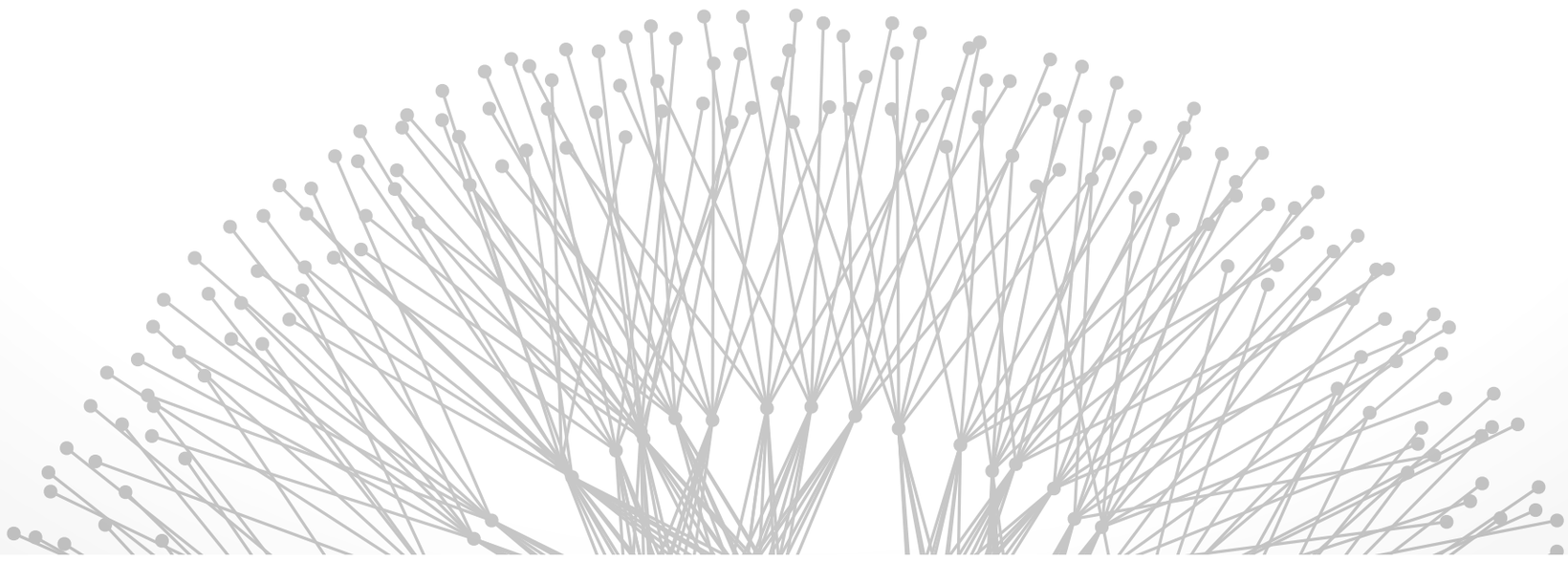
Intradomain Routing

Brighten Godfrey

CS 538 February 19 2018



Routing

Choosing paths along which messages will travel from source to destination.

Often defined as the job of Layer 3 (IP). But...

- Ethernet spanning tree protocol (Layer 2)
- Content delivery overlays, distributed hash tables, network virtualization, … (Layer 4+)

Problems for intradomain routing

Distributed path finding

React to dynamics

High reliability even with failures

Scale

Optimize link utilization (traffic engineering)

Protocols: The Building Blocks

Distance vector routing

Protocol variants

- Original ARPANET
- More recently: RIP, EIGRP

Remember vector of distances to each destination and exchange this vector with neighbors

- Initially: distance 0 from myself
- Upon receipt of vector: my distance to each destination = min of all my neighbors' distances + 1
- Send packet to neighbor with lowest dist.

Slow convergence and looping problems

- E.g., consider case of disconnection from destination

Path vector routing

Protocol variants

- BGP

Remember vector of paths to each destination and exchange path announcements

- Initially: empty path to myself
- Upon receipt of path announcement or withdrawal: selected path = shortest among active paths to destination
- Send packet along selected path

Link state routing

Protocol variants

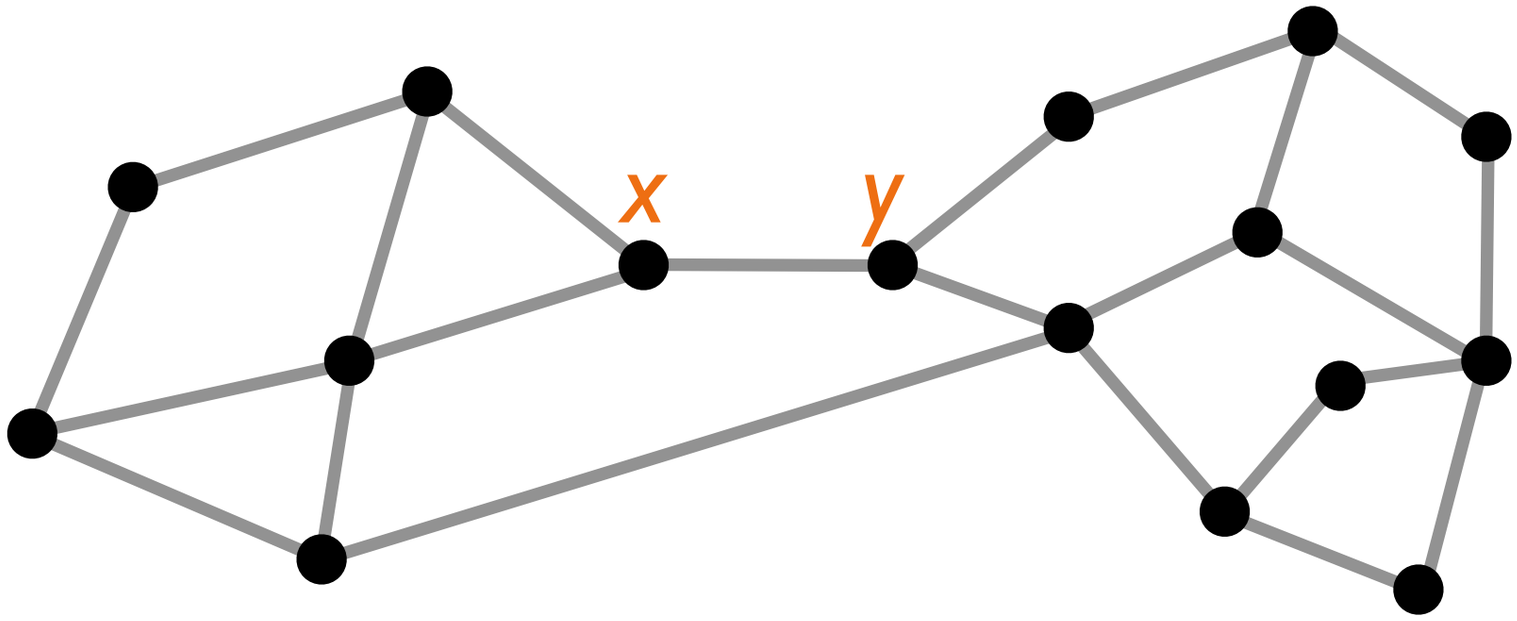
- ARPANET: McQuillan, Richer, Rosen 1980; Perlman 1983
- Intermediate System-to-Intermediate System (IS-IS)
- Open Shortest Path First (OSPF)

Algorithm

- Gossip the entire topology to everyone
- Forwarding at each hop:
  - Compute shortest path (e.g., Dijkstra's algorithm)
  - Send packet to neighbor along computed path

