BBR

2011: many reported excessive buffering and delays on the Internet (a.k.a. bufferbloat)

2012: single-connection HTTP/2 was much slower than multi-conn HTTP/1 on lossy links

2013: poor TCP throughput on WANs w/ commodity shallow-buffer switches

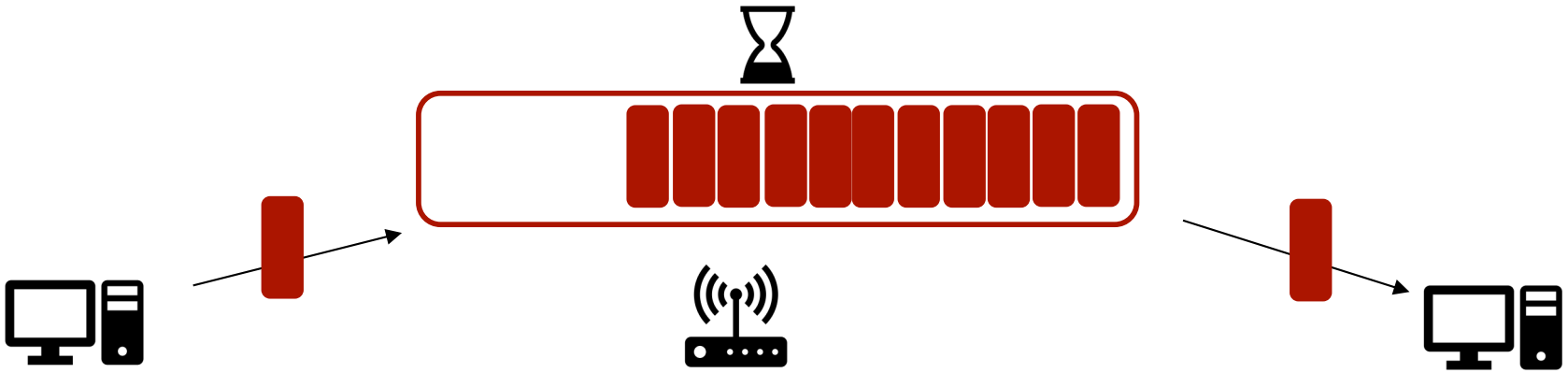
Culprit: loss-based congestion control (Reno, then CUBIC) Packet loss alone is **not** a good proxy to detect congestion

- Loss-based CC is overly sensitive to losses that come **before** congestion
- Loss-based CC bloats buffers if loss comes **after** congestion

