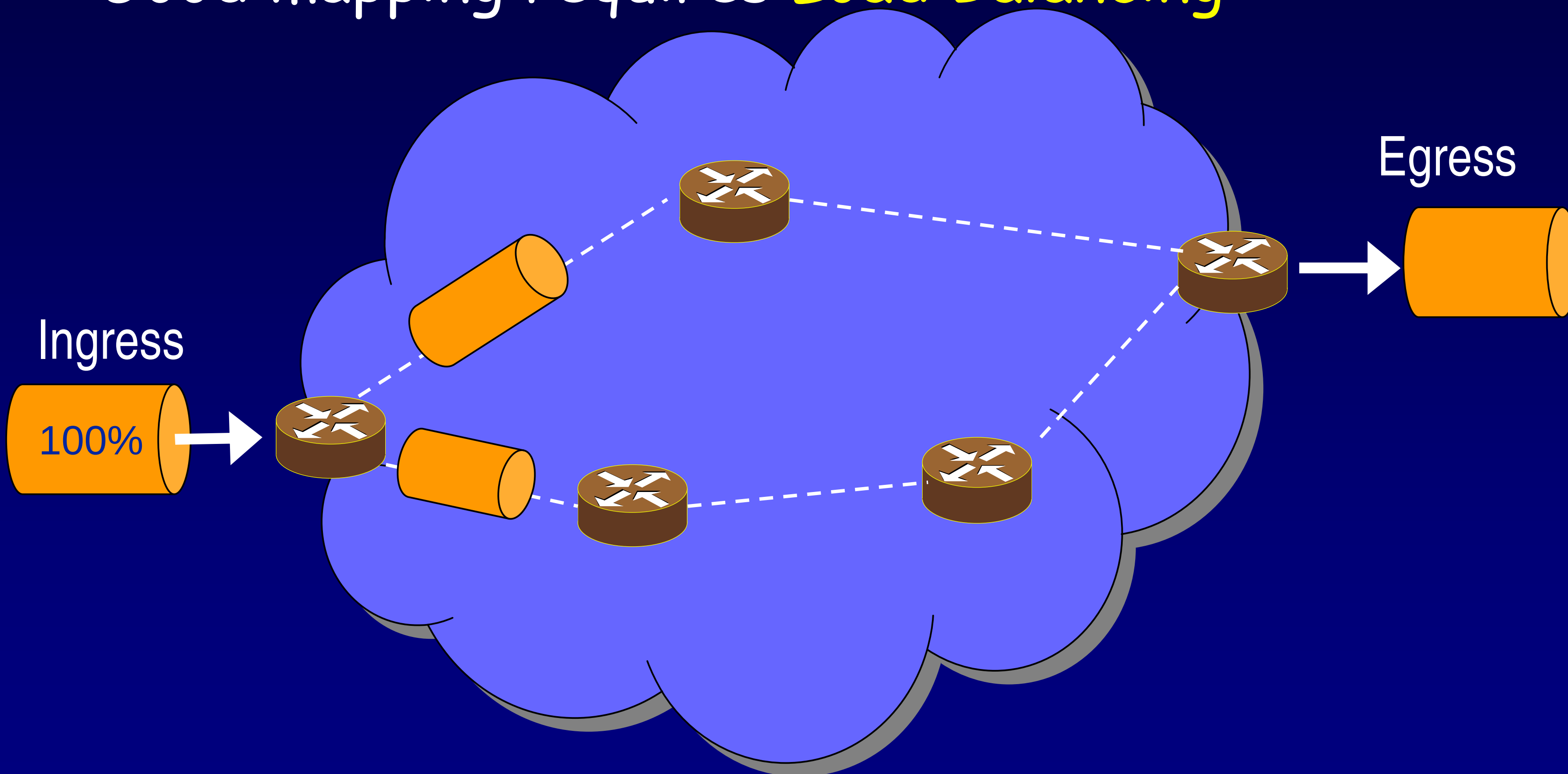
Responsive Yet Stable Traffic Engineering

