# Self-Adjusting Networks

Stefan Schmid

"We cannot direct the wind,
but we can adjust the sails."
(Folklore)

Acknowledgements:

# Trend

## Data-Centric Applications

Datacenters (“hyper-scale”)

+network


Source: Facebook

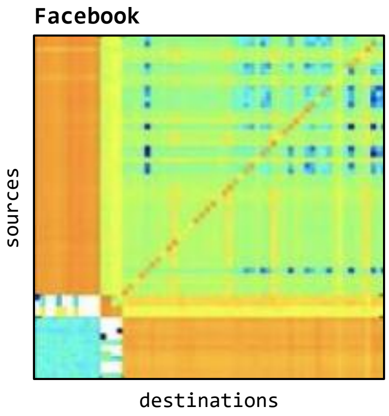
Interconnecting networks:
a **critical infrastructure**
of our digital society.

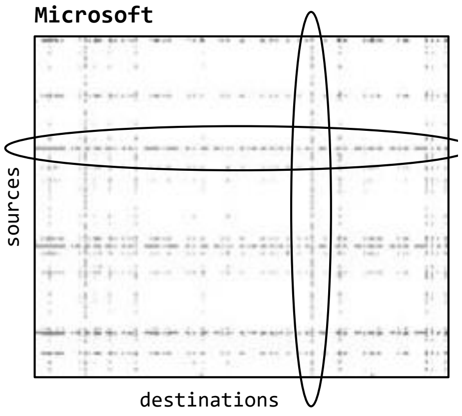
Communication Traffic:

# Big But Structured

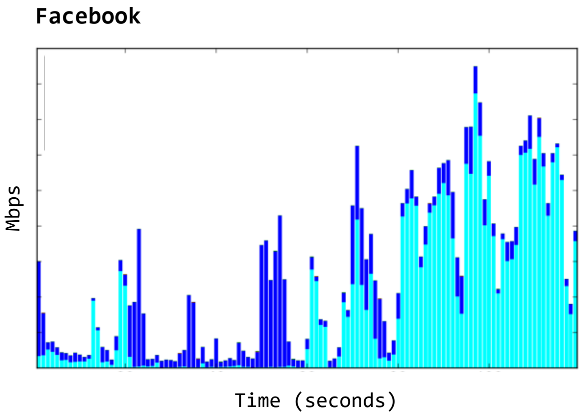
Traffic does not only grow but also has much structure:

traffic matrices sparse and skewed



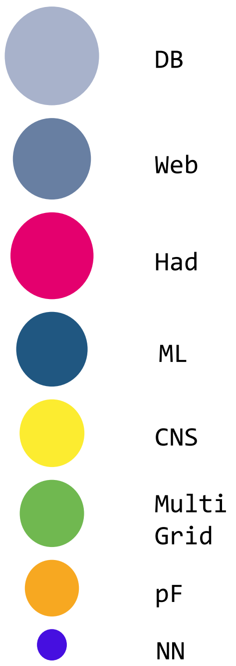




traffic bursty over time



Recent Representation of Trace Structure:

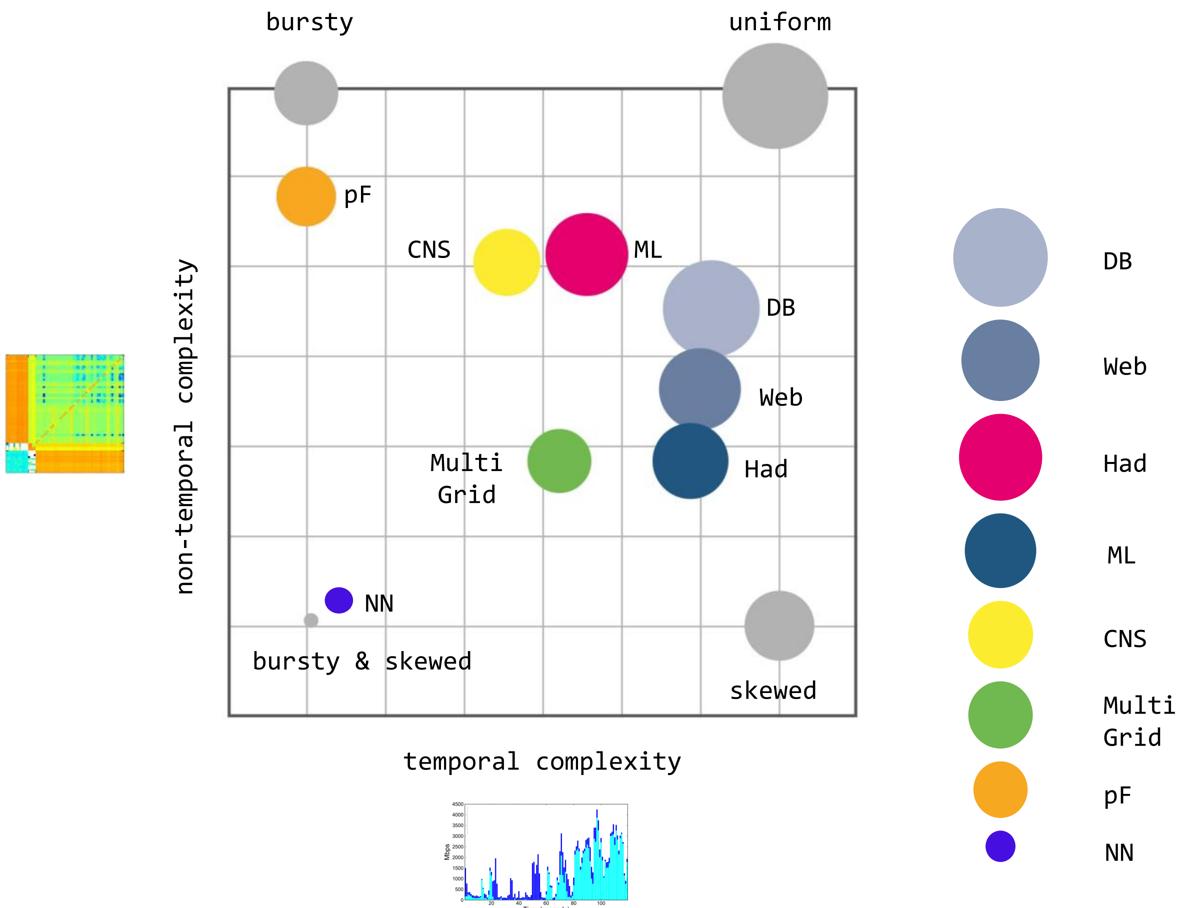
# Complexity Map




Griner et al., SIGMETRICS 2020

Recent Representation of Trace Structure:

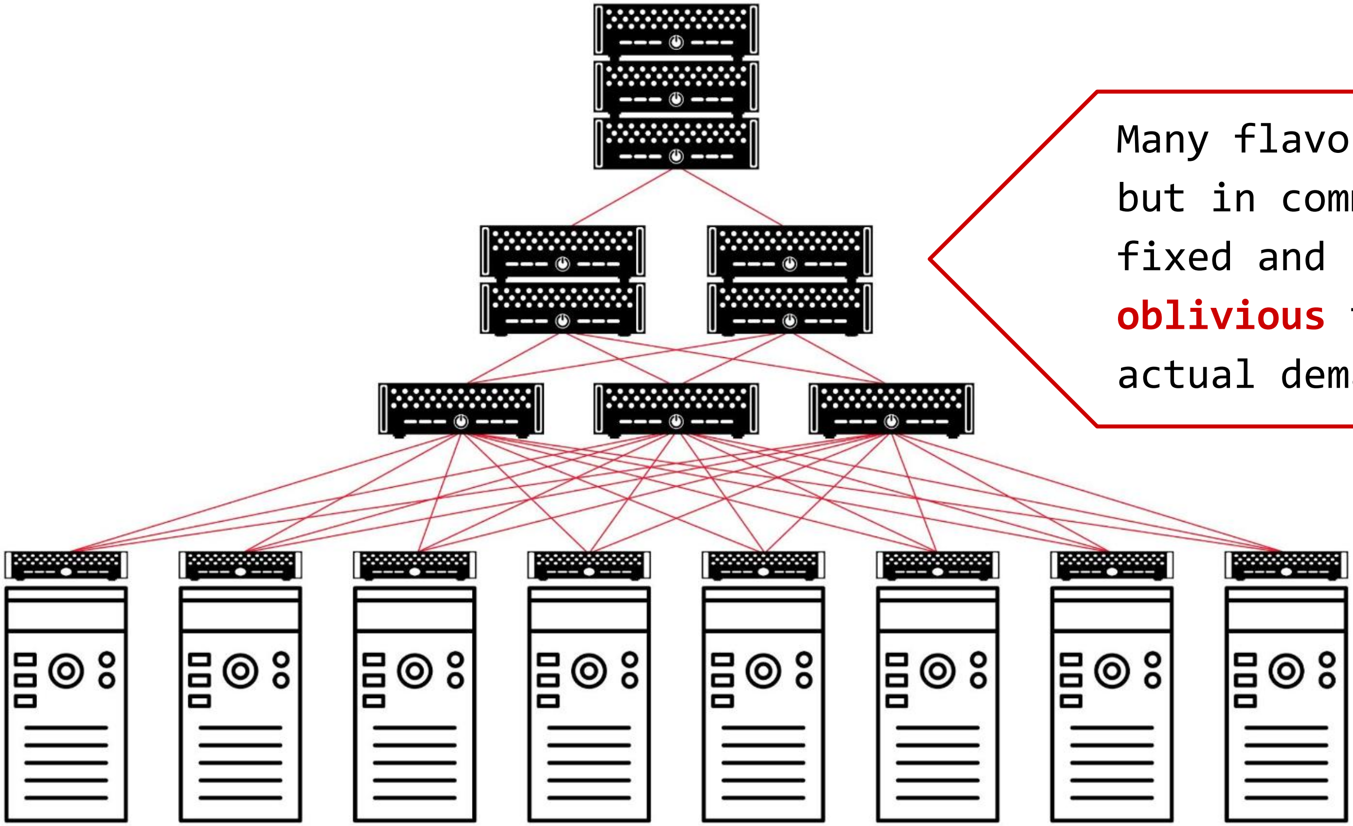
# Complexity Map




**Different structures!**

# One Solution?

## Today: Demand-Oblivious Topology

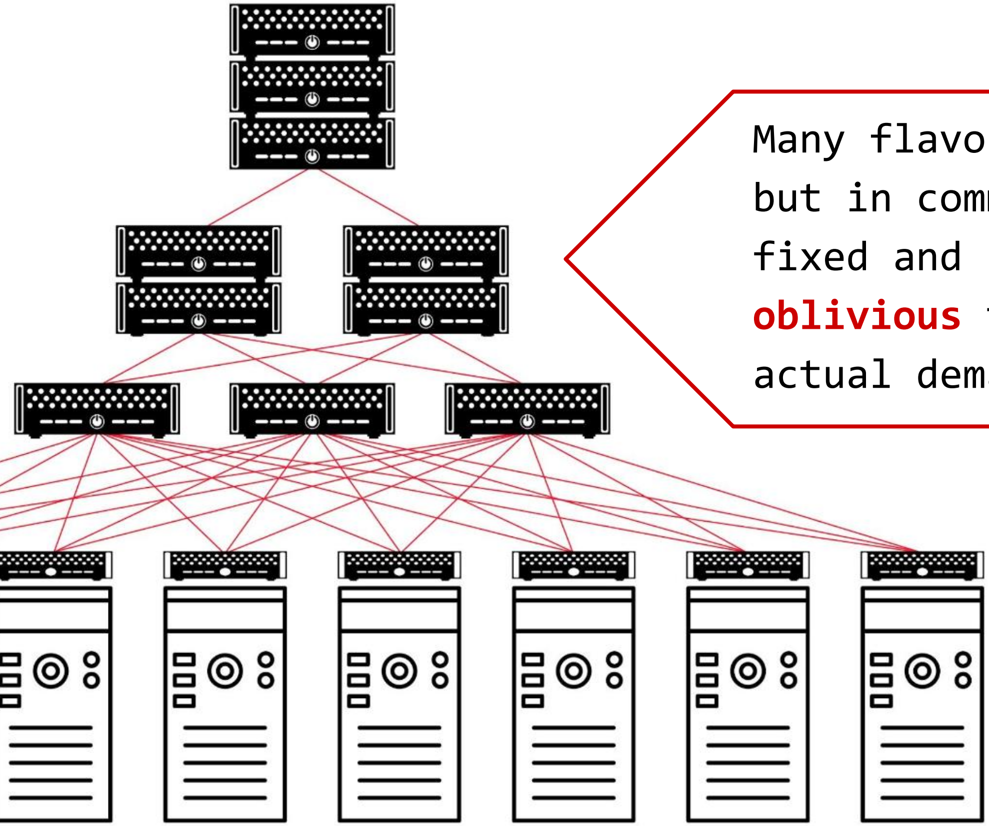


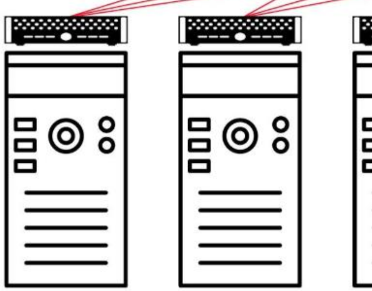
Many flavors,
but in common:
fixed and
**oblivious** to
actual demand.

# One Solution?

## Today: Demand-Oblivious Topology

**Highway which ignores actual traffic: frustrating!**

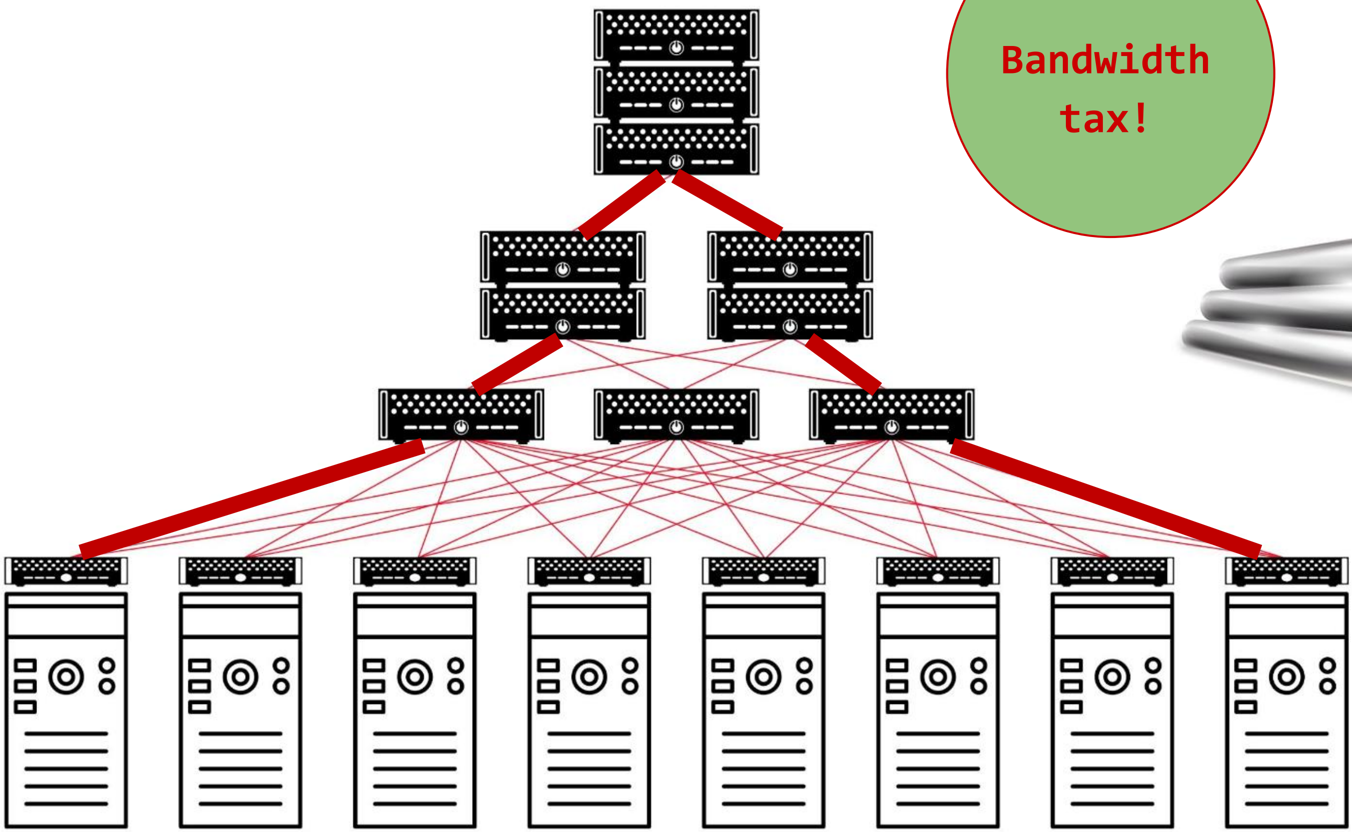




Many flavors, but in common: fixed and **oblivious** to actual demand.

# One Solution?

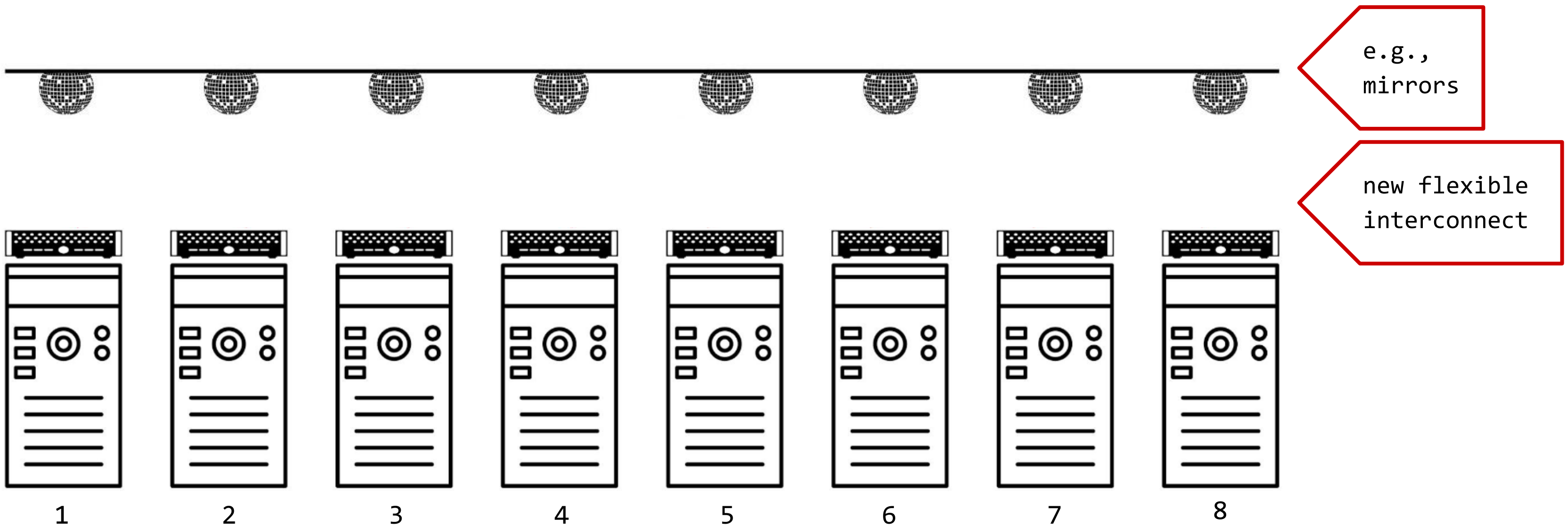
## Today: Demand-Oblivious Topology



Bandwidth
tax!

