Transport layer interactions of adaptive HTTP streaming

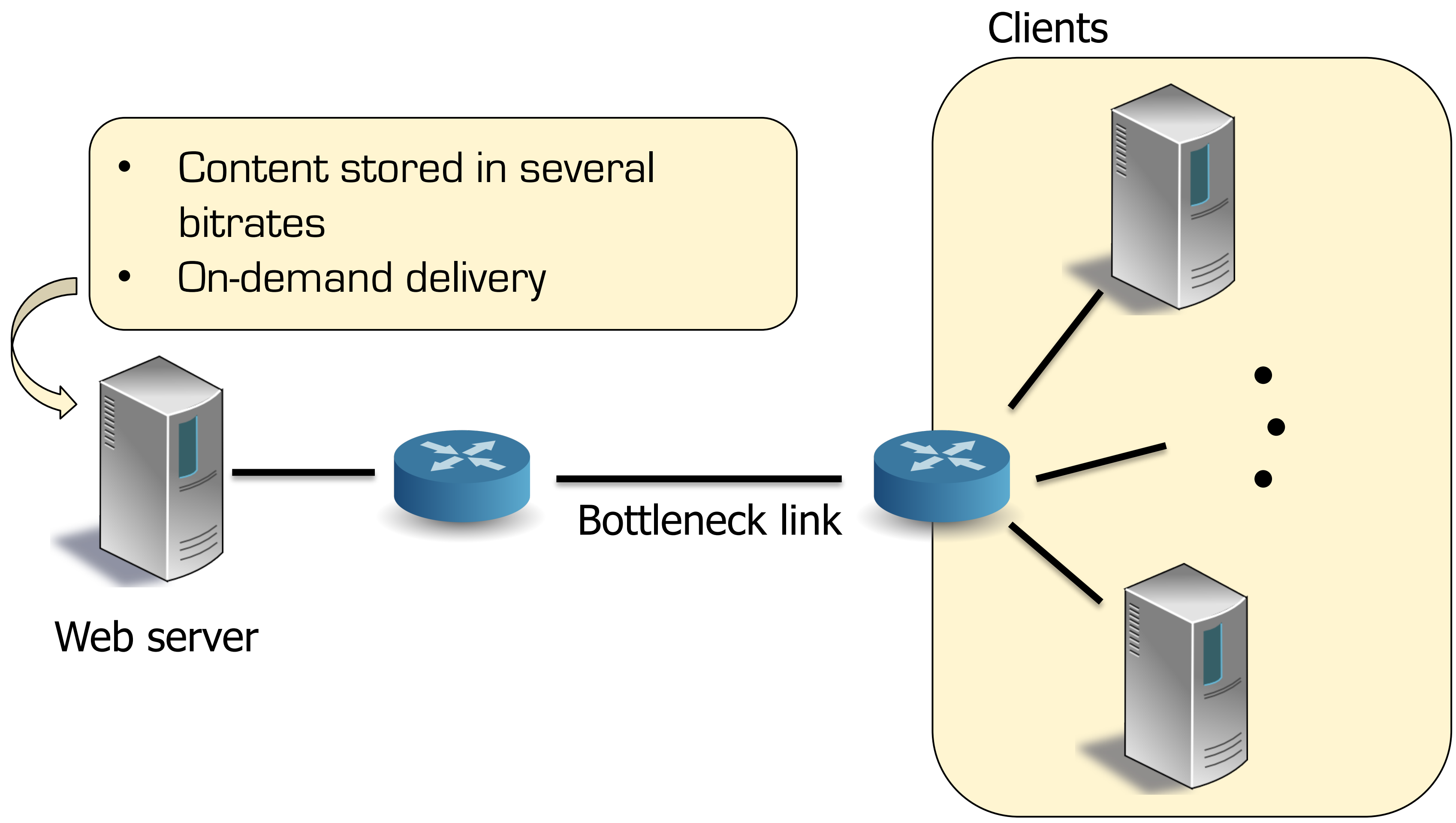
Carsten Griwodz

Simula Research Laboratory, Lysaker, Norway

and

University of Oslo, Oslo, Norway

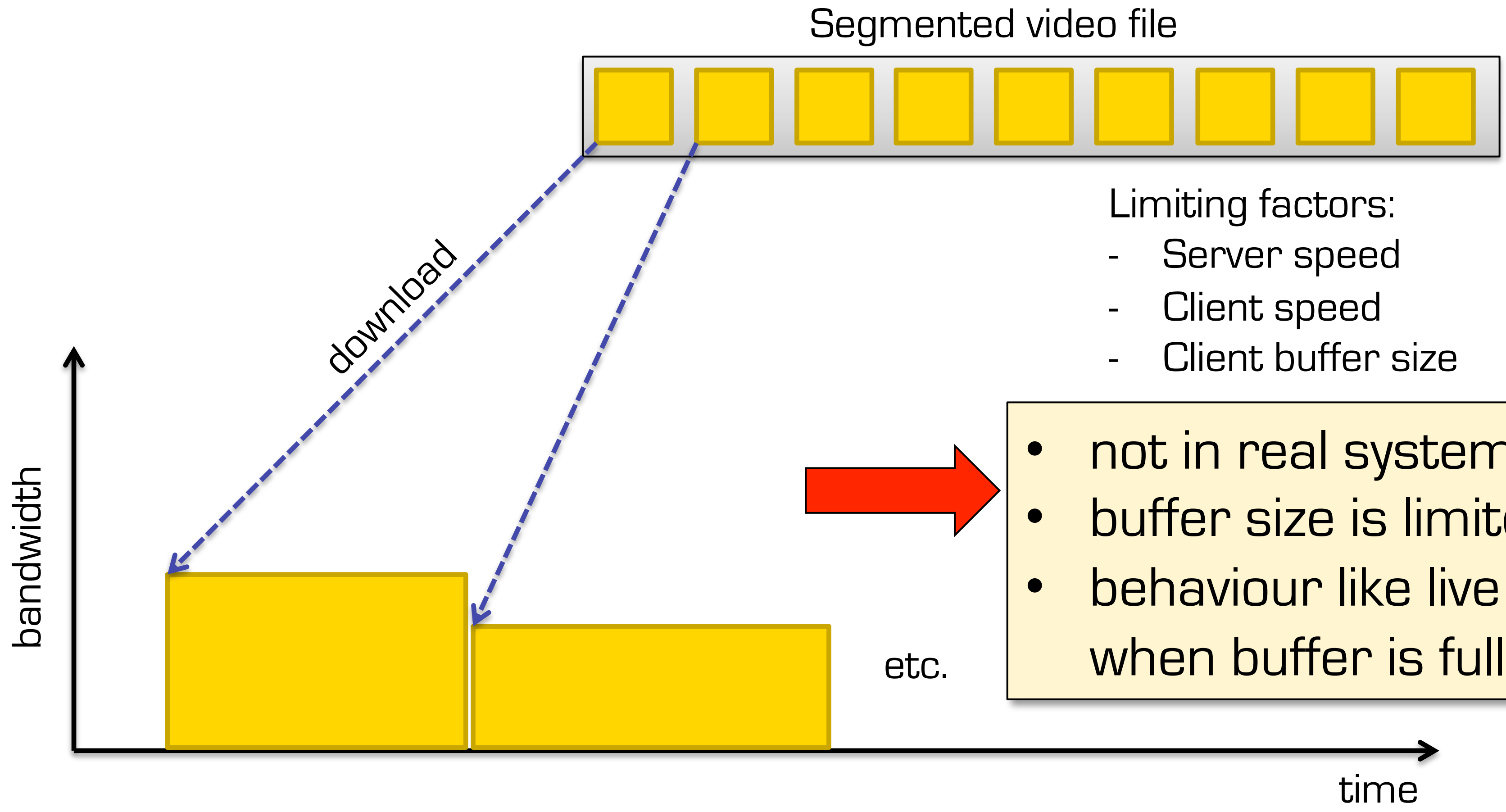
Adaptive HTTP Streaming



In Adaptive HTTP Streaming, bottlenecks don't need to be thin to degrade quality!

VoD traffic pattern

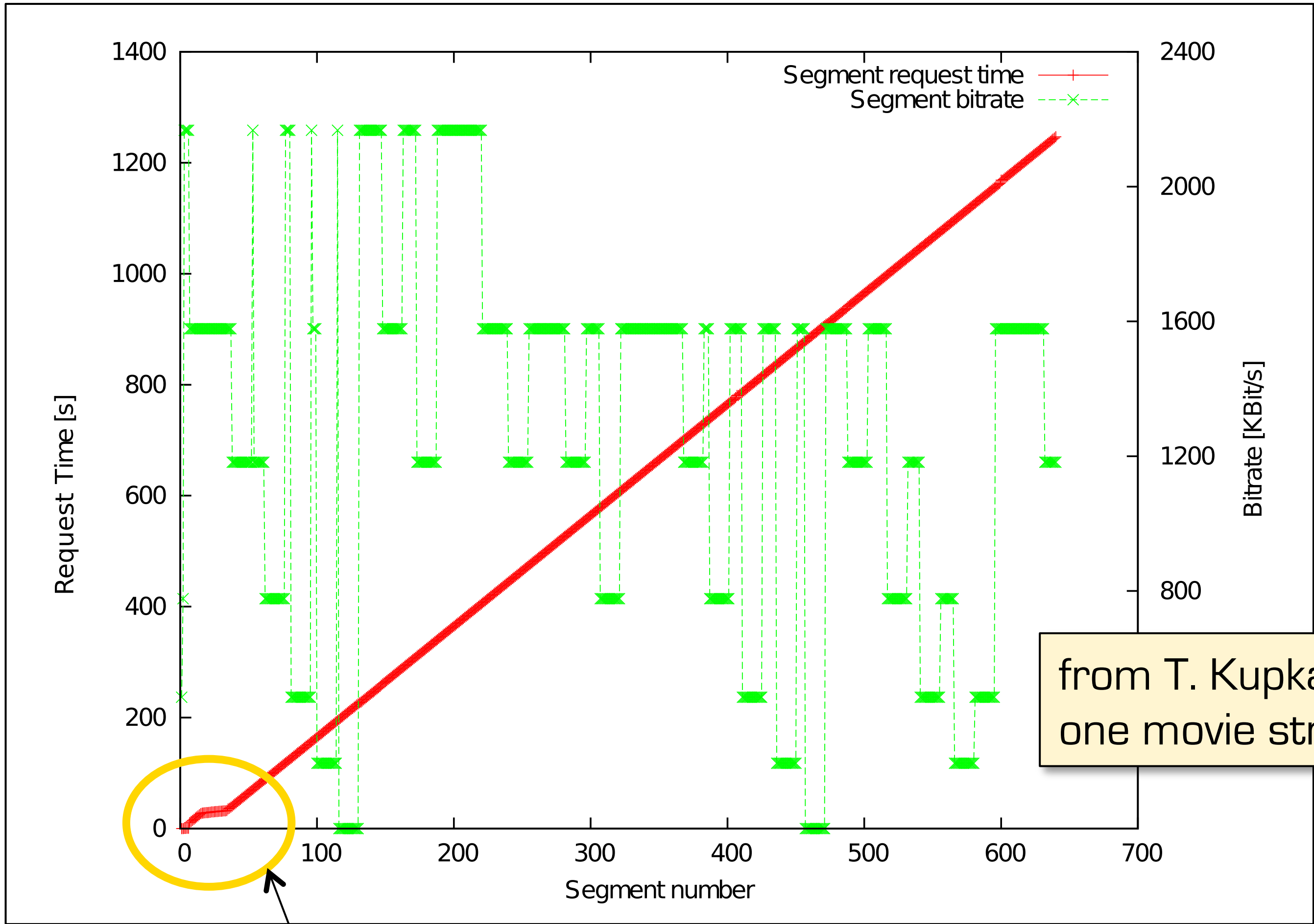
HTTP pipelining should share server-side bottlenecks fairly



Limiting factors:
- Server speed
- Client speed
- Client buffer size

- not in real systems
- buffer size is limited
- behaviour like live when buffer is full

