Timely, Reliable, and Cost-Effective Internet Transport Service using Dissemination Graphs

Amy Babay[1], Emily Wagner[1,2],
Michael Dinitz[1], and Yair Amir[1,2]

[1]Johns Hopkins University, [2]LTN Global Communications

JOHNS HOPKINS UNIVERSITY

Distributed Systems and Networks Lab
www.dsn.jhu.edu

Problem: Combining Timeliness and Reliability over the Internet

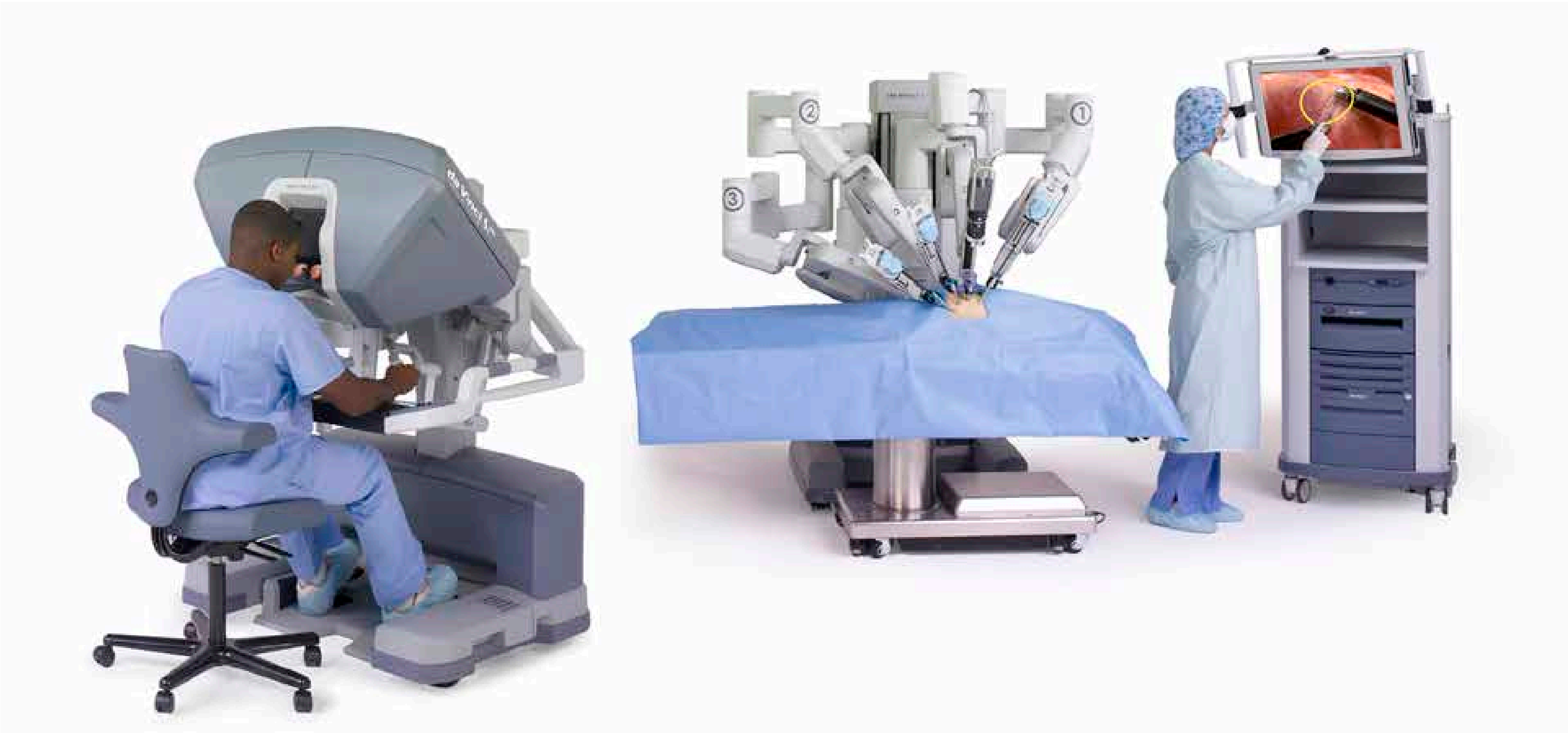
- Internet natively supports end-to-end reliable (e.g. TCP) or best-effort timely (e.g. UDP) communication
- Our goal: support applications with extremely demanding combinations of timeliness and reliability requirements in a cost-effective manner
- Applications have emerged over the past few years that require both timeliness guarantees and high reliability
  - e.g. VoIP, broadcast-quality live TV transport