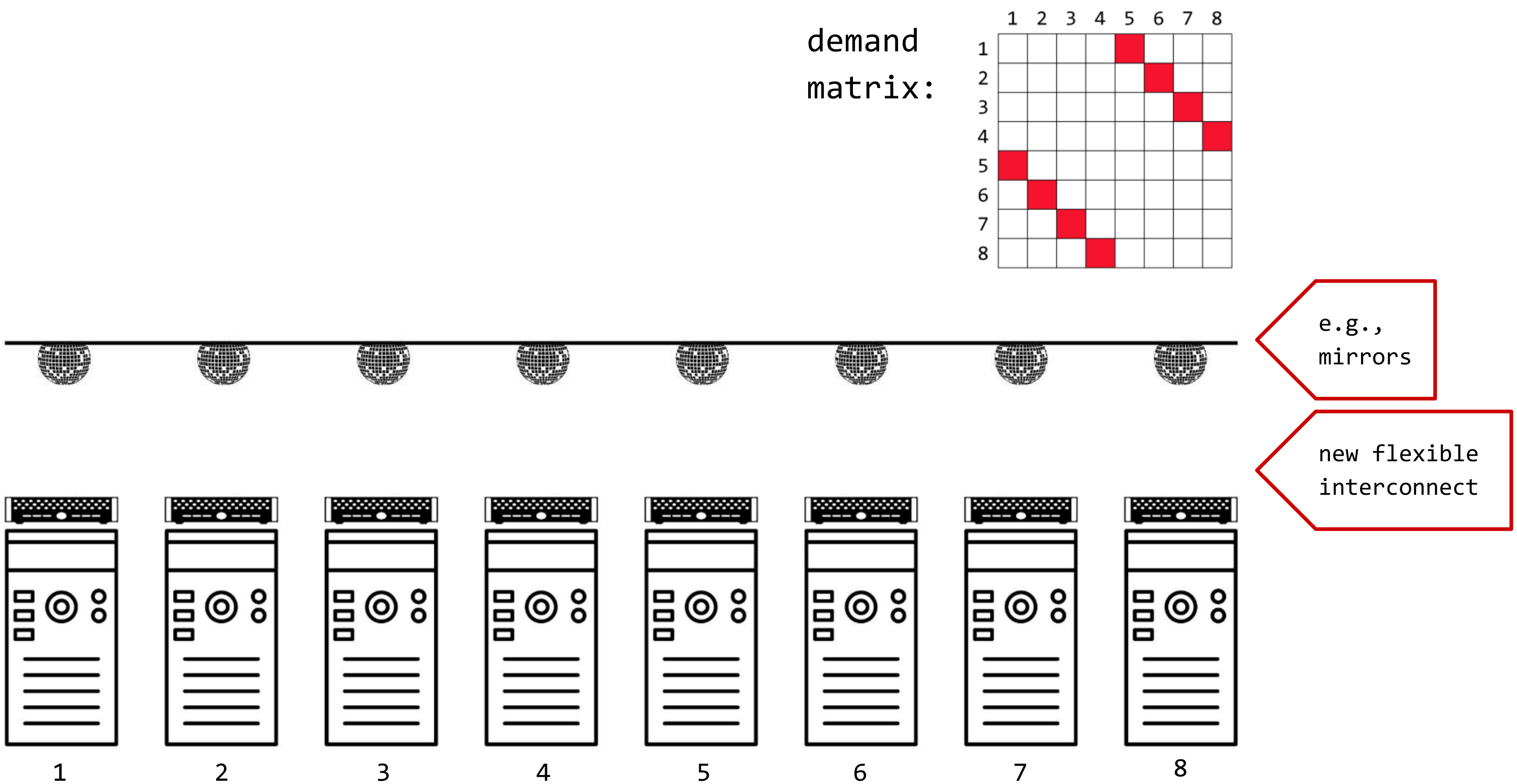
# Emerging Alternatives

## E.g., Demand-Aware Reconfigurable Datacenter



# Emerging Alternatives

## E.g., Demand-Aware Reconfigurable Datacenter

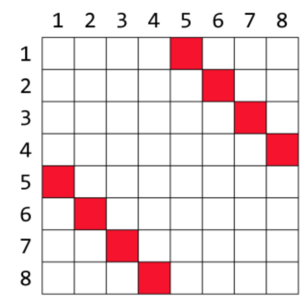


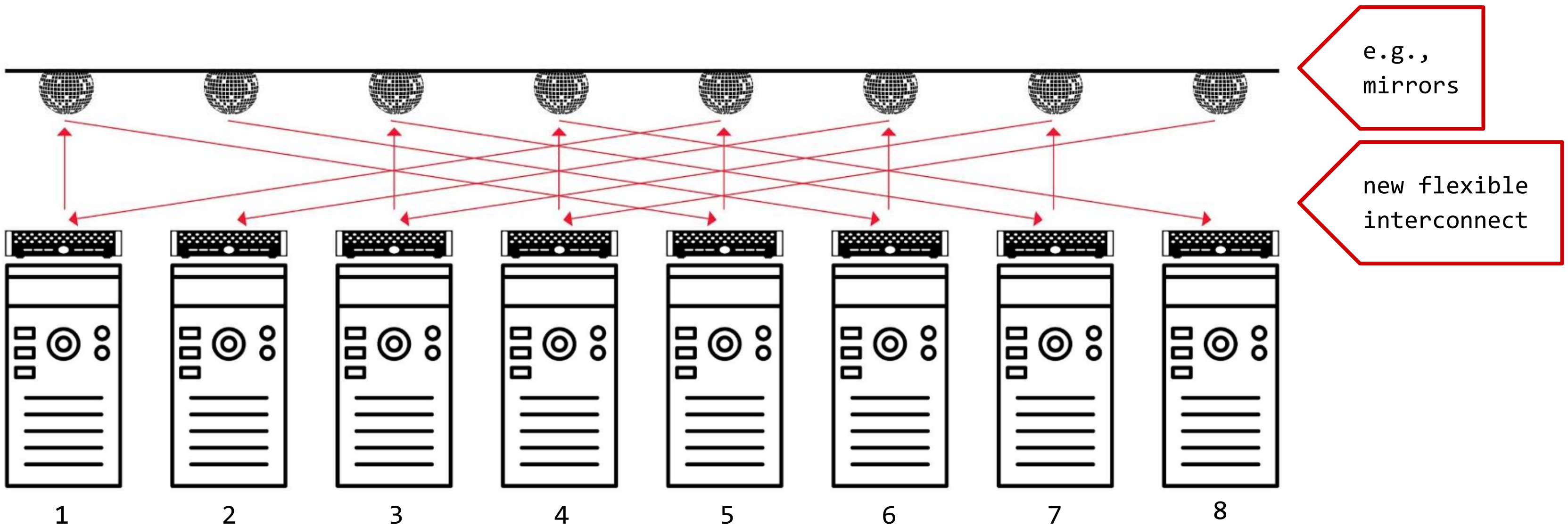
# Emerging Alternatives

## E.g., Demand-Aware Reconfigurable Datacenter

Matches demand

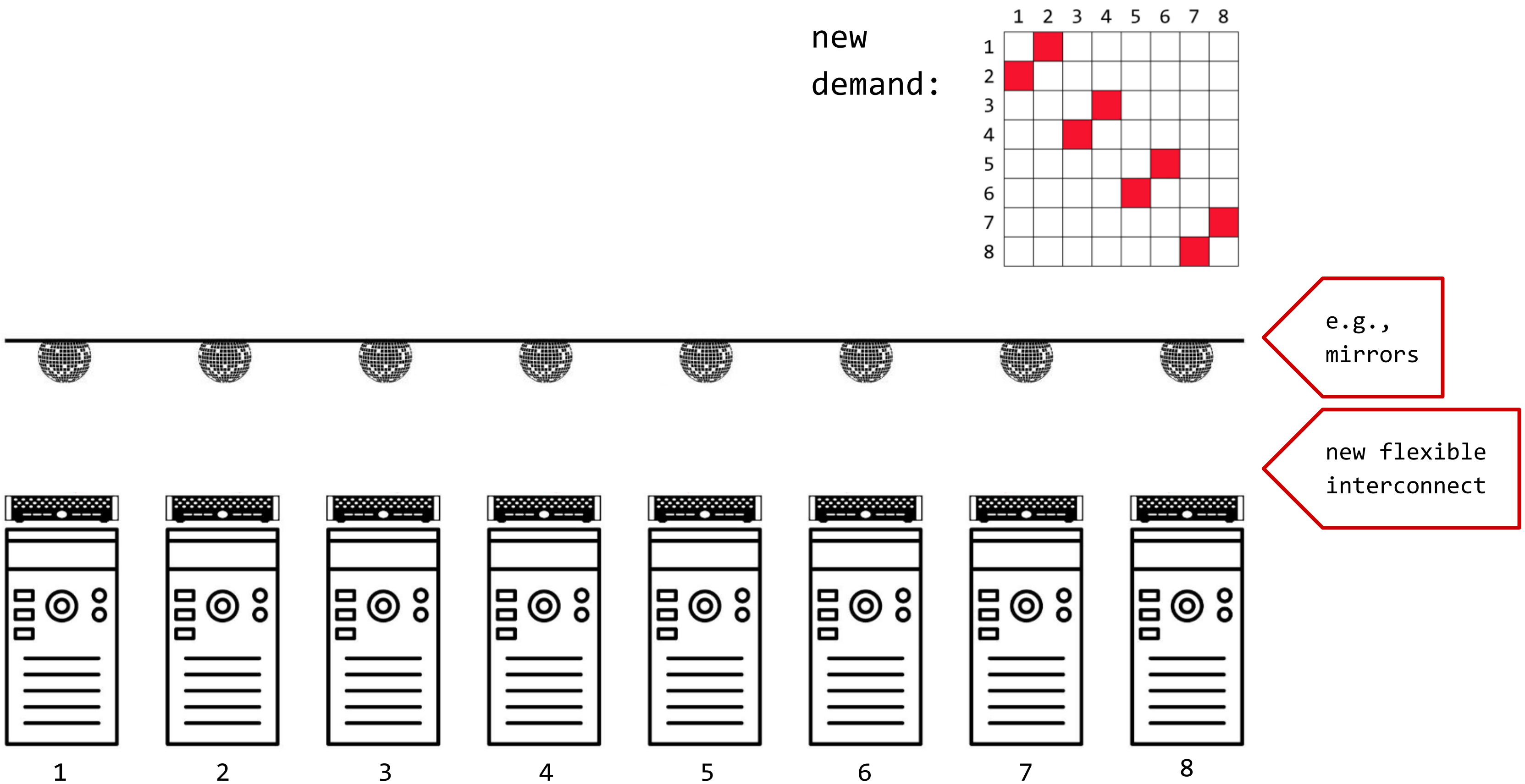
demand matrix:



# Emerging Alternatives

## E.g., Demand-Aware Reconfigurable Datacenter

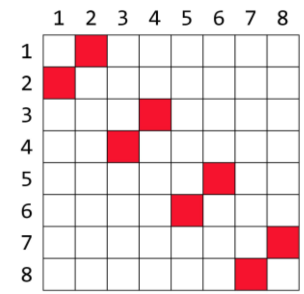


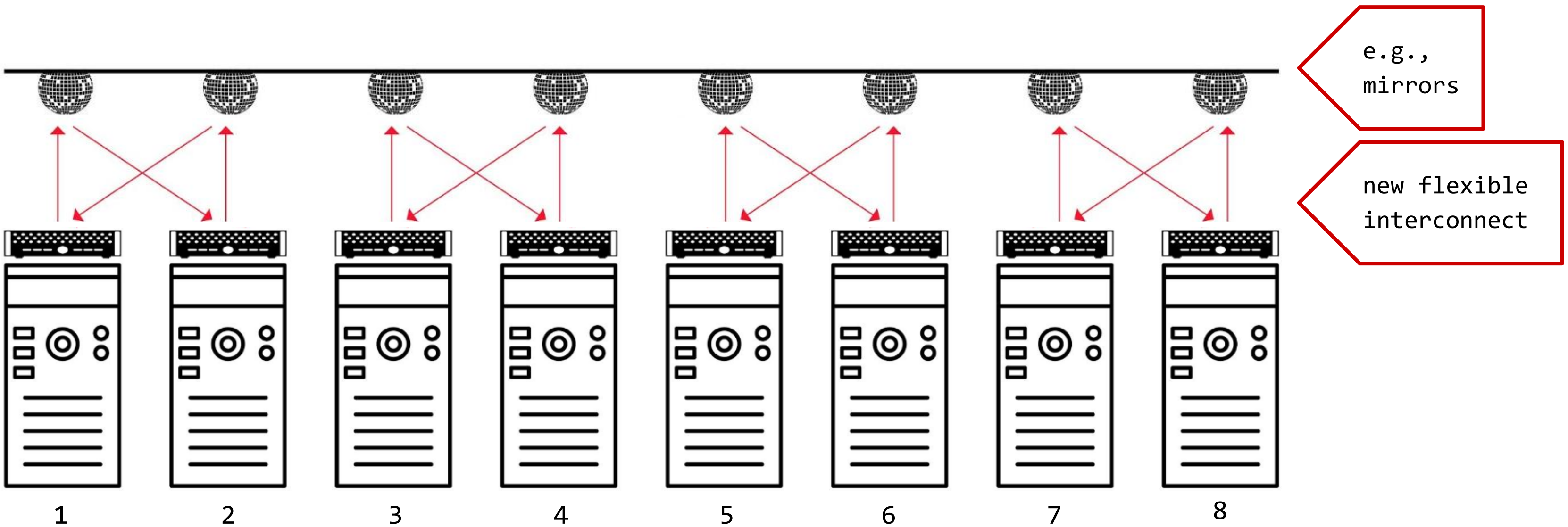
# Emerging Alternatives

## E.g., Demand-Aware Reconfigurable Datacenter

Matches demand

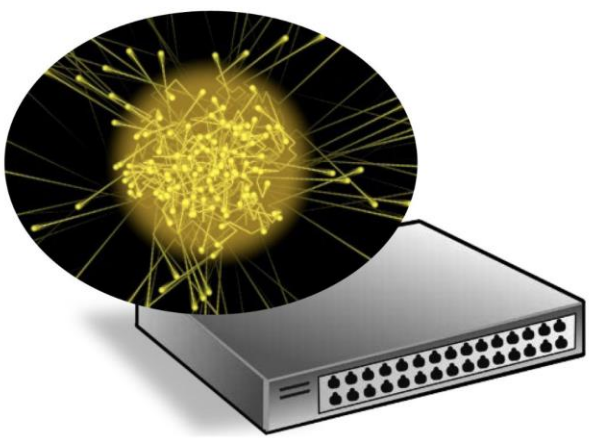
new demand:

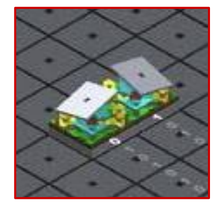


# Crazy? No!

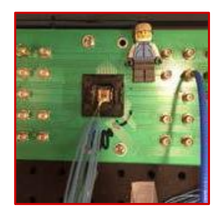
- **Spectrum** of prototypes
  - Different sizes, different reconfiguration times
  - From our ACM **SIGCOMM** workshop OptSys





Prototype 1

Prototype 2

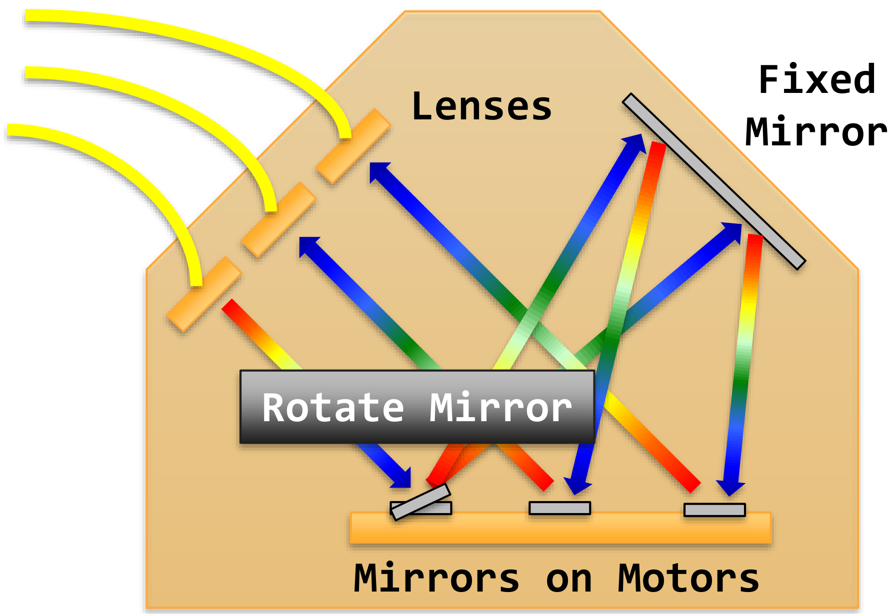


Prototype 3

# Example

## Optical Circuit Switch

- Optical Circuit Switch rapid adaption of physical layer
  - Based on rotating mirrors



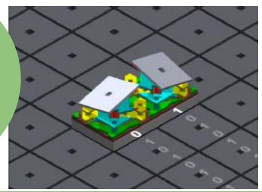

Optical Circuit Switch

By Nathan Farrington, SIGCOMM 2010

# The Big Picture

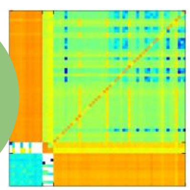
**Flexibility**

New!



**Structure**

More!

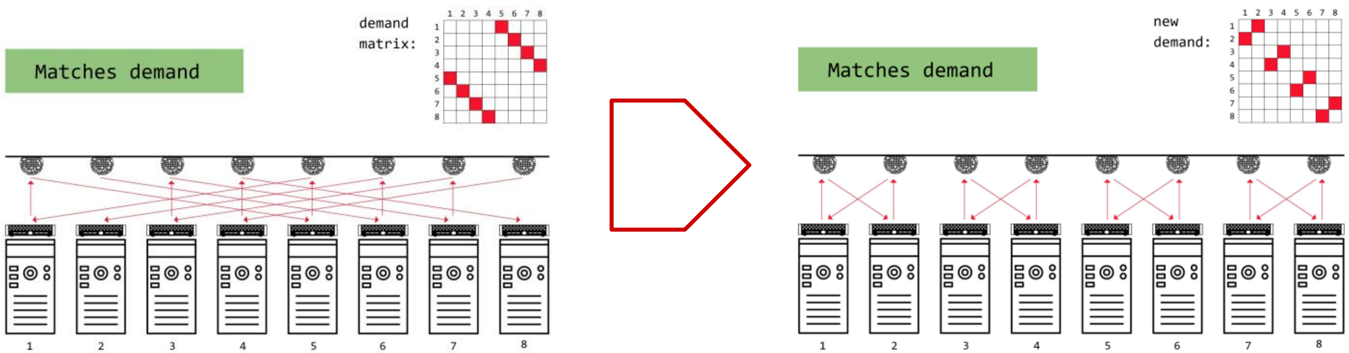


**Self-Adjusting Networks**

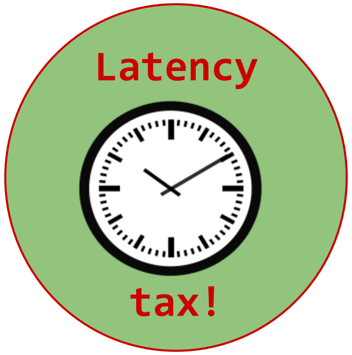
**Efficiency**

Now is the time!

# But: Introduces Tradeoff




- ProjecToR is **demand-aware** through reconfigurations
- However, reconfigurations take time



