Observations on Fairness Mechanisms Specified in Draft 1.1

Bob Doverspike, Chuck Kalmanek, Jorge Pastor, K. K. Ramakrishnan,
Aleksandra Smiljanic, Dong-Mei Wang, John Wei

AT&T Labs. Research
Florham Park, NJ

(Acknowledgment: thanks to Ed Knightly and P. Yuan of Rice University for sharing ns-2 based RPR simulator)

Introduction

- Considerable, useful work has gone into the specification of fairness mechanisms in RPR
- Our work so far: understand the mechanisms in detail, preliminary simulations
- A service provider's perspective
  - Outline subset of our perceived needs
  - How does the current specification meet our requirements
- We appreciate fact that some of the decisions have already been made on requirements
  - Target is: single bottleneck in the network only; source based fairness
- This presentation focuses on properties of current draft
- Our focus has been on dual transit buffer, aggressive scheme
- In the future, desirable to address more general models of fairness
  - Source-destination flow based fairness (metro core network)
  - Address the multiple bottleneck case (both access and metro core)

A Perspective on Service Provider's Requirements

- We are evaluating RPR as potential technology for packet transport
  - Customer access networks: potentially multiple customers on same ring, with each node being at different customer site (owned by customer or provider)
  - Metro backbone networks: carry aggregated traffic from customer access networks
- Packet transport with multiple service classes important
  - Provides customer differentiation and hence potential pricing differentiation
- Likely to use "CIR" (committed) and "EIR" (excess) rate in offering customers service
- Pricing for service is likely to be a function of CIR and EIR
  - Customers will expect a level of service that is function of the cost of the service offered
  - Charge for EIR ⇒ Some expectation that customer paying more for a larger EIR a higher burst capability

Applications likely to use FE traffic

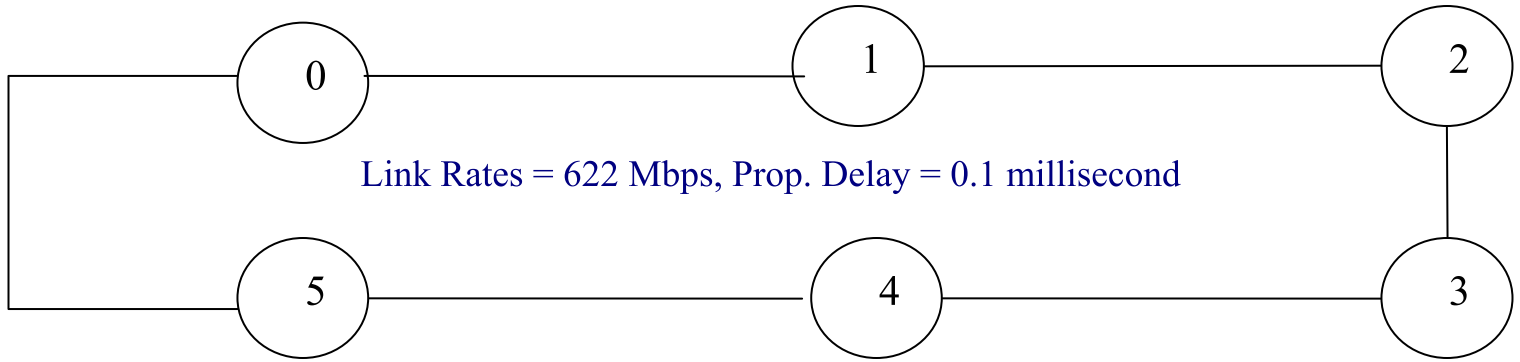
- Fairness eligible traffic likely to be used to carry application traffic that runs on top of TCP and UDP
- Applications: growing demand from streaming applications in metro area
  - Primary transport for streaming applications is RTP over UDP
  - Streaming applications increasingly use TCP
- There is some level of sensitivity to latency, even for web surfing applications
  - Because of human user involvement.
- Most applications are sensitive to loss
  - Design goal that MAC doesn't lose packets is important

