Stability of QCN: The Averaging Principle

Mohammad Alizade, Berk Atikoglu,
Abdul Kabbani, Ashvin Lakshmikantha,
Rong Pan, Balaji Prabhakar, Mick Seaman

Overview

- Paper on QCN with same authors recently written; has two main parts
  - QCN: Algorithm and theoretical model
    - This has been presented to the WG in July ‘08 at the Denver meeting
  - The Averaging Principle
    - A control-theoretic idea which can be applied to general control systems, ***not just*** congestion control systems, and which makes them more robust to increases in loop delays
    - Underlies the reason for the good stability of the QCN and BIC algorithms

- We describe the AP, apply it to BCN
  - We have also applied it to other algorithms in the Internet context
  - Since this is a second presentation, I’ll present updates

Background to the AP

- When the lags in a control loop increase, the system becomes oscillatory and eventually becomes unstable

- Feedback compensation is applied to restore stability; the two main flavors of feedback compensation in are:
  1. Determine lags (round trip times), apply the correct "gains" for the loop to be stable (e.g. XCP, RCP, FAST).
  2. Include higher order queue derivatives in the congestion information fed back to the source (e.g. REM/PI, BCN).
  - Method 1 is not suitable for us, we don't know RTTs in Ethernet
  - Method 2 requires a change to the switch implementation

- The Averaging Principle is a different method
  - It is suited to Ethernet where round trip times are unavailable
  - It doesn't need more feedback, hence switch implementations don't have to change
  - QCN and BIC-TCP already turn out to employ it

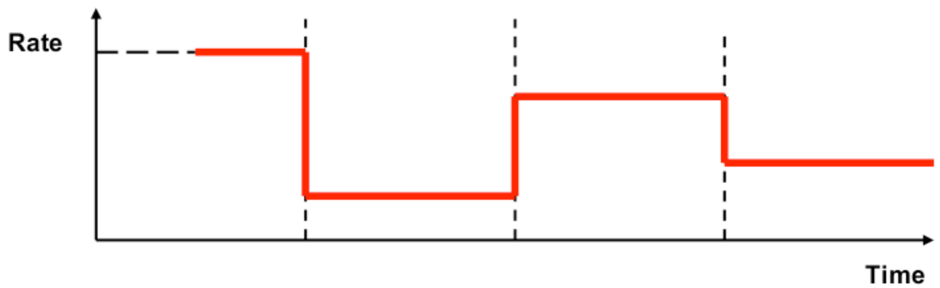
Rest of the presentation

- Statement of the Averaging Principle

- Application to:
  1. RCP: Rate Control Protocol by Dukkipatti and McKeown
  2. A “textbook” control loop, ***completely unrelated*** to congestion control
  3. (BCN: already presented on phone call, skipping here; included in appendix for completeness)

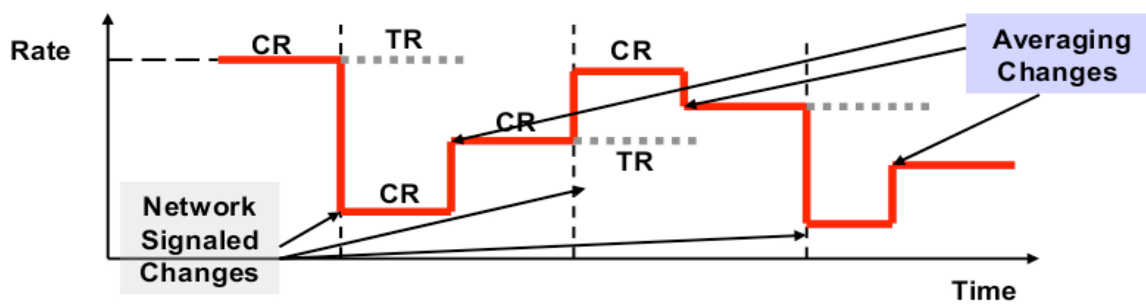
The Averaging Principle (AP)

- A source in a congestion control loop is instructed by the network to decrease or increase its sending rate (randomly) periodically



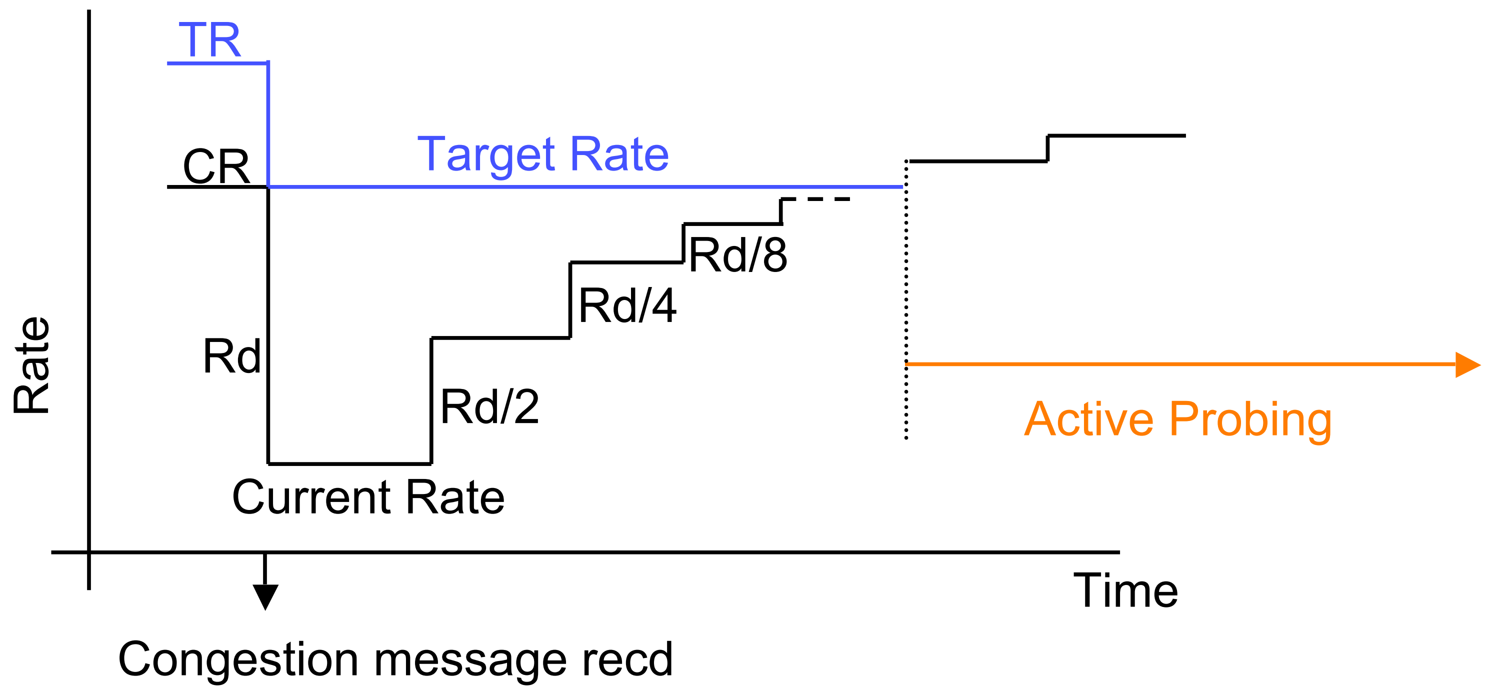
- AP: a source obeys the network whenever instructed to change rate, and then ***voluntarily*** performs ***averaging*** as below

TR = Target Rate
CR = Current Rate



Recall: QCN does 5 steps of Averaging

- In the Fast Recovery portion of QCN, there are 5 steps of averaging
- This averaging turns out to be a fundamental reason for the good stability of QCN and BIC-TCP

