End-to-End TCP Packet Dynamics

Vern Paxson

Network Research Group
Lawrence Berkeley National Laboratory
University of California, Berkeley

vern@ee.lbl.gov

Winter '97 NANOG, San Francisco

February 10, 1997

Overview:

Measurement framework.

Analysis headaches.

TCP pathologies.

Network pathologies.

Packet loss, congestion time scales, available bandwidth.

Future work.

A framework for conducting end-to-end measurements:

The Internet is <u>tremendously</u> diverse — no single site or path is “typical” or “representative”.

Key idea: run a specially designed measurement process at a number of Internet sites. (SIGCOMM '96 routing study.)

The N^2 effect: if there are N measurement sites, adding just one more adds $2N$ more Internet paths that can be measured end-to-end. Total number of measurable paths: $O(N^2)$.

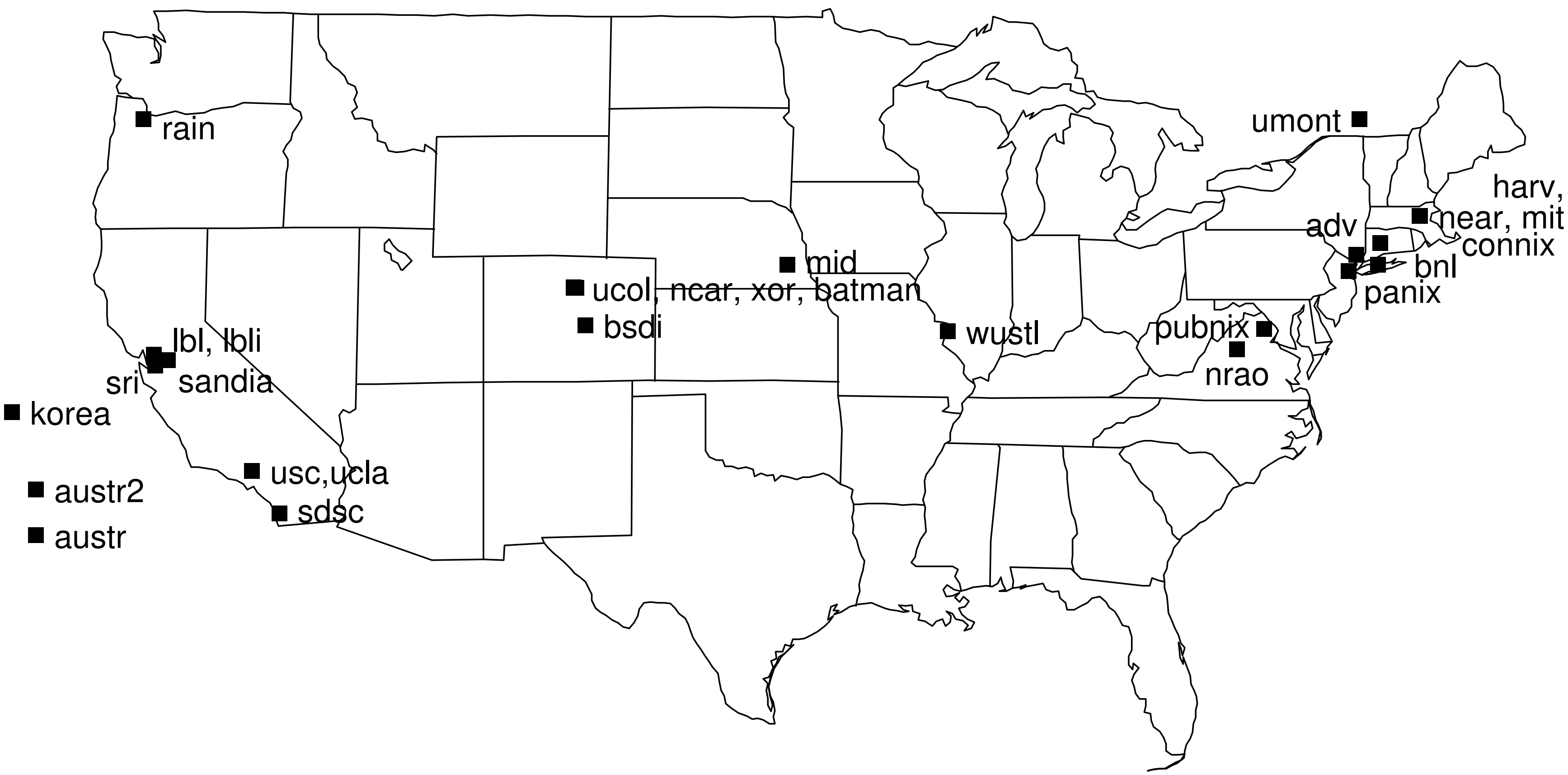
The Network Probe Daemon (NPD):

NPD runs at different sites around the Internet.

It accepts (authenticated) requests to:

- `traceroute` to a remote host
- source/sink a TCP probe
- run `tcpdump/rtcpdump` to monitor a probe
- send/reset log
- self-test

Participating sites — North America



Links traversed — North America

How to probe:

Why TCP?

- Real-world traffic
- Fine-grained time scales . . .
- . . . but with congestion control

Why not TCP?

- Endpoint TCP behavior a major analysis headache
- Self-interference a major analysis headache
- Note: acks don't self-interfere!

The raw data:

$\mathcal{N}_1$: 2,822 100 KB TCP transfers between 25 sites, traced at both ends, Dec 1994. 730,000 packets.

$\mathcal{N}_2$: 18,490 transfers between 31 sites, traced at both ends, Nov–Dec 1995. 4,850,000 packets. Big windows.

21 sites in common.

Enough traces that <u>automated</u> analysis required: `tcpanaly`.

Analysis headaches — experimental design:

10-minute max. measurement lifetime introduces bias towards underestimation.

Centralized architecture biases towards underestimation.

These could be fixed by having NPD “batch” requests.

Analysis headaches — packet filter prevarication:

Drops: present in 15% of $\mathcal{N}_1$ measurements, 10% of $\mathcal{N}_2$.

Not always correctly reported.

`tcpanaly` uses self-consistency checks to detect them.

Truncations: frequently with Solaris 2.3 (not seen in 2.4).

Extras: frequently with IRIX 5.1 and 5.2 machines (not seen in others).