# Spectrum of Topologies

**Diverse topology components:**

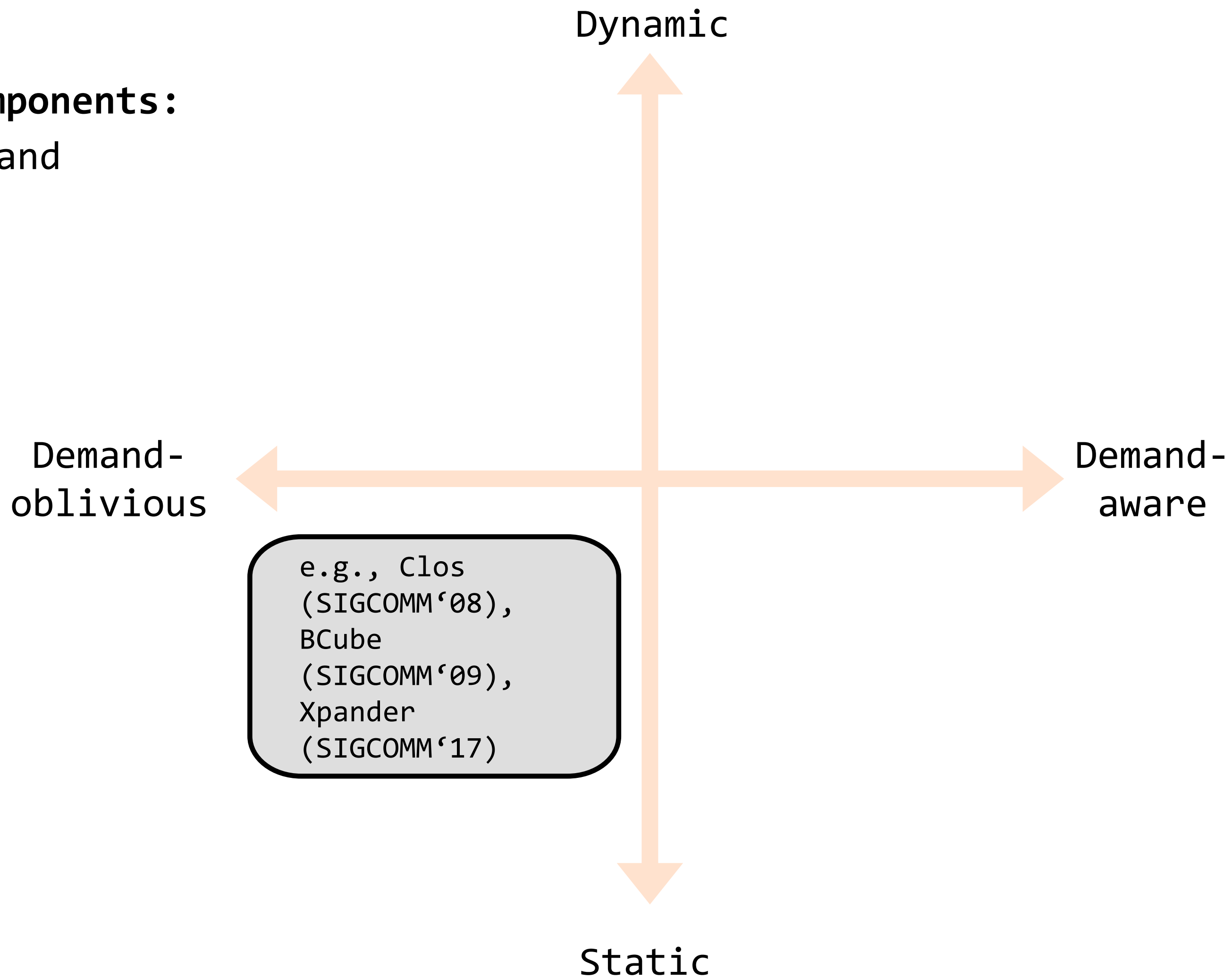
⇀ demand-oblivious and demand-aware

# Spectrum of Topologies

**Diverse topology components:**

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic

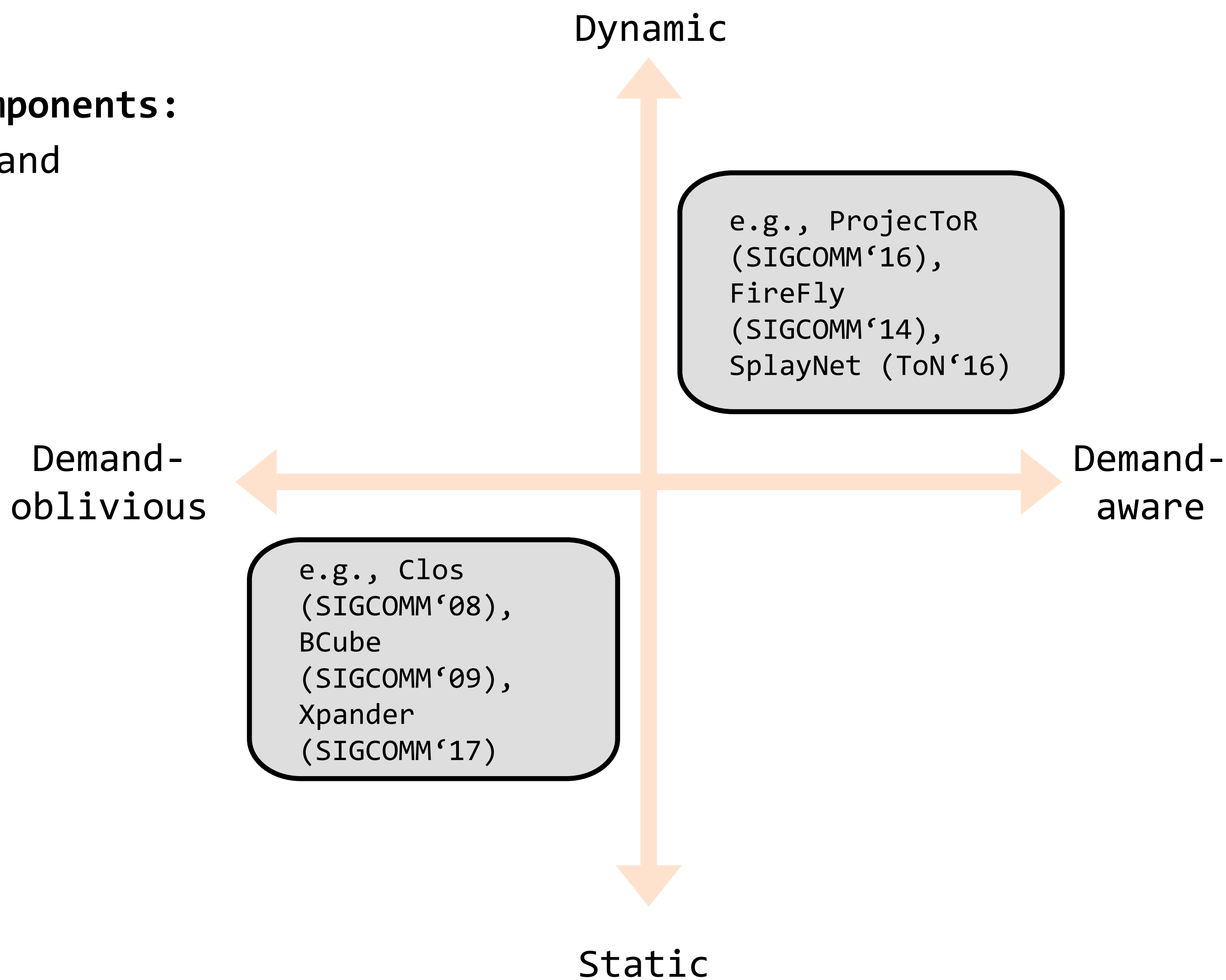


# Spectrum of Topologies

**Diverse topology components:**

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic

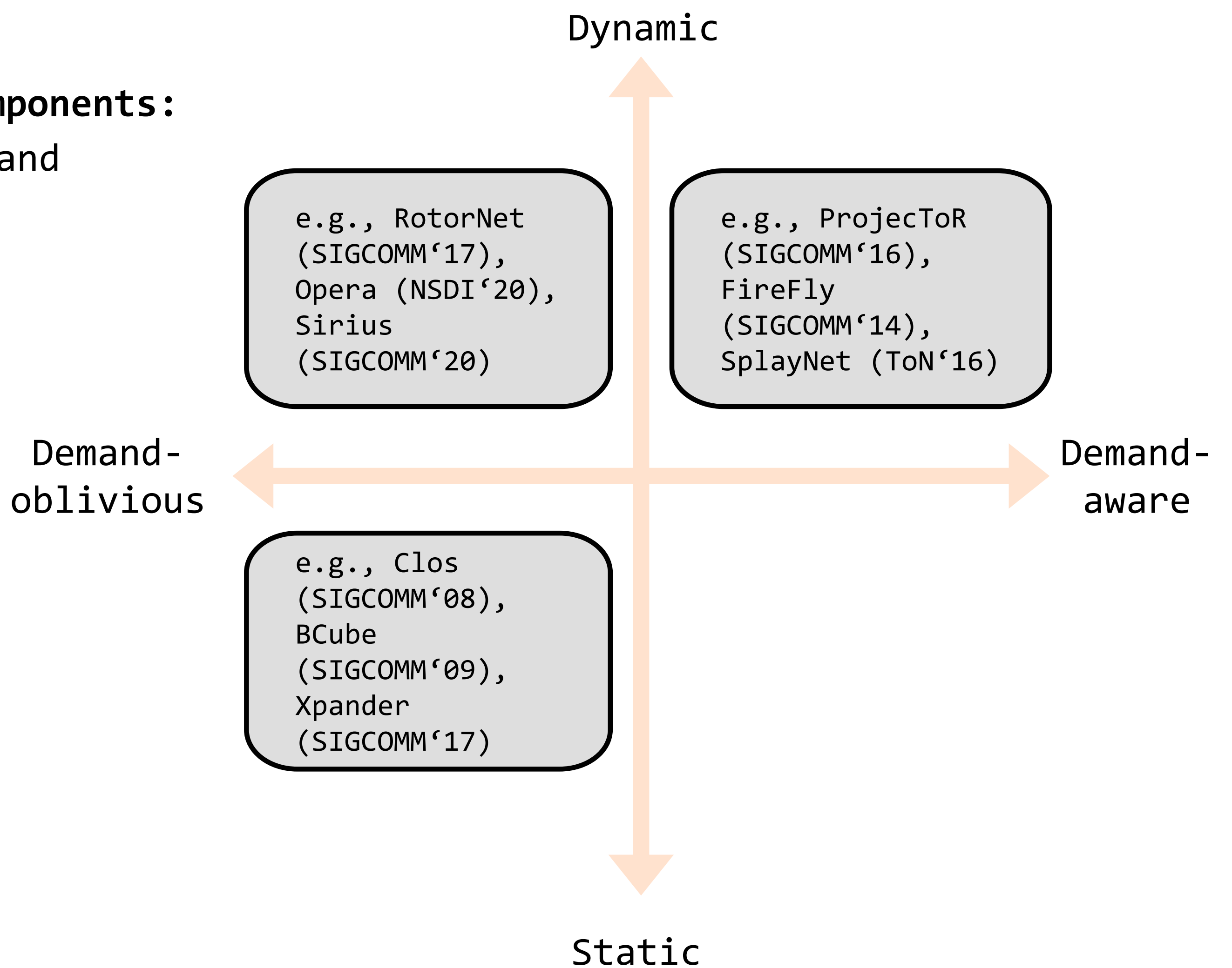


# Spectrum of Topologies

**Diverse topology components:**

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic

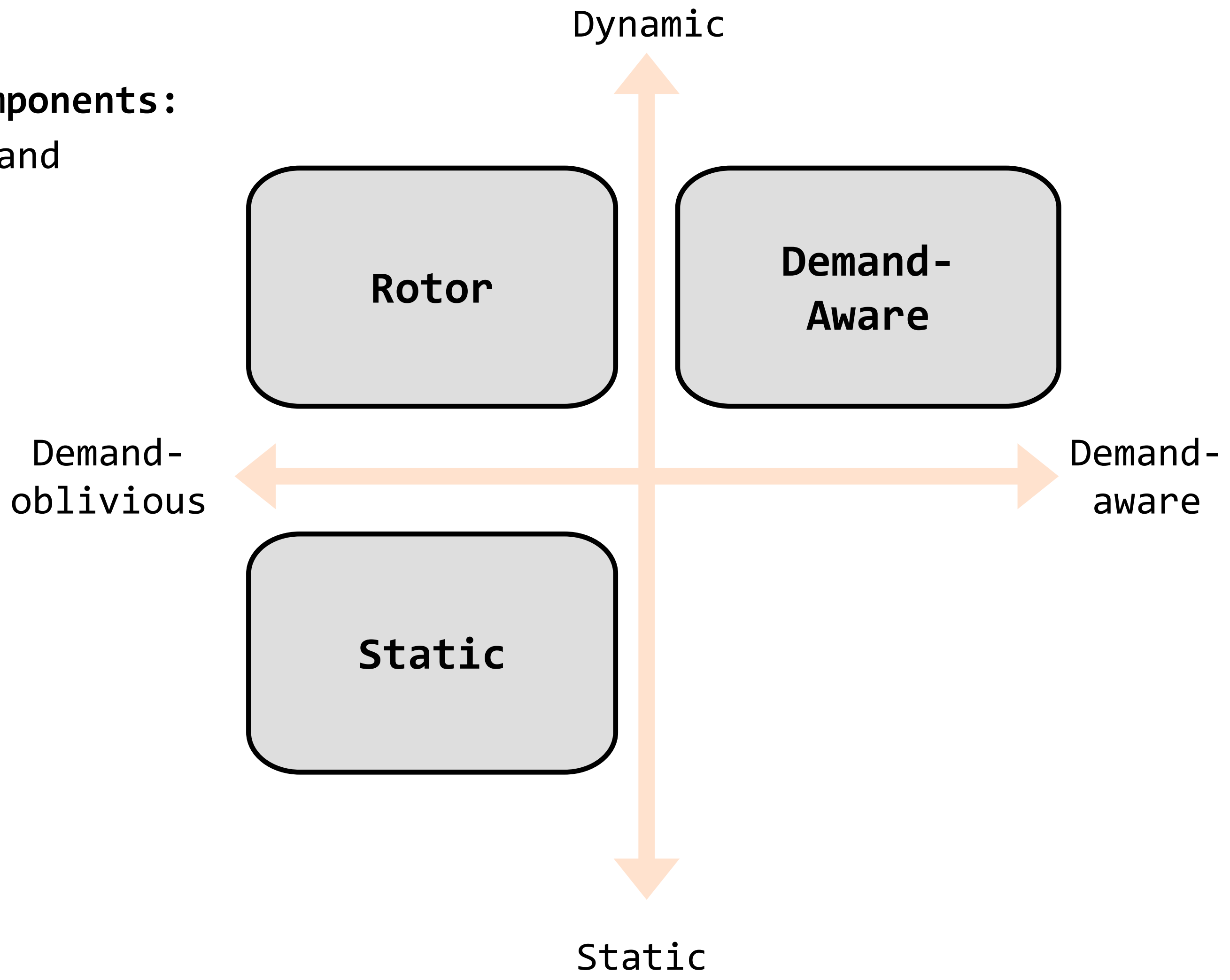


# Spectrum of Topologies

**Diverse topology components:**

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic

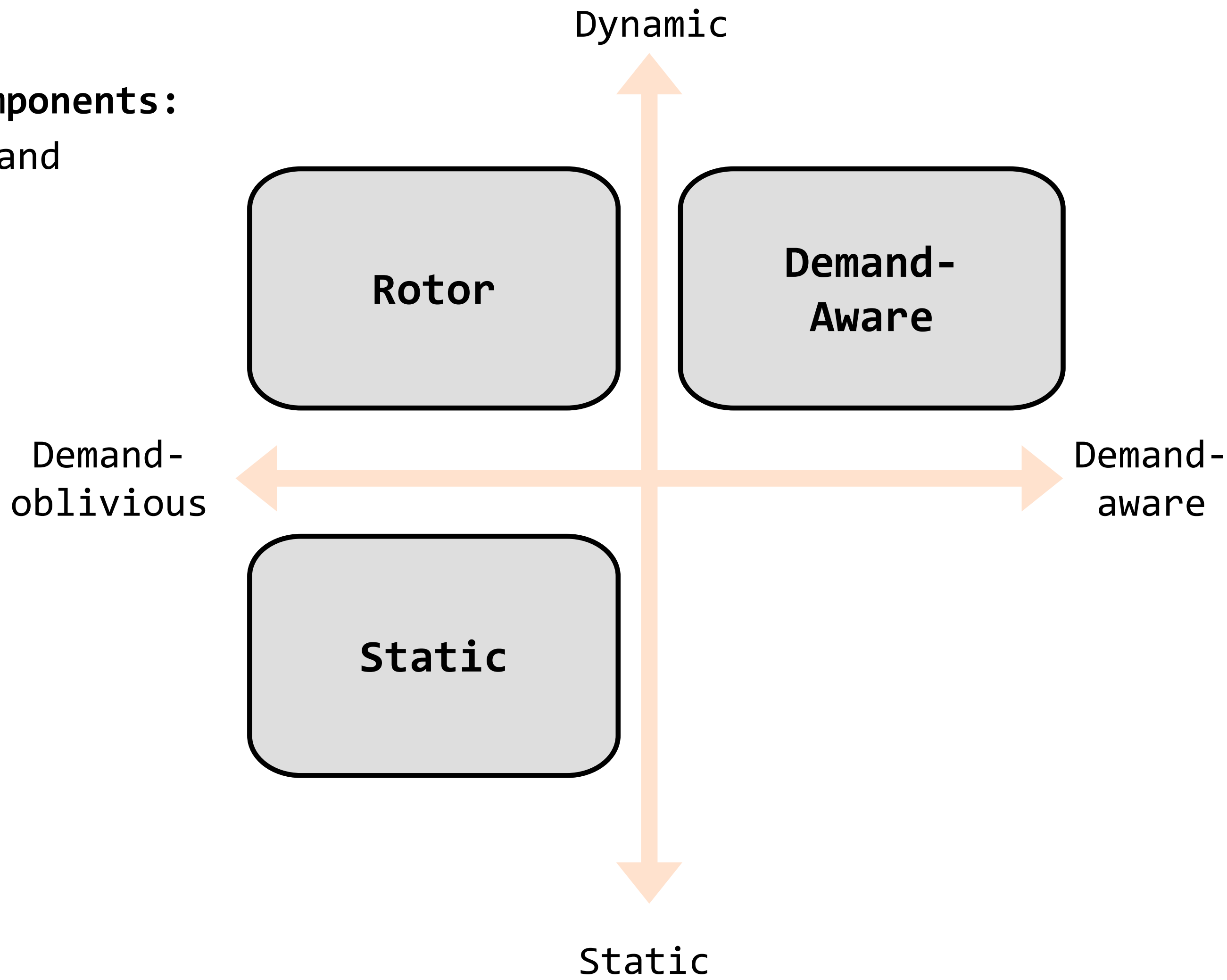


# Spectrum of Topologies

**Diverse topology components:**

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic



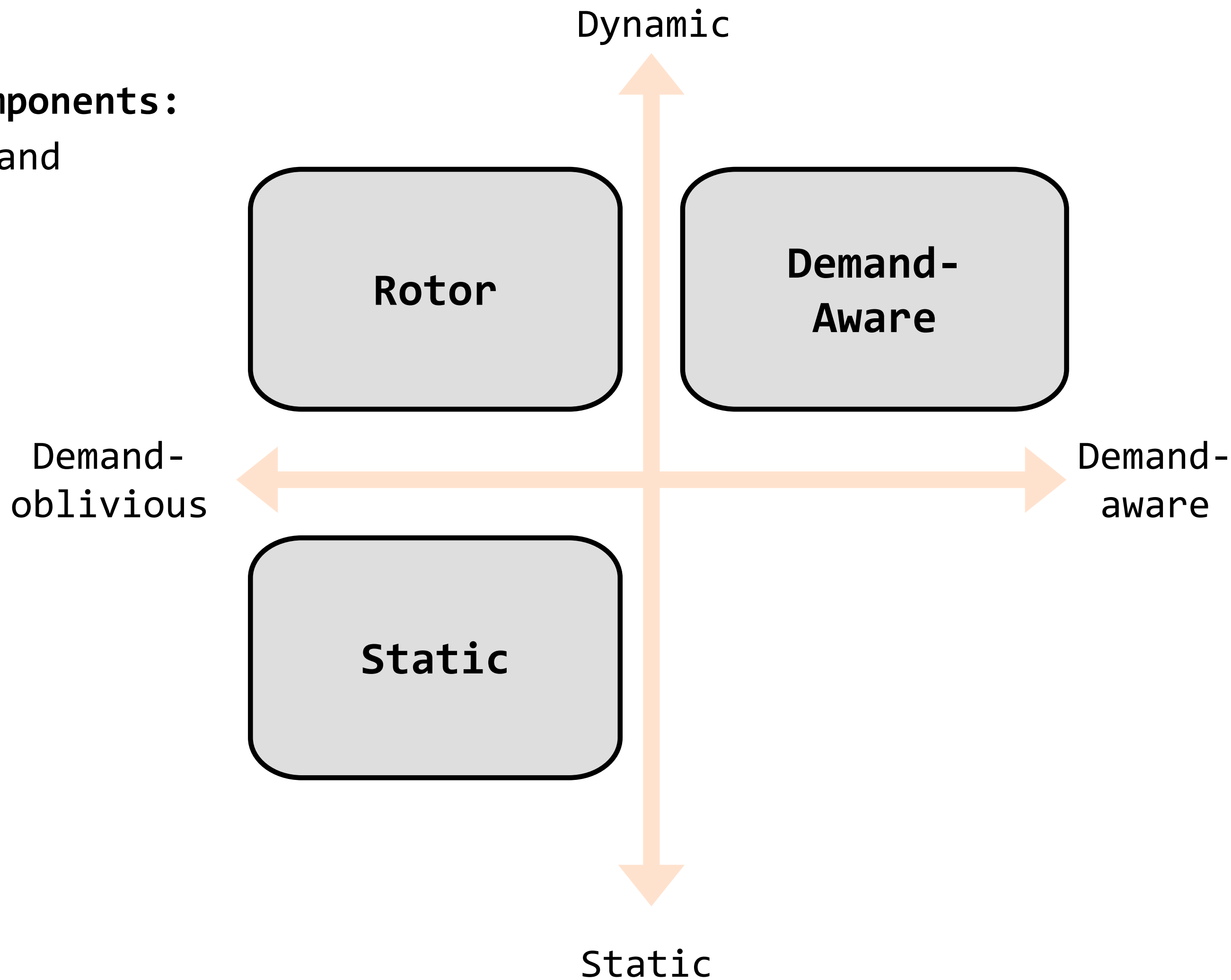

**Which approach is best?**

# Spectrum of Topologies

**Diverse topology components:**

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic





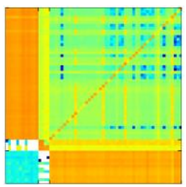
**As always in CS: It depends…**

# Agenda

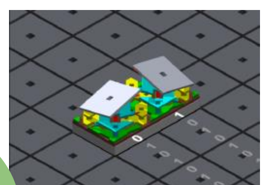
## Exploit Trends for Throughput

More!

**Structure**



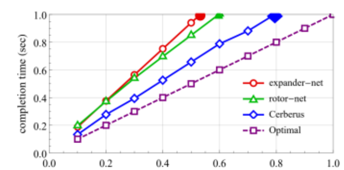
**Technology**



New!

