Revolutionizing Datacenter Networks via Reconfigurable Topologies

Stefan Schmid (TU Berlin)

“We cannot direct the wind,
but we can adjust the sails.”

(Folklore)

Acknowledgements:

We live in the age of

Distributed Computation

Datacenters (“hyper-scale”)

+network

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

Source: Facebook

We live in the age of

Distributed Computation

Datacenters (“hyper-scale”)

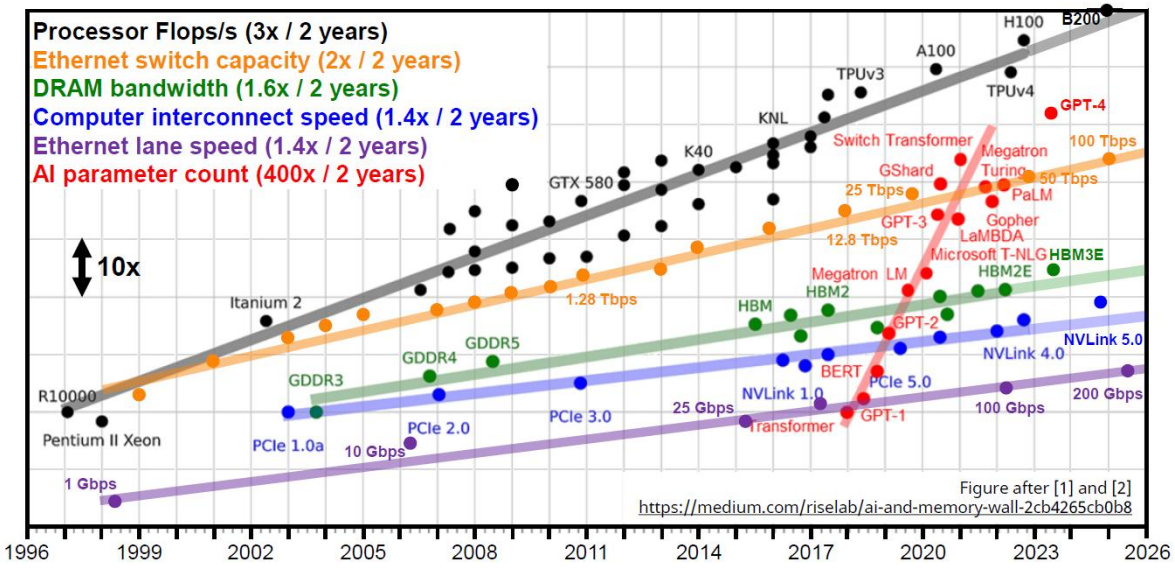
+network

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

Credits: Marco Chiesa

Technological Trends

Increasing Gap Between Compute and Network



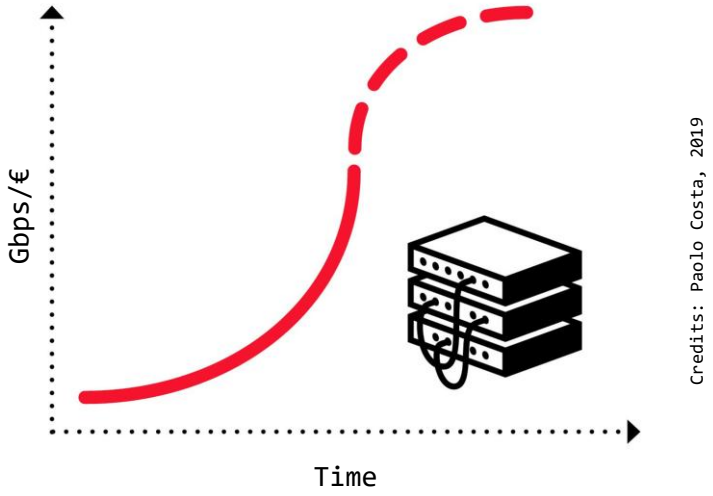
Credits: Nicola Calabretta

The Problem

Huge Infrastructure, Inefficient Use

⇢ Hence: more equipment, larger networks

⇢ Resource intensive and: **inefficient**

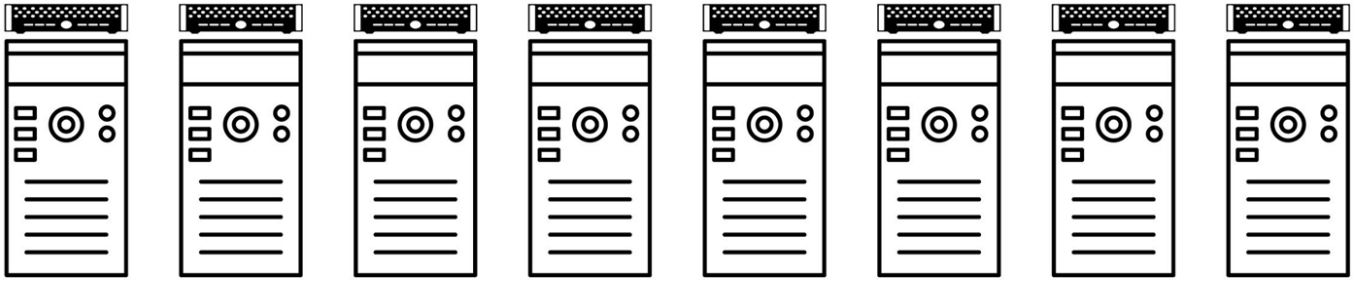


Annoying for companies, **opportunity** for researchers!

An Inefficiency

Fixed and Demand-Oblivious Topology

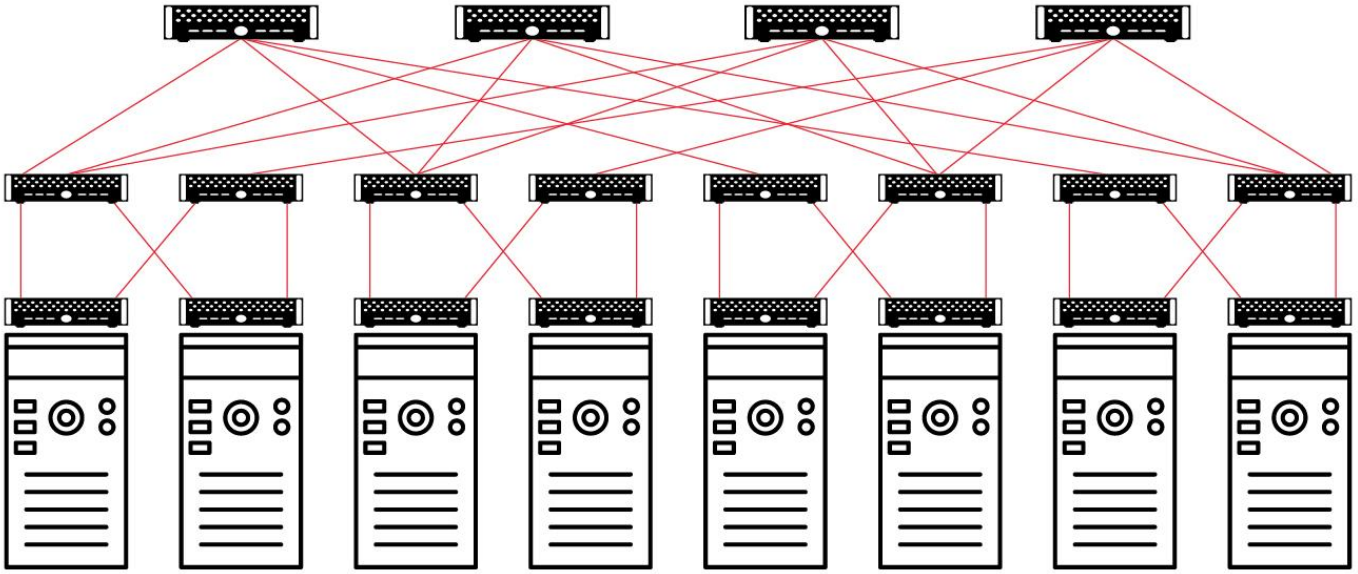
How to interconnect? Focus on this talk: scale-out network.



An Inefficiency

Fixed and Demand-Oblivious Topology

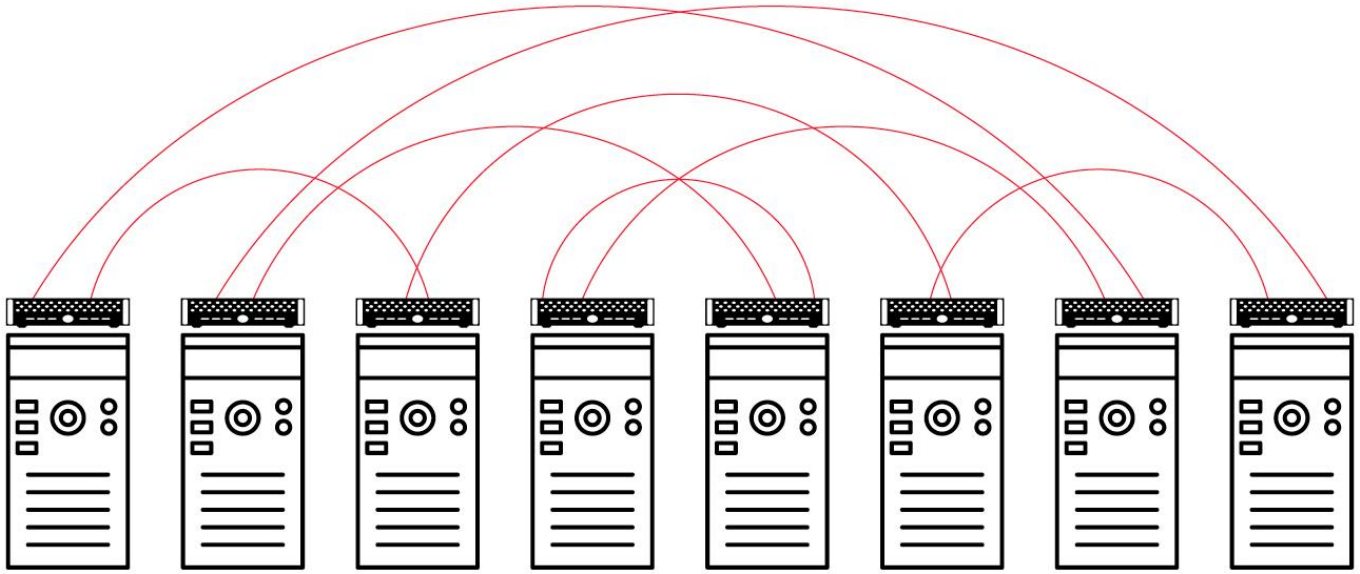
- Example: fat-tree topology (bi-regular)
  - 2 types of switches: top-of-rack (ToR) connect to hosts, additional switches connecting switches to increase throughput



An Inefficiency

Fixed and Demand-Oblivious Topology

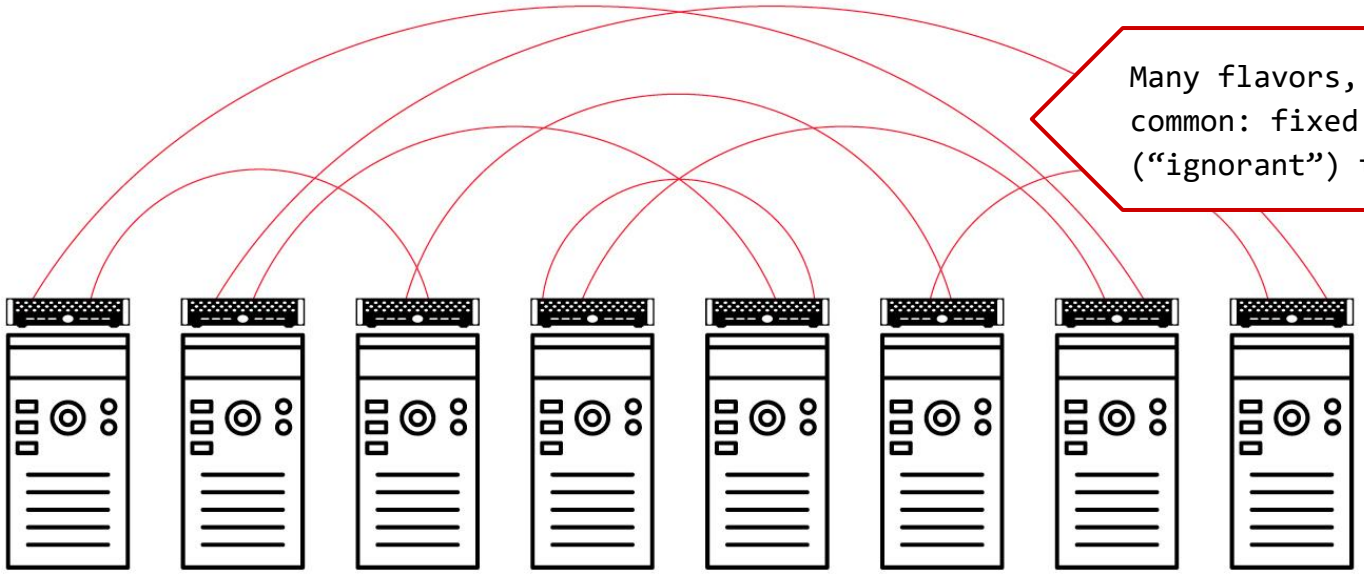
- Example: expander topology (uni-regular)
  - Only 1 type of switches:
    lower installation and management overheads



An Inefficiency

Fixed and Demand-Oblivious Topology

- Example: expander topology (uni-regular)
  - Only 1 type of switches:
    lower installation and management overheads



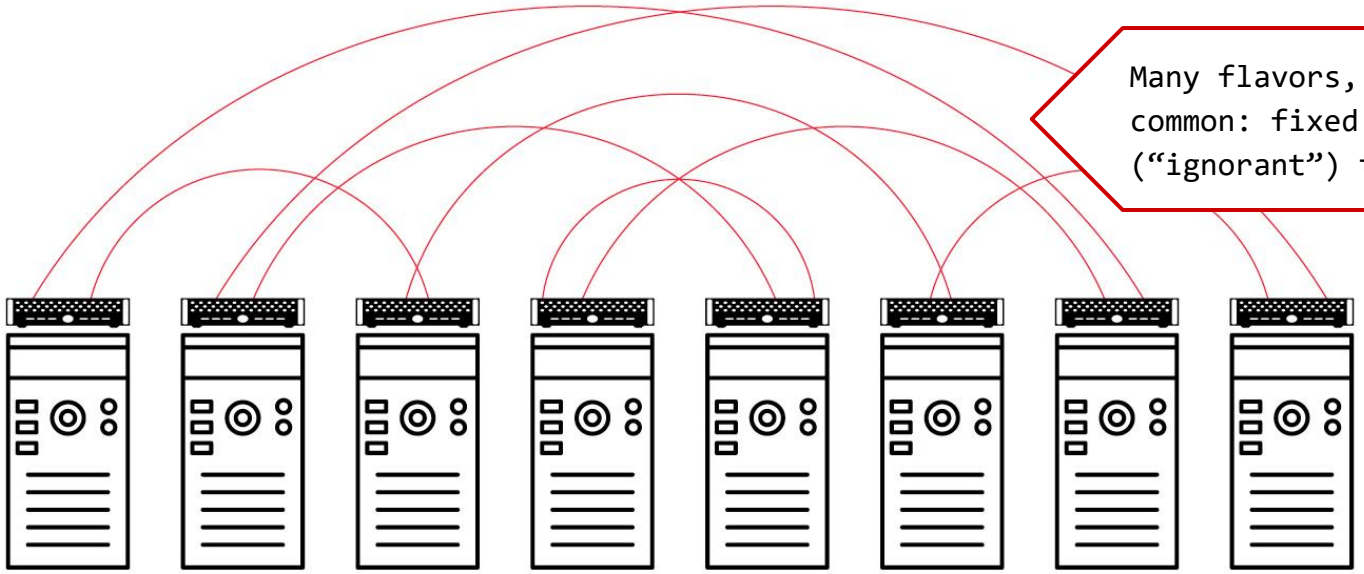
Many flavors, but in common: fixed and **oblivious** (“ignorant”) to actual demand.

An Inefficiency

Fixed and Demand-Oblivious Topology

Highway which ignores actual traffic: frustrating!

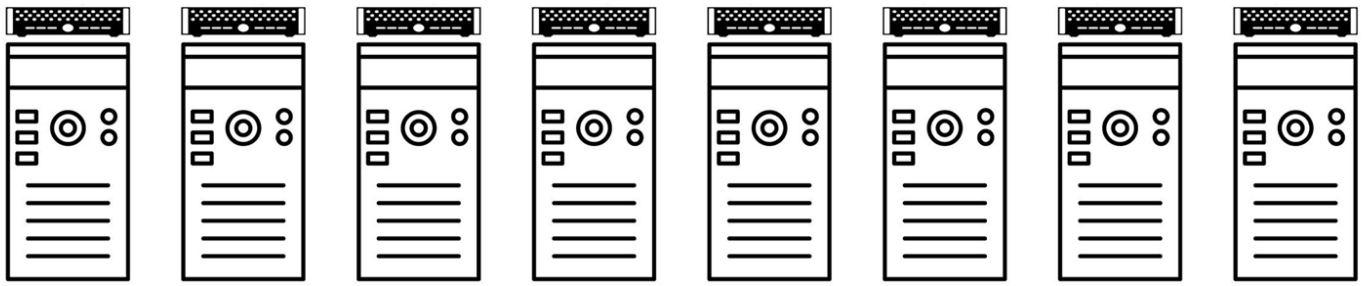
- ⇢ Example: expander topology (uni-regular)
  - → Only 1 type of switches:
    lower installation and management overheads



Many flavors, but in common: fixed and **oblivious** (“ignorant”) to actual demand.

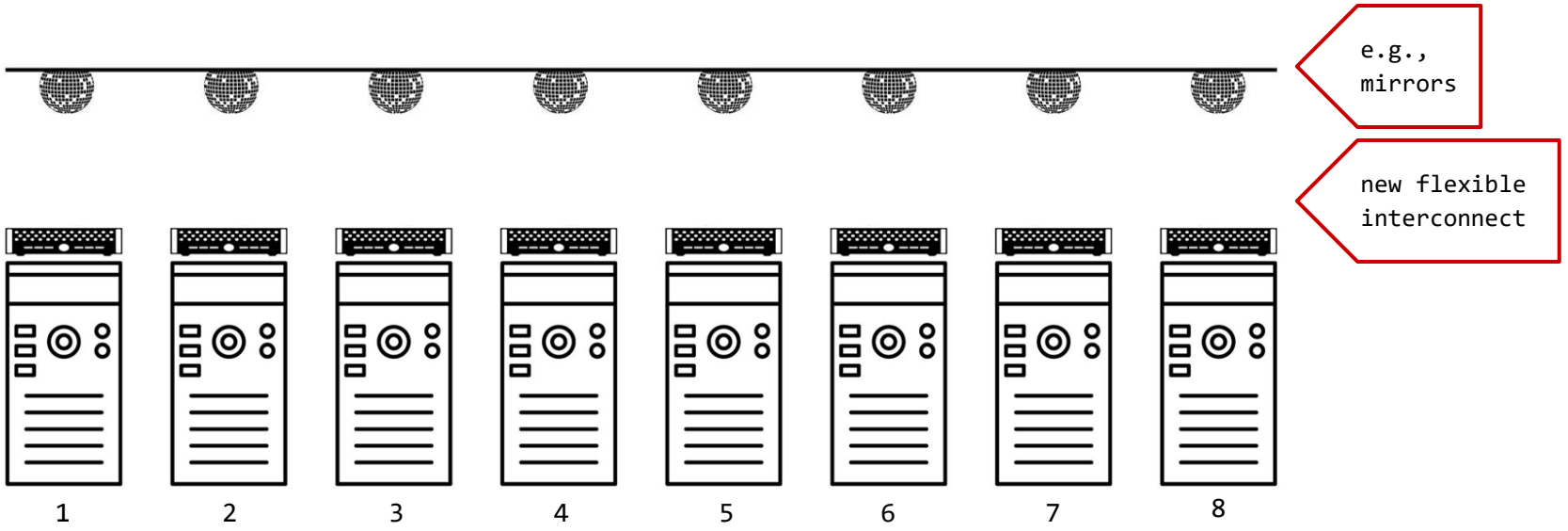
A Vision

Flexible and Demand-Aware Topologies



A Vision

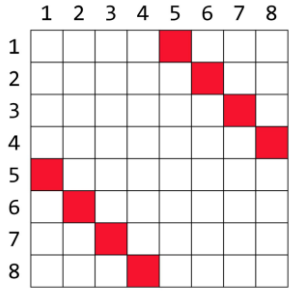
Flexible and Demand-Aware Topologies

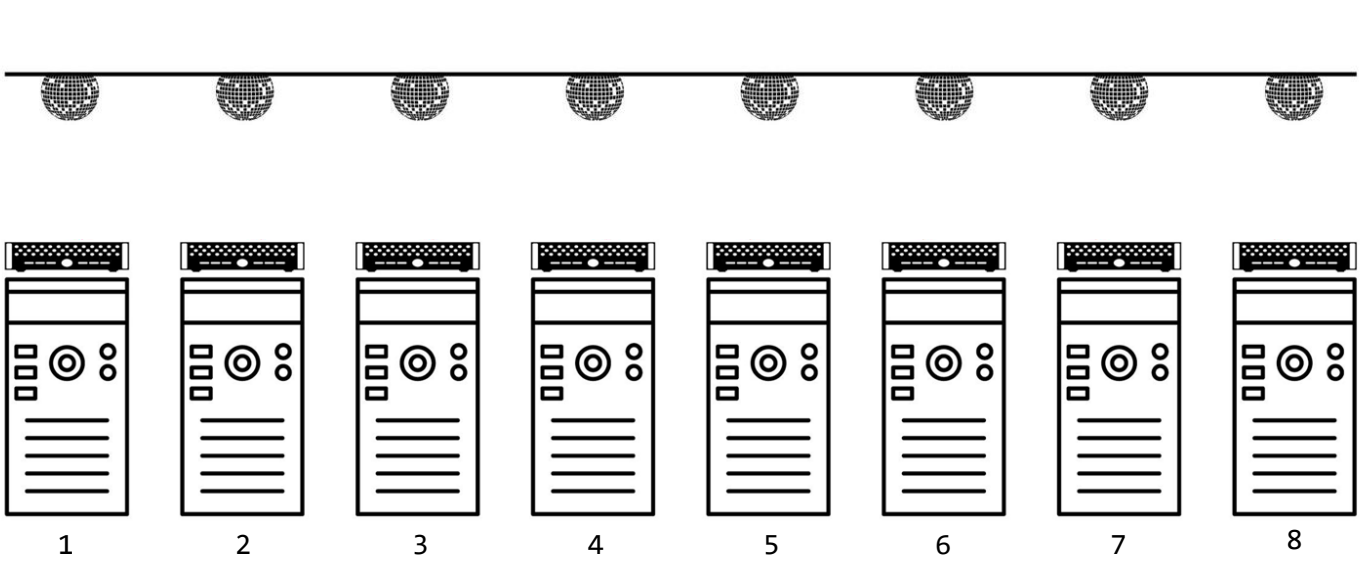


A Vision

Flexible and Demand-Aware Topologies

demand matrix:



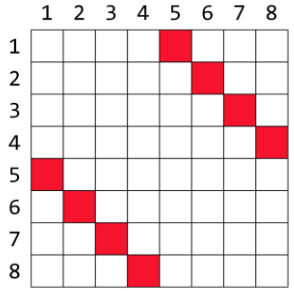


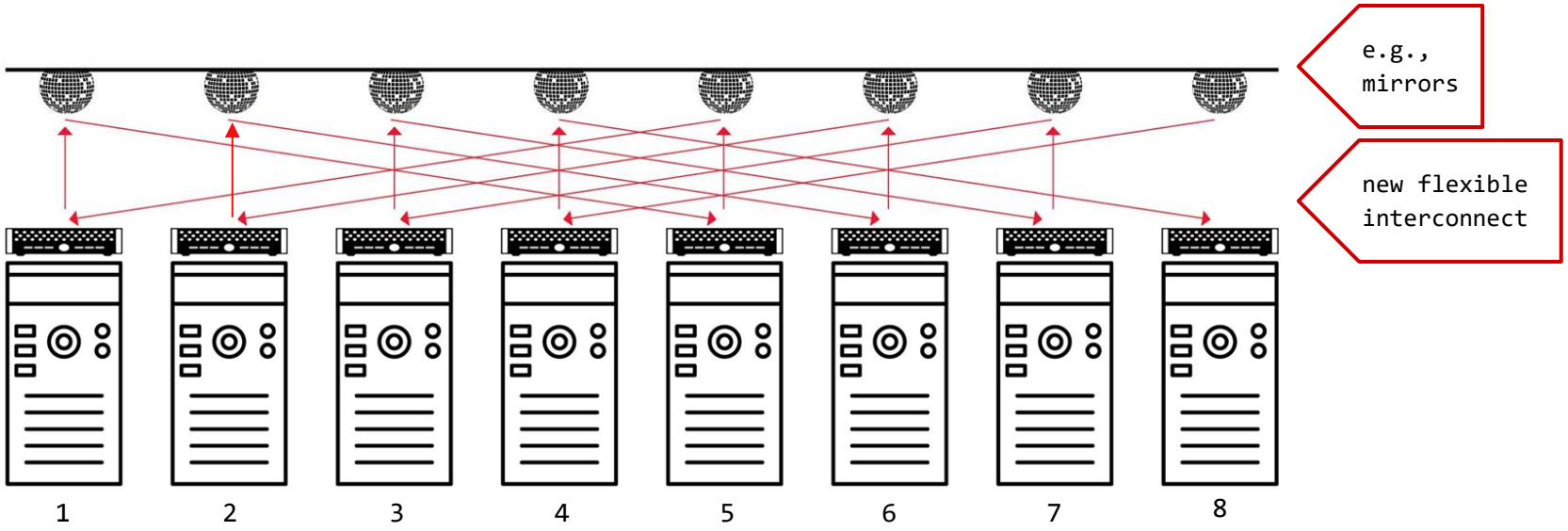
A Vision

Flexible and Demand-Aware Topologies

Matches demand

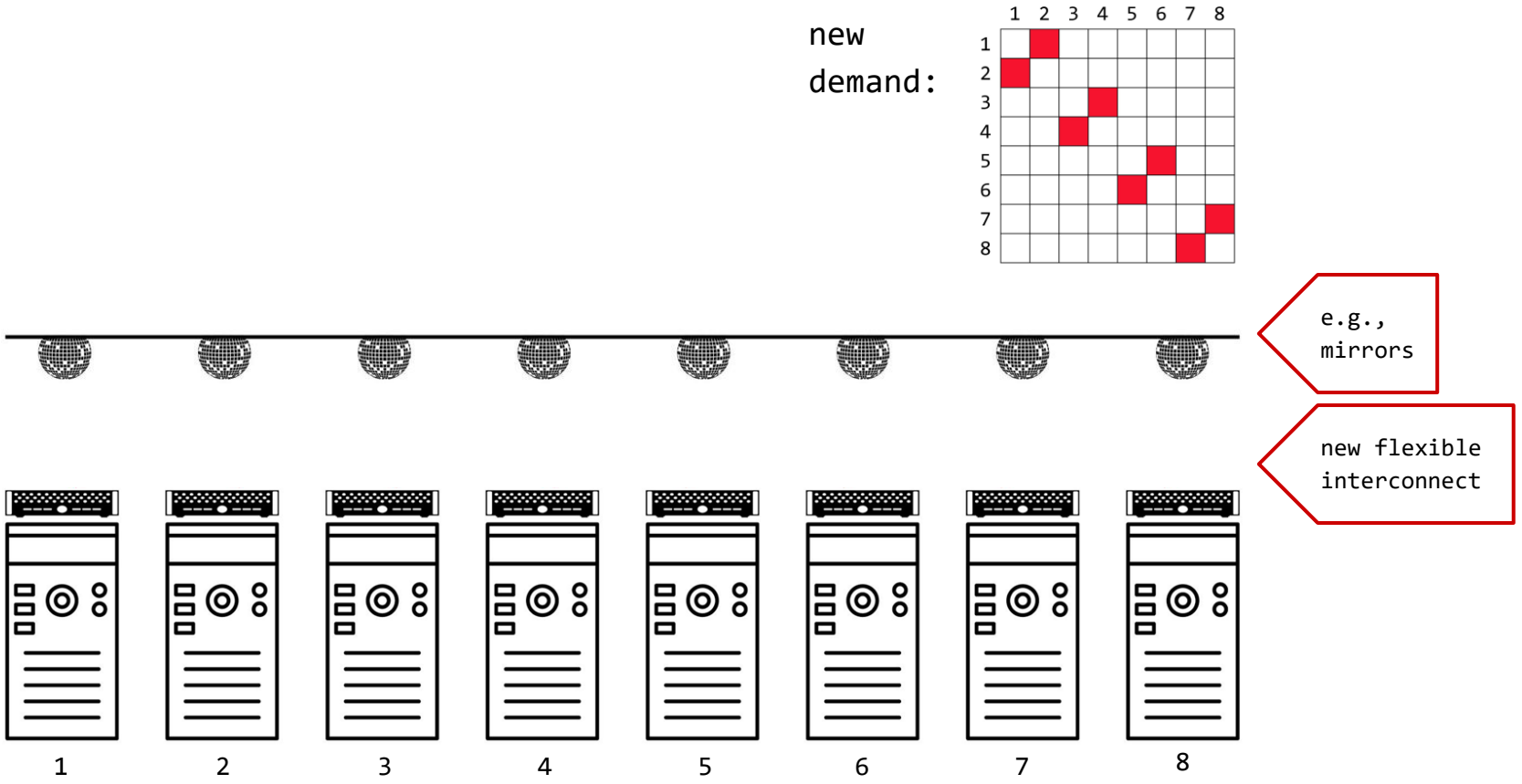
demand matrix:





A Vision

Flexible and Demand-Aware Topologies

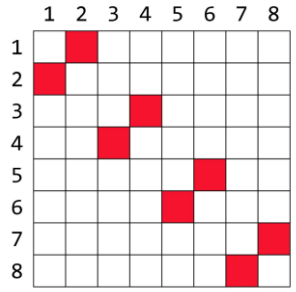


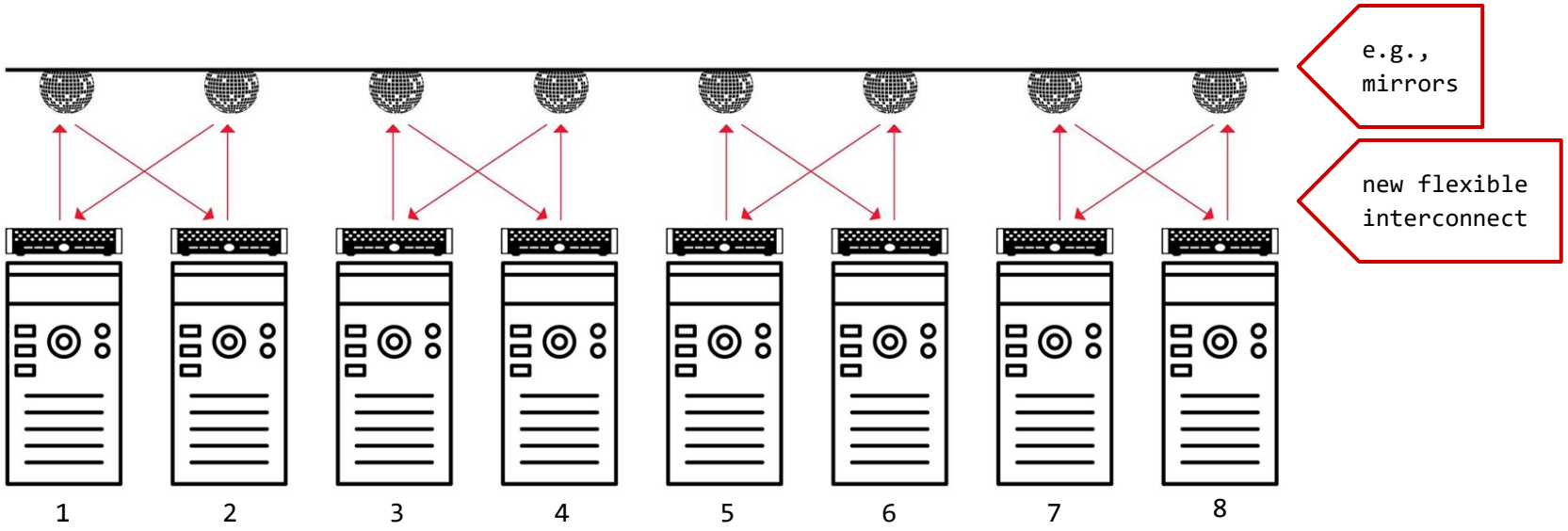
A Vision

Flexible and Demand-Aware Topologies

Matches demand

new demand:



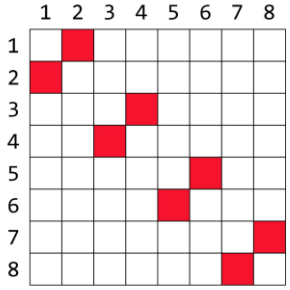


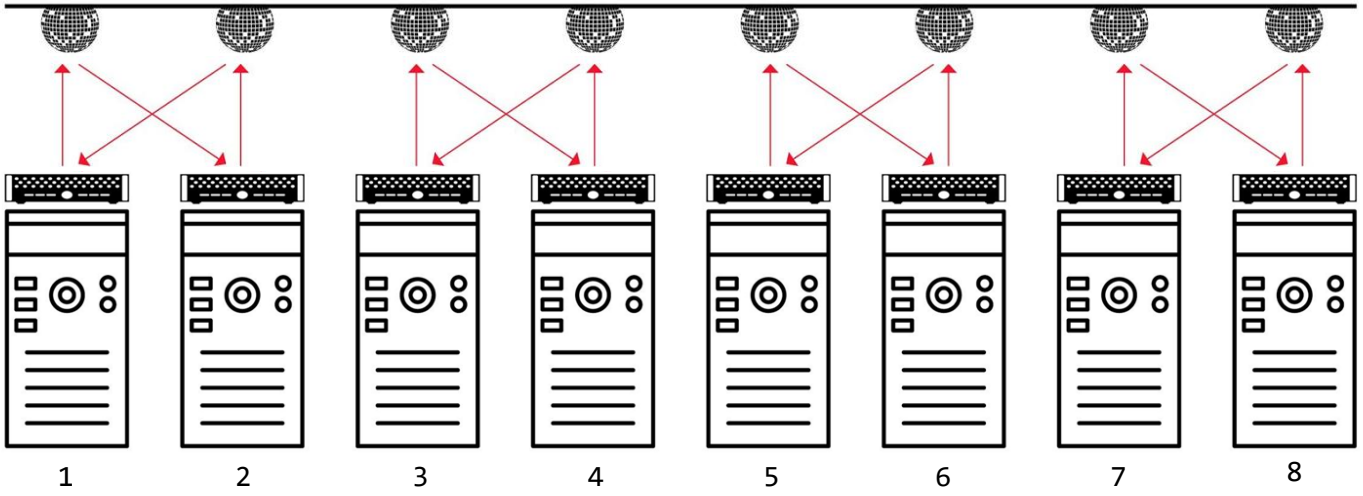
A Vision

Flexible and Demand-Aware Topologies

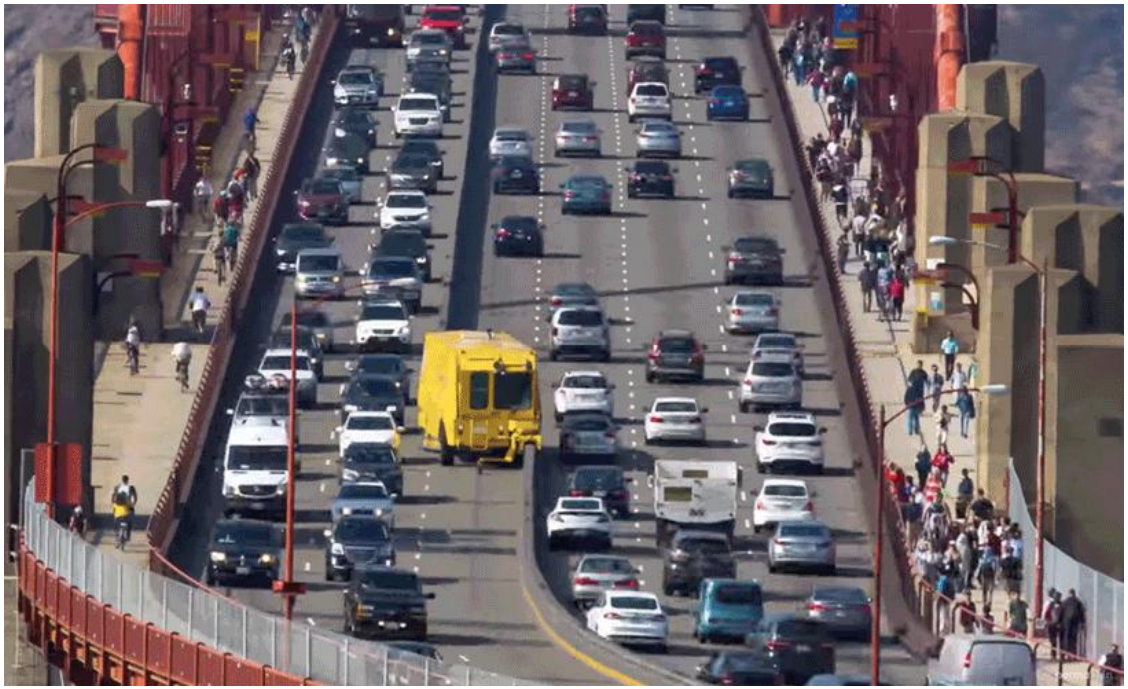
Self-Adjusting Networks

new demand:





Analogy



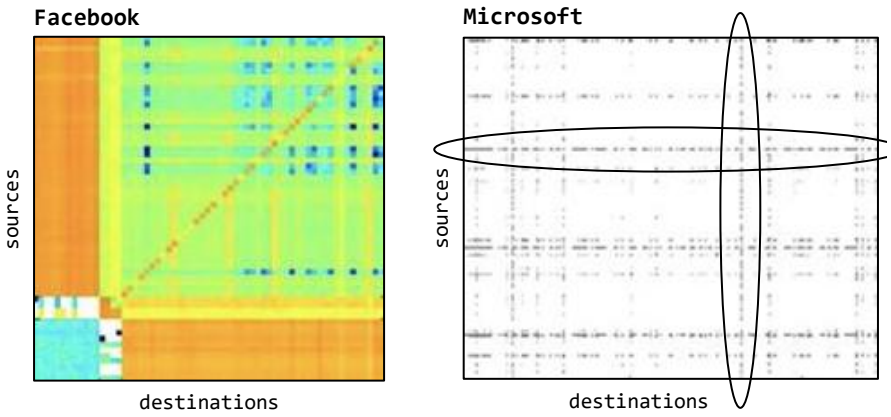
Golden Gate Zipper

The Motivation

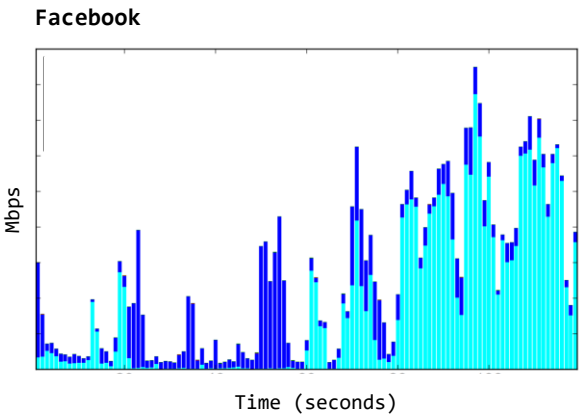
Much Structure in the Demand

Empirical studies:

traffic matrices sparse and skewed



traffic bursty over time



The hypothesis: can be exploited.