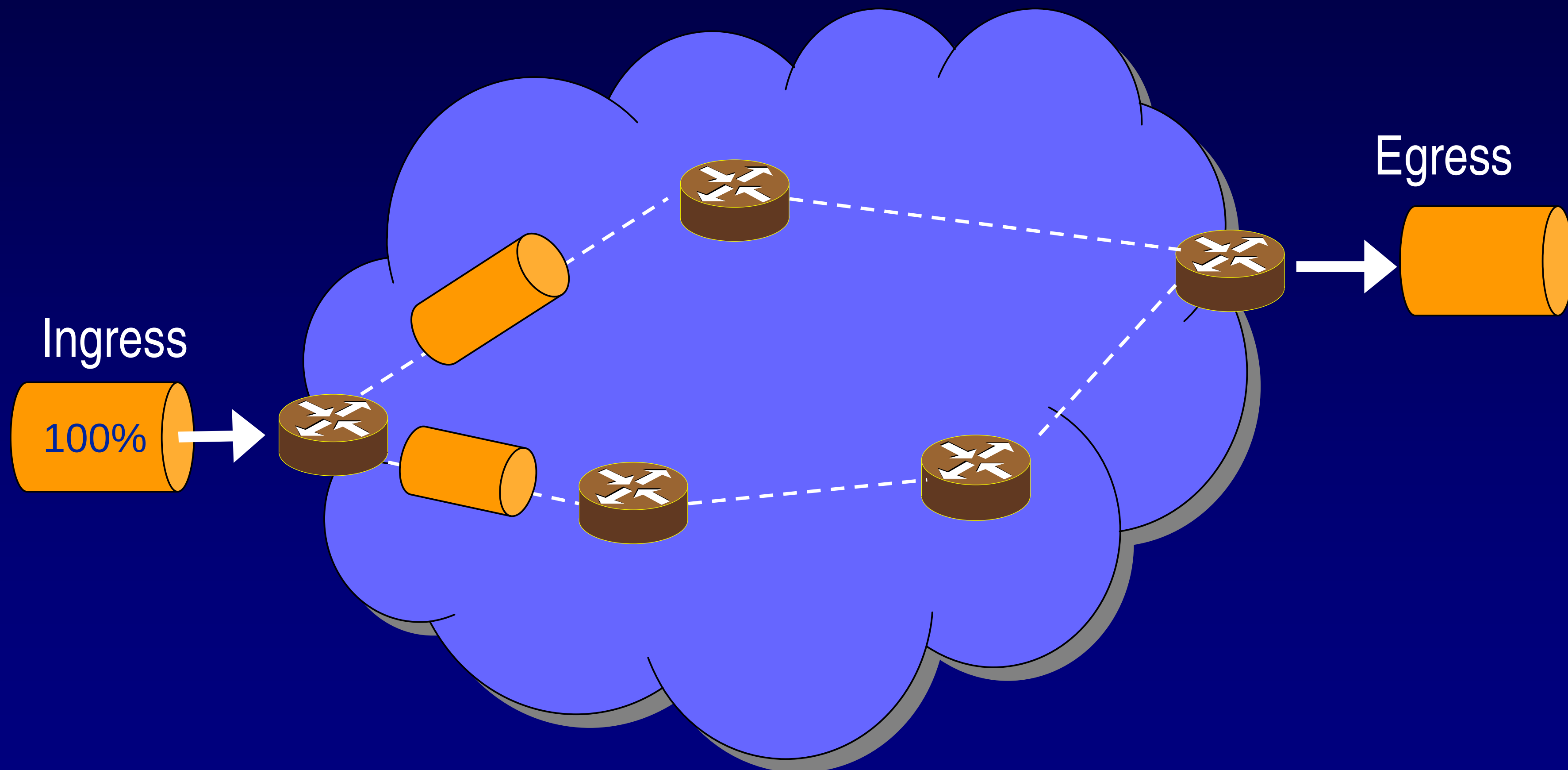
Srikanth Kandula

Dina Katabi, Bruce Davie, and Anna Charny

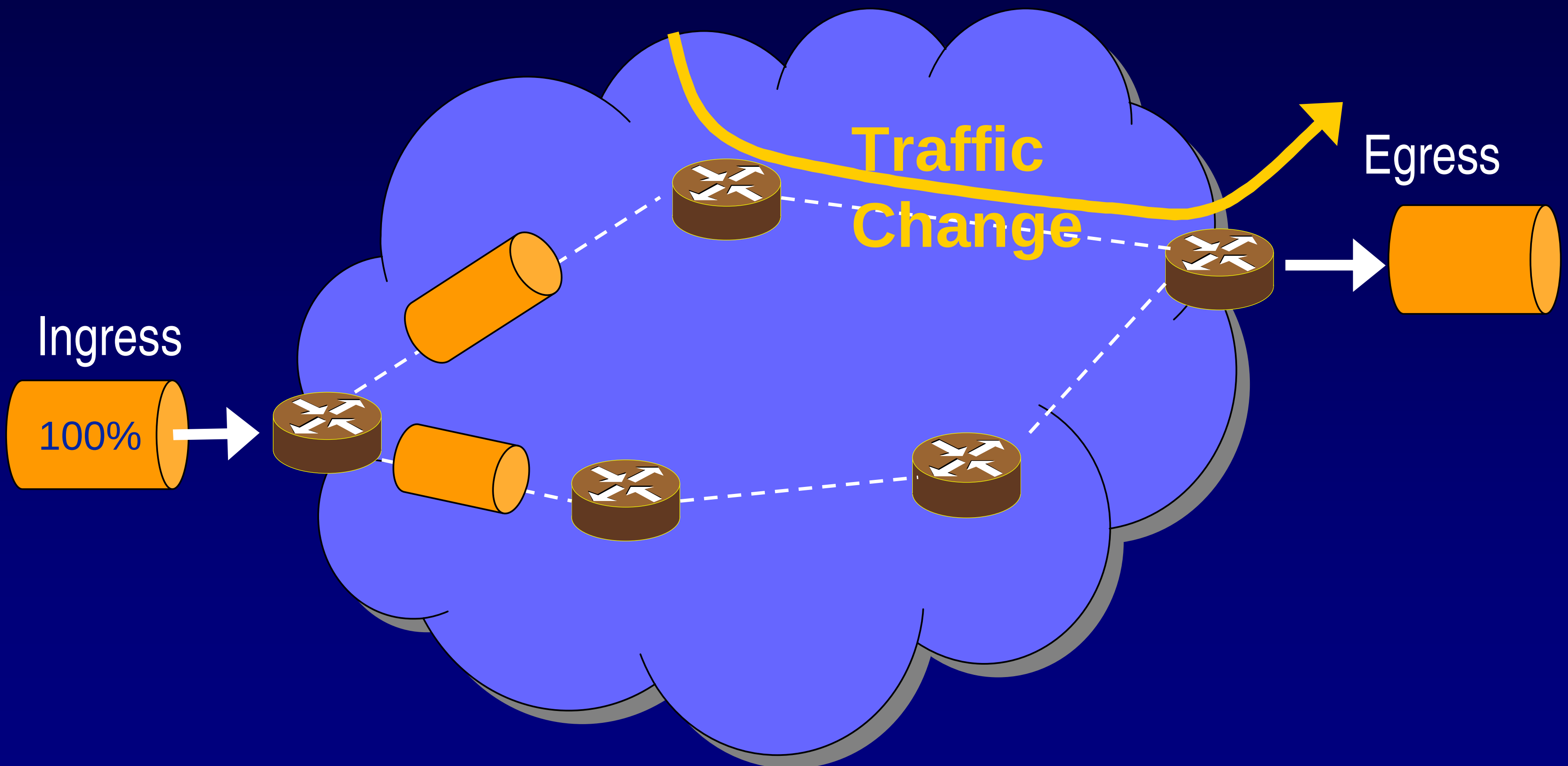
- ISPs needs to map traffic to underlying topology
- Good Mapping → Good Performance & Low Cost
- Good mapping requires Load Balancing



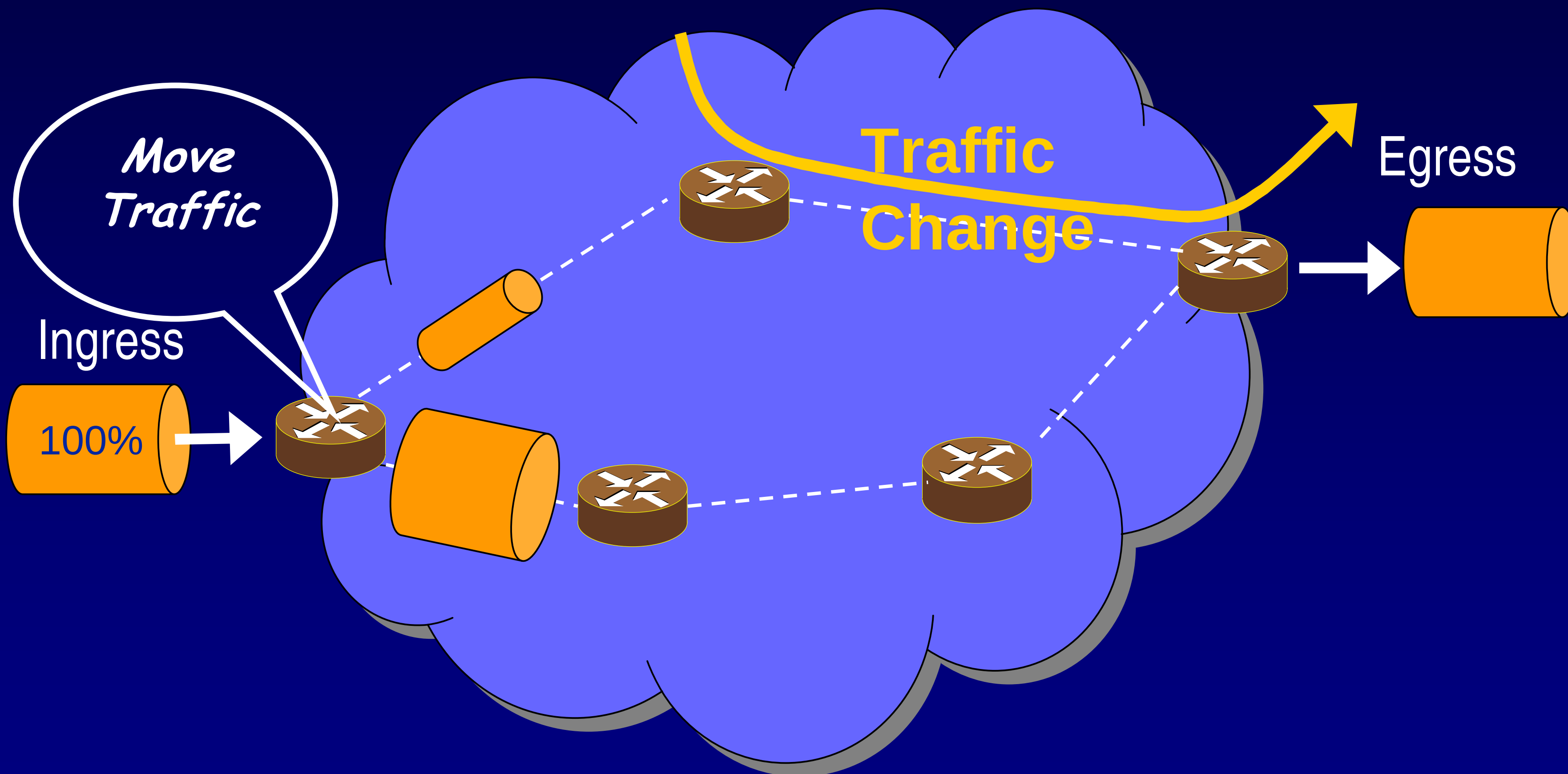
More, ISPs want to re-balance load when an unexpected event causes congestion



More, ISPs want to re-balance load when an unexpected event causes congestion → failure, BGP reroute, flash crowd, or attack

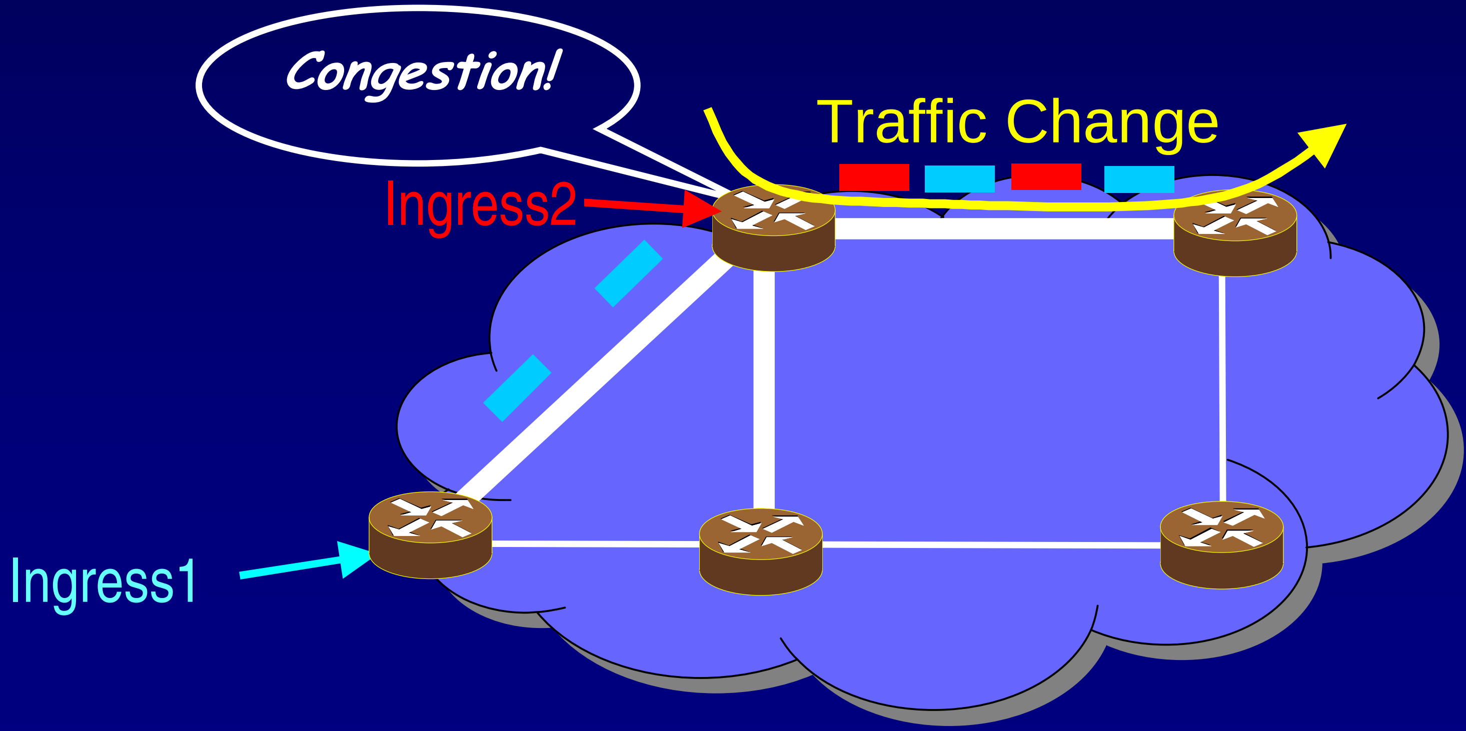


More, ISPs want to re-balance load when an unexpected event causes congestion → failure, BGP reroute, flash crowd, or attack