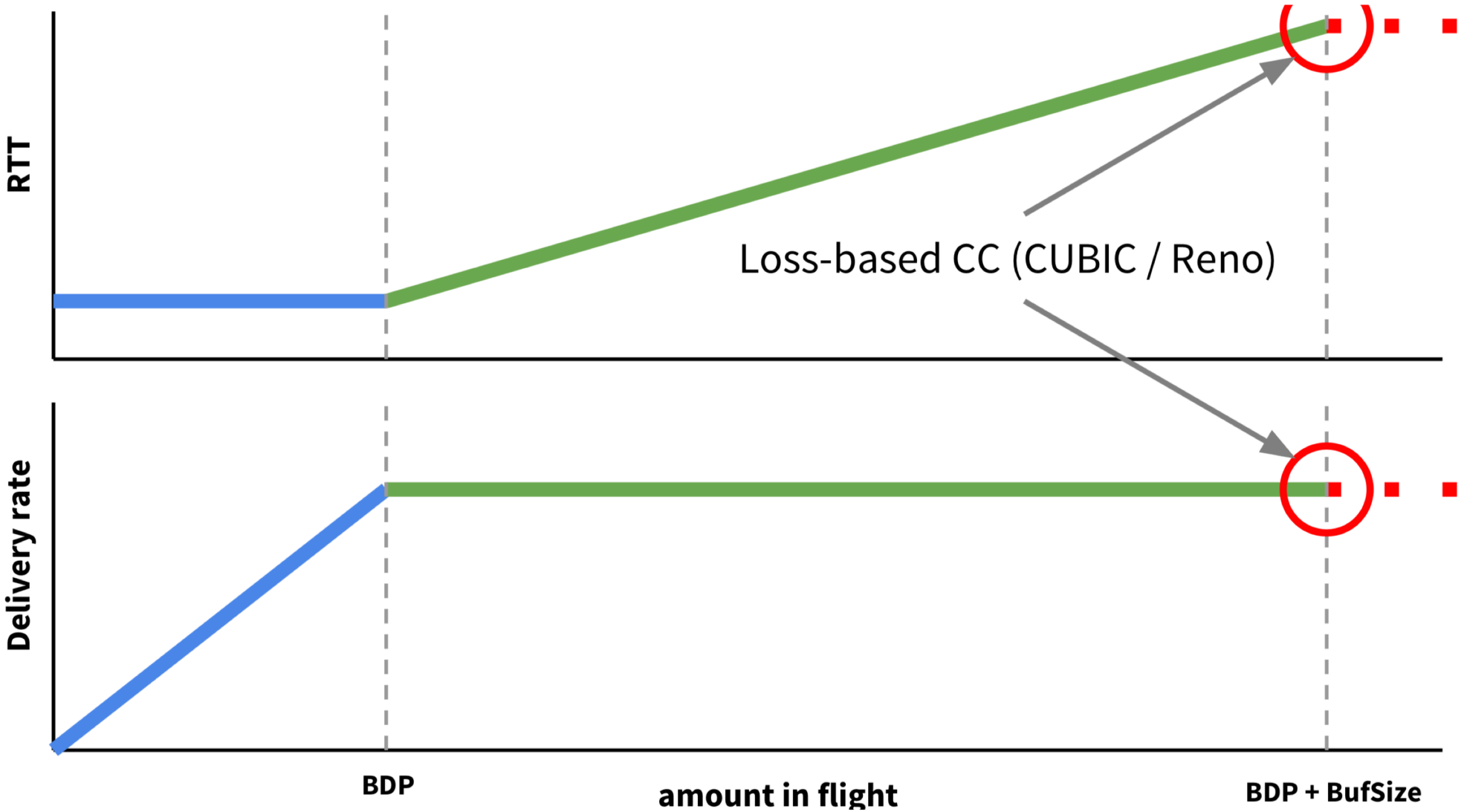
Some Slides borrowed from Neal Cardwell

Bufferbloat

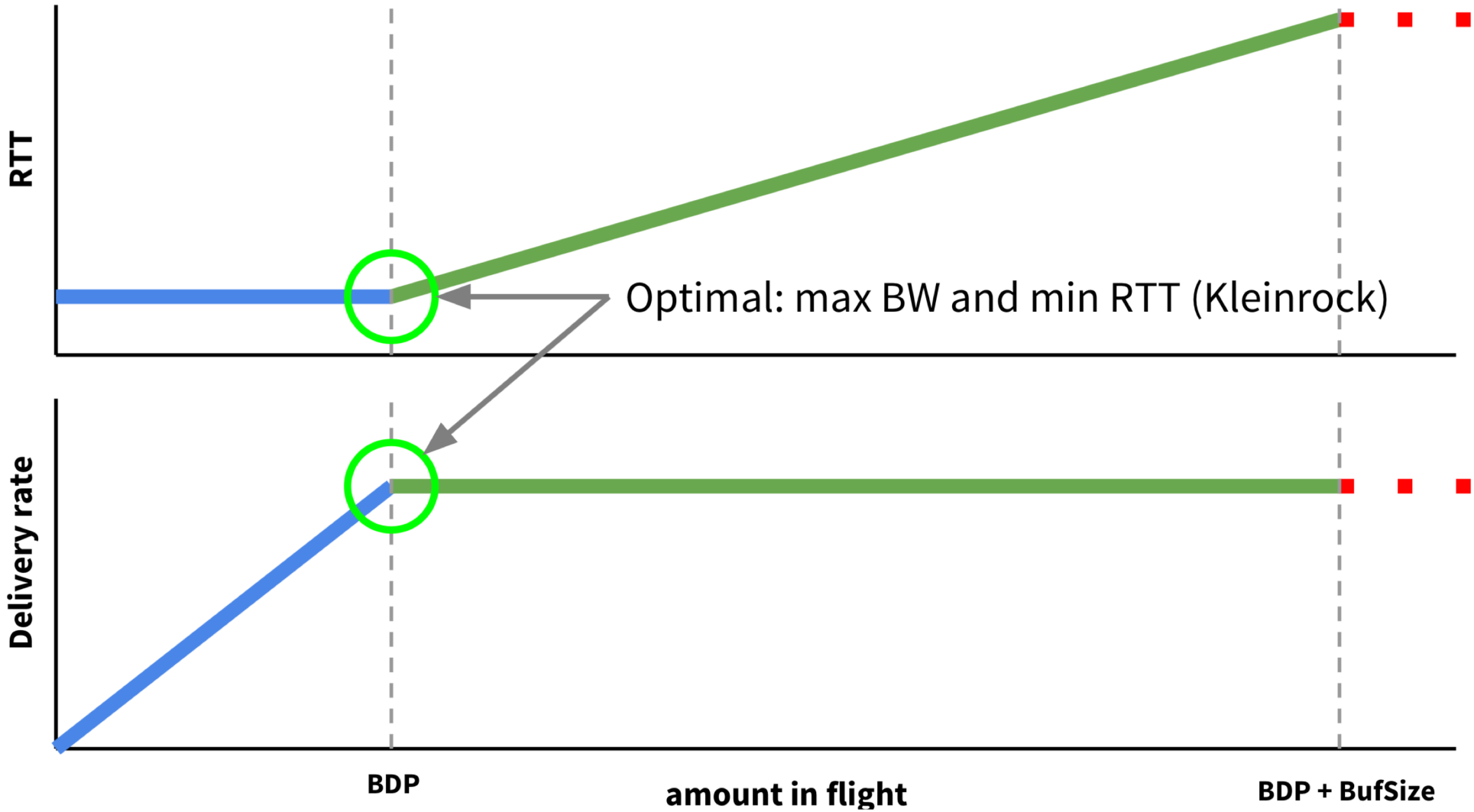
- Loss-based congestion control can induce *bufferbloat* when there is a deep buffer
  - Bufferbloat is excessive queuing delay caused by oversized buffers in the network
  - Goal in BBR: Keep buffers free so there is no queueing delay and operate at BDP
    - BDP here is that of the bottleneck buffer, defined as max bandwidth x min RTT



Where does CUBIC operate?



Optimal point?

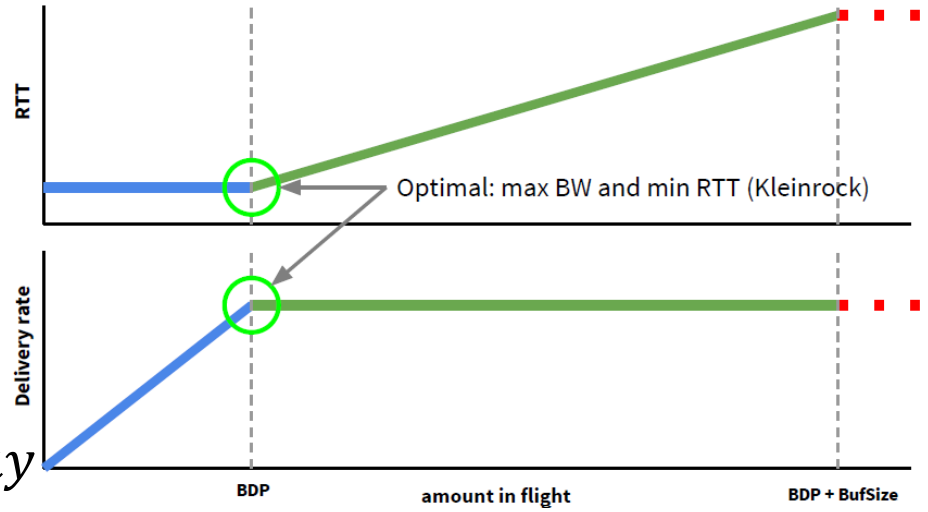


So how does BBR work?