Single choke/aggressive scheme

Our high level understanding of single choke aggressive scheme

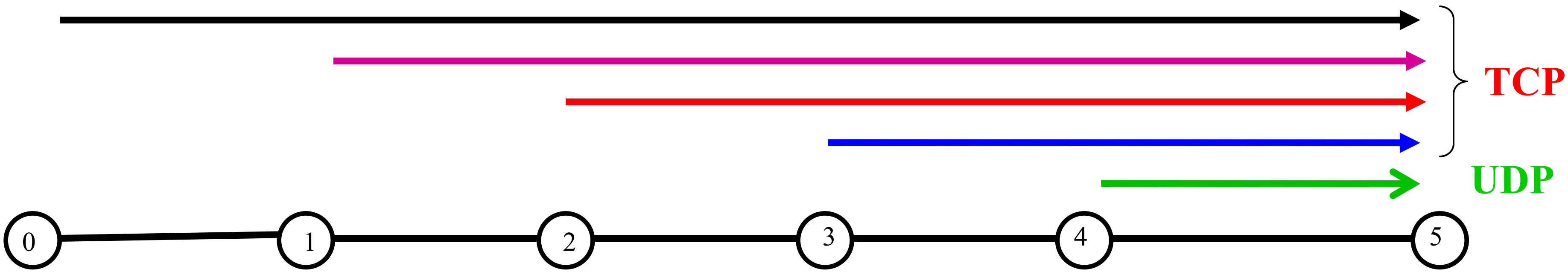
- Currently mandatory to use aggressive scheme with dual transit buffer implementations
- Goal is to fairly share single bottleneck's bandwidth in a given "congestion domain"
- When a span is "congested", backpressure mechanism using Fairness Control Messages is triggered
  - Congested ⇒ STQ buffer occupancy rises above "low threshold"
- When a span is congested, adjacent node communicates its local "add rate" to upstream nodes
  - Causes upstream nodes to reduce amount of data transmitted into the network
  - Substantial STQ buffer can receive any packets still arriving, to accommodate feedback delay
- When congestion clears, upstream nodes allowed to send at "full rate". Many details to make the scheme work in stable manner etc.

Observations: based on preliminary simulations



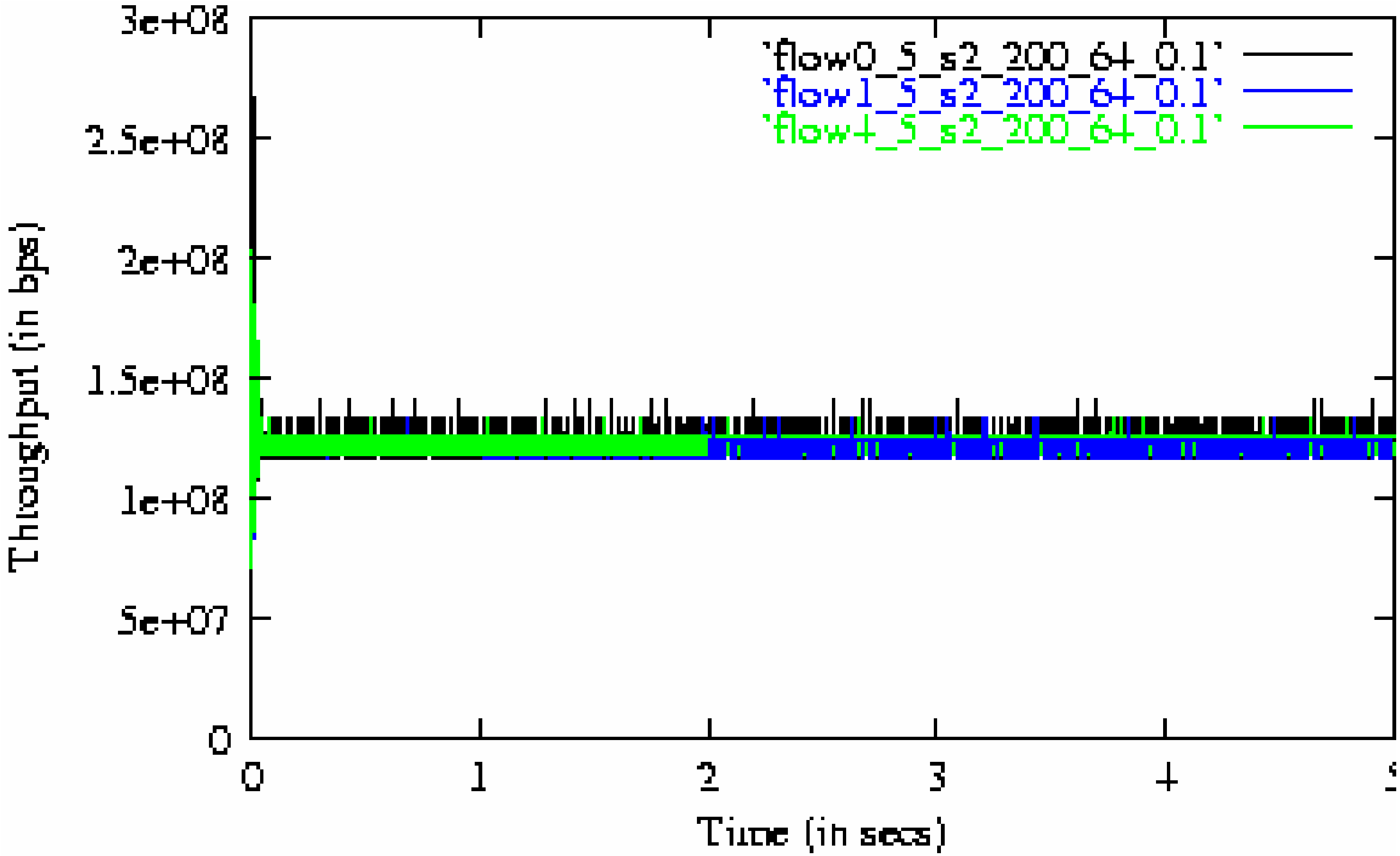
- Most of simulation results presented here based on a 6 node ring
  - Link rate = 622 Mbps; prop. delay = 0.1 millisecond; STQ = 256 Kbytes
  - Client buffer very large: 100,000 packets of buffering (experimented with smaller numbers also)
  - Single congestion domain, with one link being the bottleneck
- Experiments:
  - Steady (greedy) TCP flows (FTP); max. window size =64
  - Fixed rate UDP flows (CBR) of varying rates
  - Mixture of TCP and UDP flows