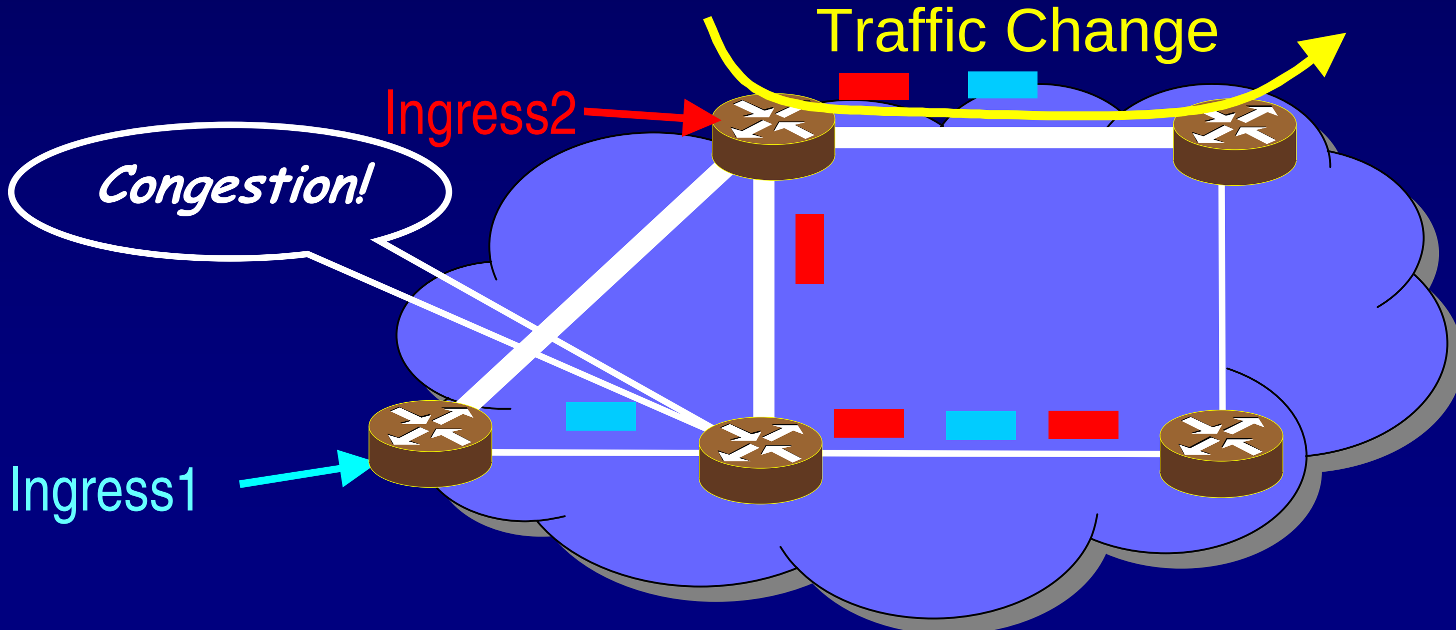
But, rebalancing load in realtime is risky

- Need to rebalance load ASAP
  - Remove congestion before it affects user's performance
- But, moving quickly → may overshoot → congestion on a different path → more drops …



But, rebalancing load in realtime is risky

- Need to rebalance load ASAP
  - Remove congestion before it affects user's performance
- But, moving quickly → may overshoot → congestion on a different path → more drops …



But, rebalancing load in realtime is risky

- Need to rebalance load ASAP
  - Remove congestion before it affects user's performance
- But, moving quickly → may overshoot → congestion on a different path → more drops …

Problem: How to make Traffic Engineering:

- Responsive: reacts ASAP
- Stable: converges to balanced load without overshooting or generating new congestion

Current Approaches

Offline TE *(e.g., OSPF-TE)*

- Avoids the risk of instability caused by realtime adaptation, but also misses the benefits
- Balances the load in steady state
- Deal with failures and change in demands by computing routes that work under most conditions
- Overprovision for unanticipated events

Online TE *(e.g., MATE)*

- Try to adapt to unanticipated events
- But, can overshoot causing drops and instability

This Talk

- TeXCP: Responsive & Stable Online TE
- Idea:
  - Use adaptive load balancing
  - But add explicit-feedback congestion control to prevent overshoot and drops
- TeXCP keeps utilization always within a few percent of optimal
- Compare to MATE and OSPF-TE, showing that TeXCP outperforms both

Typical Formalization of the TE Problem

Find a routing that:

Min Max-Utilization

- Removes hot spots and balances load
- High Max-Utilization is an indicator that the ISP should upgrade its infrastructure

Online TE involves solving 2 sub-problems

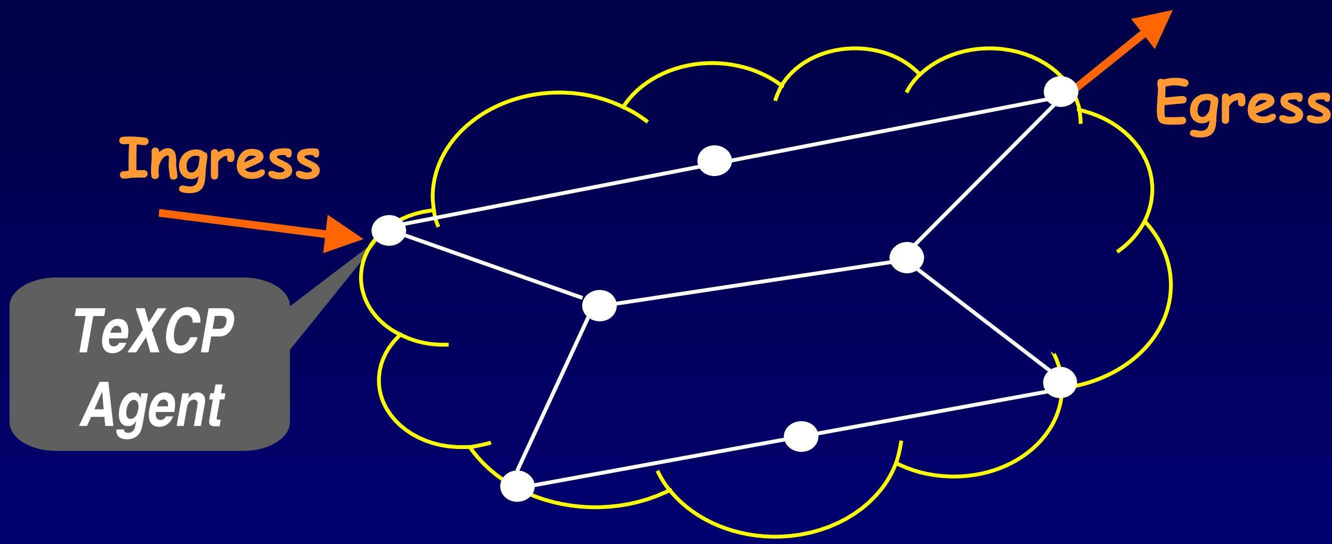
1. Find the traffic split that minimizes the Max-Utilization
2. Converge to the balanced traffic splits in a stable manner

Also, an implementation mechanism

to force traffic to follow the desired splits

Implementation: Force traffic along the right paths

Solution:

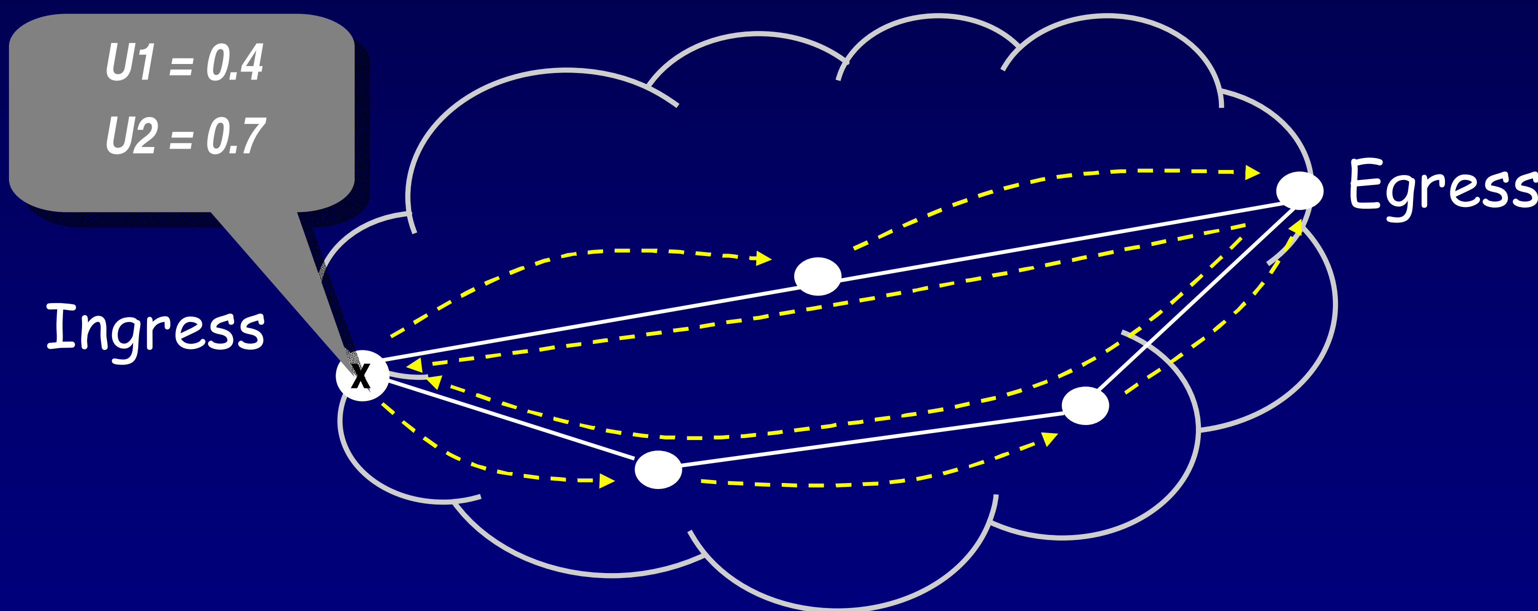


- A TeXCP agent per IE, at ingress node
- ISP configures each TeXCP agent with paths between IE
- Paths are pinned (e.g., MPLS tunnels)

