Congestion Control For Coded TCP

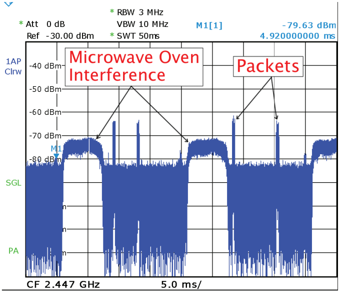
Doug Leith

Outline

- Why do error-correction coding at the transport layer ?
- Congestion control on lossy paths
- Implementation & performance measurements

Why error-correction coding at the transport layer ?

- Interference-related losses are challenging, yet increasingly common in dense 802.11 deployments
- Hidden terminals
- New dynamic devices e.g. channel bonding, will only make this worse
- Microwave interference in unlicensed band

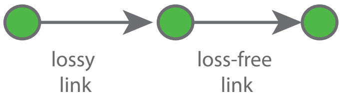


Why error-correction coding at the transport layer ?

Why not enhance error-correction at the link layer ? Link layer offers many advantages:

- Link layer has access to low-level information e.g. whether a packet loss is due to queue overflow or channel error.
- Usually quick feedback, so ARQ efficient
- Hop by hop encoding is generally more efficient than end-to-end encoding

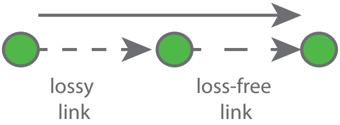
If link layer improvements are possible, make them !



Why error-correction coding at the transport layer ?

Transport layer has some compelling practical advantages:

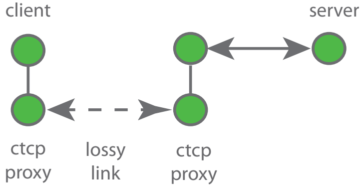
- No need for changes to installed network equipment. Recent estimate is 1.2 billion wireless devices shipped to date.
- What about servers, and especially user equipment ?



Why error-correction coding at the transport layer ?

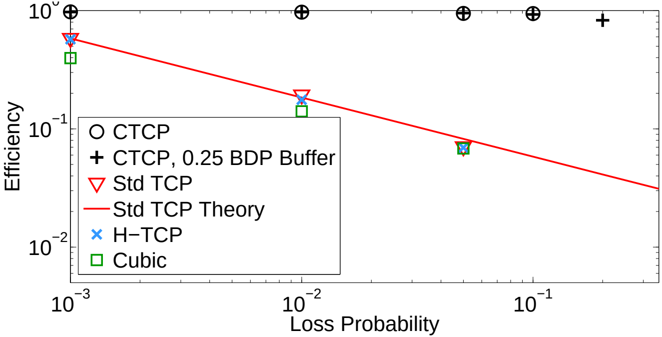
Transport layer has some compelling practical advantages:

- No need for changes to installed network equipment
- No need for root-privilege changes to user equipment, just a user-space app
- No need for changes to installed servers



Why error-correction coding at the transport layer ?

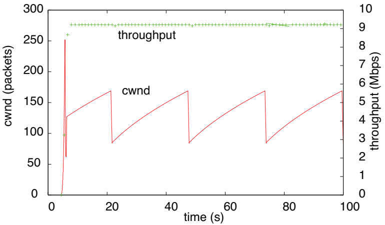
Plus potential exists for considerable performance gains.



Link 25Mbps, RTT 20ms

Congestion control on lossy paths

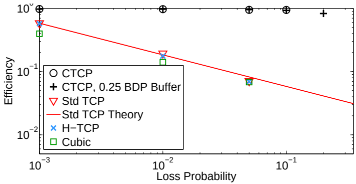
- Standard congestion control approach is window based using AIMD
- *cwnd* number of unacknowledged packets in flight
- $cwnd \leftarrow cwnd + 1$ every RTT without loss
- $cwnd \leftarrow cwnd/2$ on loss



Reno 10Mbps, 100ms RTT, buffer 1×BDP.

