Using Hadoop to Explore Internet Route Stability

Tomas Isdal
Ethan Katz-Bassett
University of Washington

Hadoop Independent Study, June 2008

Motivation

- Systems depend on knowing route performance from servers to “entire” Internet
  - iPlane, Hubble, Google
- Want up-to-date measurements, yet:
  - Want to conserve traceroutes
  - Can’t make all you want, want to be friendly
- Knowing likelihood of change could drive probing decisions
  - How often do we need to probe?
  - Focus probes on paths likely to change

Goal

To answer:

- How stable are routes on the Internet?
- For now: Prevalence, not persistence
- As many paths as possible

Related Work

- Paxson, ToN 1997
  - 37 sites, mostly academic
  - Pairwise traceroutes for 1.5 months in 1995
  - Paths heavily dominated by single prevalent route
    - 70% of (src,dst) had same router-level path >60% of time
- Zhang, tech report 2000
  - 31 NIMI hosts (25 in US, 1/2 edu, rest mostly research) plus 189 traceroute servers
  - Pairwise for Dec 99-Jan 00 (but tons of missing data)
  - Paths heavily dominated by single prevalent route
    - 85% of (src,dst) had same router-level path >90% of time

Motivation, Part 2

Do results from earlier studies hold up?

- Has the Internet changed?
- Do the results hold over longer timescales?
- Were their datasets representative?
  - Limited size
  - Heavy academic/research bias ⇒ heavy GREN backbone bias/ not representative of commercial Internet

Our Dataset

- iPlane, Harsha's dissertation work
- Daily traceroutes from ~200 PlanetLab sites to ~100,000 prefixes
  - 4.5 GB per day
- 1.5+ years of data
  - 3 TB uncompressed
  - 12 billion traceroutes
- Motivation 3: learn to use Hadoop as a tool for iPlane analysis

Hadoopifying the data

- Data stored in ~20-30 MB files (~1/site/day)
  - Binary format
  - Total size > 3TB
  - Spread out on 3 file servers
- Idea: merge to 1 day chunks and gzip
  - Copy | merge_convert | gzip | hadoop.cs | dfs
  - ~700 days, 600-700 MB/day after gzip
- Problem: 30-40 cpu minutes for 1 day of data
    - 700 days -> weeks just to get data into dfs

Hadoopifying the data

- Solution: Write a parallel distributed application

  (Wasn't it to avoid this we decided to use hadoop in the first place?)

  - Networks cluster, 80*2Ghz CPUs on 10 hosts
    - Implement controller to manages jobs
      - Max 2 concurrent copy operation per file server
      - Max 1 worker per cpu
    - Max out file servers at ~40 workers
    - Average time now ~1 min for 1 day of traceroutes
- Problem: Failures…
  - Fortunately copy to DFS is transactional