Sub-Problem: Distributedly, TeXCP agents find balanced traffic splits

Solution: TeXCP Load Balancer

- Periodically, TeXCP agent probes a path for its utilization



Probes follow the slow path like ICMP messages

Sub-Problem: Distributedly, TeXCP agents find balanced traffic splits

Solution: TeXCP Load Balancer

- Periodically, TeXCP agent probes a path for its utilization
- A TeXCP agent iteratively moves traffic from over-utilized paths to under-utilized paths
  - r_p is this agent's traffic on path p

$$\Delta r_p \propto \left(\bar{u}(t) - u_p(t)\right)$$

- Deal with different path capacity
- Deal with inactive paths (r_p =0)

Sub-Problem: Distributedly, TeXCP agents find balanced traffic splits

Solution: TeXCP Load Balancer

- Periodically, TeXCP agent probes a path for its utilization
- A TeXCP agent iteratively moves traffic from over-utilized paths to under-utilized paths
  - r_p is this agent's traffic on path p

$$\Delta r_p \propto r_p(t)\left(\hat{u}(t) - u_p(t)\right)$$

$$\hat{u} = \frac{\sum r_i u_i}{\sum r_i}$$

- Deal with different path capacity
- Deal with inactive paths (r_p =0)

Proof in paper

Sub-Problem: Converge to balanced load in a stable way

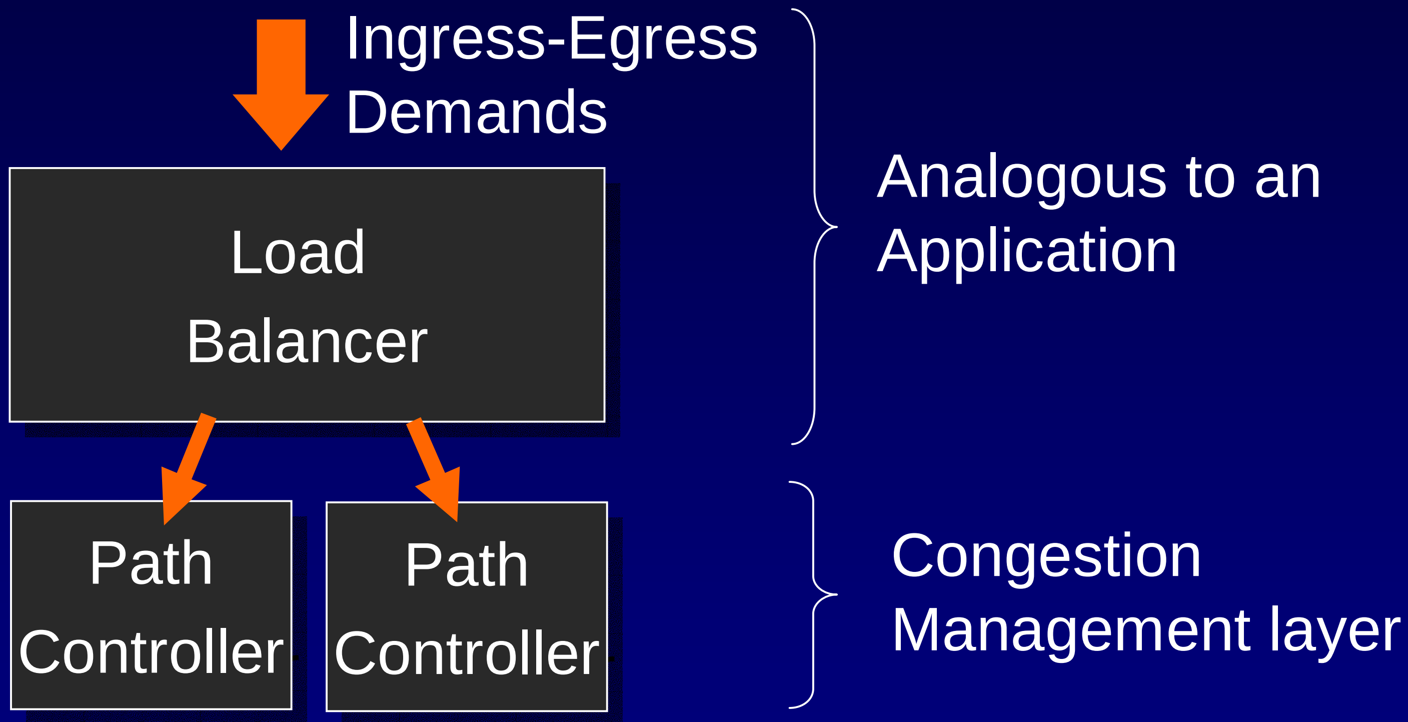
Solution: Use Experience from Congestion Control (XCP)

Congestion Control	Online TE
• Flow from sender to receiver	• Flow from ingress to egress
• Senders share the bottleneck; need coordination to prevent oscillations	• TeXCP agents share physical link; need coordination to prevent oscillations

Move in really small increments → No Overshoot!
Challenge is to move traffic quickly w/o overshoot

Congestion Management Layer between Load Balancer and Data Plane

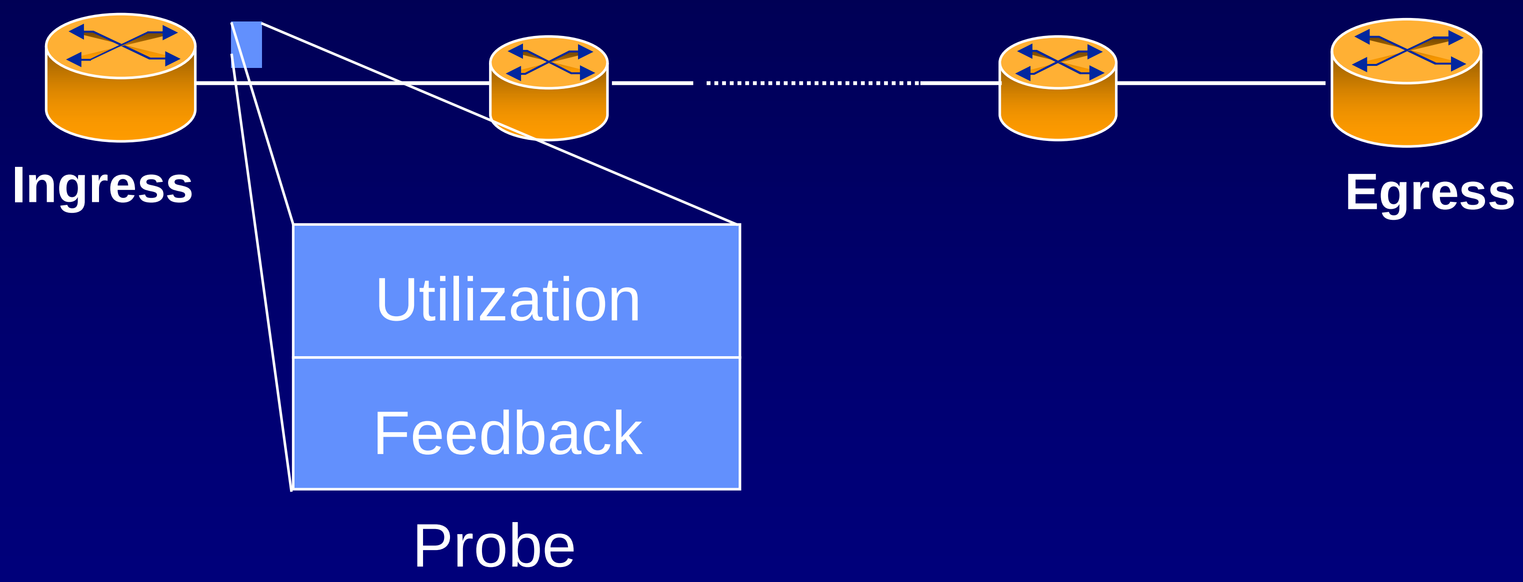
- Set of light-weight per-path congestion controllers



Unlike prior online TE, Load Balancer can push a decision to the data plane *only* as fast as the Congestion Management Layer allows it

