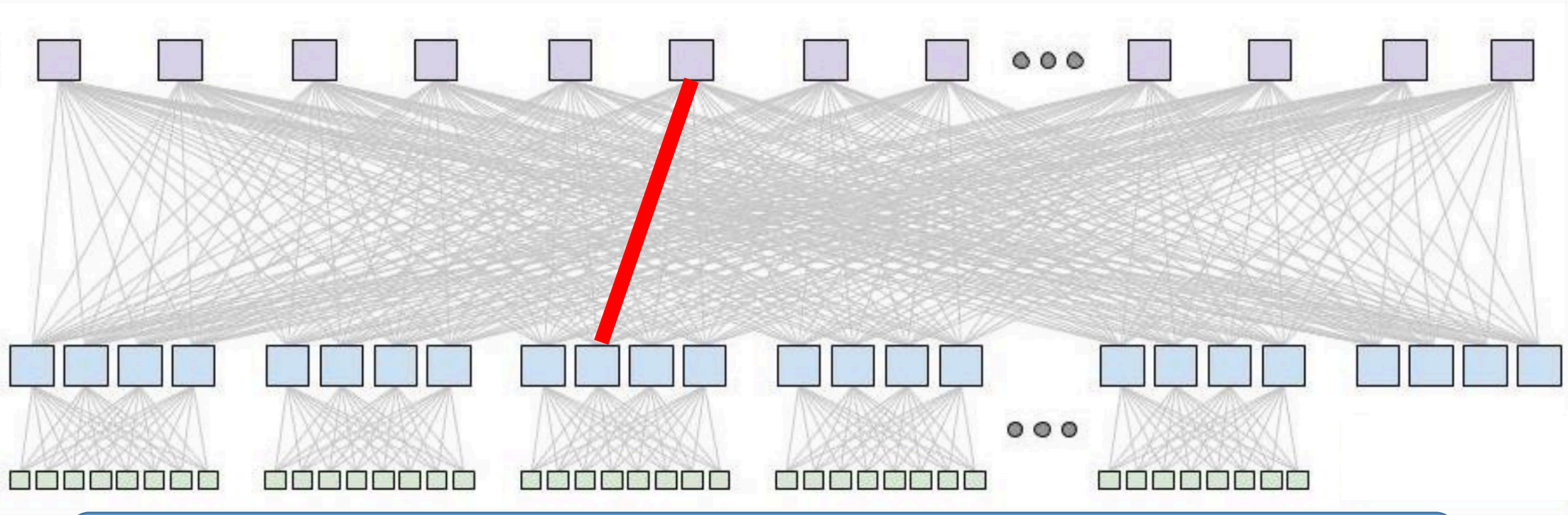
Passive realtime datacenter fault detection and localization

Arjun Roy, James Hongyi Zeng*, Jasmeet Bagga*, and Alex C. Snoeren
University of California, San Diego Facebook*

“It would be nice if we could figure out which link was causing these retransmits.”

- **Ranjeeth Dasineni**, Facebook (paraphrased)

Contemporary datacenter network



However: faults may be partial/intermittent.

Partial faults: A few examples

- **Netpilot (Sigcomm 2011):** Frame check error, unequal ECMP hashing, etc.

Wu, Xin, et al. "Netpilot: automating datacenter network failure mitigation." *ACM SIGCOMM Computer Communication Review* 42.4 (2012): 419-430.

- **Everflow (Sigcomm 2015):** TCAM bit errors, silent packet drops.

Zhu, Yibo, et al. "Packet-Level Telemetry in Large Datacenter Networks." *SIGCOMM*, 2015.

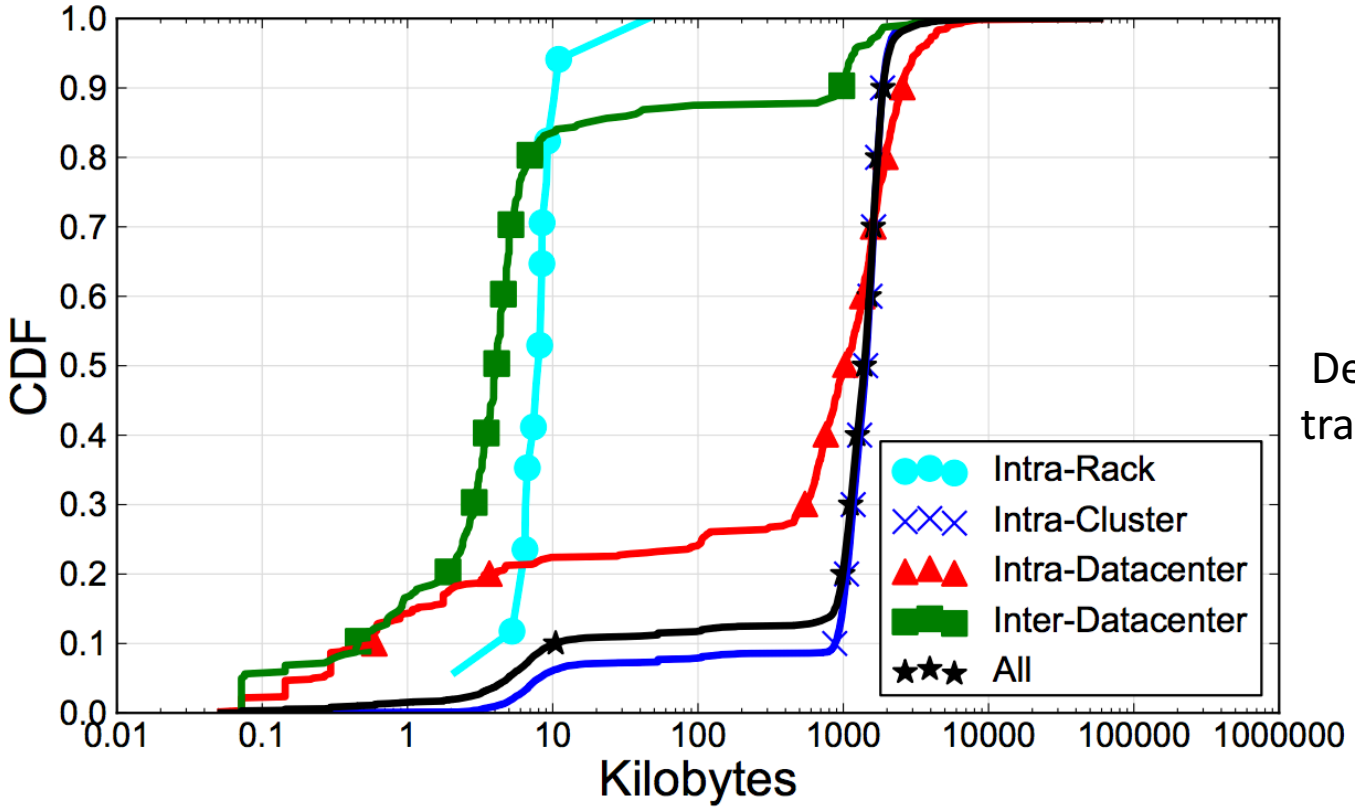
- **Pingmesh (Sigcomm 2015):** "fiber FCS…errors, switching ASIC defects, switch fabric flaw, switch software bug, NIC configuration issue, network congestions, etc. We have seen all these types of issues in our production networks."

Guo, Chuanxiong, et al. "Pingmesh: A Large-Scale System for Data Center Network Latency Measurement and Analysis." SIGCOMM, 2015.

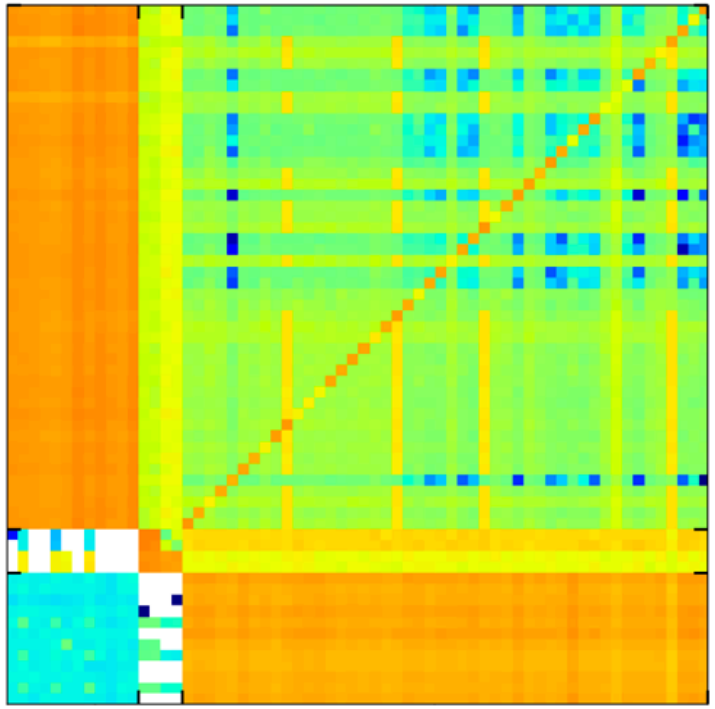
Vast body of prior work (just a small sample…)

- **Application instrumentation:** various production systems
- **Active probing:** *Pingmesh (SIGCOMM'15), NetNorad (Facebook), ATPG (CoNEXT '12), Everflow (SIGCOMM'15)*
- **Machine learning:** *NetPoirot (SIGCOMM'16)*
- **Graph algorithms:** *Gestalt (Usenix ATC '14), SCORE (NSDI '05)*
- **Path tracing:** *Everflow (SIGCOMM'15), NetNorad (Facebook), NetSight (NSDI '14), Tiny Packet Programs (SIGCOMM'14)*
- **Network instrumentation:** *FlowRadar (NSDI '16) , Planck (SIGCOMM'14), NetPilot (SIGCOMM'11)*

We exploit: highly regular load balanced traffic



Destination rack traffic magnitude



Source rack traffic magnitude

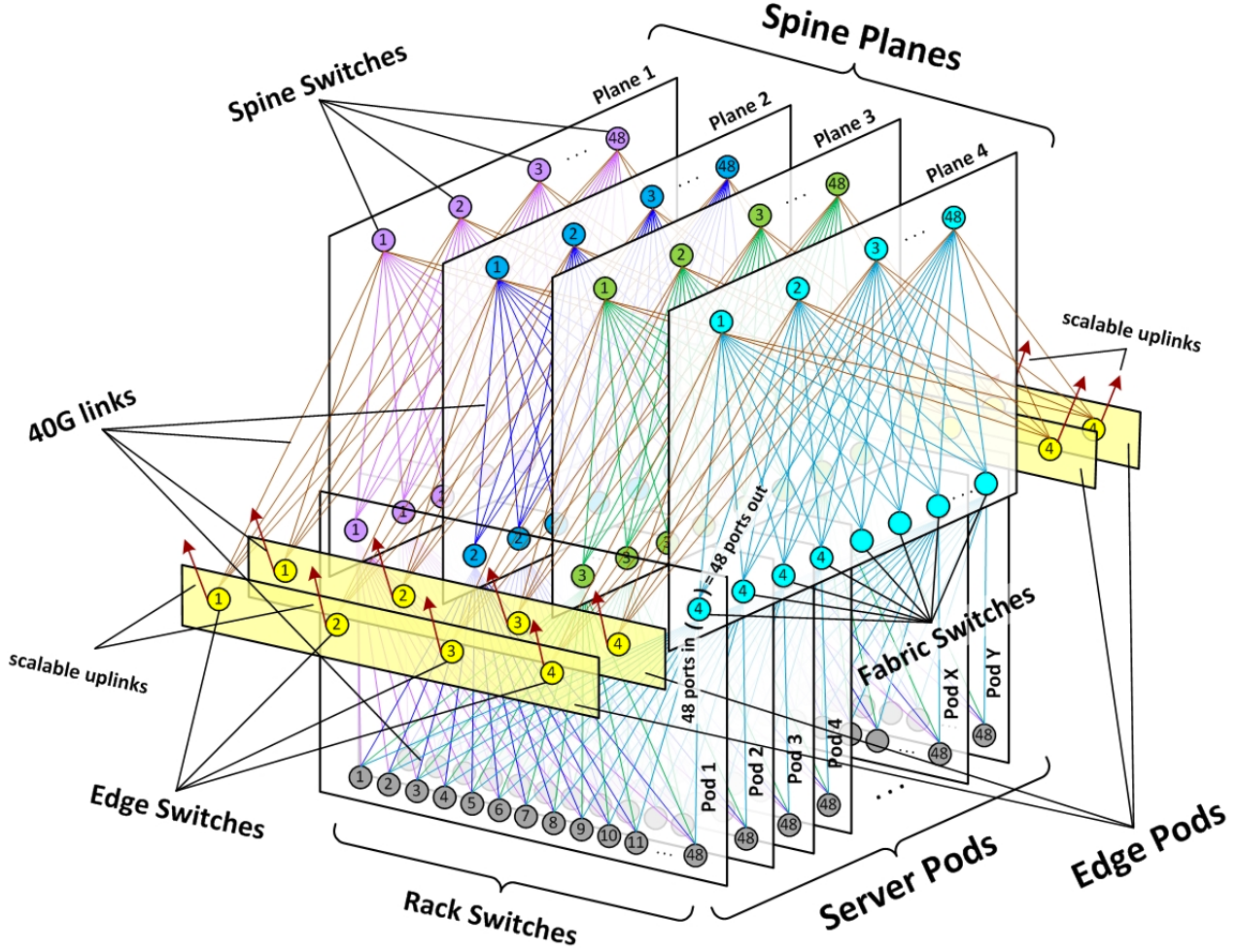
Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C. Snoeren.
Inside the Social Network's (Datacenter) Network. ACM SIGCOMM '15, London, England.

Load balanced traffic simplifies fault handling

- Evenly loaded paths means per path performance is similar ***if*** no errors.
- Network faults lead to outlier paths.
- If flow network path known, can correlate flow performance with path.