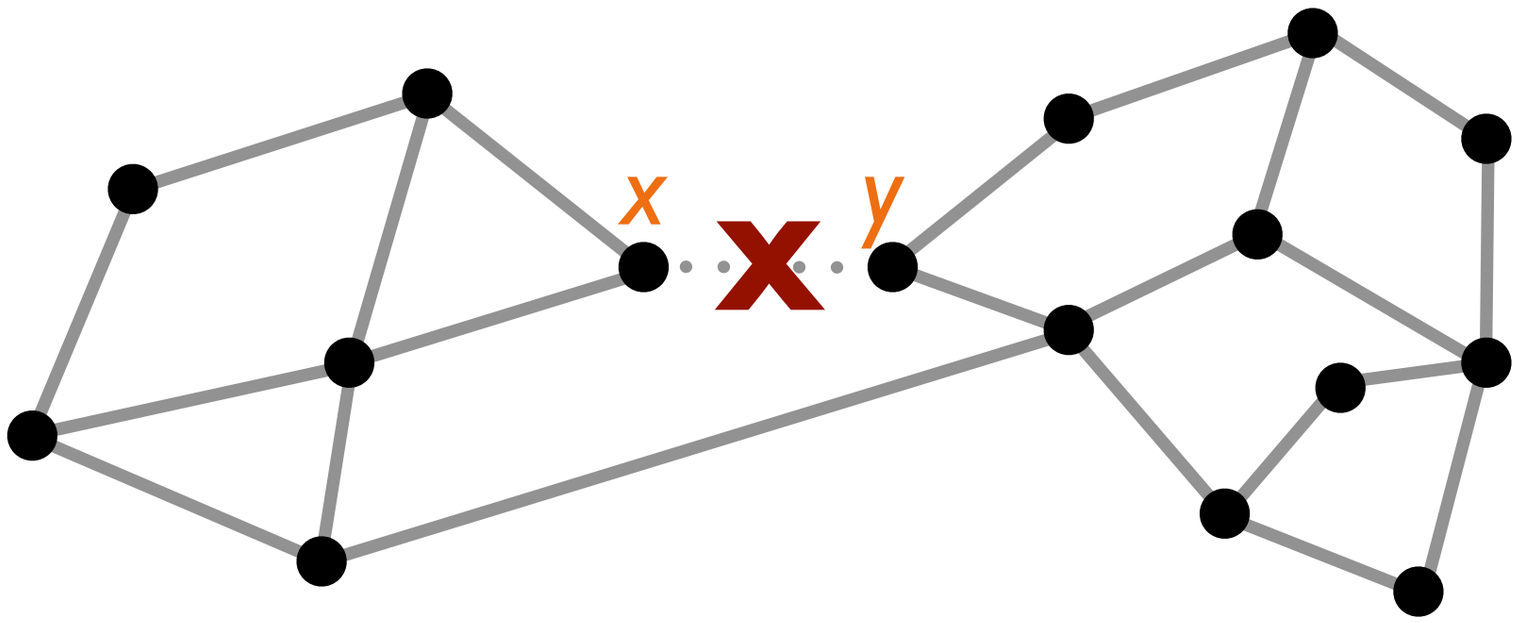
Question

We have a network...



Question

A link fails. How many total units of message does *x* send in immediate response?



...using DV or PV?

...using link state?

Link state vs. DV/PV

Disadvantages of LS

- Need consistent computation of shortest paths
  - Same view of topology
  - Same metric in computing routes
- Slightly more complicated protocol

Advantages of LS

- Faster convergence if compute time is negligible
- Gives unified global view
  - Useful for other purposes, e.g., building MPLS tables

Q: Can link state have forwarding loops?

Comparison

DV/PV	Link State
Computes paths incrementally as path announcements are disseminated	Decouples topology gossiping from routing decision
• Simple protocol • Local, independent decisions	• Slightly more complex, but not too bad • Requires consistent view of topology and routing metric across all routers

• Can have convergence problems
  • e.g. burst of updates
  • Or worse if you lack a path vector…
	• Faster convergence if compute time is negligible
	• Provides unified global view
	  • Useful for other purposes, e.g. building MPLS tables

Q: Can link state have forwarding loops?

LS variant: Source routing

Algorithm:

- Broadcast the entire topology to everyone
- Forwarding at source:
  - Compute shortest path (Dijkstra's algorithm)
  - Put path in packet header
- Forwarding at source and remaining hops:
  - Follow path specified by source

Q: Can this result in forwarding loops?

Source routing vs. link state

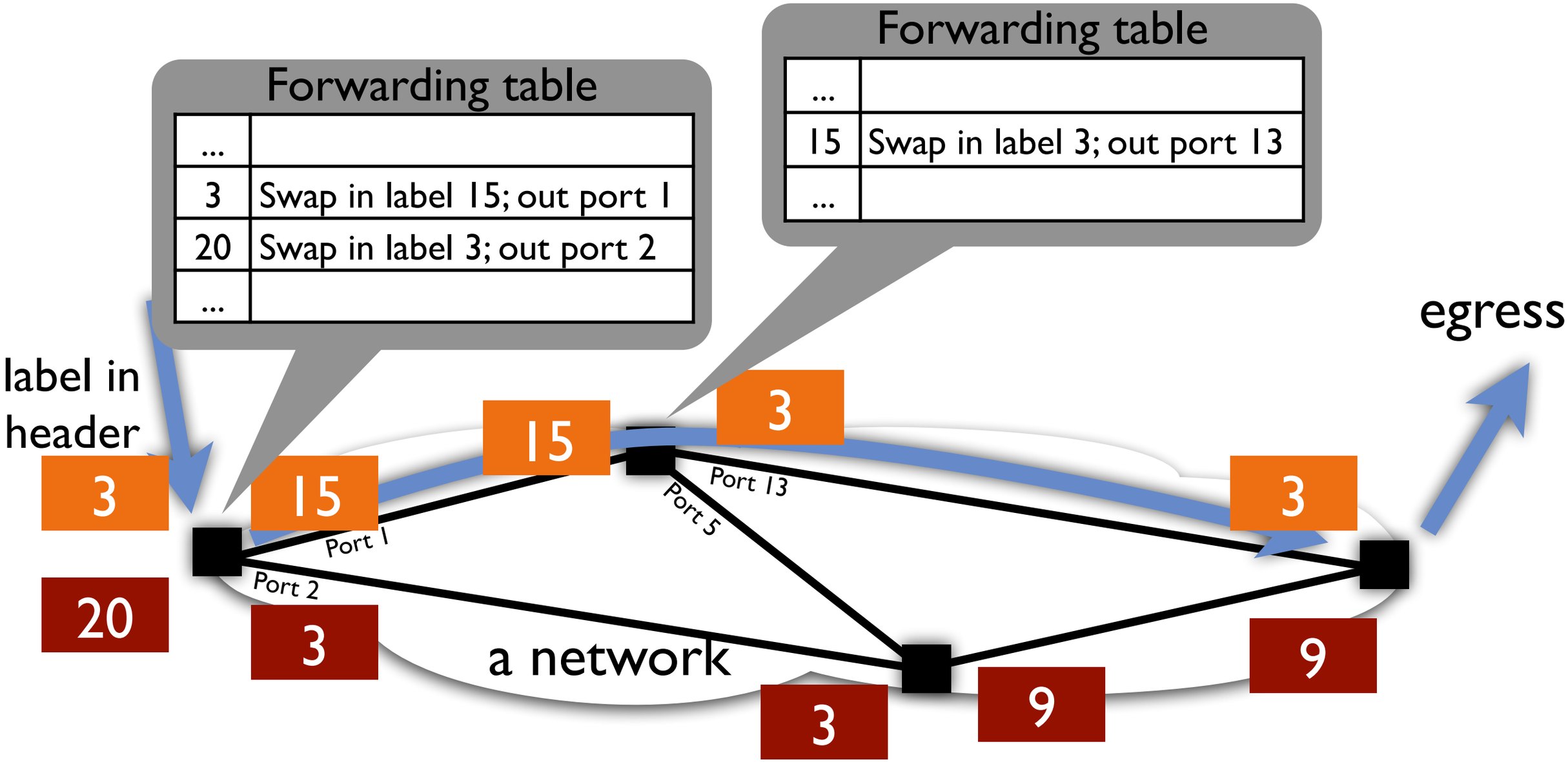
Advantages

- Essentially eliminates loops
- Compute route only once rather than every hop
- Forwarding table (FIB) size = #neighbors (not #nodes)
- Flexible computation of paths at source

