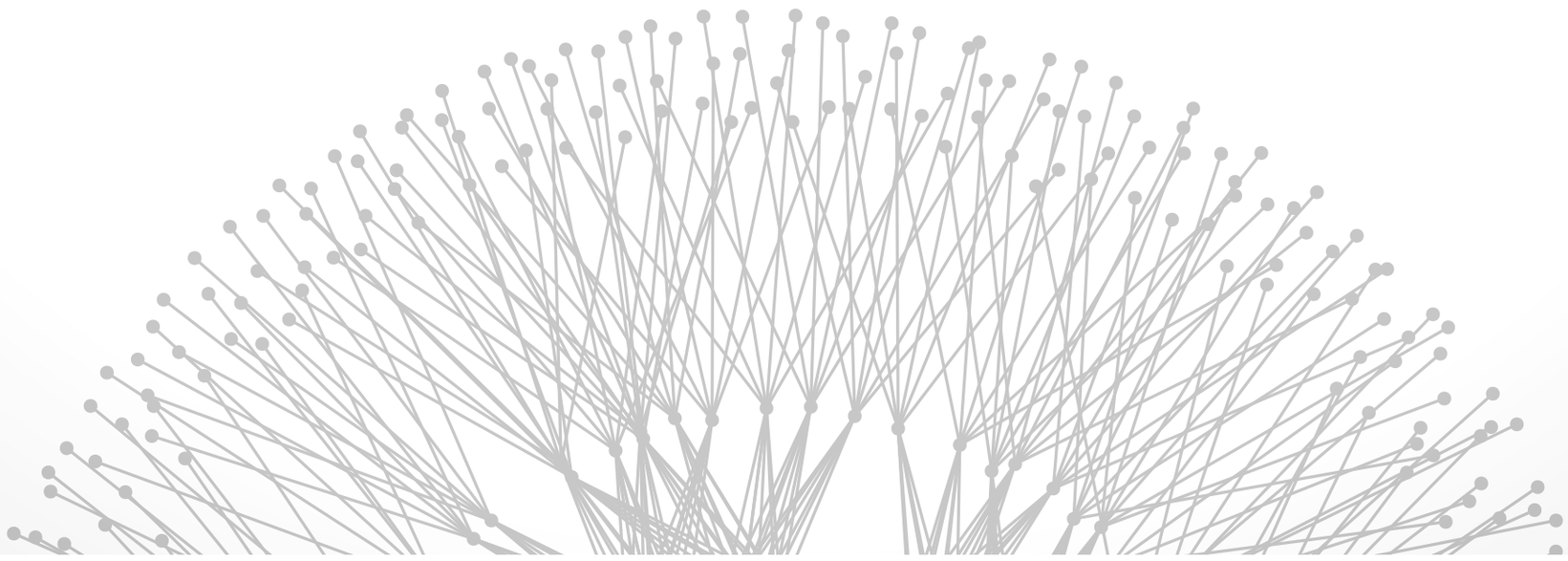
# Intradomain Routing

Brighten Godfrey

CS 538 February 20 2017



# 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)
- Distributed hash tables, content delivery overlays, ... (Layer 4+)

# Problems for intradomain routing

Distributed path finding

React to dynamics

High reliability even with failures

Scale

Optimize link utilization (traffic engineering)

# The two classic approaches

Distance Vector & Link State

Far from the only two approaches!

# Distance vector routing

Original ARPANET: distance vector routing

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
- Fix for loops in BGP: store path instead of distance

