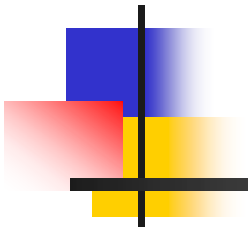
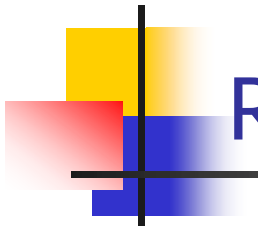


Modeling TCP/RED: a dynamical approach

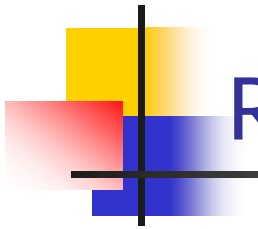
Ljiljana Trajković
ljilja@cs.sfu.ca

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



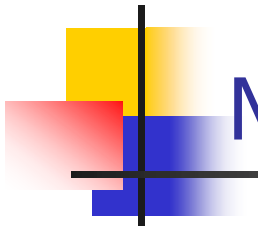
Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references



Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references



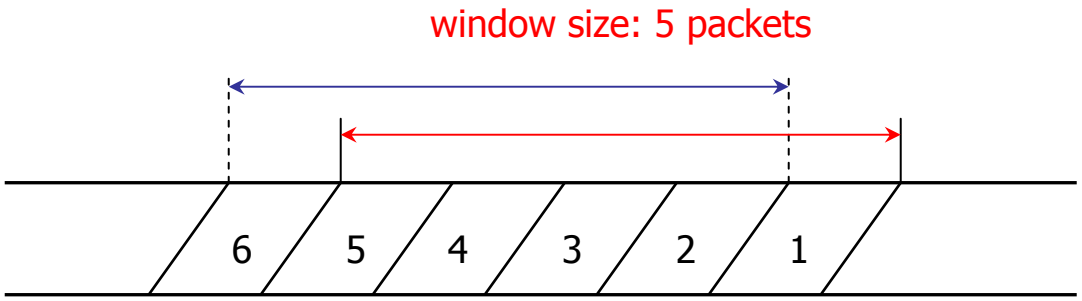
Motivation

- Modeling TCP Reno with RED:
  - examine the interactions between TCP and RED
  - understand and predict the dynamical network behavior
  - analyze the impact of system parameters

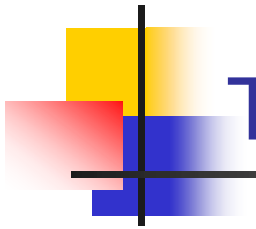
RED: Random Early Detection Gateways for Congestion Avoidance

TCP

- TCP: Transmission Control Protocol
- Fourth layer of the OSI model
- Connection oriented, reliable, and byte-stream service
- Employs window based flow and congestion control algorithms



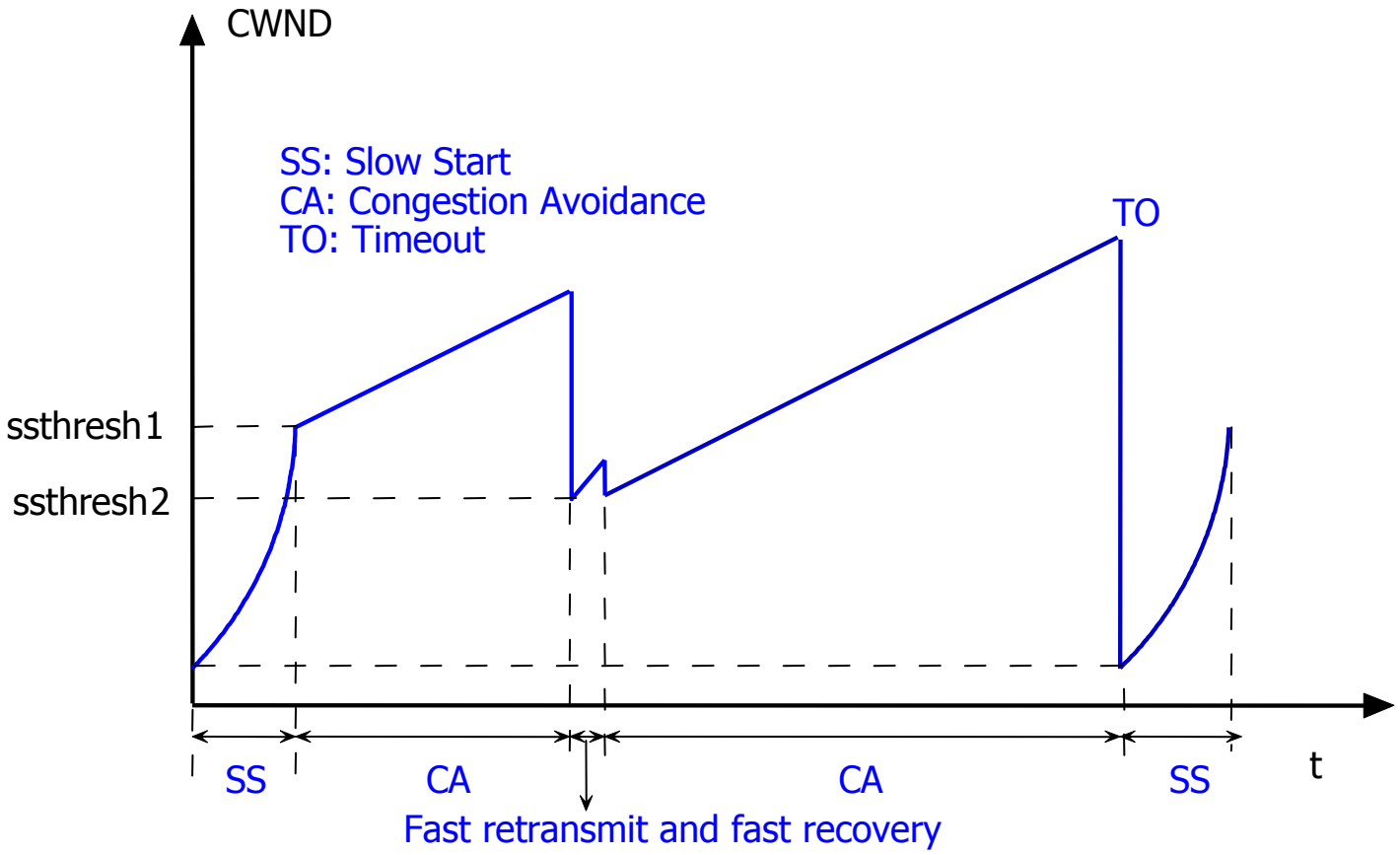
OSI: Open System Interconnection reference model

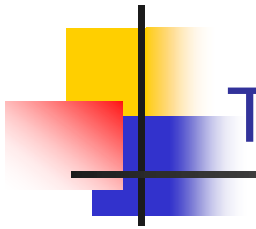


TCP

- Several flavors of TCP:
  - Tahoe: 4.3 BSD Tahoe (~ 1988)
    - slow start, congestion avoidance, and fast retransmit (RFC 793, RFC 2001)
  - Reno: 4.3 BSD Reno (~ 1990)
    - slow start, congestion avoidance, fast retransmit, and fast recovery (RFC 2001, RFC 2581)
  - NewReno (~ 1996)
    - new fast recovery algorithm (RFC 2582)
  - SACK (~ 1996, RFC 2018)

TCP Reno


CWND
SS: Slow Start
CA: Congestion Avoidance
TO: Timeout
TO
ssthresh1
ssthresh2
SS
CA
CA
SS
t
Fast retransmit and fast recovery



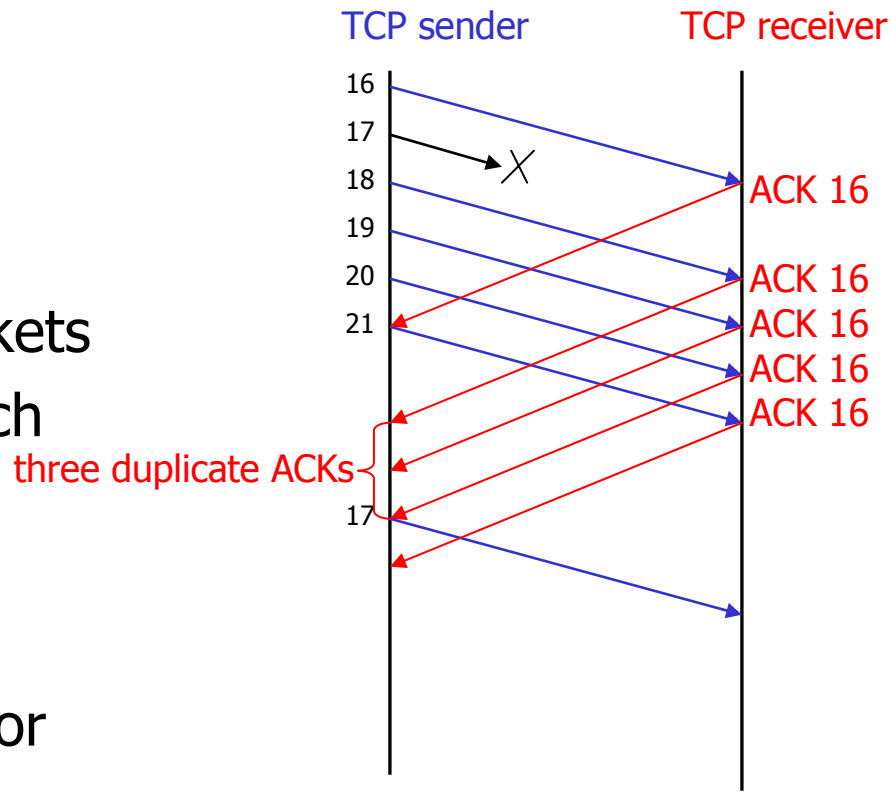
TCP Reno: slow start and congestion avoidance

- Slow start:
  - *cwnd* = *IW* (1 or 2 packets)
  - when *cwnd* < *ssthresh*
    cwnd = *cwnd* + 1 for each received *ACK*
- Congestion avoidance:
  - when *cwnd* > *ssthresh*
    cwnd = *cwnd* + 1/*cwnd* for each *ACK*

cwnd : congestion window size
IW : initial window size
ssthresh : slow start threshold
ACK : acknowledgement
RTT : round trip time

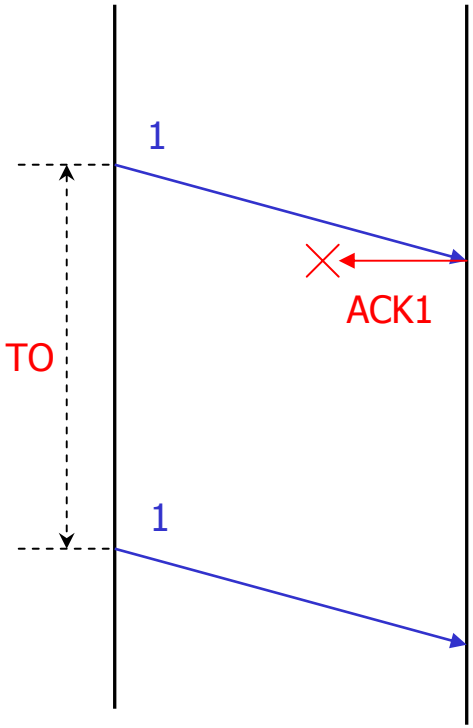
TCP Reno: fast retransmit and fast recovery

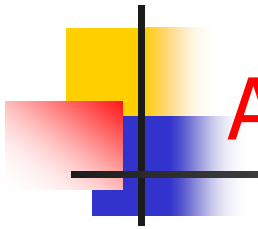
- three duplicate *ACKs* are received
- retransmit the packet
- *ssthresh* = *cwnd*/2, *cwnd* = *ssthresh* + 3 packets
- *cwnd* = *cwnd* + 1, for each additional duplicate ACK
- transmit the new data, if *cwnd* allows
- *cwnd* = *ssthresh*, if ACK for new data is received



TCP Reno: timeout

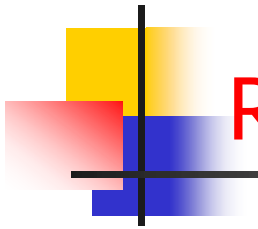
- TCP maintains a retransmission timer
- The duration of the timer is called retransmission timeout
- Timeout occurs when the ACK for the delivered data is not received before the retransmission timer expires
- TCP sender retransmits the lost packet
- *ssthresh* = *cwnd/2*
  cwnd = 1 or 2 packets





AQM: Active Queue Management

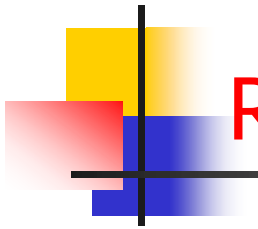
- AQM (RFC 2309):
  - reduces bursty packet drops in routers
  - provides lower-delay interactive service
  - avoids the “lock-out” problem
  - reacts to the incipient congestion before buffers overflow
- AQM algorithms:
  - RED (RFC 2309)
  - ARED, CHOKe, BLUE, …



RED

- Random Early Detection Gateways for Congestion Avoidance
  - Proposed by S. Floyd and V. Jacobson, LBN, 1993.