Reorderings: frequently with Solaris 2.3 and 2.4.

Analysis headaches — clocks:

We don't require synchronized clocks.

(Turns out they aren't a cure-all.)

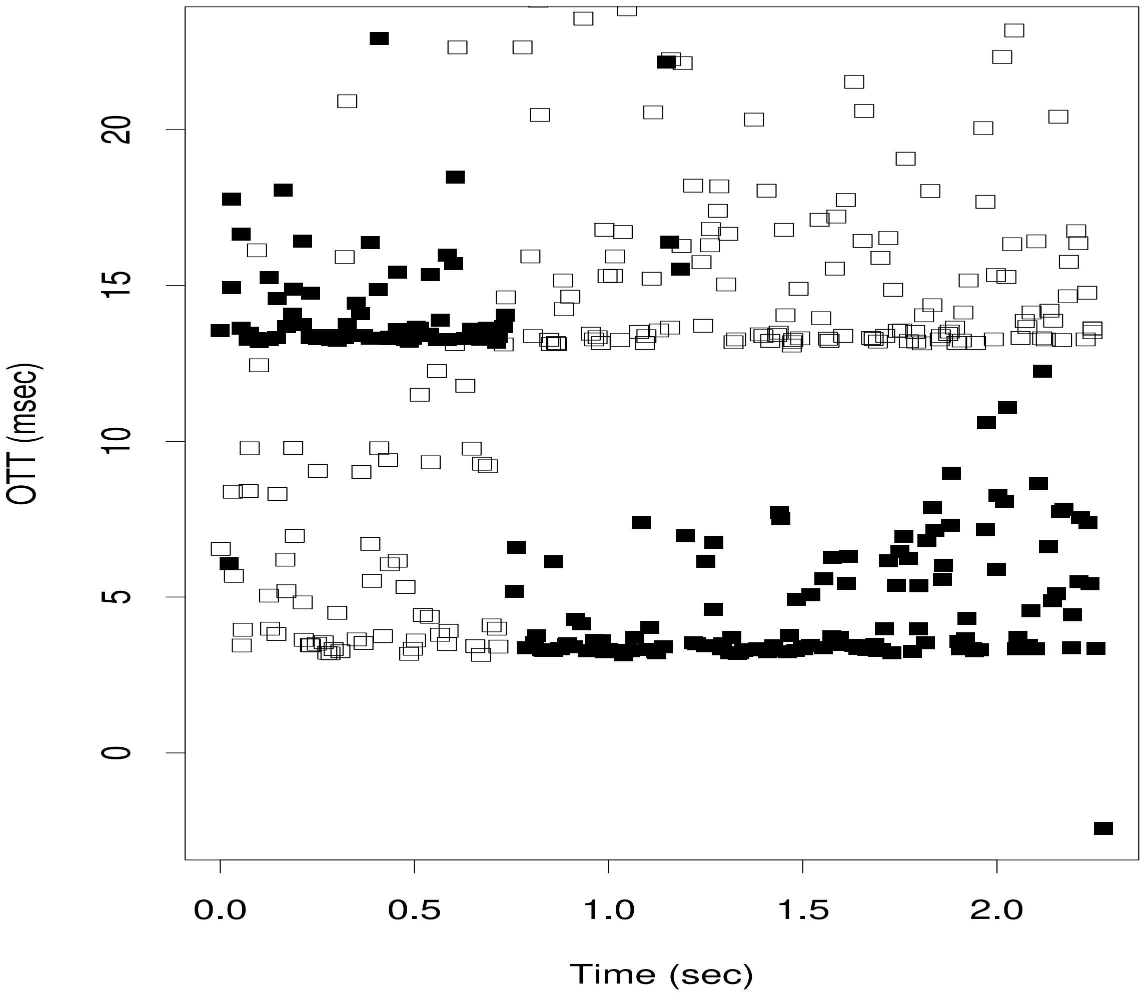
Time travel: 0.14% of $\mathcal{N}_1$ traces, 2.9% of $\mathcal{N}_2$.

Site-dependent.

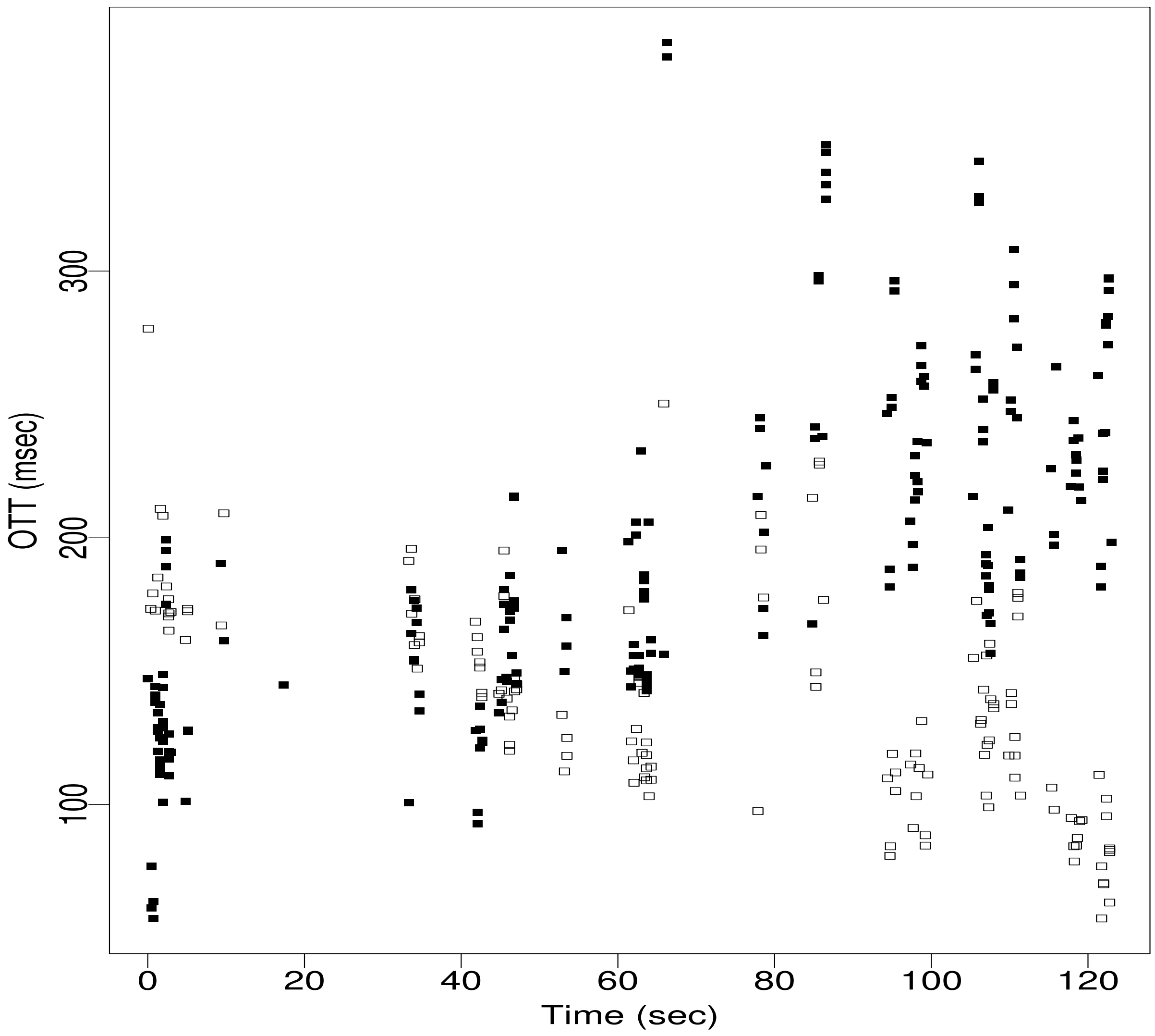
Forward adjustments: tricky to detect, but `tcpanaly` can do it.

