Evaluation of TCP congestion control mechanisms using OPNET simulator

ENSC 835: COMMUNICATION NETWORKS

FINAL PROJECT PRESENTATION
Spring 2008

Laxmi Subedi
URL: http://www.sfu.ca/~lsa38/project.html
Email: lsa38@cs.sfu.ca

Roadmap

- Introduction
- Objective and scope
- Implementation
- Simulation
- Conclusion
- Future work
- References

Introduction: Transmission Control Protocol

- reliable, error free, connection-oriented, point-to-point with flow control
- used by most applications: FTP, HTTP, SMTP
- carries about 90% internet traffic [1]
- developed based on wired network properties
- Sliding window protocol for flow control
- problems exist when an intermediate router run out of memory and drops the packets termed as Congestion
- no information from lower layers on congestion

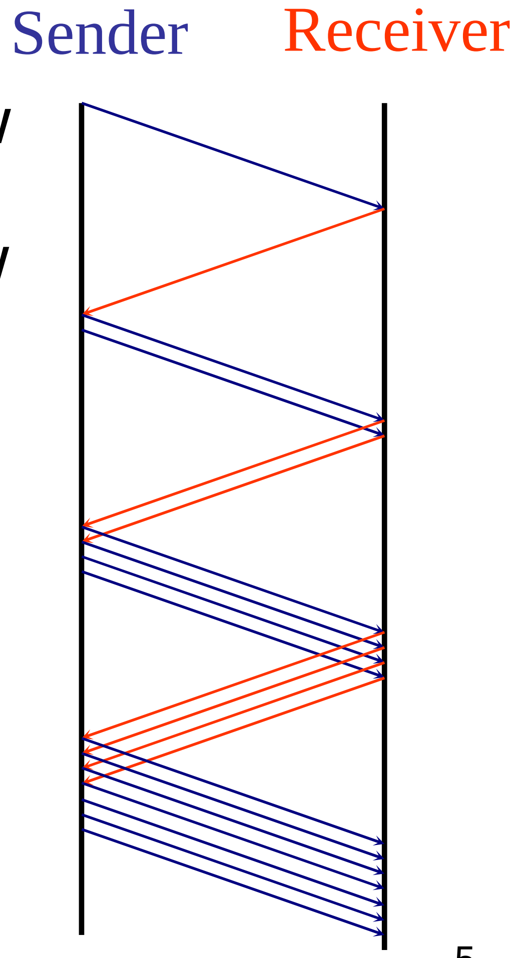
[1] I. Khalifa and Lj. Trajkovic, "An overview and comparison of analytical TCP models," (invited session) *Proc. IEEE Int. Symp. Circuits and Systems*, Vancouver, British Columbia, Canada, May 2004, vol. V, pp. 469-472.

Introduction: Transmission Control Protocol

- packet loss and delay indicate congestion
- TCP maintains congestion window accordingly
- Congestion window:
  - maximum number of bytes transmitted without ACKs sent packets
  - minimize router buffer overflow and retransmission effort
- Approaches to manage congestion window:
  - Slow Start
  - Addititive Increase Multiplicative Decrease
  - Congestion Avoidance
  - Fast retransmit and Fast recovery

Introduction: Slow Start

- initial congestion window: 1 MSS
- on receiving ACK, congestion window is set to 2
- process increases congestion window exponentially
- continues exponential increase until loss event or advertised receiver window, whichever is minimum
- rapid utilization of available bandwidth.

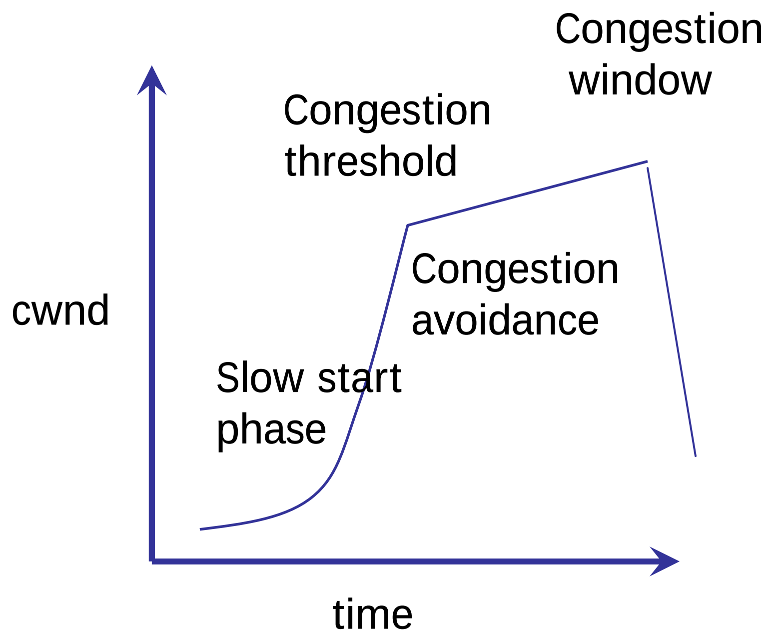


Introduction: AIMD

- Additive Increase:
  - under no congestion, window increases by 1 MSS for every RTT (Rather than for every ACK)
  - helps in additional use of bandwidth
- Multiplicative Decrease:
  - on loss event, window reduced to half of the current window size
  - continues to decrease half of the previous window on successive drops
  - minimum window size is 1 MSS

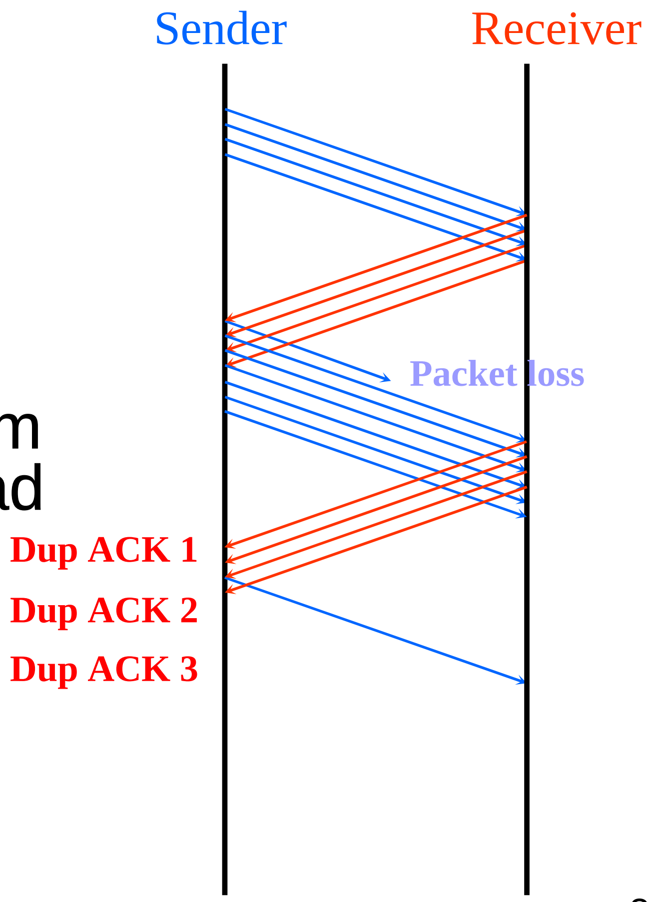
Introduction: Congestion Avoidance

- after slow start threshold, window increases by 1 MSS for every RTT rather than exponentially



