QCN Transience and Equilibrium: Response and Stability

Abdul Kabbani, Rong Pan,
Balaji Prabhakar and Mick Seaman

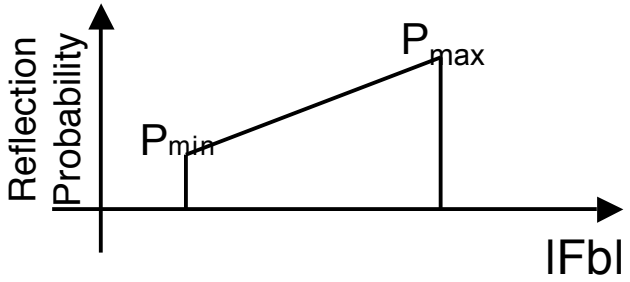
Outline of presentation

- 2-QCN
  - Overview and method for improving transient response
- Equilibrium and Scalability
  - Large number of sources and/or large RTTs
- Conclusions

Basic QCN

- 2-point architecture: Reaction Point -- Congestion Point
  1. **Congestion Points:** Sample packets, compute feedback (Fb), quantize Fb to 6 bits, and reflect only *negative* Fb values to Reaction Point with a probability proportional to Fb.

$$Fb = -(q_{off} + w\, q_{delta})$$
$$= -(\text{queue offset} + w.\text{rate offset})$$



  2. **Reaction Points:** Transmit regular Ethernet frames. When congestion message arrives: perform multiplicative decrease, fast recovery and active increase.
     - Fast recovery similar to BIC-TCP: gives high performance in high bandwidth-delay product networks, while being very simple.

Other Features

- Extra fast recovery
  - Useful for coping with bursty drops, matches offered rate to available capacity
- Drift: Timer-based, gentle rate increase
  - For failsafe operation, improves fairness by preferentially drifting low rate sources
- Fb-hat: Congestion estimation at the source
  - Allows a source to recover available bandwidth quicker
  - Will go over this today

Grabbing Extra Bandwidth: An Algorithm

- Estimate congestion at the source
  - Maintain an estimate of Fb, say Fb-hat, at each RL
  - Fb-hat is counted using a 5-bit saturation counter
  - Fb-hat is thought of as a source's estimate of congestion

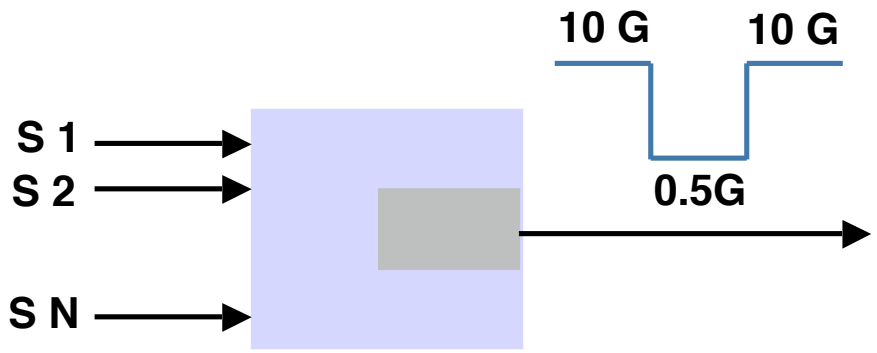
- Updating Fb-hat
  - For every Fb recd by RL: Fb-hat <-- Fb-hat + Fb
  - For every 50 pkts transmitted: Fb-hat <-- Fb-hat/2 (just right shift)

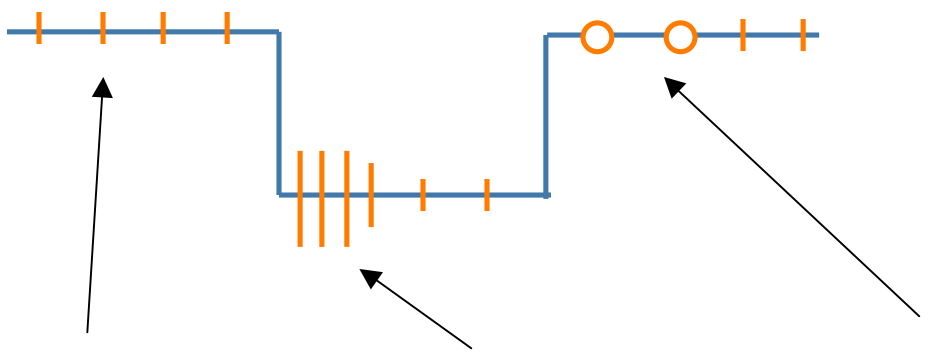
- Using Fb-hat: cycle-shrinking
  - Every time we begin a cycle of FR or AI…
    - If Fb-hat is small (e.g. 0 or 1): reduce length of cycle to 50 pkts from 100 pkts

- Idea: small Fb-hat implies no dings for a while, hence it is likely there is no congestion; so a source can quickly get to AI and grab extra bdwdth
  - Note: in equilibrium, Fb-hat will be more than 1, hence no cycle-shrinking should occur, hence stability is preserved

A Pictorial View

- Consider output-generated hotspot on a 10 G link





Small, well-spaced Fb values
Fb-hat is small

Large, closely-spaced Fb values
Fb-hat is large

Fb=0 values (not signaled)
Fb-hat goes to zero

A Principle

- Introducing Fb-hat symmetrizes the source and switch behavior
  - Switch
    - Has input: Packets or source rates
    - Observes: Qoff, Qdelta
    - Goal: Drive Q to Qeq and Qdelta to zero
    - Action taken to achieve goal: Send Fb signals to sources
  - RL
    - Has input: Fb signals from network
    - Observes: Fb-hat
    - Goal: Drive Fb-hat close to zero (i.e. just above 1, the threshold)
    - Action taken: Change transmission rate

- This is a primal-dual algorithm for congestion management
  - Primal variable, source rate: Input to switch but output from RL
  - Dual variable, Fb: Input to RL but output from switch

Distributed Control

- Primal algorithm: The control laws situated at the source, switch has static functions.
- Dual algorithm: Control laws situated at switch, source has static functions.
- QCN without Fb-hat already primal-dual, Fb-hat makes it more so.

- A principle of distributed control: The switch and source (or RL) pass just the right signals to each other so as to solve the global bandwidth partitioning problem in a distributed fashion

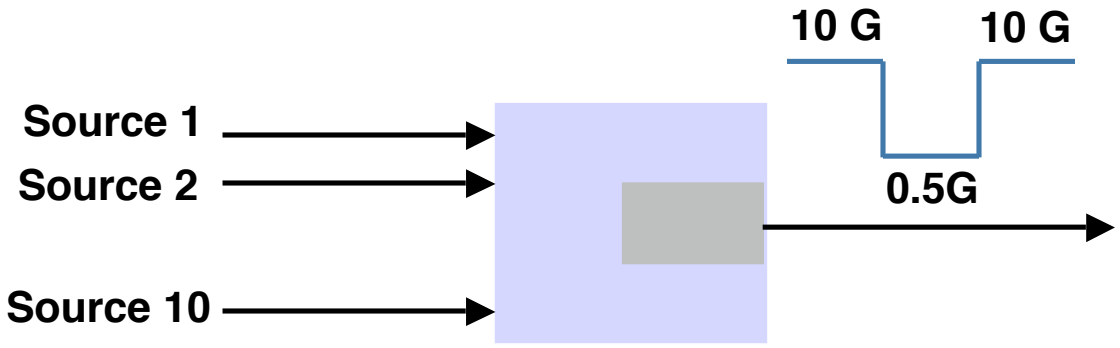
- Clearly, other algorithms can be obtained from this principle; e.g. we have tried
  1. Cycle lengths of 25, 50 and 100 pkts depending on Fb-hat values
  2. Stretching cycle lengths to 150 pkts if Fb-hat is large
  3. Letting Fb-hat go negative; this lets source know with more certainty that bdwdth is available

- These improve the transient response further
  - But they introduce a complexity--performance trade-off