TCP Simulation Setup



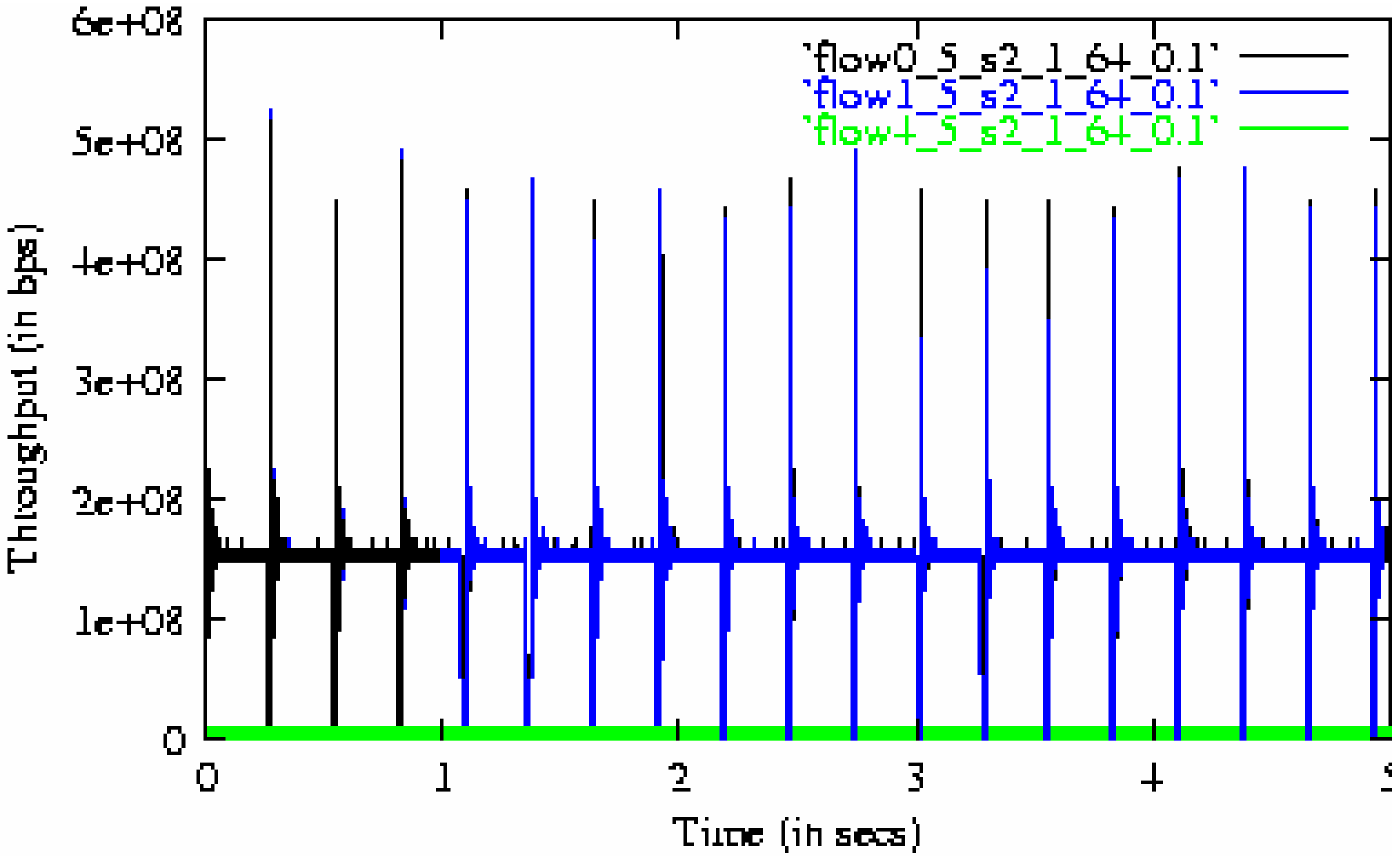
- UDP flow is CBR on last hop; rate varied
- 10 TCP flows from each RPR node; TCP co-resident w/MAC client
  - MAC does not drop packets – desirable feature
- Span 4 → 5 is bottleneck; Fair rate per source = 622/5 ≈ 124 Mbps, if all are "greedy"
  - When UDP flow's demand is reduced, fair rate for TCP flows may be higher
- Observation: UDP rate controls (strongly influences) the performance of individual TCP flows
  - TCP window grows as more packets are delivered without loss up to max. window size