Introduction: Fast retransmit and Fast recovery

- Fast retransmit:
  - uses duplicate Acks to retransmit
  - retransmits without waiting for timeout
- Fast recovery:
  - after fast retransmit, perform congestion avoidance instead of slow start
  - duplicate ACK indicates availability of network resources



Introduction: Algorithms

- Reno, New-Reno, and SACK
  - use slow start, congestion avoidance, fast retransmission and fast recovery
- Reno
  - remains in fast recovery until it receives duplicate acknowledgments
  - transmits new packet for additional received duplicate acknowledgment
- SACK
  - acknowledge out-of-order segment selectively rather than cumulatively

Introduction: Algorithms

- NewReno
  - improves retransmission during fast recovery
  - does not exit fast-recovery until acknowledgment of all data which are not acknowledged while entering fast recovery
  - uses partial ACKs (ACK segments that do not acknowledge all the information that has been sent up to the moment when they are issued)

Introduction: Literature Review

- “New-Reno TCP in absence of SACK avoids Reno TCP's performance problems when multiple packets are dropped from a window of data. But still SACK is better in case of multiple packet drops.” [2]

- “SACK performance deteriorates in case of congested network. In case of non-congested network all three algorithms have comparable performance.” [3]

[2] S. Floyd and K. Fall, “Simulation Based Comparisons of Tahoe, Reno and Sack TCP”, ACM Computer Communication Review, 1996, Vol.26, No.3: 5–21.

[3] R. Paul and Lj. Trajkovic,"Selective-TCP for wired/wireless networks,'' Proc. SPECTS 2006, Calgary, AL, Canada, Aug. 2006, pp. 339-346.

Introduction: Literature Review

- New-Reno outperforms Reno and SACK when no packet losses occur during the slow-start phase. Bandwidth-delay product leads to performance degradation of regardless of TCP versions and the bottleneck buffer size. [4]

- Performance of New-Reno is worse than Tahoe. SACK is the best and the most robust over the wireless channel. [5]

[4] H. lee, S. Lee and, Y.Choi, "The influence of the large bandwidth-delay product on TCP Reno, NewReno, and SACK", *Proc. Information Networking Conference*, Oita, Japan, 2001, pp. 327–334.

[5] F. Anjum and L. Tassiulas, "Comparative study of various TCP versions over a wireless link with correlated losses**",** *IEEE/ACM Transactions on Networking (TON)*, NJ, USA, June 2003, Vol. 11, Issue 3, pp. 370 – 383.

Project: Objective and scope

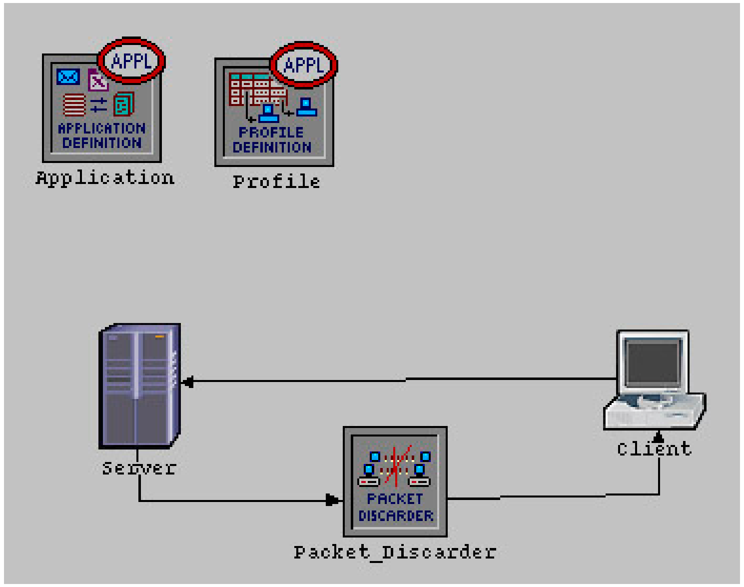
- Objective:
  - observe, analyze, and compare congestion window recovery processes
  - analyze congestion window in case of link disconnection (wireless property)
  - get familiar with OPNET simulation tool
- Scope:
  - compare Reno, SACK, and New-Reno
  - analyze Congestion window
  - simulate drop and disconnection events

Implementation: Steps

- get familiar with Opnet tool
- get familiar with algorithms implemented
- select and built appropriate nodes
- built suitable network topology
- select appropriate parameter to evaluate performance of algorithms during congestion
- select appropriate statistics
- perform analysis on collected statistics

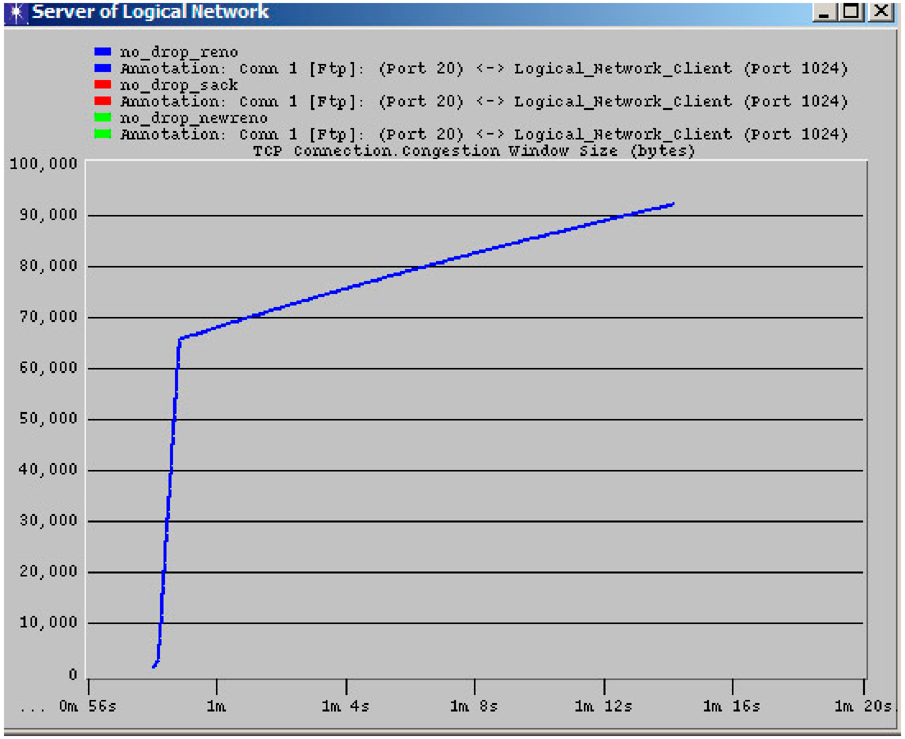
Simulation: Simple Client Server Topology