Resolution: ≈ 50 μsec to 10 msec.

Important to propagate resulting uncertainties.



OTT (msec)
0
5
10
15
20
0.0
0.5
1.0
1.5
2.0
Time (sec)



OTT (msec)
100
200
300
0
20
40
60
80
100
120
Time (sec)

Clock headaches, con't:

Skew: tricky to assess, due to network dynamics, but essential to avoid misinterpreting as queueing.

How bad can skew get? 450%!

Analysis headaches — TCP behavior:

Crucial to understand packet cause & effect: to assess RTT's, measurement drops, unusual network behavior.

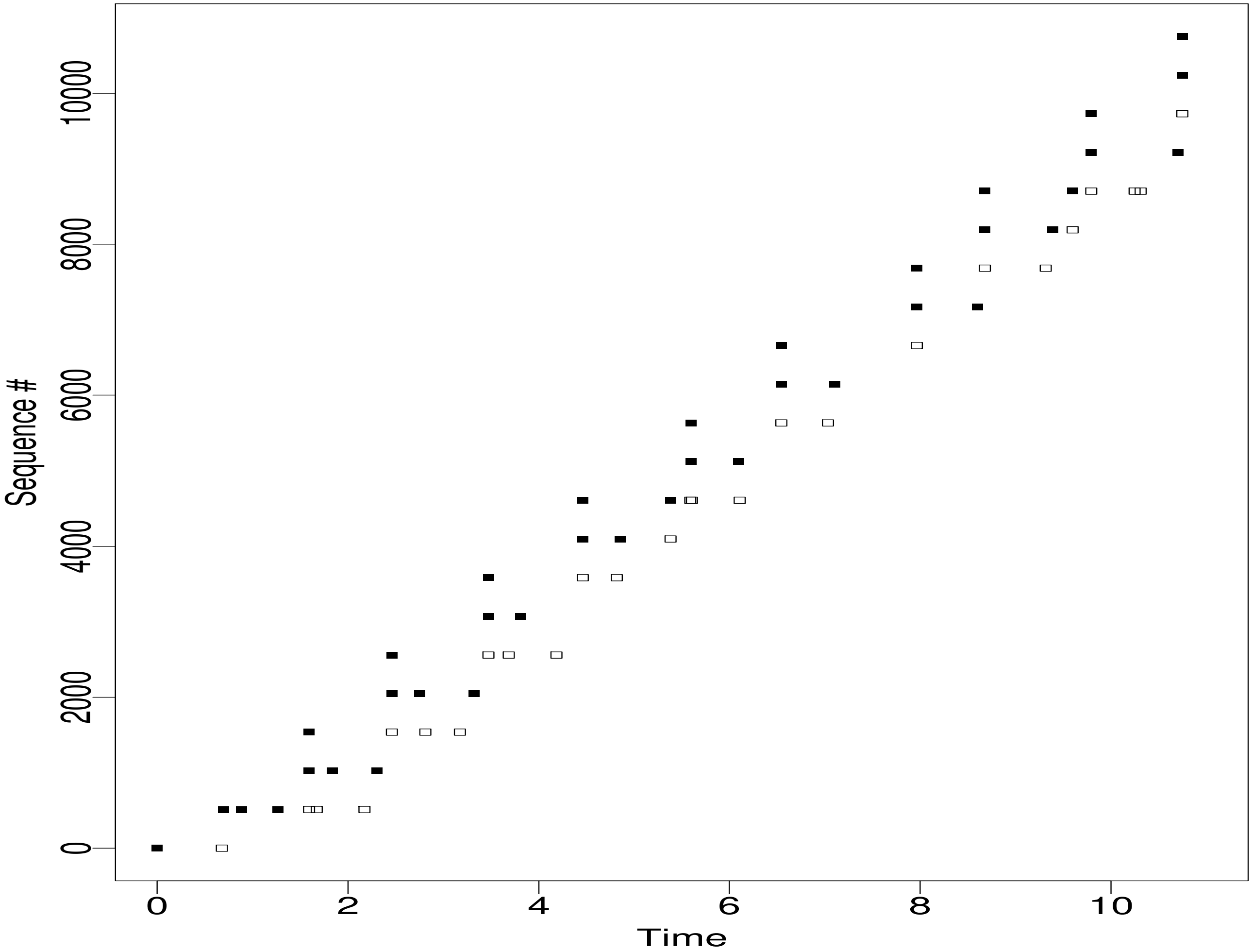
`tcpanaly` knows about 9 different TCPs:

SunOS, Solaris 2.3/2.4, Digital OSF/1, IRIX 4.0/5.1/5.2/6.2,
BSDI 1.1/2.0/2.1, NetBSD 1.0, HP/UX 9.05/10.00,
VJ 1/2, Linux 1.0

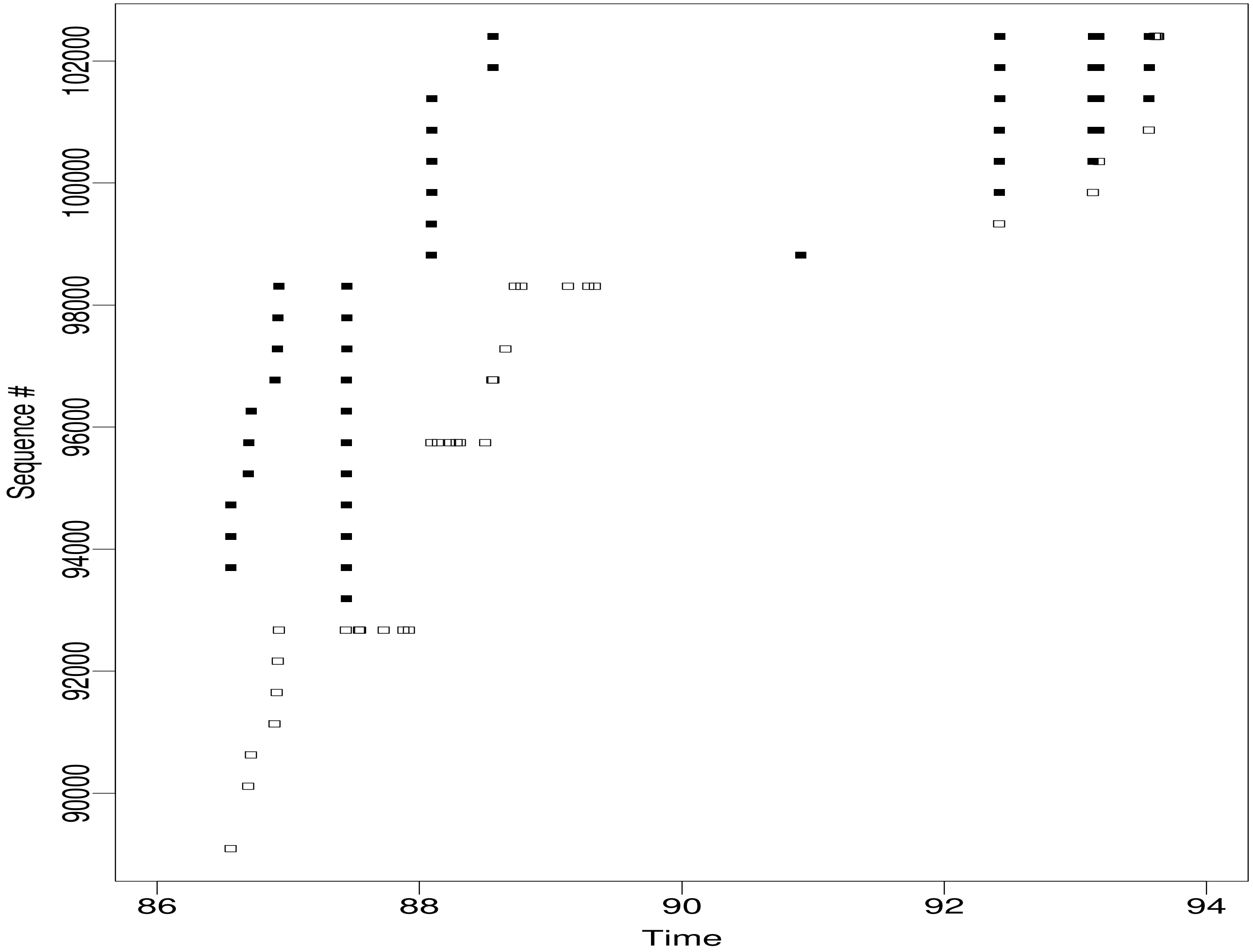
All but Solaris and Linux are Tahoe/Reno variants.

Can infer checksum errors, ICMP Source Quench.

Solaris 2.3 sender, RTT = 680 msec:



Linux 1.0 sender:



TCP pathologies:

Broken retransmit timers: Solaris 2.3/2.4! (FIXED in 2.5)

Entire window retrans. upon loss: Linux 1.0! (FIXED in 2.X)

No slow start: Trumpet/Winsock 3.0! (FIXED?)

Throws away out-of-sequence packets: Trumpet/Winsock!

If the whole world was Trumpet/Winsock 3.0 or Linux 1.0,

Metcalfe would be right.

$\Rightarrow$ Independent TCP implementation is <u>really hard</u> . . .