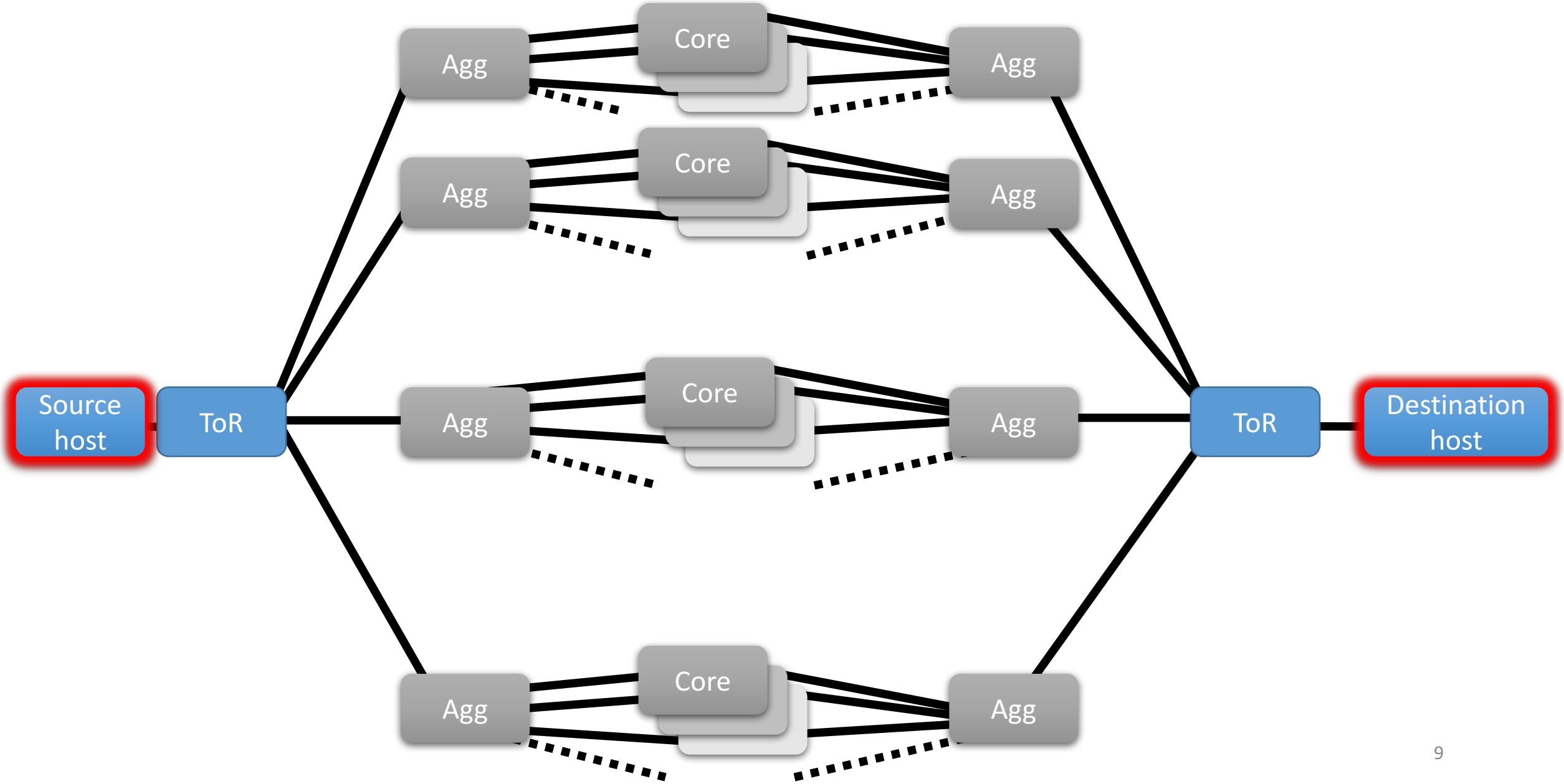
- **Approach allows us to find and localize faults:**
  - **In an application agnostic manner**
  - **Incurring no additional probing overhead**
  - **More rapidly than prior published works**

Facebook datacenter topology

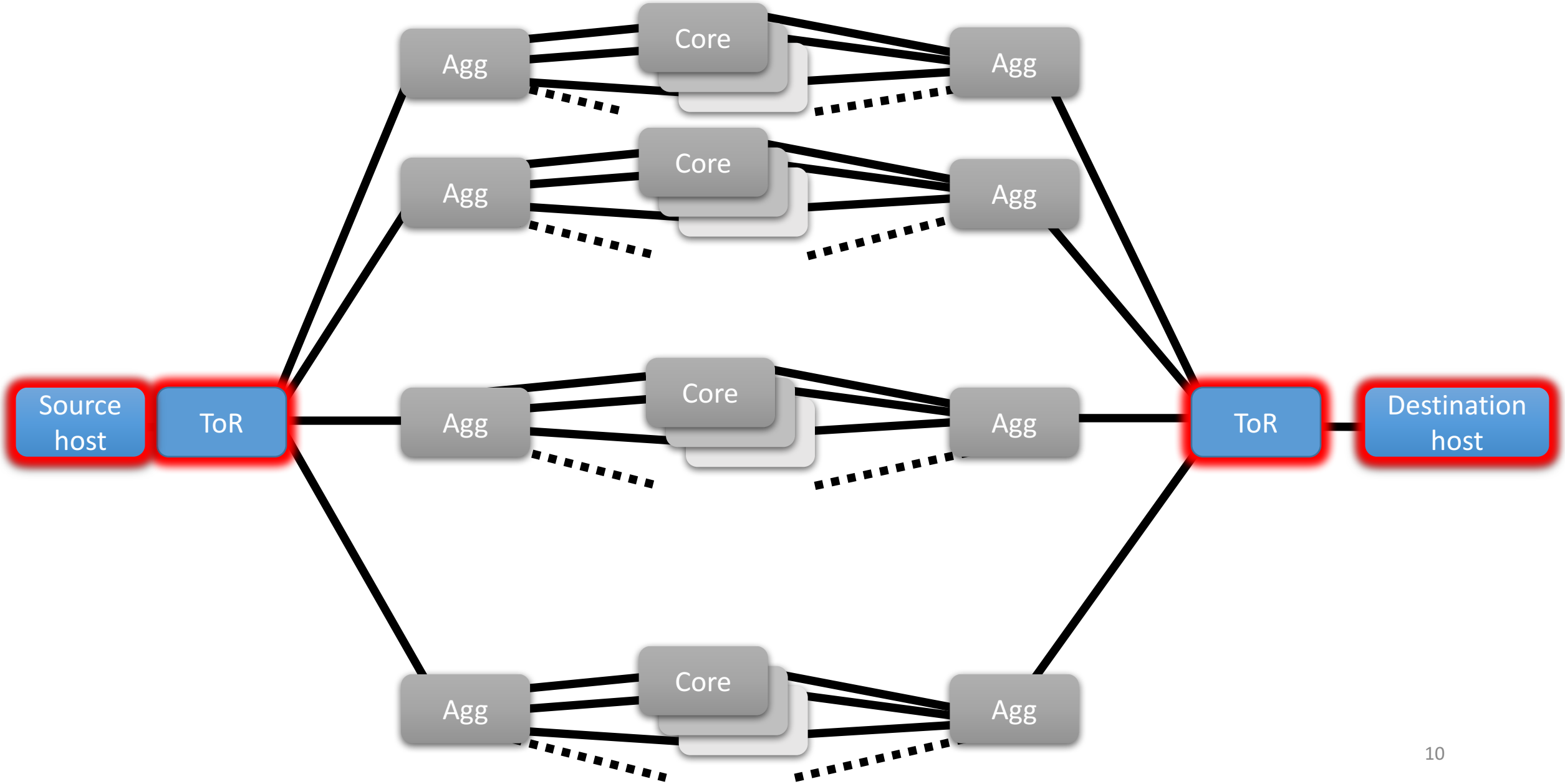


Alexey Andreyev.
Introducing data center fabric, the next-generation Facebook data center network.
https://code.facebook.com/posts/360346274145943/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/

