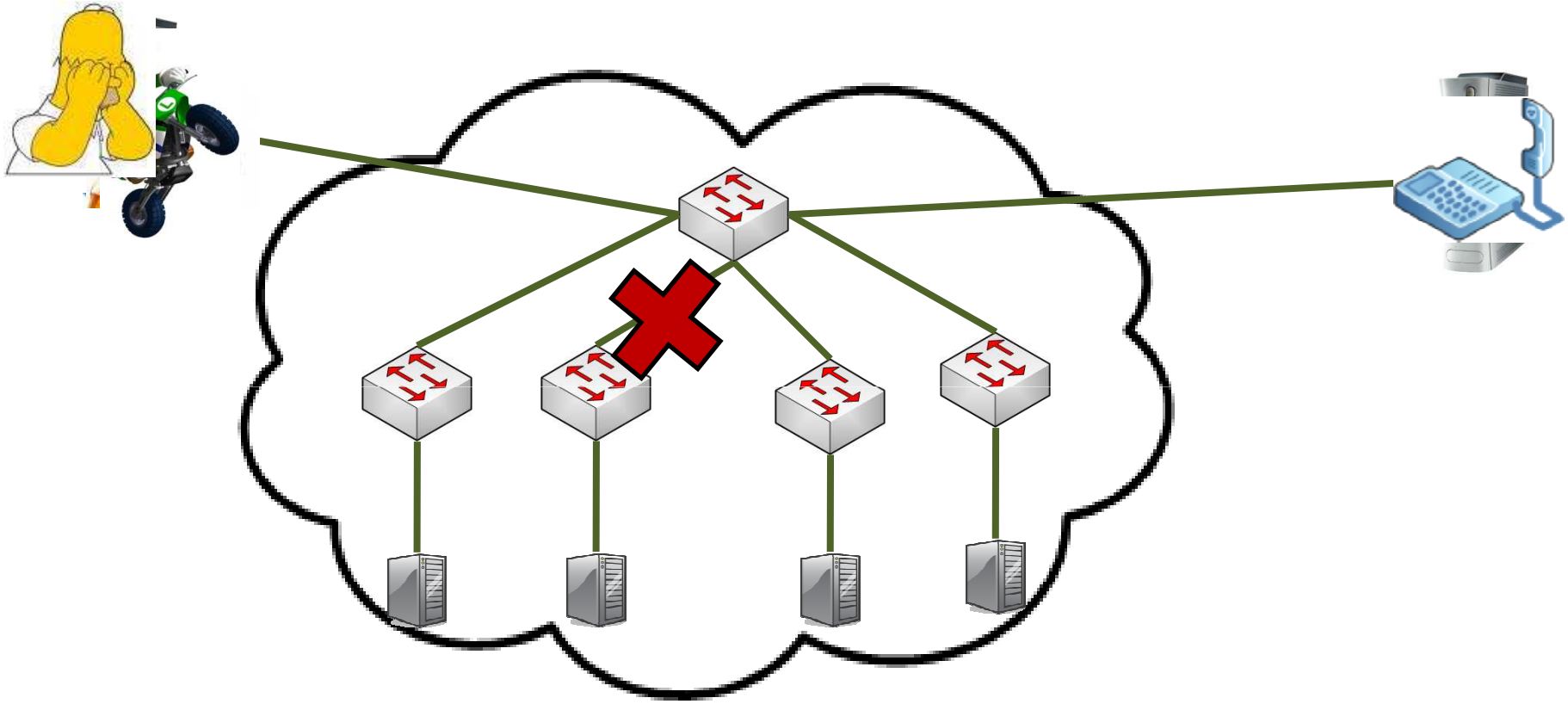
A Case for Fine Grained Traffic Engineering in Data Centers

Theophilus Benson*, Ashok Anand*, Aditya Akella*, Ming Zhang+

*University of Wisconsin, Madison

+ Microsoft Research

Why are Data Centers Important?



- IM: low B/W, loose latency
- Multimedia: low B/W, strict latency
- Games: high B/W, strict latency

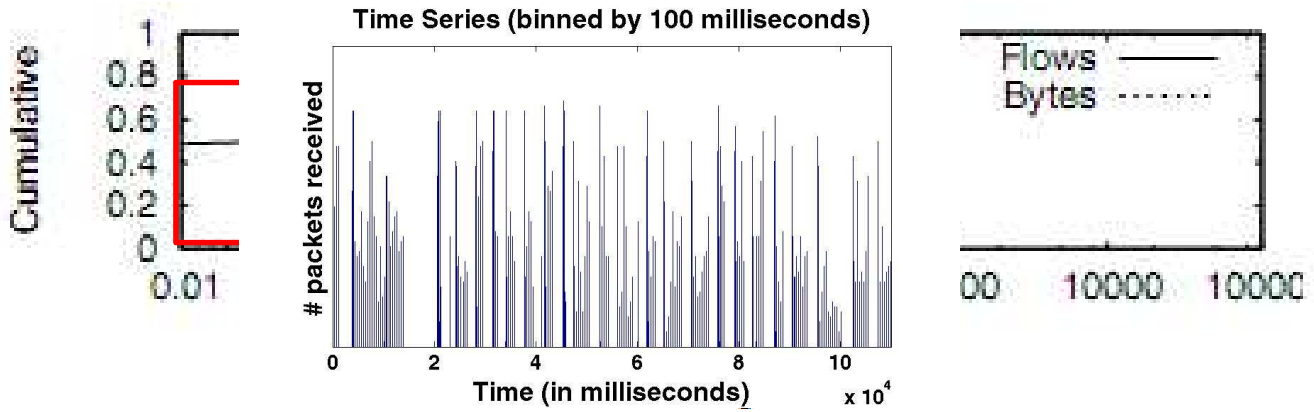
Outline

- Background
- **Traffic Engineering in data centers**
- Design goals for ideal TE
- MicroTE
- Conclusion

Options for TE in Data Centers?

