CS 525M – Mobile and Ubiquitous Computing Seminar

TCP Westwood

Written by S. Mascolo, C. Casetti,
M. Gerla, M. Sanadidi and R. Wang
Presented by Choong-Soo Lee

Introduction

- TCP suffers performance degradation in wireless environment
  - TCP uses dropped packets as a congestion signal but packets can be lost due to other reasons in wireless environment.
  - End-to-end, link layer and split connect approaches
- Local recovery is more effective than end-to-end approach in handling wireless losses but end-to-end requires minimum support from routers and base stations
- Authors propose TCP Westwood, an end-to-end solution to handle wireless losses

Outline

- Introduction
- **TCP Westwood**
- Evaluation
- Conclusion and Future Work

ACK-based measurement

- Source performs an end-to-end estimate of the available bandwidth
- Enables *faster recovery*
- When an ACK is received, it uses the corresponding data packet size to compute the bandwidth estimate
- In case of a DUPACK, an average segment size is used instead.

Filtering ACK Reception Rate

- Sample of Bandwidth Estimate

$$b_k = \frac{d_k}{t_k - t_{k-1}}$$

- We need a low-pass filter to average sampled measurements and to obtain the low frequency components of the samples.

$$\hat{b}_k = \frac{\frac{2t}{t_k - t_{k-1}} - 1}{\frac{2t}{t_k - t_{k-1}} + 1} \hat{b}_{k-1} + \frac{b_k + b_{k-1}}{\frac{2t}{t_k - t_{k-1}} + 1}$$

$1/t$: cut-off frequency

Filtering ACK Reception Rate

- An Example
  - Consider constant time interval

$$\hat{b}_k = a \cdot \hat{b}_{k-1} + \frac{(1-a)}{2}(b_k + b_{k-1})$$

where a = 0.9

- Real ACKs are irregular and we need to ensure that we have a sample within a certain interval.
  - Use of a virtual sample of 0

$$\hat{b}_{k+h} = \left(\frac{2m-1}{2m+1}\right)^h \hat{b}_k$$

Delayed and Cumulative ACKs

- Delayed ACK
  - Send ACK every other in-sequence segment
- Cumulative ACK
  - Acknowledges all segments prior to its sequence number

Delayed and Cumulative ACKs

```
cumul_ack =current_ack_seqno – last_ack_seqno;
If (cumul_ack = 0)
      accounted_for++;
      cumul_ack = 1;
endif

if (cumul_ack > 1)
      if (accounted_for >= cumul_ack)
            accounted_for = accounted_for – cumul_ack;
            cumul_ack = 1;
      else if (accounted_for < cumul_ack)
            cumul_ack = cumul_ack – accounted_for;
            accounted_for = 0;
      endif
endif

last_ack_seqno = current_ack_seqno;
acked = cumul_ack;

return acked;
```

n DUPACKs and Timeout

- n duplicate ACKs and coarse timeout imply that there is congestion in the network
  - ssthresh set to the available pipe size