TCP and UDP throughput



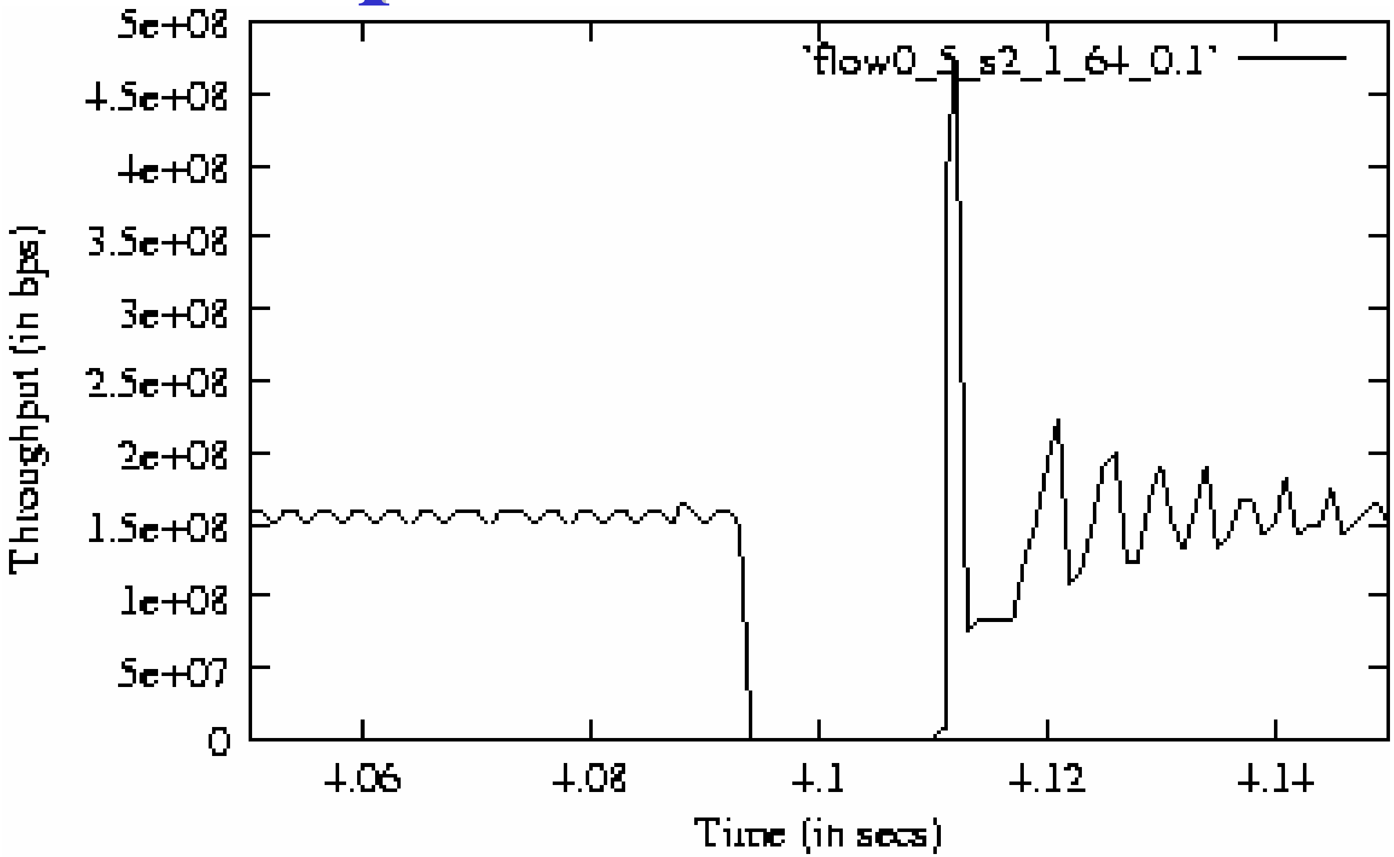
- UDP flow = 200 Mbps
- Fairness (all flows get 124 Mbps) achieved;
  - performance quite acceptable