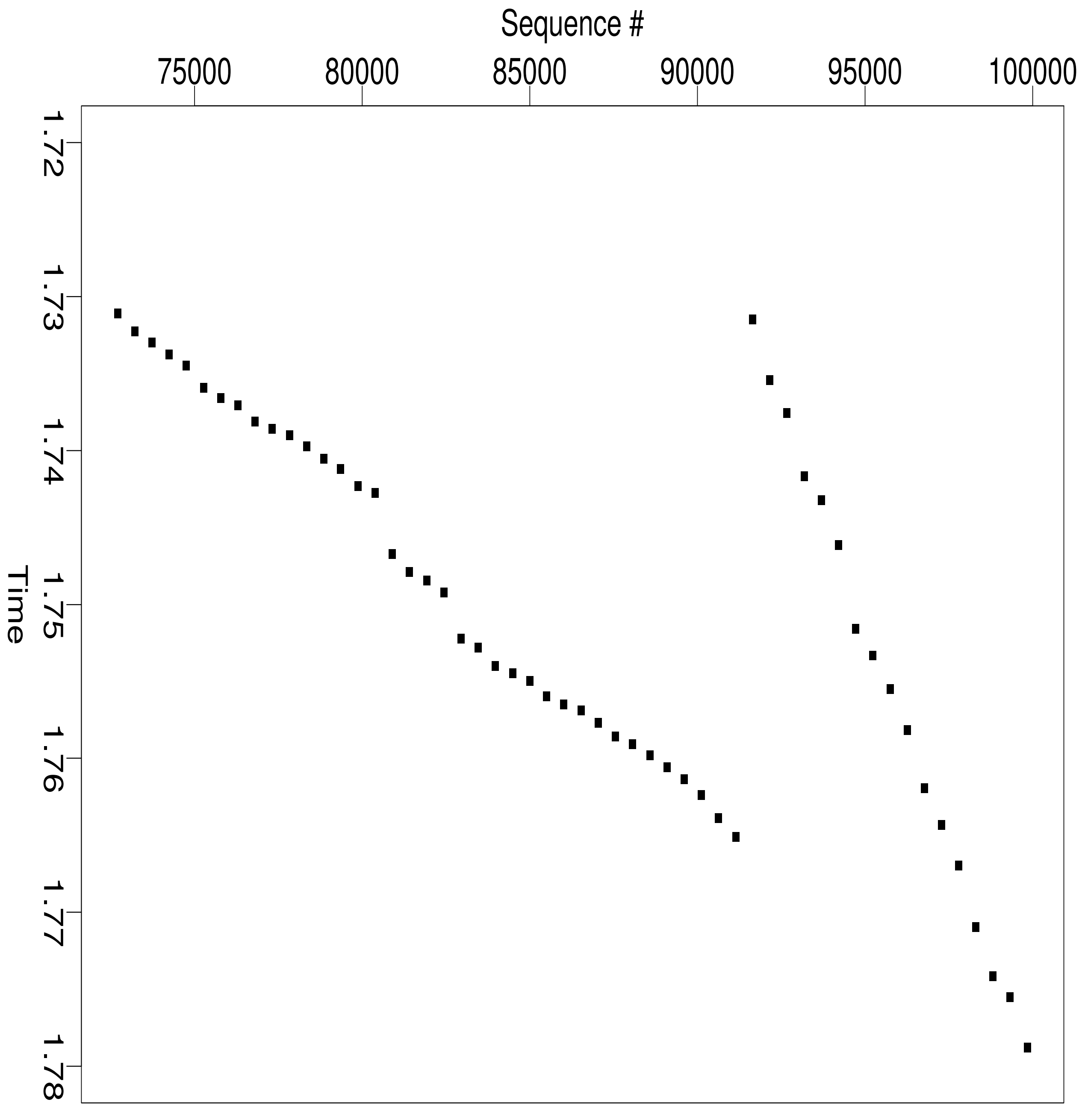
. . . but vital to Internet stability.

Network pathologies — out-of-order data delivery:

36% of traces in $\mathcal{N}_1$, 12% of traces in $\mathcal{N}_2$.
2% of packets in $\mathcal{N}_1$, 0.6% in $\mathcal{N}_2$.

But: strongly site-specific.
Worse site: 24% of all packets sent!
Biggest delay: 9 seconds!
Biggest gap: 54 packets!

Impact on performance: generally, <u>minimal</u>.
Could you lower the “dup ack” threshold? Might as well deploy SACK instead.


Sequence #
75000
80000
85000
90000
95000
100000
Time
1.72
1.73
1.74
1.75
1.76
1.77
1.78

Network pathologies — rare & unusual:

Packet replication:

In $\mathcal{N}_1$, happened once: 2 packets in, 16 out.

In $\mathcal{N}_2$, 65 times. Most striking: 1 packet in, 22 out.

Checksum errors: 1 pkt in 5,000 (after removing outlier).

BER: $4.5 \cdot 10^{-8}$.

Bad packets accepted: 1 in 300,000,000.

Packet gaining an ack field in flight: once!

Packet gaining data in flight: twice!

Bottleneck bandwidth:

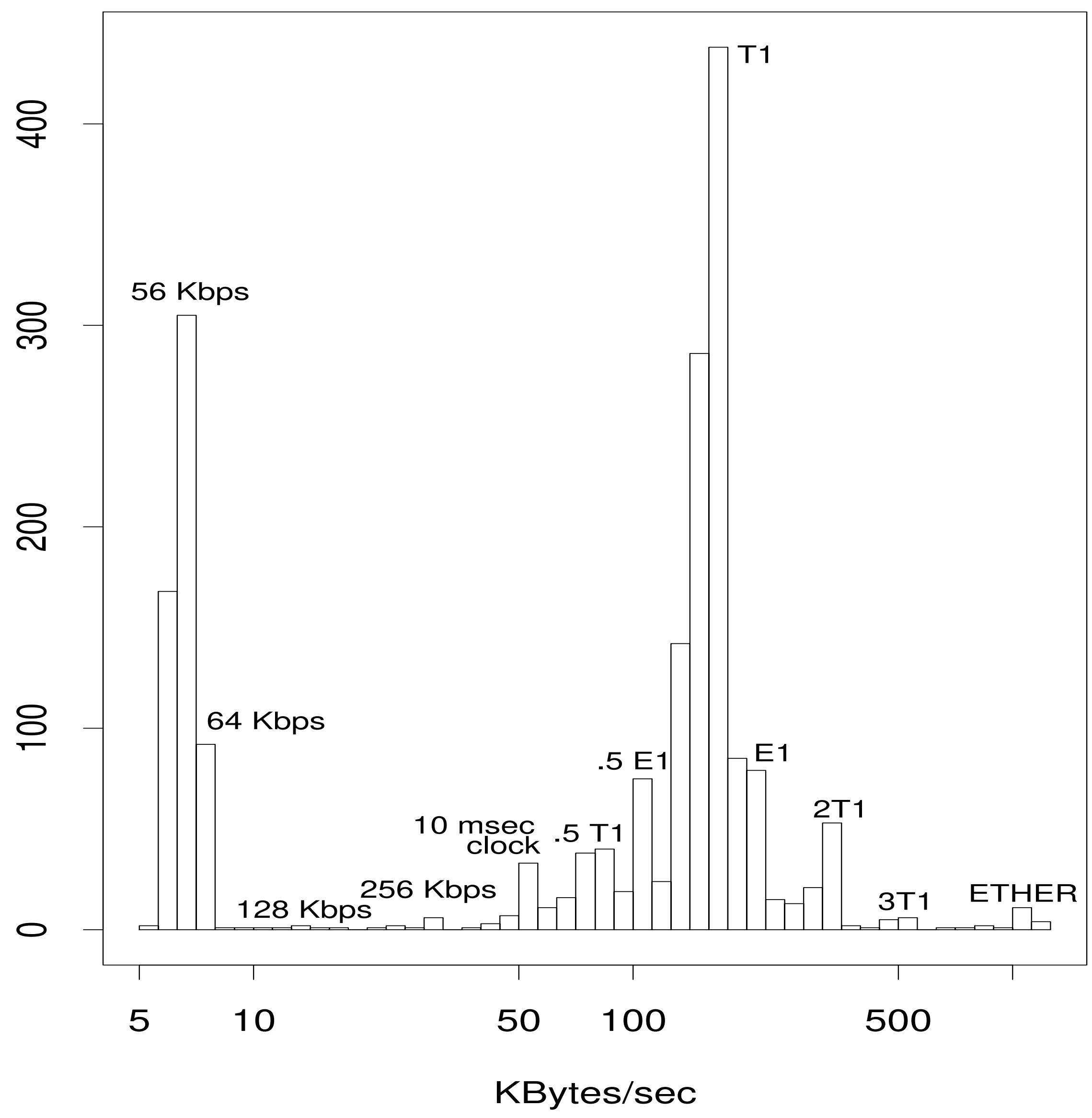
Limiting rate on a connection: the <u>bottleneck</u> bandwidth.

