# Semi-Oblivious Traffic Engineering: The Road Not Taken

Praveen Kumar (Cornell)

Yang Yuan (Cornell)

Chris Yu (CMU)

Nate Foster (Cornell)

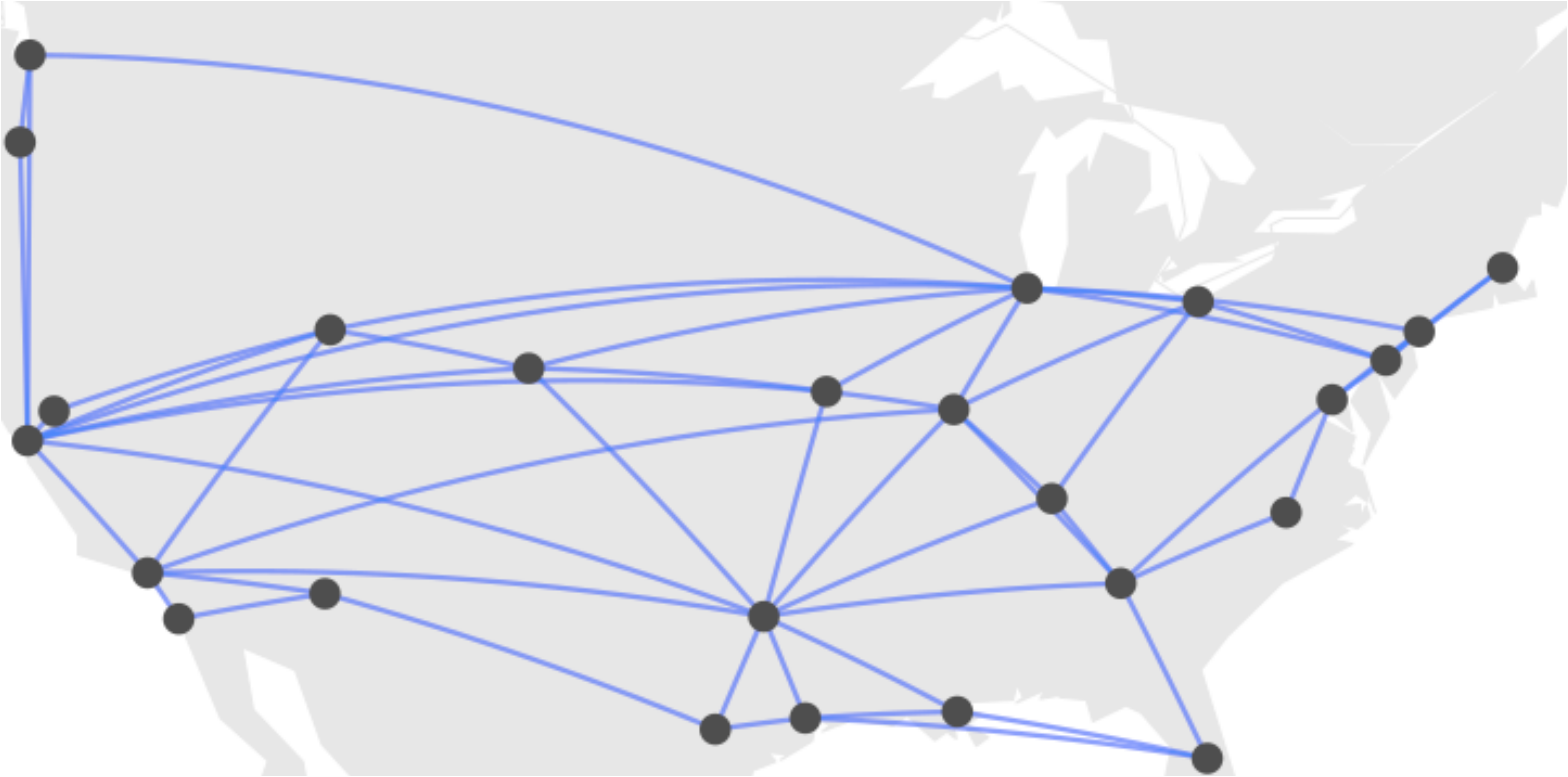
Robert Kleinberg (Cornell)

Petr Lapukhov (Facebook)

Chiun Lin Lim (Facebook)

Robert Soule (USI Lugano)