VoD traffic pattern

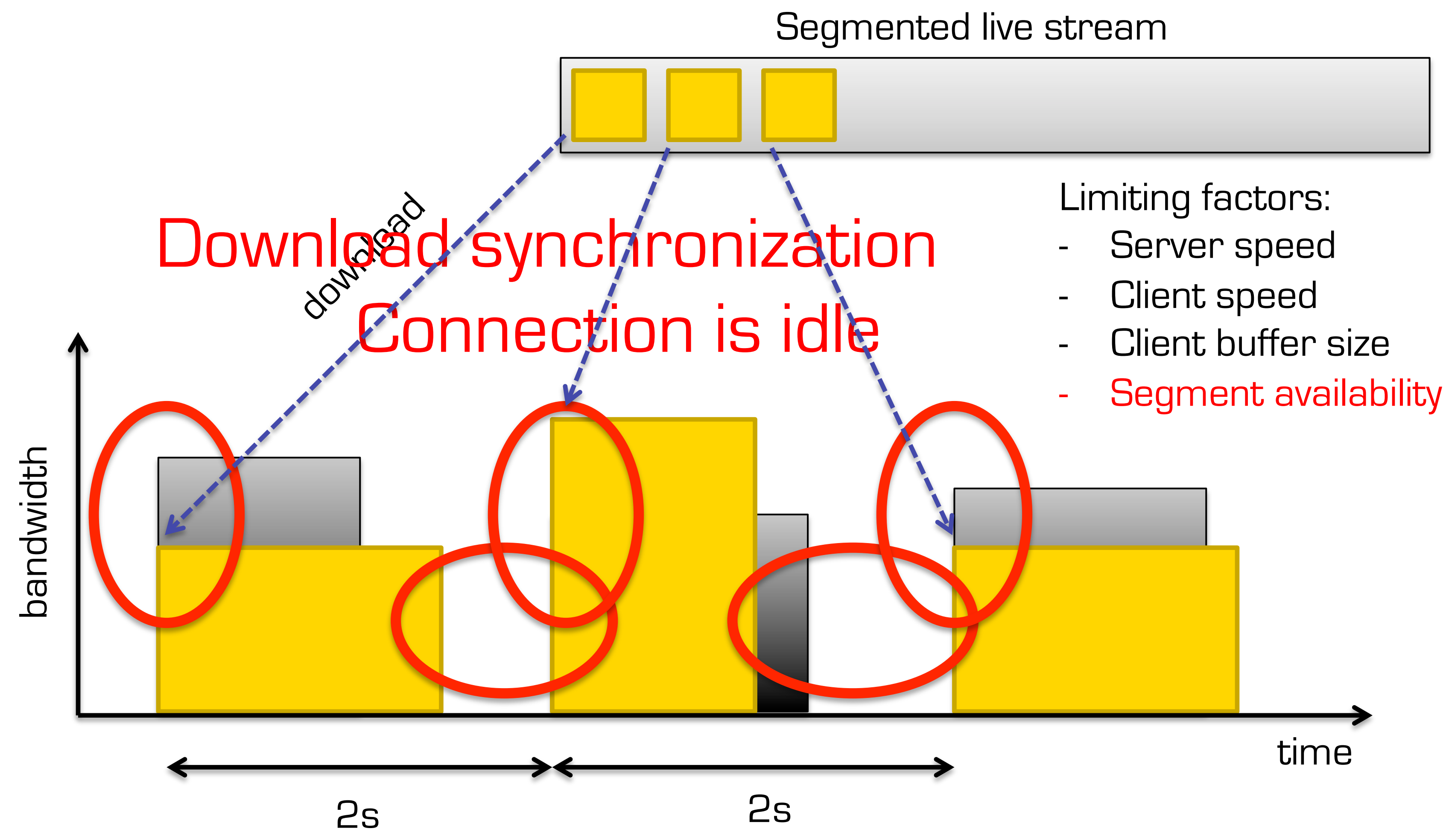


from T. Kupka (2013):
one movie streamed from ComoYo

Filling the buffer

"On the HTTP segment streaming potentials and performance improvements",
Tomas Kupka,
PhD Thesis, University of Oslo, June 2013

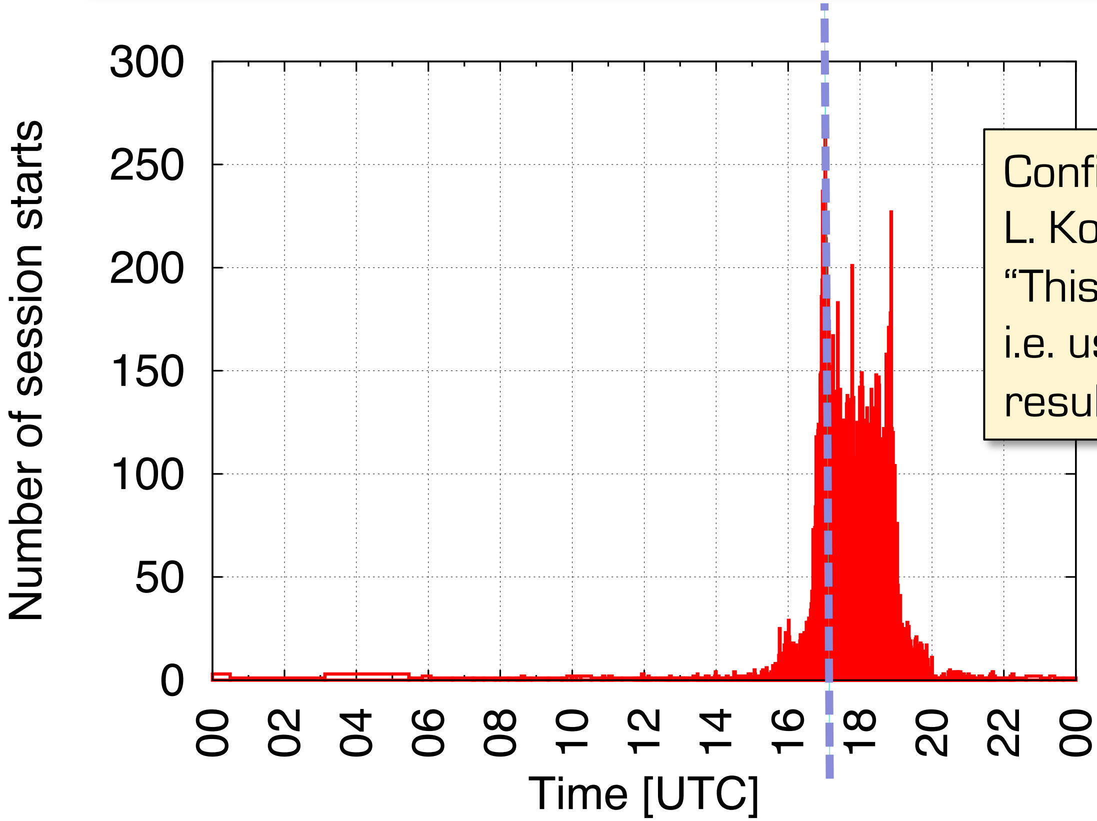
Live traffic pattern



Limiting factors:
- Server speed
- Client speed
- Client buffer size
- Segment availability

Live traffic pattern

from T. Kupka (2013:E):
Number of sessions that were started per minute (live event starting at 17:00)



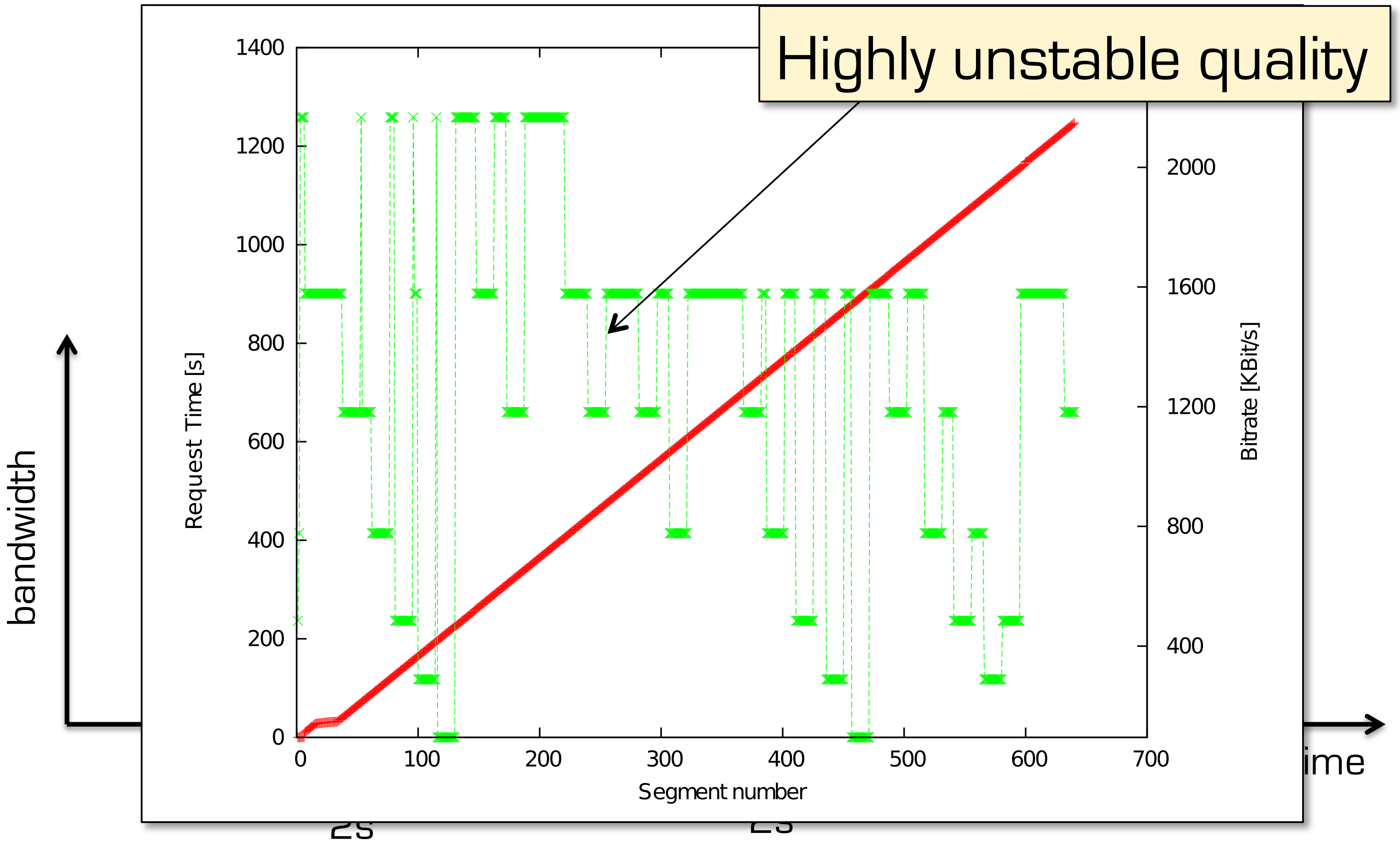
Confirms a major problem identified by L. Kontothanassis (2012):
"This video will start soon syndrome",
i.e. users arrive before start and wait,
resulting in a synchronized start burst

"Analysis of a Real-World HTTP Segment Streaming Case",
T.Kupka, C.Griwodz, P.Halvorsen, D.Johansen, T.Hovden
EuroITV 2013

"Content Delivery Considerations for Different Types of Internet Video",
Leonidas Kontothanassis,
Keynote, ACM MMSys 2012