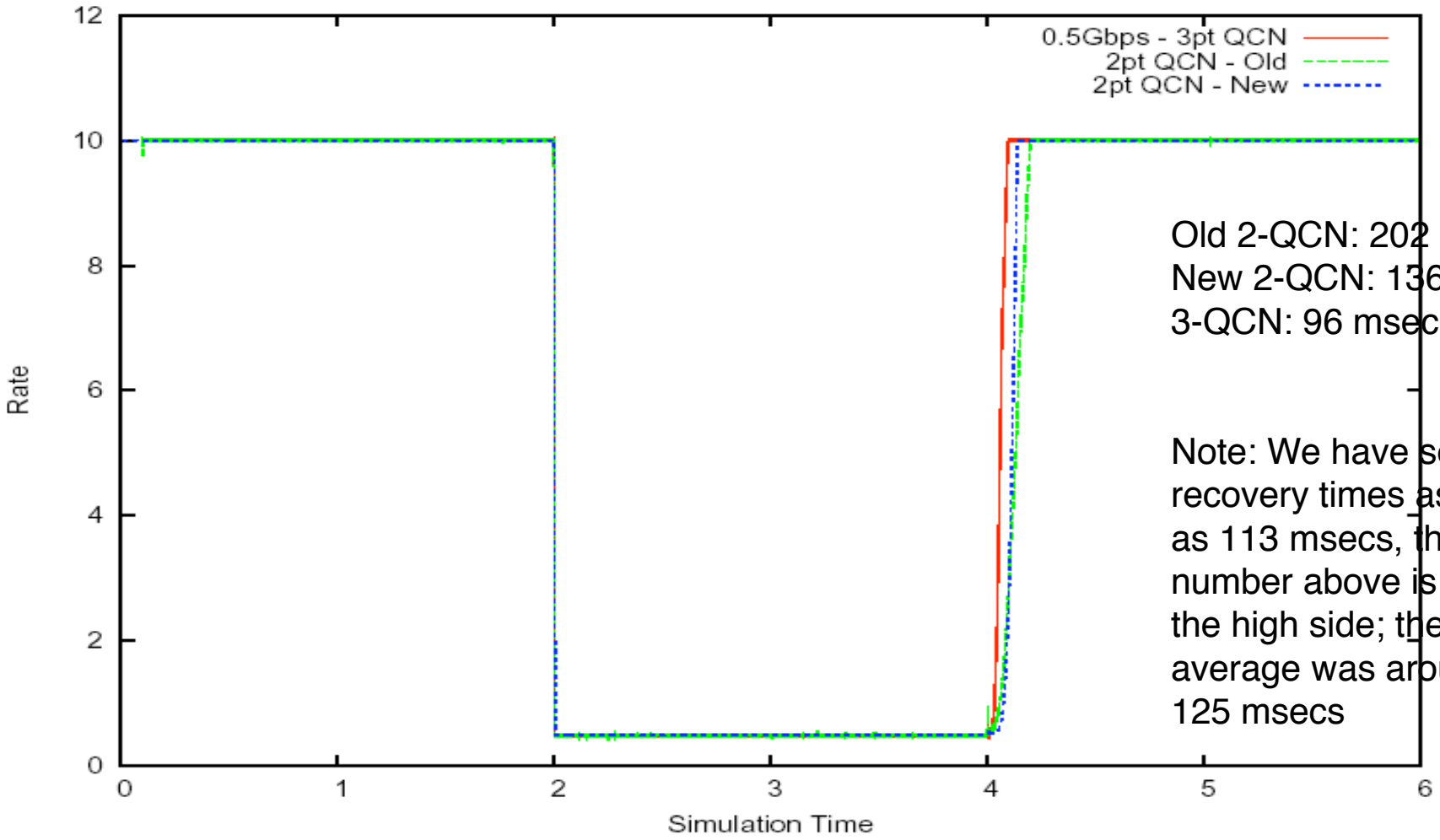
Improvement in Transience

Simulations: OG Hotspot

- Parameters
  - 10 sources share a 10 G link, whose capacity drops to 0.5G during 2-4 secs
  - Max offered rate per source: 1.05G
  - RTT = 40 microseconds
  - Buffer size = 100 pkts; Qeq = 22
  - Fb-hat saturated at 31
  - FR cycle-shrinking: 50 pkts if Fb-hat is 0 or 1, 100 pkts otherwise
  - AI: also 50 or 100 pkts depending on Fb-hat as above
  - AI amount: 25 Mbps



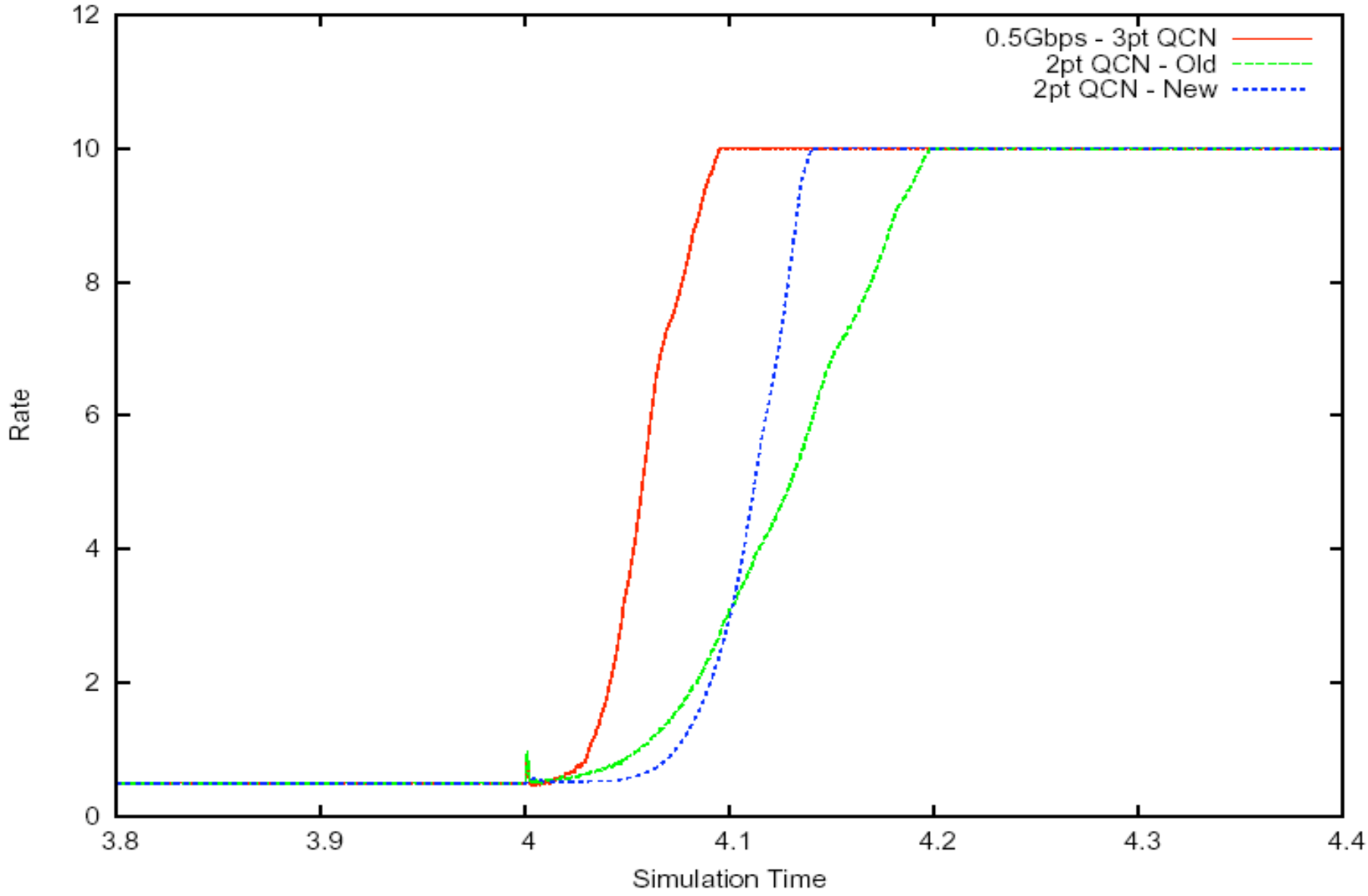
0.5G Bottleneck: Rate



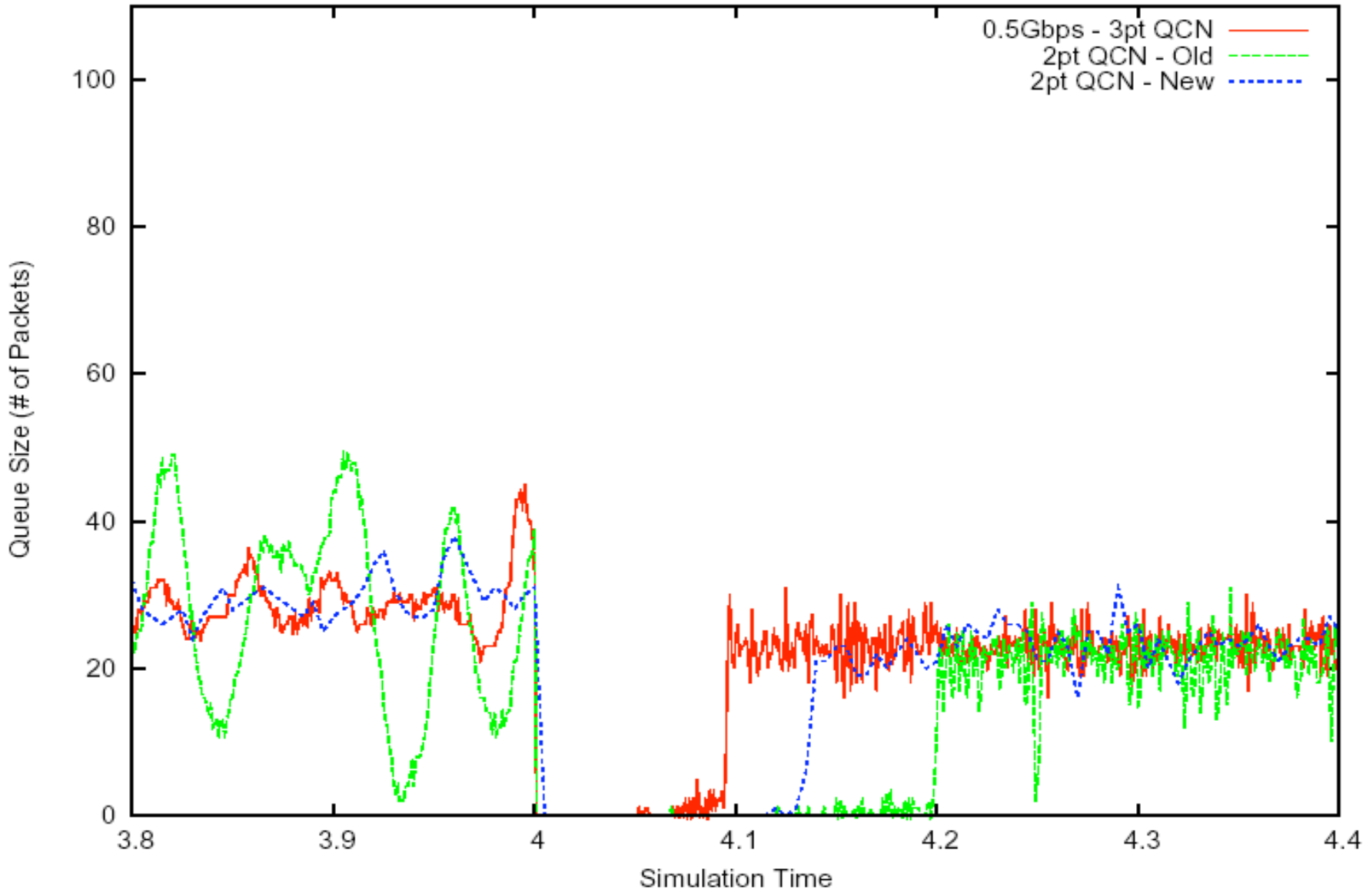
Old 2-QCN: 202 msec
New 2-QCN: 136 msec
3-QCN: 96 msec