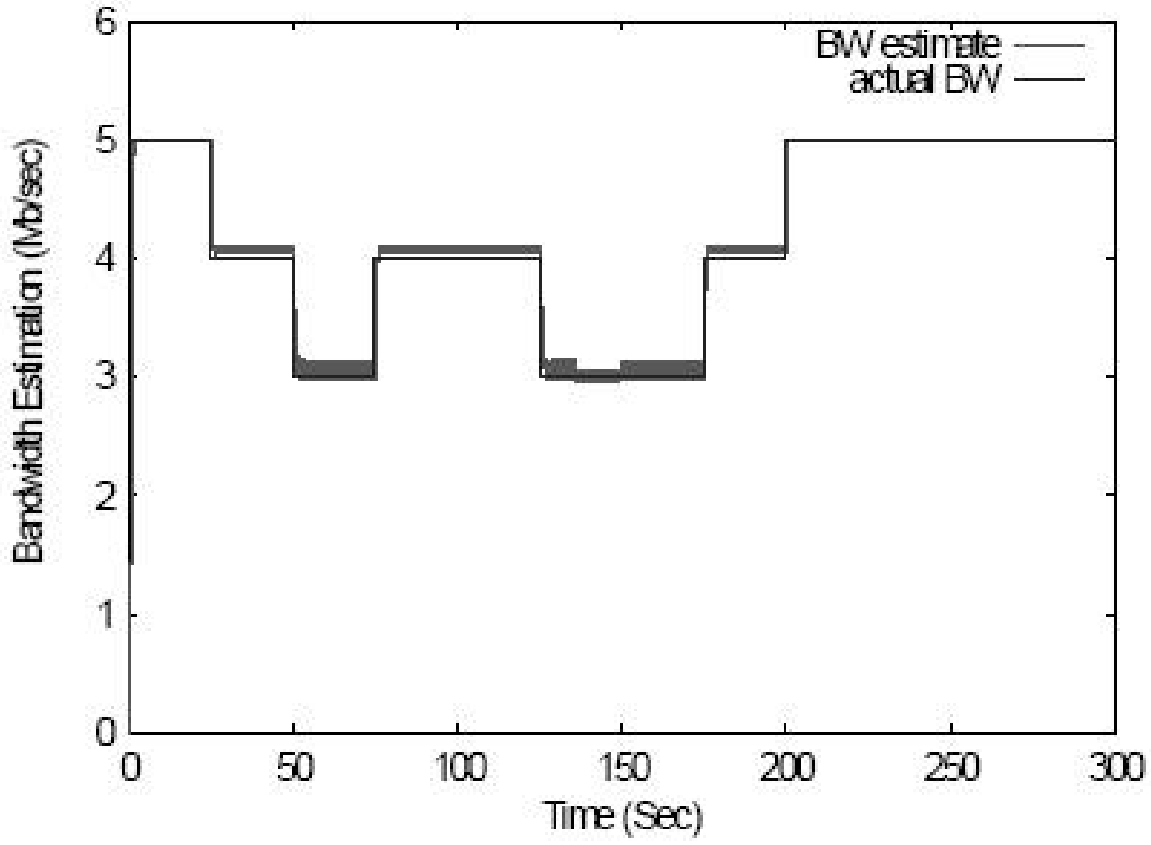
```
if (n DUPAcKs are received)
    ssthresh = (BWE * RTTmin) / seg_size;
    if (cwin > ssthresh)
        cwin = ssthresh;
    endif
endif
If (coarse timeout expires)
    ssthresh = (BWE * RTTmin) / seg_size;
    if (ssthresh < 2)
        ssthresh = 2;
    endif
    cwin = 1;
endif
```