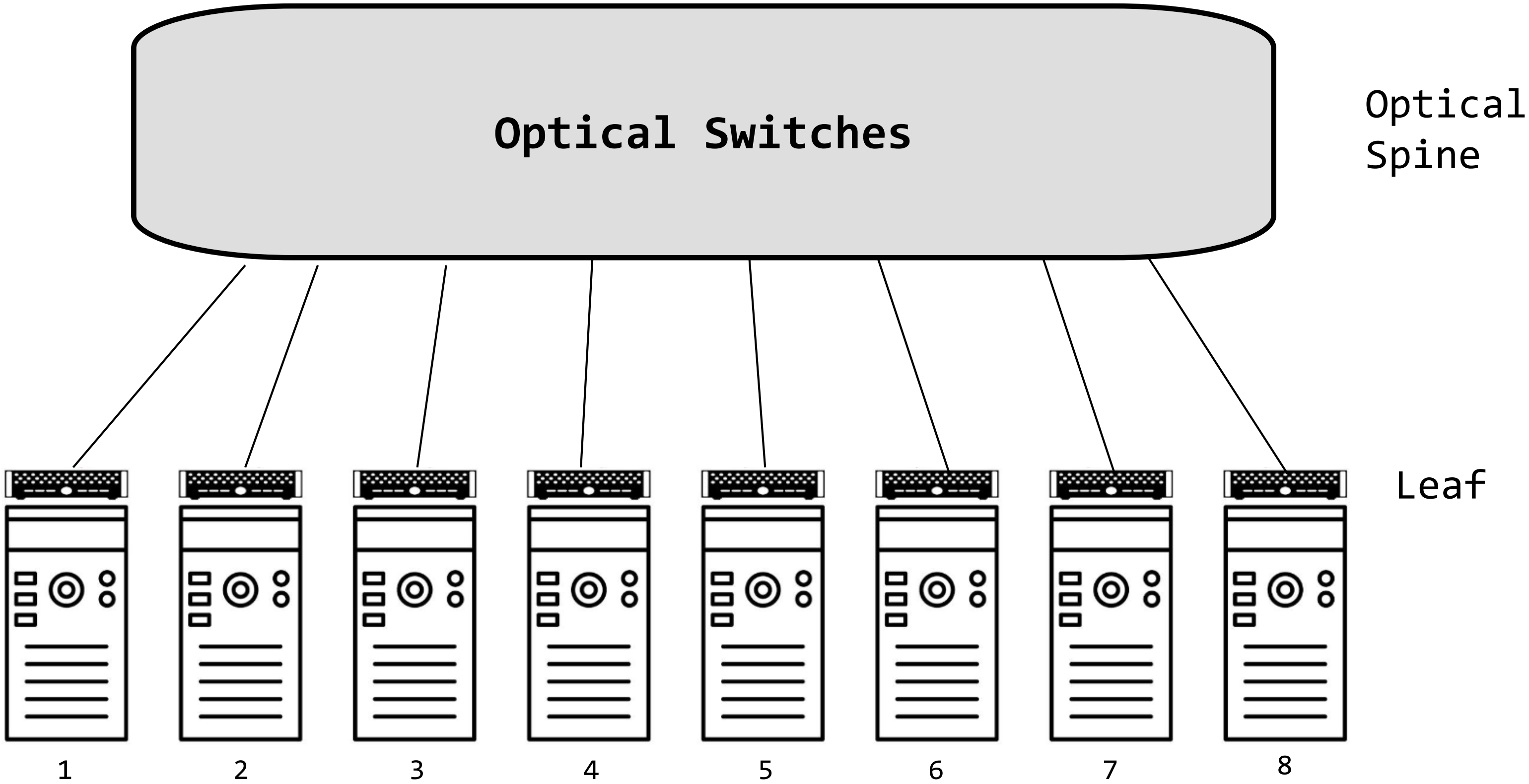
**Cerberus**

**Throughput**



# Unified Network Model

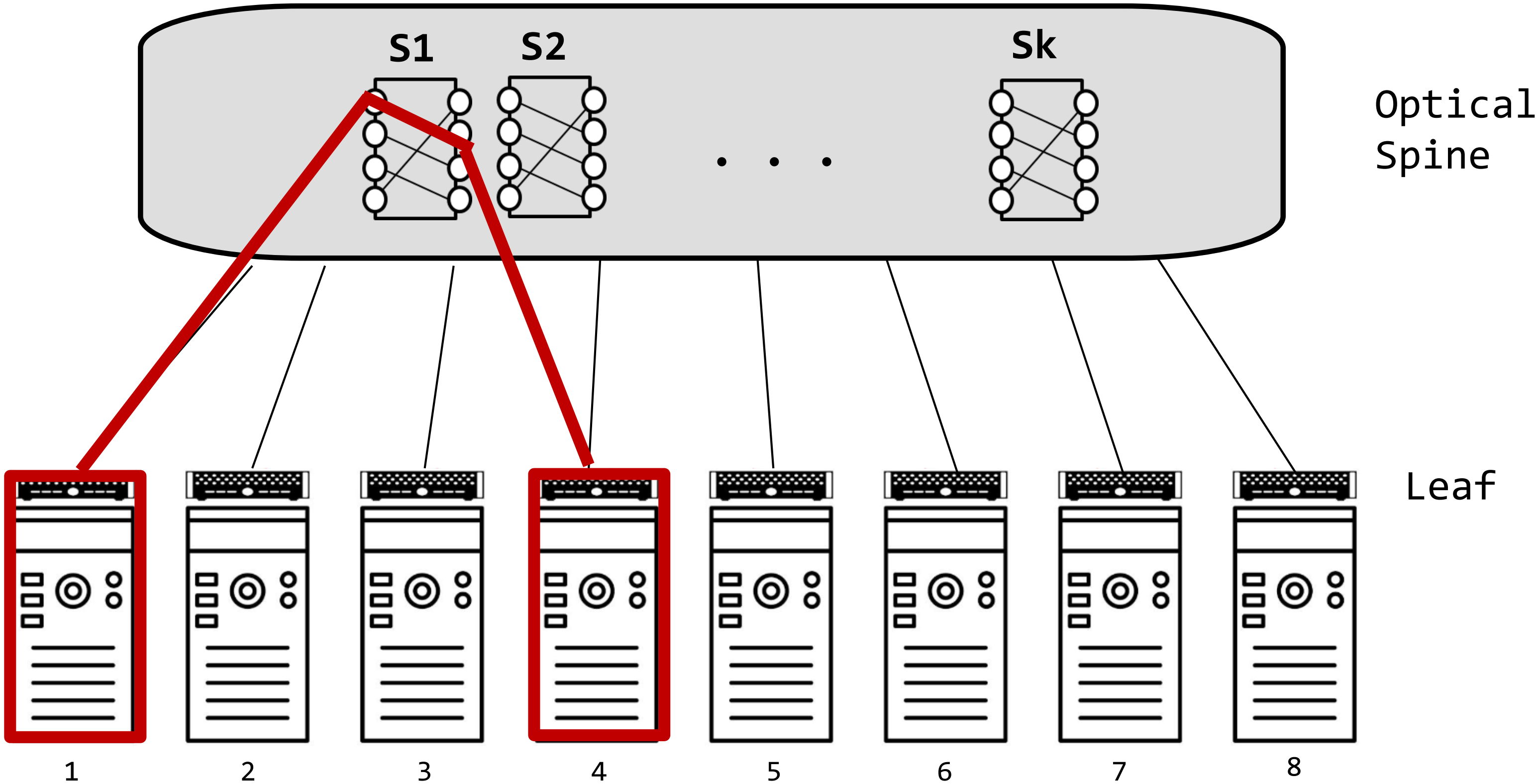
## Two-Layers ToR Interconnect




Typical rack internconnect: **ToR-Matching-ToR (TMT)** model

# Unified Network Model

## Two-Layers ToR Interconnect

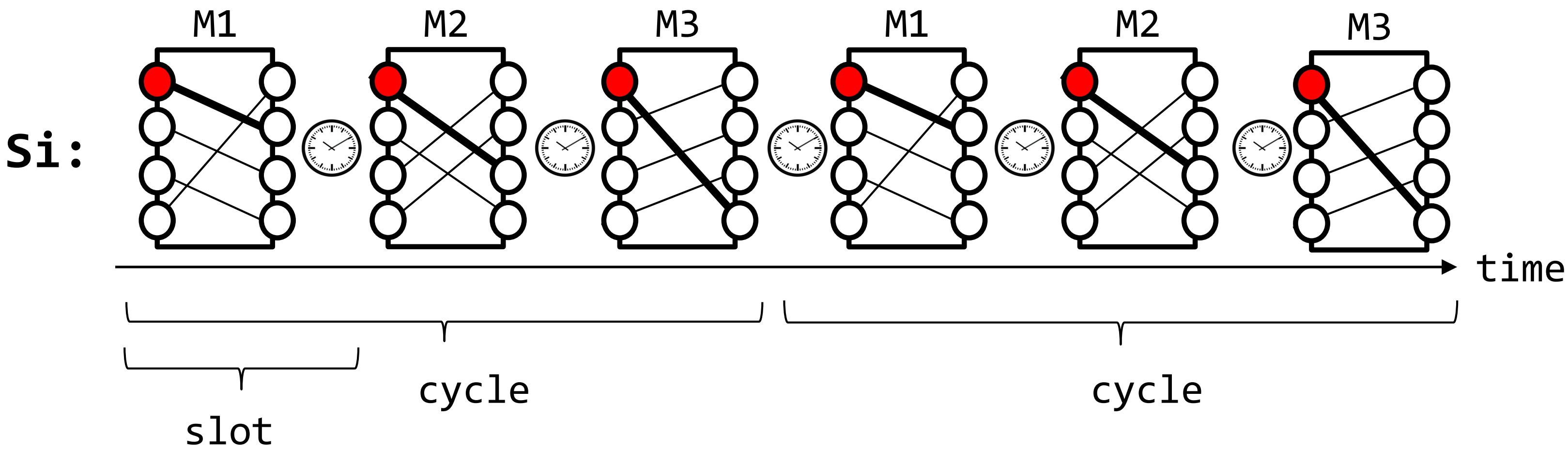


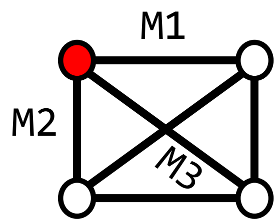

Typical rack internconnect: **ToR-Matching-ToR (TMT)** model

# Periodic Switch (Rotor)

Rotor

**Rotor switch: periodic matchings (demand-oblivious)**

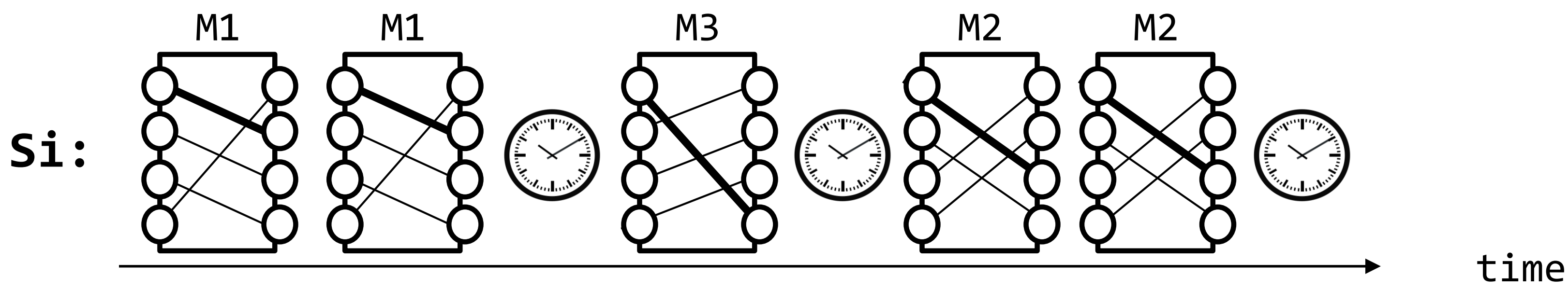


# Demand-Aware Switch

Demand-Aware

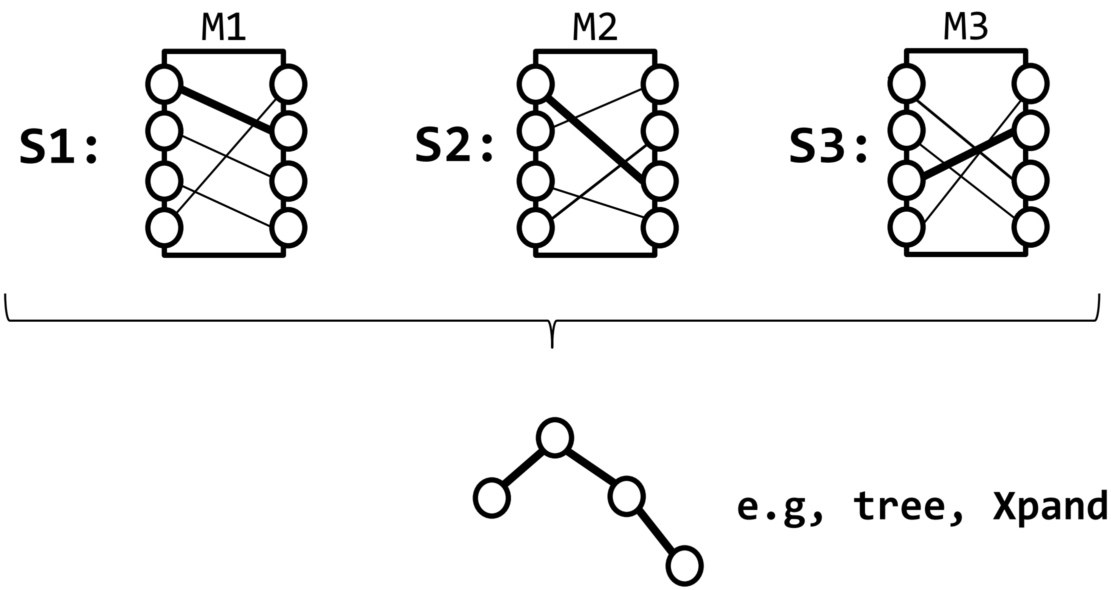
**Demand-aware switch: optimized matchings**



# Static Switch (Patch Panel)

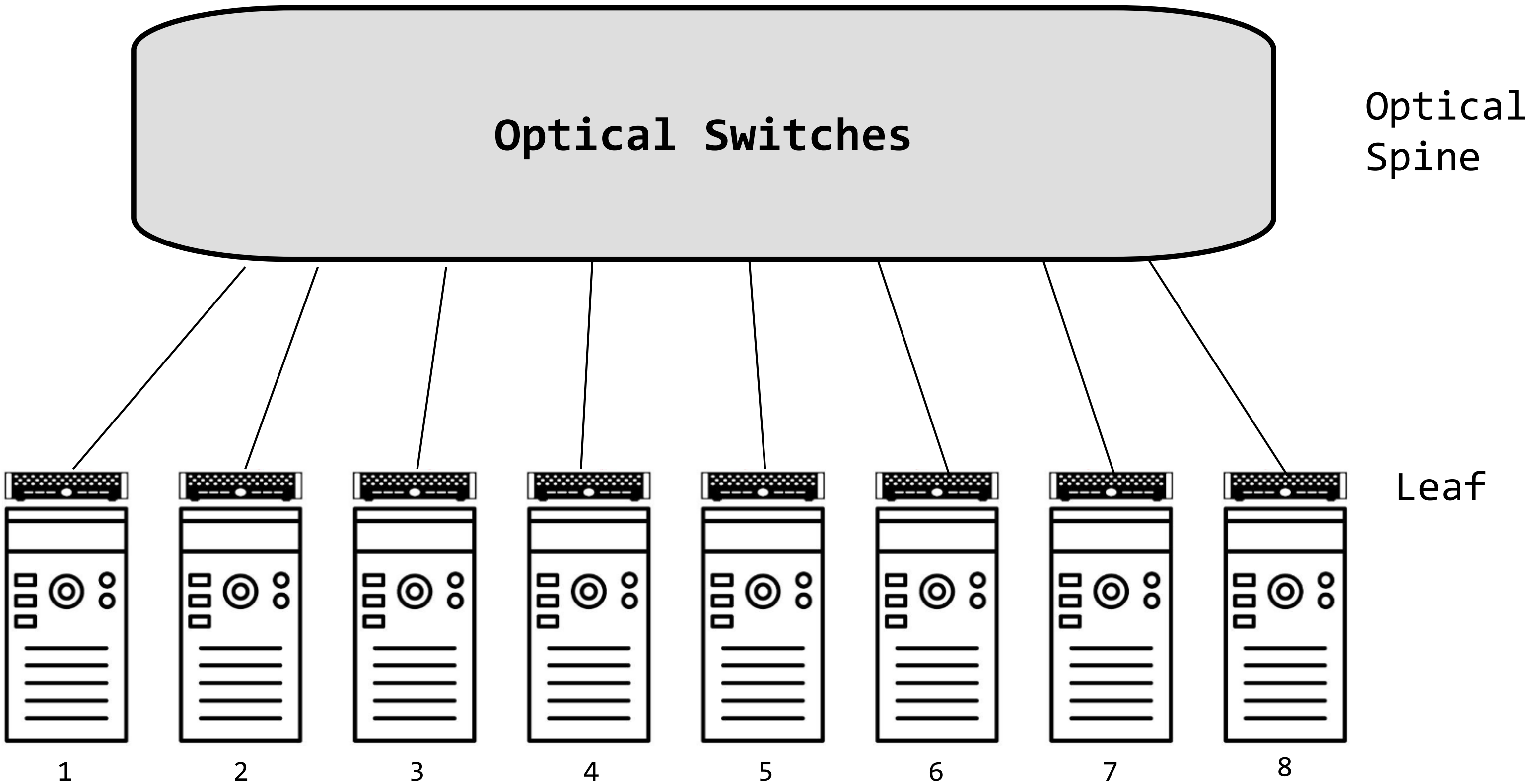
**Static**

**Static switches: combine for optimized static topology**




**e.g, tree, Xpander, Clos**

# Unified Model: From Switches to Topologies



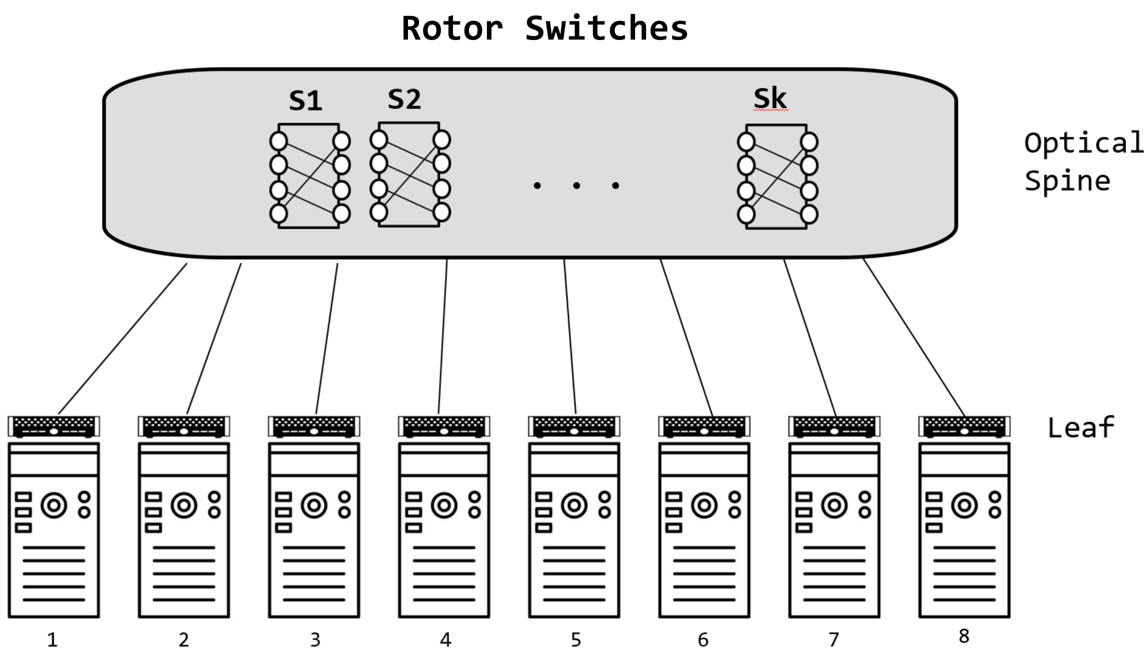

Typical rack internconnect: **ToR-Matching-ToR (TMT)** model

Unified Model

# Rotor-Net

Rotor

- All spine switches are rotor switches
- Can use 1 or 2 hop routings (VLB)
- Emulating a **complete graph** using (TDMA)

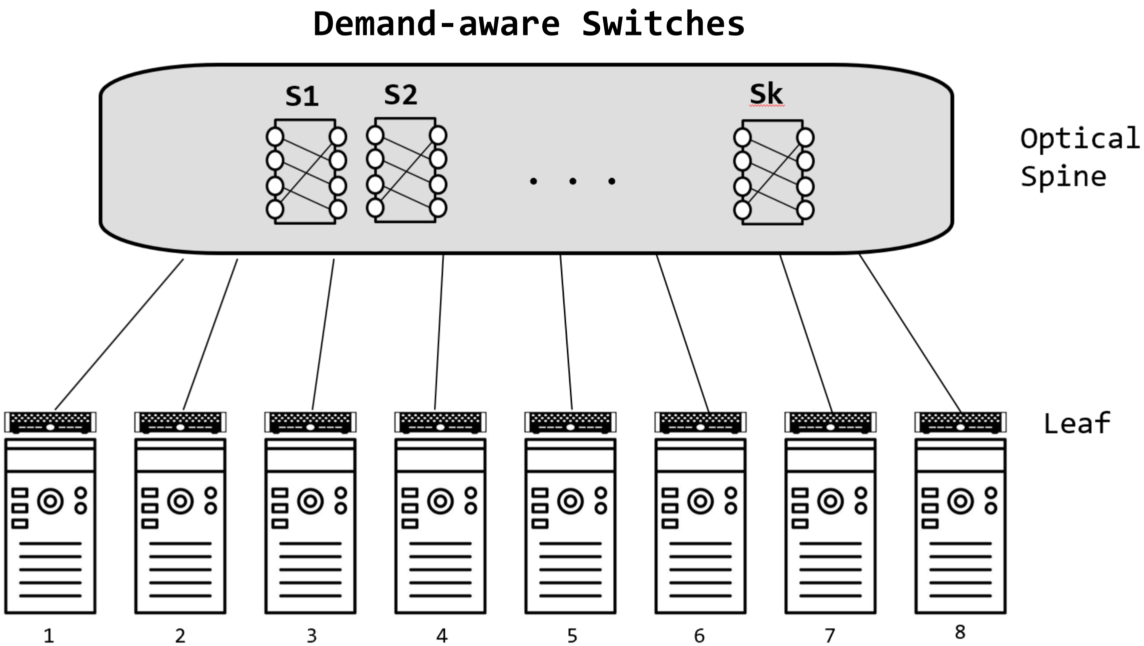



Mellette et al., SIGCOMM 2017
Ballani et al., SIGCOMM 2020

# Demand-Aware Net

**Demand-Aware**

- All spine switches are demand-aware switches
- Can use only 1 hop routings (multi-hop, in on-going work)
- Temporal / dynamic network