    S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.
  - Main concept: drop packets before the queue becomes full



RED variables and parameters

- Main variables and parameters:
  - average queue size: $\bar{q}$
  - instantaneous queue size: q
  - drop probability: p_a
  - queue weight: w_q
  - maximum drop probability: $p_{\max}$
  - queue thresholds: $q_{\min}$ and $q_{\max}$

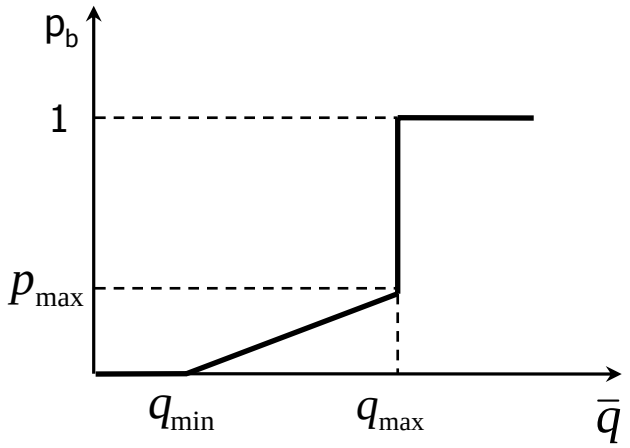
RED algorithm

Calculate:

- average queue size for each packet arrival

$$\overline{q} = (1 - w_q) \cdot \overline{q} + w_q \cdot q$$

- drop probability



RED algorithm: drop probability

- if $(q_{\min} < \bar{q} < q_{\max})$

$$p_b = p_{\max} \times \frac{\bar{q} - q_{\min}}{q_{\max} - q_{\min}} \qquad p_a = \frac{p_b}{1 - count \times p_b}$$

count: number of packets that arrived since the last packet drop

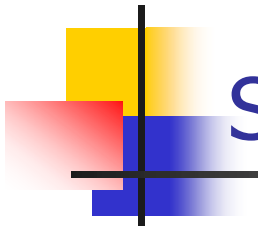
- else if $(\bar{q} > q_{\max})$

$$p_a = 1$$

- else $(\bar{q} < q_{\min})$

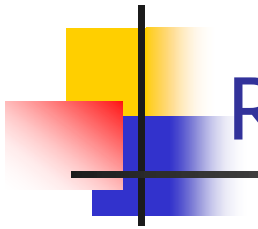
$$p_a = 0$$

- Mark or drop the arriving packet with probability p_a



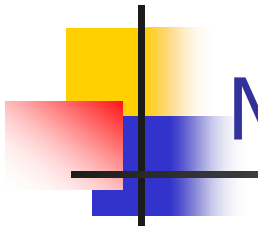
Simulation tool: ns-2

- ns-2 is a discrete event network simulator
  http://www.isi.edu/nsnam/ns
- Supports simulation of TCP, routing, and multicast protocols over wired and wireless networks
- We used ns-2 to validate the proposed S-TCP/RED model



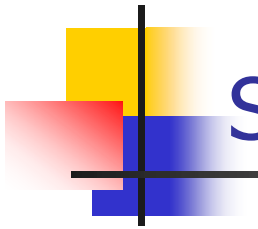
Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references



Modeling methodology

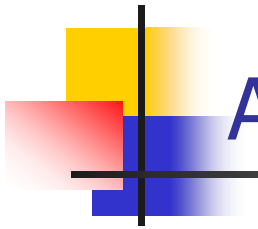
- Categories of TCP models:
  - averaged and discrete-time models
  - short-lived and long-lived TCP connections
- S-TCP/RED model:
  - discrete-time model with a long-lived connection
- State variables:
  - window size (TCP)
  - average queue size (RED)



S-TCP/RED model

- Key properties of the proposed S-TCP/RED model:
  - slow start, congestion avoidance, fast retransmit, and fast recovery (simplified)
  - Timeout:

    J. Padhye, V. Firoiu, and D. F. Towsley, "Modeling TCP Reno performance: a simple model and its empirical validation," *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 133–145, Apr. 2000.
  - Captures the basic RED algorithm

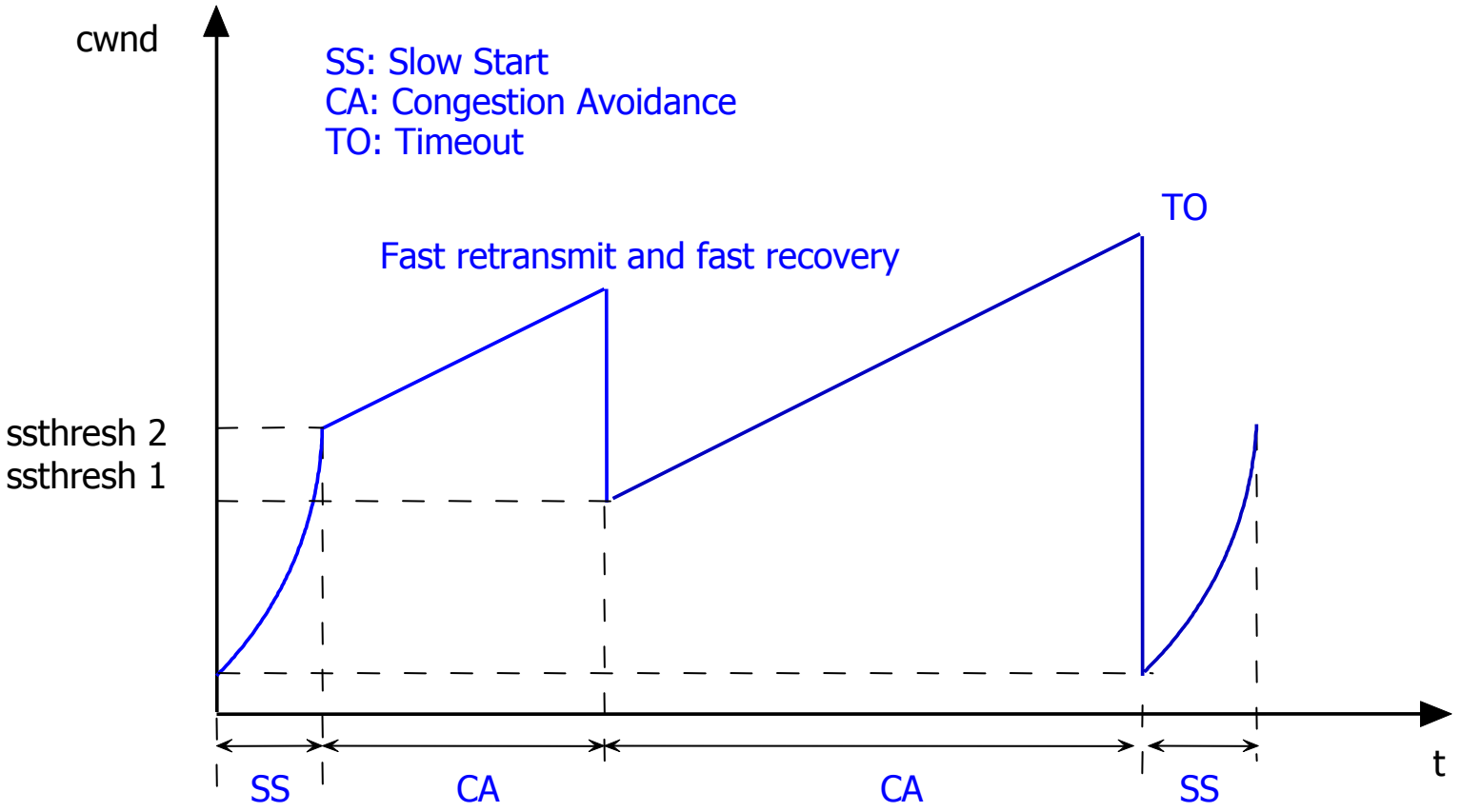


Assumptions

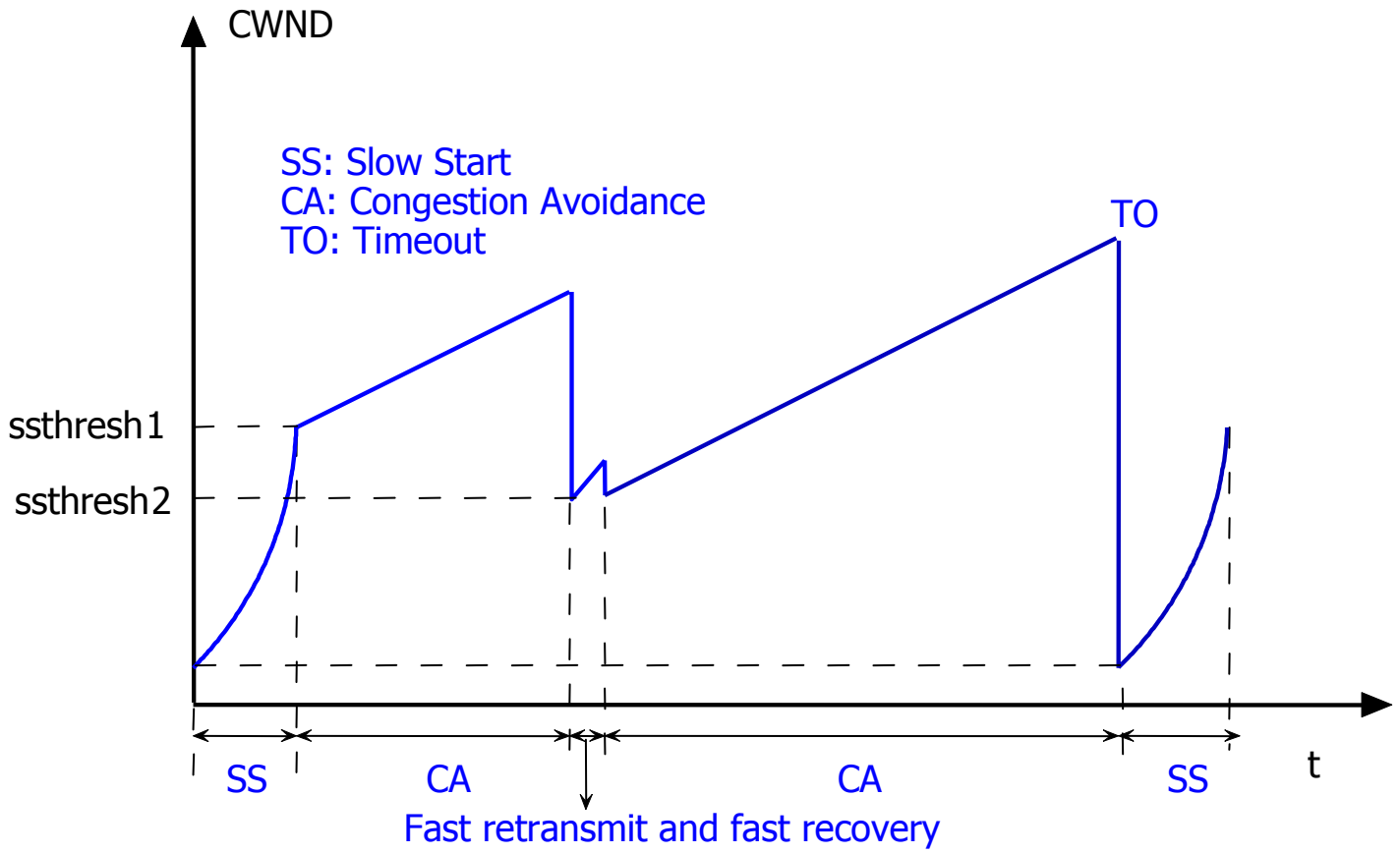
- Long-lived TCP connection
- Constant propagation delay between the source and the destination
- Constant packet size
- ACK packets are never lost
- Timeout occurs only due to packet loss
- The system is sampled at the end of every RTT interval

S-TCP/RED model simplifications

- Simplified fast recovery



TCP Reno: fast recovery



S-TCP/RED model simplifications

- TO = 5 RTT

  V. Firoiu and M. Borden, "A study of active queue management for congestion control," in *Proc. of IEEE INFOCOM 2000*, vol. 3, pp. 1435–1444, Tel-Aviv, Israel, Mar. 2000.