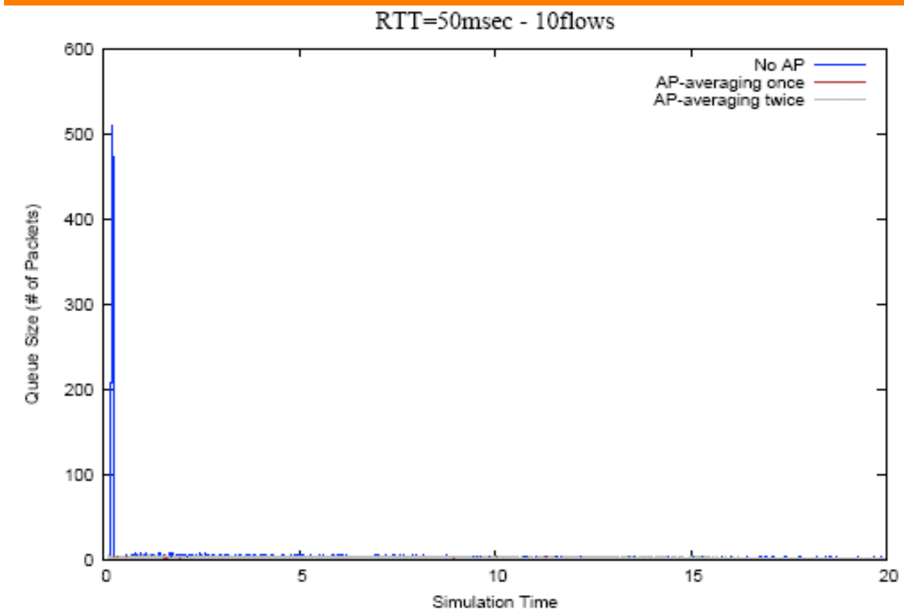


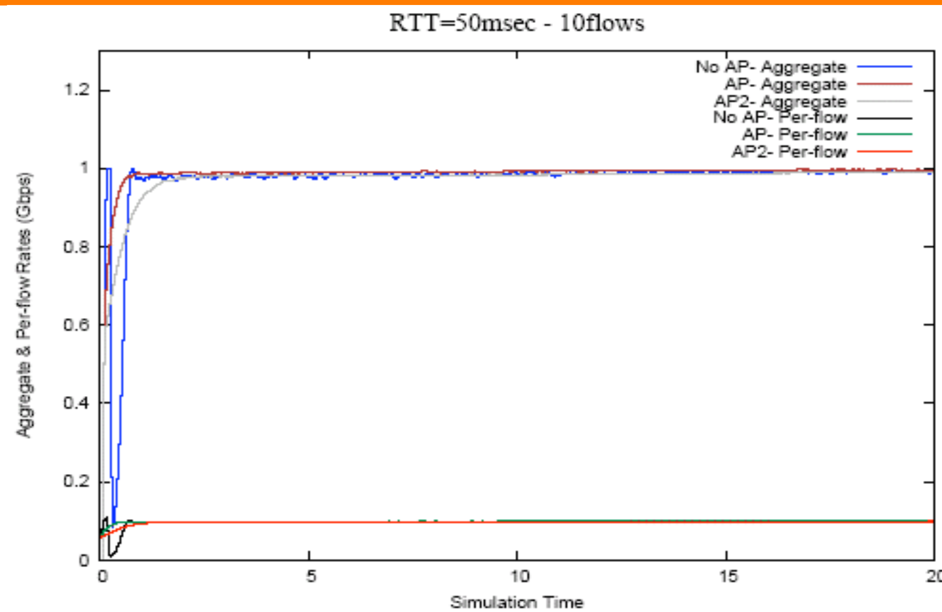
Applying the AP
RCP: Rate Control Protocol
Dukkipatti and McKeown

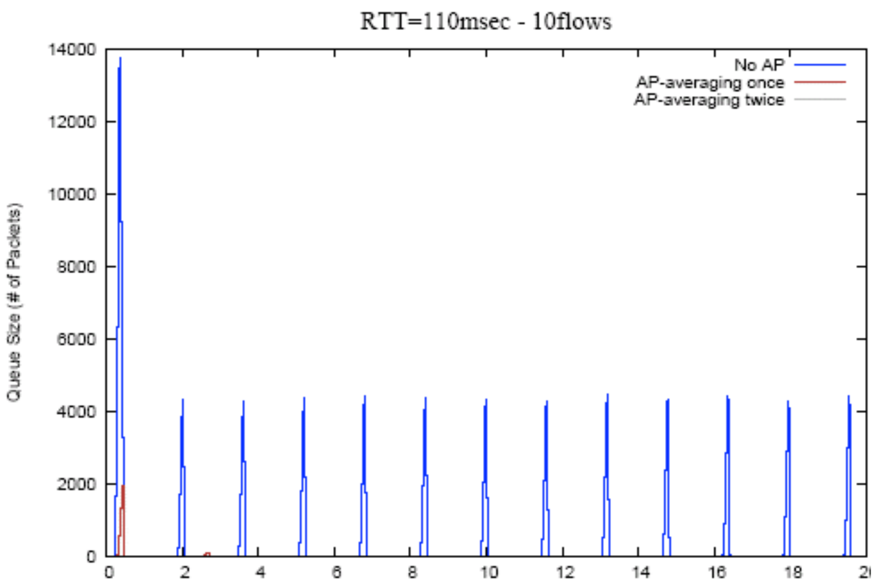
- A router computes an upper bound *R* on the rate of all flows traversing it.
- *R* recomputed every T (= 10) msec as follows: $R = R(1+Fb)$
  - $Fb = \dfrac{(T/RTT)*(\alpha*(C - I) - \beta*(Q/RTT))}{C}$
  - RTT: Round trip time estimate (set constant=T in our case)
  - C: link capacity
  - Q: Current queue size at the switch
  - I: incoming rate
  - $\alpha = 0.1$
  - $\beta = 1$

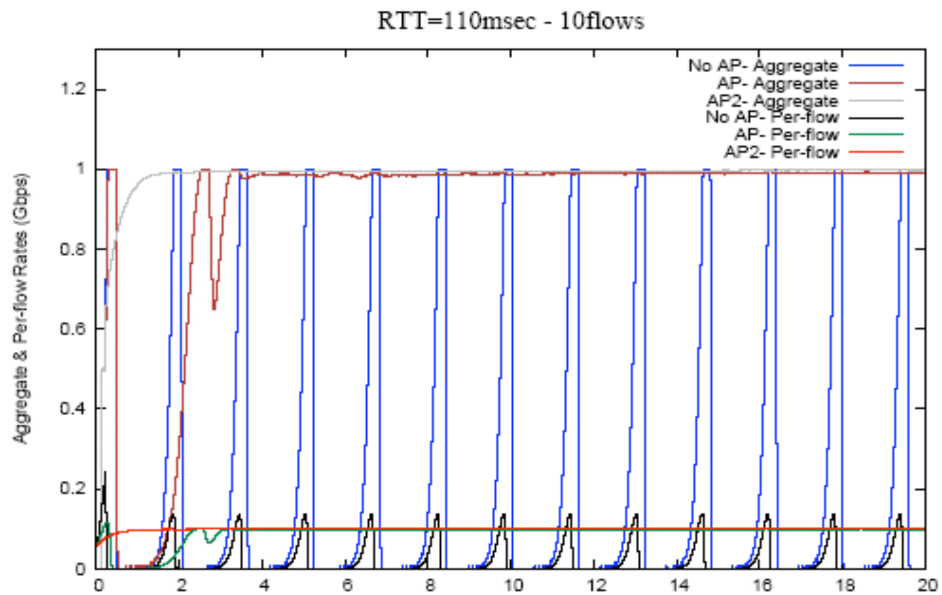
- A flow chooses the smallest advertised rate on its path.