# Link state routing

## Protocol variants

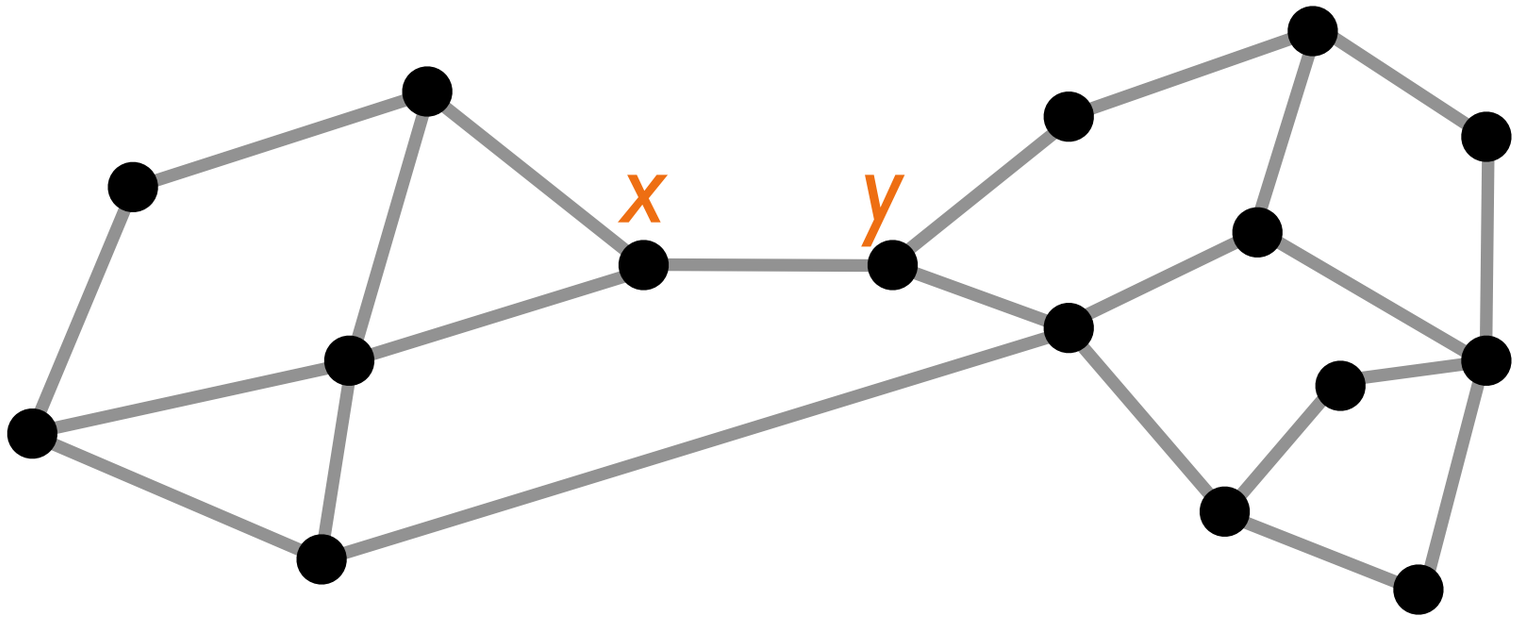
- ARPANET: McQuillan, Richer, Rosen 1980; Perlman 1983
- Intermediate System-to-Intermediate System (IS-IS)
- Open Shortest Path First (OSPF)