- Instead of waiting for losses, BBR estimates the bandwidth-delay product (BDP) of the bottleneck link to set its sending rate

$$BDP = bandwidth * propagation\ delay$$

max(delivery rate) min(RTT)

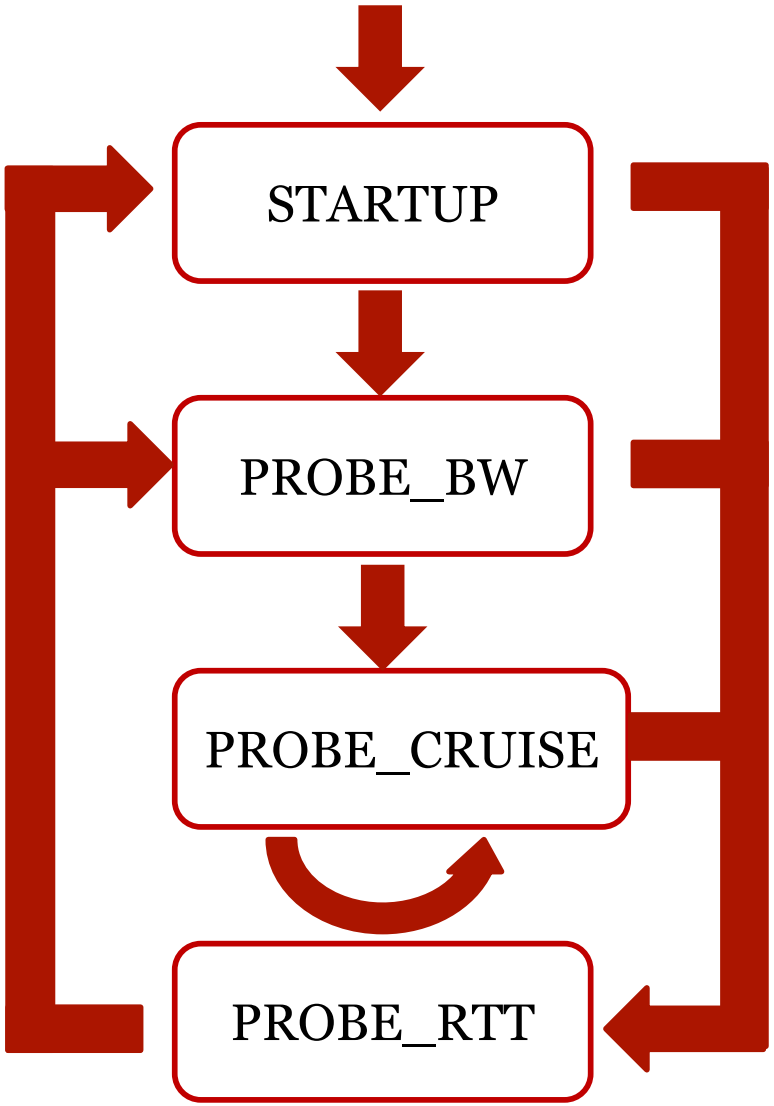


(Cardwell et al. 2017)

BBR details

- At steady state:
  - After each RTT,
    - If delivery rate is higher than previous, you can increase cwnd
    - If RTT increases, or delivery rate stagnates, then "cruise"
  - cwnd = max data rate x min RTT.
  - Every now and then drop cwnd drastically to drain the queue
  - Do nothing on losses (this is not always a great idea)

