Transition probability matrix,

$$\pi = \begin{bmatrix} p & 1-p \\ 1-q & q \end{bmatrix}$$ and

burst error, $$\varepsilon = \frac{1-p}{2-p-q}$$

Simulation parameters:

good state: 0 packet loss

bad state: 1 packet loss

p = 0.9913 and q = 0.8509

ε = 5%

A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, "A Markov-based channel model algorithm for wireless networks," *Wireless Networks,* vol. 9, no. 3, pp. 189–199, May 2003.

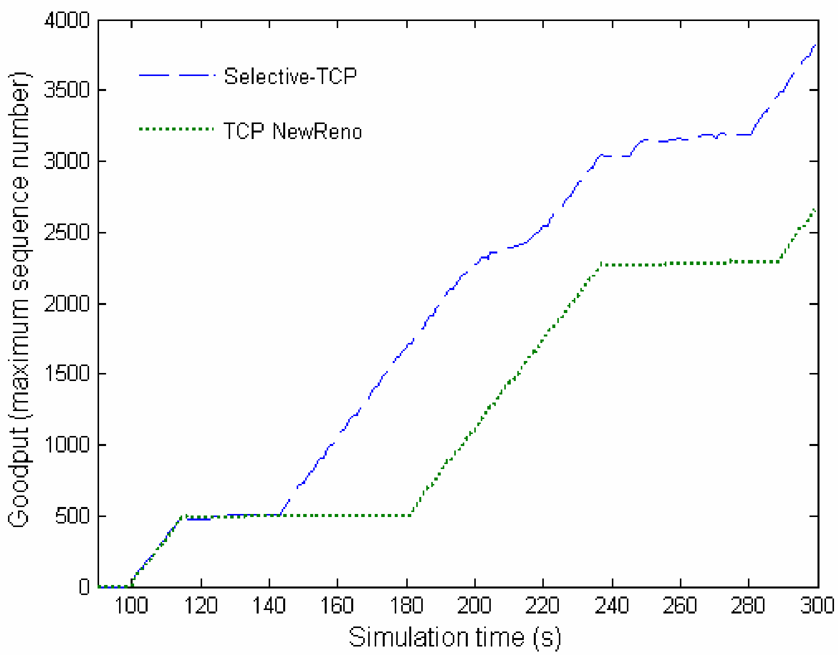
Simulation scenarios and parameters

- Simulation scenarios:
  - congested link
  - non-congested link
- Simulation parameters
  - burst error (5%): continuous lacking of data
  - random error (1%): random statistical error
  - no wireless error
- Performance measures:
  - throughput: the number of bits transmitted by the source per unit time (kbps)
  - goodput: the maximum sequence number of packets received by the destination host
  - congestion window size: size of the congestion window (kbytes)
- Selective-TCP is an extension of TCP NewReno and its performance is compared to TCP NewReno (almost 50% of the servers use NewReno)

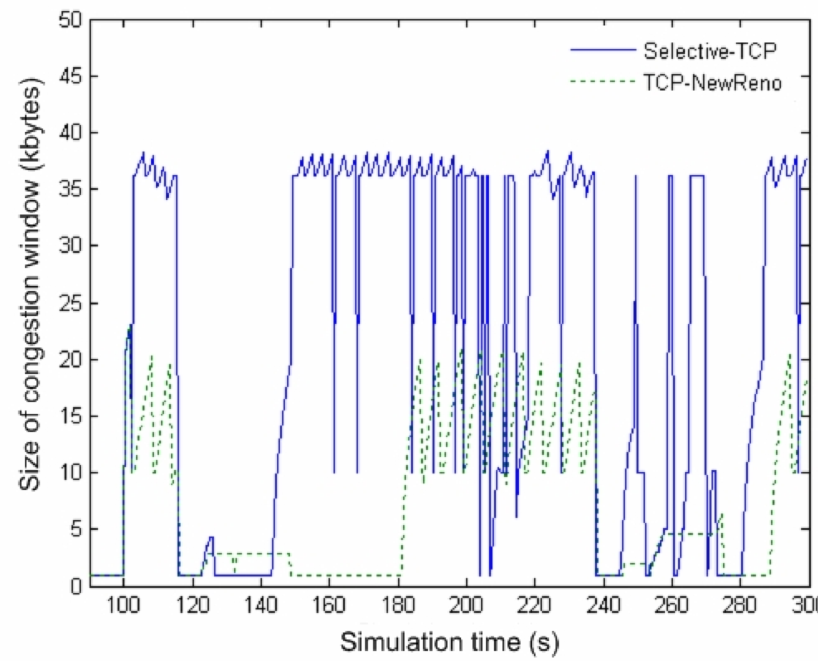
Simulation results: congested link

In the presence of congestion, with 5% burst error in the wireless links: significant increase in goodput and congestion window size