Note: We have seen recovery times as low as 113 msecs, the number above is on the high side; the average was around 125 msecs

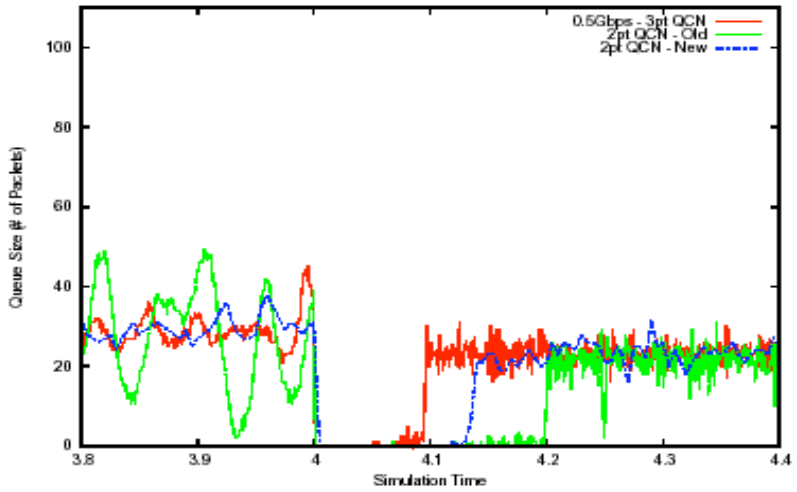
0.5G Bottleneck: Rate, Zoomed-in

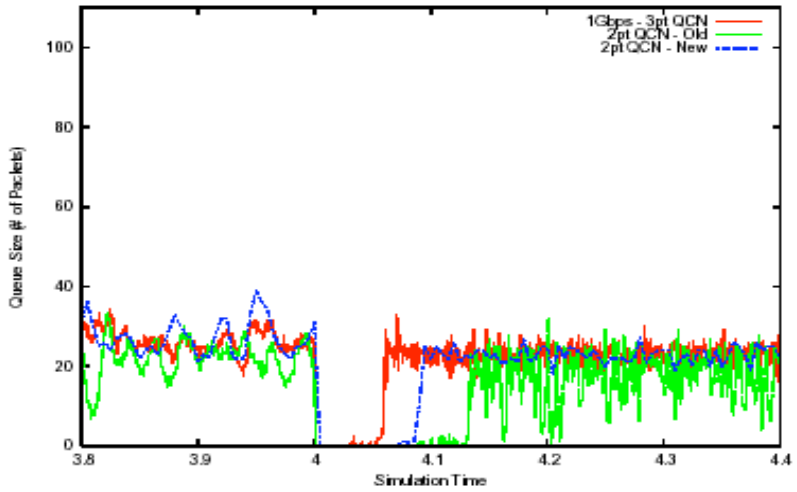


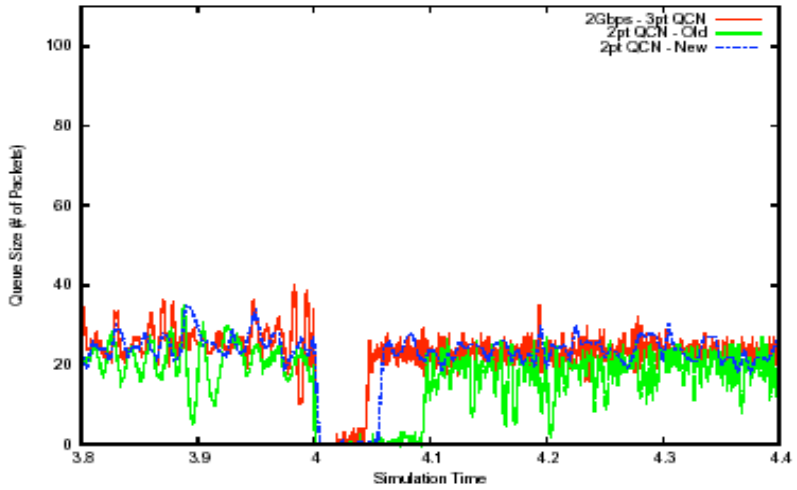
0.5G Bottleneck: Queue, Zoomed-in



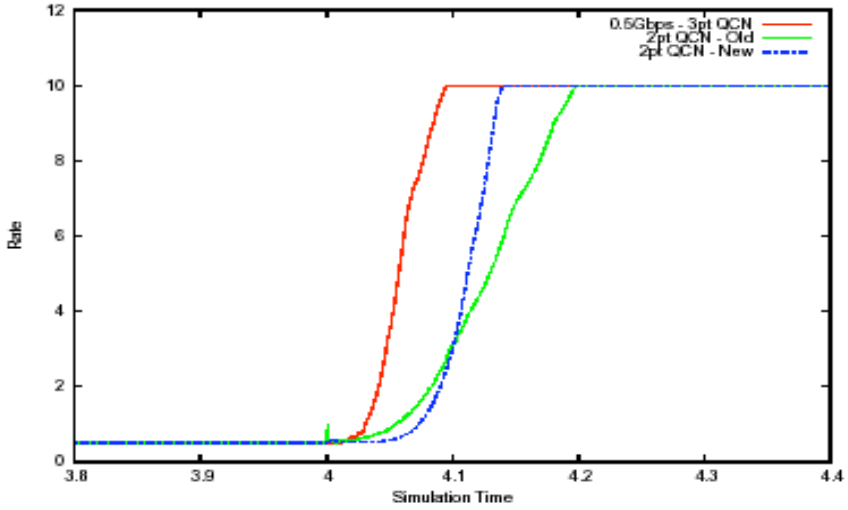
0.5G, 1G, 2G Bottleneck: Queue, Zoomed-in

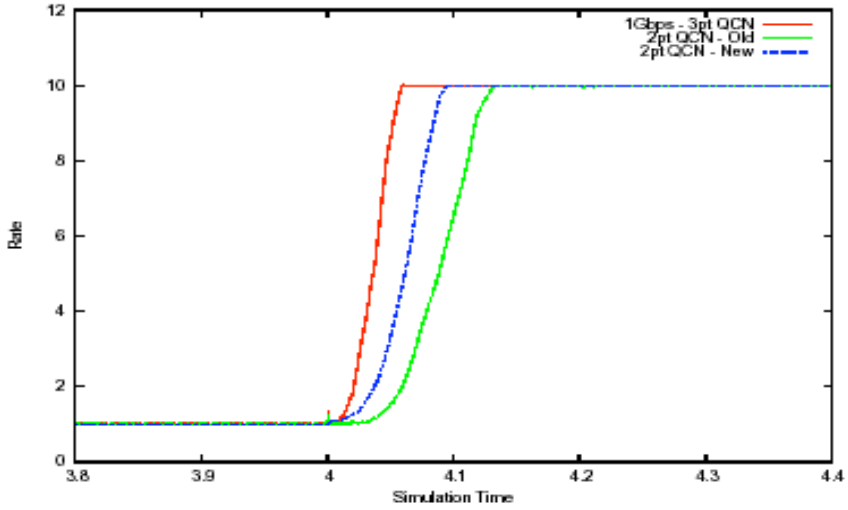


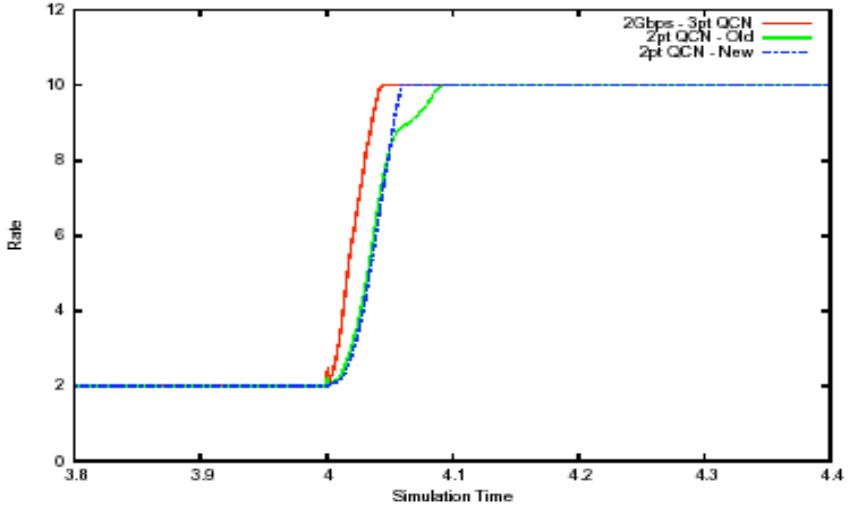




0.5G, 1G, 2G Bottleneck: Rate, Zoom







Conclusions for Transience

- Seen a method for the source to monitor congestion and quickly grab available bandwidth
  - The algorithm we have used is the simplest possible
  - Enhanced versions (different cycle lengths, Fb-hat negative, hyperactive increase) certainly improve the recovery time; in fact, one can match or beat 3-QCN.
  - However, it seems better to choose a simple version because it is adequate, esp in real deployments where flows will arrive and depart.
  - Need to draw a line in trade-off space.

