pathchar — a tool to infer characteristics of Internet paths

Van Jacobson
(van@ee.lbl.gov)

Network Research Group
Lawrence Berkeley National Laboratory
Berkeley, CA 94720

MSRI
April 21, 1997

Some history

1980 ping (Mills, et.al) — measure source–destination round-trip-time.

1985 tcp tuning (Karels & Jacobson) — measure bottleneck bandwidth.

1987 snmp (Case) — intra-domain router monitoring.

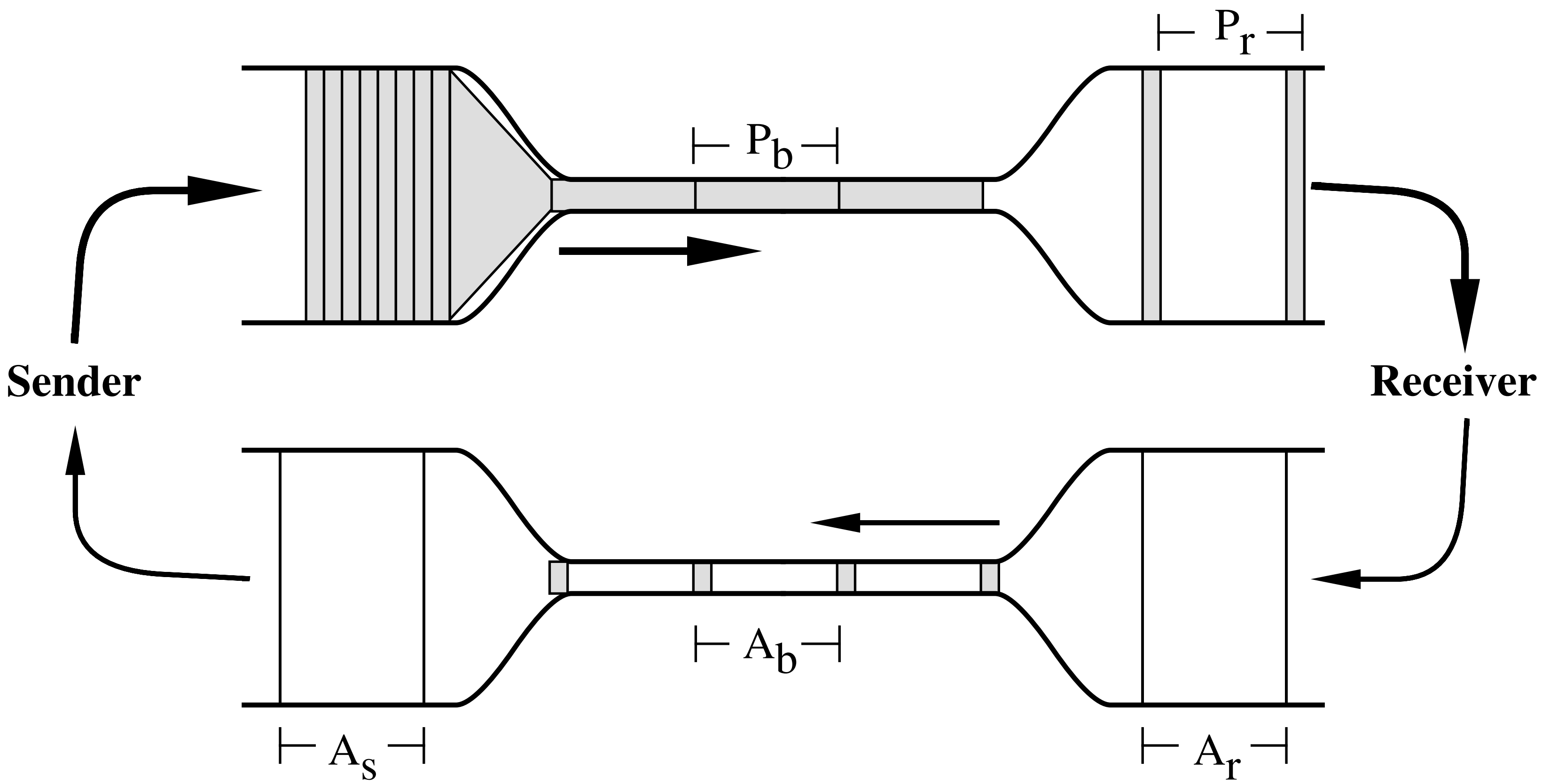
1988 traceroute (Jacobson) — enumerate source–destination path.

1991 first working version of pathchar.

1996 tcpanaly (Paxson) — measure characteristics of selected paths.