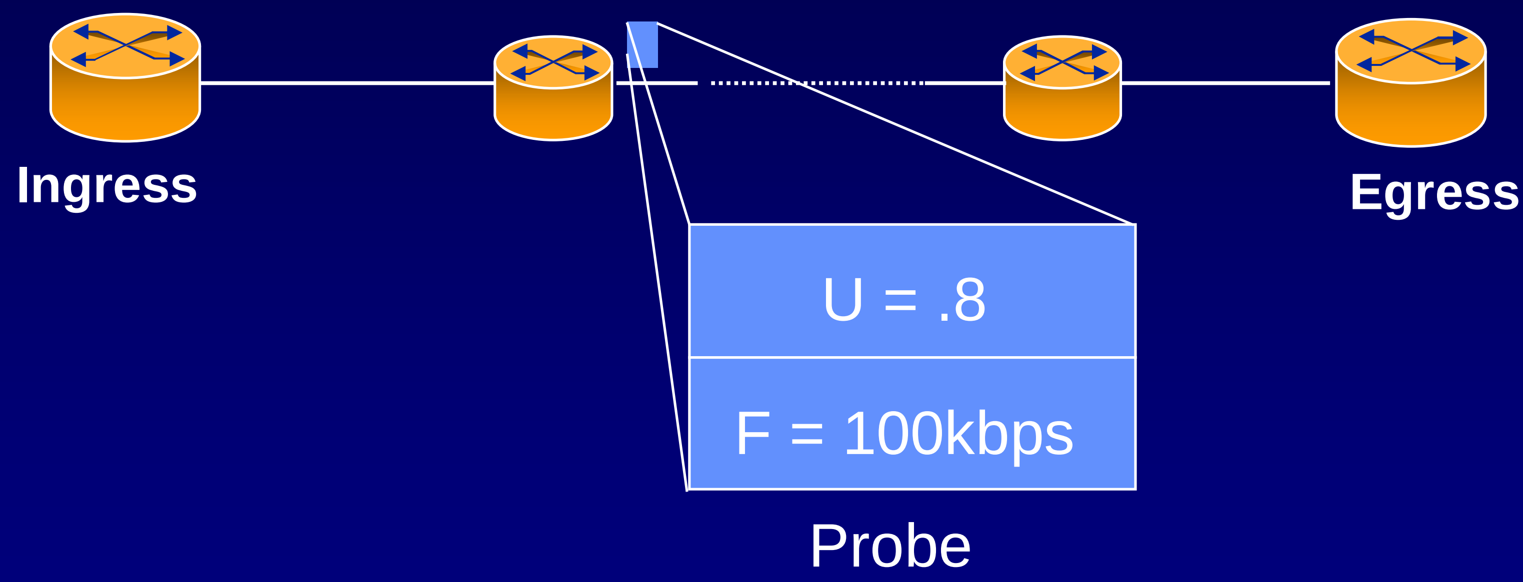
Per-Path Light-Weight Congestion Controller

- Explicit feedback from core routers (like XCP)
- Periodically, collects feedback in ICMP-like probes



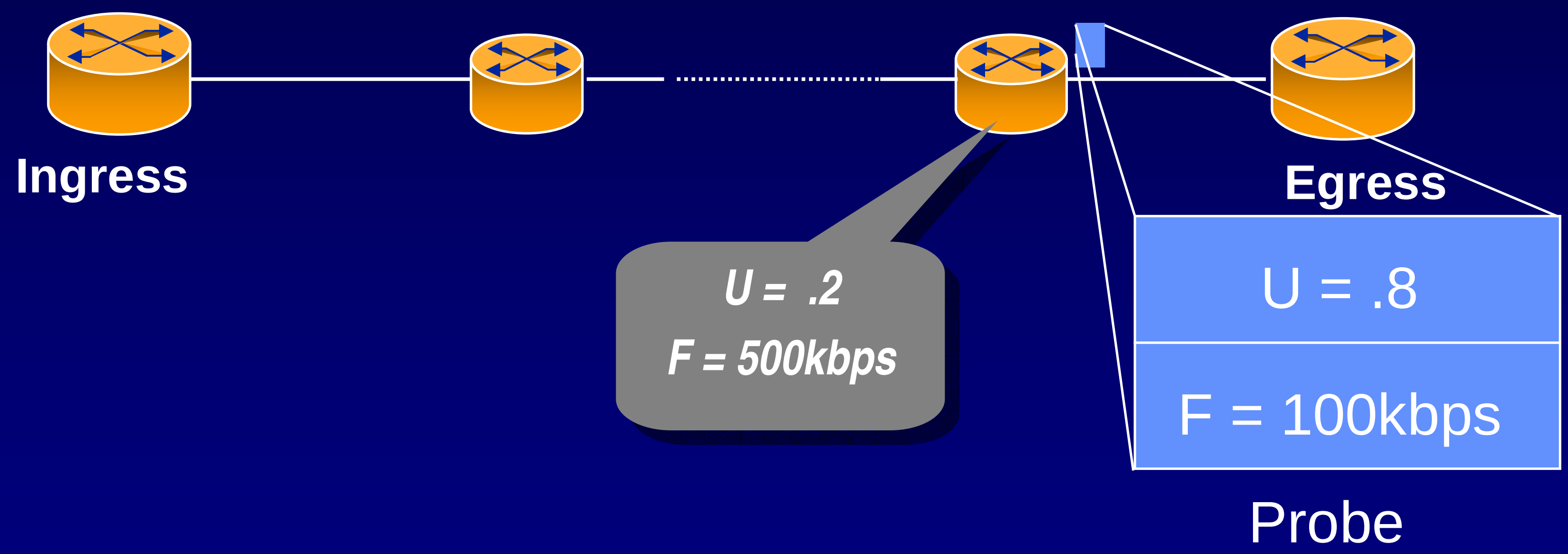
Per-Path Light-Weight Congestion Controller

- Explicit feedback from core routers (like XCP)
- Periodically, collects feedback in ICMP-like probes



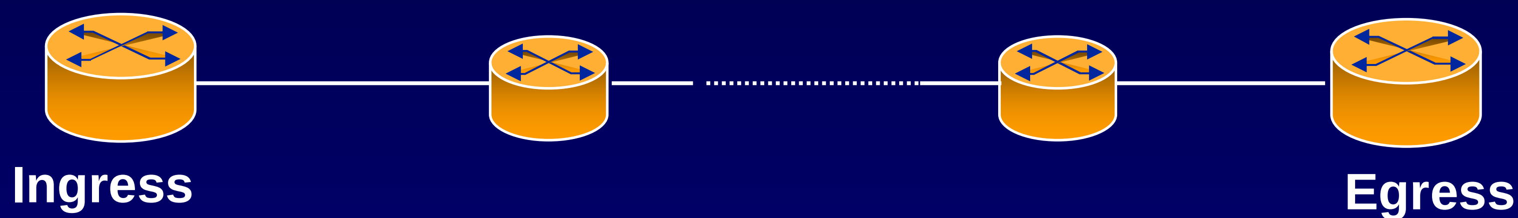
Per-Path Light-Weight Congestion Controller

- Explicit feedback from core routers (like XCP)
- Periodically, collects feedback in ICMP-like probes



Per-Path Light-Weight Congestion Controller

- Explicit feedback from core routers (like XCP)
- Periodically, collects feedback in ICMP-like probes

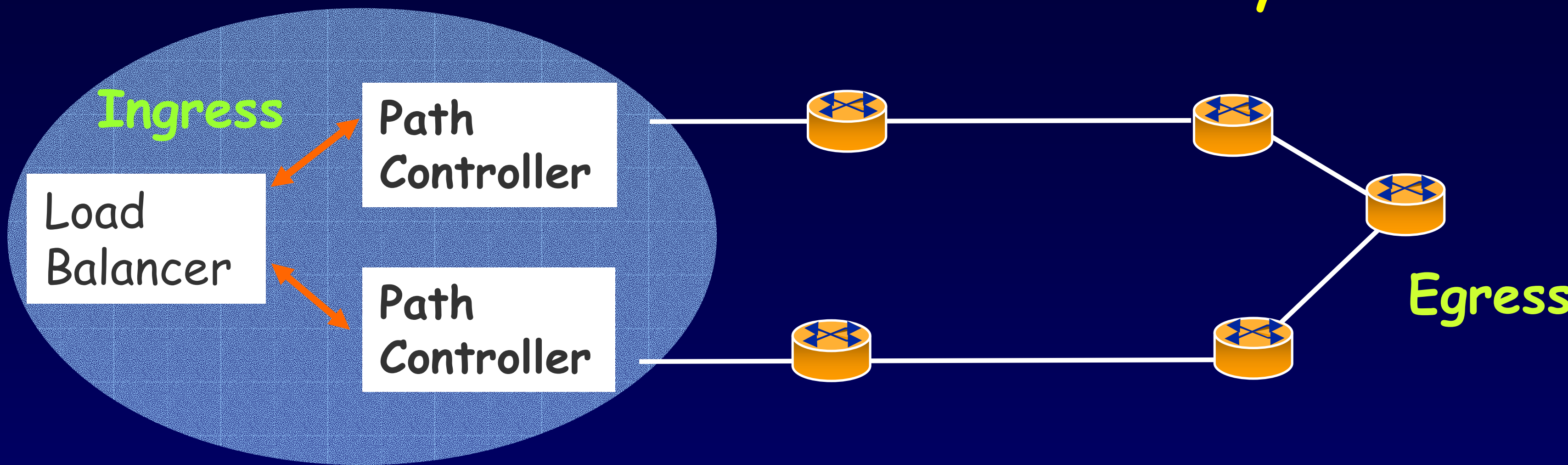


- Core router computes aggregate feedback

  Δ = Spare BW – Queue / Max-RTT
- Estimates number of IE-flows by counting probes, and divides feedback between them

Occasional explicit feedback in probes…
Need software changes only

Stability Idea



Per-path controller works at a faster timescale than load balancer → Can decouple components → Stabilize separately

Informally stated:

Theorem 1: Given a particular load split, the path controller stabilizes the traffic on each link

Theorem 2: Given stable path controllers,

- Every TeXCP agent sees balanced load on all paths
- Unused paths have higher utilization than used paths

Performance

Simulation Setup

Standard for TE

- Rocketfuel topologies
- Average demands follow gravity model
- IE-traffic consists of large # of Pareto on-off sources

