A discrete-time model of TCP with Active Queue Management

Hui Zhang
hzhange@cs.sfu.ca

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

Roadmap

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

Motivation

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

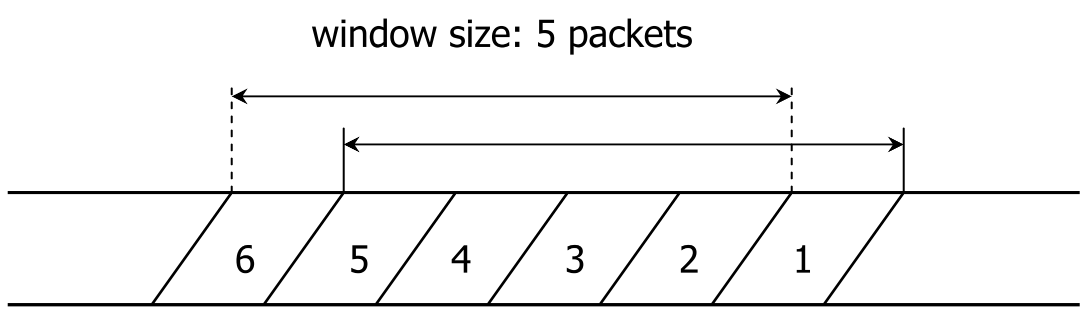
RED: Random Early Detection Gateways for Congestion Avoidance

Roadmap

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

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

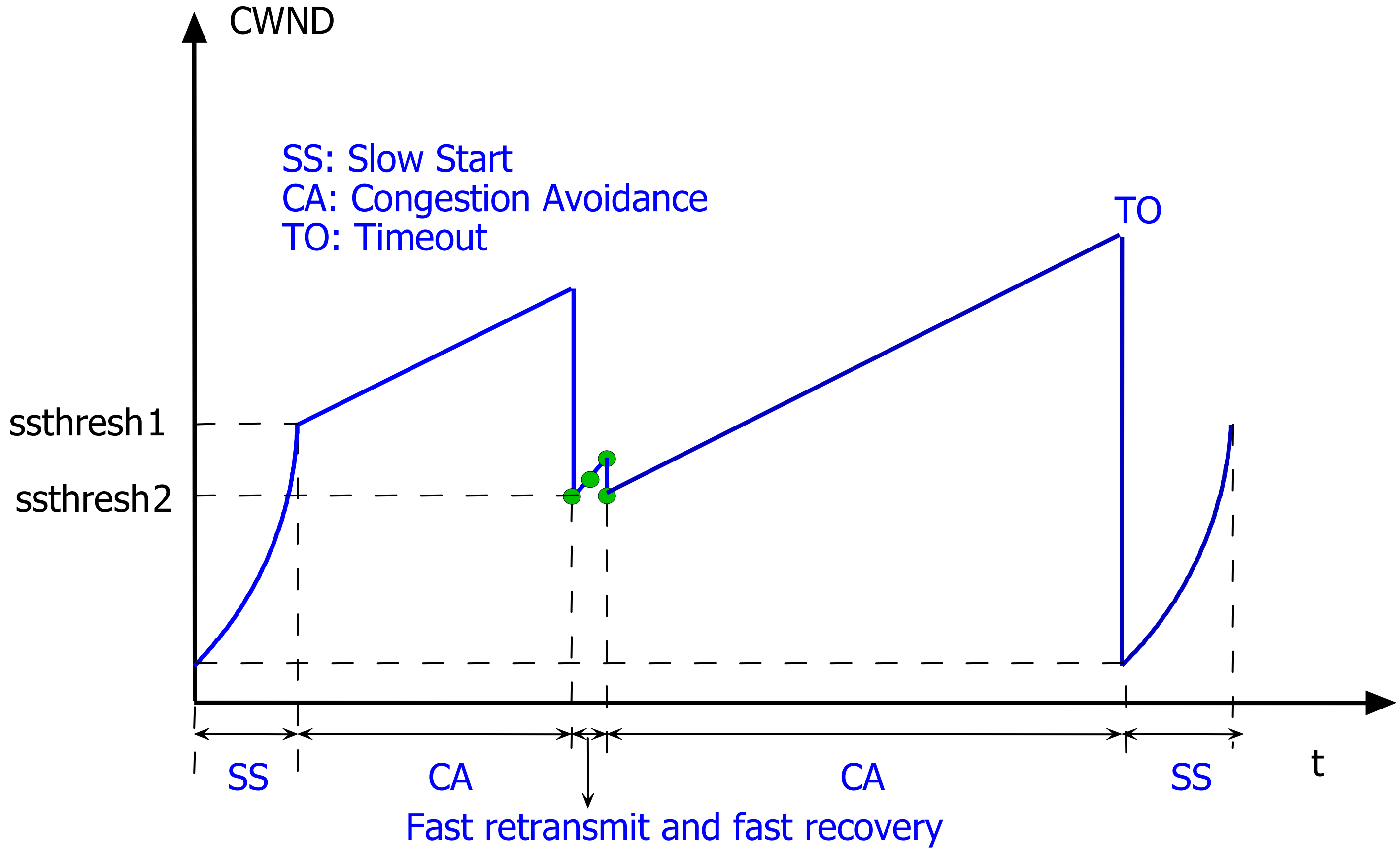


OSI: Open System Interconnection reference model

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, RFC 2582)
    - new fast recovery algorithm
  - 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

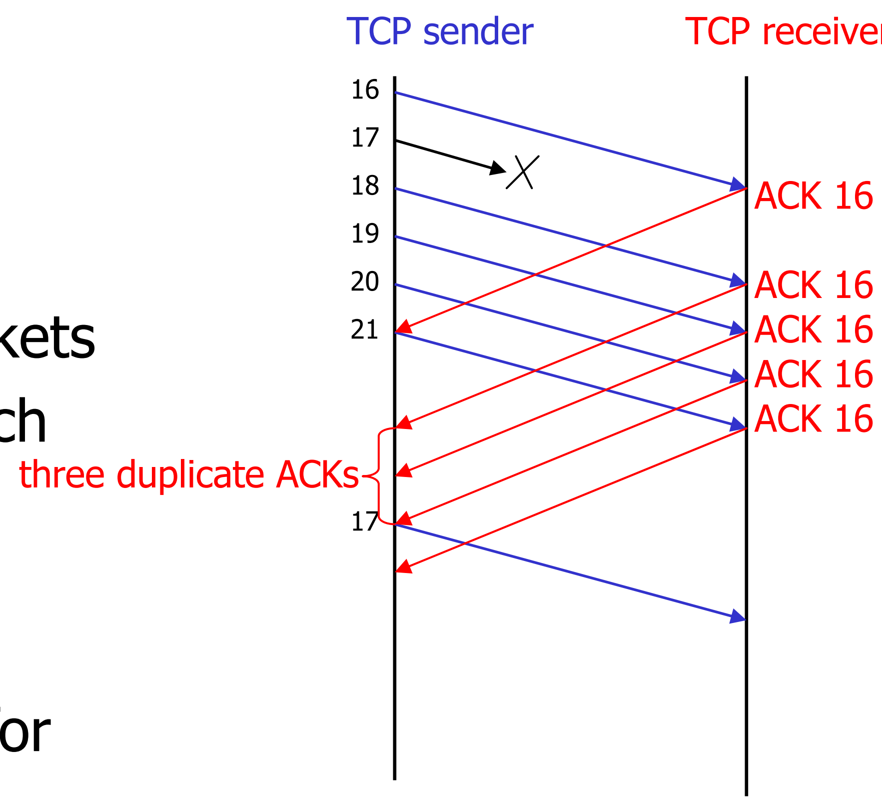
TCP Reno: slow start and congestion avoidance