- Current supported techniques
  - Equal Cost MultiPath (ECMP)
  - Spanning Tree Protocol (STP)
- Proposed (ECMP based)
  - Fat-Tree, VL2
- Other existing
  - TEXCP, COPE,…, OSPF link tuning

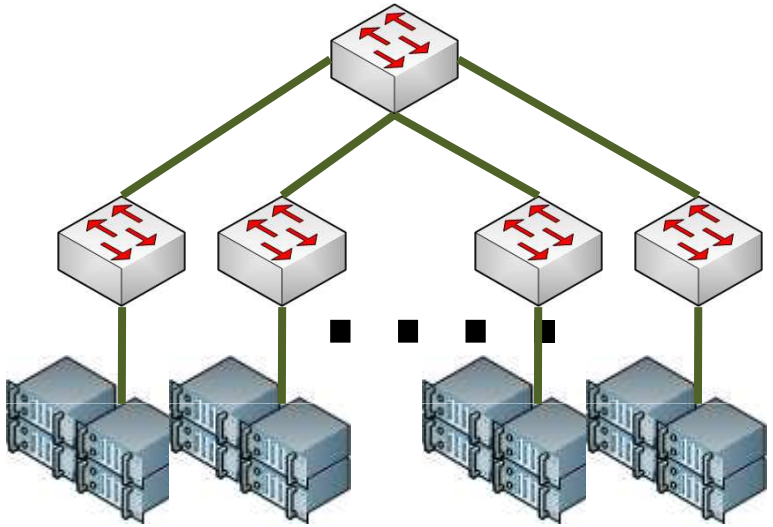
Properties of Data Center Traffic



- Flows are small and short-lived [Kandula et. al, 2009]
- Traffic is bursty [Benson et. al, 2009]
- Traffic is unpredictable at 100 secs [Maltz et. al, 2009]

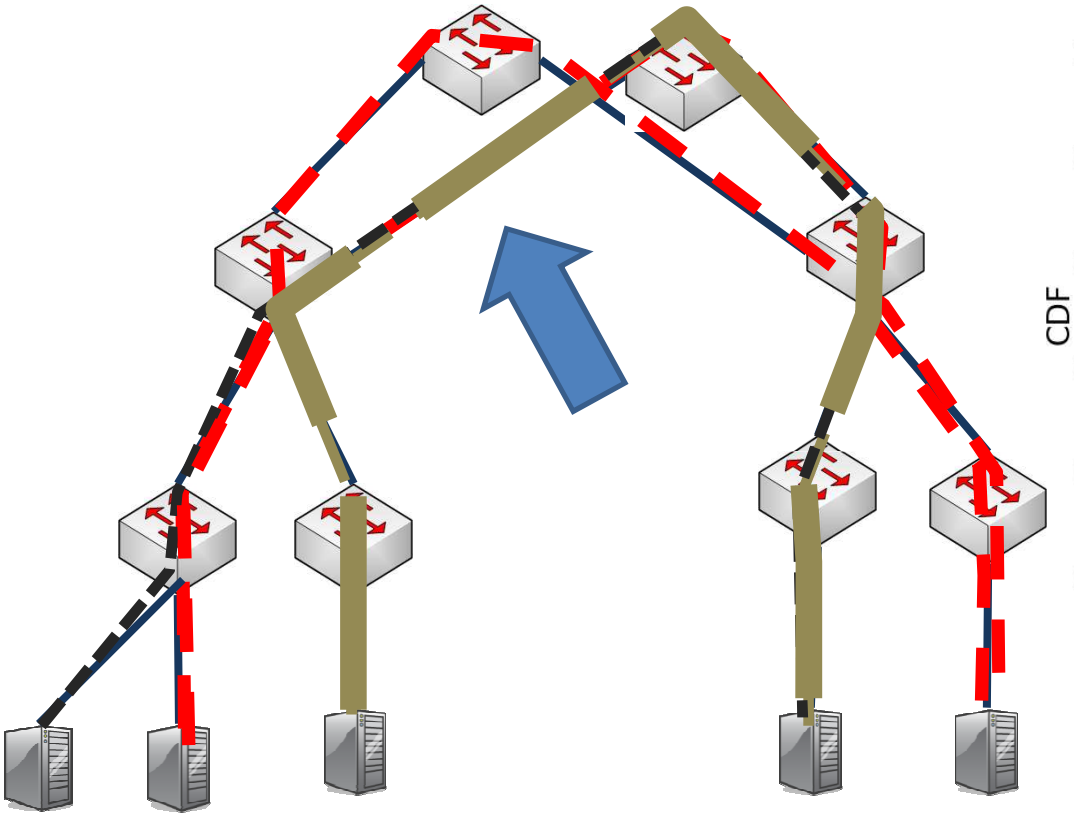
How do we evaluate TE?