Outline

- Introduction
- TCP Westwood
- **Evaluation**
- Conclusion and Future Work

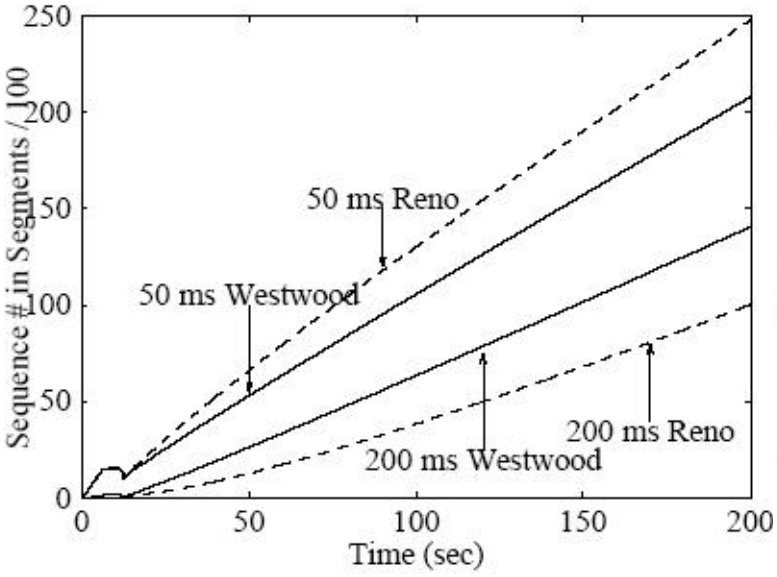
Bandwidth Estimation

- TCP Westwood with concurrent UDP Traffic

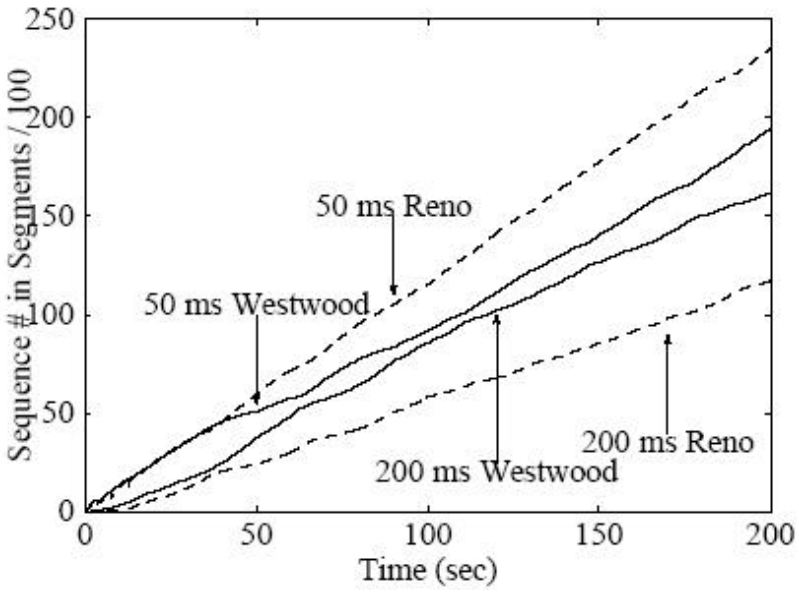


Fairness

- Fairness: flows using the same protocol should get equal share of bandwidth



without RED



with RED

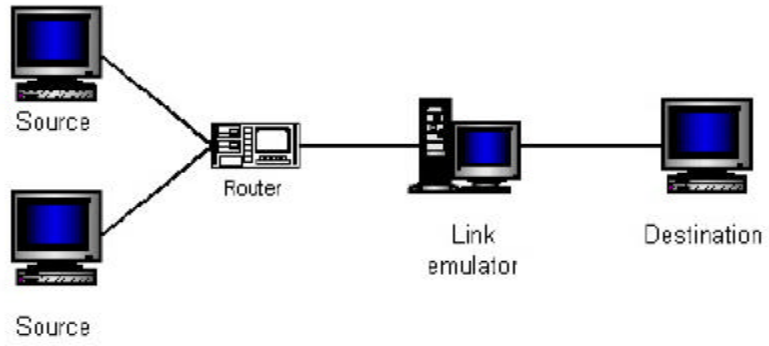
Friendliness (Simulation)

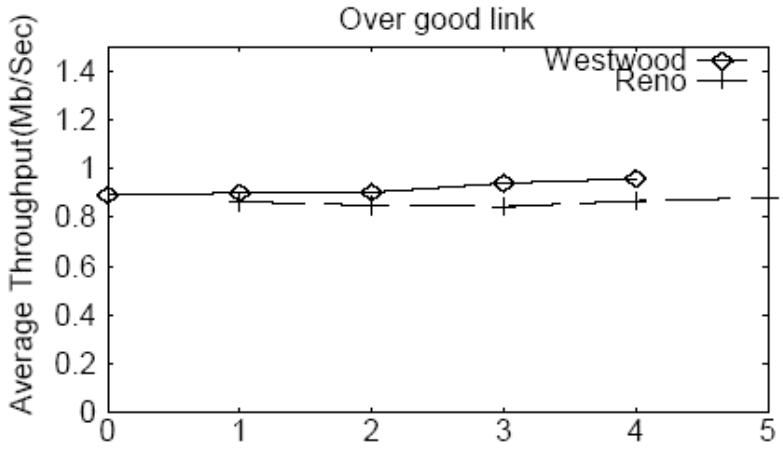
- Friendliness: protocols using different protocols should get equal share of bandwidth
- Simulation setup
  - 2 Mbps bottleneck bandwidth with 100ms round-trip time

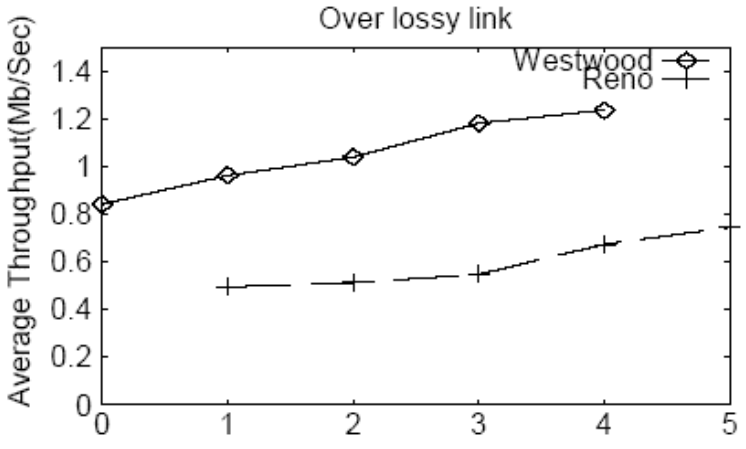
Number of Flows		Achieved Throughput (Mbps)	
TCP Reno	**TCP Westwood**	**TCP Reno**	**TCP Westwood**
20	0	0.0992	-
0	20	-	0.0994
10	10	0.0913	0.1078

Friendliness (Testbed)