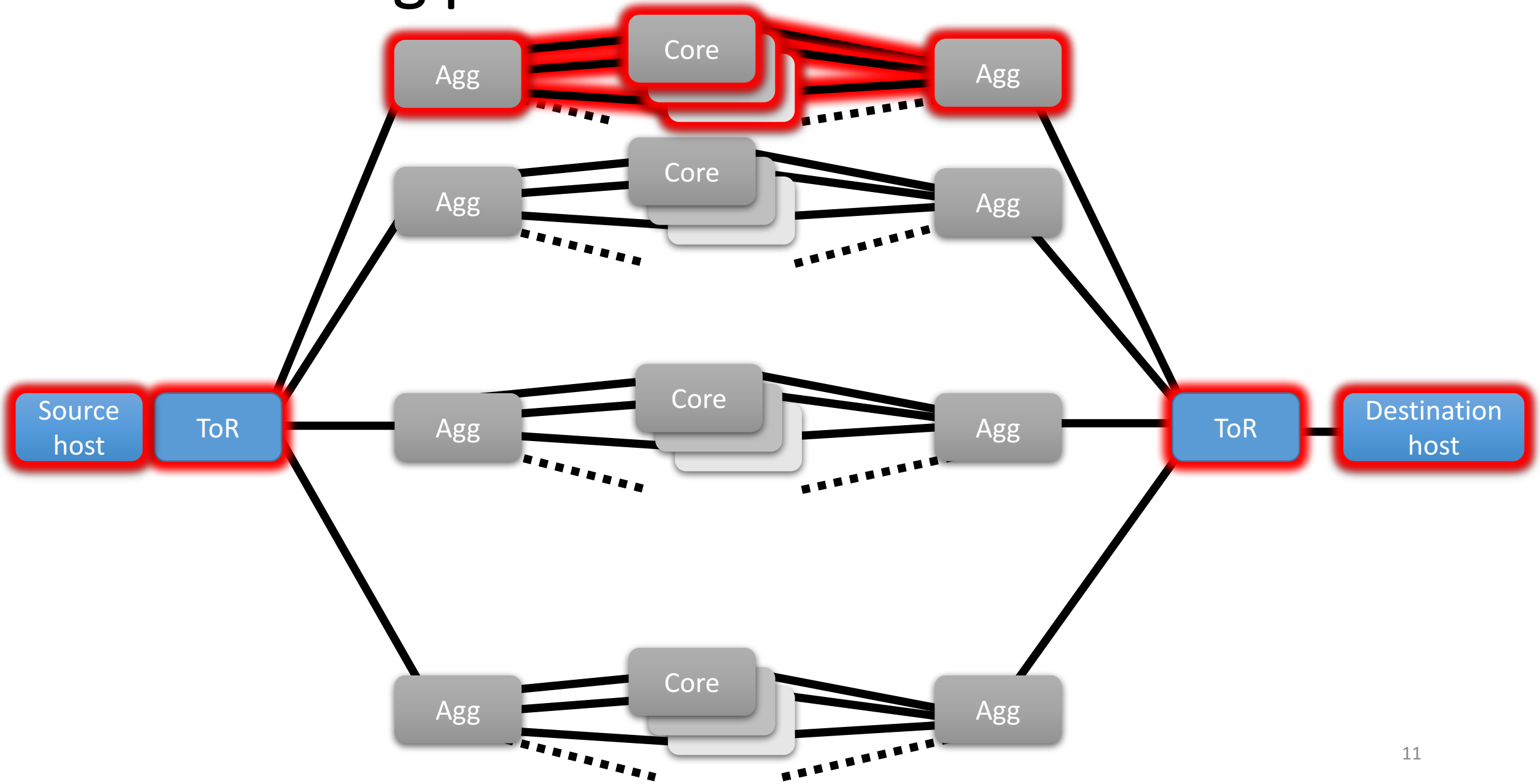
Finding path information at Facebook



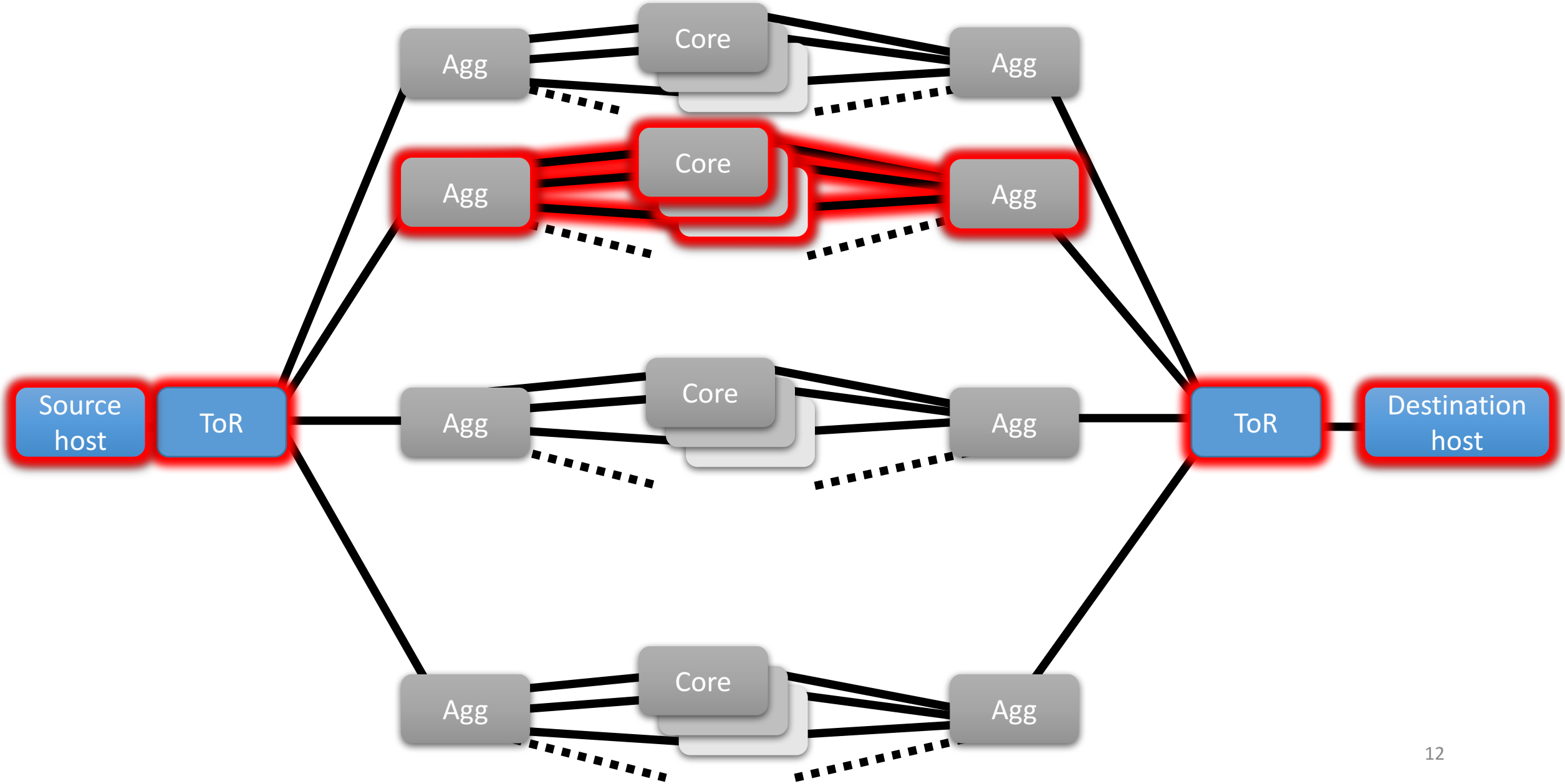
Finding path information at Facebook



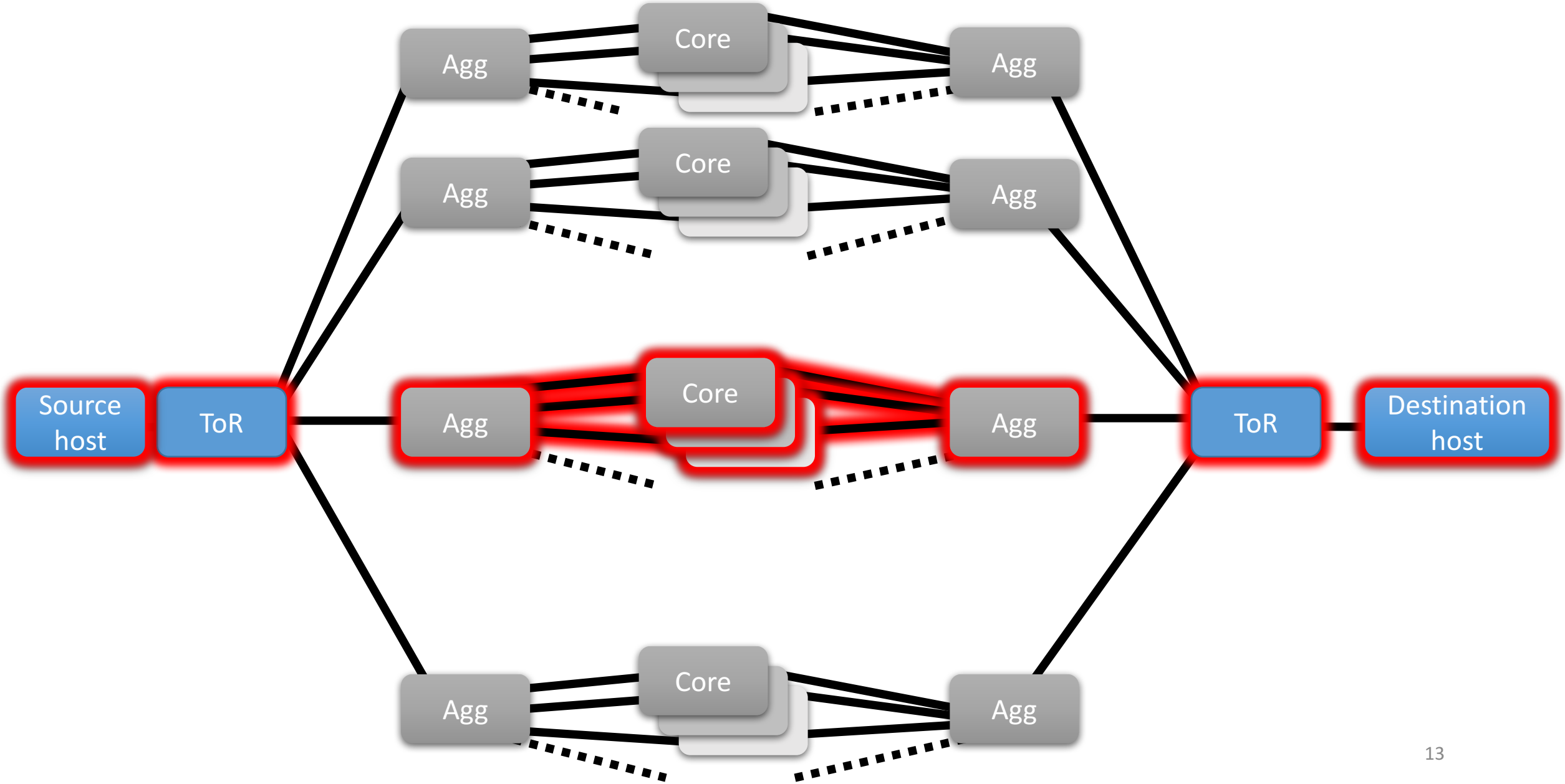
Finding path information at Facebook



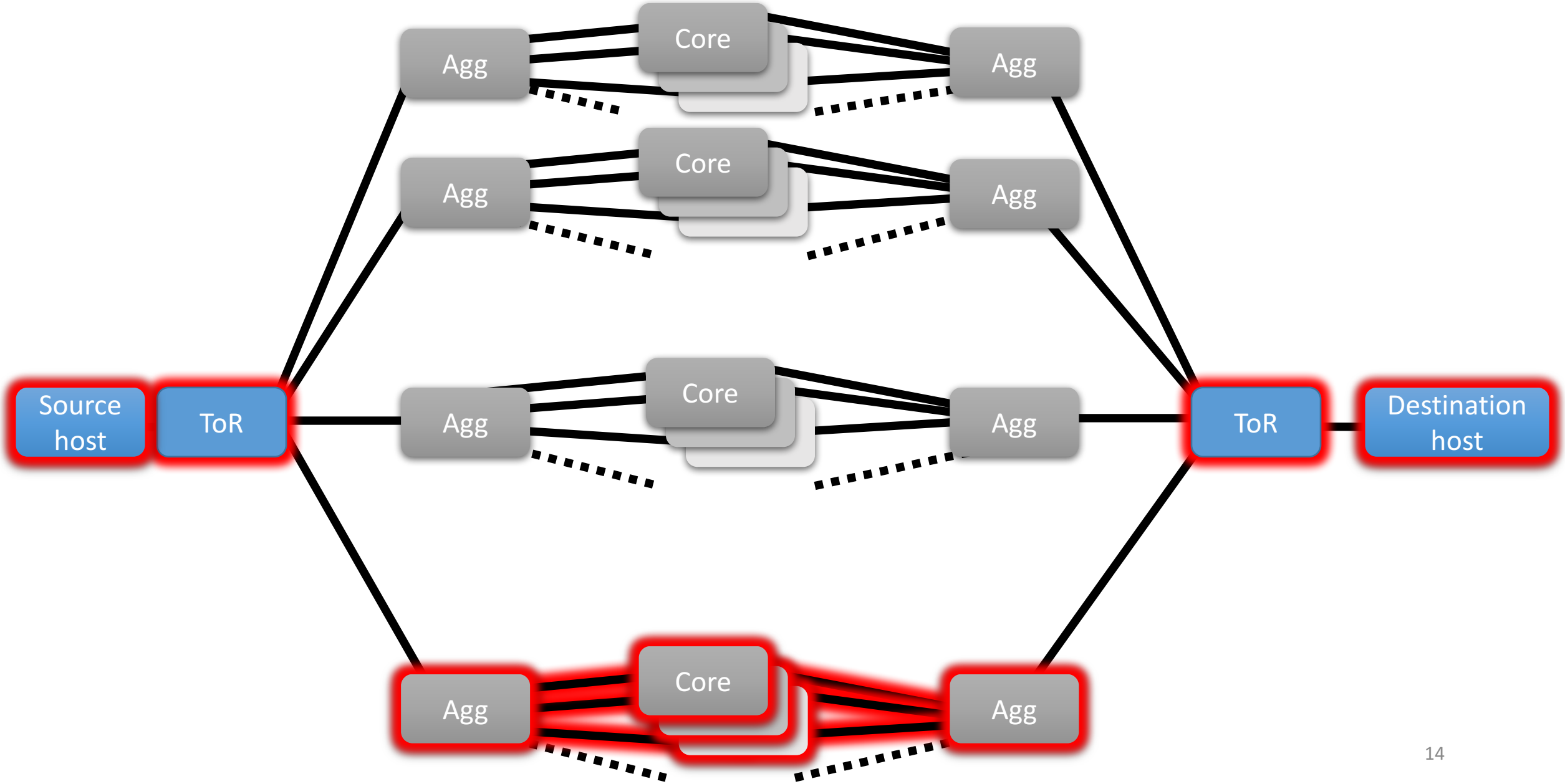
Finding path information at Facebook



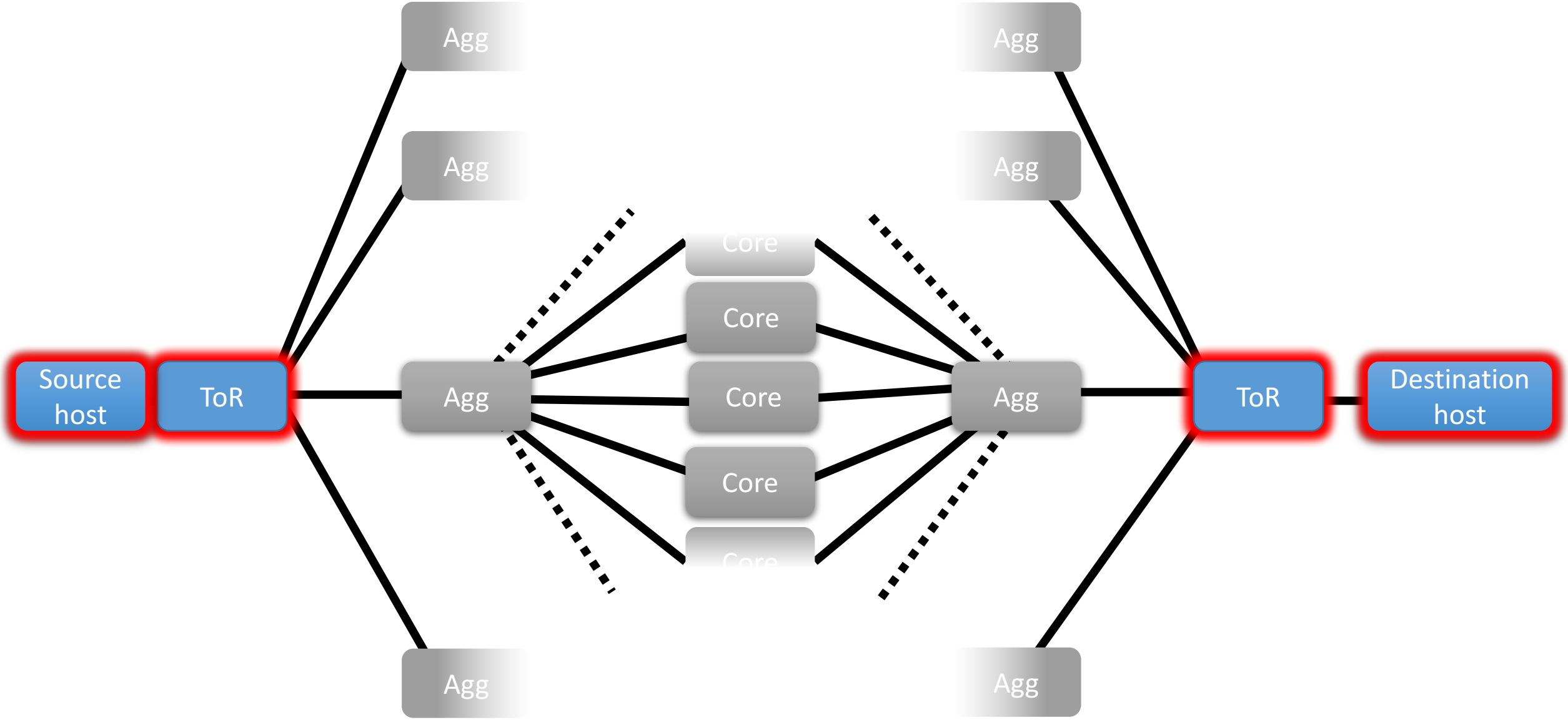
Finding path information at Facebook



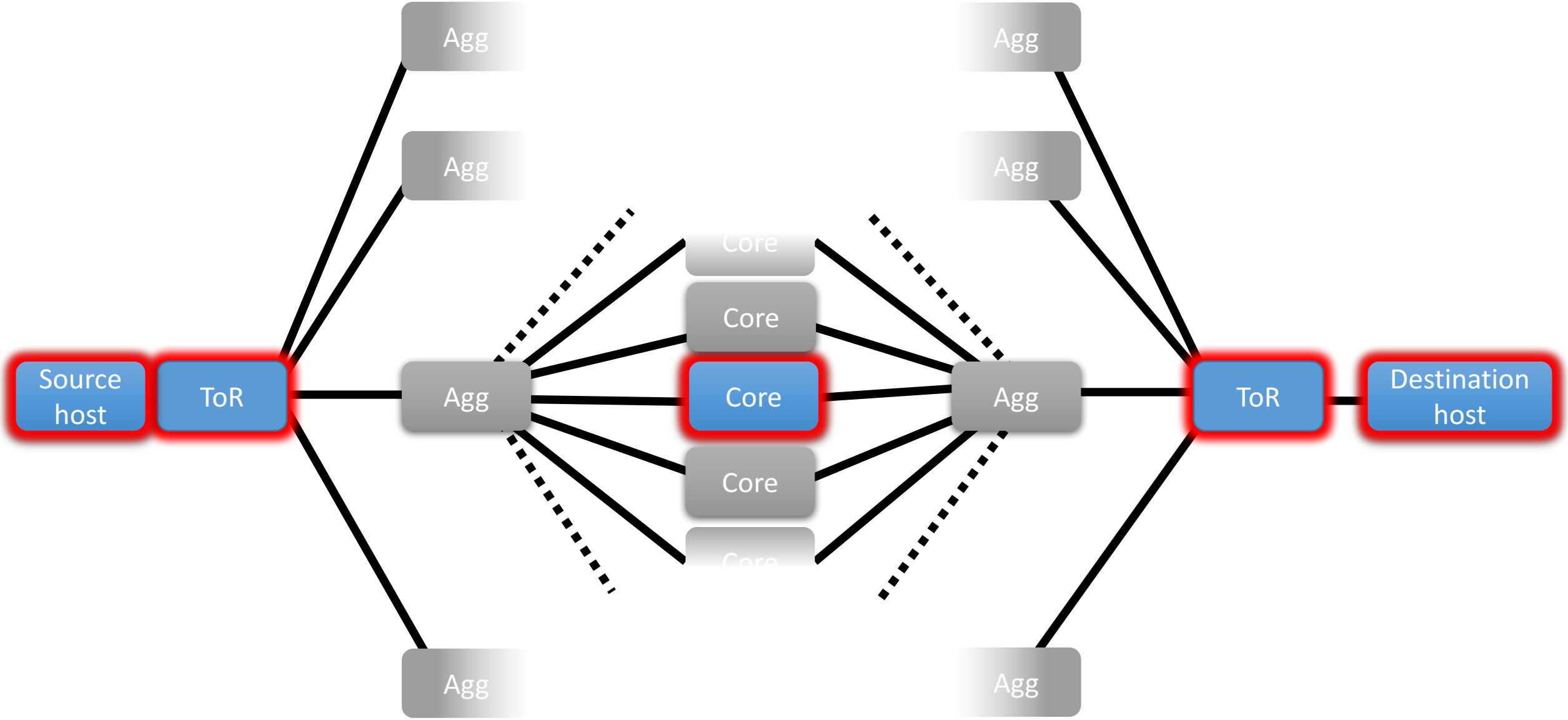
Finding path information at Facebook



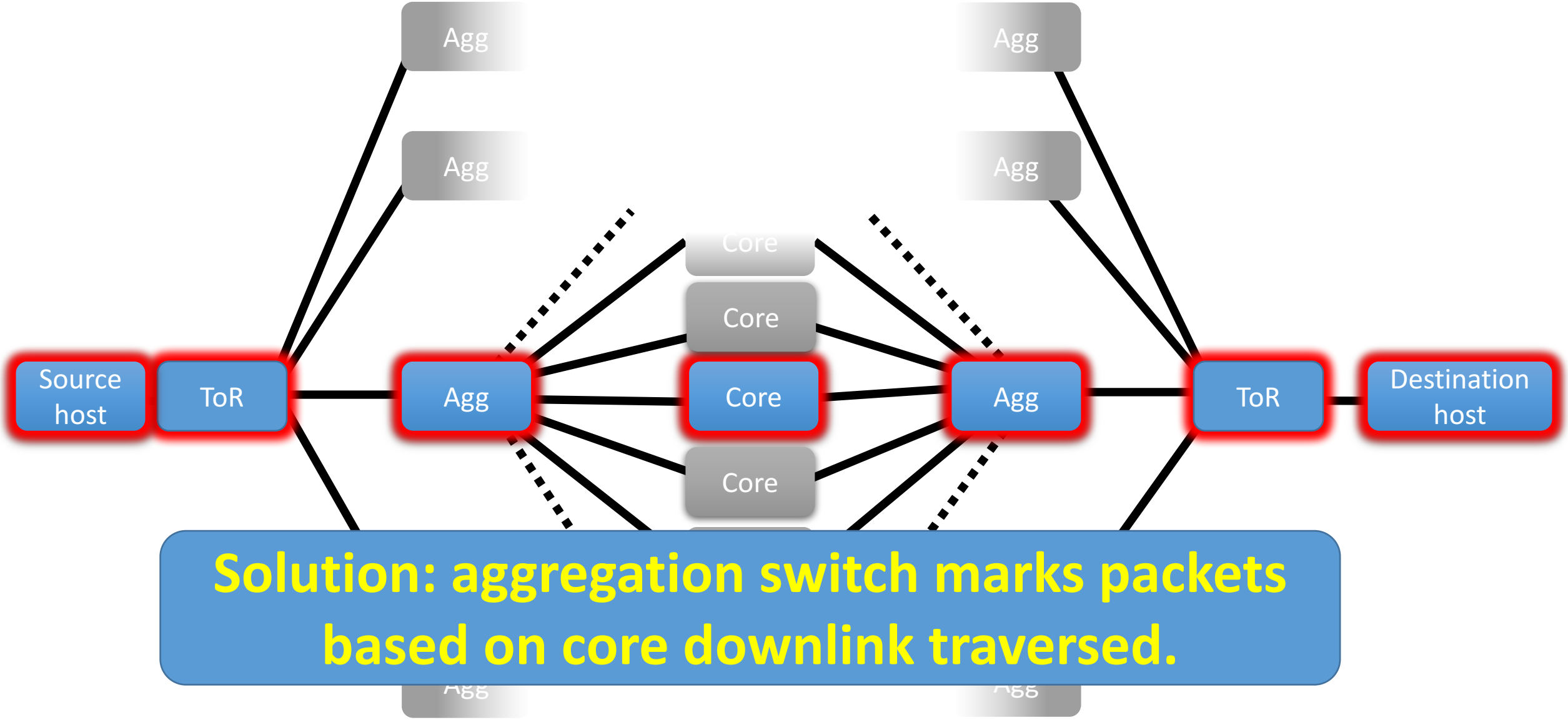
Finding path information at Facebook



Finding path information at Facebook



Finding path information at Facebook



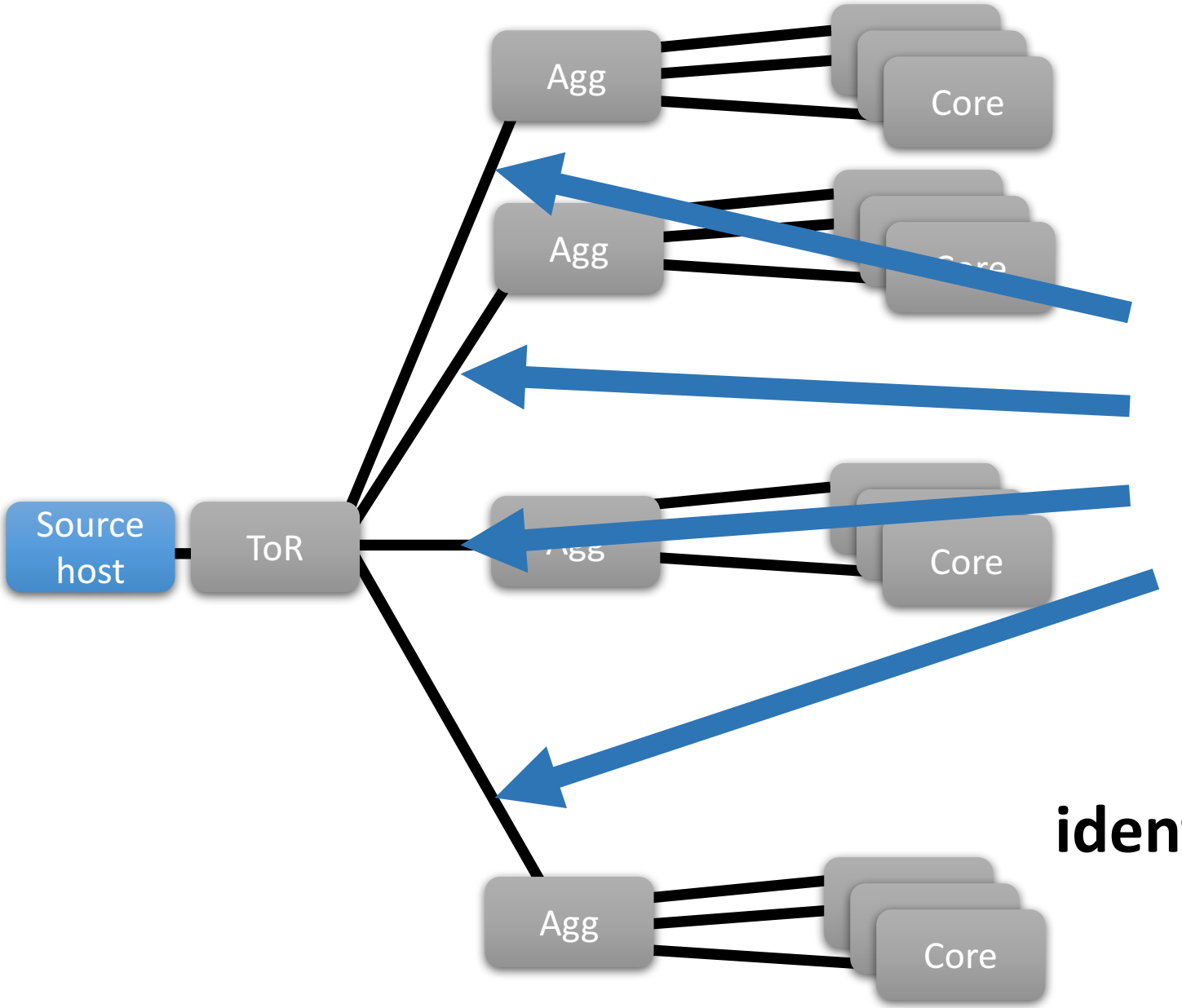
How do we use path information?

- In principle: can compare flow performance by path.
  - 1. Combinatorial disaster: O(10,000) paths from single host to remote racks.
- E
- Solution: Just compare links!

Equivalence sets:

1. **Reduces number of comparisons needed.**
2. **Pinpoints fault to specific location.**

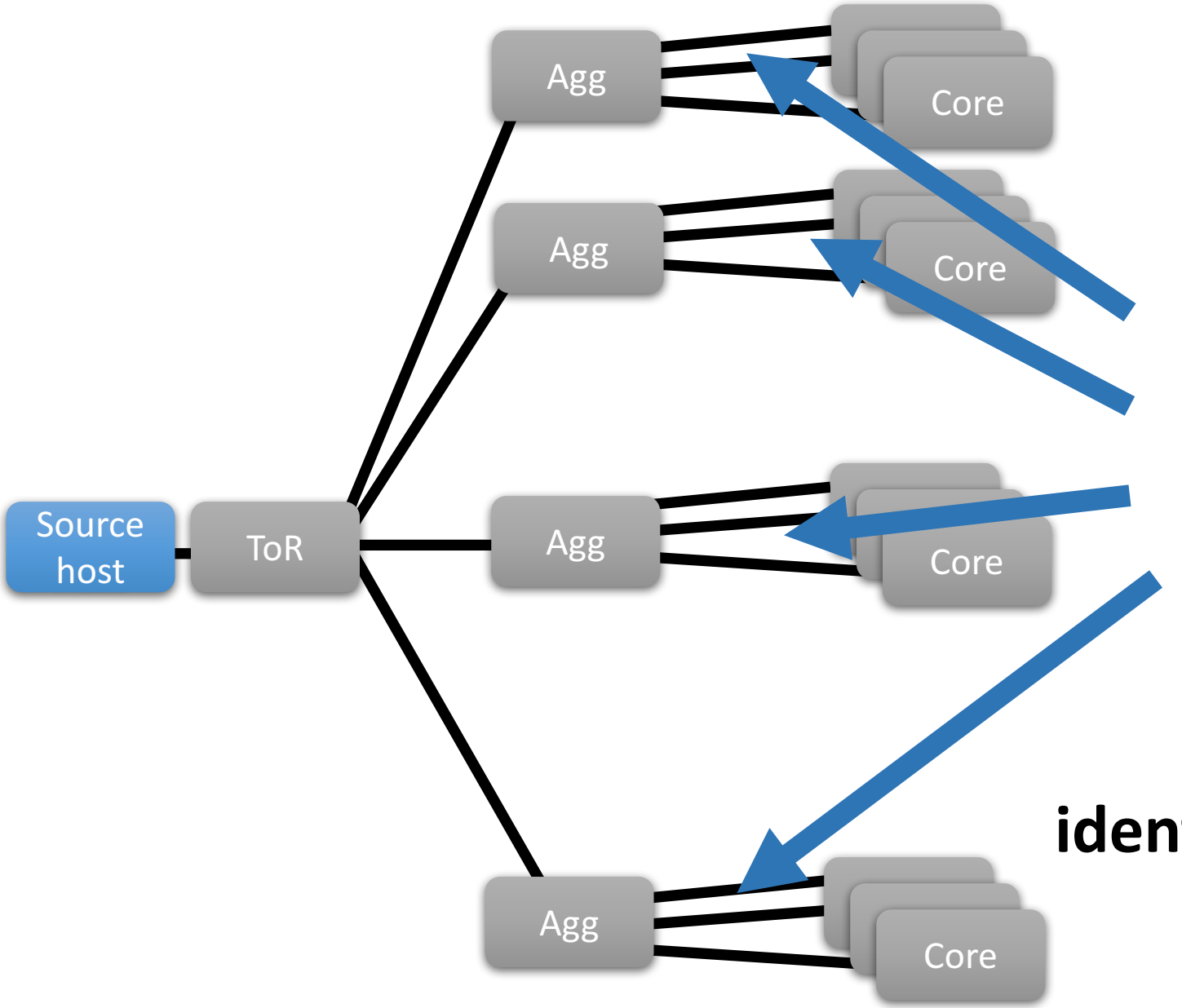
Equivalence sets in Facebook topology



Equivalence set:
4 uplinks from each ToR to pod Agg layer

…each has close to identical performance distribution in absence of errors

Equivalence sets in Facebook topology



Equivalence set:
N uplinks from
pod Agg layer
to core layer

…each has close to identical performance distribution in absence of errors

Outlier analysis with application agnostic metrics

Hosts already track metrics for congestion control or performance monitoring:

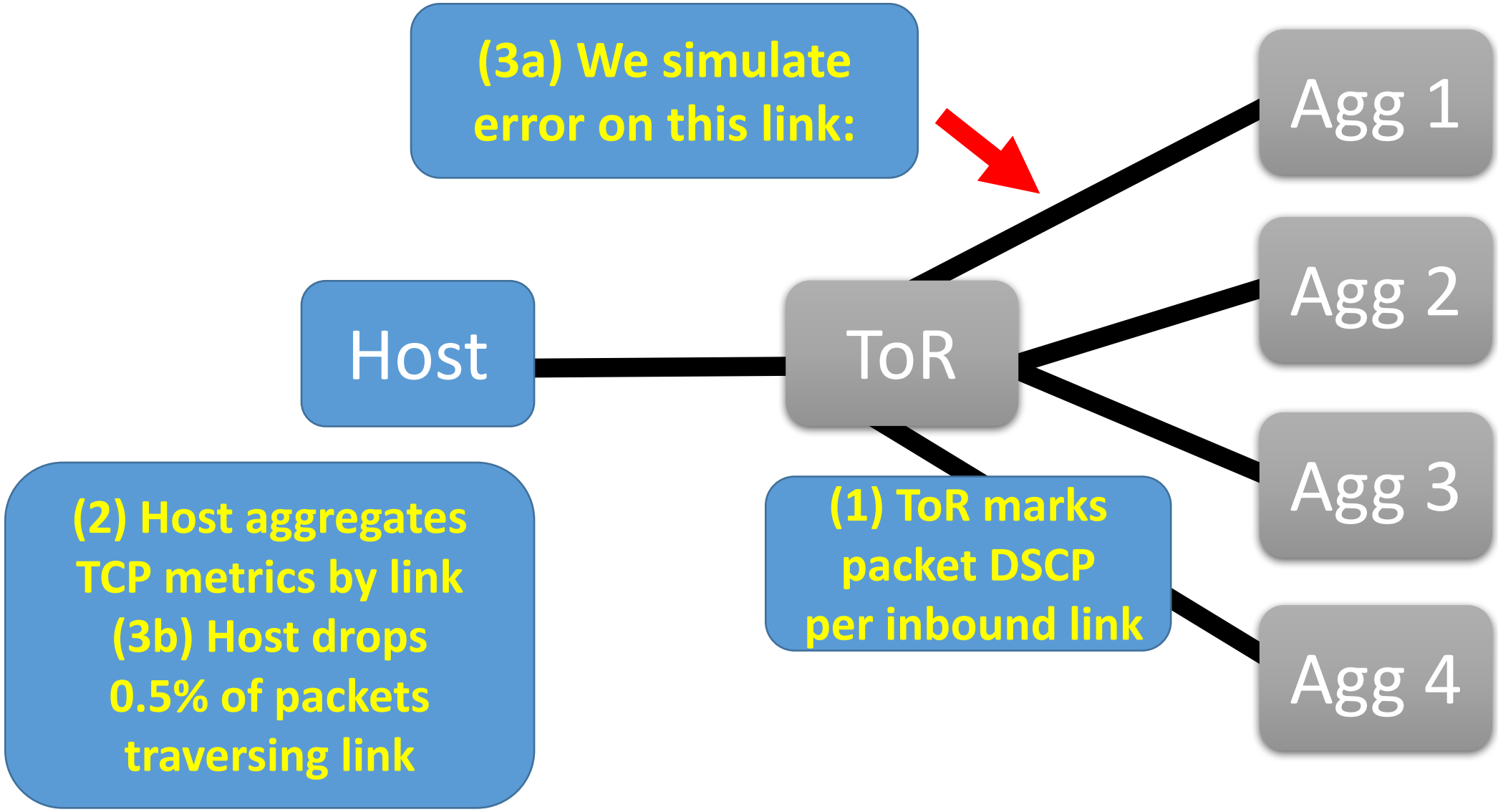
TCP Congestion window: Affected by packet loss.

TCP Retransmits: Affected by packet loss.

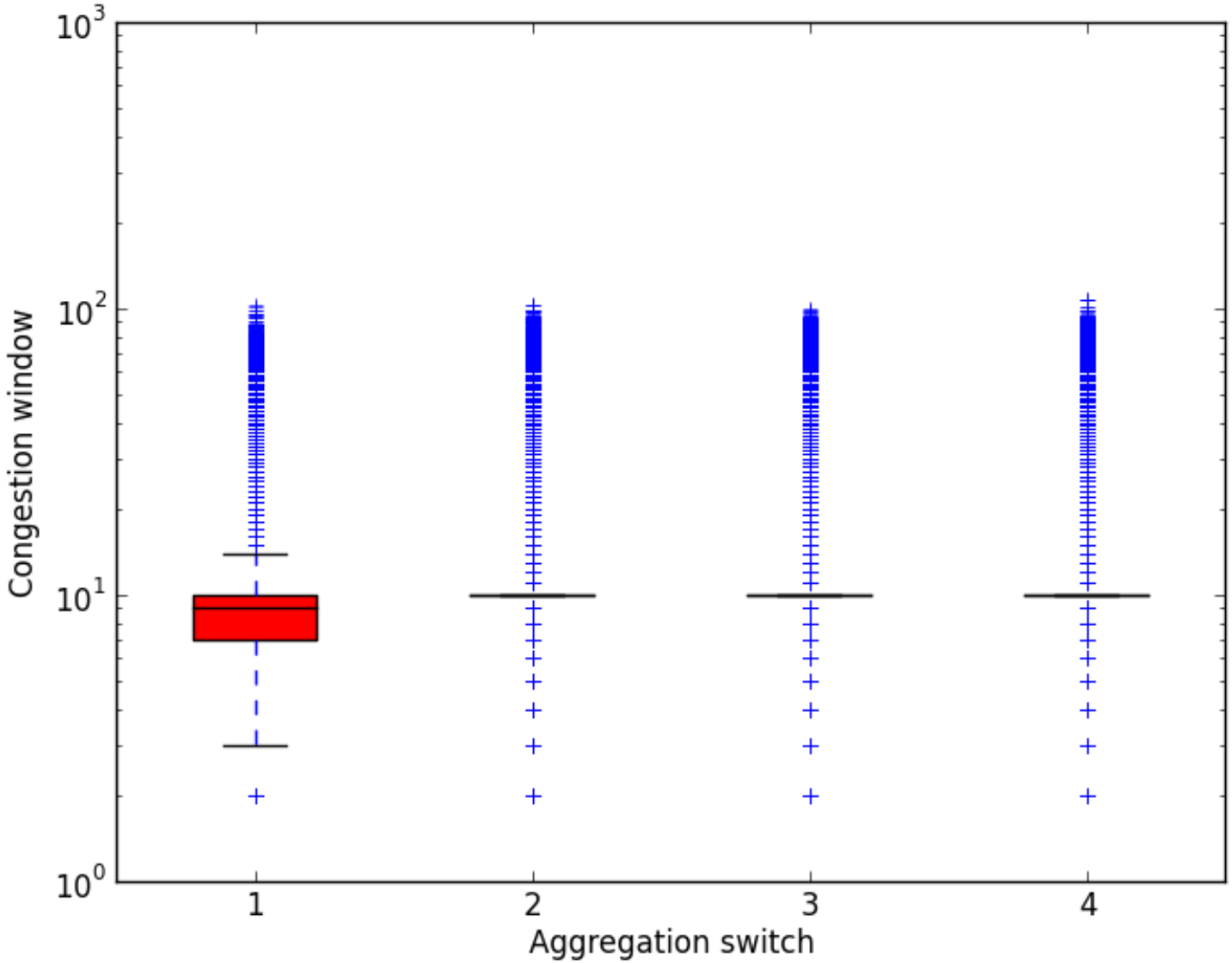
With equivalence set based grouping,
we can compare *distributions* by link.
Only link faults cause variance between links.