Congestion control on lossy paths

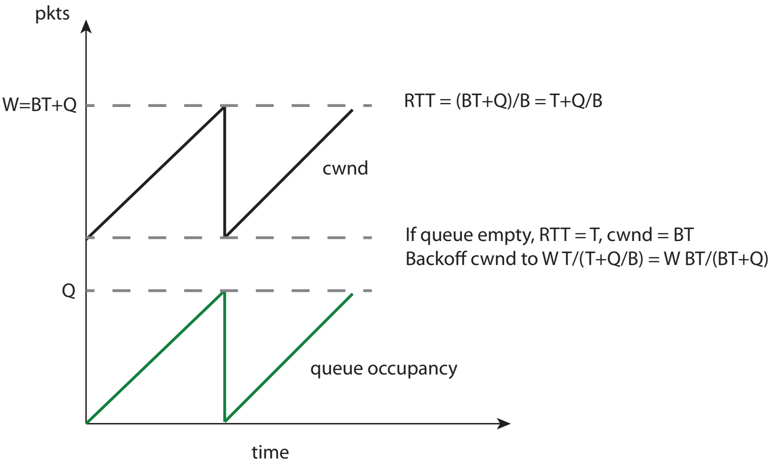
- Standard approach assumes loss = congestion
- On lossy path, backoff on loss means *cwnd* collapses to a low value
- Low throughput, many timeouts



Link 25Mbps, RTT 20ms

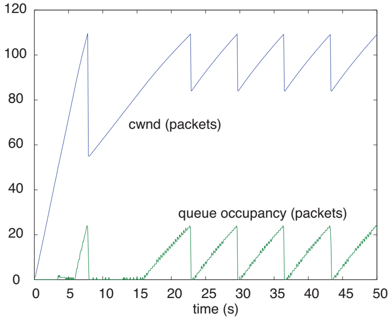
Congestion control on lossy paths

Modify AIMD backoff on loss to $cwnd \leftarrow cwnd \times \frac{RTT_{min}}{RTT}$



Congestion control on lossy paths

- Modify AIMD backoff on loss to $cwnd \leftarrow cwnd \times \frac{RTT_{min}}{RTT}$
- Never ignores packet loss
- Reverts to standard TCP on links without noise losses
- On lossy links yields dramatic improvement in throughput by avoiding *cwnd* collapse.
- An important source of the x10-x20 thoughput gains observed.



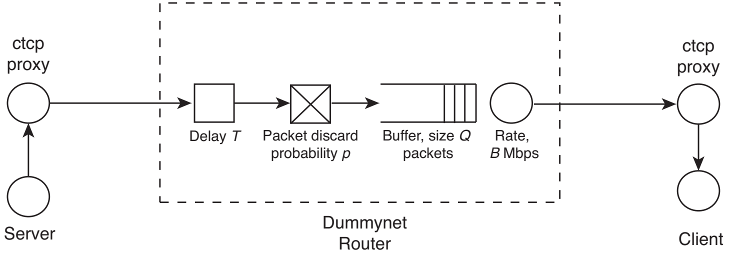
Congestion control on lossy paths

- A hybrid loss-delay approach, well established cf Compound TCP
- Builds on earlier work in H-TCP, well tested over a wide range of network conditions
- Known issues 1. Does not distinguish between:
  1. A congested lossy link (where have packet losses and the RTT is much higher than the base propagation delay)
  2. A non-congested lossy link where the base RTT fluctuates. How common are such links ?

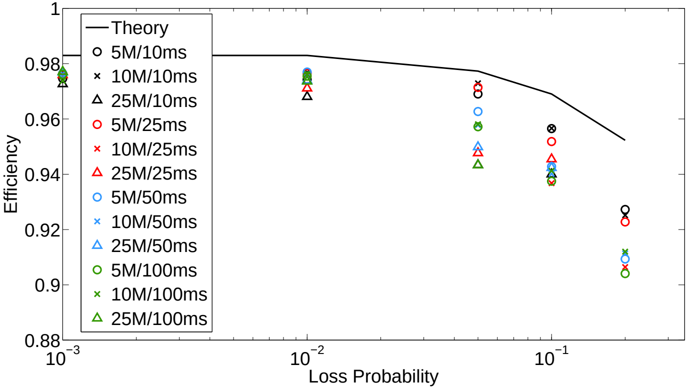
  Takes prudent approach and assumes (i) i.e. reduces send rate.
- Known issues 2. On paths with a standing queue, can be difficult to observe RTT_{min}. How common are such links ?