Disadvantages

- Computation of paths at source
- Header size: $\geq \log_2(\#nodes) \cdot |Path|$
  - Can use local rather than global next-hop identifiers
  - Then, size drops to $\geq \log_2(\#neighbors) \cdot |Path|$
- Source needs to know topology
- Harder to redirect packets in flight (to avoid a failure)

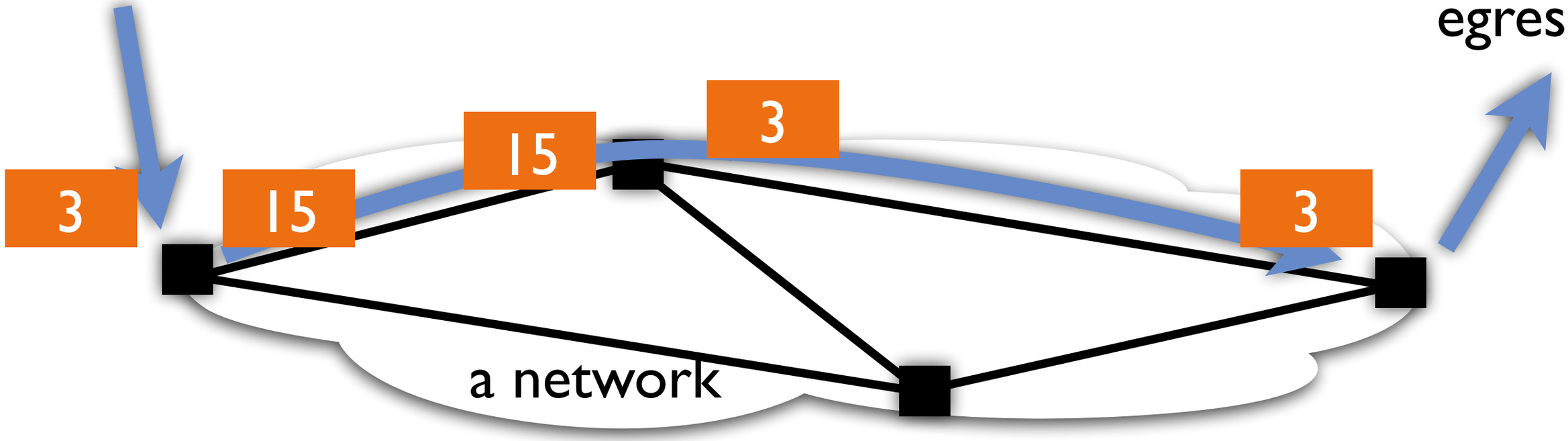
MPLS design



Why is this more flexible than shortest path routing?

MPLS design

Ingress:
Traffic classification, label packets ("forwarding equivalence class")



Control plane constructs paths and coordinates labels

Can also stack labels = concatenate paths

- used for backup paths in MPLS Fast ReRoute (FRR)

MPLS motivation

In the design doc

- High performance forwarding
- Minimal forwarding requirements, so can interface well with many types of media such as ATM
- Flexible control of traffic routing

What matters today?

Flexibility. Widely used to achieve:

- Virtual Private Network (VPN) service along dedicated paths between enterprise sites
- Control backup paths with MPLS Fast ReRoute
- Traffic engineering (load balancing)

Using the Protocols: Traditional Traffic Engineering

Traffic engineering

Key task of intradomain routing: optimize utilization

No TE: Shortest path routing

- How well does this work?

What do we actually want to accomplish?

TE: Classic ISP formulation

Given

C_{ij} Capacity of link from *i* to *j*

T_{st} Traffic demand from *s* to *t*

Objective

$\min u^*$

Subject to

u^* Max link utilization (0 to 1)

x_{ijst} Traffic volume of (*s*,*t*) flow carried over link (i,j)

$$\forall_{st}\ T_{st} = \sum_j x_{sjst}$$ Ingress flow equals demand

$$\forall_{st}\ T_{st} = \sum_i x_{itst}$$ Egress flow equals demand

$$\forall_{st}\forall_{v\notin\{s,t\}}\ \sum_i x_{ivst} = \sum_j x_{vjst}$$ Flow conservation

$$\forall_{i,j} \sum_{s,t} x_{ijst} \leq u^* C_{ij}$$ Utilization cap

TE: Classic ISP formulation

Can we solve it?

- Computationally yes: multi-commodity flow problem, represented as linear program,
- Solvable in polynomial time with LP solvers
- But how do we solve it in a distributed way with ever-changing inputs?

Is it enough?

- What about different objectives (e.g. latency)?
- What about different priorities (enterprise VPN vs. best-effort Internet flow)?

