Increasing TCP’s Initial Window

draft-hkchu-tcpm-initcwnd-00.txt

H.K. Jerry Chu - hkchu@google.com
Nandita Dukkipati - nanditad@google.com

Topics

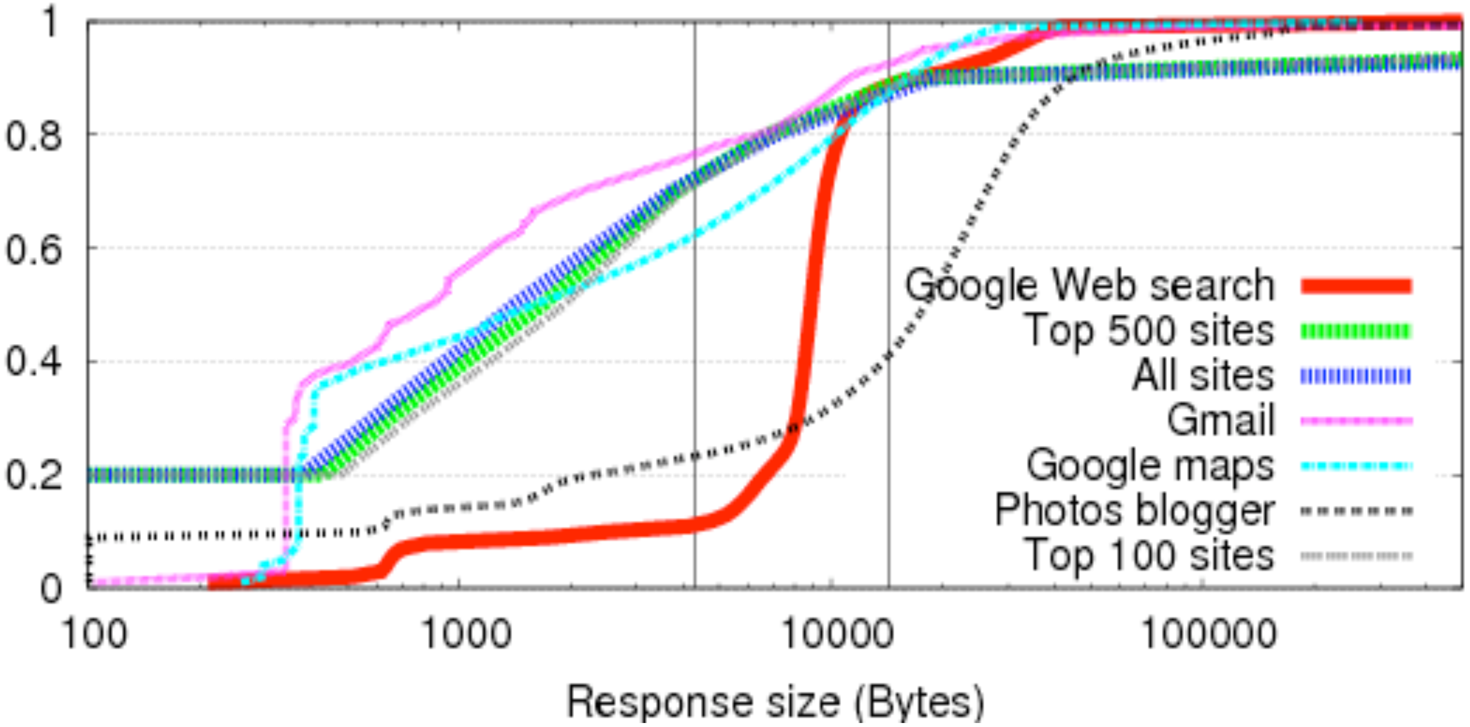
- Motivation & Justification
- Related Efforts
- Our Proposal
- Experimental Results
- Concerns
- Conclusion & Next Steps