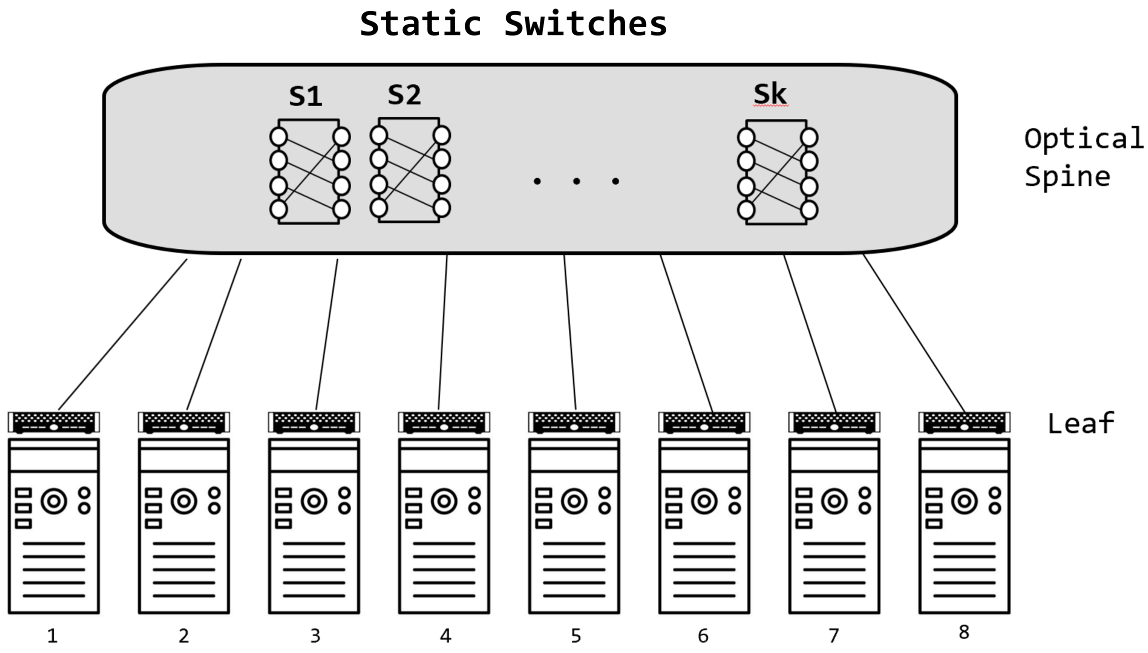



e.g., Ghobadi et al., SIGCOMM 2016

# Expander-Net

**Static**

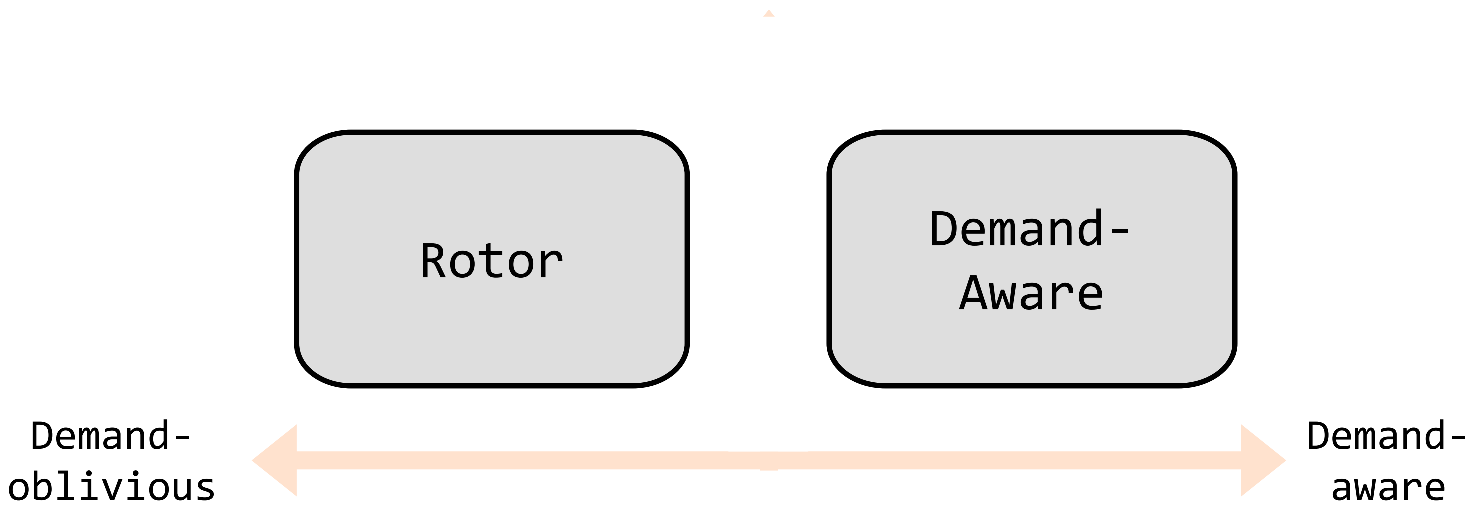
→ All spine switches are static switches
→ Uses multi-hop routing
→ Use known static topologies: e.g., Xpander*, Clos, electrical




*Valadarsky et al., CoNEXT 2016

# Design Tradeoffs (1)

## The “Awareness-Dimension”

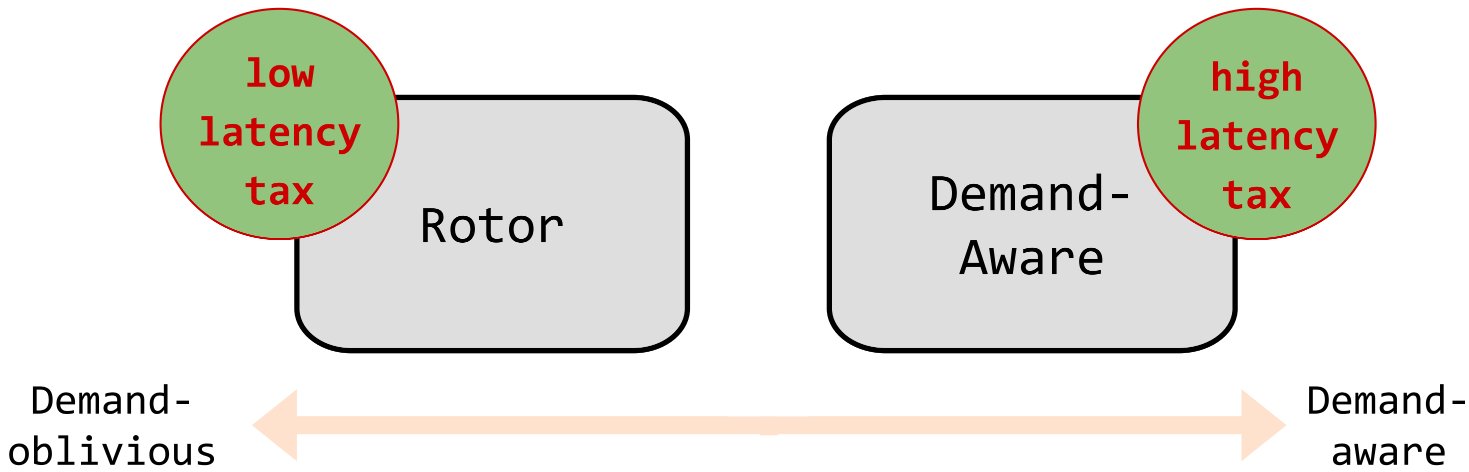



**Good for all-to-all traffic!**
→ Oblivious: very fast periodic direct connectivity
→ Simpler control plane?

**Good for elephant flows!**
→ Optimizable toward traffic
→ But slower

# Design Tradeoffs (1)

## The "Awareness-Dimension"




**Good for all-to-all traffic!**

→ Oblivious: very fast periodic direct connectivity

→ Simpler control plane?

**Good for elephant flows!**

→ Optimizable toward traffic

→ But slower

**Compared to static networks: latency tax!**

# Design Tradeoffs (2)

## The “Flexibility-Dimension”

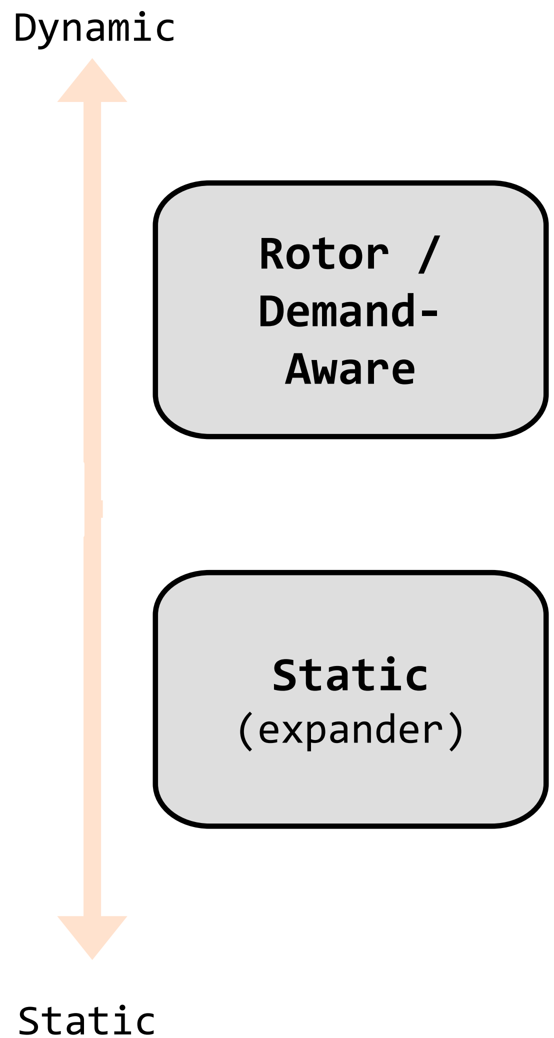
**Good for high throughput!**

⇀ direct connectivity saves bandwidth along links

**Good for low latency!**

⇀ no need to wait for reconfigurable links

⇀ **compared to dynamic: bandwidth tax (multi-hop)**



# Design Tradeoffs (2)

## The “Flexibility-Dimension”

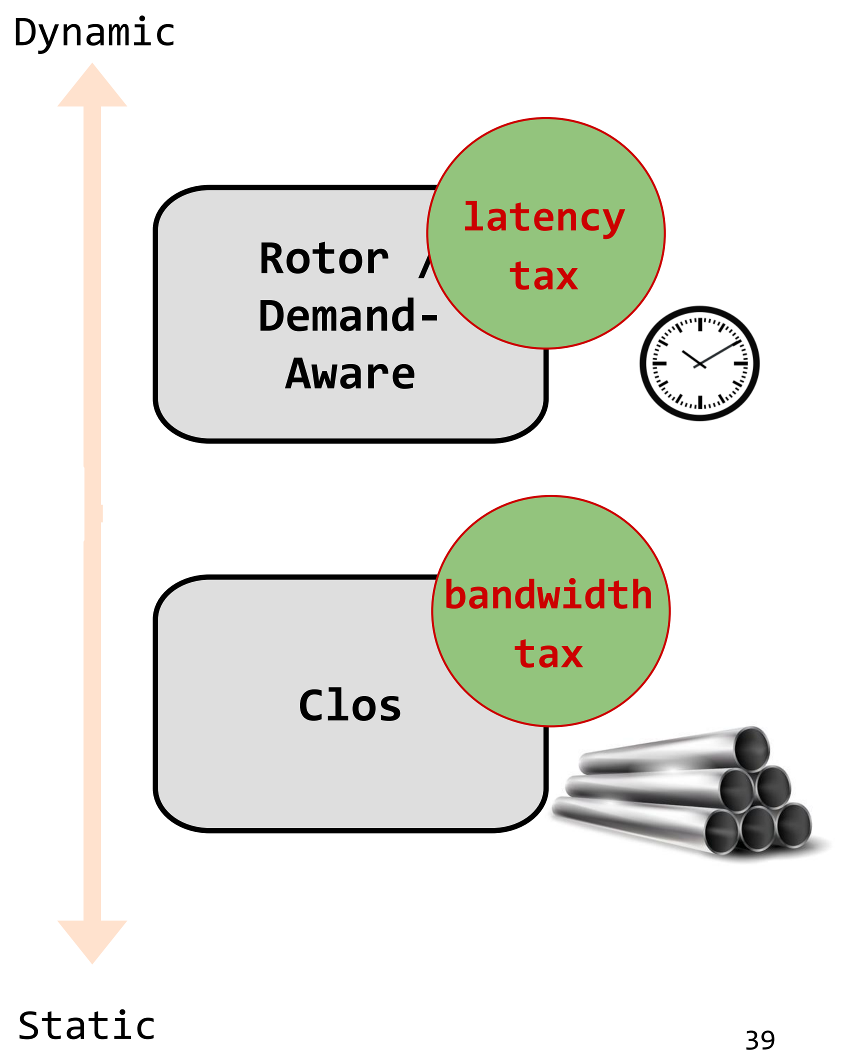
**Good for high throughput!**

→ Direct connectivity saves bandwidth along links

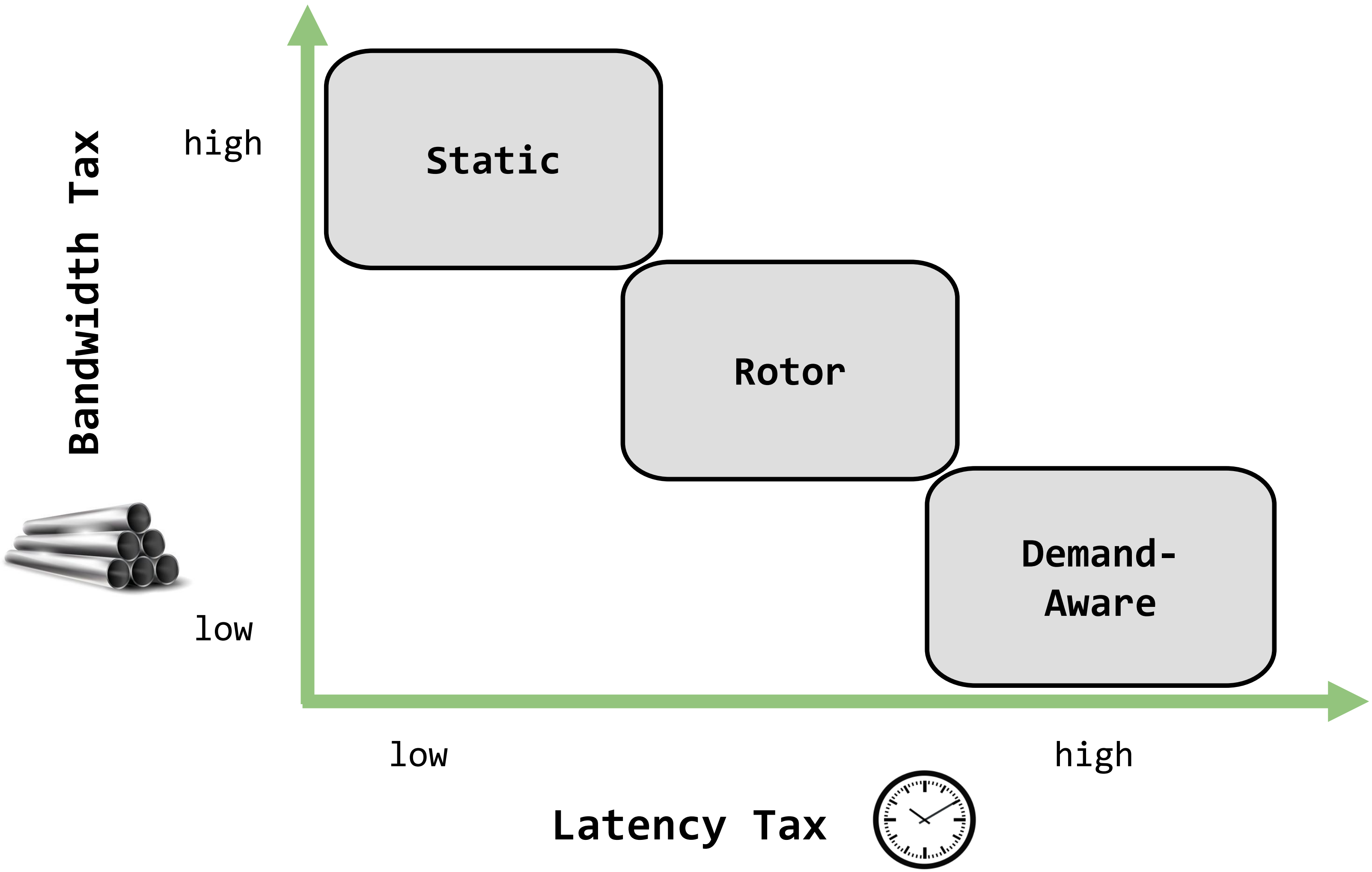
**Good for low latency!**

→ No need to wait for reconfigurable links

→ **Compared to dynamic: bandwidth tax (multi-hop)**



# Summary: Tax Map



# The Spectrum of Traffic

**Diverse patterns:**

⇀ Shuffling/Hadoop:
all-to-all

⇀ All-reduce/ML: ring or
tree traffic patterns

- ⇀ Elephant flows

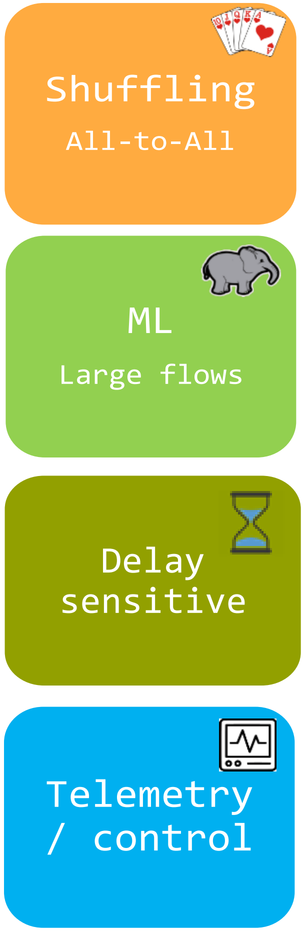
⇀ Query traffic: skewed

- ⇀ Mice flows

⇀ Control traffic: does not evolve
but has non-temporal structure

**Diverse requirements:**

⇀ ML is bandwidth hungry,
small flows are latency-
sensitive



# Main Observations

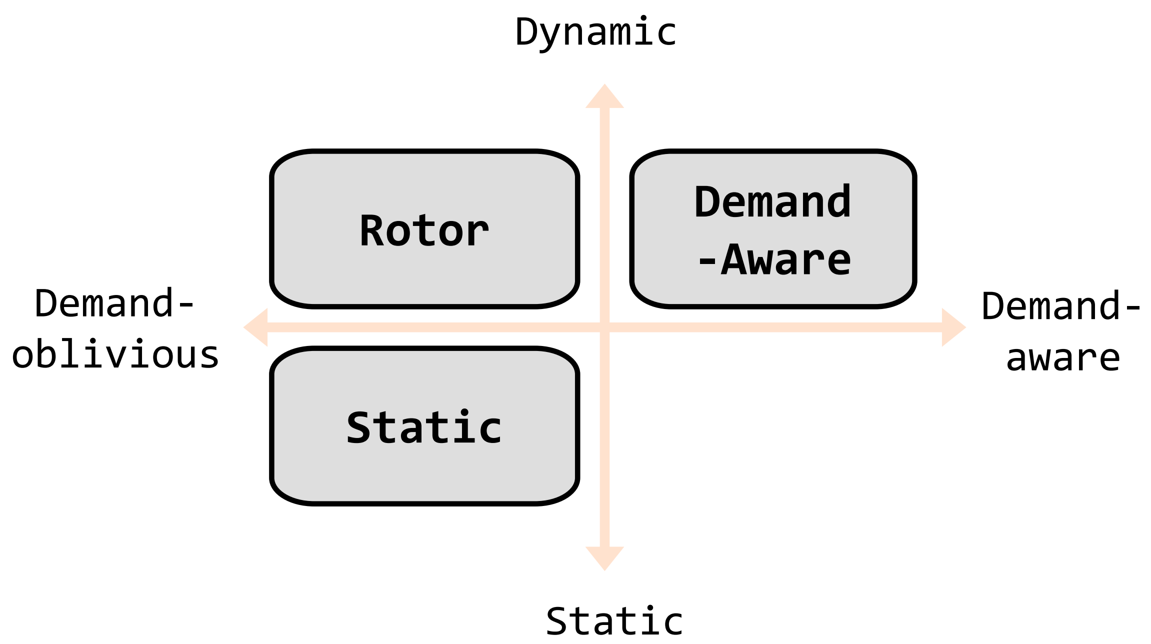
- ⇢ **Observation 1:** Different topologies provide different tradeoffs.
- ⇢ **Observation 2:** Different traffic requires different topology types.
- ⇢ **Observation 3:** A **mismatch of demand** and topology can decrease **throughput** and increase **flow completion times**.

# Main Observations

- **Observation 1:** Different topologies provide different tradeoffs.
- **Observation 2:** Different traffic requires different topology types.
- **Observation 3:** A **mismatch of demand** and topology can decrease **throughput** and increase **flow completion times**.