BBR details

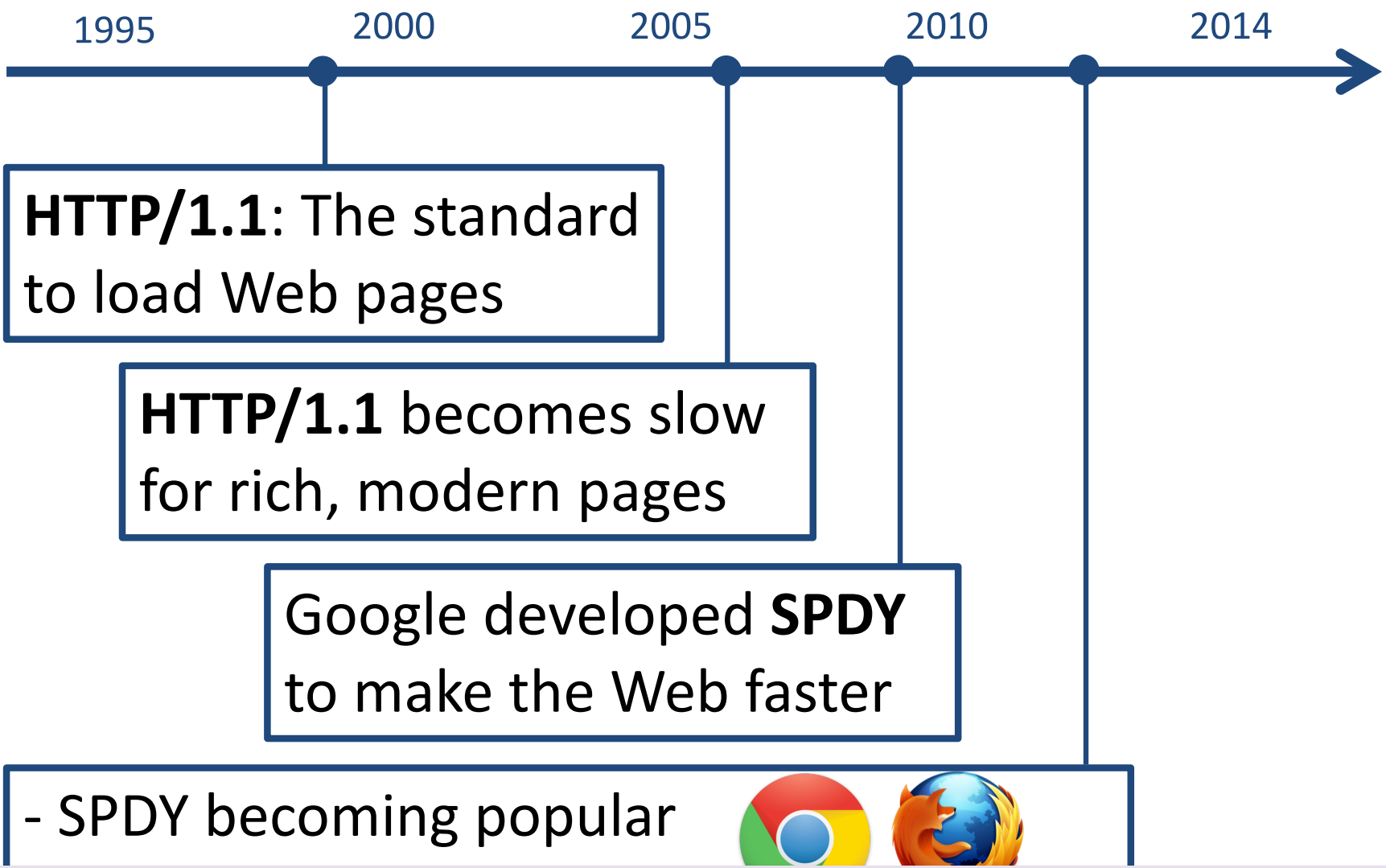


Initial studies on BBR

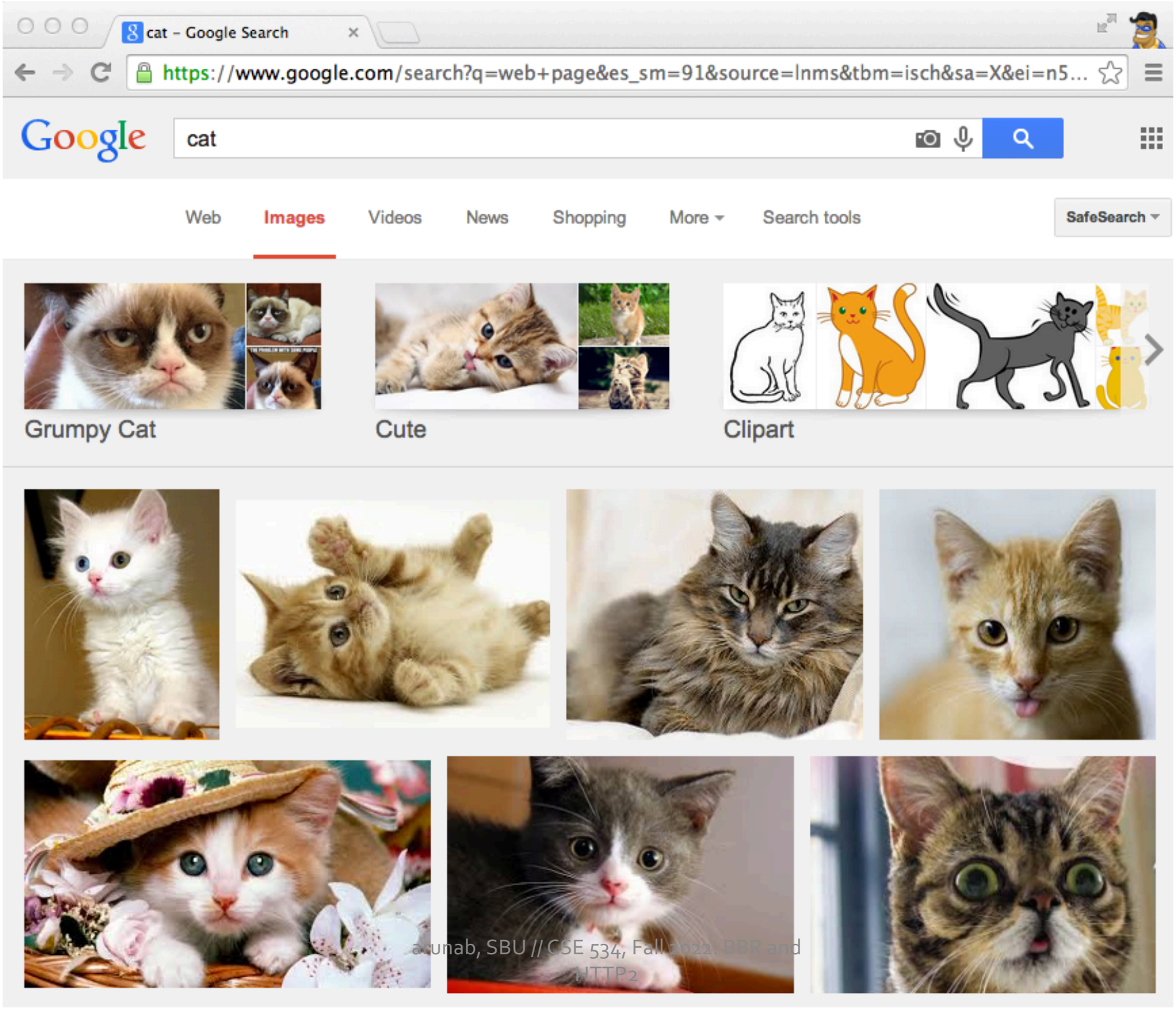
- Works better than CUBIC when buffers are small
- Can be highly unfair under loss
- New version BBR2 to solve some initial issues

How speedy is SPDY?

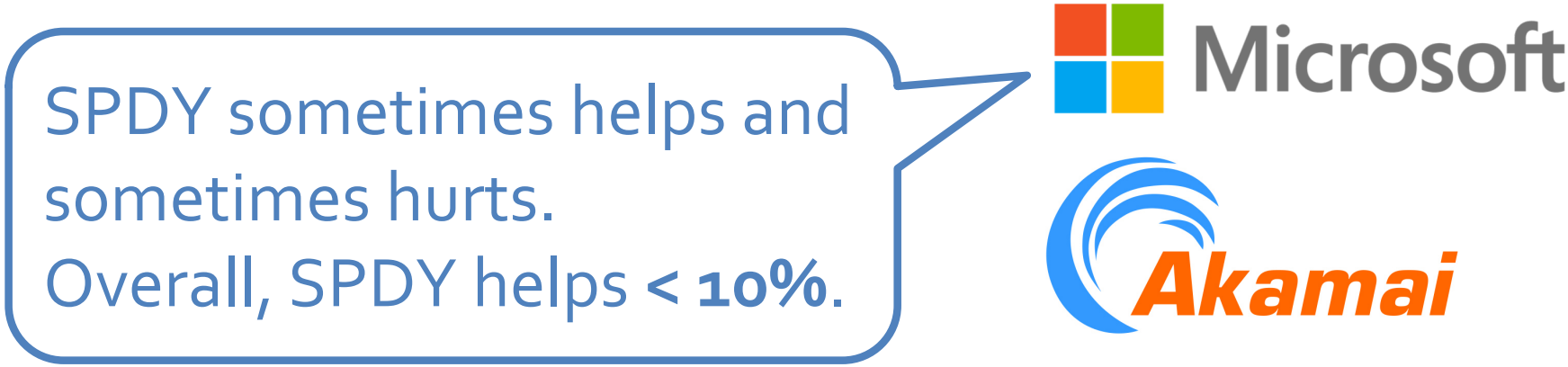
Aruna Balasubramanian



How much better is SPDY compared to HTTP in terms of improving page load time?