1997 first pathchar release.

Bottleneck bandwidth estimation
(and why it doesn't generalize)



Path discovery - traceroute

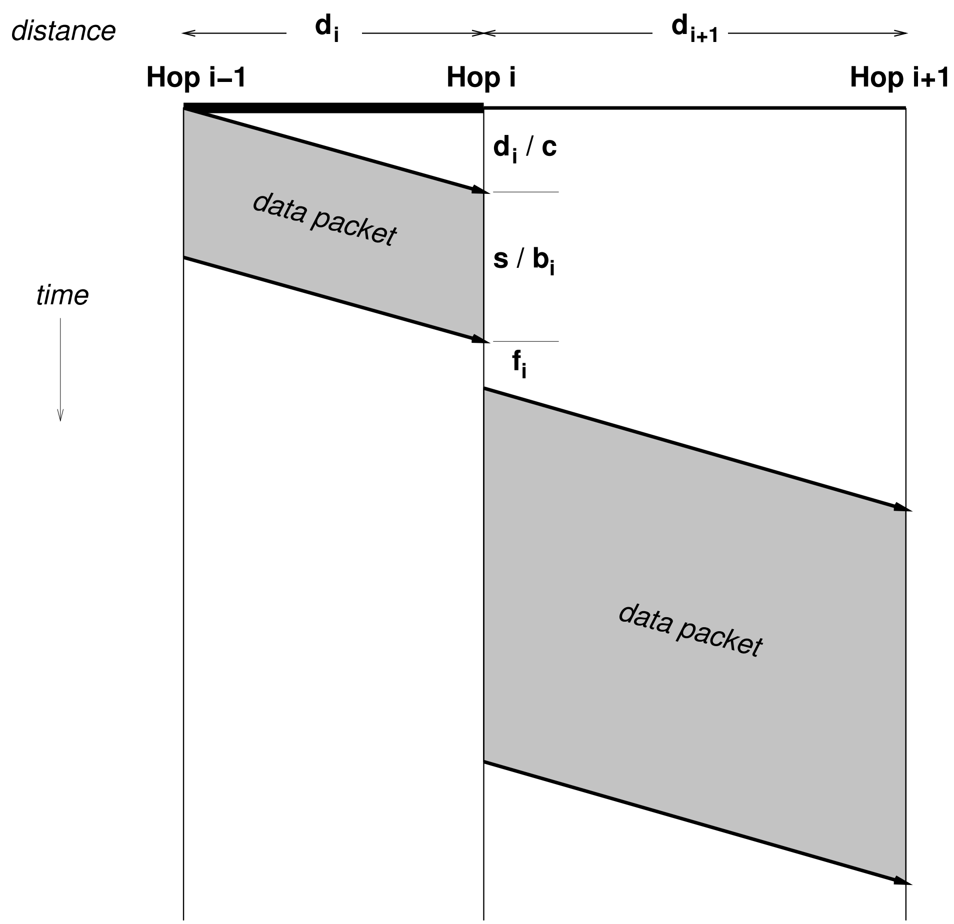
```
% traceroute ka9q.ampr.org
 1  ir40gw.lbl.gov (131.243.1.1)  1.773 ms  1.371 ms  1.096 ms
 2  er1gw.lbl.gov (131.243.128.11)  2.355 ms  1.649 ms  1.503 ms
 3  lbl-lc1-1.es.net (198.128.16.11)  2.628 ms  2.501 ms  1.897 ms
 4  gac-atms.es.net (134.55.24.6)  151.258 ms  150.189 ms  147.582 ms
 5  CERFNETGWY.GAT.COM (192.73.7.163)  143.865 ms  147.481 ms  145.009 ms
 6  * sdsc-ga.cerf.net (134.24.20.15)  139.885 ms  173.316 ms
 7  mobydick-fddi.cerf.net (134.24.252.3)  156.924 ms *  287.051 ms
 8  qualcomm-sdsc-ds3.cerf.net (134.24.47.200)  136.821 ms *  147.951 ms
 9  * krypton-e2.qualcomm.com (192.35.156.2)  148.756 ms  131.125 ms
10  ascend-max.qualcomm.com (129.46.54.31)  158.475 ms  143.450 ms  146.958 ms
11  karnp50.qualcomm.com (129.46.90.33)  3435.416 ms *  173.813 ms
12  unix.ka9q.ampr.org (129.46.90.35)  196.909 ms  176.182 ms  188.959 ms
```

How does traceroute work?

IP packets contain a *time-to-live* field that is initialized by the original sender then decremented by one at each intermediate router. If the field is decremented to zero, the packet is discarded and an error indication packet (an ICMP "time exceeded") is sent back to the original sender.