## Algorithm

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

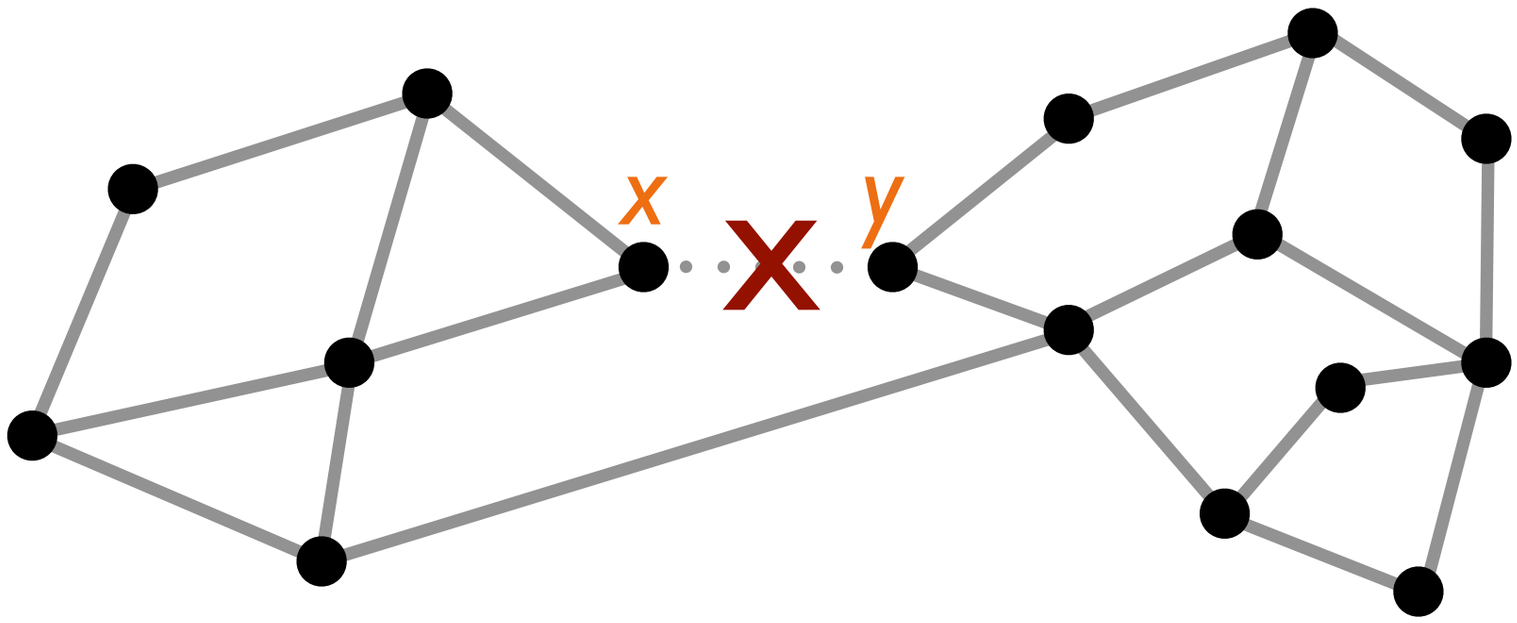
# Question

We have a network...



