QCN: Algorithm for P-code

Abdul Kabbani, Rong Pan, Balaji Prabhatkar and Mick Seaman
Two items to be discussed

1. SONAR/Fb99 approaches
   - Sensitivity of probing for available bandwidth
   - Complexity of having timer at switch (issue raised at Atlanta)

2. Timer-supported basic QCN
   - Adapts the drift timer at the source
   - This gives low recovery time in addition to stability
   - P-code will be posted by Rong after call
Recall: SONAR and Fb99

- In SONAR and Fb99
  - A source (RL) performs an endless loop of FR and AI; meanwhile, it is also probing for “available bandwidth”
  - Switches detect “bandwidth availability” based on queue going down “for a while”; timer used at the switch
  - When bandwidth becomes available on all paths fed by an RL
    - RL goes into Hyper AI
Recall: The SONAR Algorithm at RL

- **FR**
  - 5 cycles, 100 pkts each

- **AI**
  - 100 pkts/cycle, Ri Mbps increase

- **SONAR Ping**

- **Bdwdth NOT Available**

- **Bdwdth Available**

- **HAI**
  - 100 pkts/cycle,
  - Ri x cycle_cnt Mbps

\[ F_b \leq 0 \]
Remarks

• The RL can be in one of 3 states: FR, AI or HAI

• SONAR and Fb99 put HAI *in parallel* with FR and AI
  – That is, when bandwidth availability is detected, RL goes into HAI
  – It *does not send any further packets* in FR and AI modes

• The parallel structure achieved
  – Stability: Safe operation of RL during congestion: endless loop of FR--AI
  – Rapid response: Because of HAI triggered by a *timer*

• But there is a lot of sensitivity to correctly detecting “bandwidth availability”
  – We will see that it is not an easy condition to detect, esp for 2 sources and large RTT

• Conclusion: Place HAI *in series* with FR and AI for safe operation
  – In order to achieve rapid response use a timer at RL in addition to byte-counter
Basic QCN: FR--AI Rates
Basic QCN: FR--AI
Queue sizes
QCN with SONAR
HAI makes rate loss more severe
4srcs, rate
4srcs, qsize

- **NOTE**
  - With 4 sources and long RTT of 500 us, the stability and utilization are pretty good
  - So, a very small number of sources AND a very large RTT cause problems for aggressive sources
Summary

• “Bandwidth availability” tricky to detect
  – Going into HAI based on incorrect detection degrades performance

• Need to put HAI in series with FR and AI
  – This ensures stability

• Need to use a timer at the source
  – This ensures quick recovery of bandwidth
Timer-supported QCN

- **Byte-Counter**
  - 5 cycles of FR (150KB per cycle)
  - AI cycles afterwards (75KB per cycle)
  - Fb < 0 sends timer to FR

- **RL**
  - In FR if *both* byte-ctr and timer in FR
  - In AI if *only one of* byte-ctr or timer in AI
  - In HAI if *both* byte-ctr and timer in AI
  - Note: RL goes to HAI only after 500 pkts have been sent

- **Timer**
  - 5 cycles of FR (T msec per cycle)
  - AI cycles afterwards (T/2 msec/cycle)
  - Fb < 0 sends timer to FR
Rate Adjustments

When RL is in FR
- Upon completion of a byte-ctr or timer cycle: \( CR = (CT + TR)/2 \)
- EFR and Target rate reduction enabled during first cycle of byte-ctr

When RL is in AI
- Upon completion of byte-ctr or timer cycle: \( TR = TR + R_{AI}; \) \( CR = (CR + TR)/2 \)
(We’ve used \( R_{AI} = 5 \text{ Mbps} \))

When RL is in HAI
- Events = completion of byte-ctr or timer cycles
- Events numbered \( i = 1, 2, \ldots \)
- At the end of event number \( i \):
  \( TR = TR + (i*R_{HAI}); \) \( CR = (CT + TR)/2; \)
(We’ve used \( R_{HAI} = 50 \text{ Mbps} \))
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 = 50 usec
  - Buffer size = 100 pkts (150KB); Qeq = 22
  - T = 10 msecs
  - $R_{AI} = 5$ Mbps
  - $R_{HAI} = 50$ Mbps
Recovery Time

Stability not compromised
Recovery time = 80 msec
Stability (RTT = 50 usecs)
Stability (RTT = 50 usecs)

10 sources - RTT=50usec - adaptive timer= 10 or 5msec - adaptive counter=100 or 50pkts - 5Mbps AI - HA increment=50Mbps

100 sources - RTT=50usec - adaptive timer= 10 or 5msec - adaptive counter=100 or 50pkts - 5Mbps AI - HA increment=50Mbps
Stability (RTT = 50 usecs)

300 sources - RTT=50usec - adaptive timer= 10 or 5msec - adaptive counter=100 or 50pks
- 5Mbps AI - HAincrement=50Mbps

Rate vs Simulation Time

Queue Size vs Simulation Time

400 sources - RTT=50usec - adaptive timer= 10 or 5msec - adaptive counter=100 or 50pks
- 5Mbps AI - HAincrement=50Mbps

Rate vs Simulation Time

Queue Size vs Simulation Time
Stability (long RTT = 500 usecs)

2 sources - RTT=500 usec - adaptive timer= 10 or 5msec - adaptive counter=100 or 50pkt
- 5Mbit/s AI - HAIncrement=50Mbit/s

4 sources - RTT=500 usec - adaptive timer= 10 or 5msec - adaptive counter=100 or 50pkt
- 5Mbit/s AI - HAIncrement=50Mbit/s
Stability (RTT = 500 usecs)
Stability (500 usecs)
Conclusion

• Discussed trickiness of detecting available bandwidth

• Need to put HAI **in series** with FR and AI
  – Gives good stability; timer gives good recovery time

• Timer-supported QCN
  – Key ideas and features
    • Transmit 500 pkts before going to HAI
    • Byte-ctr gives network sufficient opportunity to send feedback to RL
    • Timer hastens recovery for slow rate sources
    • No change to switch operation from basic QCN; esp no timer at switch
    • Parameter free (no dynamic parameter choice needed)
  – Very similar to basic QCN
  – Can be further enhanced, e.g. with Fb-hat or Fb99
    • Need to seriously consider benefits of enhancements first!