Motivation #1

- Speed up *slow* start
  - Internet is dominated by Web traffic and short lived connections that never exit *slow* start
  - See Altas Internet Observatory 2009 Annual Report (technical plenary on Thur.)
- Web objects and pages growing in size

quantiles	Average	30	40	50	60	70	80	90
KB per Get	8.12	0.59	0.92	1.41	2.28	3.72	7.1	18.68
KB per Page	384	132	181	236	304	392	521	776

CDF of HTTP Response Sizes



Motivation #2

- IW=10 saves up to 4 round trips
- Reverse the trend of browsers opening more and more simultaneous connections
  - Six per domain
  - IE8 is shown to open up to 180 simultaneous connections to the same server (when server advertises 30 domain names)!
  - Works against TCP's congestion control mechanism
  - Congestion manager (CM) is difficult to implement
- Allow more fast recovery through fast retransmit

Justification – why is IW=10 safe?

- Huge bandwidth growth since IW=4KB (1998)
  - – Average b/w has reached 1.7Mbps world wide
  - – Narrowband (<256Kbps) has shrunk to 5%
- Browsers open many simultaneous connections
  - – Effectively test network with bursts much larger than IW=4KB
- TCP is already bursty
  - – Slow start bursts pkts out at twice the bottleneck b/w

Related Efforts

- Fast/Quick/Jump/Swifter/… Starts
  - Any one ready for standardization and deployment?
- Persistent HTTP
  - Benefit limited by connection persistency
  - Does not help the first data chunk, often the largest
- HTTP pipelining
  - Can benefit more from a larger IW
  - Limited deployment due to little support from proxies

Related Efforts (cont')

- SPDY - Google's Web experimental protocol
  - "An Argument For Changing TCP Slow Start" http://sites.google.com/a/chromium.org/dev/spdy /An_Argument_For_Changing_TCP_Slow_Start.pdf
- Congestion manager
  - complex to implement
- Cwnd cache
  - Similar to the temporal sharing of TCP states in RFC2140 but aggregated on a per /24 subnet basis
- NetDB
  - Global database of subnet attributes from past history

Our Proposal

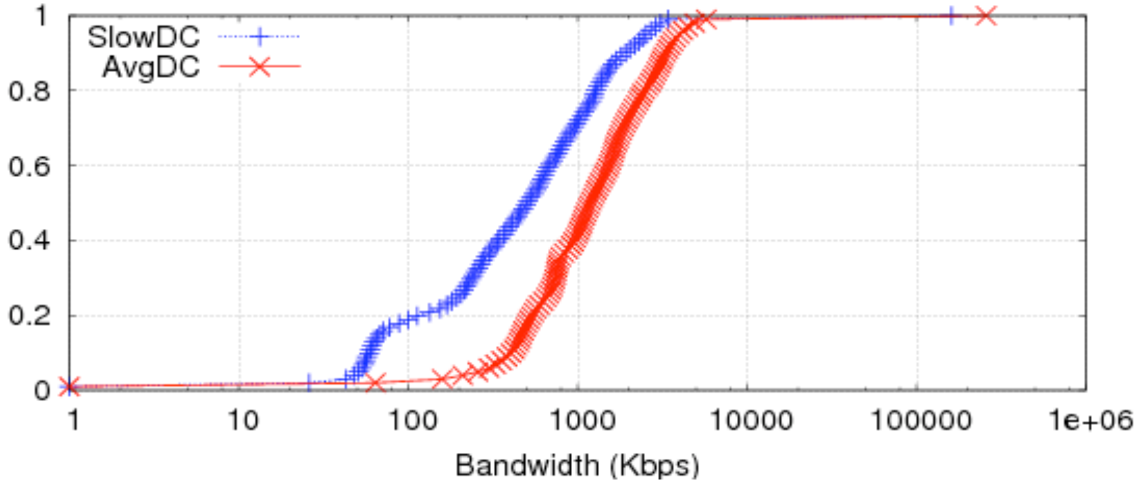
- Increase IW to 10 or higher
  - All experimental data shown here are from IW=10
  - Ongoing experiments continue with IW=16
- Design principle - KISS
  - No state sharing across connections
  - IW a fixed value or based on data collected during 3WHS
  - No pacing required
- May consider a non-standard response function when loss occurs during IW