The source address of the ICMP "time exceeded" identifies the router that discarded the data packet. So if packets are sent to the final destination but with the ttl set to n, the router n hops along the path is forced to identify itself.

A model for per-hop forwarding time



Forwarding time to hop n for packet of size s:

$$T(n,s) = \sum_{i=1}^{n} \left[\frac{s}{b_i} + \frac{d_i}{c} + f_i \right]$$

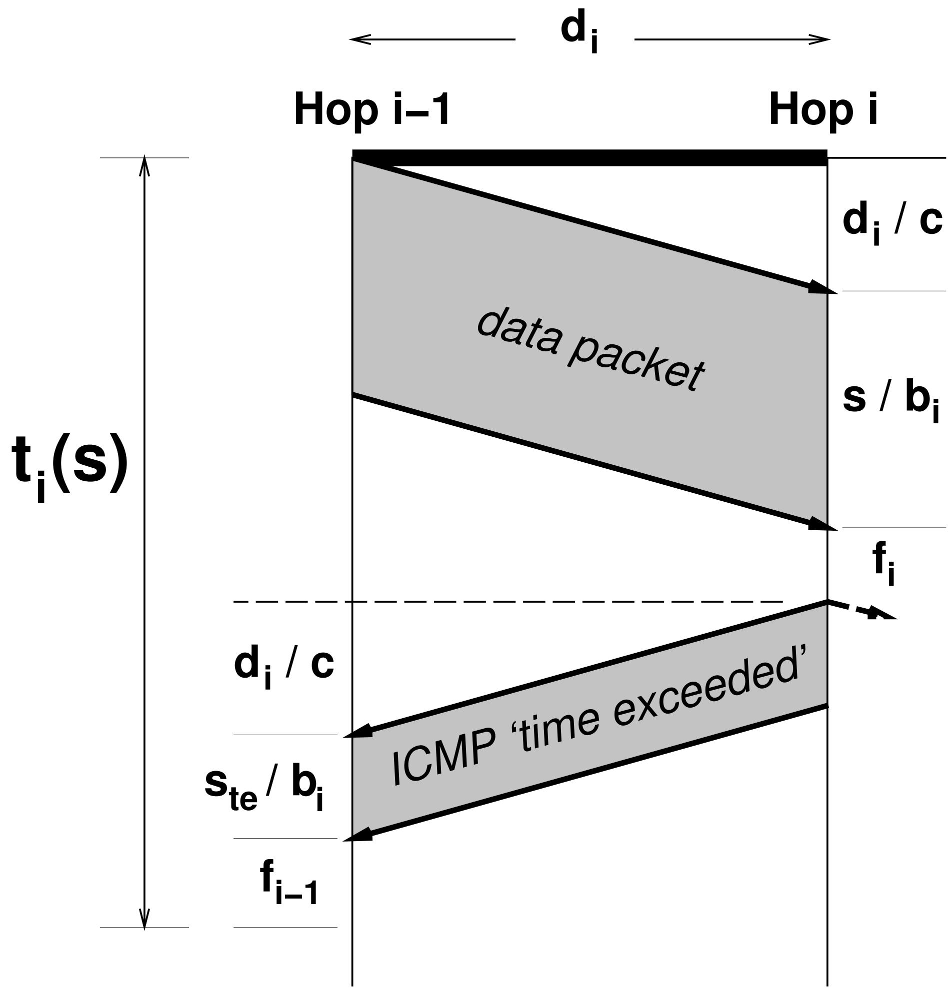
A model for ICMP-time-exceeded

$$t_i(s) = \frac{s}{b_i} + \frac{d_i}{c} + f_i + \frac{s_{te}}{b_i} + \frac{d_i}{c} + f_{i-1}$$