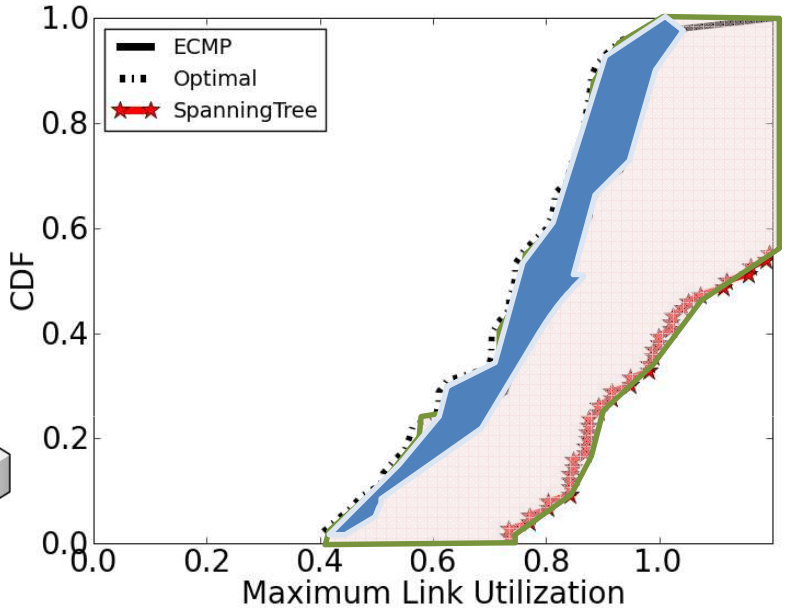
- Data center traces
  - Cloud data center
    - Map-reduce app
    - ~1500 servers,
    - ~80 switches
    - 1 sec snapshots for 24 hours



- Simulator
  - Input:
    - Traffic matrix, Topology ,Traffic Engineering
  - Output:
    - link utilization