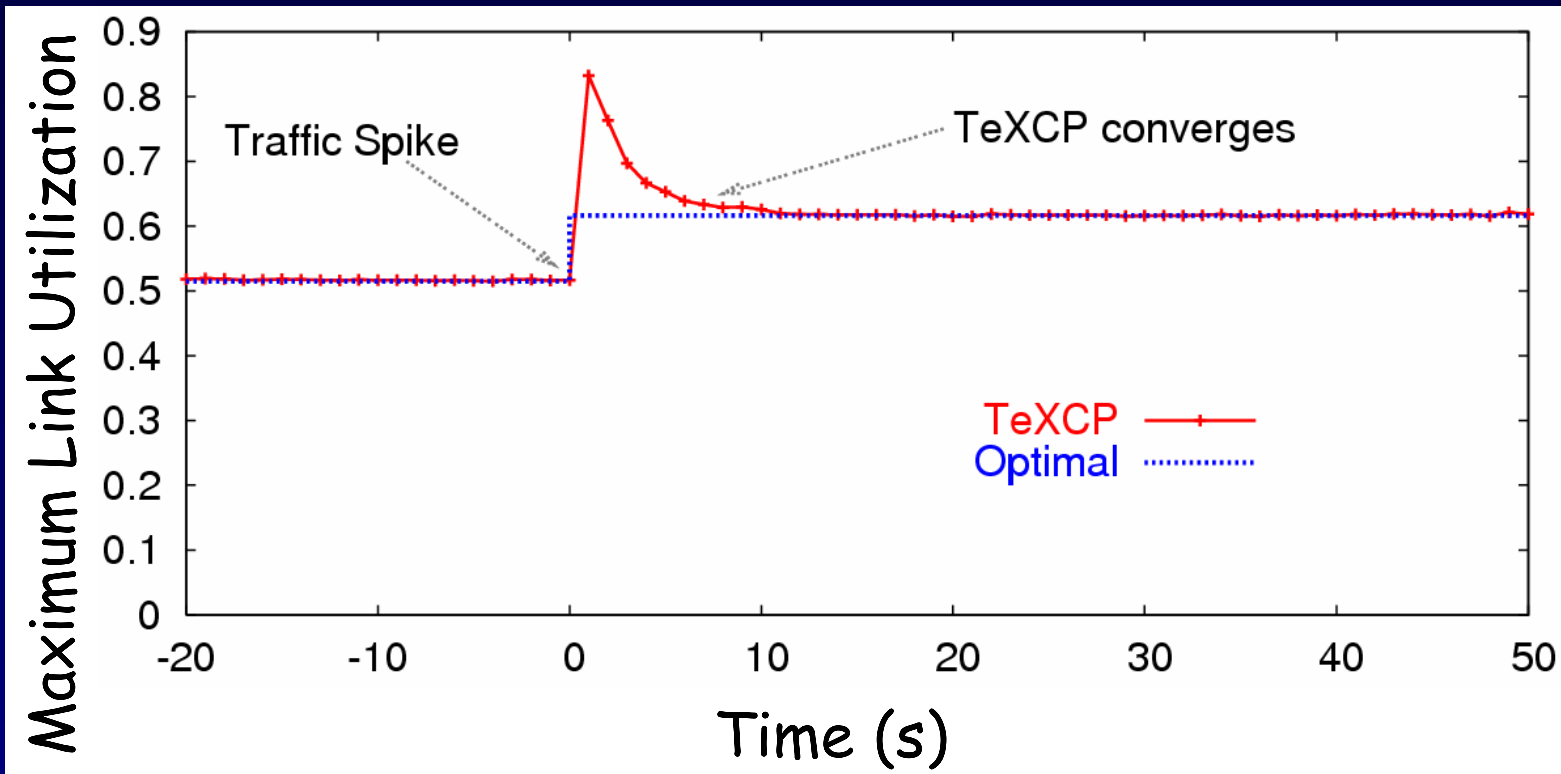
TeXCP Parameters:

- Each agent is configured with 10 shortest paths
- Probe for explicit feedback every 0.1s
- Load balancer re-computes a split every 0.5s

Compare to Optimal Max-Utilization

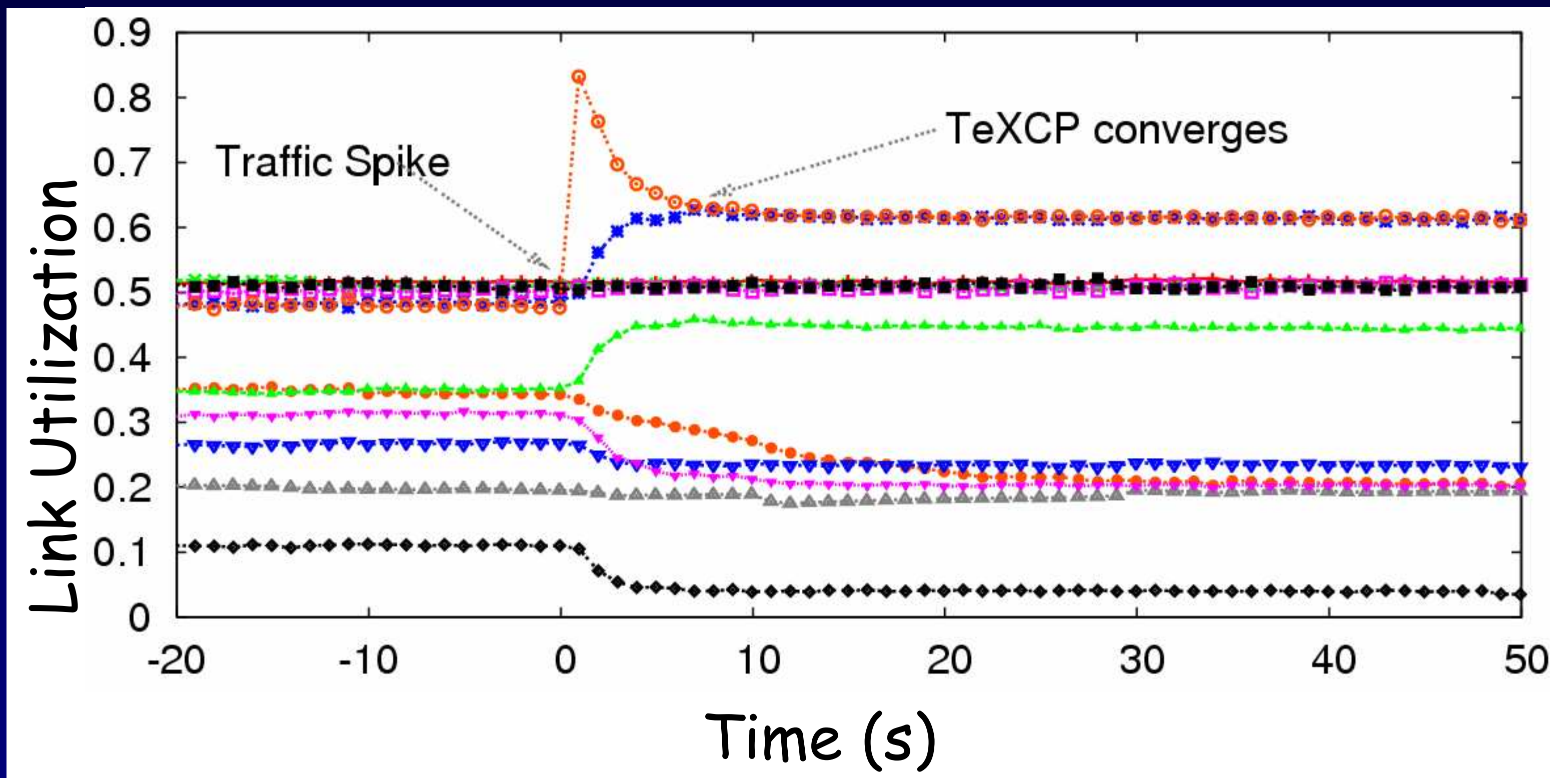
- Obtained with a centralized oracle that has Immediate and exact demands info, and uses as many paths as necessary

TeXCP Balances Load Without Oscillations



TeXCP converges to a few percent of optimal

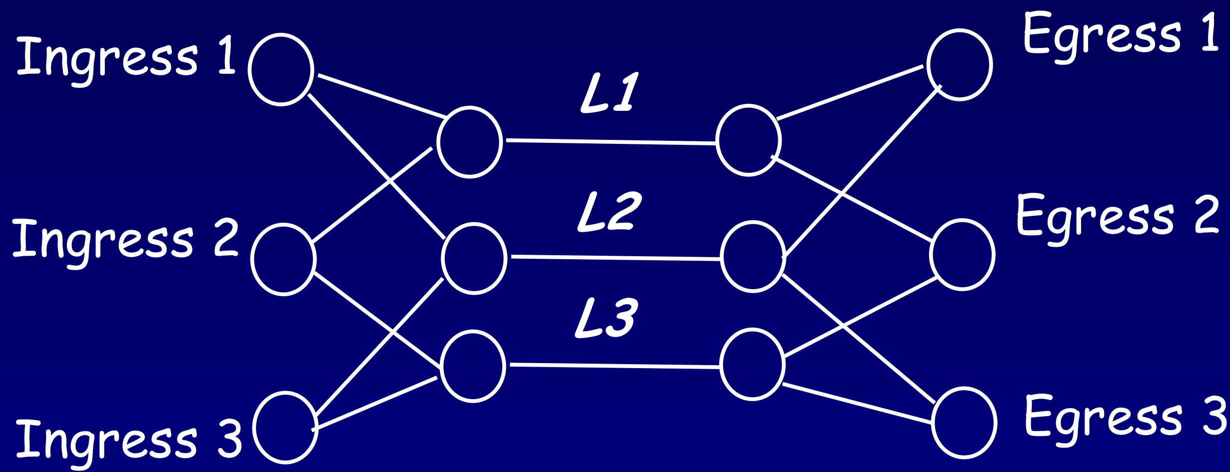
TeXCP Balances Load Without Oscillations