Classic TE solutions

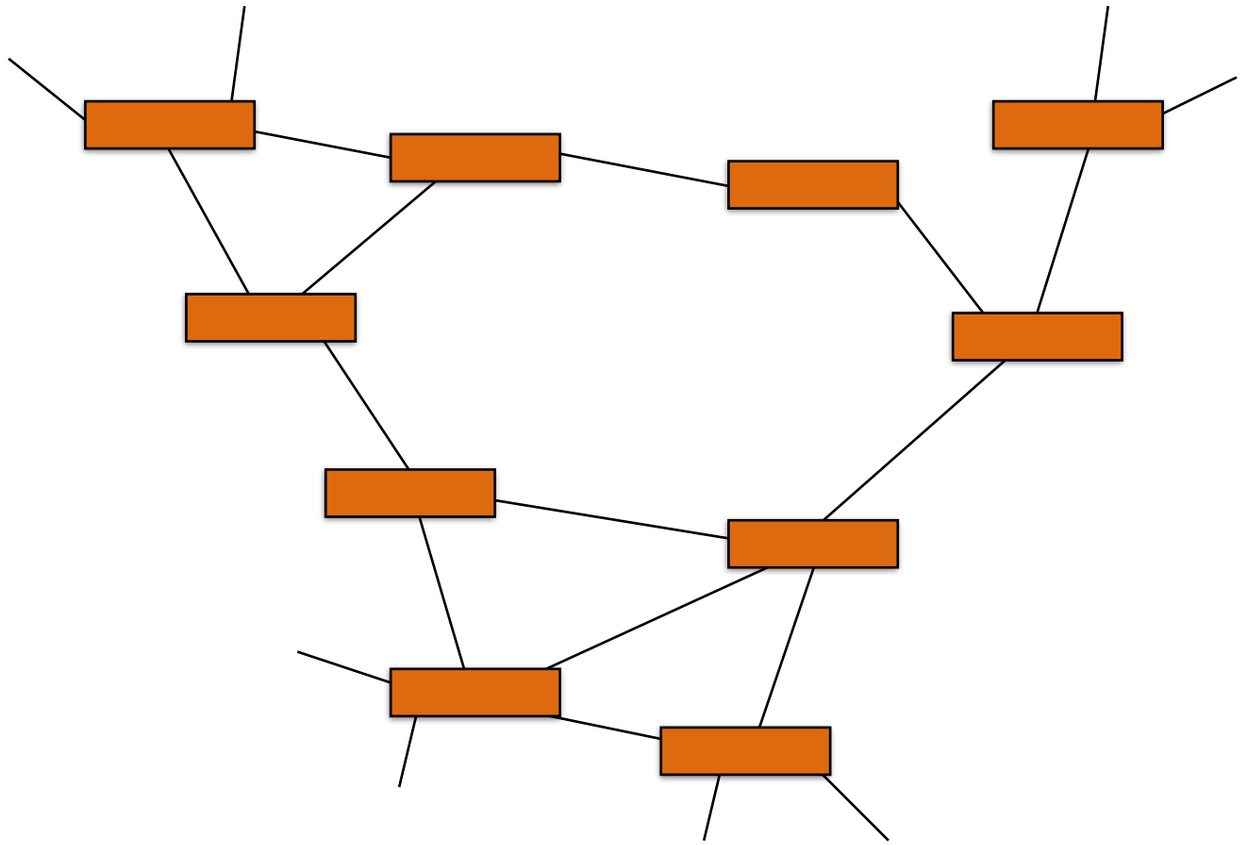
Approach 1: Optimize OSPF weights

- e.g. OSPF-TE
- Need to propagate everywhere: can't change often
- Artificial constraints make it difficult to optimize
  - Same weights apply to all traffic
  - So all traffic at one ingress follows same paths

Approach 2: Allocate traffic to explicit MPLS paths

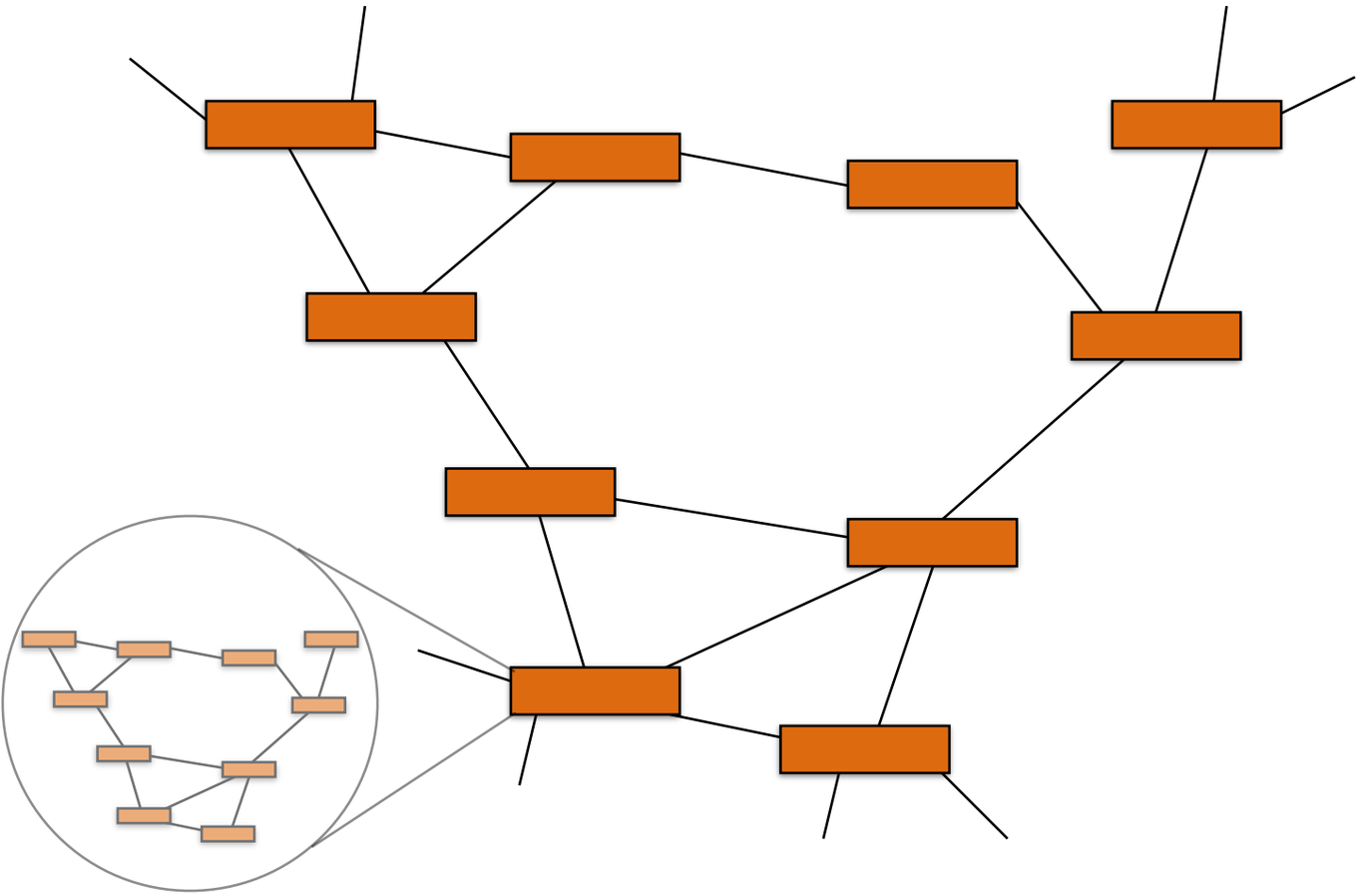
- Control protocol like RSVP-TE reserves capacity and constructs MPLS tunnels at each router along path
- Tradeoff: improves path choice but also state in routers
  - Not all possible paths will be available

MPLS with RSVP-TE



[Slide Credit: Ankit Singla]