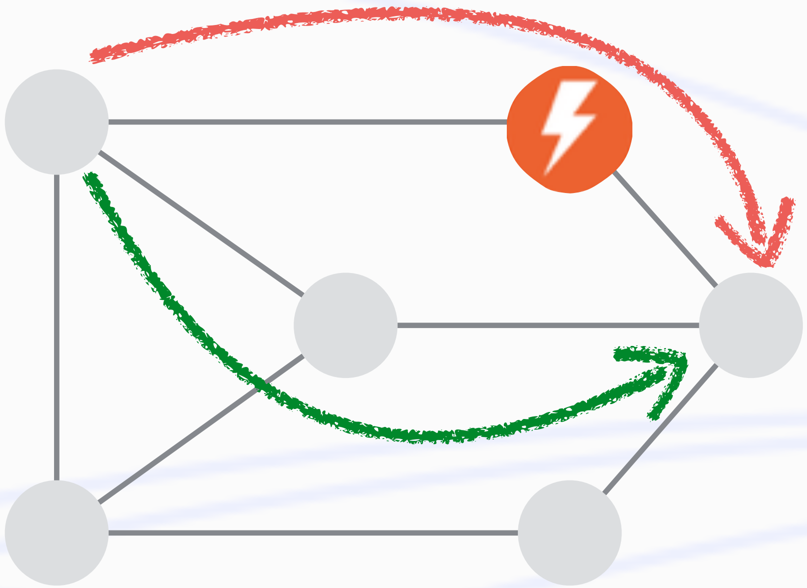
# WAN Traffic Engineering



# WAN Traffic Engineering

## Objectives

Performance



Robustness

Latency

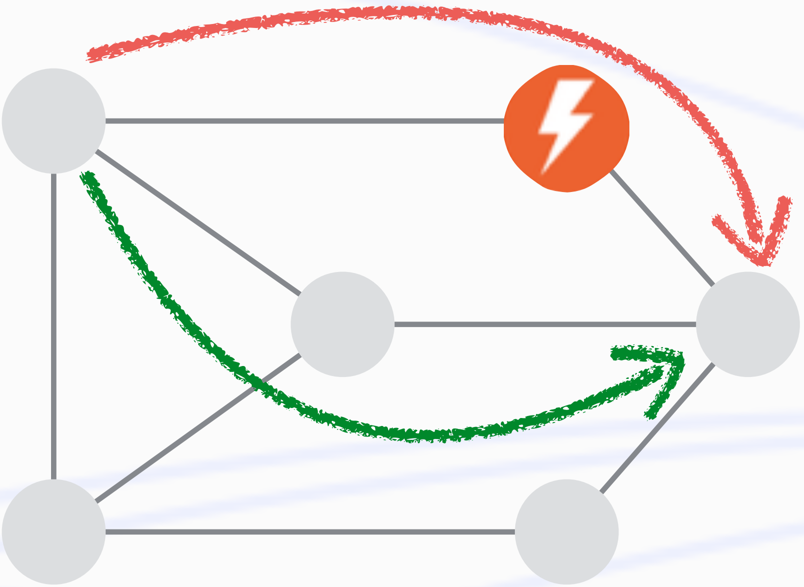
Operational simplicity

## Challenges

# WAN Traffic Engineering

## Objectives

Performance



Robustness

Latency

Operational simplicity

## Challenges

Unstructured topology

Heterogeneous capacity

Unexpected failures

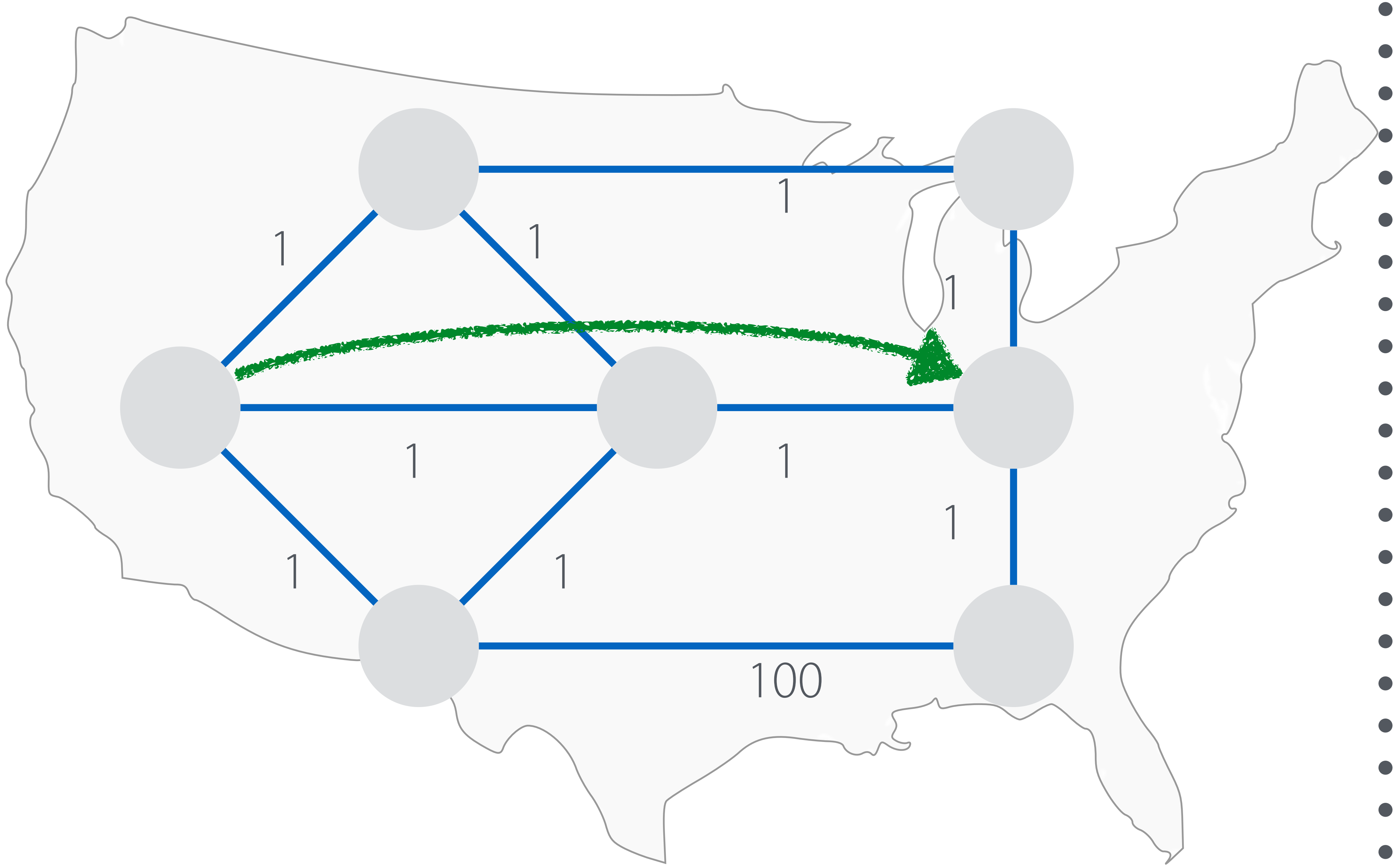
Misprediction & Traffic Bursts

Device limitations

Update overheads

# TE Approaches

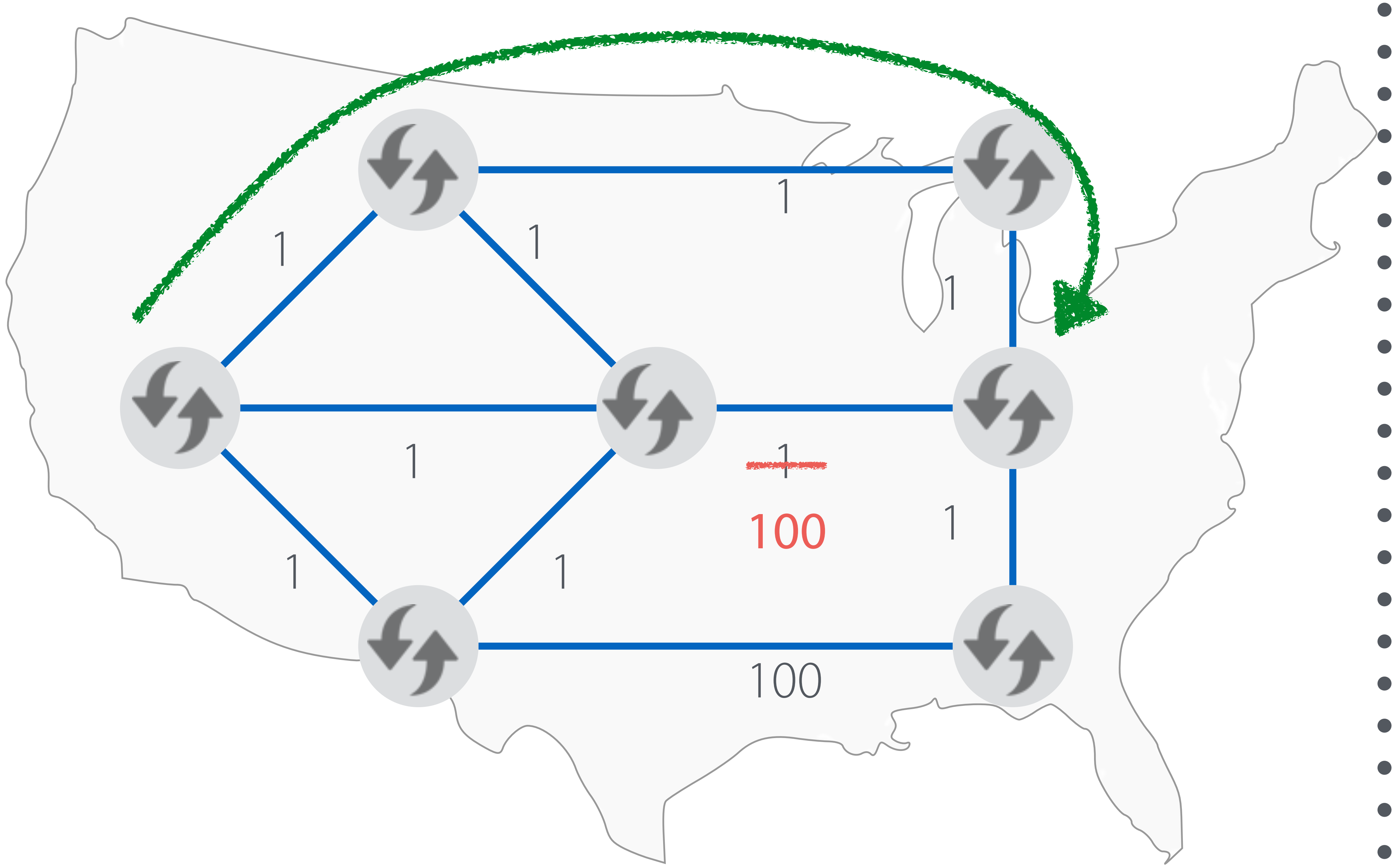
## Traditional Distributed



## SDN-Based Centralized

# TE Approaches

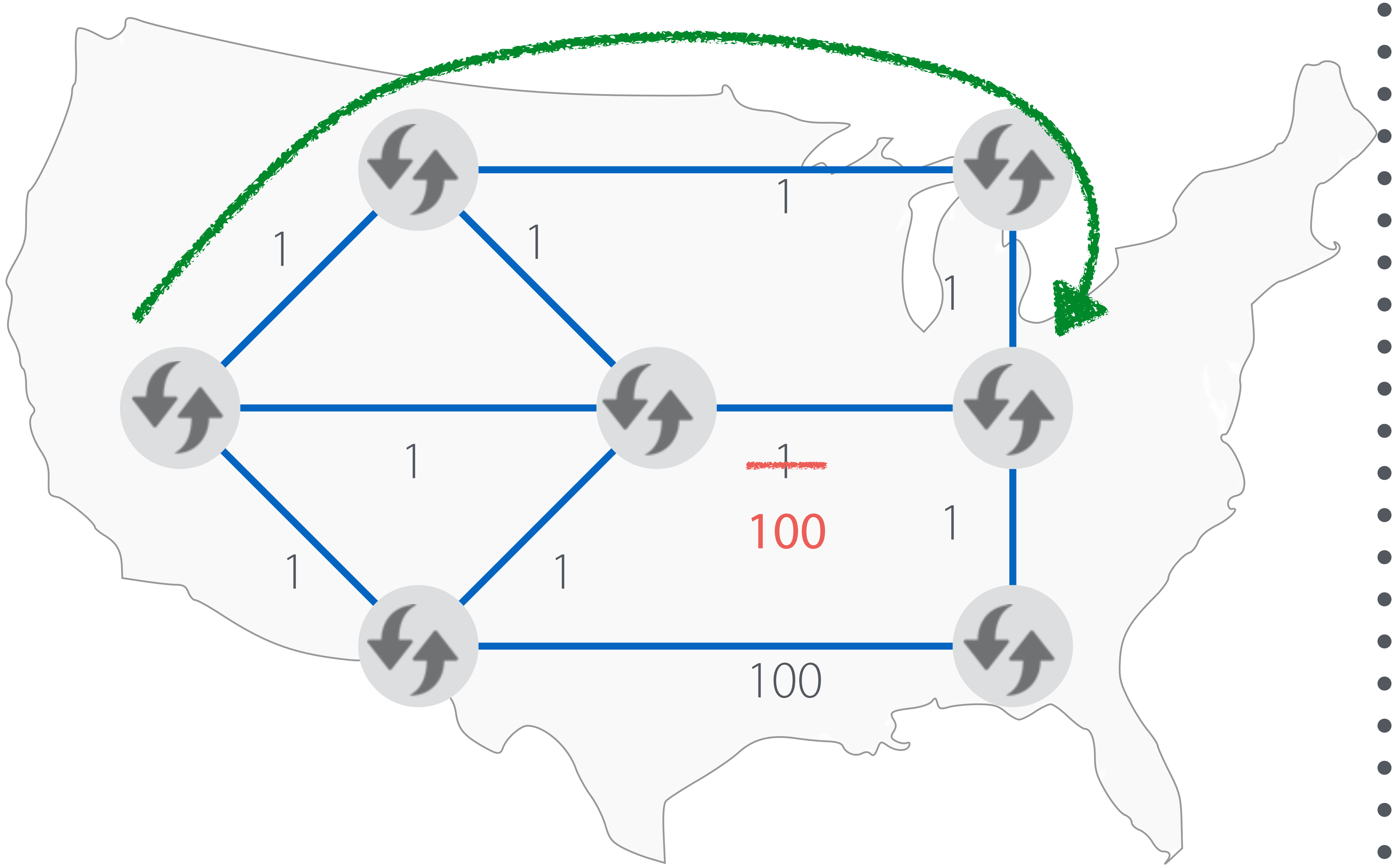
## Traditional Distributed



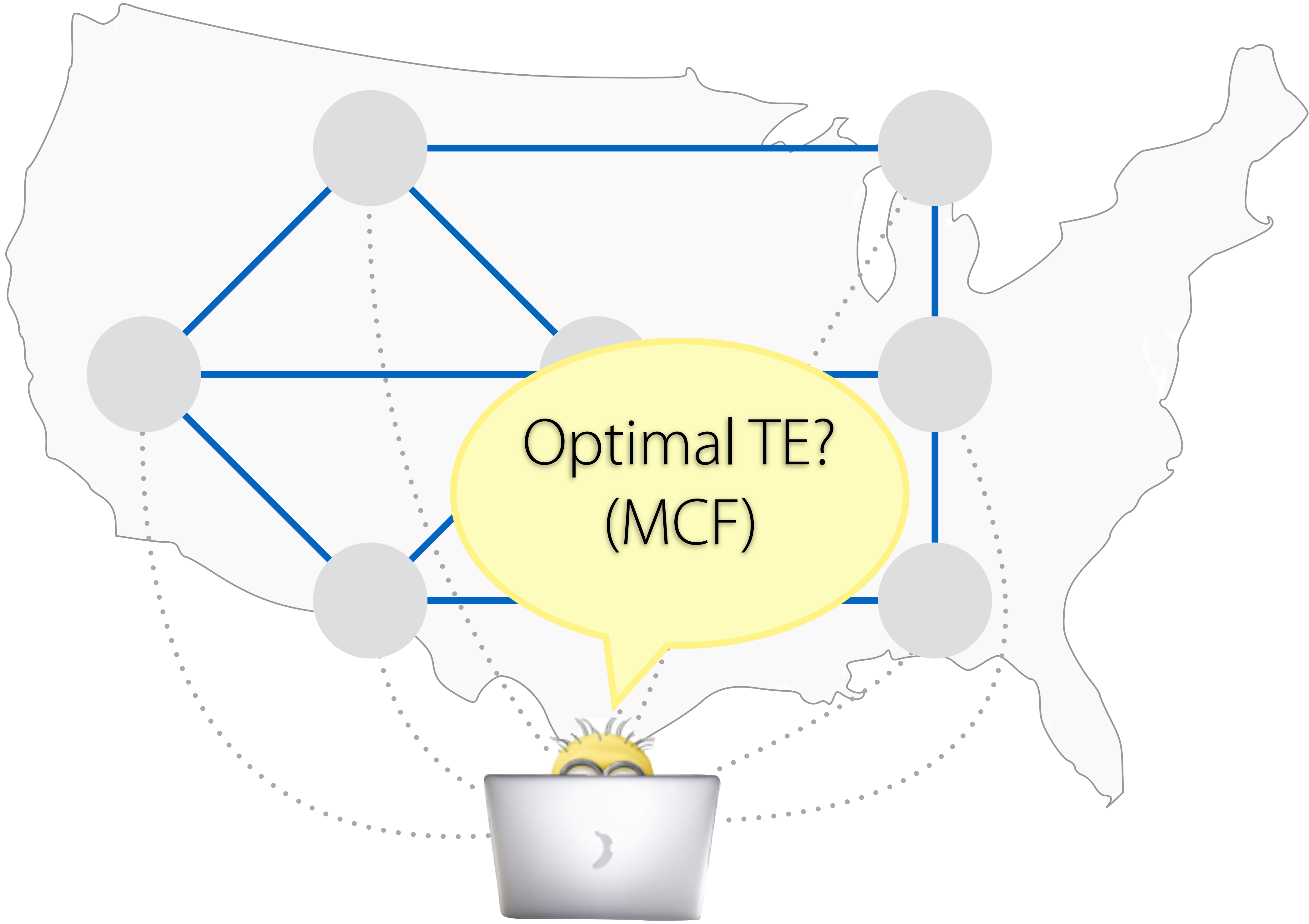
## SDN-Based Centralized

# TE Approaches

## Traditional Distributed

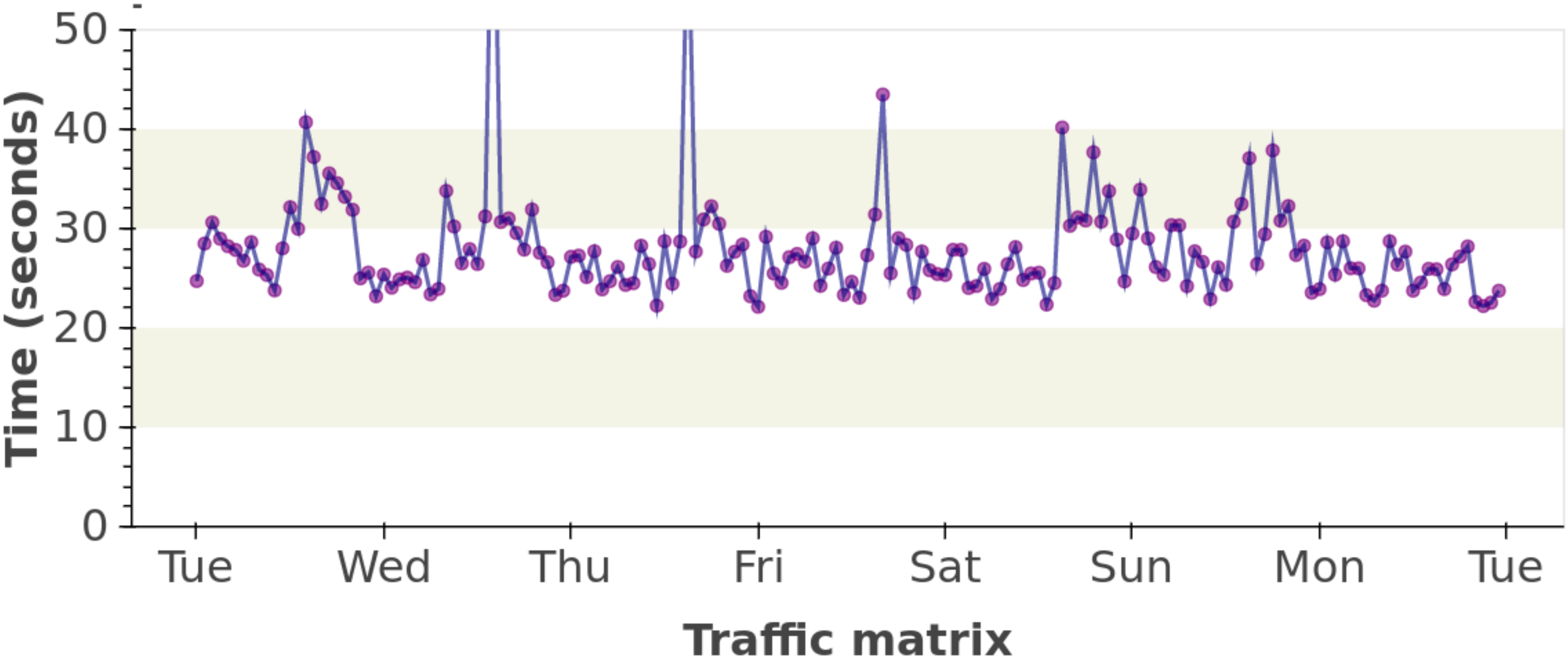


## SDN-Based Centralized



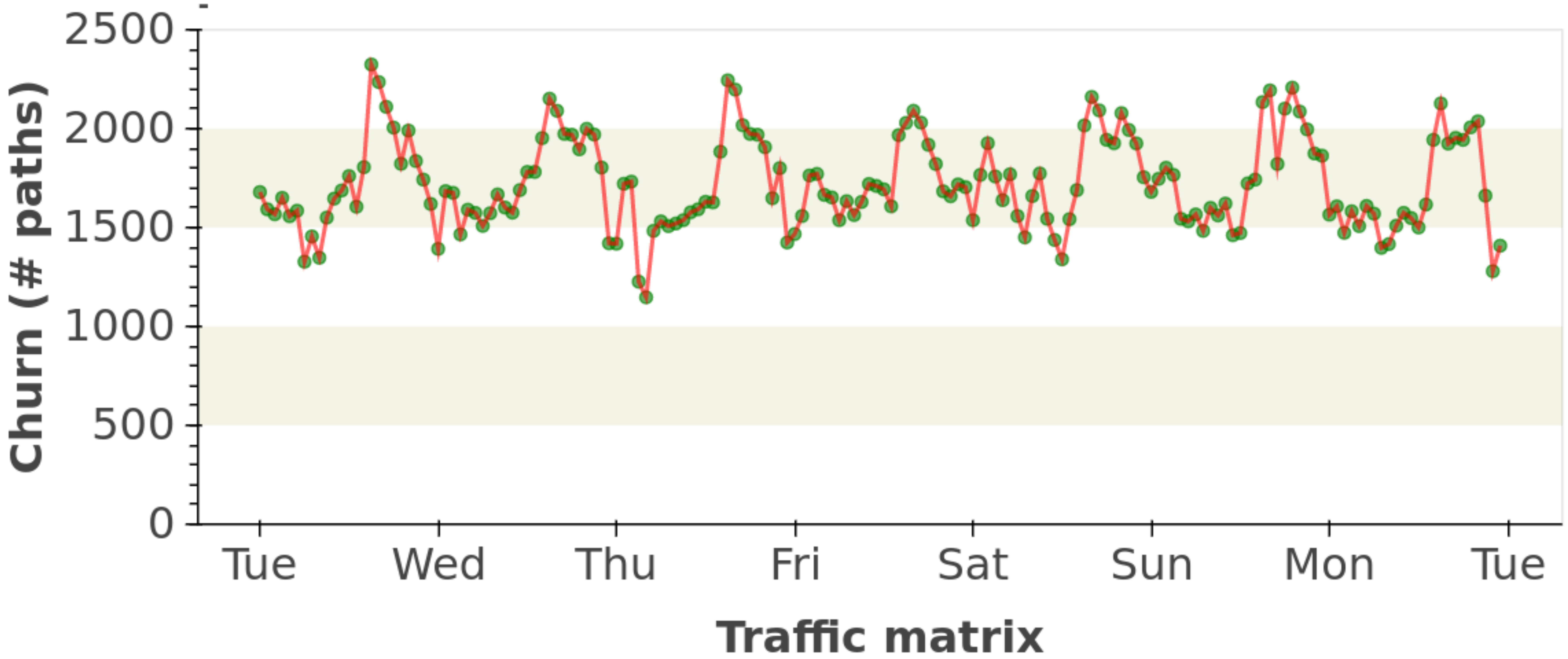
# Operational Cost of Optimality

## Solver Time

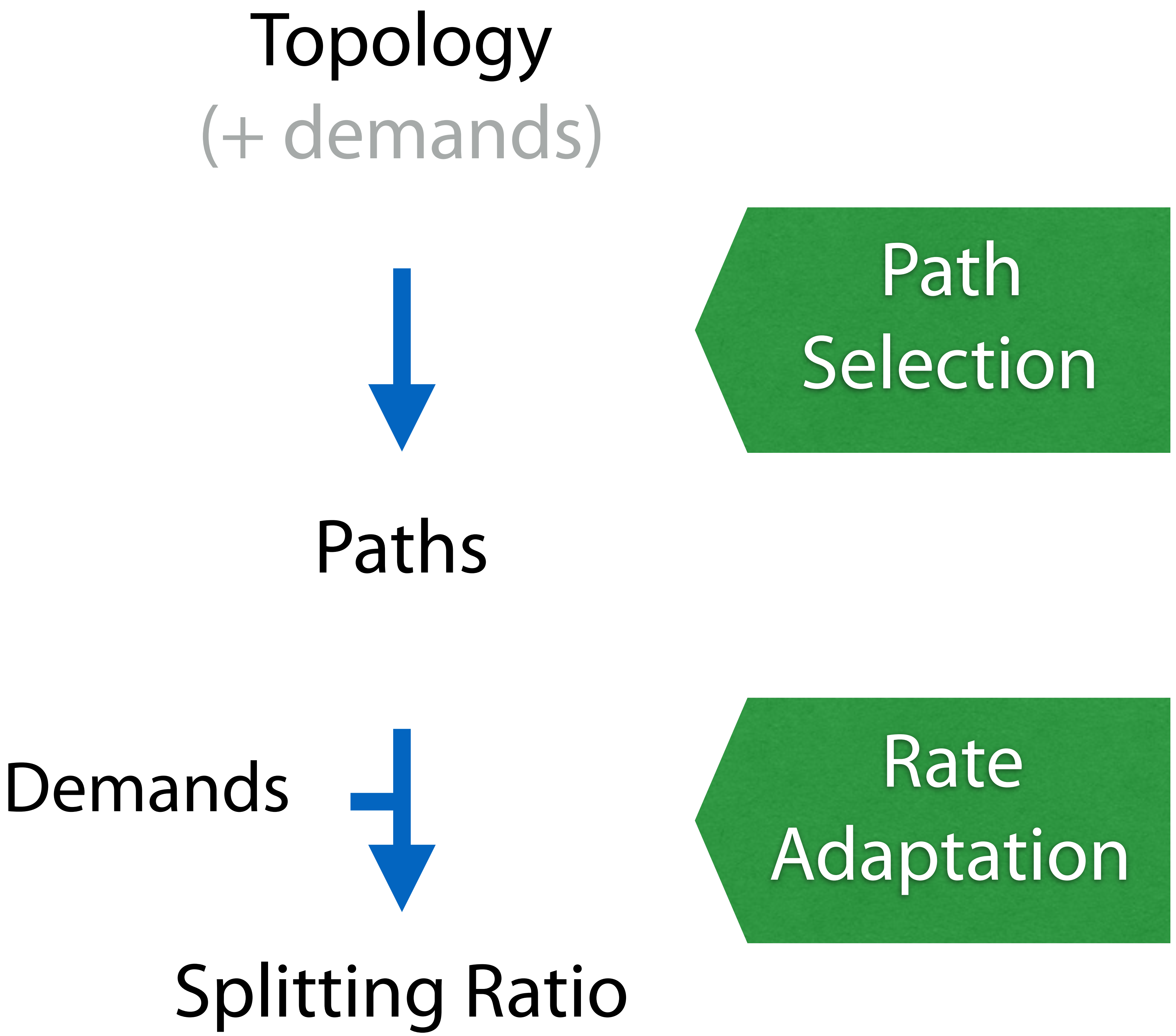


# Operational Cost of Optimality

## Path Churn



# Towards a Practical Model



Topology