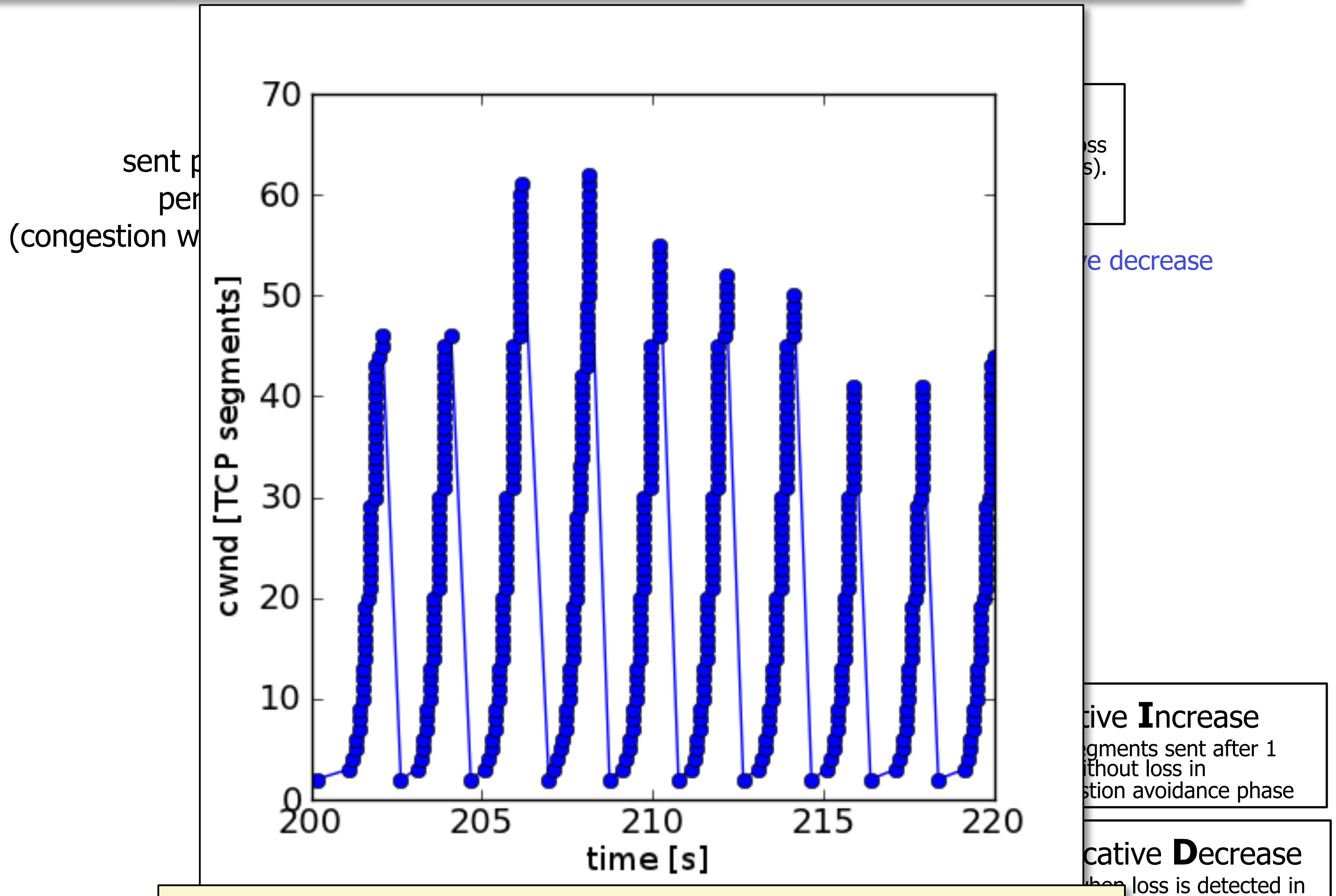
Traffic patterns

Why should we care?



Bottleneck behaviour

TCP congestion window development: 2 sec segments



connection hardly ever leave slow start

Bottleneck behaviour

What do we know?

The server application dispatches an entire segment at once (except Akamai server with matching client)

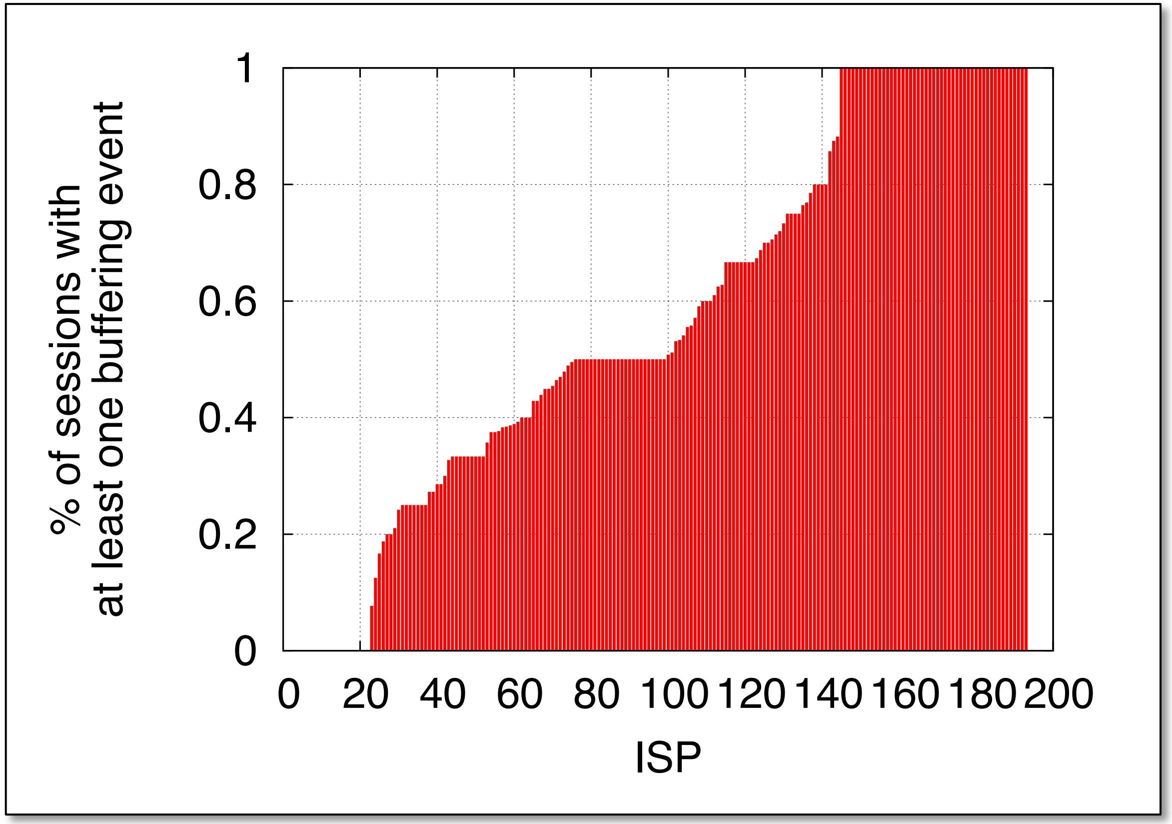
Order of burst arrival at the bottleneck queue changes due to self-clocking effects, jitter and RTT differences

A connection that manages to leave slow start is penalized: no return to slow start growth even when falling under previous threshold

Loosing in one RTT may force a client to introduce several seconds pre-buffering

Effects of Live Adaptive TCP Streaming

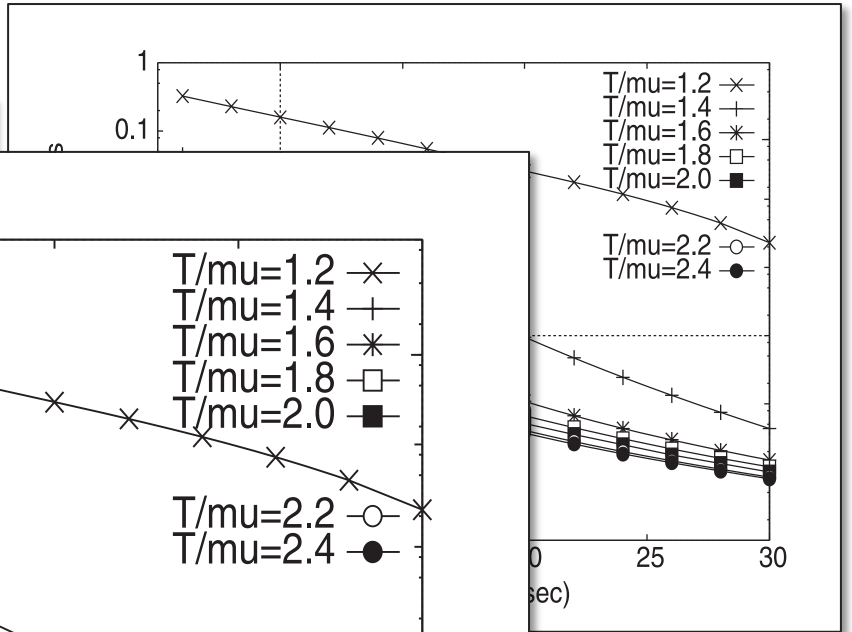
If the BW loss is sudden, and prefetch time must be increased, we experience buffer underruns ("hickups")

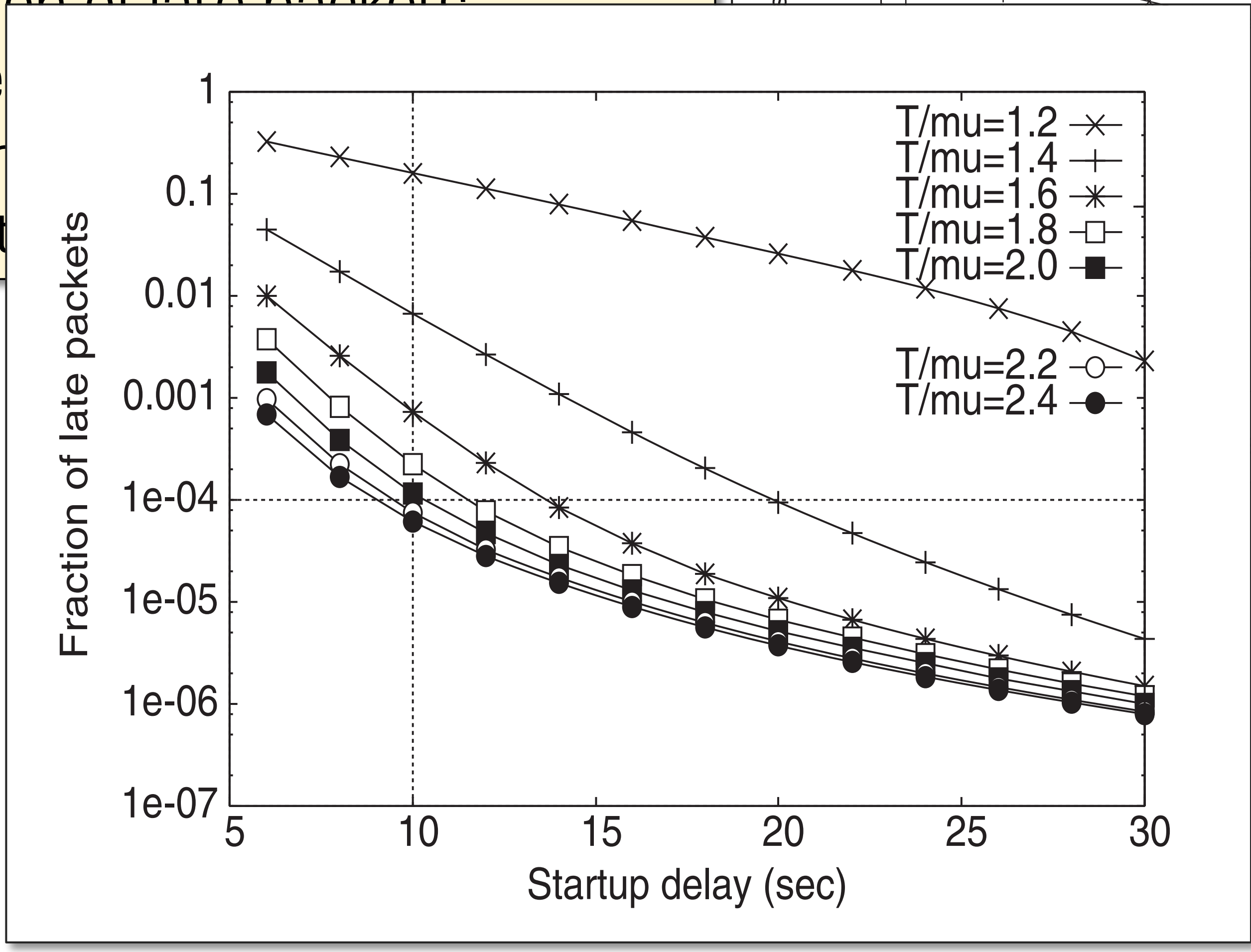


from T. Kupka (2013:E):
The percentage of sessions with at least one buffer underrun by ISP