Draw Backs of Existing TE

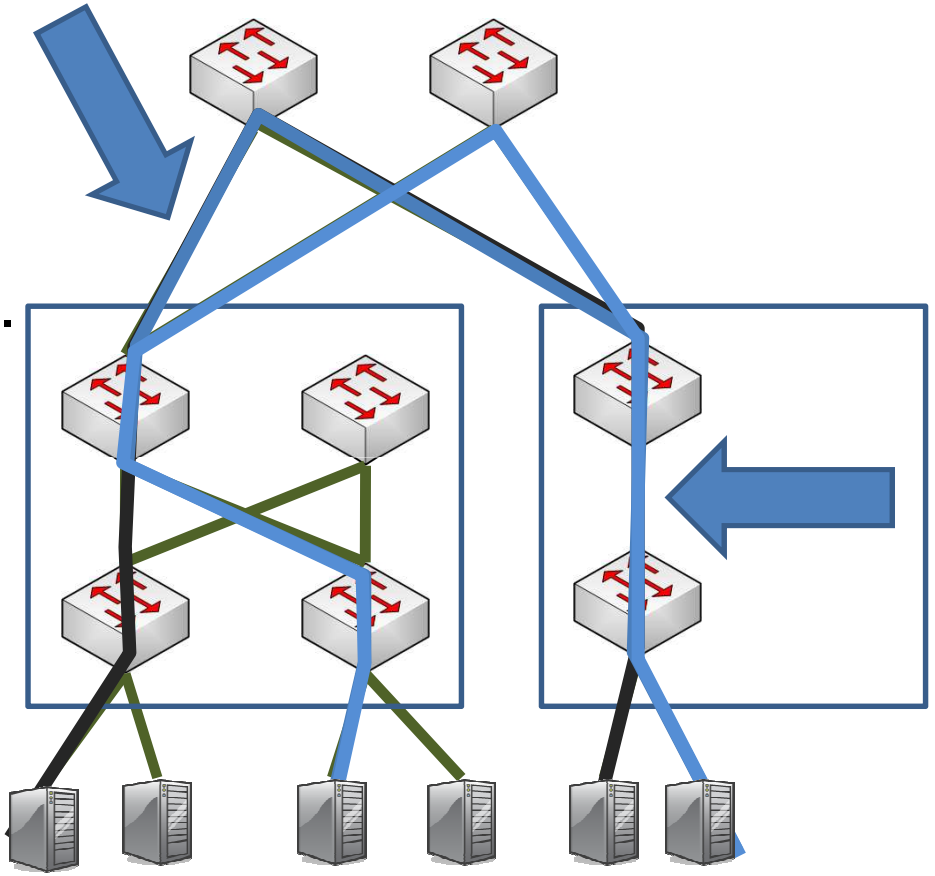




- STP does not use multiple path
- ECMP does not adapt to burstiness

Draw Backs of Proposed TE

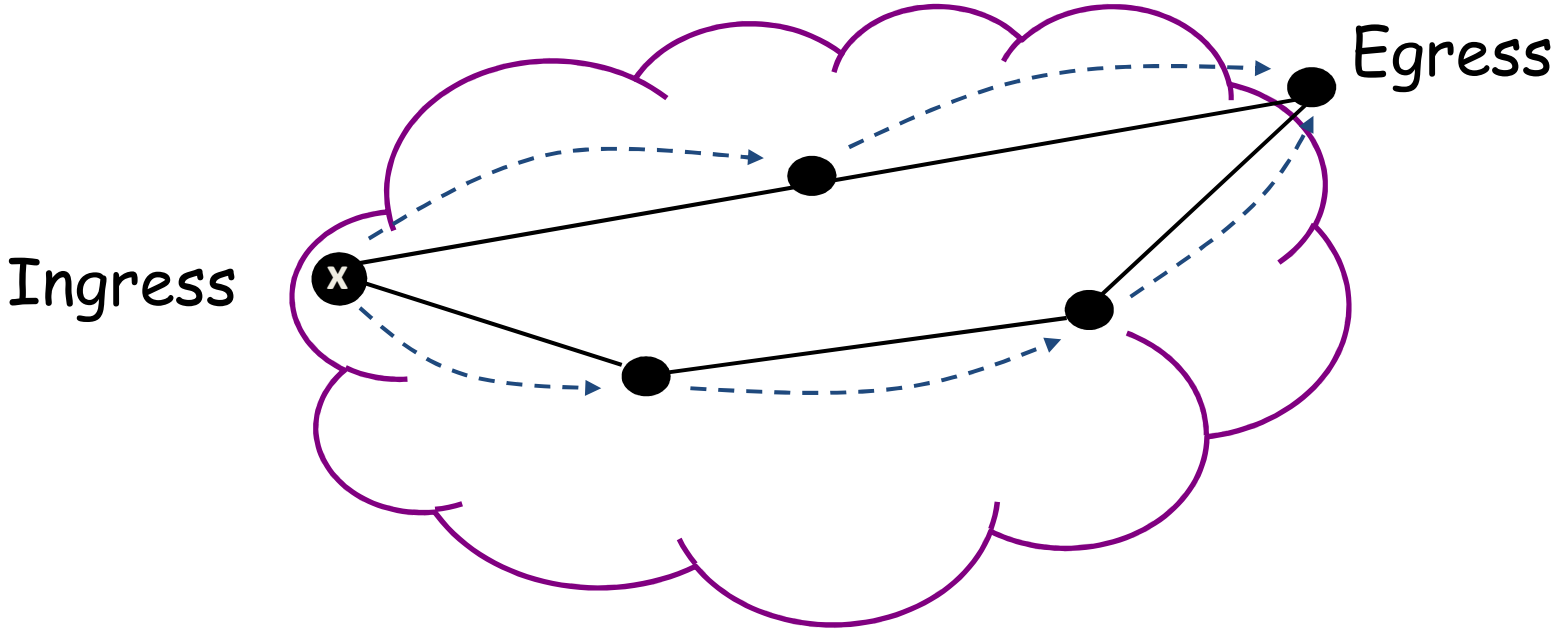
- Fat-Tree
  - Rehash flows
  - Local opt. != global opt.
- VL2
  - Coarse grained flow assignment



- VL2 & Fat-Tree do not adapt to burstiness

Draw Backs of Other Approaches

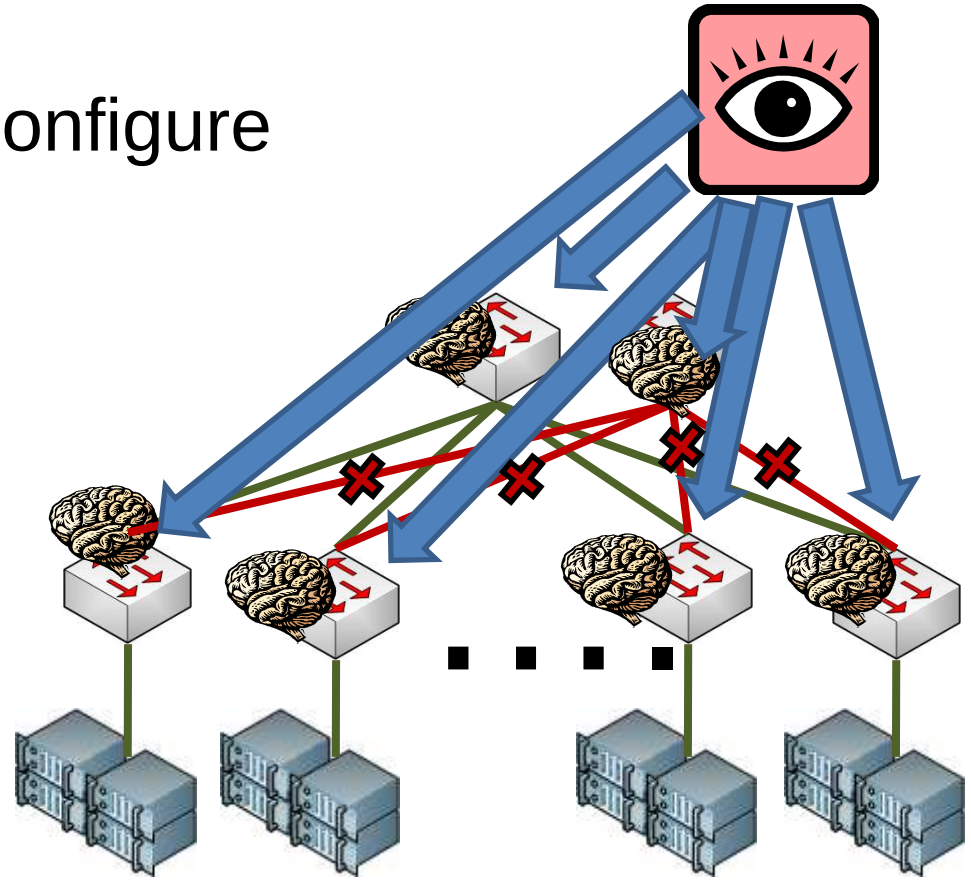
- TEXCP, COPE …. OSPF link tuning



- Unable to react fast enough (below 100 secs)