RCP-AP Stability

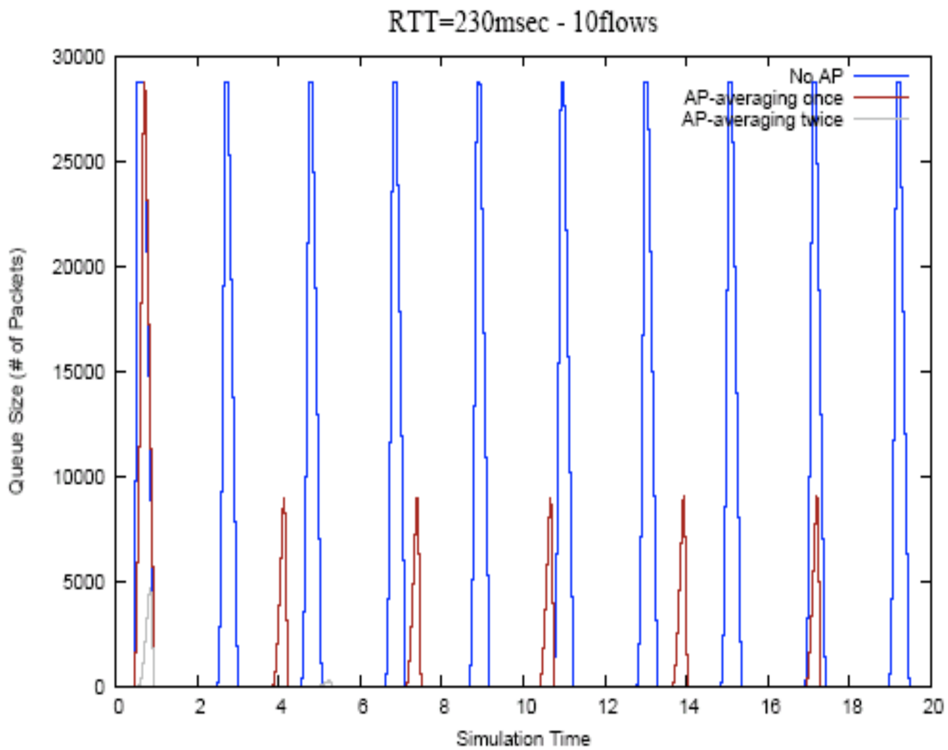


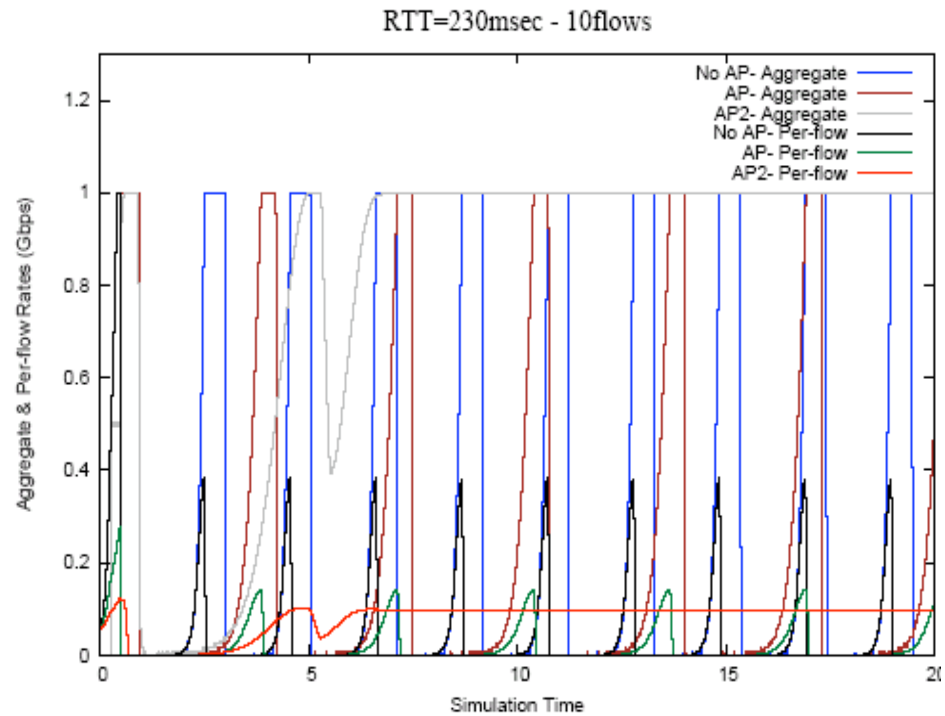






RCP-AP Stability





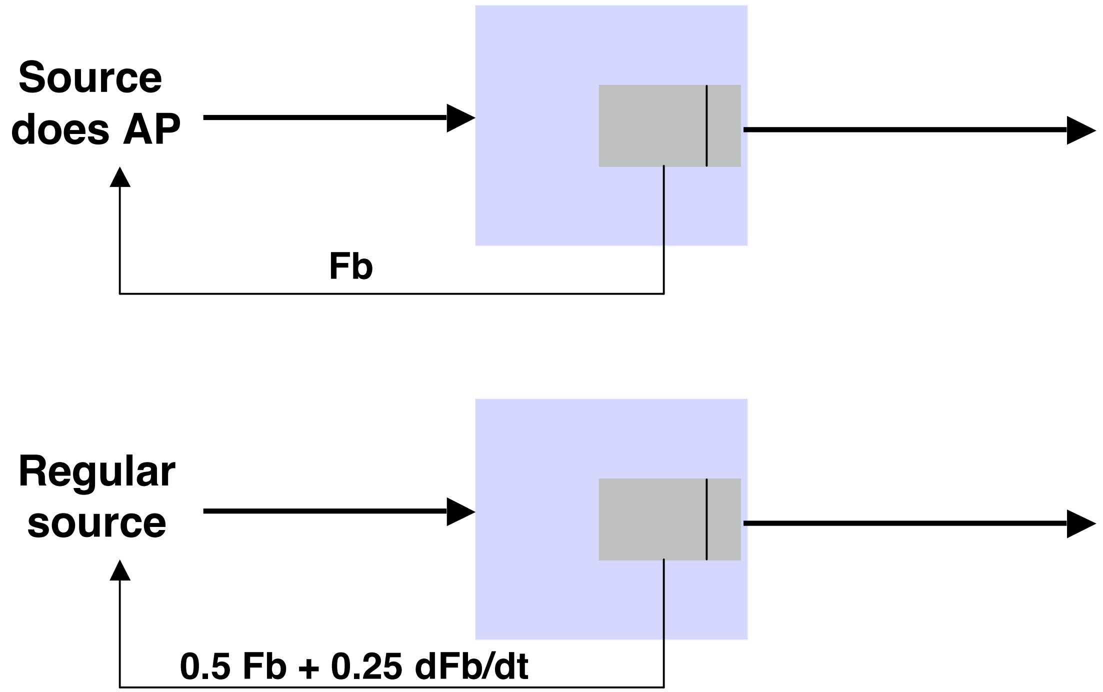
- Conclusions:
  - The AP improves (doubles, for small numbers of sources) stability with respect to lags; hence, it is useful in high bandwidth delay product networks.
  - Averaging twice or more leads to better stability, but makes the system more sluggish.

Understanding the AP

- As mentioned earlier, the two major flavors of feedback compensation are:
  1. Determine lags, chose appropriate gains
  2. Feedback higher derivatives of state

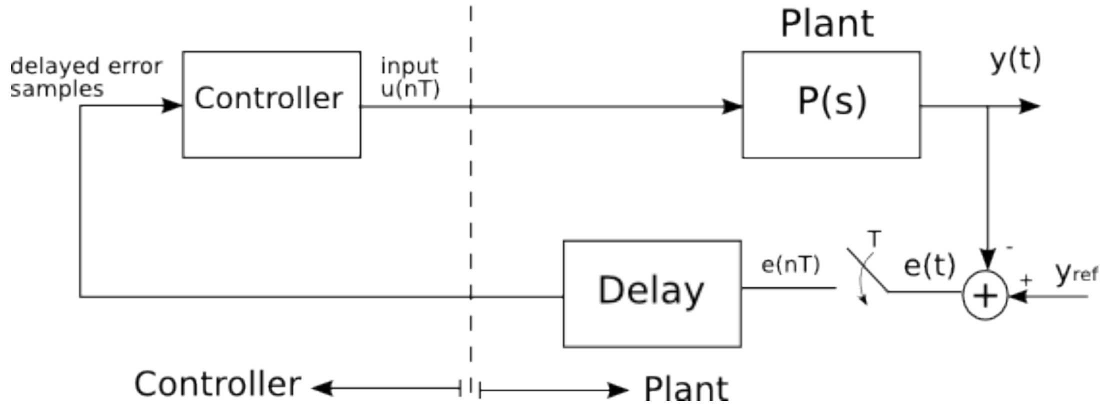
- We prove that the AP is sense equivalent to both of the above!
  - This is great because we don't need to change network routers and switches
  - And the AP is really very easy to apply; no lag-dependent optimizations of gain parameters needed

AP Equivalence: Single Source Case



- Systems 1 and 2 are discrete-time models for an AP enabled source, and a regular source respectively.
- ***Main Result:*** Systems 1 and 2 are algebraically equivalent. That is, given identical input sequences, they produce identical output sequences.
  - Therefore the AP is equivalent to adding a derivative to the feedback ***and*** reducing the gain!
  - Thus, the AP does both known forms of feedback compensation without knowing RTTs or changing switch implementations