- Testbed implementation using Linux

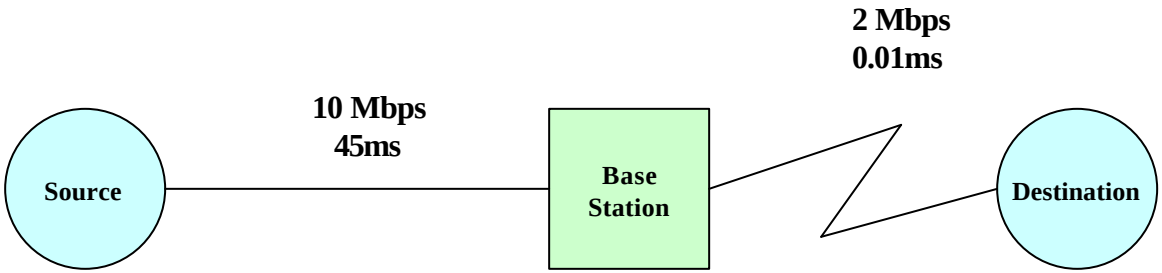






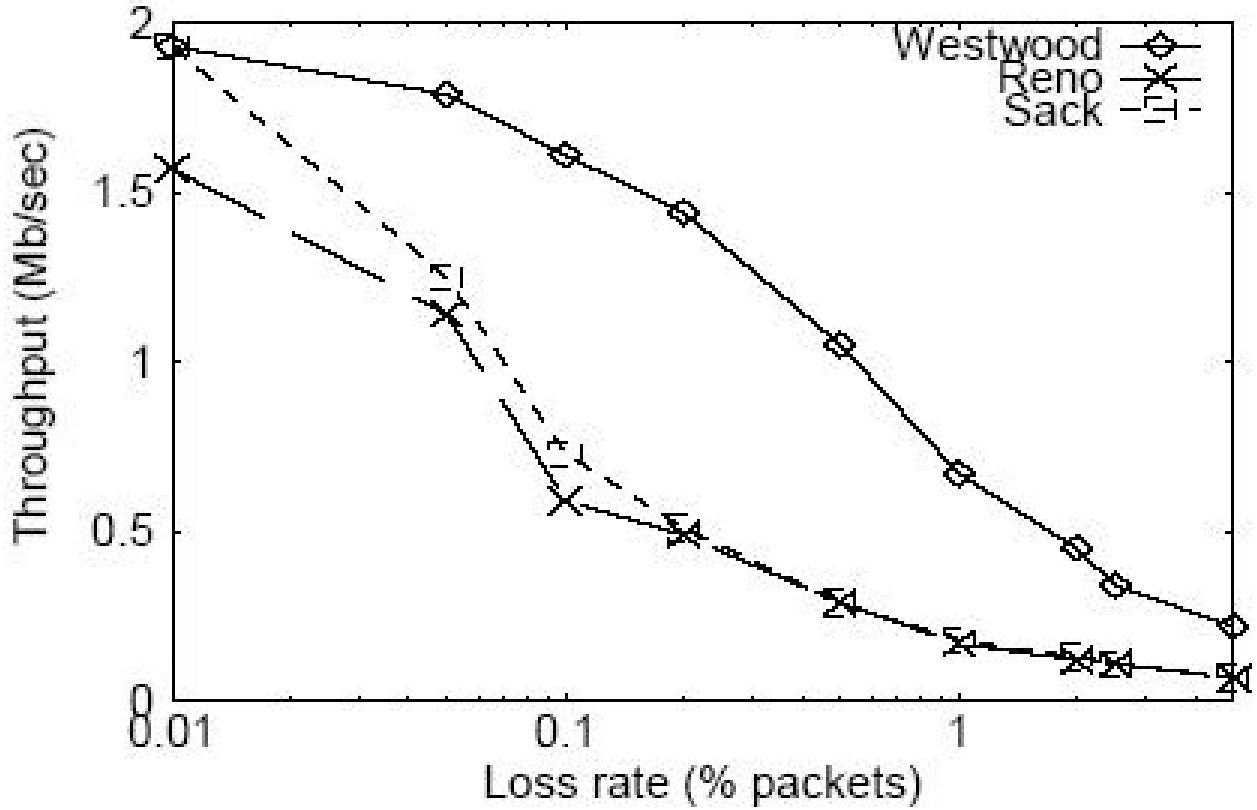
Performance (Setup)