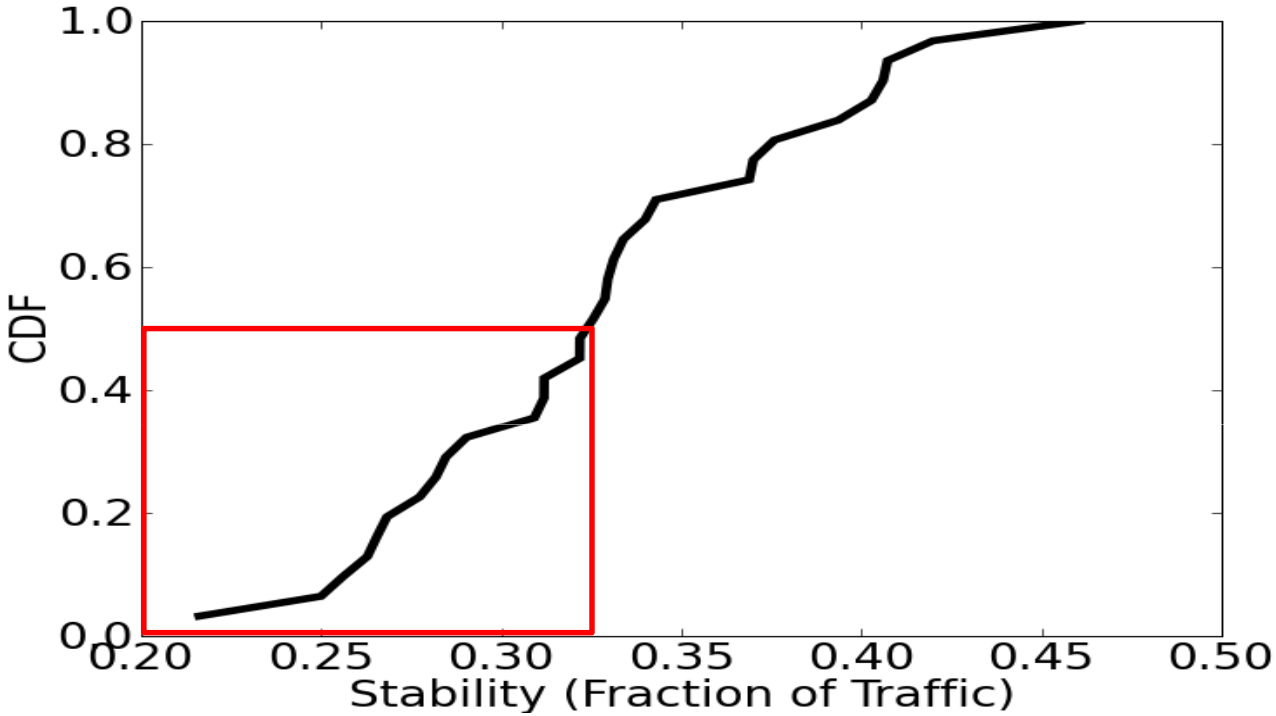
Design Requirements for TE

- Calculate paths & reconfigure network
  - Use all network paths
  - Use global view
  - Must react quickly



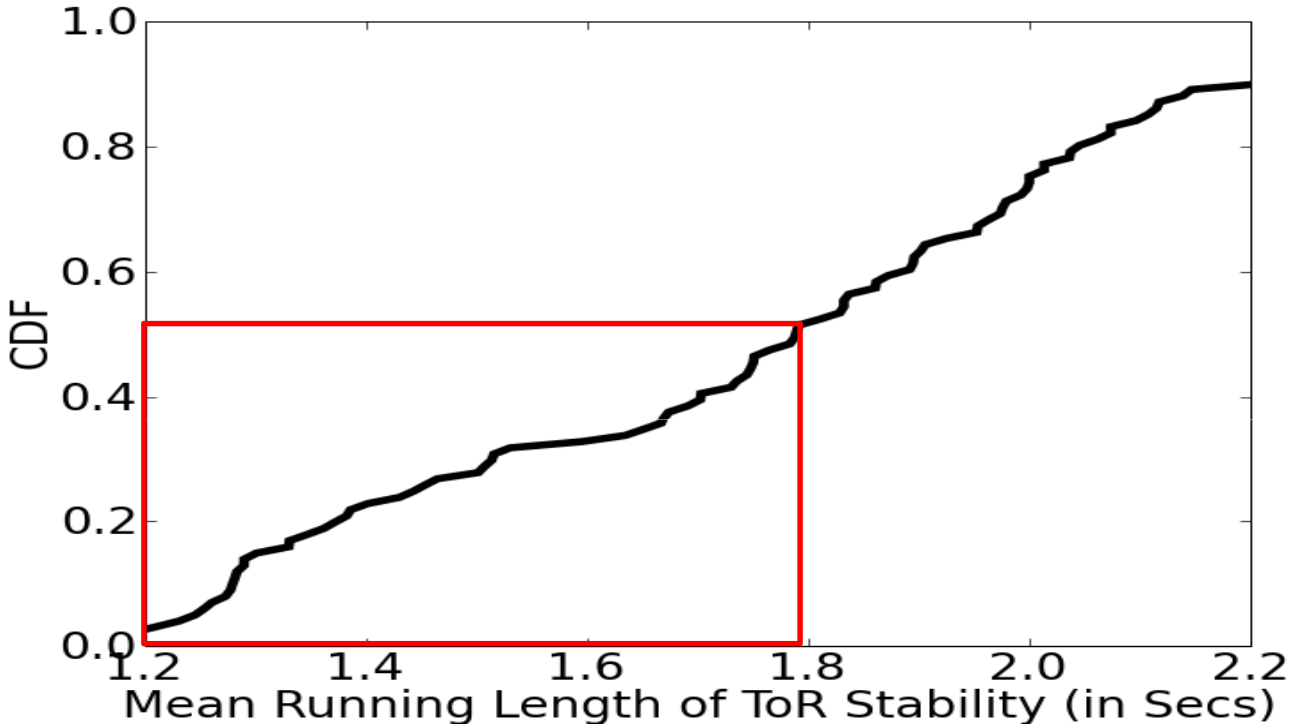
- How predictable is traffic?