A Generic Control Example

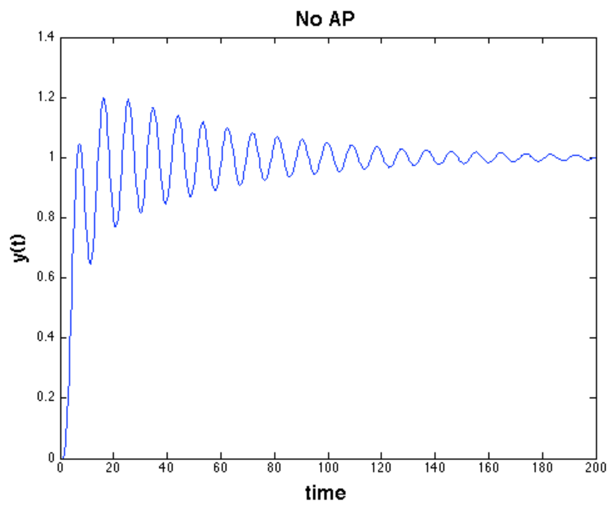


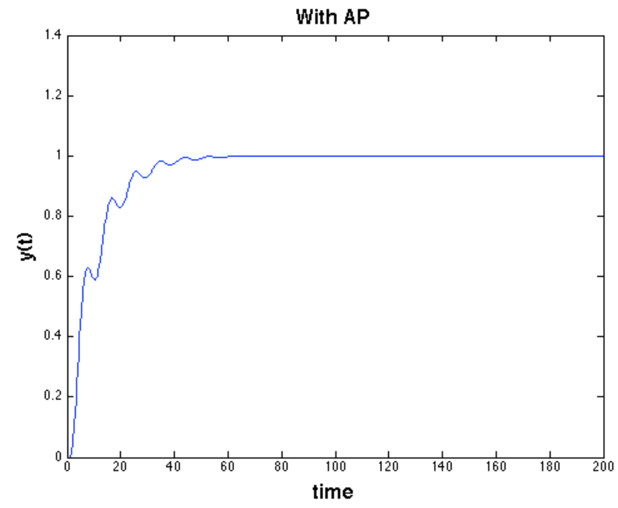
- As an example, we consider the plant transfer function:

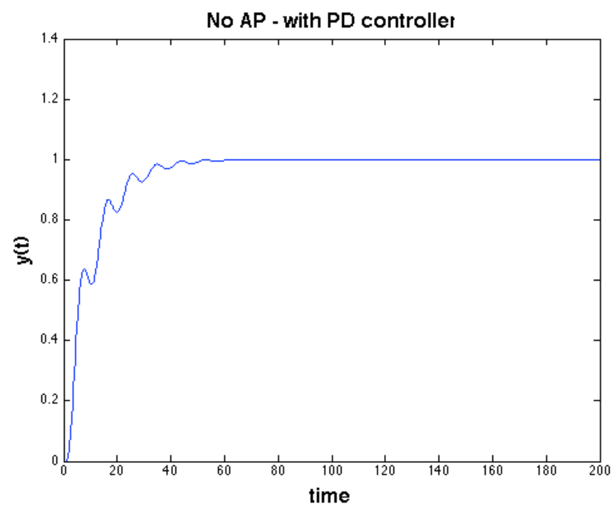
$$P(s) = (s+1)/(s^3+1.6s^2+0.8s+0.6)$$

Step Response

Basic AP, No Delay

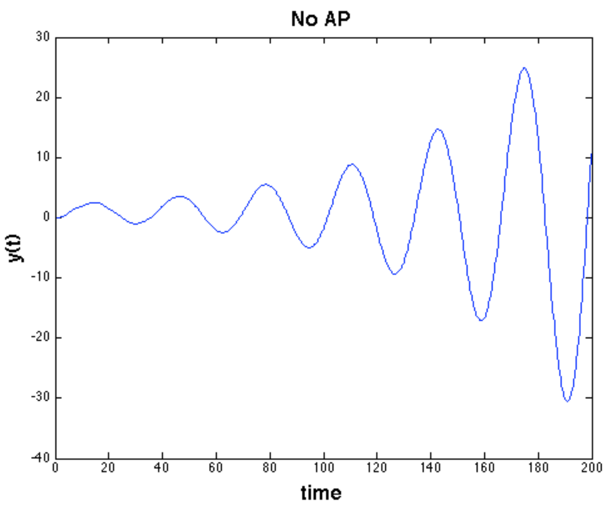


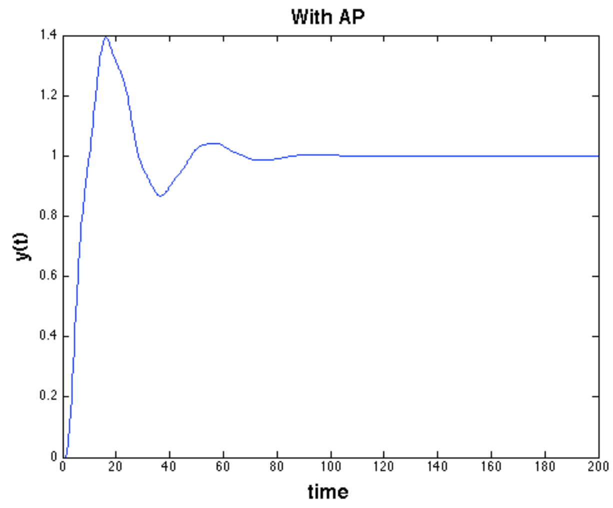


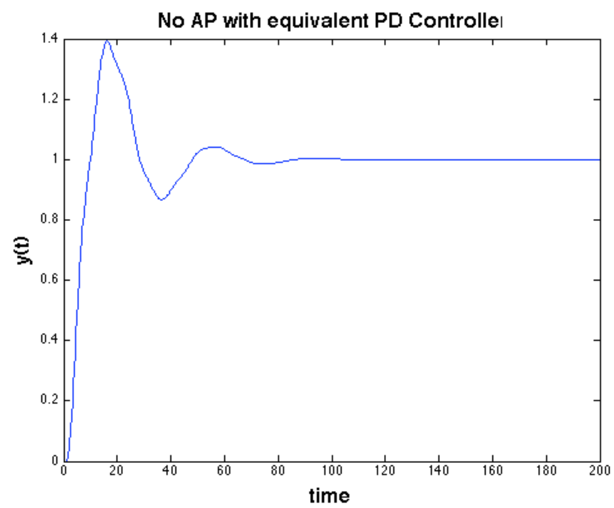


Step Response

Basic AP, Delay = 8 seconds

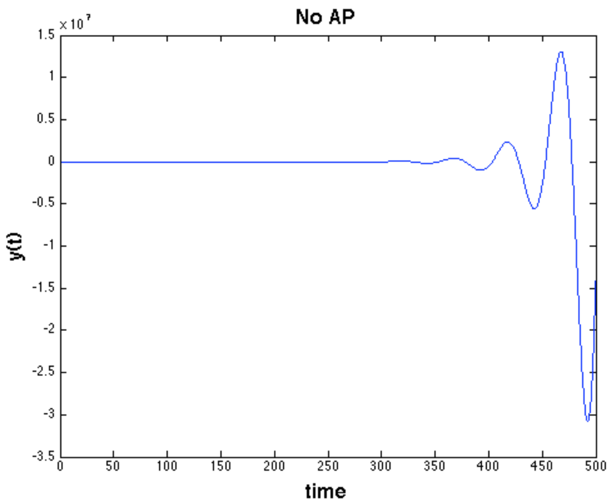


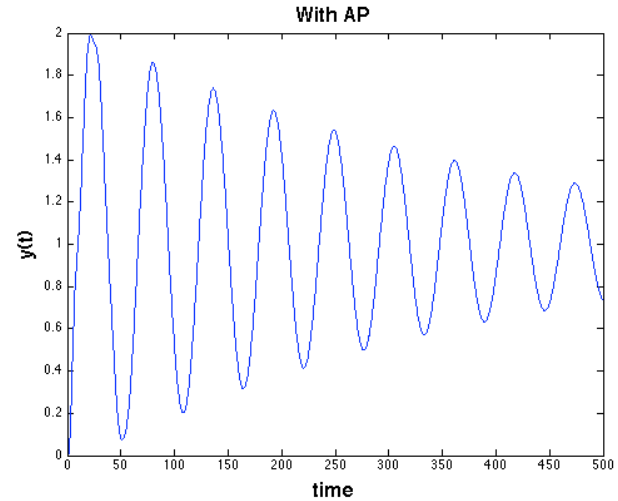


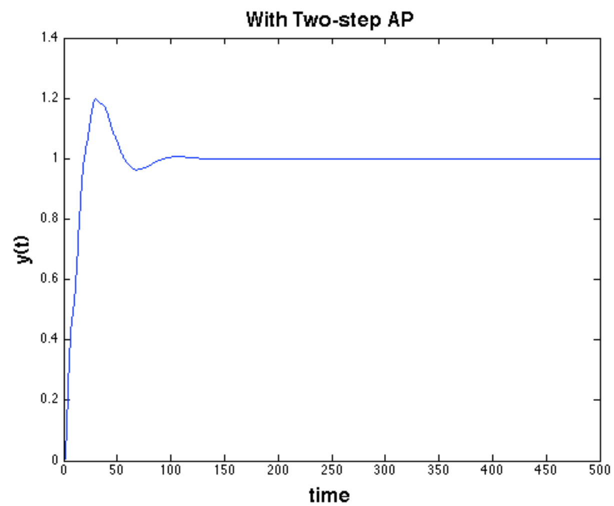


Step Response

Two-step AP, Delay = 14 seconds

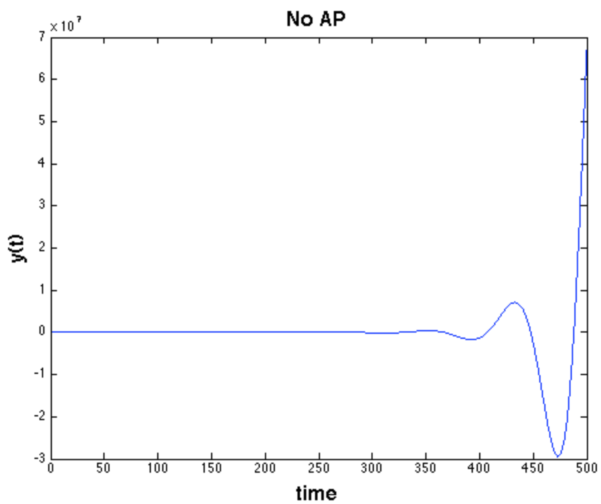


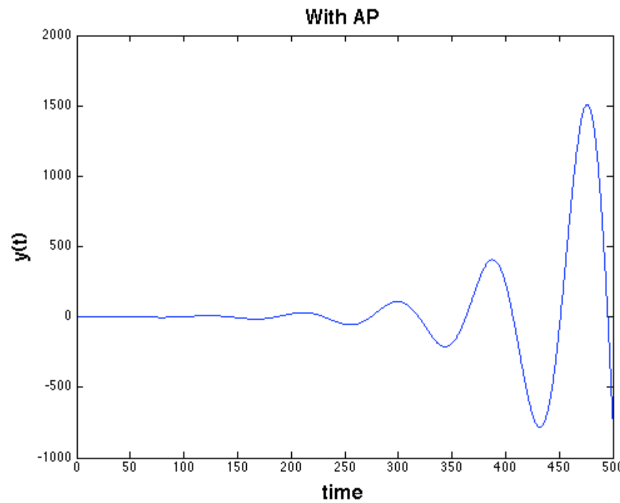


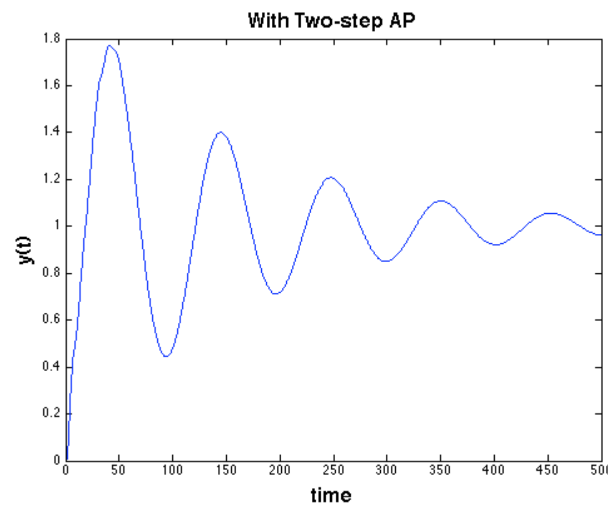


Step Response

Two-step AP, Delay = 25 seconds







Two-step AP is even more stable than Basic AP

Recall that QCN does 5 steps of AP in the Fast Recovery cycle: very stable

Conclusion

- The AP is a simple method for making many control loops (not just congestion control loops) more robust to increasing lags
- Gives a clear understanding as to the reason why the BIC-TCP and QCN algorithms have such good delay tolerance: they do averaging repeatedly
  - There is a theorem which deals explicitly with the QCN-type loop
- Variations of the basic principle are possible; i.e. average more than once, average by more than half-way, etc
  - The theory is fairly complete in these cases

Appendix: BCN with AP

Averaging applied to BCN

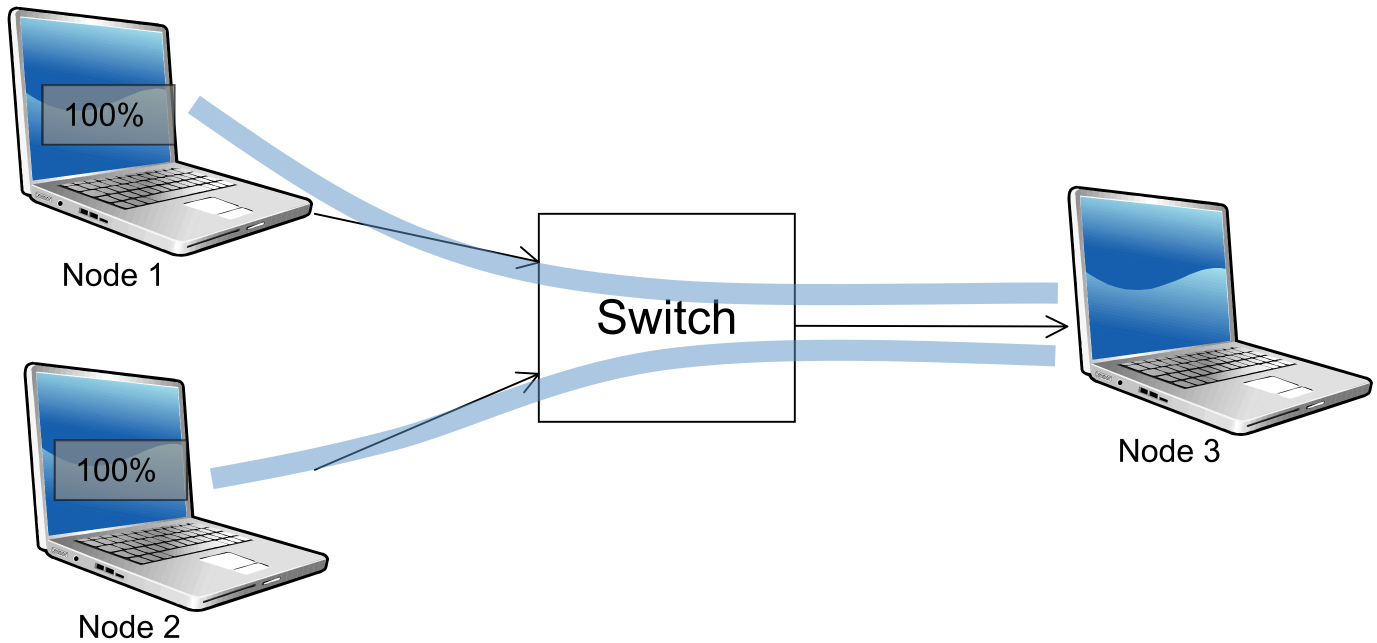
- Algorithm
  - When Fb (positive or negative) is received:
    - Apply Fb to modify Current Rate
    - Set Target Rate = old Current Rate
  - Apply averaging after 50 packets are sent:
    - Current Rate = *0.5* * Target Rate + 0.5 * Current Rate

- Recall: Rule for modifying current rate in BCN

$$R \leftarrow \begin{cases} R + G_i R_u F_b & \text{if } F_b \geq 0 \\ R(1 - G_d |F_b|) & \text{if } F_b < 0 \end{cases}$$