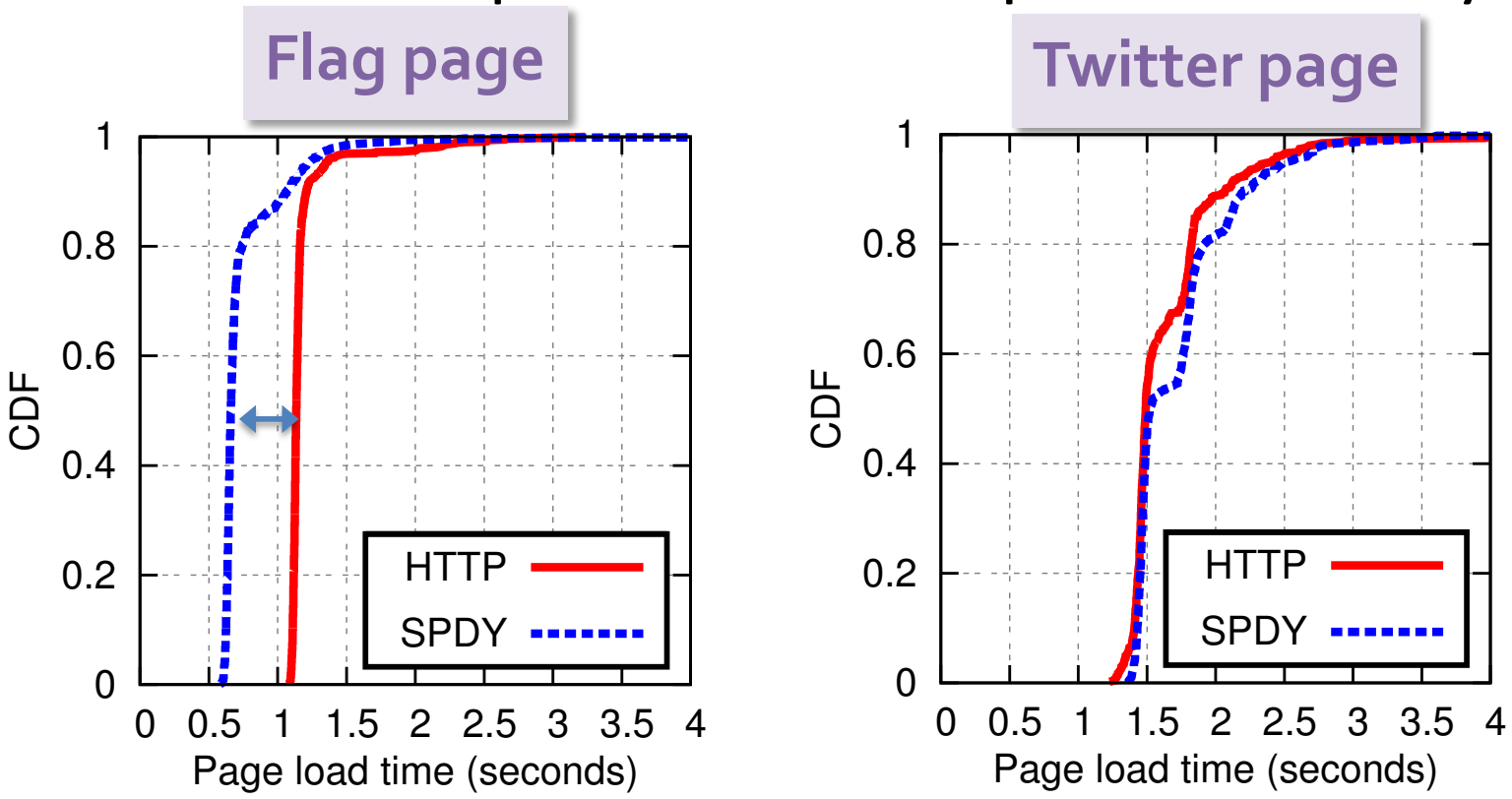

cat - Google Search
https://www.google.com/search?q=web+page&es_sm=91&source=lnms&tbm=isch&sa=X&ei=n5...
Google
cat
Web
Images
Videos
News
Shopping
More
Search tools
SafeSearch
Grumpy Cat
Cute
Clipart

How well does SPDY perform?



Why this discrepancy?