# Question

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

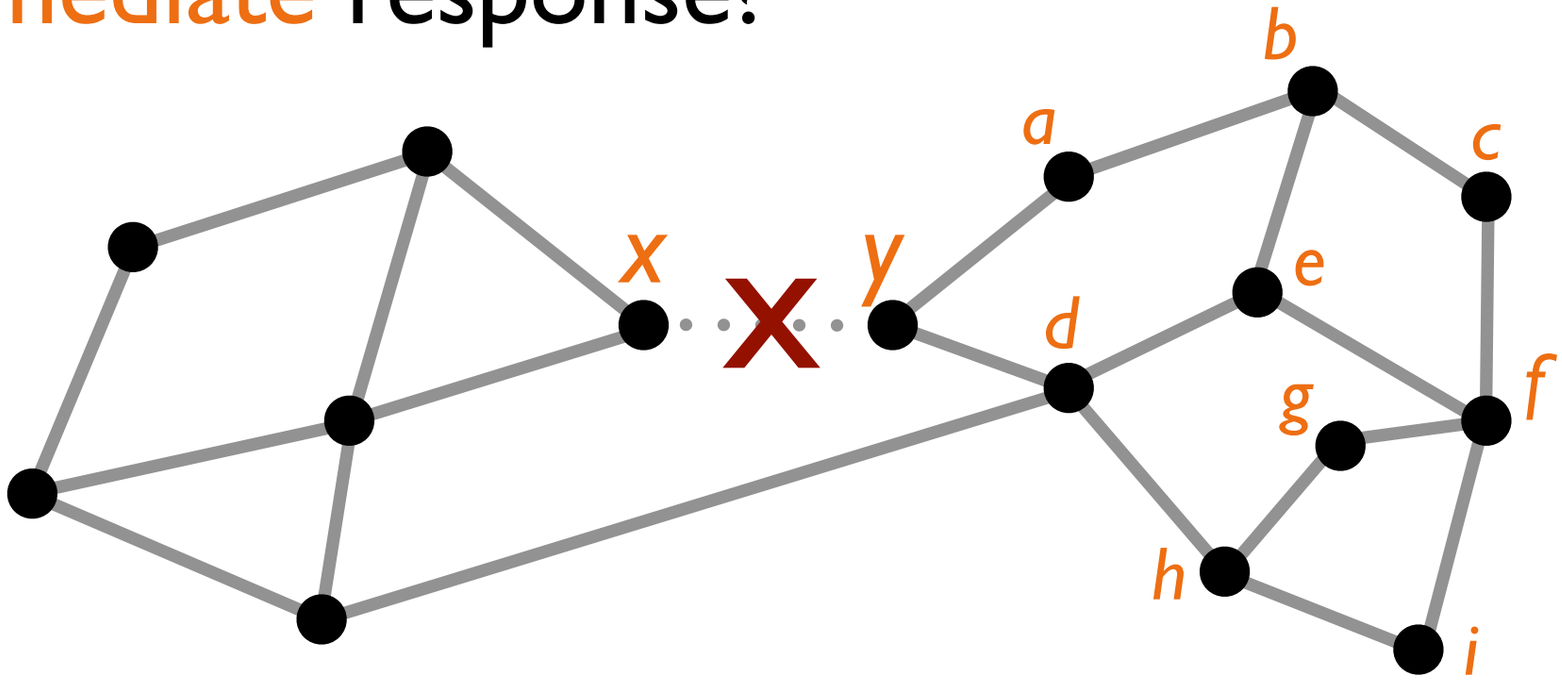



...using distance vector?

...using link state?