Relation to the TCP streaming rule of thumb

the fracti
when the
(determi
is T/mu t

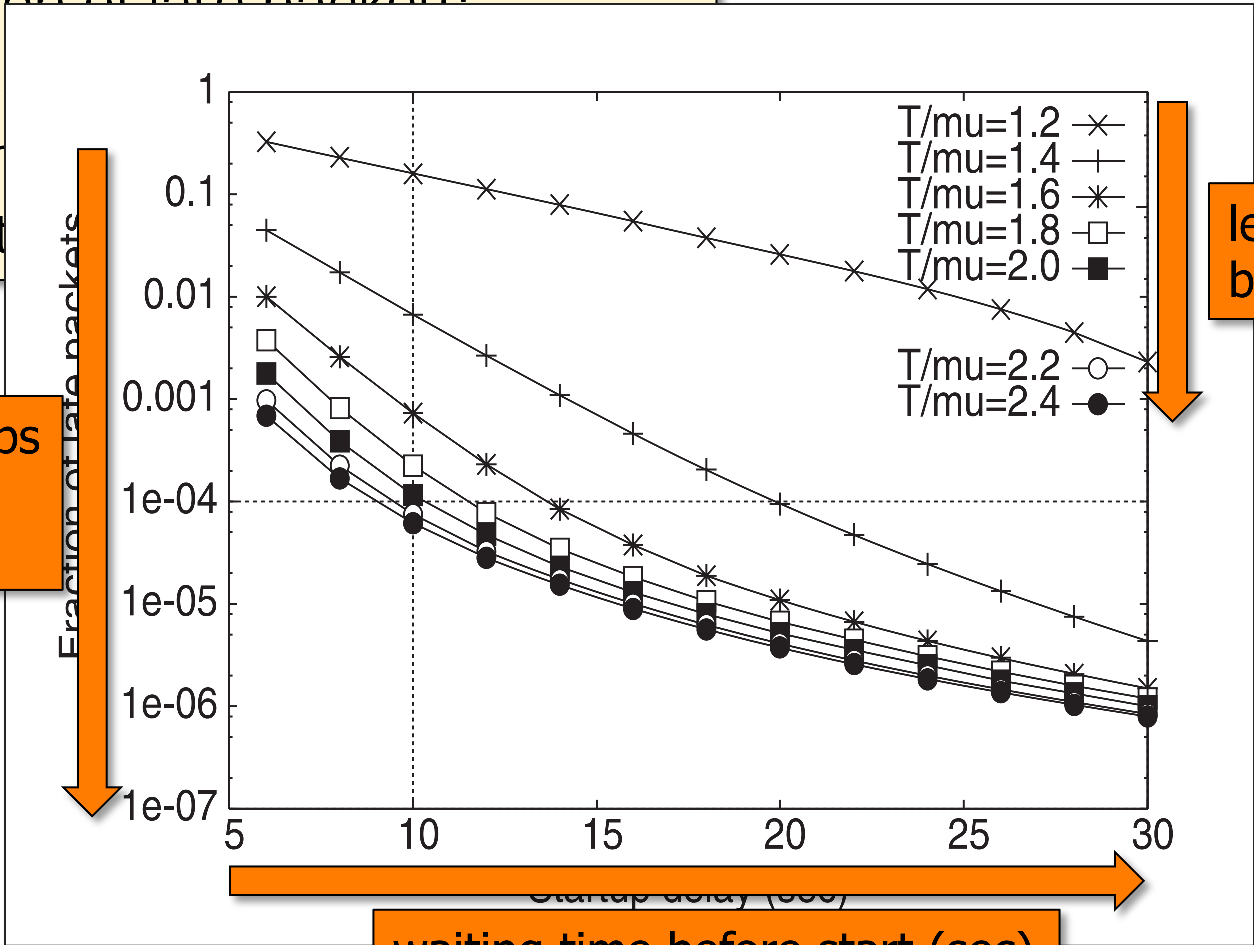




"Multimedia Streaming via TCP: An Analytic Performance Study",
B. Wang, J. Kurose, P. Shenoy, D. Towsley, *ACM TOMCCAP 4:2, 2008*

Relation to the TCP streaming rule of thumb

the fraction of late packets
when the
(determir
is T/mu t



fewer hickups in video playback

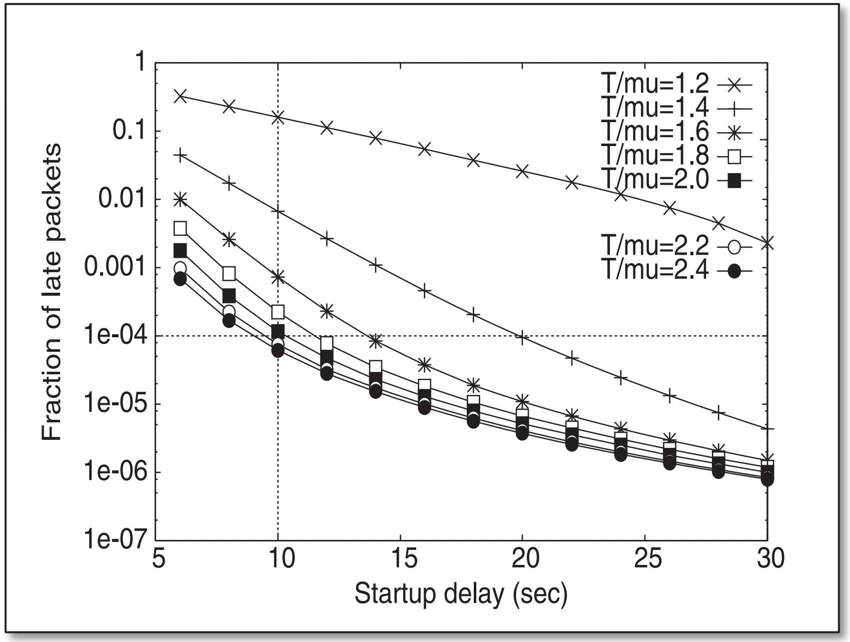
less effective bandwidth use

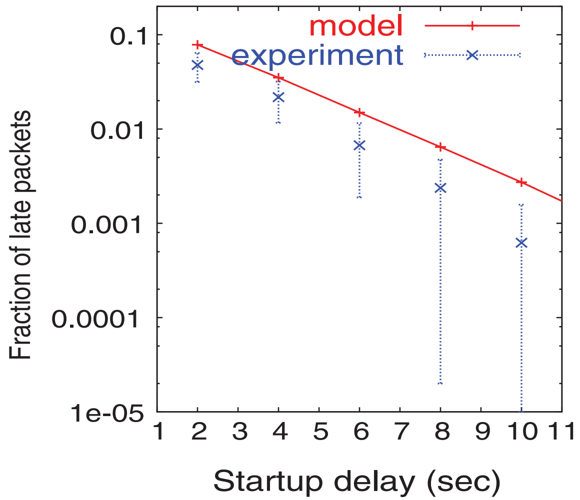
waiting time before start (sec)

"Multimedia Streaming via TCP: An Analytic Performance Study",
B. Wang, J. Kurose, P. Shenoy, D. Towsley, *ACM TOMCCAP 4:2, 2008*

Relation to the TCP streaming rule of thumb

the fraction of late packets
when the available bandwidth
(determined by RTT)
is T/mu times the streaming rate





real-world verification

rule of thumb:
if 5 sec startup delay is acceptable,
available bandwidth of twice the
streaming rate is sufficient
(*cited in really many papers*)

"Multimedia Streaming via TCP: An Analytic Performance Study",
B. Wang, J. Kurose, P. Shenoy, D. Towsley, *ACM TOMCCAP 4:2, 2008*

Relation to the TCP streaming rule of thumb

Does this rule of thumb apply to adaptive HTTP streaming?

Not entirely

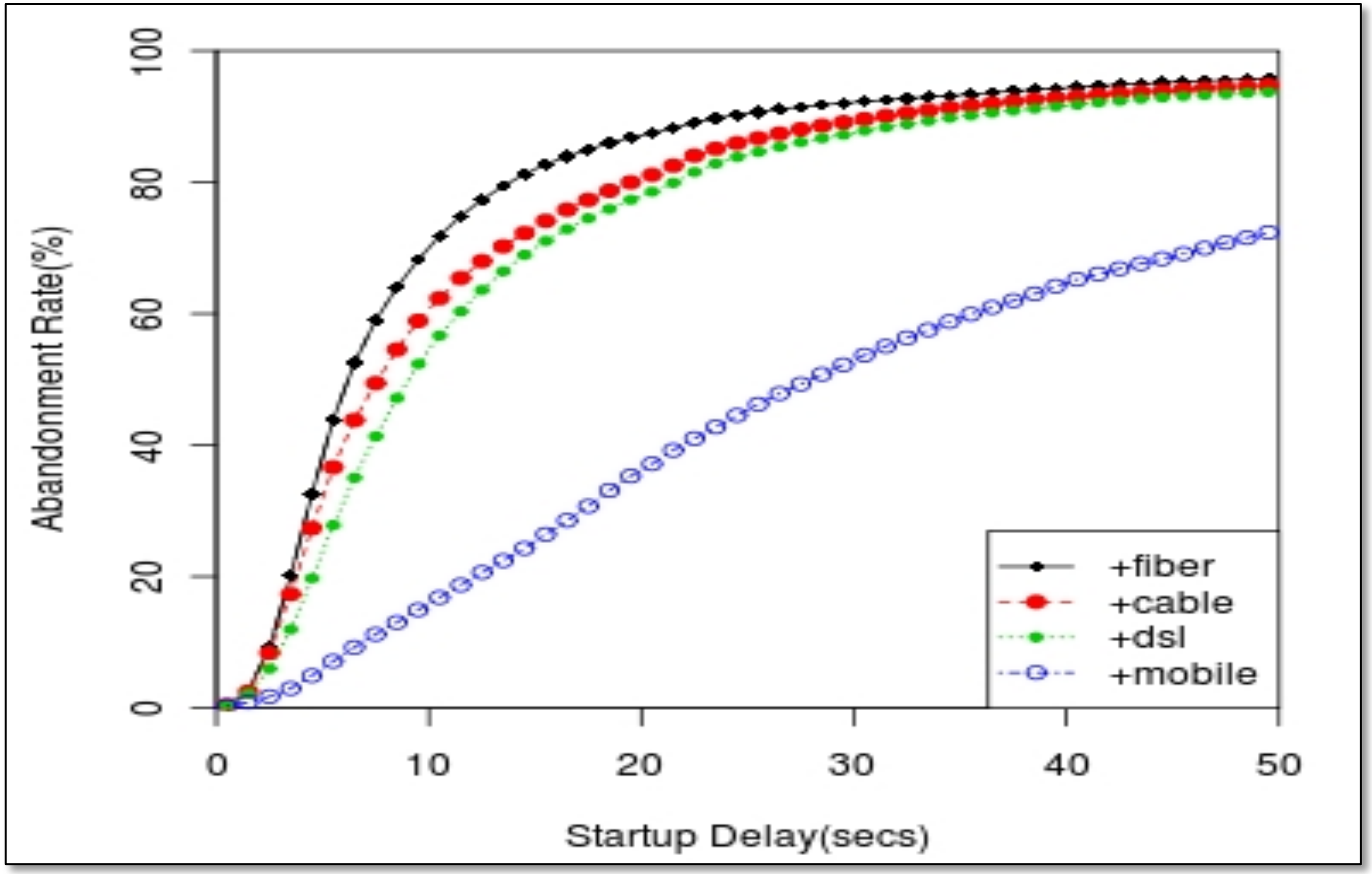
- Adaptive HTTP sessions may remain in slow start
- The server-sided bottleneck is not stable, the assumption of eventual fair sharing never holds
- An adaptive HTTP stream can revise its bandwidth decision every 2 seconds
- Hickups can be avoided by downscaling

Furthermore

The rule fails for 1 in 1000 segments
5sec pre-buffering may already be too much

Abandonment due to startup latency

Surprisingly high numbers of users abandon Akamai downloads



"Video Stream Quality Impacts Viewer Behavior: Inferring Causality using Quasi-Experimental Designs", S. Krishnan and R. Sitaraman, *ACM IMC 2012*

Abandonment due to startup latency

Based on a huge dataset:
- 10 days
- 12 content providers
- 102 000 videos
- 23 million views from three continents