Different from <u>available</u> bandwidth.

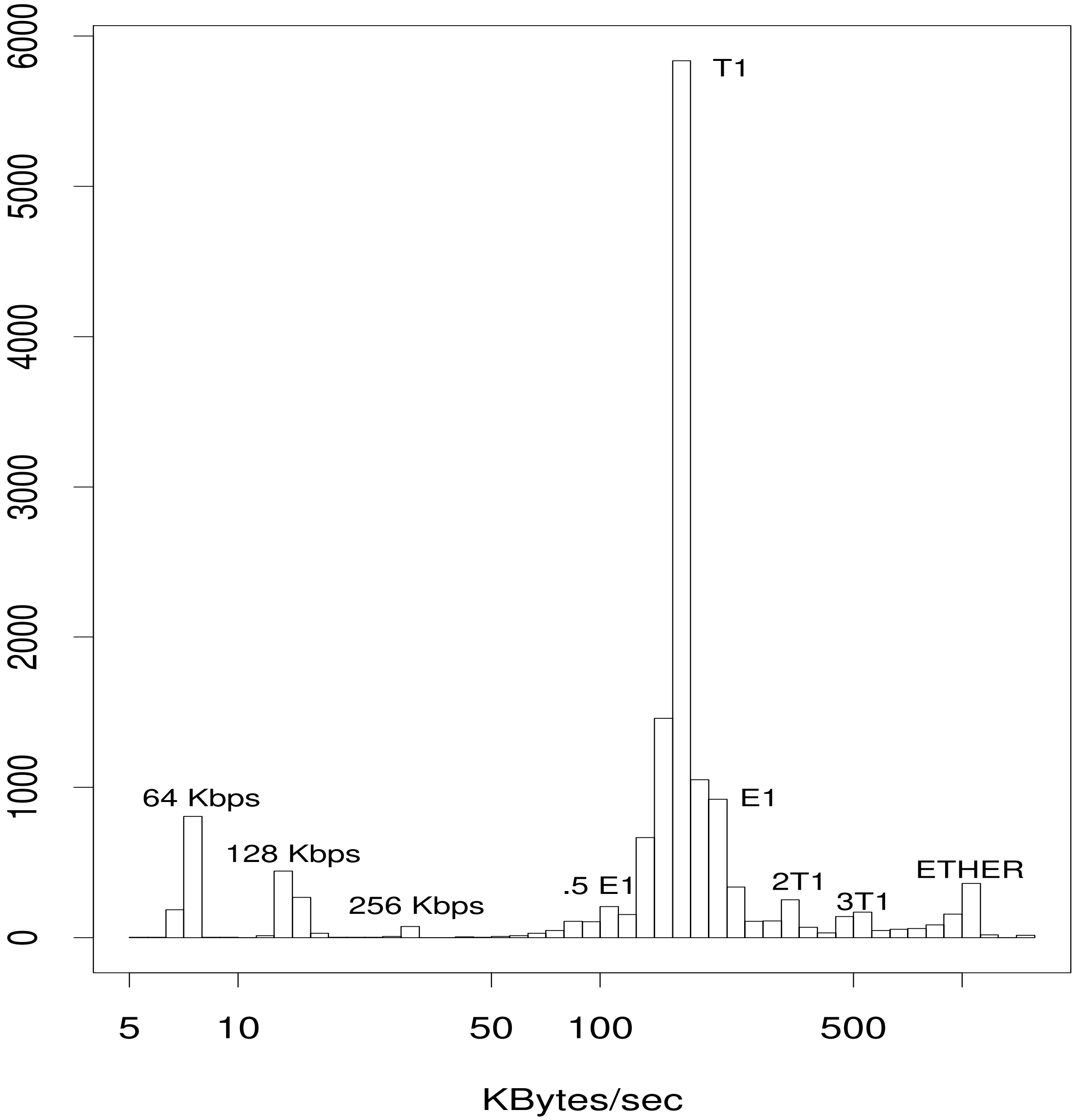
If you know it $\Rightarrow$ self-interference time constant, Q_b.

Robust estimation based on “packet bunch” and searching for mode or <u>modes</u>.

Measuring at the <u>receiver</u> a big win here.



56 Kbps
64 Kbps
128 Kbps
256 Kbps
10 msec
clock
.5 T1
.5 E1
T1
E1
2T1
3T1
ETHER
0
100
200
300
400
5
10
50
100
500
KBytes/sec



6000
5000
4000
3000
2000
1000
0
T1
64 Kbps
128 Kbps
256 Kbps
.5 E1
E1
2T1
3T1
ETHER
5
10
50
100
500
KBytes/sec

Packet loss:

Overall loss rate (sites in common): 2.7% in $\mathcal{N}_1$, 5.2% in $\mathcal{N}_2$.

But: need to account for bigger $\mathcal{N}_2$ windows.

Do this by looking at <u>ack</u> loss rates: 2.88% vs. 5.14%.

Loss-free (“quiescent”) connections: 52% vs. 49%.