(+ demands)

Path Selection

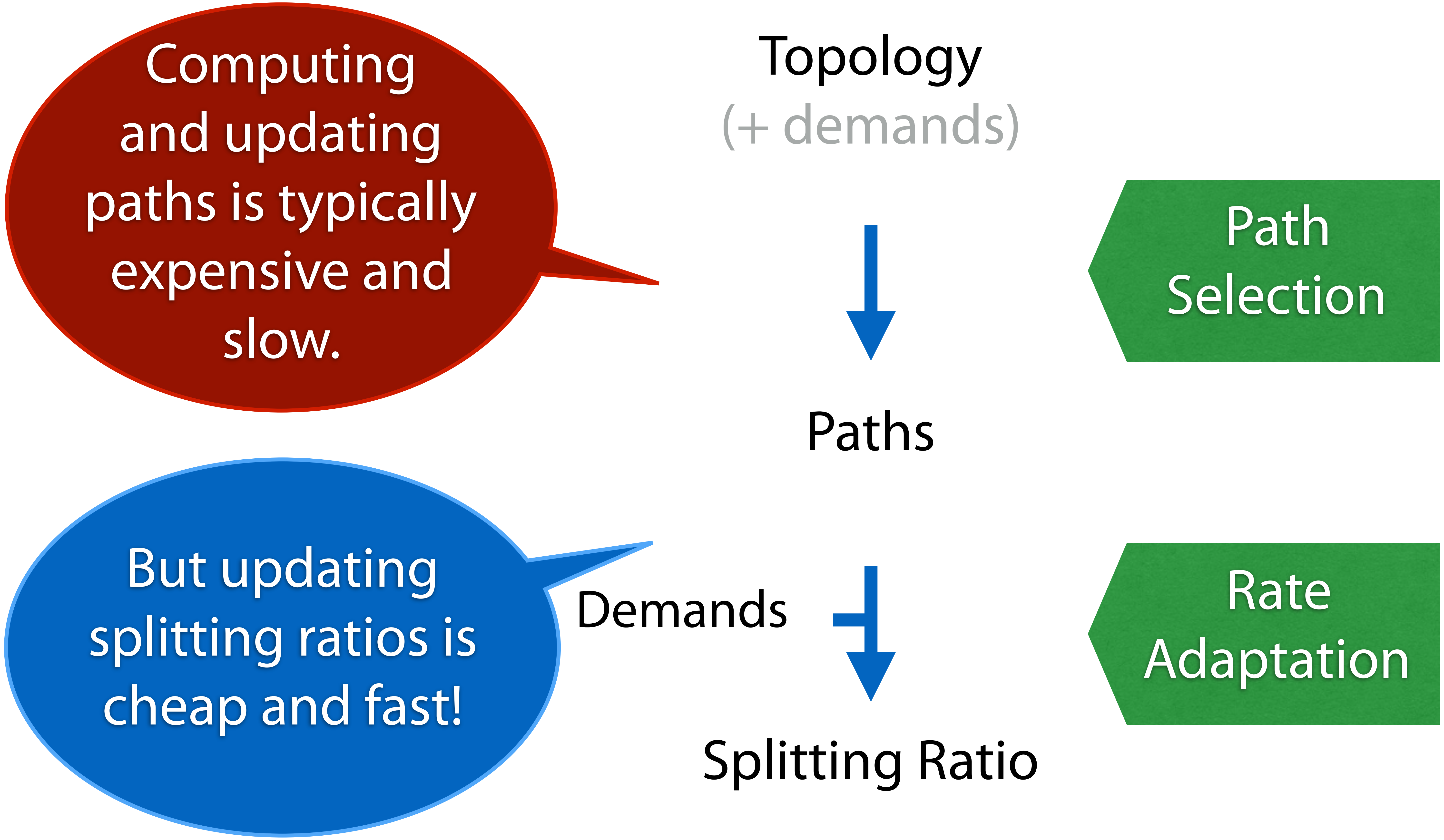
Paths

Demands

Rate Adaptation

Splitting Ratio

# Towards a Practical Model



Computing and updating paths is typically expensive and slow.

But updating splitting ratios is cheap and fast!

Topology

(+ demands)

Paths

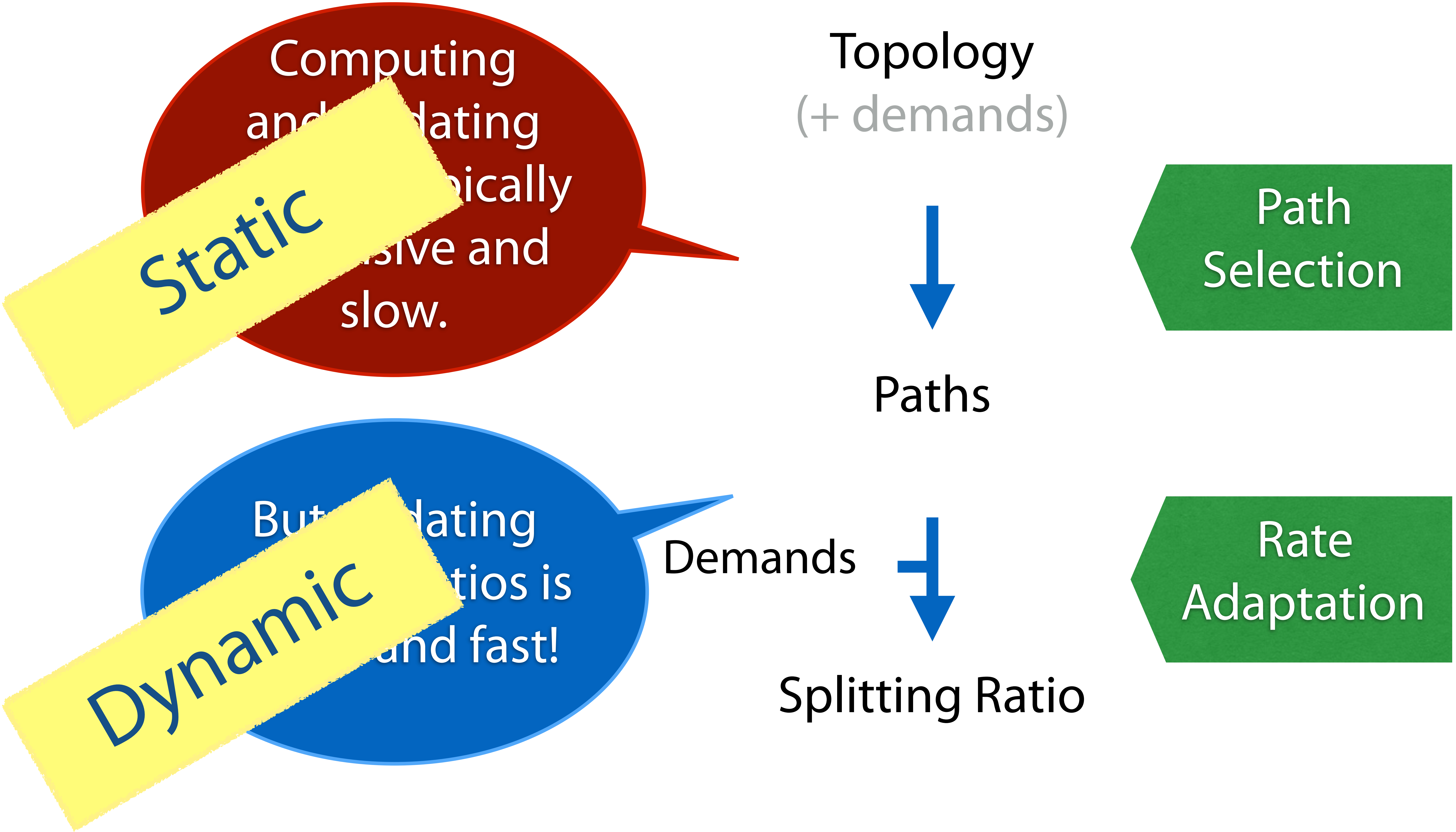
Demands

Splitting Ratio

Path Selection

Rate Adaptation

# Towards a Practical Model



Computing
and dating
ically
sive and
slow.

Static

But dating
tios is
and fast!

Dynamic

Topology
(+ demands)

Paths

Demands

Splitting Ratio

Path Selection

Rate Adaptation

# Path Selection Challenges

- Selecting a good set of paths is tricky!
  - **Route** the demands (ideally, with competitive **latency**)
  - React to **changes in demands** (diurnal changes, traffic bursts, etc.)
  - Be robust under **mis-prediction** of demands
  - Have sufficient extra capacity to route demands in presence of **failures**
  - …

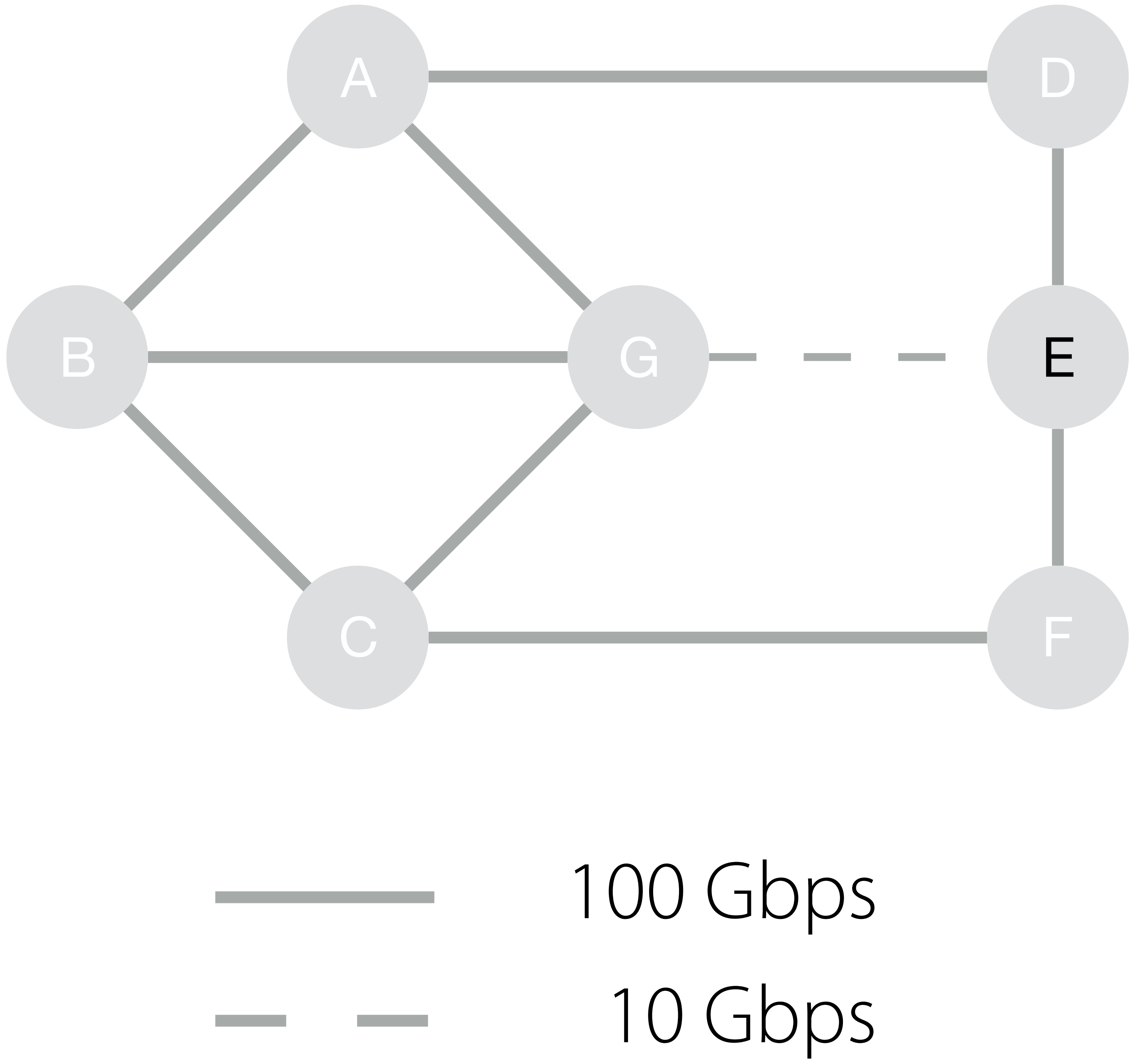
# Approach

A static set of cleverly-constructed paths can provide near-optimal performance and robustness!