This study shows when users gives up before seeing *anything*

Viewers with better connectivity have less patience for startup delay and abandon sooner

The study does not show whether users abandon after receiving low quality at low startup delay

No study considering startup-delay and quality in combination has been conducted yet

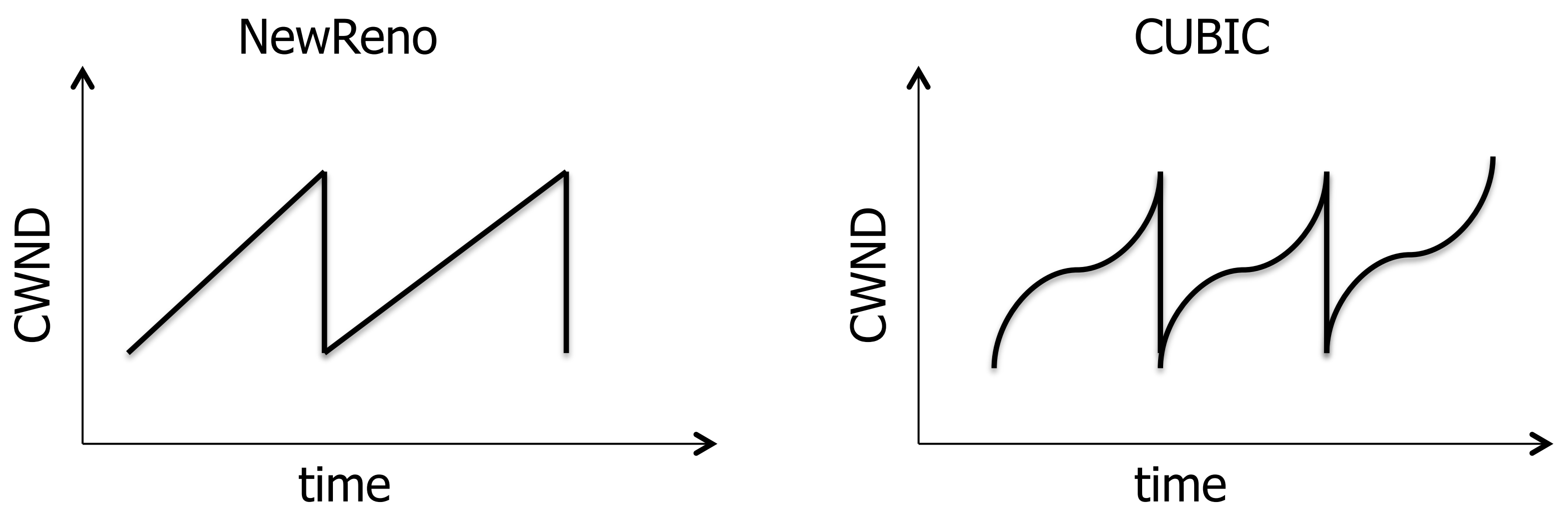
"Video Stream Quality Impacts Viewer Behavior: Inferring Causality using Quasi-Experimental Designs", S. Krishnan and R. Sitaraman, *ACM IMC 2012*

Choices for quality and liveness

Evaluated modifications

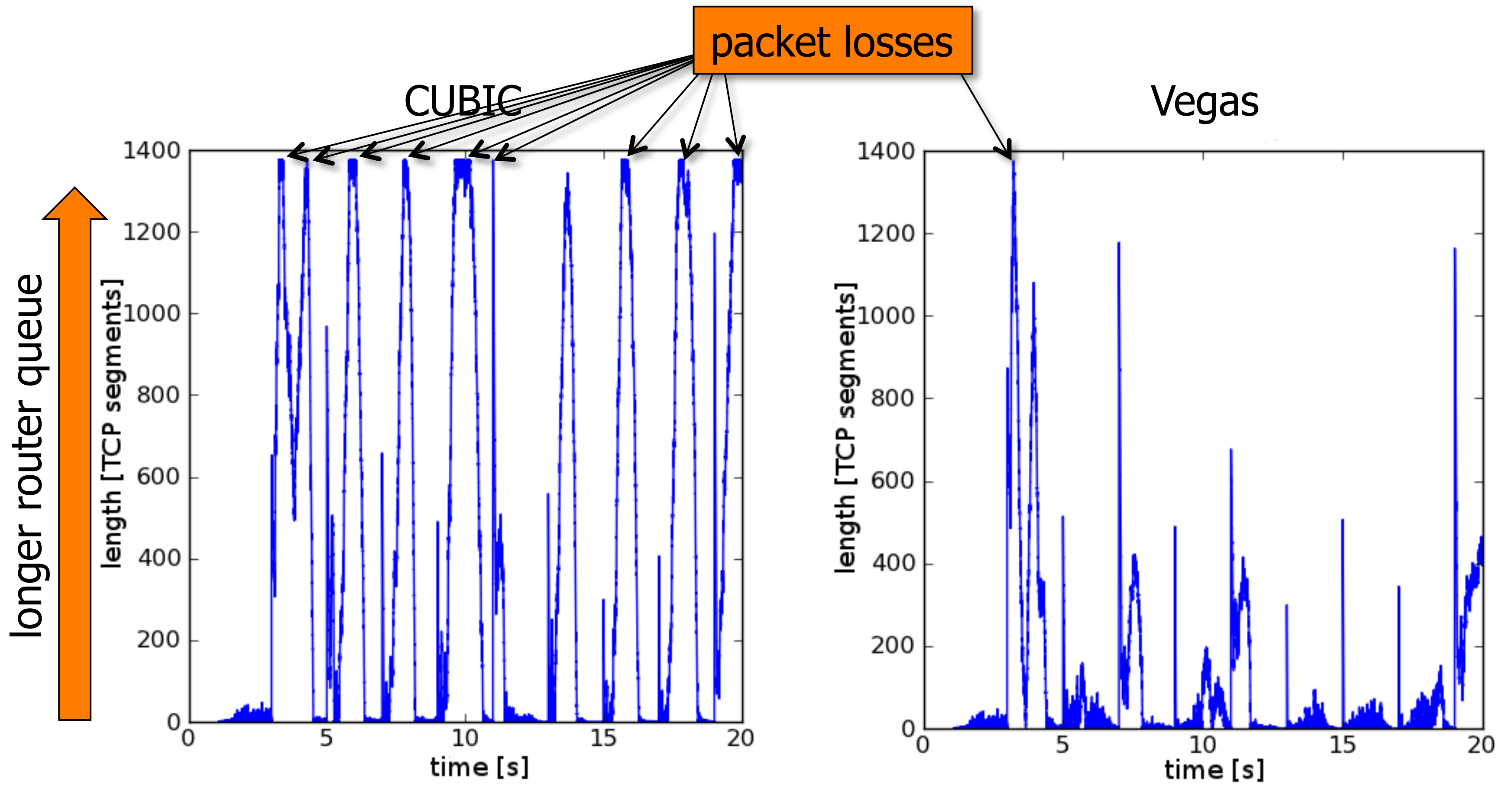
TCP congestion control alternatives

TCP exhibits an new set of problems due to a new kind of On-Off traffic



"Performance of On-Off Traffic Stemming From Live Adaptive Segmented HTTP Video Streaming",
T.Kupka, C.Griwodz, P.Halvorsen
IEEE LCN 2012

TCP congestion control alternatives



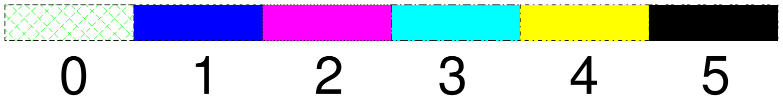
TCP Vegas shares the network much better than TCP CUBIC

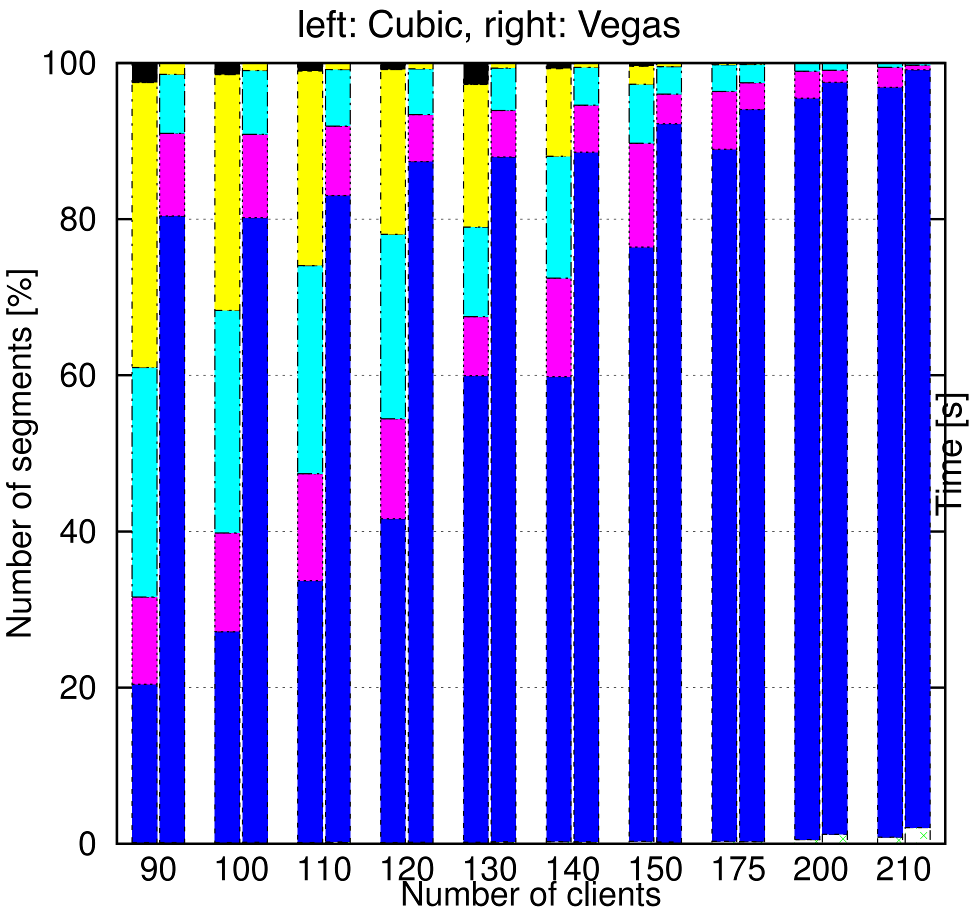
Unfortunately, TCP Vegas looses in sharing

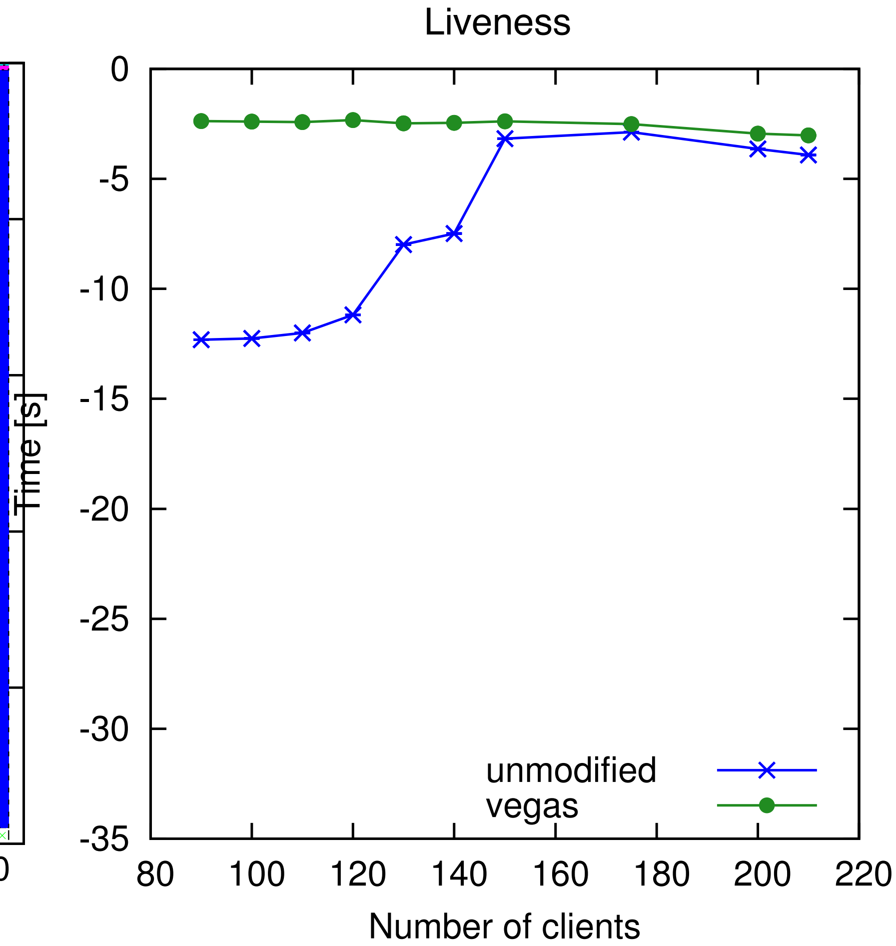
TCP congestion control alternatives

lowest quality

highest quality





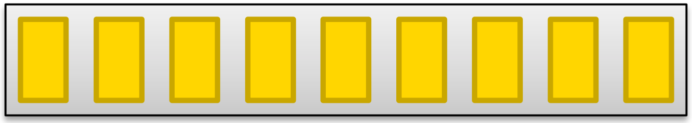


Cubic delivers higher quality

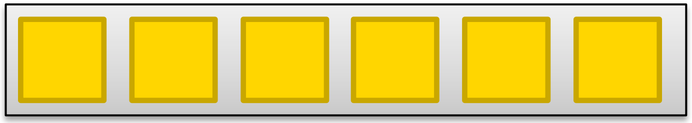
Vegas delivers better liveness

Evaluated modifications

Increasing segment length: give TCP more time



vs.