- 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

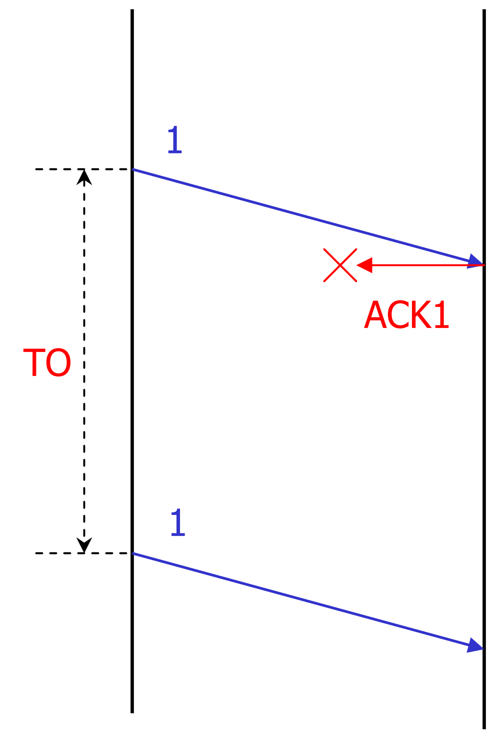
TCP Reno: fast retransmit and fast recovery

- 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, and 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 $q_{\max}$

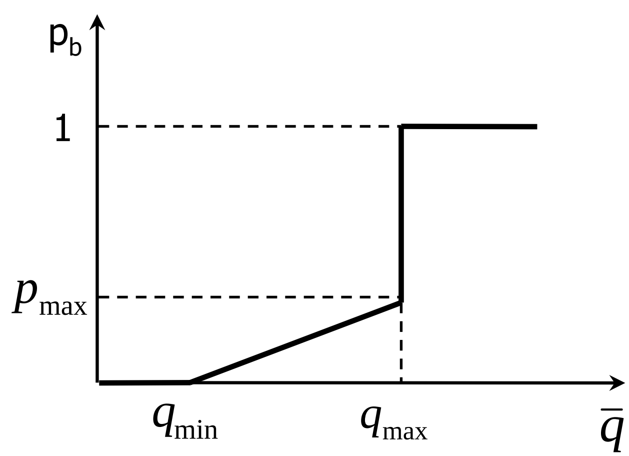
RED algorithm

Calculate:

- average queue size for each packet arrival

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

- 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

ARED

- Adaptive RED aims to improve the robustness of RED with minimal modifications:

  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/.

- Maintain a stable average queue size (target)
- Adjust p_{max} in response to dynamical changes of the average queue size

ARED algorithm

- Update $p_{\max}$

  For every interval

  - If $\bar{q} > target$

    $$p_{\max} = p_{\max} + \alpha$$

  - Else if $\bar{q} < target$

    $$p_{\max} = p_{\max} \times \beta$$

$$target : [q_{\min} + 0.4(q_{\max} - q_{\min}), q_{\min} + 0.6(q_{\max} - q_{\min})]$$

$$\alpha : \min(0.01, p_{\max}/4)$$

$$\beta : 0.9$$

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-RED model

Roadmap

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

Modeling methodology

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

S-RED model

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

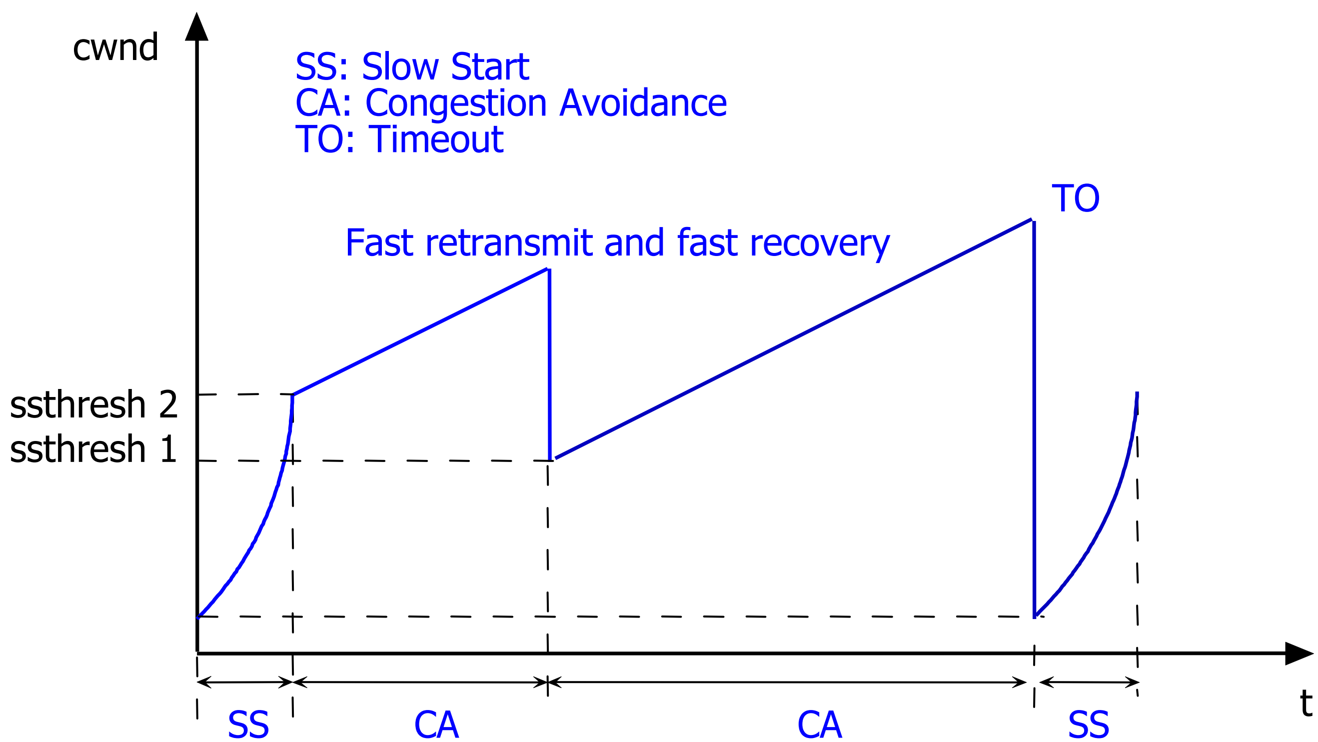
    J. Padhye, V. Firoiu, and D. F. Towsley, “Modeling TCP Reno performance: a simple model and its empirical validation,” *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 133-145, Apr. 2000.
  - Captures the basic RED algorithm

Assumptions

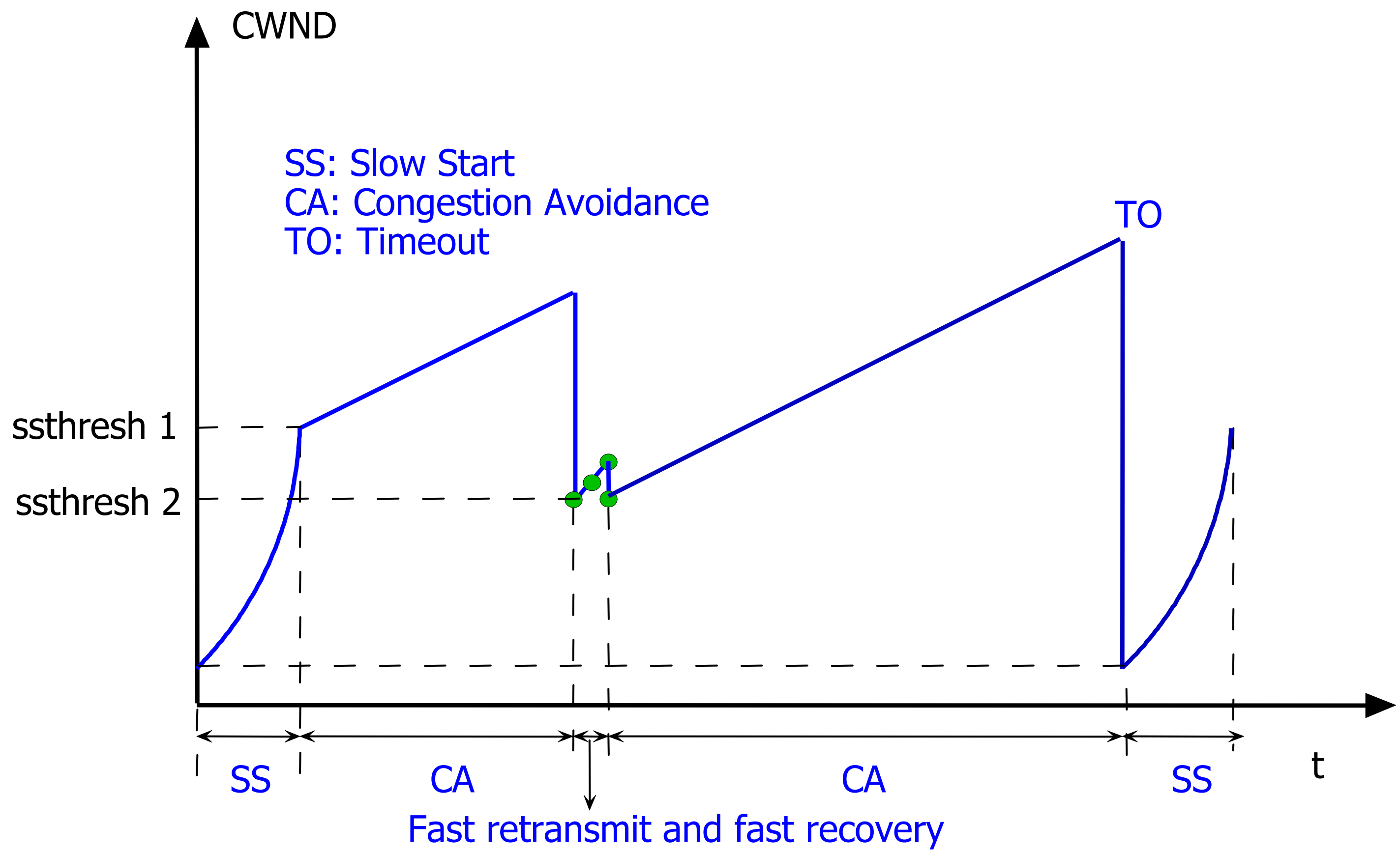
- Long-lived TCP connection
- Constant propagation delay between the source and the destination
- Constant packet size
- Timeout occurs only due to packet loss
- The system is sampled at the end of every RTT