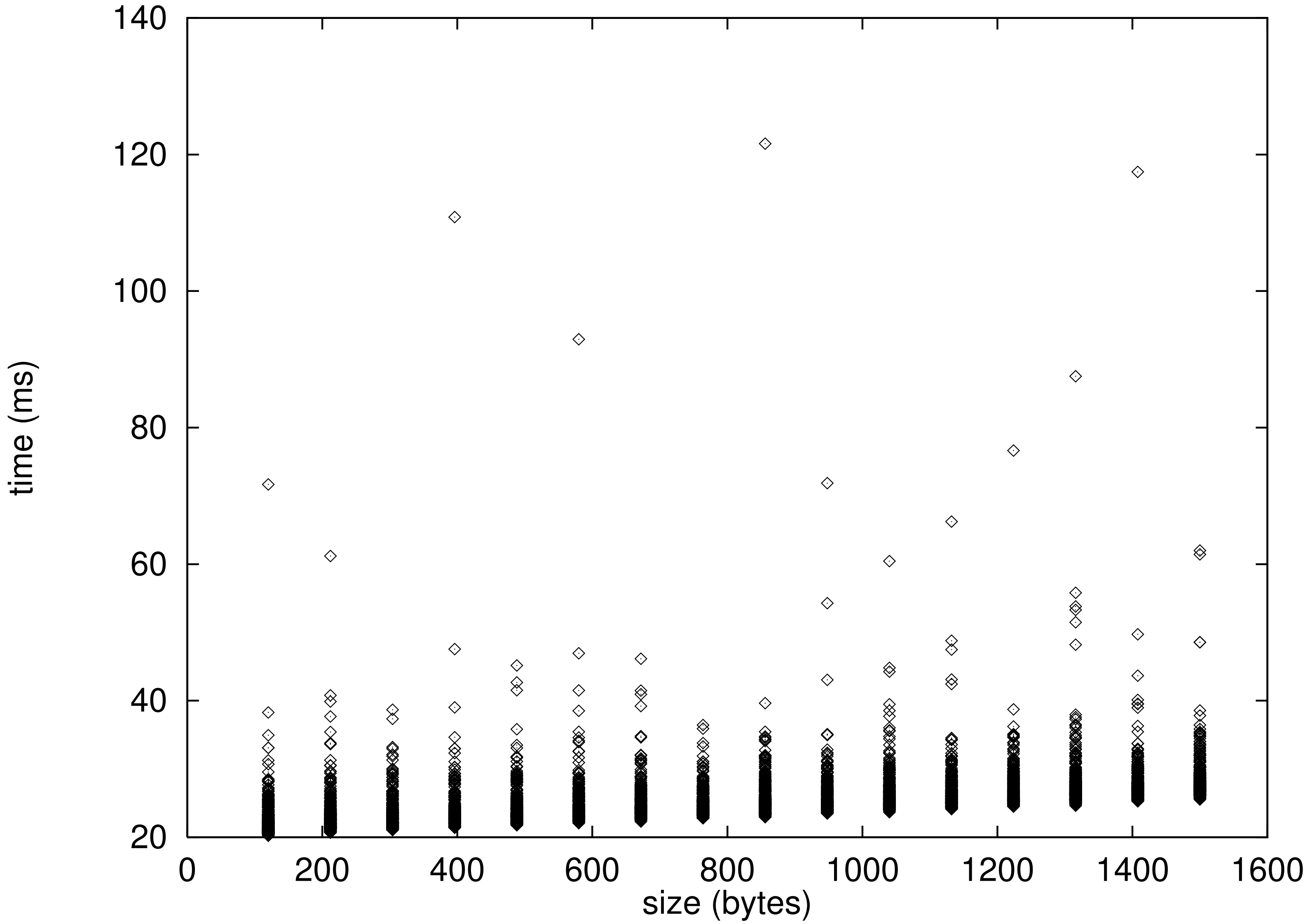
and

$$t_i(s) \approx \frac{s}{b_i} + \frac{s_{te}}{b_i} + 2\left[\frac{d_i}{c} + f_i\right]$$

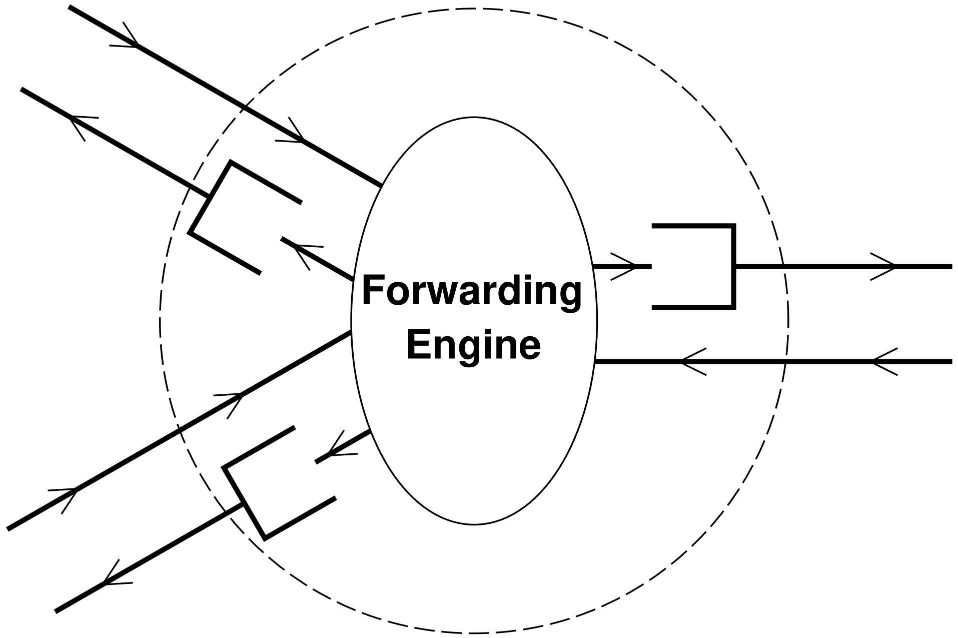
(if $f_{i-1} \approx f_i$)