Problem 1: SPDY performance depends on many factors



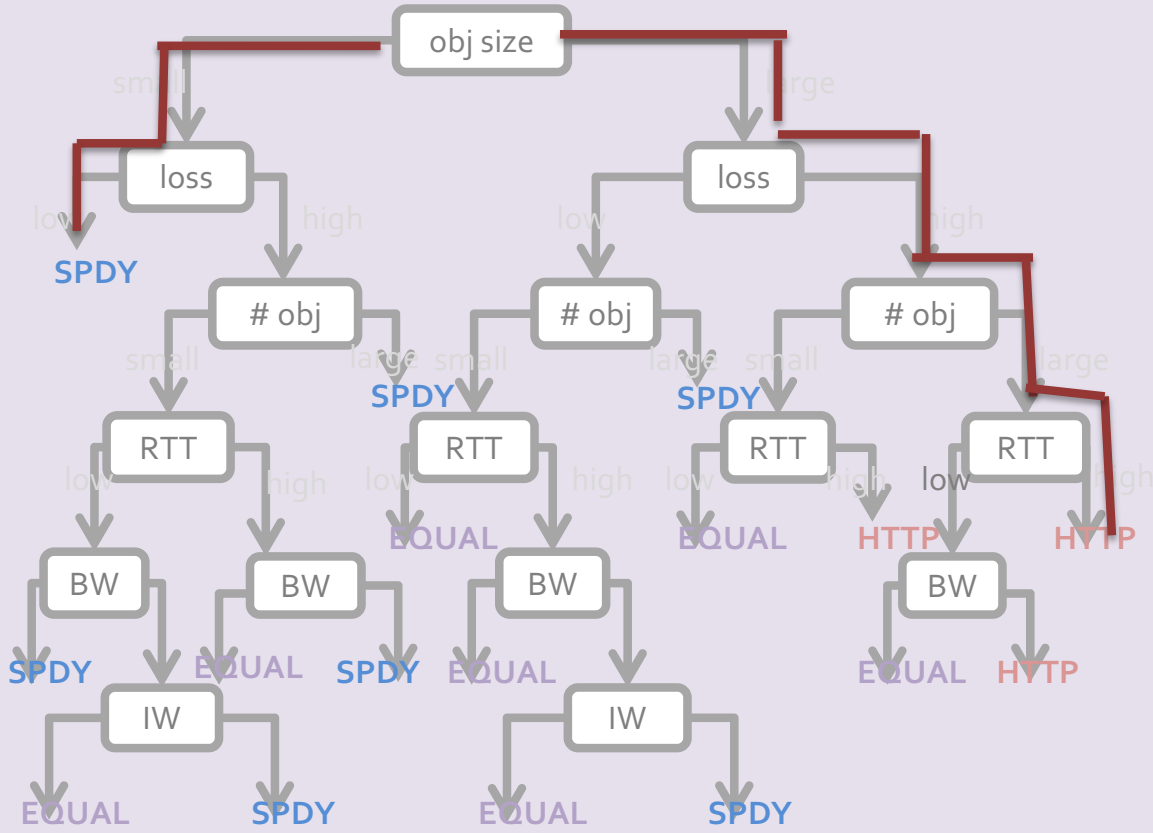
Problem 2: Difficult to measure SPDY improvement because web page loads have very high variance

Range of parameters

	Factors	Range
Network parameters	RTT	20ms, 100ms, 200ms
	Bandwidth	1Mbps, 10Mbps
	Loss rate	0, .5%, 1%, 2%
TCP settings	TCP IW	3, 10, 21, 32
Synthetic objects	Web obj. size	100B, 1K, 10K, 100K, 1M
	# of objects	2, 8, 16, 32, 64, 128, 512
Real objects	Object size/#	From Alexa top 200

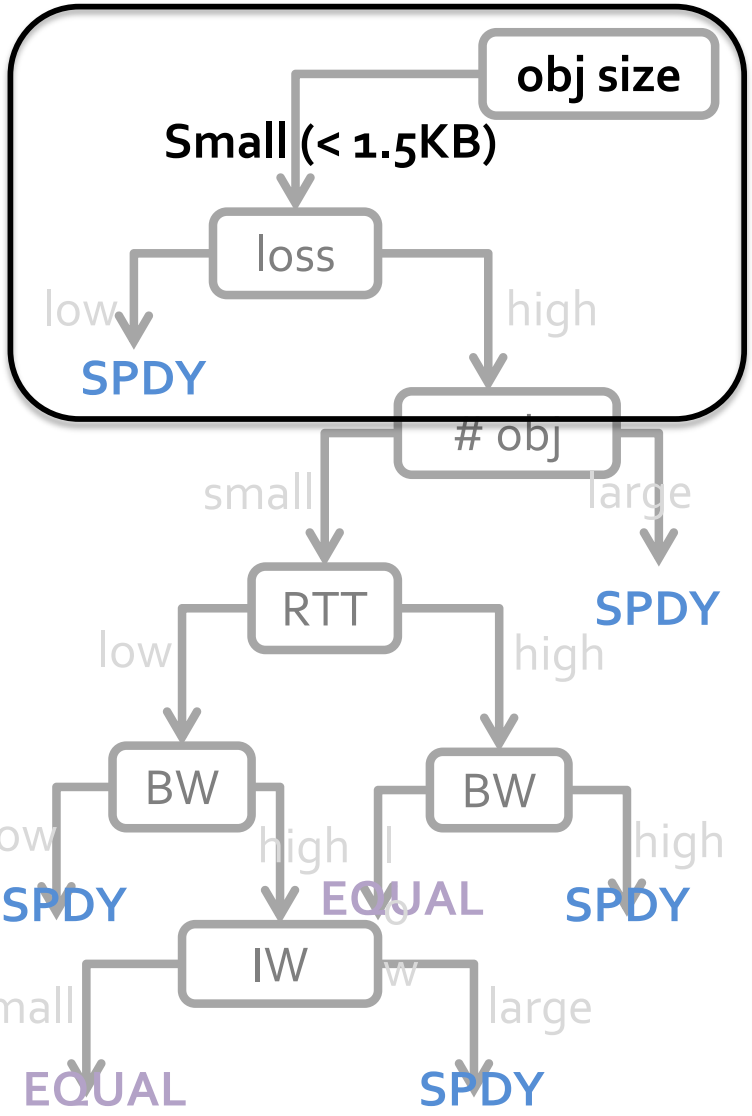
Over 5000 experiments and data points

Methodology: Decision Tree Analysis



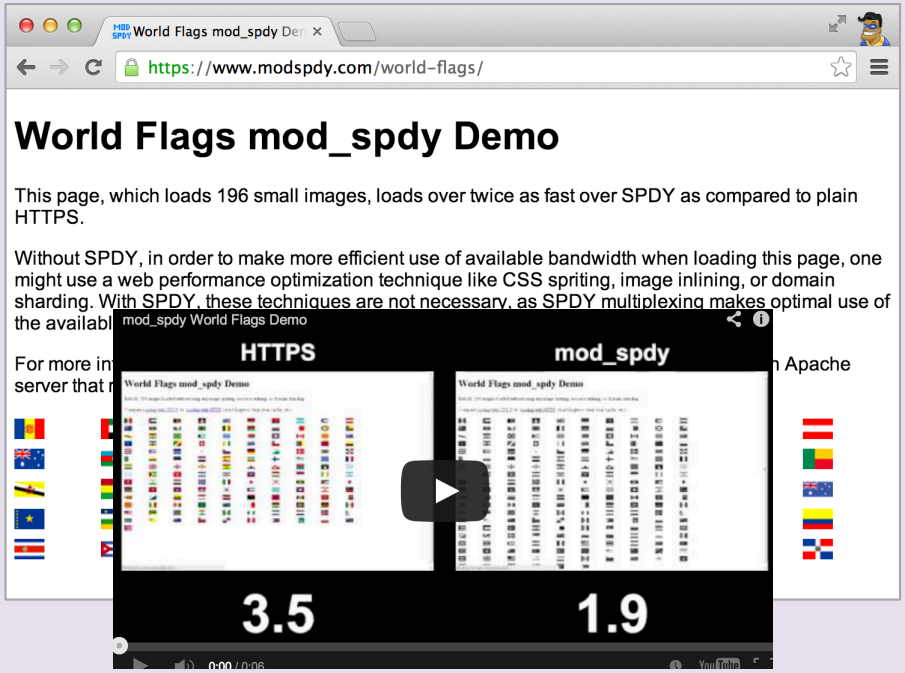
When does SPDY help performance?