Is Data Center Traffic Predictable?



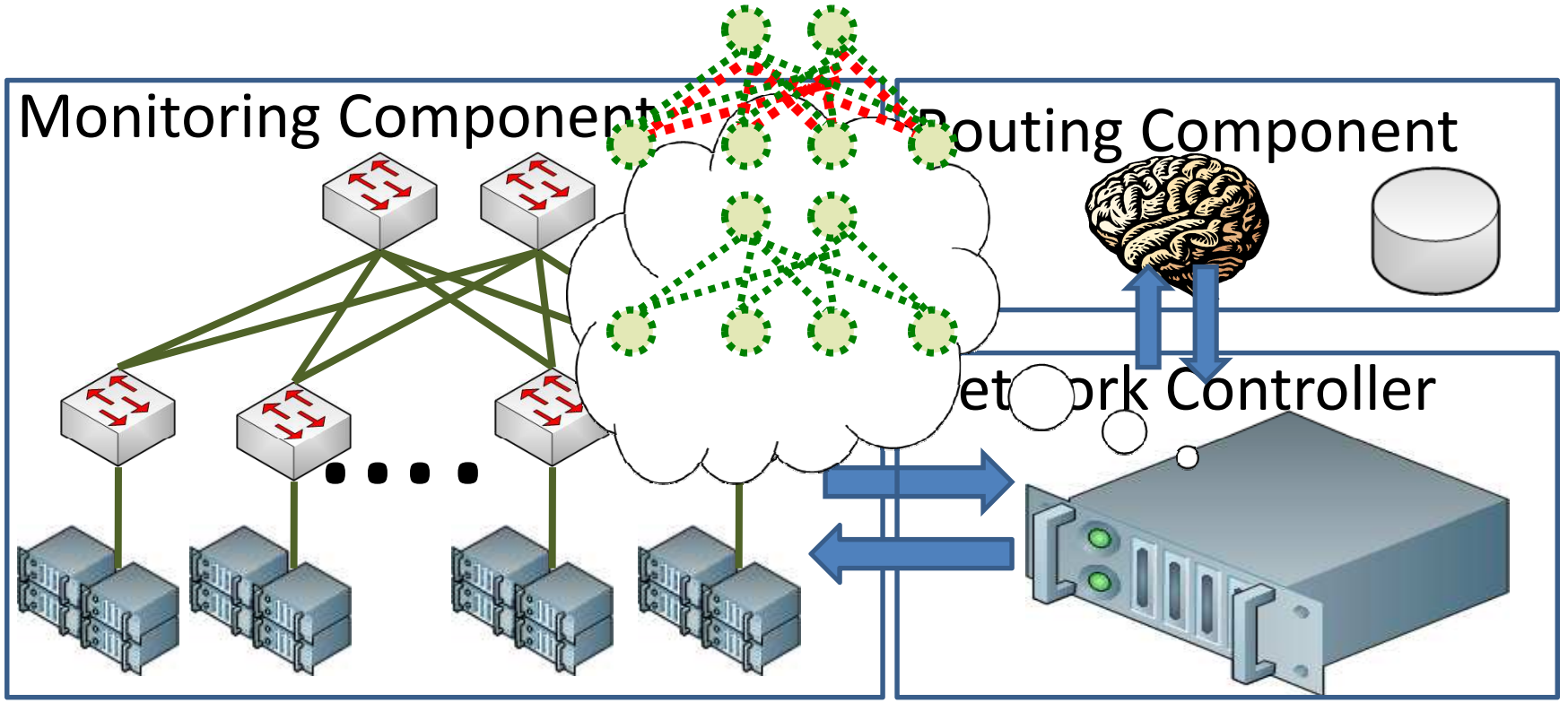
- YES! 33% of traffic is predictable

How Long is Traffic Predictable?