Utilizations of all links in the network change without oscillations

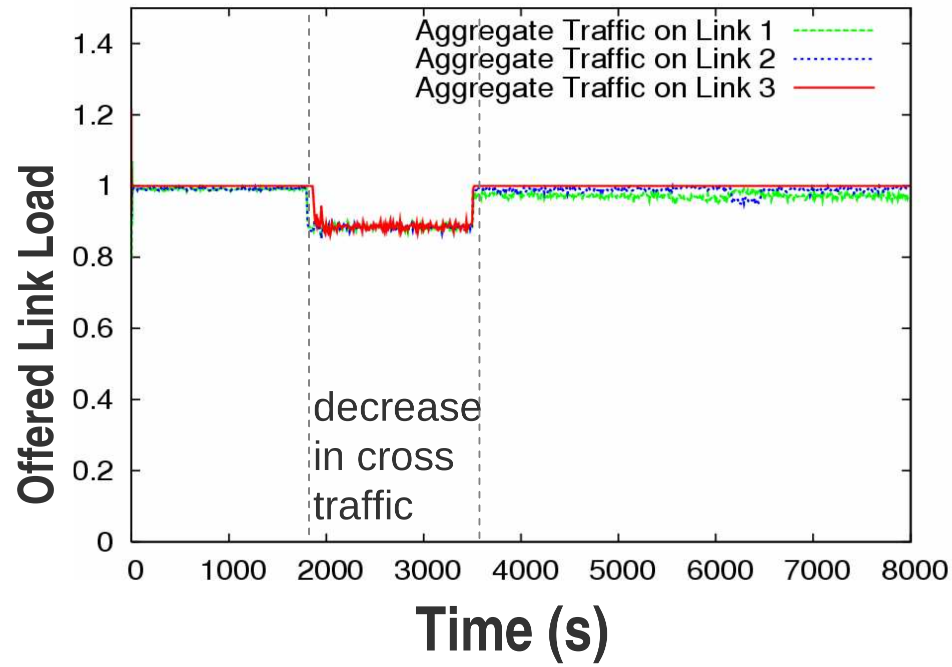
Comparison with MATE

- MATE is the state-of-the-art in online TE
- All simulation parameters are from the MATE paper

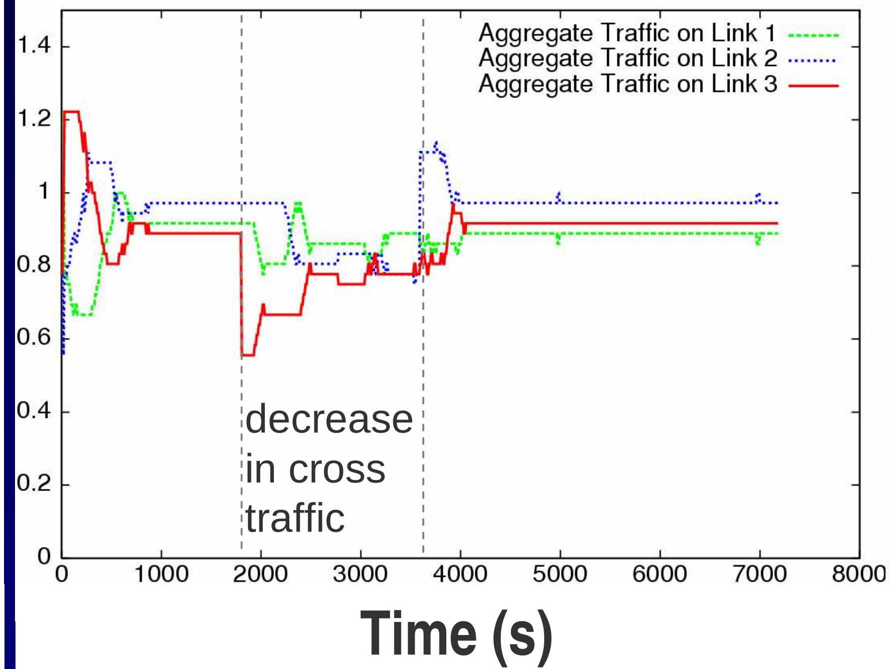


TeXCP balances load better than MATE

TeXCP



MATE



Avg. drop rate in MATE is 20% during convergence

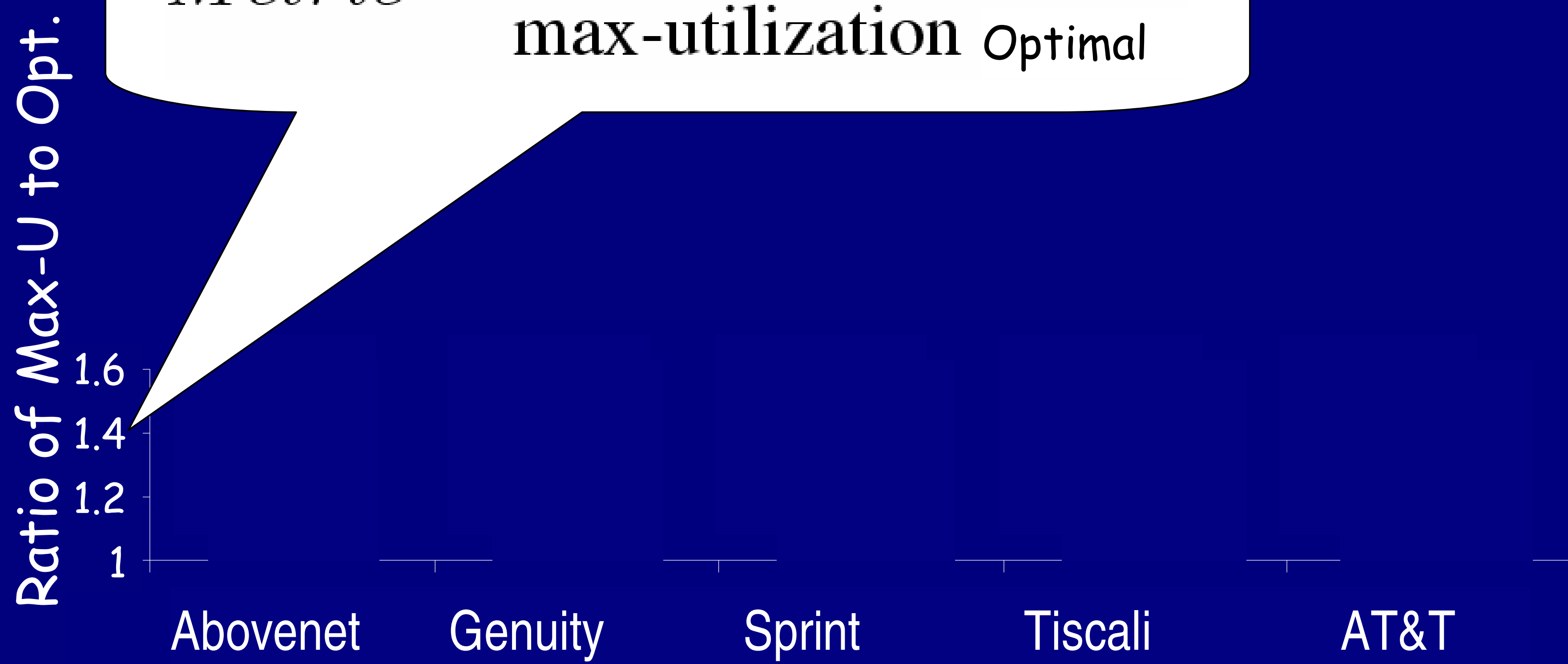
Explicit feedback allows TeXCP to react faster and without oscillations

Comparison with OSPF-TE

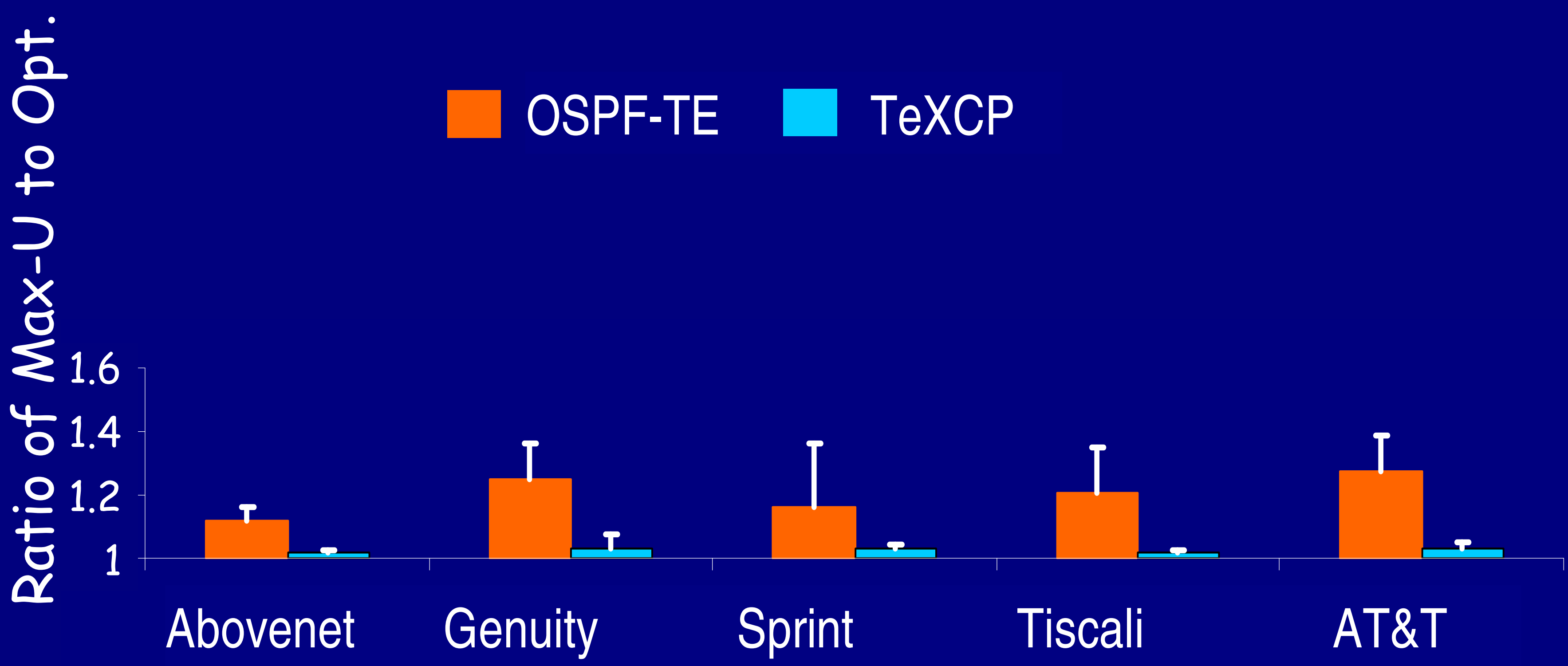
- OSPF-TE is the most-studied offline TE scheme
- It computes link weights, which when used in OSPF balance the load
- OSPF-TE-FAIL is an extension that optimizes for failures
- OSPF-TE-Multi-TM is an extension that optimizes for variations in traffic demands