SPDY helps when web objects are small

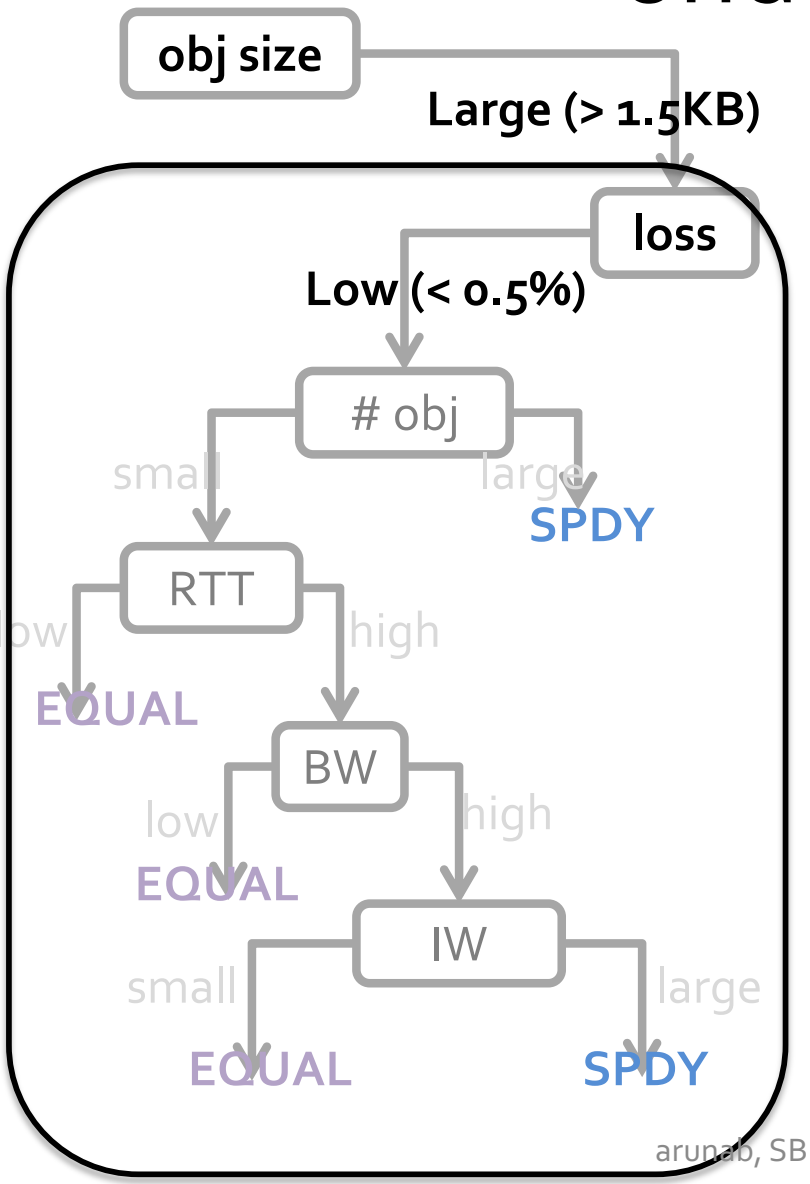


Explanation

Unlike HTTP, a TCP segment can carry multiple Web objects in SPDY.