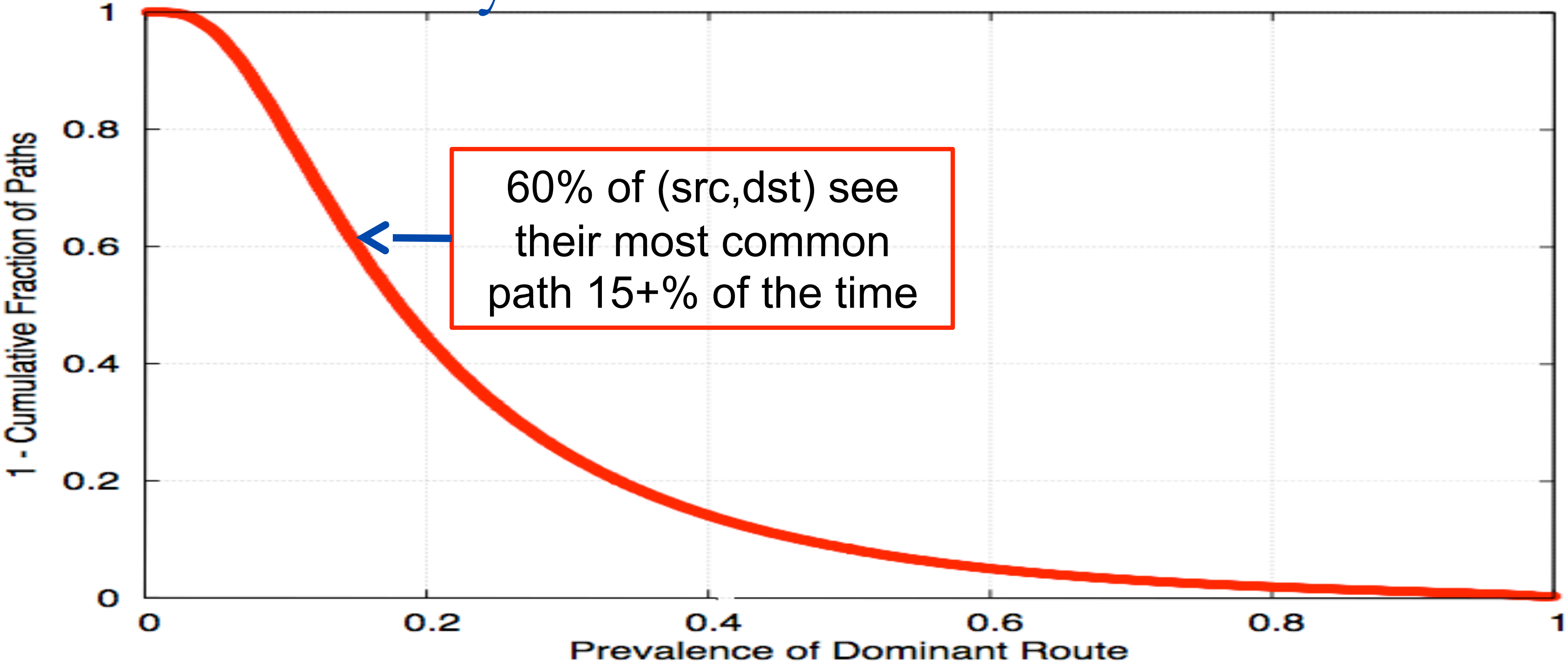
Cleaning the Data

- Exact src, dst varies by day
- Target set updated partway through
- Traceroutes that don’t reach
- Loops
- Missing, duplicated hops
- Aliases
- Load-balancing

Map/Reduce

- Input file: 1 day’s traceroutes, gzipped txt, one traceroute per line, ~700MB
- Map:
  - Input: 1 traceroute
  - Preprocess and clean input:
    - Discard if bad
    - Standardize src, dst, route
  - Output: (<src, dst>, Hash(route))
- Reduce:
  - Input: (<src, dst>, List of Hash(route))
  - Output: (<src, dst>, List of <Hash(route),cnt>)

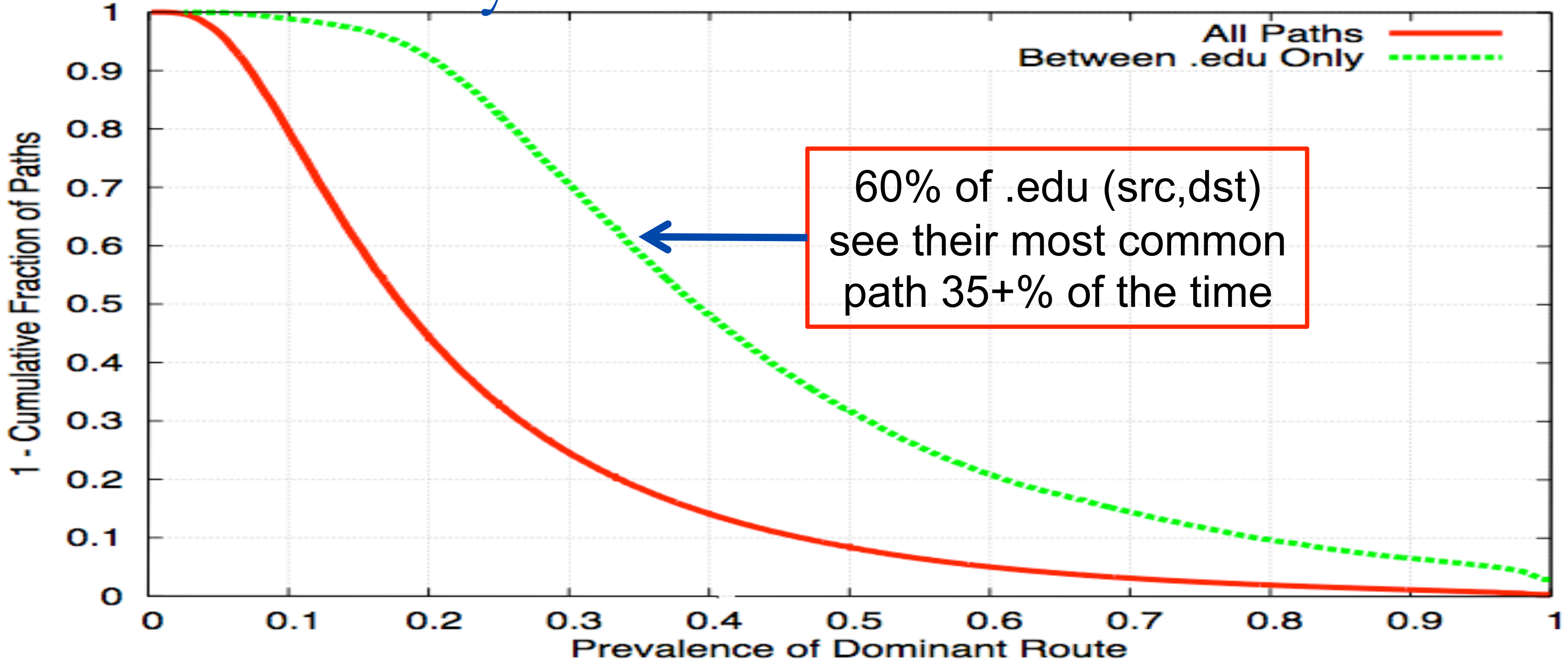
Preliminary Results



60% of (src,dst) see their most common path 15+% of the time

- <src IP, dst IP> ⇒ IP-level path
- Consider only pairs with 50+ measurements
- Unlike previous work, no dominant paths

Preliminary Results



60% of .edu (src,dst) see their most common path 35+% of the time

Why the discrepancy with previous work?

- Duration of study? Internet changed? **Dataset biases?**
- GREN backbone not representative

What We Learned and What's Left

- Hadoop makes this type of analysis easy
- Importing data into DFS is not trivial
- Datasets bias results
  - PL-PL measurements not representative
  - PL-world?

Future:

- Persistence
- PoP, AS-level paths
- Analysis of failed traceroutes
- Can we classify which are stable?