**So: Can we match traffic to topology?**

# Examples:
# Match or Mismatch?



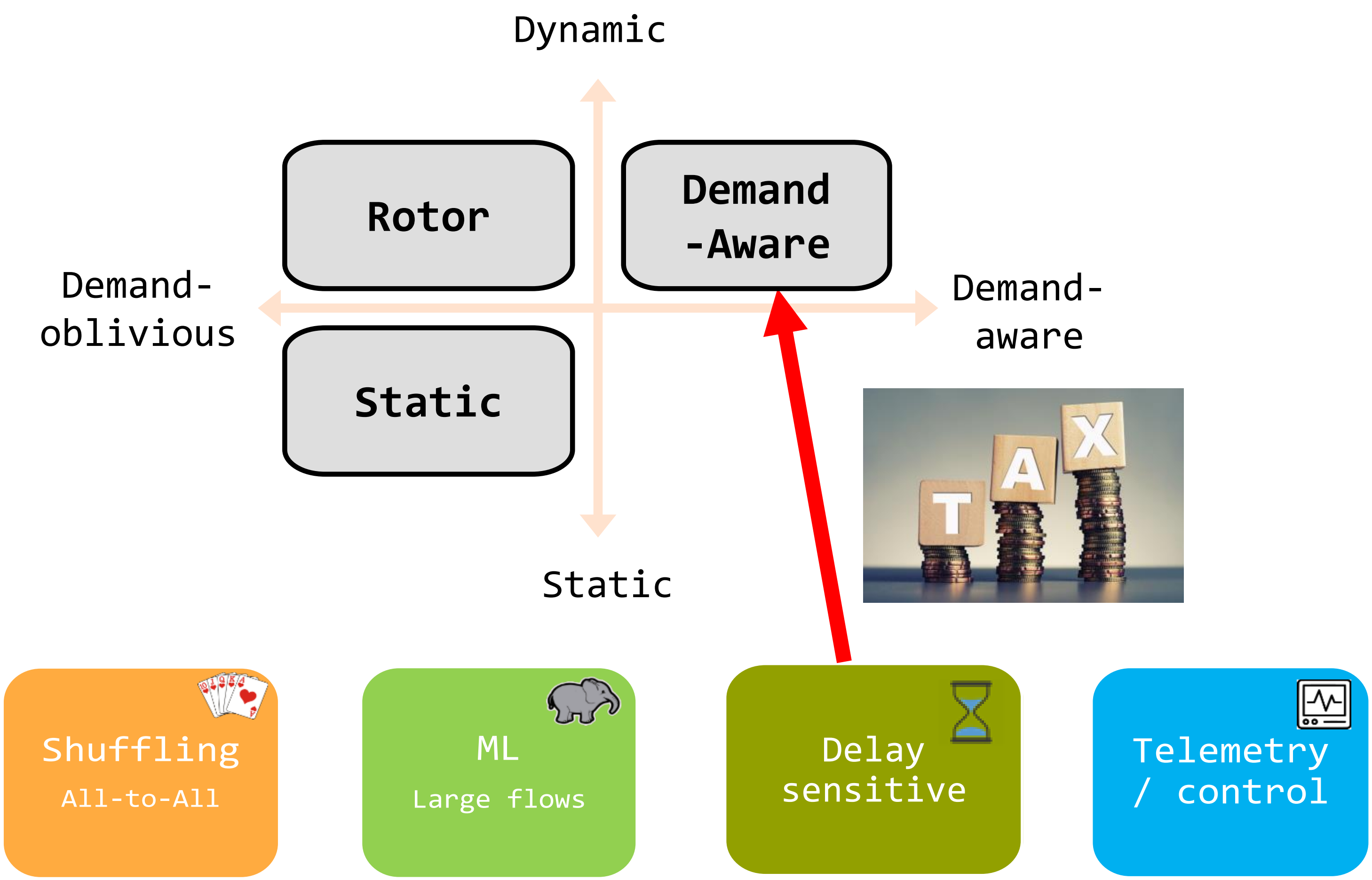

**Shuffling**
All-to-All

**ML**
Large flows

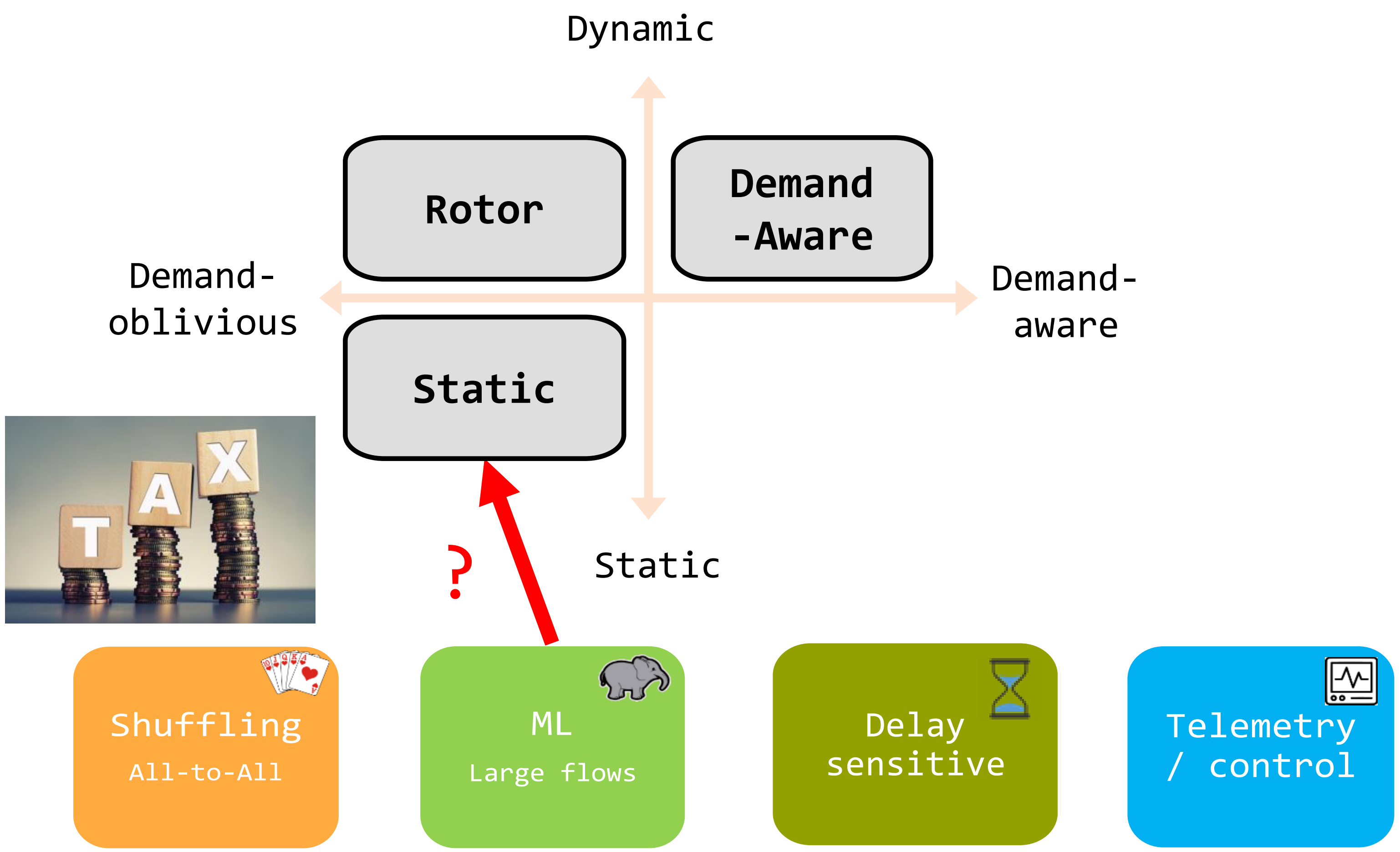
**Delay sensitive**

**Telemetry / control**

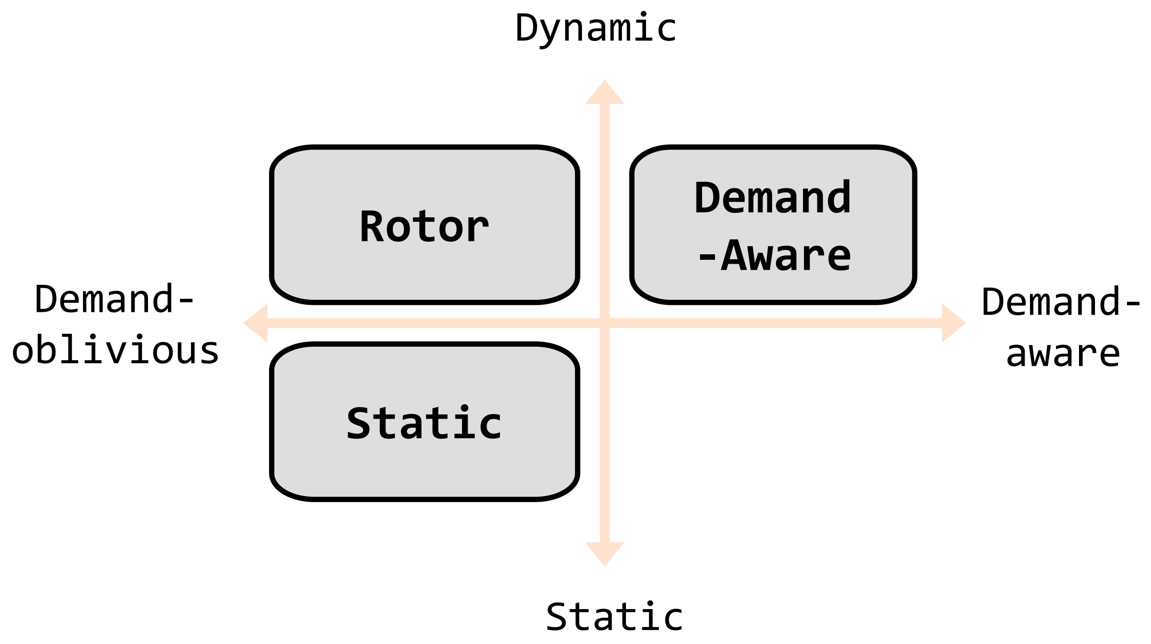
# Examples:
# Match or Mismatch?



# Examples:
# Match or Mismatch?



# Cerberus: It's a Match!



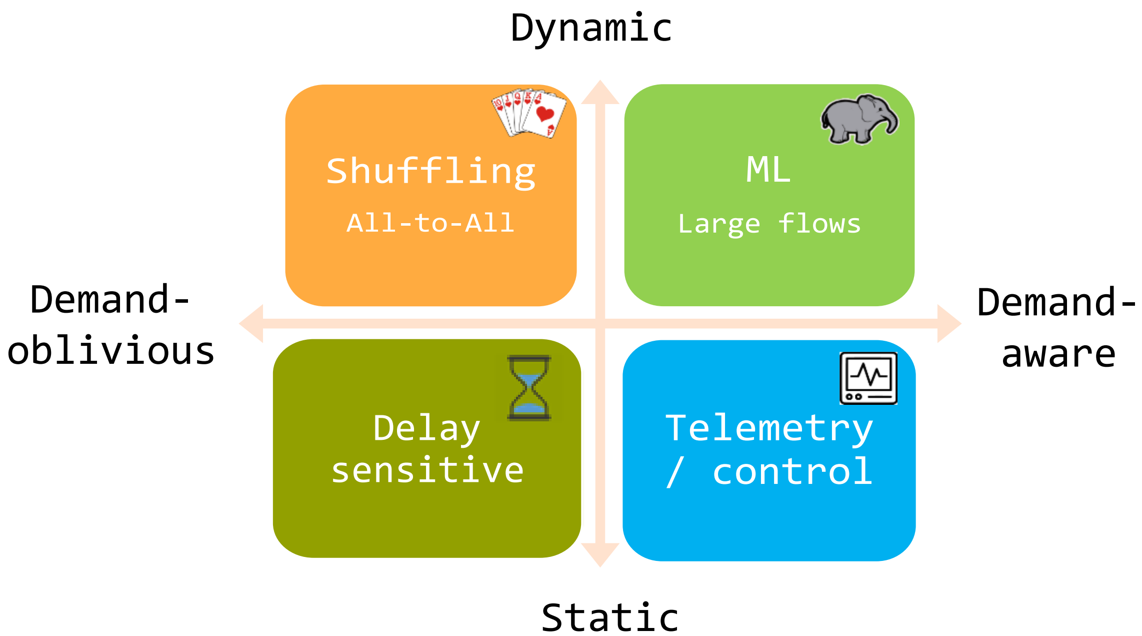

**Shuffling**
All-to-All

**ML**
Large flows

**Delay sensitive**

**Telemetry / control**

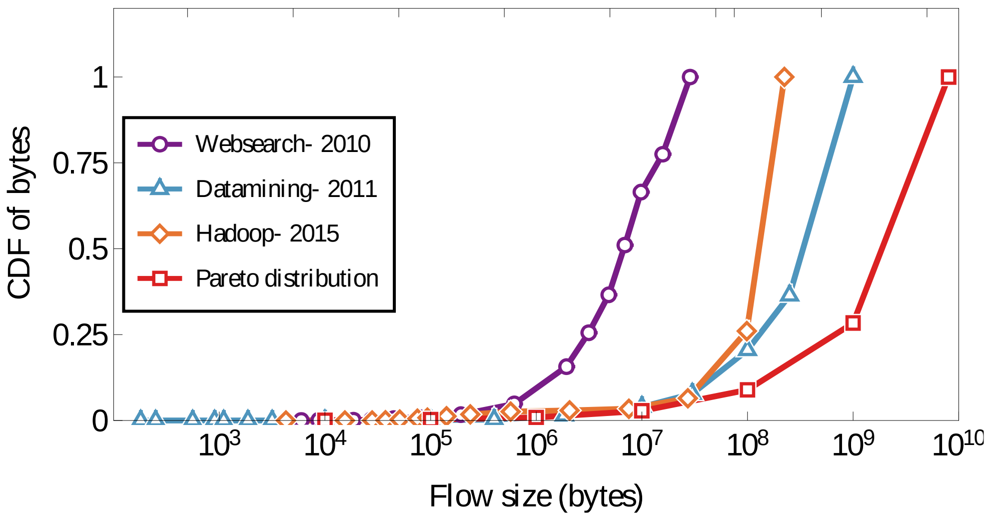
# Cerberus: It's a Match!




Our system Cerberus* serves traffic on the “best topology”!

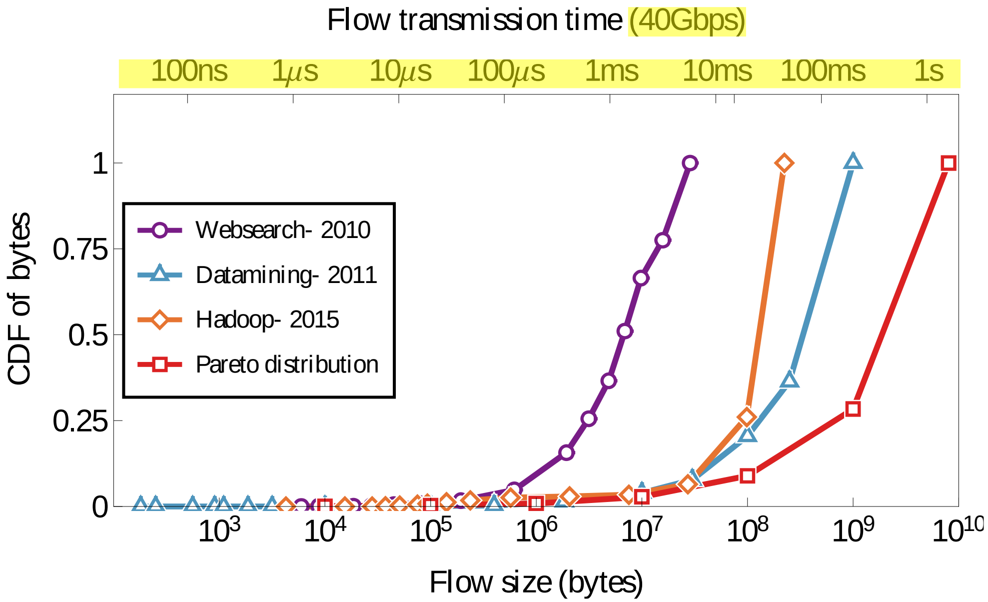
* Griner et al., ACM SIGMETRICS 2022

# Flow Size Matters




⇢ **Observation 1:** Most flows are small, most bytes in big flows.

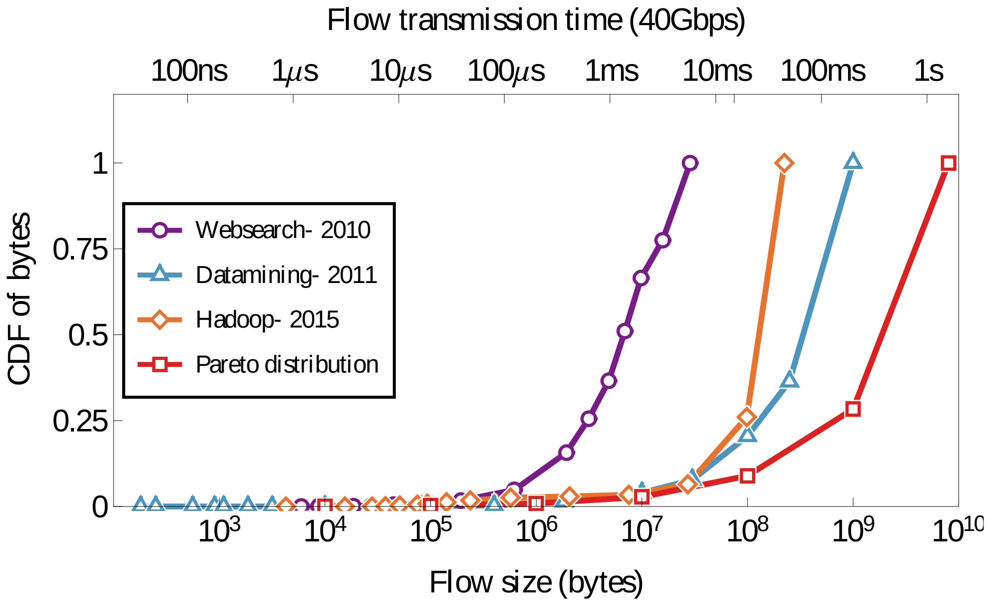
# Flow Size Matters




⇢ **Observation 1:** Most flows are small, most bytes in big flows.

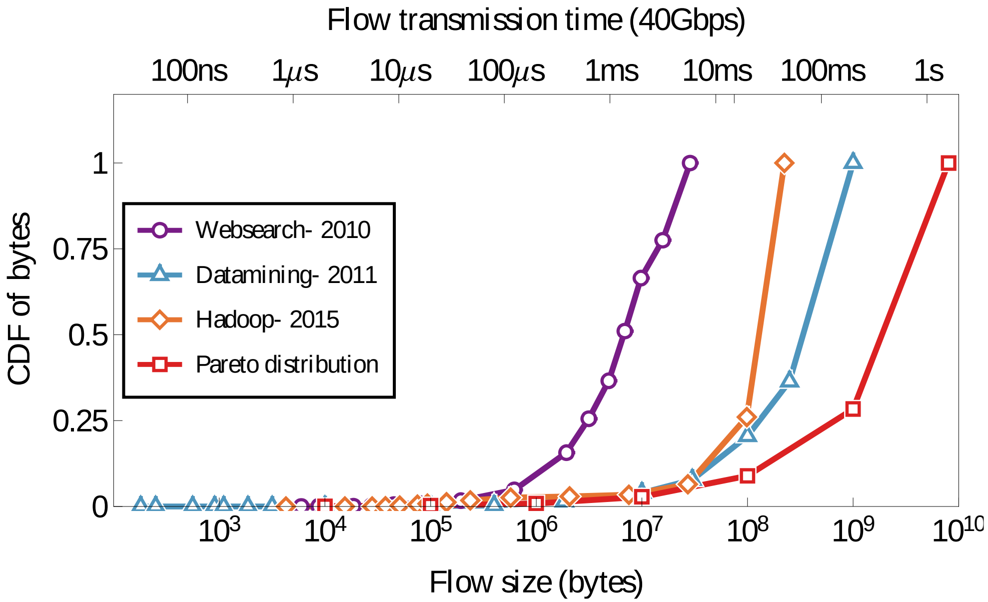
⇢ **Observation 2:** The transmission time of a flow depends on its size.

# Flow Size Matters



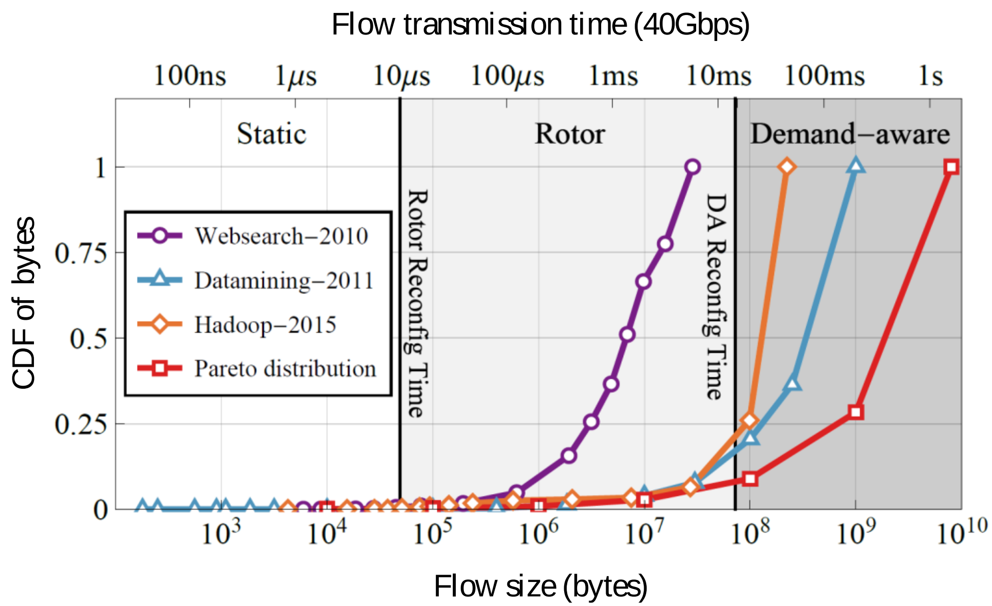

- **Observation 1:** Most flows are small, most bytes in big flows.
- **Observation 2:** The transmission time of a flow depends on its size.
- **Observation 3:** For small flows, flow completion time suffers if network needs to be reconfigured first.

# Flow Size Matters




- **Observation 1:** Most flows are small, most bytes in big flows.
- **Observation 2:** The transmission time of a flow depends on its size.
- **Observation 3:** For small flows, flow completion time suffers if network needs to be reconfigured first.
- **Observation 4:** For large flows, reconfiguration time may amortize.