2 second segments

10 second segments

smooth HD

HTTP streaming

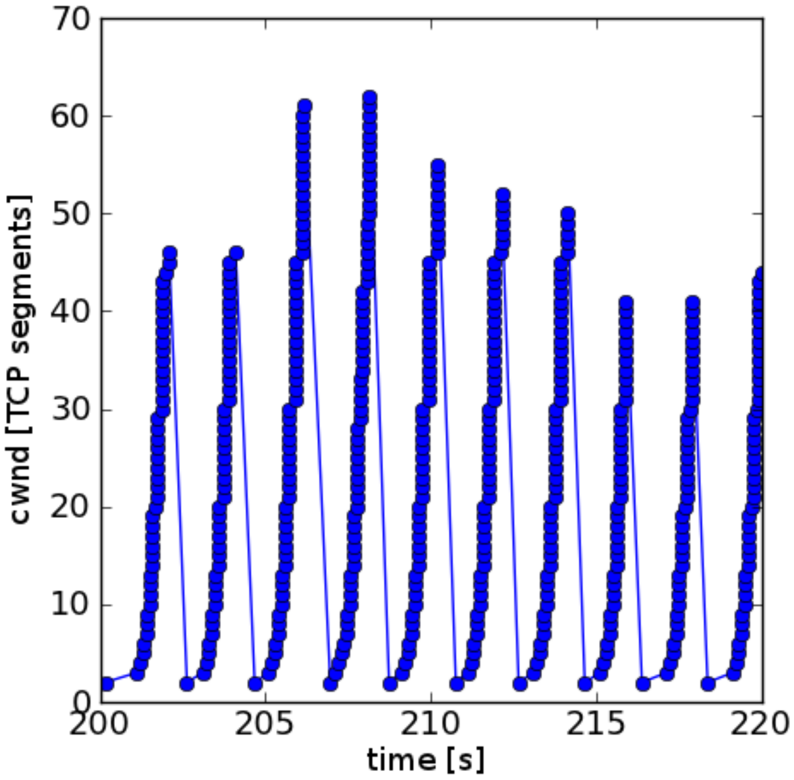
HTTP live streaming

Netview Technology

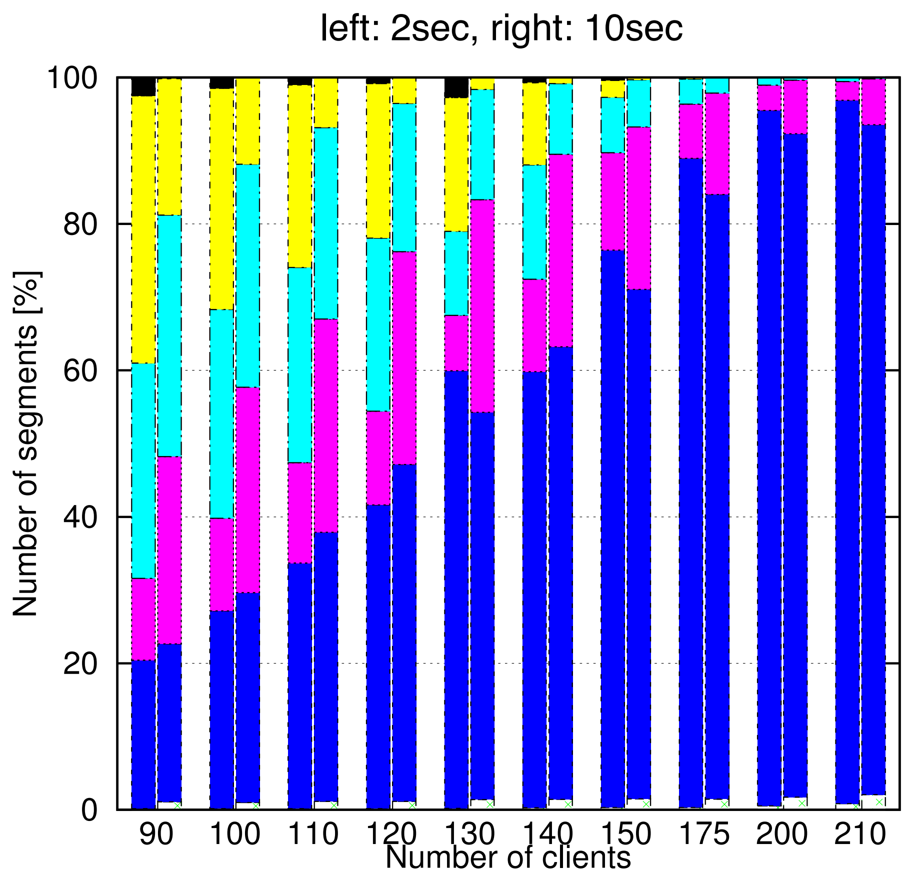
2 second vs. 10 second segments

lowest quality | 0 | 1 | 2 | 3 | 4 | 5 | highest quality

TCP CWND for 2 sec. segments

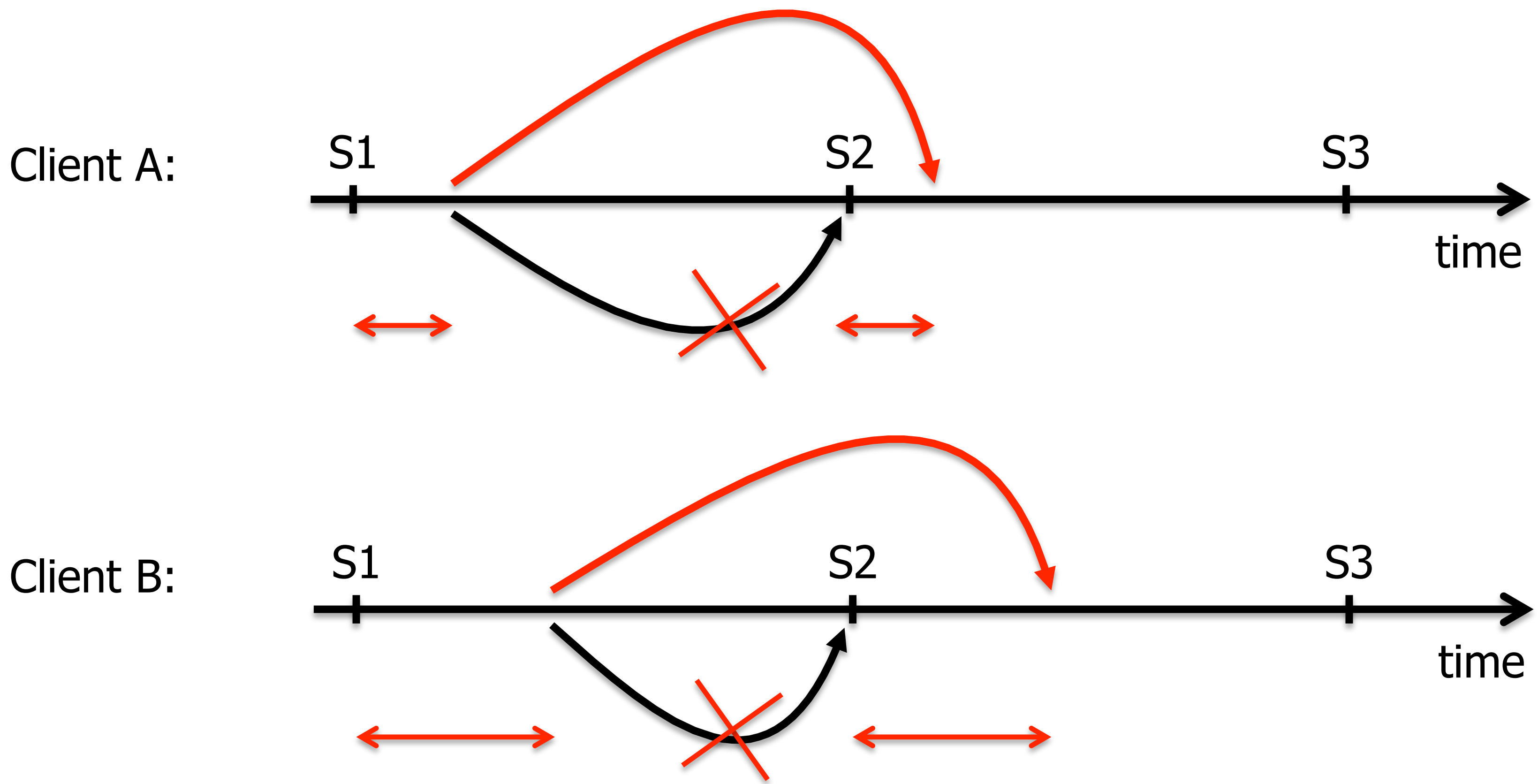


Quality not better for longer segments



Evaluated modifications

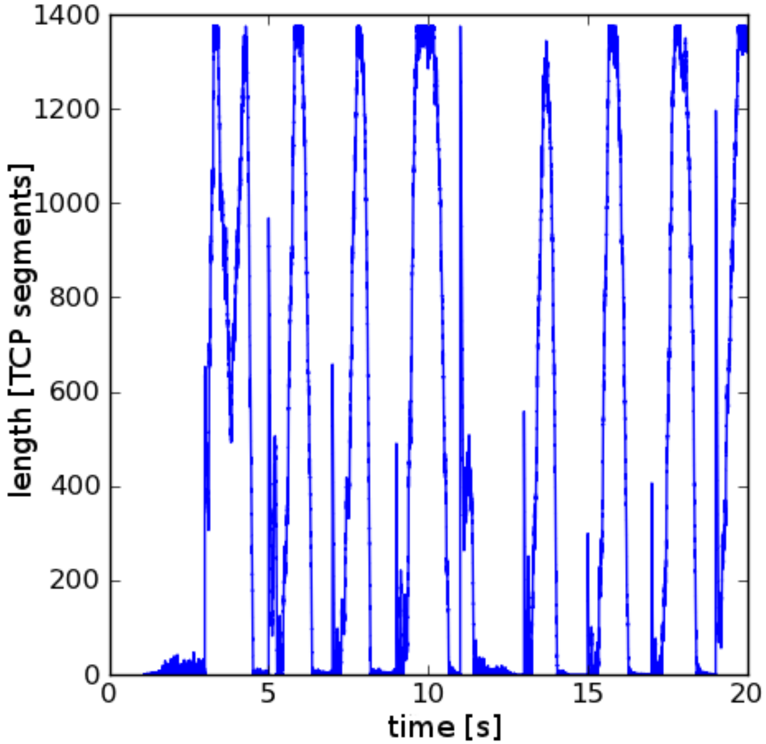
Client request distribution



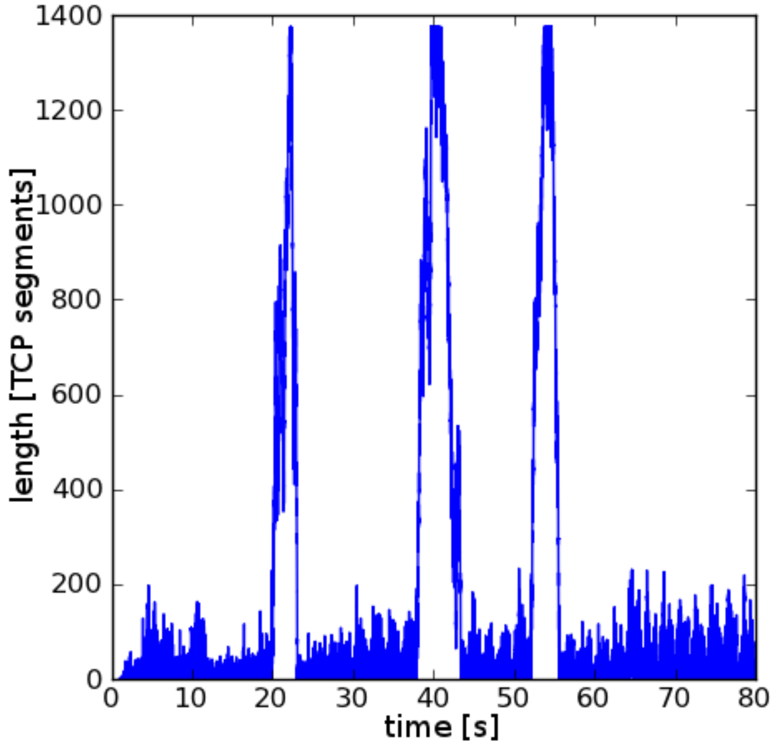
Client request distribution

Self-inflicted congestion: Only the slow start effects lead to queue overflow. Loss of goodput.

Synchronous requests



Distributed requests

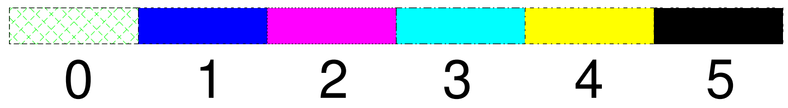


Uniform distribution maximizes goodput

This helps ***not at all*** if congestion is due to high total load.