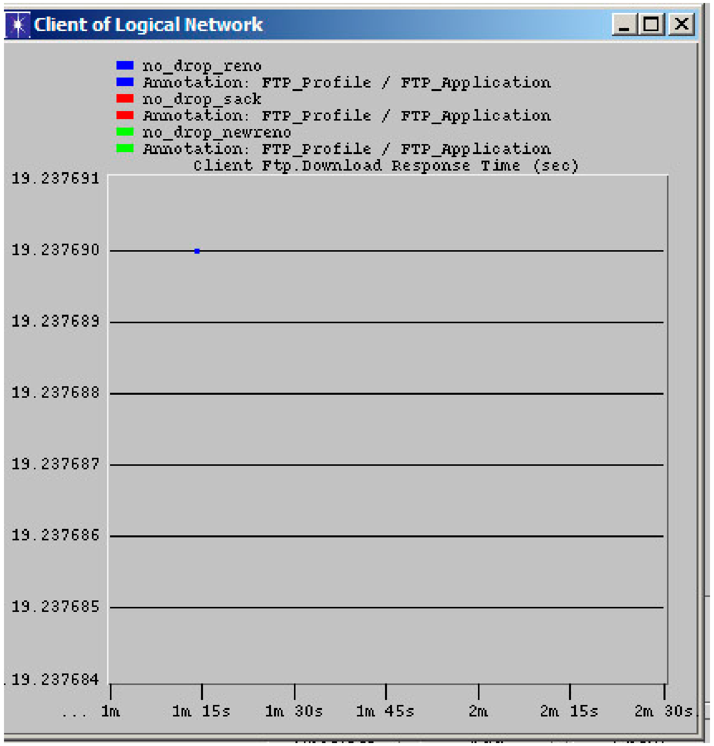
- Client and Server connected with 1.5 Mbps line
- Packet discarder, between server to client link, impose packet loss
- File of size 3 MB is transferred from server to client using ftp application
- Different packet loss scenario is created for 0.5 second