Ack loss-rate if not “quiescent” (“busy”): 5.7% vs. 9.2%.

Packet loss by region:

Region	$Quies_1$	$Quies_2$	$Busy_1$	$Busy_2$	Δ
Europe	48%	58%	5.3%	5.9%	$+11\%$
U.S.	66%	69%	3.6%	4.4%	$+21\%$
Into Europe	40%	31%	9.8%	16.9%	$+73\%$
Into U.S.	35%	52%	4.9%	6.0%	$+22\%$
All regions	53%	52%	5.6%	8.7%	$+54\%$

Packet loss patterns:

Type of loss	P_l^u		P_l^c	
	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$
"Loaded" data pkt	2.8%	4.5%	49%	50%
"Unloaded" data pkt	3.3%	5.3%	20%	25%
Ack	3.2%	4.3%	25%	31%

$\Rightarrow$ Loss not well described as independent.

$\Rightarrow$ Loss <u>outages</u> have <u>infinite variance</u>.

Redundant Retransmissions (RRs):

Type of RR	Sol. 2.3/2.4$_1$	Sol. 2.3/2.4$_2$	Other$_1$	Other$_2$
% all packets	6%	6%	1%	2%
% retrans.	66%	59%	26%	28%
Unavoidable	14%	33%	44%	17%
Coarse feed.	1%	1%	51%	80%
Bad RTO	84%	66%	4%	3%

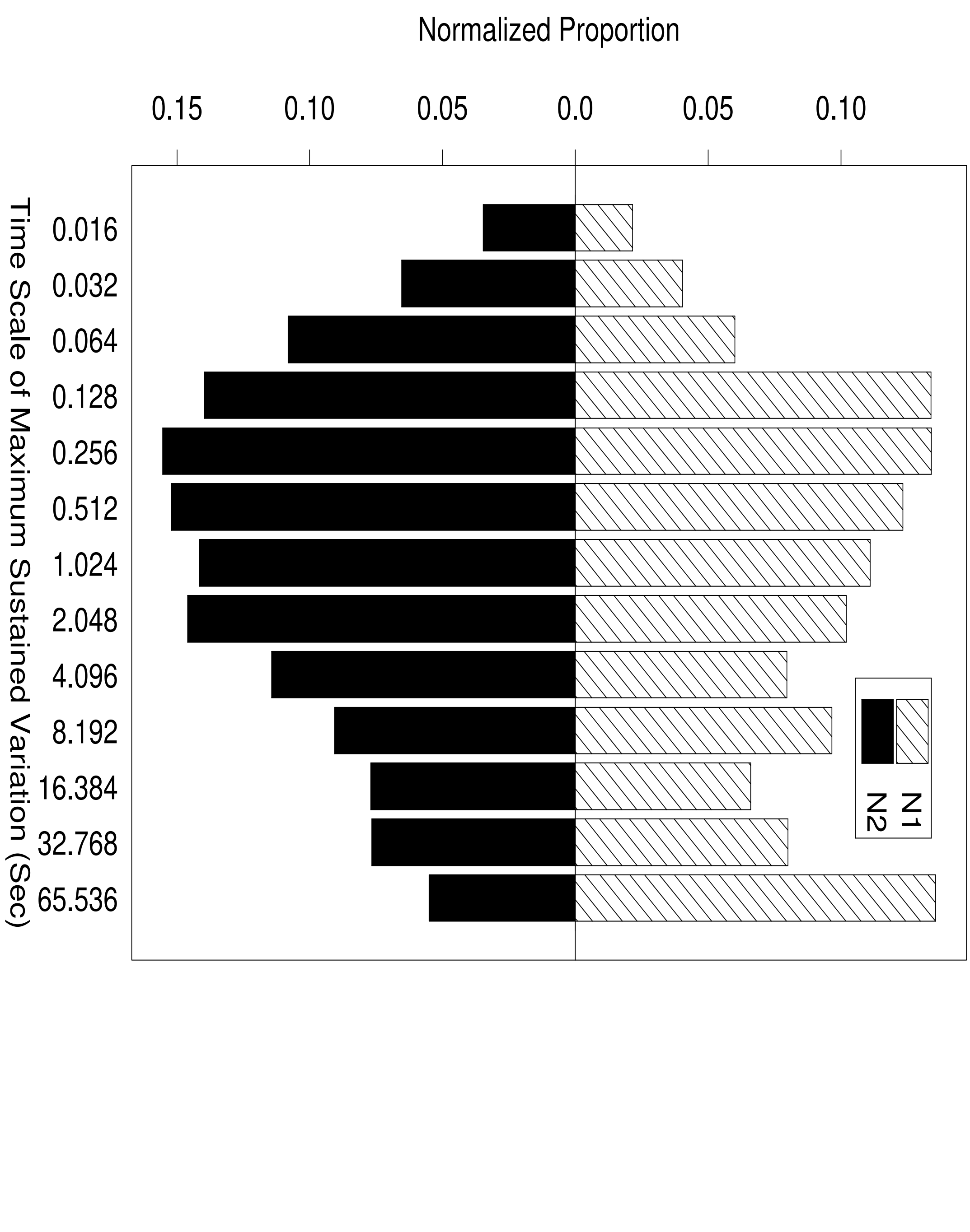
$\Rightarrow$ Fixing timers, deploying SACK fixes the vast majority.

One-way transit time (OTT):

For transport protocol dynamics, RTT is of great interest.
For network dynamics, OTT variations more relevant.

If: eliminate bad clocks, reordered packets, compressed packets, traces with TTL shifts . . .
. . . then: argue that remaining OTT variation reflects queueing/congestion.

Look for $\hat{\tau}$, the time scale of the most variation.