Goodput vs. simulation time



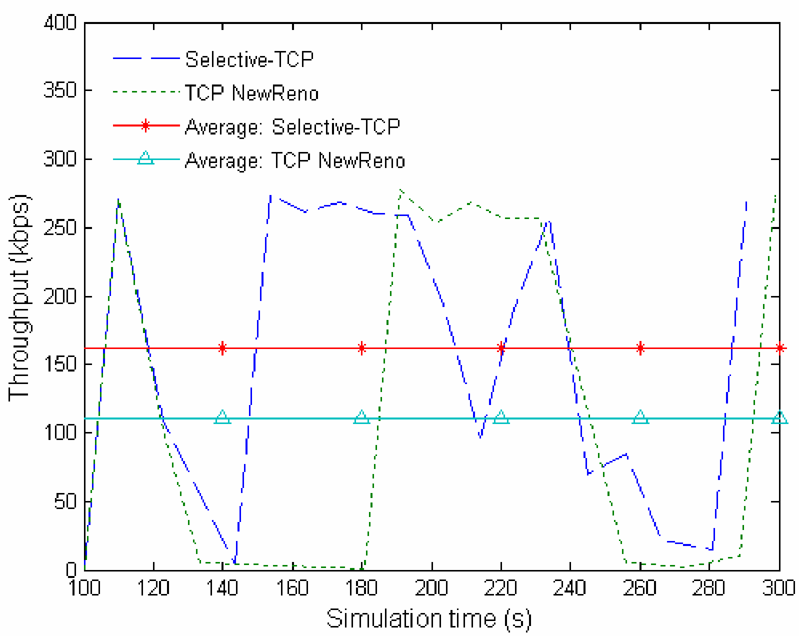
Congestion window vs. simulation time



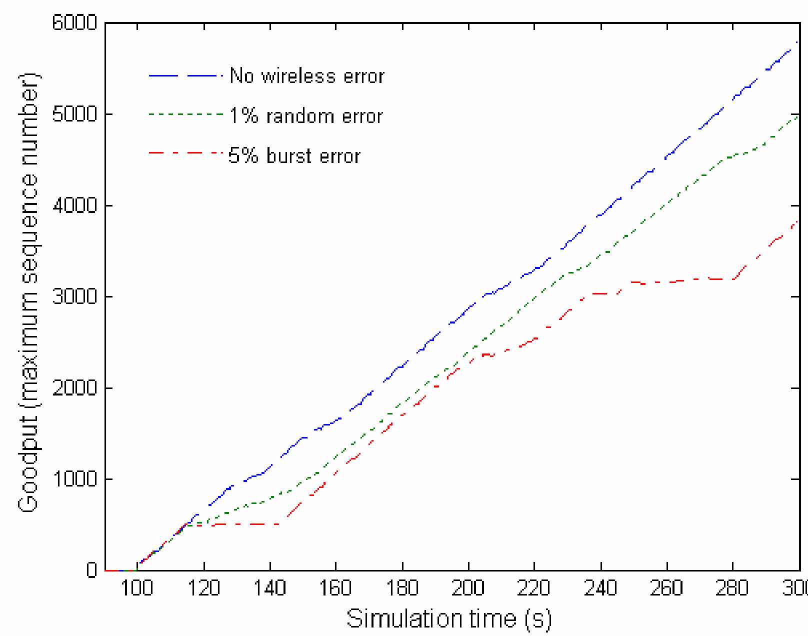
Simulation results: congested link

In the presence of congestion: average network throughput increases

Throughput vs. simulation time