Client server topology

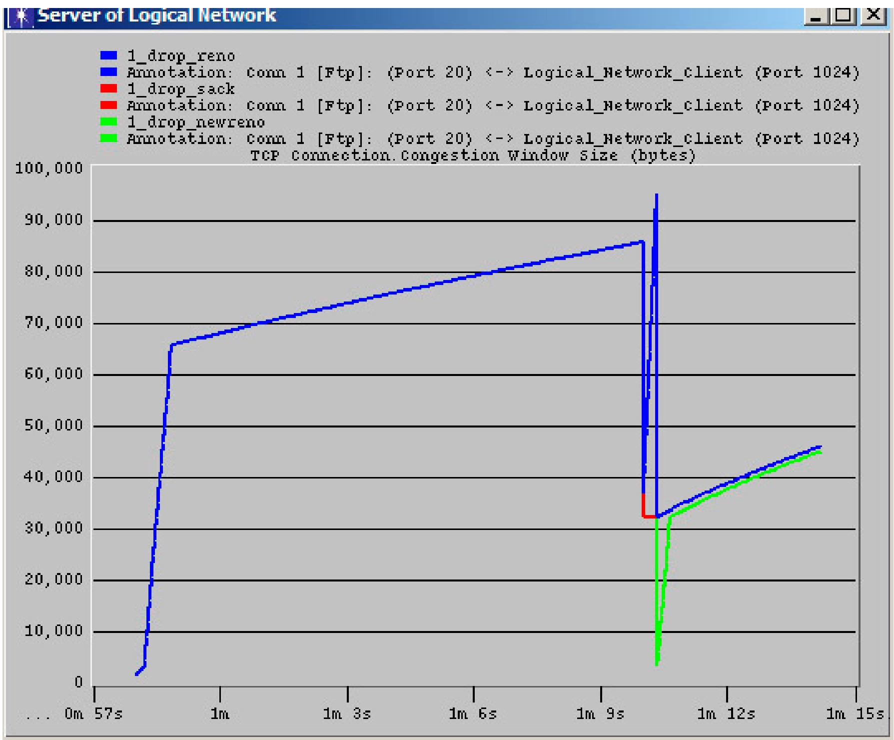
No packet loss



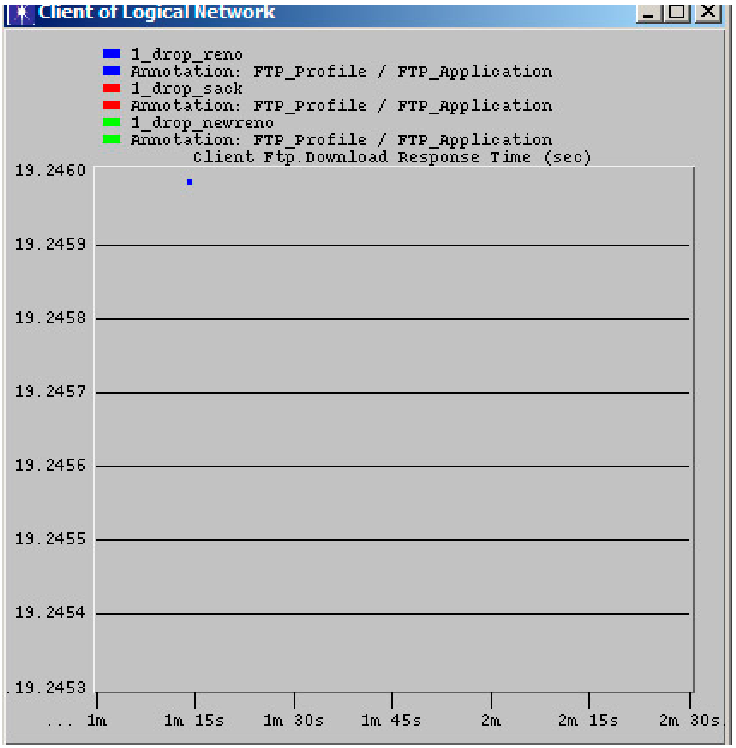


Congestion window (Reno, SACK, NewReno) Download response time

One packet loss

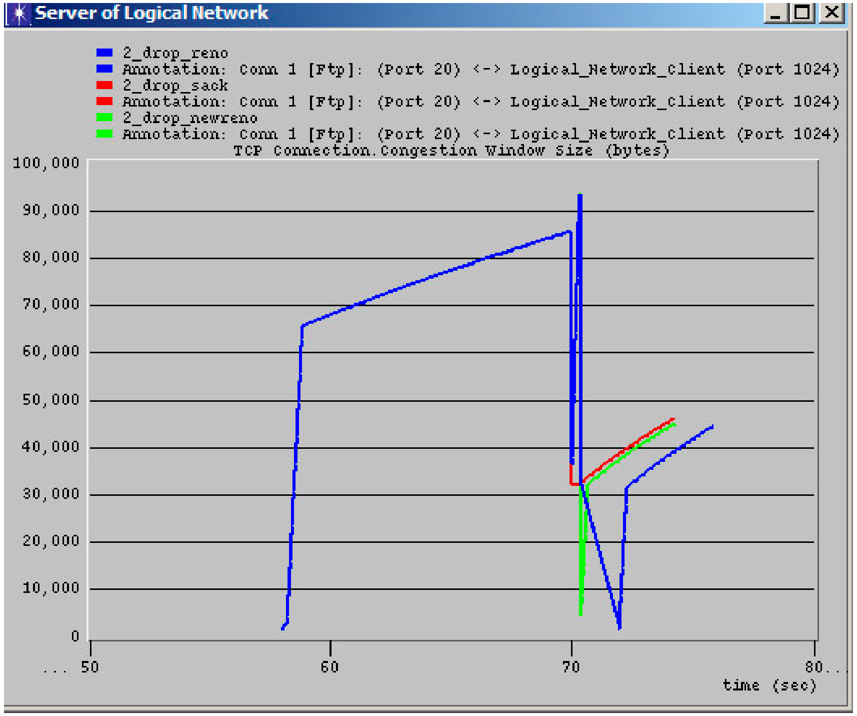


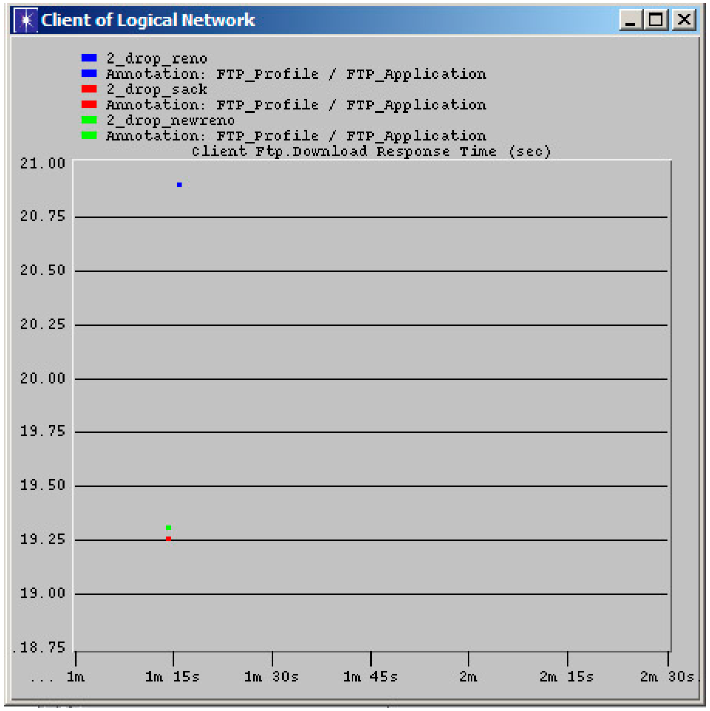
Congestion window (Reno, SACK, NewReno)