State-of-the-art: Combining Timeliness and Reliability over the Internet

200ms one-way latency requirement, 99.999% reliability guarantee

40ms one-way propagation delay across North America

New Challenges: Combining Timeliness and Reliability



130ms **round-trip** latency requirement

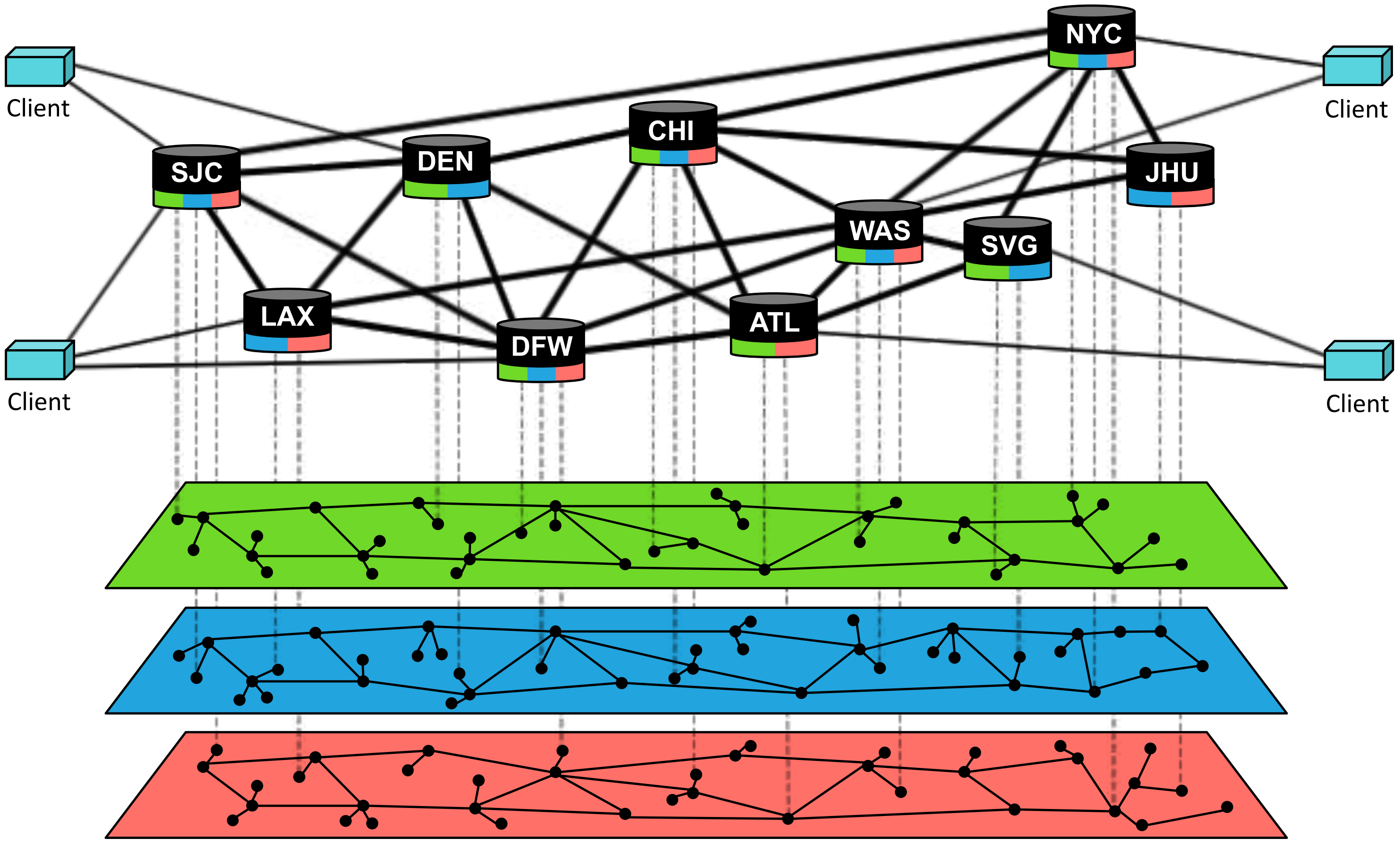
New Challenges: Combining Timeliness and Reliability

130ms **round-trip** latency requirement

80ms round-trip propagation delay across North America

State-of-the-art: Combining Timeliness and Reliability over the Internet

- Structured overlay networks enable specialized routing and recovery protocols



Related Work

- Overlay Routing
  - Detour [SAAB+99] IEEE Micro 1999
  - RON [ABKM01] SOSP 2001
  - One-hop source routing [GMGLW04] OSDI 2004
  - Spines (effective latency) [ADGHT06] Trans. Multimedia 2006
- Overlay Recovery
  - Hop-by-hop reliability [AD03] DSN 2003
  - OverQos [SSBK04] NSDI 2004
- Redundant Dissemination
  - Disjoint Paths
    - [SCG01] SOSP 2001, [PHS02] MobiHoc 2002, [ASB03] IMC 2003, [NZ03] INFOCOMM 2003, [OTBS+16] ICDCS 2016
  - Potentially overlapping paths
    - [KPKYL10] On the Move to Meaningful Internet Systems 2010

Addressing New Challenges: Dissemination Graph Approach

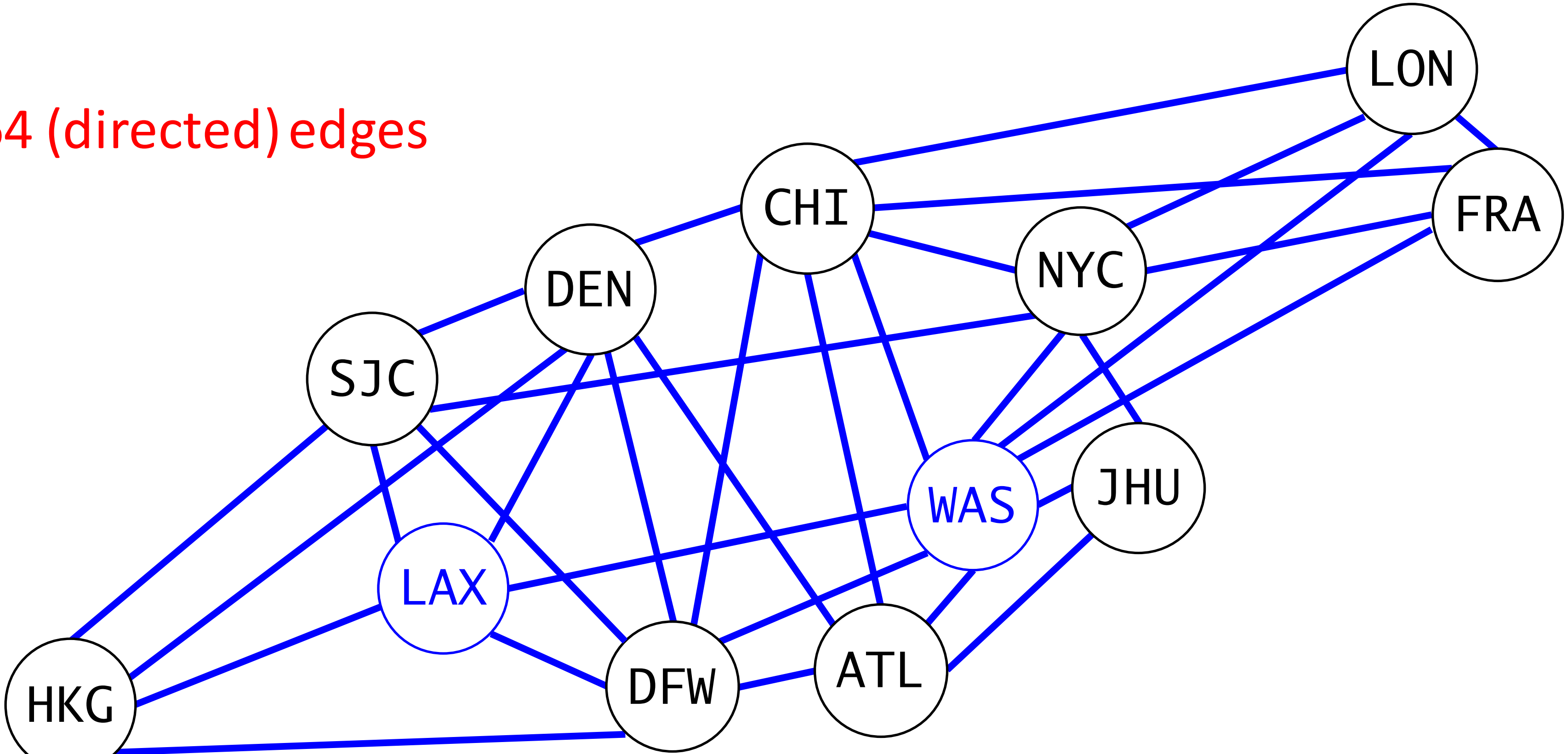
- Stringent latency requirements give less flexibility for buffering and recovery
- Core idea: Send packets redundantly over a subgraph of the network (a dissemination graph) to maximize the probability that at least one copy arrives on time

How do we select the subgraph (subset of overlay links) on which to send each packet?