- Independent loss model in ground radio environment



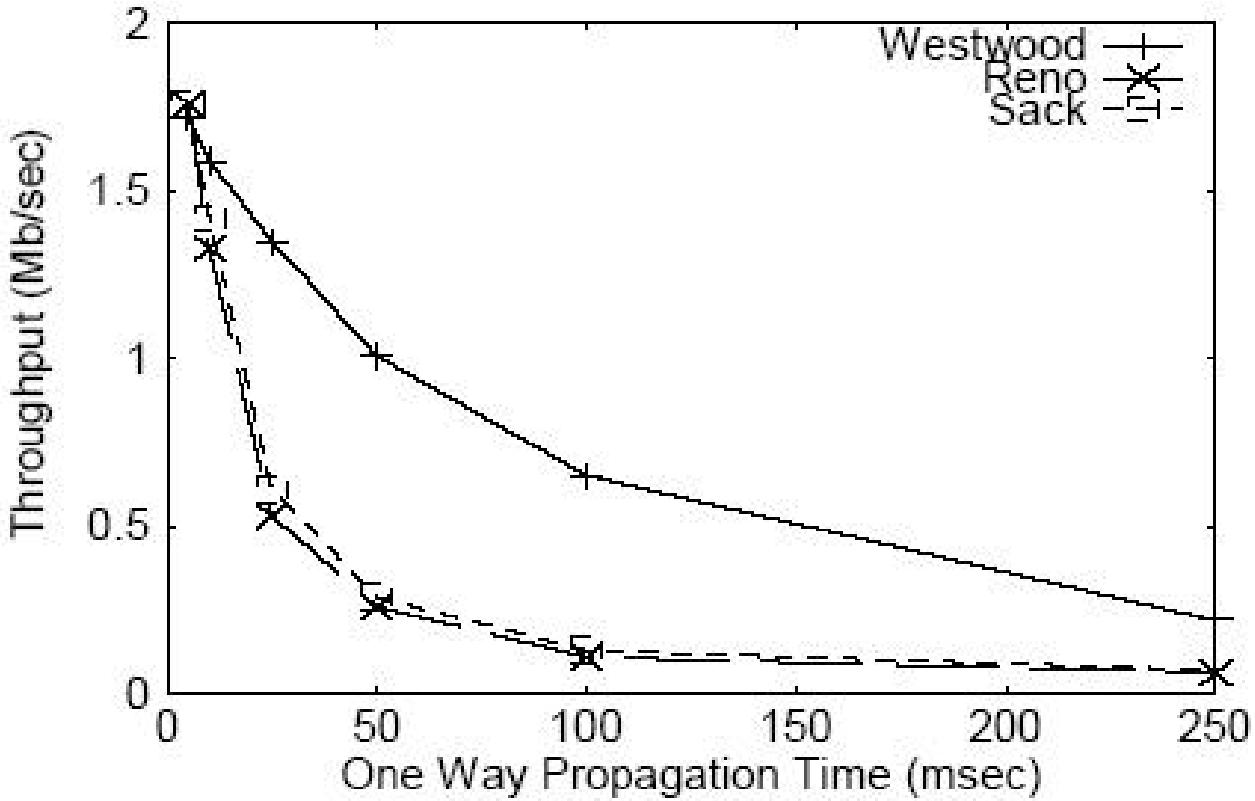
Performance

- Throughput vs. error rate



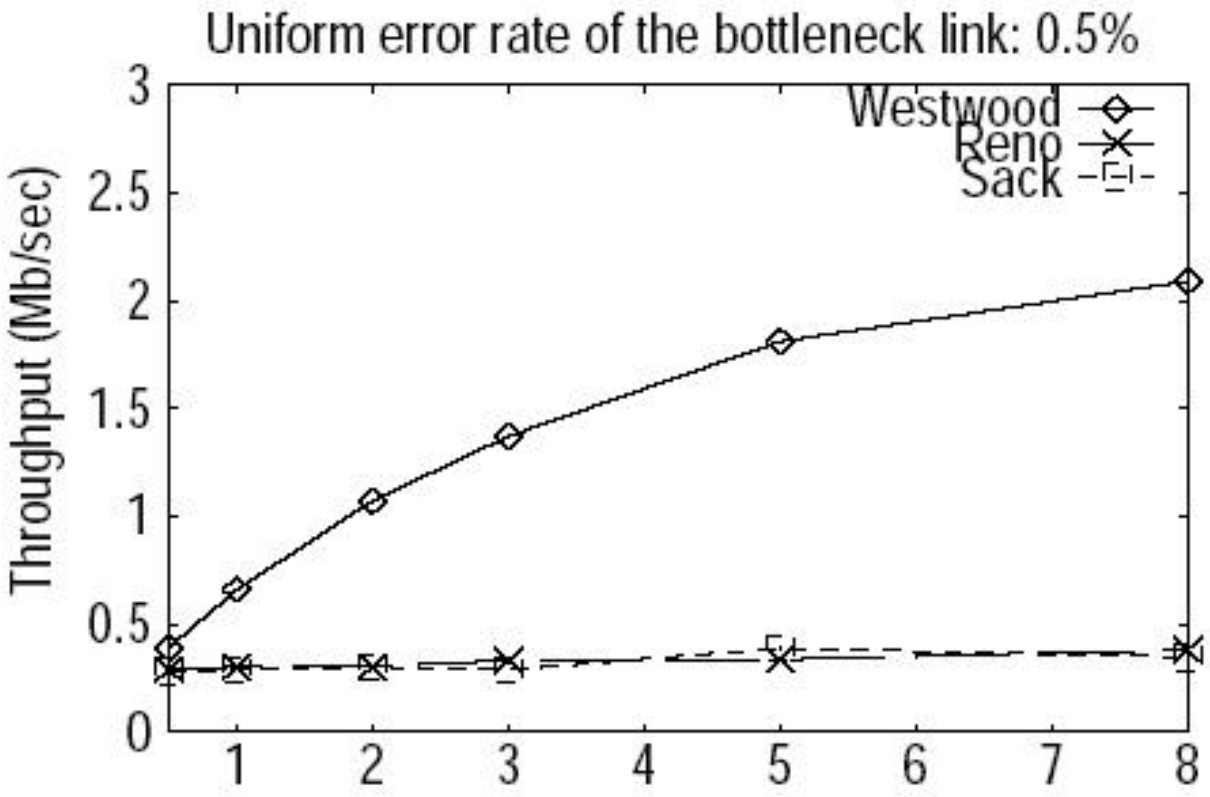
Performance

- Throughput vs. one-way propagation delay