# Question

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




## ...using distance vector?

20

"My distance to *y* changed!
My distance to *a* changed!
My distance to *b* changed!
...
My distance to *i* changed!"
...to each of 2 neighbors

## ...using link state?

2

"Oh hey, link *x*-*y* failed"
...to each of 2 neighbors

# Link state vs. distance vector

## 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
- Gives 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)

# Traffic engineering

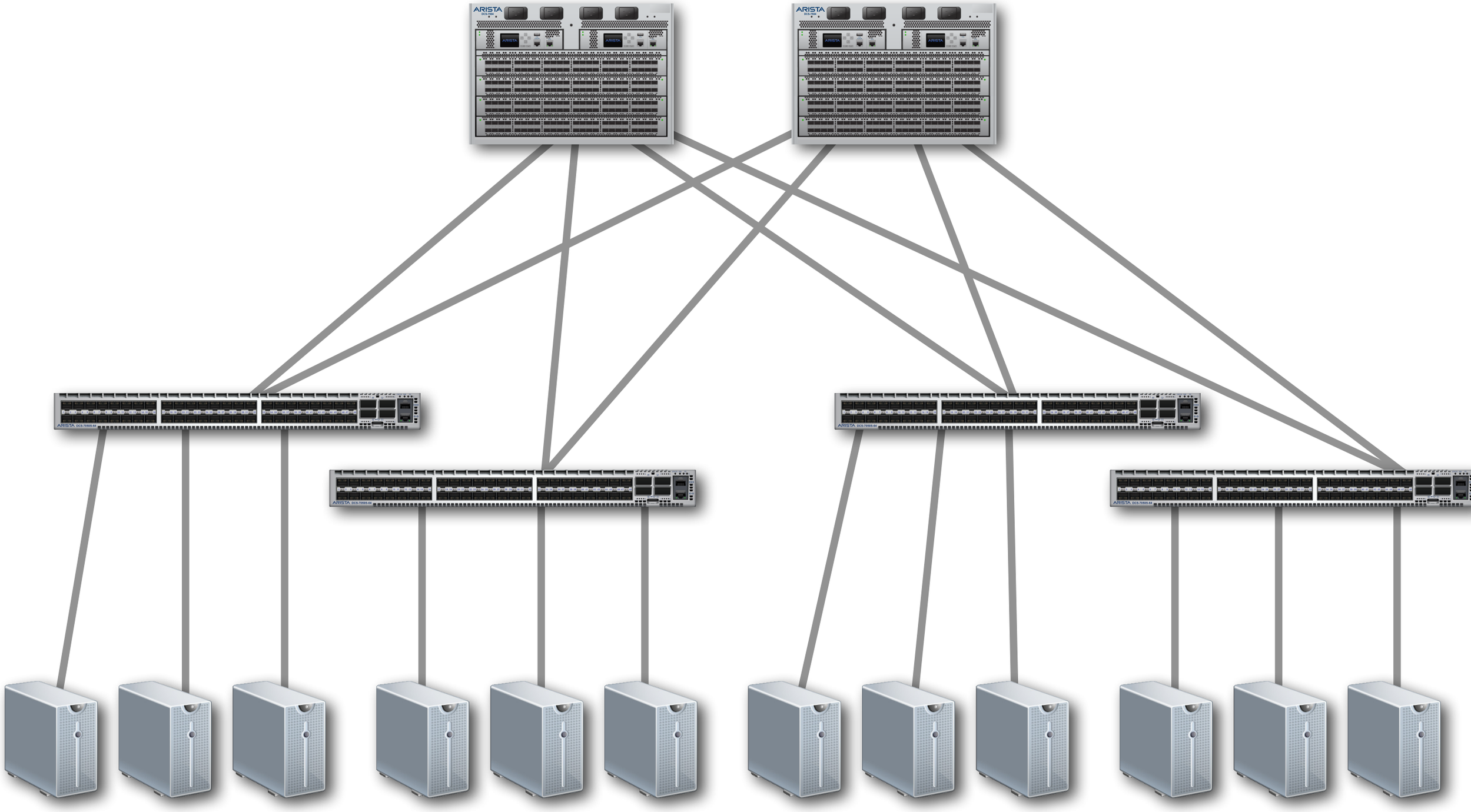
Key task of intradomain routing: optimize utilization

No TE: Shortest path routing

- How well does this work?

A start: Equal Cost Multipath Protocol (ECMP)

- Each router splits traffic across equally short next-hops
- Hash header to pin flow to a pseudorandom path (why?)
- When do you think this works well?