Desired path properties:

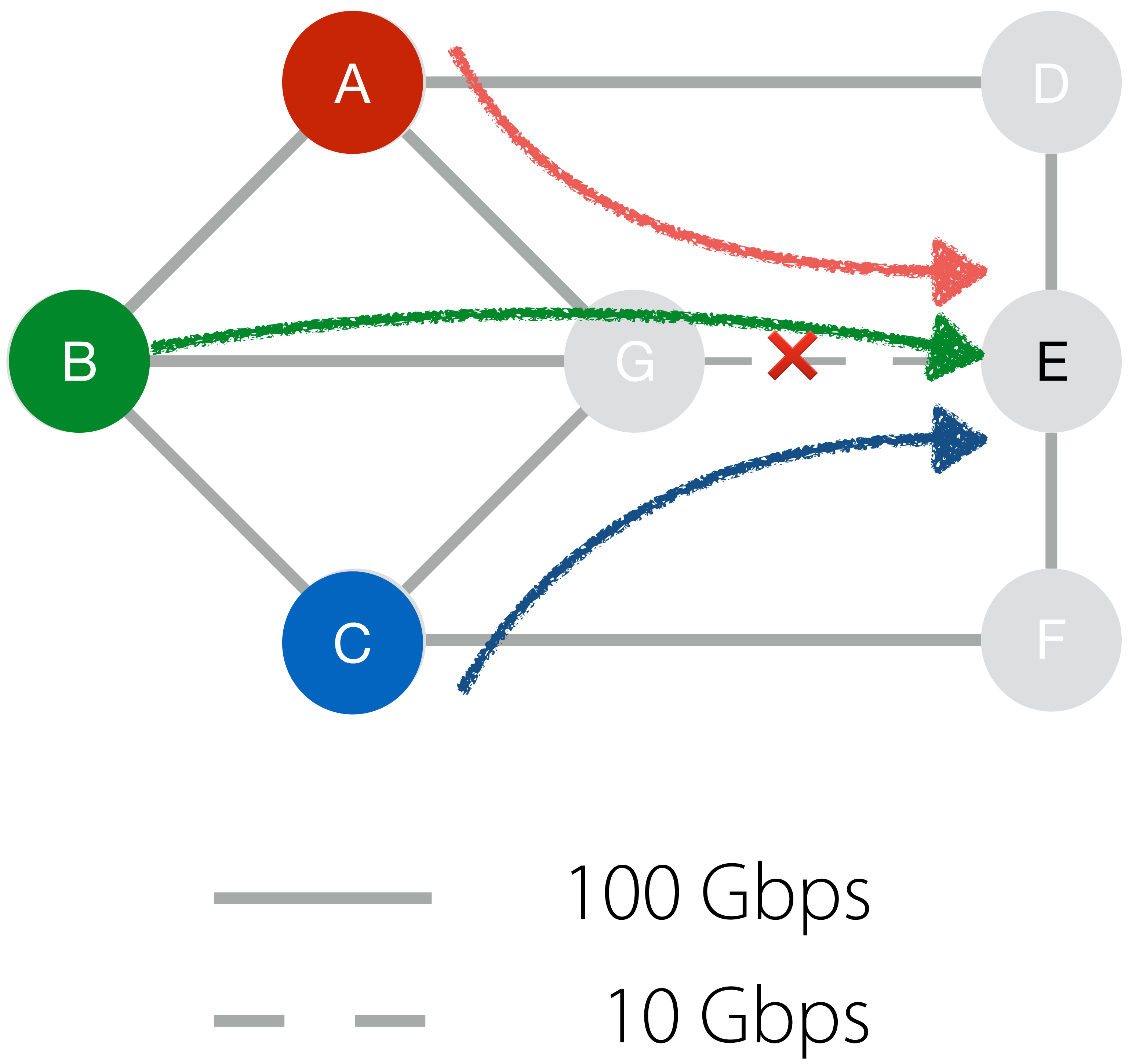
- ***Low stretch*** for minimizing latency
- ***High diversity*** for ensuring robustness
- ***Good load balancing*** for performance
  - Capacity aware
  - Globally optimized

# Path Properties: Capacity Aware



- Traditional approaches to routing based on shortest paths (e.g., ECMP, KSP) are generally not capacity aware

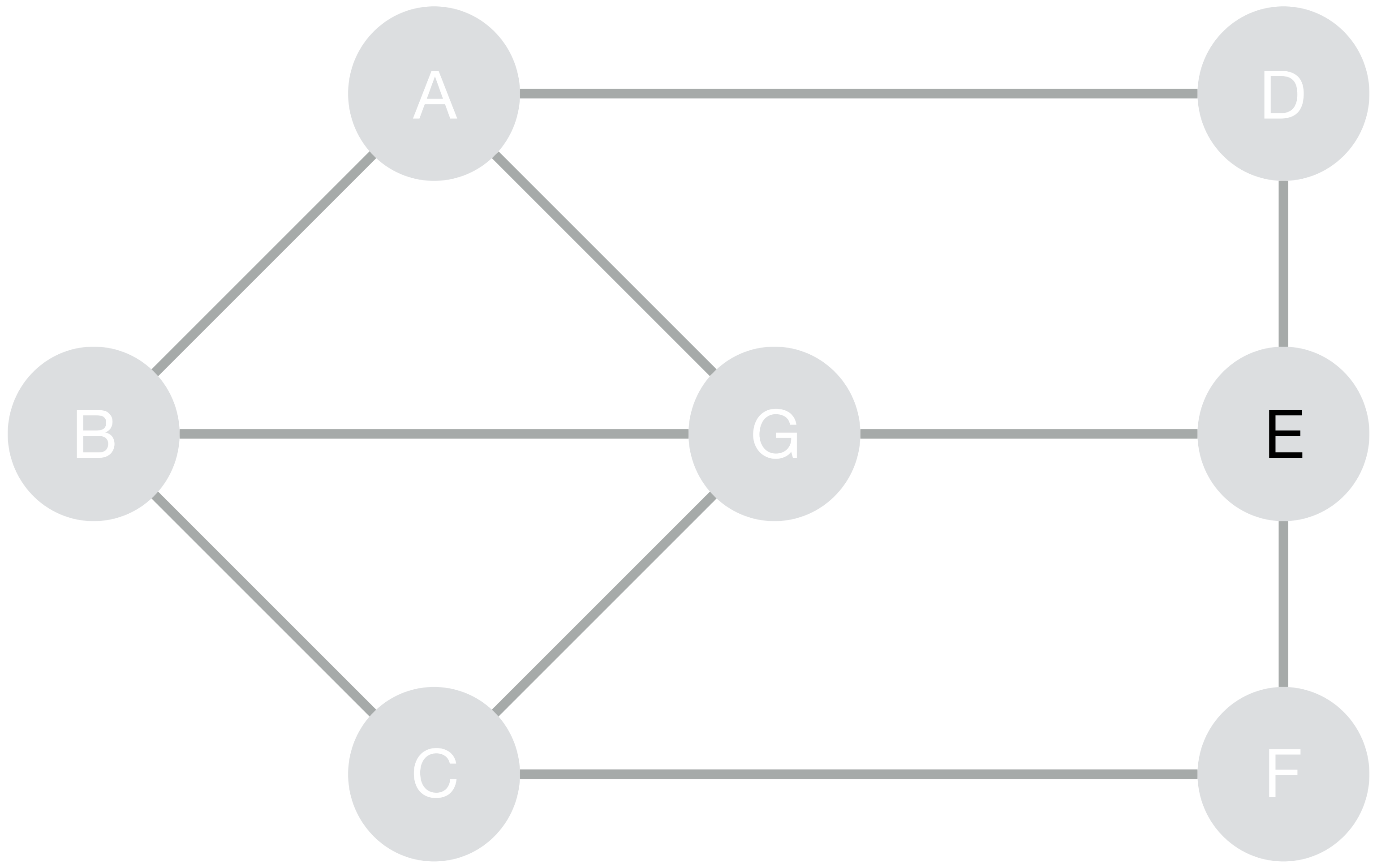
# Path Properties: Capacity Aware



- Traditional approaches to routing based on shortest paths (e.g., ECMP, KSP) are generally not capacity aware

# Path Properties: Globally Optimal

Other approaches based on greedy algorithms are capacity aware, but are still not globally optimal

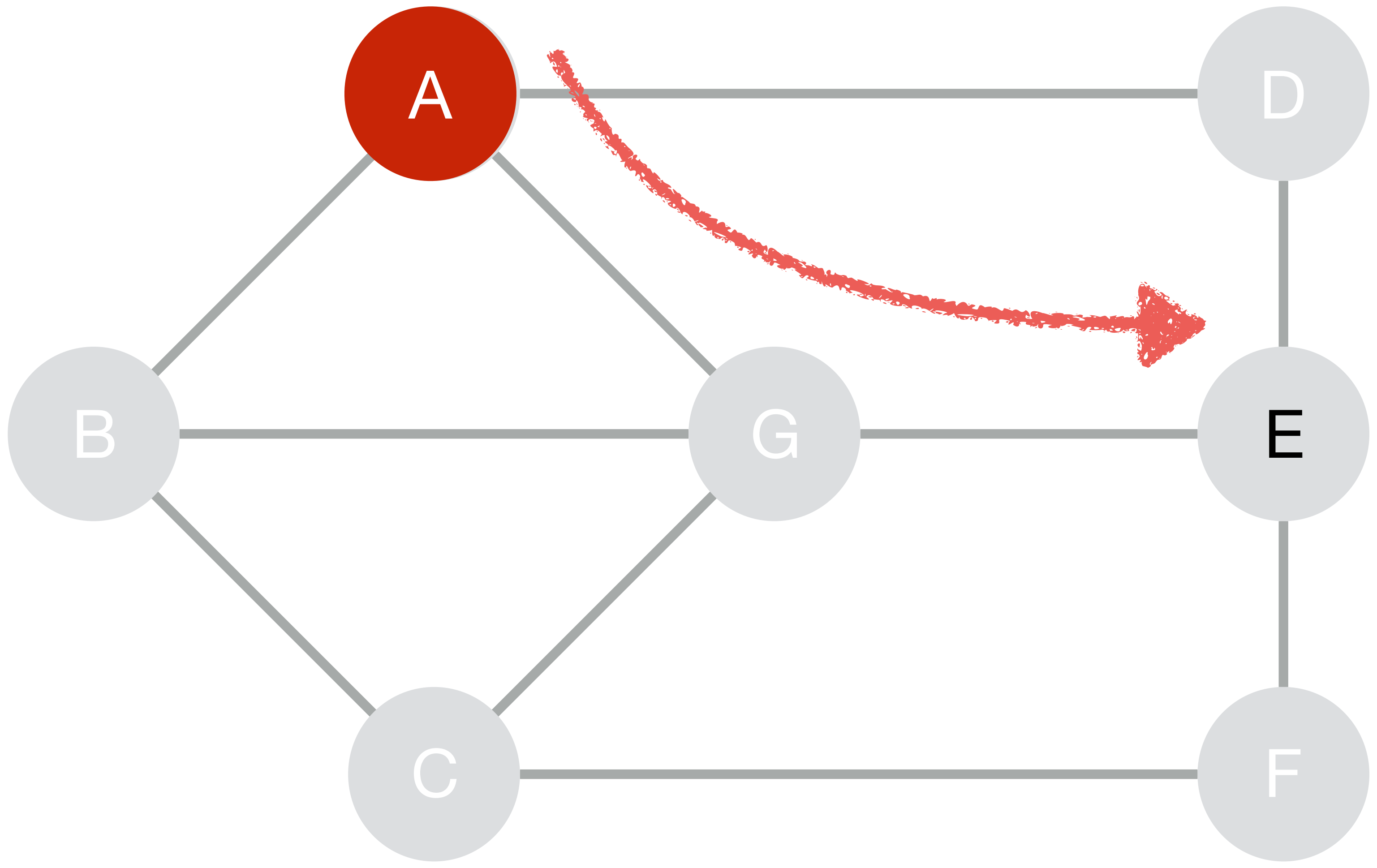


CSPF

Globally optimal

# Path Properties: Globally Optimal

Other approaches based on greedy algorithms are capacity aware, but are still not globally optimal

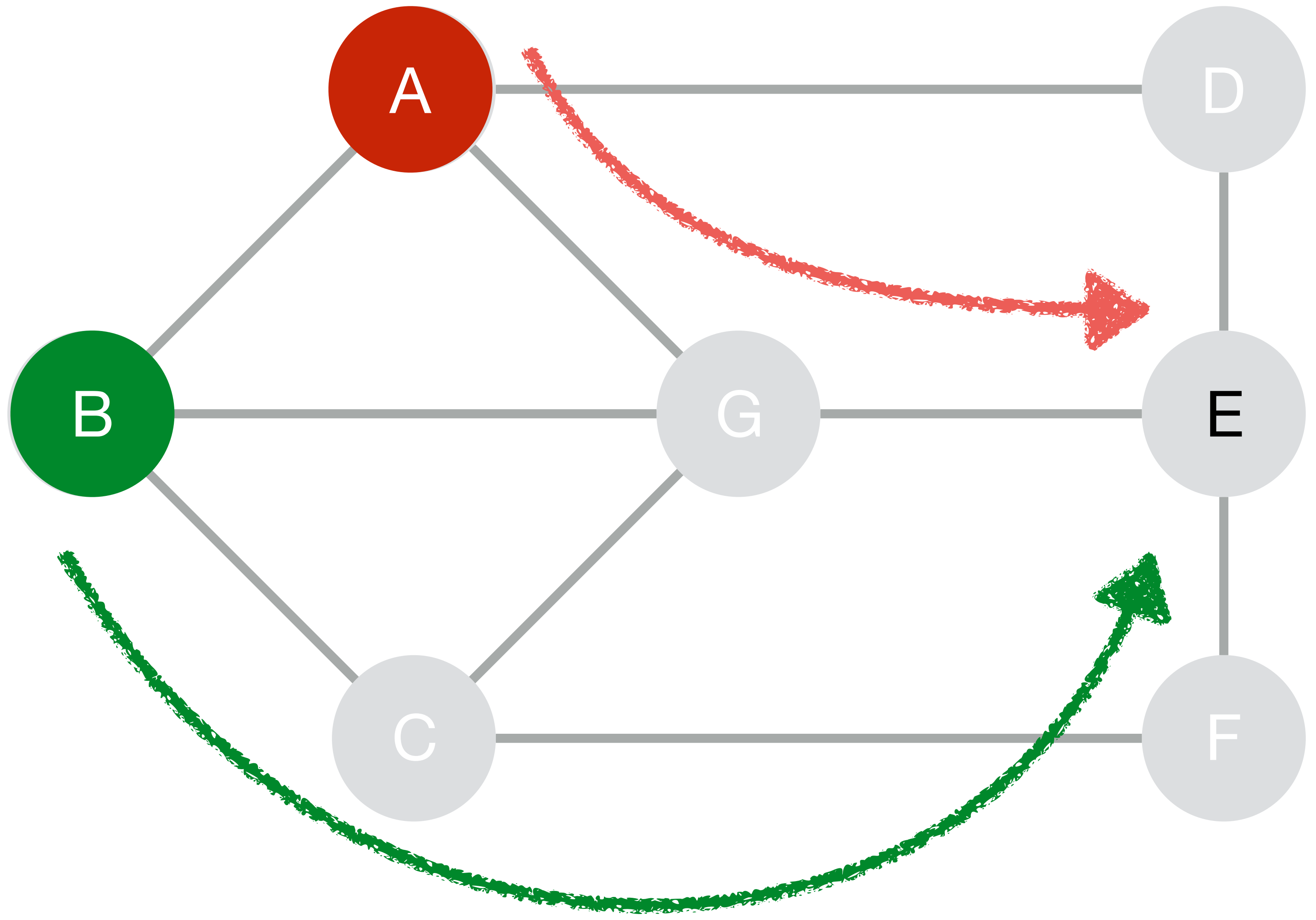


CSPF

Globally optimal

# Path Properties: Globally Optimal

Other approaches based on greedy algorithms are capacity aware, but are still not globally optimal