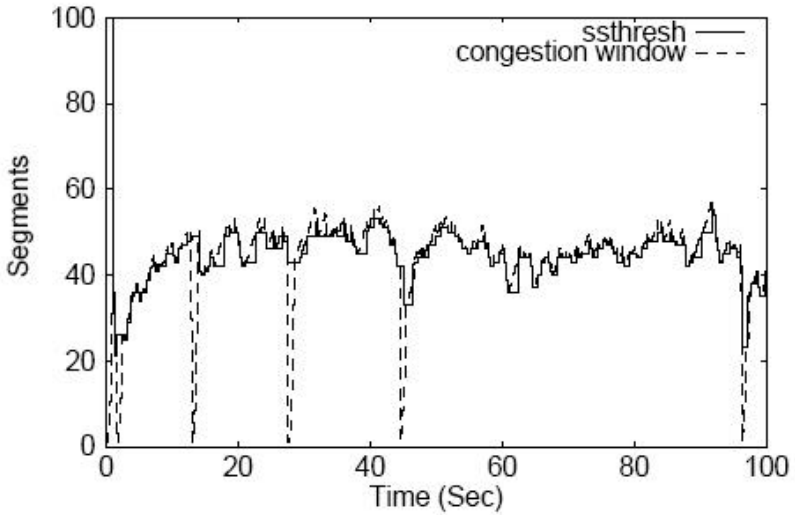
Performance

- Throughput vs. link capacity

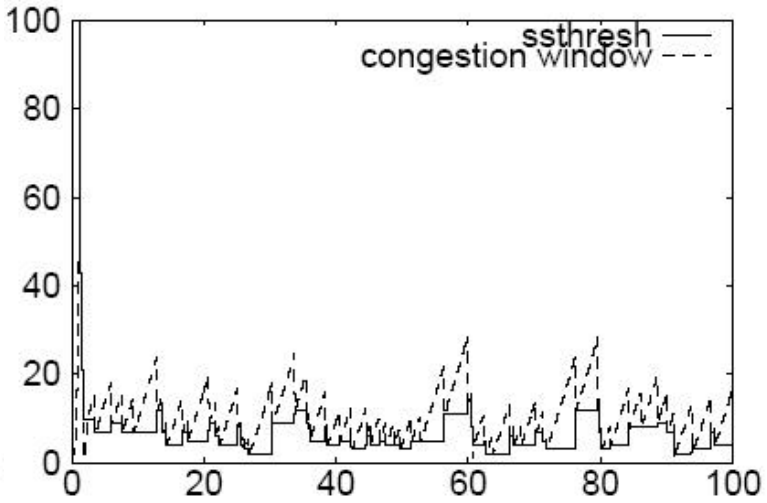


Performance

- cwin and ssthresh



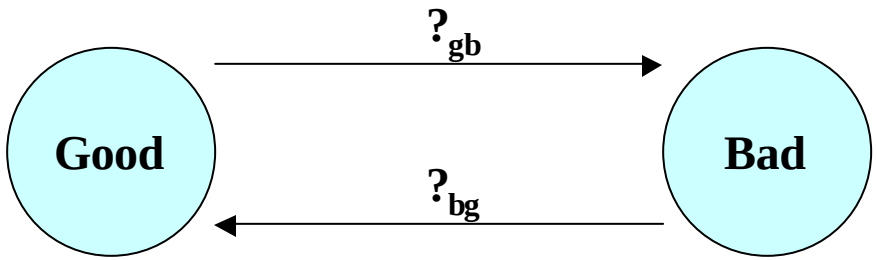
TCP Westwood



TCP Reno

Performance

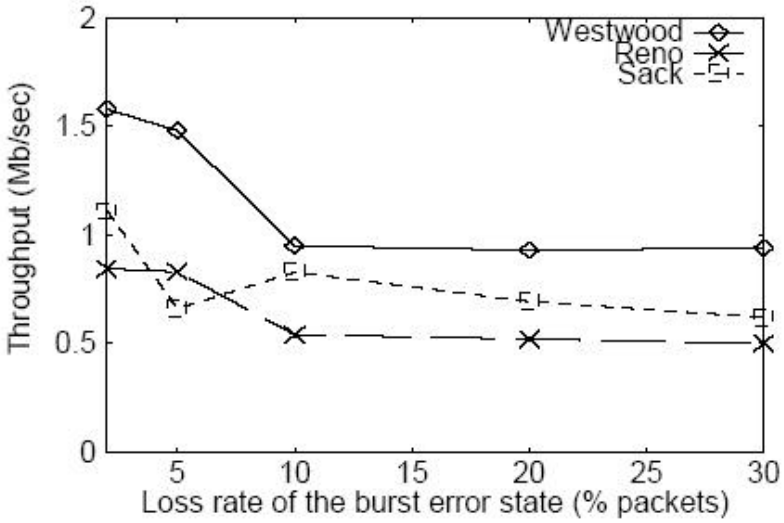