Raw measured data for one hop



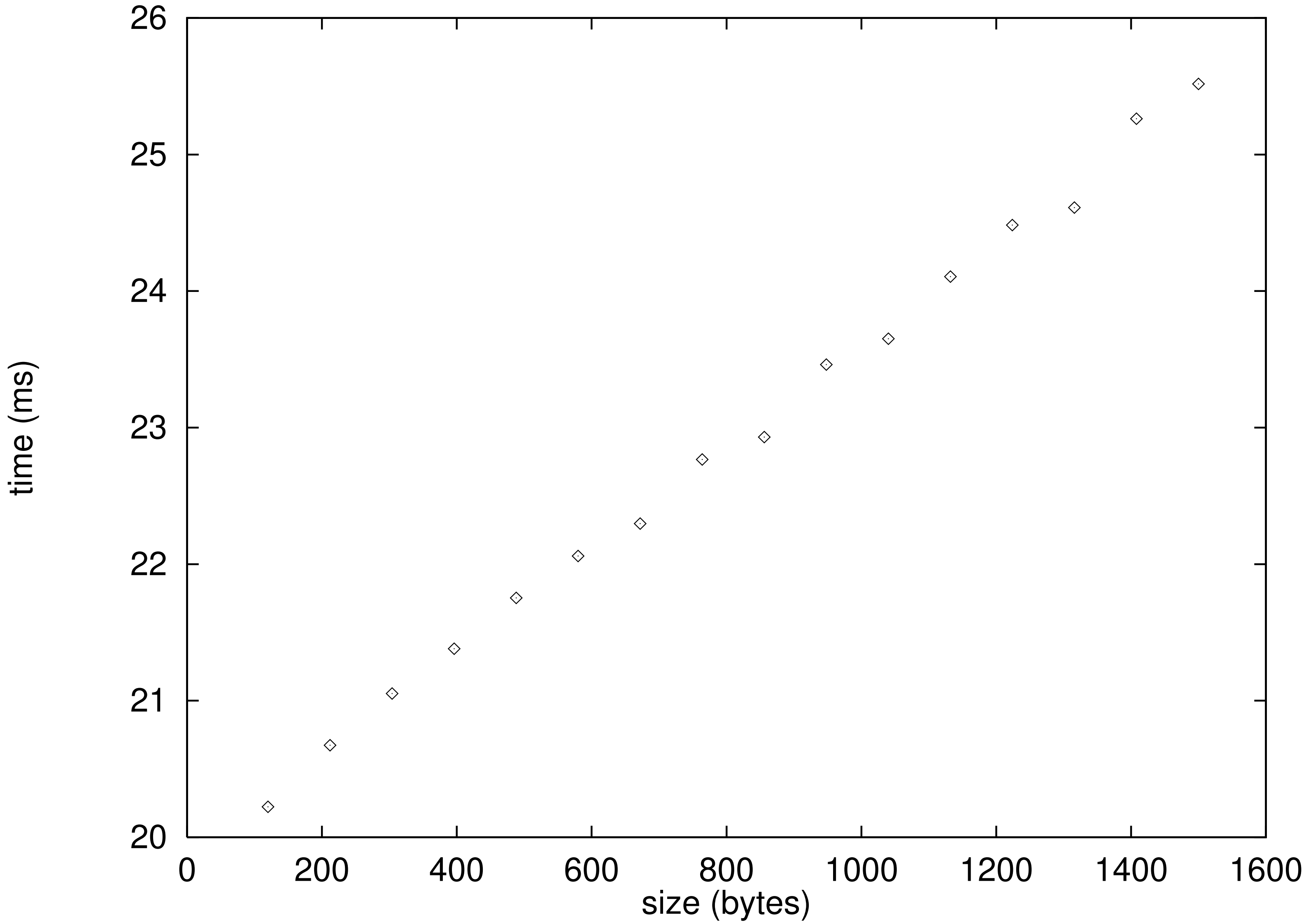
What went wrong? (we left out the queues)



A router contains links, a forwarding engine and queues. Our model included the (deterministic) links and forwarding but not the random, unpredictable, queuing time due to competing traffic.