S-RED model simplifications

- Simplified fast recovery



TCP Reno: fast recovery



S-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: *count* not used

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

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

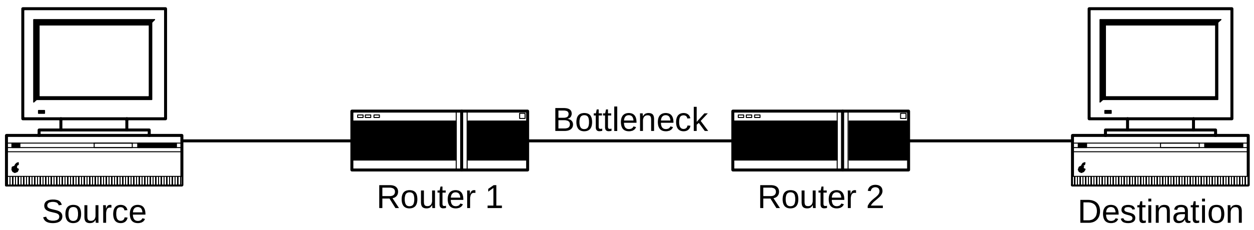
$$p_a = \frac{p_b}{1 - count \times p_b}$$

$$\xrightarrow{p_a = p_b}$$

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

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

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

- Calculate the average queue size:

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

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

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

  - 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-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-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-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-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: $p_k \cdot W_k \ge 1.5$

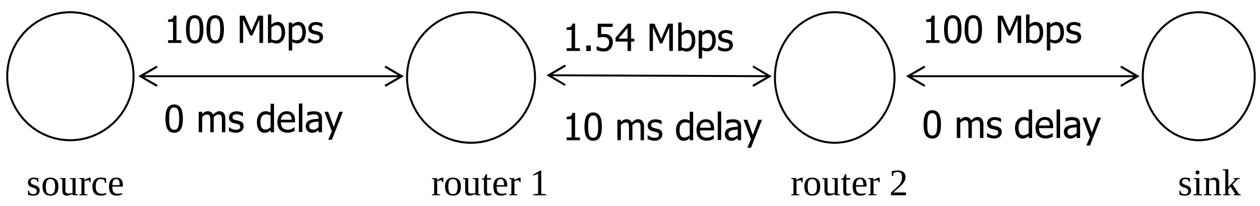
$$W_{k+1} = 0$$

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

Roadmap

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

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:

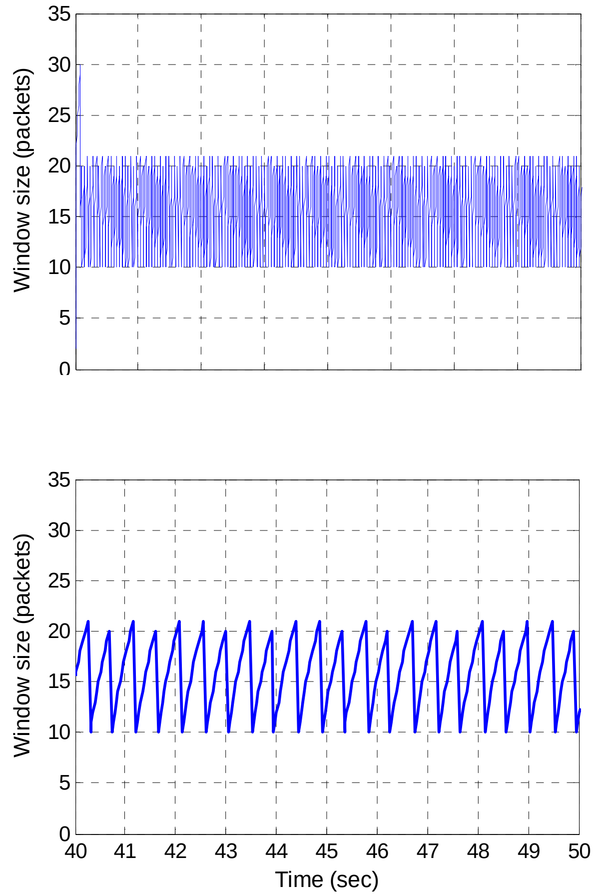
  S. Floyd, "RED: Discussions of Setting Parameters," Nov. 1997: http://www.icir.org/floyd/REDparameters.txt

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)