- RED: parameter count is not used

if $(q_{\min} < \bar{q} < q_{\max})$

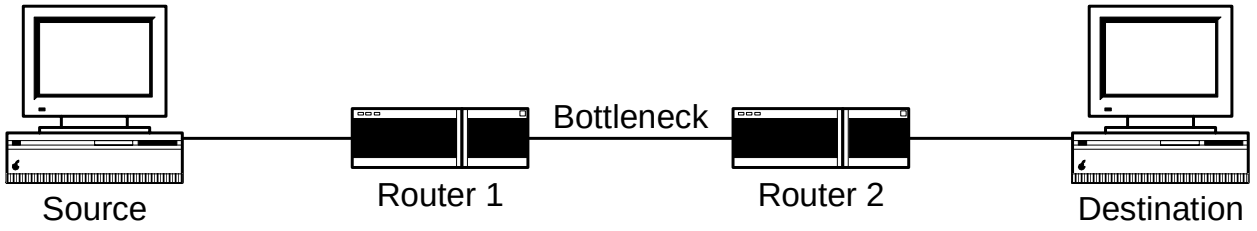
$$p_b = p_{\max} \times \frac{\bar{q} - q_{\min}}{q_{\max} - q_{\min}}$$

$$p_a = \frac{p_b}{1 - count \times p_b}$$

if $(q_{\min} < \bar{q} < q_{\max})$

$$p_a = p_b \longrightarrow p_a = p_{\max} \times \frac{\bar{q} - q_{\min}}{q_{\max} - q_{\min}}$$

Network topology



- Components: one source, two routers, and one destination
- The link between routers 1 and 2 is the only bottleneck
- RED algorithm is deployed in router 1

S-TCP/RED: parameters and variables

- Variables:

  w : window size $\overline{q}$: average queue size

  p : drop probability q : instantaneous queue size

- Parameters:

  q_{max}: maximum queue threshold

  q_{min}: minimum queue threshold

  p_{max}: maximum drop probability

  w_q : queue weight d : propagation delay

  M : packet size C : link capacity

S-TCP/RED: a discrete-time model for TCP Reno with RED

- Calculate the average queue size:

$$q_{k+1} = q_k + W_{k+1} - \frac{C}{M}(d + \frac{q_k \cdot M}{C})$$

$$= W_{k+1} - \frac{C \cdot d}{M} \qquad (1)$$

$$\bar{q}_{k+1} = (1 - w_q) \cdot \bar{q}_k + w_q \cdot q_{k+1} \qquad (2)$$

  - the average queue size is updated after each packet arrival
  - $\bar{q}_{k+1}$ is updated W_{k+1} times in *k*+1-th round

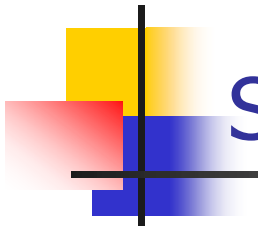
From (1) and (2):

$$\bar{q}_{k+1} = (1 - w_q)^{W_{k+1}} \cdot \bar{q}_k + (1 - (1 - w_q)^{W_{k+1}}) \cdot \max(W_{k+1} - \frac{C \cdot d}{M}, 0)$$

S-TCP/RED model: drop probability

- Calculate the drop probability:

$$p_{k+1} = \begin{cases} 0 & if\ \bar{q}_{k+1} \le q_{\min} \\ 1 & if\ \bar{q}_{k+1} \ge q_{\max} \\ \dfrac{\bar{q}_{k+1} - q_{\min}}{q_{\max} - q_{\min}} p_{\max} & otherwise \end{cases}$$



S-TCP/RED model: three cases

- No packet lost:
  - slow start
  - congestion avoidance
- Single packet lost:
  - fast retransmit
  - fast recovery
- At least two packets lost:
  - timeout

S-TCP/RED model: case 1

- No packet lost: $p_k \cdot W_k < 0.5$

$$W_{k+1} = \begin{cases} \min(\, 2W_k,\ ssthresh\ ) & if\ W_k < ssthresh \\ \min(\, W_k + 1,\ rwnd\ ) & if\ W_k \geq ssthresh \end{cases}$$

$$\overline{q}_{k+1} = (1 - w_q)^{W_{k+1}} \cdot \overline{q}_k + (1 - (1 - w_q)^{W_{k+1}}) \cdot \max(W_{k+1} - \frac{C \cdot d}{M}, 0)$$

ssthresh: slow start threshold

rwnd: receiver's advertised window size

S-TCP/RED model: cases 2 and 3

- One packet lost: $0.5 \le p_k \cdot W_k < 1.5$

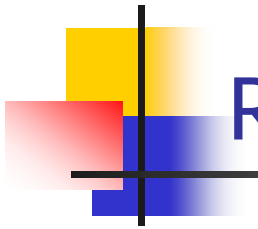
$$W_{k+1} = \frac{1}{2} W_k$$

$$\bar{q}_{k+1} = (1 - w_q)^{W_{k+1}} \cdot \bar{q}_k + (1 - (1 - w_q)^{W_{k+1}}) \cdot \max(W_{k+1} - \frac{C \cdot d}{M}, 0)$$

- At least two packets lost: $p_k \cdot W_k \ge 1.5$

$$W_{k+1} = 0$$

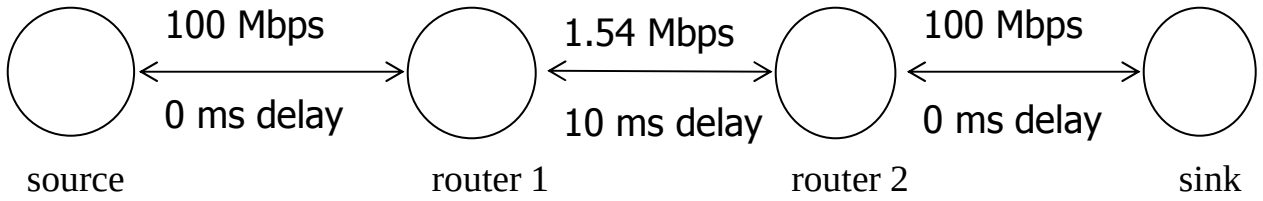
$$\bar{q}_{k+1} = \bar{q}_k$$



Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references

Simulation scenario



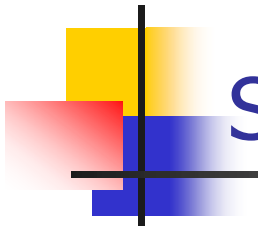
- source to router1:
  - link capacity: 100 Mbps with 0 ms delay
- router 1 to router 2: the only bottleneck in the network
  - link capacity: 1.54 Mbps with 10 ms delay
- router 2 to sink:
  - link capacity: 100 Mbps with 0 ms delay

RED: default parameters

- RED parameters:

  S. Floyd, "RED: Discussions of Setting Parameters," Nov. 1997: http://www.icir.org/floyd/REDparameters.txt

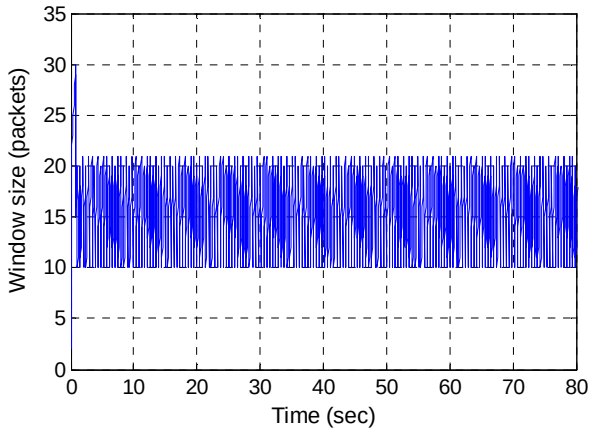
Queue weight (w_q)	0.002
Maximum drop probability (p_{max})	0.1
Minimum queue threshold (q_{min})	5 (packets)
Maximum queue threshold (q_{max})	15 (packets)

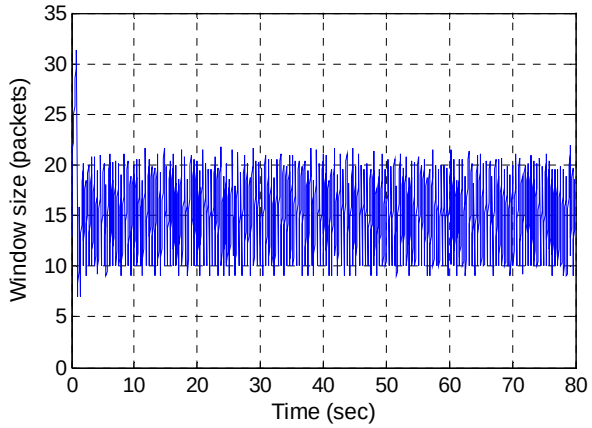


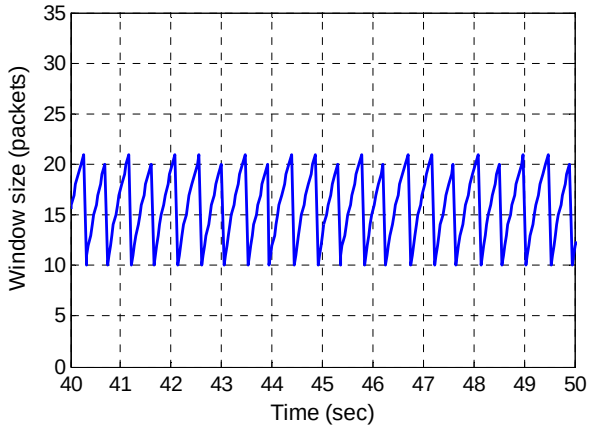
S-TCP/RED model validation

- Waveforms of the state variables with default parameters:
  - window size
  - average queue size
- Validation for various values of the system parameters:
  - queue weight: w_q
  - maximum drop probability: p_{max}
  - queue thresholds: q_{min} and q_{max} , $q_{max}/q_{min} = 3$