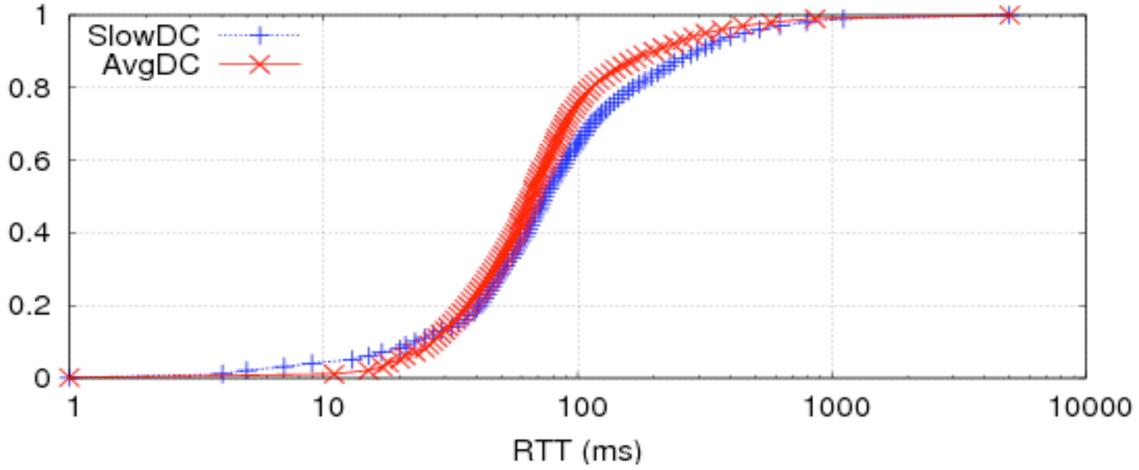
Experiment Setup

- Experiments with larger IW in several data centers over past few months
- Front-end servers configuration
  - Linux TCP implementation, CUBIC cong. control
  - `initcwnd` option in `ip route` command
- Multiple connections opened by applications are served from the same data center
- Results from two representative data centers for two consecutive weeks