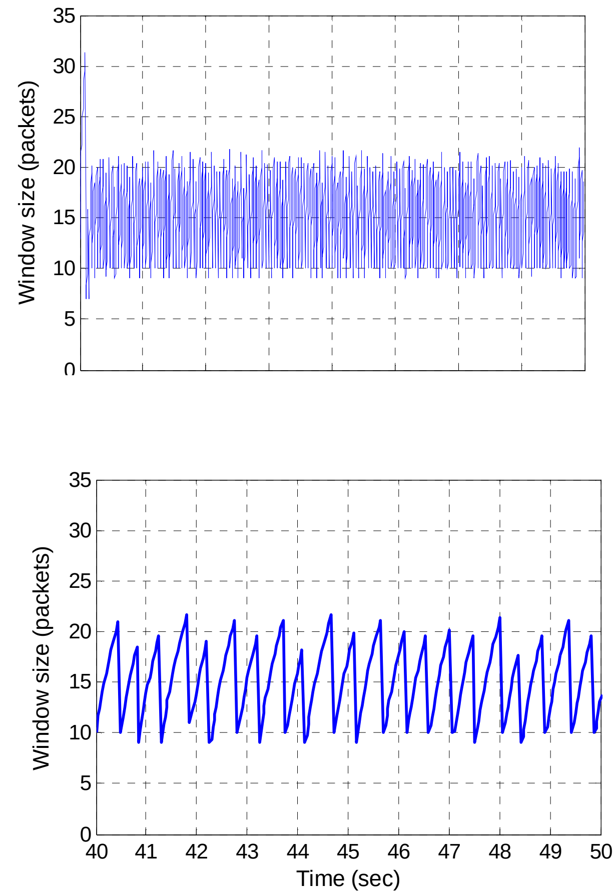
S-RED model validation

- 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

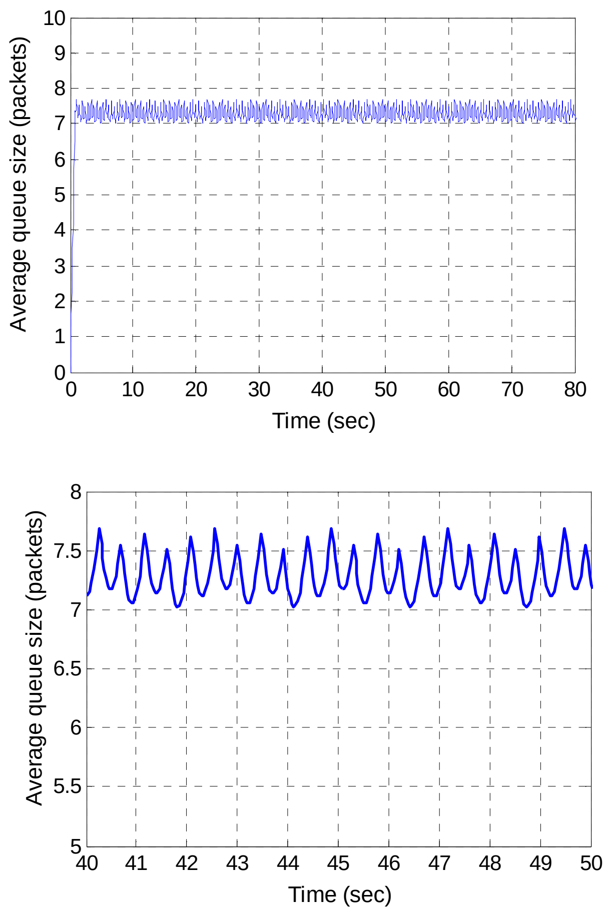


S-RED model

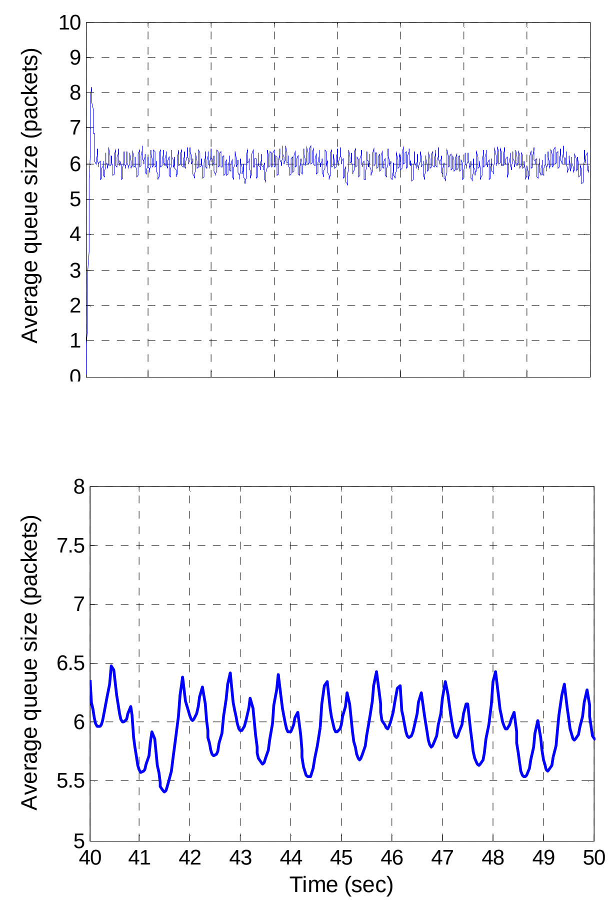


ns-2

Average queue size: waveforms



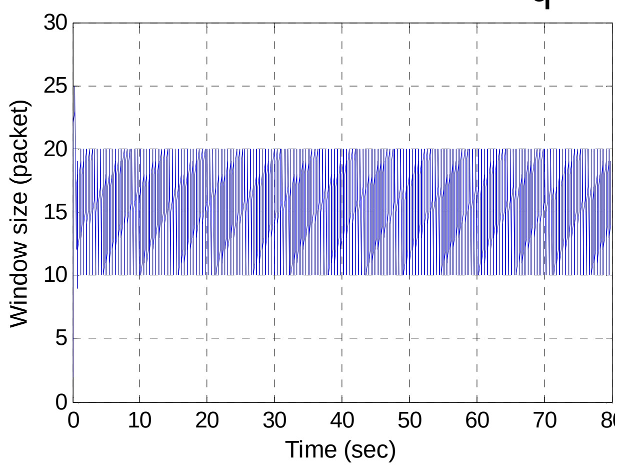
S-RED model



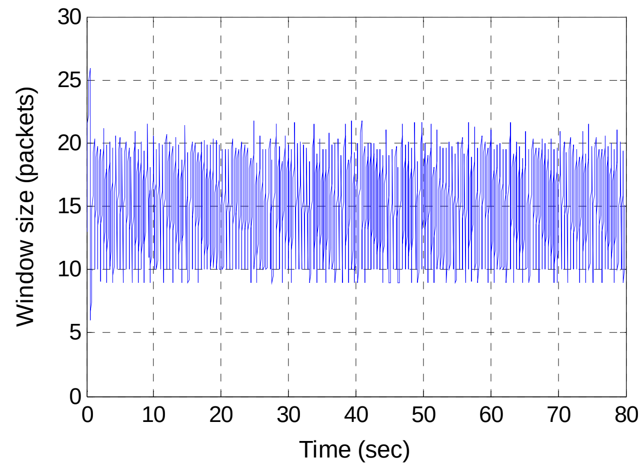
ns-2

Model validation: w_q

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



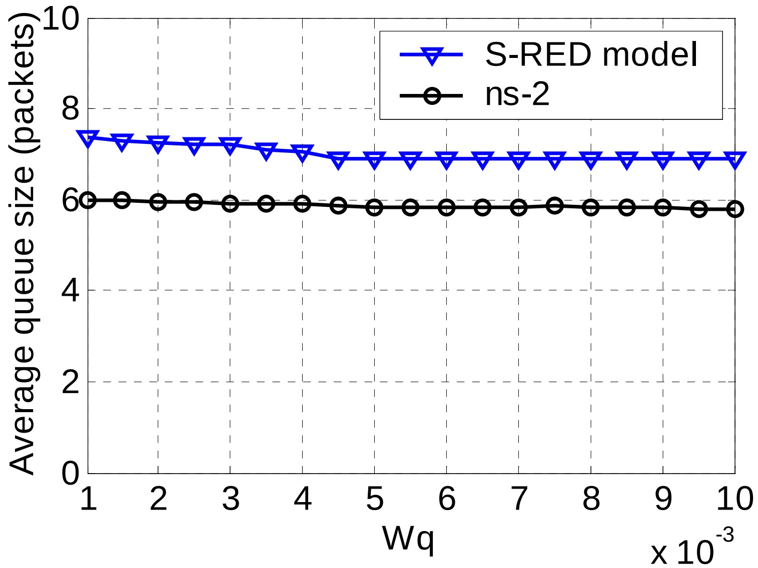
S-RED



ns-2

Model validation: w_q

- average queue size during steady state:



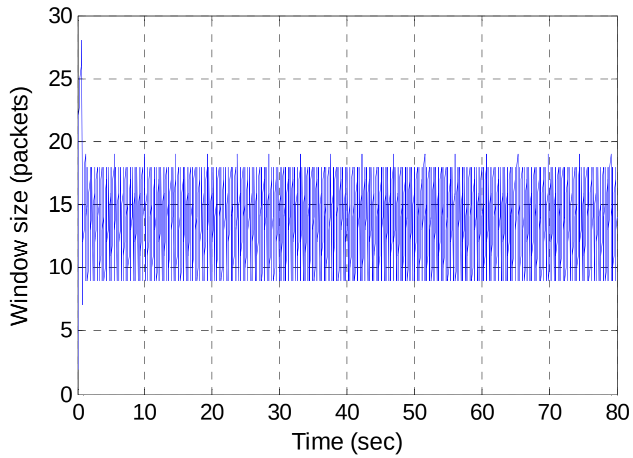
Model validation: w_q

- Comparison of system variables:

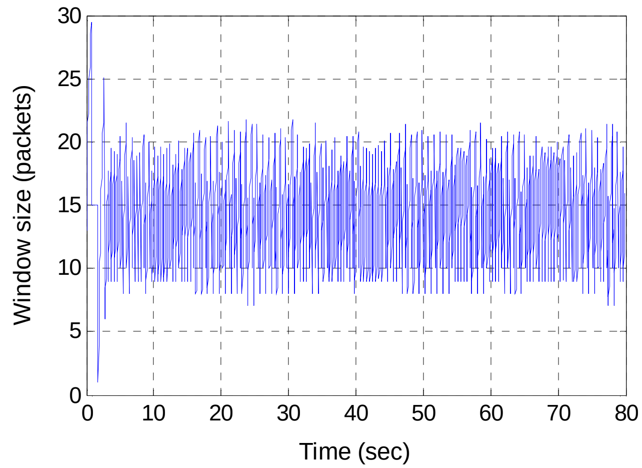
Parameters	Average RTT (msec)			Sending rate (packets/sec)			Drop rate (%)		
weight (w_q)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)
0.001	40.3	36.1	11.63	384.99	384.71	0.073	0.55	0.54	1.29
0.002	39.9	36.0	10.83	384.98	384.77	0.056	0.56	0.55	2.56
0.004	39.4	36.2	8.80	385.11	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



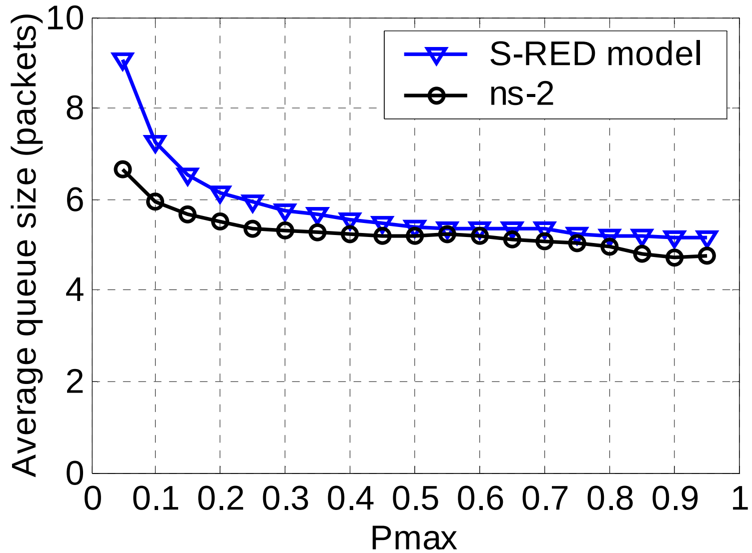
S-RED



ns-2

Model validation: p_{max}

- average queue size during steady state:



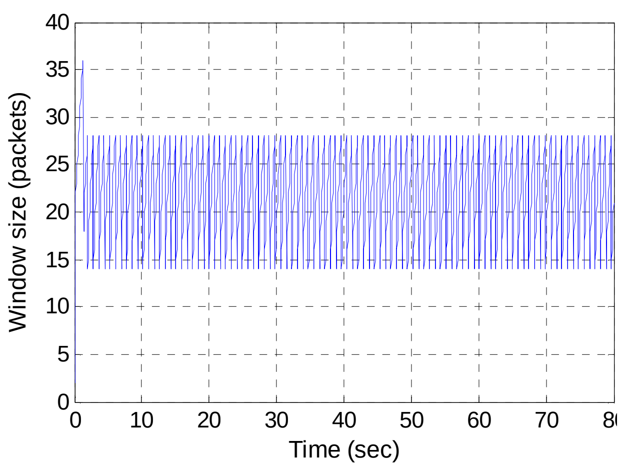
Model validation: p_{max}

- Comparison of system variables:

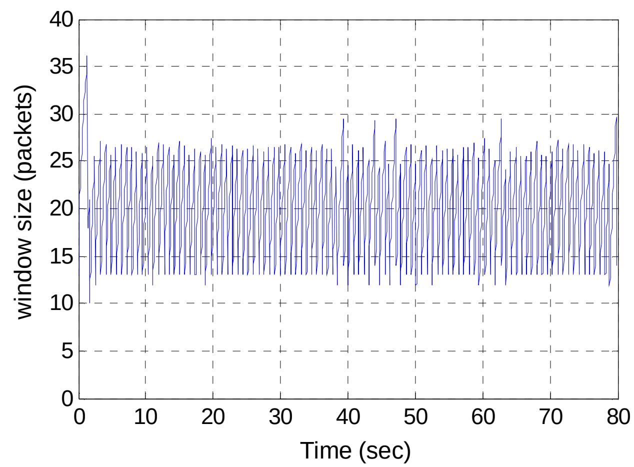
	Average RTT (msec)			Sending rate (packets/sec)			Drop rate (%)		
p_{max}	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)	S-RED model	ns-2	Δ (%)
0.05	44.3	38.1	16.27	385.13	384.70	0.11	0.45	0.51	-11.76
0.10	39.9	36.0	10.83	384.98	384.77	0.06	0.56	0.55	2.56
0.25	36.5	34.5	5.80	384.93	384.73	0.05	0.65	0.59	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



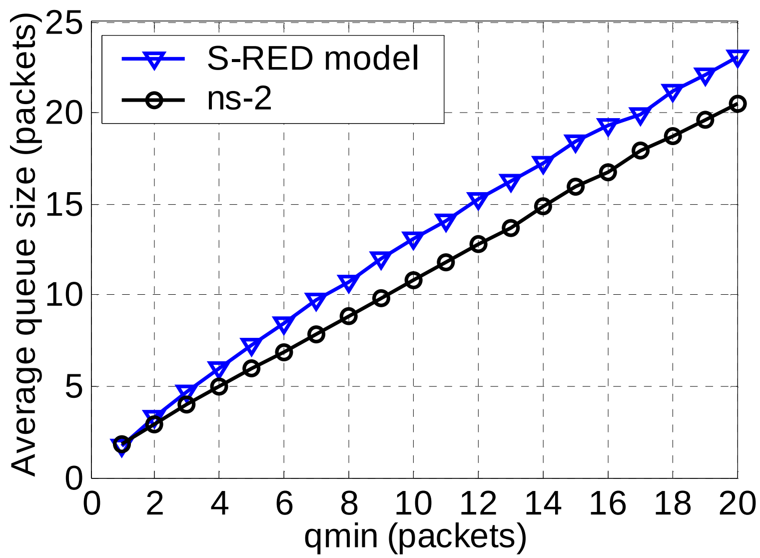
S-RED



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-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

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

Comparison: S-RED model vs. M-model

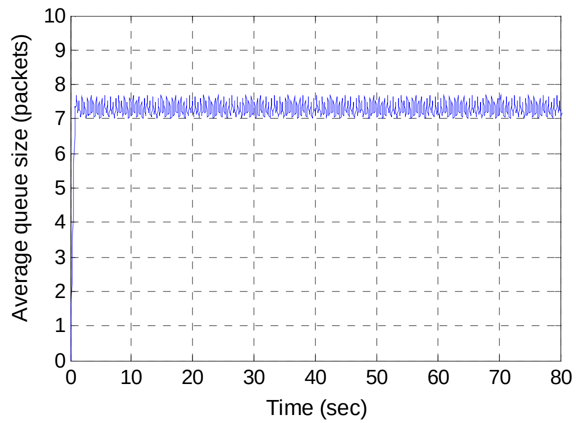
- M-model:

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

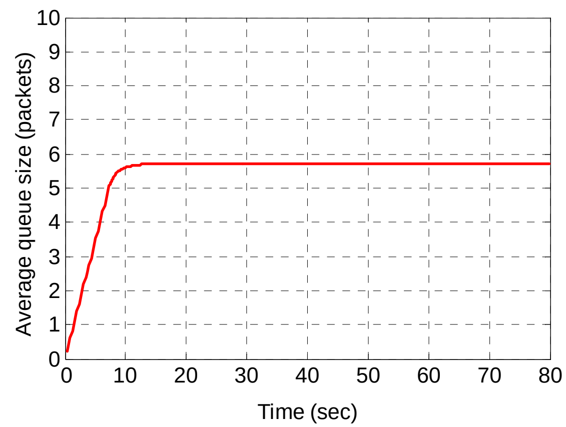
  P. Ranjan, E. H. Abed, and Richard J. La, "Nonlinear instabilities in TCP-RED," in *Proc. IEEE INFOCOM 2002*, New York, NY, USA, June 2002, vol. 1, pp. 249-258
- One state variable: average queue size