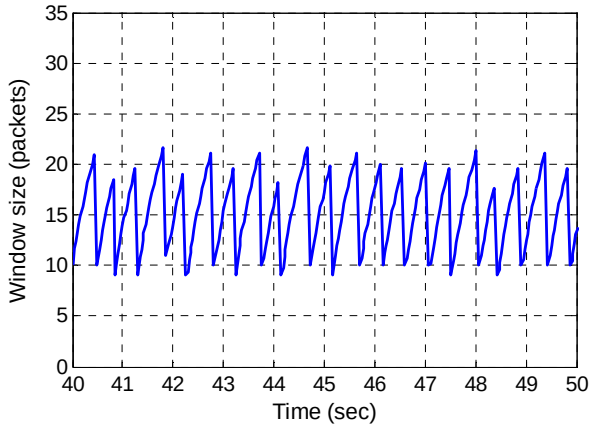
Window size: waveforms





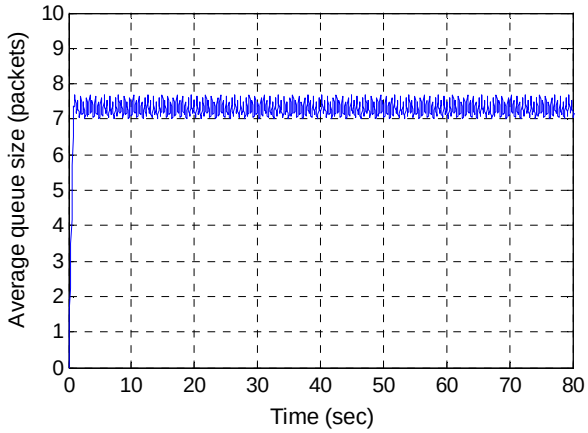


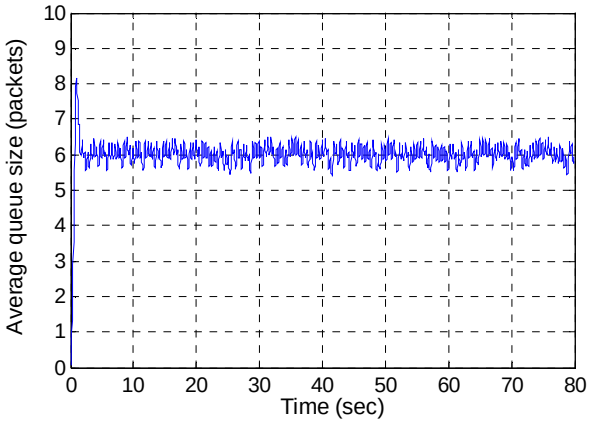
S-TCP/RED model

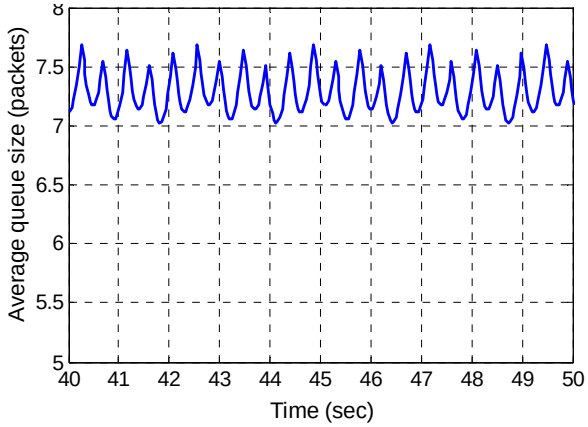


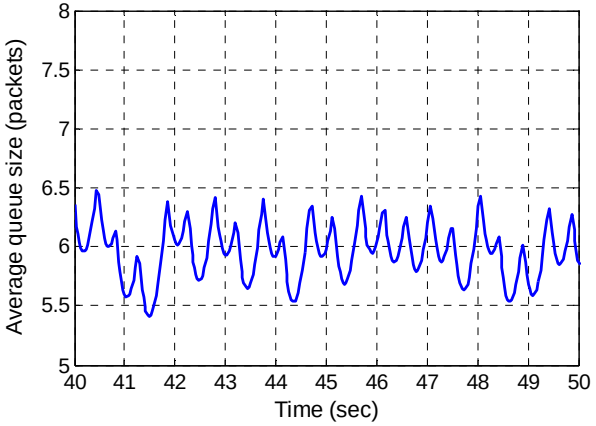
ns-2

Average queue size: waveforms







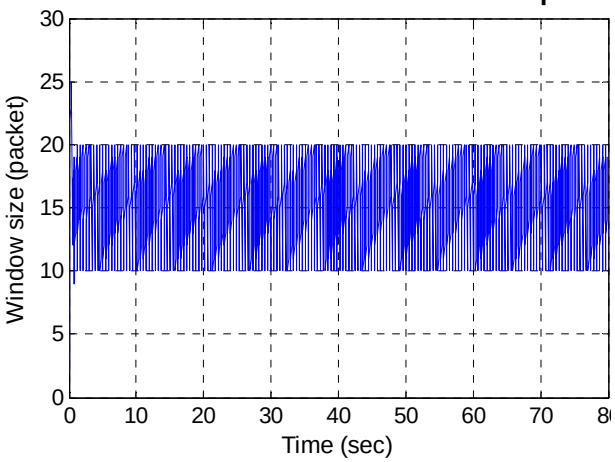


S-TCP/RED model

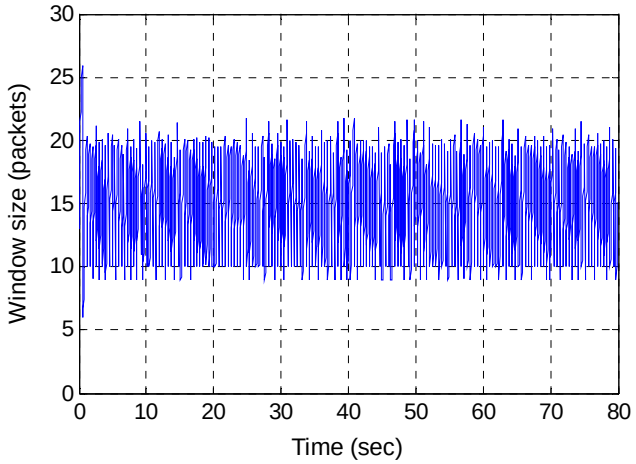
ns-2

Model validation: w_q

- w_q =[0.001, 0.01], with other parameters default
  - window size: w_q = 0.006



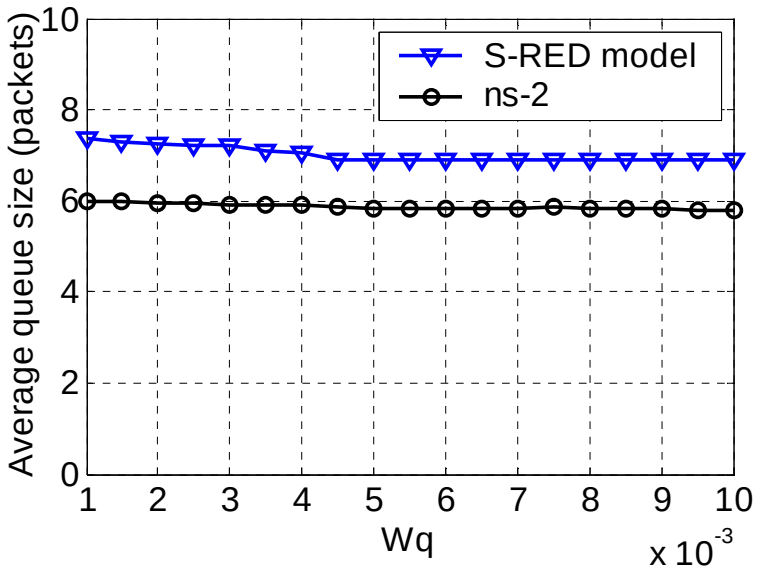
S-TCP/RED model



ns-2

Model validation: w_q

- average queue size during steady state:



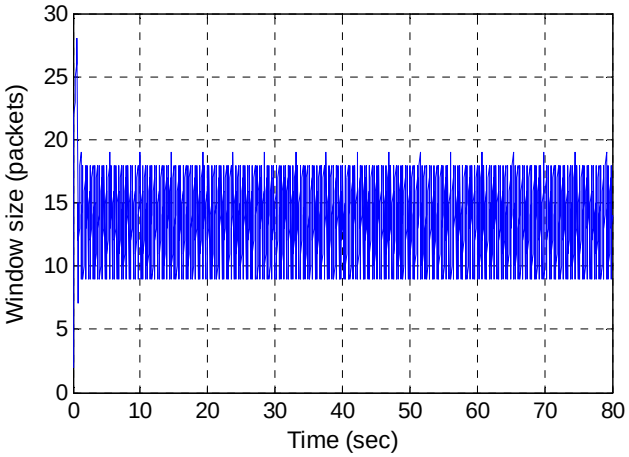
Model validation: w_q

- Comparison of system variables:

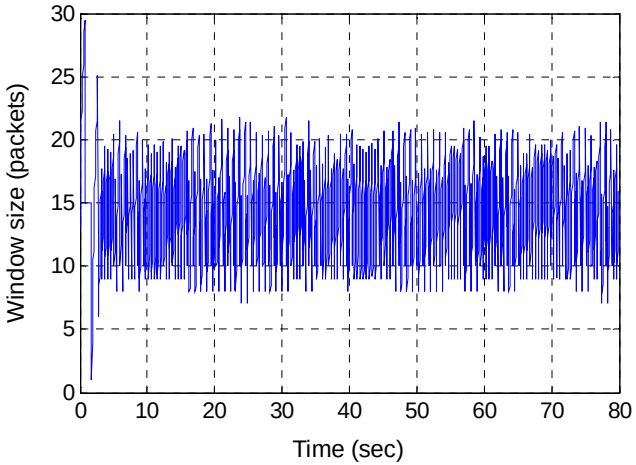
Parameters	Average RTT (msec)			Sending rate (packets/sec)			Drop rate (%)		
weight (w_q)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)
0.001	40.3	36.1	11.63	384.99	384.71	0.073	0.55	0.54	1.29
0.002	39.9	36.0	10.83	384.98	384.77	0.056	0.56	0.55	2.56
0.004	39.4	36.2	8.80	385.11	384.79	0.083	0.59	0.56	6.12
0.006	39.0	35.8	8.93	385.08	384.73	0.093	0.60	0.56	7.91
0.008	39.0	35.8	8.90	385.10	384.68	0.109	0.61	0.55	11.11
0.010	38.9	35.7	8.96	385.02	384.70	0.083	0.61	0.55	11.72

Model validation: p_{max}

- p_{max} = [0.05, 0.95], with other parameters default
  - window size: waveforms, p_{max} = 0.5



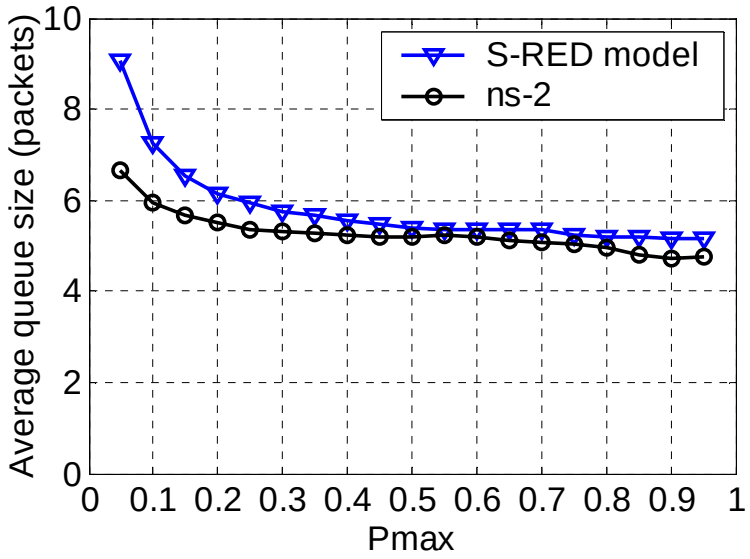
S-TCP/RED



ns-2

Model validation: p_{max}

- average queue size during steady state:



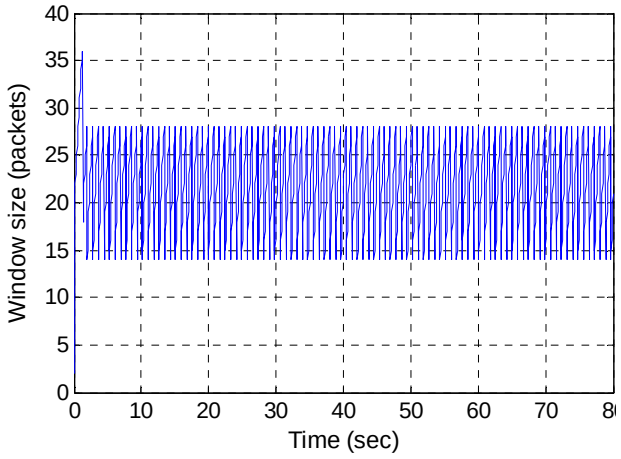
Model validation: p_{max}

- Comparison of system variables:

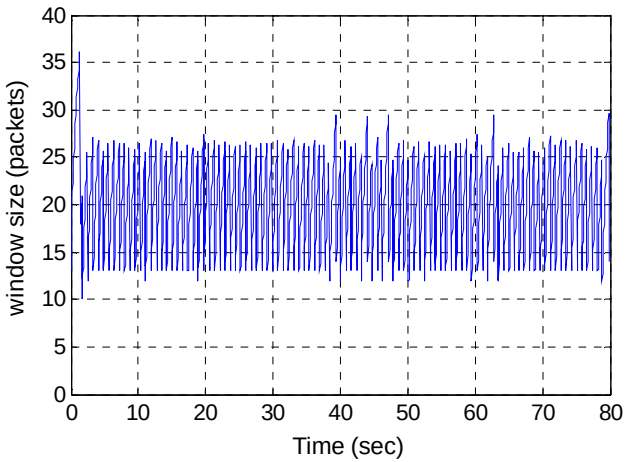
	Average RTT (msec)			Sending rate (packets/sec)			Drop rate (%)		
p_{max}	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)
0.05	44.3	38.1	16.27	385.13	384.70	0.11	0.45	0.51	-11.76
0.10	39.9	36.0	10.83	384.98	384.77	0.06	0.56	0.55	2.56
0.25	36.5	34.5	5.80	384.93	384.73	0.05	0.65	0.59	11.28
0.50	35.3	34.0	3.80	384.98	379.37	1.48	0.73	0.61	19.09
0.75	34.8	35.1	-0.85	384.63	357.55	7.60	0.74	0.65	14.37