- TE must react in under 2 seconds

MicroTE: Architecture

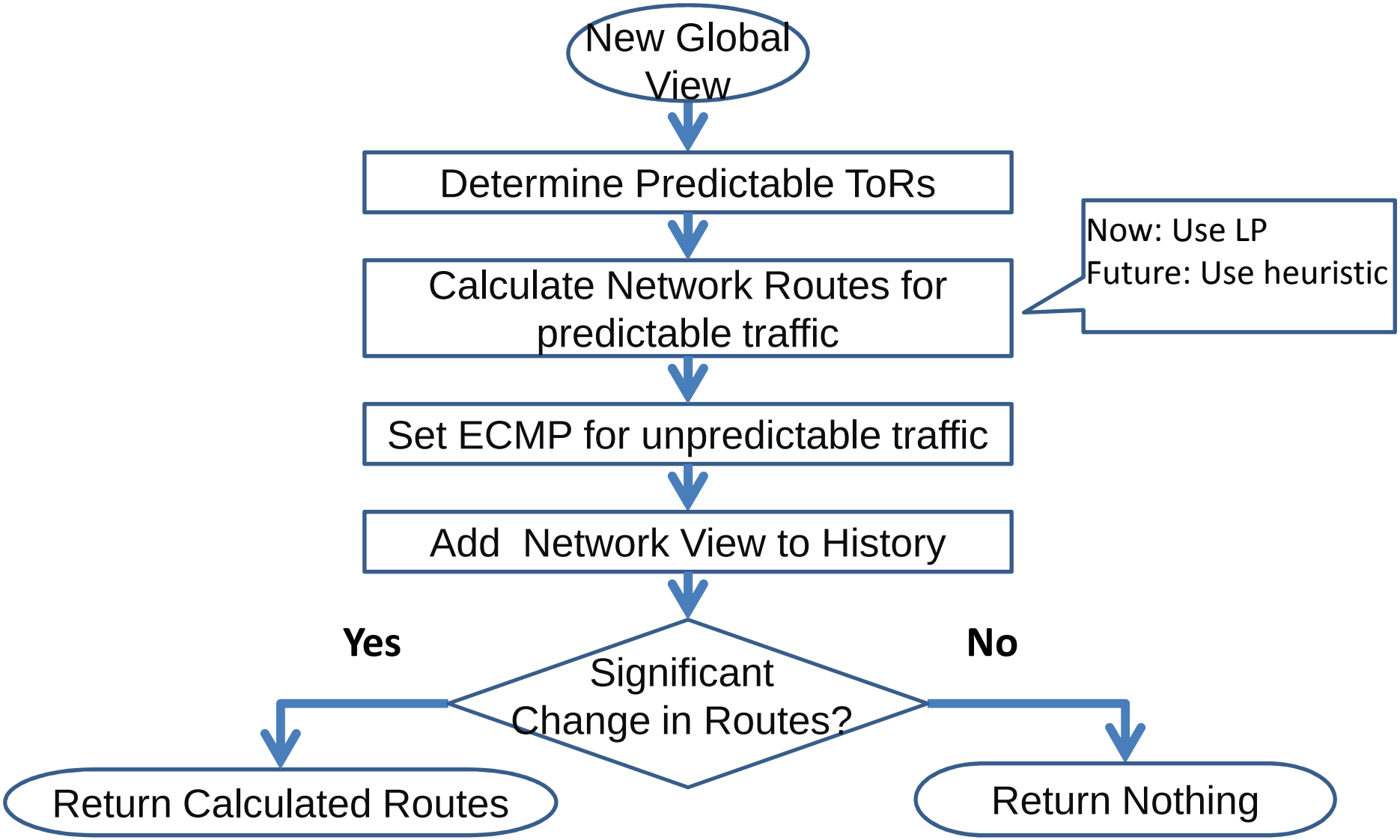


- Based on OpenFlow framework
- Global view:
  - created by network controller
- React to predictable traffic:
  - routing component tracks demand history
- All N/W paths:
  - routing component creates routes using all paths