Scalability, Stability

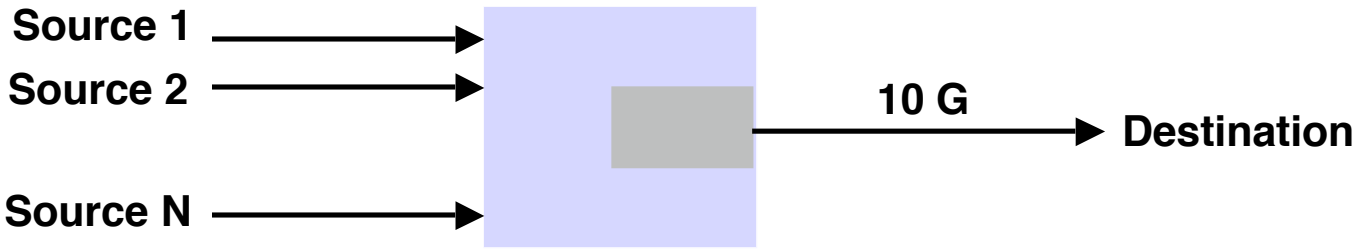
Scalability Analysis

- This refers to understanding the behavior of QCN when N, the number of sources, and the RTT get very large; main question
  - How does stability depends on parameters?

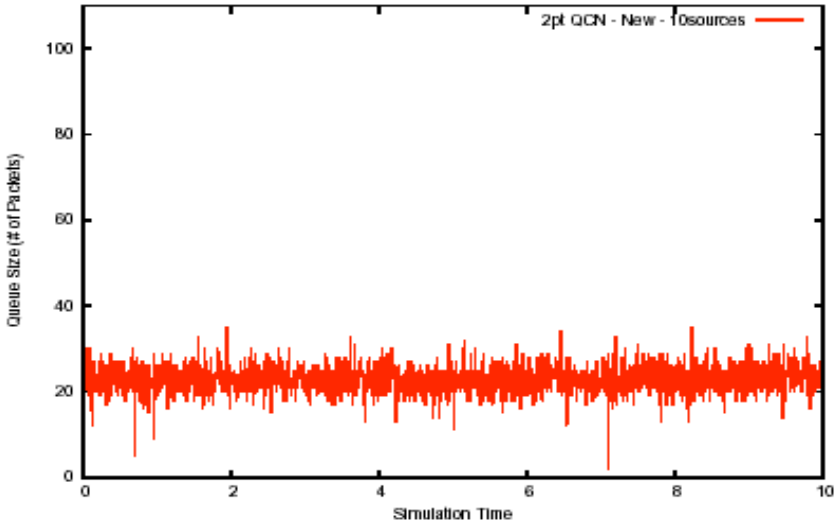
- Ultimately this is addressed through a theoretical model. We use a Markov chain based model, different from the standard linearized analysis using differential equations .
  - The Markov model contains more information, because it is the source of the differential equation model. The differential equations describe the *mean* behavior of the system; i.e. average rates, not instantaneous; whereas the Markov model gives the complete stochastic description.
  - Linearization around the operating point gives *local* stability.
  - Finally, the Markov model can be used to model both equilibrium and transience.
  - We are developing the theory; for now, we use simulations to observe stability wrt large numbers of sources and RTT.

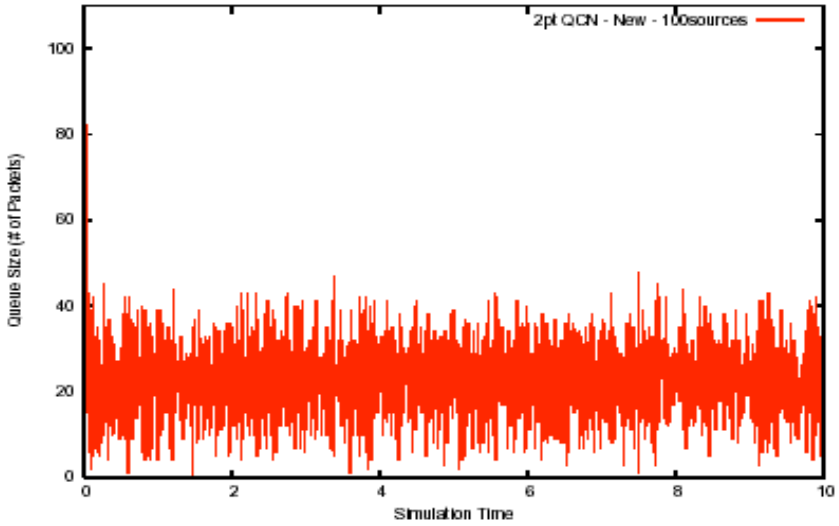
Simulations

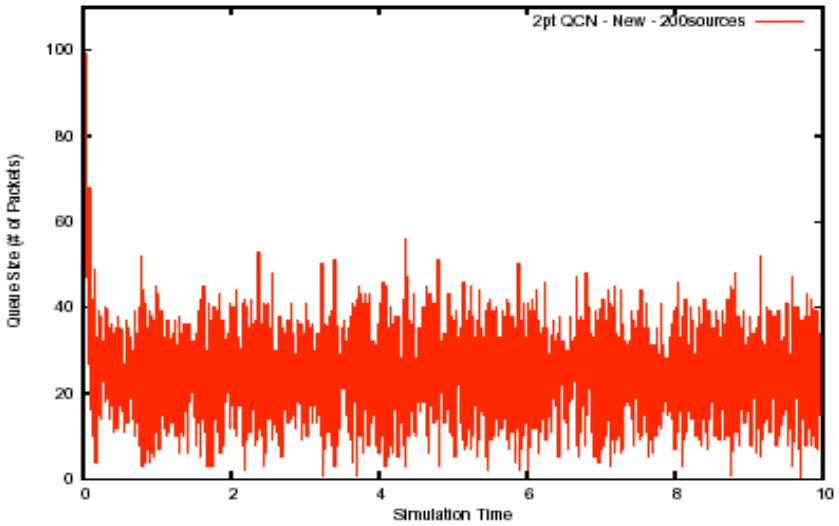
- Parameters
  - N sources share a 10 G link
  - Starting rate of each source = 100/N G
  - RTT = 40, 100, 200, 300, 400 microseconds
  - Buffer size = 100 pkts; Qeq = 22
  - Fb-hat saturated at 31
  - FR cycle-shrinking: 50 pkts if Fb-hat is 0 or 1, 100 pkts otherwise
  - AI: also 50 or 100 pkts depending on Fb-hat as above
  - AI amount: 25 Mbps