Download response time

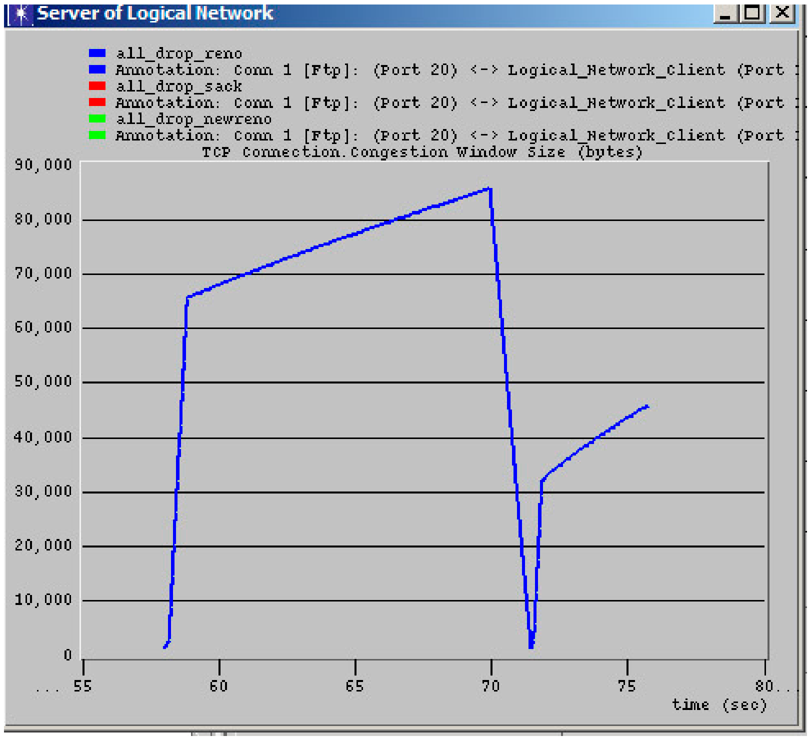
Two packets loss

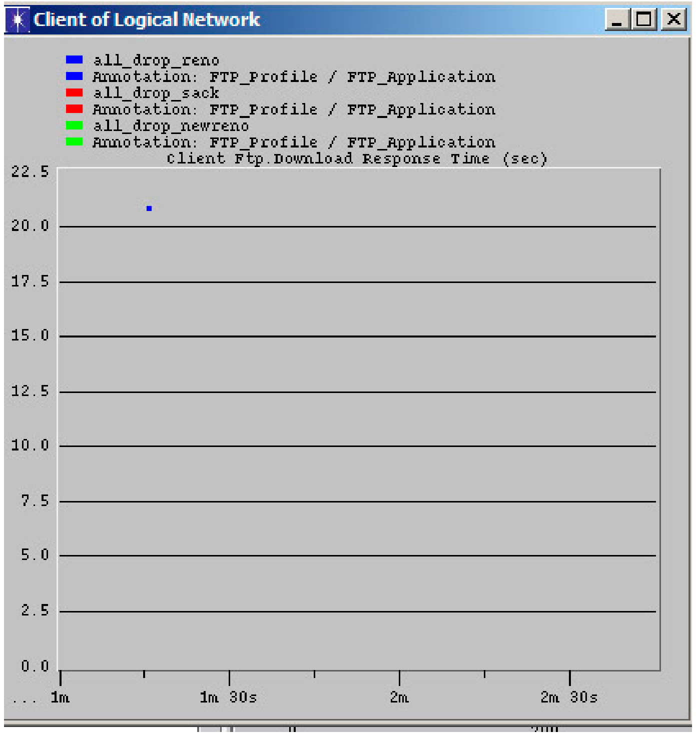




Congestion window (Reno, SACK, NewReno)   Download response time

All packets loss



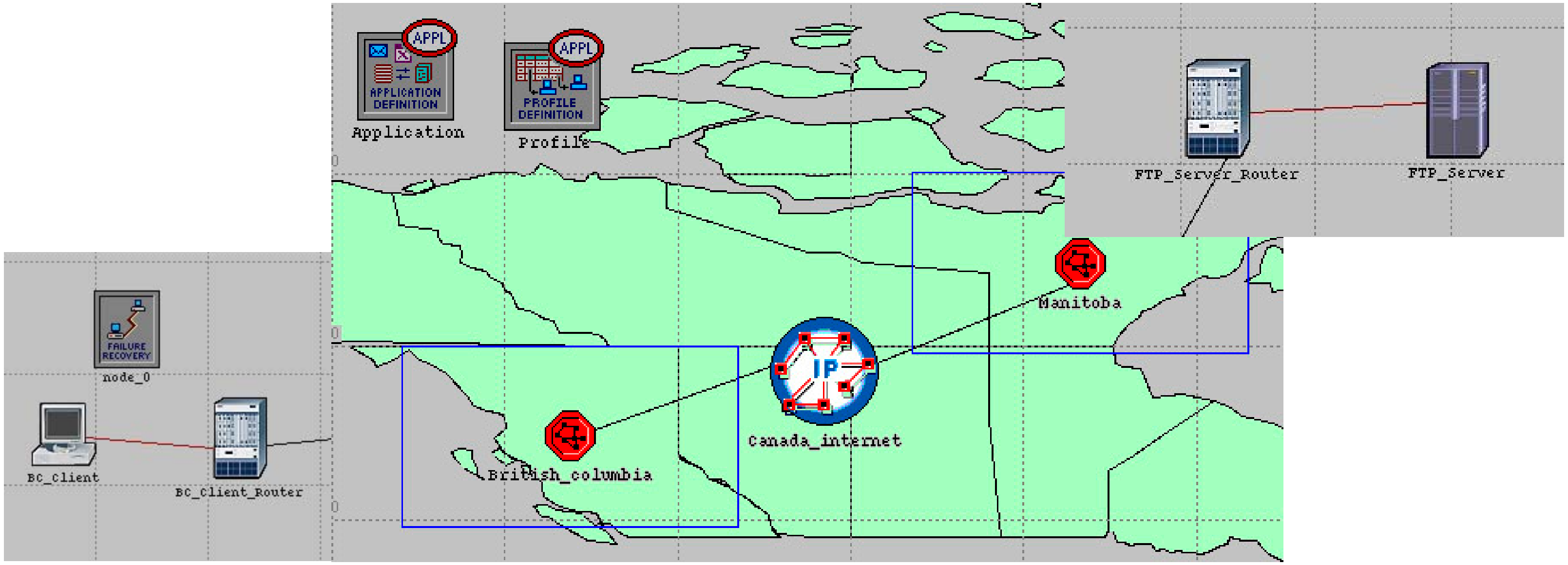


Congestion window (Reno, SACK, NewReno) Download response time

Simulation: Client and Server with disconnection node

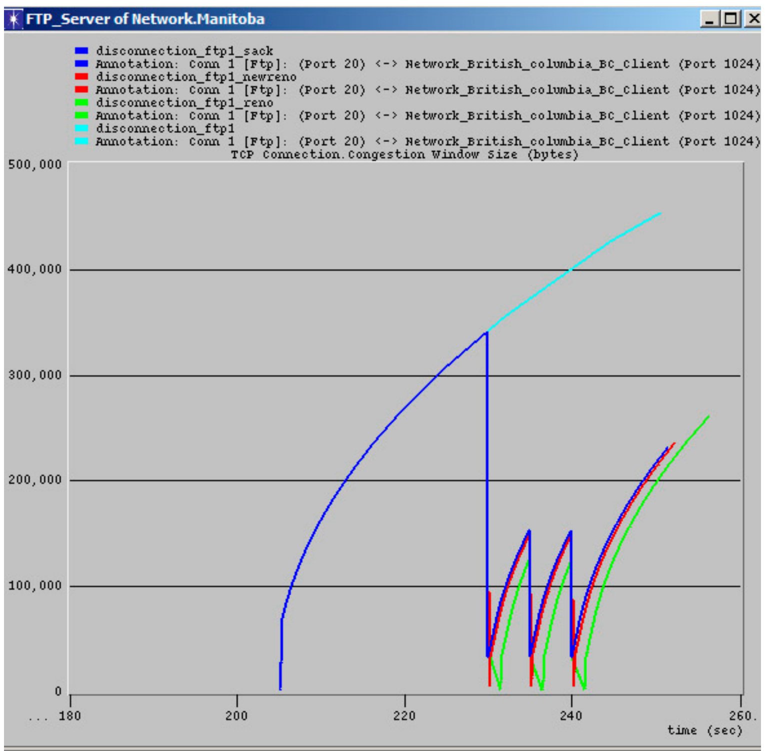
- consist of two subnets: Client and Server in each
- client and server connected to routers: 100 Mbps link
- routers are connected to the Internet cloud: 45 Mbps link
- 150 MB is transferred using ftp application
- link between client and router is disconnected via failure recovery node
- disconnection time intervals:
  - 0.05s, 0.1s, and 0.2s
  - 10s

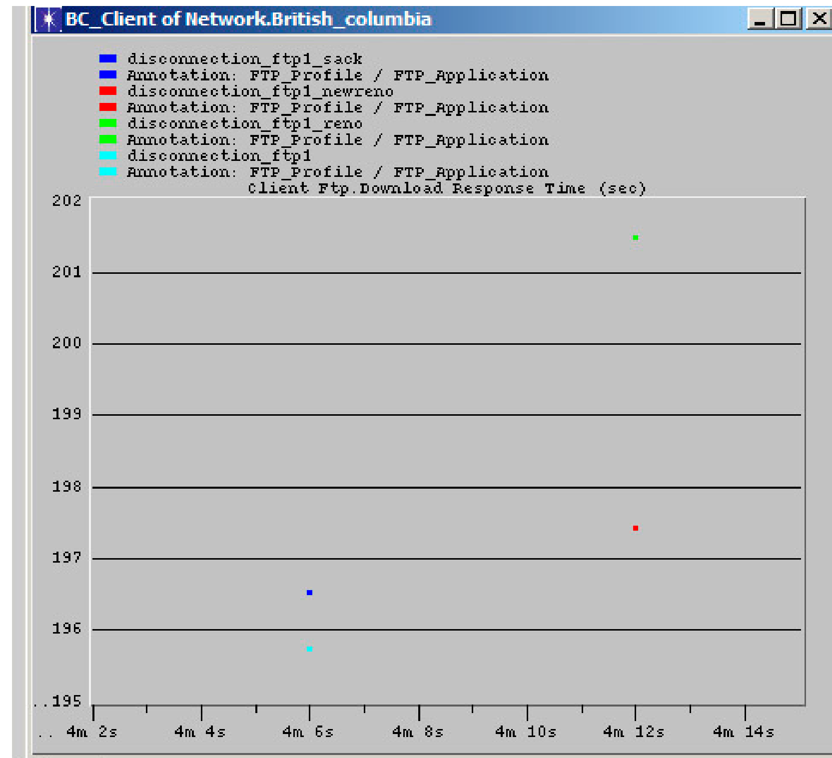
Simulation: Client and Server with disconnection node