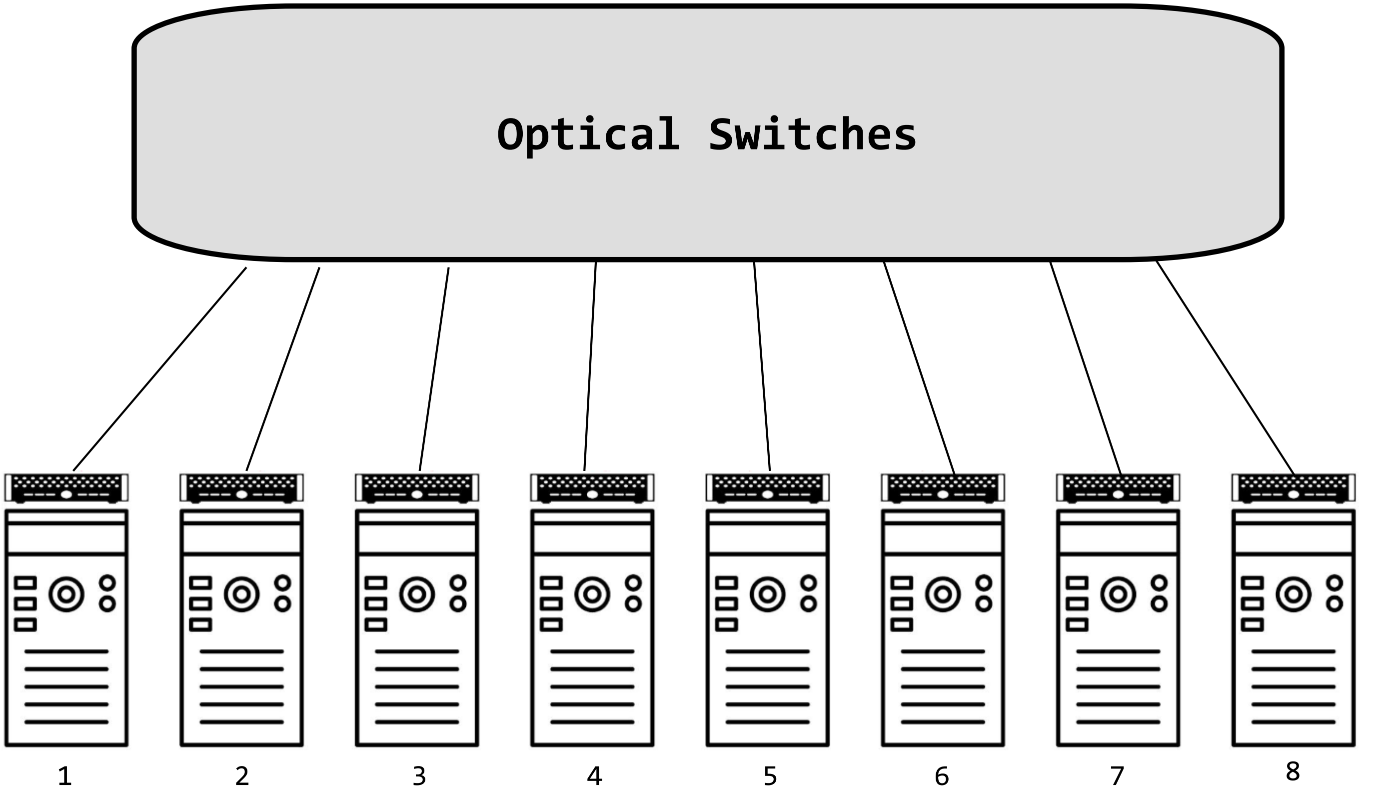
# Flow Size Matters



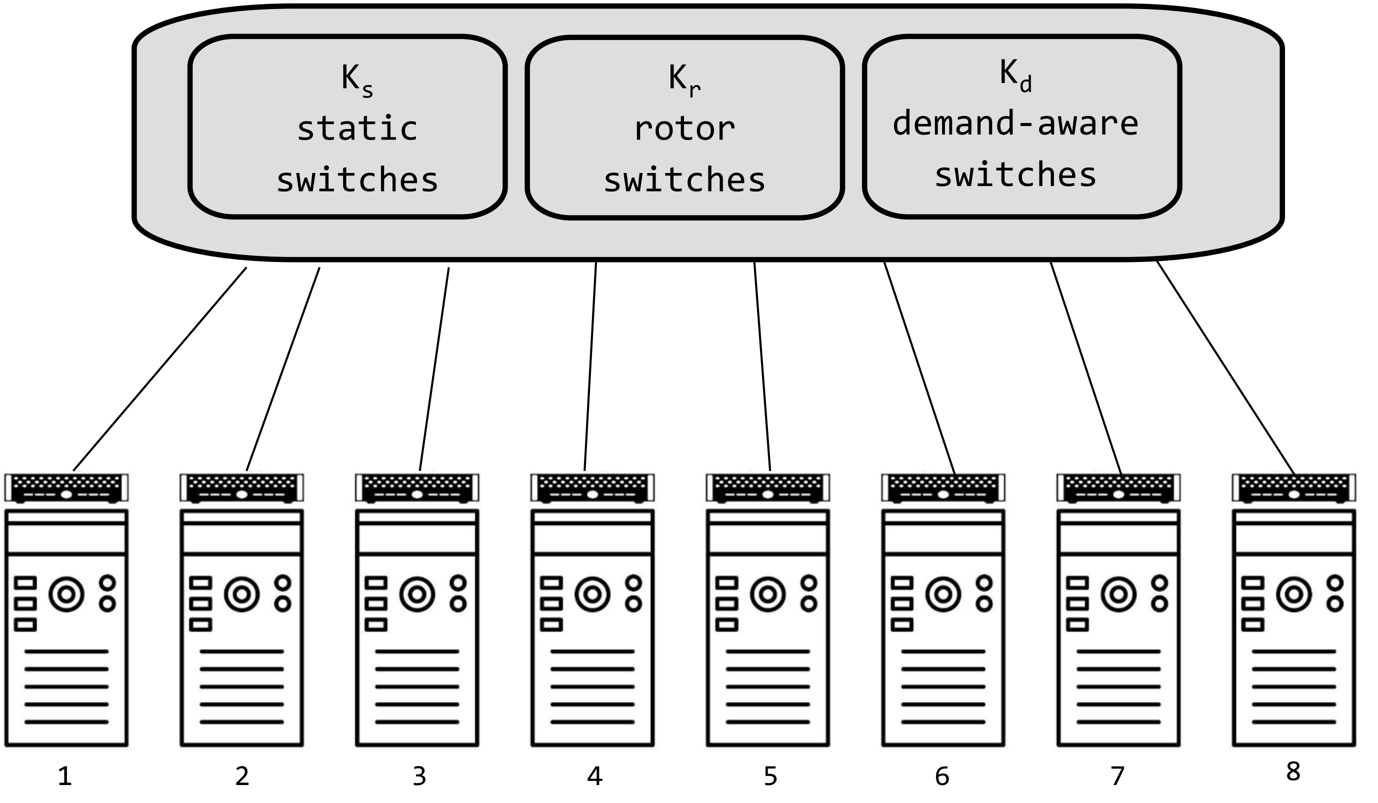

It's a Match!

- **Observation 1:** Most flows are small, most bytes in big flows.
- **Observation 2:** The transmission time of a flow depends on its size.
- **Observation 3:** For small flows, flow completion time suffers if network needs to be reconfigured first.
- **Observation 4:** For large flows, reconfiguration time may amortize.

# Cerberus

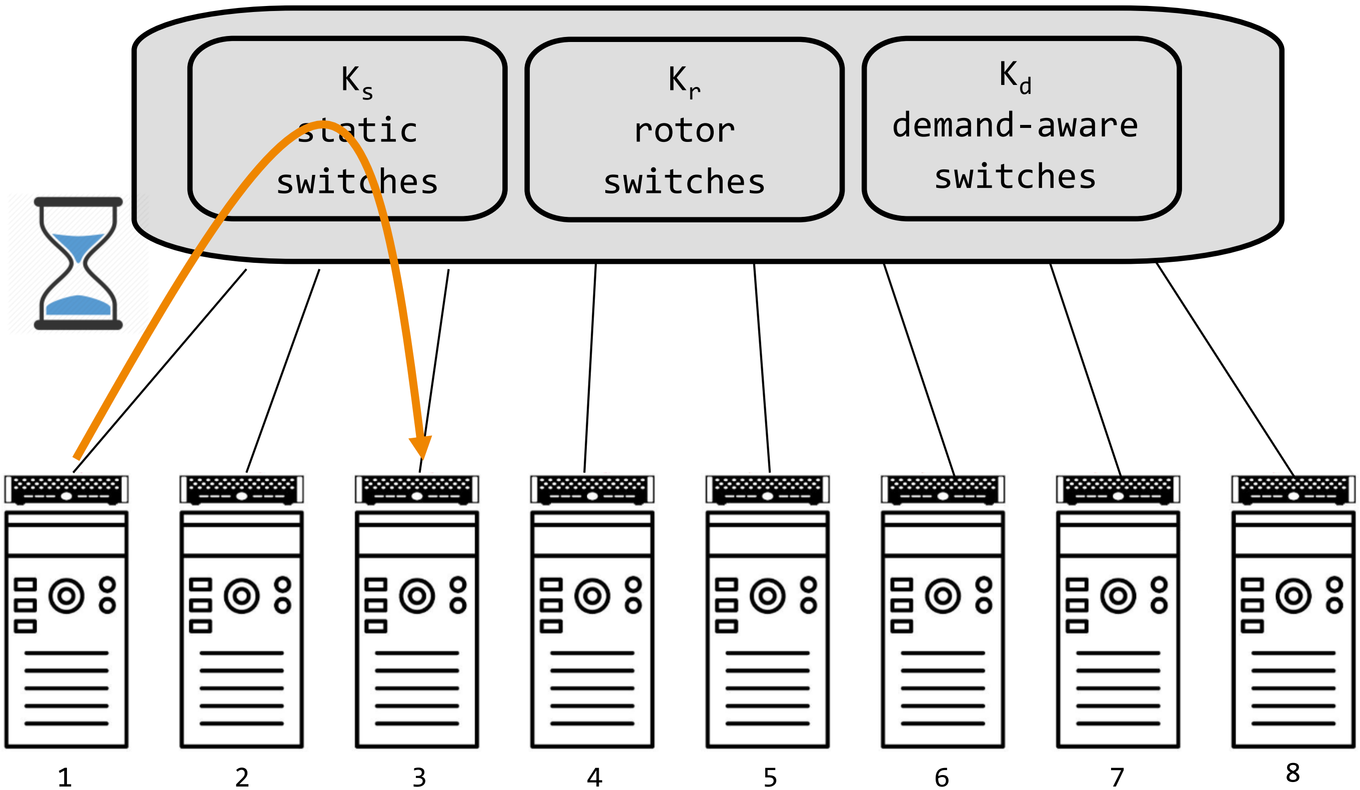


# Cerberus*



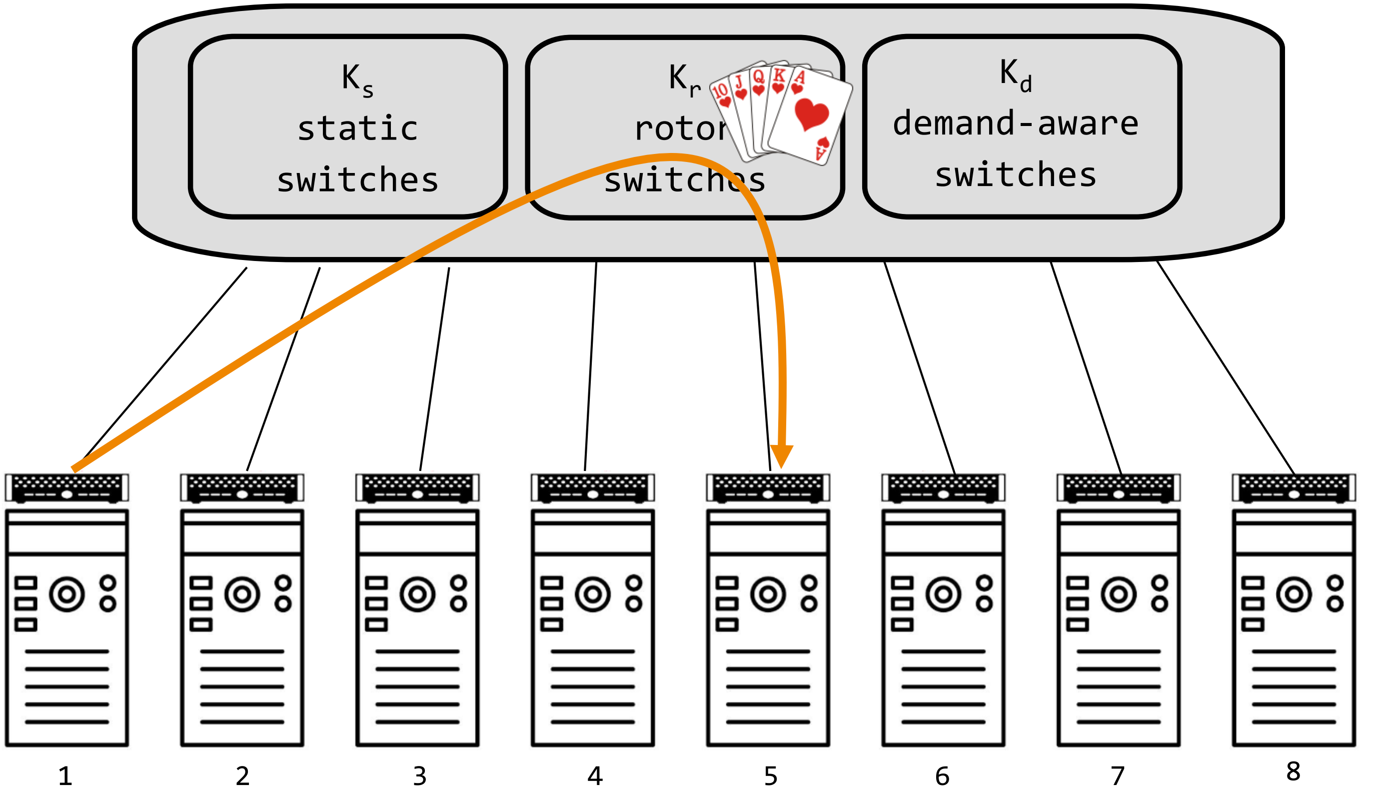

* 3-headed dog from Greek mythology

# Cerberus



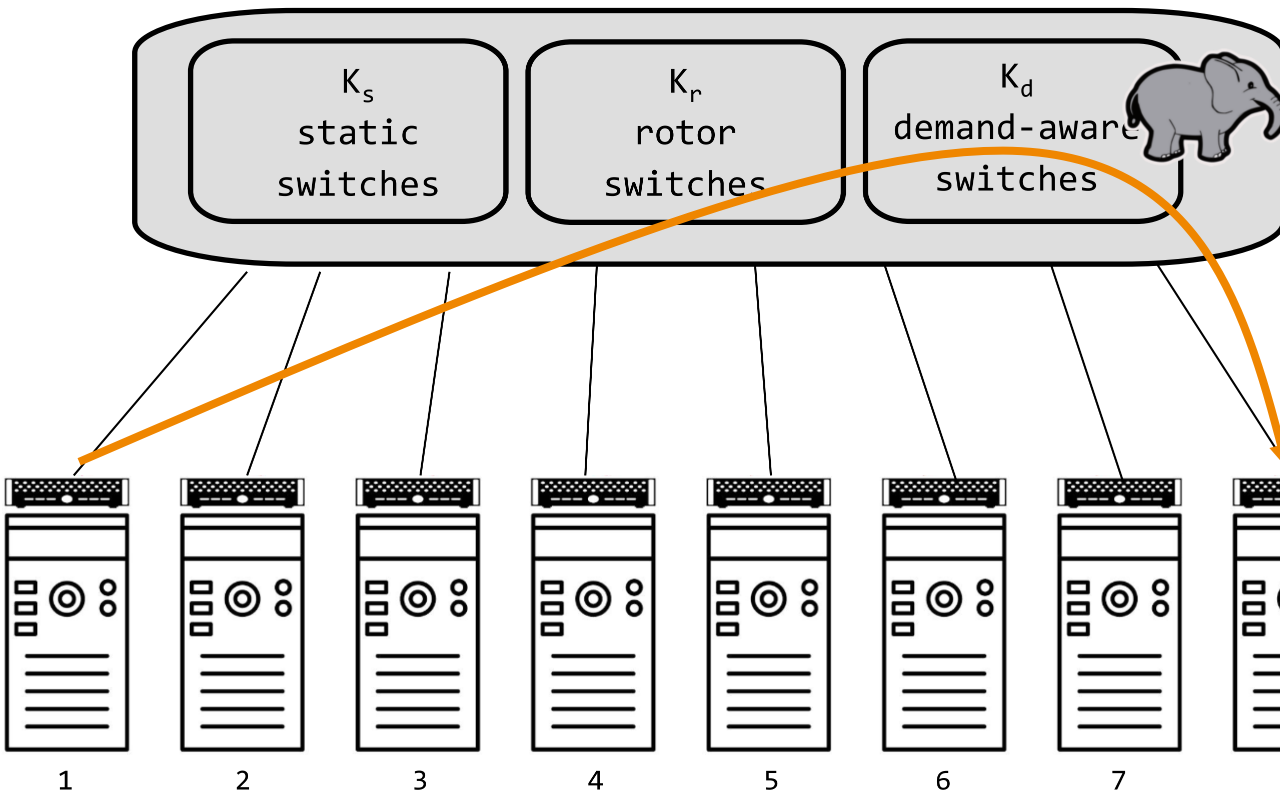

**Scheduling:** Small flows go via static switches…

# Cerberus




**Scheduling:** … medium flows via rotor switches…

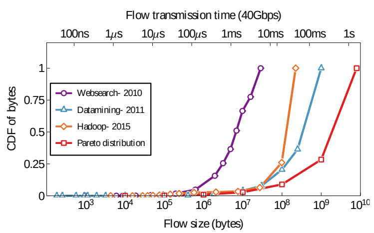
# Cerberus




**Scheduling:** … and large flows via demand-aware switches
(if one available, otherwise via rotor).

# Cerberus Framework

**flow sizes**




**n ToRs**
**k spine switches**
**reconfig times**
$R_r, R_d, \delta$

**Cerberus**

**Optimal Partition**
(static, rotor, demand-aware)

**flow size thresholds**
(small, medium, large)

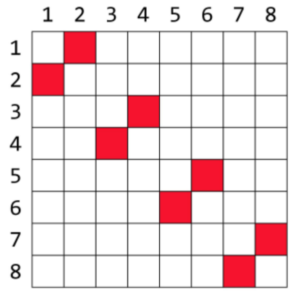
**Throughput analysis**

vs Rotor-Net and Expander-Net

# Throughput Analysis

Demand Matrix

$T$

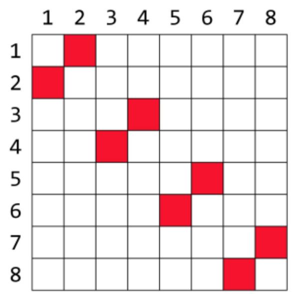



**Metric:** throughput of a demand matrix…

Abdu et al., SC 2016
Namyar et al., SIGCOMM 2021

# Throughput Analysis

Demand Matrix

$T$  $\times\ \theta(T)$



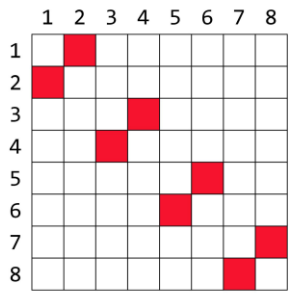
**Metric:** throughput of a demand matrix… … is the maximal scale down factor by which traffic is feasible $0 \leq \theta(T) \leq 1$.

Abdu et al., SC 2016
Namyar et al., SIGCOMM 2021

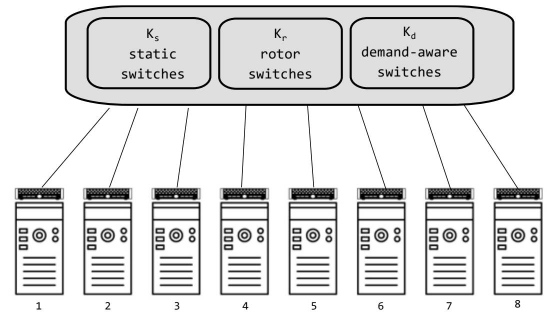
# Throughput Analysis

Demand Matrix

$T$



$\times\ \theta(T)\ \Rightarrow$




**Metric:** throughput of a demand matrix…

… is the maximal scale down factor by which traffic is feasible $0 \leq \theta(T) \leq 1$.