19

BCN with AP: Scenario and Workload



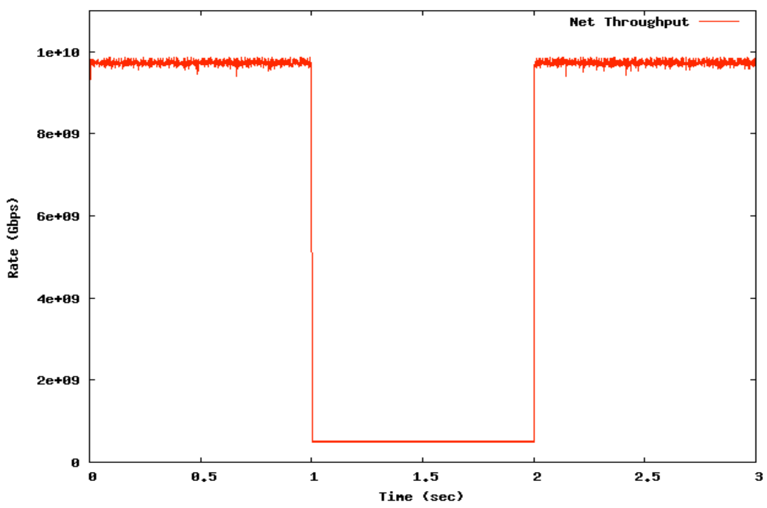
- 2 flows destined to node destined node 3 (First flow from Node 1, second flow from Node 2)
- Each flow is at maximum rate (10Gbps)
- Traffic: uniform
- Duration: 3s
- Service rate at switch is decreased to 0.5G from 1s to 2s
- RTT: varying

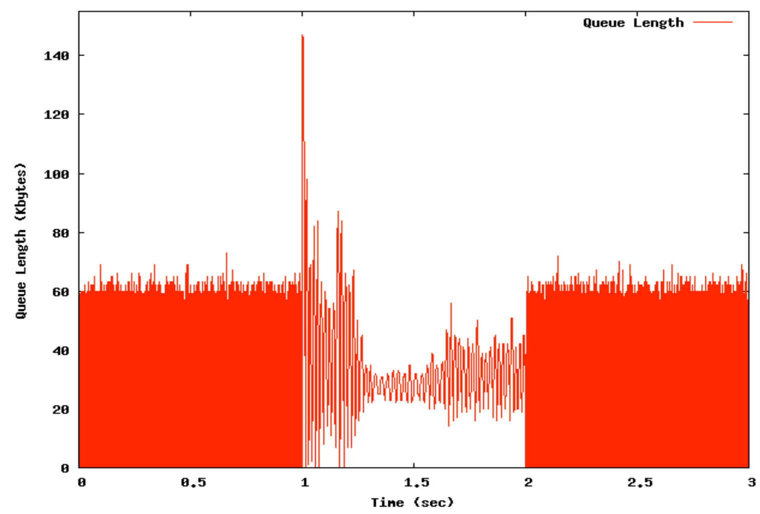
BCN parameters

- – Qeq = 375
- – Qsc = 1600
- – Qmc = 2400
- – Qsat disabled
- – Ecm00 disabled
- – Gi = 0.53333 (varies with RTT)
- **– W= 0 or 2**
- – Gd = 0.00026667
- – Ru = 1,000,000
- – Rd = 1,000,000
- – Td = 1ms
- – Rmin = 1,000,000