Ref: http://code.google.com/speed/articles/tcp_initcwnd_paper.pdf

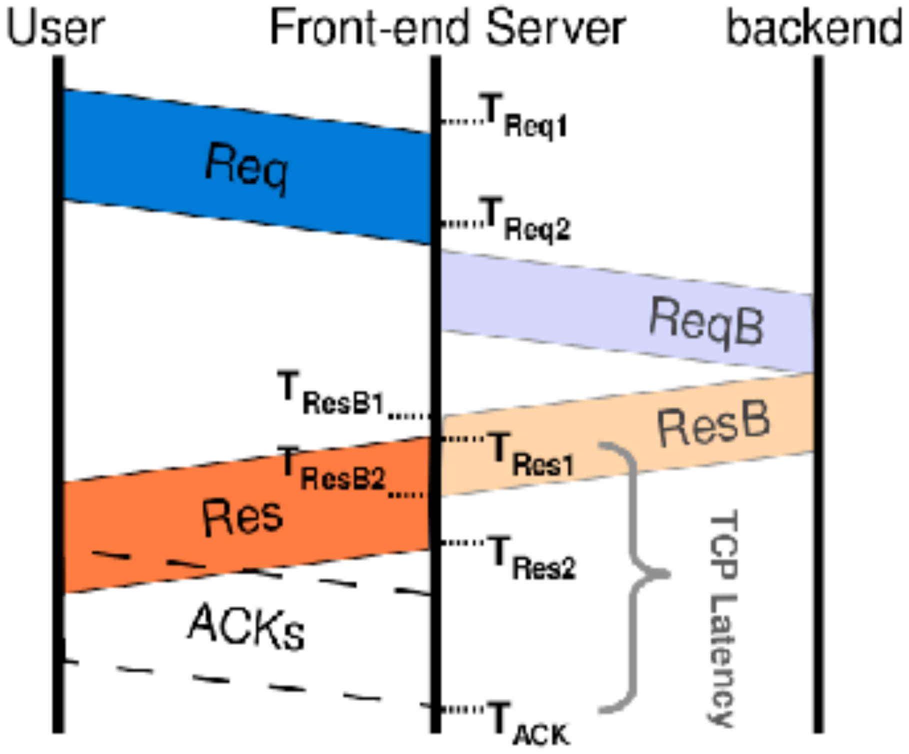
User Network Characteristics





- Median BW
  - AvgDC: 1.2Mbps
  - SlowDC: 500Kbps
- Median RTT ~ 70ms

Metrics of Interest and Datasets



Dataset	# Subnets	# Responses	Vol. (TB)
AvgBaseData	1M	5.5B	39.3
AvgExpData	1M	5.5B	39.4
SlowBaseData	800K	1.6B	9.3
SlowExpData	800K	1.6B	9.1

- Logged HTTP transactions
- Metrics
  - TCP Latency
  - Retransmission rate

Outline of Experiment Results

- Are client receive windows large enough?
- Impact of IW=10
  - Overview of Web search latency
  - Impact of subnets of varying BW, RTT, BDP
  - Impact on responses of different sizes