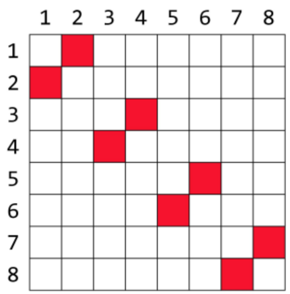
Throughput of network $\boldsymbol{\theta}^*$: worst case $T$

Abdu et al., SC 2016
Namyar et al., SIGCOMM 2021

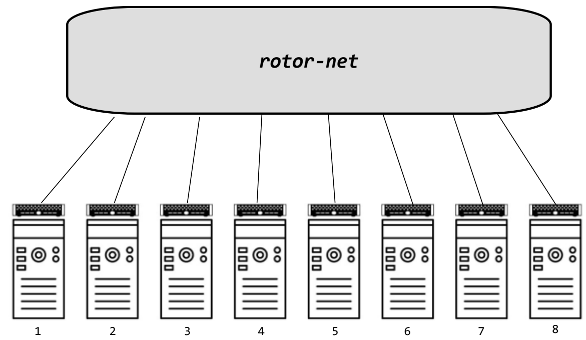
# Throughput: Rotor-Net

Demand Matrix

$T$




Permutation matrix




$$\theta(T) \leq \frac{1}{2-\phi(T)} \cdot \frac{\delta}{R_r + \delta}$$

Skew parameter

Bandwidth tax

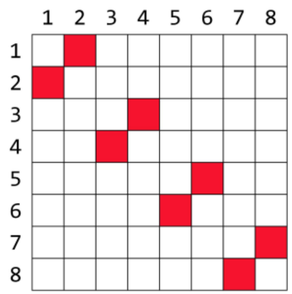
Latency tax

$$\theta^* \leq \frac{n}{2n-1} \cdot \frac{\delta}{R_r + \delta}$$

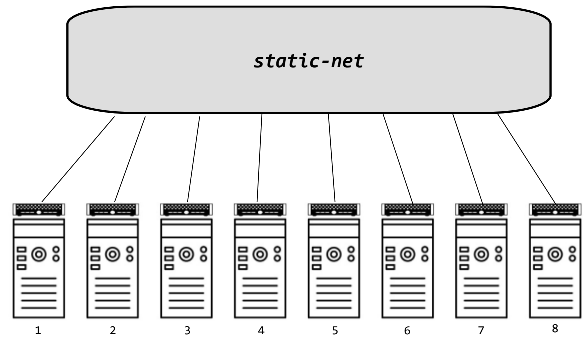
# Throughput: Expander-Net

Demand Matrix

$T$




Permutation matrix




$$\theta^* \leq \frac{1}{\mathrm{epl}(G(k))}$$

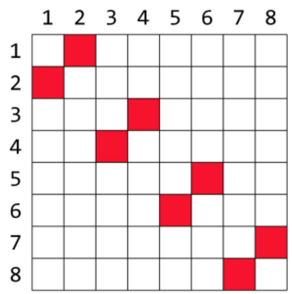

Bandwidth tax

Expected path length

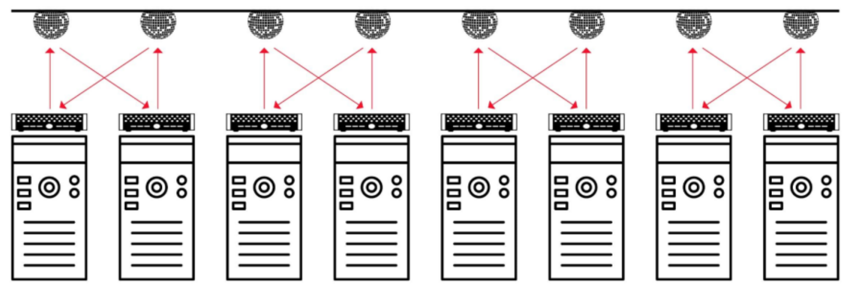
# Throughput: Demand-Aware

Demand Matrix

$T$




Permutation matrix

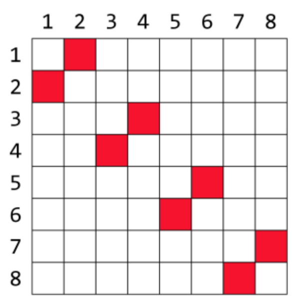


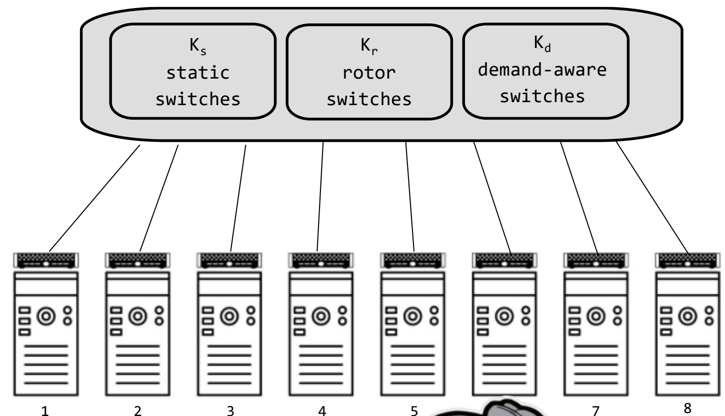
Permutation matrix is the best demand matrix for demand-aware net!

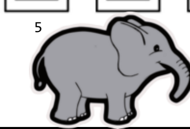
# Throughput: Cerberus

Demand Matrix

$T$



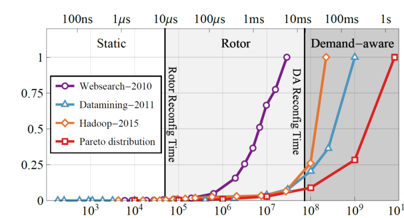
$K_s$
static
switches

$\overset{\theta}{=}$

$K_r$
rotor
switches

$\overset{\theta}{=}$

$K_d$
demand-aware
switches




$$\theta(T) = \frac{\hat{T}(1,\ell)}{nk_d^*}\left(R_d\mathbb{E}\left[\frac{1}{|f|}\right] + \frac{1}{r}\right)$$

 Bandwidth tax

Latency tax 

# Throughput: Summary

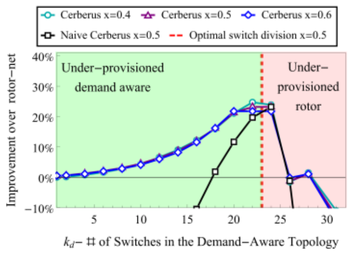
Demand Matrix

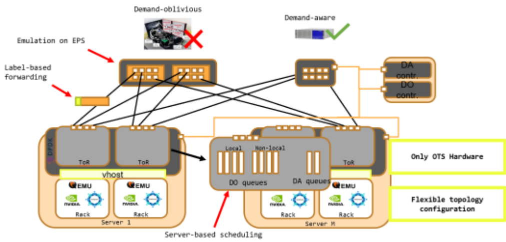

For the given input parameters: $n$, $k$, $R_d$, $R_r$

| | *expander-net* | *rotor-net* | CERBERUS |
|---|---|---|---|
| BW-Tax | ✓ | ✓ | ✗ |
| LT-Tax | ✗ | ✓ | ✓ |
| $\theta(T)$ | Thm 2 | Thm 3 | Thm 5 |
| $\theta^*$ | 0.53 | 0.45 | Open |
| Datamining | 0.53 | 0.6 | 0.8 (+33%) |
| Permutation | 0.53 | 0.45 | $\approx$ 1 (+88%) |
| Case Study | 0.53 | 0.66 | 0.9 (+36%) |

# Conclusion

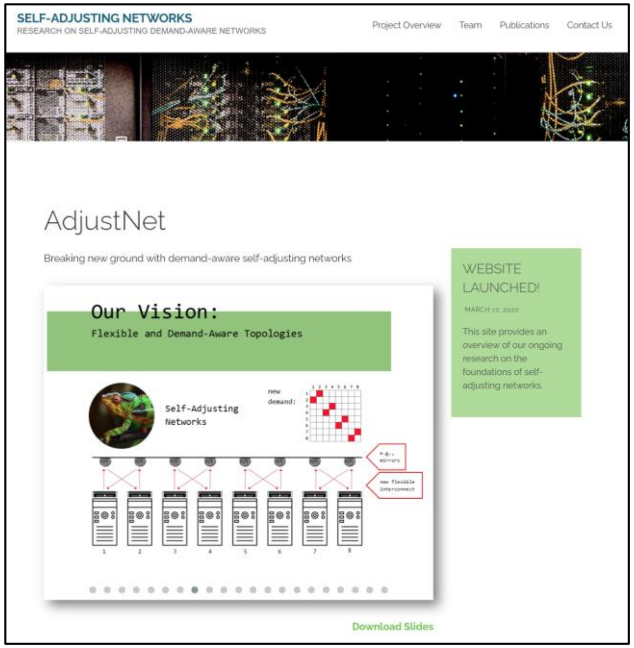
- Diverse traffic requires diverse technologies/topologies
- Cerberus aims to assign traffic to its best topology
  - Depending on flow size
- Skipped: simulations and prototype
- Many challenges
  - Impact on routing and congestion control
  - Sensitivity analysis
  - Simulation & prototyping

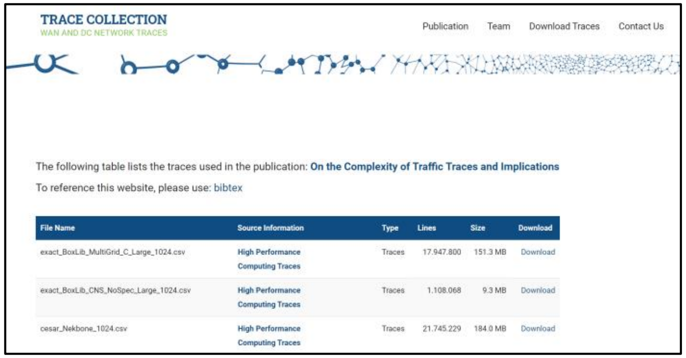



Zerwas et al., ANCS 2021

# Websites




**http://self-adjusting.net/**
**Project website**

**https://trace-collection.net/**
**Trace collection website**

**Thank you!**

# Further Reading

**Cerberus: The Power of Choices in Datacenter Topology Design***

A Throughput Perspective

CHEN GRINER, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel
JOHANNES ZERWAS, Technical University of Munich, Germany
ANDREAS BLENK, Technical University of Munich, Germany
MANYA GHOBADI, Computer Science and Artificial Intelligence Laboratory, MIT, USA
STEFAN SCHMID, Faculty of Computer Science, University of Vienna, Austria
CHEN AVIN, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel

The bandwidth and latency requirements of modern datacenter applications have led researchers to propose various topology designs using static, dynamic demand-oblivious (rotor), and/or dynamic demand-aware switches. However, given the diverse nature of datacenter traffic, there is little consensus about how these designs would fare against each other. In this work, we analyze the throughput of existing topology designs under different traffic patterns and study their unique advantages and potential costs in terms of bandwidth and latency "tax". To overcome the identified inefficiencies, we propose CERBERUS, a unified, two-layer leaf-spine optical datacenter design with three topology types. CERBERUS systematically matches different traffic patterns

**On the Complexity of Traffic Traces and Implications**

CHEN AVIN, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel
MANYA GHOBADI, Computer Science and Artificial Intelligence Laboratory, MIT, USA
CHEN GRINER, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel
STEFAN SCHMID, Faculty of Computer Science, University of Vienna, Austria

This paper presents a systematic approach to identify and quantify the types of structures featured by packet traces in communication networks. Our approach leverages an information-theoretic methodology, based on iterative randomization and compression of the packet trace, which allows us to systematically remove and measure dimensions of structure in the trace. In particular, we introduce the notion of *trace complexity* which approximates the entropy rate of a packet trace. Considering several real-world traces, we show that trace complexity can provide unique insights into the characteristics of various applications. Based on our approach,

# Further Reading

## Static DAN

### Demand-Aware Network Designs of Bounded Degree

Chen Avin  Kaushik Mondal  Stefan Schmid

**Abstract** Traditionally, networks such as datacenter interconnects are designed to optimize worst-case performance under *arbitrary* traffic patterns. Such network designs can however be far from optimal when considering the *actual* workloads and traffic patterns which they serve. This insight led to the development of demand-aware datacenter interconnects which can be reconfigured depending on the workload.

Motivated by these trends, this paper initiates the algorithmic study of demand-aware networks (DANs), and in particular the design of bounded-degree networks. The inputs to the network design problem are a discrete communication request distribution, $\mathcal{D}$, defined over communicating pairs from the node set $V$, and a bound, $\Delta$, on the maximum degree. In turn, our objective is to design an (undirected) demand-aware network $N = (V, E)$ of bounded-degree $\Delta$, which provides short routing paths between frequently communicating nodes distributed across $N$. In particular, the designed network should minimize the *expected path length* on $N$

**1 Introduction**

The problem studied in this paper is motivated by the advent of more flexible datacenter interconnects, such as ProjecToR [29,31]. These interconnects aim to overcome a fundamental drawback of traditional datacenter network designs: the fact that network designers must decide *in advance* on how much capacity to provision between electrical packet switches, e.g., between Top-of-Rack (ToR) switches in datacenters. This leads to an undesirable tradeoff [42]: either capacity is over-provisioned and therefore the interconnect expensive (e.g., a fat-tree provides full-bisection bandwidth), or one may risk congestion, resulting in a poor cloud application performance. Accordingly, systems such as ProjecToR provide a reconfigurable interconnect, allowing to establish links flexibly and in a *demand-aware manner*. For example, direct links or at least short communication paths can be established between frequently communicating ToR switches. Such links can be implemented using a bounded number of lasers, mirrors,

## Overview: Models

### Toward Demand-Aware Networking: A Theory for Self-Adjusting Networks

Chen Avin
Ben Gurion University, Israel
avin@cse.bgu.ac.il

Stefan Schmid
University of Vienna, Austria
stefan_schmid@univie.ac.at

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
The authors take full responsibility for this article's technical content. Comments can be posted through CCR Online.

**ABSTRACT**

The physical topology is emerging as the next frontier in an ongoing effort to render communication networks more flexible. While first empirical results indicate that these flexibilities can be exploited to reconfigure and optimize the network toward the workload it serves and, e.g., providing the same bandwidth at lower infrastructure cost, only little is known today about the fundamental algorithmic problems underlying the design of reconfigurable networks. This paper initiates the study of the theory of demand-aware, self-adjusting networks. Our main position is that self-adjusting networks should be seen through the lense of self-adjusting datastructures. Accordingly, we present a taxonomy classifying the different algorithmic models of demand-oblivious, fixed demand-aware, and reconfigurable demand-aware networks, introduce a formal model, and identify objectives and evaluation metrics. We also demonstrate, by examples, the inherent

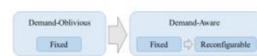



**Figure 1: Taxonomy of topology optimization**

design of efficient datacenter networks has received much attention over the last years. The topologies underlying modern datacenter networks range from trees [7, 8] over hypercubes [9, 10] to expander networks [11] and provide high connectivity at low cost [1].

Until now, these networks also have in common that their topology is *fixed* and ***oblivious*** to the actual demand (i.e.,

## Static Optimality

### ReNets: Toward Statically Optimal Self-Adjusting Networks

Chen Avin[1] Stefan Schmid[2]
[1] Ben Gurion University, Israel [2] University of Vienna, Austria

**Abstract**

This paper studies the design of *self-adjusting* networks whose topology dynamically adapts to the workload, in an *online* and *demand-aware* manner. This problem is motivated by emerging optical technologies which allow to reconfigure the datacenter topology at runtime. Our main contribution is *ReNet*, a self-adjusting network which maintains a balance between the benefits and costs of reconfigurations. In particular, we show that *ReNets* are *statically optimal* for arbitrary sparse communication demands, i.e., perform at least as good as any fixed demand-aware network designed with a perfect knowledge of the *future* demand. Furthermore, *ReNets* provide *compact* and *local* routing, by leveraging ideas from self-adjusting datastructures.

**1 Introduction**

Modern datacenter networks rely on efficient network topologies (based on fat-trees [1], hypercubes [2, 3], or expander [4] graphs) to provide a high connectivity at low cost [5]. These datacenter networks have in common that their topology is *fixed* and *oblivious* to the actual demand (i.e., workload or communication pattern) they currently serve. Rather, they are designed for all-to-all communication patterns, by ensuring properties such as full bisection bandwidth or $O(\log n)$ route lengths between *any* node pair in a constant-degree $n$-node network. However, demand-oblivious networks can be inefficient for more *specific* demand patterns, as they usually arise in

## Robust DAN

### rDAN: Toward Robust Demand-Aware Network Designs

Chen Avin[1] Alexandr Hercules[1] Andreas Loukas[2] Stefan Schmid[3]
[1] Ben-Gurion University, IL [2] EPFL, CH [3] University of Vienna, AT & TU Berlin, DE

**Abstract**

We currently witness the emergence of interesting new network topologies optimized towards the traffic matrices they serve, such as demand-aware datacenter interconnects (e.g., ProjecToR) and demand-aware peer-to-peer overlay networks (e.g., SplayNets). This paper introduces a formal framework and approach to reason about and design robust demand-aware networks ($DAN$). In particular, we establish a connection between the communication frequency of two nodes and the path length between them in the network, and show that this relationship depends on the *entropy* of the communication matrix. Our main contribution is a novel robust, yet sparse, family of networks, short *rDAN*s, which guarantee an expected path length that is proportional to the entropy of the communication patterns.

## Dynamic DAN

### SplayNet: Towards Locally Self-Adjusting Networks

Stefan Schmid*, Chen Avin*, Christian Scheideler, Michael Borokhovich, Bernhard Haeupler, Zvi Lotker

***Abstract*—This paper initiates the study of locally self-adjusting networks: networks whose topology adapts dynamically and in a decentralized manner, to the communication pattern $\sigma$. Our vision can be seen as a distributed generalization of the self-adjusting datastructures introduced by Sleator and Tarjan [22]: In contrast to their splay trees which dynamically optimize the lookup costs from a *single node* (namely the tree root), we seek to minimize the routing cost between arbitrary *communication pairs* in the network.**

**As a first step, we study distributed binary search trees (BSTs), which are attractive for their support of greedy routing. We introduce a simple model which captures the fundamental tradeoff between the benefits and costs of self-adjusting networks. We present the *SplayNet* algorithm and formally analyze its performance, and prove its optimality in specific case studies. We also introduce lower bound techniques based on interval cuts and edge expansion, to study the limitations of any demand-optimized network. Finally, we extend our study to multi-tree networks, and highlight an intriguing difference between classic and distributed splay trees.**

I. Introduction

In the 1980s, Sleator and Tarjan [22] proposed an appealing new paradigm to design efficient Binary Search Tree (BST) datastructures: rather than optimizing traditional metrics such toward static metrics, such as the diameter or the length of the longest route: the self-adjusting paradigm has not spilled over to distributed networks yet.

We, in this paper, initiate the study of a distributed generalization of self-optimizing datastructures. This is a non-trivial generalization of the classic splay tree concept: While in classic BSTs, a *lookup request* always originates from the same node, the tree root, distributed datastructures and networks such as skip graphs [2], [13] have to support *routing requests* between arbitrary pairs (or *peers*) of communicating nodes; in other words, both the source as well as the destination of the requests become variable. Figure 1 illustrates the difference between classic and distributed binary search trees.

In this paper, we ask: Can we reap similar benefits from self-adjusting *entire networks*, by adaptively reducing the distance between frequently communicating nodes?

As a first step, we explore fully decentralized and self-adjusting Binary Search Tree networks: in these networks, nodes are arranged in a binary tree which respects node identifiers. A BST topology is attractive as it supports greedy routing: a node can decide locally to which port to forward a request given its destination address.

## Concurrent DANs

### CBNet: Minimizing Adjustments in Concurrent Demand-Aware Tree Networks

Otavio Augusto de Oliveira Souza[1] Olga Goussevskaia[1] Stefan Schmid[2]
[1] Universidade Federal de Minas Gerais, Brazil [2] University of Vienna, Austria

***Abstract*—This paper studies the design of demand-aware network topologies: networks that dynamically adapt themselves toward the demand they currently serve, in an online manner. While demand-aware networks may be significantly more efficient than demand-oblivious networks, frequent adjustments are still costly. Furthermore, a centralized controller of such networks may become a bottleneck.**

CBNet is based on concepts from self-adjusting data structures, and in particular, CBTrees [12]. CBNet gradually adapts the network topology toward the communication pattern in an online manner, i.e., without previous knowledge of the demand distribution. At the same time, *bidirectional semi-splaying* and counters are used to maintain state, minimize reconfiguration

# Selected References

**On the Complexity of Traffic Traces and Implications**
Chen Avin, Manya Ghobadi, Chen Griner, and Stefan Schmid.
ACM SIGMETRICS, Boston, Massachusetts, USA, June 2020.

**Survey of Reconfigurable Data Center Networks: Enablers, Algorithms, Complexity**
Klaus-Tycho Foerster and Stefan Schmid.
**SIGACT News**, June 2019.

**Toward Demand-Aware Networking: A Theory for Self-Adjusting Networks (Editorial)**
Chen Avin and Stefan Schmid.
ACM SIGCOMM Computer Communication Review (**CCR**), October 2018.

**Dynamically Optimal Self-Adjusting Single-Source Tree Networks**
Chen Avin, Kaushik Mondal, and Stefan Schmid.
14th Latin American Theoretical Informatics Symposium (LATIN), University of Sao Paulo, Sao Paulo, Brazil, May 2020.

**Demand-Aware Network Design with Minimal Congestion and Route Lengths**
Chen Avin, Kaushik Mondal, and Stefan Schmid.
38th IEEE Conference on Computer Communications (**INFOCOM**), Paris, France, April 2019.

**Distributed Self-Adjusting Tree Networks**
Bruna Peres, Otavio Augusto de Oliveira Souza, Olga Goussevskaia, Chen Avin, and Stefan Schmid.
38th IEEE Conference on Computer Communications (**INFOCOM**), Paris, France, April 2019.

**Efficient Non-Segregated Routing for Reconfigurable Demand-Aware Networks**
Thomas Fenz, Klaus-Tycho Foerster, Stefan Schmid, and Anaïs Villedieu.
**IFIP Networking**, Warsaw, Poland, May 2019.

**DaRTree: Deadline-Aware Multicast Transfers in Reconfigurable Wide-Area Networks**
Long Luo, Klaus-Tycho Foerster, Stefan Schmid, and Hongfang Yu.
IEEE/ACM International Symposium on Quality of Service (**IWQoS**), Phoenix, Arizona, USA, June 2019.

**Demand-Aware Network Designs of Bounded Degree**
Chen Avin, Kaushik Mondal, and Stefan Schmid.
31st International Symposium on Distributed Computing (**DISC**), Vienna, Austria, October 2017.

**SplayNet: Towards Locally Self-Adjusting Networks**
Stefan Schmid, Chen Avin, Christian Scheideler, Michael Borokhovich, Bernhard Haeupler, and Zvi Lotker.
IEEE/ACM Transactions on Networking (**TON**), Volume 24, Issue 3, 2016. Early version: IEEE **IPDPS** 2013.

**Characterizing the Algorithmic Complexity of Reconfigurable Data Center Architectures**
Klaus-Tycho Foerster, Monia Ghobadi, and Stefan Schmid.
ACM/IEEE Symposium on Architectures for Networking and Communications Systems (**ANCS**), Ithaca, New York, USA, July 2018.