Network topology to simulate disconnected network

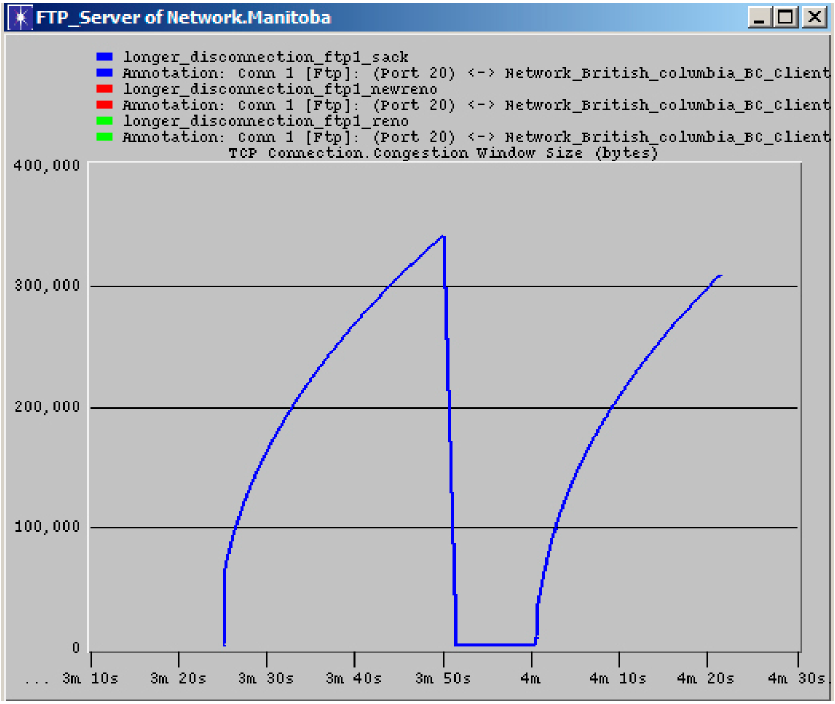
Disconnection for 0.05s, 0.1s, 0.2s

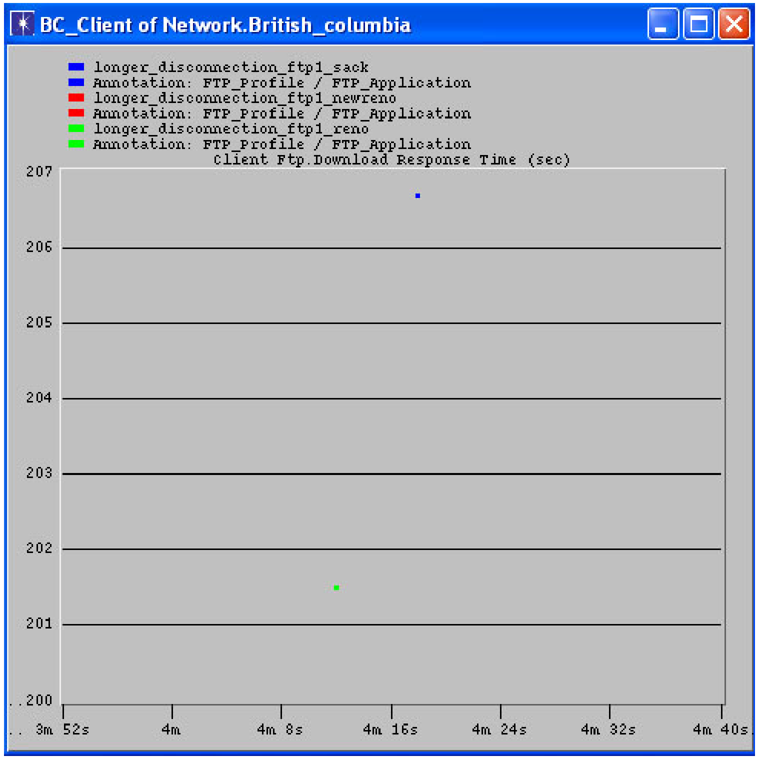




Congestion window (SACK, Reno, NewReno) Download response time

Disconnection for 10s





Congestion window (Reno,SACK, NewReno) Download response time

Simulation: Client and Server with congested network

- consists of multiple subnets: multiple clients and servers
- client and server connected to each subnet router by 100 Mbps link
- 45 Mbps link connects routers to backbone Internet
- database and ftp application are used in server
- profile using server application are run on client
- congested network is simulated