Link layer-agnostic measurements

Testbed setup:

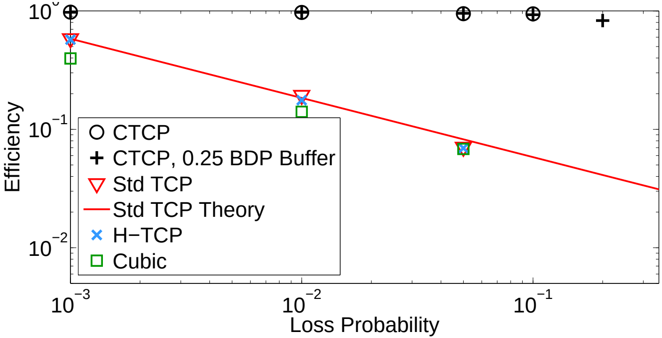


Link layer-agnostic measurements



Link 25Mbps, RTT 20ms

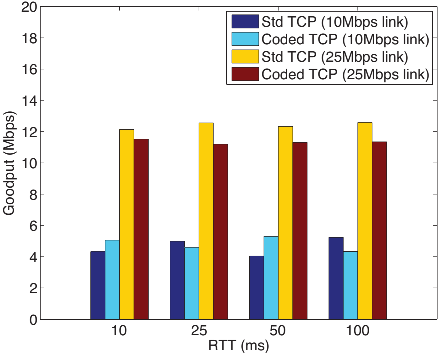
Link layer-agnostic measurements



Link 25Mbps, RTT 20ms

Link layer-agnostic measurements

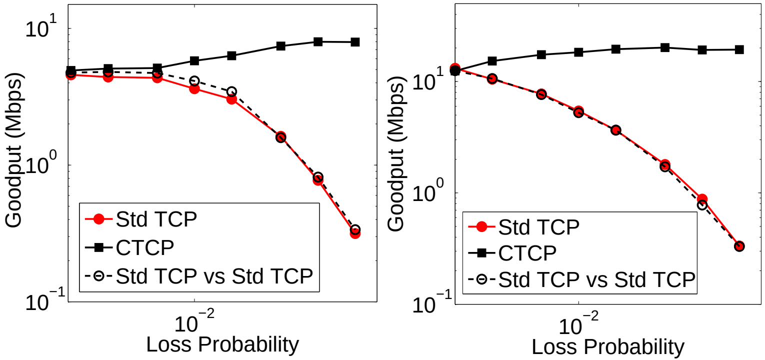
Friendliness - loss-free path



Standard TCP and a CTCP flow sharing a loss-free link

Link layer-agnostic measurements

Friendliness - lossy path

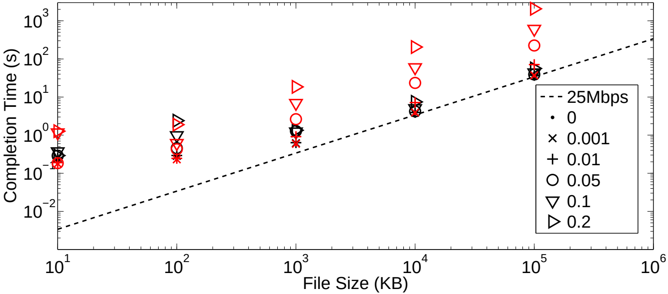


10Mbps link, RTT=25ms 25Mbps link, RTT=25ms

TCP and CTCP flow sharing link (solid lines), and (ii) two TCP flows sharing link (dashed line).

Link layer-agnostic measurements

Application performance - HTTP

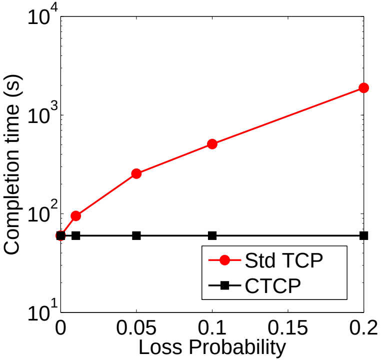


25Mbps link, RTT 10ms

Standard TCP (red) and CTCP (black)

Link layer-agnostic measurements

Application performance - Video streaming

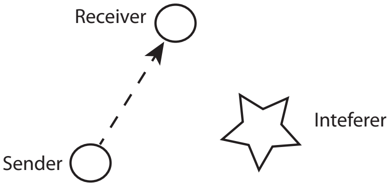


25Mbps link, RTT 10ms, 60s video playout
Standard TCP (red) and CTCP (black)