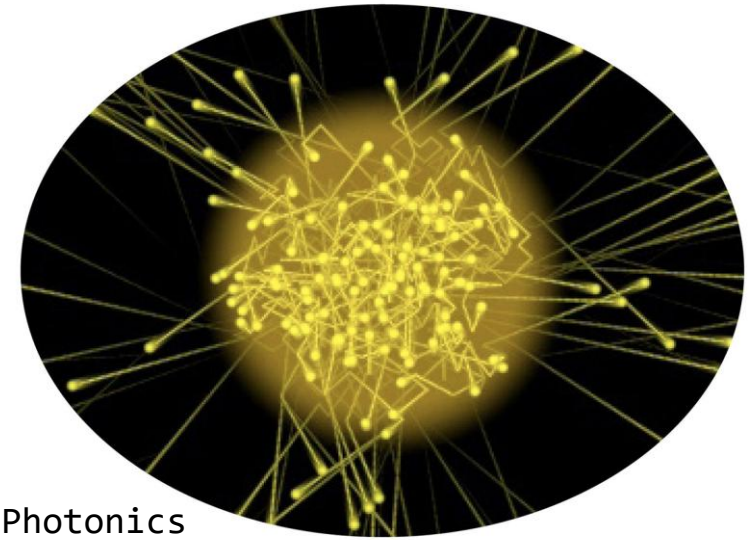
Traffic is also clustered: bi-clustering results

Small Stable Clusters

Opportunity: *exploit* with little reconfigurations!

Förster et al., Analyzing the Communication Clusters in Datacenters. WWW 2023

Sounds Crazy? Emerging Enabling Technology.



Photonics

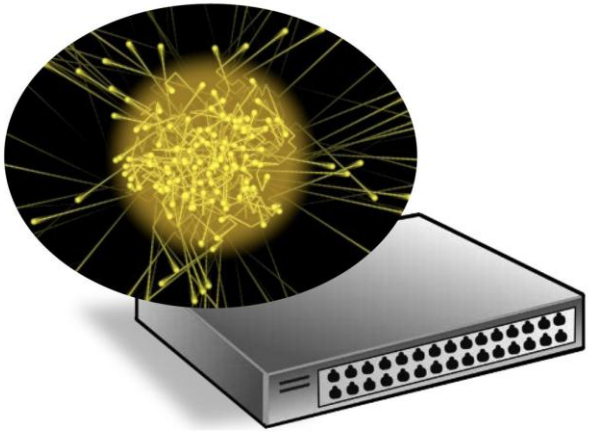
H2020:
“Photonics one of only five key enabling technologies for future prosperity.”

US National Research Council:
“Photons are the new Electrons.”

Enabler

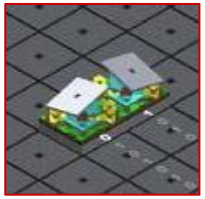
Novel Reconfigurable Optical Switches

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



Prototype 1

Moving antenna (ms)



Prototype 2

Moving mirrors (mus)

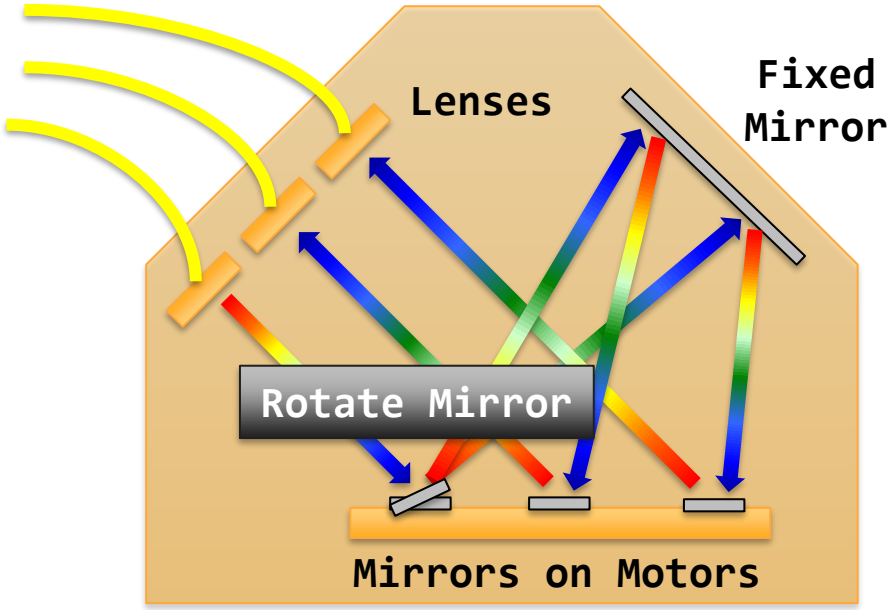
Prototype 3

Changing lambdas (ns)

Example

Optical Circuit Switch

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



Optical Circuit Switch

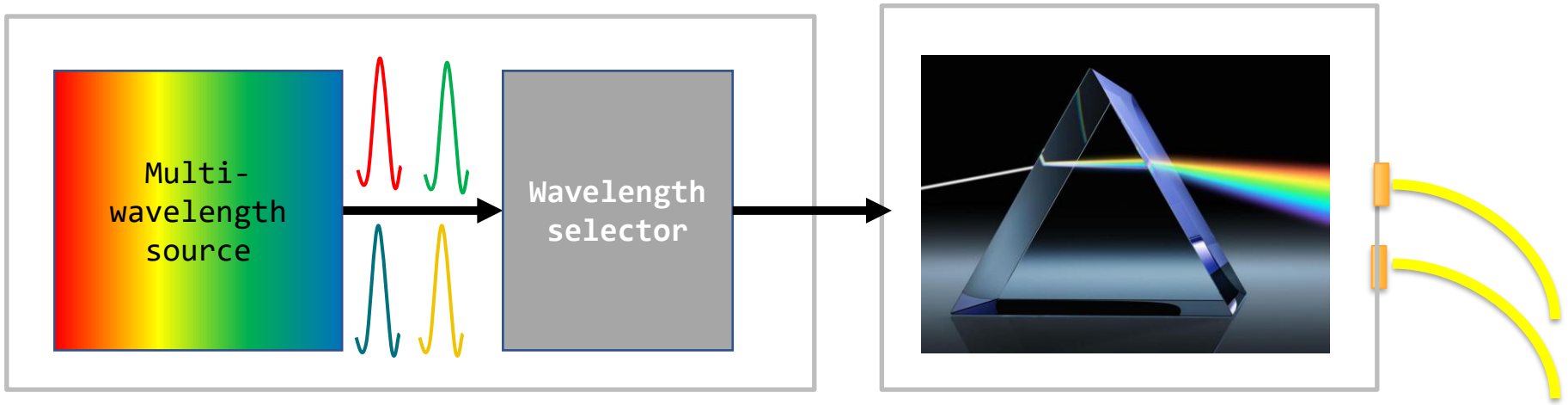
By Nathan Farrington, SIGCOMM 2010

Another Example

Tunable Lasers

⇢ Depending on wavelength, forwarded differently

⇢ Optical switch is passive



Electrical switch with tunable laser

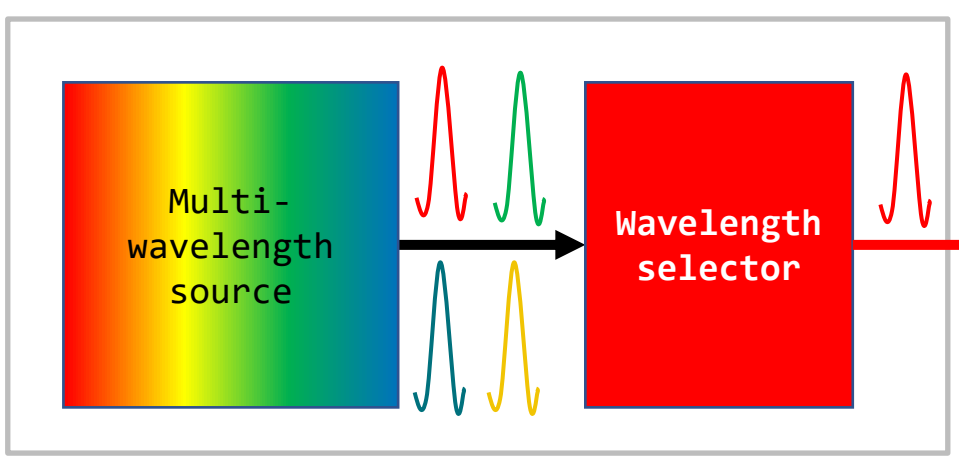
Optical switch
Passive

Another Example

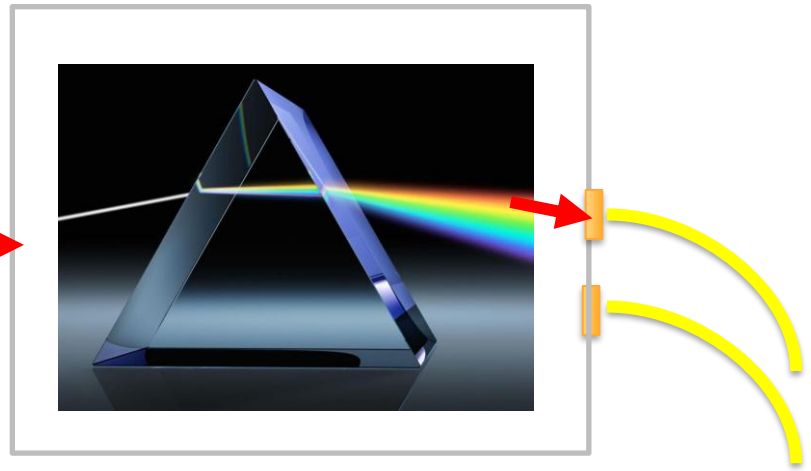
Tunable Lasers

⇢ Depending on wavelength, forwarded differently

⇢ Optical switch is passive



Electrical switch with tunable laser



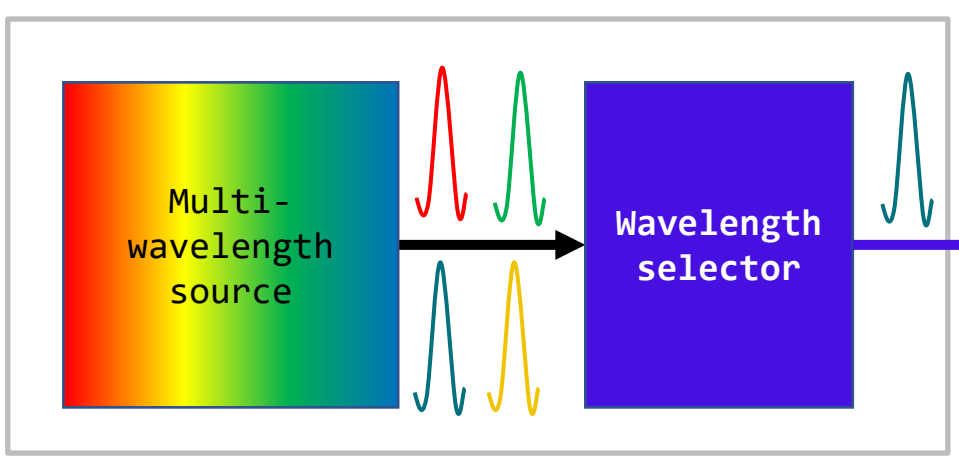
Optical switch
Passive

Another Example

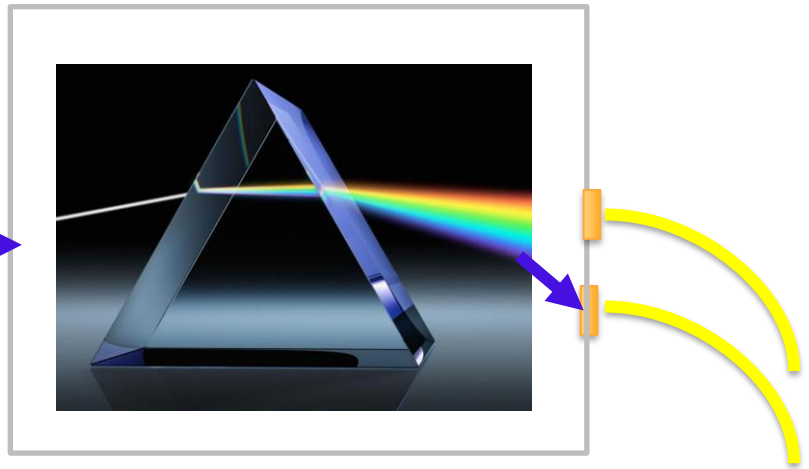
Tunable Lasers

⇢ Depending on wavelength, forwarded differently

⇢ Optical switch is passive



Electrical switch with tunable laser

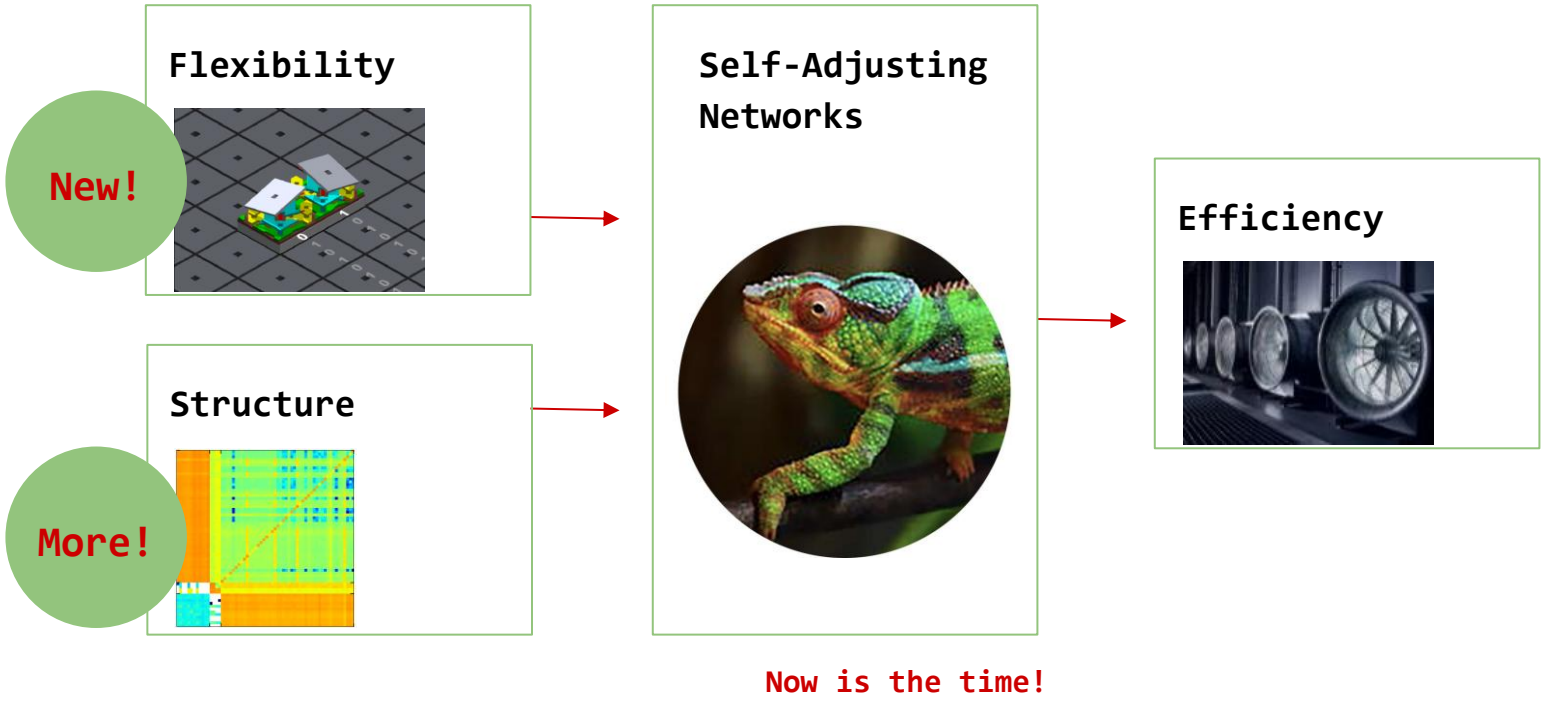


Optical switch
Passive

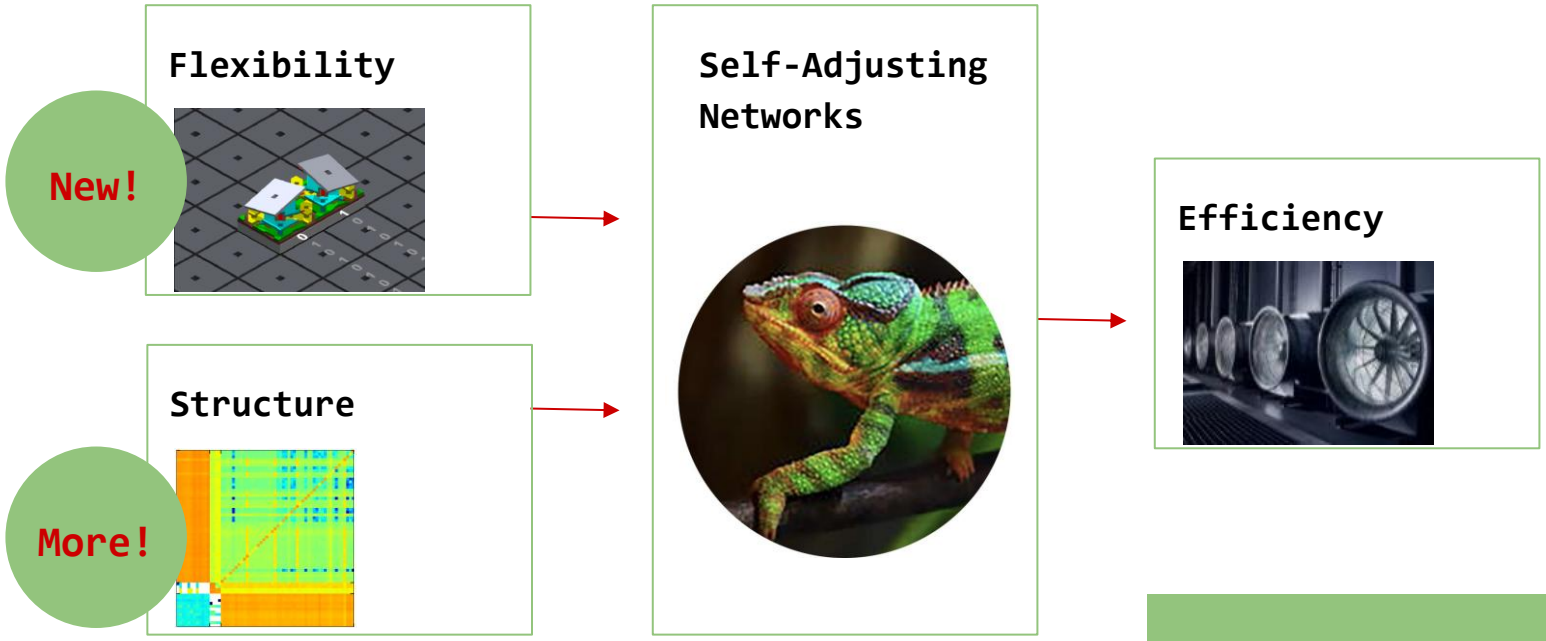
First Deployments

E.g., Google’s Datacenter Jupiter

The Big Picture



The Big Picture



Missing: Foundations of demand-aware, self-adjusting networks.

Unique Position

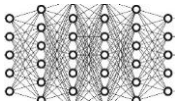
Demand-Aware, Self-Adjusting Systems

Everywhere, but mainly in software

Algorithmic trading

NETFLIX

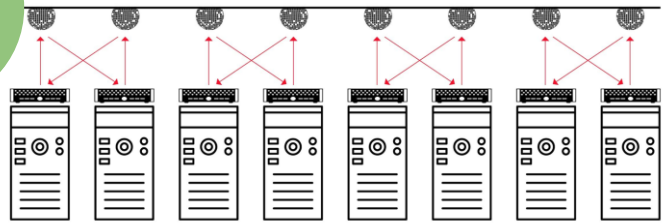
Recommender systems



Neural networks

VS

Our focus in this talk: in hardware



Question:

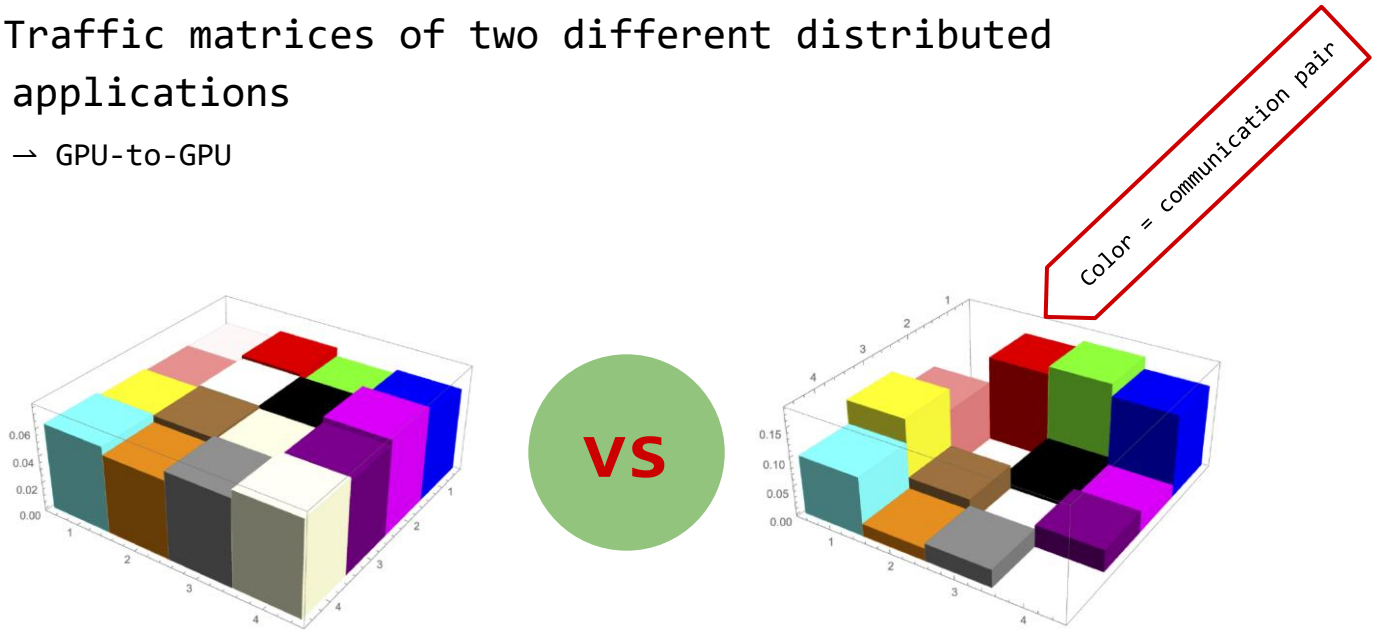
How to Quantify such “Structure” in the Demand?

Intuition

Which demand has more structure?

⇢ Traffic matrices of two different distributed ML applications

→ GPU-to-GPU

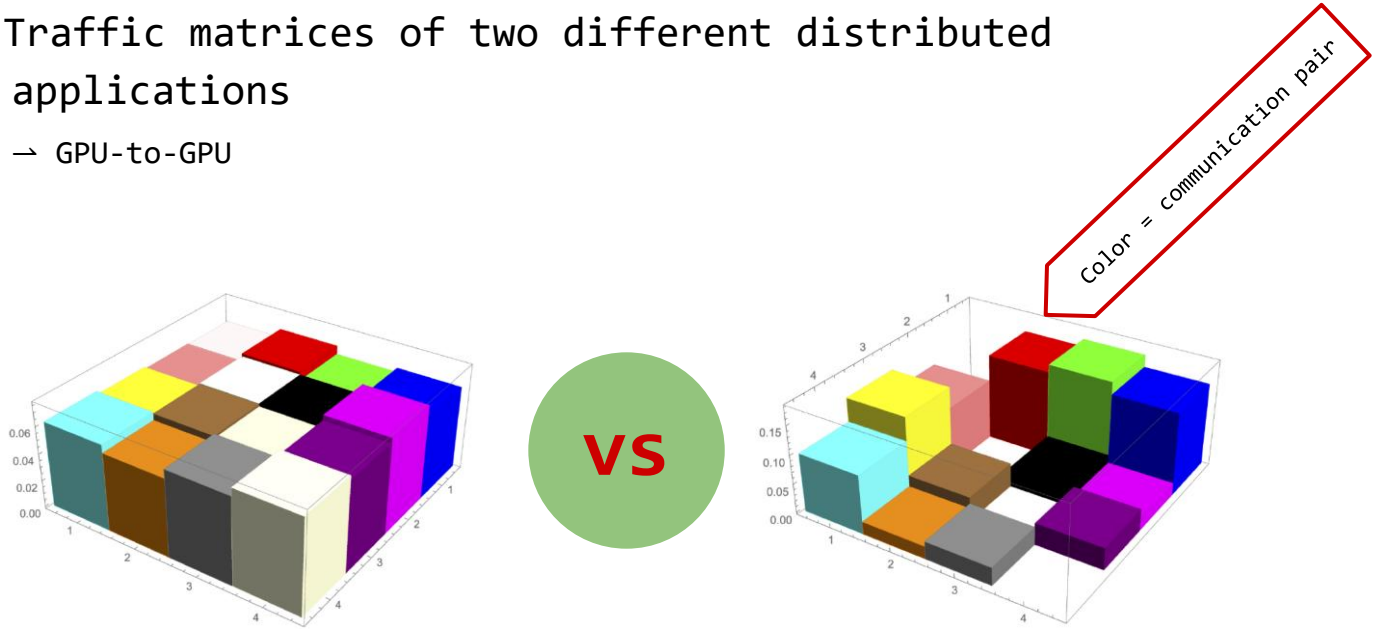


Intuition

Which demand has more structure?

⇢ Traffic matrices of two different distributed ML applications

→ GPU-to-GPU



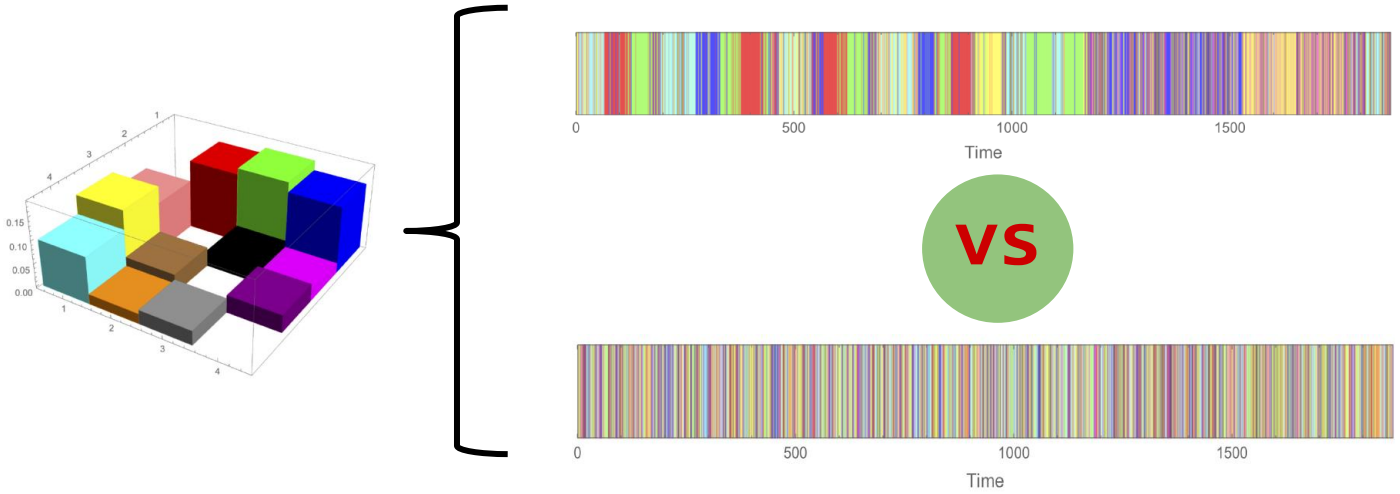
More uniform **More structure**

Intuition

Spatial vs temporal structure

⇢ Two different ways to generate same traffic matrix:
  ⇀ Same non-temporal structure

⇢ Which one has more structure?

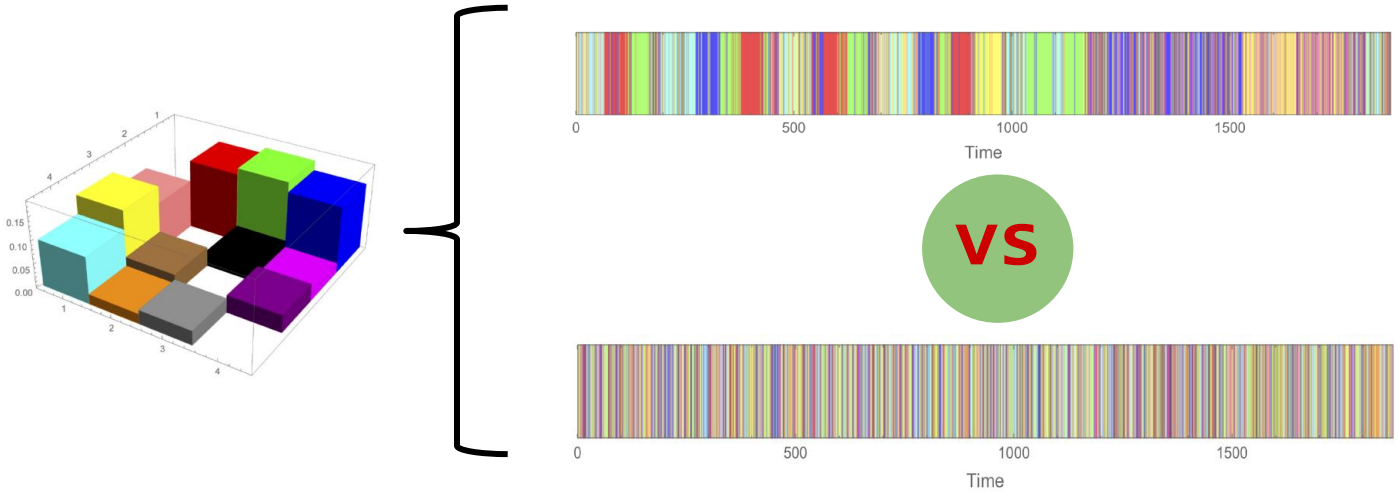


Intuition

Spatial vs temporal structure

⇢ Two different ways to generate same traffic matrix:
  ⇀ Same non-temporal structure

⇢ Which one has more structure?

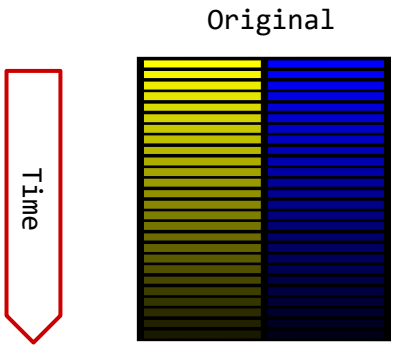


Systematically?

Trace Complexity

Information-Theoretic Approach

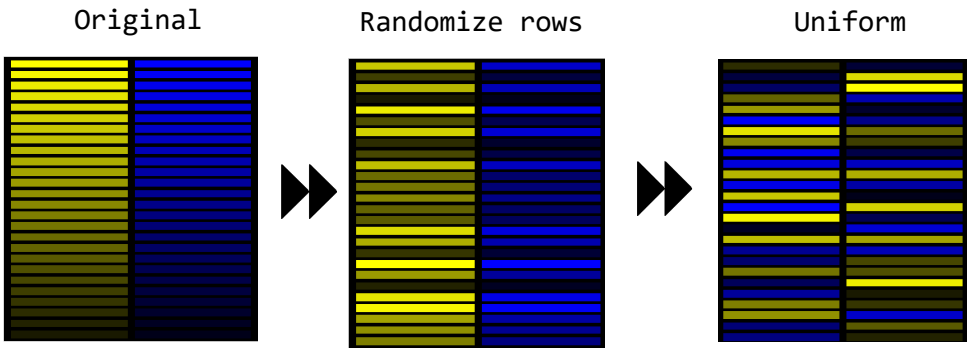
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

“Shuffle&Compress”



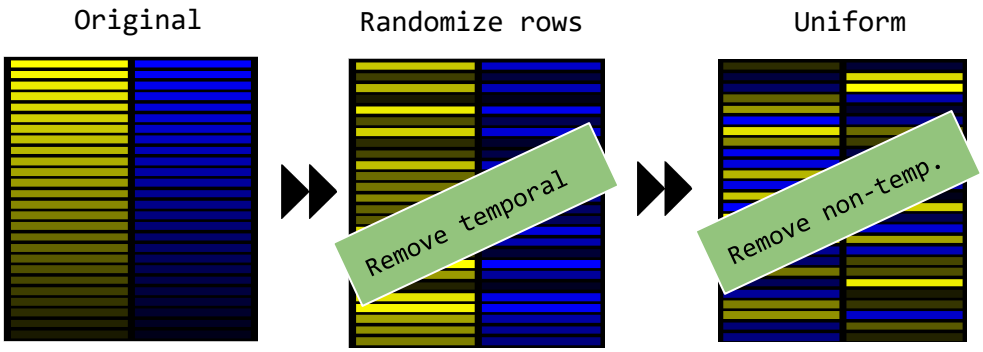
Increasing complexity (systematically randomized)

More structure (compresses better)

Trace Complexity

Information-Theoretic Approach

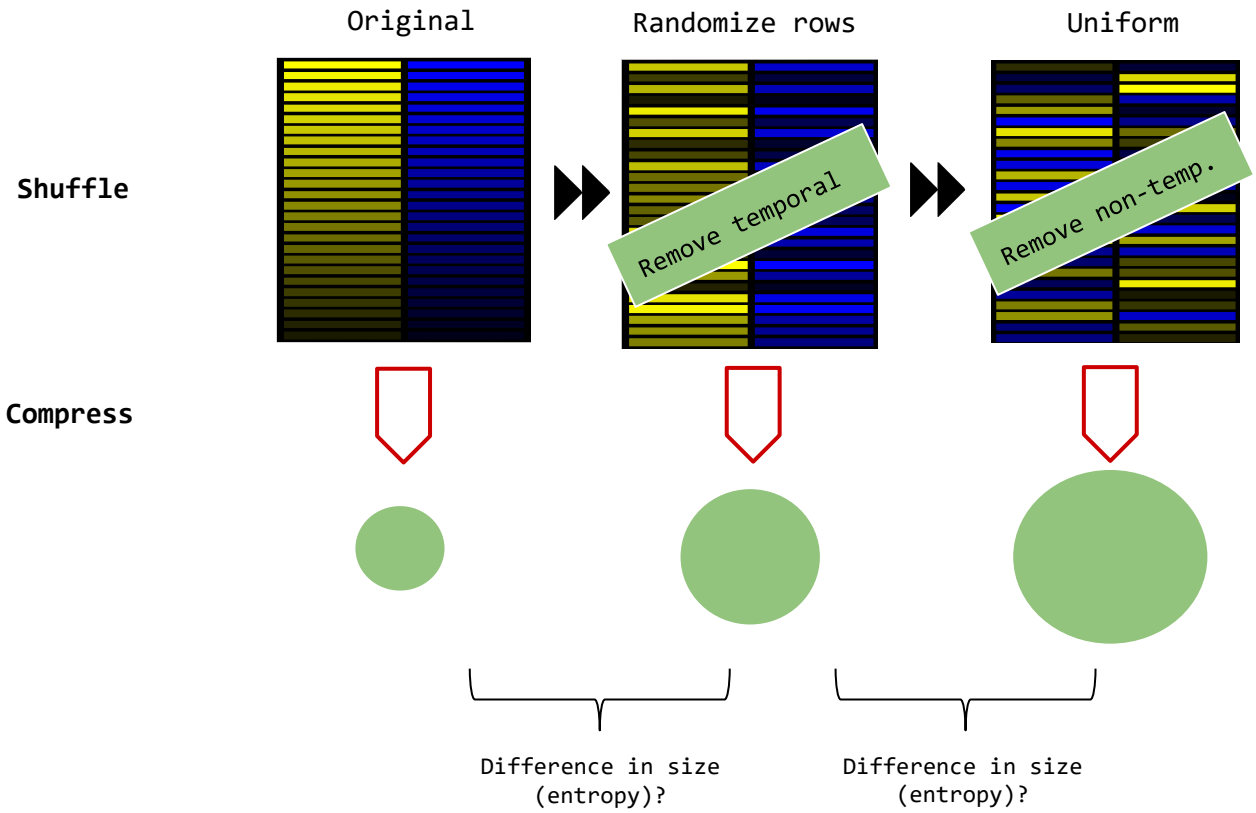
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

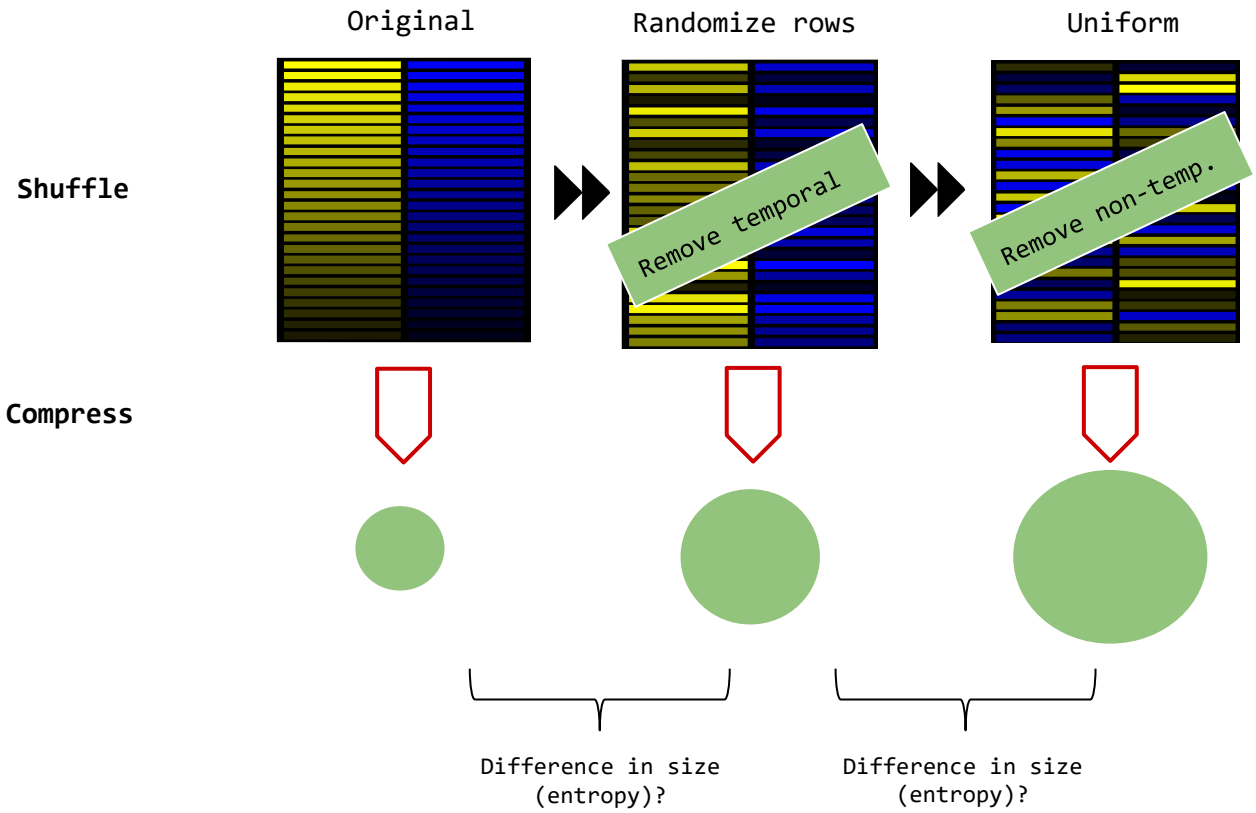
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

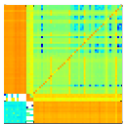
“Shuffle&Compress”

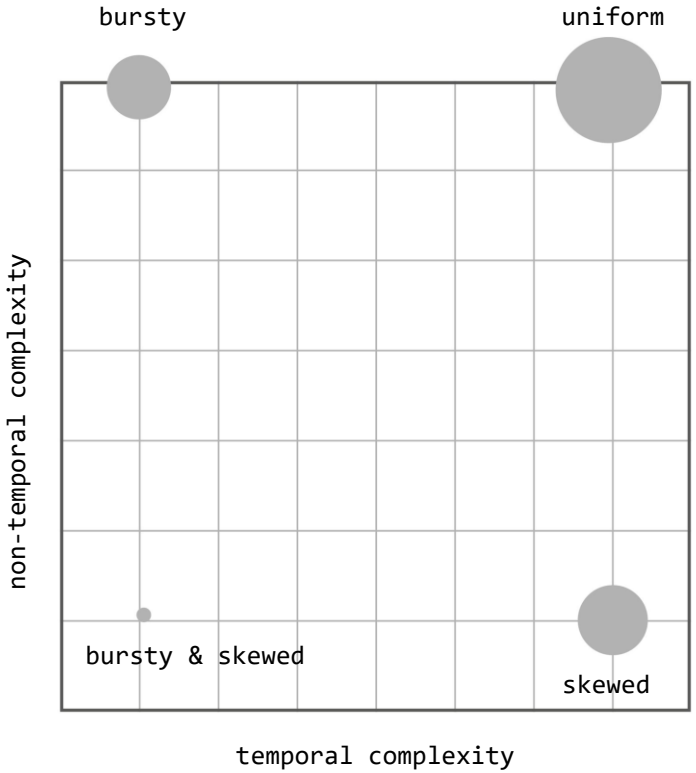


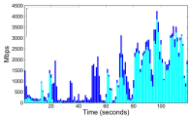
Can be used to define 2-dimensional complexity map!