ARISTA
ARISTA
ARISTA
ARISTA
ARISTA

# Traffic engineering: the classics

Key task of intradomain routing: optimize utilization

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: path choice vs. little state in routers

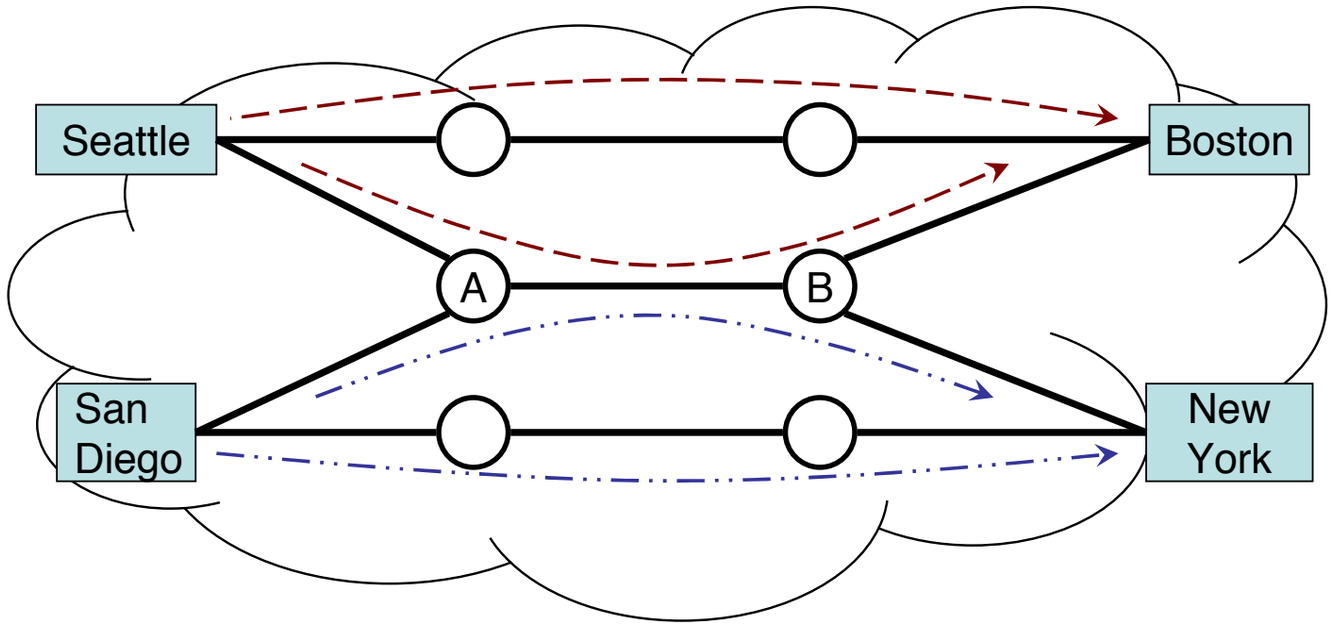
# TeXCP [Kandula et al 2005]

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

- 10 MPLS tunnels in implementation

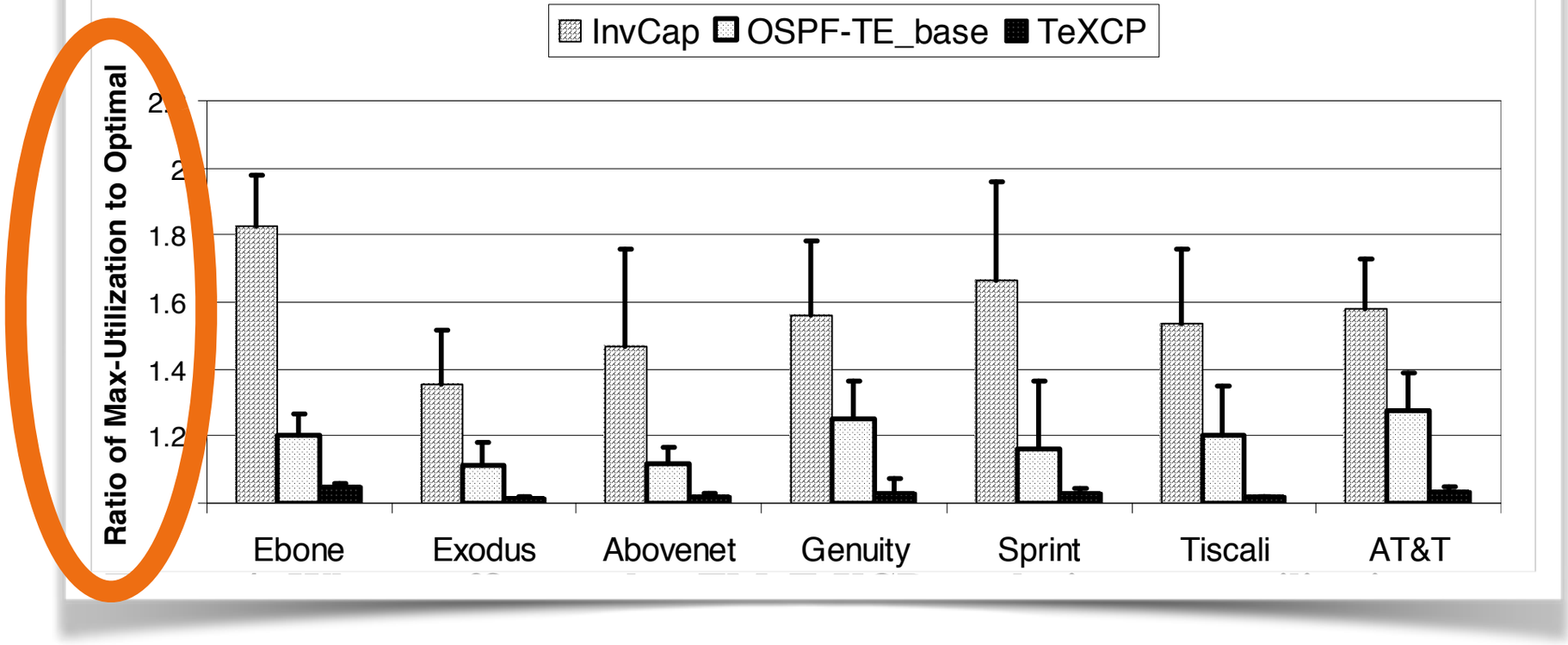
Dynamically at each ingress node:

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




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

# TeXCP results

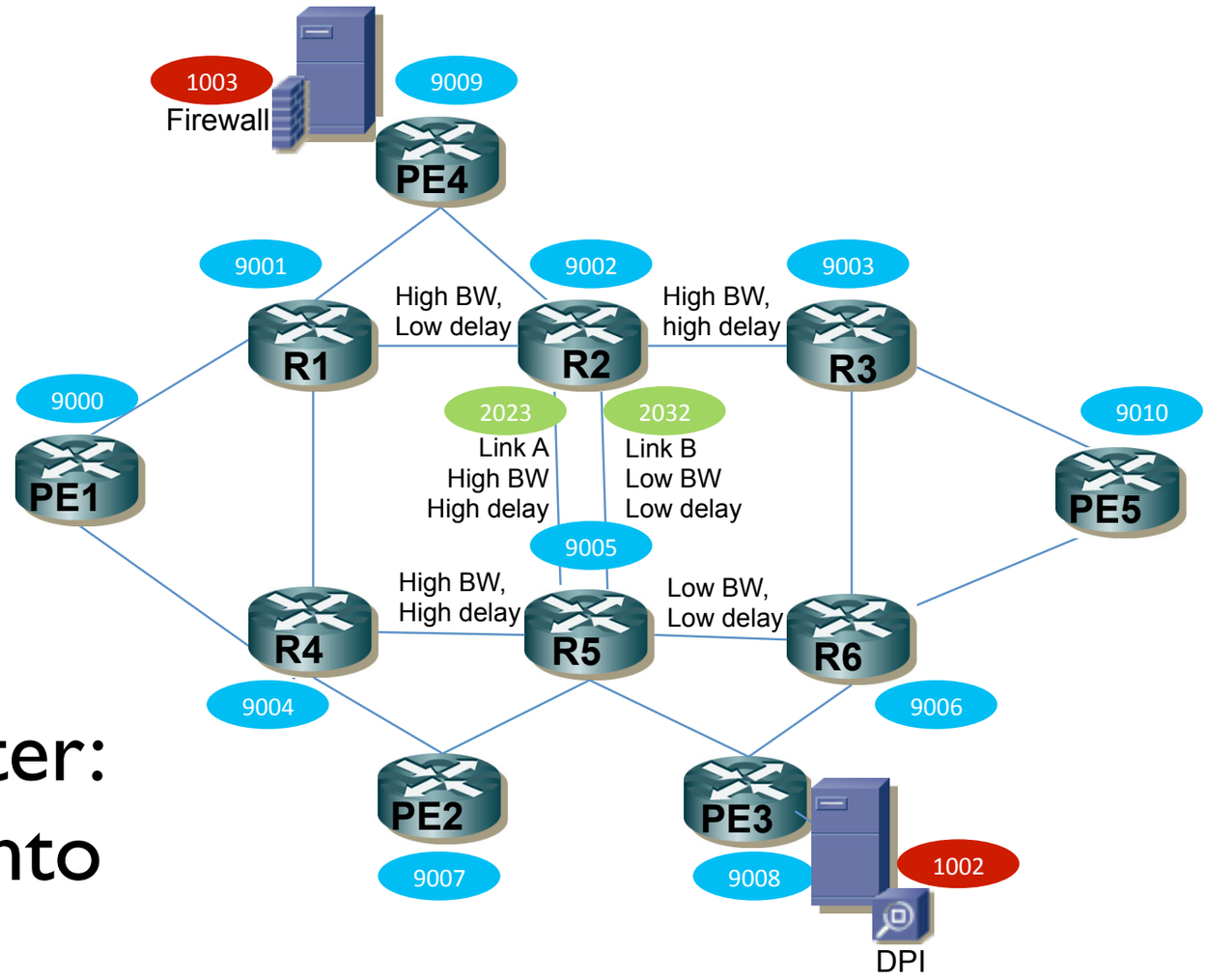



Q: In OSPF-TE, "Finding optimal link weights that minimize the max-utilization is NP-hard". Why is this harder than finding the best possible (non-OSPF) solution?

# Background: Segment Routing

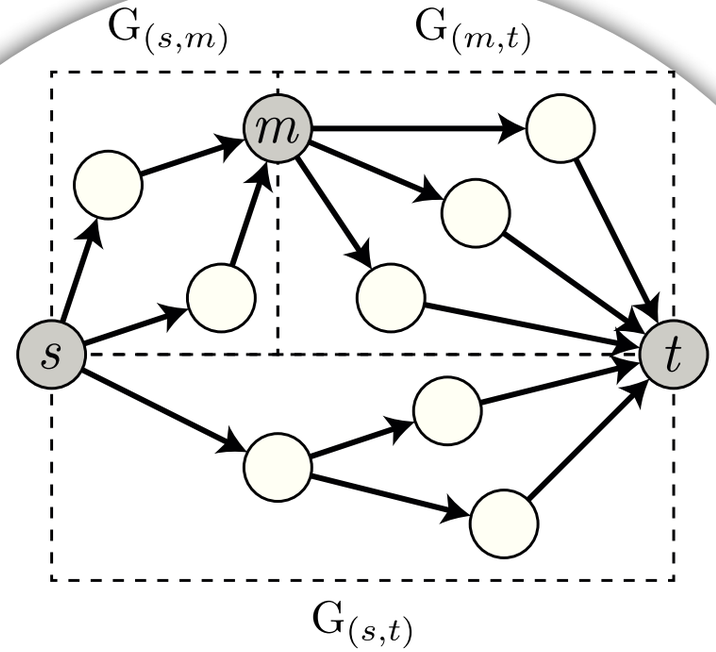
Idea: source routing by composing path segments