Model validation: q_{min} and q_{max}

- q_{min}= [1, 20] packets, q_{max}/q_{min} = 3, with other parameters default
  - window size: waveforms, q_{min} = 10 packets



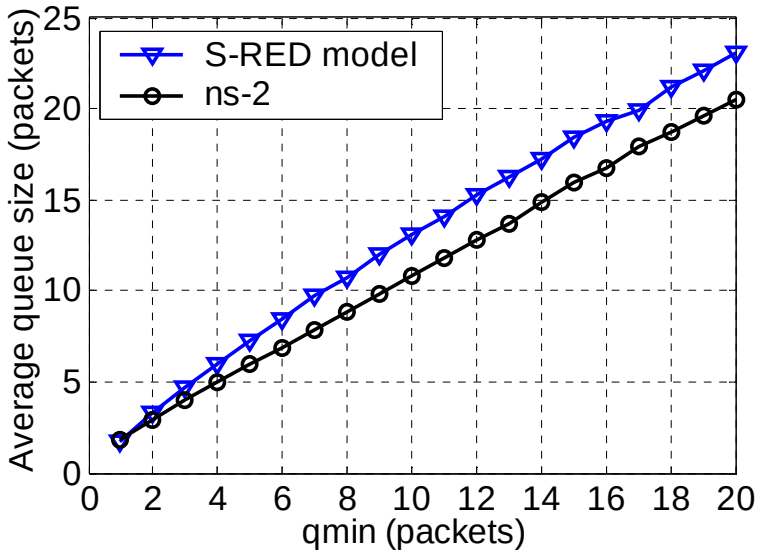
S-TCP/RED



ns-2

Model validation: q_{min} and q_{max}

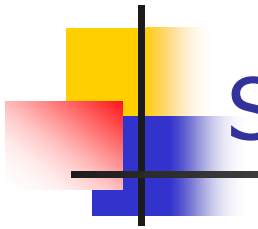
- average queue size during steady state:



Model validation: q_{min} and q_{max}

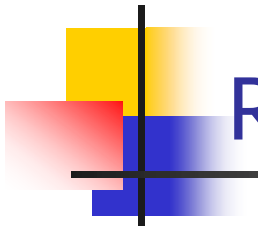
- Comparison of system variables:

	Average RTT (msec)			Sending rate (packets/sec)			Drop rate (%)		
q_{min} (packets)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)
3	33.4	31.1	7.4	383.22	382.44	0.20	0.78	0.71	10.01
5	39.9	36.0	10.83	384.98	384.77	0.06	0.56	0.55	2.56
10	54.7	48.1	13.72	385.10	384.85	0.06	0.31	0.33	-6.34
15	67.7	60.3	12.27	385.06	384.83	0.06	0.20	0.22	-10.71
20	79.1	73.0	8.36	385.30	384.95	0.09	0.15	0.16	-5.66



S-TCP/RED: model evaluation

- Waveforms of the window size:
  - match the ns-2 simulation results
- The average queue size:
  - mismatch, but similar trend
- System variables RTT, sending rate, and drop rate:
  - reasonable agreement with ns-2 simulation results, depending on the system parameters



Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references

S-TCP/RED: modification

- The difference in the average queue size between S-TCP/RED model and ns-2 is due to the simplification of p:

if $(q_{\min} < \bar{q} < q_{\max})$ if $(q_{\min} < \bar{q} < q_{\max})$

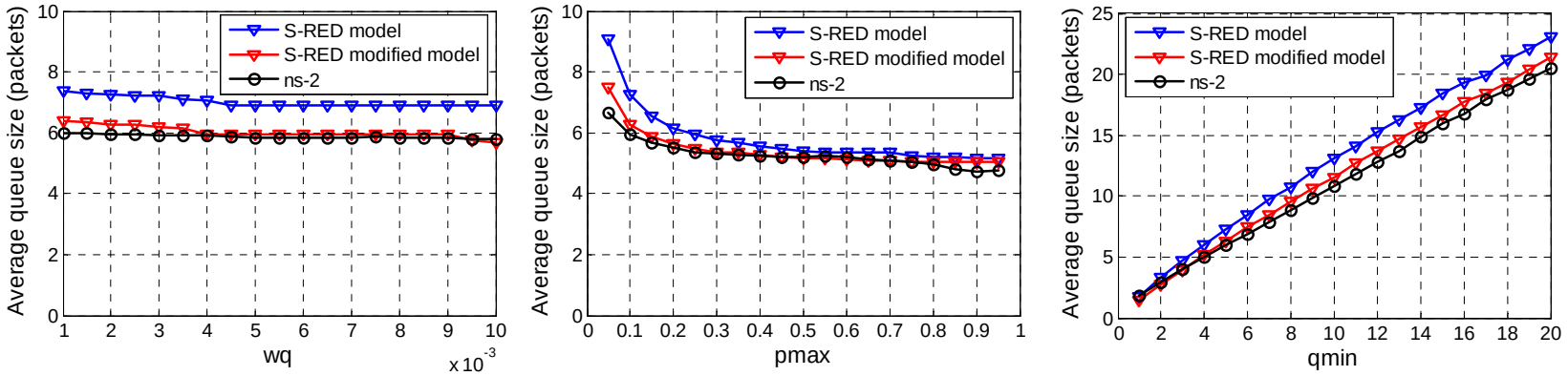
$$p_b = p_{\max} \times \frac{\bar{q} - q_{\min}}{q_{\max} - q_{\min}} \xrightarrow{p_a = p_b} p_a = p_{\max} \times \frac{\bar{q} - q_{\min}}{q_{\max} - q_{\min}}$$

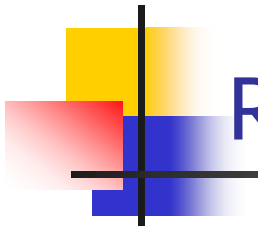
$$p_a = \frac{p_b}{1 - count \times p_b}$$

- Modification of p_a: $p_a = \alpha \cdot p_b \quad (\alpha > 1)$

Modified S-TCP/RED model

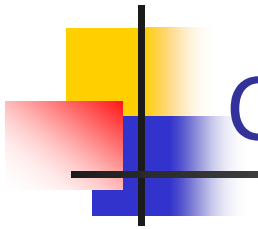
- Modification: $\alpha = 1.8$





Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references



Comparison: S-TCP/RED vs. M-model

- M-model:

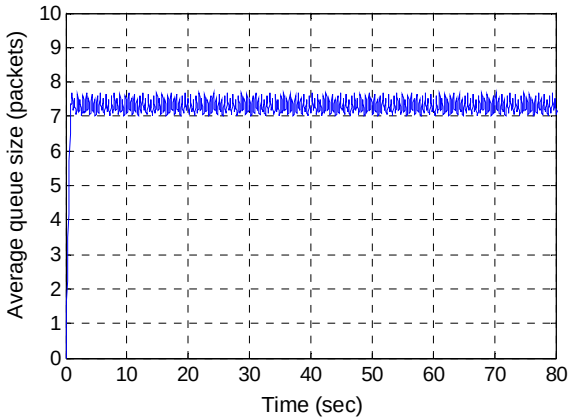
  A discrete nonlinear dynamical model of TCP Reno with RED proposed by a research group from University of Maryland:

  P. Ranjan, E. H. Abed, and R. J. La, "Nonlinear instabilities in TCP-RED," in *Proc. IEEE INFOCOM 2002*, New York, NY, USA, June 2002, vol. 1, pp. 249–258 and *IEEE/ACM Trans. on Networking*, vol. 12, no. 6, pp. 1079–1092, Dec. 2004.

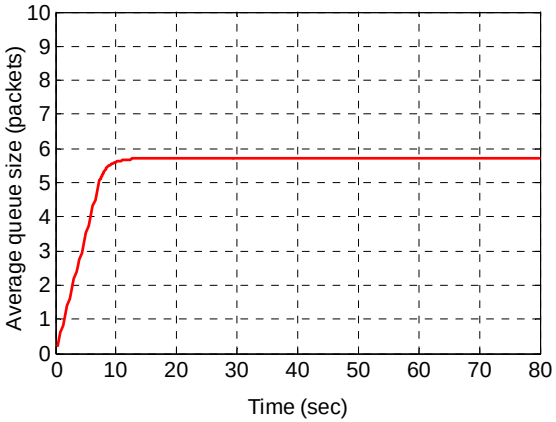
- One state variable: average queue size

Model comparison: default parameters

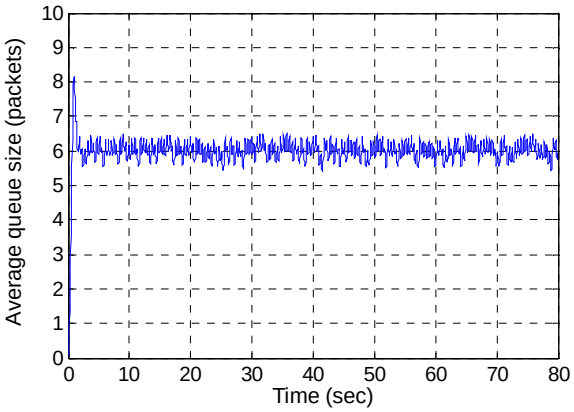
- The waveform of the average queue size with default RED parameters:



S-TCP/RED



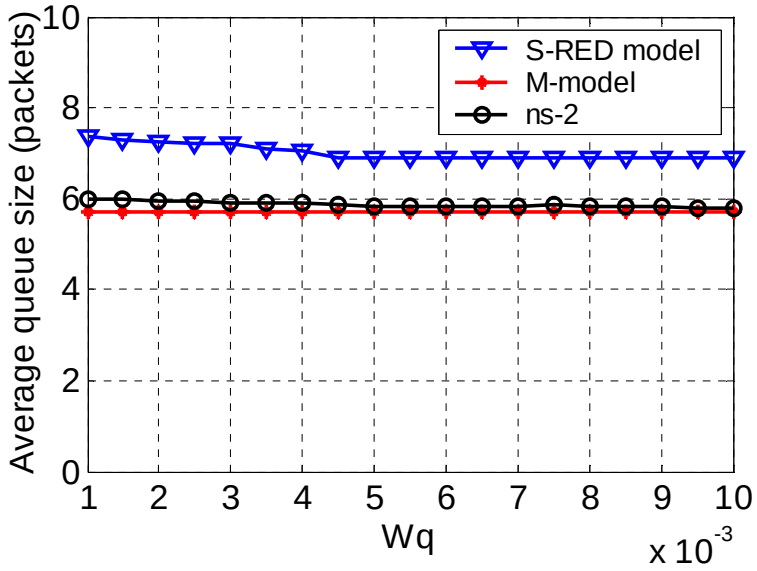
M-model



ns-2

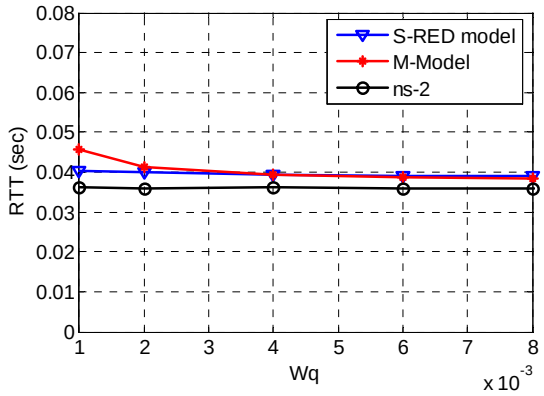
Model comparisons: w_q

- w_q = [0.001, 0.01], with other parameters default
  - average queue size during steady state:

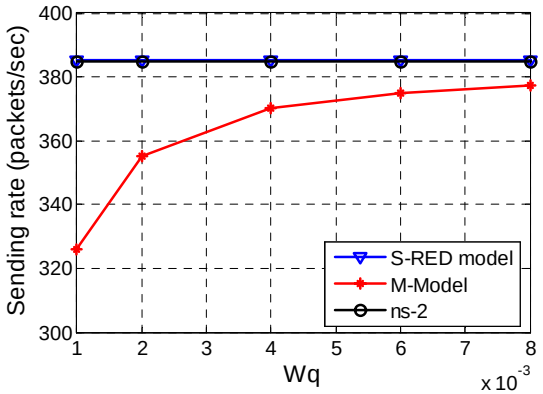


Model comparisons: w_q

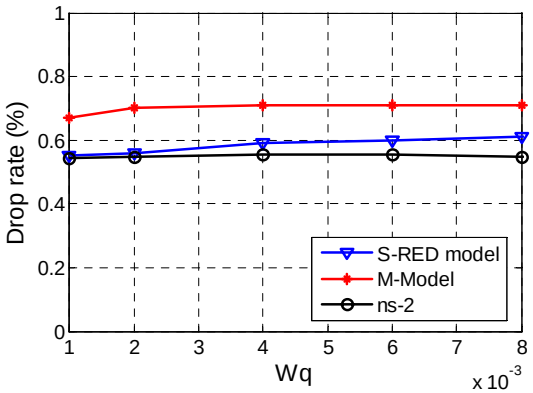
- system variables:



RTT



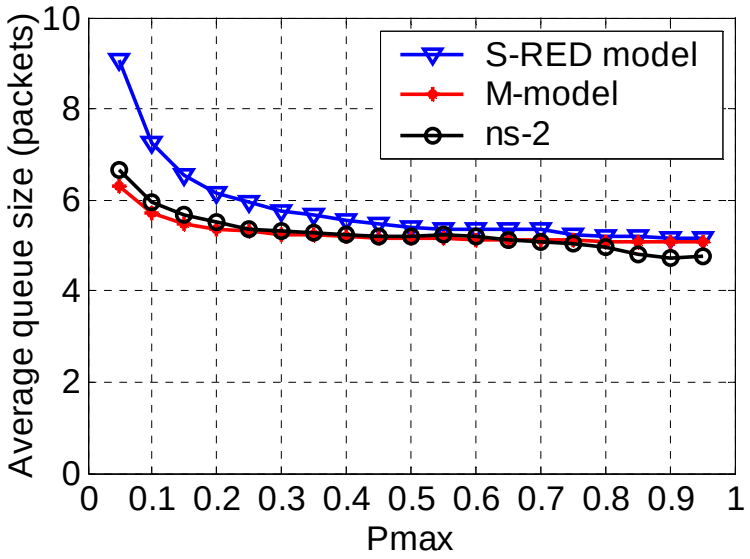
sending rate



drop rate

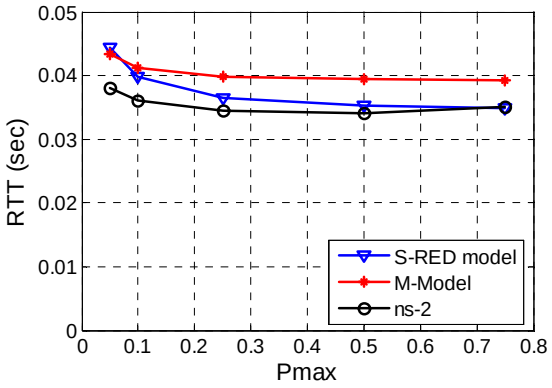
Model comparisons: p_{max}

- p_{max}= [0.05, 0.95], with other parameters default
  - average queue size during steady state:

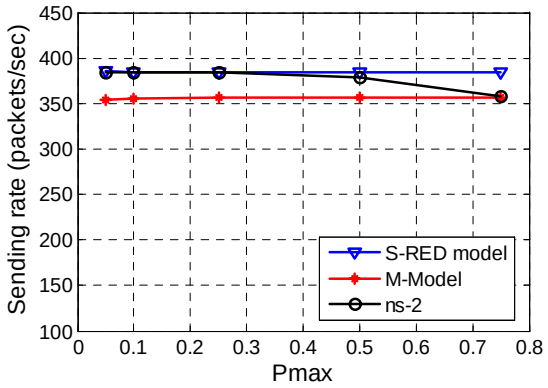


Model comparisons: p_{max}

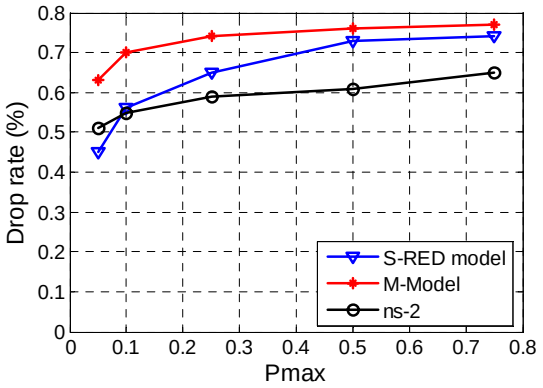
- system variables:



RTT



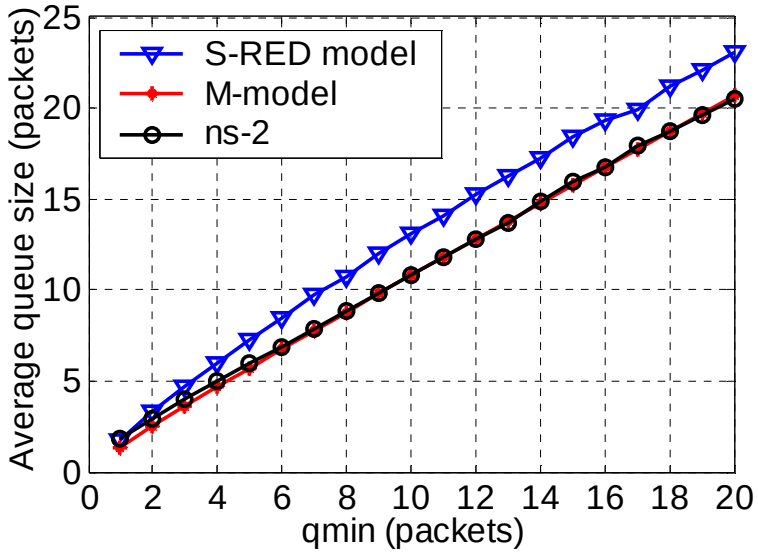
sending rate



drop rate

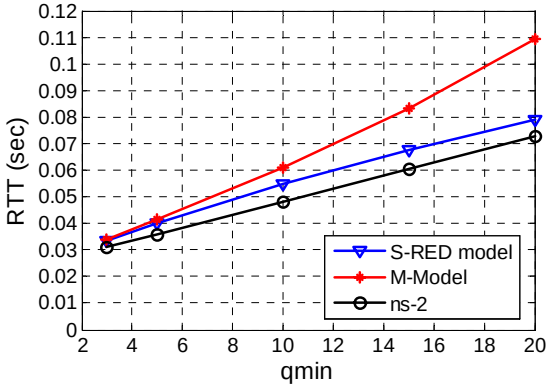
Model comparisons: q_{min} and q_{max}

- q_{min}= [1, 20] packets, q_{max}/q_{min} =3, with other parameters default
  - average queue size during steady state:

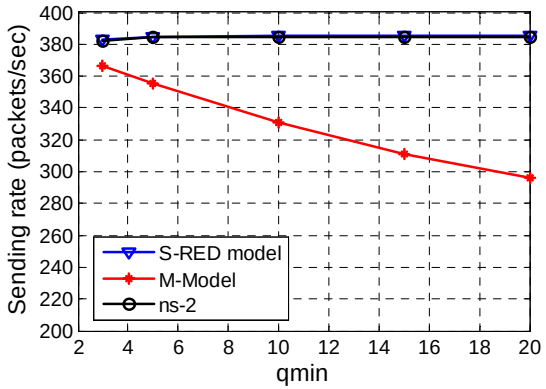


Model comparisons: q_{min} and q_{max}

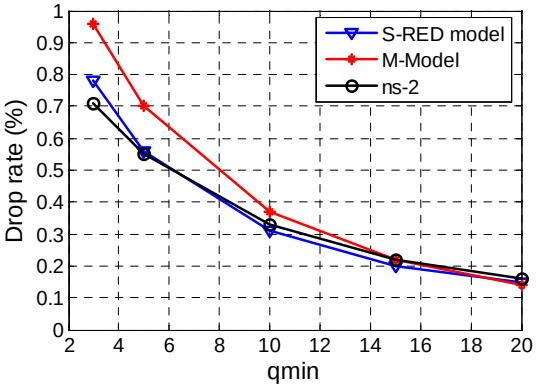
- system variables:



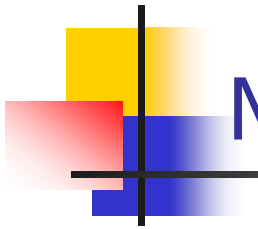
RTT



sending rate

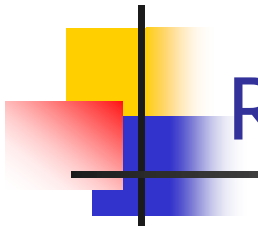


drop rate



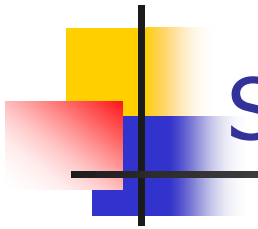
Model comparison: summary

- S-TCP/RED model captures dynamical details of TCP/RED
- RTT, sending rate, and drop rate: S-TCP/RED model, in general, matches the ns-2 simulation results better than the M-model
- M-model: average queue size
  - constant during steady-state
  - matches better the ns-2 simulation results



Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references



Simplified S-TCP/RED model

- M-model, a discrete nonlinear dynamical model of TCP Reno with RED:

  P. Ranjan, E. H. Abed, and R. J. La, "Nonlinear instabilities in TCP-RED," in *Proc. IEEE INFOCOM 2002*, New York, NY, USA, June 2002, vol. 1, pp. 249–258 and *IEEE/ACM Trans. on Networking*, vol. 12, no. 6, pp. 1079–1092, Dec. 2004.

- The proposed simplified S-TCP/RED model is:
  - simple and intuitively derived
  - able to capture detailed dynamical behavior of TCP/RED systems
  - has been verified via ns-2 simulations

Simplified S-TCP/RED model: one state variable

- Variables:
  - $\overline{q_{k+1}}$: average queue size in round k+1
  - $\overline{q_k}$: average queue size in round k
  - w_q: queue weight in RED
  - N: number of TCP connections
  - K: constant = $\sqrt{3/2}$
  - p_k: drop probability in round k
  - C: capacity of the link between the two routers
  - d: round-trip propagation delay
  - M: packet size
  - *rwnd*: receiver's advertised window size

Simplified S-TCP/RED model: case 1

- Drop probability: $p_k \neq 0$

$$q_{k+1} = q_k + B(p_k) \cdot RTT_{k+1} \cdot N - \frac{C \cdot RTT_{k+1}}{M}$$

$$= q_k + \frac{K}{\sqrt{p_k} \cdot RTT_{k+1}} \cdot RTT_{k+1} \cdot N - \frac{C}{M}(d + \frac{q_k \cdot M}{C})$$

$$= \frac{K \cdot N}{\sqrt{p_k}} - \frac{C \cdot d}{M}$$

where:

$B(p_k)$	: TCP sending rate
$B(p_k) \cdot RTT_{k+1} \cdot N$	: the number of incoming packets
$C \cdot \frac{RTT_{k+1}}{M}$	: the number of outgoing packets

Simplified S-TCP/RED model: case 1

The average queue size is:

$$\bar{q}_{k+1} = (1 - w_q) \cdot \bar{q}_k + w_q \cdot q_{k+1}$$

hence

$$\bar{q}_{k+1} = (1 - w_q) \cdot \bar{q}_k + w_q \cdot \max\left(\frac{N \cdot K}{\sqrt{p_k}} - \frac{C \cdot d}{M}, 0\right)$$