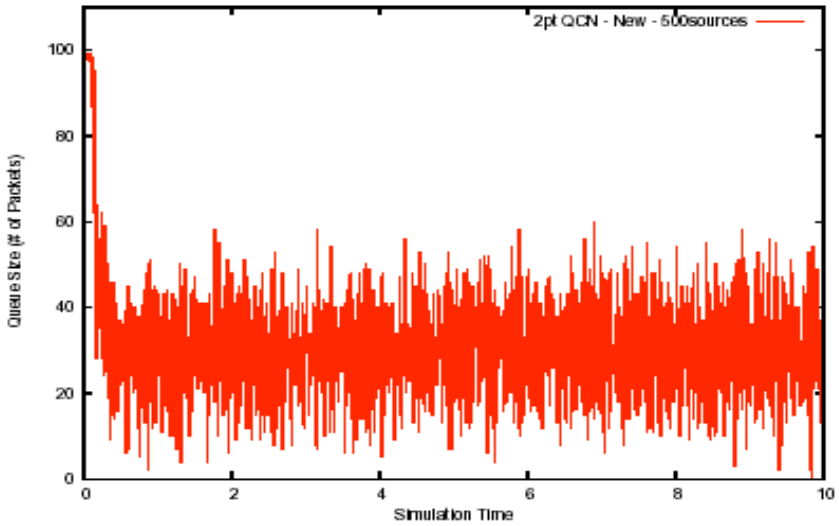


Varying # of Sources: Queue Size
RTT = 40 usecs

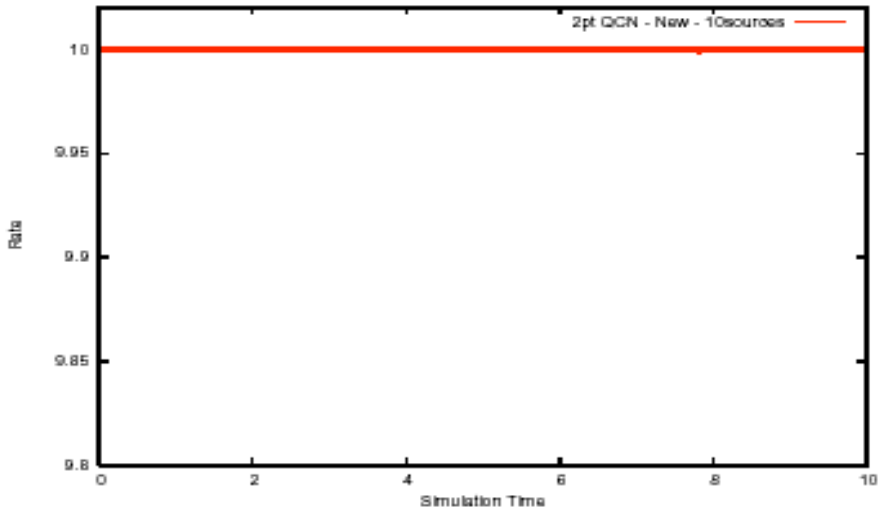


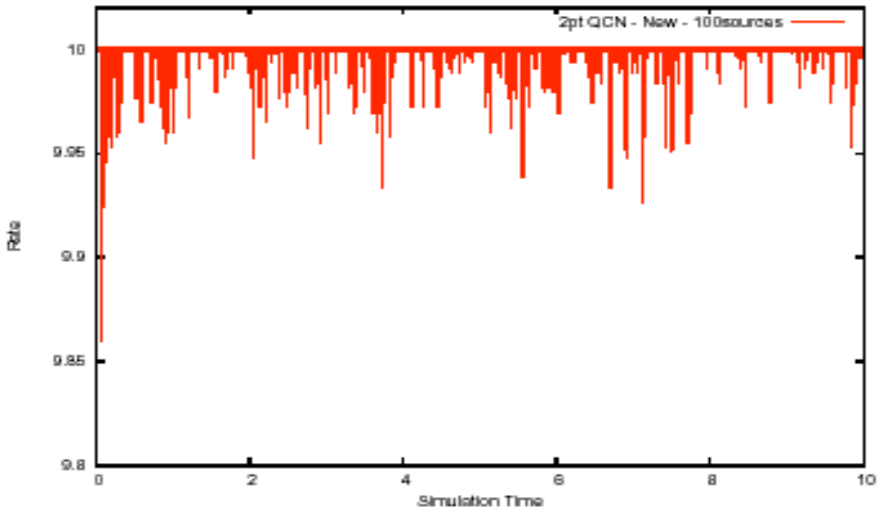


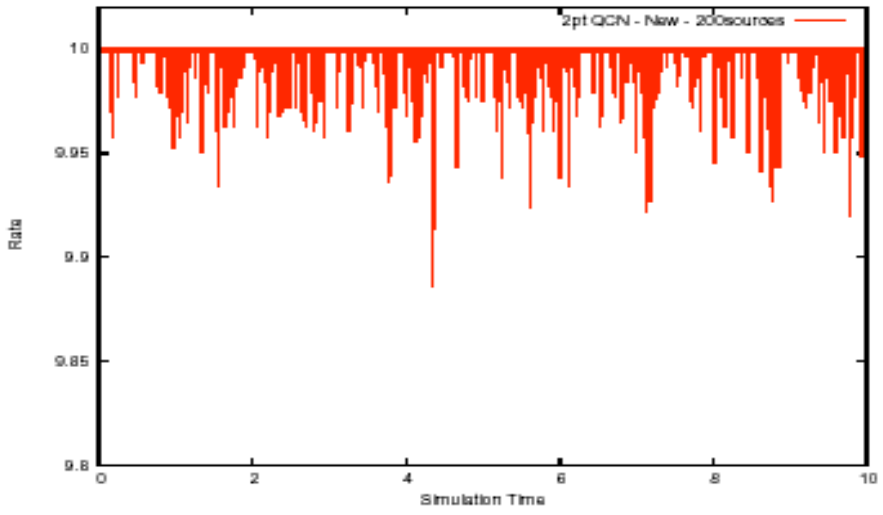


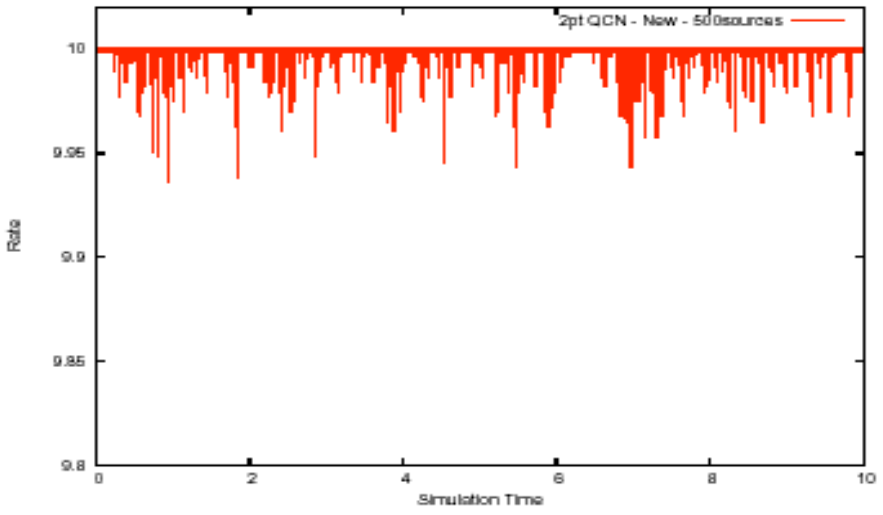


Varying # of Sources: Rate
RTT = 40 usecs

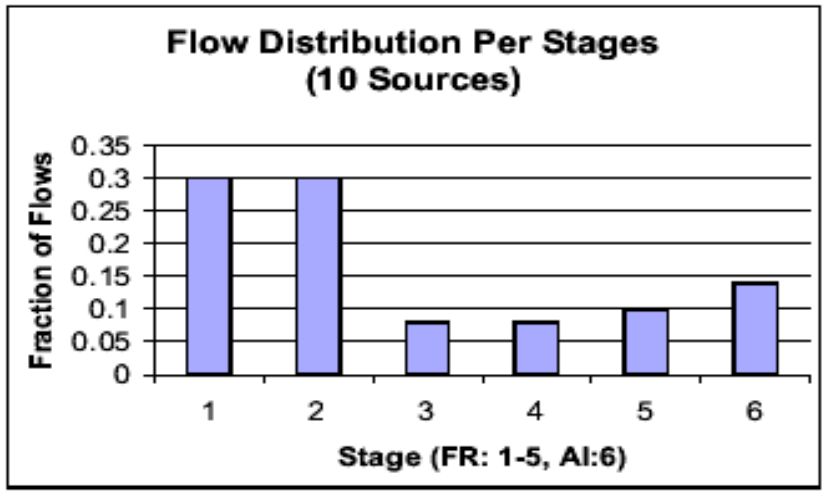


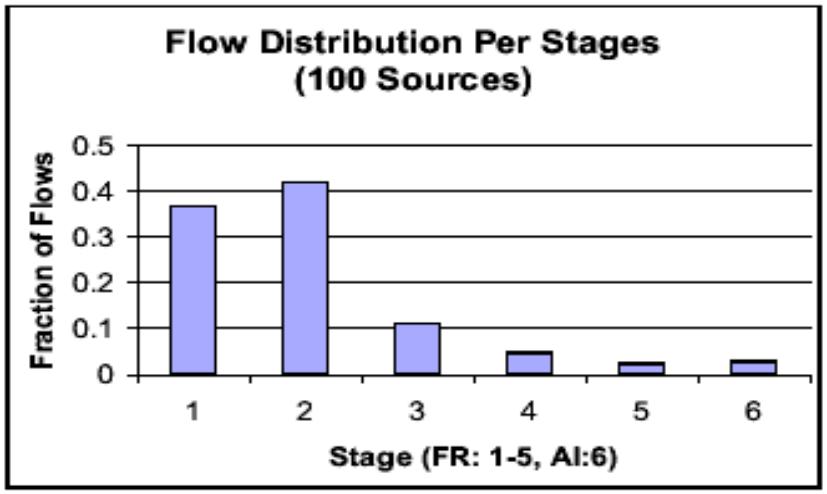


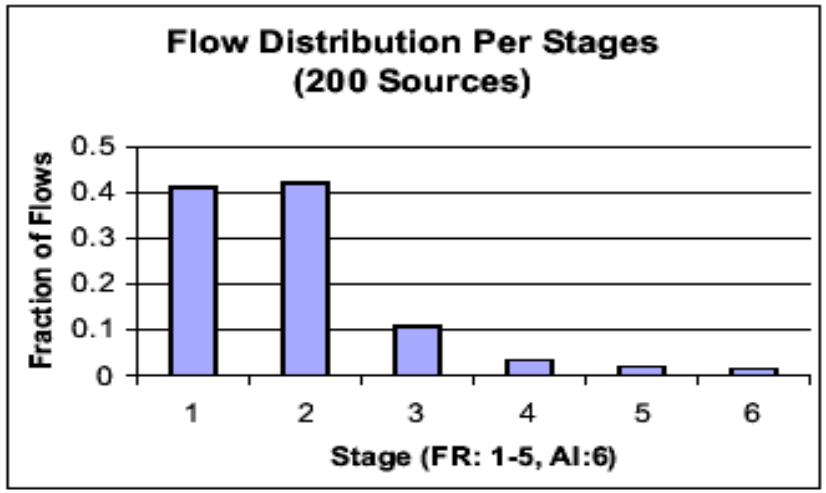


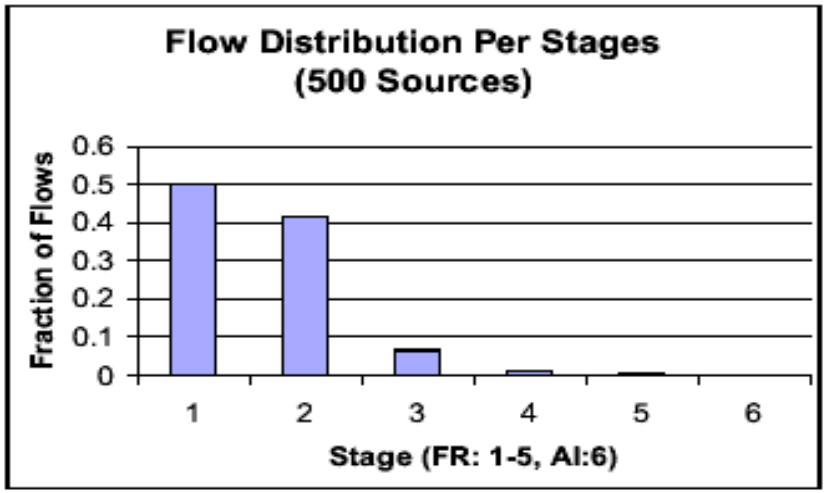