TCP and UDP throughput



- UDP flow = 1 Mbps (green); two TCP flows (node 0 and 1) shown
- Long term fairness achieved (all TCP flows get 124 Mbps)
  - Overall TCP throughput {(max. sequence #)/(time)} is reasonable
- But, TCP flows experience considerable oscillation in throughput, in a synchronous manner: is this bad?

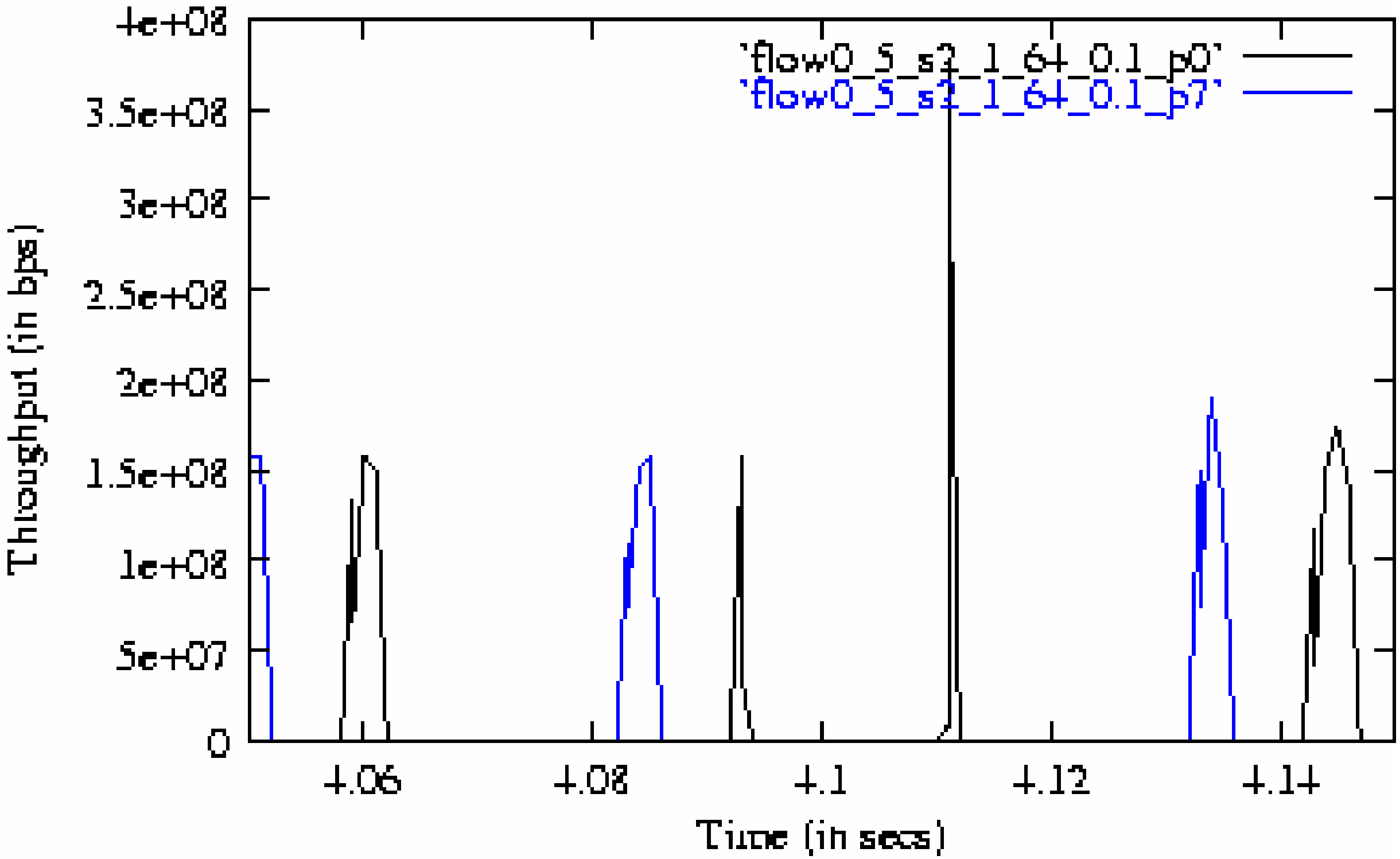