- Burst error models
  - Two state Markov model



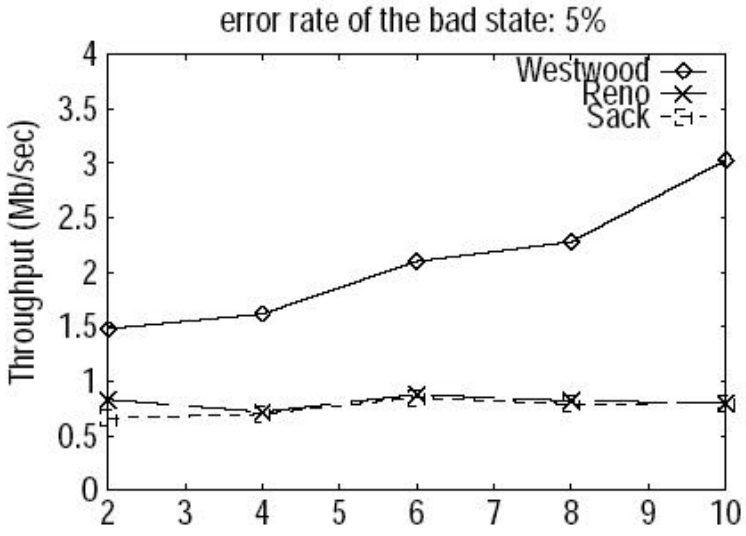
  - It is used to model fading and blackout.

Performance

- Fading
  - Good State (8 sec, 0.001%)
    Bad State (4 sec, 0 to 30%)
  - Throughput vs. error rate