Our Methodology

Complexity Map



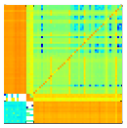


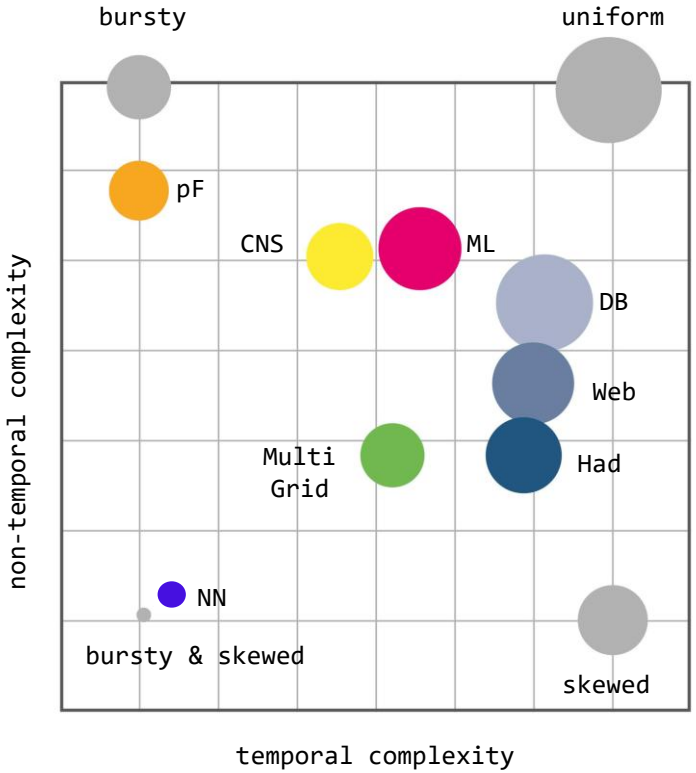


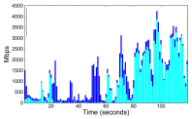
Our **approach**: iterative **randomization and compression** of trace to identify dimensions of structure.

Our Methodology

Complexity Map





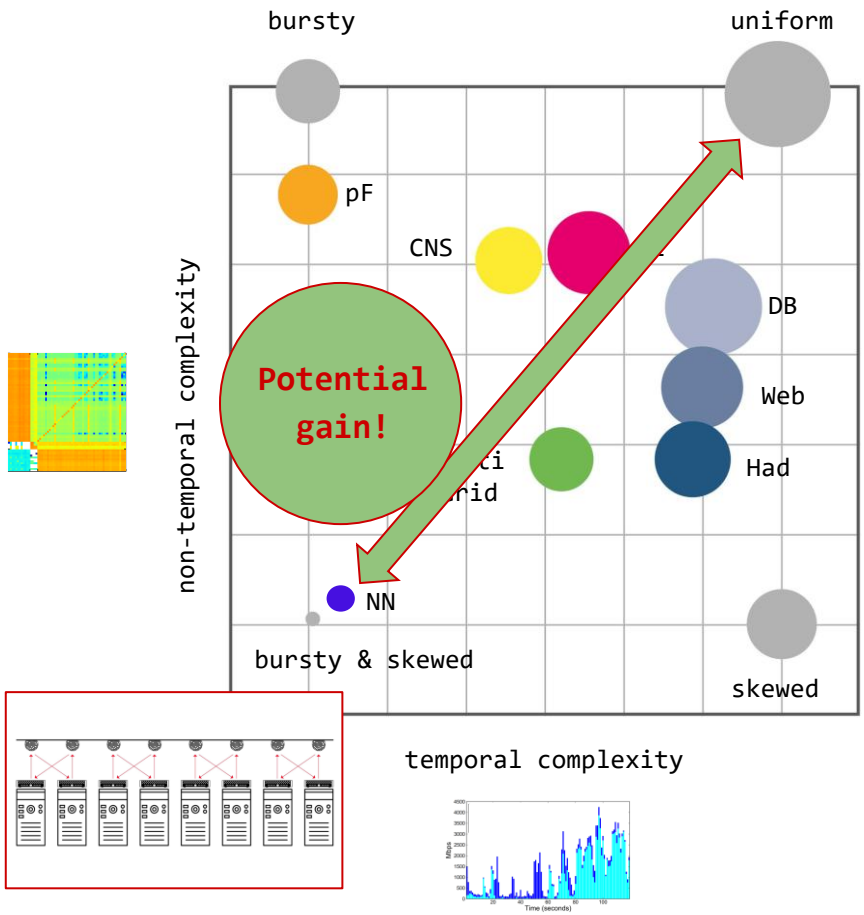


Our **approach**: iterative **randomization and compression** of trace to identify dimensions of structure.

Different structures!

Our Methodology

Complexity Map



Our **approach**: iterative **randomization and compression** of trace to identify dimensions of structure.

Different structures!

Further Reading

ACM SIGMETRICS 2020

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, we also propose a traffic generator model able to produce a synthetic trace that matches the complexity levels of its corresponding real-world trace. Using a case study in the context of datacenters, we show that insights into the structure of packet traces can lead to improved demand-aware network designs: datacenter topologies that are optimized for specific traffic patterns.

CCS Concepts: • **Networks** → **Network performance evaluation**; **Network algorithms**; **Data center networks**; • **Mathematics of computing** → *Information theory*;

Additional Key Words and Phrases: trace complexity, self-adjusting networks, entropy rate, compress, complexity map, data centers

ACM Reference Format:
Chen Avin, Manya Ghobadi, Chen Griner, and Stefan Schmid. 2020. On the Complexity of Traffic Traces and Implications. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 1, Article 20 (March 2020), 29 pages. https://doi.org/10.1145/3379486

20

1 INTRODUCTION

Packet traces collected from networking applications, such as datacenter traffic, have been shown to feature much *structure*: datacenter traffic matrices are sparse and skewed [16, 39], exhibit

The Natural Question:

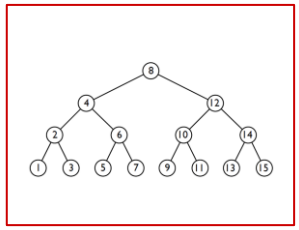
Given This Structure, What Can Be Achieved? Metrics and Algorithms?

A first insight: entropy of the demand.

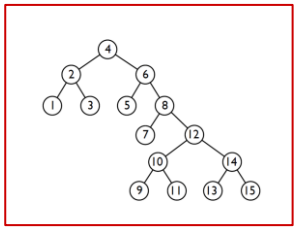
Insight:

Connection to Datastructures

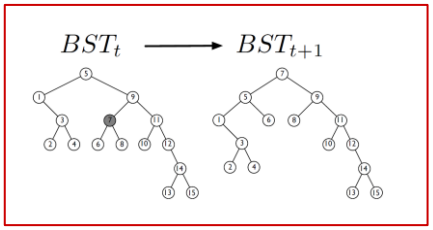
Traditional BST



Demand-aware BST



Self-adjusting BST

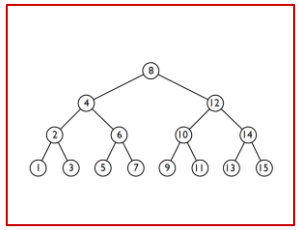


More structure: improved **access cost**

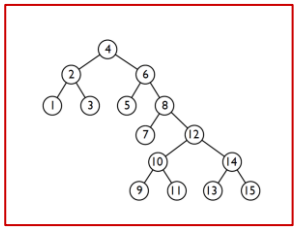
Insight:

Connection to Datastructures & Coding

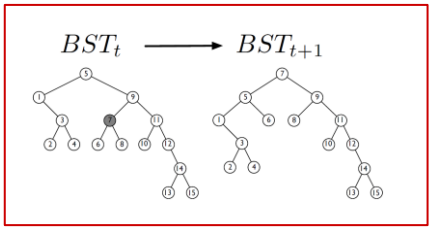
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)



Self-adjusting BST
(Dynamic Huffman coding)

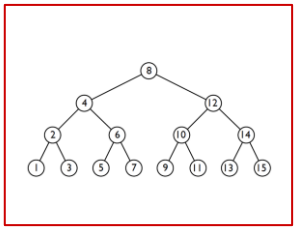


More structure: improved **access cost** / shorter **codes**

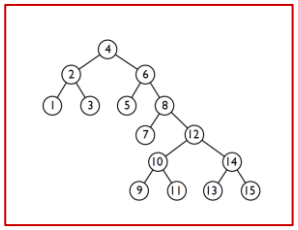
Insight:

Connection to Datastructures & Coding

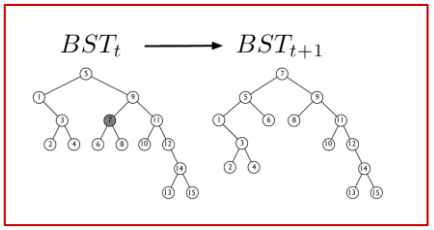
Traditional BST
(Worst-case coding)



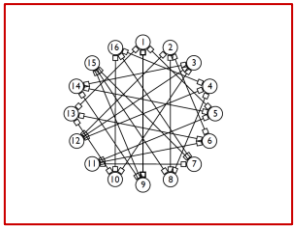
Demand-aware BST
(Huffman coding)

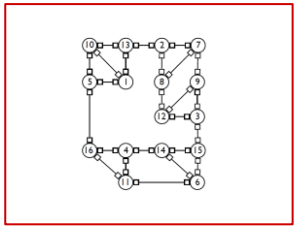


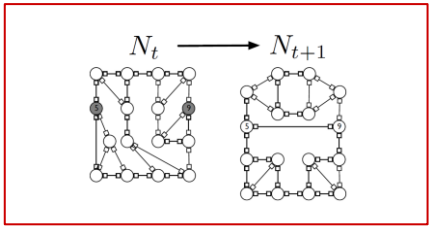
Self-adjusting BST
(Dynamic Huffman coding)



More structure: improved **access cost** / shorter **codes**





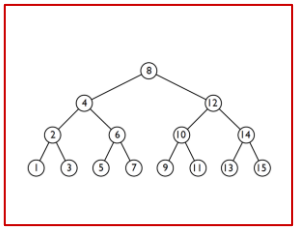


Similar **benefits**?

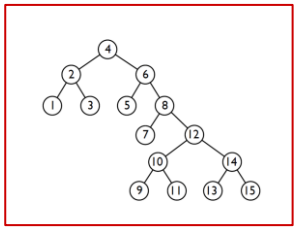
Insight:

Connection to Datastructures & Coding

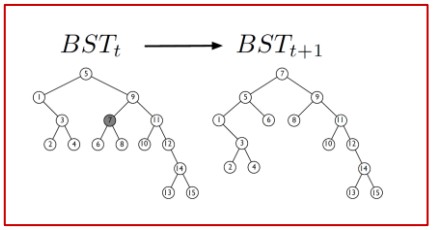
Traditional BST
(Worst-case coding)



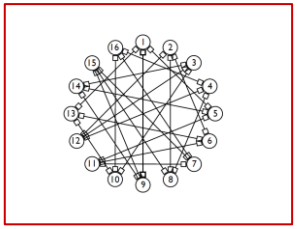
Demand-aware BST
(Huffman coding)

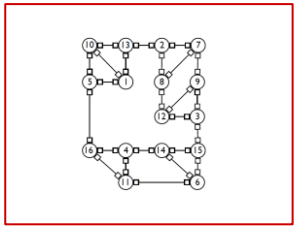


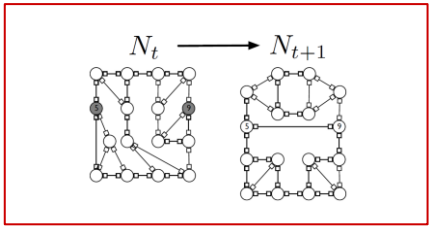
Self-adjusting BST
(Dynamic Huffman coding)



More structure: improved **access cost** / shorter **codes**







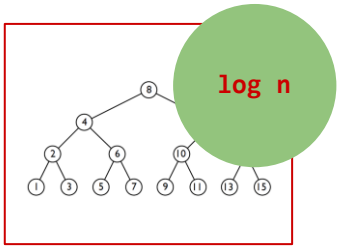
Similar **benefits**?

More than
an analogy!

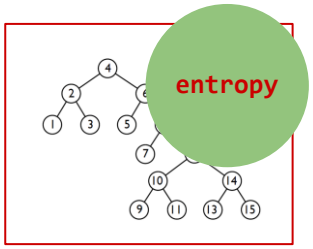
Insight:

Connection to Datastructures & Coding

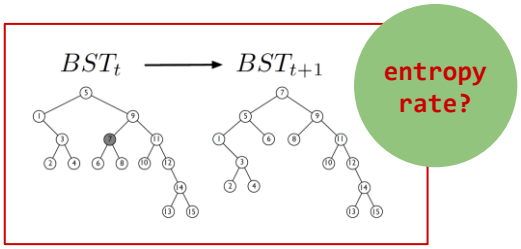
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)



Self-adjusting BST
(Dynamic Huffman coding)



Reduced expected **route lengths**!

More than an analogy!

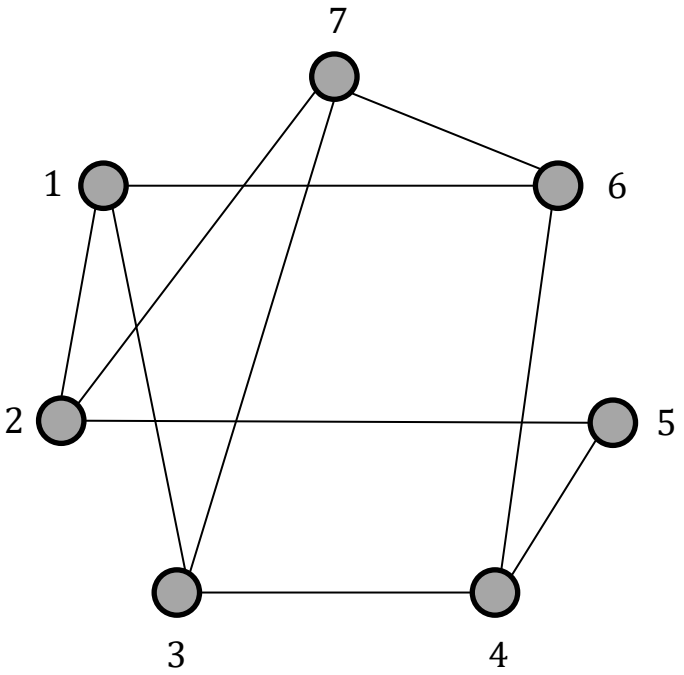
Generalize methodology:
... and transfer entropy bounds and algorithms of data-structures to networks.

First result:
Demand-aware networks of asymptotically optimal route lengths.

Case Study "Route Lengths"

Constant-Degree Demand-Aware Network

Sources \ Destinations	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$	0	$\frac{3}{65}$	$\frac{4}{65}$	0	0	0
6	$\frac{2}{65}$	0	0	0	0	0	$\frac{3}{65}$
7	$\frac{3}{65}$	$\frac{2}{65}$	$\frac{1}{13}$	0	0	$\frac{3}{65}$	0

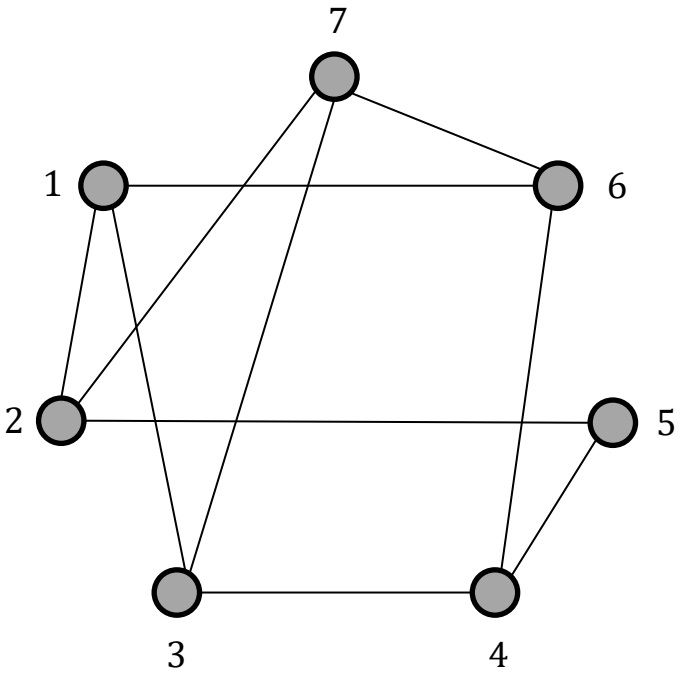


$$\mathrm{ERL}(\mathcal{D},\mathrm{N}) = \sum_{(u,v)\in\mathcal{D}} p(u,v)\cdot d_N(u,v)$$

Case Study "Route Lengths"

Constant-Degree Demand-Aware Network

Sources \ Destinations	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$	0	$\frac{3}{65}$	$\frac{4}{65}$	0	0	0
6	$\frac{2}{65}$	0	0	0	0	0	$\frac{3}{65}$
7	$\frac{3}{65}$	$\frac{2}{65}$	$\frac{1}{13}$	0	0	$\frac{3}{65}$	0



$$\mathrm{ERL}(\mathcal{D},N) = \sum_{(u,v)\in\mathcal{D}} p(u,v)\cdot d_N(u,v)$$

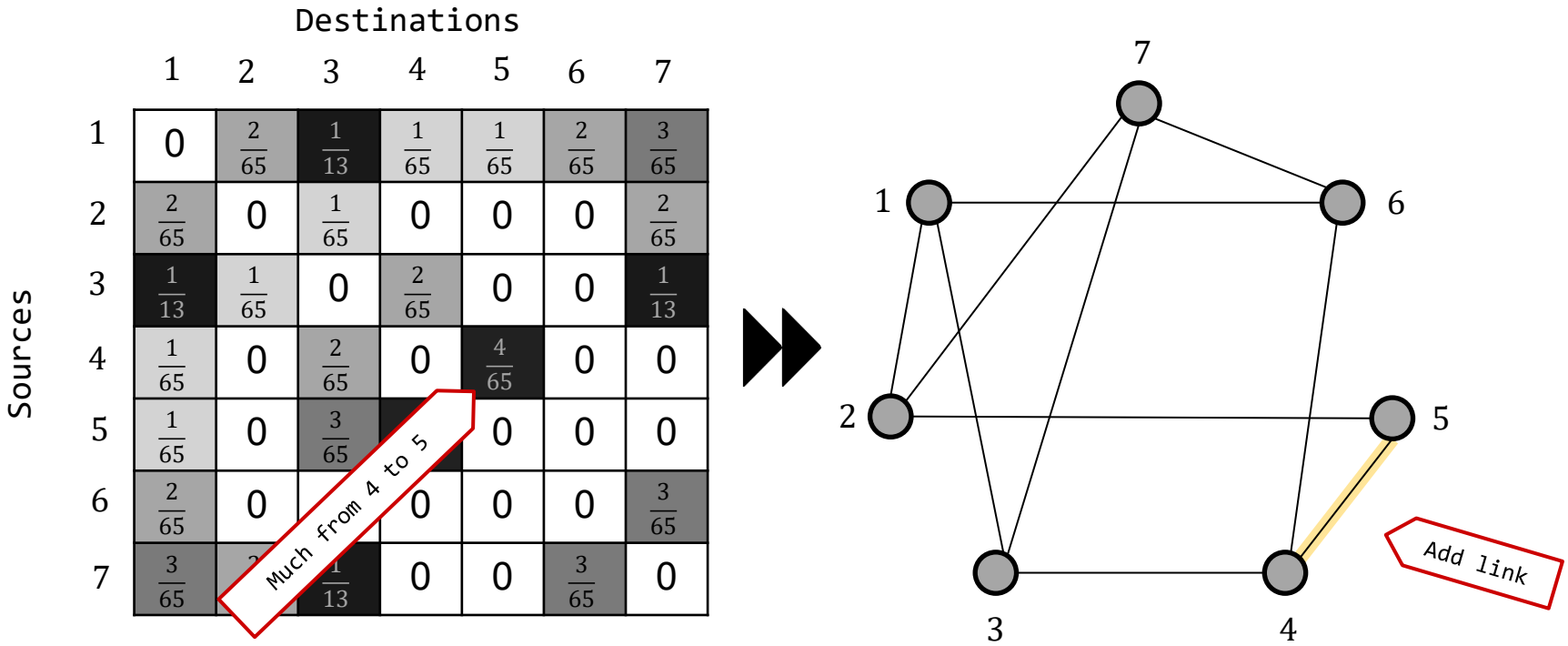
Expected Route Length

Demand

DAN

Case Study “Route Lengths”

Constant-Degree Demand-Aware Network



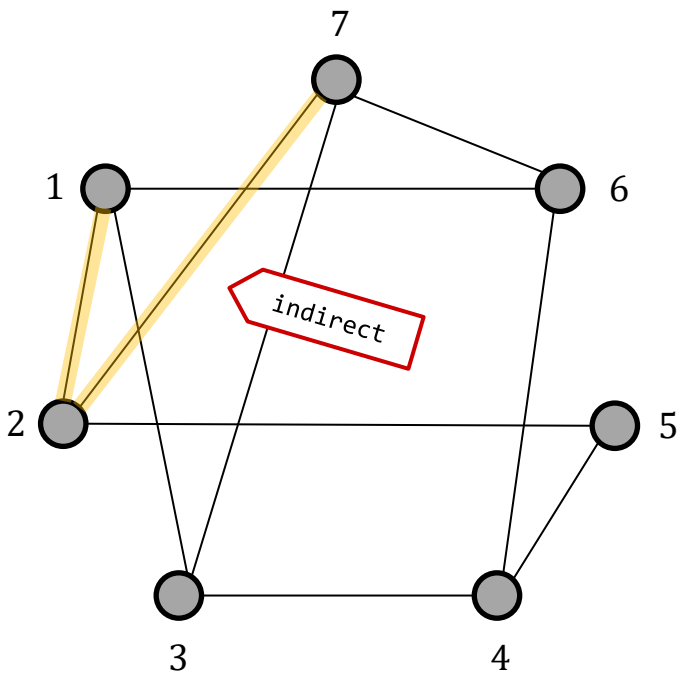
$$\mathrm{ERL}(\mathcal{D},\mathrm{N}) = \sum_{(u,v)\in\mathcal{D}} p(u,v)\cdot d_N(u,v)$$

Case Study "Route Lengths"

Constant-Degree Demand-Aware Network

Communicated with many

Sources \ Destinations	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$	0	$\frac{3}{65}$	$\frac{4}{65}$	0	0	0
6	$\frac{2}{65}$	0	0	0	0	0	$\frac{3}{65}$
7	$\frac{3}{65}$	$\frac{2}{65}$	$\frac{1}{13}$	0	0	$\frac{3}{65}$	0



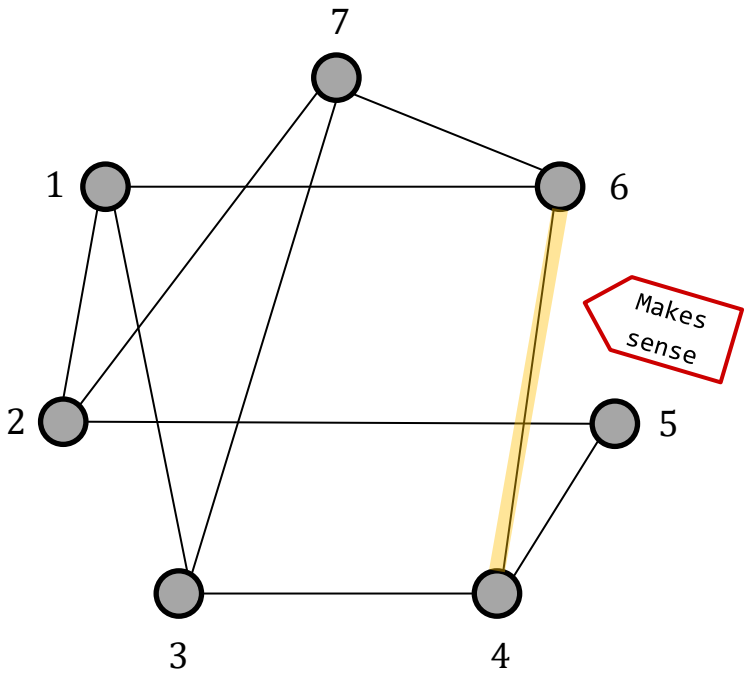
$$\mathrm{ERL}(\mathcal{D},N) = \sum_{(u,v)\in\mathcal{D}} p(u,v)\cdot d_N(u,v)$$

Case Study "Route Lengths"

Constant-Degree Demand-Aware Network

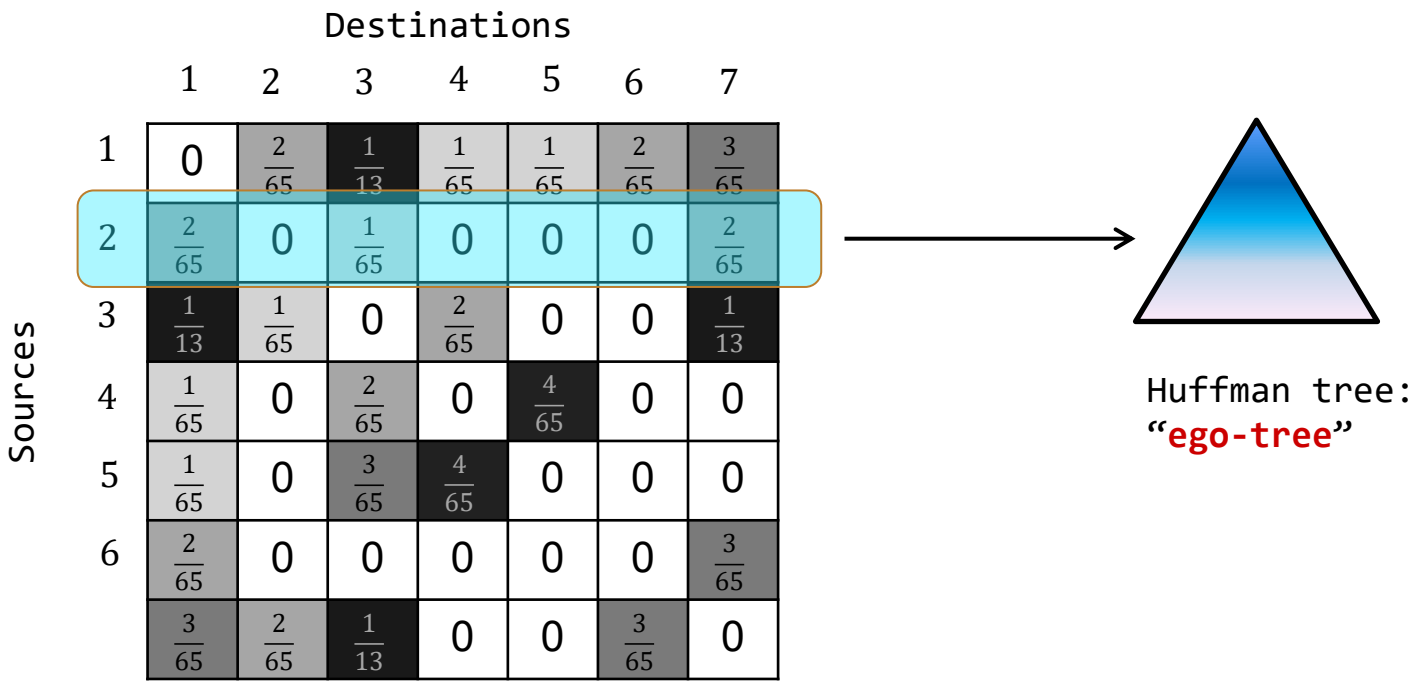
Sources \ Destinations	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	$\frac{2}{65}$	
3	$\frac{1}{13}$	$\frac{1}{65}$	0		0	$\frac{1}{13}$	
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{2}{65}$	0	0
5	$\frac{1}{65}$	0	$\frac{3}{65}$	$\frac{4}{65}$	0	0	0
6	$\frac{2}{65}$	0	0	0	0	0	$\frac{3}{65}$
7	$\frac{3}{65}$	$\frac{2}{65}$	$\frac{1}{13}$	0	0	$\frac{3}{65}$	0

Don't communicate

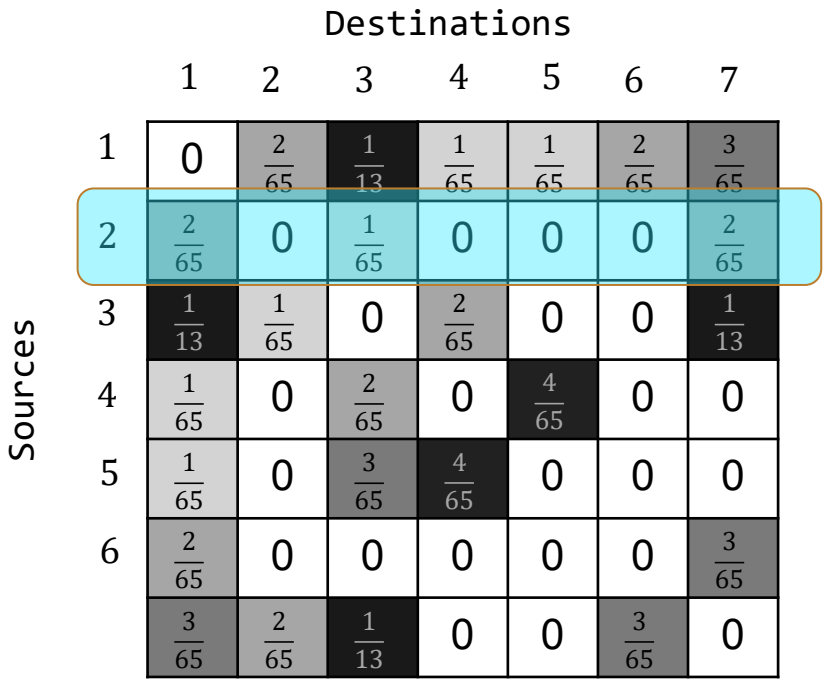


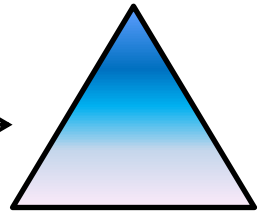
$$\mathrm{ERL}(\mathcal{D},N) = \sum_{(u,v)\in\mathcal{D}} p(u,v)\cdot d_N(u,v)$$

Algorithm: Idea



Algorithm: Idea



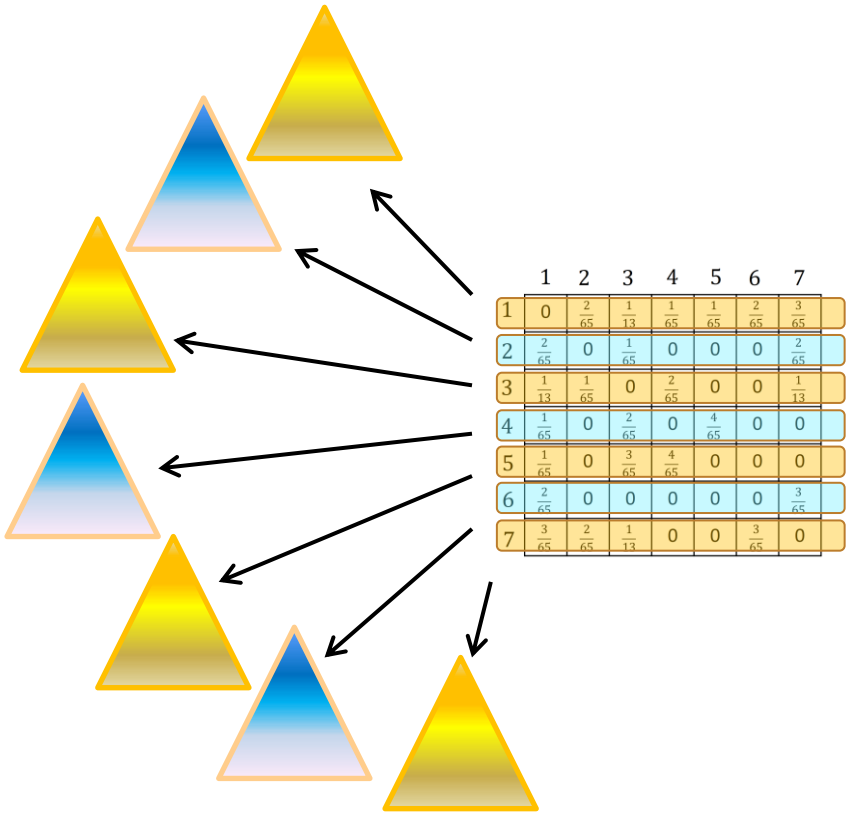


Huffman tree:
"**ego-tree**"

Cost:
Entropy!