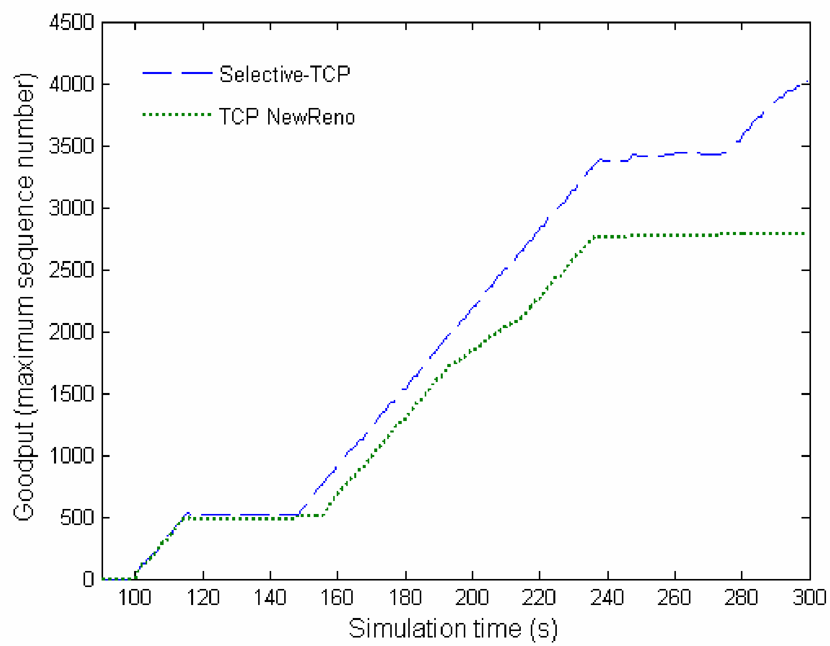
Goodput vs. simulation time



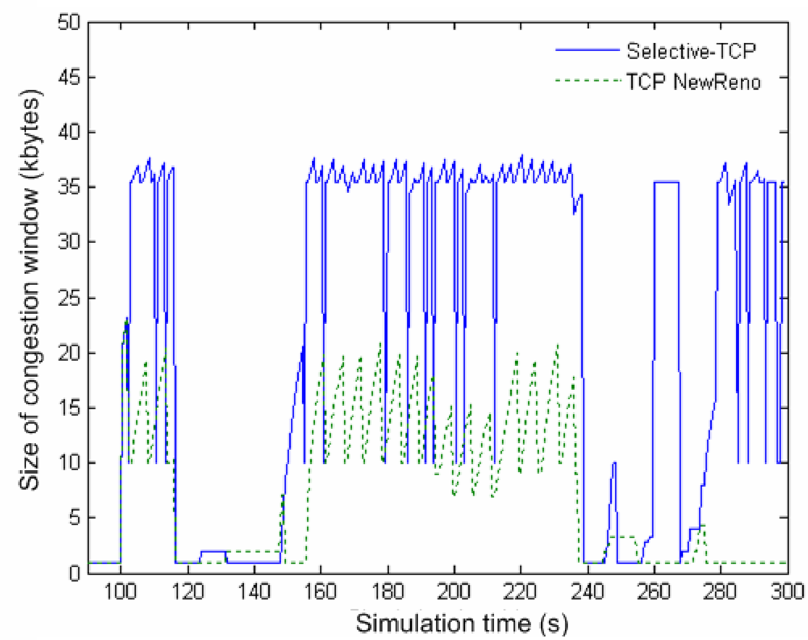
Simulation results: non-congested link

In the absence of congestion, with 5% burst error in the wireless links: increased goodput and congestion window size