Simplified S-TCP/RED model: case 2

- Drop probability: $p_k = 0$

$$q_{k+1} = q_k + B(p_k) \cdot RTT_{k+1} \cdot N - \frac{C \cdot RTT_{k+1}}{M}$$

$$= q_k + \frac{rwnd}{RTT_{k+1}} \cdot RTT_{k+1} \cdot N - \frac{C}{M}(d + \frac{q_k \cdot M}{C})$$

$$= rwnd \cdot N - \frac{C \cdot d}{M}$$

The average queue size is:

$$\bar{q}_{k+1} = (1 - w_q) \cdot \bar{q}_k + w_q \cdot q_{k+1}$$

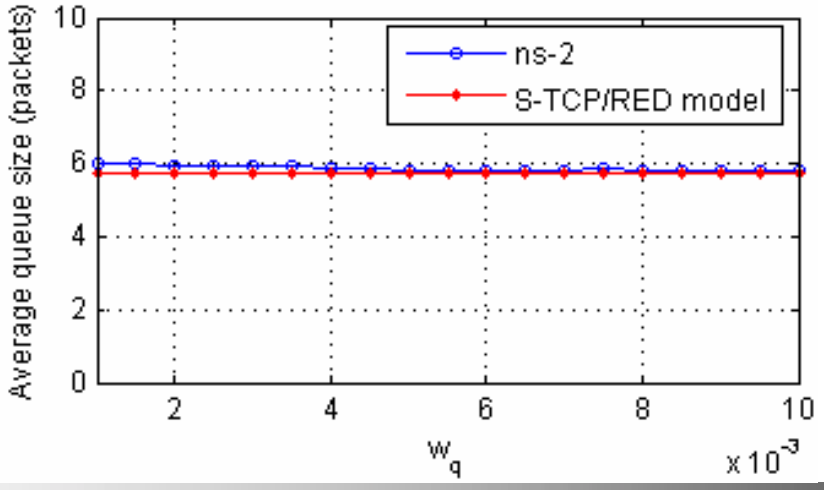
hence

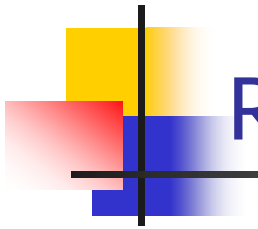
$$\bar{q}_{k+1} = (1 - w_q) \cdot \bar{q}_k + w_q \cdot (rwnd \cdot N - \frac{C \cdot d}{M})$$

Simplified S-TCP/RED model

- Dynamical model of TCP/RED:

$$\bar{q}_{k+1} = \begin{cases} (1-w_q)\cdot \bar{q}_k + w_q \cdot \max\left(\frac{N\cdot K}{\sqrt{p_k}} - \frac{C\cdot d}{M}, 0\right) \ if \ \ p_k \neq 0 \\ (1-w_q)\cdot \bar{q}_k + w_q \cdot (rwnd \cdot N - \frac{C\cdot d}{M}) \ \ if \ p_k = 0 \end{cases}$$



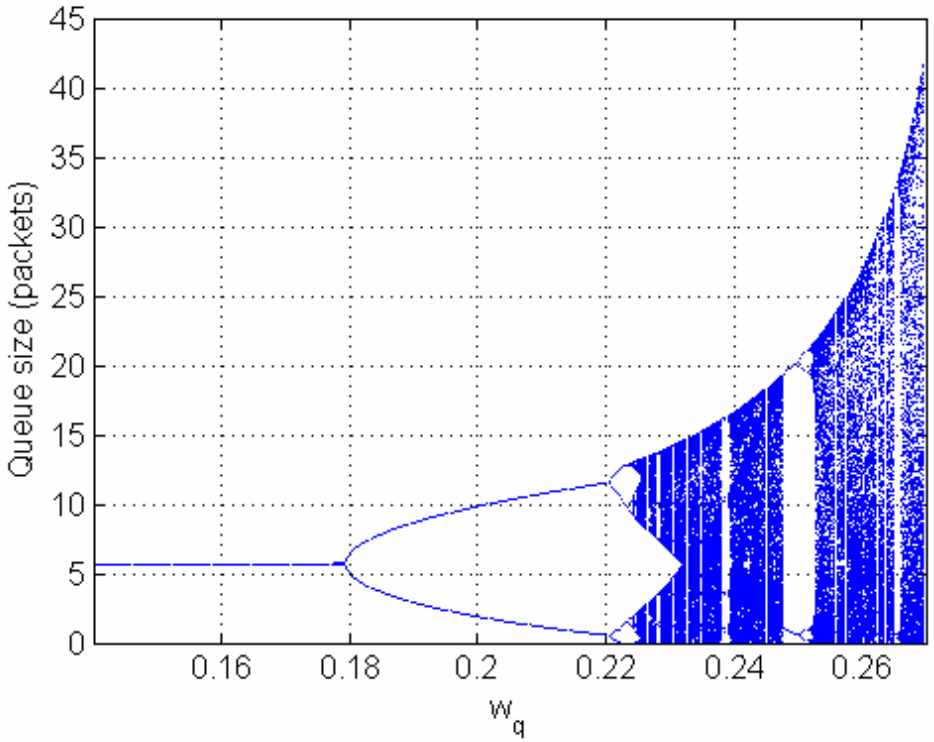


Roadmap

- Introduction
- Discrete-time model of TCP Reno with RED:
  - S-TCP/RED: model with two state variables
  - model validation
  - modifications
- Comparison of TCP/RED models
- Simplified S-TCP/RED: model with one state variable
- Bifurcation diagrams
- Conclusions and references