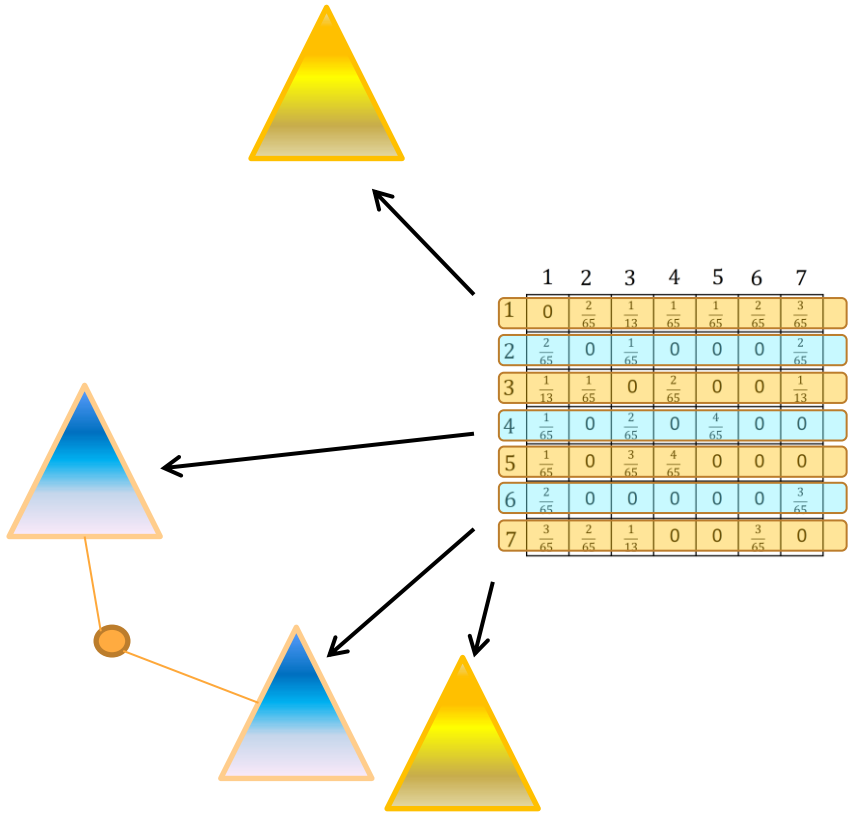
Entropy Upper Bound

- ⇢ Idea for algorithm:
  - ⇀ Union of trees
  - ⇀ Reduce degree
  - ⇀ But keep distances



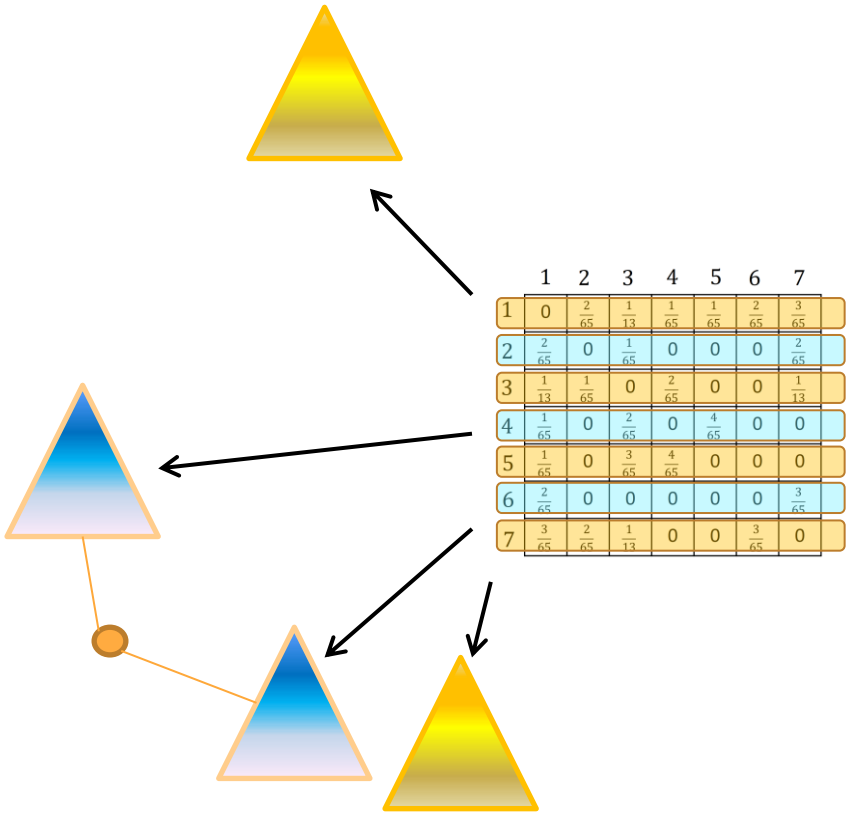
Entropy Upper Bound

- Idea for algorithm:
  - Union of trees
  - Reduce degree
  - But keep distances
- Ok for sparse demands
  - Not everyone gets tree
  - Helper nodes



Entropy Upper Bound

Gives upper bound of conditional entropy*



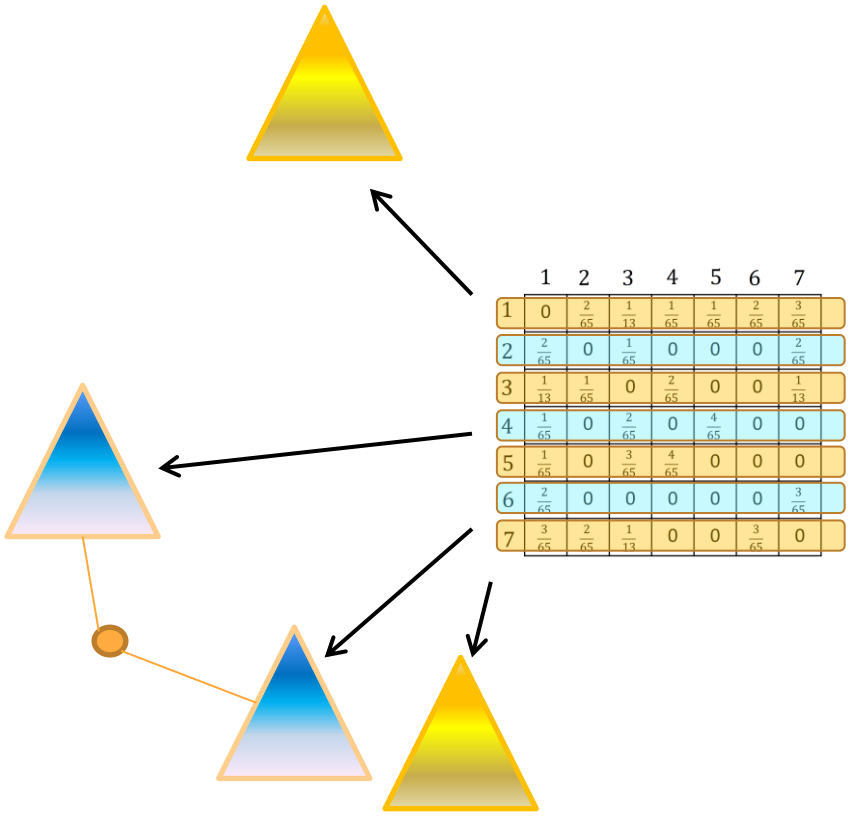
	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$	0	$\frac{3}{65}$	$\frac{4}{65}$	0	0	0
6	$\frac{2}{65}$	0	0	0	0	0	$\frac{3}{65}$
7	$\frac{3}{65}$	$\frac{2}{65}$	$\frac{1}{13}$	0	0	$\frac{3}{65}$	0

*$O(H(X|Y)+H(Y|X)$

Entropy Upper Bound

Gives upper bound of conditional entropy*

Matching lower bound for sparse demands!



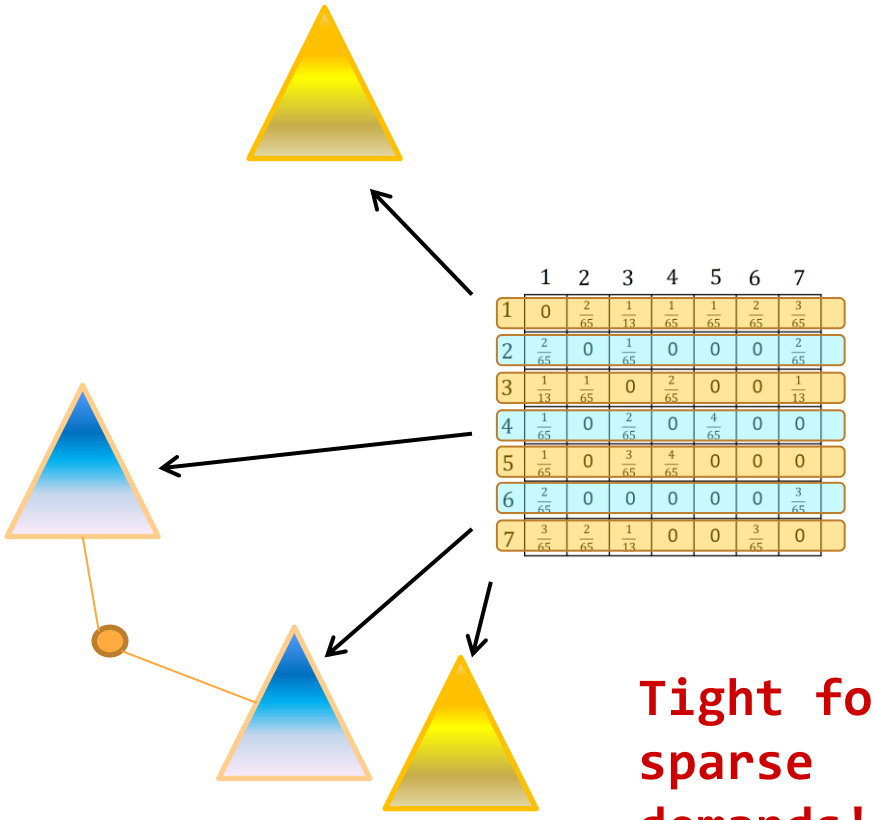
	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$	0	$\frac{3}{65}$	$\frac{4}{65}$	0	0	0
6	$\frac{2}{65}$	0	0	0	0	0	$\frac{3}{65}$
7	$\frac{3}{65}$	$\frac{2}{65}$	$\frac{1}{13}$	0	0	$\frac{3}{65}$	0

*$O(H(X|Y)+H(Y|X))$

Entropy Upper Bound

Gives upper bound of conditional entropy*

Matching lower bound for sparse demands!



	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$	0	$\frac{3}{65}$	$\frac{4}{65}$	0	0	0
6	$\frac{2}{65}$	0	0	0	0	0	$\frac{3}{65}$
7	$\frac{3}{65}$	$\frac{2}{65}$	$\frac{1}{13}$	0	0	$\frac{3}{65}$	0

Tight for sparse demands!

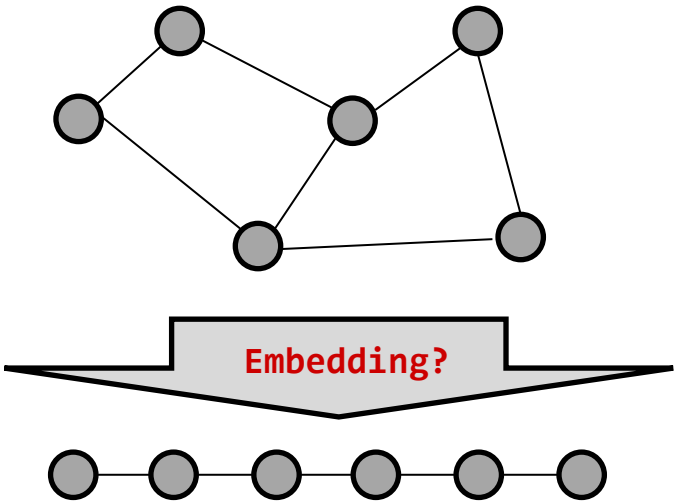
*$O(H(X|Y)+H(Y|X))$

Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\triangle$=2: A Minium Linear Arrangement (**MLA**) Problem

- → Minimizes sum of virtual edges

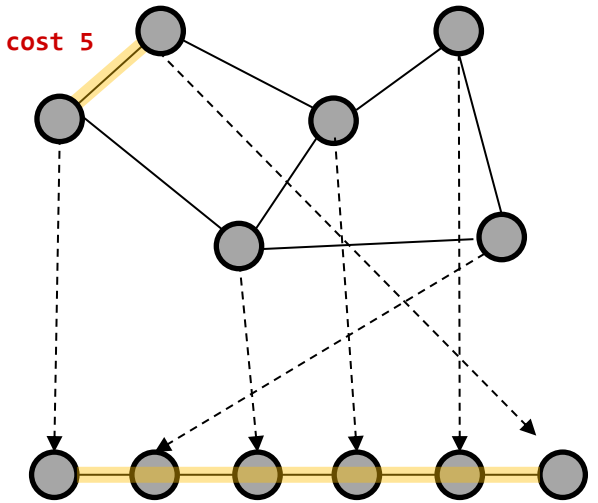


Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\triangle$=2: A Minium Linear Arrangement (**MLA**) Problem

- ⇀ Minimizes sum of virtual edges



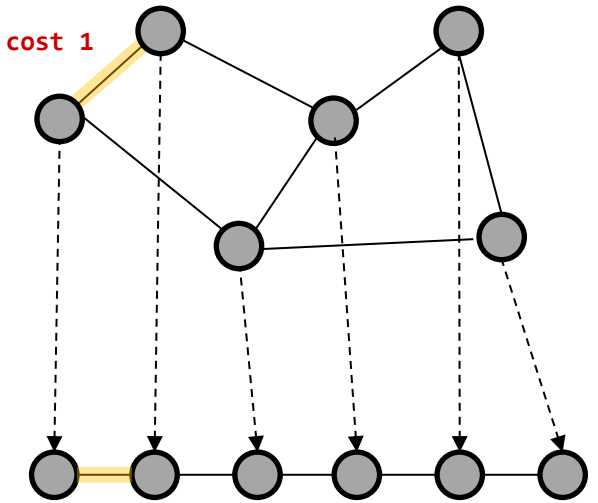
Bad!

Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\triangle$=2: A Minium Linear Arrangement (**MLA**) Problem

- ⇀ Minimizes sum of virtual edges



Good!

Related Problem:

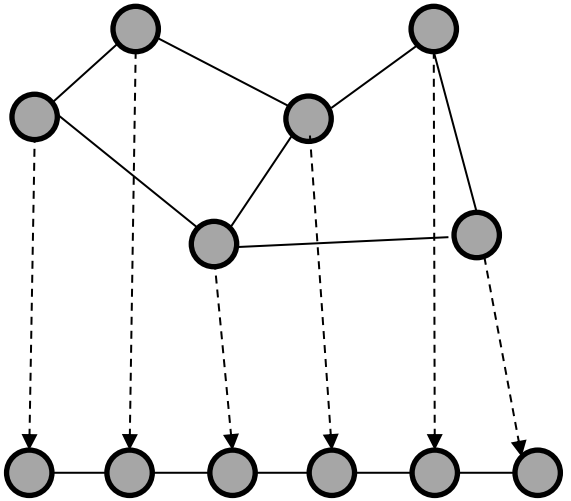
Virtual Network Embedding Problem (VNEP)

Example $\triangle$=2: A Minium Linear Arrangement (**MLA**) Problem

- → Minimizes sum of virtual edges

MLA is **NP-hard**

- → … and so is our problem!



Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\triangle$=2: A Minium Linear Arrangement (**MLA**) Problem

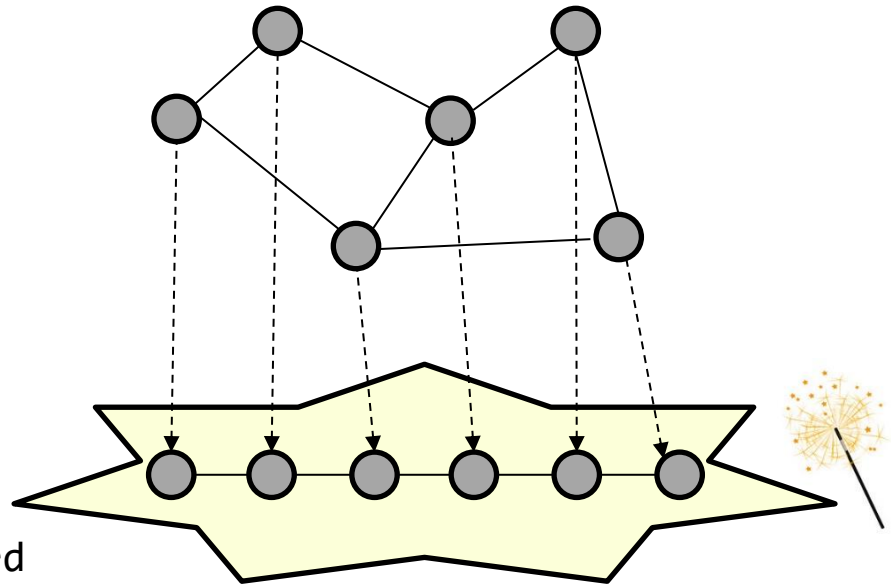
- → Minimizes sum of virtual edges

MLA is **NP-hard**

- → … and so is our problem!

But what about **$\triangle$>2**?

- → Embedding problem still hard
- → But we have a new **degree of freedom**!



Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\triangle$=2: A Minium Linear Arrangement (**MLA**) Problem

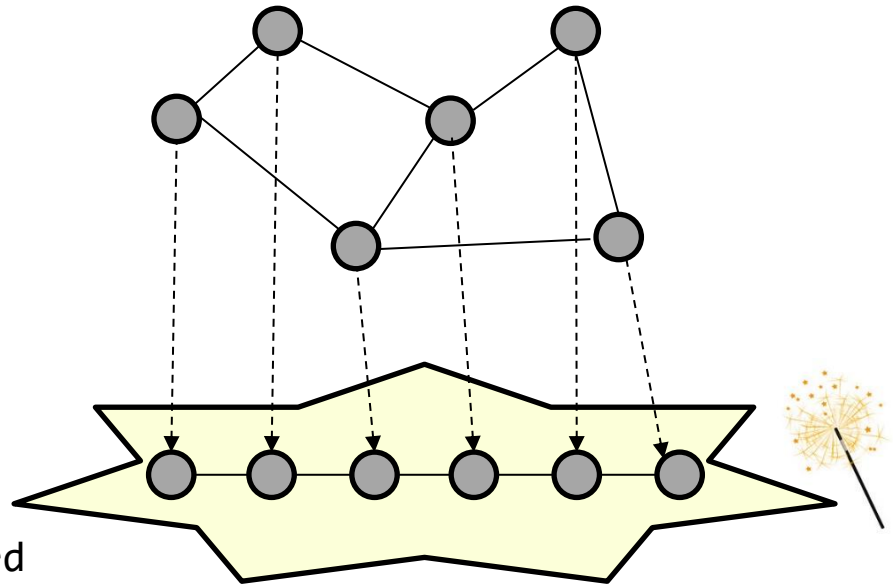
- → Minimizes sum of virtual edges

MLA is **NP-hard**

- → … and so is our problem!

But what about **$\triangle$>2**?

- → Embedding problem still hard
- → But we have a new **degree of freedom**!



Simplifies problem?!

Remark

VNEP vs DAN Design?

⇢ Idea to design demand-aware network:
  ⇢ Choose a "**universally** good" **host graph**, e.g., an **expander**, in oblivious manner
  ⇢ Optimally **embed** the demand matrix (**guest graph**)

Remark

VNEP vs DAN Design?

⇢ Idea to design demand-aware network:
  ⇢ Choose a "**universally** good" **host graph**, e.g., an **expander**, in oblivious manner
  ⇢ Optimally **embed** the demand matrix (**guest graph**)

⇢ Expected path length can be **Ω(log log *n*) factor** more than on the optimal host graph, for networks of size *n*.

Another Related Problem

Low Distortion Spanners

⇢ Classic problem: find ***sparse***, ***distance-preserving*** (low-distortion) spanner of a graph

⇢ But:

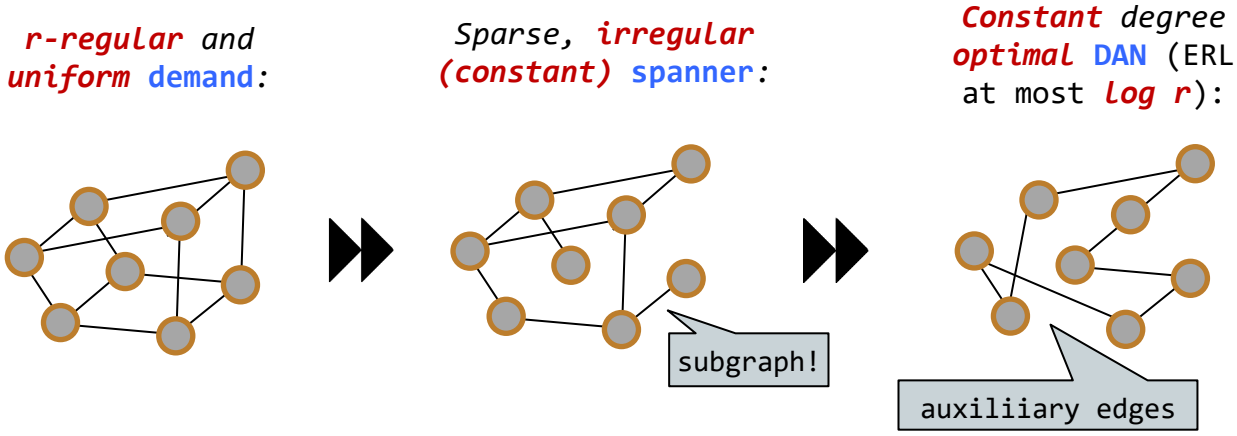
- ⇢ Spanners aim at low distortion ***among all pairs***; in our case, we are only interested in the **distortion** between **communication neighbors** (in demand matrix)
- ⇢ We allow ***auxiliary edges*** (not a subgraph): similar to geometric spanners
- ⇢ We require ***constant degree***

From Spanners to DANs

An Algorithm

⇢ Yet, can ***leverage the connection*** to spanners sometimes!

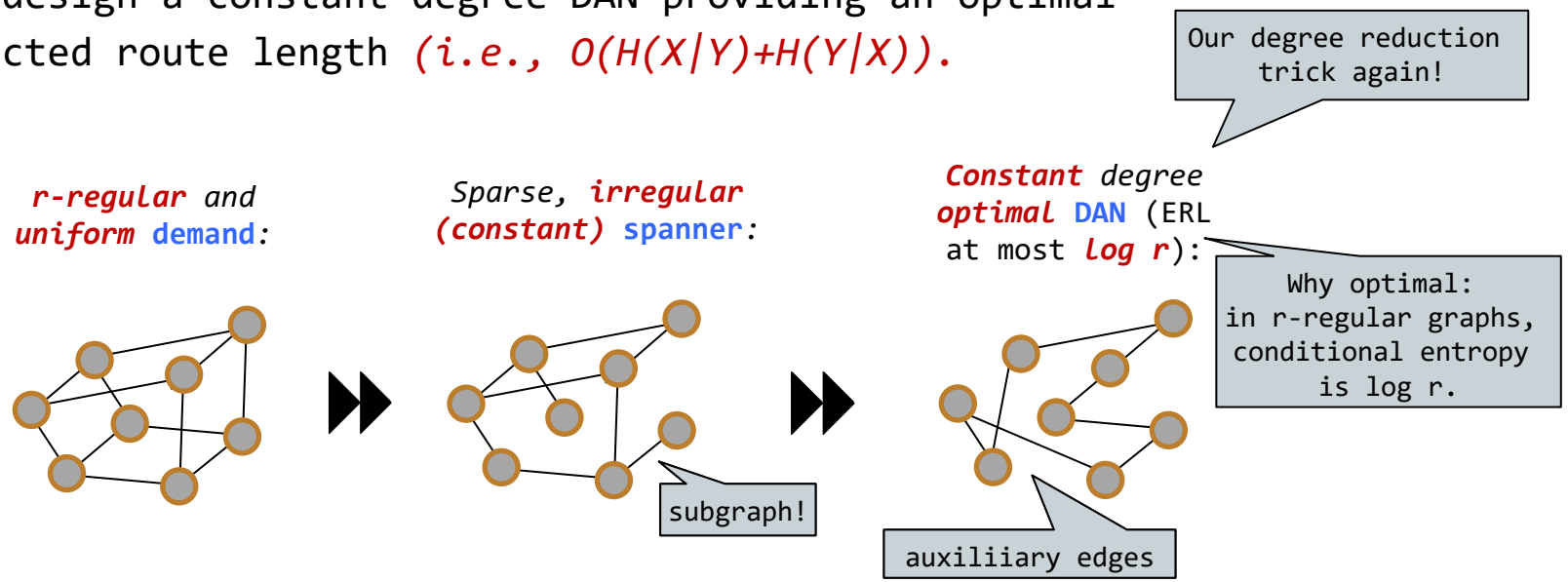
Theorem: If demand matrix is ***regular and uniform***, and if we can find a constant distortion, linear sized (i.e., ***constant, sparse***) spanner for this request graph: then we can design a constant degree DAN providing an optimal expected route length *(i.e., O(H(X|Y)+H(Y|X)).*



An Algorithm

⇢ Yet, can ***leverage the connection*** to spanners sometimes!

Theorem: If demand matrix is ***regular and uniform***, and if we can find a constant distortion, linear sized (i.e., ***constant, sparse***) spanner for this request graph: then we can design a constant degree DAN providing an optimal expected route length *(i.e., O(H(X|Y)+H(Y|X)).*



Further Reading

Dist.Comp. 2020

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, Δ, on the maximum degree. In turn, our objective is to design an (undirected) demand-aware network $N = (V, E)$ of bounded degree Δ, which provides

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 overprovisioned 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 man-*

Further Reading

Dist.Comp. 2020

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, Δ, on the maximum degree. In turn, our objective is to design an (undirected) demand-aware network $N = (V, E)$ of bounded degree Δ, which provides

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 man-*

Also includes spanner result and other special graphs (e.g., bounded doubling dimension). Open problem: dense demands.

Further Reading

OPODIS 2024

Efficient Algorithms for Demand-Aware Networks and a Connection to Virtual Network Embedding

Aleksander Figiel
TU Berlin, Germany

Janne H. Korhonen
TU Berlin, Germany

Neil Olver
London School of Economics and Political Science, UK

Stefan Schmid
TU Berlin, Germany
Fraunhofer SIT, Berlin, Germany

Abstract

Emerging optical switching technologies enable demand-aware datacenter networks, whose topology can be flexibly optimized toward the traffic they serve. This paper revisits the *bounded-degree network design problem* underlying such demand-aware networks. Namely, given a distribution over communicating node pairs (represented has a demand graph), we want to design a network with bounded maximum degree (called host graph) that minimizes the expected communication distance.

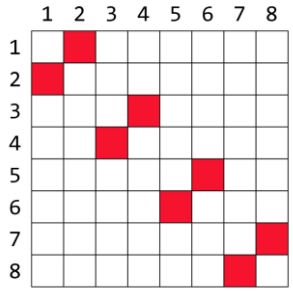
We improve the understanding of this problem domain by filling several gaps in prior work. First, we present the first practical algorithm for solving this problem on arbitrary instances without violating the degree bound. Our algorithm is based on novel insights obtained from studying a new *Steiner node* version of the problem, and we report on an extensive empirical evaluation, using several real-world traffic traces from datacenters, finding that our approach results in improved

VNEP vs DAN

Throughput of DANs?

Demand Matrix

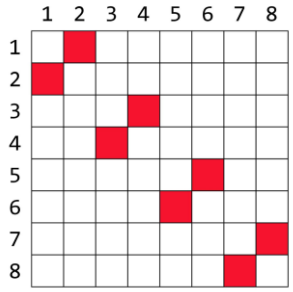
T



Metric: throughput of a demand matrix…

Throughput of DANs?

Demand Matrix

T 

$\times\ \theta(T)$

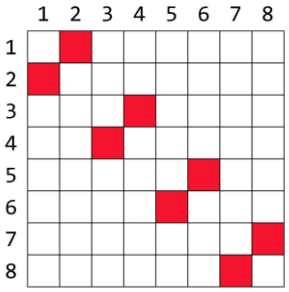
Metric: throughput of a demand matrix…

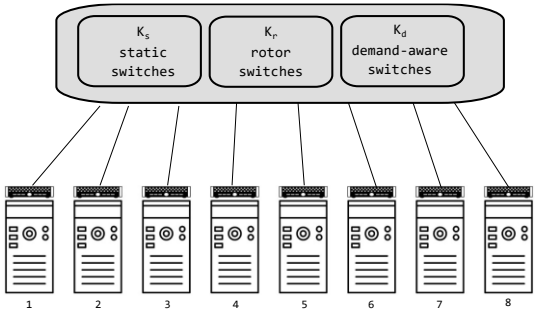
… is the maximal scale down factor by which traffic is feasible

$0 \leq \theta(T) \leq 1.$

Throughput of DANs?

Demand Matrix

T 

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

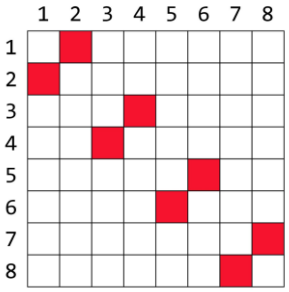
Metric: throughput of a demand matrix…

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

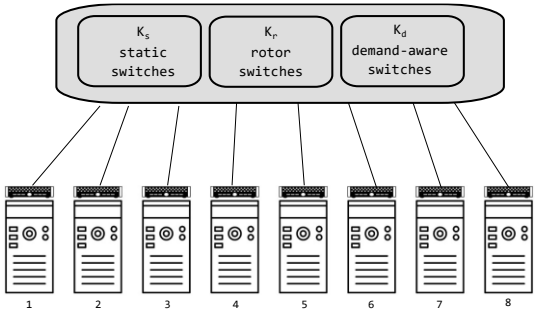
Throughput of network $\boldsymbol{\theta^*}$: worst case T

Throughput of DANs?

Demand Matrix

T 

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



Metric: throughput of a demand matrix…

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

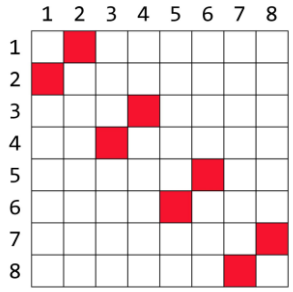
Throughput of network $\boldsymbol{\theta^*}$: worst case T

Definition extends to dynamic networks!

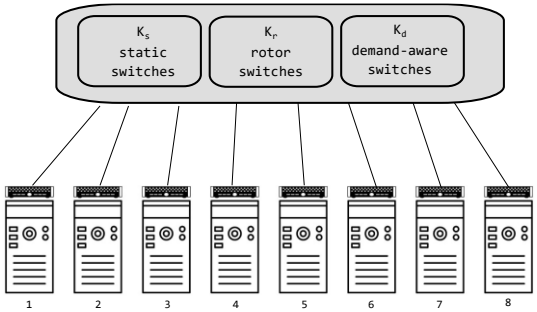
Abdu et al., SC 2016
Namyar et al., SIGCOMM 2021

Throughput of DANs?

Demand Matrix



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



Metric: throughput of a demand matrix…

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

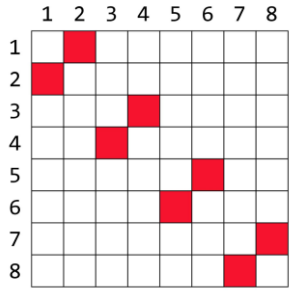
Throughput of network $\boldsymbol{\theta^*}$: worst case T

Definition extends to dynamic networks!

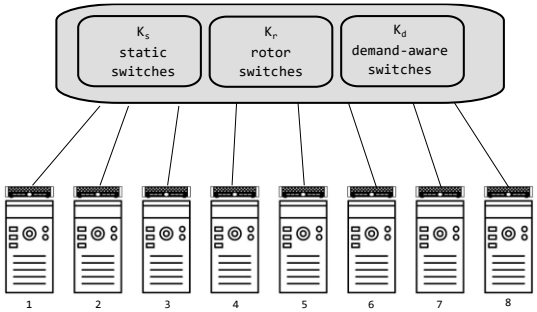
T typically assumed to be doubly stochastic (entries sum to 1).

Throughput of DANs?

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

Definition extends to dynamic networks!

T typically assumed to be doubly stochastic (entries sum to 1).

Worst T for different networks?

Worst *T* for Oblivious

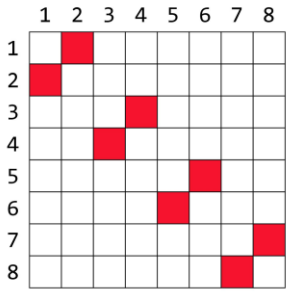
⇀ E.g., worst T for (oblivious) expander?

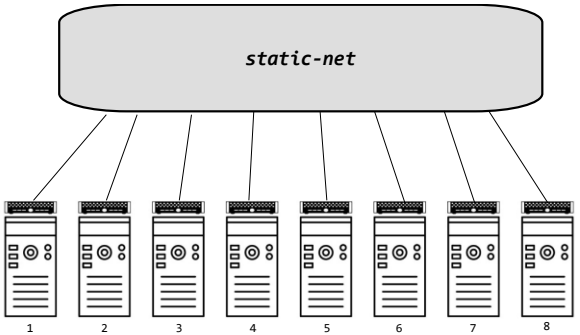
Worst T for Oblivious

⇀ E.g., worst T for (oblivious) expander?

Demand Matrix

T





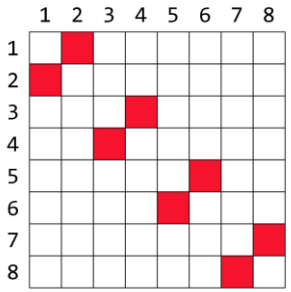
Permutation matrix
is the worst demand

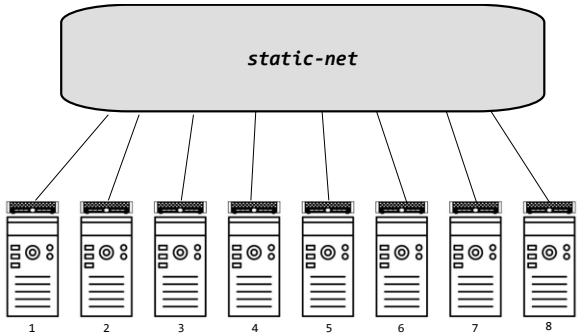
Worst T for Oblivious

→ E.g., worst T for (oblivious) expander?

Demand Matrix

T





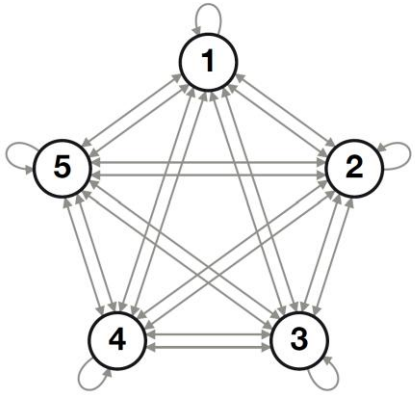
Permutation matrix
is the worst demand

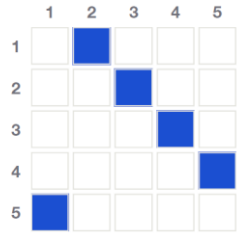
→ Actually the best T for demand-aware!

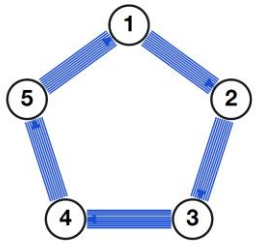
Throughput:

Many Open Questions (1)

- **Known:** worst case throughput θ^* of *demand-oblivious* networks at most ½ (tight)
  - Due to Keslassy, Chang, McKeown, Lee.
  - Best fixed graph: (2n-1) regular, two links to every other node





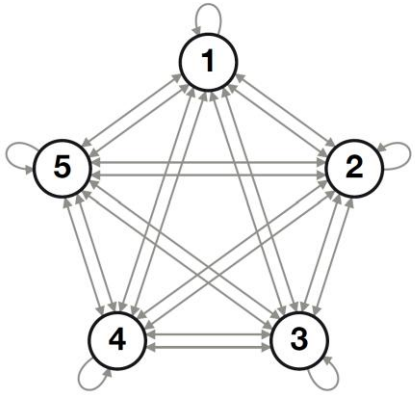


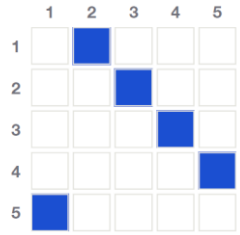
- For this graph, worst T is permutation
- Knowing the demand, each node *pointing all* (2n-1) links at its successor gives perfect *throughput 1*

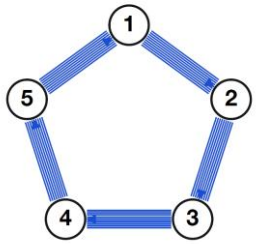
Throughput:

Many Open Questions (1)

→ **Known:** worst case throughput θ^* of *demand-oblivious* networks at most ½* (tight)
  → Due to Keslassy, Chang, McKeown, Lee.
  → Best fixed graph: (2n-1) regular, two links to every other node







→ For this graph, worst T is permutation
→ Knowing the demand, each node *pointing all* (2n-1) links at its successor gives perfect *throughput 1*

* Same bound also holds for *dynamic* oblivious networks

Throughput:

Many Open Questions (2)

→ ***Unknown:*** worst case throughput of ***demand-aware*** networks? Can never be worse than demand-oblivious, but is it always strictly better in the worst case as well?

Input: A doubly stochastic demand matrix M and a number $0 \le \theta \le 1$.
Question: Is there a graph G such that the throughput of G with respect to M is at least θ?

DEMAND-OBLIVIOUS

$$\theta^\star_{\mathrm{OB}} = \max_G \min_M \theta(G, M)$$

Fix G first — then face the worst demand.

DEMAND-AWARE

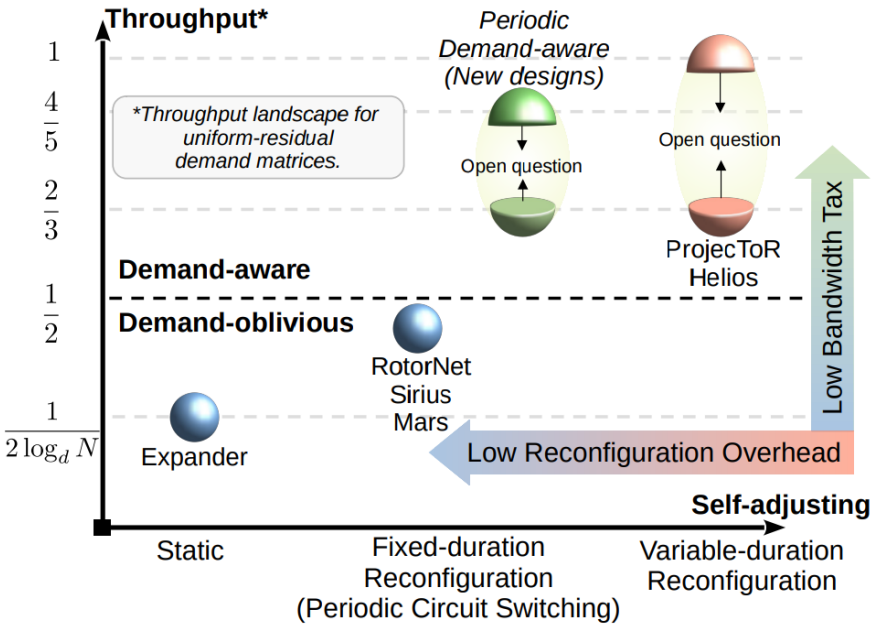
$$\theta^\star_{\mathrm{DA}} = \min_M \max_G \theta(G, M)$$

See M first — then choose the best G.

ACM PODC 2026: The demand-aware throughput θ^* asymptotically approaches at least ***5/8***. But: A ***probabilistic method*** proof.