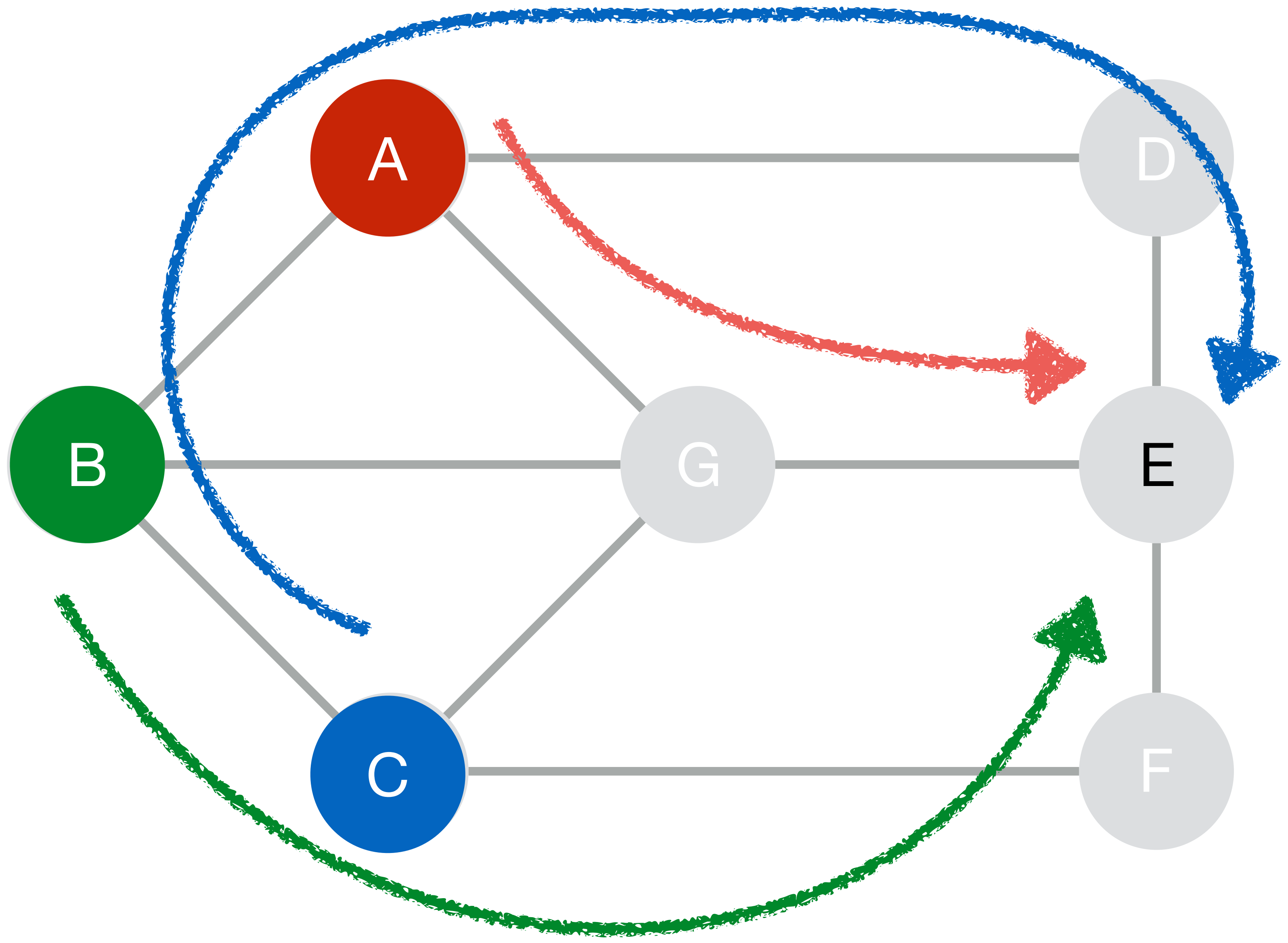


CSPF

Globally optimal

# Path Properties: Globally Optimal

Other approaches based on greedy algorithms are capacity aware, but are still not globally optimal

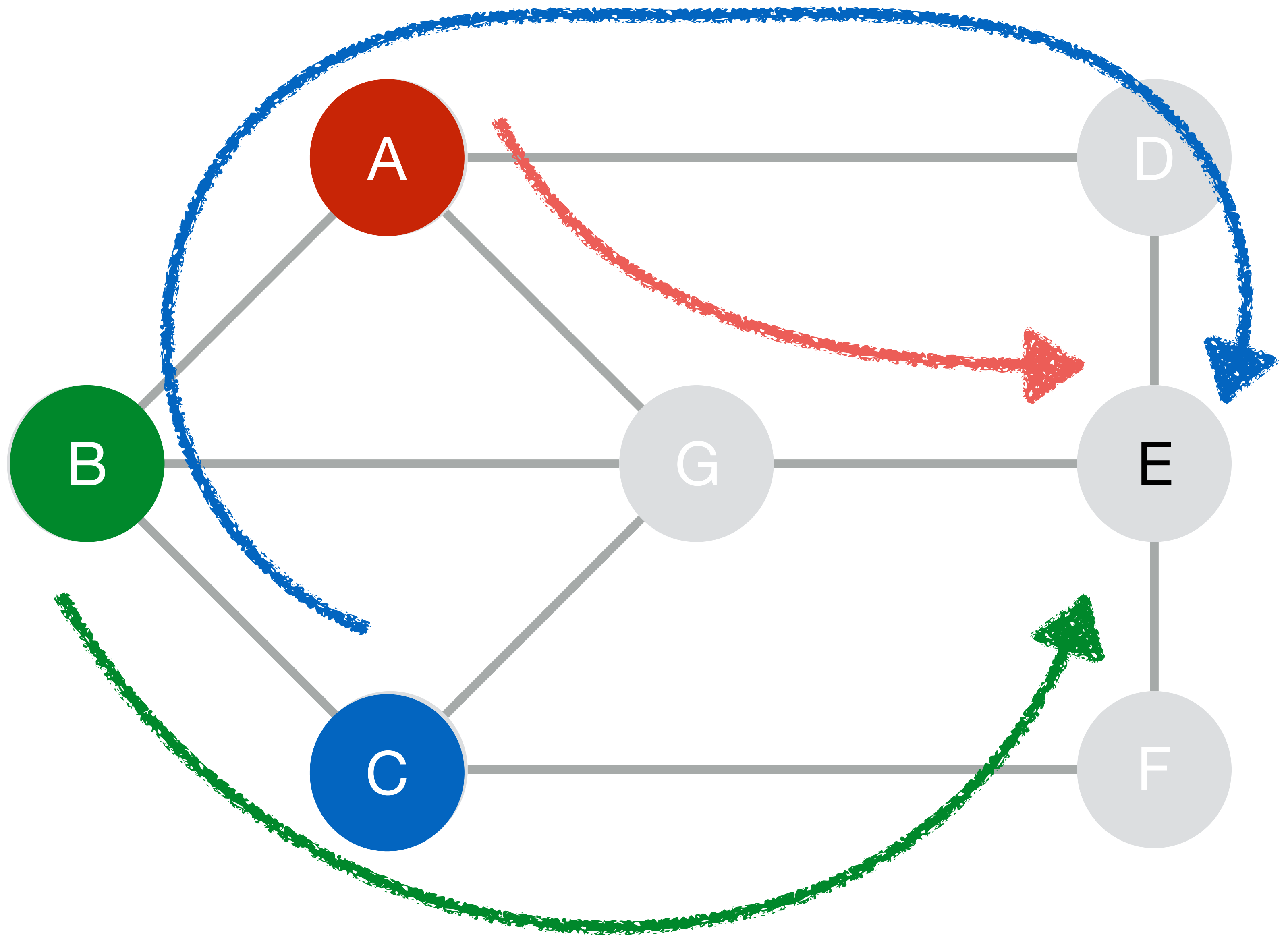


CSPF

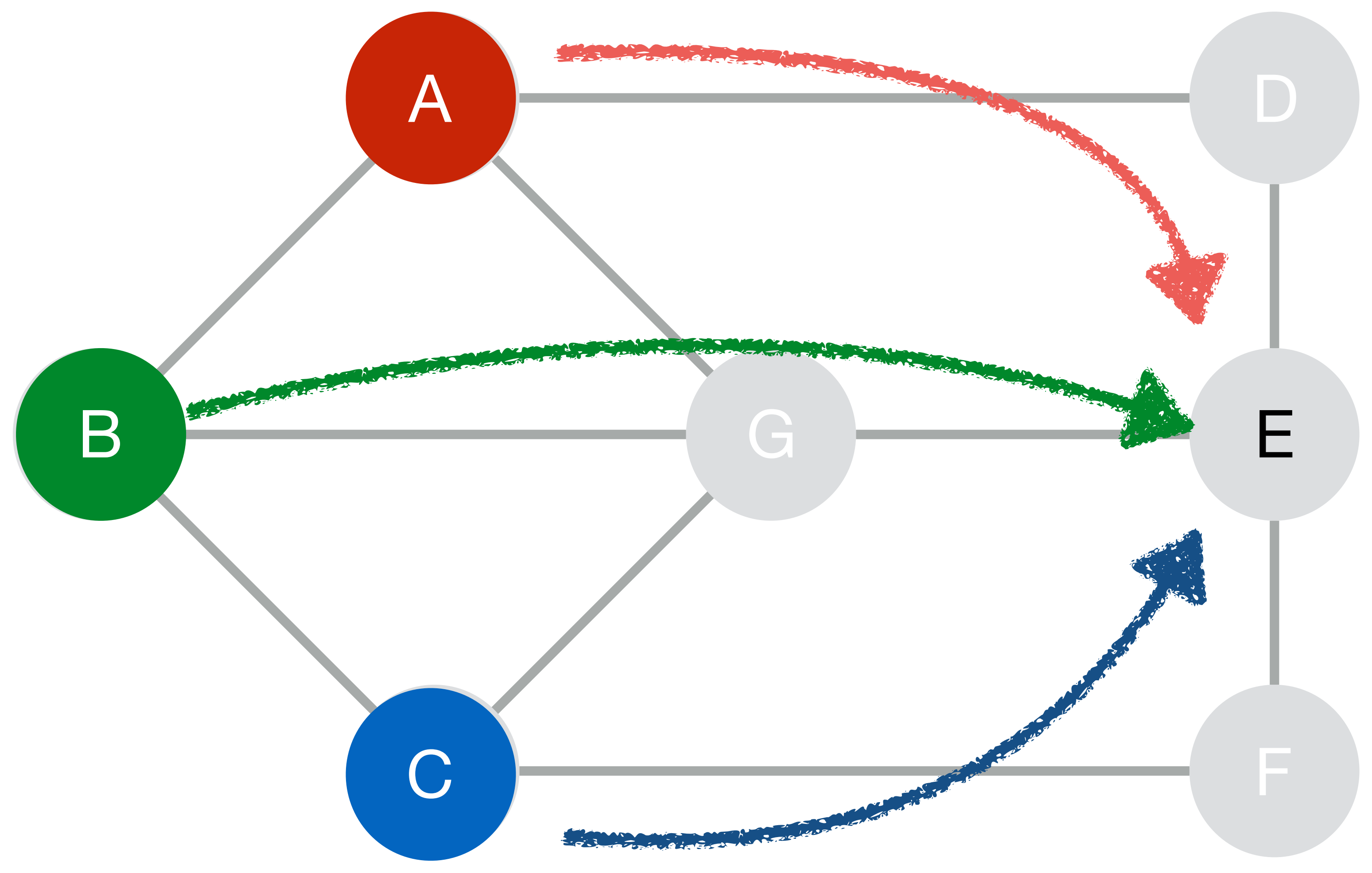
Globally optimal

# Path Properties: Globally Optimal

Other approaches based on greedy algorithms are capacity aware, but are still not globally optimal



CSPF



Globally optimal

# Path Selection

| Algorithm | Load balanced | | Diverse | Low-stretch |
|---|---|---|---|---|
| | Capacity aware | Globally Optimized | | |
| SPF / ECMP | ✗ | ✗ | ✗ | ✓ |
| CSPF | ✓ | ✗ | ✗ | ✓ |
| k-shortest paths | ✗ | ✗ | ? | ✓ |
| Edge-disjoint KSP | ✗ | ✗ | ✓ | ✓ |
| MCF | ✓ | ✓ | ✗ | ✗ |
| VLB | ✗ | ✗ | ✓ | ✗ |
| B4 | ✓ | ✓ | ✗ | ? |

# Path Selection

| **Algorithm** | Load balanced | | Diverse | Low-stretch |
|---|---|---|---|---|
| | Capacity aware | Globally Optimized | | |
| SPF / ECMP | ✗ | ✗ | ✗ | ✓ |
| CSPF | ✓ | ✗ | ✗ | ✓ |
| k-shortest paths | ✗ | ✗ | ? | ✓ |
| Edge-disjoint KSP | ✗ | ✗ | ✓ | ✓ |
| MCF | ✓ | ✓ | ✗ | ✗ |
| VLB | ✗ | ✗ | ✓ | ✗ |
| B4 | ✓ | ✓ | ✗ | ? |

# Path Selection

| **Algorithm** | Load balanced | | Diverse | Low-stretch |
|---|---|---|---|---|
| | Capacity aware | Globally Optimized | | |
| SPF / ECMP | ✗ | ✗ | ✗ | ✓ |
| CSPF | ✓ | ✗ | ✗ | ✓ |
| k-shortest paths | ✗ | ✗ | ? | ✓ |
| Edge-disjoint KSP | ✗ | ✗ | ✓ | ✓ |
| MCF | ✓ | ✓ | ✗ | ✗ |
| VLB | ✗ | ✗ | ✓ | ✗ |
| B4 | ✓ | ✓ | ✗ | ? |