Initial Approaches to Selecting a Dissemination Graph

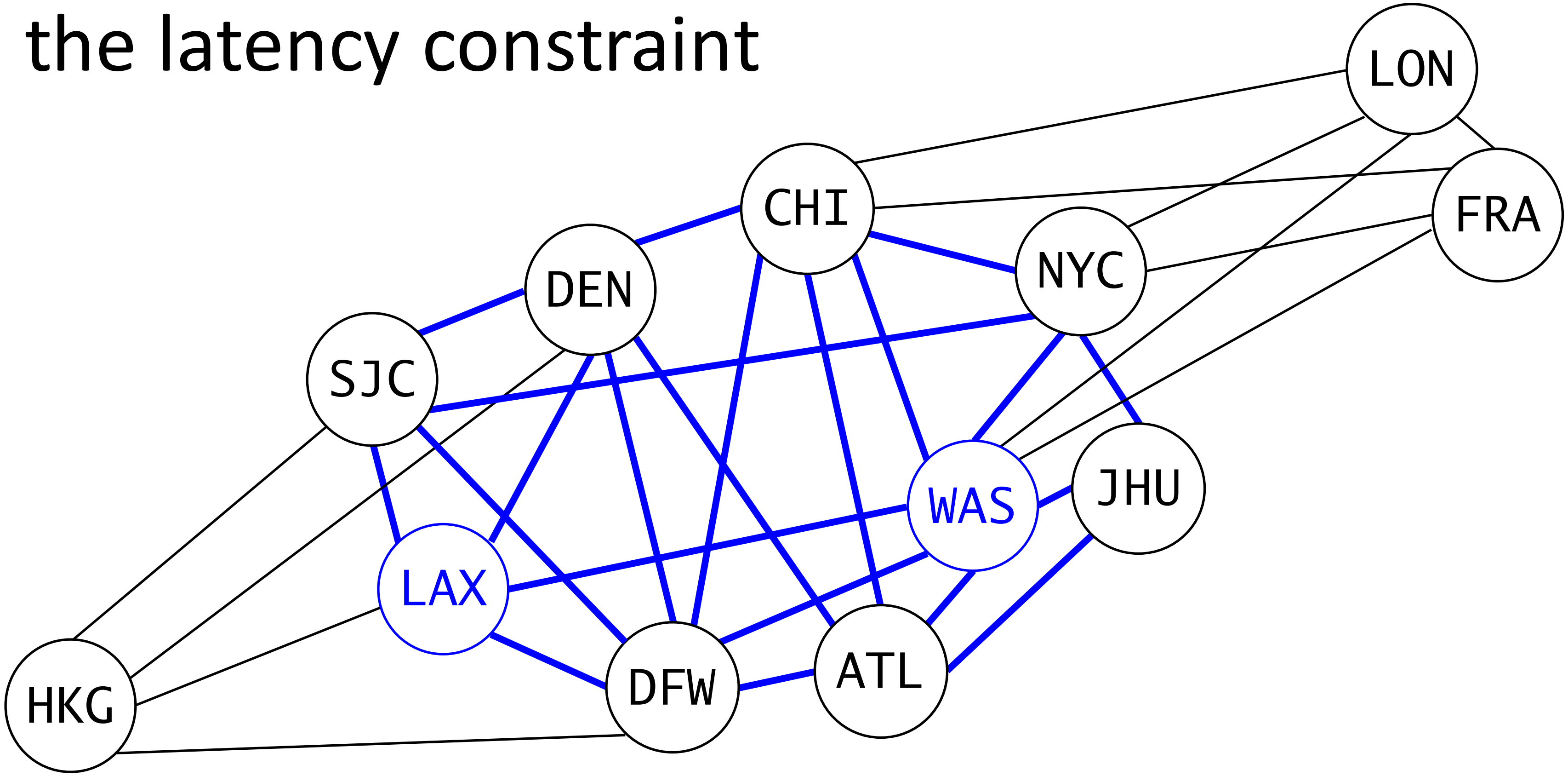
- Overlay Flooding: send on all overlay links
  - Optimal in timeliness and reliability but expensive

64 (directed) edges



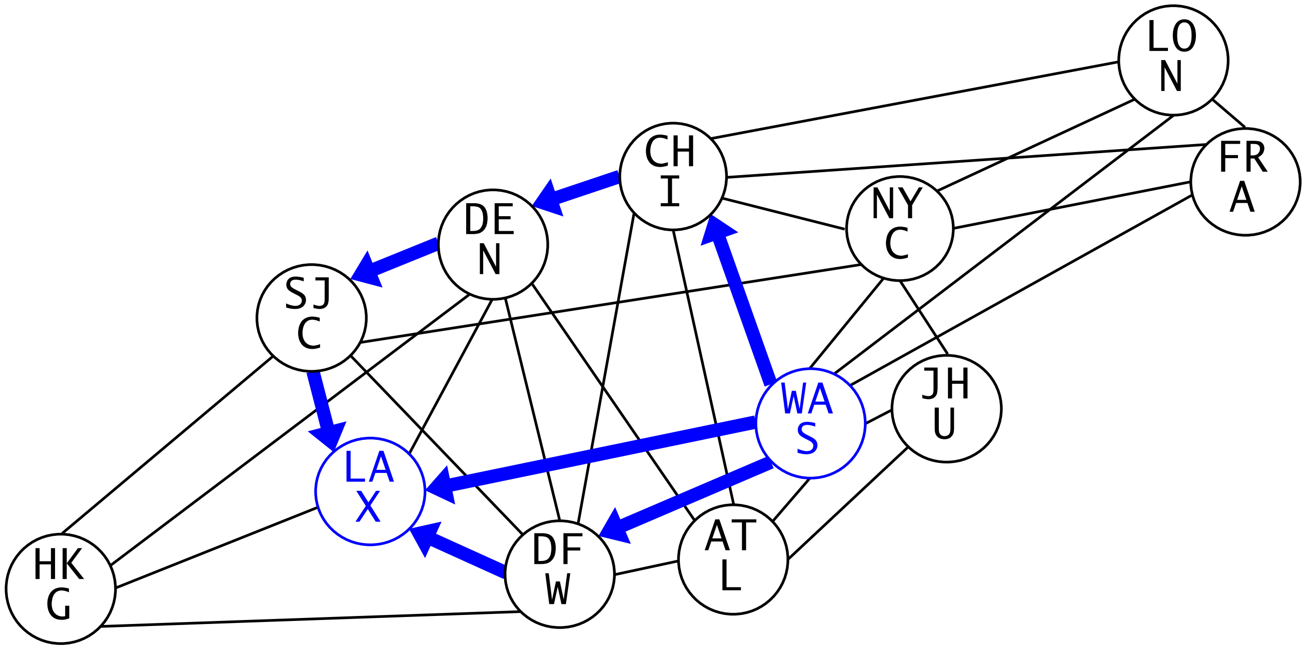
Initial Approaches to Selecting a Dissemination Graph