Model comparison: default parameters

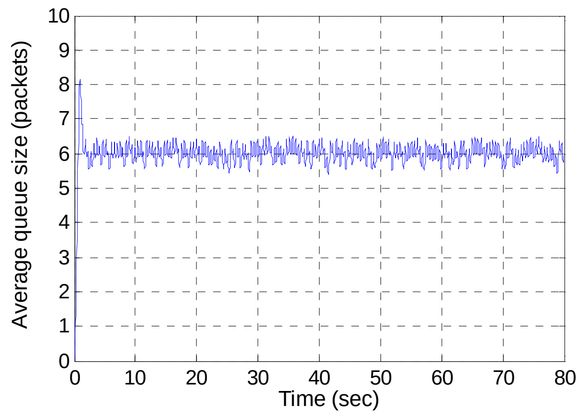
- The waveform of the average queue size with default RED parameters:



S-RED



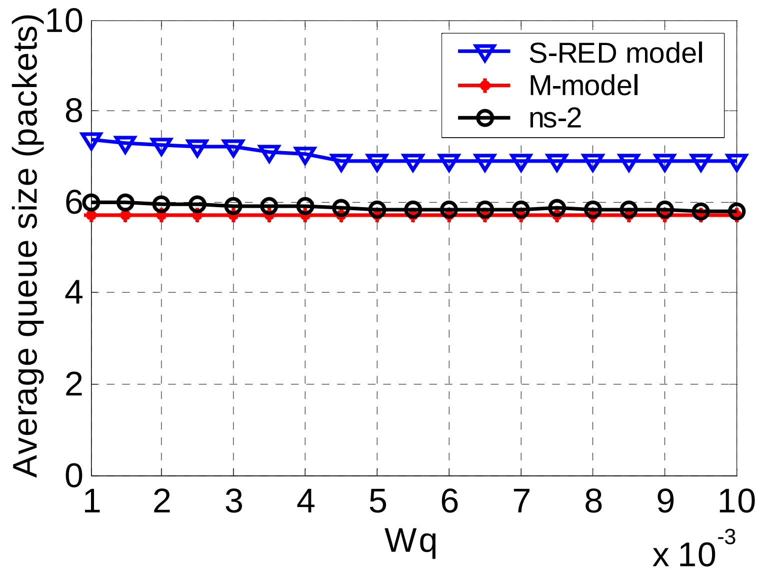
M-model



ns-2

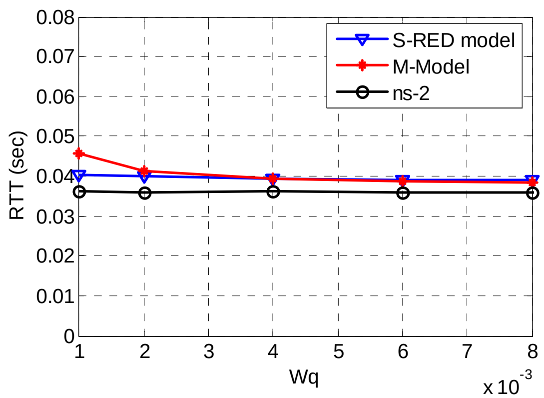
Model comparisons: w_q

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

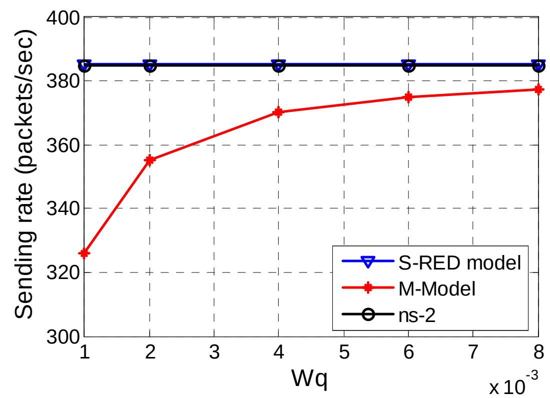


Model comparisons: w_q

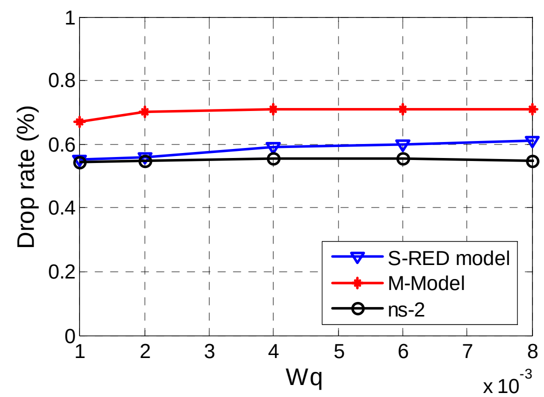
- system variables:



RTT



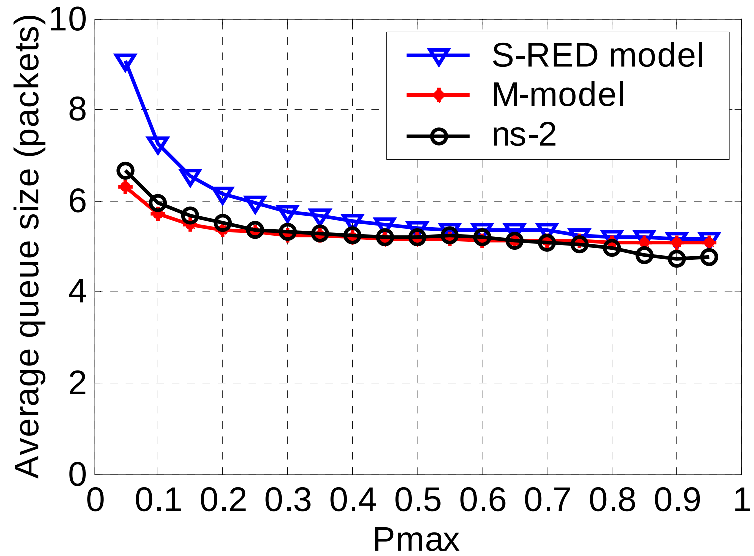
sending rate



drop rate

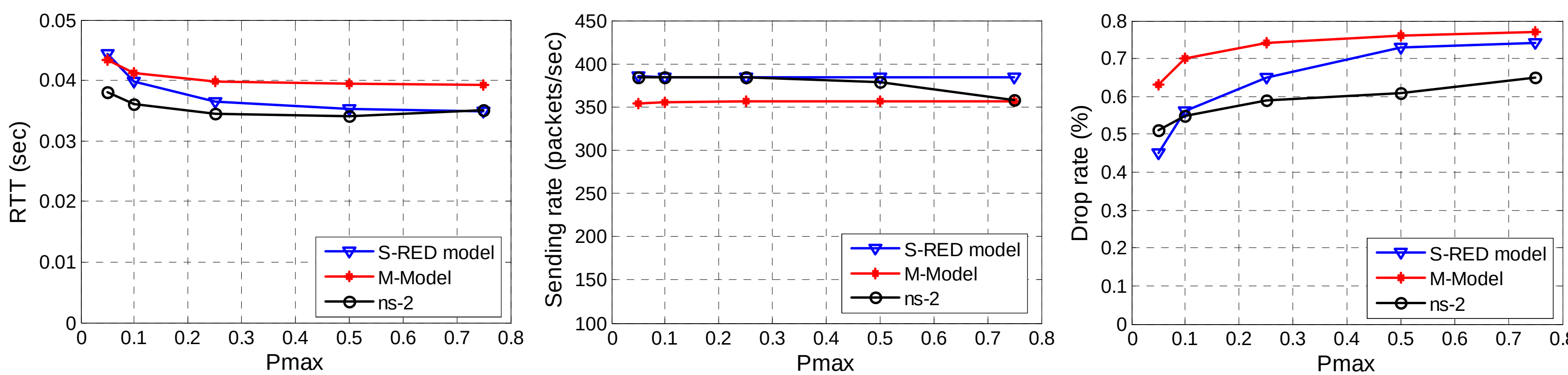
Model comparisons: p_{max}

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



Model comparisons: p_{max}

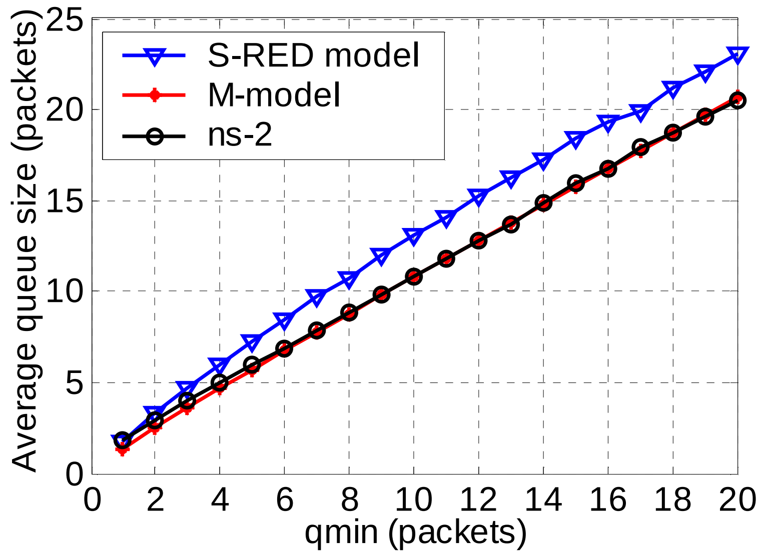
- system variables:



RTT | sending rate | drop rate

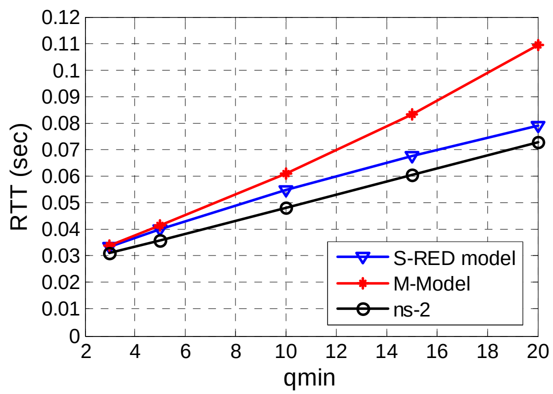
Model comparisons: q_{min} and q_{max}

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

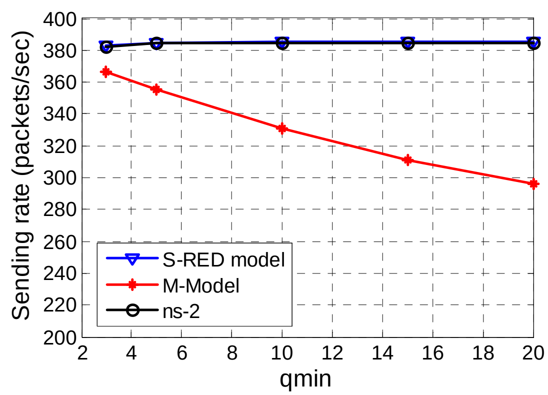


Model comparisons: q_{min} and q_{max}

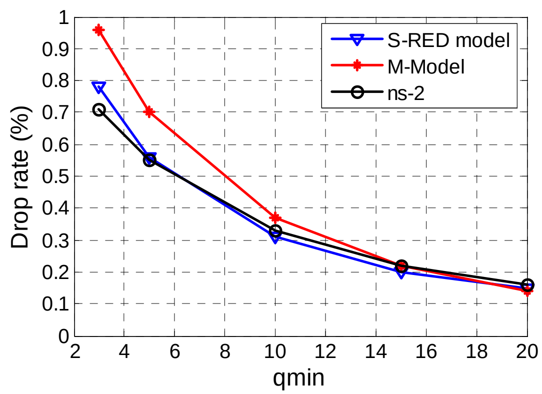
- system variables:



RTT



sending rate



drop rate

Model comparison: summary

- S-RED model captures dynamical details of TCP/RED
- RTT, sending rate, and drop rate: S-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

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

S-RED: modification

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

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

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

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

$\xrightarrow{p_a = p_b}$

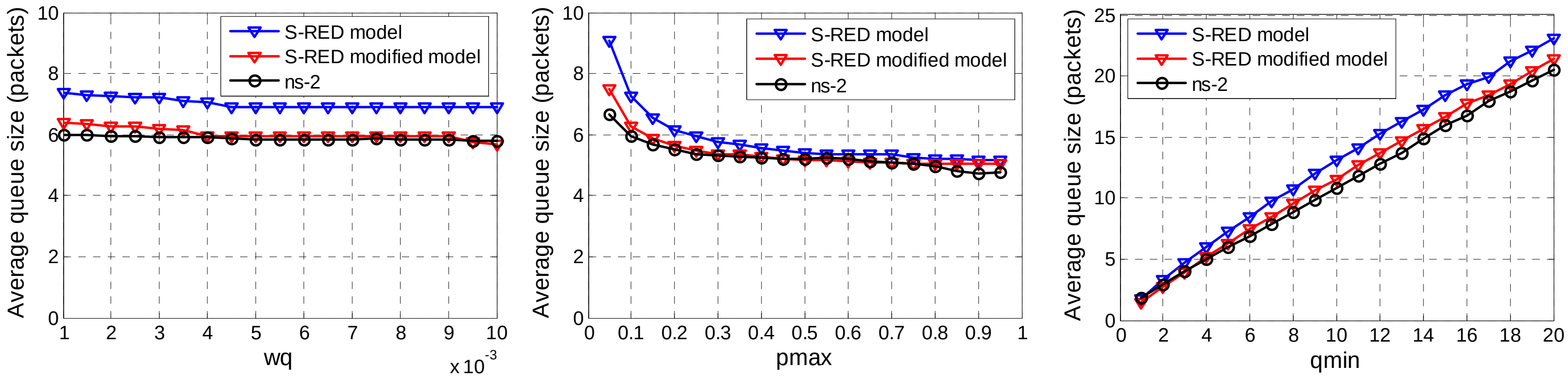
If $(q_{\min} < \bar{q} < q_{\max})$

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

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

Modified S-RED model

- Modification: $\alpha = 1.8$



Roadmap

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

S-ARED model

- ARED algorithm:
  - extension to the RED algorithm
  - designed to improve RED's robustness
  - achieves a specified target value for the average queue size by dynamically updating $p_{\max}$
- S-ARED model:
  - developed based on S-RED model
  - keeps all the assumptions of S-RED model
  - state variables: window size and average queue size
  - interval for update $p_{\max}$: C·RTT (C = 10)

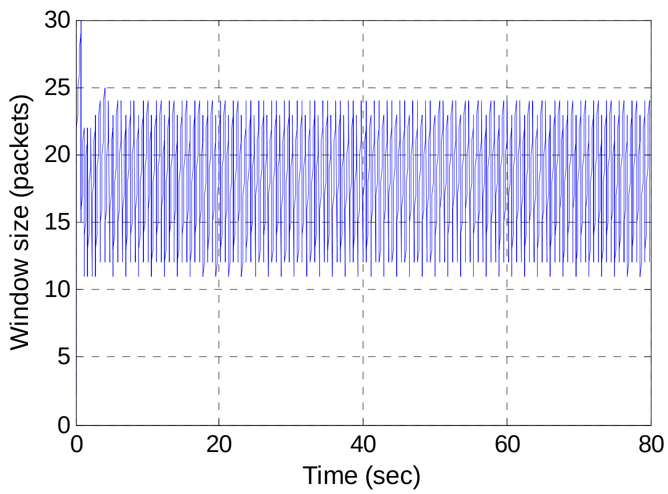
S-ARED: model validation

- The simulation scenario used to validate S-ARED model is identical to the scenario used for S-RED model
- Parameters:

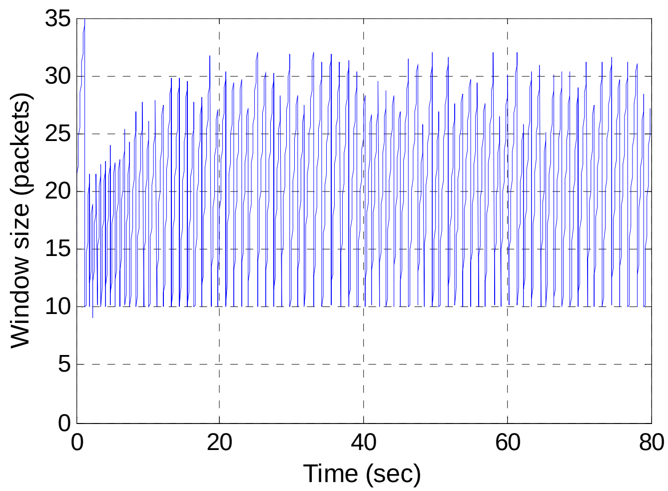
Packet size M (bytes)	500
Maximum drop probability p_{max}	0.1
Minimum queue threshold q_{min} (packets)	5
Maximum queue threshold q_{max} (packets)	15
Queue weight w_q	0.002
Sample_interval C	10