Goodput vs. simulation time



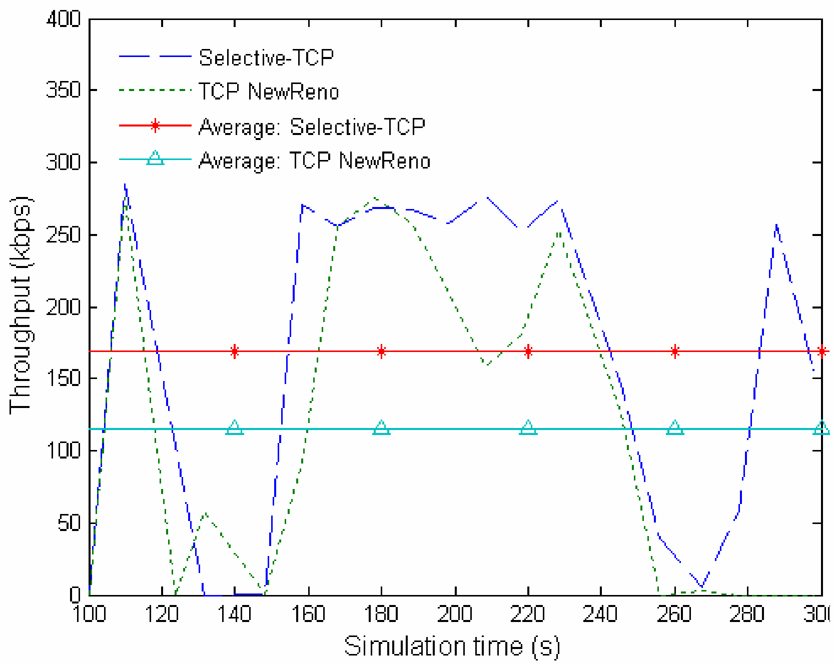
Congestion window vs. simulation time



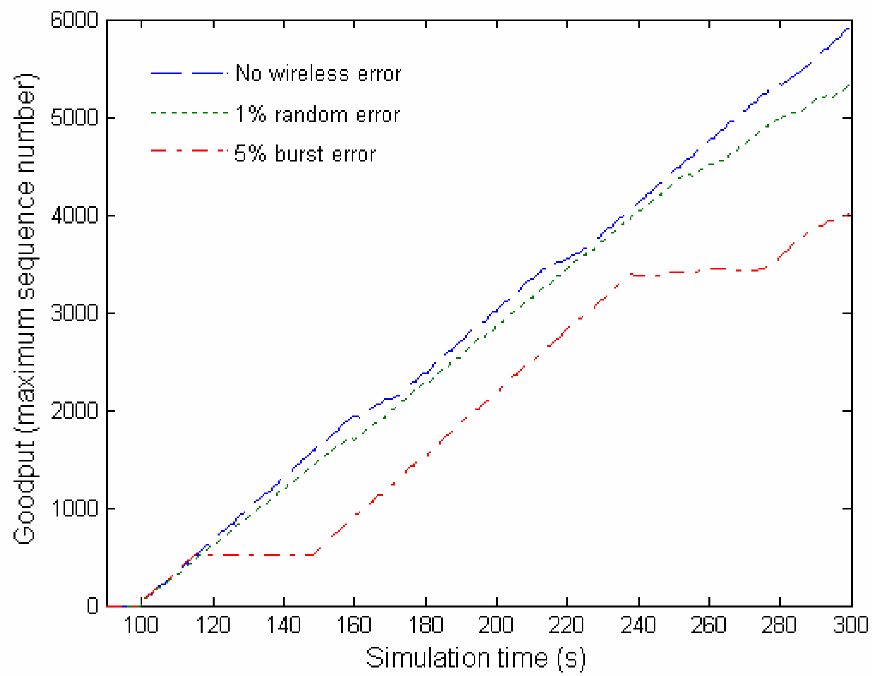
Simulation results: non-congested link

In the absence of congestion: average throughput increases by 45% compared to TCP NewReno

Throughput vs. simulation time



Goodput vs. simulation time



Roadmap

- Motivation
- Background and related work
- Selective-TCP:
  - algorithm
  - implementation
  - performance evaluation
- Performance comparison: Selective-TCP vs. TCP variants
- Conclusions and references

Comparison with TCP variants

- We compared Selective-TCP to:
  - TCP Reno
  - TCP NewReno
  - TCP SACK
  - TCP Westwood
  - TCP Packet Control algorithm

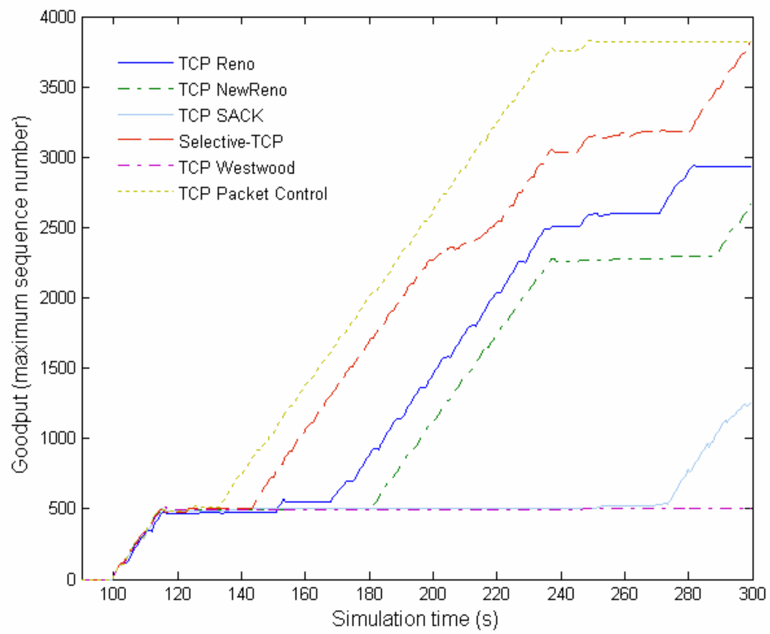
(TCP Reno, TCP NewReno, TCP SACK, TCP Westwood: end-to-end protocols)