Simulation: Client and Server with congested network

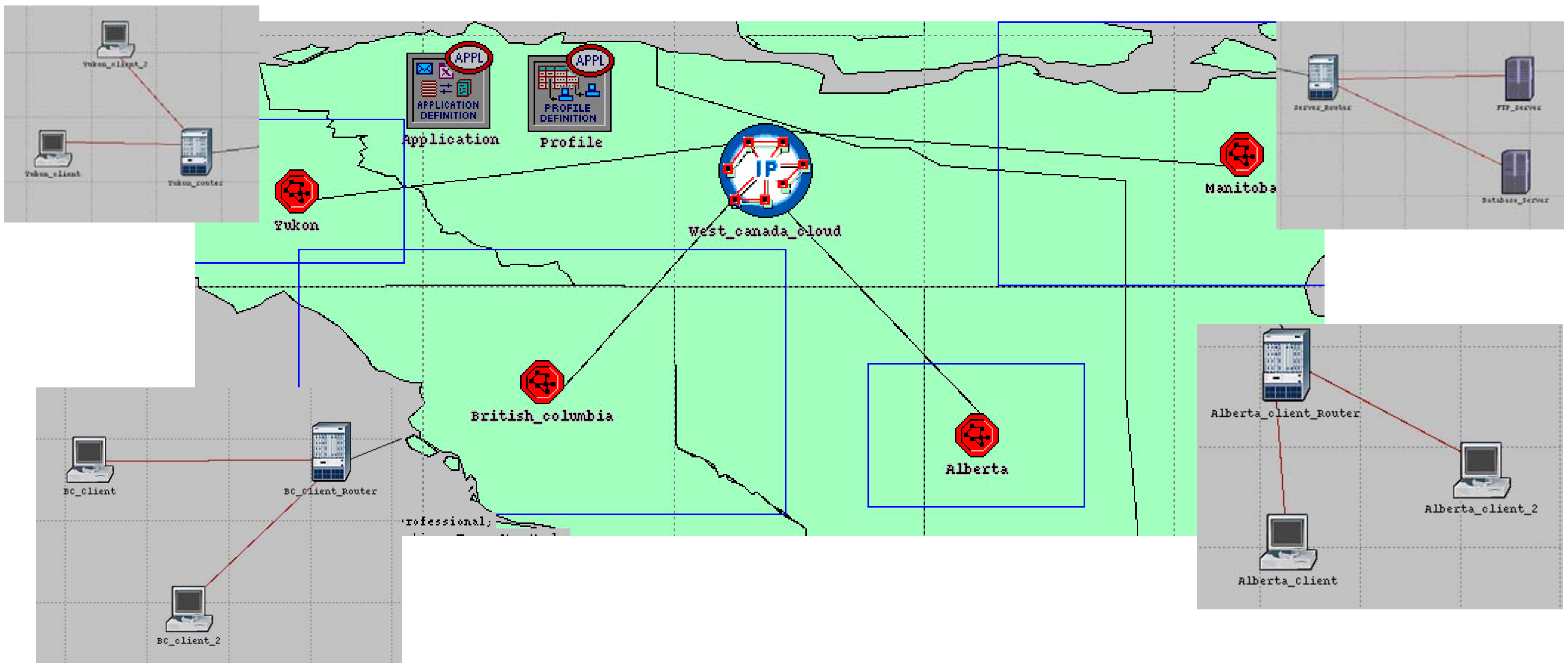
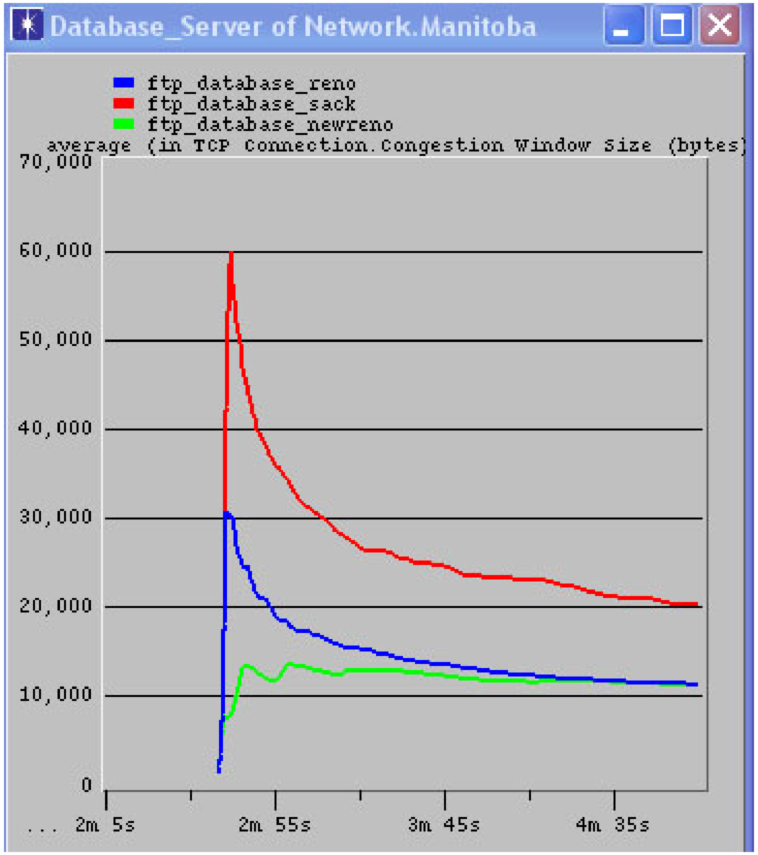