QCN: Distribution of flows in FR, AI

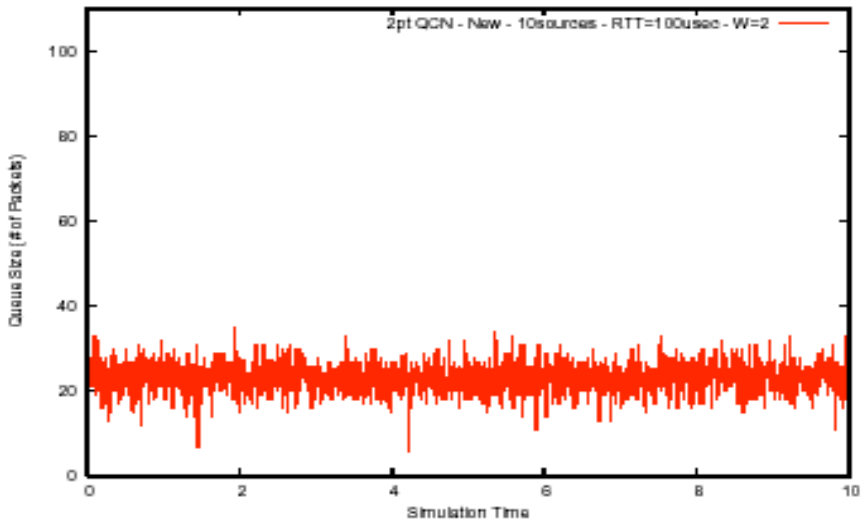


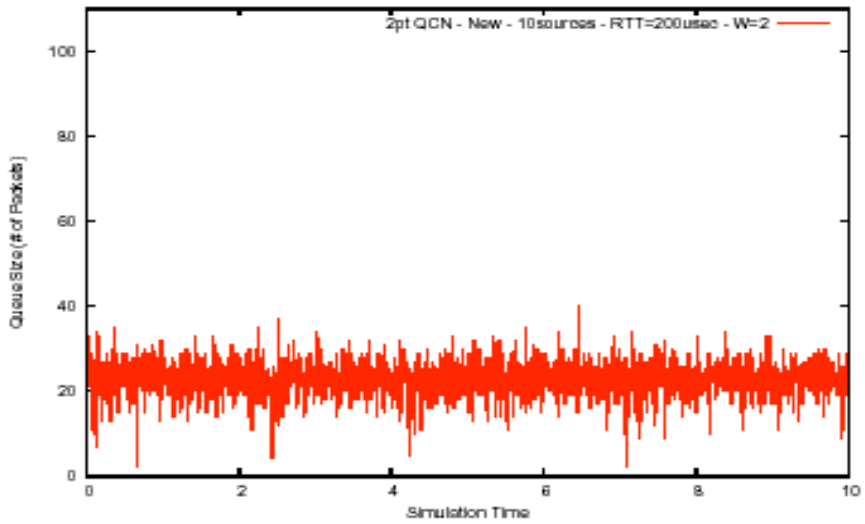


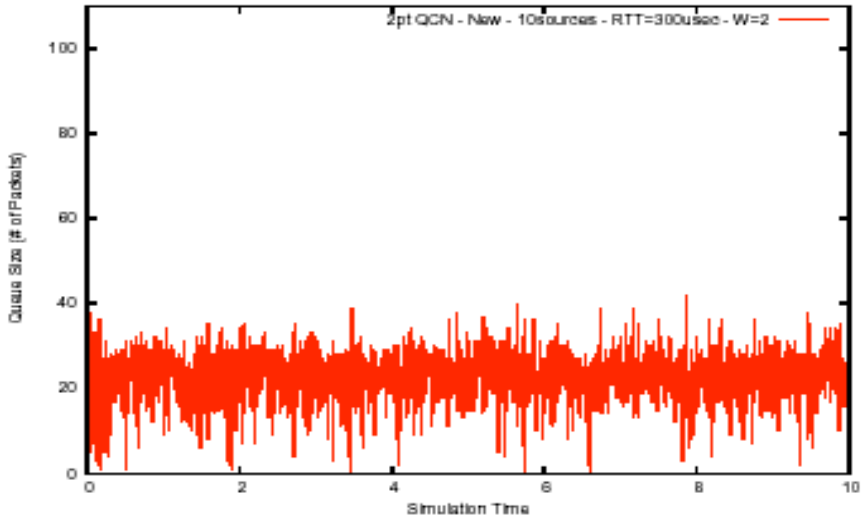


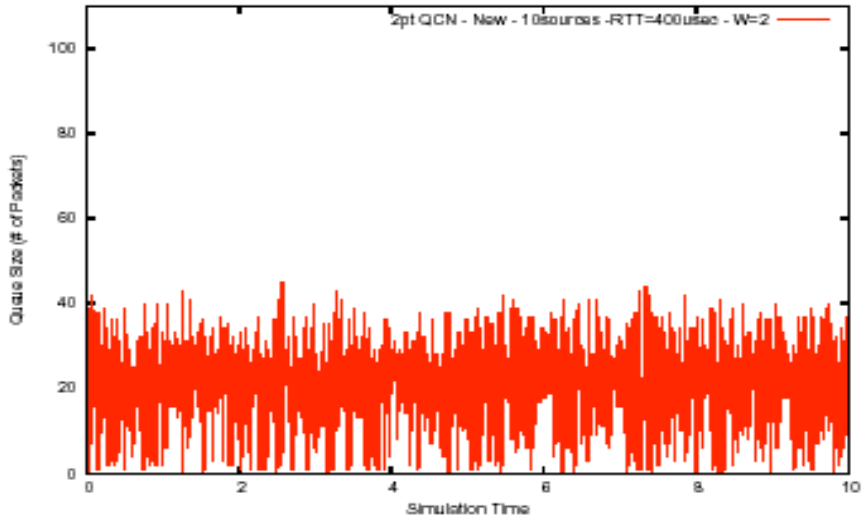


Varying RTT: Queue Size
Number of Sources = 10

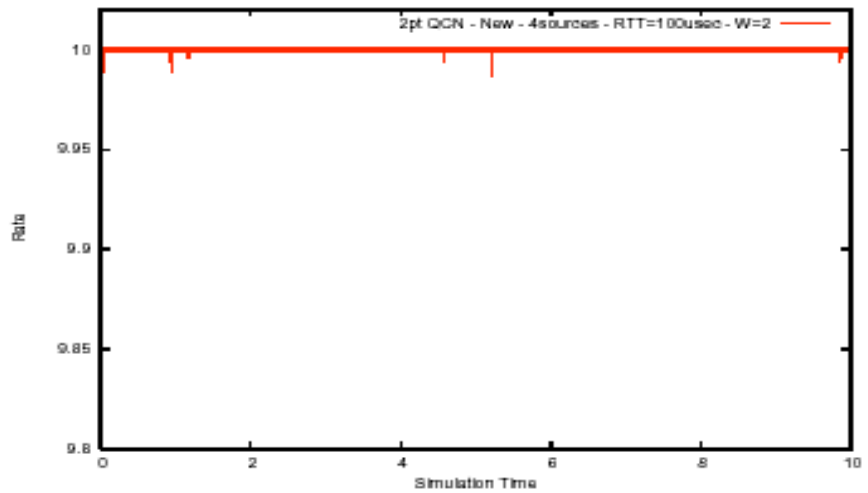


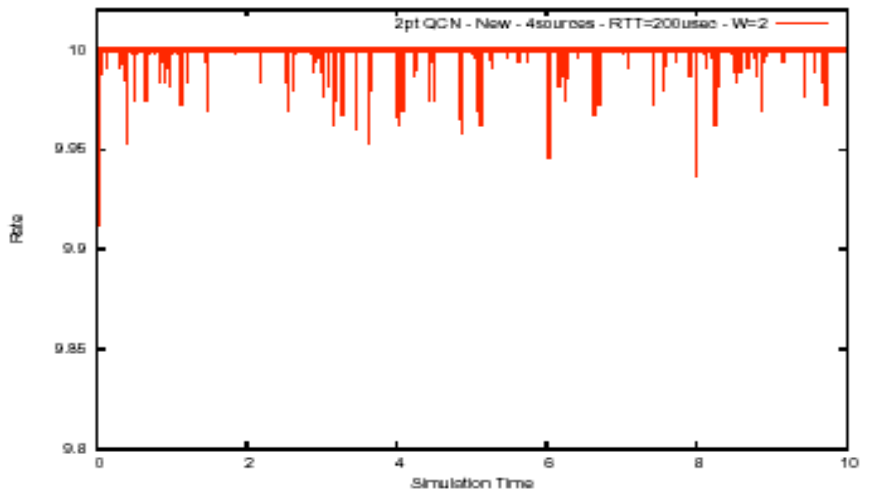


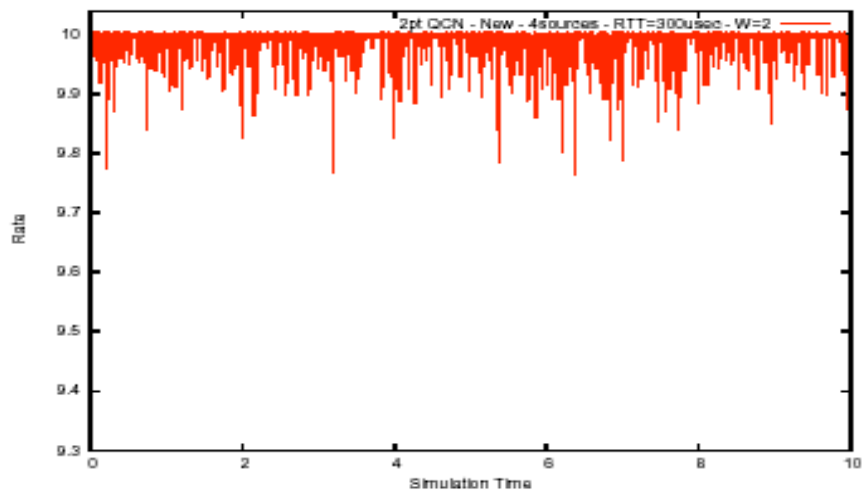


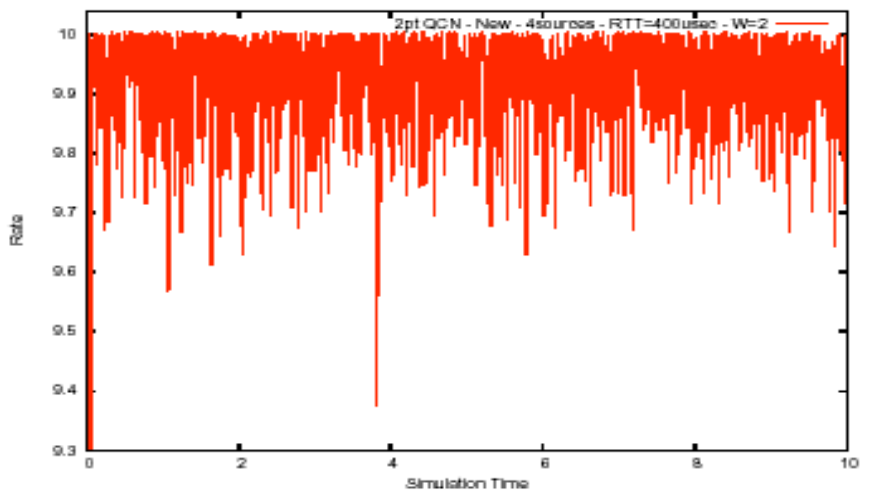


Varying RTT: Rate
Number of Sources = 10