(TCP Packet Control algorithm: link layer based protocol)

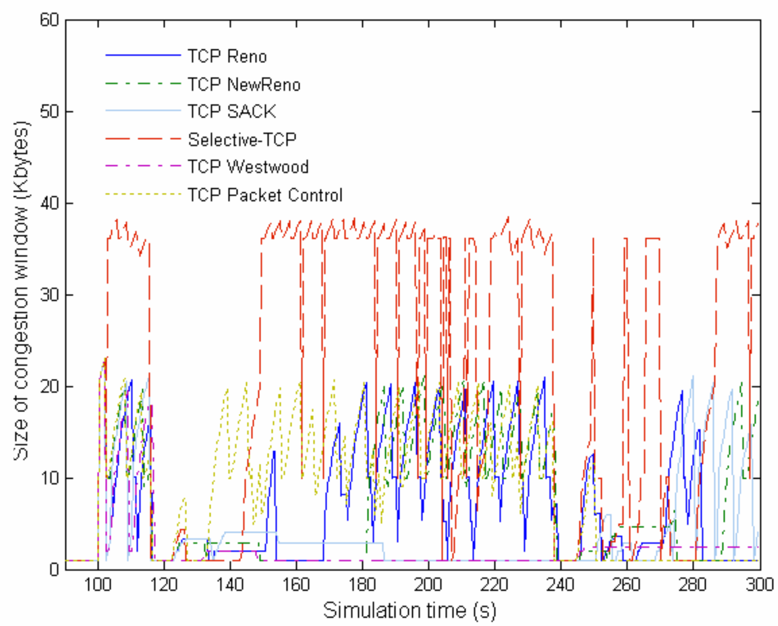
Simulation results: congested link

In the presence of congestion, with 5% burst error in the wireless links: TCP packet control algorithm and Selective-TCP achieve better goodput than other TCP variants