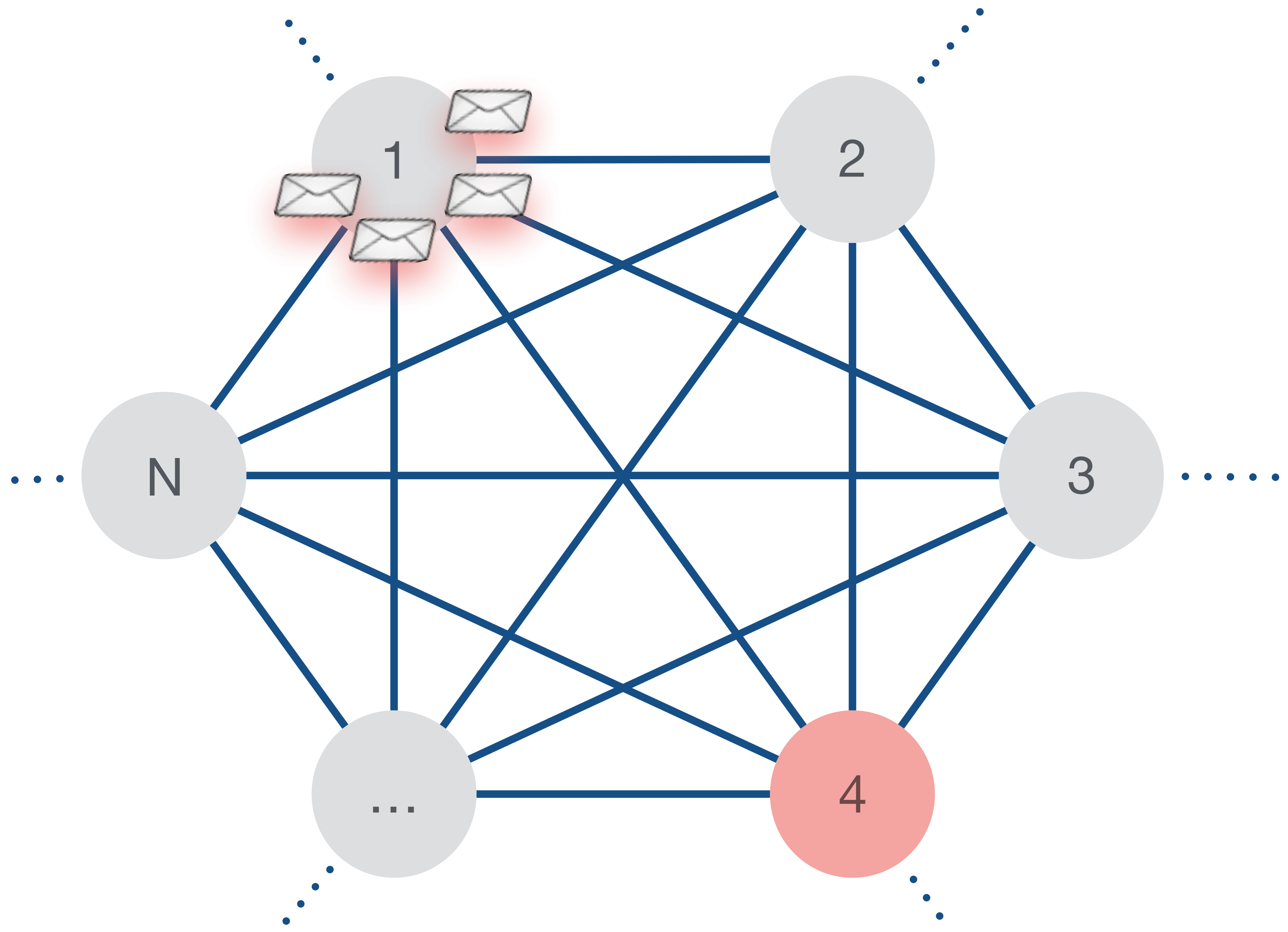
# Path Selection

| Algorithm | Load balanced | | Diverse | Low-stretch |
|---|---|---|---|---|
| | Capacity aware | Globally Optimized | | |
| SPF / ECMP | ✗ | ✗ | ✗ | ✓ |
| CSPF | ✓ | ✗ | ✗ | ✓ |
| k-shortest paths | ✗ | ✗ | ? | ✓ |
| Edge-disjoint KSP | ✗ | ✗ | ✓ | ✓ |
| MCF | ✓ | ✓ | ✗ | ✗ |
| VLB | ✗ | ✗ | ✓ | ✗ |
| B4 | ✓ | ✓ | ✗ | ? |

# Oblivious Routing

# VLB

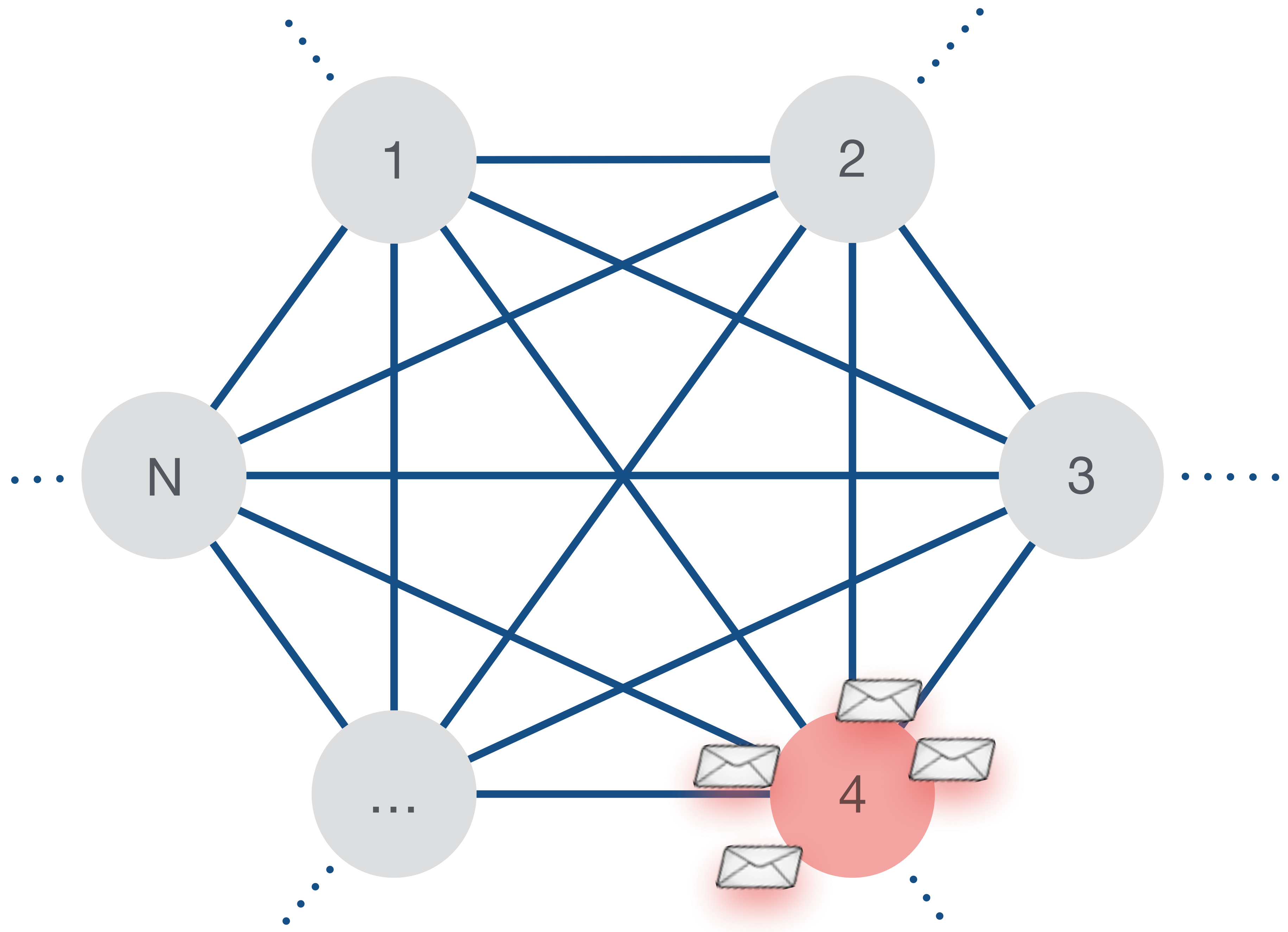
## Mesh



- Route through random intermediate node
- Works well for mesh topologies
- WANs are not mesh-like
    - Good resilience
    - Poor performance & latency

# VLB

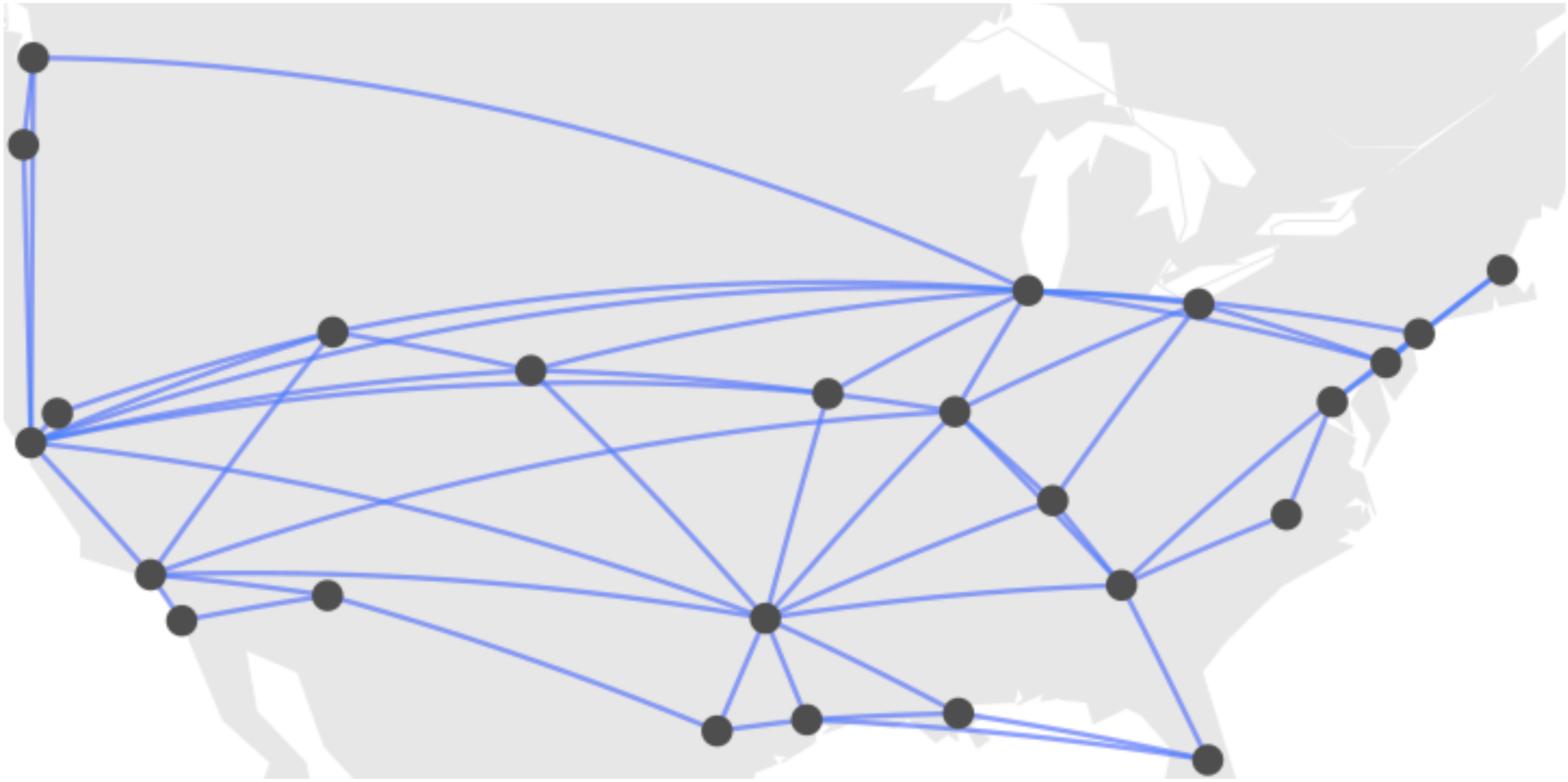
## Mesh



- Route through random intermediate node
- Works well for mesh topologies
- WANs are not mesh-like
  - Good resilience
  - Poor performance & latency

# VLB

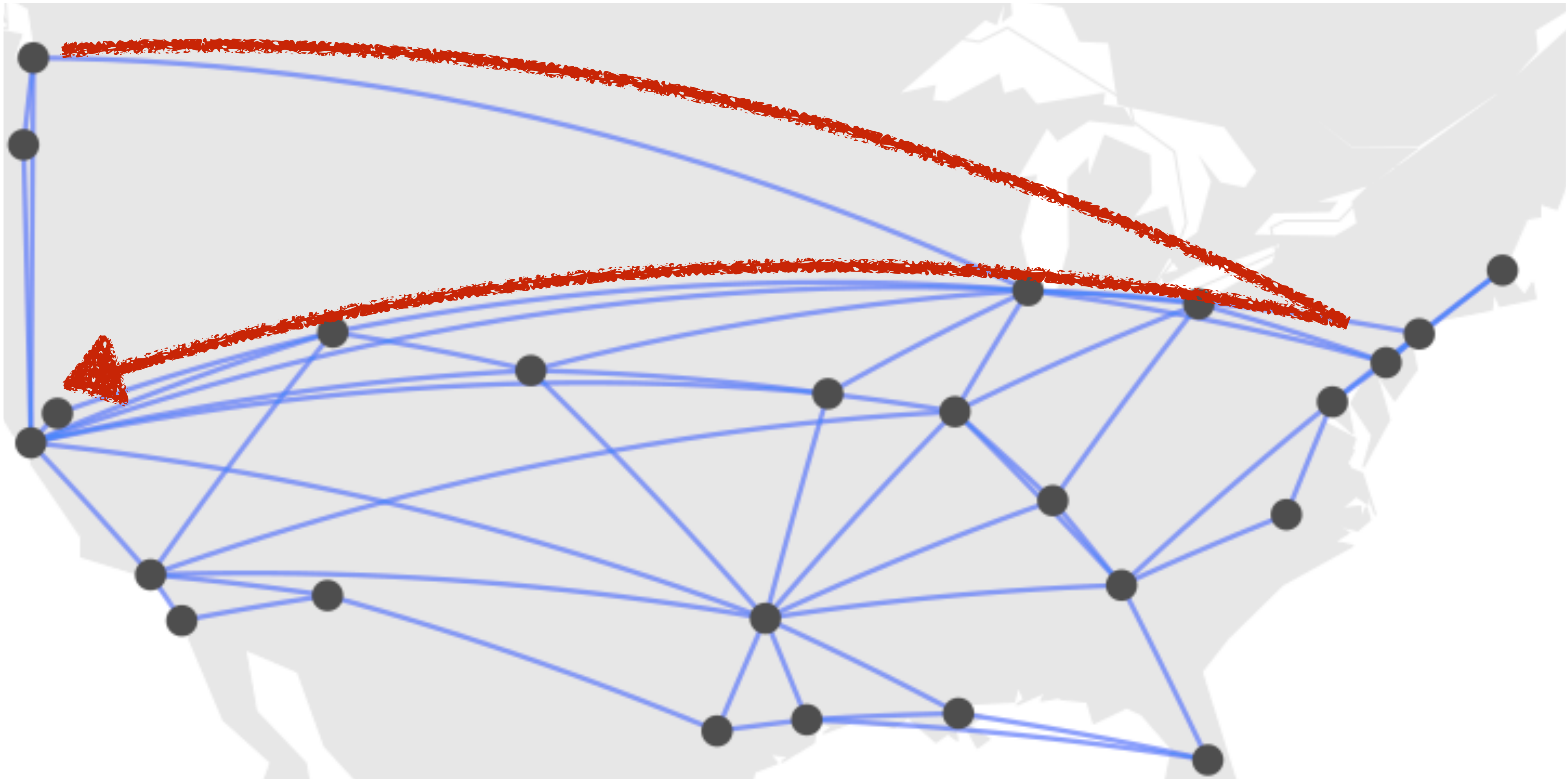
## Not Mesh



- Route through random intermediate node
- Works well for mesh topologies
- WANs are not mesh-like
  - Good resilience
  - Poor performance & latency