Client request distribution

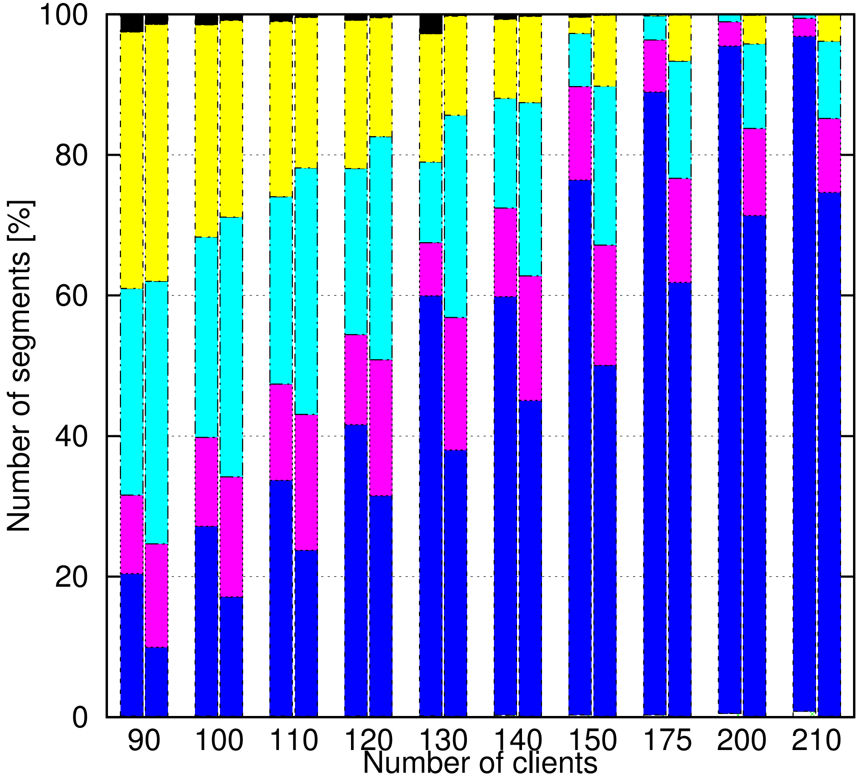
lowest quality



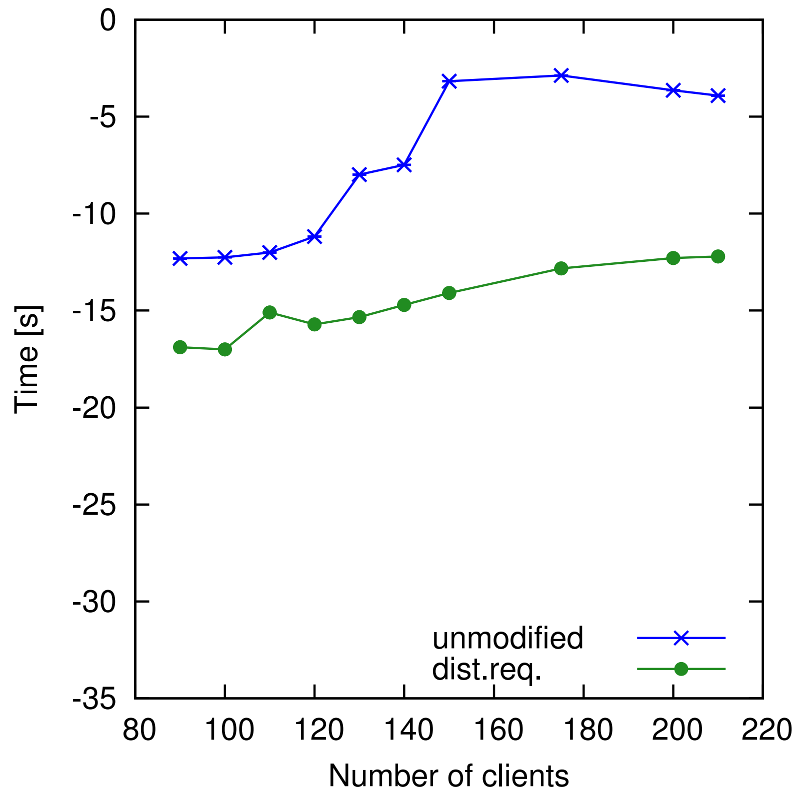
highest quality

Distribution of the number of segments of different quality

left: synchronized, right: distributed



Liveness



Recap

TCP CC alternatives: TCP Vegas is good, but not practical

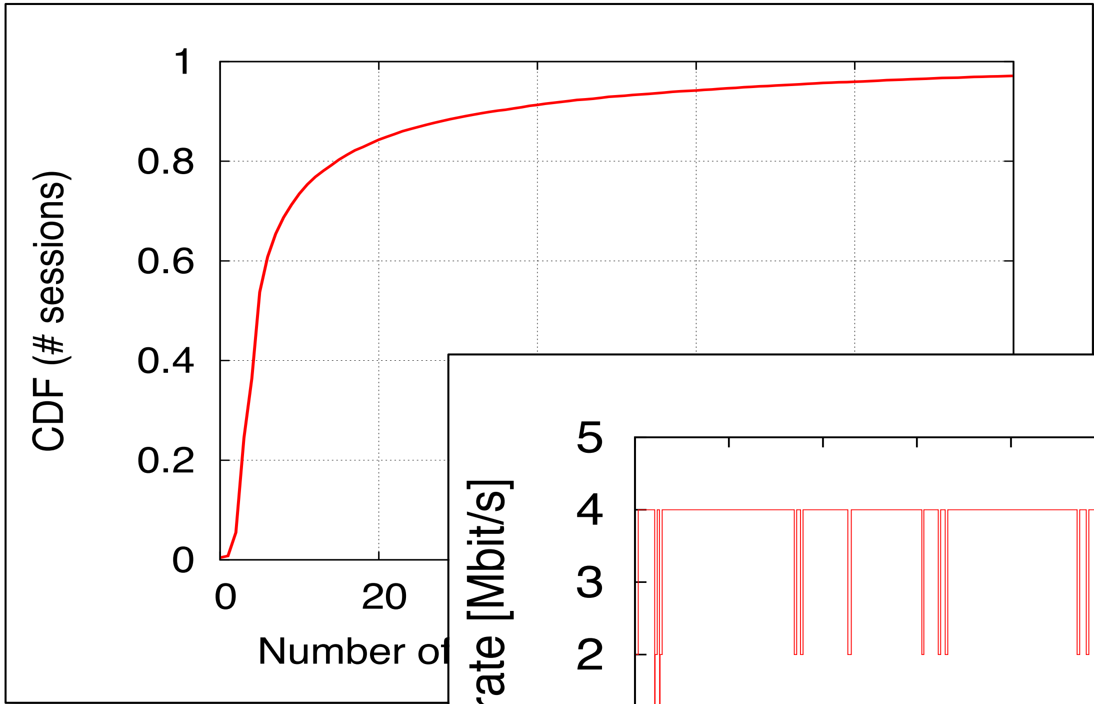
Segment duration:
No evidence of longer segments being better (network view)

Client request distribution leads to good quality and liveness

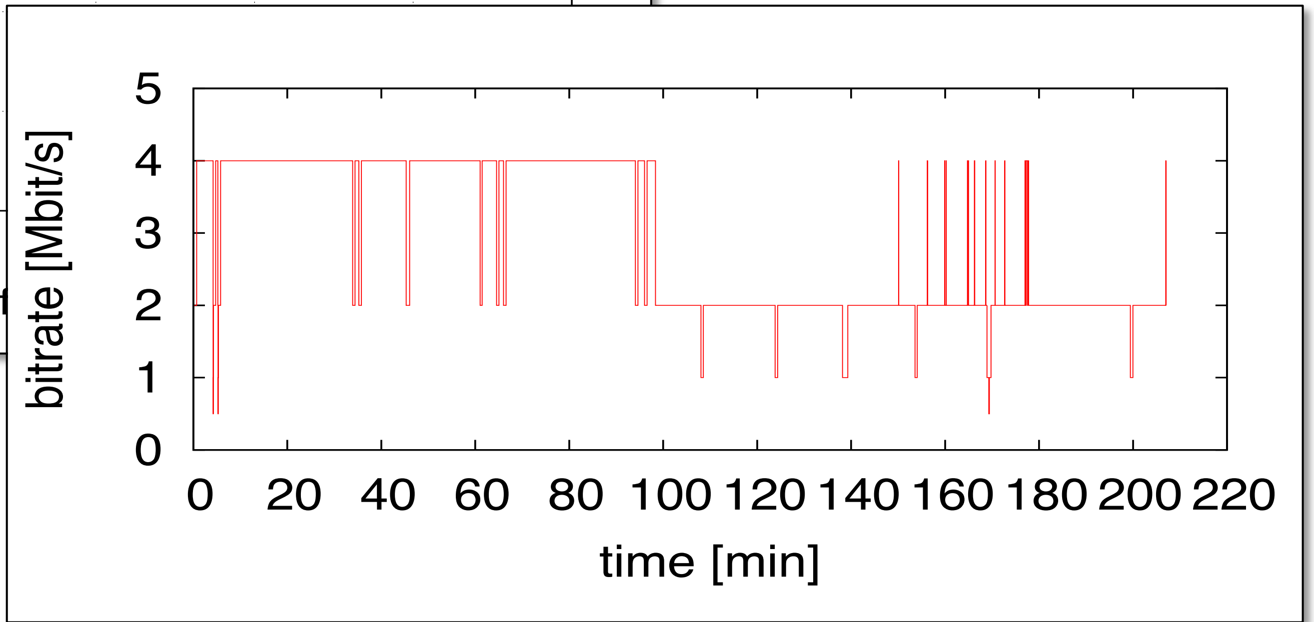
Smoothing techniques

Effects of Live Adaptive TCP Streaming

Number of bitrate switches per session



random session example



"Analysis of a Real-World HTTP Segment Streaming Case", T.Kupka, C.Griwodz, P.Halvorsen, D.Johansen, T.Hovden *EuroITV 2013*