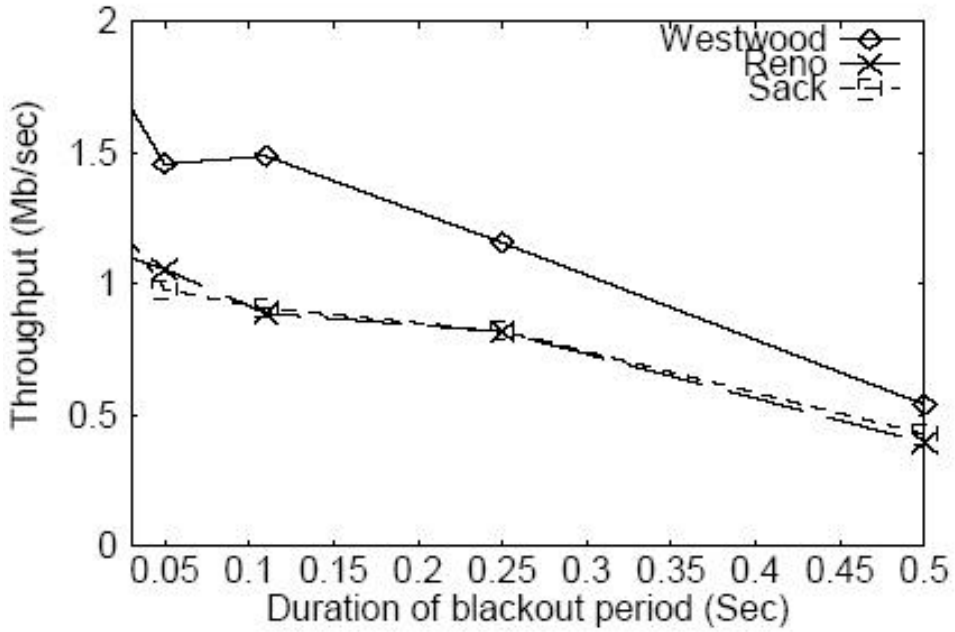
Bad State



Two-State Model

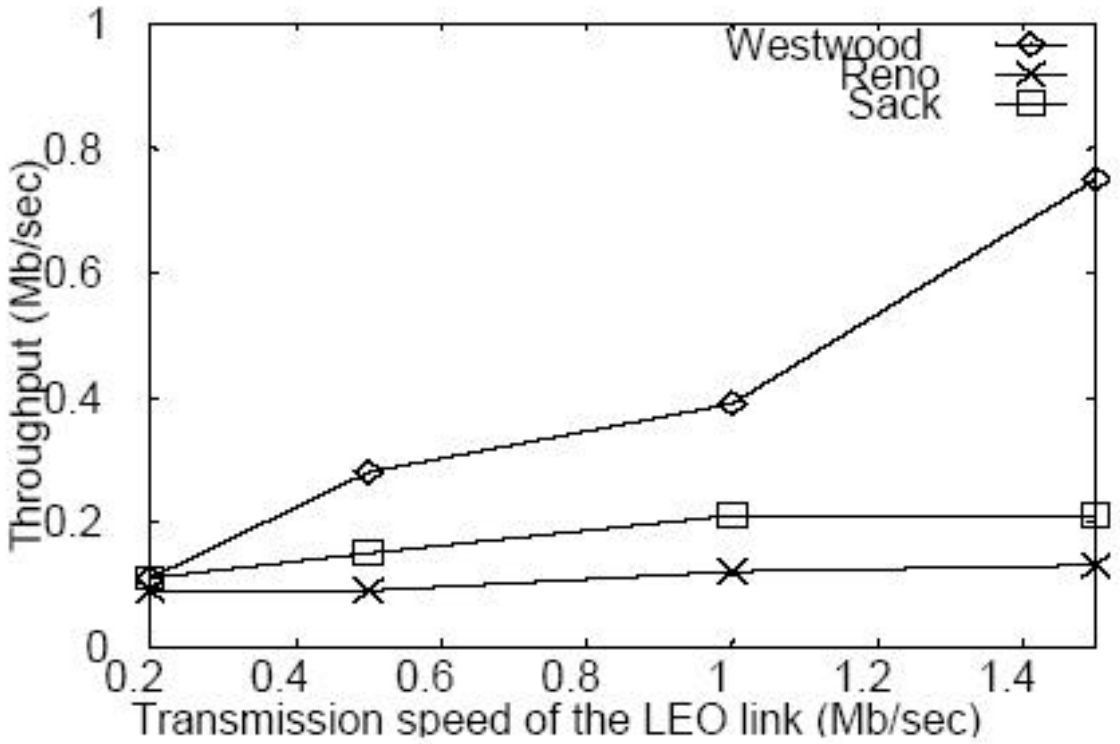
Performance

- Blackout
  - Good state (4sec)
  - Bad state (0 to 0.5 sec) (blackout)
  - Throughput vs. average duration



Performance

- Throughput vs. link capacity of the Satellite link



Outline

- Introduction
- TCP Westwood
- Evaluation
- **Conclusion and Future Work**

Conclusion

- The authors proposed a new version of TCP, TCP Westwood.
  - TCP Westwood offers *faster recovery* without over-shrinking cwin
  - TCP Westwood's congestion control relies on both packet losses and bandwidth estimation
  - TCP Westwood improves performance over wide range of wireless/wired scenarios

Future Work

- Include TCP NewReno feature for recovering from multiple losses in the same window
- Backward path could be the bottleneck
  - Need to define fairness between DATA and ACK streams
- More comparison between TCP Westwood and link-layer techniques
- Further refine bandwidth estimation and filter method for better friendliness
- Develop control theoretical model to study TCP Westwood's stability

Questions