# VLB

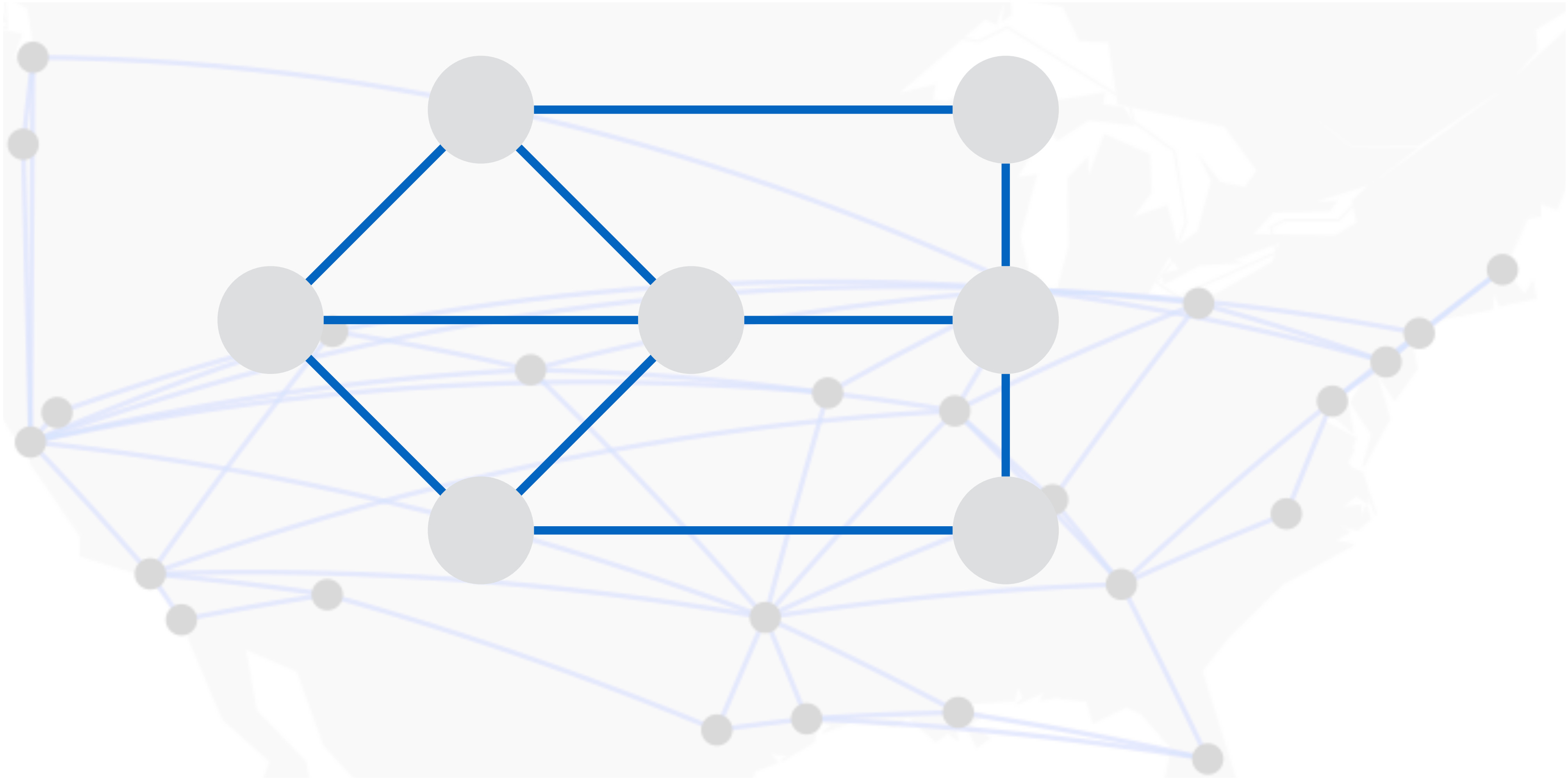
## Not Mesh



- Route through random intermediate node
- Works well for mesh topologies
- WANs are not mesh-like
  - Good resilience
  - Poor performance & latency

# Oblivious [Räcke '08]

## Not Mesh

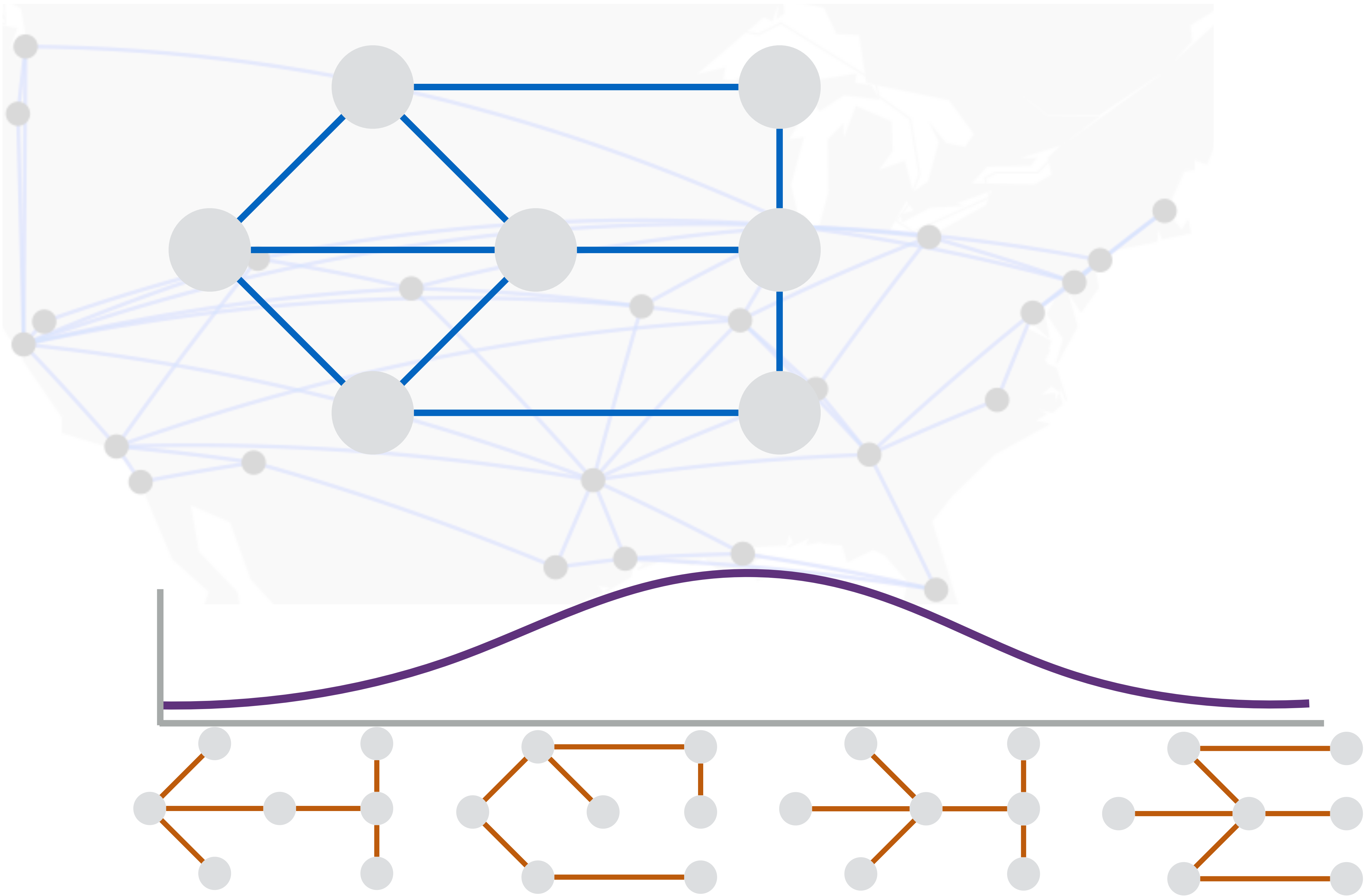


Low-stretch routing trees

- Generalizes VLB to non-mesh
- Distribution over routing trees
    - Approximation algorithm for low-stretch trees [FRT '04]
    - Penalize links based on usage
- *O(log n)* competitive

# Oblivious [Räcke '08]

## Not Mesh



Low-stretch routing trees

- Generalizes VLB to non-mesh
- Distribution over routing trees
  - Approximation algorithm for low-stretch trees [FRT '04]
  - Penalize links based on usage
- *O(log n)* competitive

# Path Selection

| Algorithm | Load balanced | | Diverse | Low-stretch |
|---|---|---|---|---|
| | Capacity aware | Globally Optimized | | |
| SPF / ECMP | ✗ | ✗ | ✗ | ✓ |
| CSPF | ✓ | ✗ | ✗ | ✓ |
| k-shortest paths | ✗ | ✗ | ? | ✓ |
| Edge-disjoint KSP | ✗ | ✗ | ✓ | ✓ |
| MCF | ✓ | ✓ | ✗ | ✗ |
| VLB | ✗ | ✗ | ✓ | ✗ |
| B4 | ✓ | ✓ | ✗ | ? |
| SMORE / Oblivious | ✓ | ✓ | ✓ | ✓ |

# SMORE: Semi-Oblivious Routing

*Oblivious Routing* computes a set of paths which are low-stretch, robust and have good load balancing properties

Path Selection

*LP Optimizer* balances load by dynamically adjusting splitting ratios used to map incoming traffic flows to paths

Rate Adaptation

# Semi-Oblivious Routing in Practice?

- ▼ Previous work [Hajiaghayi et al.] established a worst-case competitive ratio that is not much better than oblivious routing: Ω(log(n)/log (log(n)))

- ▲ But the real-world does not typically exhibit worst-case scenarios

- ▲ e.g., there is an correlation between demands and link capacities as network designs evolve

- **Question:** How well does semi-oblivious routing perform in practice?

# Evaluation

# Facebook's WAN

- Overview
  - Common network design for content providers
  - Several large data centers (DCs) and points-of-presence (PoPs)
  - Mix of latency-sensitive customer traffic + background elastic traffic
- Method
  - Collected accurate snapshot of network state - topology, TMs, etc.
  - Simulations to study performance characteristics

# TE Systems - Comparison

## Traditional

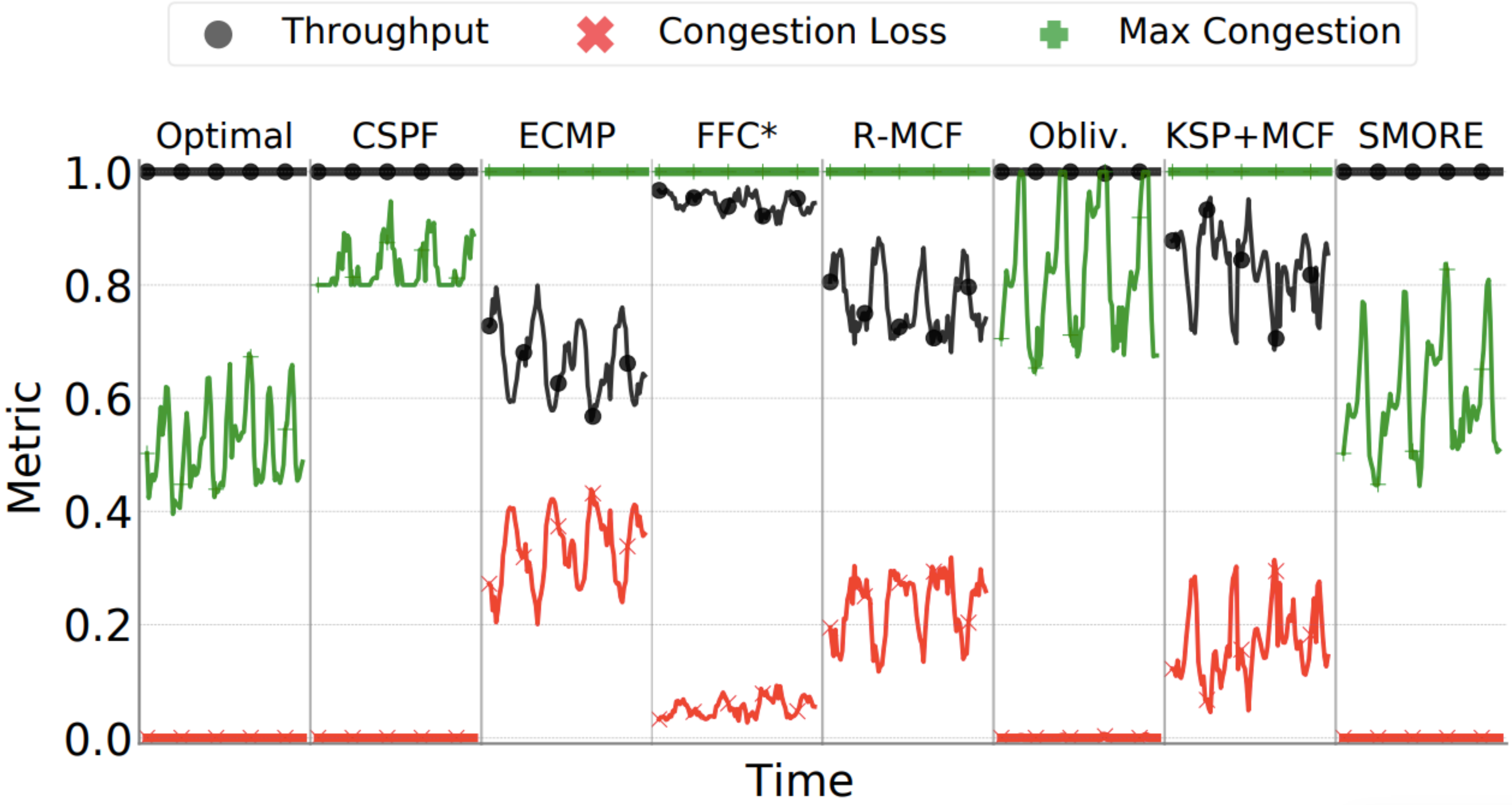
- OSPF
- ECMP
- CSPF
- MCF
- Omniscient MCF (“Optimal”)
- …

## Contemporary

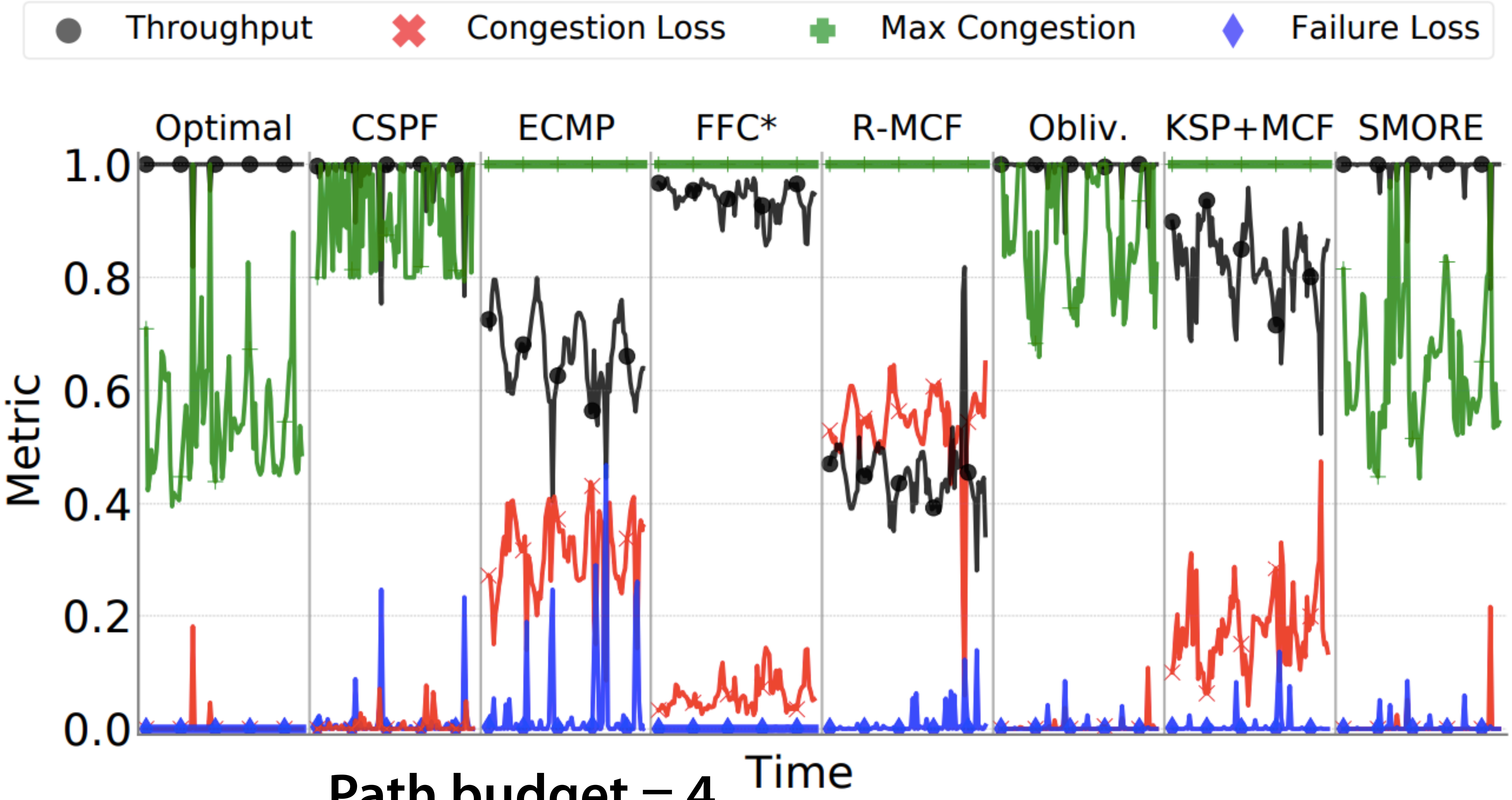
- Oblivious [STOC ’08]
- VLB [INFOCOM ‘08]
- Robust MCF [SIGMETRICS ‘11]
- KSP + MCF [SIGCOMM ’13]
- FFC* [SIGCOMM ’15]
- …