Goodput vs. simulation time



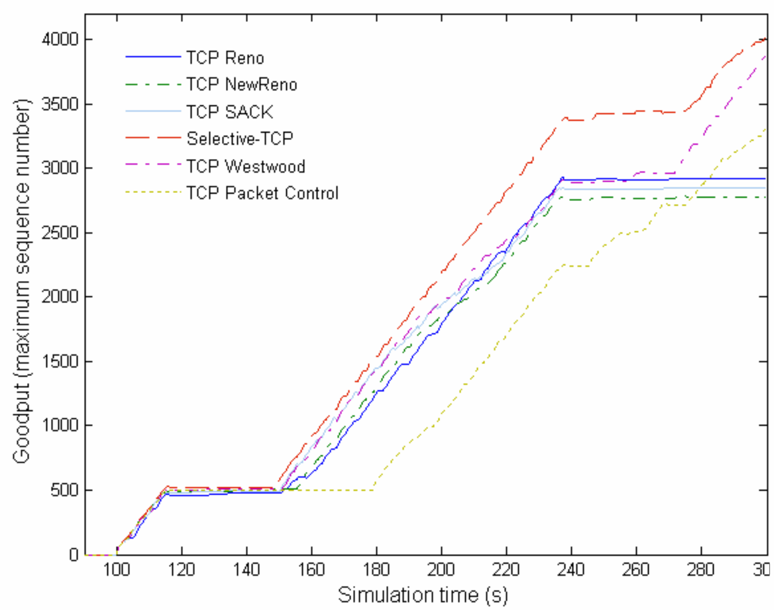
Congestion window vs. simulation time



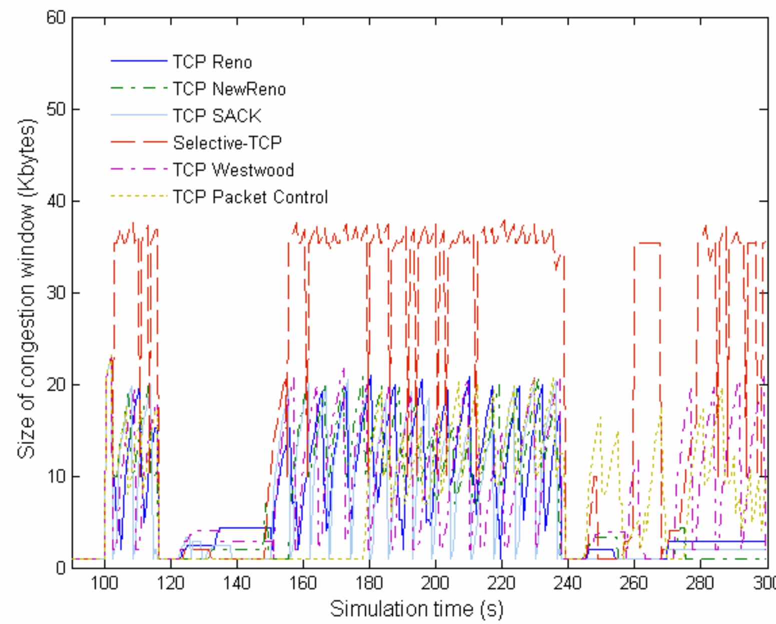
Simulation results: non-congested link

In the absence of congestion: Selective-TCP performs better than other TCP variants

Goodput vs. simulation time



Congestion window vs. simulation time



Roadmap

- Motivation
- Background and related work
- Selective-TCP
  - algorithm
  - implementation
  - performance evaluation
- Performance comparison: Selective-TCP vs. TCP variants
- Conclusions and references

Conclusions

- Selective-TCP:
  - is an end-to-end approach
  - distinguishes wireless losses from congestion losses
  - takes corrective action depending on type of losses
  - improves goodput up to 45% in mixed wired/wireless networks in the presence of 5% burst error, when compared to TCP NewReno

Conclusions

- Compared to TCP variants, Selective-TCP:
  - shows better bandwidth utilization than other protocols
  - performs better in absence of congested link in the network and achieves high goodput in both cases: congestion and no congestion
  - achieves the highest goodput among the end-to-end protocols

References

1. S. Biaz and N. H. Vaidya, "Discriminating congestion losses from wireless losses using inter-arrival times at the receiver," in *Proc. IEEE Symposium on Application-Specific Systems and Software Engineering and Technology*, Richardson, TX, Mar. 1999, pp. 10–17.
2. Consultative Committee for Space Data Systems, *Space Communications Protocol Specification—Transport Protocol (SCPS-TP)*, Blue Book, issue 1, May 1999.
3. D. Anantharaman, "Performance analysis of snack in satellite networks through simulation," M.S. Thesis, Lamar University, Lamar, TX, 2004.
4. H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *Computer Communication Review*, vol. 26, no. 4, pp. 256–269, Aug. 1996.
5. J. McDougall and S. Miller, "Sensitivity of wireless network simulations to a two-state Markov model channel approximation," in *Proc. GLOBECOM*, San Francisco, CA, Dec. 2003, pp. 697–701.
6. A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, "A Markov-based channel model algorithm for wireless networks," *Wireless Networks,* vol. 9, no. 3, pp. 189–199, May 2003.
7. ns-2 [Online]. Available: http://www.isi.edu/nsnam/ns.

8. M. Allman, V. Paxson, and W. Stevens, "TCP congestion control" *IETF RFC 2581*, Apr. 1999.
9. S. Floyd and T. Henderson, "The NewReno modification to TCP's fast recovery algorithm," *IETF RFC 2582*, Apr. 1999.
10. M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgement options," *IETF RFC 2018*, Apr. 1996.
11. C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP Westwood: end-to-end congestion control for wired/wireless networks," *Wireless Networks*, vol. 8, no. 5, pp. 467–479, Sept. 2002.
12. W. G. Zeng and Lj. Trajkovic, "TCP packet control for wireless networks," in *Proc. IEEE Int. Conf. on Wireless and Mobile Computing, Networking and Communications* (*WiMob 2005*), Montreal, Canada, Aug. 2005, pp. 196–203.