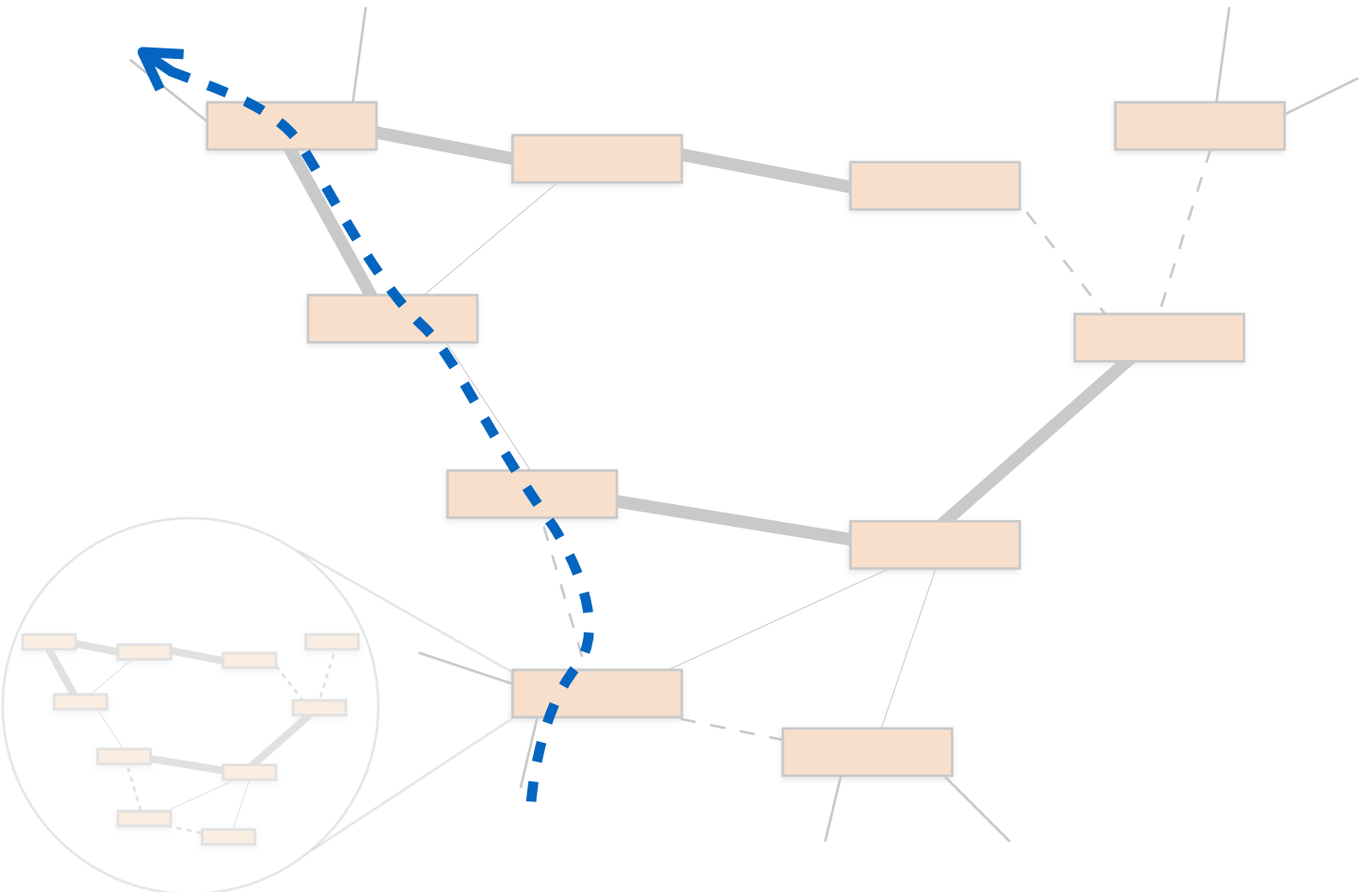
MPLS with RSVP-TE



- Link-state protocol (OSPF / IS-IS)

[Slide Credit: Ankit Singla]

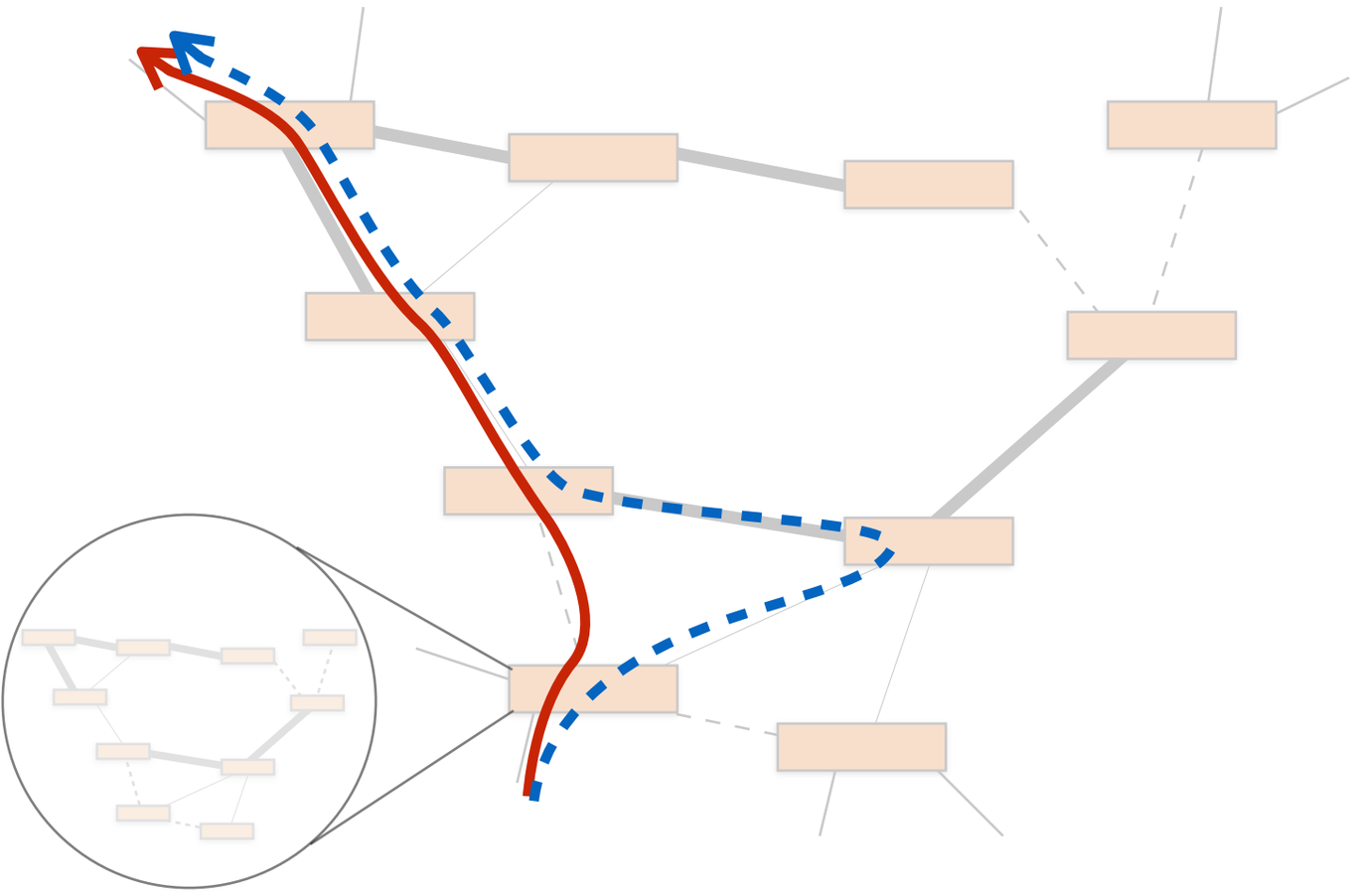
MPLS with RSVP-TE



1. Link-state protocol (OSPF / IS-IS)
2. Also flood available bandwidth info
3. Fulfill tunnel provisioning requests

[Slide Credit: Ankit Singla]