Normalized Proportion
0.15
0.10
0.05
0.0
0.05
0.10
Time Scale of Maximum Sustained Variation (Sec)
0.016
0.032
0.064
0.128
0.256
0.512
1.024
2.048
4.096
8.192
16.384
32.768
65.536
N1
N2

Available bandwidth:

Idea: compare OTT increase due to connection's own packets, vs. total OTT increase:

$$\alpha = \frac{\sum_i \psi_i}{\sum_j \gamma_j}.$$

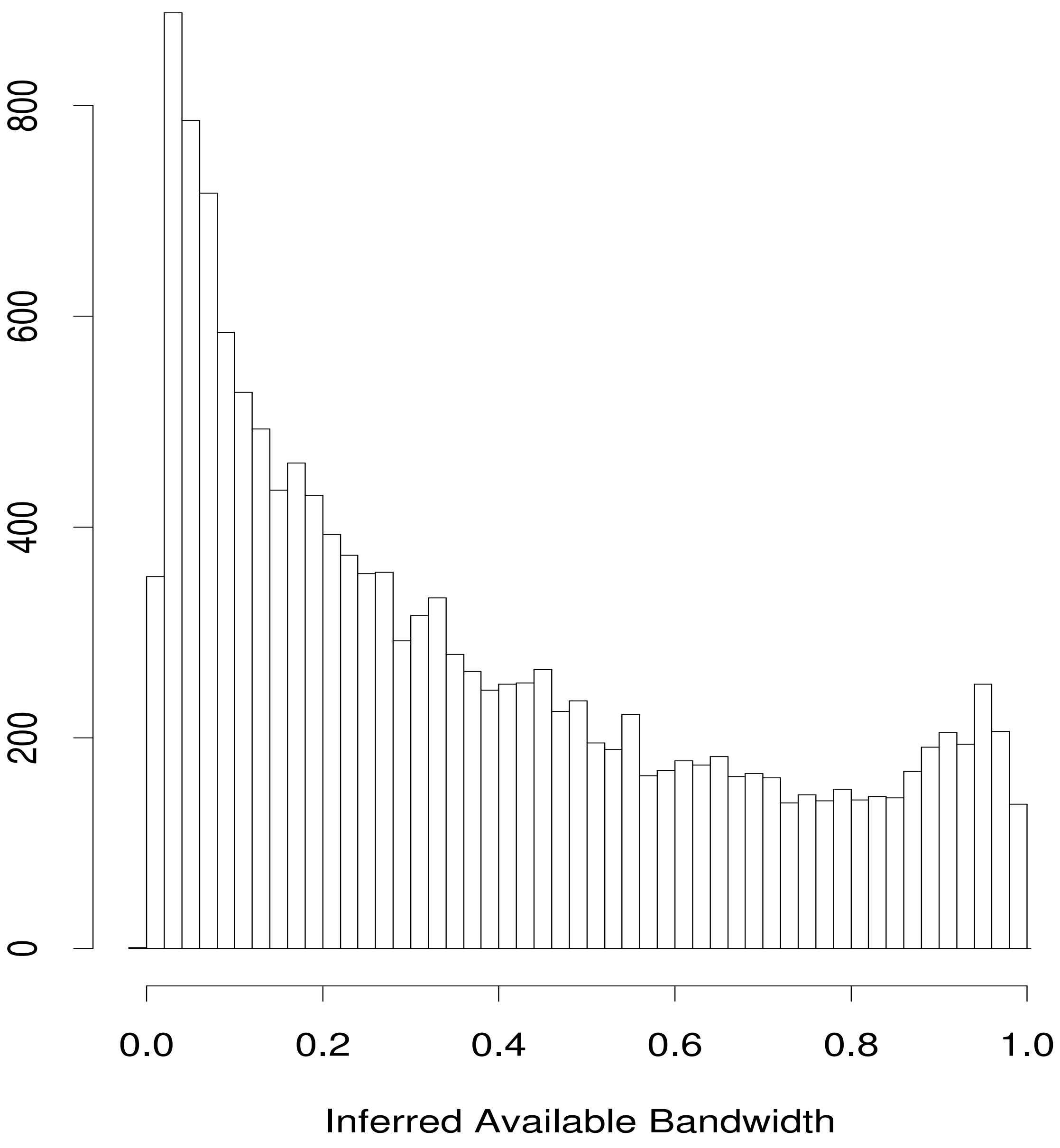
If $\alpha \approx 1$, then almost all delay due to connection's own load:

$\Rightarrow$ network is unloaded.

If $\alpha \approx 0$, then almost none of delay due to connection's load:

$\Rightarrow$ network heavily loaded.

α correlates strongly with normalized, overall throughput.


0
200
400
600
800
0.0
0.2
0.4
0.6
0.8
1.0
Inferred Available Bandwidth

The Future:

Building “Internet measurement infrastructure.”

(NSF funding, w/ Matt Mathis & Jamshid Mahdavi)

Idea: run industrial-strength NPDs at a wide range of sites.

Conduct large-scale measurements of Internet clouds.

Archive and electronically publish the results.

Modular architecture: plug in different measurements (IPPM).

Circa next NANOG, we'll be looking for volunteers . . .

Acknowledgements:

Guy Almes and Bob Camm (`adv`); Jos Alsters (`unij`); Jean-Chrysostome Bolot (`inria`); Hans-Werner Braun, Kim Claffy, and Bilal Chinoy (`sdsc`); Randy Bush (`rain`); Jon Crowcroft and Atanu Ghosh (`ucl`); Peter Danzig and Katia Obraczka (`usc`); Mark Eliot (`sri`); Robert Elz (`austr`); Teus Hagen (`oce`); Steinar Haug and Håvard Eidnes (`sintef1`, `sintef2`); John Hawkinson (`near` and `panix`); TR Hein (`xor`); Tobias Helbig and Werner Sinze (`ustutt`); Paul Hyder (`ncar`); Alden Jackson (`sandia`); Kate Lance (`austr2`); Craig Leres (`lbl`); Kurt Lidl (`pubnix`); Peter Linington, Alan Ibbetson, Peter Collinson,

Acknowledgements, con't:

and Ian Penny (`ukc`); Steve McCanne (`lbli`); John Milburn (`korea`); Walter Mueller (`umann`); Evi Nemeth, Mike Schwartz, Dirk Grunwald, Lynda McGinley (`ucol`, `batman`); Francóis Pinard (`umont`); Jeff Polk and Keith Bostic (`bsdi`); Todd Satogata (`bnl`); Doug Schmidt and Miranda Flory (`wustl`); Sorell Slaymaker and Alan Hannan (`mid`); Don Wells and Dave Brown (`nrao`); Gary Wright (`connix`); John Wroclawski (`mit`); Cliff Young and Brad Karp (`harv`); and Lixia Zhang, Mario Gerla, and Simon Walton (`ucla`).