Caveat: Can be difficult to determine if an affect is due to a faulty link, overloaded hosts, application variance, etc.

Demonstrating equivalence sets from Agg to ToR

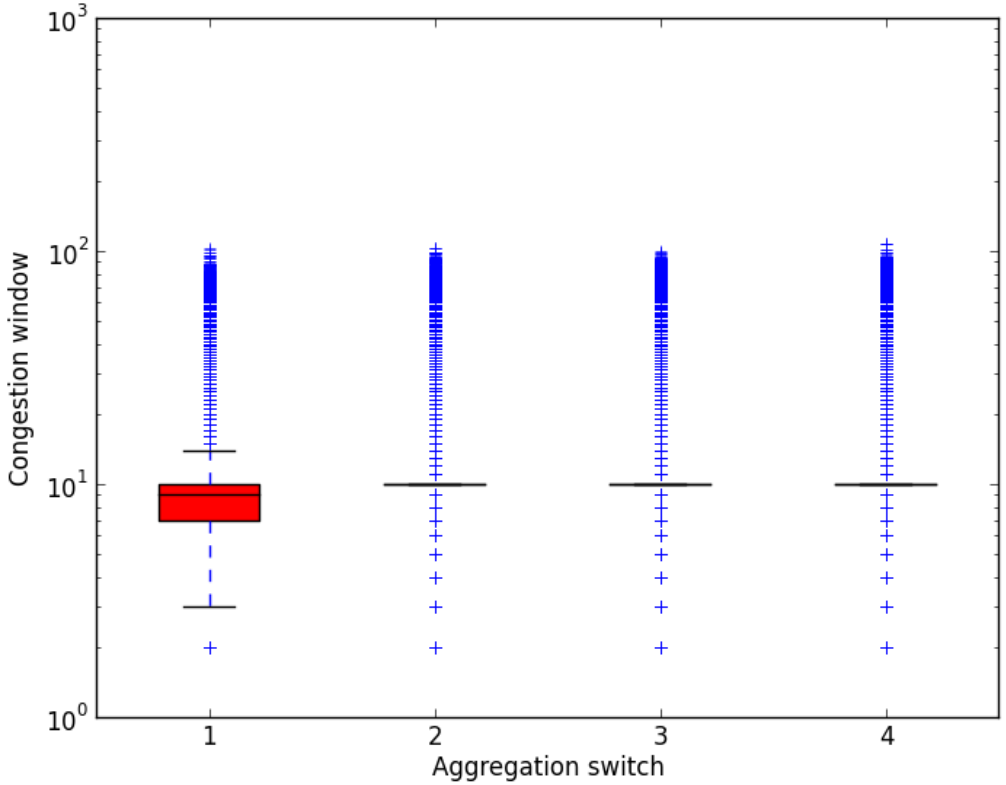


TCP Congestion window in Agg to ToR equivalence set

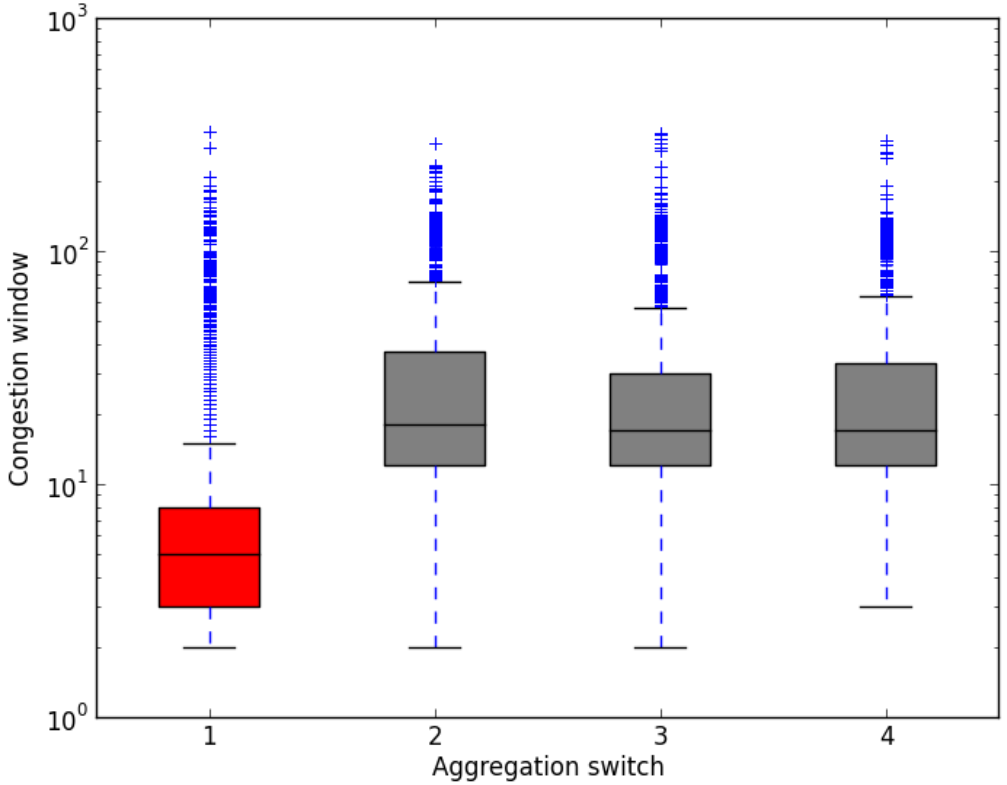


Cache server

Congestion window signal is application agnostic

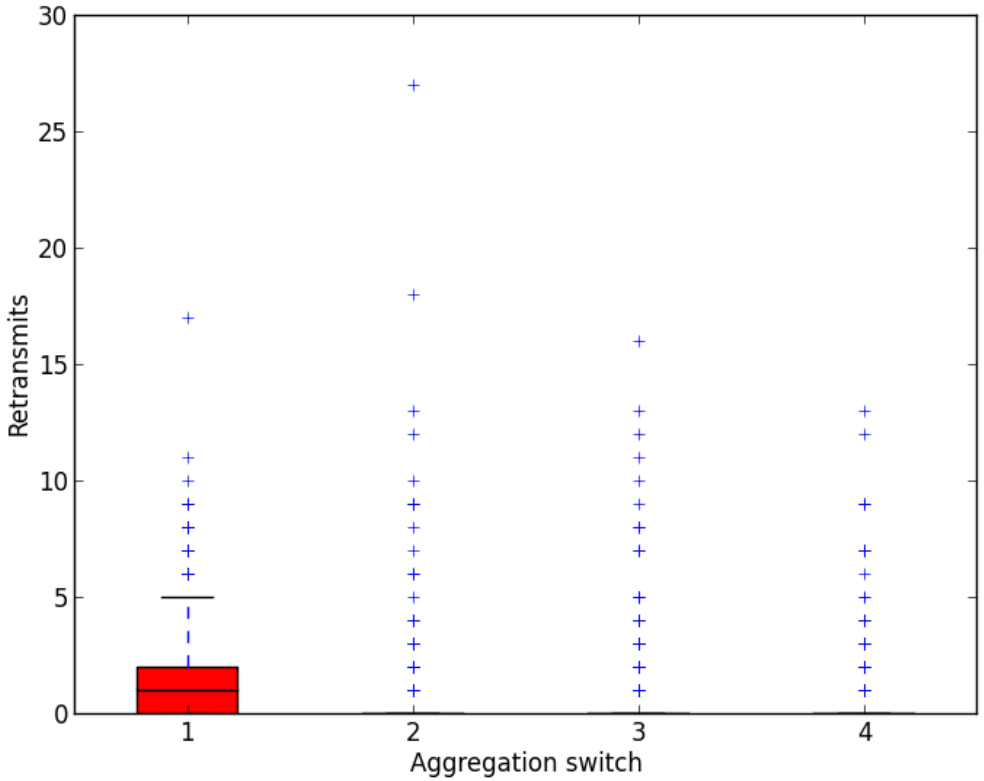


Cache server

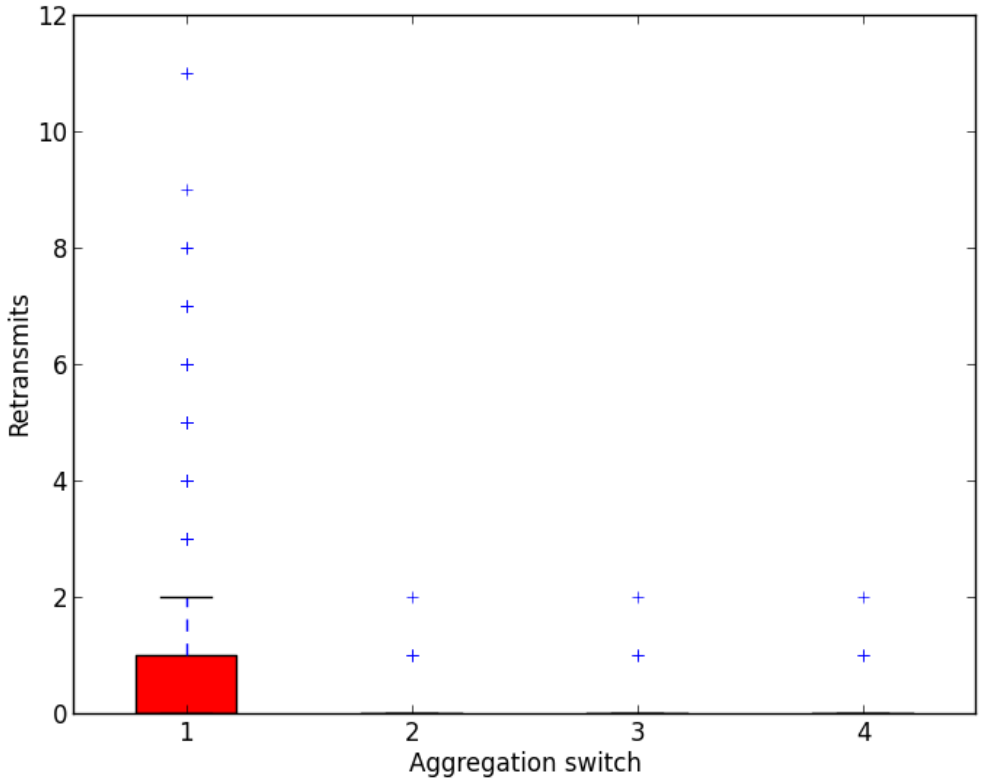


Web server

We use: TCP retransmits in our work



Cache server



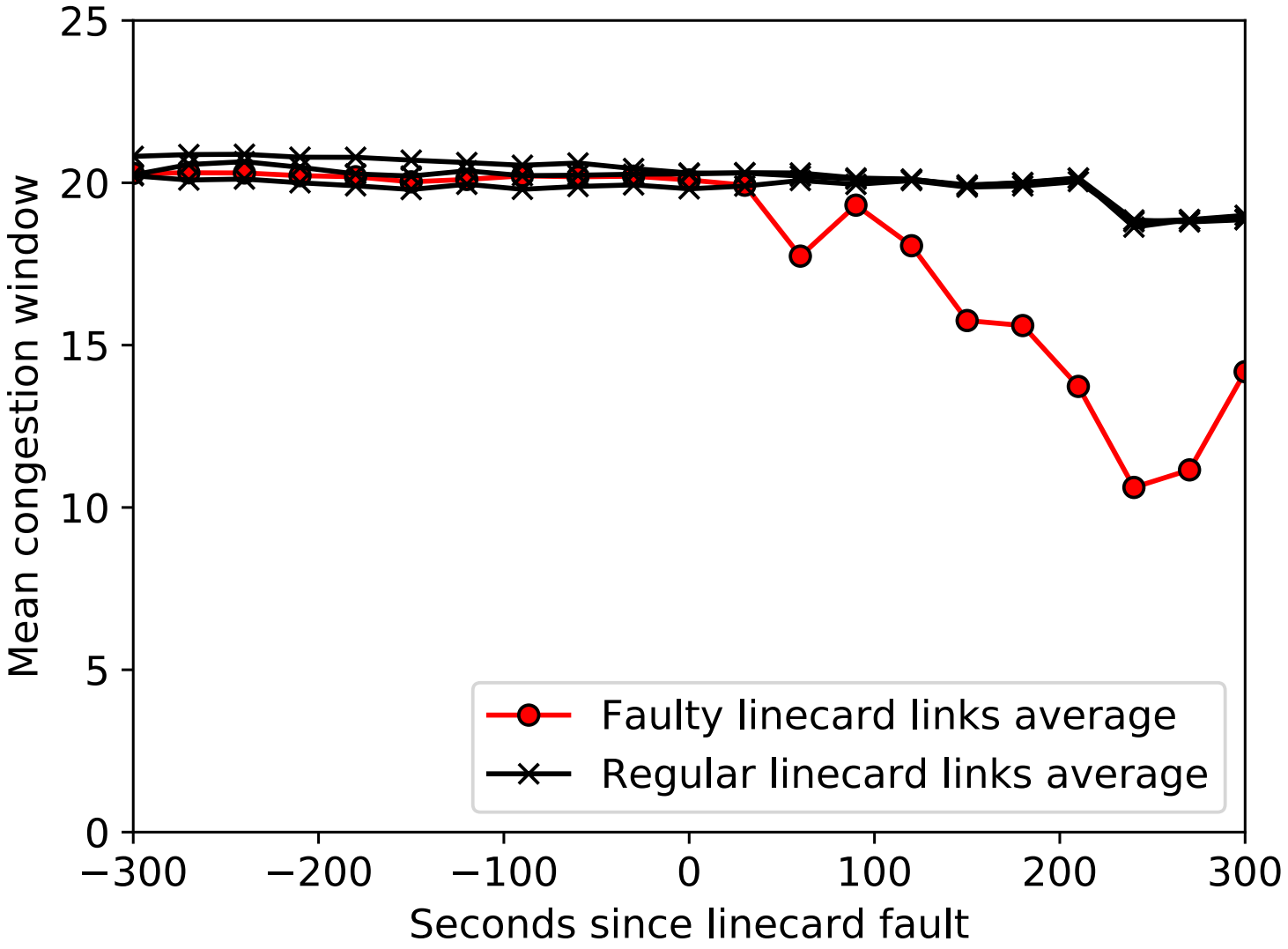
Web server

Detecting faults in production

- Monitored traffic through pod aggregation switch.
  1. No faults injected.
  2. Collected TCP metric data on 30 web server hosts.
  3. Equivalence set: four linecards connecting to core layer (each linecard has equal share of uplinks).

- On January 25th, a single linecard had a software fault.
  1. Linecard controller software hung.
  2. BGP routes timed out, production traffic through linecard routed away.
  3. A few minutes later, NetNORAD flagged unresponsive linecard.

Fault visible to our approach in 30 seconds



Classifying faulty links

- “Does this link have more retransmits per flow than the other links?”
- “Do two distributions have the same mean, or is one greater?”

Classifier: compare each link to other links with one sample Student’s T-Test.

Online fault monitoring with T-Test alone

- In principle: can setup a system that uses end host T-Test result to tell us which network links are faulty.
- However: by itself this is susceptible to *False Positives.*
- Can’t afford false positives in network with O(10,000) links!

Accounting for false positives

- However, two characteristics aid us:

Chi-squared test: determines if any links are outliers.