Open-source implementations at http://github.com/cornell-netlab/yates

# Performance

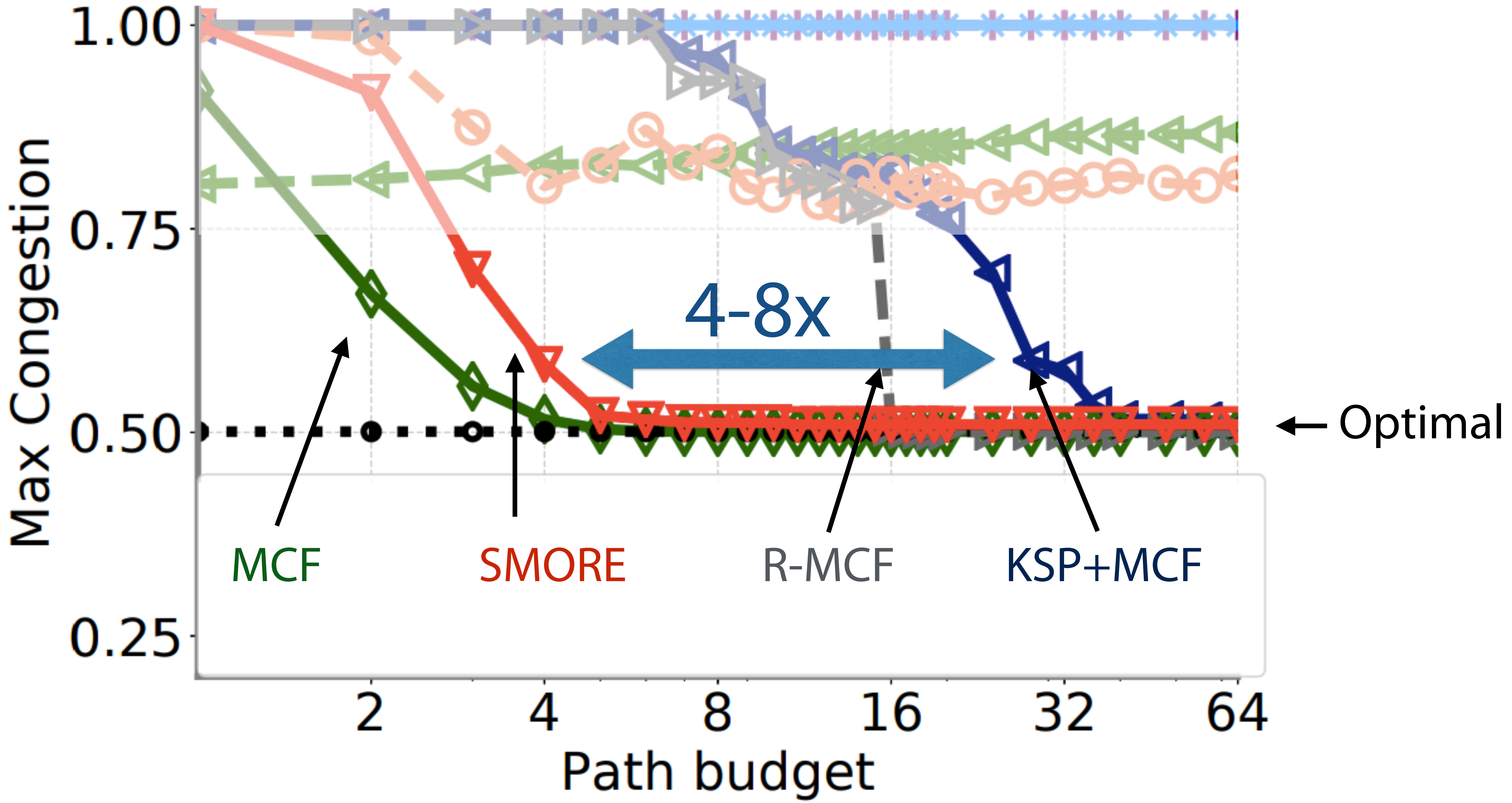


# Robustness



**Path budget = 4**

# Operational Constraints - Path Budget

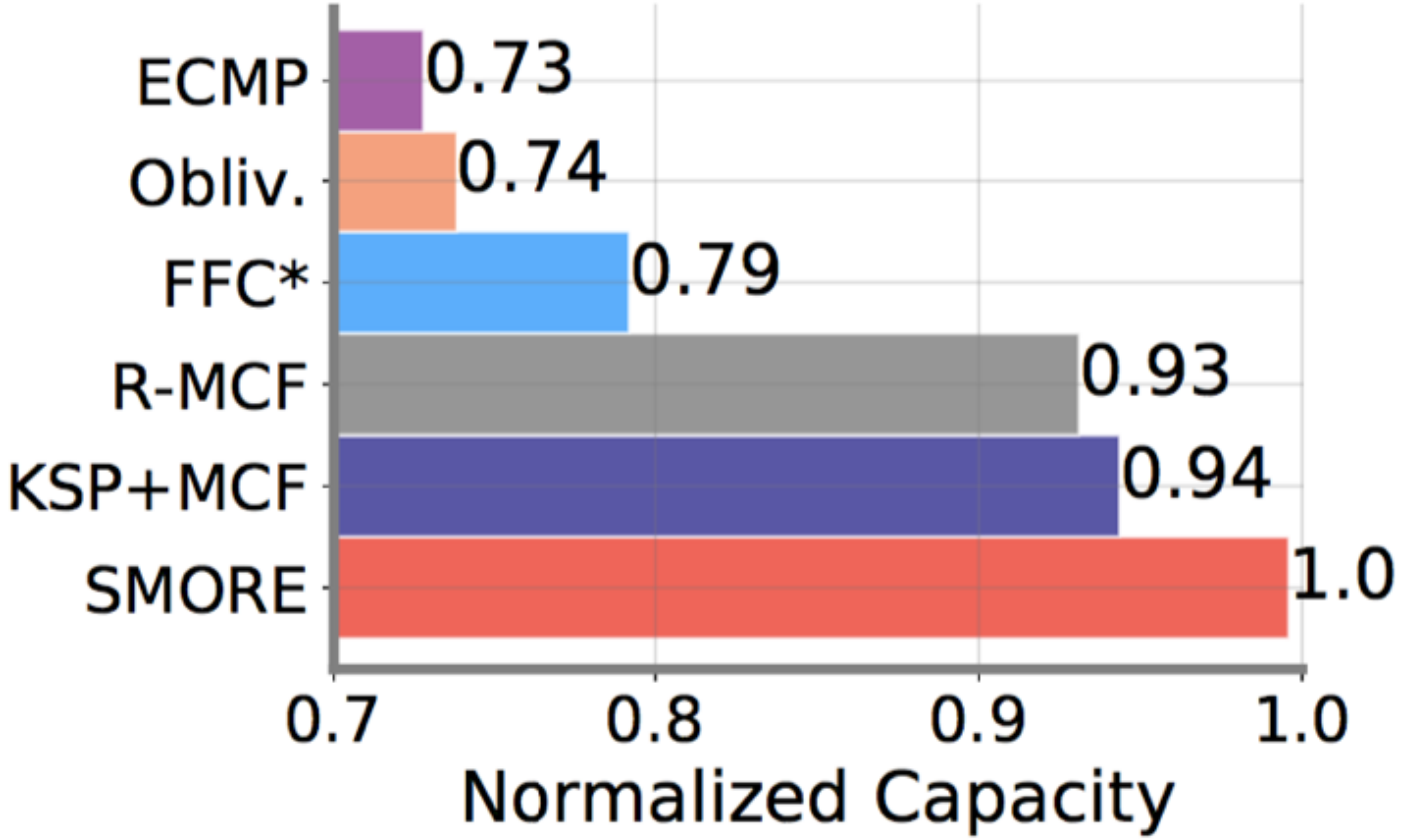


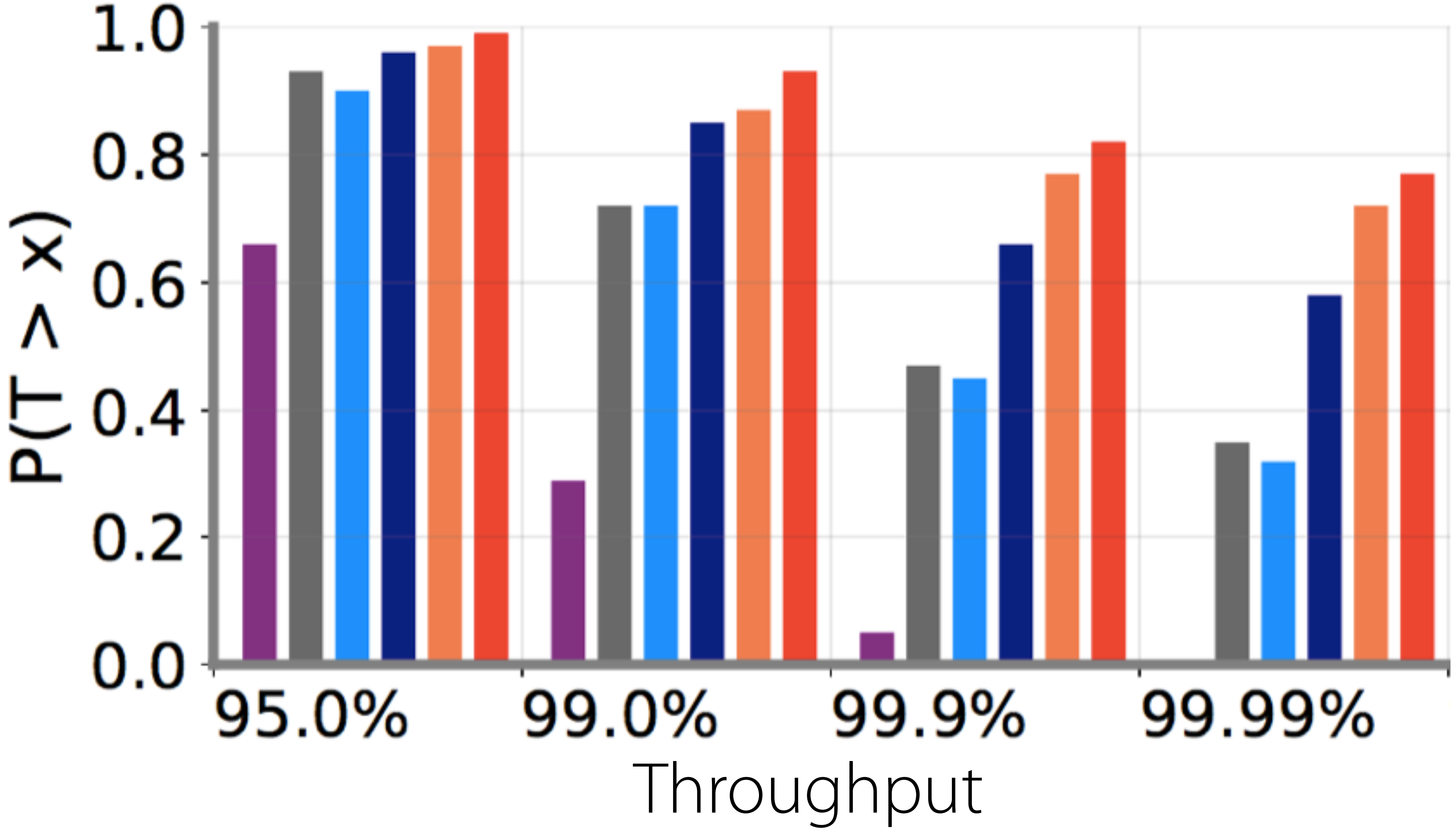
# Large Scale Simulations

- Conducted larger set of simulations on Internet Topology Zoo
- 30 topologies from ISPs and content providers
- Multiple traffic matrices (gravity model), failure models and operational conditions

# Do these results generalize?

Yes*





Probability of achieving SLA

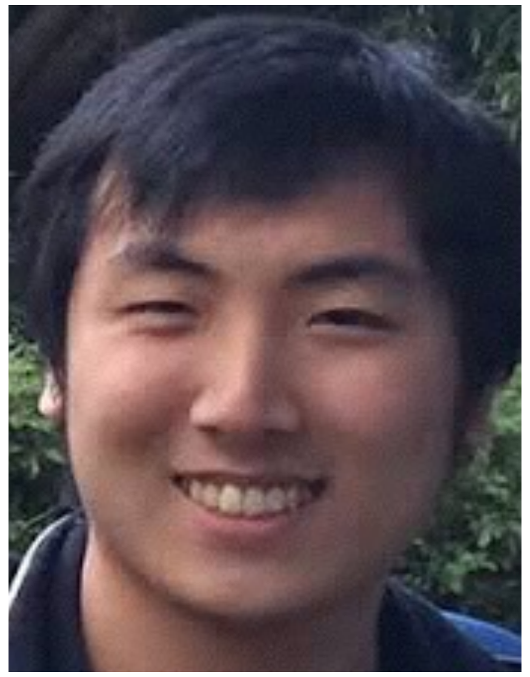
# Takeaways

- **Path selection** plays an outsized role in the performance of TE systems
- **Semi-oblivious TE** meets the competing objectives of performance and robustness in modern networks
  - **Oblivious routing** for path selection + **Dynamic load-balancing**
- Ongoing and future-work:
  - Apply to other networks (e.g. non-Clos DC topologies)
  - SR-based implementations and deployments

# Thank You!

SMORE: Oblivious routing + Dynamic rate adaptation

Yang Yuan
Cornell



Chris Yu
CMU

Nate Foster
Cornell

Bobby Kleinberg
Cornell

Petr Lapukhov
Facebook

Chiun Lin Lim
Facebook

Robert Soule
Lugano

https://github.com/cornell-netlab/yates