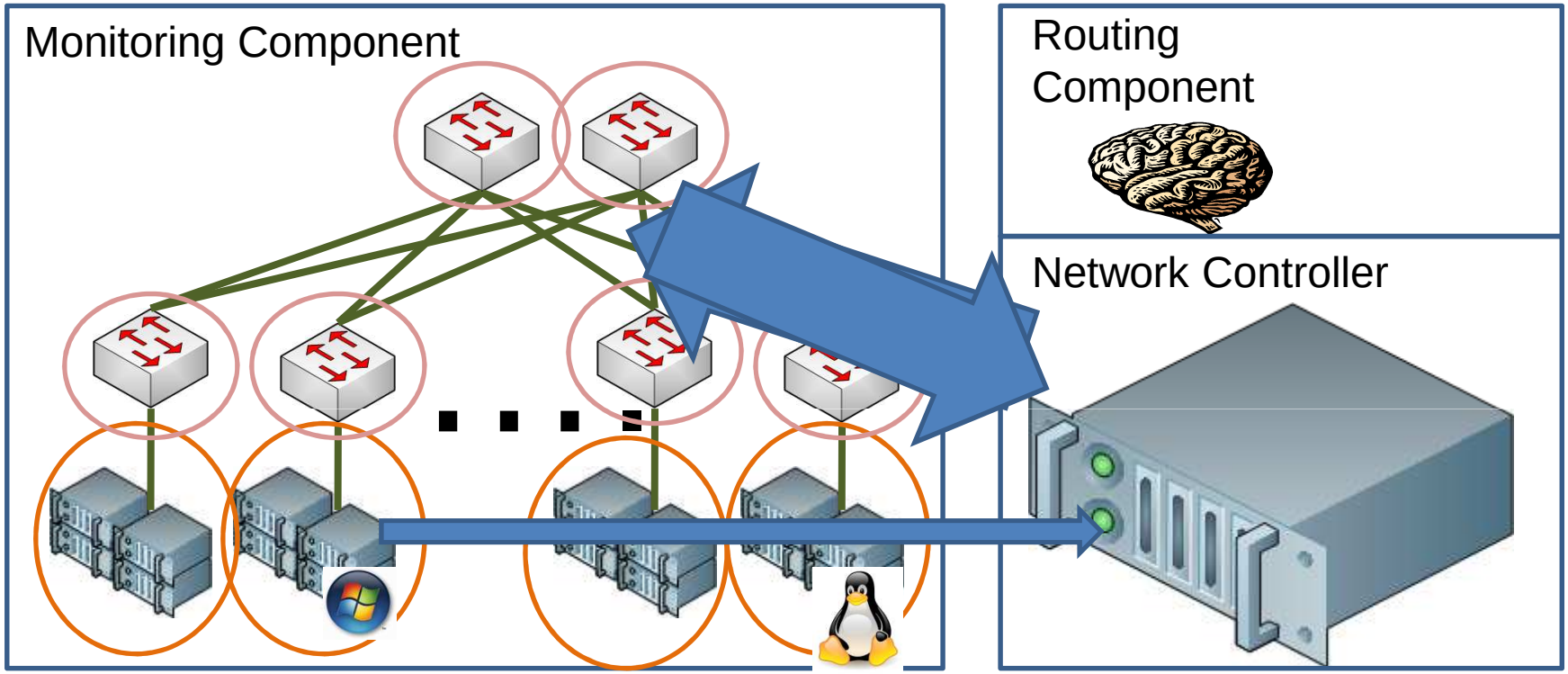
Routing Component

- Step 1: Determine predictable traffic
- Step 2: Route along rarely utilized paths
  - Currently use LP
  - Faster Algorithm == future work
- Step 3: Set ECMP for other traffic
- Step 4: Return routes

Routing Component



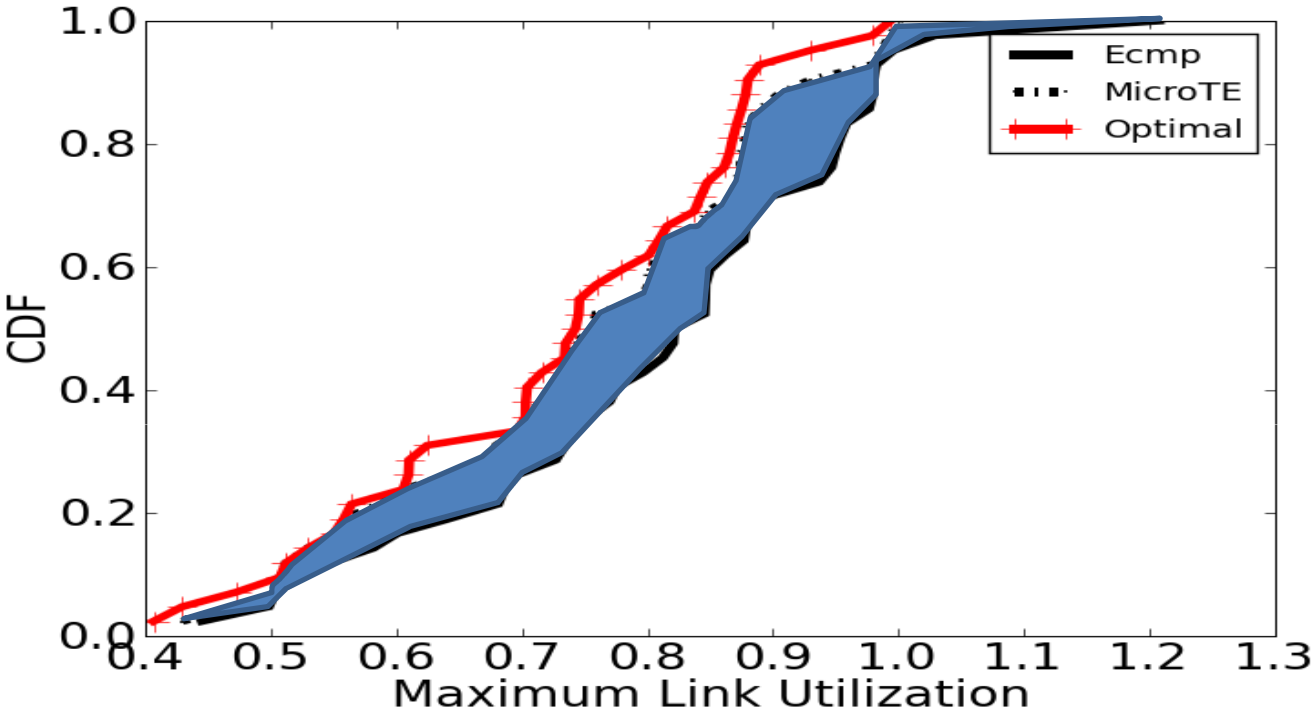
Tradeoffs: Monitoring Component



- Switch based
  - Low complexity
  - High overhead

- End-host based
  - Low overhead
  - High complexity

Preliminary Evaluation



- Outperforms ECMP
- Slightly worse than optimal

Conclusion

- Study existing TE
  - Found them lacking (15-20%)
- Study data center traffic
  - Discovered traffic predictability (33% for 2 secs)
- Guidelines for ideal TE
- MicroTE
  - Implementation of ideal TE
  - Preliminary evaluation

Thank You

- Questions?