- Time-Constrained Flooding: flood only on edges that can reach the destination within the latency constraint



Initial Approaches to Selecting a Dissemination Graph

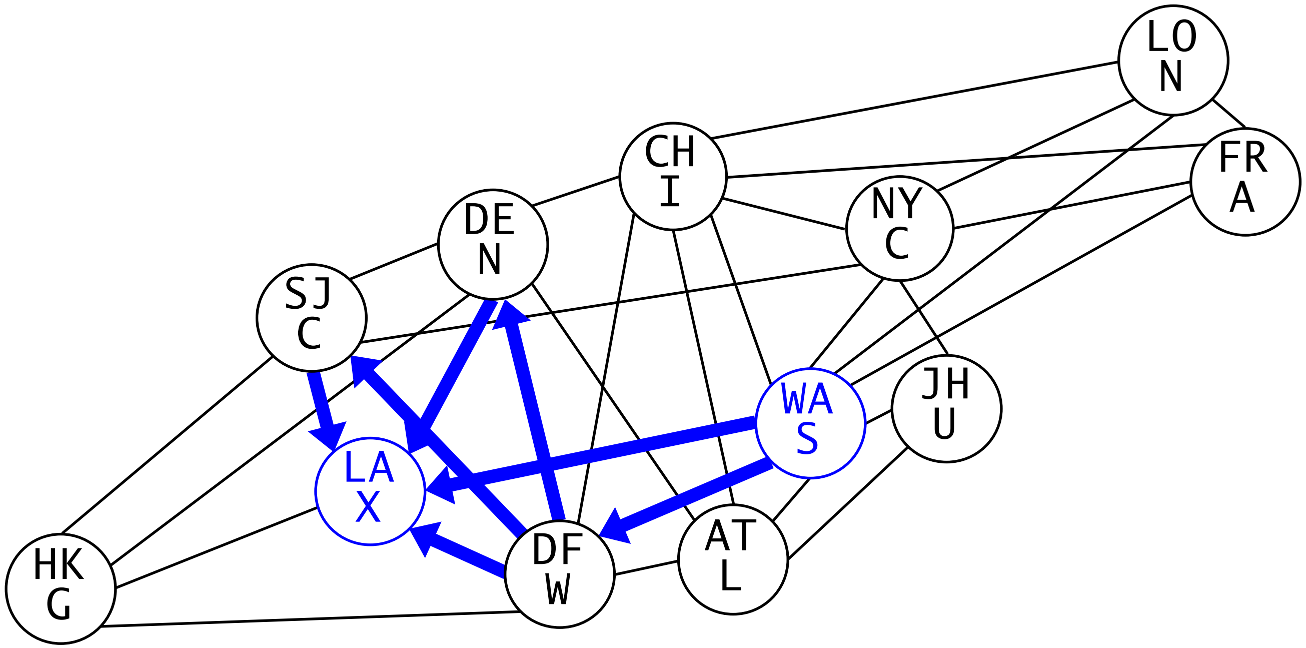
- Disjoint Paths: send on several paths that do not share any nodes (or edges)
  - Good trade-off between cost and timeliness/reliability
  - Uniformly invests resources across the network



Selecting an Optimal Dissemination Graph

Can we use knowledge of the network characteristics to do better?

Invest more resources in more problematic regions:



Data-Informed Dissemination Graphs

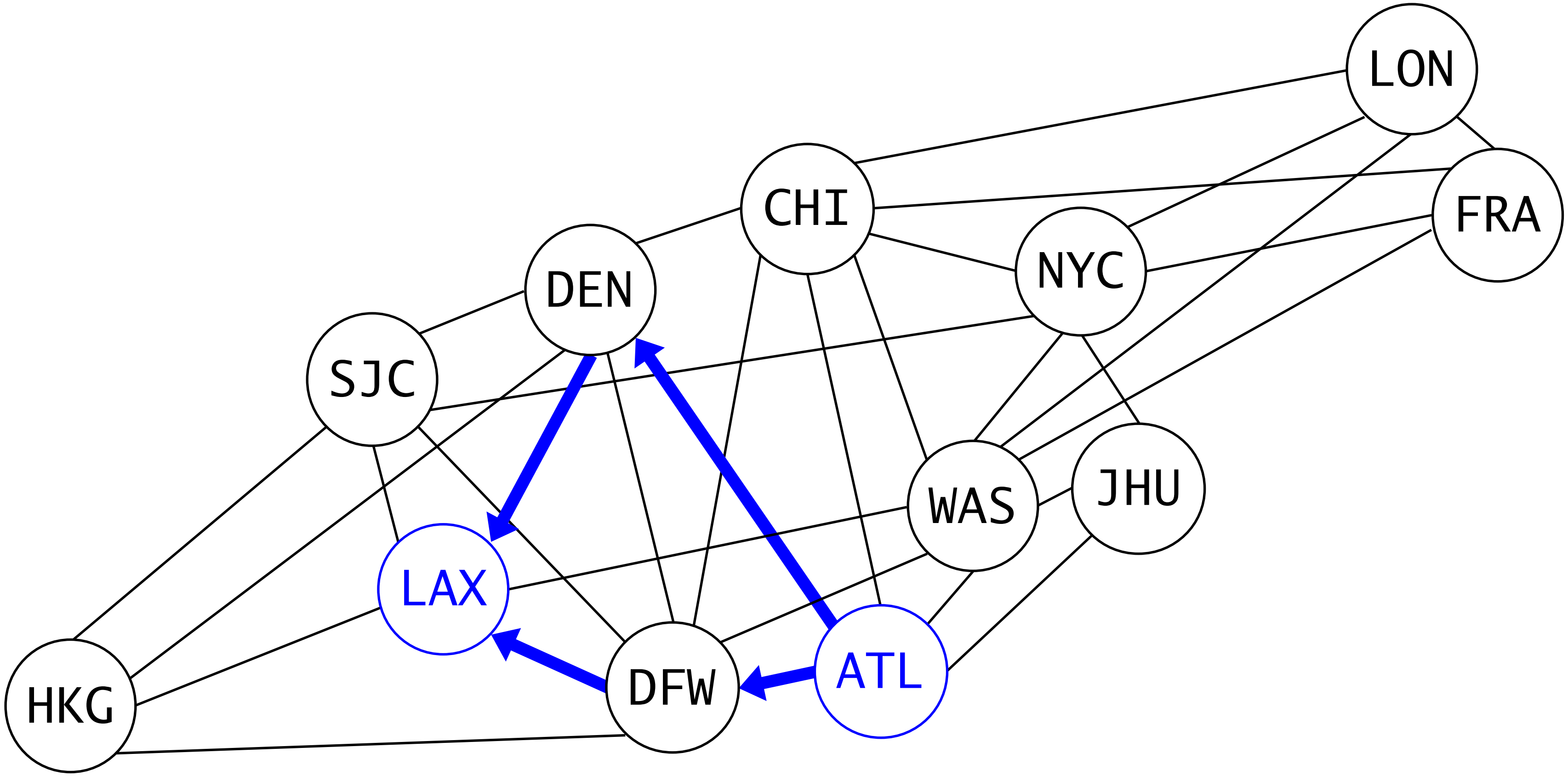
- Goal: Learn about the types of problems that occur in the field and tailor dissemination graphs to address common problem types
- Collected data on a commercial overlay topology (www.ltnglobal.com) over 4 months
- Key findings:
  - Two disjoint paths provide relatively high reliability overall
    - Good building block for most cases
  - Almost all problems not addressed by two disjoint paths involve either:
    - A problem at the source
    - A problem at the destination
    - Problems at both the source and the destination