  - Latency in mobile subnets
  - Effect on retransmission rate
  - Impact on applications with concurrent TCP connections

Client Receive Windows

receive window of first HTTP request

OS	% >15KB	Average
FreeBSD	91%	58KB
iPhone	66%	87KB
Linux	6%	10KB
Mac	93%	270KB
Win 7	94%	41KB
Win Vista	94%	35KB
Win XP	88%	141KB

- Greater than 90% TCP connections have large enough receive windows to benefit from using IW=10

TCP Latency for Web Search

AvgDC

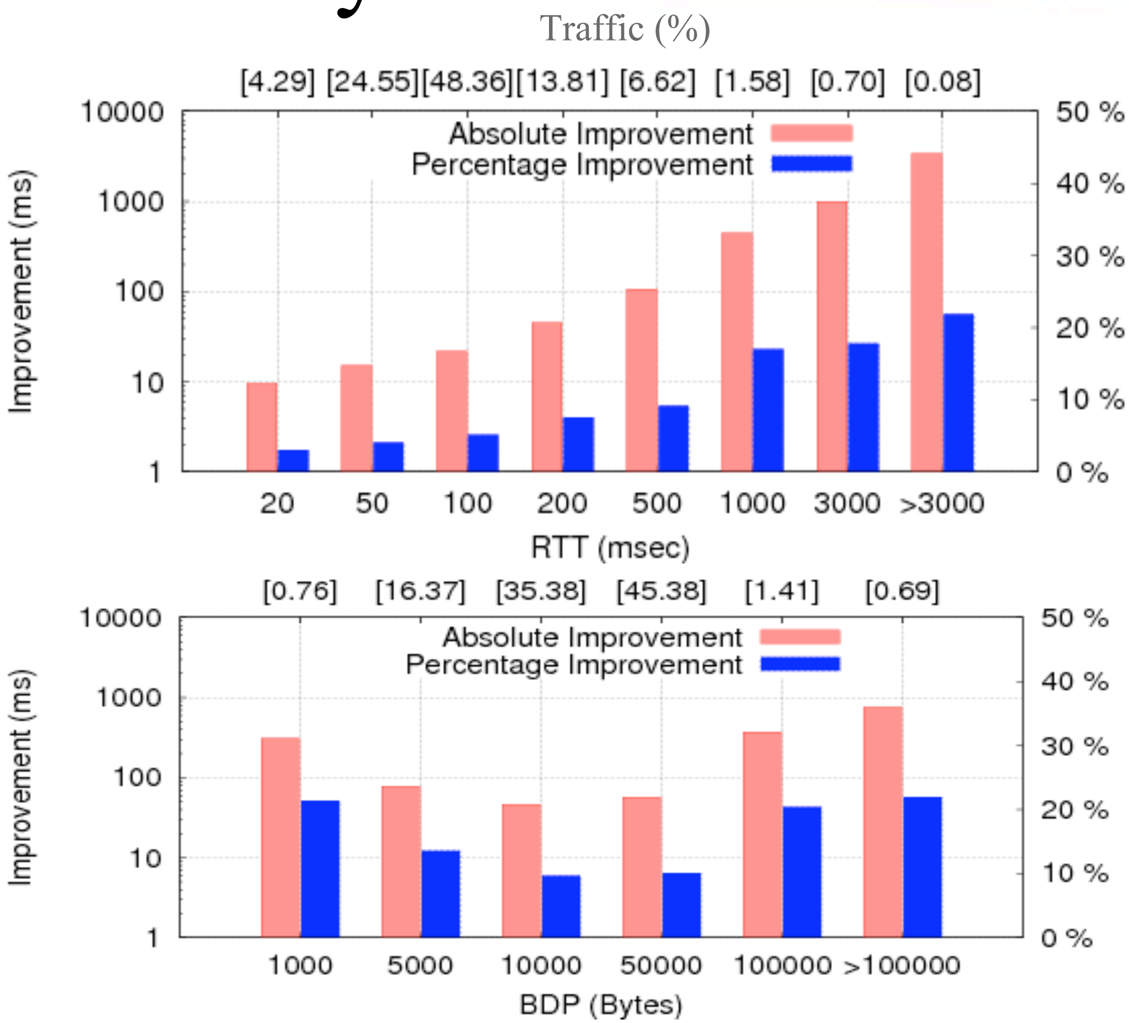
Qtls	Exp	Base	Diff %
10	174	193	9.84%
50	363	388	6.44%
90	703	777	9.52%
95	1001	1207	17.07%
99	2937	3696	20.54%
99.9	8463	10883	22.24%
Average	514	582	11.7%