More Research Directions

→ More open questions, for dynamic hybrid datacenter networks



Addanki et al., arXiv 2025:
https://arxiv.org/pdf/2405.20869

Addanki et al., Vermillion:
https://arxiv.org/pdf/2504.09892

Further Reading

ACM PODC 2026

A Separation Between Optimal Demand-Oblivious and Demand-Aware Network Throughput

Matthias Bentert[1], Chen Avin[2], and Stefan Schmid[1]

[1]Technische Universität Berlin, Germany
[2]Ben-Gurion University of the Negev, School of Electrical and Computer Engineering, Israel

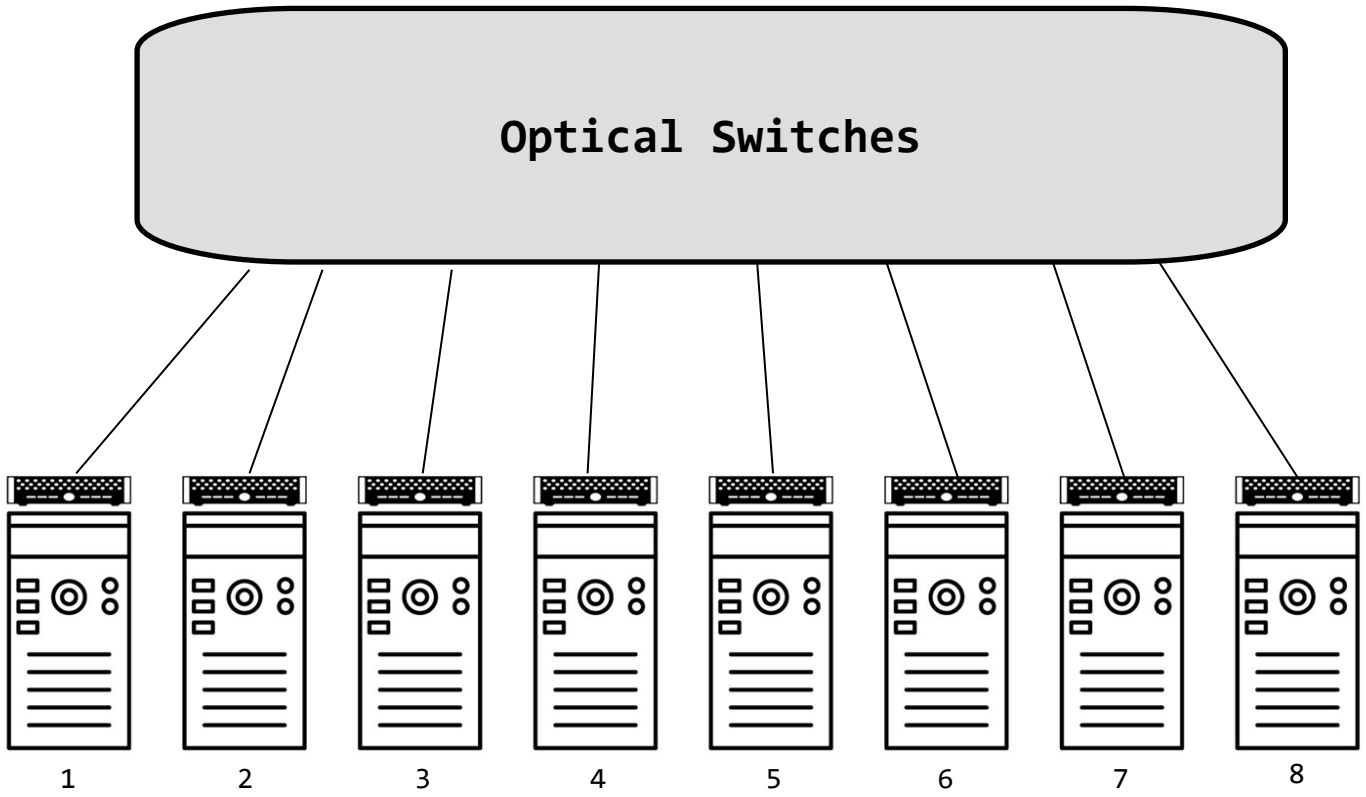
Abstract

The performance of distributed applications often critically depends on the interconnecting network or more specifically on its throughput: how fast data can be carried across a network. Over the last years, great progress has been made in understanding *demand-oblivious* throughput: how fast a given demand matrix describing pairwise communication requirements, can be served on a *given* network. However, surprisingly little is known today about the achievable *demand-aware* throughput: the throughput on a network topology which can be *optimized* toward the demand. Such demand-aware networks have recently gained popularity in datacenters and are enabled by emerging reconfigurable optical technologies.

From Static to Dynamic Neworks

Reconfigurable Datacenters in Reality:

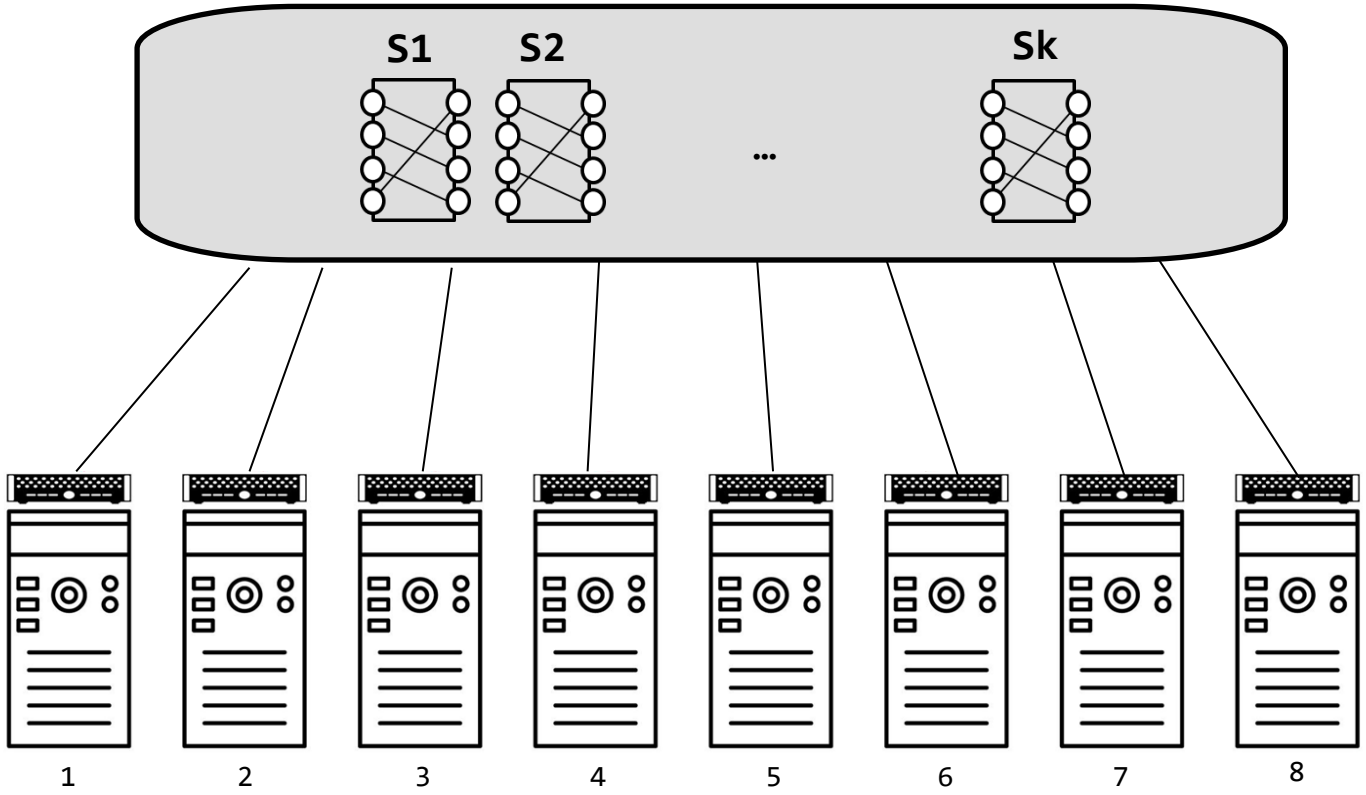
It's about Dynamic Matchings



Typical rack interconnect: ToR-Matching-ToR (TMT) model

Reconfigurable Datacenters in Reality:

It's about Dynamic Matchings

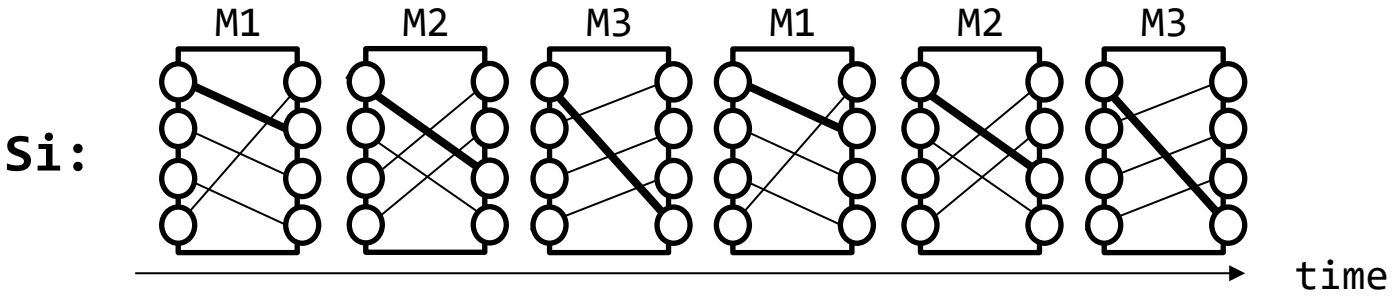


Optical switches provide *matchings* (-> *bounded degree*!)

Two Types of Reconfigurable Optical Switches

Periodically Rotating

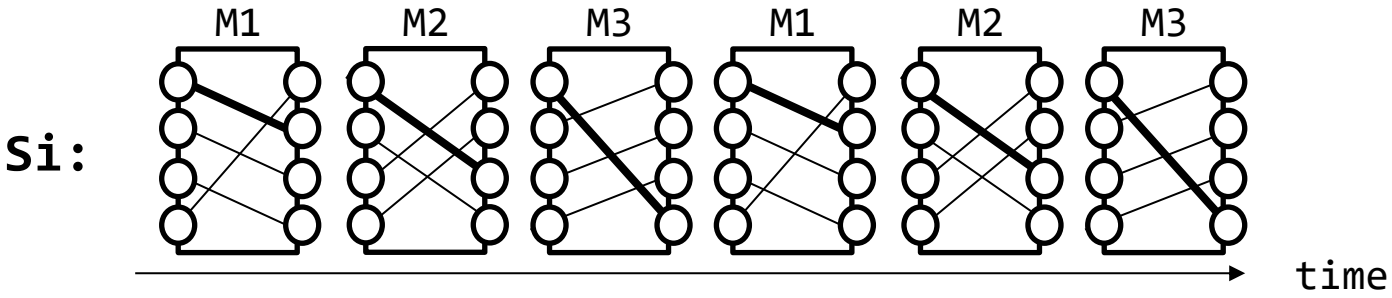
Rotor switch: periodic matchings (demand-oblivious)



Two Types of Reconfigurable Optical Switches

Periodically Rotating

Rotor switch: periodic matchings (demand-oblivious)

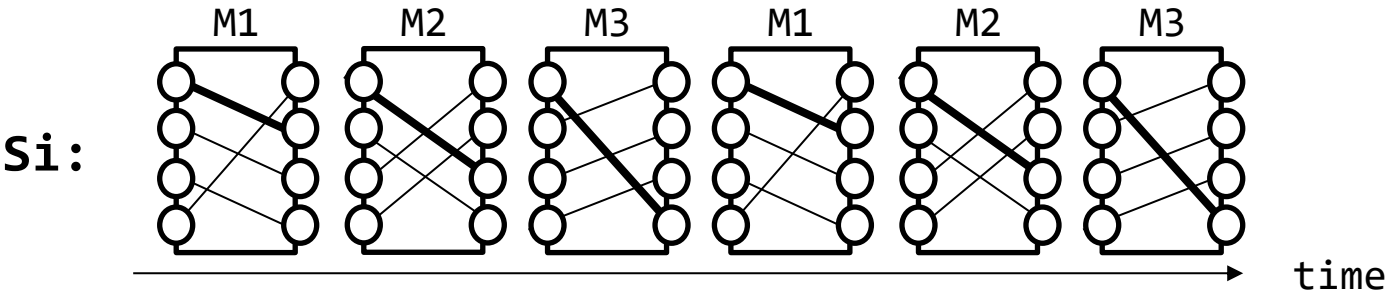


Advantage/disadvantage?

Two Types of Reconfigurable Optical Switches

Periodically Rotating

Rotor switch: periodic matchings (demand-oblivious)



Advantage/disadvantage?

\+ *Direct* connections

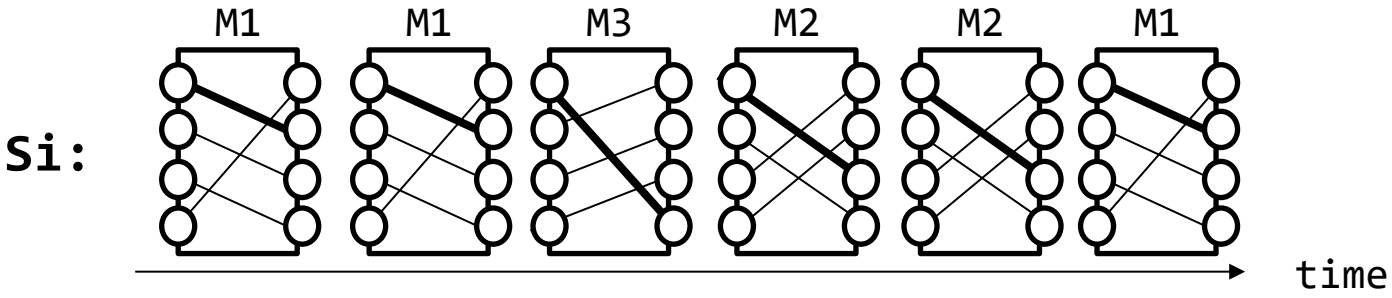
\- Not always connected

\- *Reconfiguration delays*

Two Types of Reconfigurable Optical Switches

Demand-Aware Dynamic

Demand-aware switch: optimized matchings

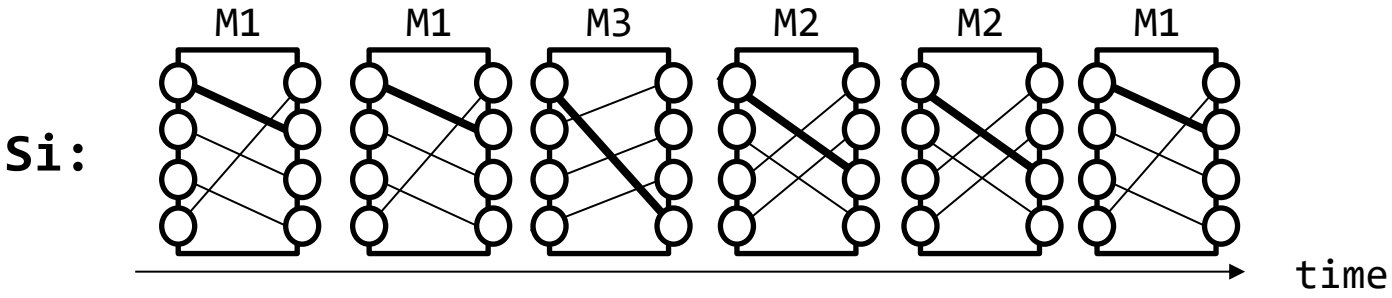


Advantage/disadvantage?

Two Types of Reconfigurable Optical Switches

Demand-Aware Dynamic

Demand-aware switch: optimized matchings



Advantage/disadvantage?

\+ Direct *optimized* connections

\- Not always connected

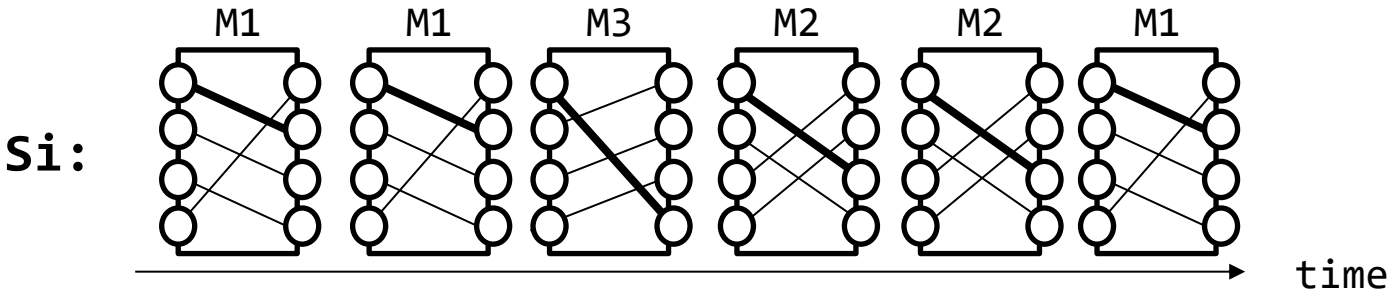
\- Reconfiguration delays

\- *Optimization delays*

Two Types of Reconfigurable Optical Switches

Demand-Aware Dynamic

Demand-aware switch: optimized matchings

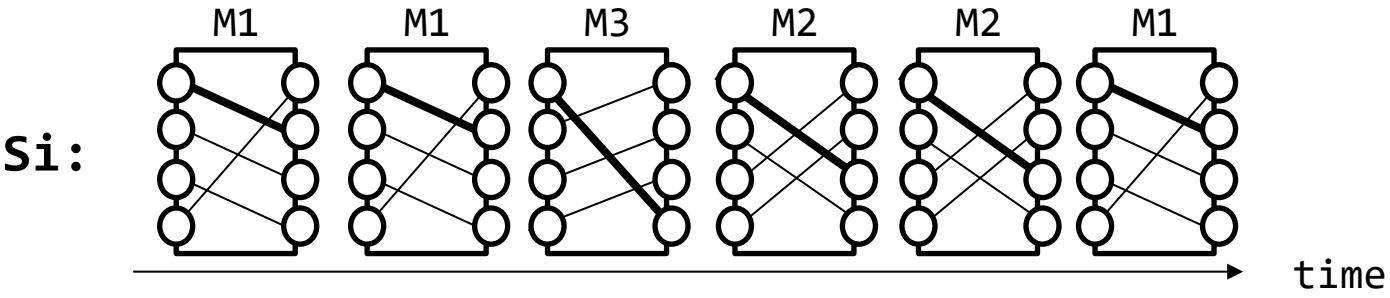


Algorithms?

Two Types of Reconfigurable Optical Switches

Demand-Aware Dynamic

Demand-aware switch: optimized matchings



Algorithms?

→ E.g., *Birkhoff von Neumann* Decomposition

→ But does not support reconfiguration delays and multi-hop

Opportunity: Tech Diversity

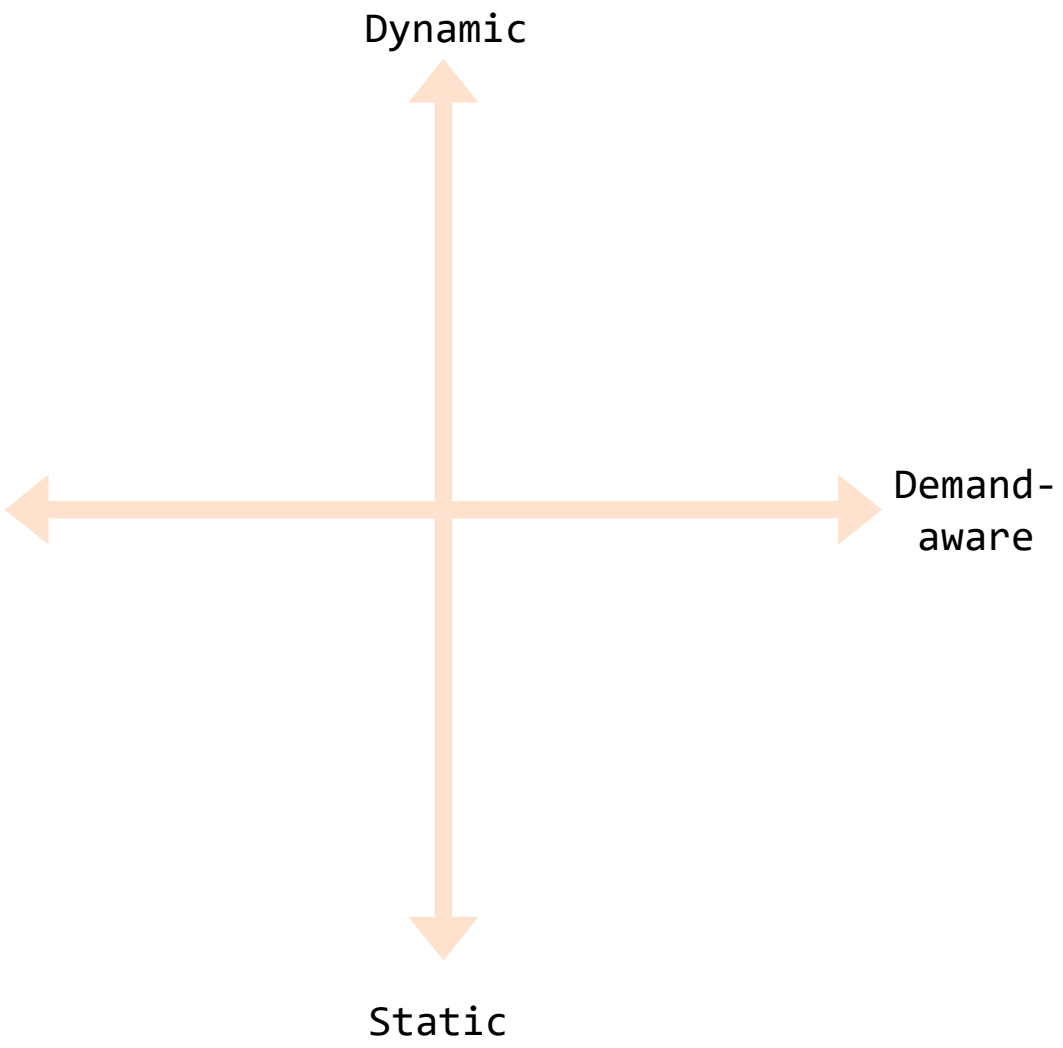
Diverse topology components:

→ demand-oblivious and
demand-aware

Opportunity: Tech Diversity

Diverse topology components:

- demand-oblivious and demand-aware
- static vs dynamic

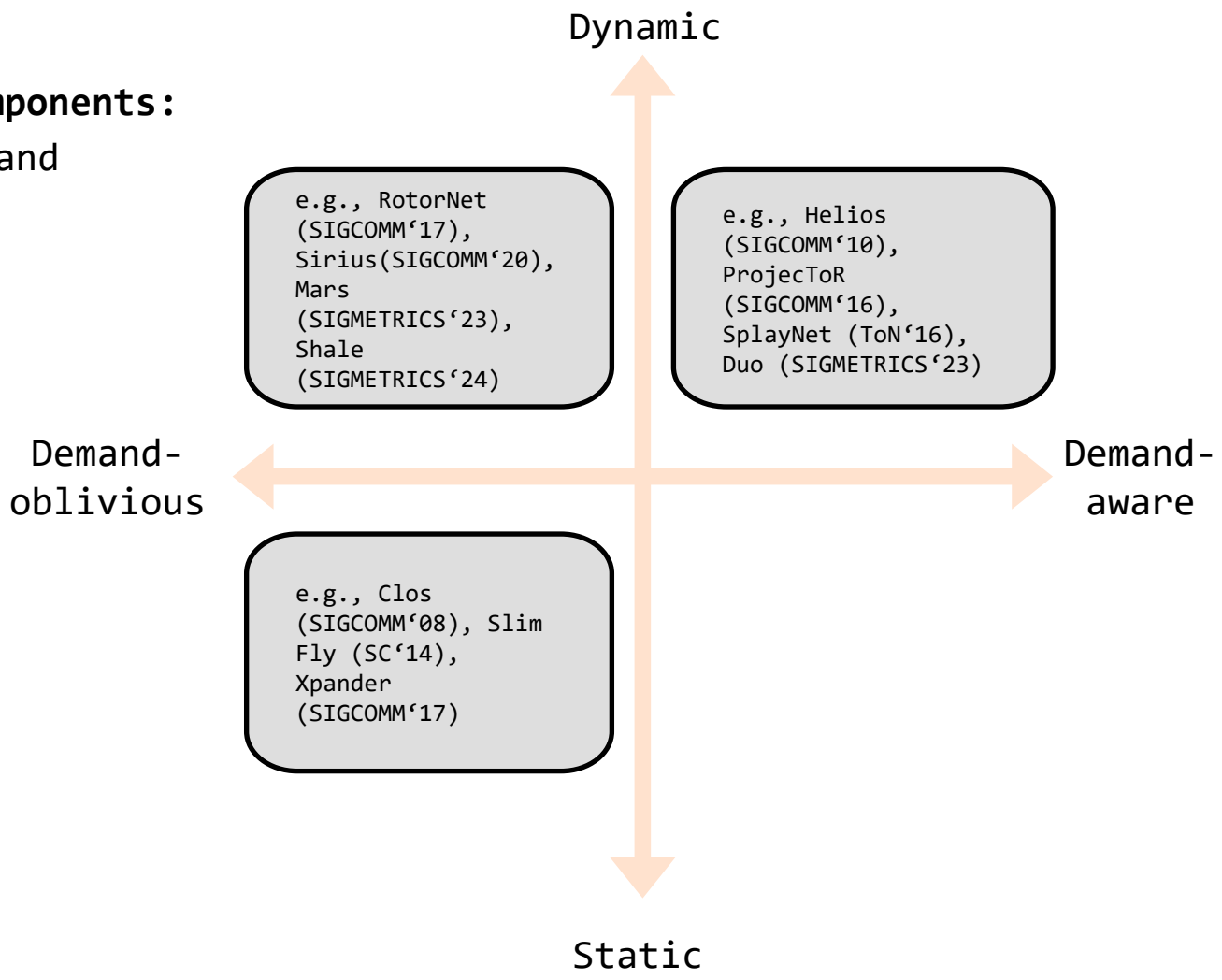


Opportunity: Tech Diversity

Diverse topology components:

→ demand-oblivious and demand-aware

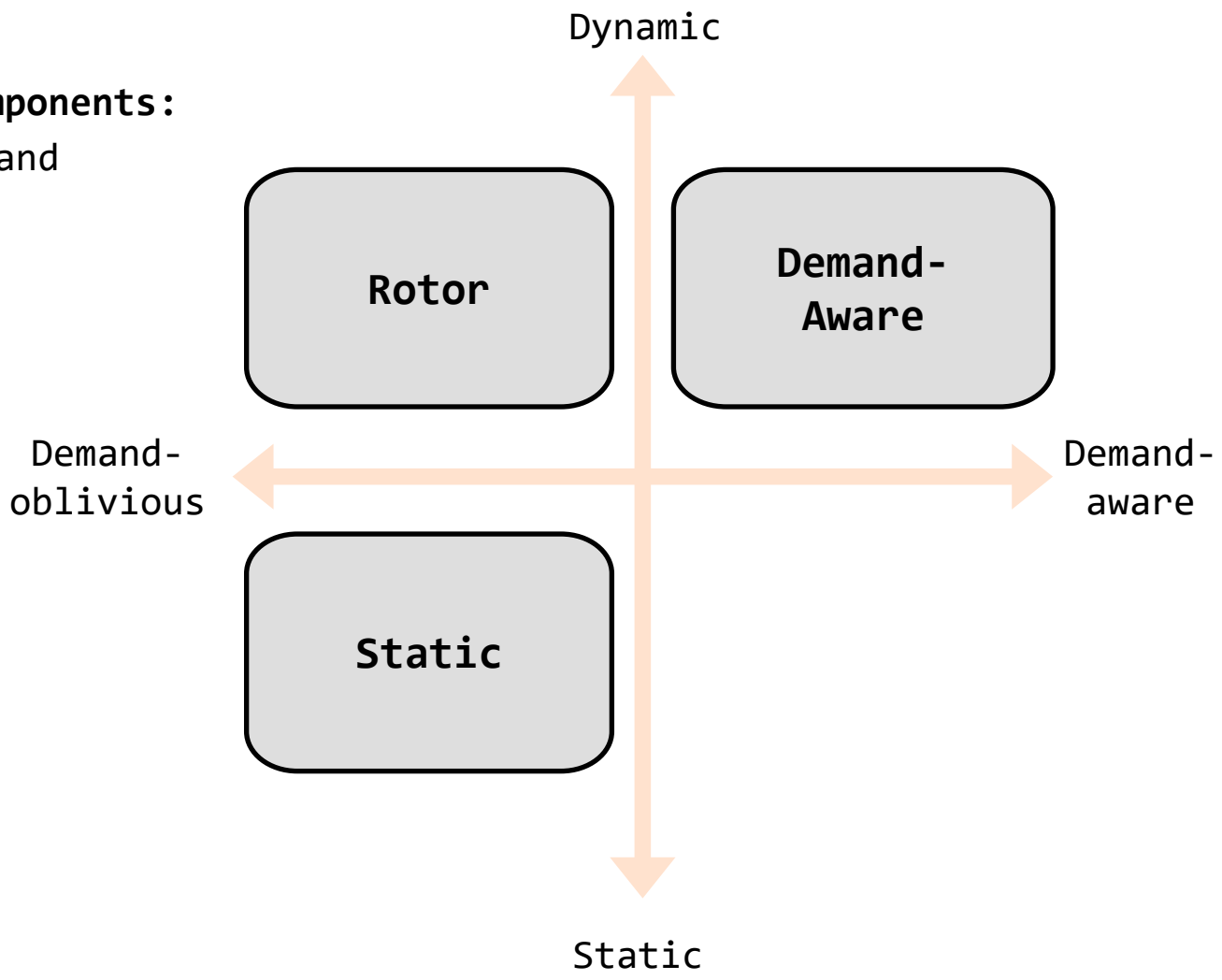
→ static vs dynamic



Opportunity: Tech Diversity

Diverse topology components:

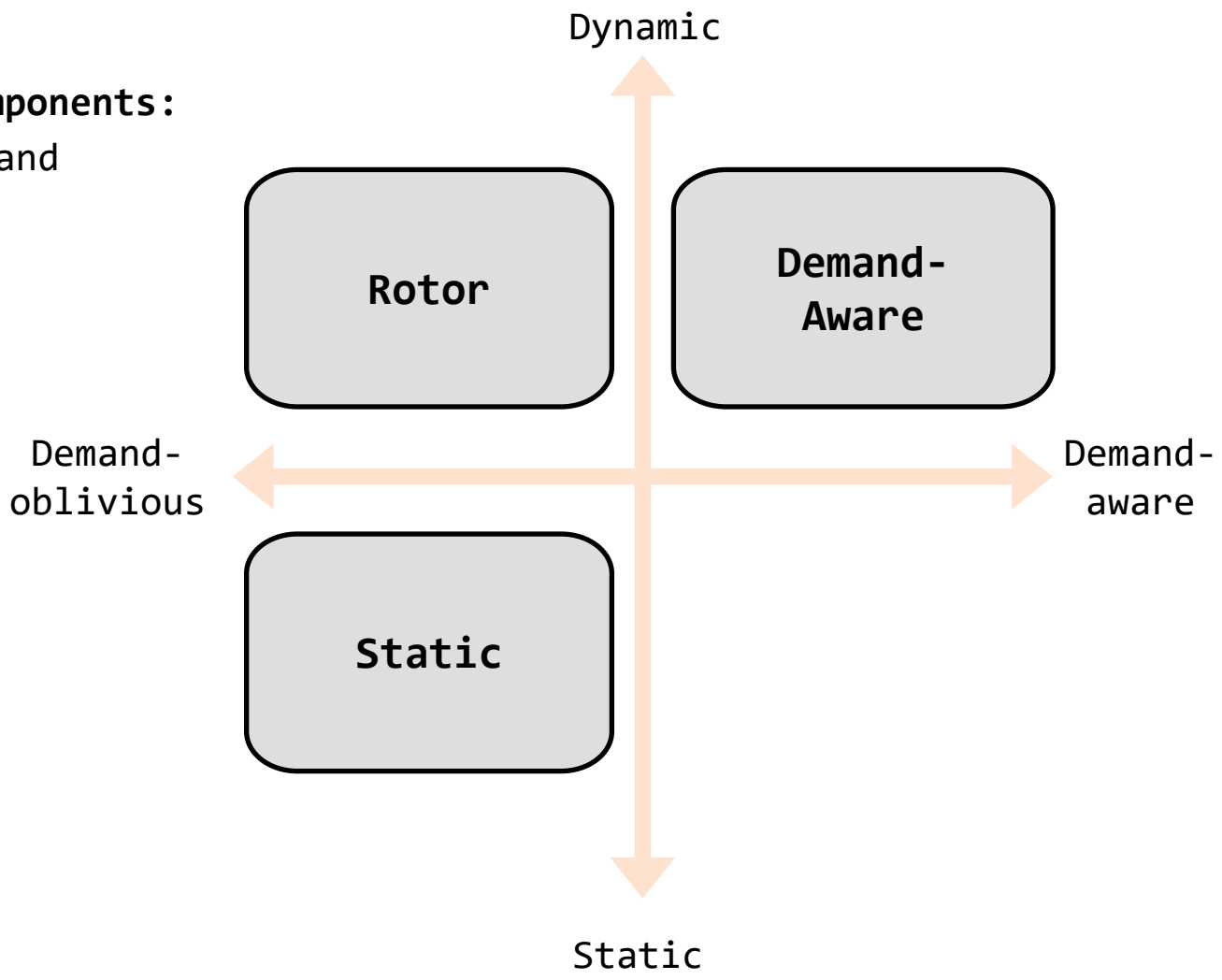
- demand-oblivious and demand-aware
- static vs dynamic



Opportunity: Tech Diversity

Diverse topology components:

- demand-oblivious and demand-aware
- static vs dynamic

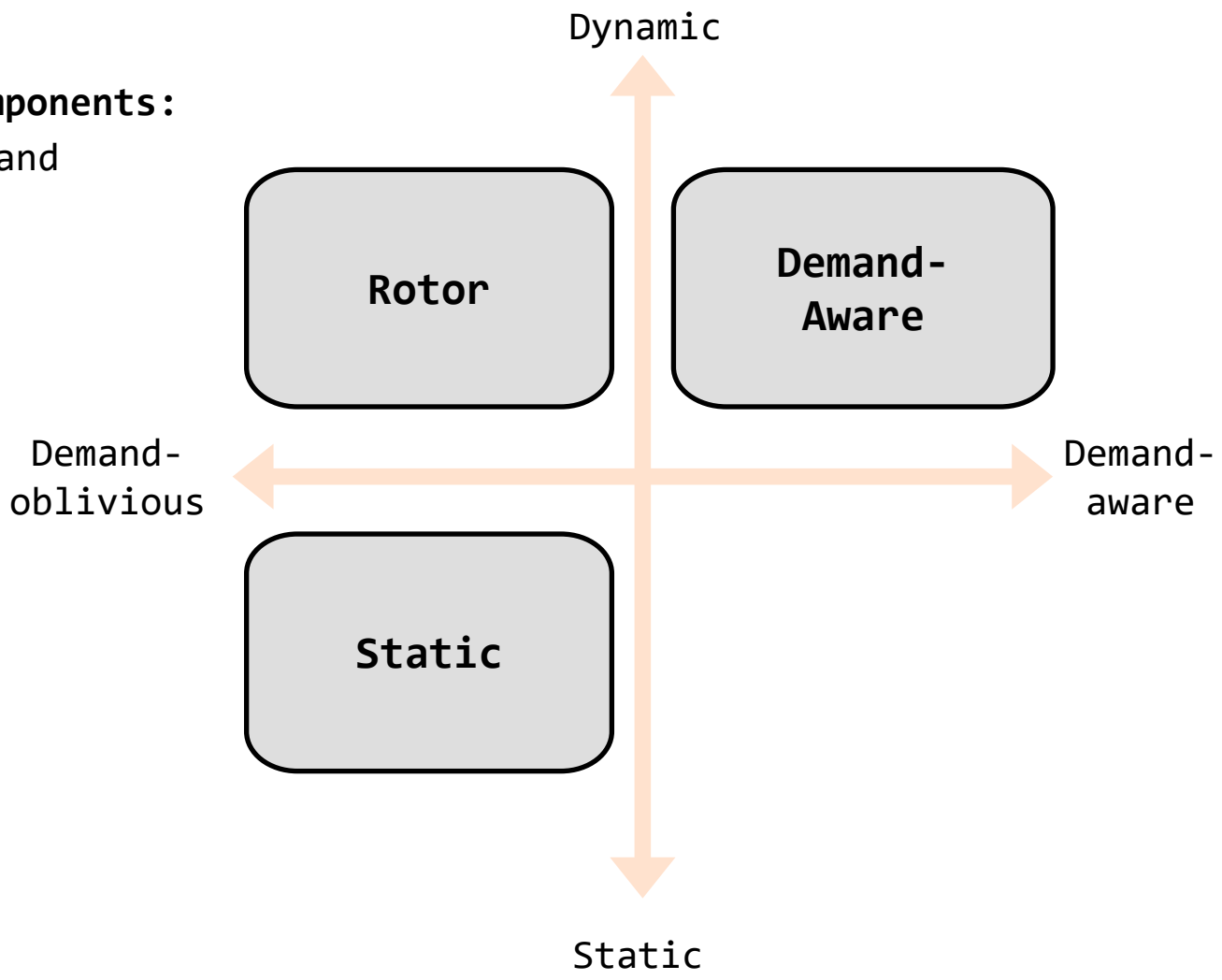


Which approach is best?

Opportunity: Tech Diversity

Diverse topology components:

- demand-oblivious and demand-aware
- static vs dynamic



Which approach is best?

As always in CS: It depends…

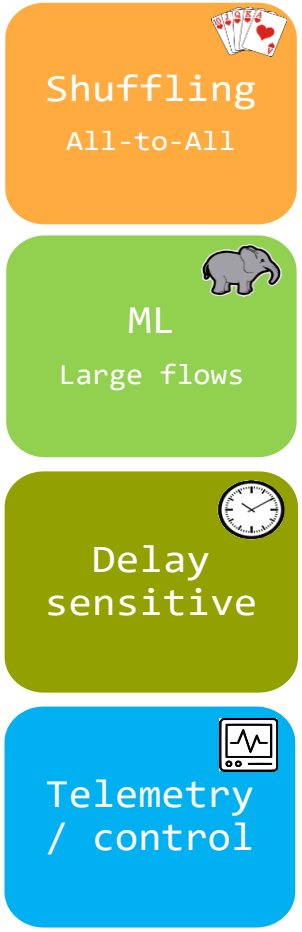
Challenge: Traffic Diversity

Diverse patterns:

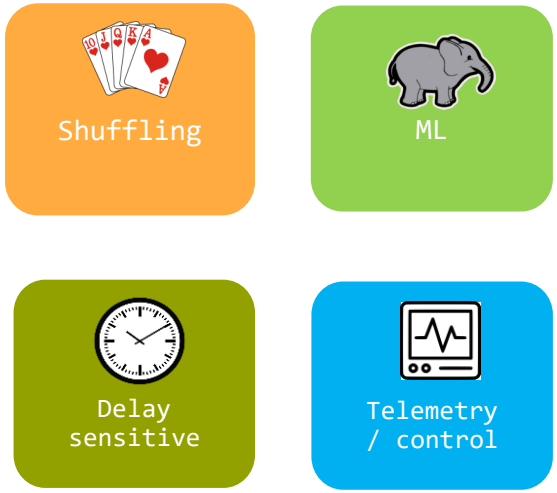
- Shuffling/Hadoop: all-to-all
- All-reduce/ML: ring or tree traffic patterns
  - Elephant flows
- Query traffic: skewed
  - Mice flows
- Control traffic: does not evolve but has non-temporal structure

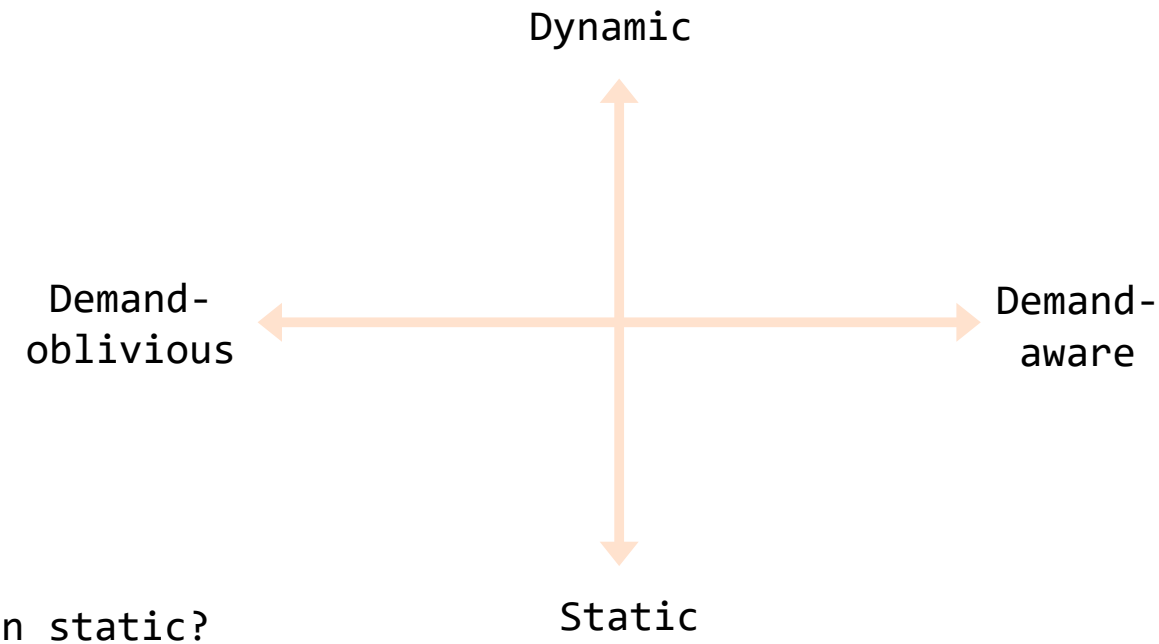
Diverse requirements:

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



Examples:
Match or Mismatch?