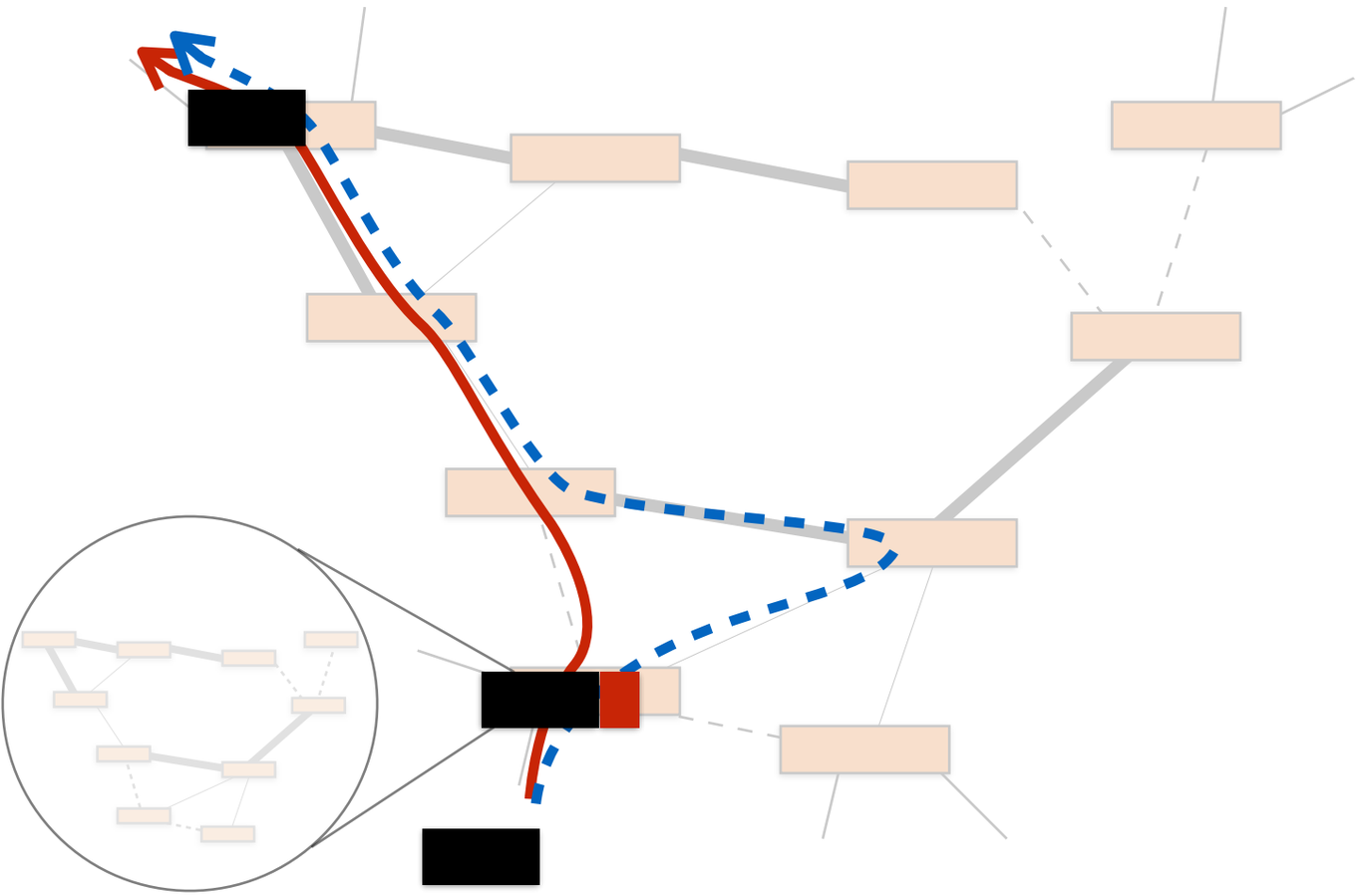
MPLS with RSVP-TE



1. Link-state protocol (OSPF / IS-IS)
2. Also flood available bandwidth info
3. Fulfill tunnel provisioning requests
4. Update network state, flood info

[Slide Credit: Ankit Singla]

MPLS with RSVP-TE



1. Link-state protocol (OSPF / IS-IS)
2. Also flood available bandwidth info
3. Fulfill tunnel provisioning requests
4. Update network state, flood info

[Slide Credit: Ankit Singla]

More Advanced TE

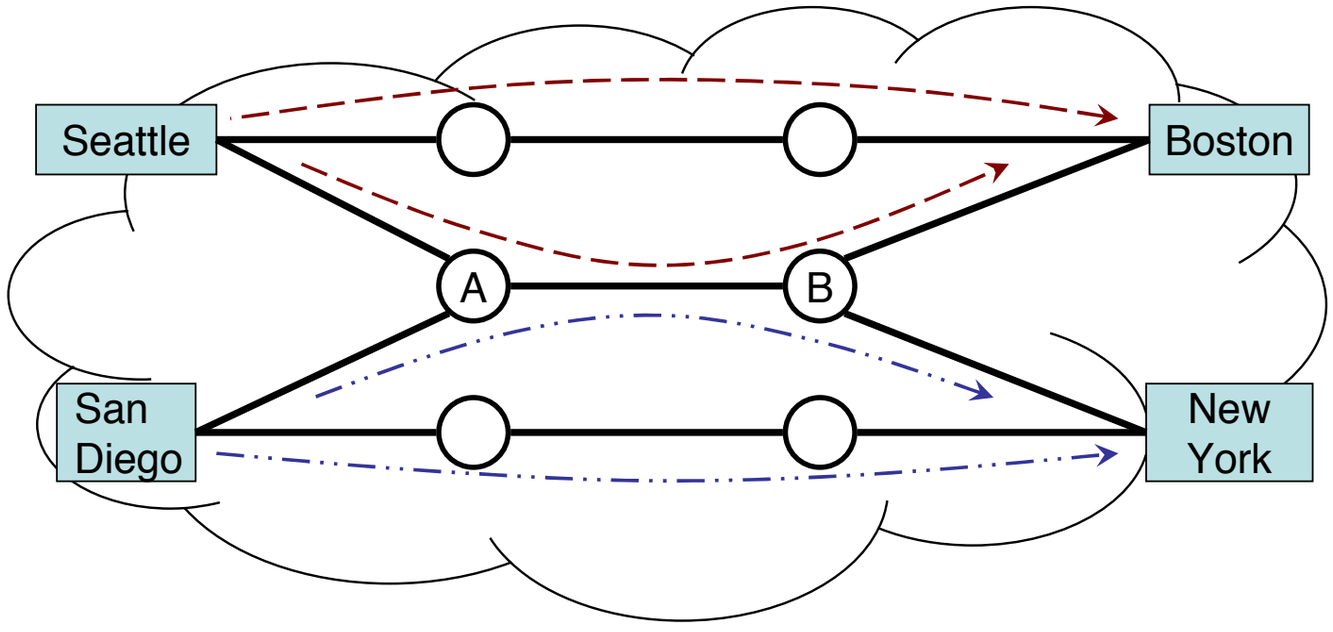
TeXCP [Kandula et al 2005]