Dissemination Graphs with Targeted Redundancy

- Our approach:
  - Pre-compute four graphs per flow:
    - Two disjoint paths (static)
    - Source-problem graph
    - Destination-problem graph
    - Robust source-destination problem graph
  - Use two disjoint paths graph in the normal case
  - If a problem is detected at the source and/or destination of a flow, switch to the appropriate pre-computed dissemination graph
  - Converts optimization problem to classification problem

Dissemination Graphs with Targeted Redundancy: Case Study

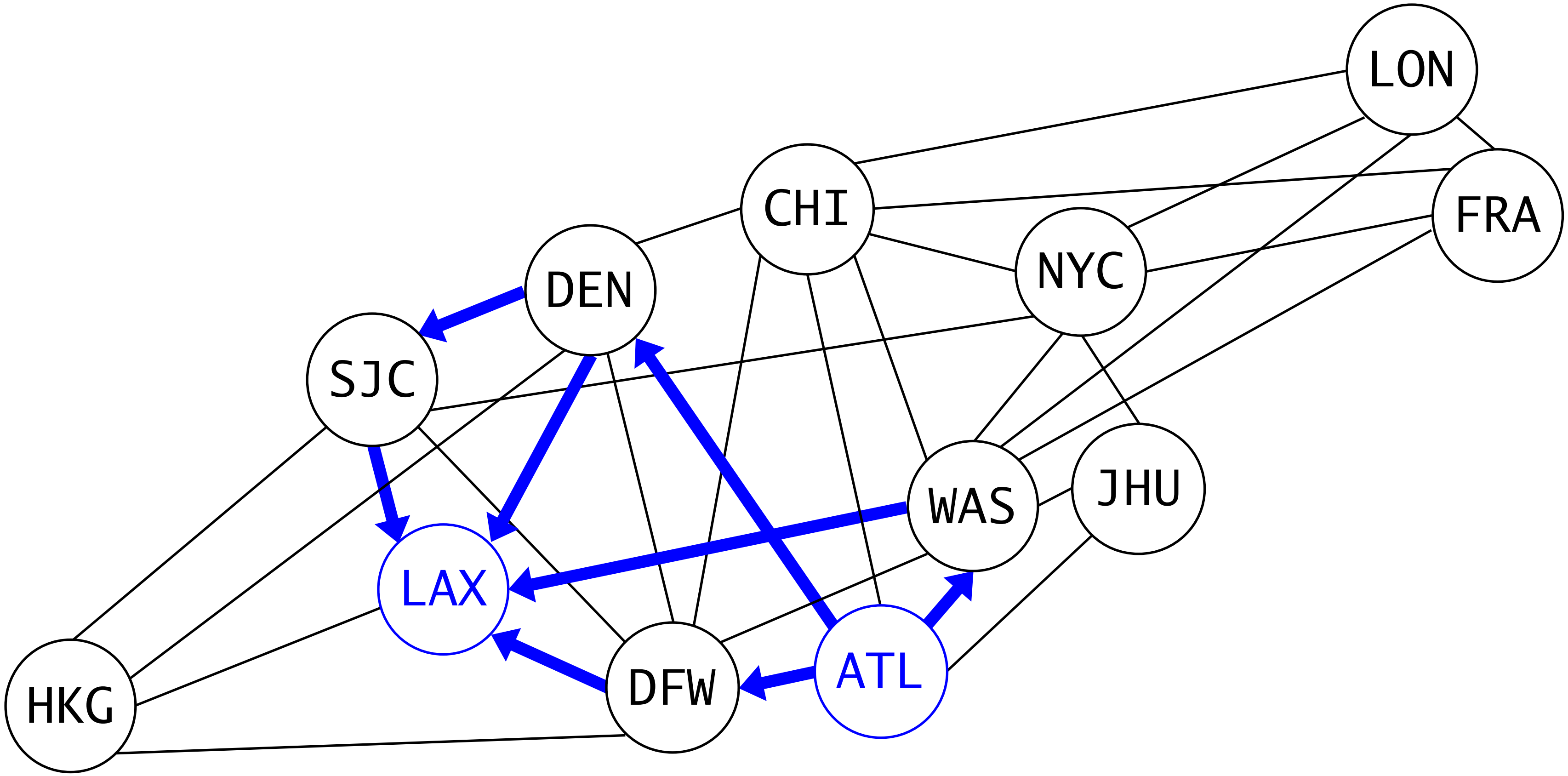
- Case study: Atlanta -> Los Angeles



Two node-disjoint paths dissemination graph (4 edges)