P-Value ≈ 1: "Yes, all the links being marked faulty by hosts at similar rates."

P-Value ≈ 0: "No, a subset has a comparatively high percentage of hosts claiming fault."

1. "Are all the links being marked faulty by hosts at similar rates?"
2. "Are hosts flagging a particular subset of links as faulty at higher rates?"

Evaluation in the datacenter

- Small detection surface; did not detect any ‘organic’ partial faults.
- Approach: inject ‘simulated’ faults to evaluate approach.
- Induced a variety of fault scenarios to challenge our system.

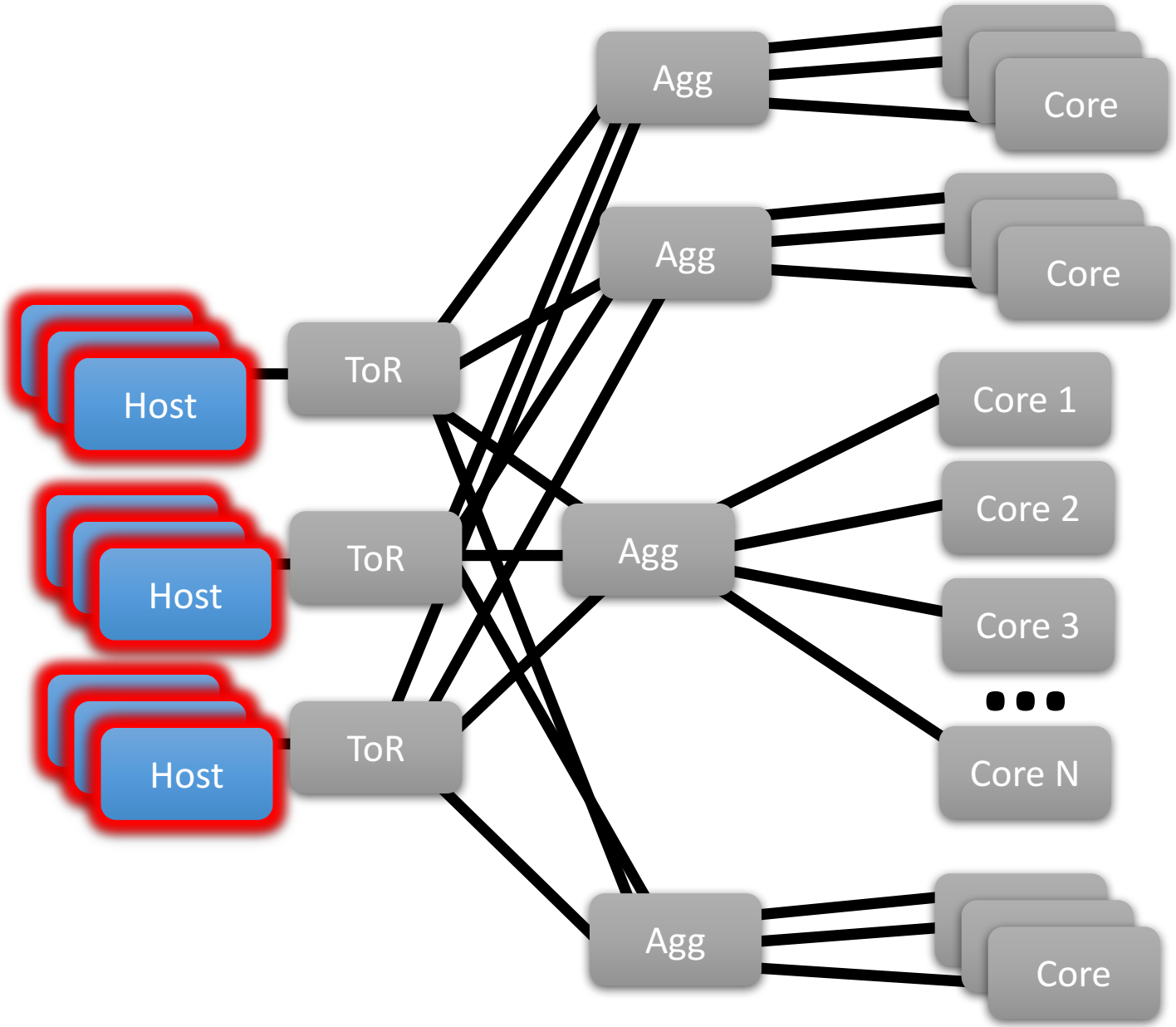
Evaluation in the datacenter: fault scenarios

- Miniscule faults: faults can have ***very*** low drop rates.
- Concurrent faults: multiple faults can occur simultaneously.
- Masked faults: larger fault can mask effect of miniscule fault.
- Correlated faults: hardware fault can impact multiple nearby links, confounding outlier analysis.

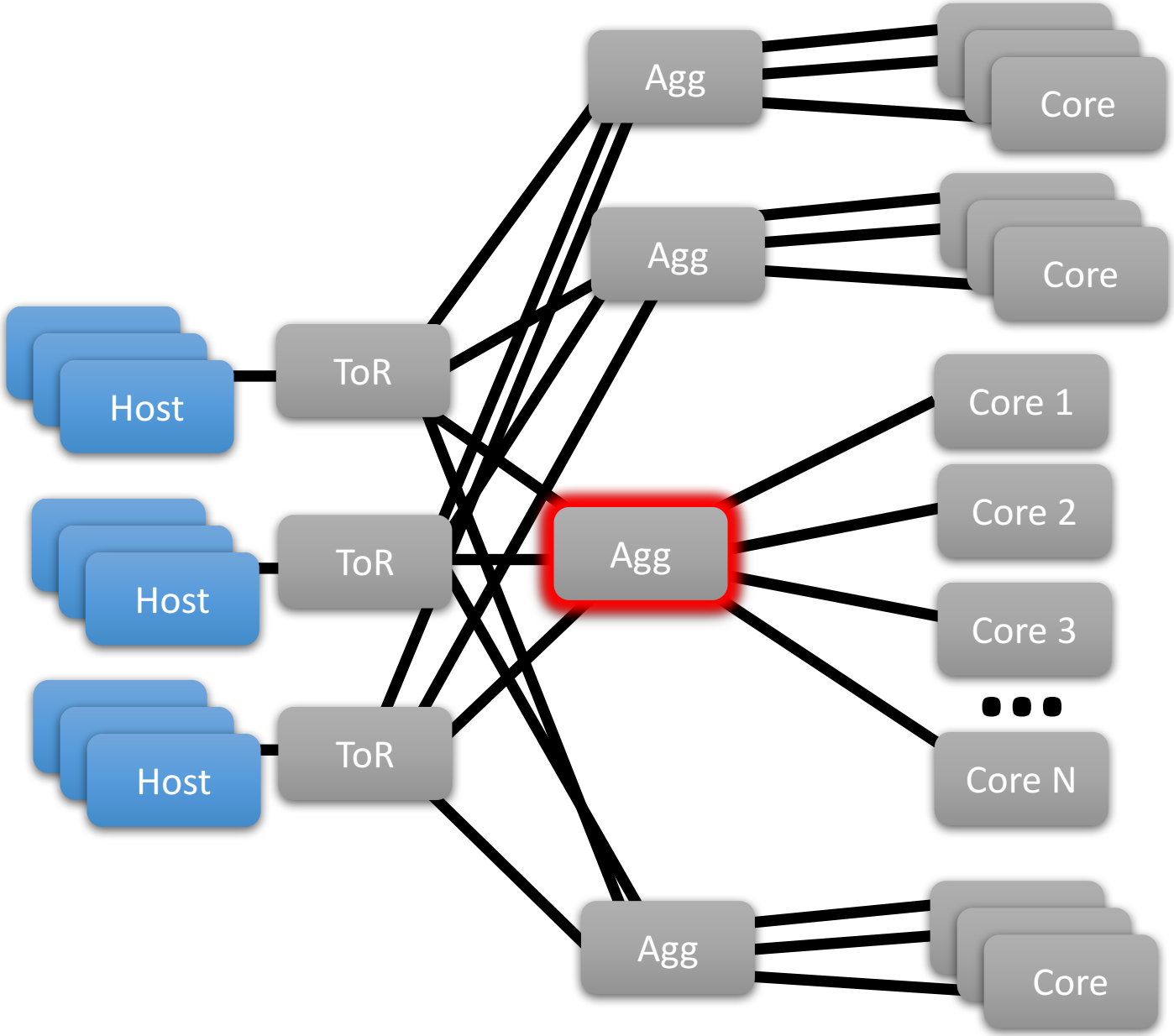
Evaluation in the datacenter: fault scenarios

- **Miniscule faults: faults can have *very* low drop rates.**
- Concurrent faults: multiple faults can occur simultaneously.
- Masked faults: larger fault can mask effect of miniscule fault.
- Correlated faults: hardware fault can impact multiple nearby links, confounding outlier analysis.

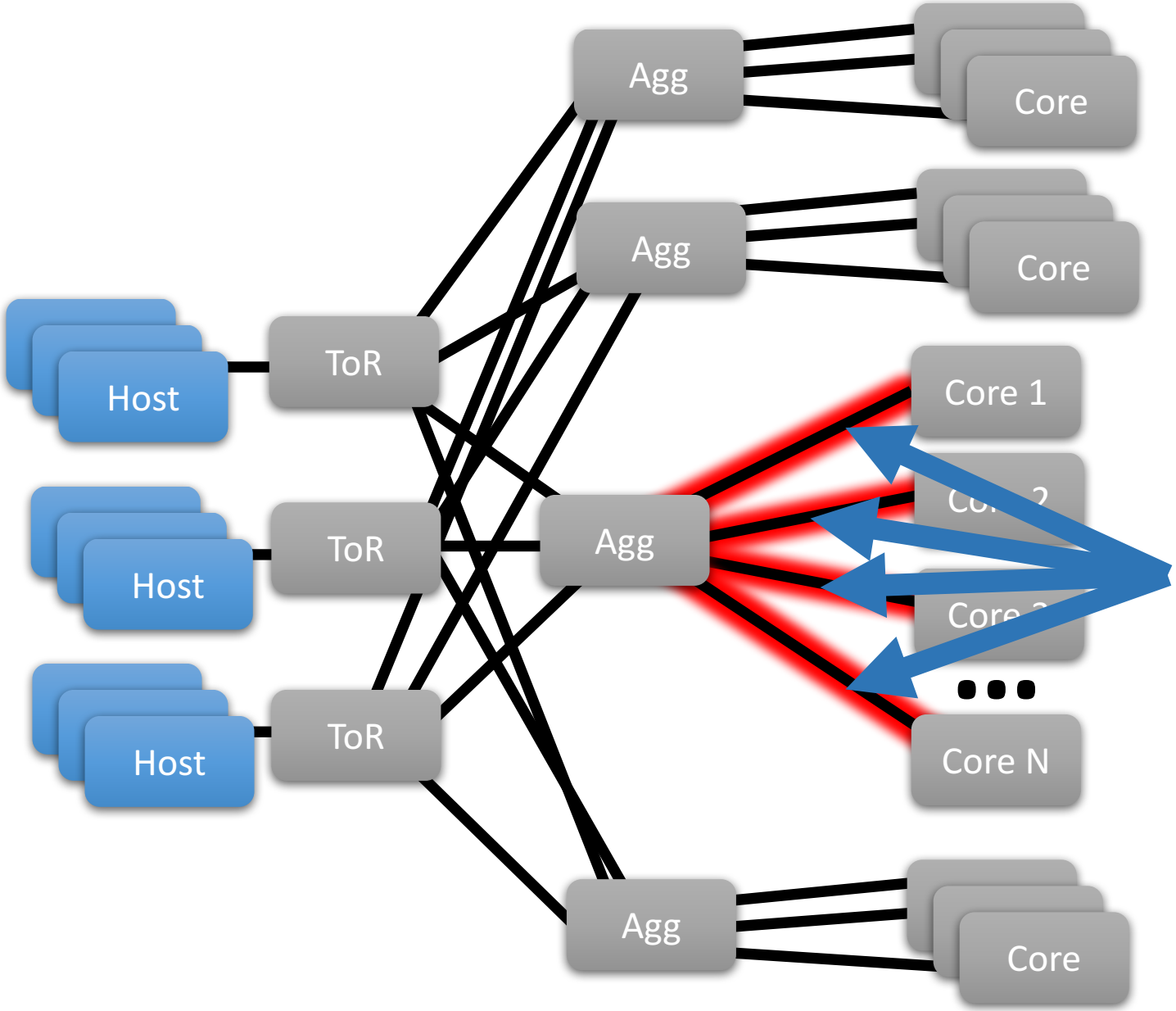
Finding miniscule faults: experiment setup