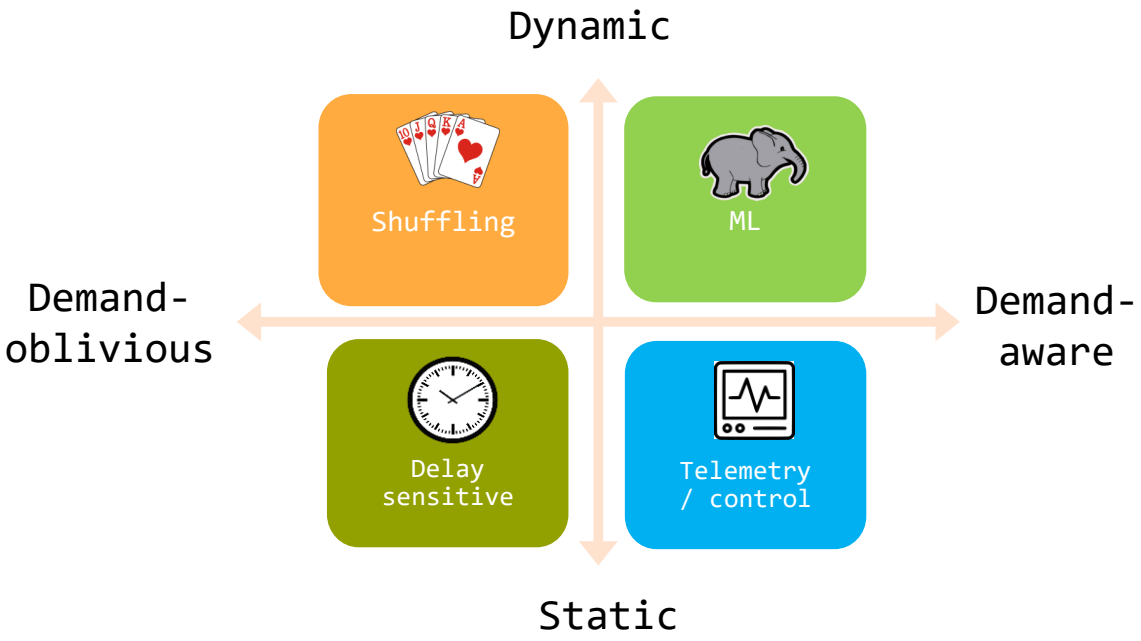




Serving elephant flows on static?

Bad idea! Bandwidth tax.

Conceptual Solution



Conceptually ideal solution:

Cerberus* serves traffic on the "best topology"!

* Griner et al., ACM SIGMETRICS 2022

Further Reading

Sigmetrics 2022

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 with their most suitable topology type: e.g., latency-sensitive flows are transmitted via a static topology, all-to-all traffic via a rotor topology, and elephant flows via a demand-aware topology. We show analytically and in simulations that CERBERUS can improve throughput significantly compared to alternative approaches and operate datacenters at higher loads while being throughput-proportional.

CCS Concepts: • **Networks** → **Network design principles**; **Network performance analysis**; **Data center networks**;

Can we optimize reconfigurable datacenters for specific ML workloads?

Collective Communications

- Distributed HPC and ML applications often perform very ***specific and predictable*** communication operations

- Known as ***Collective Communications***

- Examples: ***Nvidia's NCCL***, AMD's RCCL,....
  - Basically reincarnations of MPI
  - "CCL" stands for Collective Communication ***Library***

Collective Communications

- Distributed HPC and ML applications often perform very ***specific and predictable*** communication operations

- Known as ***Collective Communications***

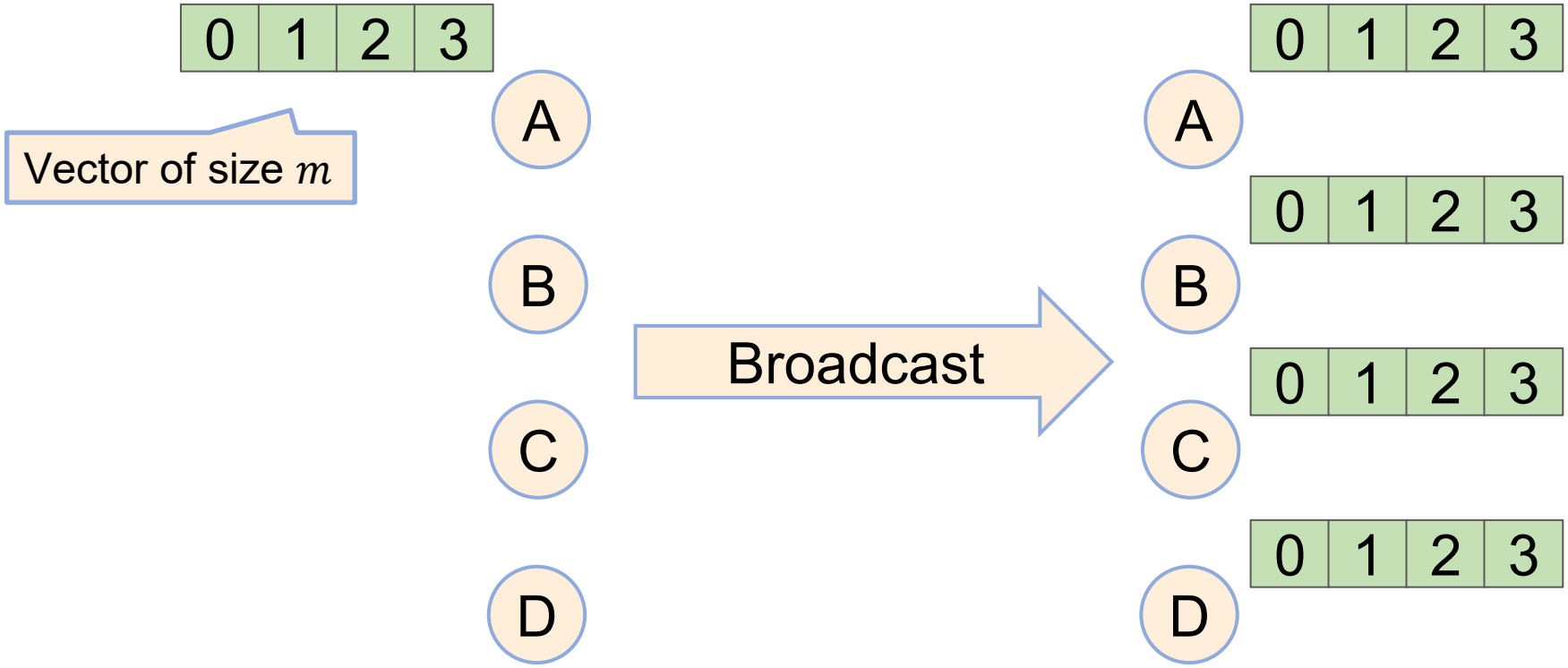
- Examples: ***Nvidia's NCCL***, AMD's RCCL,....
  - Basically reincarnations of MPI
  - "CCL" stands for Collective Communication ***Library***

Importantly:

- Predictable: simpler control plane ("***offline*** vs online")
- Collective algorithms realize primitives in **communication phases**
- Specifically: communication is ***pair-wise*** ("***matching***")

Example of a Collective Communication

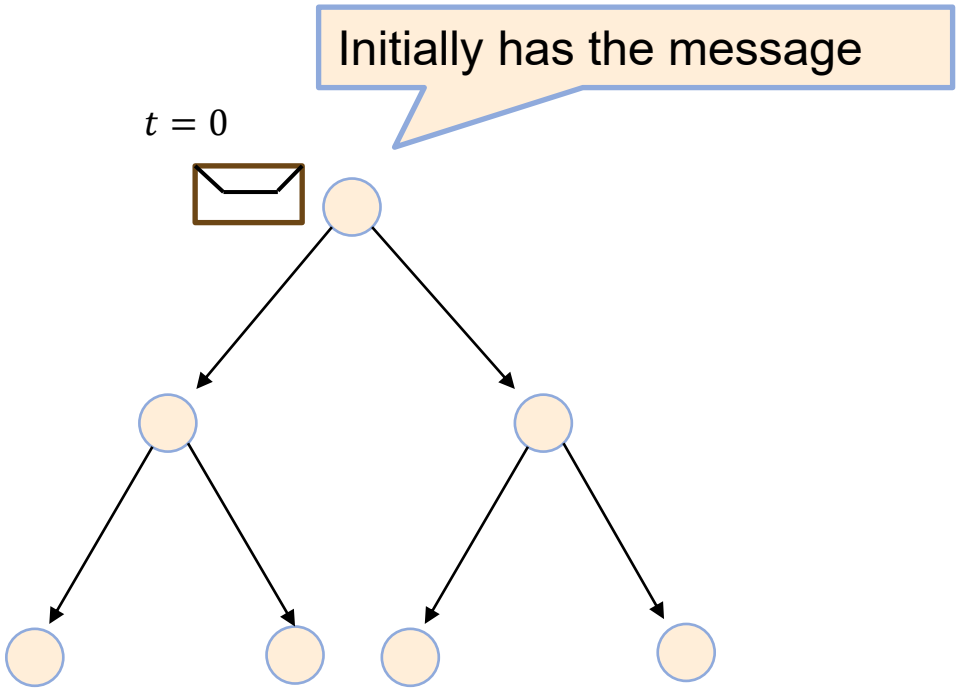
Broadcast



A single node transmits its entire data *to all other nodes*.

Example of a Collective Communication

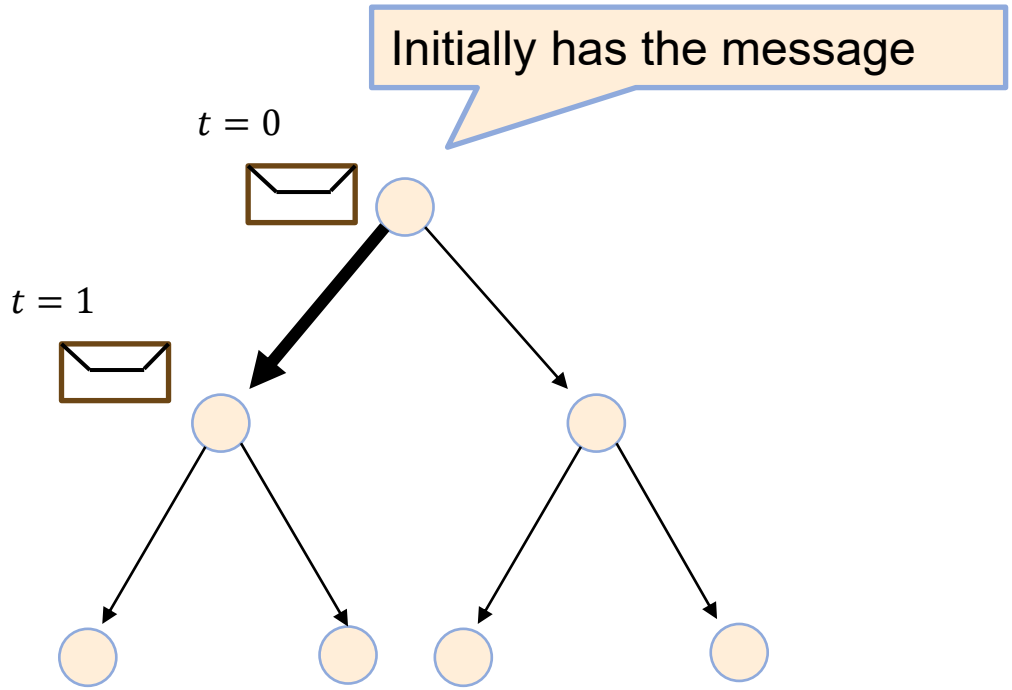
Broadcast Algorithm



Realization in *collectives algorithm*: *pair-wise* matchings!

Example of a Collective Communication

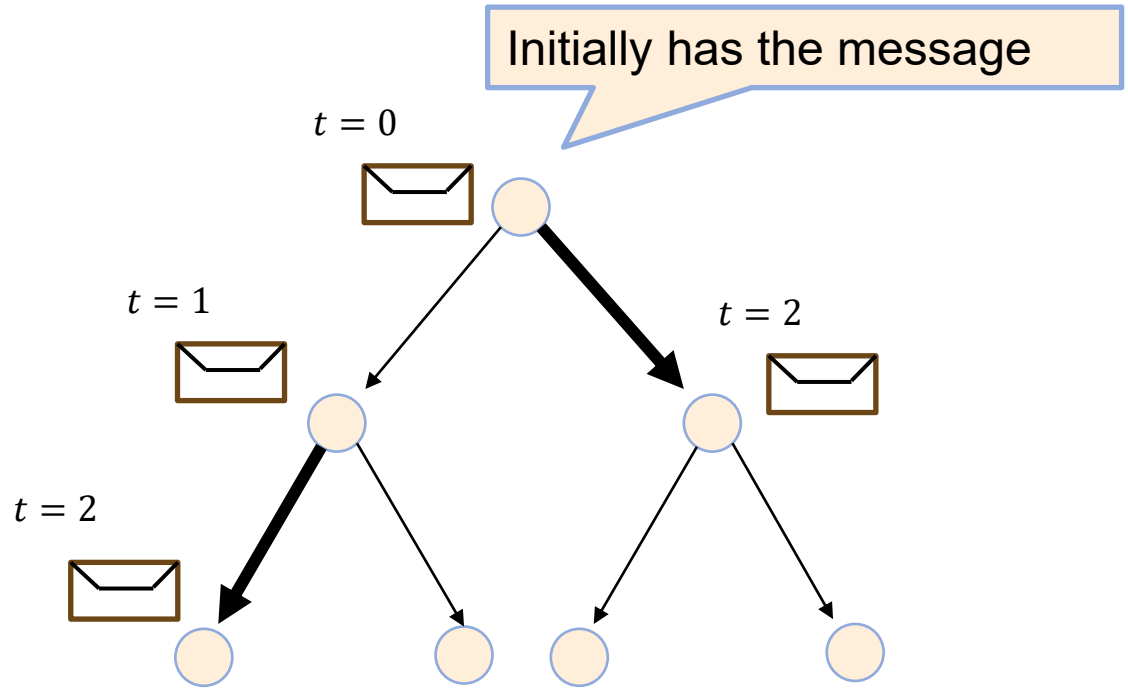
Broadcast Algorithm



Realization in *collectives algorithm*: *pair-wise* matchings!

Example of a Collective Communication

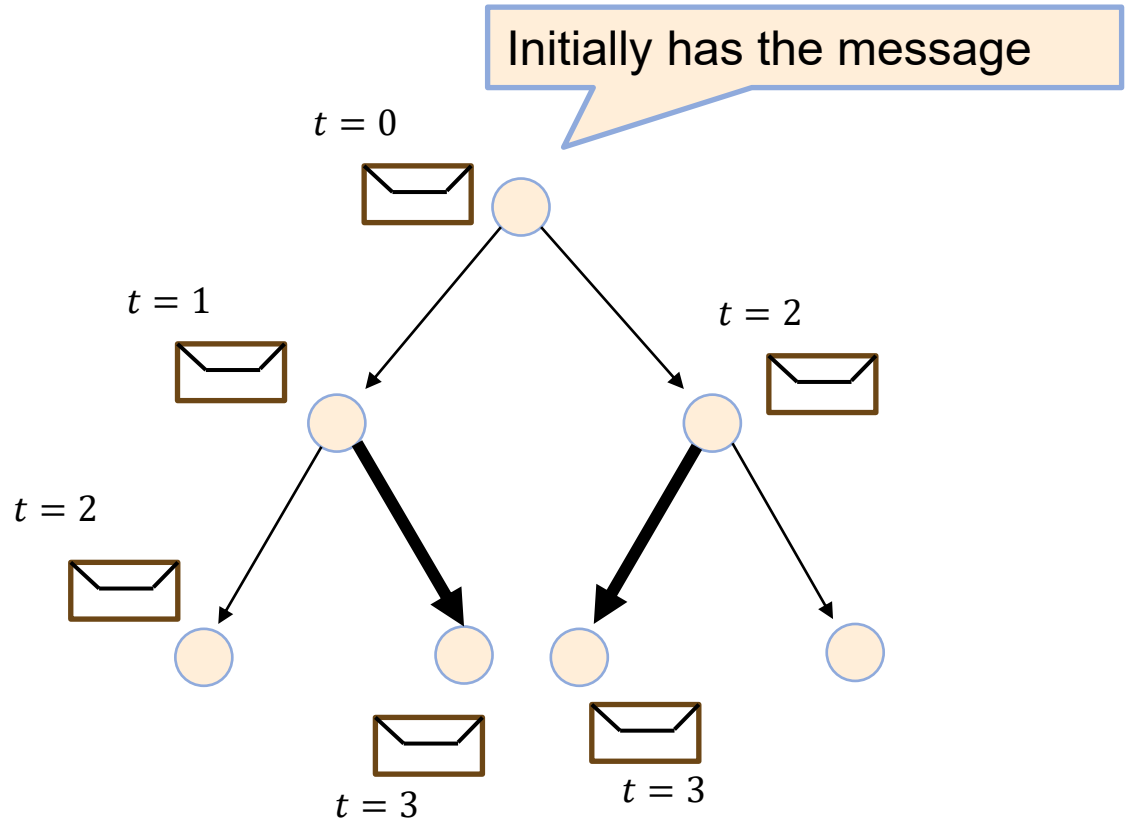
Broadcast Algorithm



Realization in *collectives algorithm*: *pair-wise* matchings!

Example of a Collective Communication

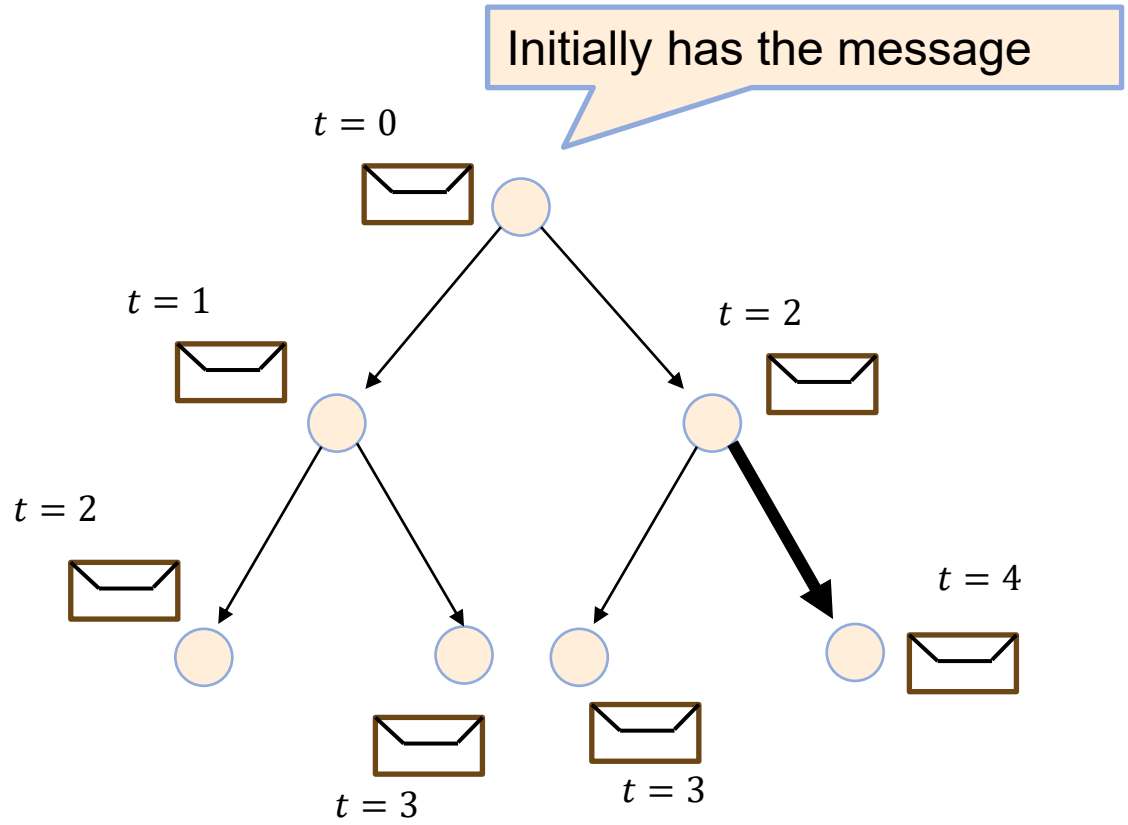
Broadcast Algorithm



Realization in *collectives algorithm*: *pair-wise* matchings!

Example of a Collective Communication

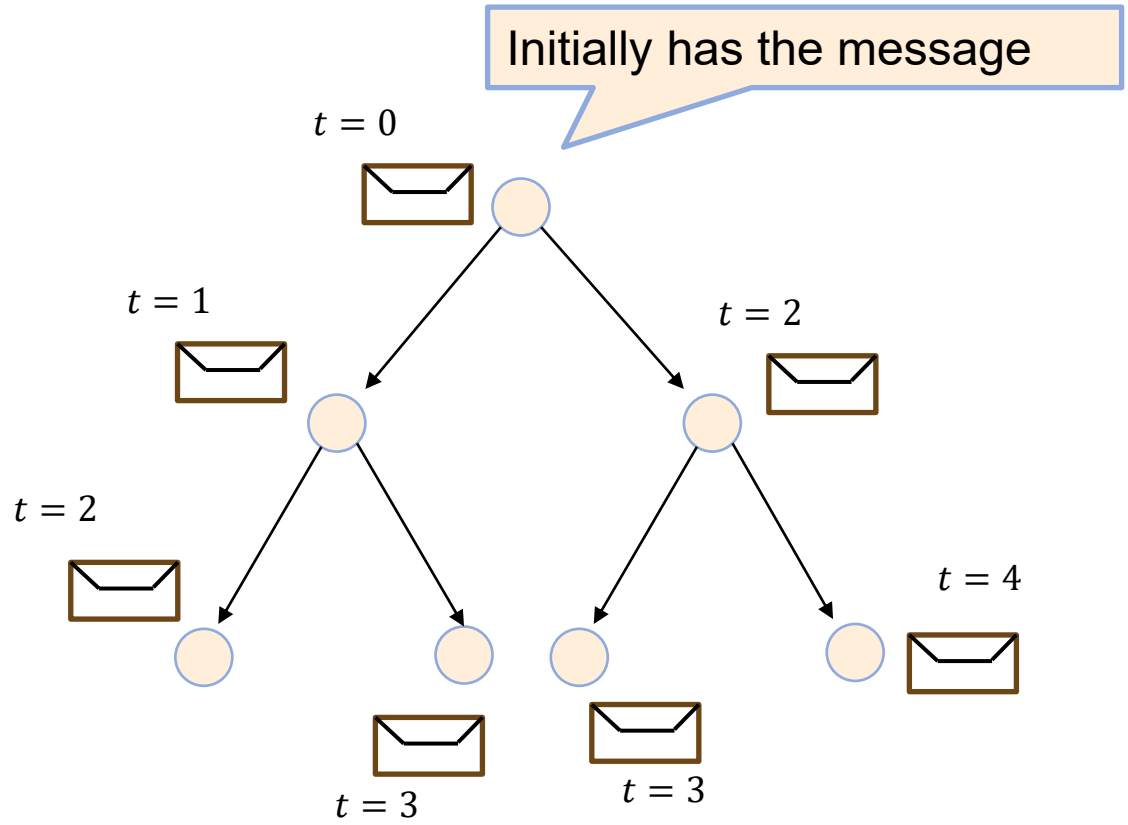
Broadcast Algorithm



Realization in *collectives algorithm*: *pair-wise* matchings!

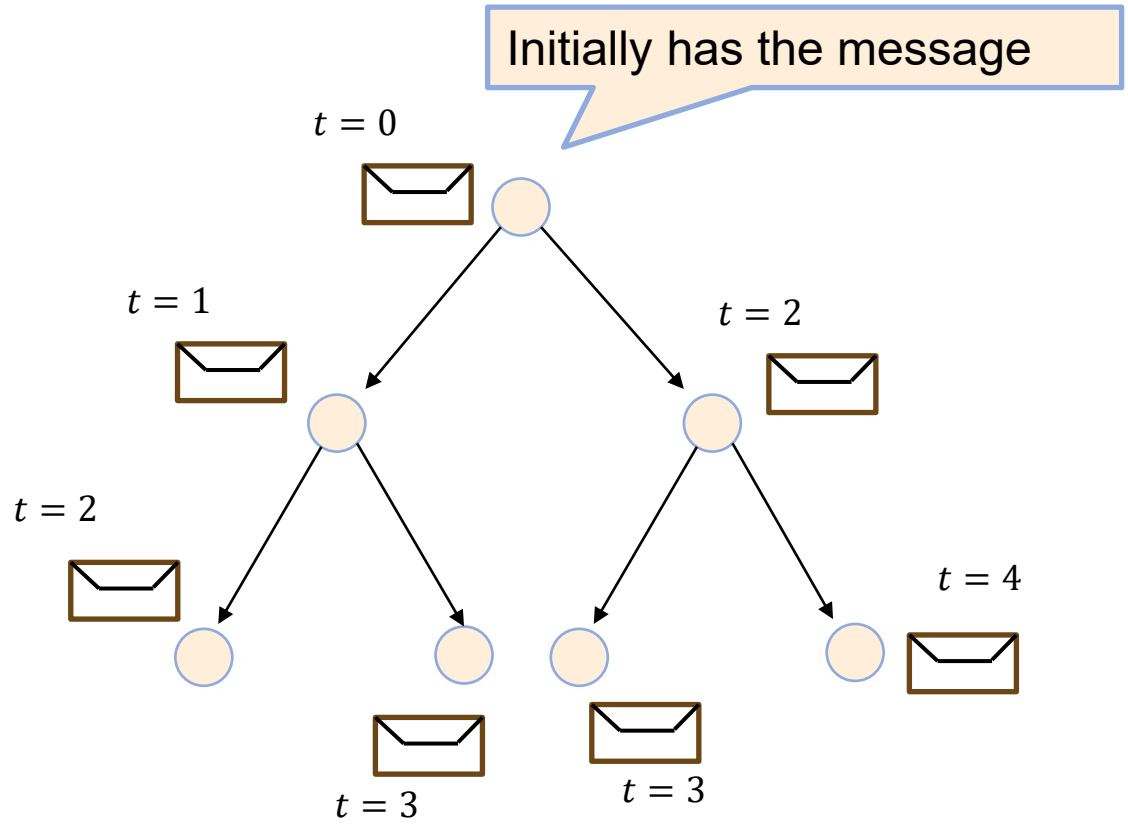
Example of a Collective Communication

Broadcast Algorithm



Example of a Collective Communication

Broadcast Algorithm



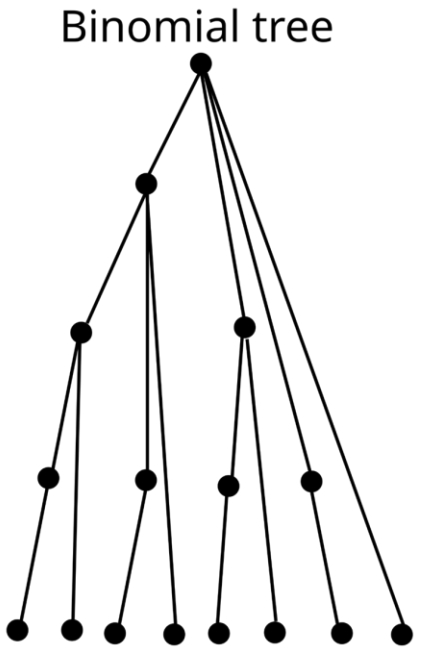
Faster solution with *binomial tree*: but always a *matching*!

Example of a Collective Communication

Broadcast Algorithm

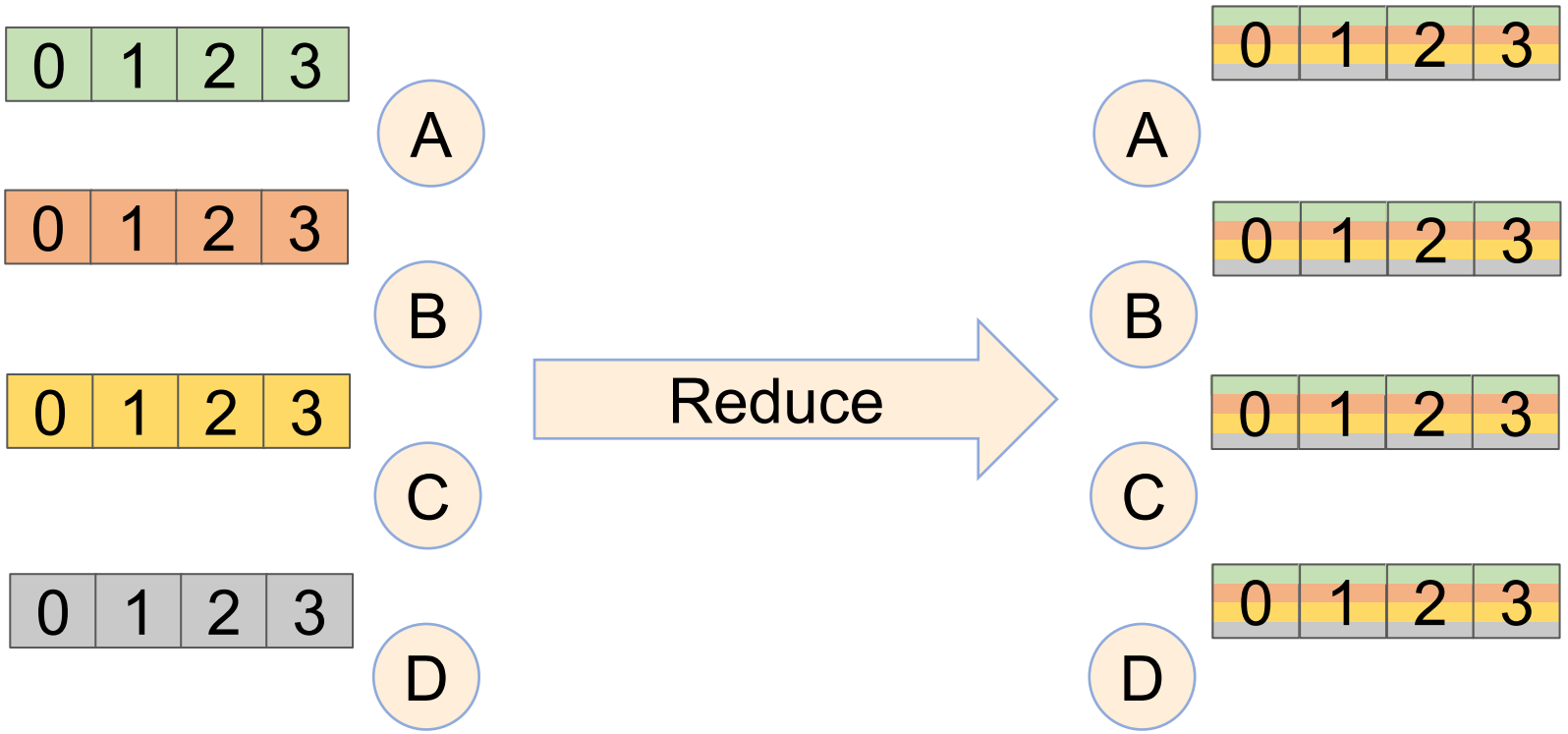
Keep helping in future too!

- There are optimizations…
- E.g., nodes continue helping, aka ***binomial tree***
- But always a ***matching***!



Example of a Collective Communication

AllReduce

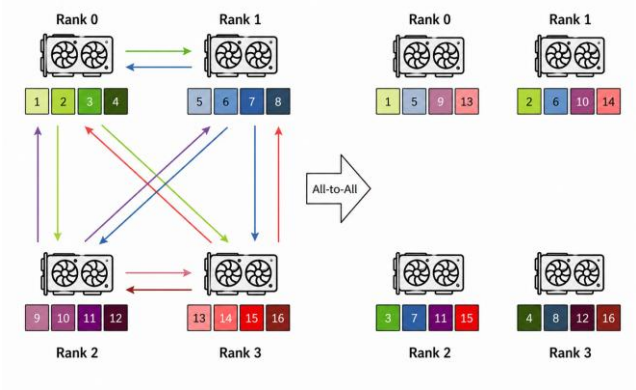


Every node transmits its data *to every other node*, and *aggregation* is performed (e.g, sum, min, max).

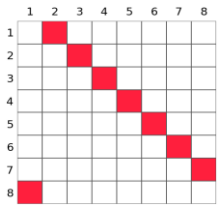
Example of a Collective Communication

All-to-All

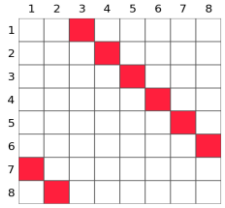
→ Each node sends a ***unique*** message to ***every*** other node



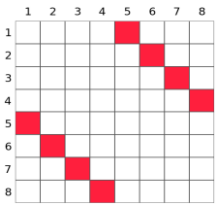
→ Algorithmic realization in phases (matchings): ***Bruck's algorithm***



K= 0

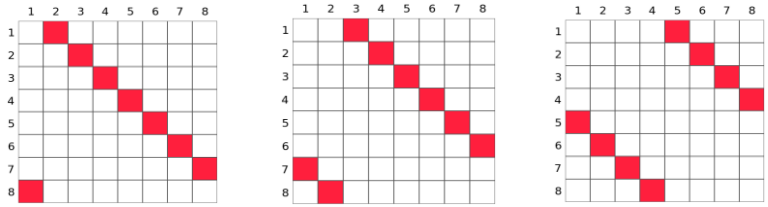


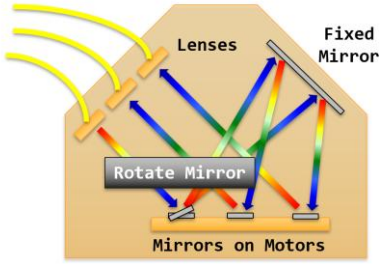
K= 1

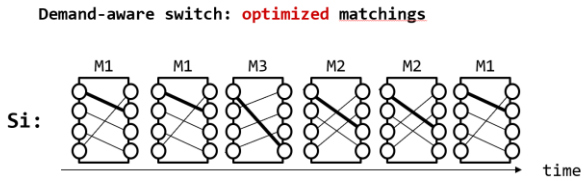


K= 2

A Perfect Match!







→ Schedule of ***dynamic matchings*** can optimally serve ***Bruck's workload***
(always direct, no multi-hop)

→ Algorithms: e.g., ***Birckhoff von Neumann*** decompositions

Further Reading

HotOptics@Sigcomm 2026

Revisiting Bruck: Phase-Efficient All-to-All Communication in Reconfigurable Networks

Anton Juerss
Weizenbaum Institute & TU Berlin
Berlin, Germany
anton.juerss@weizenbaum-institut.de

Stefan Schmid
TU Berlin & Weizenbaum Institute
Berlin, Germany
stefan.schmid@tu-berlin.de

Abstract

All-to-All communication is a key performance bottleneck for distributed machine learning (ML) and high-performance computing (HPC) workloads, where dense traffic increasingly stresses scale-up interconnects. While these ML and HPC workloads have driven unprecedented infrastructure demand, optical reconfigurable networks (ORNs) offer a promising path forward as they can reconfigure the network at runtime. By adapting the physical topology to the active workload, they improve communication cost and bandwidth utilization. However, optical reconfigurable networks introduce a fundamental trade-off for collective communication: each reconfiguration requires global synchronization, during which communication is suspended for at a non-negligible delay. Additionally, their benefit is critically contingent on whether the collective consists of structured phases that can be served by sparse and reusable topology states.

In this paper, we revisit Bruck's All-to-All implementation and demonstrate the benefits of topology optimization in which both communication pattern and reconfiguration strategy are co-designed. We present ReTri, a bidirectional All-to-All schedule for ORNs based on the Trivance algorithm. ReTri uses balanced ternary block propagation to complete All-to-All in $\lceil \log_3 n \rceil$ phases. The reconfiguration strategy induced by ReTri's pairwise bidirectional exchanges allows reconfiguration delays to be amortized across multiple phases. Preliminary simulations show that ReTri improves completion time by up to 10× over Pairwise All-to-All, even for millisecond-scale reconfiguration delays, and improves reconfig-

1 Introduction

To meet the increasing compute and memory demands of deep learning workloads, including recommendation systems and Mixture of Expert models, modern distributed systems connect thousands of accelerators in hyperscale datacenters [12, 17, 21, 28]. In these large-scale ML systems, efficient communication across accelerators is crucial for training and inference performance, since activations, embeddings, and tokens must be synchronized [17]. These synchronizations are typically realized through All-to-All collective communication, where each accelerator sends distinct data to every other accelerator [27]. Their impact on end-to-end performance is substantial, accounting for up to 55% of MoE end-to-end training time [17]. The dense communication pattern of All-to-All makes it a major performance bottleneck and increasingly stresses scale-up interconnects [15]: every accelerator exchanges distinct data, raising serious concerns about congestion and requiring high network bandwidth [17, 21].

The design of datacenter interconnect fabrics therefore plays a central role in the scalability and efficiency of large-scale ML and HPC systems [7, 12, 21]. Whereas conventional, electrically switched networks are power-intensive and may lead to performance bottlenecks, optical reconfigurable networks (ORNs) have emerged as a promising alternative. ORNs establish bidirectional high-bandwidth optical links between endpoints, introducing the enhanced capability to adjust and optimize the physical topology [3, 4, 11]. However, direct optical connectivity incurs non-negligible reconfiguration delay [7, 10, 20]. Whether reconfigura-

Challenges: Impact of dynamic networks on other layers and scalable control plane

More Challenges:

Network Layer?