Dissemination Graphs with Targeted Redundancy: Case Study

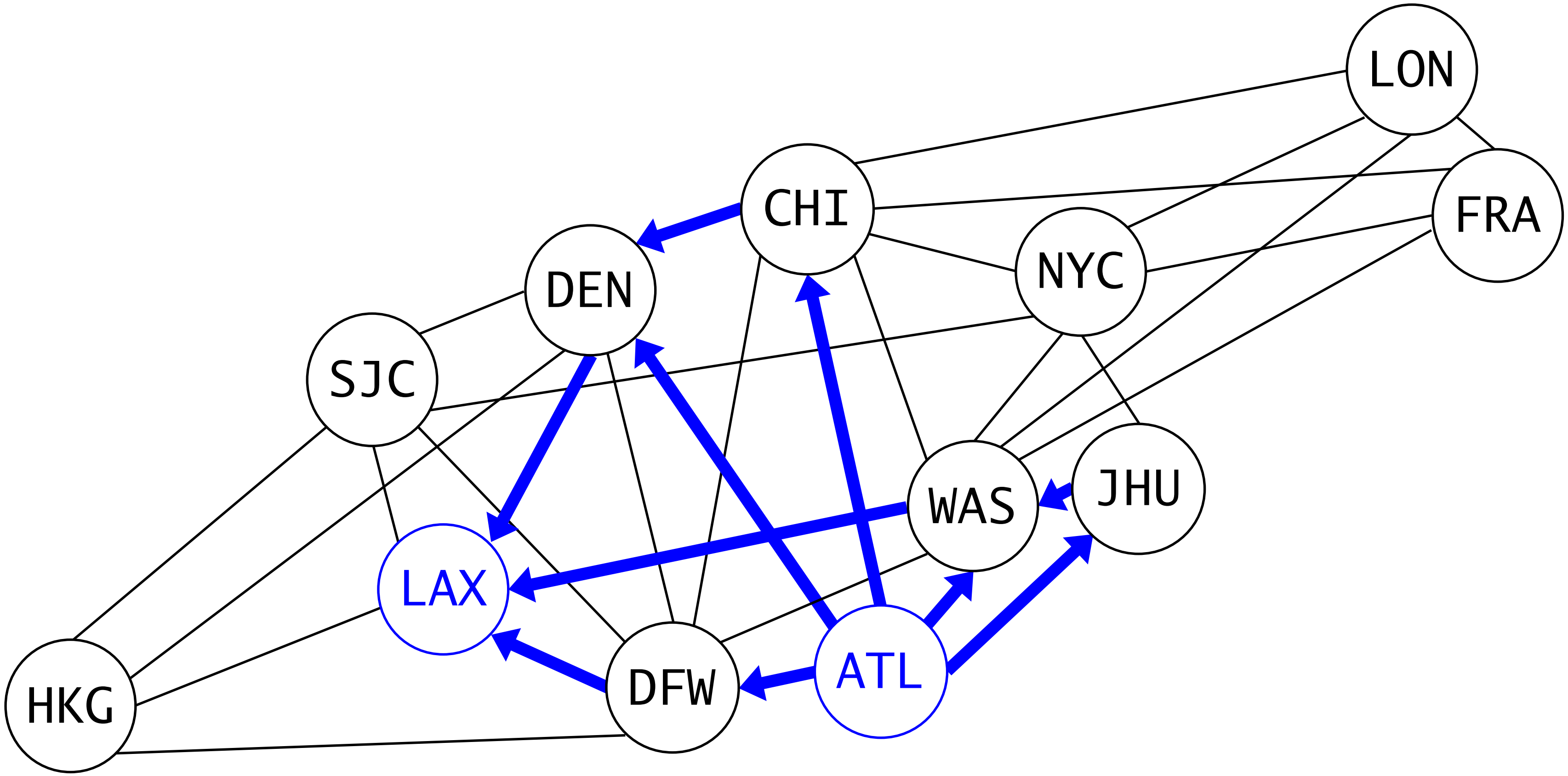
- Case study: Atlanta -> Los Angeles



Destination-problem dissemination graph (8 edges)