SPDY helps when objects are large, under low loss



Explanation

By of multiplexing over single TCP, SPDY achieves

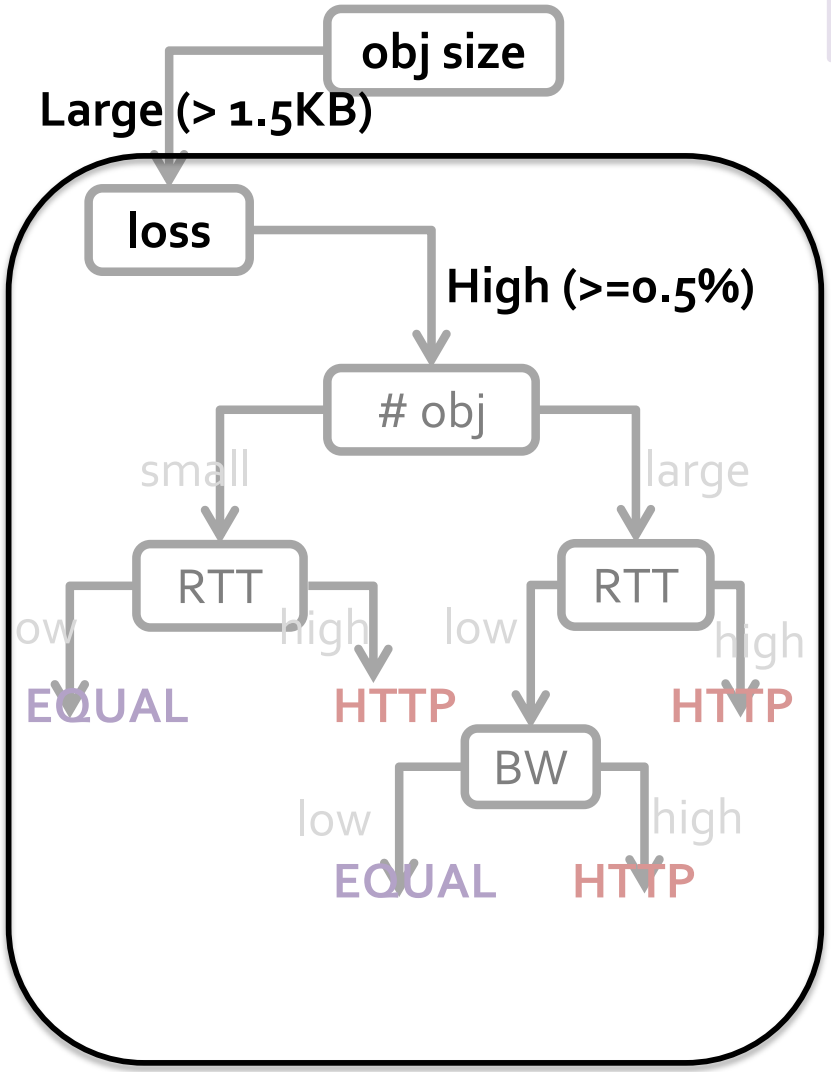
→ Lower **retransmissions**

→ **Better** use of the pipe

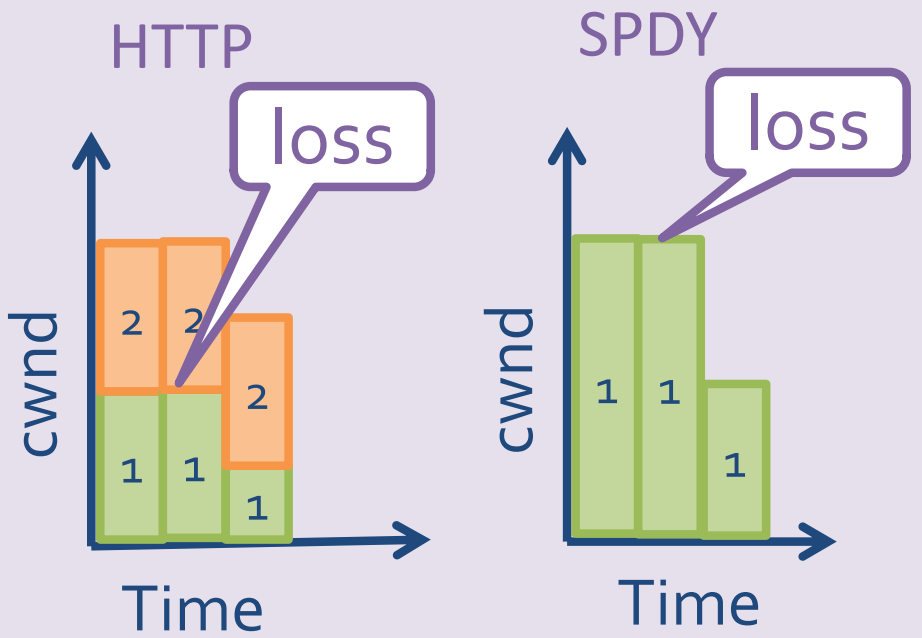
→ **Lower** TCP set up time in the critical path

When does SPDY hurt?

SPDY hurts under high loss

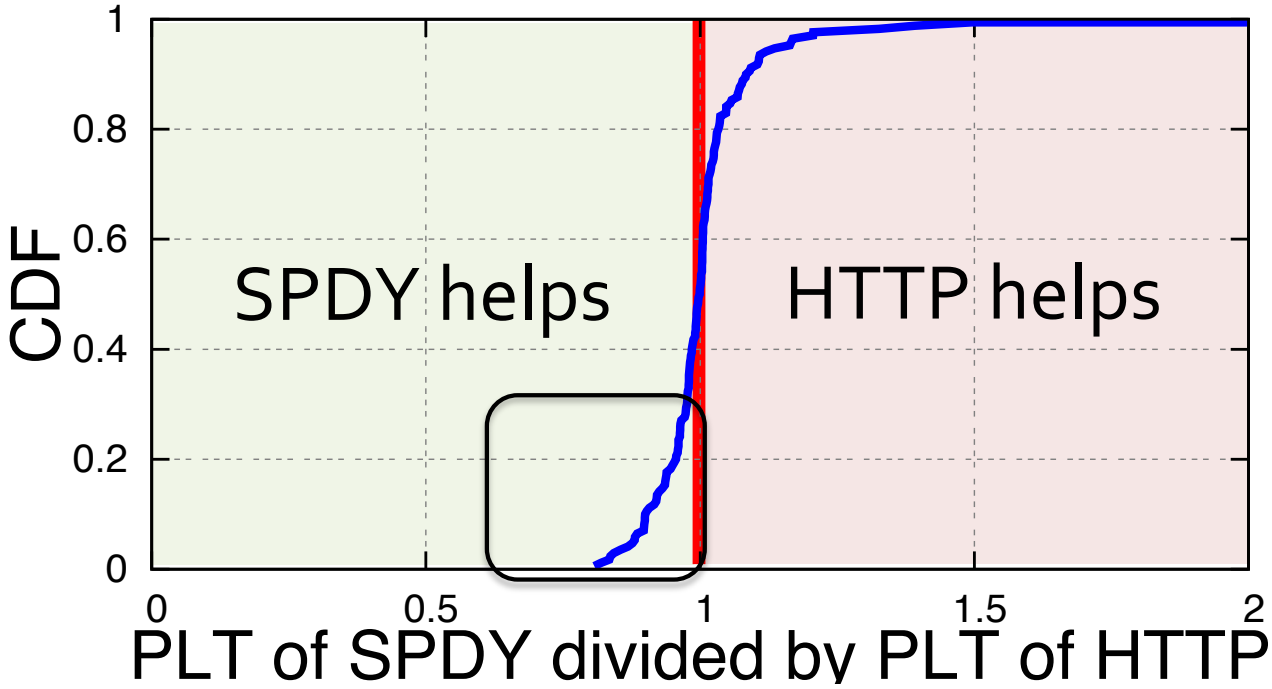


Explanation



In SPDY, cwnd reduces more aggressively under loss.

BUT SPDY vs HTTP on the web



RTT=20ms
Bandwidth
=10Mbps

Improvement when SPDY is deployed on the Web is much smaller