Pre-construct small set of paths between every ingress-egress pair

- 10 MPLS tunnels in implementation

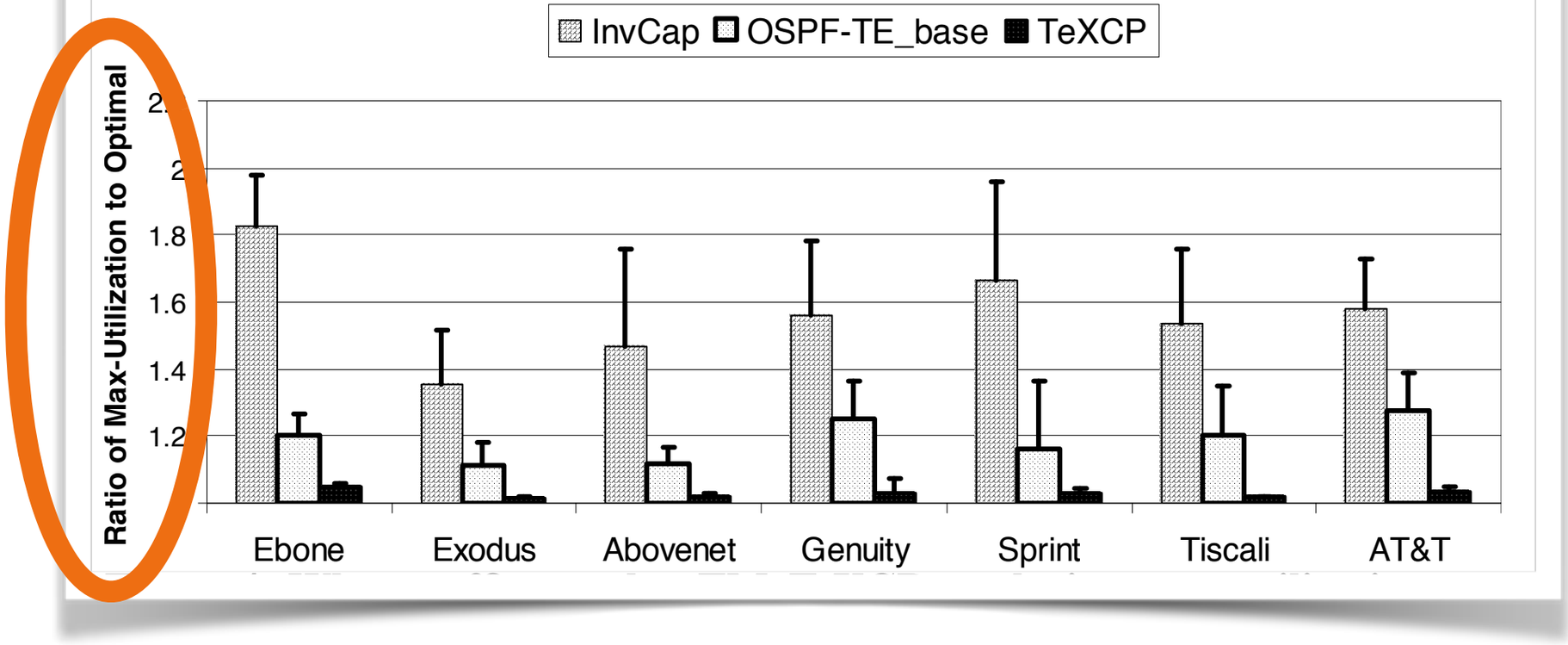
Dynamically at each ingress node:

- Probe utilization, latency of each path
- Dynamically reallocate traffic between paths



[Kandula et al, "Walking the Tightrope", SIGCOMM 2005]

TeXCP results

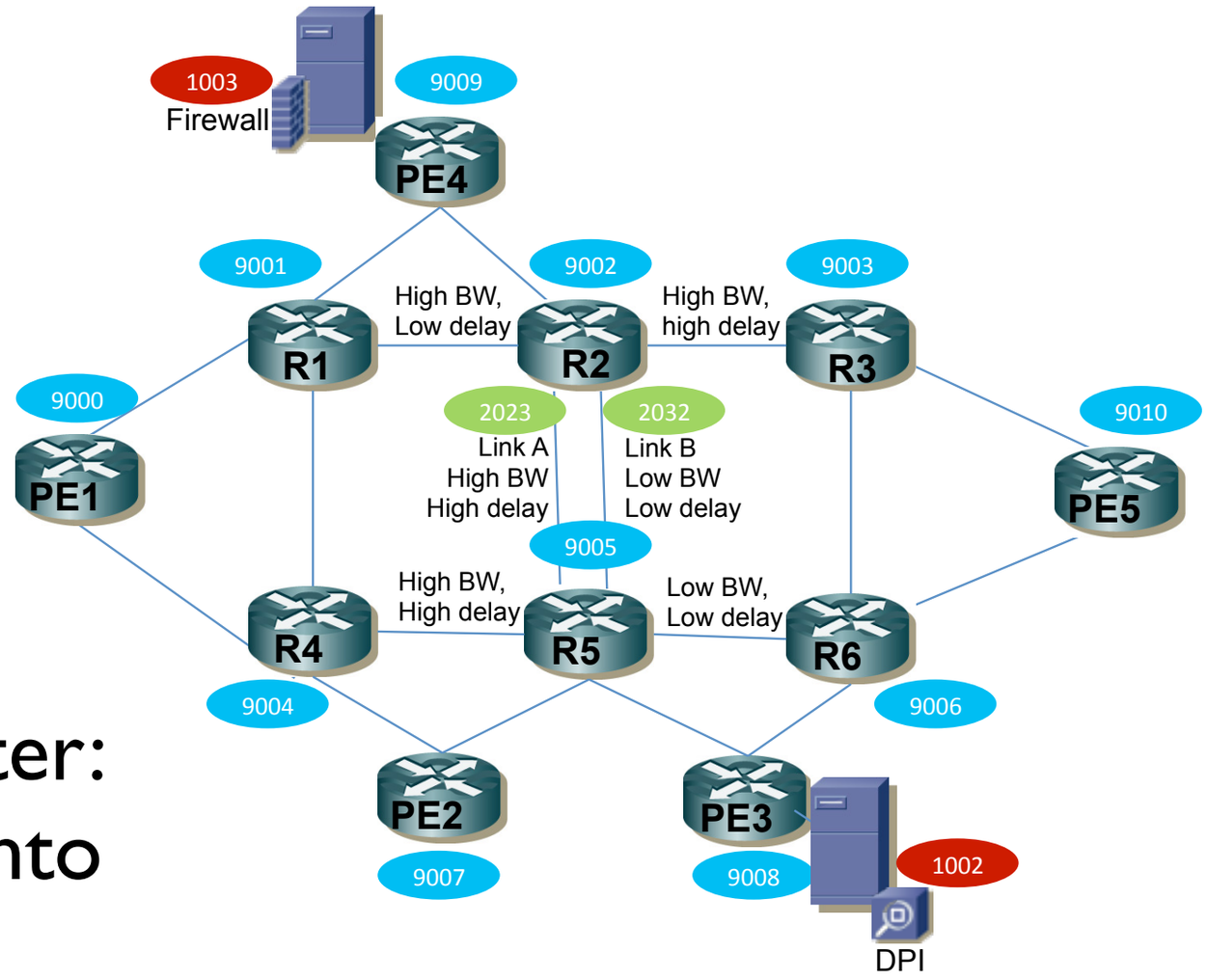


Q: In OSPF-TE, "Finding optimal link weights that minimize the max-utilization is NP-hard". Why is this harder than finding the best possible (non-OSPF) solution?

Background: Segment Routing

Idea: source routing by composing path segments

- Segment identifies
  - link or service (local)
  - router (global)
- Associated actions at router:
  - Push a new segment onto front of packet
  - Continue forwarding along a specified segment
  - Go to Next segment in packet
- Can be implemented with MPLS



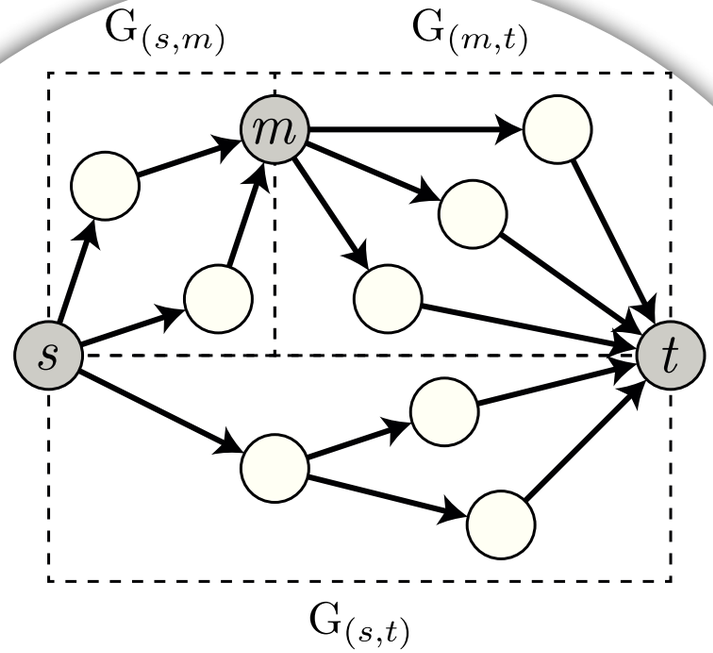
DEFO

A Declarative and Expressive Approach to Control Forwarding Paths in Carrier-Grade Networks