Client-only techniques for long-term smoothness

Ni (2011) showed: Quality oscillations more frequent than 1Hz are perceived as flickering

Adaptive HTTP streaming is safe from this

Still, we have advocated long-term stable quality: not at all costs, but on longer time scales

Desirability apparently verified in perceptual studies
(sorry, no reference)

Other researchers do the same

"Flicker Effects in Adaptive Video Streaming to Handheld Devices",
Pengpeng Ni, Ragnhild Eg, Alexander Eichhorn, Carsten Griwodz, Pål Halvorsen
ACM Multimedia 2011

Client-only techniques for long-term smoothness

Netview adaptive strategy

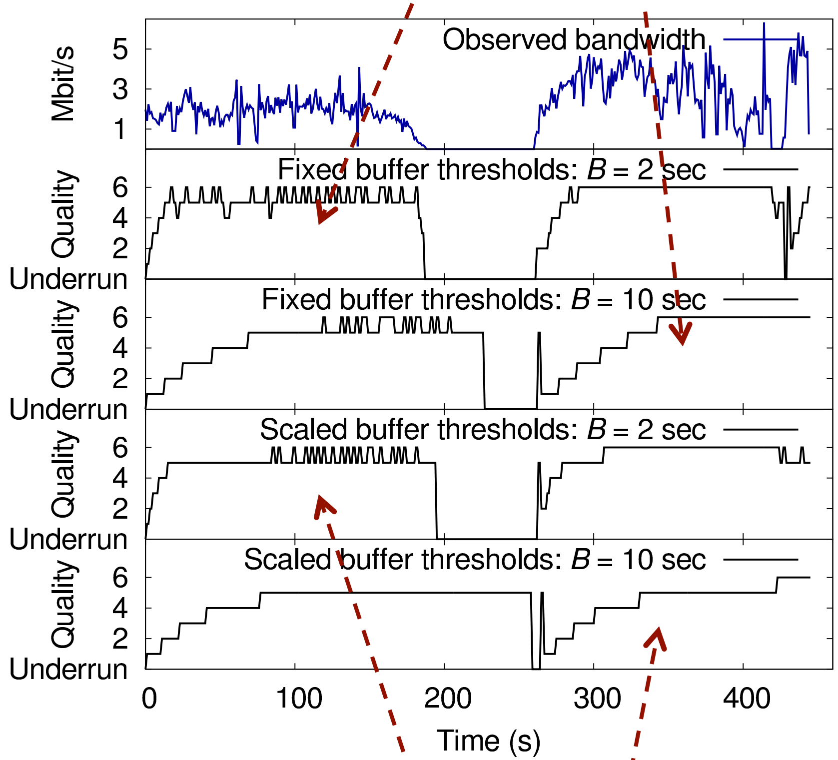
- `no upscaling for 20sec after a drop`

- `never select a layer that exceeds the current download rate (sample size 1 segment)`

- `never select a layer that exceeds:`
  $T_N = \alpha * t_N + (1-\alpha) * T_{N-1}$
  $\alpha = .25$
  t_N `=throughput observed for segment N`

- `for upscaling consider buffered` **`seconds`**

buffer must hold 2sec/10sec of current quality



buffer must hold $B*(R_N-R_1)/(R_2-R_1)$ seconds, where R_I is bandwidth req'mt of level I, for B=2sec/10sec

Client-only techniques for long-term smoothness

FESTIVE
Fair, Efficient, and Stable adapTIVE
- remove unfairness
- increase stability

"Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE",
J. Jiang, V. Sekar, H. Zhang
CoNext 2012

PANDA
Probe AND Adapt
- increase stability
- asymmetric rate shifting

"Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale",
Z.Li, X.Zhu, J.Gahm, R.Pan, A.Begen, D.Oran
ArXiv 2013

ELASTIC
fEedback Linearization Adaptive STreamIng Controller
- remove On-Off periods for all but highest quality
- allow oscillation around average quality

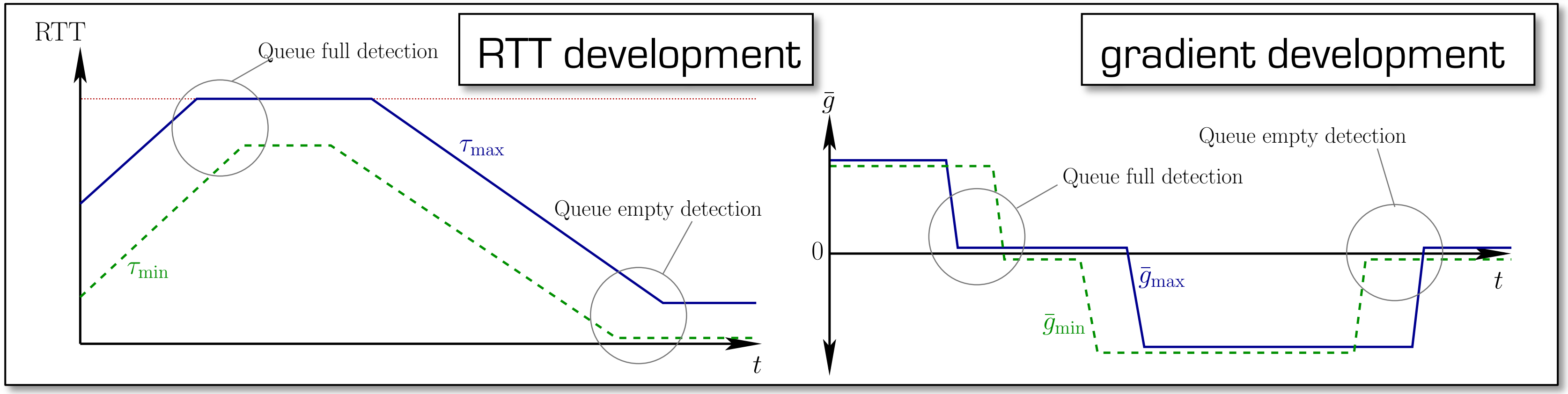
"ELASTIC: a Client-side Controller for Dynamic Adaptive Streaming over HTTP (DASH)",
L. De Cicco, V. Caldaralo, V. Palmisano, S. Mascolo
IEEE PV 2013

Some new transport development

New TCP developments: CDG

CDG - CAIA Delay Gradient TCP

Idea: infer the queue state (growing, shrinking, stable) and stay at a stable queue size



- estimate the gradient of queue occupancy development by observing the development of $\mathtt{RTT}_{\mathtt{min}}$ and $\mathtt{RTT}_{\mathtt{max}}$ over time (in sample intervals of 1 RTT)
- once per RTT compute a backoff probability based on queue development
- with this probability, reduce CWND by factor 0.7, or increase by 1

"Revisiting TCP Congestion Control using Delay Gradients",
David A. Hayes and Grenville Armitage
IFIP Networking 2011

New TCP developments: CDG

CDG - CAIA Delay Gradient TCP

Relation to adaptive HTTP streaming

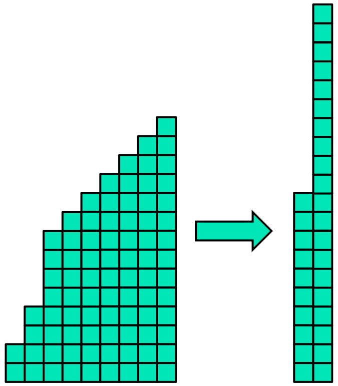
- Server-side only change, easy to deploy
- In highly multiplexed queues, a new CDG stream in slow-start will increase early backoff probability for all CDG streams, and some will make space
- As soon as slowstart threshold is reached, new flow levels out as well; prevents overshoot in initial quality estimation

- Initial behavior like slow-start, therefore not applicable when most requests are synchronized
- Does not change initial CWND behaviour and does therefore not prevent oscillation

"Revisiting TCP Congestion Control using Delay Gradients", David A. Hayes and Grenville Armitage *IFIP Networking 2011*

New TCP developments: iw10

new Linux default of Initial CWND=10



- 10 packets are generally not enough for a segment
- But the number of RTTs for entire segment download is reduced, shorter competition period; can be beneficial in conjunction with uniform distribution of downloads until congestion is experienced

- Initial burst is worse, a new segment download in slow-start will affect more active segment downloads than old-fashioned slow start

- M. Scharf: simulations show that iw10 increases loss probability by 0.5%
- J. Chu: confirms, but User Completion Time does not suffer

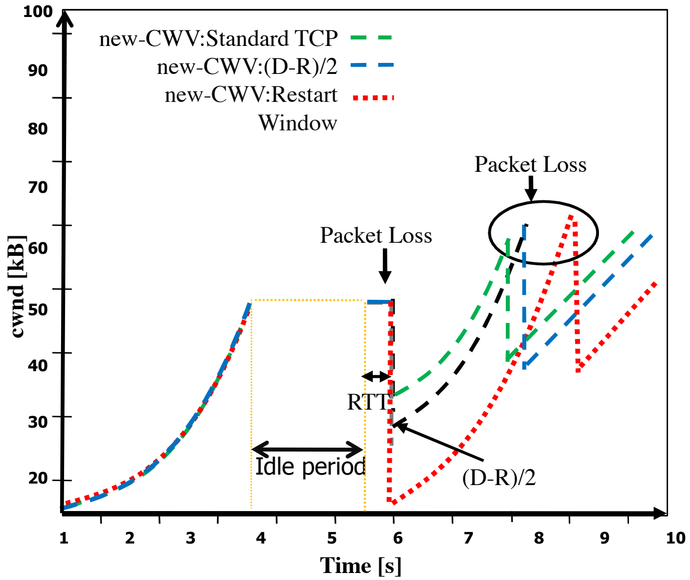
"A Testbed study on IW10 vs IW3",
Jerry Chu
ICCRG meeting at IETF 79, 2010

"Performance and Fairness Evaluation of IW10 and Other Fast Startup Schemes",
Michael Scharf
ICCRG meeting at IETF 80, 2011

New TCP developments: New CWV

New congestion window validation

- Meant for CWND estimation for any flow with idle periods
- Compared to CWV, New CWV does never decay



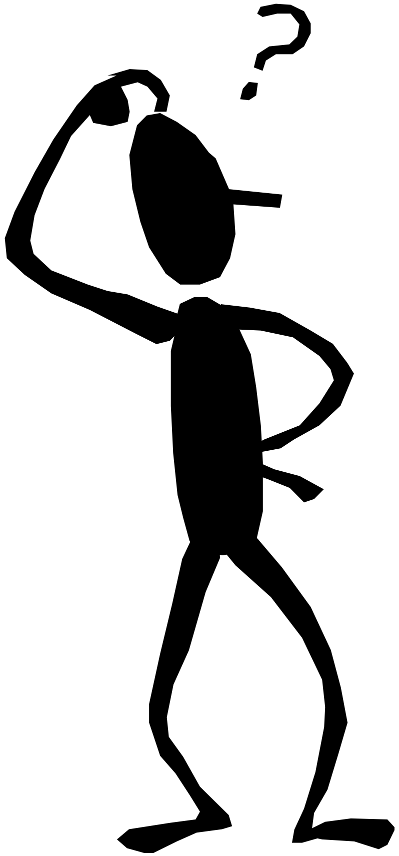
New-CWV assumes a validated phase, which is normal CWND development even after an idle period, and a non-validated phase

- If congestion is experienced in the non-validated phase,
  and L is the first lost packet according to a SACK options, then
  CWND = (last CWND-L)/2

- desired/required: a combination with pacing
- reducing the burstiness during validation

"Enhancing TCP Performance to support Variable-Rate Traffic",
Arjuna Sathiaseelan, Raffaello Secchi, Gorry Fairhurst, Israfil Biswas,
CSW@CoNext 2012

Questions? Comments?



Contact information:

Carsten Griwodz

griff@simula.no

http://mpg.ndlab.net

Acknowledgements

our work has been supported by

RITE
Reducing Internet Transport Latency

IAD
Information Access Disruptions

and Norwegian Research Council grants

187828 Verdione

193034 Perceval

176847 Hystream