Close-up behavior of one node



<u>Short term behavior of aggregate of 10 TCPs from node 0</u>

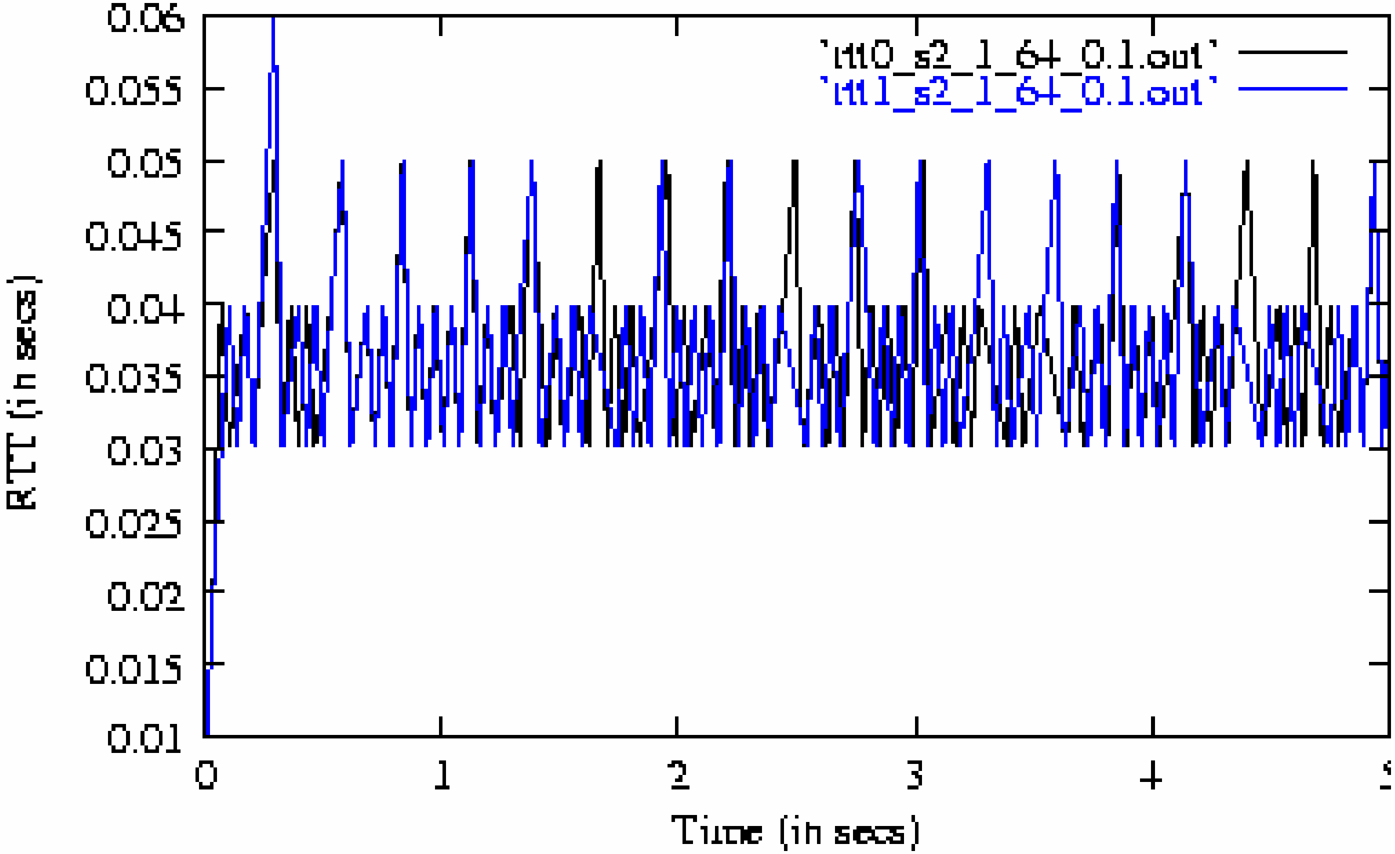
- All the ten TCPs being nearly idle for 15-20 milliseconds
  - Packets buffered during idle period. Drained in burst when congestion clears
- Large spike in throughput drains client buffer (≈ 500 Mbps)