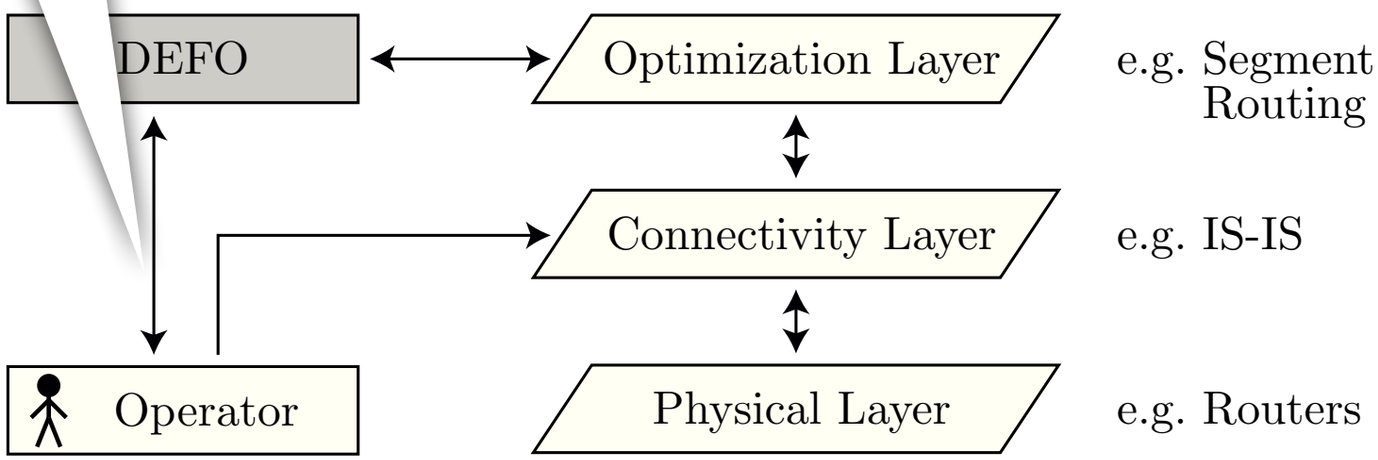
Hartert, Vissicchio, Schaus, Bonaventure, Filsfils, Telkamp, Francois

SIGCOMM 2015

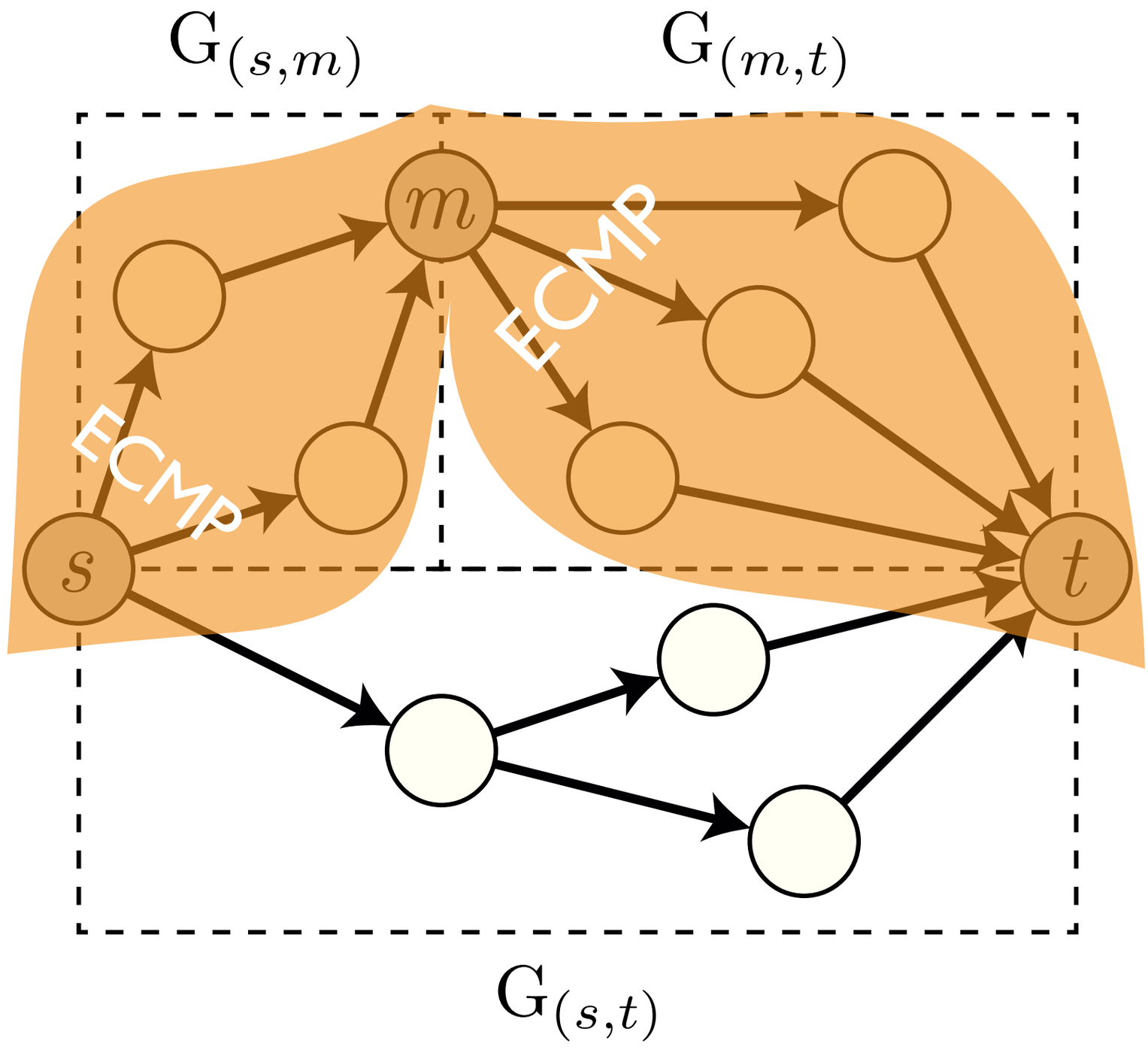
```
val goal = new Goal(topology){
 for(d<-Demands) add(d.deviations <= 2)
 for(l<-topology.links) add(l.load <= 0.9 l.capacity)
 minimize(MaxLoad)}
```



… for each ingress-egress traffic bundle



DEFO



DEFO discussion

What's the benefit of using a middlepoint instead of an explicit path?

What are the advantages & disadvantages of DEFO compared to TeXCP?

Announcements

Wednesday

Project proposal feedback later today

Readings

- OpenFlow (McKeown et al, 2008)
- Fabric: A Retrospective on Evolving SDN (Casado et al, HotSDN 2012)
- Recommended, but no review: The Future of Networking (Shenker, ONS 2011)