Comparison with OSPF-TE under Static Load

$$Metric = \frac{\text{max-utilization}_{\,Tech.}}{\text{max-utilization}_{\,\text{Optimal}}}$$

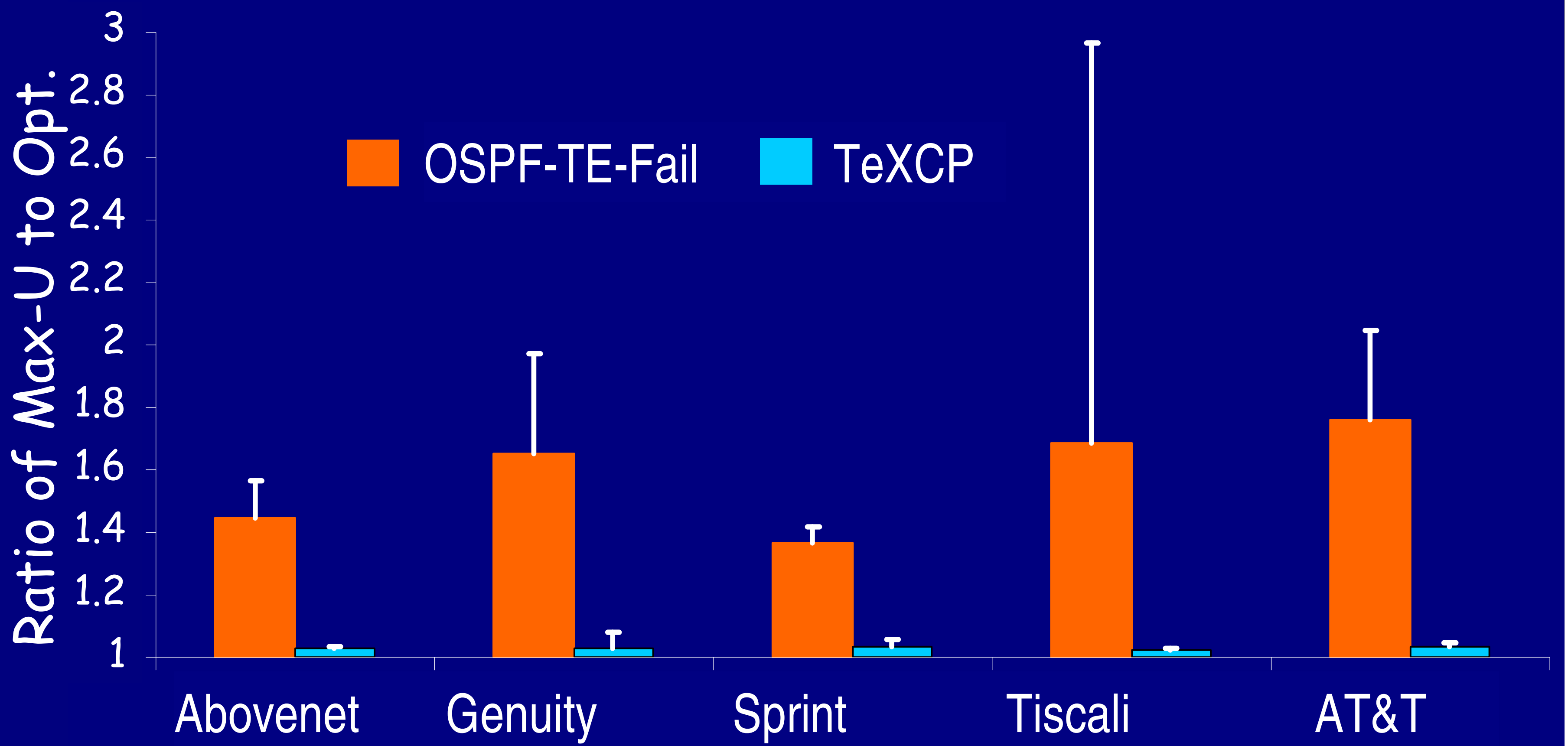


Comparison with OSPF-TE under Static Load



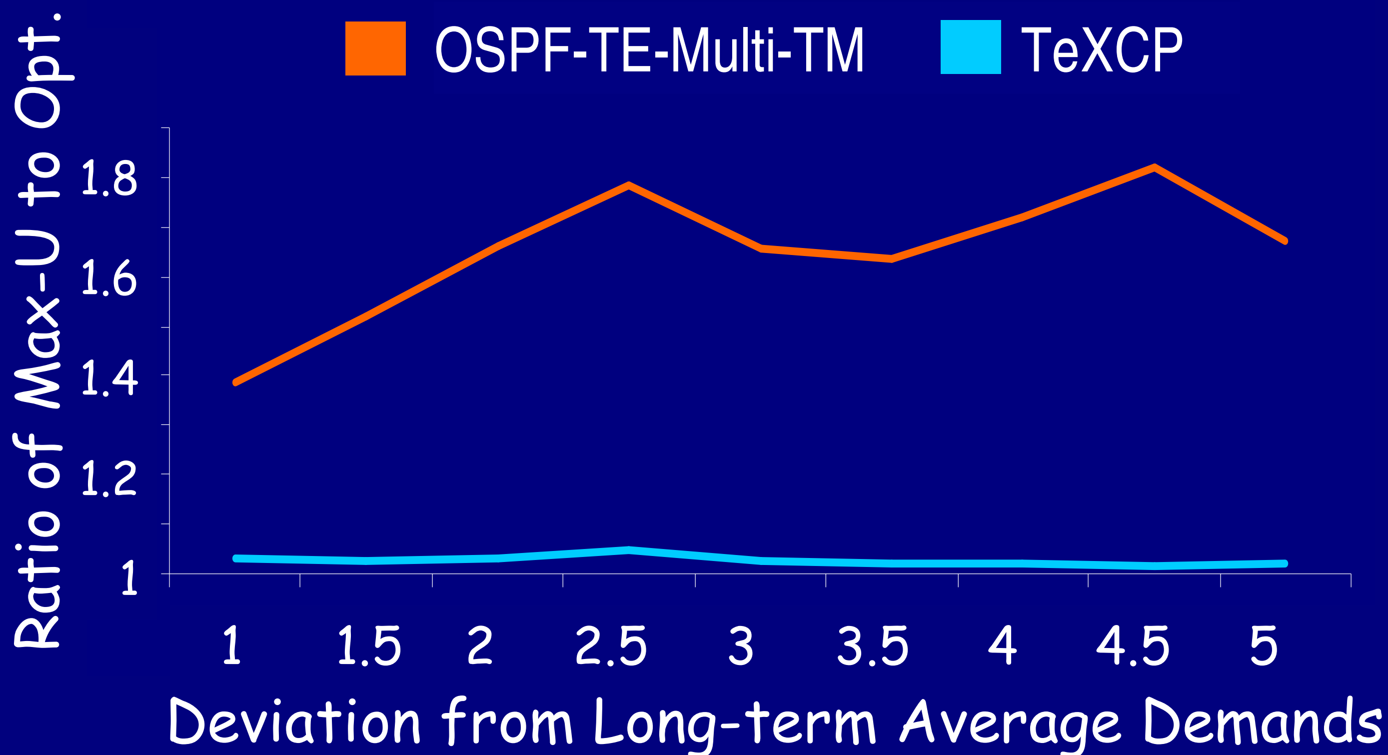
TeXCP is within a few percent of optimal, outperforming OSPF-TE

Comparison with OSPF-TE-Fail



TeXCP allows an ISP to support same failure resilience with about ½ the capacity !

Performance When Traffic Deviates From Long-term Averages



TeXCP reacts better to realtime demands!

Conclusion

- TeXCP: Responsive & Stable Online TE
- Combines load balancing with a Cong. Mngt. Layer to prevent overshoot and drops
- TeXCP keeps utilization always within a few percent of optimal
- Compared to MATE, it is faster and does not overshoot
- Compared to OSPF-TE
  - it keeps utilization 20% to 100% lower
  - it supports the same failure resilience with ½ the capacity → major savings for the ISP

http://nms.lcs.mit.edu/projects/texcp/