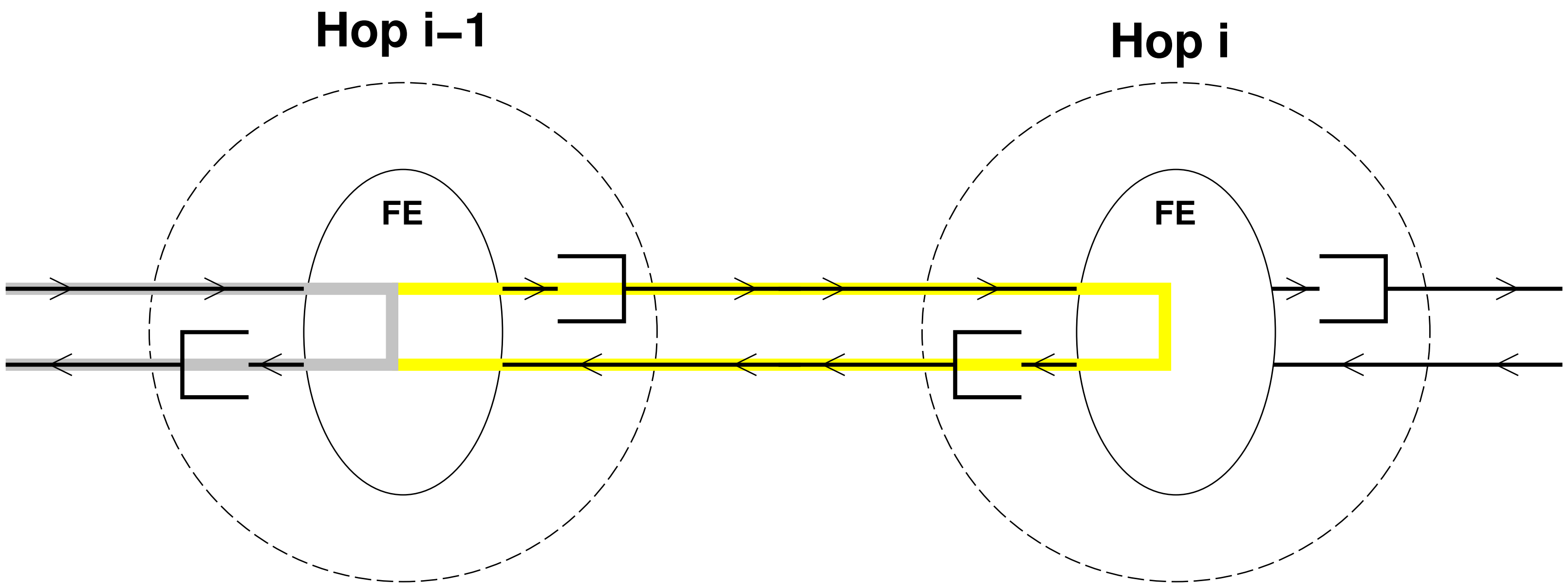
But queuing can only add to the deterministic time. If we take a lot of samples at well spaced, random times, the ensemble min should approximate our forwarding model.

Min-filtered data for the same hop



Note this works both ways — the difference between the unfiltered data and the ensemble min is the queuing time distribution. So one set of raw measurements gives the per-hop bandwidth, prop delay, queue time and drop rate.

Each time the ttl increases by one, we measure two new links, two new queues and one new forwarding engine:



(The gray shade marks the measurement path for hop $n-1$. The yellow is what's added for ttl n.)

Unfortunately, the queue and drop rate estimates are for the *convolution* of this hop's behavior with all the upstream queues on both the forward and reverse path.

I.e., a drop rate of $d\%$ and/or queue of q ms. at hop n will result in drop rates and queues of at least d and q for all hops $m > n$.