Fig: Network topology used to simulate congested network

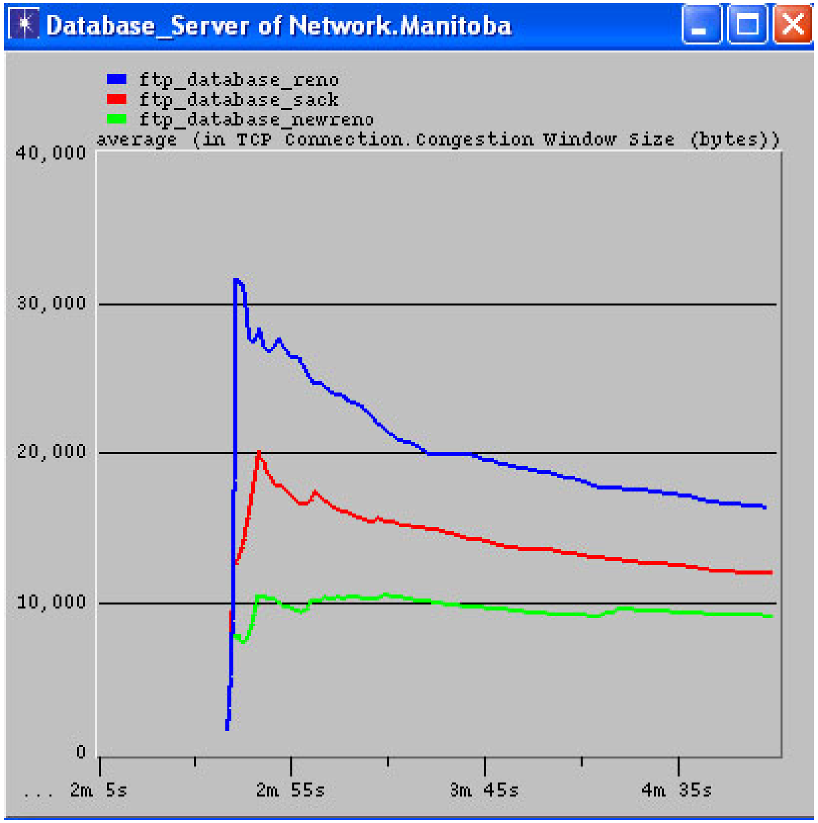
Congestion window at port 1025



Average congestion window at port 1025

Reno, Sack, and New-Reno

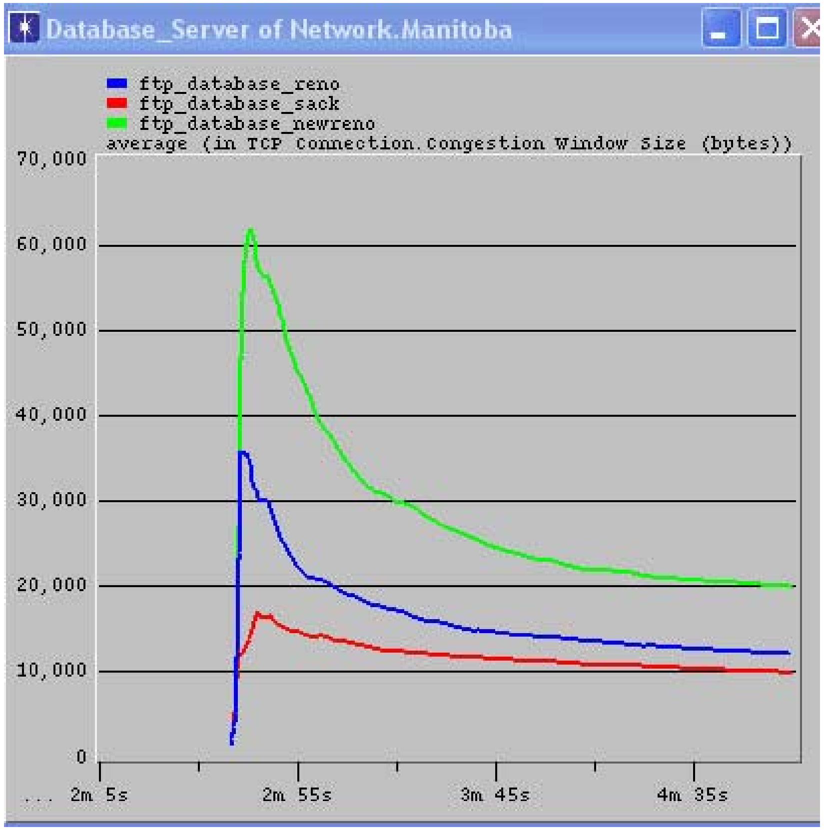
Congestion window at port 1026



Average congestion window at port 1026

Reno, Sack, and New-Reno

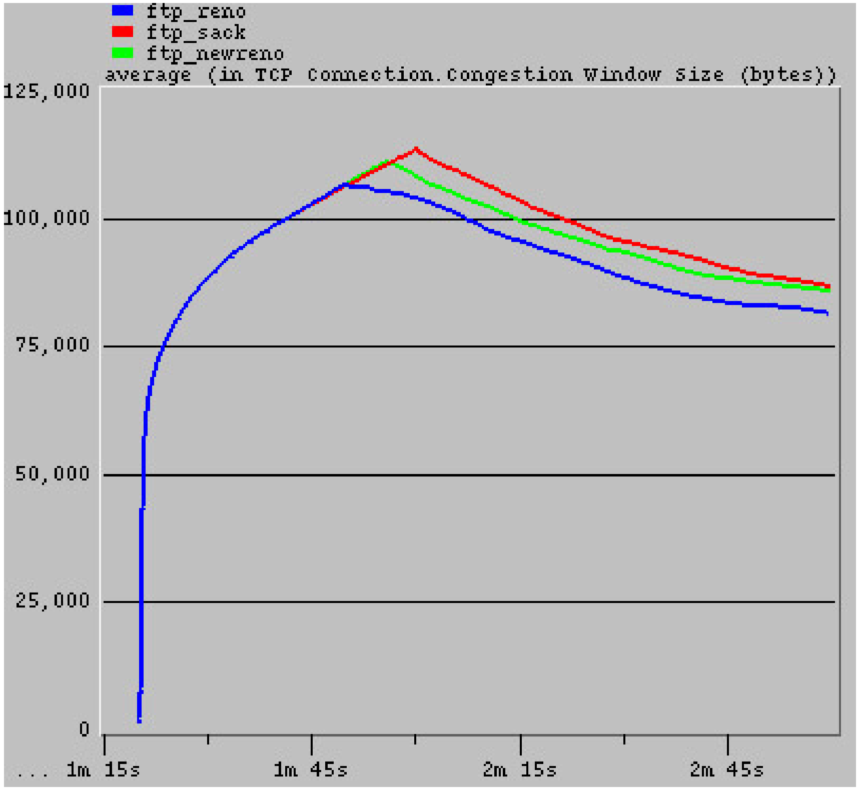
Congestion window at port 1027



Average congestion window at port 1027

Reno, Sack, and New-Reno

Slightly congested network



Average congestion window for low congested network

Reno, Sack, and New-Reno

Conclusion

- Simple client server model:
  - Reno, SACK, and NewReno recovers congestion window similarly in case of all and no packets loss
    - SACK performs best for multiple packets loss
    - Reno performs worst among three algorithms
- Client and server with disconnected node:
  - All algorithms are incapable to distinguish link disconnection

Conclusion

  - window recovery process varies with disconnection interval
  - SACK performs better for short disconnection

- Multiple clients and servers model:
  - conflicting behavior for heavily congestion network
  - SACK performs better in slightly congested network

Future Work

- model wireless node with disconnection behavior for detailed analysis
- analyze variation in drop time for more further performance analysis
- investigate to identify the cause of conflicting behavior of Reno, NewReno, and SACK algorithms for heavily congested network
- implement new algorithms and compare their performance with these basic algorithms

References

- [1] S. Floyd and K. Fall, "Simulation Based Comparisons of Tahoe, Reno and Sack TCP", ACM Computer Communication Review, 1996, Vol.26, No.3: 5–21 [1]
- [2] S.Floyd and T.Henderson "The New-Reno Modification to TCP's Fast Recovery Algorithm" RFC 2582, Apr 1999. [2]
- [3] R. Paul and Lj. Trajkovic,"Selective-TCP for wired/wireless networks," Proc. SPECTS 2006, Calgary, AL, Canada, Aug. 2006, pp. 339-346. [3]
- [4] W. G. Zeng and Lj. Trajkovic,"TCP packet control for wireless networks," Proc. IEEE Int. Conf. on Wireless and Mobile Computing, Networking and Communications (WiMob 2005), Montreal, Canada, Aug. 2005, pp. 196-203, vol. 2. [4]
- [5] Z. Chen and M.M. Ali, "The performance of TCP congestion control algorithm over high-speed transmission links," IEEE Canadian Conf., Department of Electrical and Computer Engineering, Concordia University, Montreal, Canada, May 2004, pp.1371 - 1374 Vol.3. [5]

References

- [6] G. Holland and N. Vaidya,"Analysis of TCP performance over mobile ad hoc networks," Proc. ACM/IEEE Int. Conf. on Mobile Computing, Networking,, Seattle, Washington, United States, 1999, pp. 219-230.[6]
- [7] F. Anjum and L. Tassiulas, "Comparative study of various TCP versions over a wireless link with correlated losses**",** *IEEE/ACM Transactions on Networking (TON)*, NJ, USA, June 2003, Vol. 11, Issue 3, pp. 370 – 383. [7]
- [8] H. lee, S. Lee and, Y.Choi, "The influence of the large bandwidth-delay product on TCP Reno, NewReno, and SACK", *Proc. Information Networking Conference*, Oita, Japan, 2001, pp. 327–334. [8]
- [9] RFC [online]. Available: http://www.ietf.org/rfc.html.