- 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

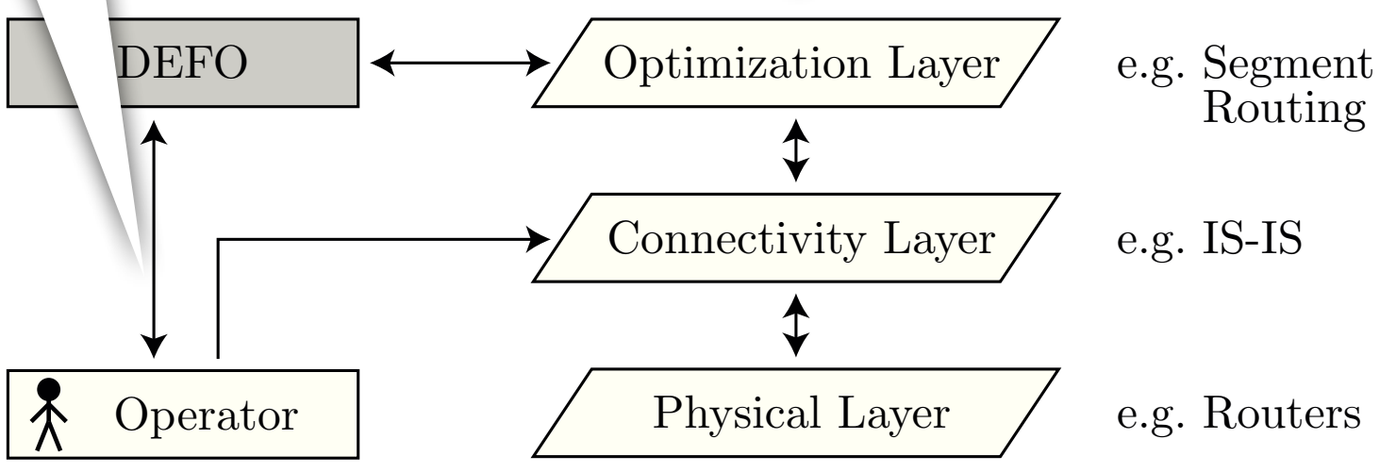


# DEFO [Hartert et al 2015]

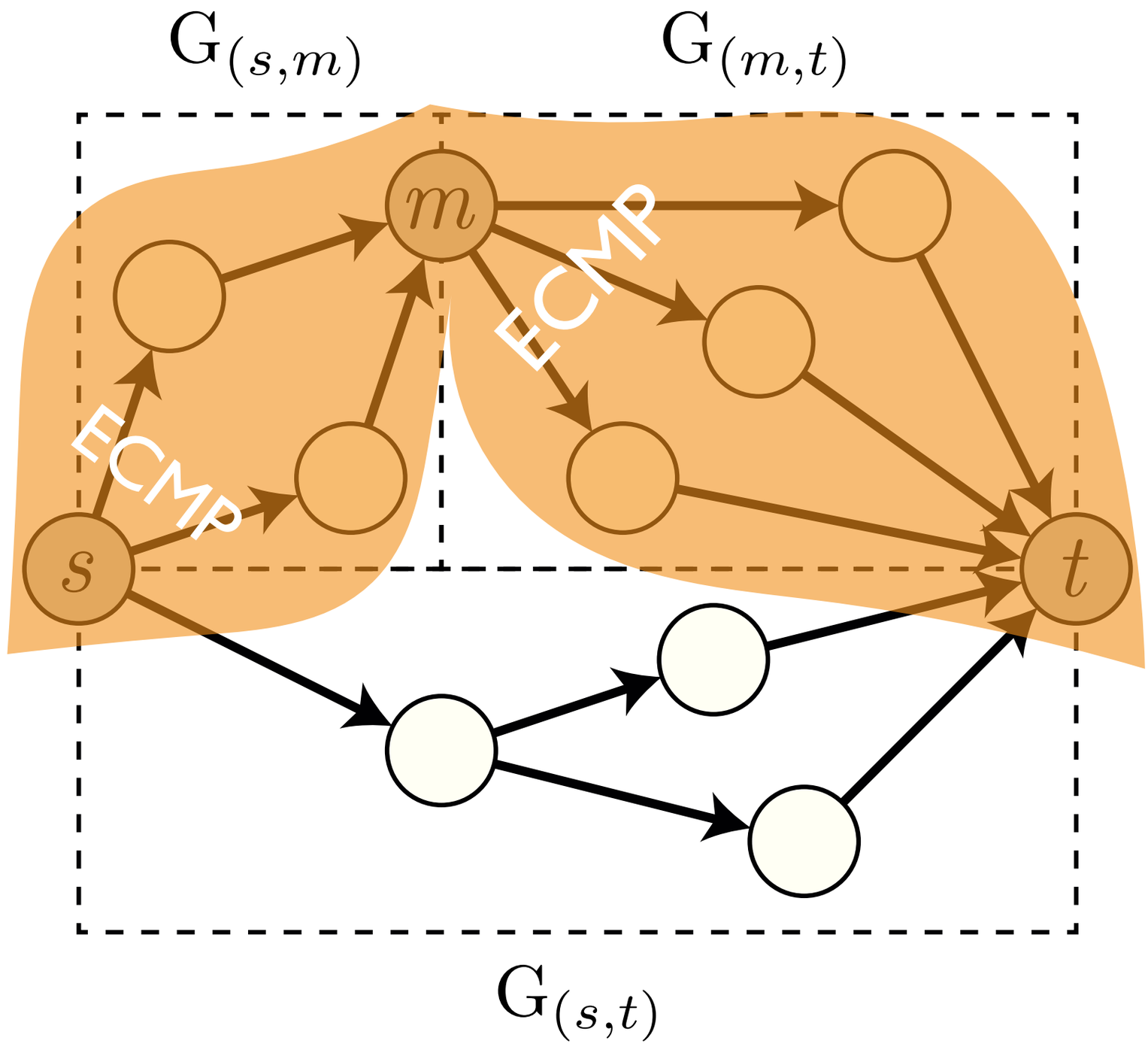
```
val goal = new Goal(topology){
 for(d<-Demands) add(d.deviations <= 2)
 for(l<-topology.links) add(l.load <= 0.9 l.capacity)
 minimize(MaxLoad)}
```




… for each ingress-egress traffic bundle



# DEFO [Hartert et al 2015]



# 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 proposals and assignments returned

Readings

- BGP routing policies in ISP networks (Caesar and Rexford, IEEE Network Magazine, Nov/Dec 2005)
- Anatomy of a Large European IXP (Ager et al., SIGCOMM 2012)