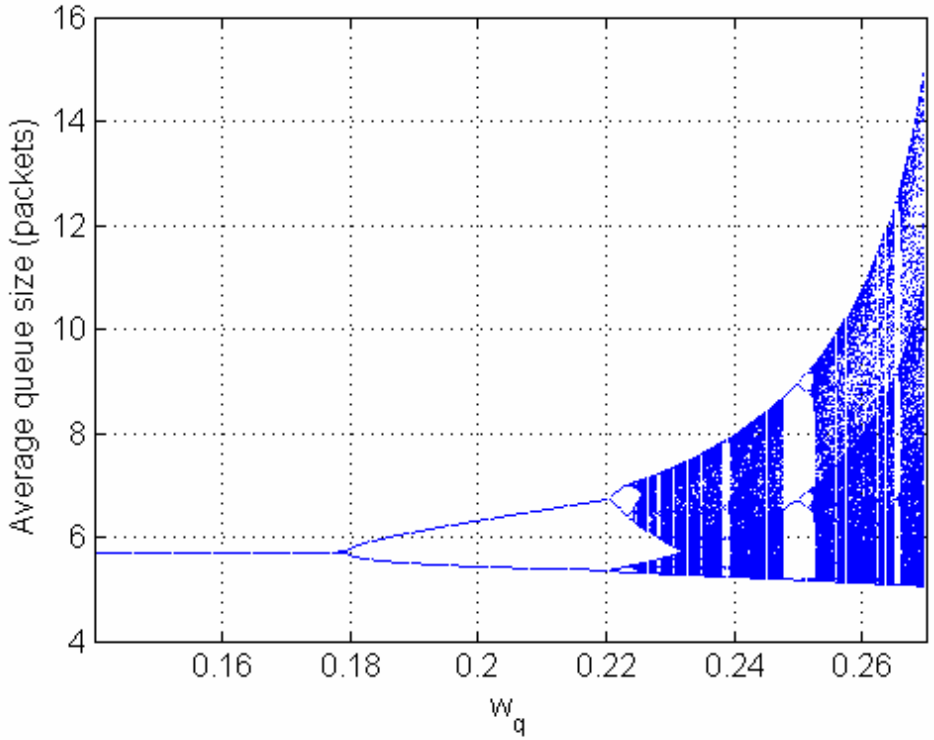
Queue size vs. w_q

- $p_{max} = 0.1$, $q_{min} = 5$, $q_{max} = 15$



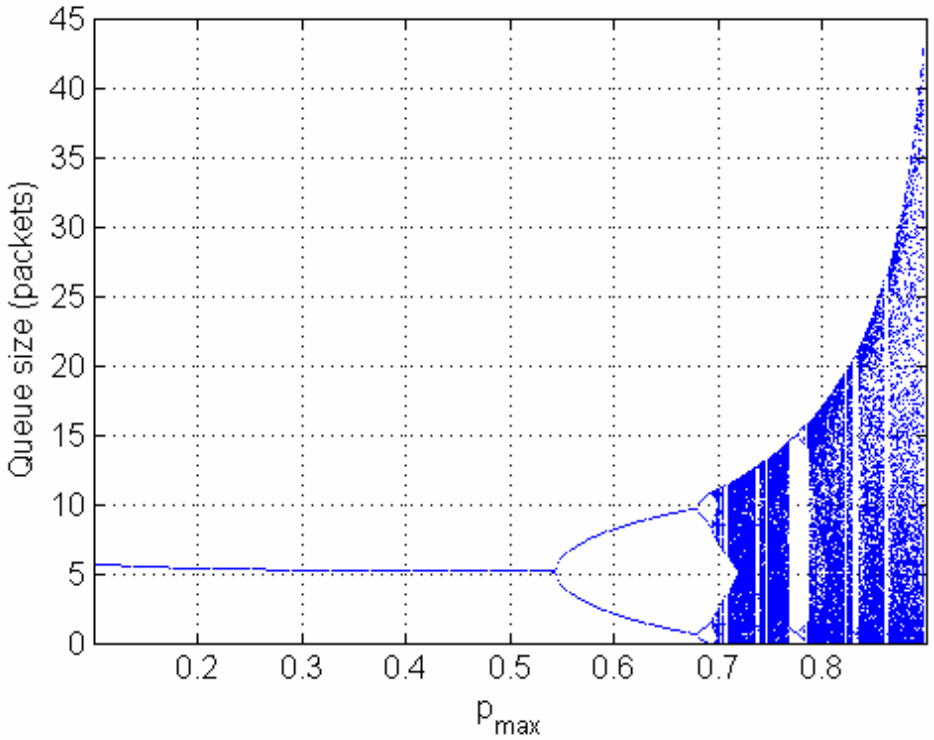
Average queue size vs. w_q

- $p_{max} = 0.1$, $q_{min} = 5$, $q_{max} = 15$



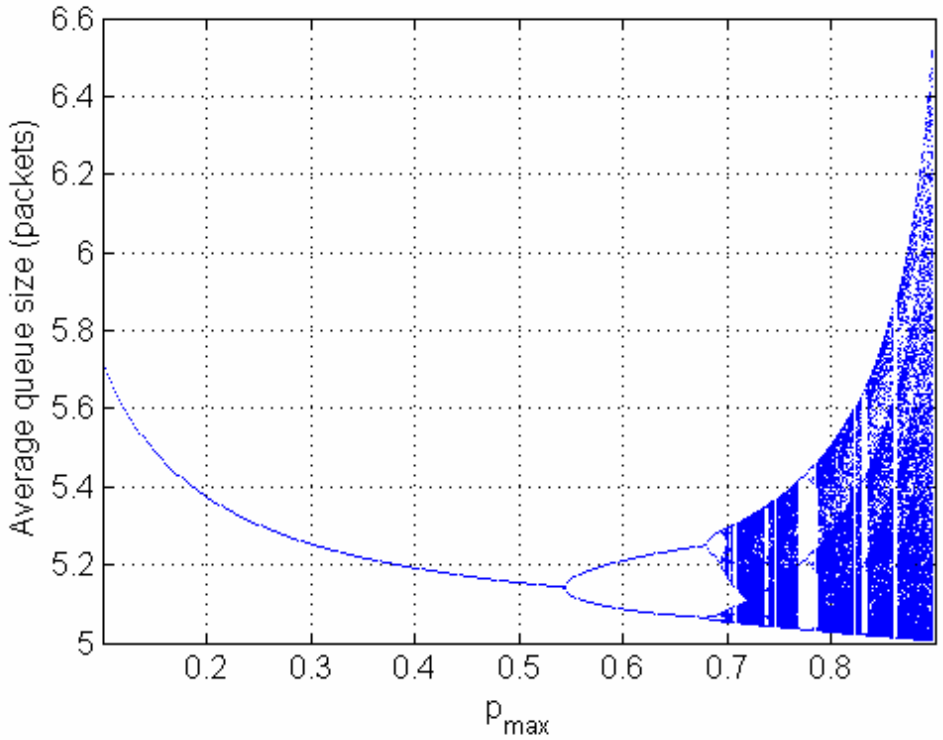
Queue size vs. p_{max}

- w_{qx} = 0.04, q_{min} = 5, q_{max} = 15



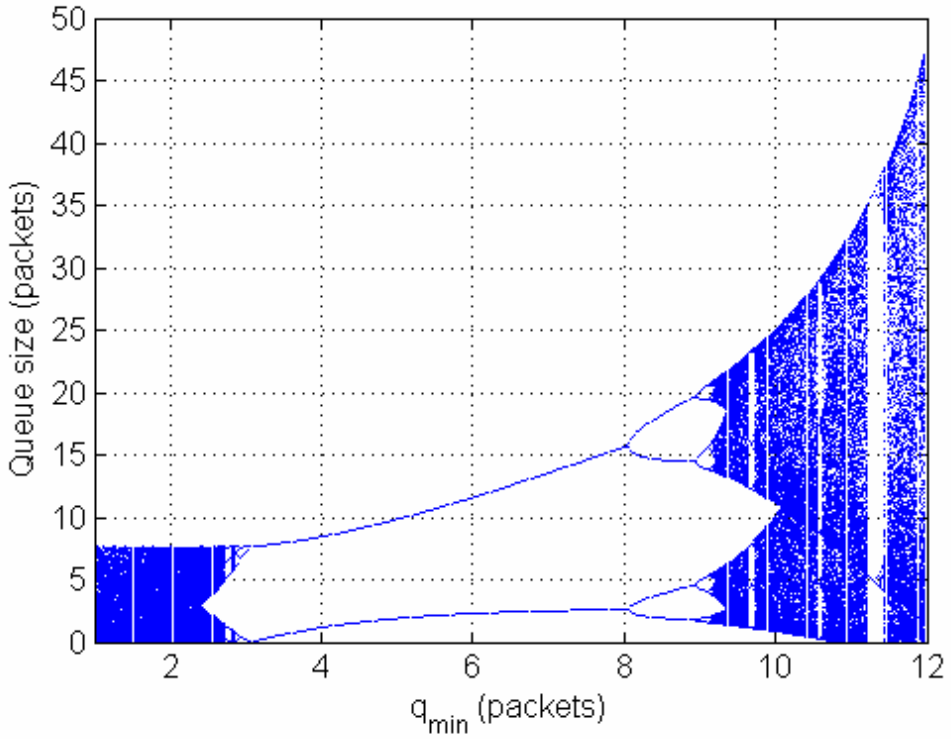
Average queue size vs. p_{max}

- $w_q = 0.04$, $q_{min} = 5$, $q_{max} = 15$



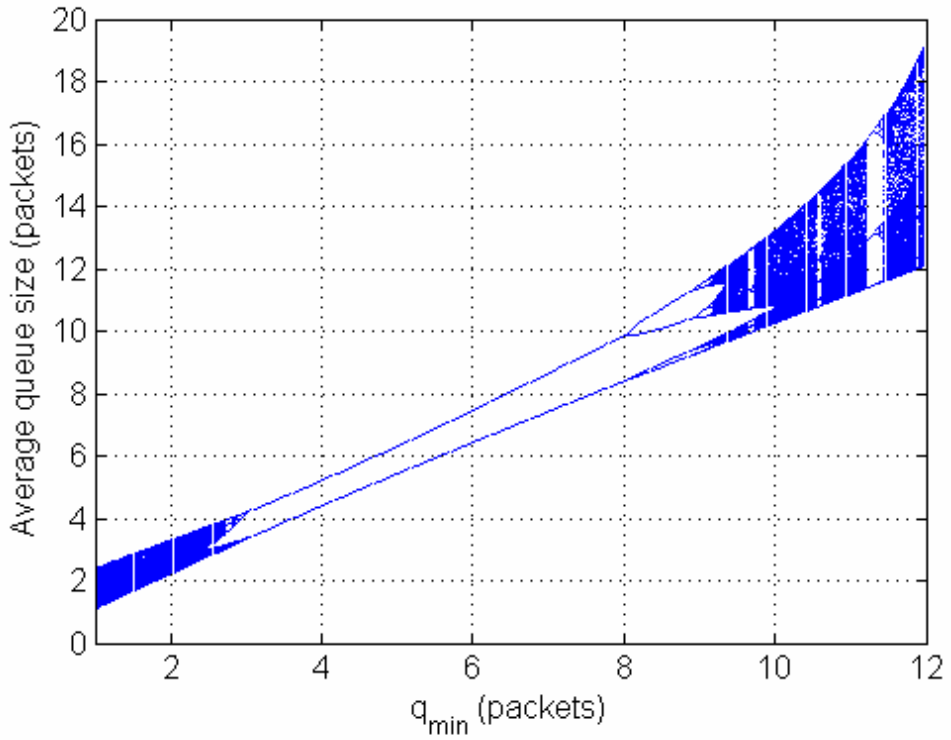
Queue size vs. q_{min}/q_{max}

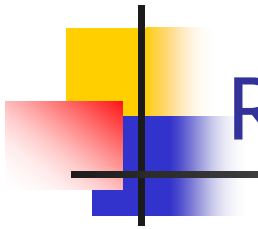
- $w_q = 0.2$, $p_{max} = 0.1$, $q_{max} = 3 \times q_{min}$



Average queue size vs. q_{min}/q_{max}

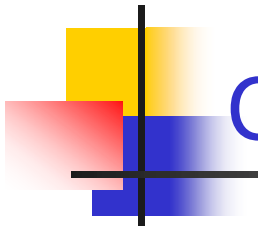
- $w_q = 0.2$, $p_{max} = 0.1$, $q_{max} = 3 \times q_{min}$





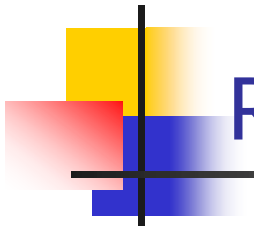
Roadmap

- Introduction
- Discrete-time models of TCP Reno with RED:
  - S-TCP/RED: with two state variables
  - simplified S-TCP/RED: one state variable
- Model validation
- Comparison of TCP/RED models
- S-TCP/RED: modifications
- Conclusions and references



Conclusions

- We developed two discrete-time models for TCP Reno with RED
- S-TCP/RED models include:
  - slow start, congestion avoidance, fast retransmit, timeout, elements of fast recovery, and RED
- Proposed models were validated by comparing their performance to ns-2 simulations
- They capture the main features of the dynamical behavior of TCP Reno with RED
- S-TCP/RED model with one state variable was used to study bifurcation and chaos in TPC/RED systems with a single connection



References: TCP and RED

[1] M. Allman, V. Paxson, and W. Steven, "TCP congestion control," *IETF Request for Comments (RFC) 2581*, Apr. 1999.

[2] B. Barden et al., "Recommendations on queue management and congestion avoidance in the Internet," *IETF Request for Comments (RFC) 2309*, Apr. 1998.

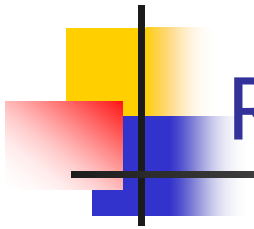
[3] R. Braden, "Requirements for Internet hosts: communication layers," *IETF Request for Comments (RFC) 1122*, Oct. 1989.

[4] C. Casetti, M. Gerla, S. Lee, S. Mascolo, and M. Sanadidi, "TCP with faster recovery," in *Proc. MILCOM 2000*, Los Angeles, CA, USA, Oct. 2000, vol. 1, pp. 320–324.

[5] K. Fall and S. Floyd, "Simulation-based comparison of Tahoe, Reno, and SACK TCP," *ACM Communication Review*, vol. 26, no. 3, pp. 5–21, July 1996.

[6] V. Jacobson, "Congestion avoidance and control," *ACM Computer Communication Review*, vol. 18, no. 4, pp. 314–329, Aug. 1988.

[7] V. Jacobson, "Modified TCP congestion avoidance algorithm," ftp://ftp.isi.edu/end2end/end2end-interest-1990.mail, Apr. 1990.



References: TCP and RED

[8] J. Padhye and S. Floyd, "On inferring TCP behavior," in *Proc. ACM SIGCOMM 2001*, San Diego, CA, USA, Aug. 2001, pp. 287–298.

[9] V. Paxson and M. Allman, "Computing TCP's retransmission timer," *IETF Request for Comments (RFC) 2988*, Nov. 2000.

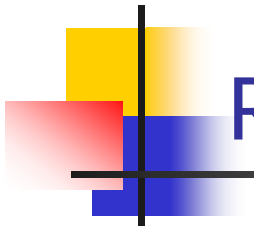
[10] J. Postel, "Transmission control protocol," *IETF Request for Comments (RFC) 793*, Sept. 1981.

[11] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The protocols*. New York, NY: Addison-Wesley, 1994.

[12] S. Floyd, "RED: discussions of setting parameters," Nov. 1997: http://www.icir.org/floyd/REDparameters.txt.

[13] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[14] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: an algorithm for increasing the robustness of RED's active queue management," Aug. 2001: http://www.icir.org/floyd/papers/.



References: TCP/RED models

[15] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *Proc. IEEE INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000, vol. 3, pp. 1742–1751.

[16] V. Firoiu and M. Borden, "A study of active queue management for congestion control," *in Proc. IEEE INFOCOM 2000*, Tel-Aviv, Israel, Mar. 2000, vol. 3, pp. 1435–1444.

[17] J. Padhye, V. Firoiu, and D. F. Towsley, "Modeling TCP Reno performance: a simple model and its empirical validation," *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 133–145, Apr. 2000.

[18] C. V. Hollot, V. Misra, D. Towsley, and W. B. Gong, "A control theoretic analysis of RED," in *Proc. IEEE INFOCOM 2001*, Anchorage, AK, Apr. 2001, vol. 3, pp. 1510–1519.

[19] C. V. Hollot, V. Misra, D. Towsley, and W. B. Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Trans. on Automatic Control*, vol. 47, no. 6, pp. 945–959, June 2002.

References: TCP/RED models

[20] P. Ranjan and E. H. Abed, "Bifurcation analysis of TCP-RED dynamics," in *Proc. ACC* , Anchorage, AK, USA, May 2002, vol. 1, pp. 2443–2448.

[21] P. Ranjan, E. H. Abed, and R. J. La, "Nonlinear instabilities in TCP- RED," in *Proc. IEEE INFOCOM 2002*, New York, NY, USA, June 2002, vol. 1, pp. 249–258.

[22] R. J. La, P. Ranjan, and E. H. Abed, "Nonlinearity of TCP and instability with RED," in *Proc. SPIE ITCom*, Boston, MA, USA, July 2002, pp. 283–294.

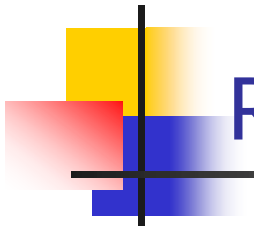
[23] P. Ranjan, R. J. La, and E. H. Abed, "Bifurcations of TCP and UDP traces under RED," in *Proc. 10th Mediterranean Conference on Control and Automation (MED) 2002*, Lisbon, Portugal, July 2002.

[24] P. Ranjan and E. H. Abed, "Chaotic behavior in TCP-RED," in *Proc. 41st IEEE Conference on Decision and Control*, Las Vegas, NE, Dec. 2002, pp. 540–542.

[25] P. Ranjan, E. H. Abed, and R. J. La "Communication delay and instability in rate-controlled networks," in *Proc. 42nd Conference on Decision and Control*, Maui, HI, Dec. 2003.

[26] R. J. La, "Instability of a tandem network and its propagation under RED," *IEEE Trans. Automatic Control*, vol. 49, no. 6, pp. 1006–1011, June 2004.

[27] P. Ranjan, E. H. Abed, and R. J. La, "Nonlinear instabilities in TCP-RED, *IEEE/ACM Trans. on Networking*, vol. 12, no. 6, pp. 1079–1092, Dec. 2004.



References: TCP/RED models

[28] I. Khalifa and Lj. Trajković, "An overview and comparison of analytical TCP models," in *Proc. IEEE International Symposium on Circuits and Systems*, Vancouver, BC, Canada, May 2004, vol. V, pp. 469–472.

[29] M. Liu, H. Zhang, and Lj. Trajković, "Stroboscopic model and bifurcations in TCP/RED," in *Proc. IEEE International Symposium on Circuits and Systems*, Kobe, Japan, May 2005, pp. 2060–2063.

[30] H. Zhang, M. Liu, V. Vukadinović, and Lj. Trajković, "Modeling TCP/RED: a dynamical approach," *Complex Dynamics in Communication Networks*, Springer Verlag, Series: Understanding Complex Systems, 2005, pp. 251–278.

[31] M. Liu, A. Marciello, M. di Bernardo, and Lj. Trajković, "Continuity-induced bifurcations in TCP/RED communication algorithms," to be presented at *IEEE International Symposium on Circuits and Systems*, Kos, Greece, May 2006.