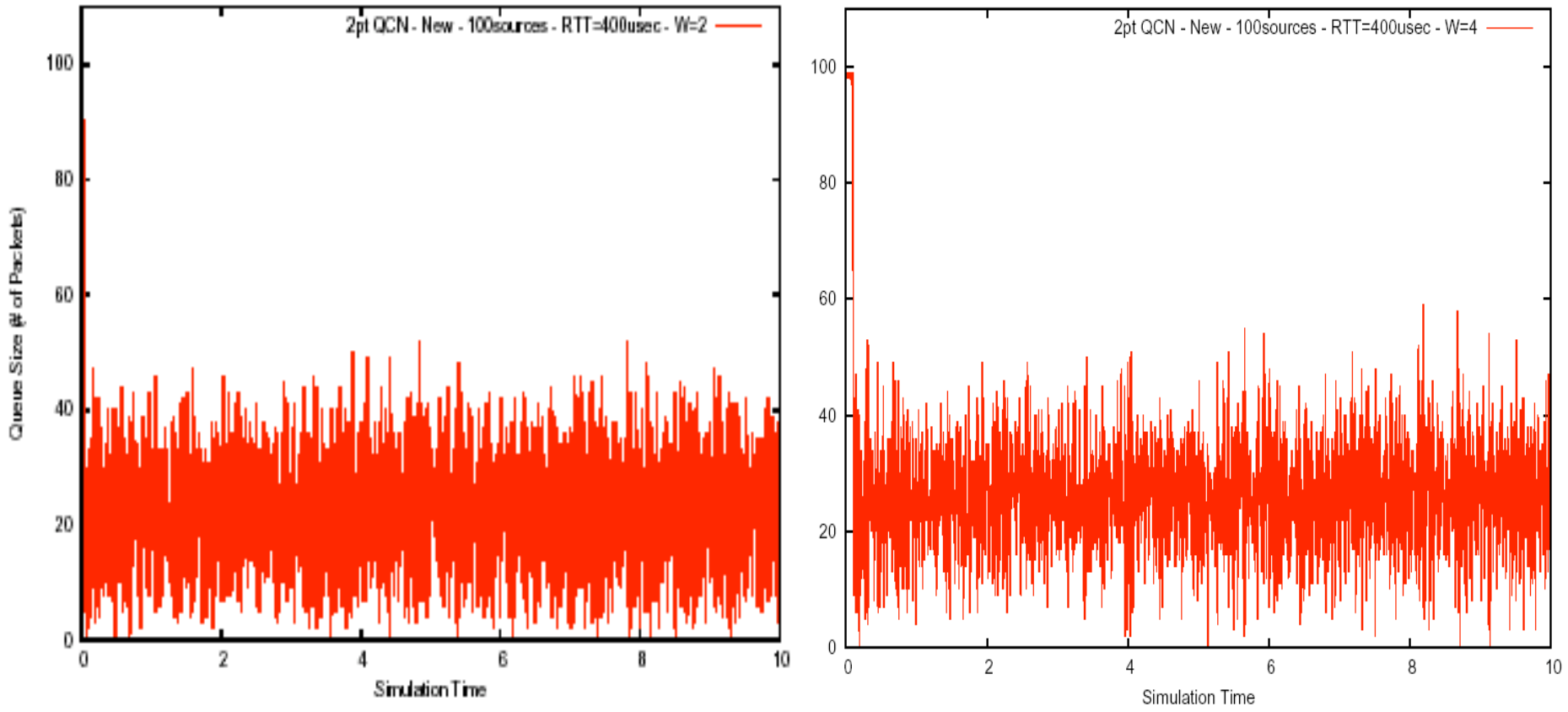






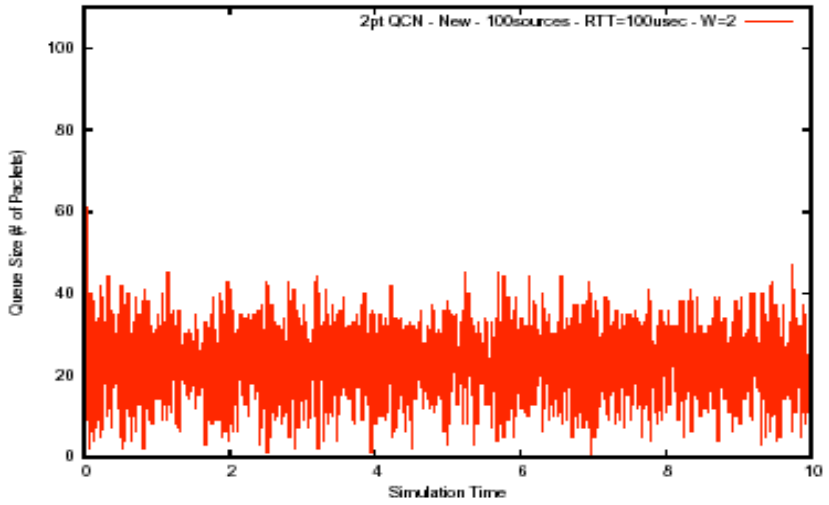


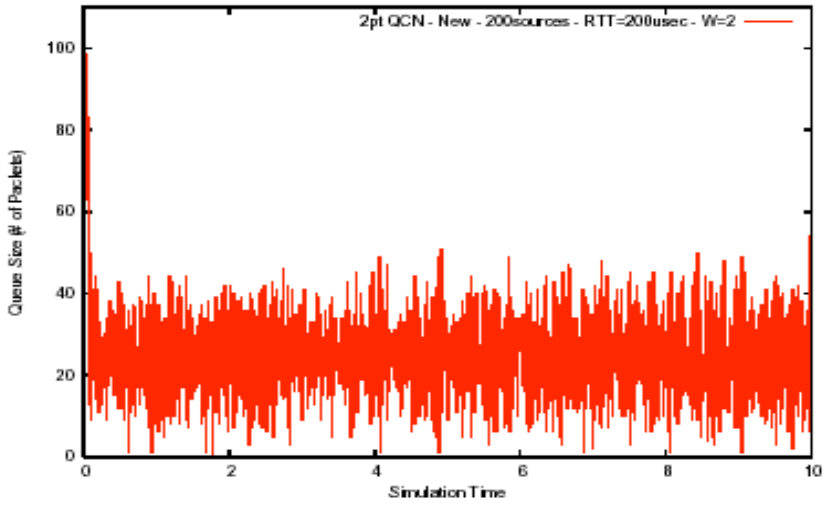
100 sources, RTT = 400 usecs w=2 vs w=4

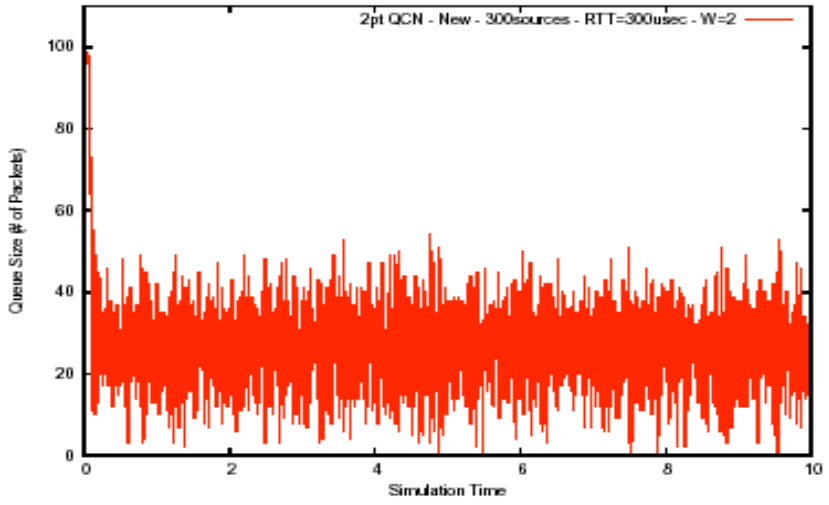


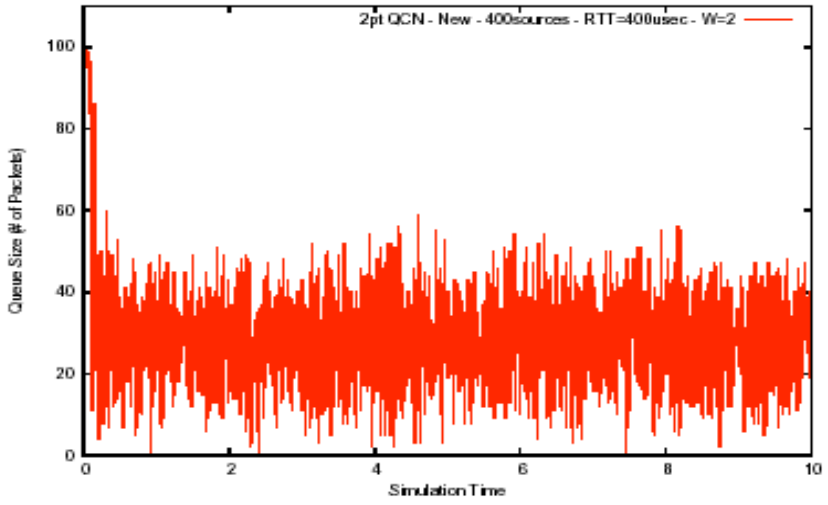
- Since Fb = q_{off} + wq_{delta}, increasing w weights the derivative more and hence leads to less wiggly queues. But because of the springiness of binary increase, even w = 2 performs well at large RTTs.