Finding miniscule faults: experiment setup

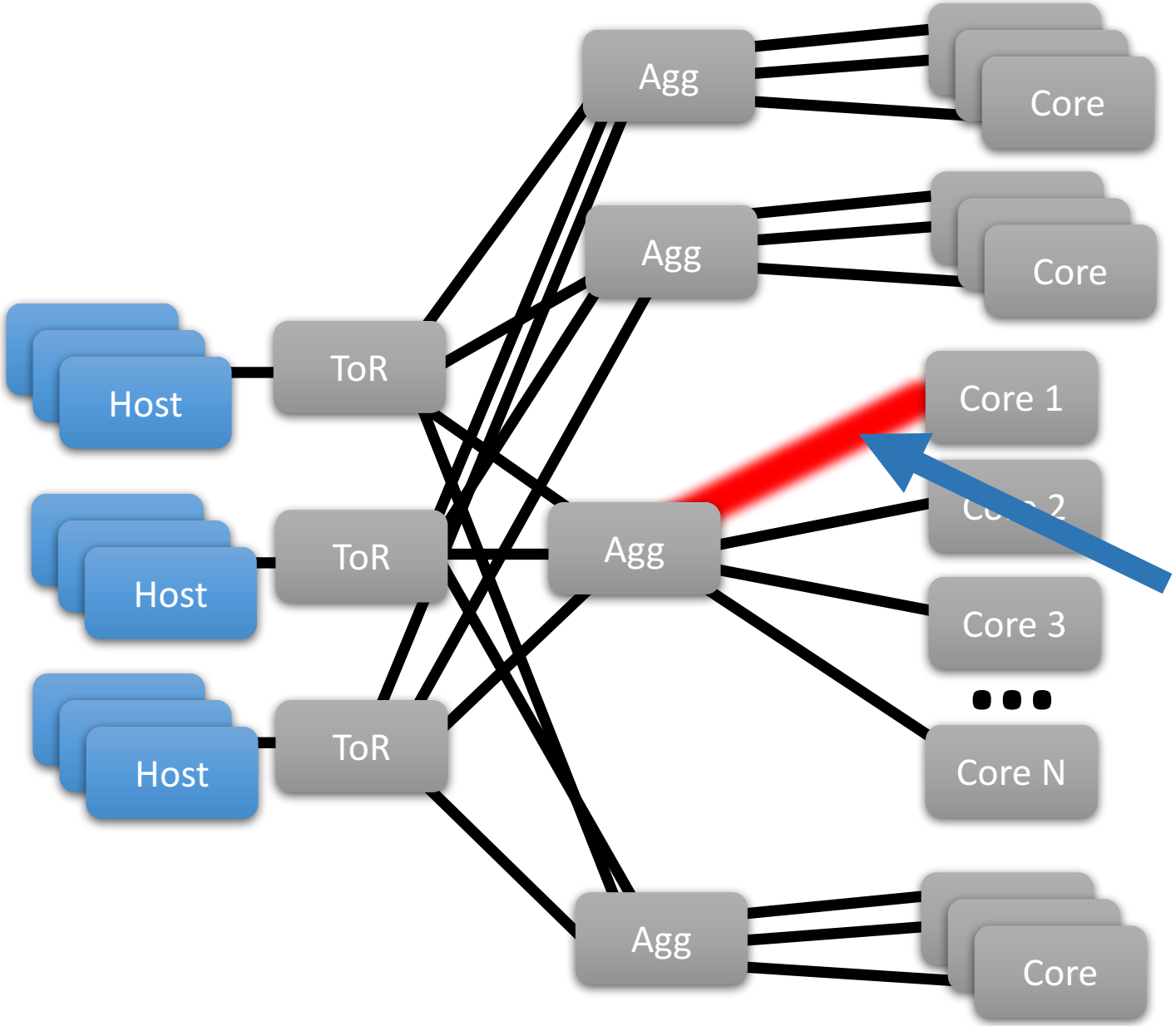


Finding miniscule faults: experiment setup



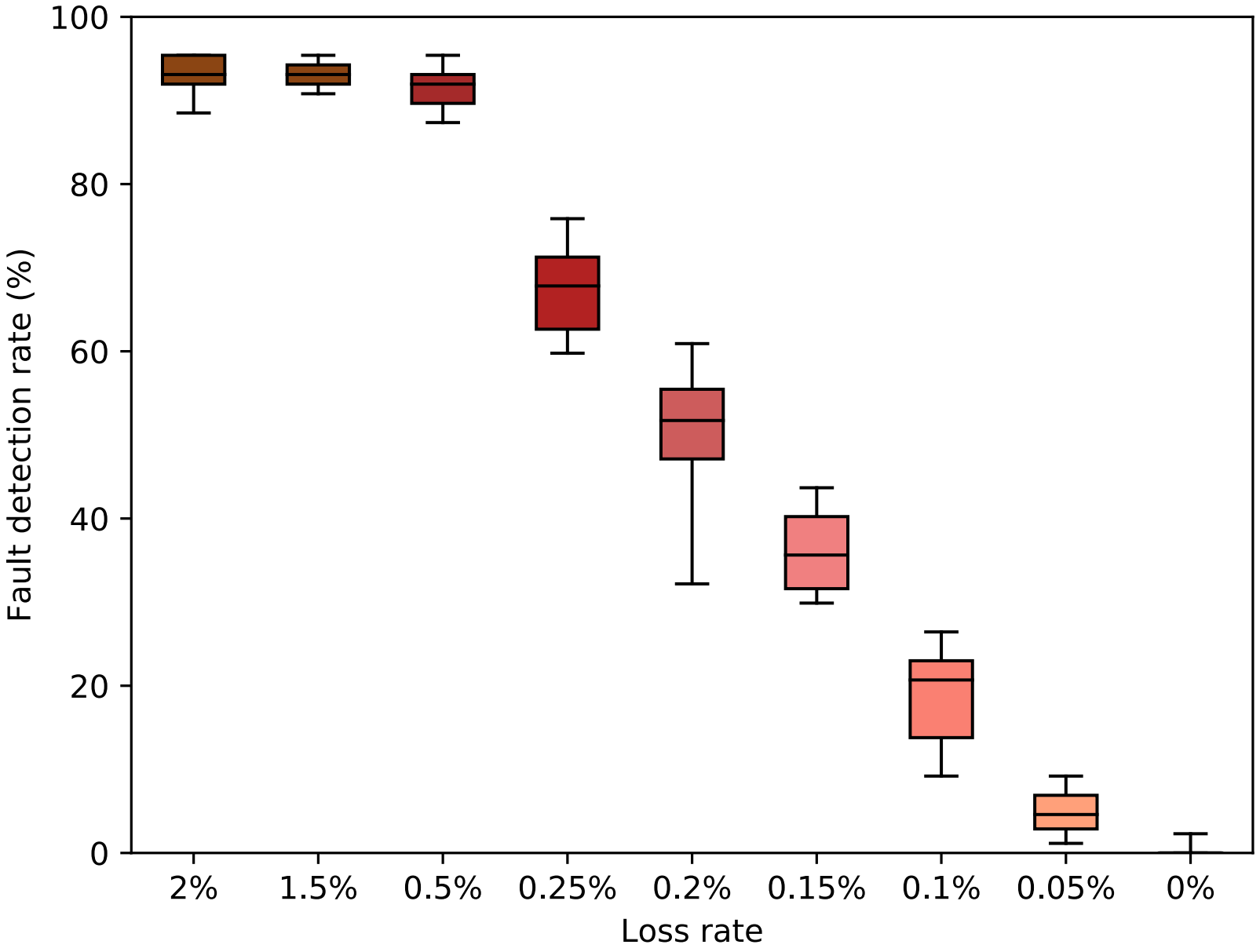
Equivalence set:
N uplinks from pod Agg layer to core layer

Finding miniscule faults: experiment setup

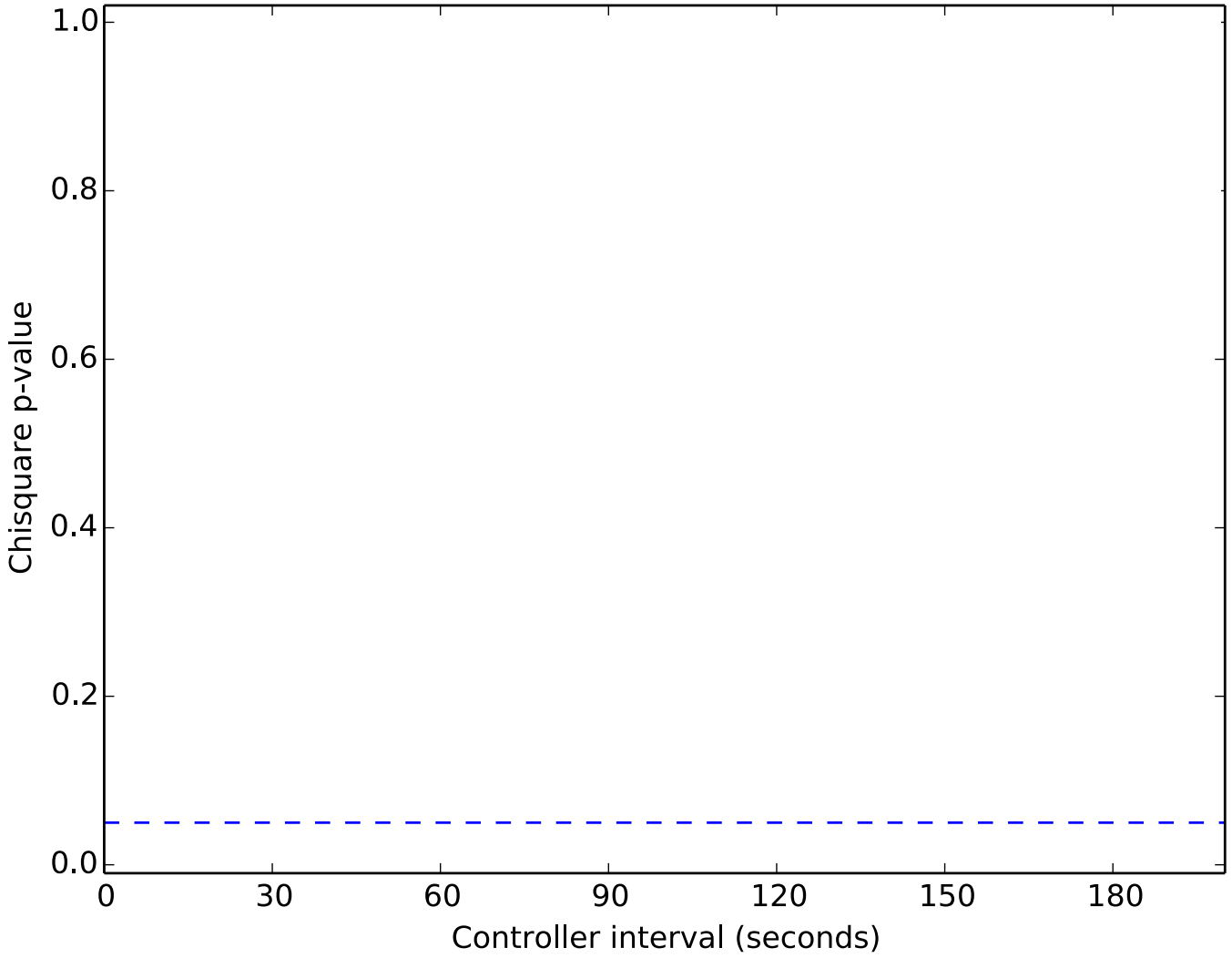


Partial fault induced on single Core to Agg downlink.

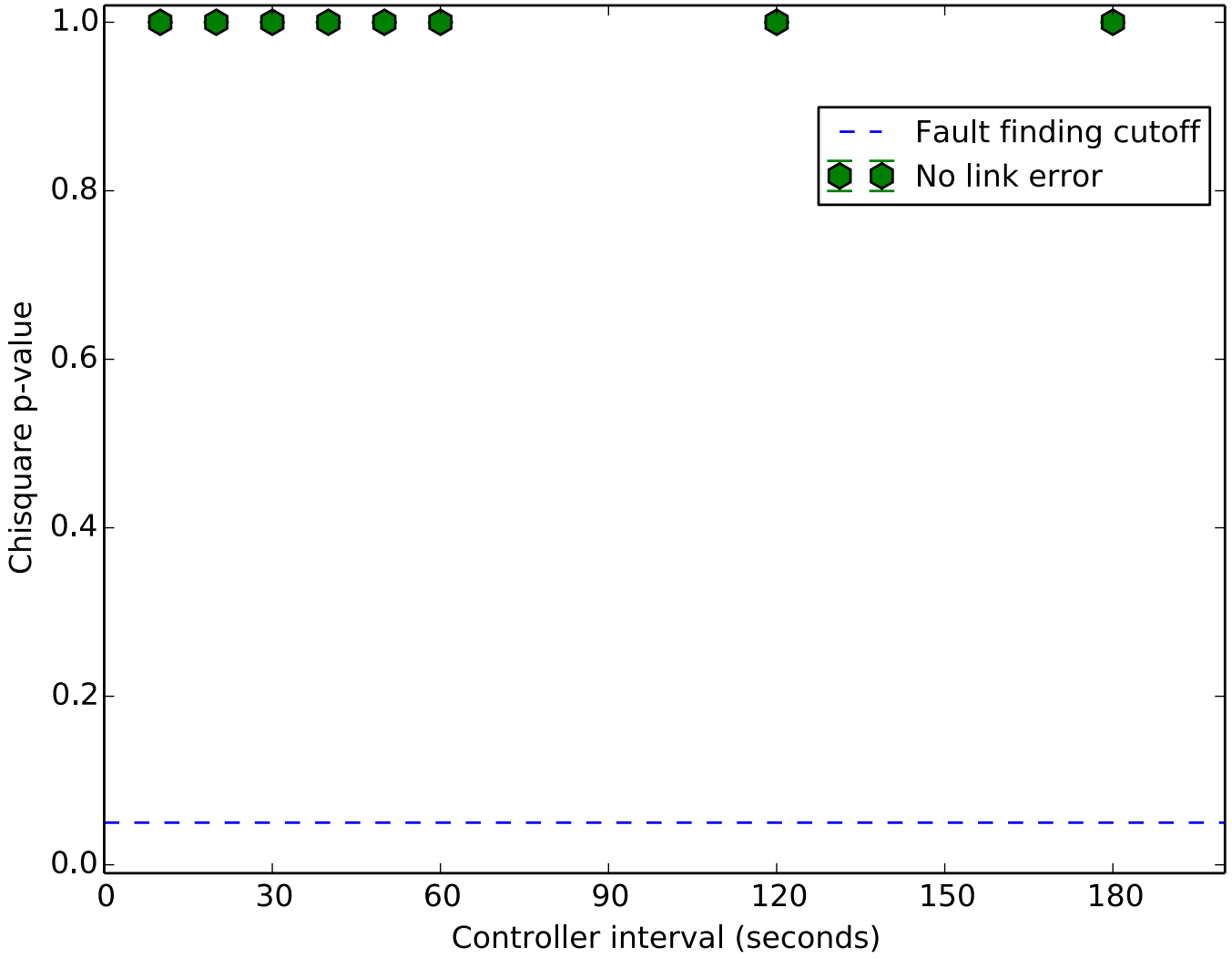
Fault detection rate vs drop rate



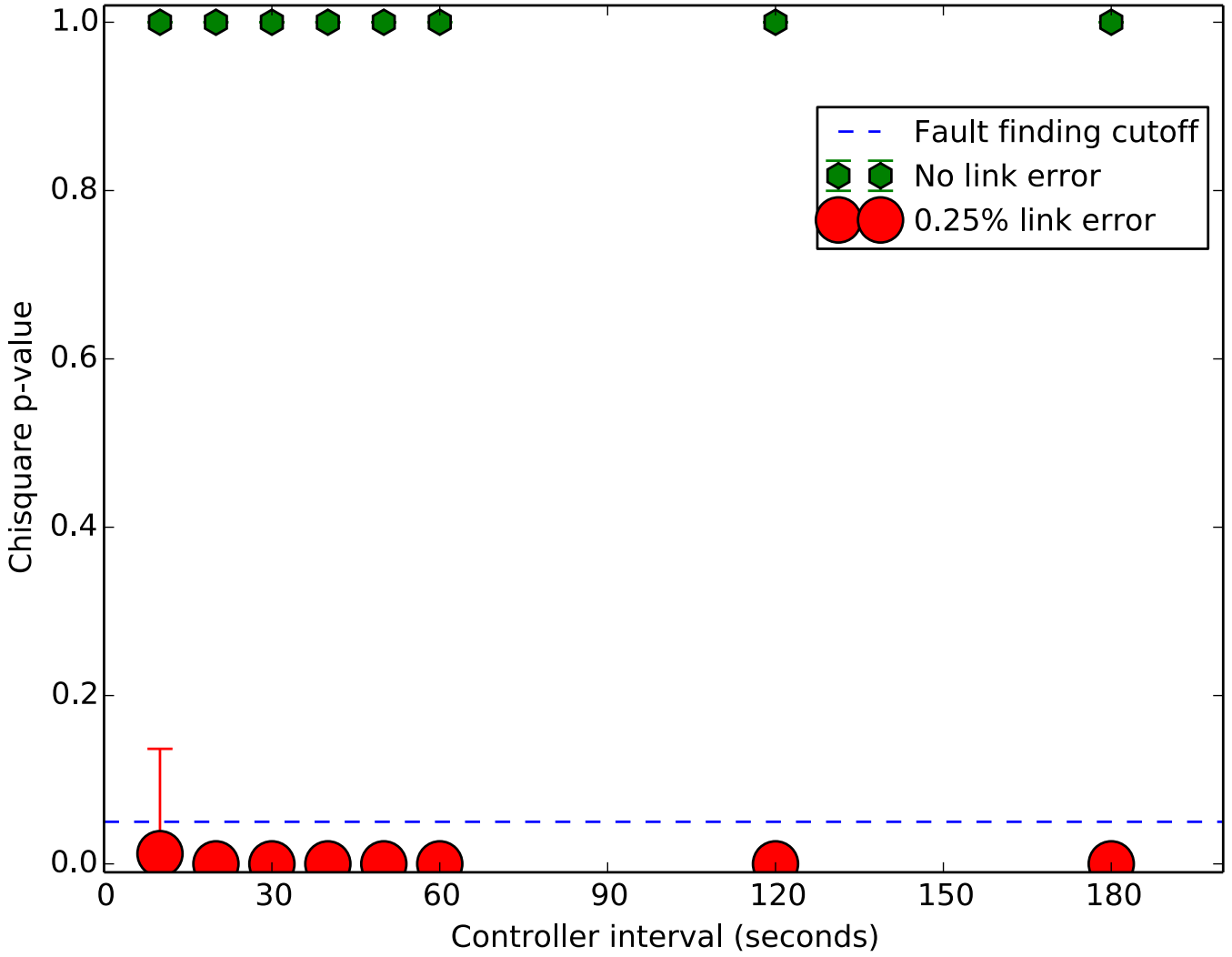
Miniscule faults: choosing between detection speed and sensitivity