SlowDC

Qtls	Exp	Base	Diff %
10	204	211	3.32%
50	458	474	3.38%
90	1067	1194	10.64%
95	1689	1954	13.56%
99	5076	5986	15.20%
99.9	16091	18661	13.77%
Average	751	823	8.7%

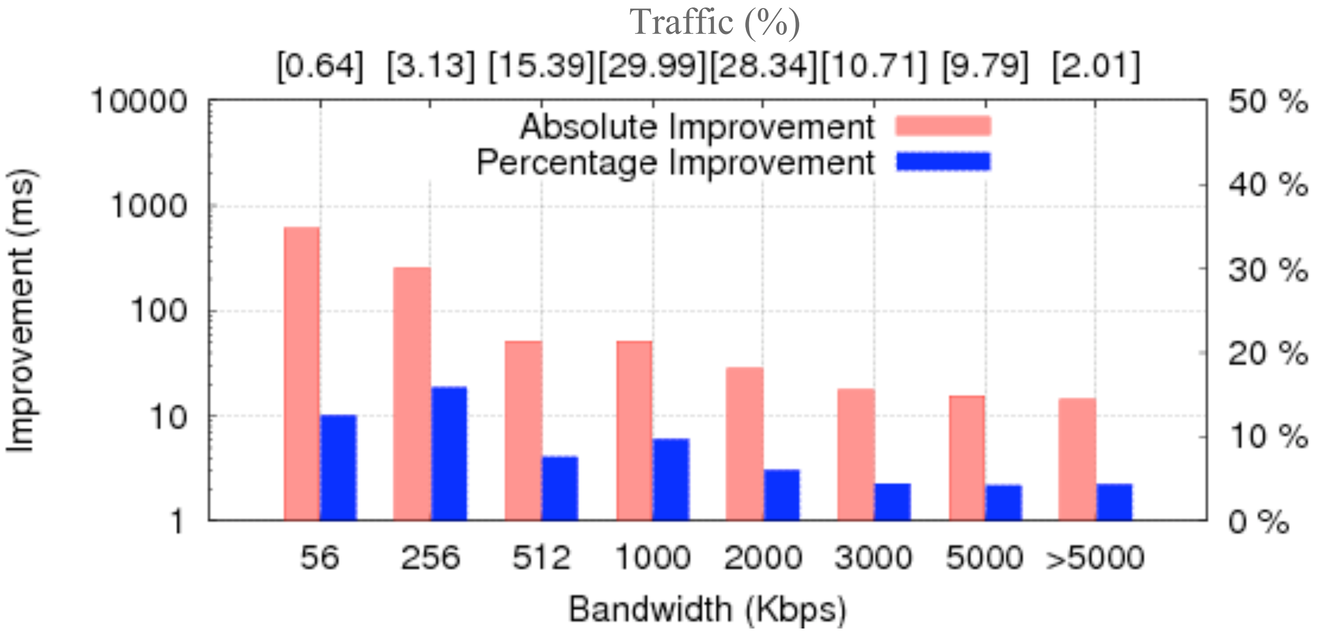
Latency measured in milliseconds

Latency as Functions of BW, RTT, BDP



- Largest improvements (~20%) are for high RTT and BDP networks

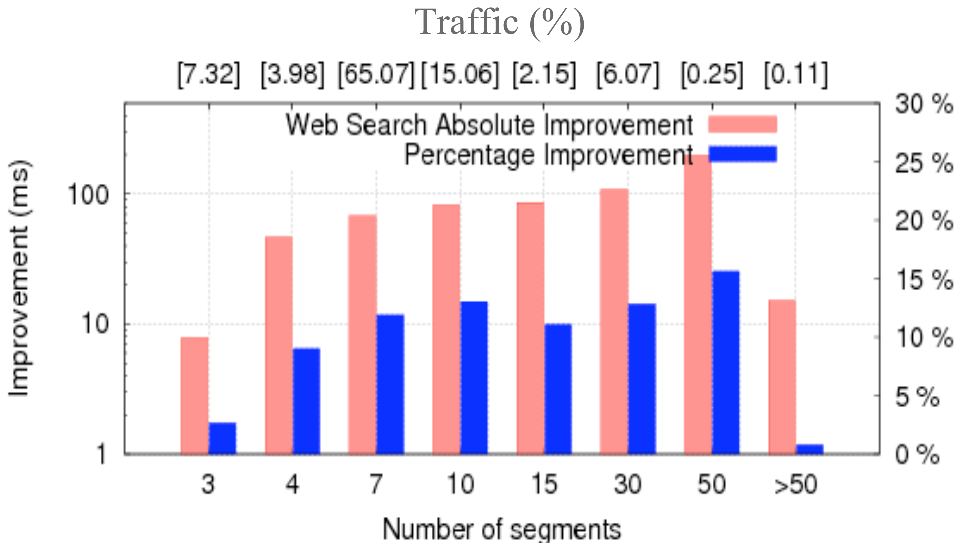
Latency as Functions of BW, RTT, BDP



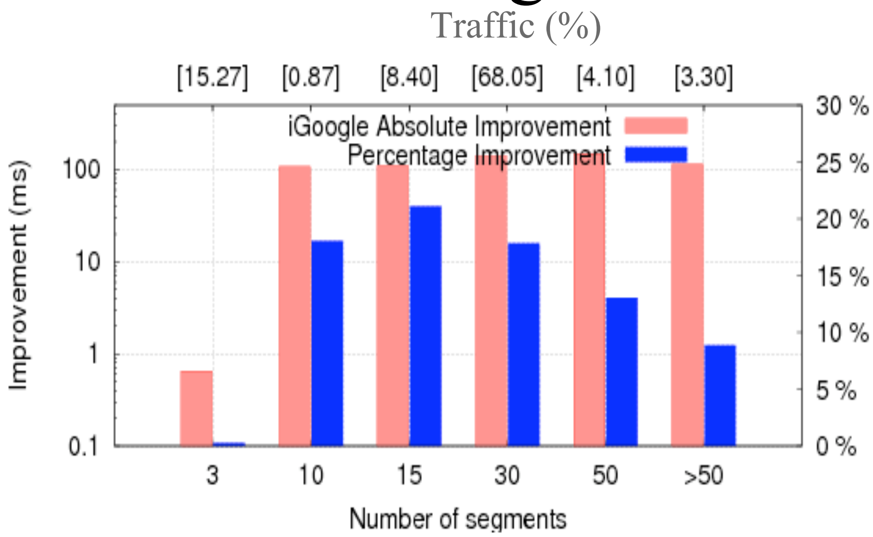
- Slow start latency = $N_{\text{slow-start}}$ * RTT + response-size/BW
- Low BW subnets show significant improvements
  - Fewer slow start rounds, faster loss recovery