Behavior of individual TCPs



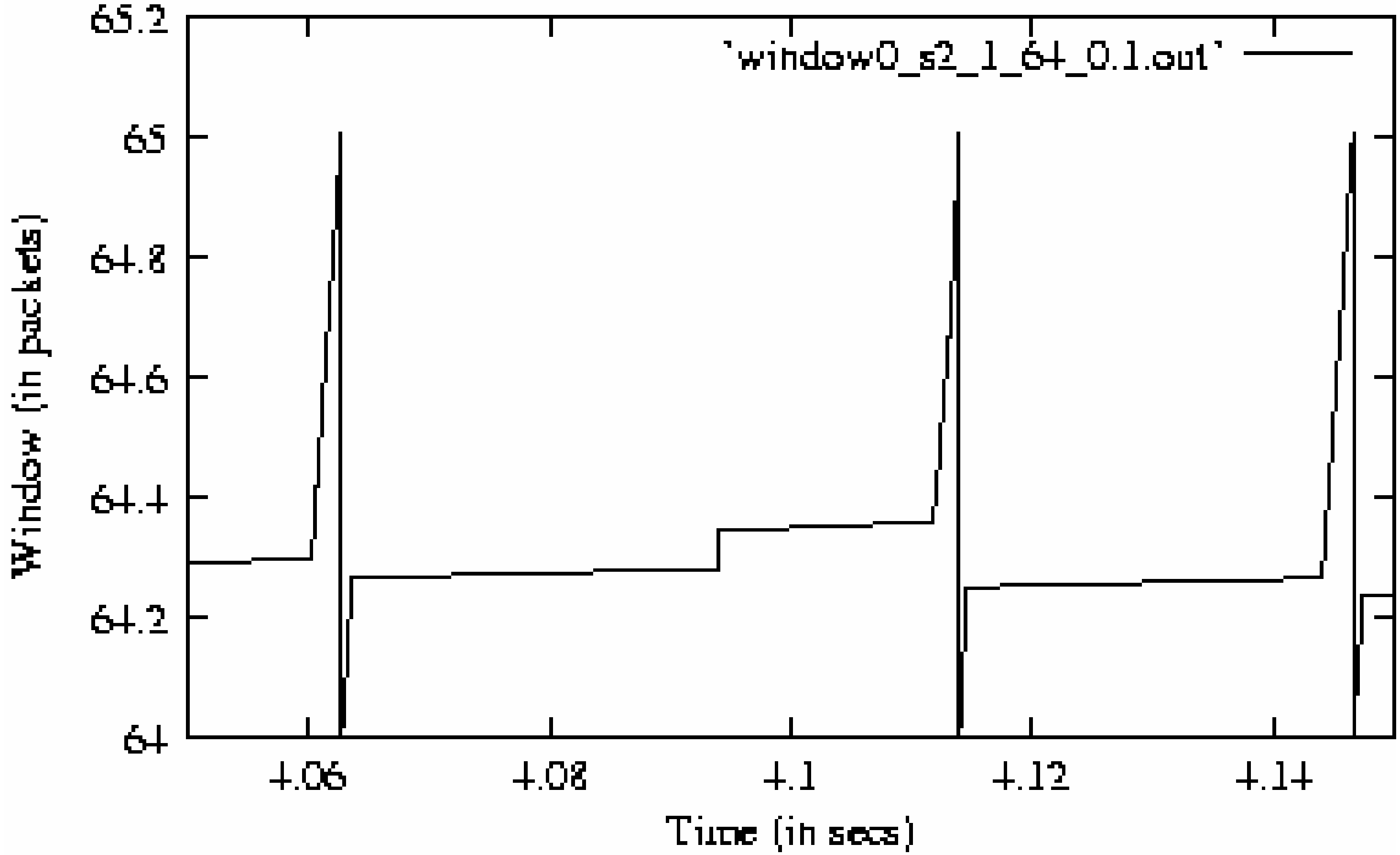
- Each individual TCP from node 0 served in round robin manner
- But all TCPs sources at node (in fact all nodes) go down to near 0 (waiting for acks) when last hop is congested
  - Recovery of lost throughput is through a large spike when congestion clears