Varying RTT and # of Sources: Queue Size

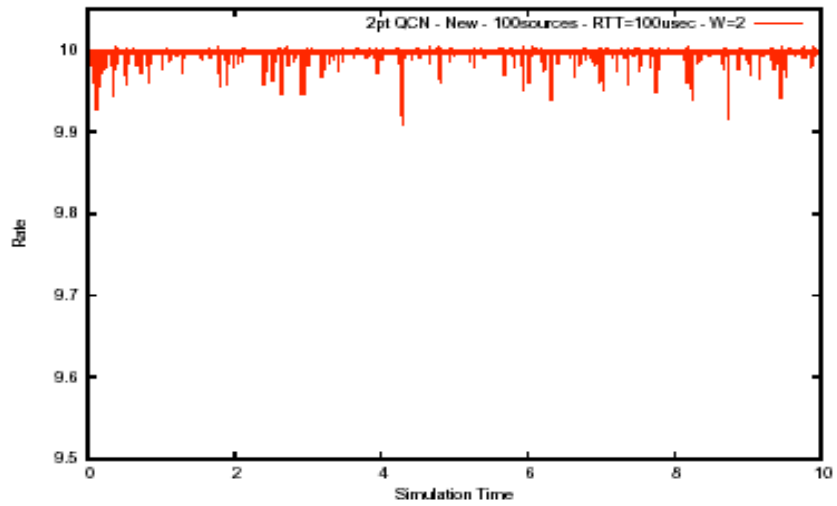


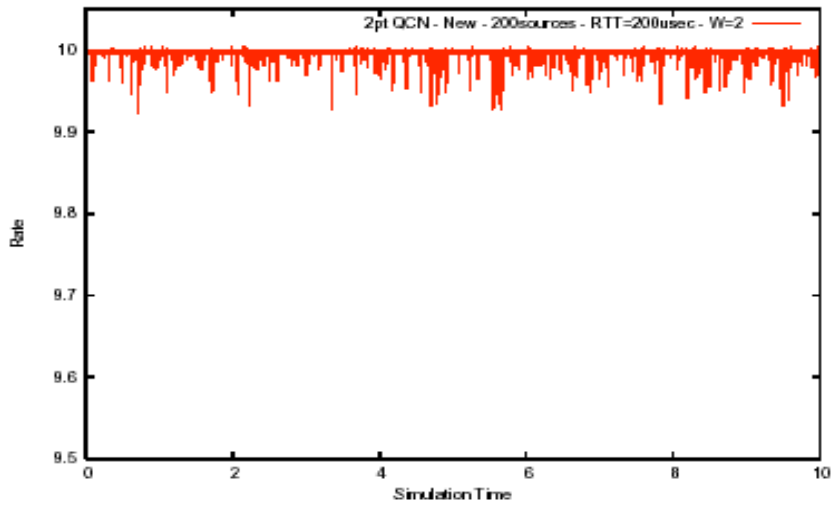


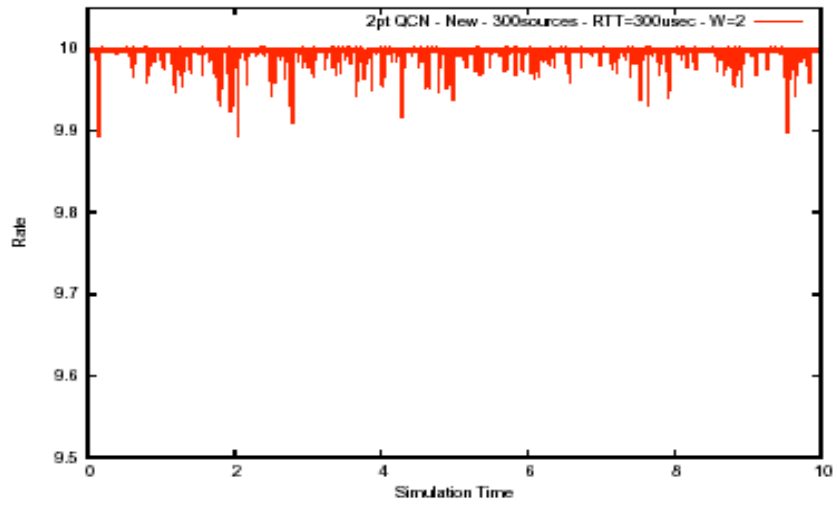


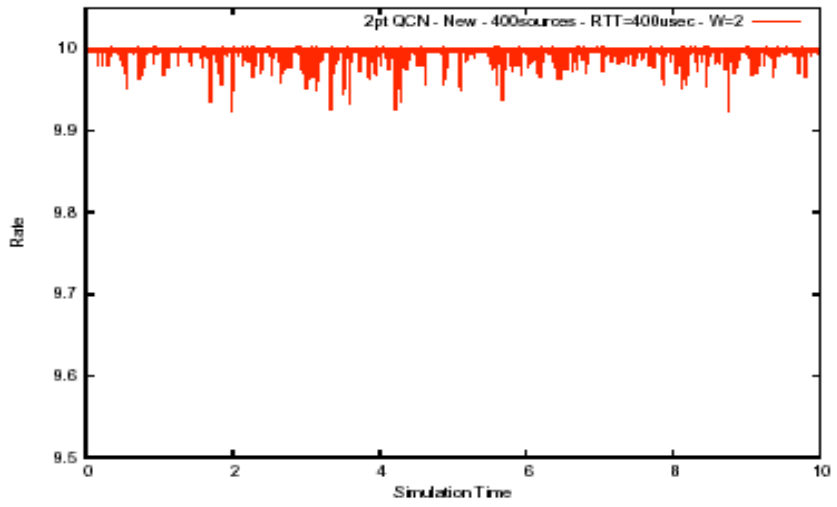


Varying RTT and # of Sources: Rate









Conclusions

- QCN shows good stability when both the number of sources and RTT are varied.
- We are currently developing a Markov chain model to analyze the scaling behavior that would capture both the equilibrium and transient modes.