Dissemination Graphs with Targeted Redundancy: Case Study

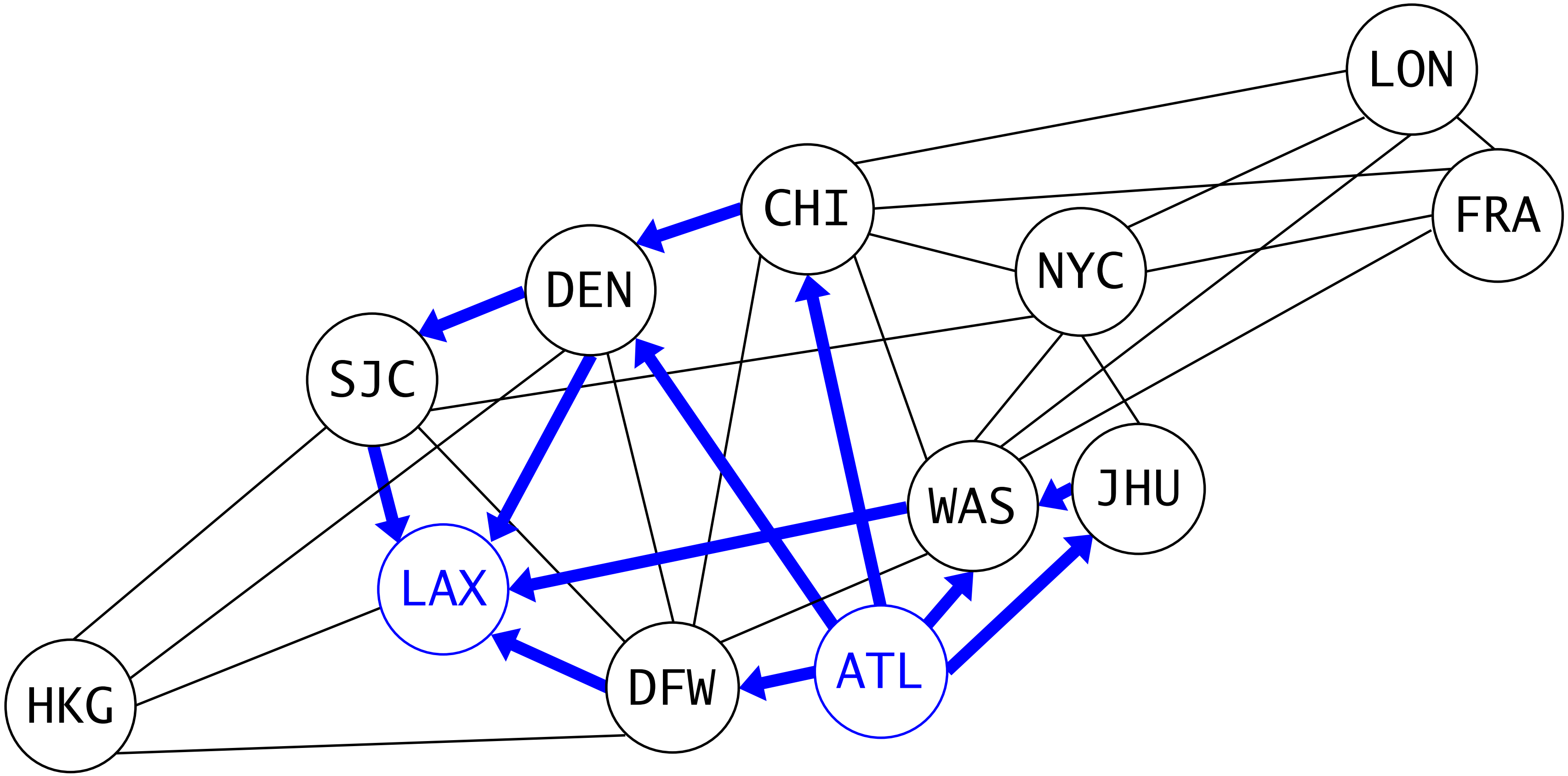
- Case study: Atlanta -> Los Angeles



Source-problem dissemination graph (10 edges)

Dissemination Graphs with Targeted Redundancy: Case Study

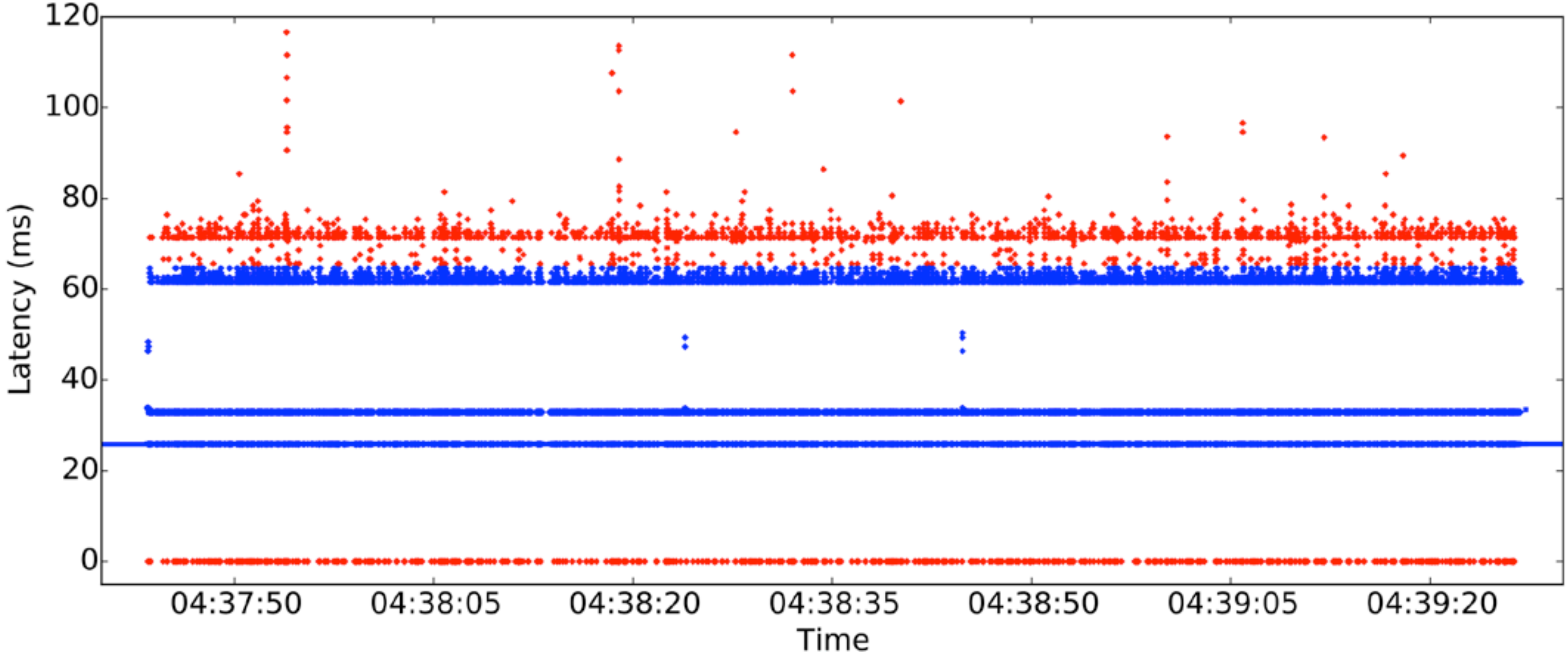
- Case study: Atlanta -> Los Angeles



Robust source-destination-problem dissemination graph (12 edges)

Dissemination Graphs with Targeted Redundancy: Case Study

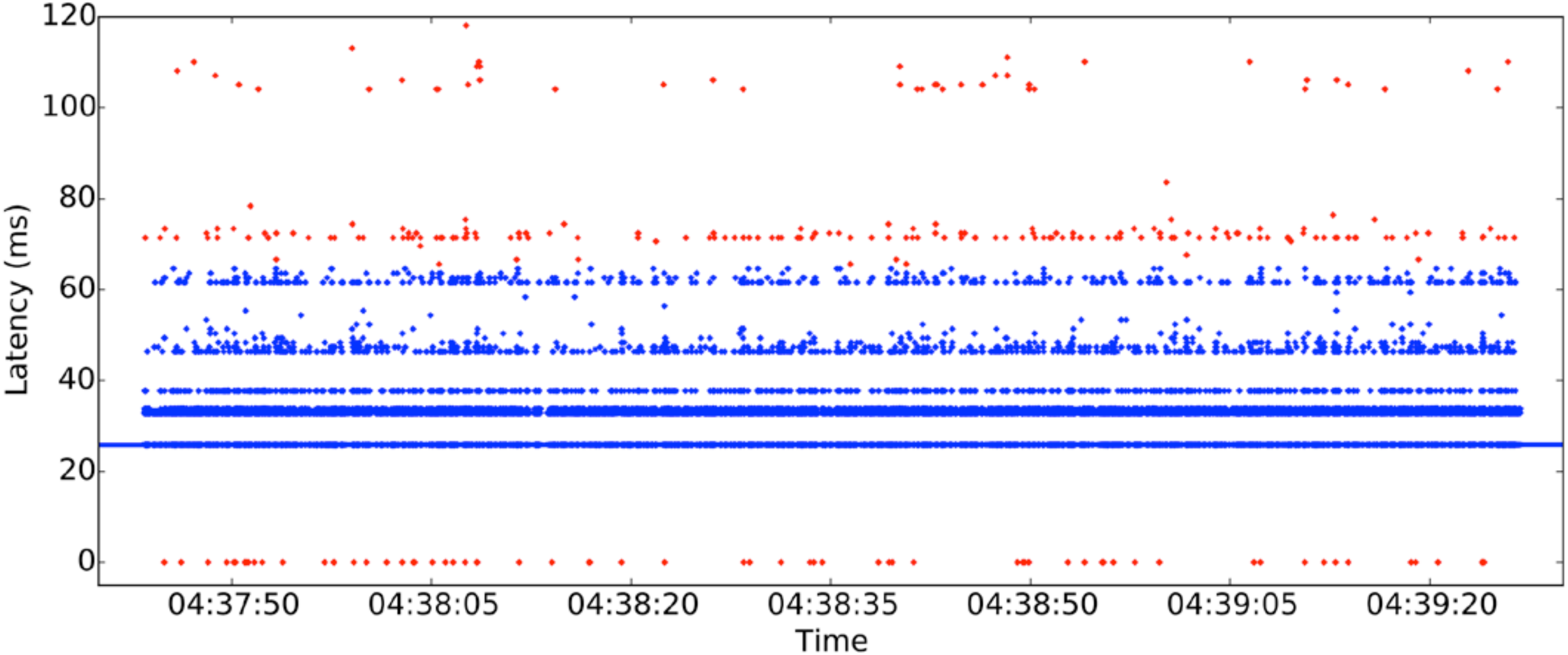
- Case study: Atlanta -> Los Angeles; August 15, 2016