Convolved estimates are useful but isolating each hop's behavior, as is done for the bandwidth and delay estimates, would be much nicer (see 'future work').

pathchar example

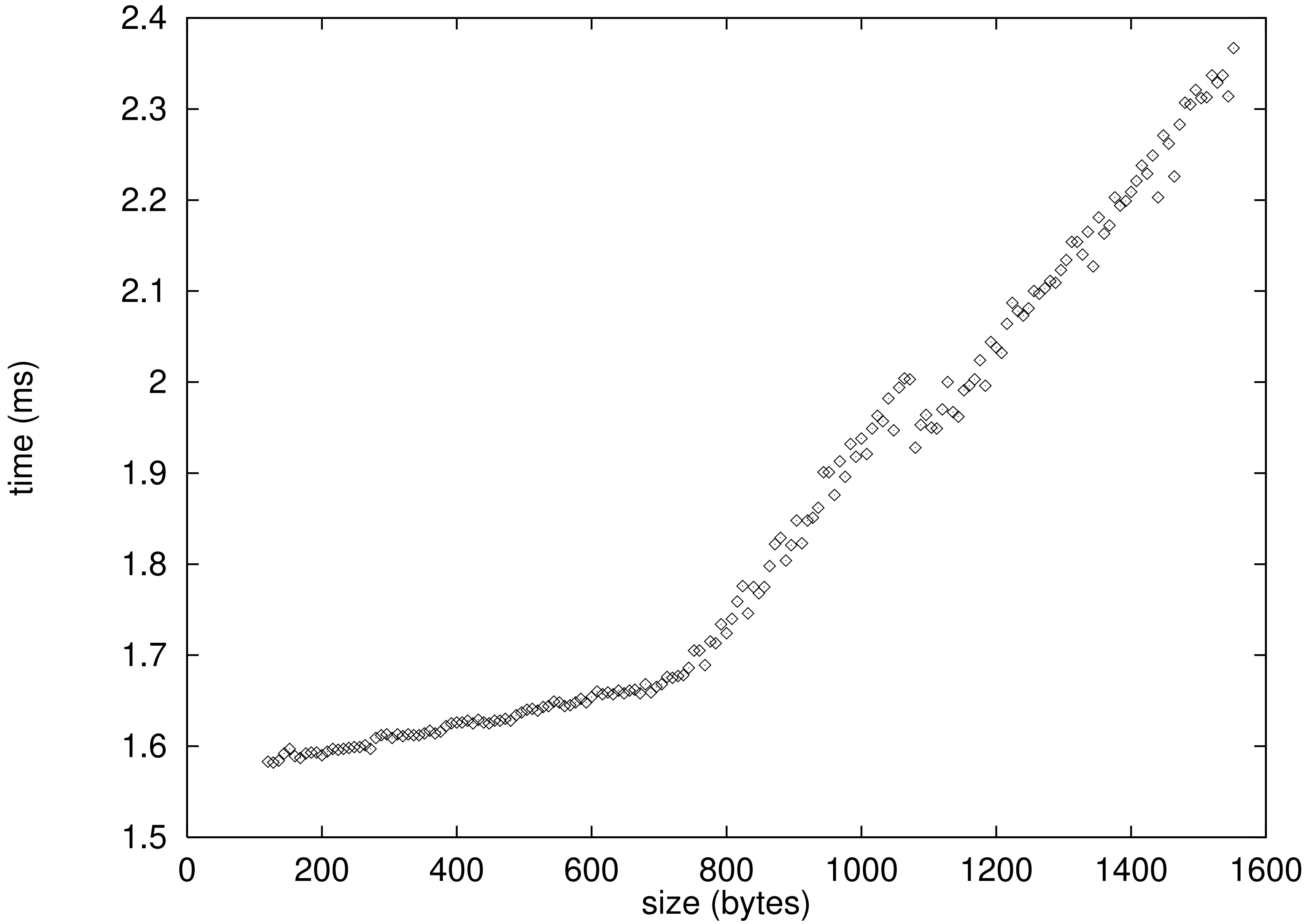
```
% pathchar ka9q.ampr.org
 0 localhost
 |   8.7 Mb/s,   292 us (1.97 ms)
 1 ir40gw.lbl.gov (131.243.1.1)
 |    29 Mb/s,   132 us (2.64 ms)
 2 er1gw.lbl.gov (131.243.128.11)
 |    91 Mb/s,   189 us (3.15 ms)
 3 lbl-lc1-1.es.net (198.128.16.11)
 |    25 Mb/s,   6.92 ms (17.5 ms)
 4 gac-atms.es.net (134.55.24.6)
 |    11 Mb/s,   424 us (19.4 ms)
 5 CERFNETGWY.GAT.COM (192.73.7.163)
 |   7.0 Mb/s,   1.20 ms (23.5 ms)
 6 sdsc-ga.cerf.net (134.24.20.15)
```