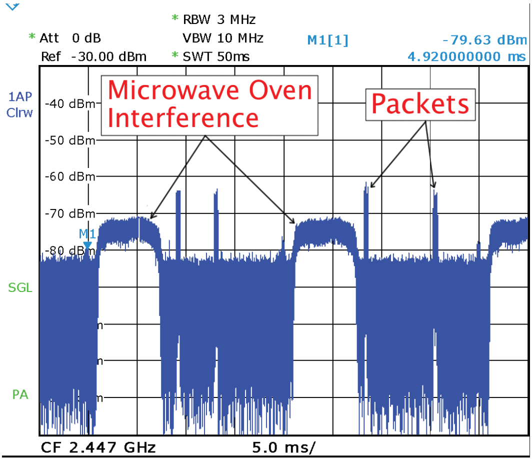
802.11 wireless measurements

Testbed setup:

- Proprietary 802.11 featrures disabled
- 802.11 rate control manual
- Cubic as standard TCP.

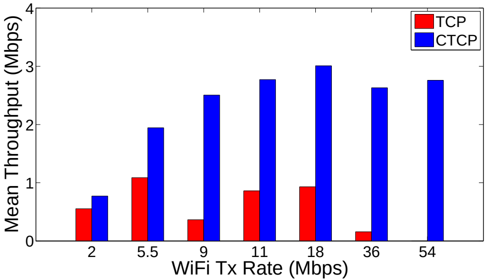


802.11 wireless measurements



802.11 wireless measurements

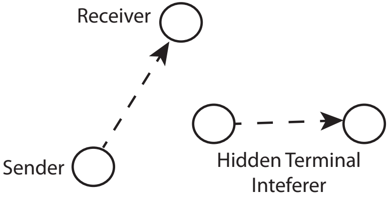
Microwave oven interference



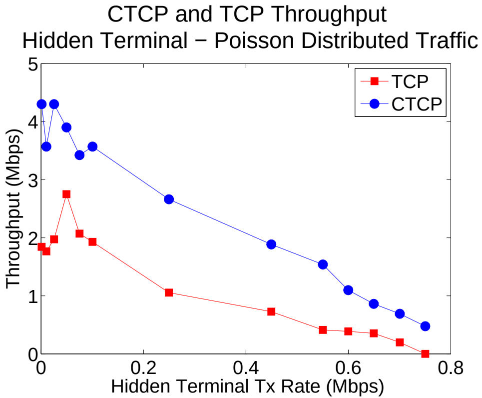
802.11 wireless measurements

Hidden terminal setup:

- Modified 802.11 driver to disable carrier sense
- Poisson interference traffic



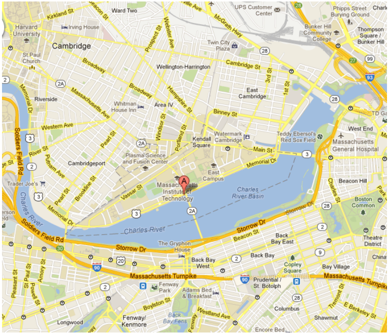
802.11 wireless measurements



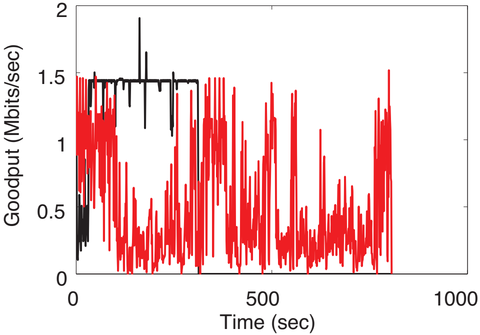
Throughput vs intensity of hidden terminal interference when using standard TCP (Cubic TCP) and CTCP over an 802.11b/g wireless link.

802.11 hot spot measurements

- Various public WiFi networks in the greater Boston area
- Downloaded a 50 MB file from a server located on MIT campus to a laptop
- Default operating system (Ubuntu) settings are used for all network parameters on client and server.



802.11 hot spot measurements



Standard TCP (red), CTCP (black)

Sheraton Hotel, Needham.

No link loss

Standard TCP, 5% loss

Coded TCP, 5% loss

23rd May 2013, MAC OS X 10.7.4