Latency for Varying Sizes of Responses

Web Search



iGoogle



- Absolute improvement increases with size
- Response sizes <=3 segments perform no worse of than baseline

Per-subnet Latency and Mobile Networks

Web Search in AvgDC

/24 subnet latency

Qtls	Exp	Base	Diff %
10	301	317	5.32%
50	421	450	6.89%
90	943	1060	12.4%
95	1433	1616	12.77%
99	3983	4402	10.52%
99.9	9903	11581	16.95%

Mobile subnets

Qtls	Exp	Base	Diff %
10	468	508	7.8%
50	517	564	8.4%
90	1410	1699	17%
95	2029	2414	15.9%
99	4428	5004	11.5%
99.9	9428	10639	11.4%

- Higher improvements in mobile because of larger RTTs

Effect on Retransmission Rate

AvgDC	Exp	Base	Diff
Web Search	1.73 [5.63]	1.55 [5.82]	0.18 [-0.2]
Maps	4.17 [7.78]	3.27 [7.18]	0.9 [0.6]
iGoogle	1.52 [11.2]	1.17 [9.79]	0.35 [1.41]
Overall	2.29 [6.26]	1.98 [6.24]	0.31 [0.02]

SlowDC	Exp	Base	Diff
Web Search	3.5 [10.44]	2.98 [10.2]	0.52 [0.26]
Maps	5.79 [9.32]	3.94 [7.36]	1.85 [1.97]
iGoogle	2.8 [19.88]	1.88 [13.6]	0.92 [6.29]
Overall	4.21 [8.21]	3.54 [8.04]	0.67 [0.17]

An entry has two parts: retrx rate [% responses with >0 retrx]

- Most increase in retransmission rate from applications using multiple concurrent connections

Applications using Multiple Concurrent Connections

Google Maps Latency

AvgDC

Qtls	Exp	Base	Diff [%]
10	47	48	2.08%
50	220	225	2.22%
90	653	679	3.83%
95	1107	1143	3.15%
99	2991	3086	3.08%
99.9	7514	7792	3.57%

SlowDC

Qtls	Exp	Base	Diff [%]
10	19	27	29.6%
50	170	176	3.4%
90	647	659	1.8%
95	1172	1176	0.3%
96	1401	1396	-0.4%
97	1742	1719	-1.3%
99	3630	3550	-2.3%
99.9	10193	9800	-4%

- Effective IW for Maps in experiment is 80-120 segments
- Latency improves on average in AvgDC and SlowDC

Concerns

- What happens if everyone switches to IW=10?
  - congestion collapse unlikely since congestion backoff mechanism remains in place
- Negative impact to slow or mobile network?
  - Our experiments did not show much
- How does IW=10 flows affect flows with IW=3?
- How does IW=10 affect non-web or long lived connections?

Conclusion & Next Steps

- A moderate increase of IW seems to be the best "near-term" solution to relieve the slow-start logjam
- Propose to TCPM for adoption as a WG item
- More tests and analysis are needed!
- We would like to call for volunteers to help out!

Backup Slides

1st Attempt - Cwnd Cache

- Similar to the temporal sharing of TCB states proposed in RFC2140, but aggregated on per /24 subnet basis
- Medium implementation complexity
- Memory vs cache hit rate
- Suffers low cache-hit rate due to load balancers

2nd attempt - NetDB

- A global database of per-subnet (/24)/time-slot bw/rtt/cwnd estimates from past history
- Effectiveness depends on the accuracy of the data
- High implementation complexity
- Doesn’t adapt to dynamic congestion condition
- Google-only solution