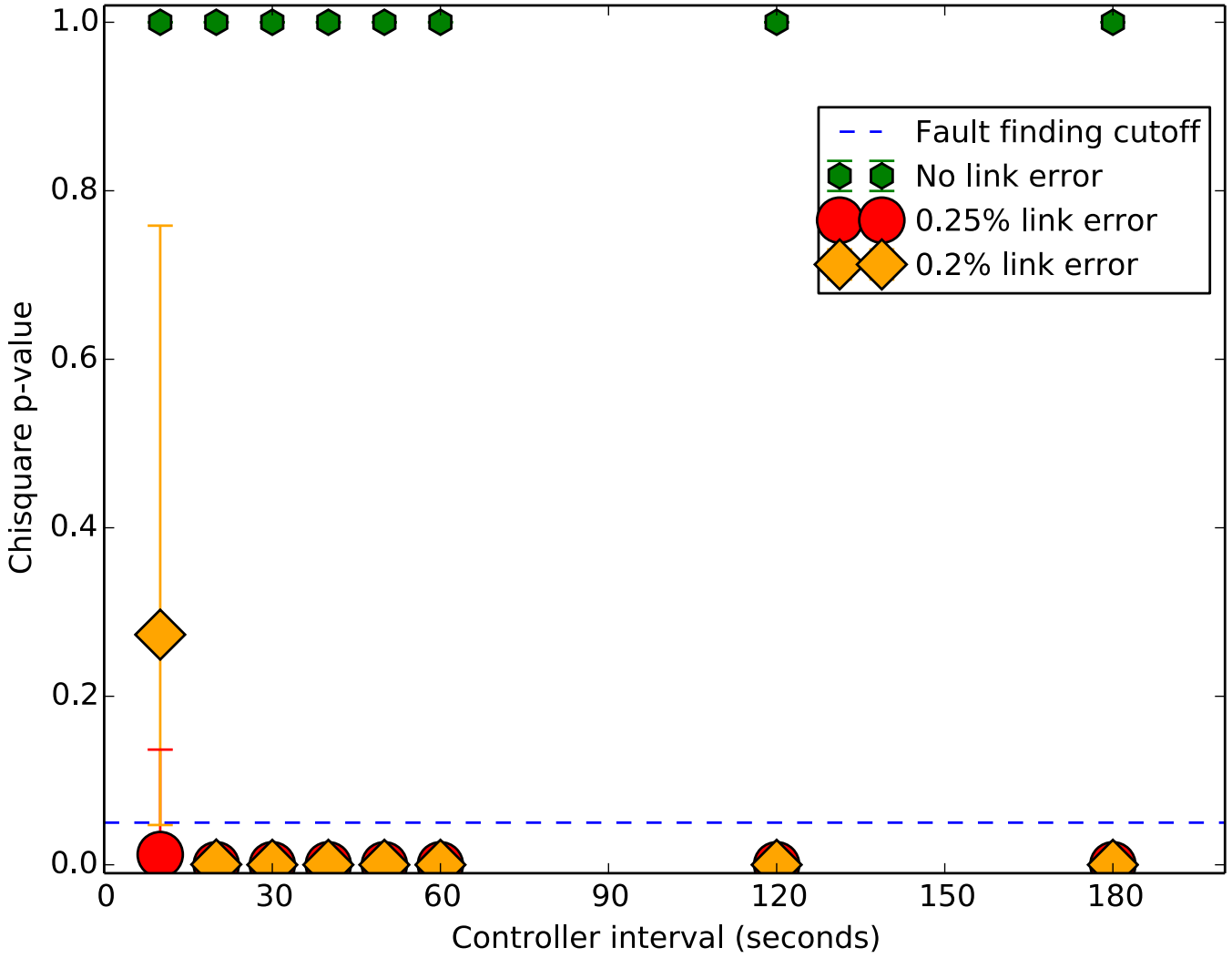
Miniscule faults: choosing between detection speed and sensitivity



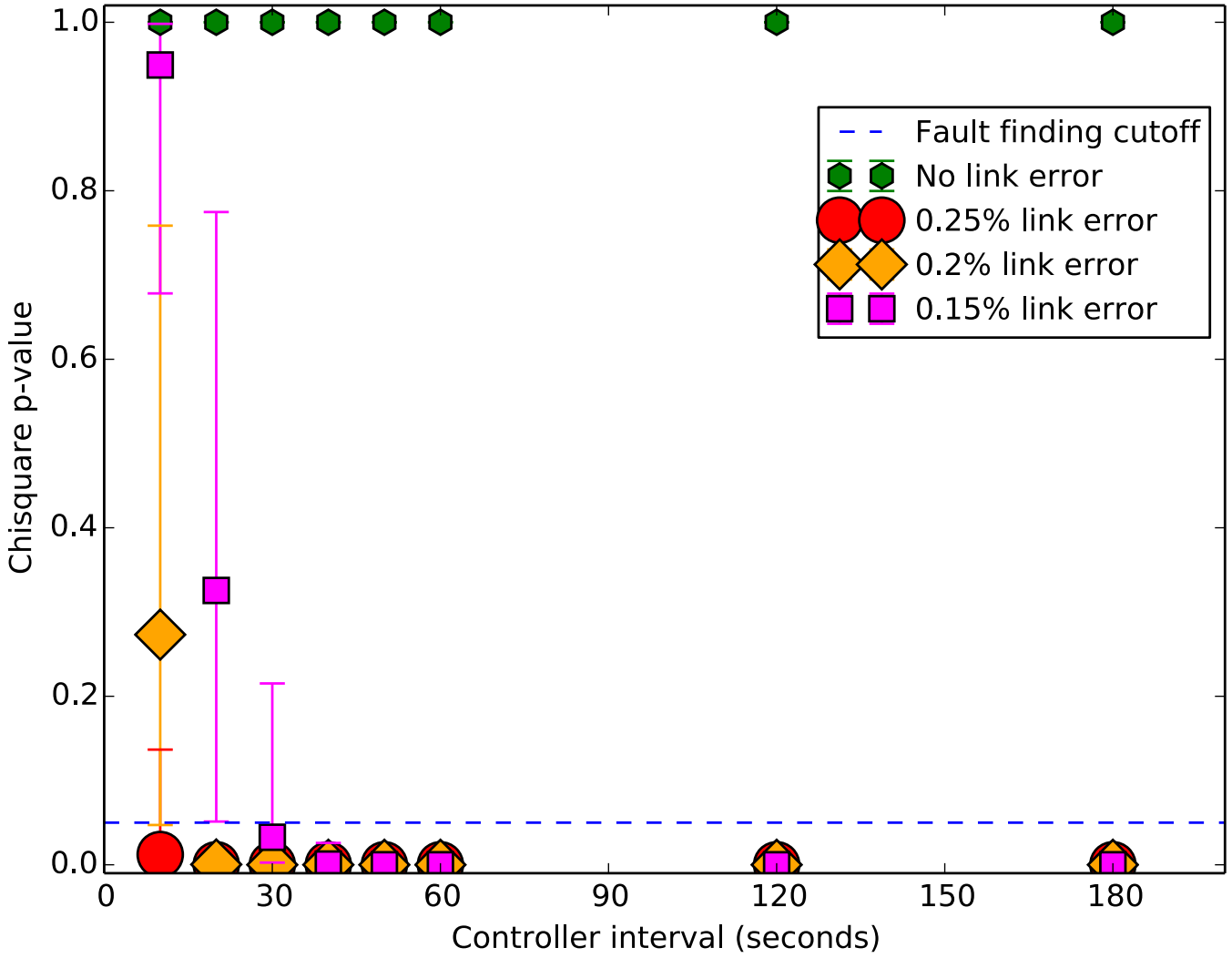
Miniscule faults: choosing between detection speed and sensitivity



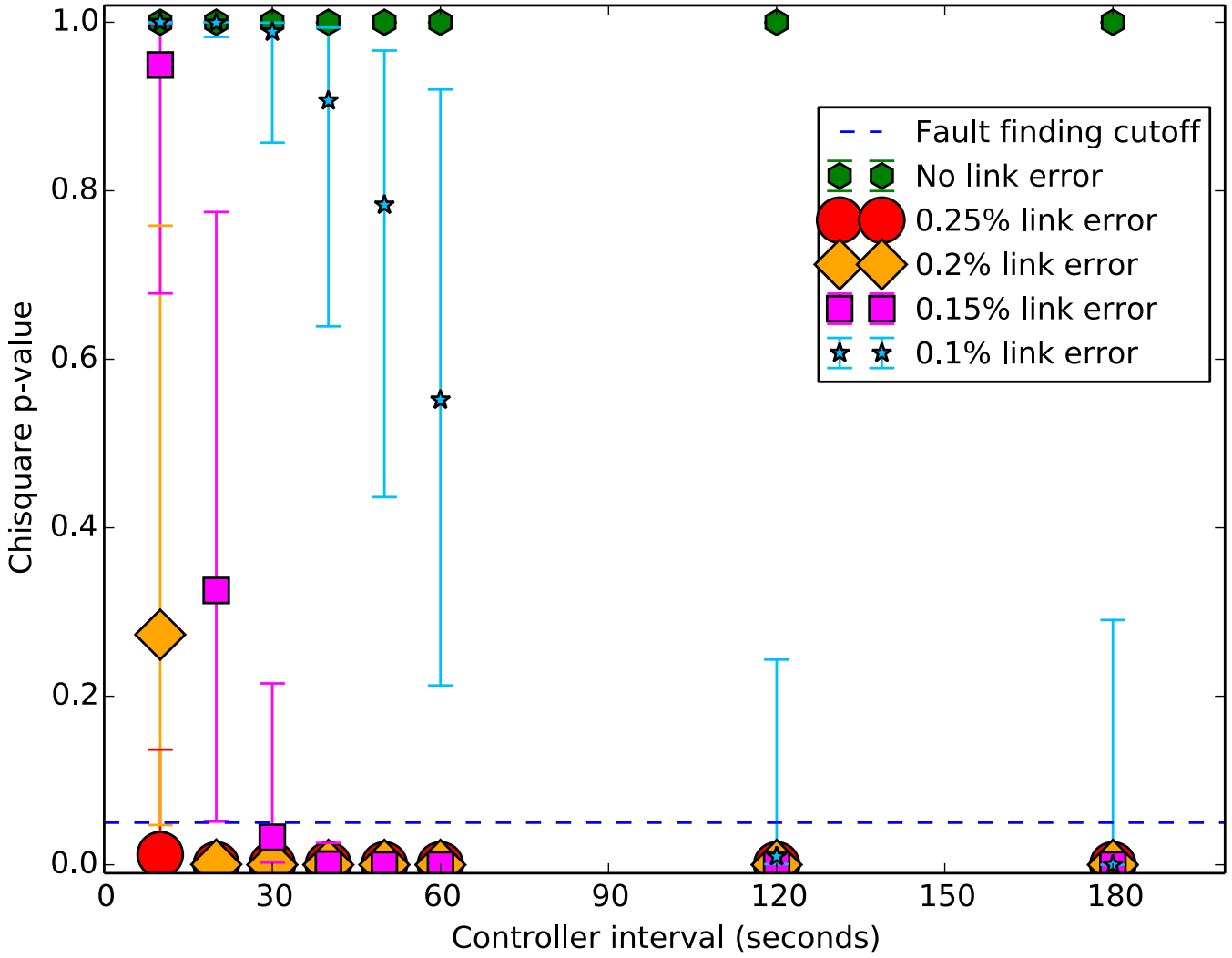
Miniscule faults: choosing between detection speed and sensitivity



Miniscule faults: choosing between detection speed and sensitivity

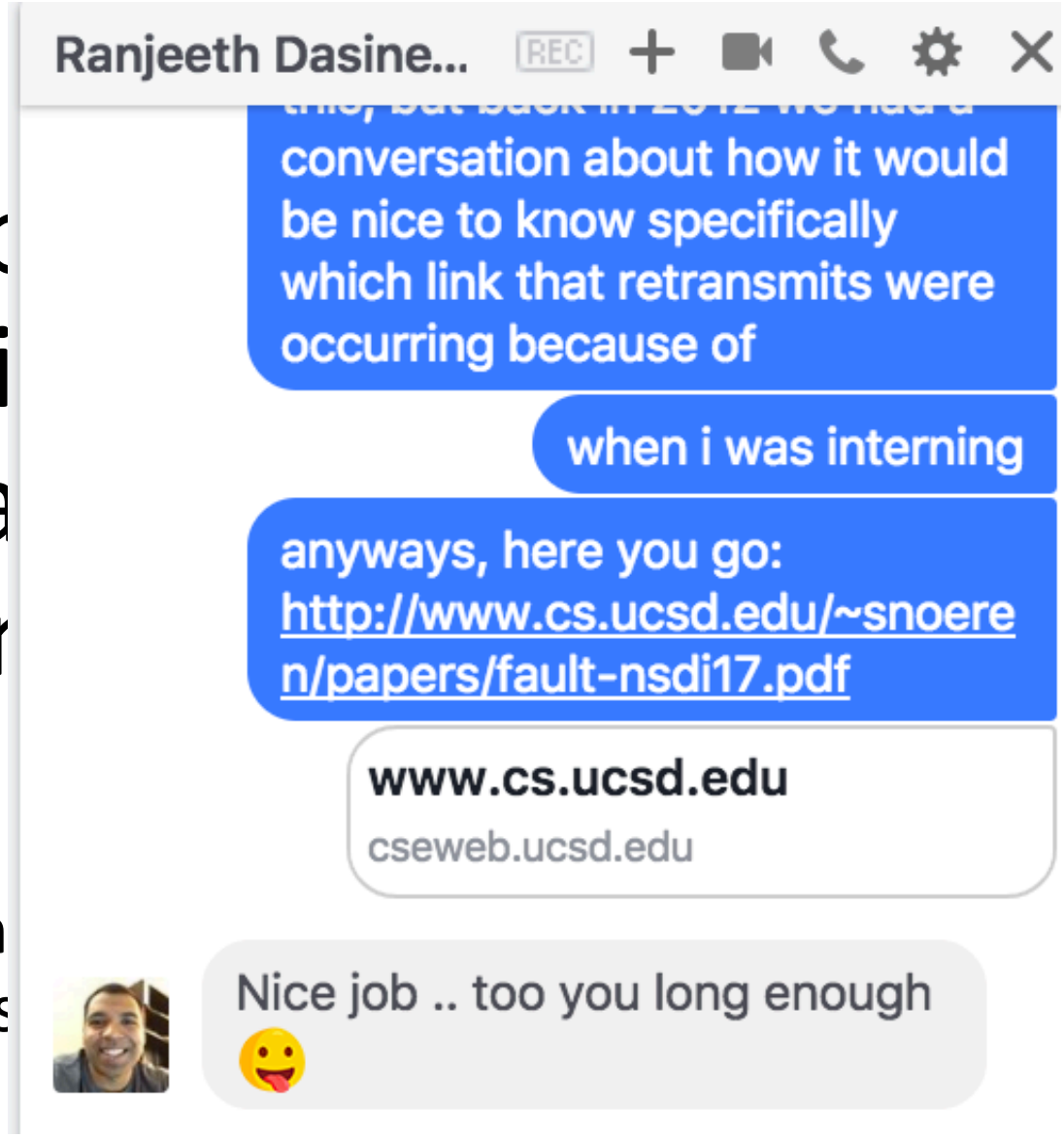


Miniscule faults: choosing between detection speed and sensitivity



“It would b
if we could fi
which link wa
these retrar

Ranjeeth Dasinen
(paraphras



cns

center for networked systems

Interpreting the T-Test

1. **T-Statistic:** “Does this link have more or less retransmits than average?”

   - **Positive** T-statistic means larger than average.
   - **Negative** T-statistic means smaller than average.

2. **P-Value:** “Is the difference in mean big enough to concern us?”

   - **Close to 0** means this link could be an outlier.
   - **Close to 1** means we are not concerned.

Interpreting the T-Test