RTT Behavior of TCP flows



- ❑ Large delay even though total propagation delay = 0.5 milliseconds
  - ➢ Reflects considerable size queues at the ingress clients
- ❑ Oscillations in RTT follow the pattern of aggregate throughput
  - ➢ Unclear if this oscillation in delay is acceptable
    - ❖ Interactive streaming applications using TCP or RTP/UDP likely to impacted

TCP Window up close



- TCP idle during15-20 milliseconds of congestion at last hop (which includes time to communicate upstream when congestion clears)
  - TCP sequence # doesn't grow while waiting for acks.
  - Packets are still in local ingress node's client buffer

Observations on Behavior

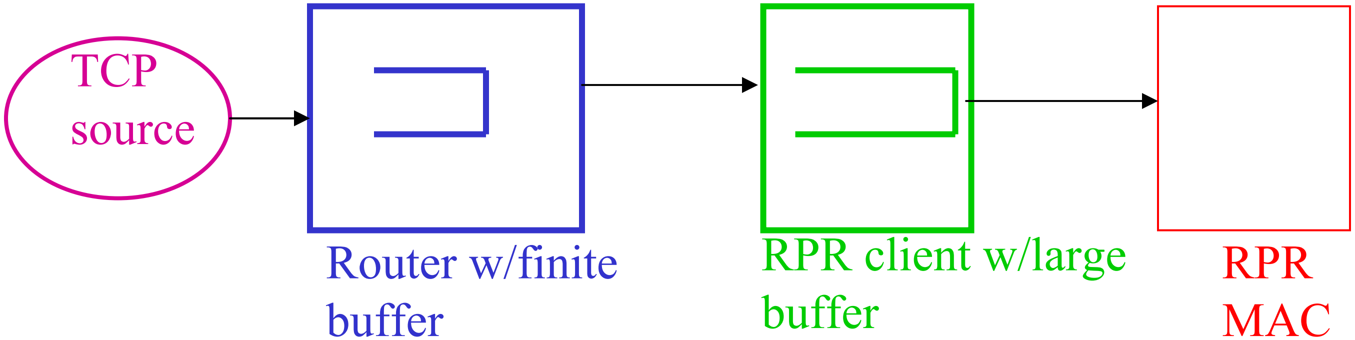
- Overall throughput of individual TCP's over the long term is almost fair across source nodes
  - Unfairness and oscillations in the short term
  - Client buffers can store packets generated by TCP while local MAC's transmit rate constrained
- However, packets have to be drained quickly after congestion clears
  - EIR has to be sufficiently high to enable this
- Service provider may not wish to provision each individual node to have such a large EIR (nearly line rate)
  - Even if we provision such large EIR: may result in loss at receiver?
- Customer may also not wish to pay for such a large EIR
- Provisioned EIR will cause policer on ingress to limit injection of packets by source
  - RPR MAC may not succeed in recovering lost throughput

Observations (contd…)

- Primary reason for oscillatory behavior: downstream node's low add rate
  - Upstream node limited to unreasonably unfair, low rate over short term
- Oscillations may be mitigated by determining fair rate on shorter time scales, based on who is sharing the bottleneck link
- By providing computed "fair rate" to upstream nodes
  - Allowed rate of upstream nodes is determined by the actual link bandwidth available
  - doesn't cause a complete shut-down of sources for brief periods of time
- Approximations of fair rate that may not be "true max-min" fair rate may be acceptable in the short term

Limitations in our understanding

- We have not yet simulated an overall system with routers, with limited buffers, end-hosts and additional non-RPR links
- Limited EIR and bursty behavior – may result in loss of packets?
- Interaction between TCP's congestion control upon loss of packets and RPR mechanisms need to be understood better
  - Past experience indicates that this is a critical piece of understanding needed in development of multiple layers of congestion control mechanisms
  - Where do policers and shapers reside?
- What happens when we include enough other services (some not TCP) and therefore start to include mild loss? How will throughput be affected?



Recommended Changes

- Currently, single transit buffer and "conservative" mode for fair rate operation are coupled together. Are they?
- We feel it is desirable to have multiple transit buffers to isolate traffic classes and interactions that may cause priority inversion
- Conservative mode for fair rate allocation appears to have potential to reduce oscillatory behavior (improvements needed?)
- Proposal: Reflect in Text and Pseudo code of Fairness section
  - Conservative or aggressive mode to be usable in general
- Allows implementors to choose
  - Single transit buffer or dual (or multiple) transit buffer
- Allows service providers to choose to deploy
  - Aggressive mode or conservative mode