Packets received and dropped over a 110-second interval using (adaptive) two disjoint paths (3982 lost/late packets, 20 packets with latency over 120ms not shown)

Dissemination Graphs with Targeted Redundancy: Case Study

- Case study: Atlanta -> Los Angeles; August 15, 2016

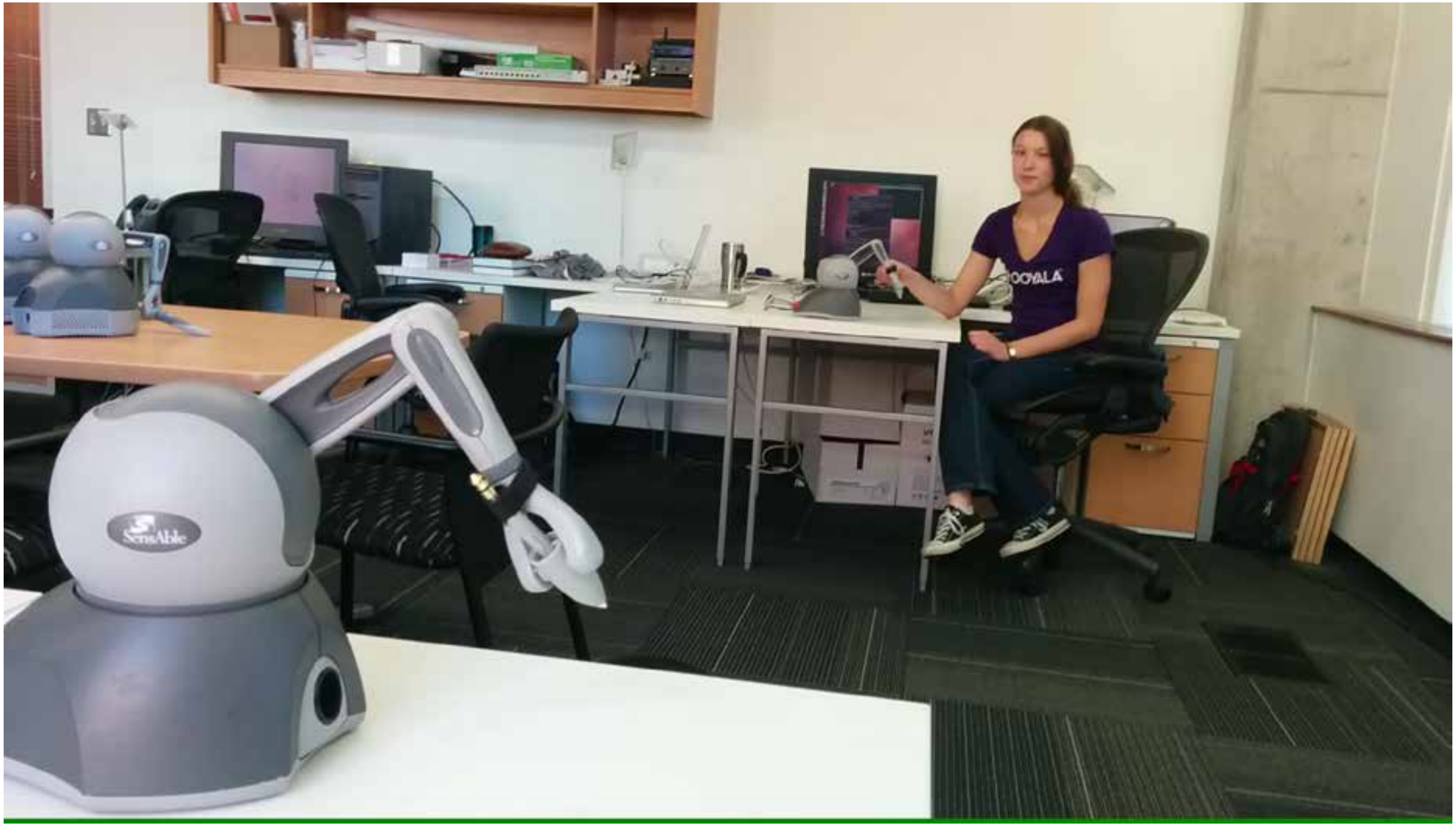


Packets received and dropped over a 110-second interval using our dissemination-graph-based approach to add targeted redundancy at the destination (299 lost/late packets)

Dissemination Graphs with Targeted Redundancy: Results

- 4 weeks of data collected over 4 months
  - Packets sent on each link in the overlay topology every 10ms
- Analyzed 16 transcontinental flows
  - All combinations of 4 cities on the East Coast of the US (NYC, JHU, WAS, ATL) and 2 cities on the West Coast of the US (SJC, LAX)
  - 1 packet/ms simulated sending rate
- Captures over 99% of the benefit of (optimally reliable) time-constrained flooding
- Costs slightly more than two disjoint paths (about twice the cost of the single best path)

Applications: Remote Manipulation



Applications: Remote Robotic Ultrasound

- Collaboration with JHU/TUM CAMP lab (https://camp.lcsr.jhu.edu/)