S-ARED: validation

- window size: waveforms



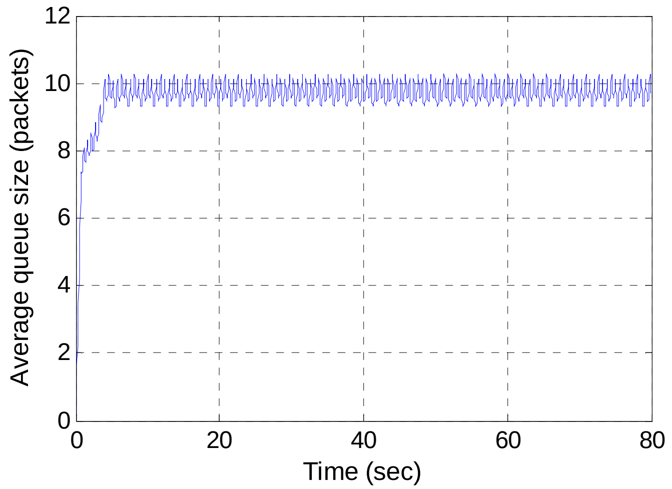
S-ARED



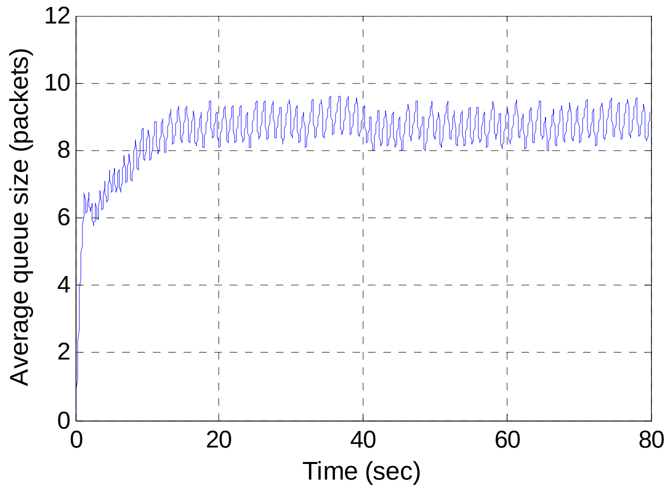
ns-2

S-ARED: validation

- average queue size: waveforms



S-ARED



ns-2

S-ARED: validation

S-ARED:

- captures the basic TCP/ARED system
- window size and average queue size: ~ 10 % difference with the ns-2 simulation results
- RTT and sending rate match well the ns-2 simulation results
- drop probability shows large mismatch with ns-2

	S-ARED model	ns-2	∆ (%)
Window size (packets)	17.66	20.10	-12.13
Average queue size (packets)	9.69	8.69	11.50
RTT (ms)	45.9	44.2	3.84
Sending rate (packets/s)	385.03	377.41	2.02
Drop probability (%)	0.44	0.26	69

Road map

- Motivation
- Background
- S-RED: a discrete-time model of TCP Reno with RED
- Model validation of S-RED
- Comparison of TCP/RED models
- S-RED: a modification
- S-ARED: extension to S-RED
- Conclusions
- Future work

Conclusions

- We developed a second-order discrete-time model for TCP Reno with RED
- The S-RED model includes slow start, congestion avoidance, fast retransmit, timeout, elements of fast recovery, and RED
- The model is validated by comparing its performance to ns-2 simulations
- The S-RED model can capture the main features of the dynamic behavior of TCP Reno with RED
- We compared S-RED model with the M-model

Conclusions

- After modifying the drop probability, the S-RED model shows a better agreement with ns-2 simulation results
- We also introduced the S-ARED model, an extension to the S-RED model
- S-RED model may be used for performance evaluation of TCP/RED based AQM schemes

Future work

- Analyze nonlinear phenomena of TCP/RED:
  - chaos and bifurcation
- Evaluate performance of other TCP/RED based AQM schemes:
  - CHOKe
- Develop new AQM algorithms

References: TCP and RED

[1] M. Allman, V. Paxson, and W. Steven, "TCP Congestion Control," *Request for Comment (RFC) 2581*, Apr. 1999.

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

[3] R. Braden, "Requirements for Internet hosts: communication layers," *Request for Comment (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] 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/.

[7] 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.

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

References: TCP and RED

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

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

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

[12]] V. Paxson and M. Allman, "Computing TCP's Retransmission Timer," *Request for Comment (RFC) 2988*, Nov. 2000.

[13] J. Postel, "Transmission control protocol," *Request for Comment (RFC) 793*, Sept. 1981.

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

References: modeling

[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] 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.

[18] 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.

[19] 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.

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

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

Thank You !

Questions ?

References: TCP and RED

[1] M. Allman, V. Paxson, and W. Steven, "TCP Congestion Control," *Request for Comment (RFC) 2581*, Apr. 1999.

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

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

[4] L. Brakmo and L. Peterson, "TCP Vegas: end to end congestion avoidance on a global Internet," *IEEE Journal on Selected Areas in Communication*, vol. 13, no. 8, pp. 1465-1480, Oct. 1995.

[5] 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.

[6] 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.

[7] 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.

[8] 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 and RED

[9] 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.

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

[11] 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.

[12] 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.

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

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

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

[16] R. Pan, K. Psounis, and B. Prabhakar, "CHOKe, A stateless active queue management scheme for approximating fair bandwidth allocation," in *Proc. IEEE INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000, vol. 2, pp. 942-951.

References: TCP and RED

[17] V. Paxson and M. Allman, "Computing TCP's Retransmission Timer," *Request for Comment (RFC) 2988*, Nov. 2000.

[18] J. Postel, "Transmission control protocol," *Request for Comment (RFC) 793*, Sept. 1981.

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

References: modeling

[20] 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.

[21] C. Casetti and M. Meo, "A new approach to model the stationary behavior of TCP connections," in *Proc. INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000, vol. 1, pp. 367-375.

[22] W. C. Y. Chan and C. K. Tse, "Study of bifurcation in current programmed DC/DC boost converters: from quasi-periodicity to period-doubling," *IEEE Trans. Circuit and Systems I*, vol. 44, no. 12, pp. 1129-1142, Dec. 1997.

[23] K. W. E. Cheng, M. Liu, and J. Wu, "Chaos study and parameter-space analysis of the DC-DC buck-boost converter," *IEE Proceedings-Electric Power Applications*, vol. 150, pp. 126-138, Mar. 2003.

[24] 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.

[25] P. Kuusela, P. Lassila, J. Virtamo, and P. Key, "Modeling RED with idealized TCP sources," in *9th IFIP Conference on Performance Modeling and Evaluation of ATM and IP Networks 2001*, Budapest, Hungary, June 2001, pp. 155-166.

[26] 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.

References: modeling

[27] S. H. Low and D. E. Lapsley, "Optimization flow control I: basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, no. 6, pp. 861-874, Dec. 1999.

[28] S. H. Low, F. Paganini, J. Wang, S. Adlakha, and J. C. Doyle, "Dynamics of TCP/RED and a scalable control," in *Proc. IEEE INFOCOM 2002*, New York, NY, USA., June 2002, vol. 1, pp. 239-248.

[29] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM Computer Communication Review*, vol. 27, no. 3, pp. 67-82, July 1997.

[30] M. Mellia, I. Stoica, and H. Zhang, "TCP model for short lived flows," *IEEE Communications Letters*, vol. 6, no. 2, pp. 85-87, Feb. 2002.

[31] V. Misra, W. B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proc. ACM SIGCOMM 2000*, Stockholm, Sweden, Aug. 2000, pp. 151-160.

[32] 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.

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

References: modeling

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

[35] R. Roy, R. C. Mudumbai, and S. S. Panwar, "Analysis of TCP congestion control using a fluid model," in *Proc. IEEE ICC 2001*, Helsinki, Finland, June 2001, vol. 8, pp. 2396-2403.