⇢ *ECMP* reconvergence?! Benefits of *Valiant* routing?
⇢ How to avoid packet *reorderings*? *RDMA* network cards don't like them!
⇢ Routing in hybrid networks: segregated vs *non-segregated*?
⇢ First ideas: *local* routing! Techniques from dynamic P2P systems?

Duo: A High-Throughput Reconfigurable Datacenter Network Using Local Routing and Control

JOHANNES ZERWAS, TUM School of Computation, Information and Technology, Technical University of Munich, Germany
CSABA GYÖRGYI, University of Vienna and ELTE Eötvös Loránd University, Austria and Hungary
ANDREAS BLENK, Siemens AG, Germany
STEFAN SCHMID, TU Berlin & Fraunhofer SIT, Germany
CHEN AVIN, Ben-Gurion University, Israel

The performance of many cloud-based applications critically depends on the capacity of the underlying datacenter network. A particularly innovative approach to improve the throughput in datacenters is enabled by emerging optical technologies, which allow to dynamically adjust the physical network topology, both in an oblivious or demand-aware manner. However, such topology engineering, i.e., the operation and control of dynamic datacenter networks, is considered complex and currently comes with restrictions and overheads.

We present Duo, a novel demand-aware reconfigurable rack-to-rack datacenter network design realized with a simple and efficient control plane. Duo is based on the well-known de Bruijn topology (implemented using a small number of optical circuit switches) and the key observation that this topology can be enhanced using dynamic ("opportunistic") links between its nodes.

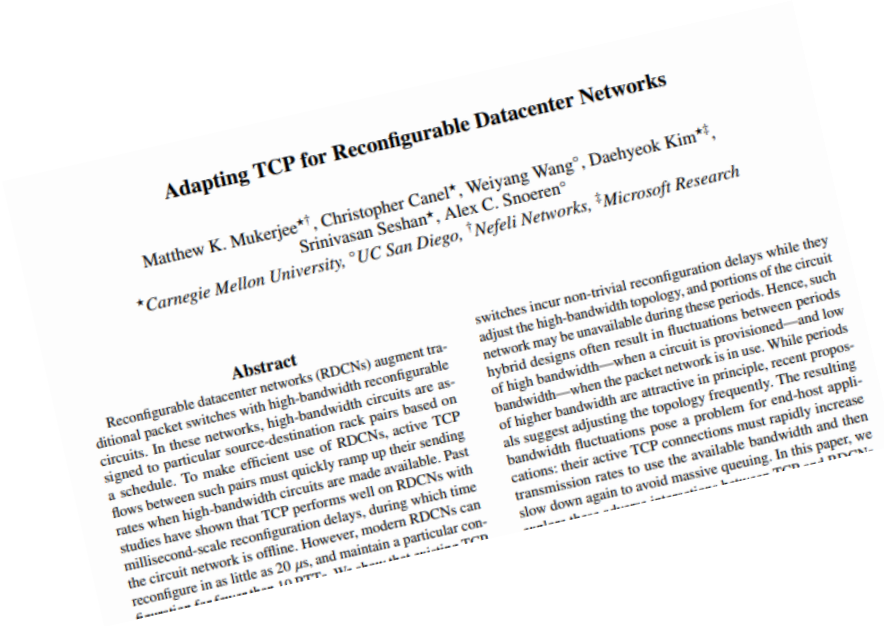
In contrast to previous systems, Duo has several desired features: i) It makes effective use of the network

More Challenges:

Congestion Control?

⇢ First ideas for quickly reacting TCP: ***ReTCP***, ***PowerTCP***, …

⇢ Or better completely different approach? Even centralized?!



Adapting TCP for Reconfigurable Datacenter Networks

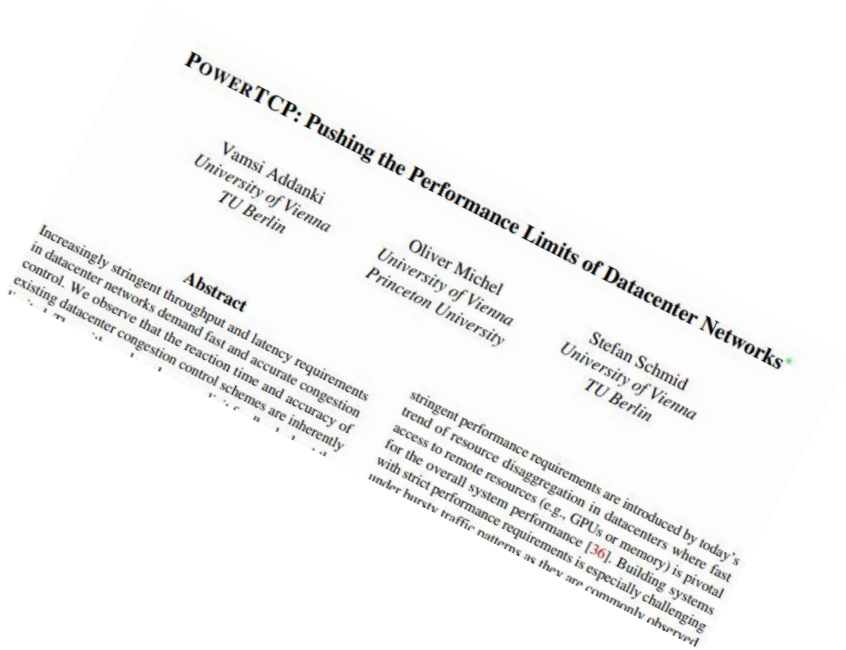
Matthew K. Mukerjee*†, Christopher Canel*, Weiyang Wang°, Daehyeok Kim*‡, Srinivasan Seshan*, Alex C. Snoeren°

*Carnegie Mellon University, °UC San Diego, †Nefeli Networks, ‡Microsoft Research

Abstract

Reconfigurable datacenter networks (RDCNs) augment traditional packet switches with high-bandwidth reconfigurable circuits. In these networks, high-bandwidth circuits are assigned to particular source-destination rack pairs based on a schedule. To make efficient use of RDCNs, active TCP flows between such pairs must quickly ramp up their sending rates when high-bandwidth circuits are made available. Past studies have shown that TCP performs well on RDCNs with millisecond-scale reconfiguration delays, during which time the circuit network is offline. However, modern RDCNs can reconfigure in as little as 20 μs, and maintain a particular con-

switches incur non-trivial reconfiguration delays while they adjust the high-bandwidth topology, and portions of the circuit network may be unavailable during these periods. Hence, such hybrid designs often result in fluctuations between periods of high bandwidth—when a circuit is provisioned—and low bandwidth—when the packet network is in use. While periods of higher bandwidth are attractive in principle, recent proposals suggest adjusting the topology frequently. The resulting bandwidth fluctuations pose a problem for end-host applications: their active TCP connections must rapidly increase transmission rates to use the available bandwidth and then slow down again to avoid massive queuing. In this paper, we



POWERTCP: Pushing the Performance Limits of Datacenter Networks*

Vamsi Addanki
University of Vienna
TU Berlin

Oliver Michel
University of Vienna
Princeton University

Stefan Schmid
University of Vienna
TU Berlin

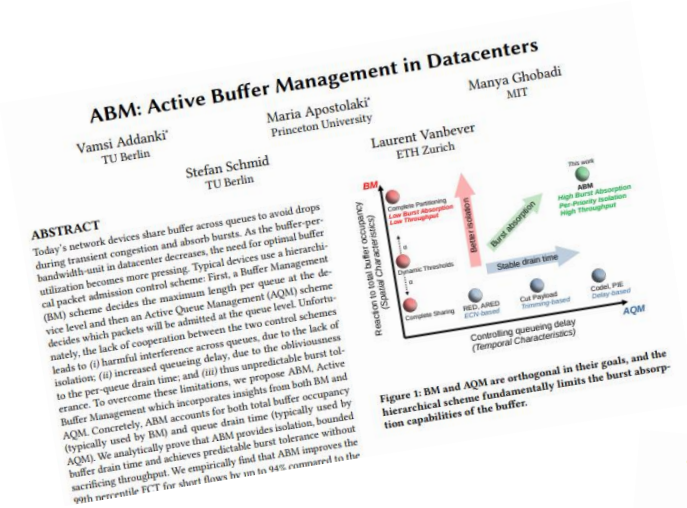
Abstract

Increasingly stringent throughput and latency requirements in datacenter networks demand fast and accurate congestion control. We observe that the reaction time and accuracy of existing datacenter congestion control schemes are inherently

stringent performance requirements are introduced by today's trend of resource disaggregation in datacenters where fast access to remote resources (e.g., GPUs or memory) is pivotal for the overall system performance [36]. Building systems with strict performance requirements is especially challenging under bursty traffic patterns as they are commonly observed

More Challenges:

Buffering?



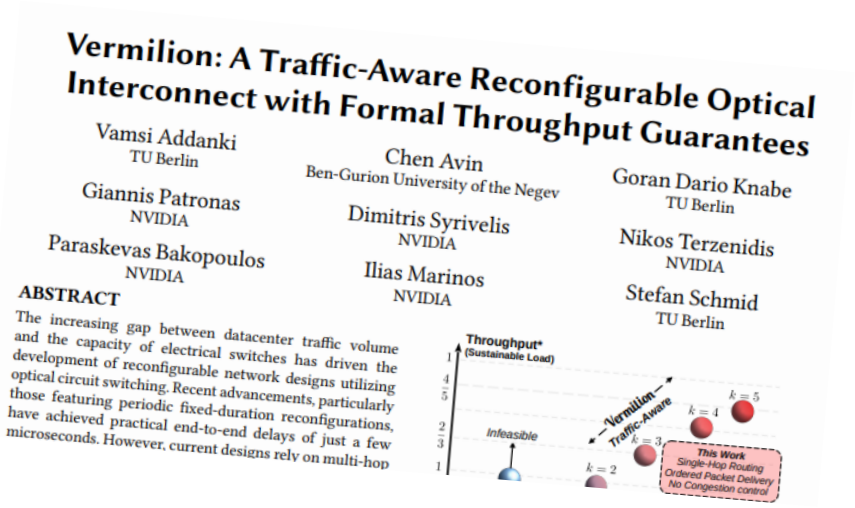
ABM: Active Buffer Management in Datacenters

Vamsi Addanki* TU Berlin
Maria Apostolaki* Princeton University
Manya Ghobadi MIT
Stefan Schmid TU Berlin
Laurent Vanbever ETH Zurich

ABSTRACT

Today's network devices share buffer across queues to avoid drops during transient congestion and absorb bursts. As the buffer-per-bandwidth-unit in datacenter decreases, the need for optimal buffer utilization becomes more pressing. Typical devices use a hierarchical packet admission control scheme: First, a Buffer Management (BM) scheme decides the maximum length per queue at the device level and then an Active Queue Management (AQM) scheme decides which packets will be admitted at the queue level. Unfortunately, the lack of cooperation between the two control schemes leads to *(i)* harmful interference across queues, due to the obliviousness isolation, *(ii)* increased queueing delay, due to the lack of to the per-queue drain time; and *(iii)* thus unpredictable burst tolerance. To overcome these limitations, we propose ABM, Active Buffer Management which incorporates insights from both BM and AQM. Concretely, ABM accounts for both total buffer occupancy (typically used by BM) and queue drain time (typically used by AQM). We analytically prove that ABM provides isolation, bounded buffer drain time and achieves predictable burst tolerance without sacrificing throughput. We empirically find that ABM improves the 99th percentile FCT for short flows by up to 94% compared to the

Figure 1: BM and AQM are orthogonal in their goals, and the hierarchical scheme fundamentally limits the burst absorption capabilities of the buffer.



Vermilion: A Traffic-Aware Reconfigurable Optical Interconnect with Formal Throughput Guarantees

Vamsi Addanki TU Berlin
Chen Avin Ben-Gurion University of the Negev
Goran Dario Knabe TU Berlin
Giannis Patronas NVIDIA
Dimitris Syrivelis NVIDIA
Nikos Terzenidis NVIDIA
Paraskevas Bakopoulos NVIDIA
Ilias Marinos NVIDIA
Stefan Schmid TU Berlin

ABSTRACT

The increasing gap between datacenter traffic volume and the capacity of electrical switches has driven the development of reconfigurable network designs utilizing optical circuit switching. Recent advancements, particularly those featuring periodic fixed-duration reconfigurations, have achieved practical end-to-end delays of just a few microseconds. However, current designs rely on multi-hop

Case Study:

PowerTCP

Existing congestion control algorithms based on either

⇢ State (“voltage”) like BDP, queue length, loss, e.g.:
  ⇢ DCTCP: uses ECN/loss
  ⇢ Swift: RTT
  ⇢ HPCC: inflight packets

⇢ Gradient (“current”) like reaction to queue length change
  ⇢ Timely: RTT-gradient based

Case Study:

PowerTCP

Existing congestion control algorithms based on either

⇢ State (“voltage”) like BDP, queue length, loss, e.g.:
  ⇢ DCTCP: uses ECN/loss
  ⇢ Swift: RTT
  ⇢ HPCC: inflight packets

⇢ Gradient (“current”) like reaction to queue length change
  ⇢ Timely: RTT-gradient based

☺ Can achieve near-zero queue equilibrium

☹ Slow reaction

Case Study:

PowerTCP

Existing congestion control algorithms based on either

⇢ State (“voltage”) like BDP, queue length, loss, e.g.:
  ⇢ DCTCP: uses ECN/loss
  ⇢ Swift: RTT
  ⇢ HPCC: inflight packets

⇢ Gradient (“current”) like reaction to queue length change
  ⇢ Timely: RTT-gradient based

☺ Fast reaction

☹ No equilibrium

Case Study:

PowerTCP

Existing congestion control algorithms based on either

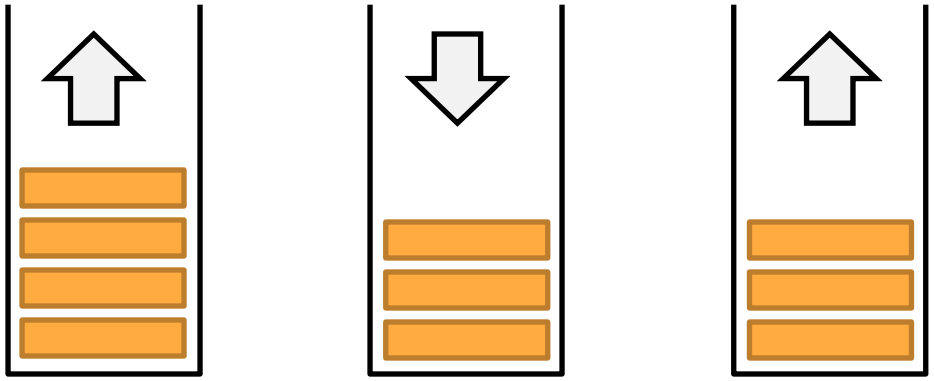
⇢ State (“voltage”) like BDP, queue length, loss, e.g.:
  ⇢ DCTCP: uses ECN/loss
  ⇢ Swift: RTT
  ⇢ HPCC: inflight packets

⇢ Gradient (“current”) like reaction to queue length change
  ⇢ Timely: RTT-gradient based

Limitation: using only one of the two may miss useful information for fine-grained adaptions!

Limitation of SOTA

⇢ Consider a queue which may be in three different states:



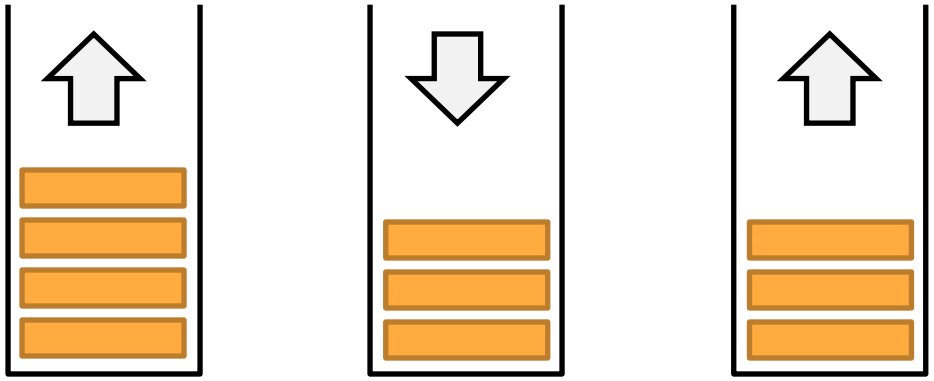
1
growing

2
shrinking

3
growing

Limitation of SOTA

⇢ Consider a queue which may be in three different states:



1
growing

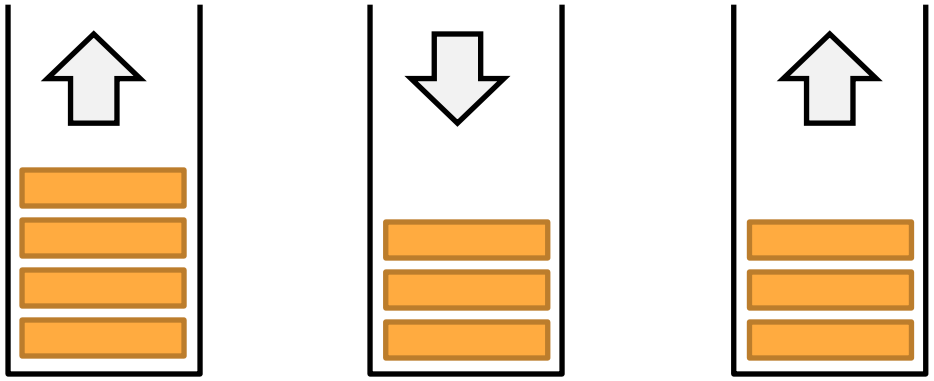
2
shrinking

3
growing

2 and 3: impossible to
distinguish for voltage-based CCA

Limitation of SOTA

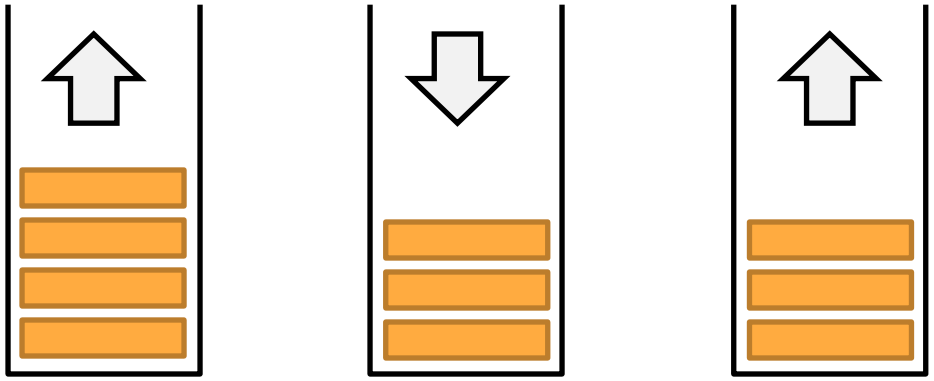
⇢ Consider a queue which may be in three different states:



1 and 3: impossible to
distinguish for current-based CC

Limitation of SOTA

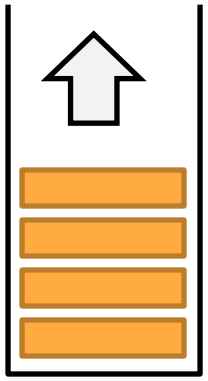
⇢ Consider a queue which may be in three different states:



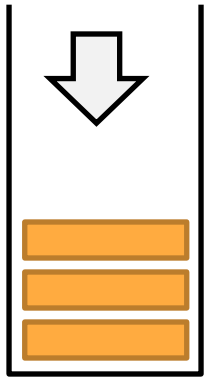
We need both: Power (Voltage x Current)

Limitation of SOTA

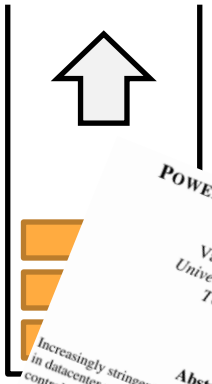
⇢ Consider a queue which may be in three different states:



1
growing



2
shrinking



POWERTCP: Pushing the Performance Limits of Datacenter Networks*

Vamsi Addanki
University of Vienna
TU Berlin

Oliver Michel
University of Vienna
Princeton University

Stefan Schmid
University of Vienna
TU Berlin

Abstract

Increasingly stringent throughput and latency requirements in datacenter networks demand fast and accurate congestion control. We observe that the reaction time and accuracy of existing datacenter congestion control schemes are inherently limited. They either rely only on explicit feedback about the network state (e.g., queue lengths in DCTCP) or only on variations of state (e.g., RTT gradient in TIMELY). To overcome these limitations, we propose a novel congestion control algorithm, PowerTCP, which achieves much more fine-grained congestion control by adapting to the bandwidth-window product (henceforth called power). PowerTCP leverages in-band network telemetry to react to changes in the network instantaneously without loss of throughput and while keeping queues short. Due to its fast reaction time, our algorithm is particularly well-suited for dynamic network environments and bursty traffic patterns. We show analytically and empirically that PowerTCP can significantly outperform the state-of-the-art in both traditional datacenter topologies and emerging stringent performance requirements are introduced by today's trend of resource disaggregation in datacenters where fast access to remote resources (e.g., GPUs or memory) is pivotal for the overall system performance [36]. Building systems with strict performance requirements is especially challenging under bursty traffic patterns as they are commonly observed in datacenter networks [12, 16, 47, 53, 55].

These requirements introduce the need for fast and accurate network resource management algorithms that optimally utilize the available bandwidth while minimizing packet latencies and flow completion times. Congestion control (CC) plays an important role in this context being "a key enabler (or limiter) of system performance in the datacenter" [34]. In fact, fast reacting congestion control is not only essential to efficiently adapt to bursty traffic [29, 48], but is also becoming increasingly important in the context of emerging reconfigurable datacenter networks (RDCNs) [13, 14, 20, 33, 38, 39, 50]. In these networks, a congestion control algorithm must be able to quickly ramp up its sending rate when high-bandwidth

We need both: Power (Voltage x Current)

Inspired: POWERTCP

A control-theoretic approach

More benefits of optical & reconfigurable switching

So far: focus on throughput performance.

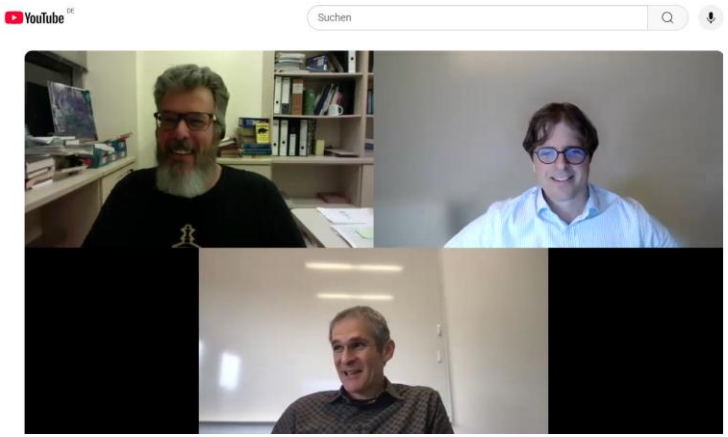
Benefit 1:

Evolving Datacenters

⇢ Reconfigurable datacenter networks naturally support *heterogeneous* network elements

⇢ And therefore also *incremental* hardware upgrades

See interview with Amin Vahdat, Google in CACM: https://www.youtube.com/watch?v=IxcV1gu8ETA

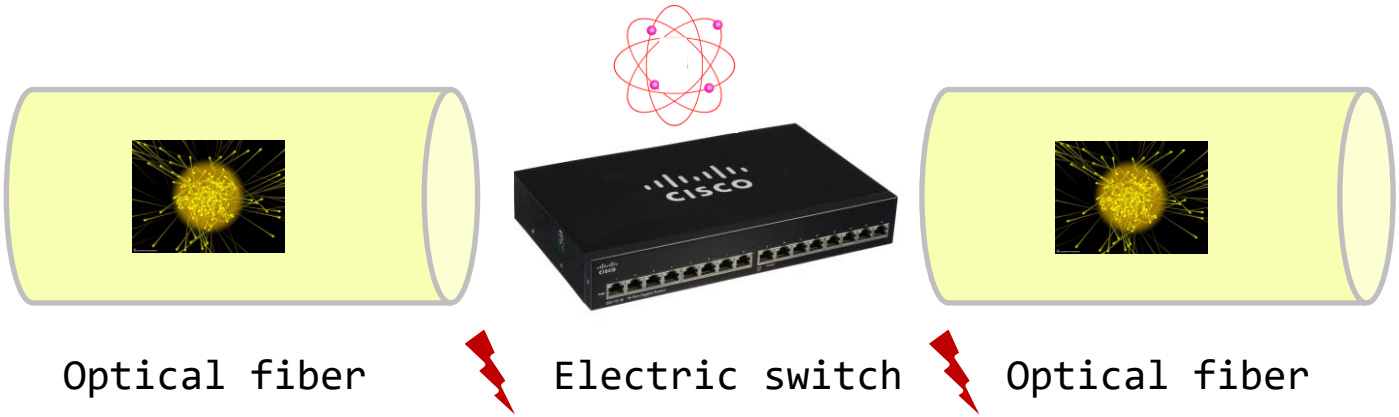


Benefit 2:

Energy and Latency

⇢ No need to *convert* photons in fiber to electrons in switch (and back)

⇢ Can safe *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)

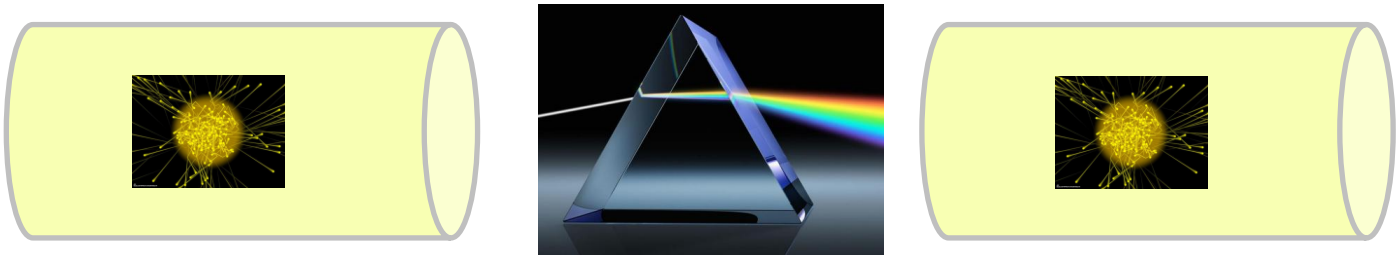


Benefit 2:

Energy and Latency

⇢ No need to *convert* photons in fiber to electrons in switch (and back)

⇢ Can safe *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)



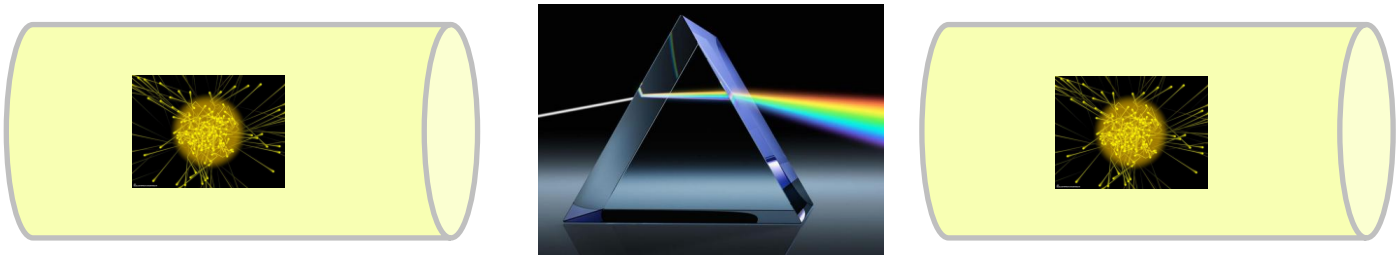
Optical fiber —— Optical switch —— Optical fiber

Benefit 2:

Energy and Latency

⇢ No need to *convert* photons in fiber to electrons in switch (and back)

⇢ Can safe *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)



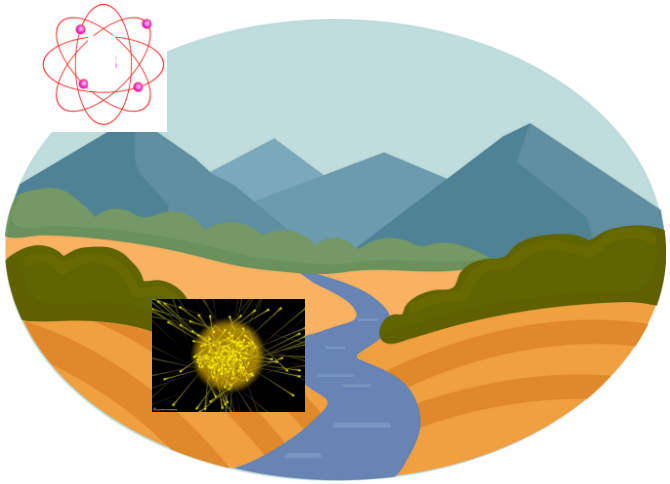
Optical fiber —— Optical switch —— Optical fiber

⇢ Interesting for emerging *distributed datacenters*!

Benefit 3:

Resilience

Floodings in South Germany destroyed much electrical network infrastructure



Solution: deploy optical infrastructure (in valleys) and electrical *on hills* where safe?

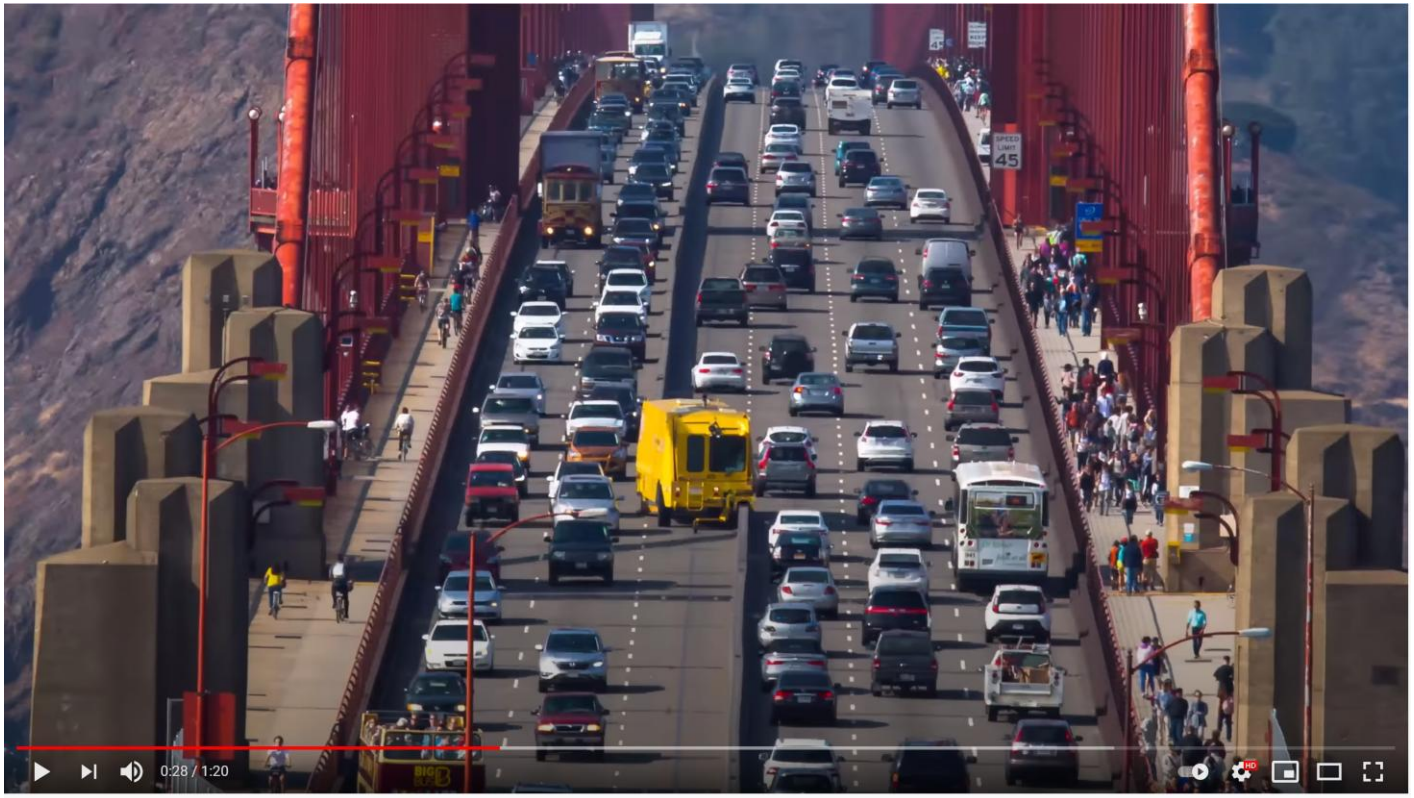
Conclusion: Tip of Iceberg!

⇢ Opportunity: *structure* in demand and *reconfigurable* networks
- ⇀ It's about matchings: finding heavy matchings that can be offloaded to optical network, statically and dynamically

⇢ Many research opportunities
- ⇀ Optimal static networks for *dense demands*
- ⇀ Throughput: tight bounds and NP-hardness
- ⇀ *Scheduling* dynamic networks (offline and online!) and beyond *BvN*
- ⇀ *Routing* and congestion control
- ⇀ *Hybrid* datacenters
- ⇀ *Scalable control* plane
- ⇀ *Application-specific* self-adjusting networks?

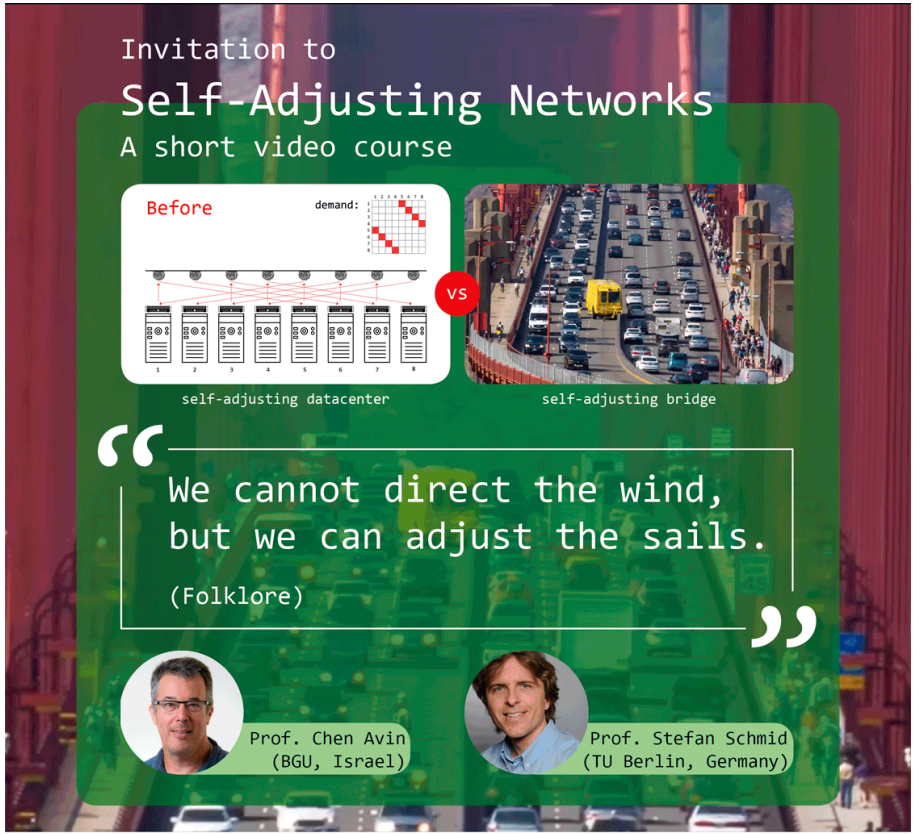
⇢ Many more *opportunities* for optical networks

Thank you! Questions?



Slides available here:

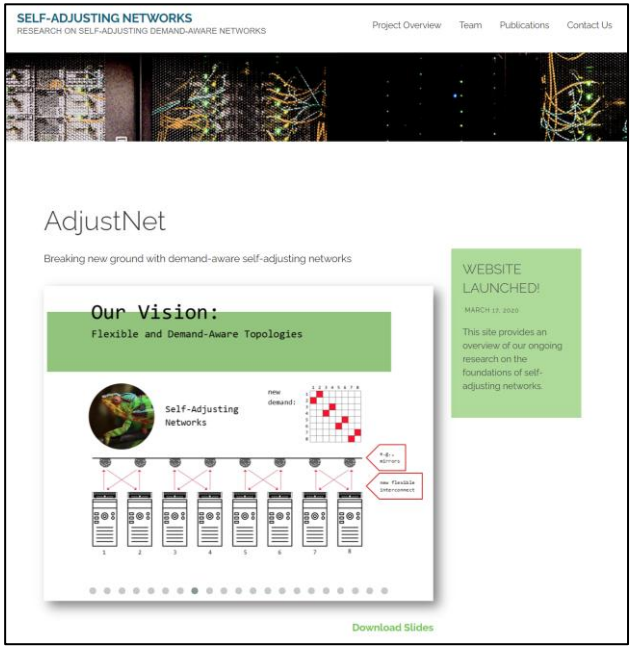
Online Video Course



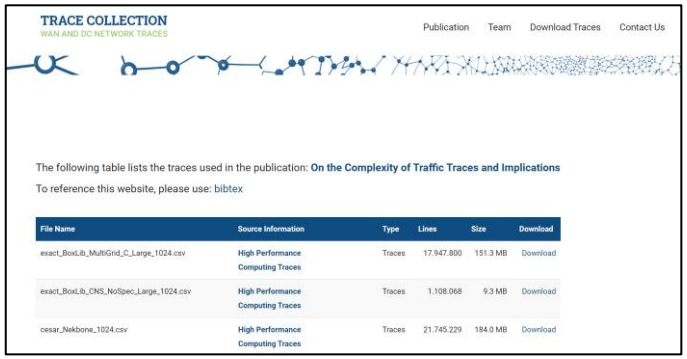
YouTube Interview & CACM

Revolutionizing Datacenter Networks via Reconfigurable Topologies
Chen Avin and Stefan Schmid.
Communications of the ACM (**CACM**), 2025.
Watch here: https://www.youtube.com/@self-adjusting-networks-course

Websites



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



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

June'25 CACM Article

Revolutionizing Datacenter Networks via Reconfigurable Topologies

CHEN AVIN, is a Professor at Ben-Gurion University of the Negev, Beersheva, Israel

STEFAN SCHMID, is a Professor at TU Berlin, Berlin, Germany

With the popularity of cloud computing and data-intensive applications such as machine learning, datacenter networks have become a critical infrastructure for our digital society. Given the explosive growth of datacenter traffic and the slowdown of Moore's law, significant efforts have been made to improve datacenter network performance over the last decade. A particularly innovative solution is reconfigurable datacenter networks (RDCNs): datacenter networks whose topologies dynamically change over time, in either a demand-oblivious or a demand-aware manner. Such dynamic topologies are enabled by recent optical switching technologies and stand in stark contrast to state-of-the-art datacenter network topologies, which are fixed and oblivious to the actual traffic demand. In particular, reconfigurable demand-aware and "self-adjusting" datacenter networks are motivated empirically by the significant spatial and temporal structures observed in datacenter communication traffic. This paper presents an overview of reconfigurable datacenter networks. In particular, we discuss the motivation for such reconfigurable architectures, review the technological enablers, and present a taxonomy that classifies the design space into two dimensions: static vs. dynamic and demand-oblivious vs. demand-aware. We further present a formal model and discuss related research challenges. Our article comes with complementary video interviews in which three leading experts, Manya Ghobadi, Amin Vahdat, and George Papen, share with us their perspectives on reconfigurable datacenter networks.

