NDN Congestion Control

Motivation, Assumptions, and Early Design

Klaus Schneider, Beichuan Zhang

March 24, 2016

Why NDN Congestion Control is Hard

NDN Architecture makes Congestion Control hard:

1. Unknown Endpoints
2. Universal Caching
3. FIB Aggregation
4. Multipath Forwarding
5. Deployment as IP Overlay or Dual-Stack

More specifically:

- Hard to set a good timeout
- Hard to set a good congestion window
- Hard to signal congestion to the right consumer

Clarifying Assumptions

Work in the literature disagrees about the assumptions.
Critical for design implications!

- Can we assume to know the link bandwidth ?
- Can we identify flows? (probably not)
  - Naming conventions? Header fields?
  - Is per-flow fairness feasible (state overhead) or even desirable (fairness might work differently in NDN) ?
- How much in-network state is feasible?
- Are *per-route labels* scalable and practical?
- Effect of caching strategies ?

Design Goals

Ongoing work, intend to publish at ICN 2016.

1. **First do no harm!**
   – Work with reliable and unreliable traffic
2. Don't rely on Timers
   - Avoid packet drops
   - Use explicit congestion notification
   - Use timeouts as backup with really high values (e.g. 2 seconds)
3. Exploit multipath routing
   - Make decisions hop-by-hop
   - Use NDN forwarding to "forward around congestion"
4. Don't use per-route labels
5. Consider overlay and dual stack scenario
6. Consider caching