21

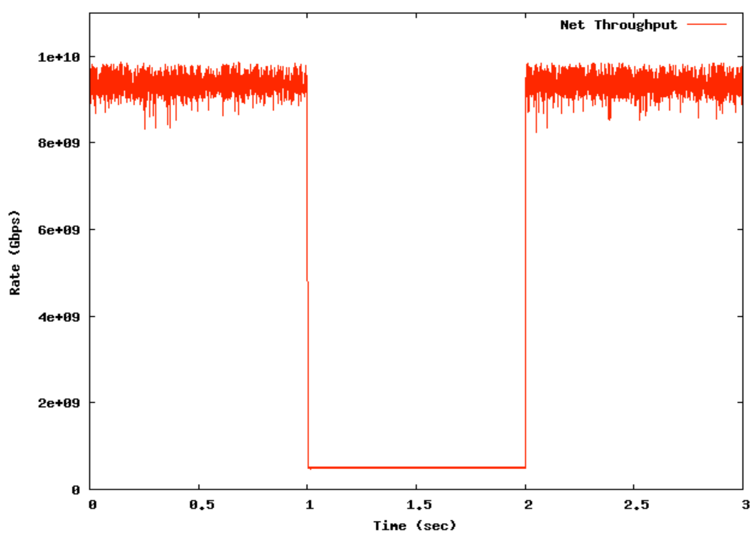
BCN v1.0
Fb = Qoff



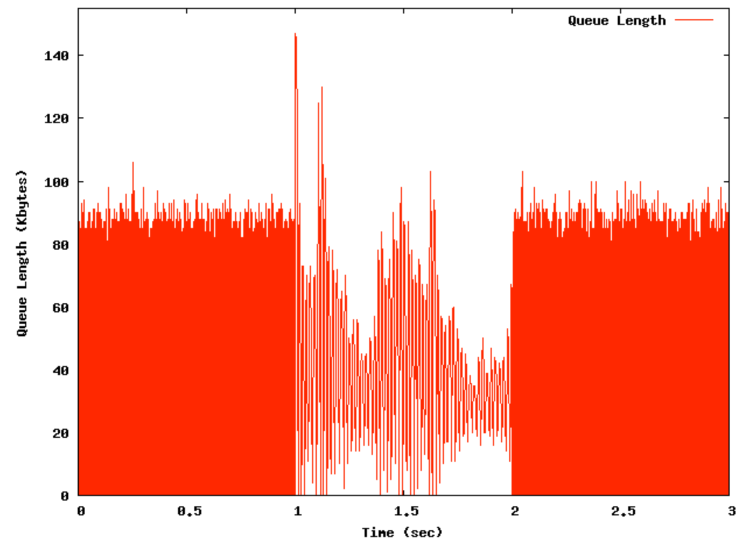


BCN v1.0
50 us

Fb = Qoff

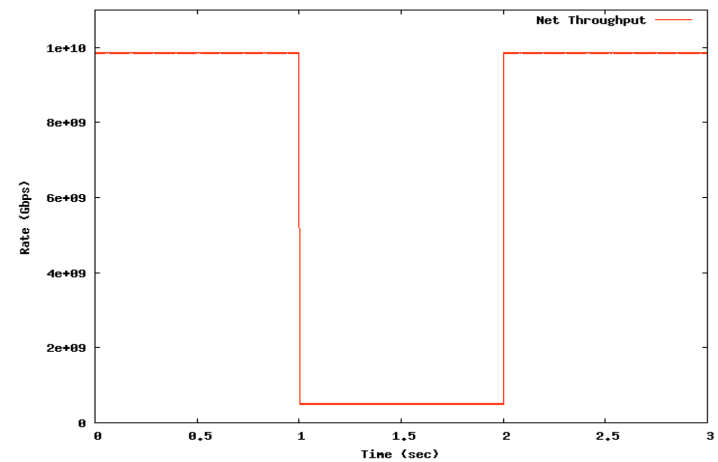


BCN v1.0
100 us

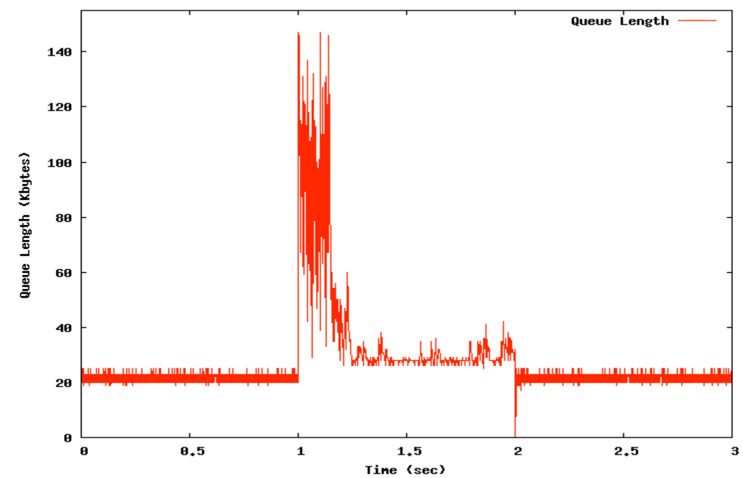


22

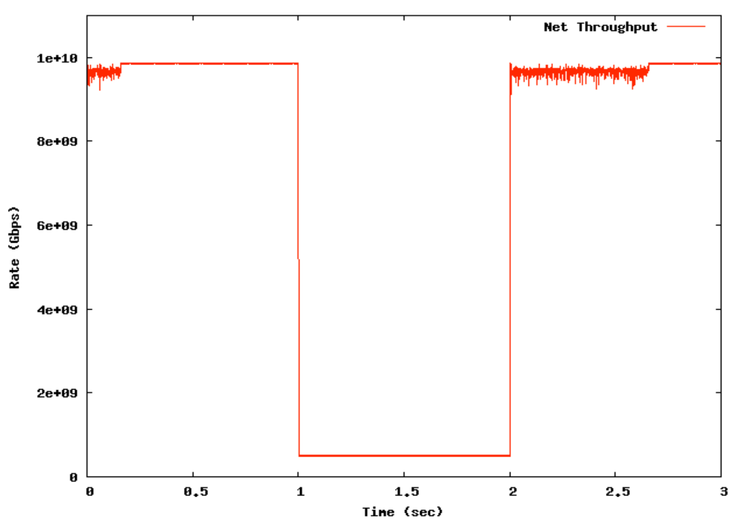
BCN v2.0
Fb = Qoff + 2 Qdelta



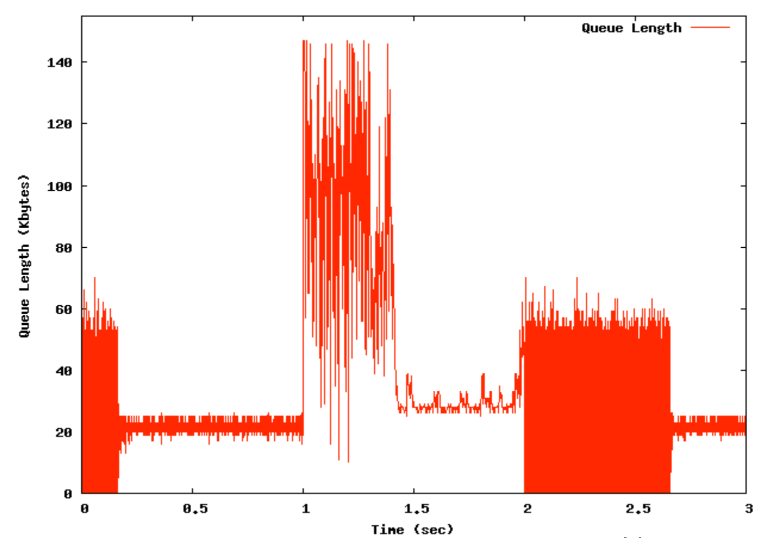
BCN v2.0
50 us



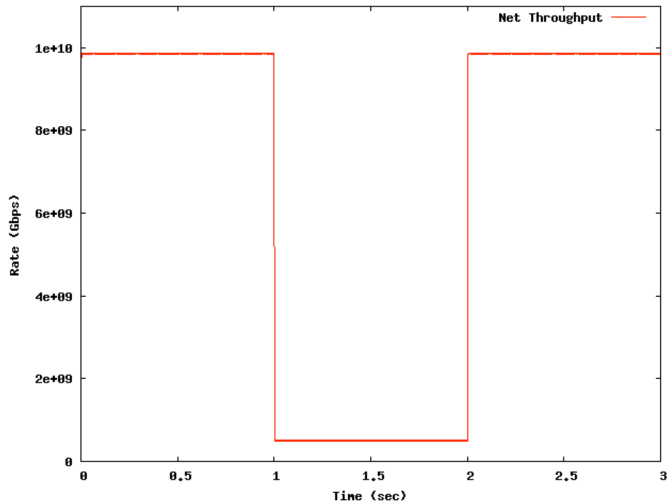
Fb = Qoff + 2Qdelta



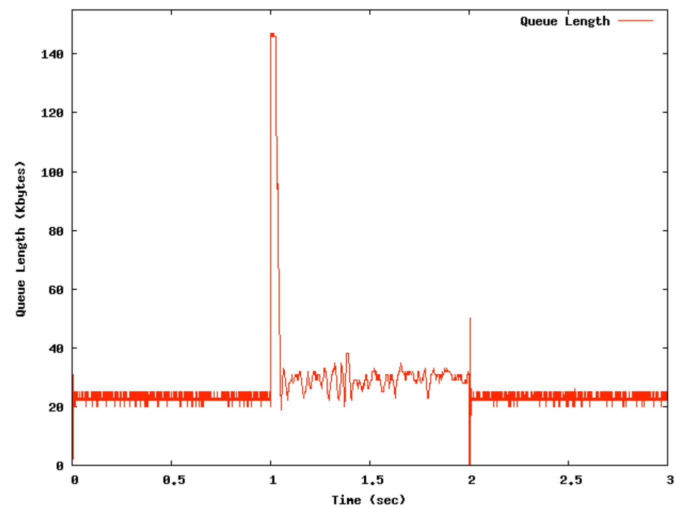
BCN v2.0
100 us



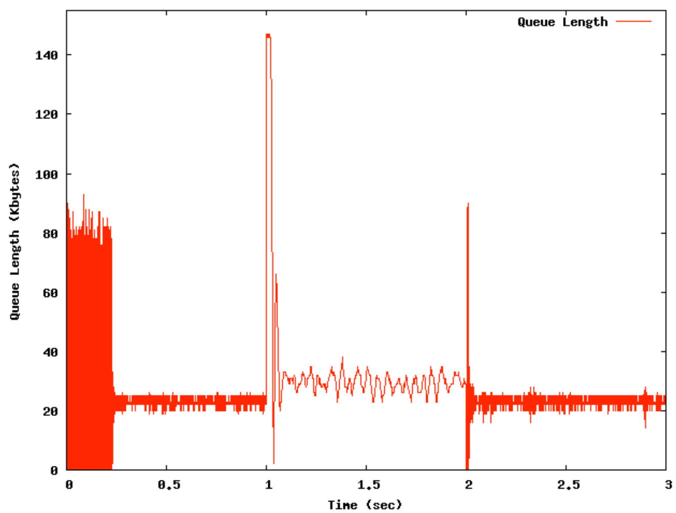
BCN v1.0 with AP
Fb = Qoff



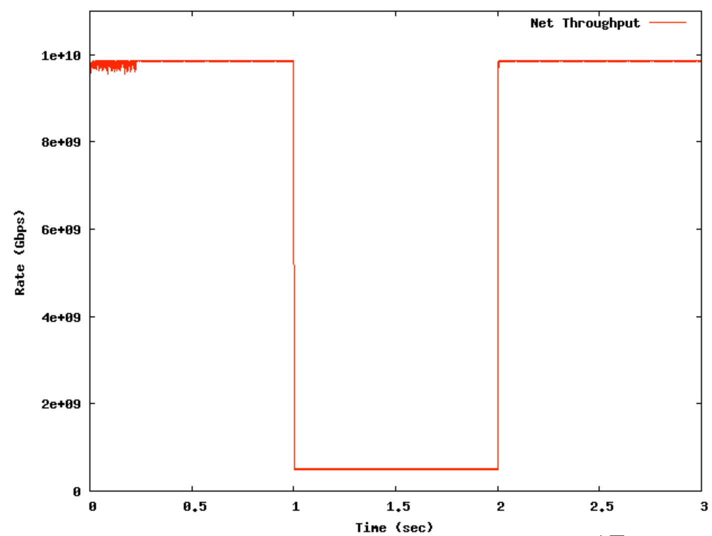
BCN *v1.0*
with AP
50 us



Fb = Qoff



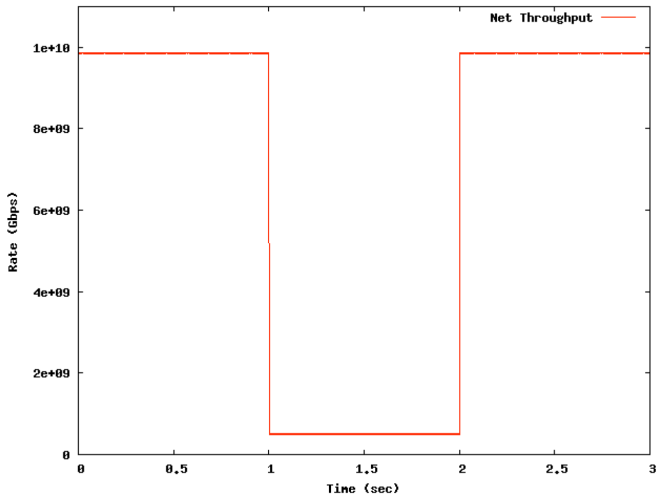
BCN *v1.0*
with AP
100 us



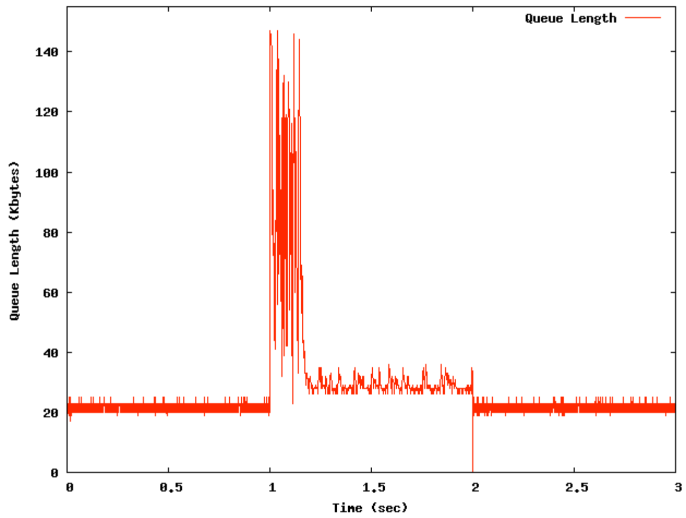
Summary of BCN v1.0 with AP

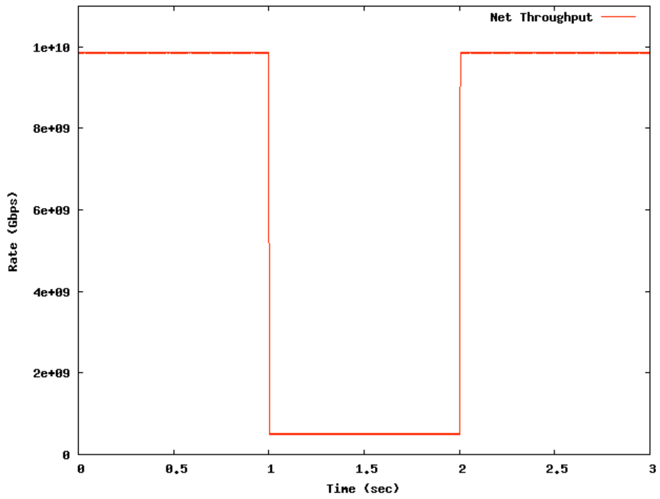