KEY INSIGHTS

- **Datacenter networks have become a critical infrastructure** for our digital society, serving explosively growing communication traffic.
- **Reconfigurable datacenter networks (RDCNs)** which can adapt their topology dynamically, based on innovative **optical switching technologies**, bear the potential to improve datacenter network performance, and to simplify datacenter planning and operations.
- Demand-aware dynamic topologies are particularly interesting because of the **significant spatial and temporal structures** observed in real-world traffic, e.g., related to distributed machine learning.
- The study of RDCNs and self-adjusting networks raises many **novel technological and research challenges** related to their design, control, and performance.

References (1)

Revolutionizing Datacenter Networks via Reconfigurable Topologies
Chen Avin and Stefan Schmid.
Communications of the ACM (**CACM**), 2025.

Cerberus: The Power of Choices in Datacenter Topology Design (A Throughput Perspective)
Chen Griner, Johannes Zerwas, Andreas Blenk, Manya Ghobadi, Stefan Schmid, and Chen Avin.
ACM **SIGMETRICS** and ACM Performance Evaluation Review (**PER**), Mumbai, India, June 2022.

Mars: Near-Optimal Throughput with Shallow Buffers in Reconfigurable Datacenter Networks
Vamsi Addanki, Chen Avin, and Stefan Schmid.
ACM **SIGMETRICS** and ACM Performance Evaluation Review (**PER**), Orlando, Florida, USA, June 2023.

Duo: A High-Throughput Reconfigurable Datacenter Network Using Local Routing and Control
Johannes Zerwas, Csaba Györgyi, Andreas Blenk, Stefan Schmid, and Chen Avin.
ACM **SIGMETRICS** and ACM Performance Evaluation Review (**PER**), Orlando, Florida, USA, June 2023.

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

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

Credence: Augmenting Datacenter Switch Buffer Sharing with ML Predictions
Vamsi Addanki, Maciej Pacut, and Stefan Schmid.
21st USENIX Symposium on Networked Systems Design and Implementation (**NSDI**), Santa Clara, California, USA, April 2024.

PowerTCP: Pushing the Performance Limits of Datacenter Networks
Vamsi Addanki, Oliver Michel, and Stefan Schmid.
19th USENIX Symposium on Networked Systems Design and Implementation (**NSDI**), Renton, Washington, USA, April 2022.

TCP's Third Eye: Leveraging eBPF for Telemetry-Powered Congestion Control
Jörn-Thorben Hinz, Vamsi Addanki, Csaba Györgyi, Theo Jepsen, and Stefan Schmid.
SIGCOMM Workshop on eBPF and Kernel Extensions (**eBPF**), Columbia University, New York City, New York, USA, September 2023.

References (2)

ABM: Active Buffer Management in Datacenters
Vamsi Addanki, Maria Apostolaki, Manya Ghobadi, Stefan Schmid, and Laurent Vanbever.
ACM **SIGCOMM**, Amsterdam, Netherlands, August 2022.

ExRec: Experimental Framework for Reconfigurable Networks Based on Off-the-Shelf Hardware
Johannes Zerwas, Chen Avin, Stefan Schmid, and Andreas Blenk.
16th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (**ANCS**), Virtual Conference, December 2021.

Demand-Aware Network Design with Minimal Congestion and Route Lengths
Chen Avin, Kaushik Mondal, and Stefan Schmid.
IEEE/ACM Transactions on Networking (**TON**), 2022.

A Survey of Reconfigurable Optical Networks
Matthew Nance Hall, Klaus-Tycho Foerster, Stefan Schmid, and Ramakrishnan Durairajan.
Optical Switching and Networking (**OSN**), Elsevier, 2021.

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.

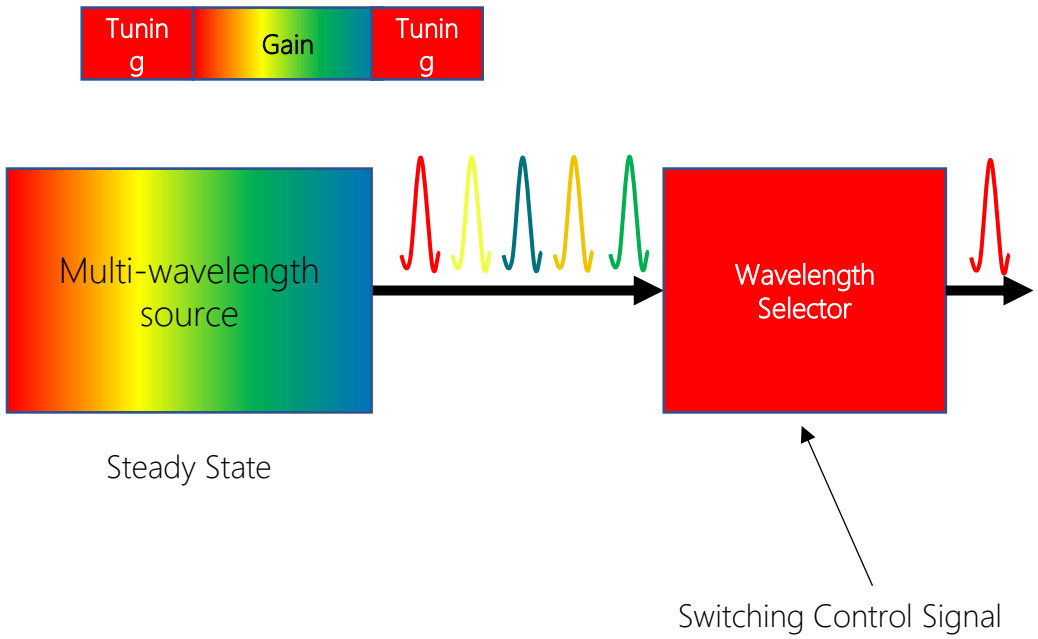
.

Bonus Material

Hogwarts Stair

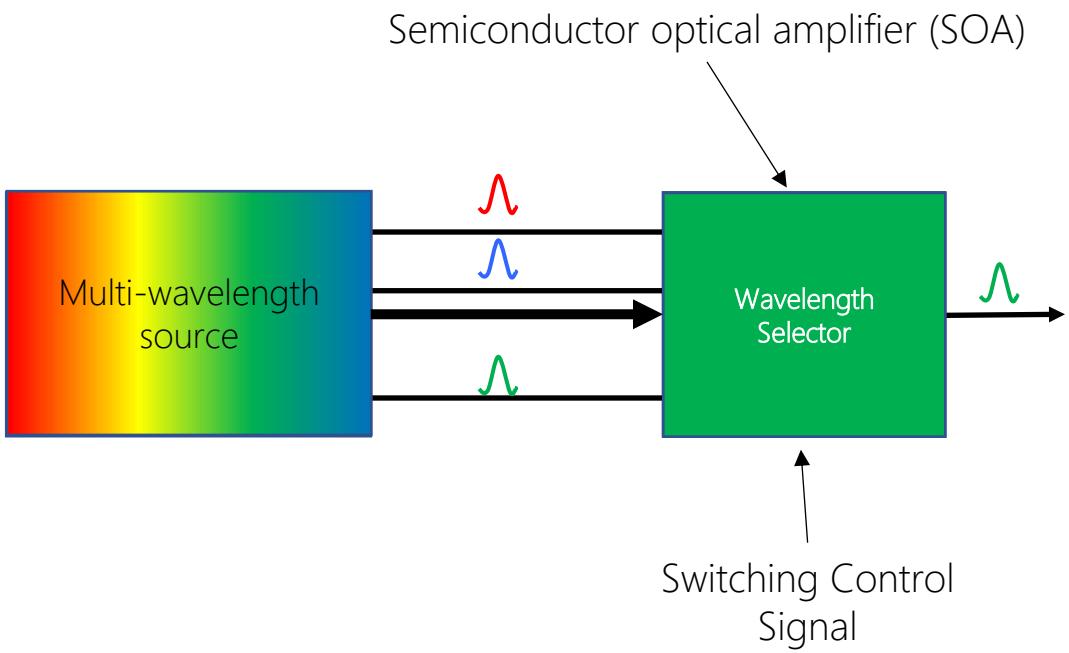
Idea

Disaggregated Laser



Ballani et al., Sirius, ACM SIGCOMM 2020.

Example Design



Sirius also implemented other designs
(details in the paper)

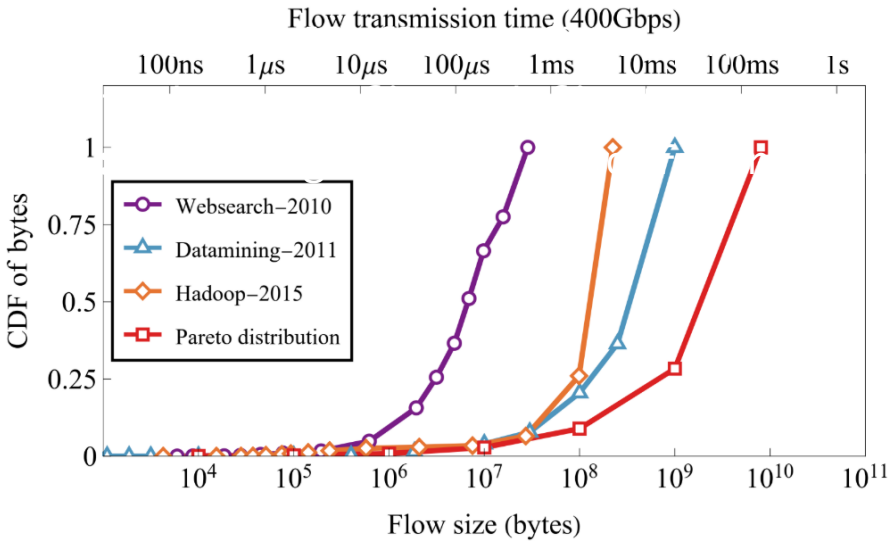
Optimal Hybrid RDCNs

Flow Size Matters

On what should topology type depend? We argue: flow size.

Flow Size Matters

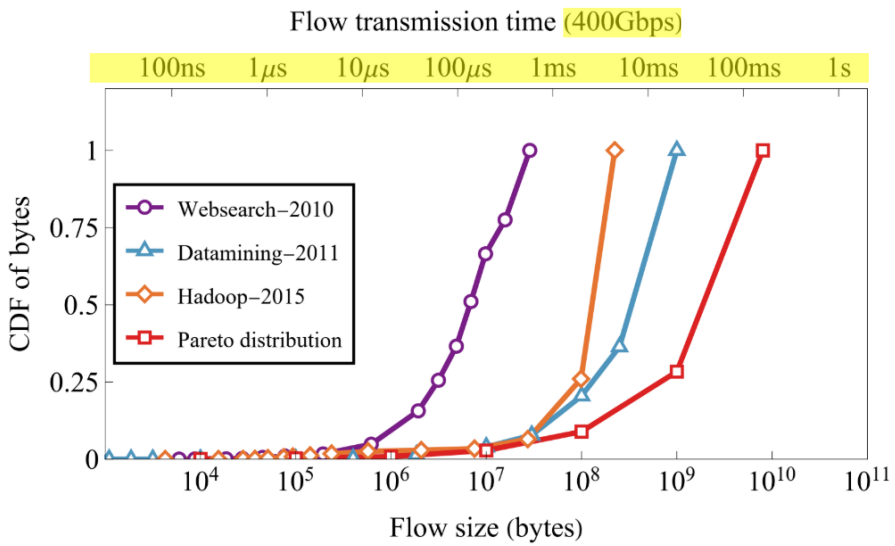
On what should topology type depend? We argue: flow size.



⇢ **Observation 1:** Different apps have different flow size distributions.

Flow Size Matters

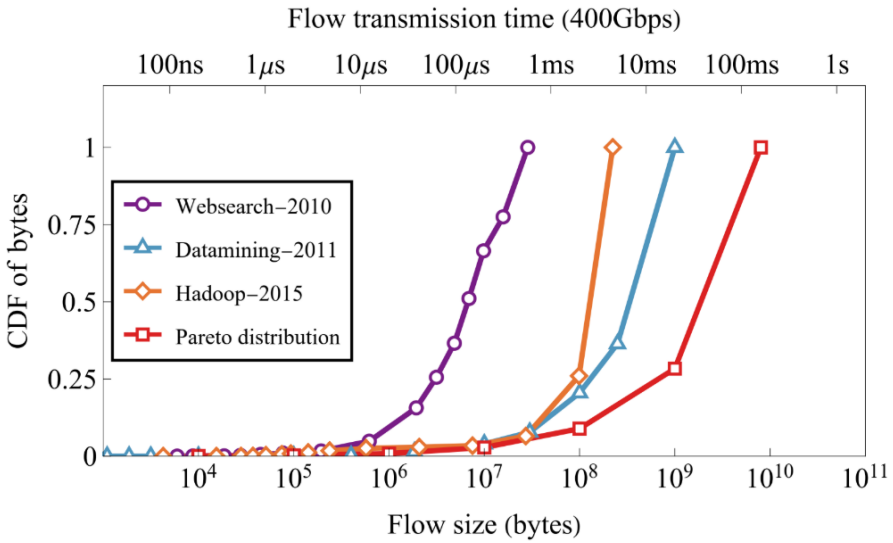
Similar tradeoff for
800Gbps or 1.6 Tpbs



⇢ **Observation 1:** Different apps have different flow size distributions.

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

Flow Size Matters



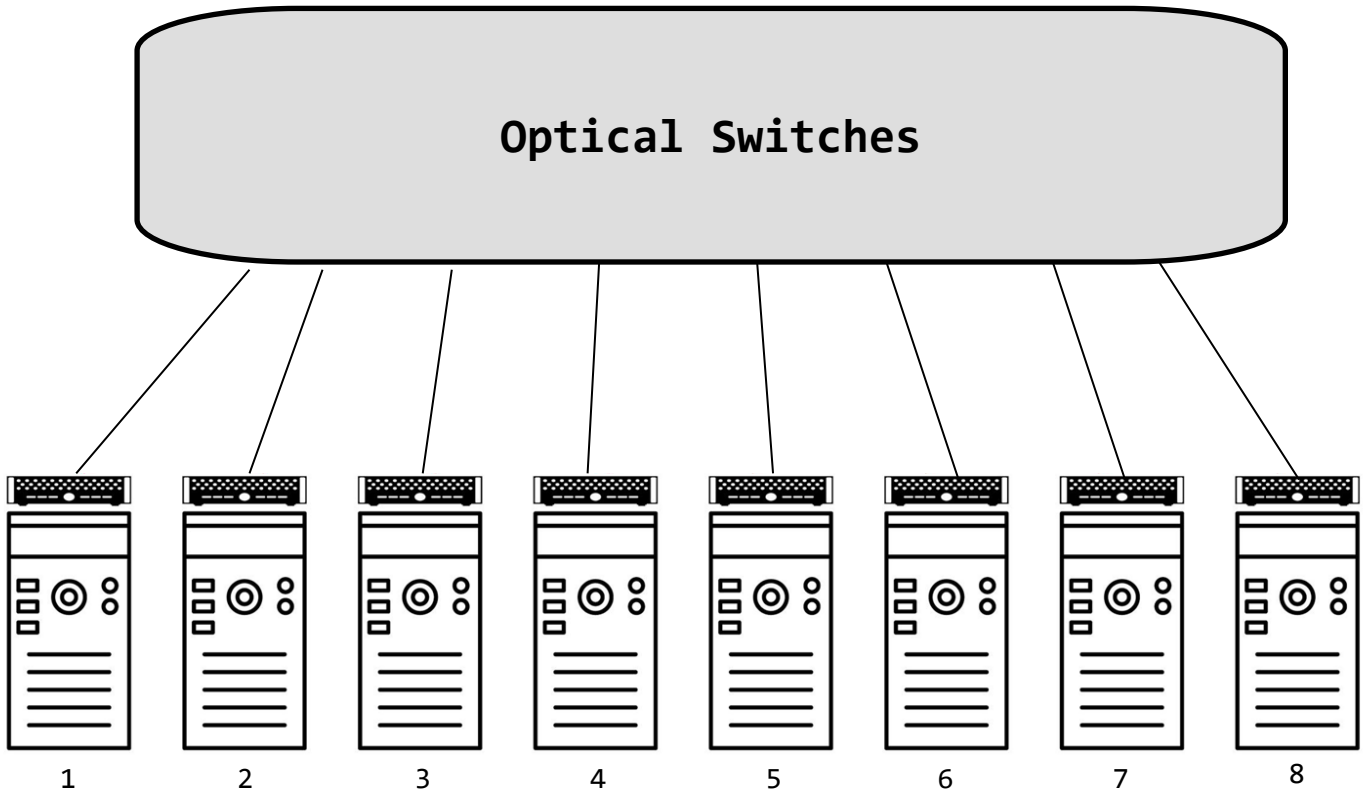
⇢ **Observation 1:** Different apps have different flow size distributions.

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

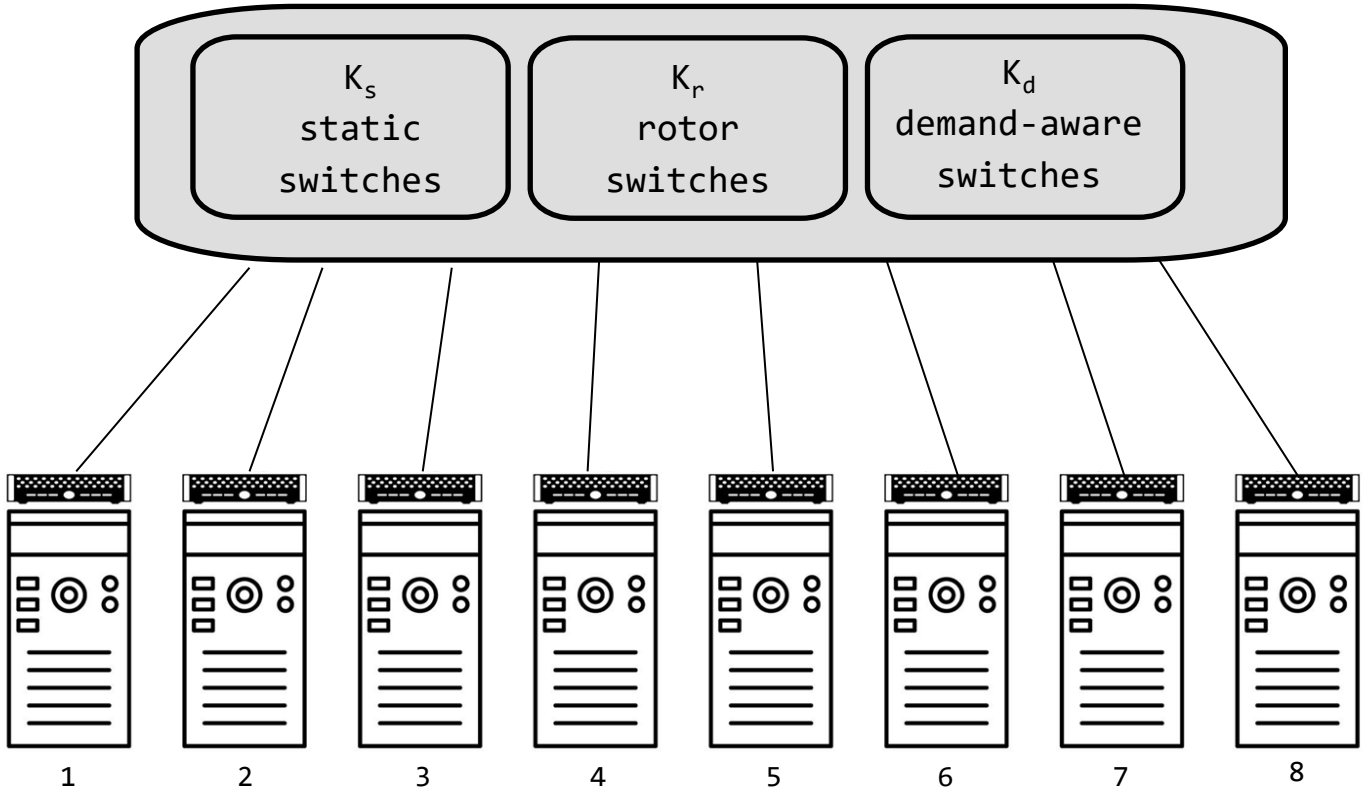
⇢ **Observation 3:** For small flows, flow completion time suffers if network needs to be reconfigured first.

⇢ **Observation 4:** For large flows, reconfiguration time may amortize.

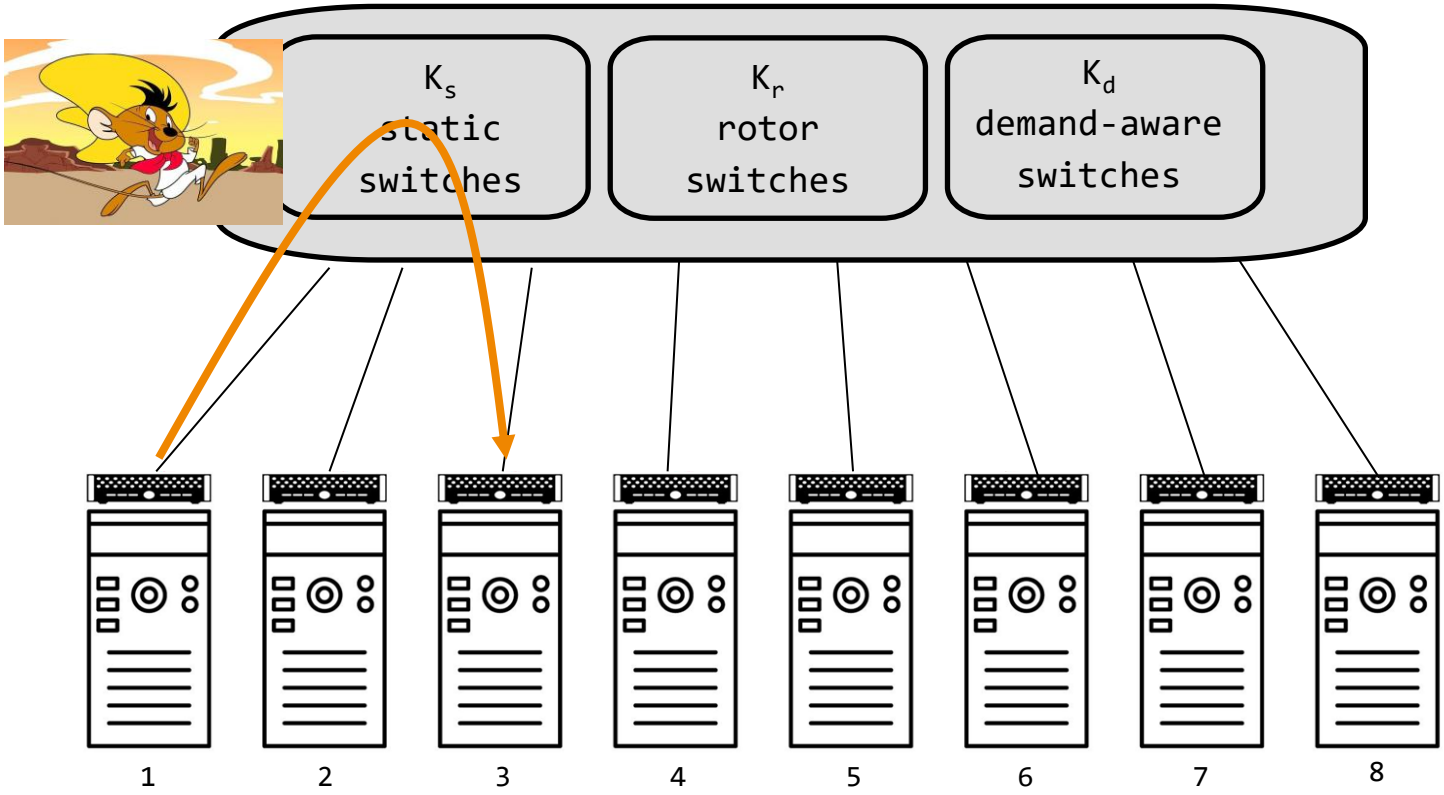
Optimal: Cerberus



Optimal: Cerberus

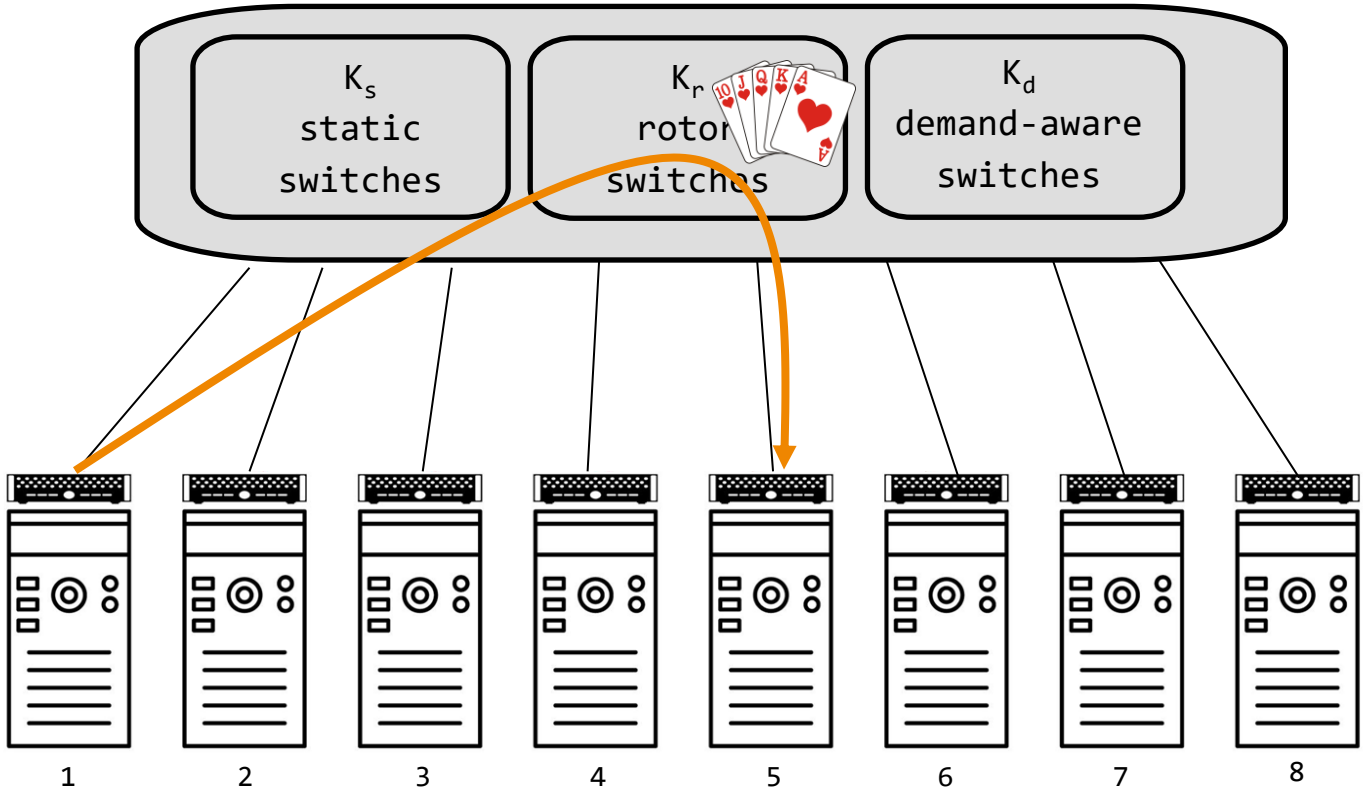


Optimal: Cerberus



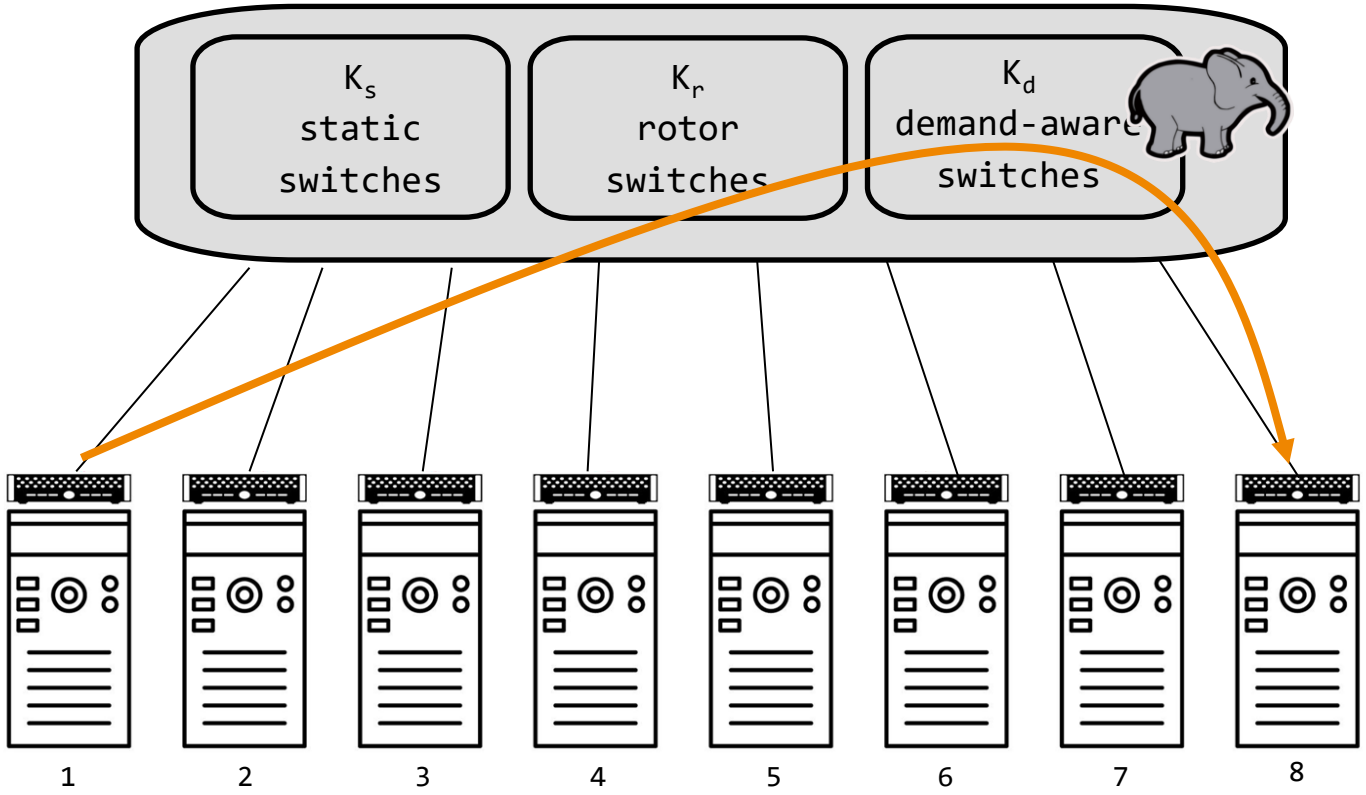
Scheduling: Small flows go via static switches…

Optimal: Cerberus



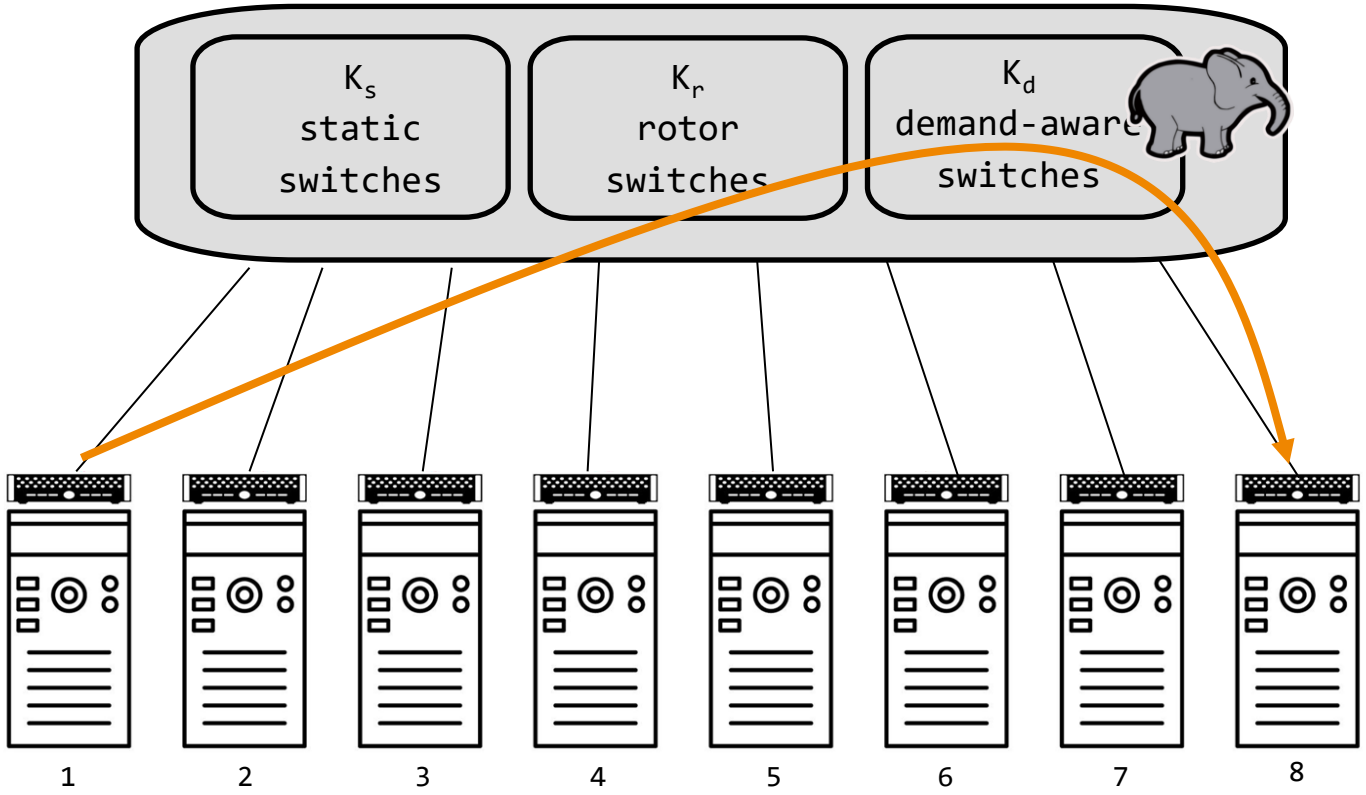
Scheduling: … medium flows via rotor switches…

Optimal: Cerberus



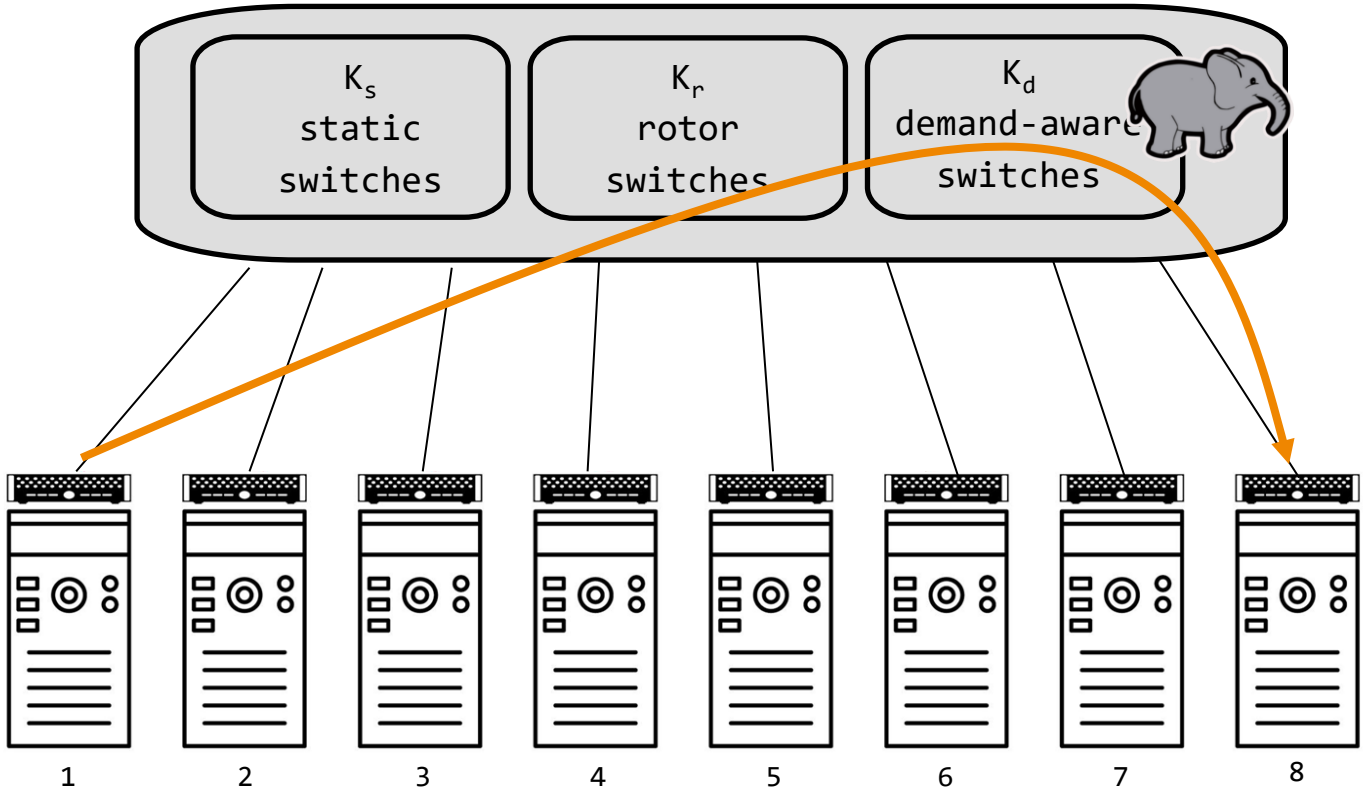
Scheduling: … and large flows via demand-aware switches
(if one available, otherwise via rotor).

Optimal: Cerberus



How good is it? Open problem. But there are bounds.

Optimal: Cerberus



How good is it? Open problem. But there are bounds.

How to realize such an architecture?! Scalable control plane?