pathchar example (cont.)

```
 6 sdsc-ga.cerf.net (134.24.20.15)
 |   ?? b/s,   -263 us (22.8 ms)
 7 mobydick-fddi.cerf.net (134.24.252.3)
 |    46 Mb/s,   -51 us (22.9 ms)
 8 qualcomm-sdsc-ds3.cerf.net (134.24.47.200)
 |   9.0 Mb/s,   17 us (24.3 ms)
 9 krypton-e2.qualcomm.com (192.35.156.2)
 |   5.5 Mb/s,   1.04 ms (28.6 ms)
10 ascend-max.qualcomm.com (129.46.54.31)
 |   56.7 Kb/s,   6.19 ms (253 ms)
11 karnp50.qualcomm.com (129.46.90.33)
 |    10 Mb/s,   -50 us (253 ms),  +q 3.32 ms (6.58 KB) *2
12 unix.ka9q.ampr.org (129.46.90.35)

rtt 32 ms (253 ms), bottleneck 56.7 Kb/s, pipe 4706 bytes
```

Implementation Difficulties — non-linearities



Implementation Difficulties — signal-to-noise

Largest packet that can make it through the network without being fragmented is (usually) 1500 bytes. So the *maximum* change we can cause by varying packet size is:

Ethernet	(10 Mb/s)	1.2 ms
T3	(45 Mb/s)	267 us
FDDI	(100 Mb/s)	120 us
OC-3	(150 Mb/s)	80 us
OC-12	(622 Mb/s)	19 us

Typical round-trip prop times are tens or hundreds of ms so fit involves extracting O(100us) variation from measurements 1000 times larger.

Queue fluctuations are generally on the order of the prop time so “noise” is at least 1000 times larger than “signal”.

Implementation Difficulties — noise amplification

The bandwidth and delay estimates at each hop involve a difference from the previous hop. There are at least three ways to calculate this:

1. Fit $t_i(s)$ to $M_{i,s} - M_{i-1,s}$ (where $M_{i,s}$ is a filtered measurement of a probe of size s sent to hop i).

2. Fit $t_i(s)$ to $M_{i,s} - T(i-1,s)$

3. Fit $T(i,s)$ to $M_{i,s}$ then subtract $T(i-1,s)$.

Differencing amplifies noise so (1) makes the SNR problem worse. (2) propagates errors from one hop into all future hops so path end results are meaningless. Summing damps noise so (3) improves the SNR problem and only propagates errors forward one hop.

Problems — Multiple Paths

```
% pathchar bells.cs.ucl.ac.uk
...
 7 h9-1.t36-0.New-York2.t3.ans.net (140.223.37.21)
 |     70 Mb/s,  672 us (79.9 ms),  +q 1.94 ms
 8 f0-0.c36-11.New-York2.t3.ans.net (140.223.36.222)
                        -> 204.151.184.14 (6024)
                        -> 204.151.184.38 (6798)
                        -> 204.151.184.10 (9814)
                        -> 204.151.184.18 (7361)
                        -> 204.151.184.26 (5986)
 |    1.0 Mb/s,  38.5 ms (169 ms),  +q 17.7 ms,  4% dropped
 9?Dante-UKERNA.t3.ans.net (204.151.184.22)
 |    ?? b/s,    -1.1 ms (164 ms)
10 atm-gw.ulcc.ja.net (193.63.94.83)
 |    4.5 Mb/s,  0.98 ms (169 ms),  +q 17.0 ms
11 ucl.lonman.ja.net (194.83.100.61)
...
```

Problems — Hidden Hops (e.g., bridges, ATM, V.42)

```
% pathchar mbone.nsi.nasa.gov
 0 localhost
 |   8.7 Mb/s,   298 us (1.98 ms)
 1 ir40gw.lbl.gov (131.243.1.1)
 |    30 Mb/s,   127 us (2.64 ms)
 2 er1gw.lbl.gov (131.243.128.11)
 |    91 Mb/s,   189 us (3.15 ms)
 3 lbl-lc1-1.es.net (198.128.16.11)
 |   8.5 Mb/s,   2.16 ms (8.86 ms)
 4 ames-lbl-atms.es.net (134.55.28.1)
 |    83 Mb/s,   -158 us (8.69 ms)
 5 mbone.nsi.nasa.gov (192.203.230.241)
```

Current status

- Tool working on FreeBSD 2.2, Linux 2.x, Solaris and SunOS (some other ports in progress).
- Feedback from a few courageous alpha testers being incorporated in preparation for first public beta release.
- Hope to get out (source & binary) beta release sometime in next two weeks.

Future Work

- Get the release out.
- Automatically adapt number of probes based on “quality” of fit.
- De-convolve queue and loss-rate distributions.
- Better filtering and fit.
- Deal with “hidden” hops.