- We see that the AP provides an automatic stabilization to BCN v1.0 which is at least as good as that provided by BCN v2.0
  - The difference is that the AP does not require Qdelta
  - Qdelta requires a change at all switches, which we can avoid using the AP
- Now, suppose we take BCN v2.0, where Qdelta is already available
  - We saw in the Los Gatos meeting in Jan 08 that BCN v2.0 needs gain adjustments to be stable at large RTTs (200 us or so)
  - Can the AP be applied to BCN v2.0 to improve its stability?

BCN v2.0 vs BCN v2.0 with AP
RTT = 10 us

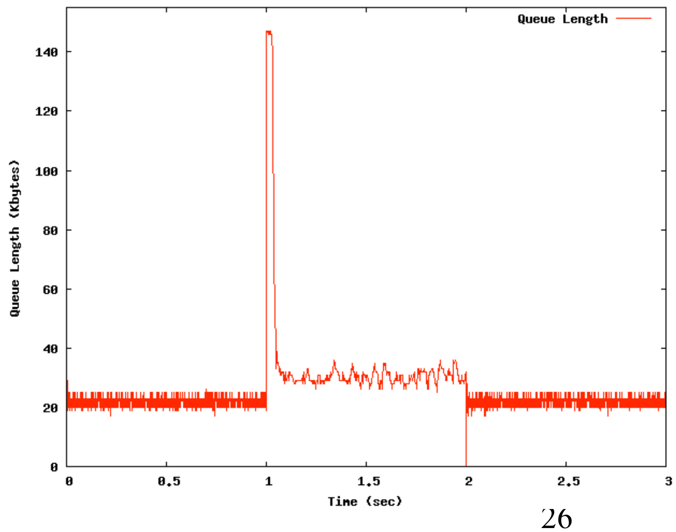


BCN *v2.0*

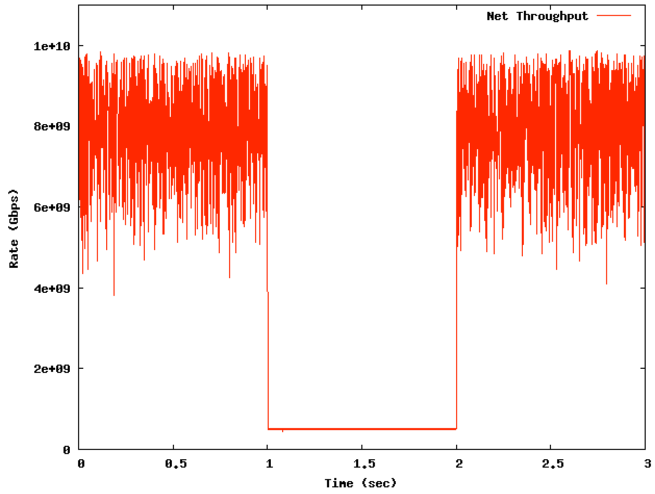




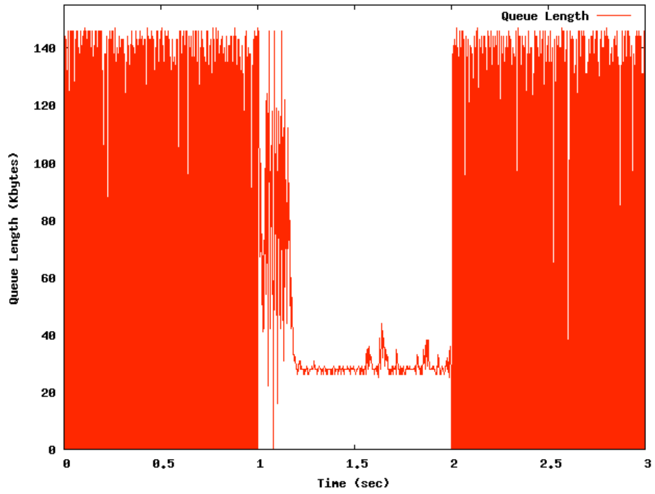
BCN *v2.0* with AP

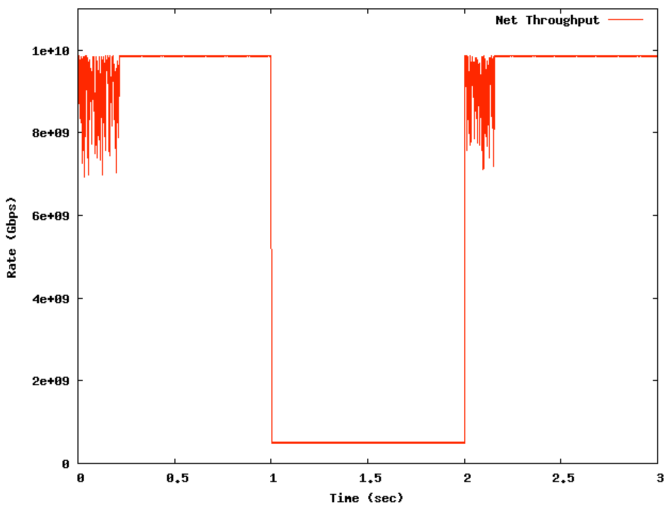


BCN v2.0 vs BCN v2.0 with AP
RTT = 250 us



BCN *v2.0*
(Los Gatos,
Jan ‘08)





BCN *v2.0*
with AP