

From Topology-Aware to Fabric-Adaptive: Rethinking Datacenter Networks for AI

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”)



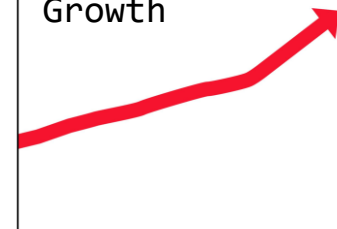
NETFLIX



+network

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

Traffic
Growth



Source: Facebook

We live in the age of

Distributed Computation

Datacenters (“hyper-scale”)



NETFLIX



+network

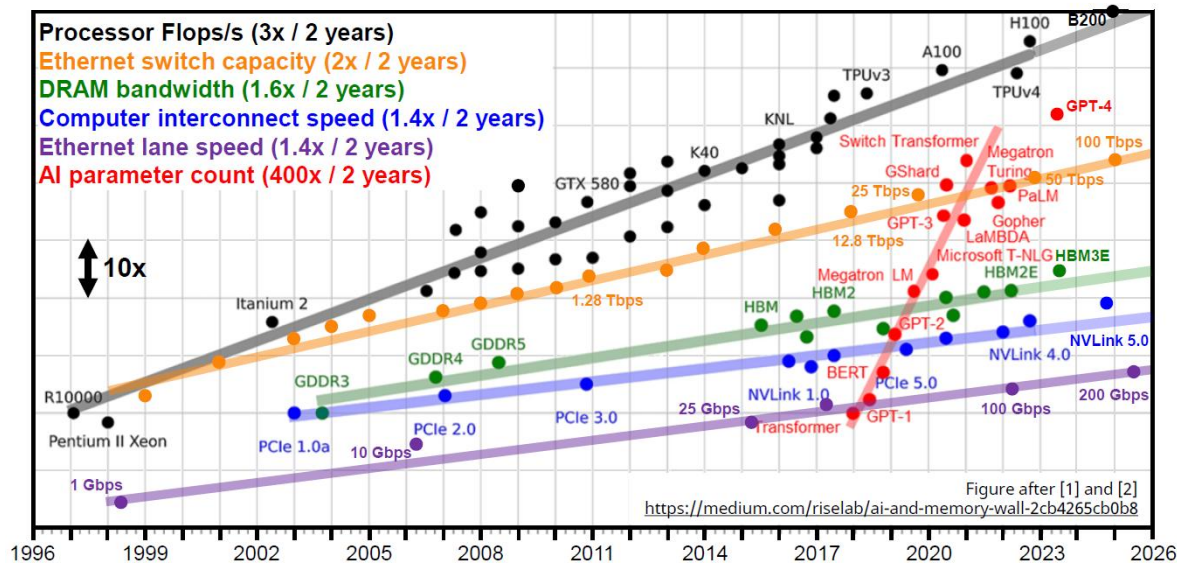
Interconnecting networks:
a **critical infrastructure**
of our digital society.



Credits: Marco Chiesa

Technological Trends

Increasing Gap Between Compute and Network

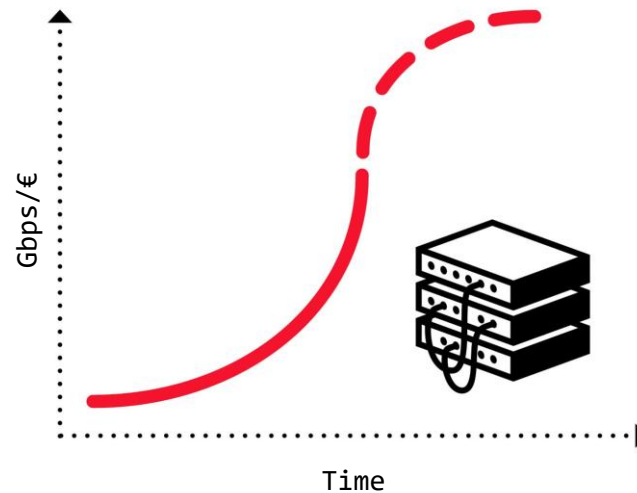


Credits: Nicola Calabretta

The Problem

Huge Infrastructure, Inefficient Use

- Hence: more equipment, larger networks
- Resource intensive and: **inefficient**



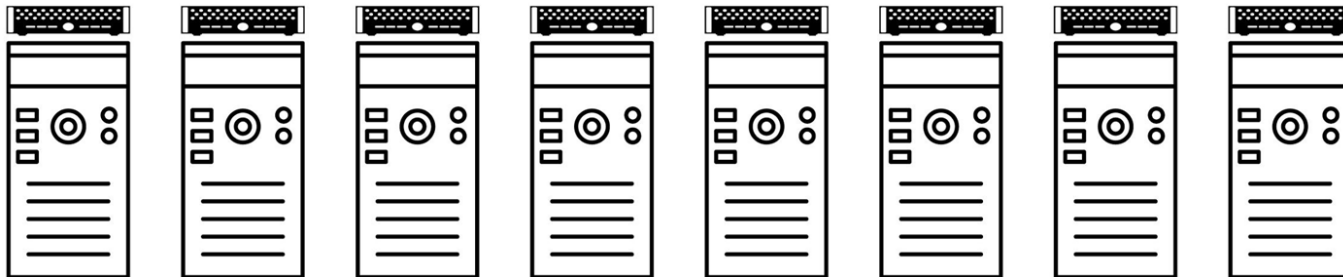
Credits: Paolo Costa, 2019

Annoying for companies,
opportunity for researchers!

An Inefficiency

Fixed and Demand-Oblivious Topology

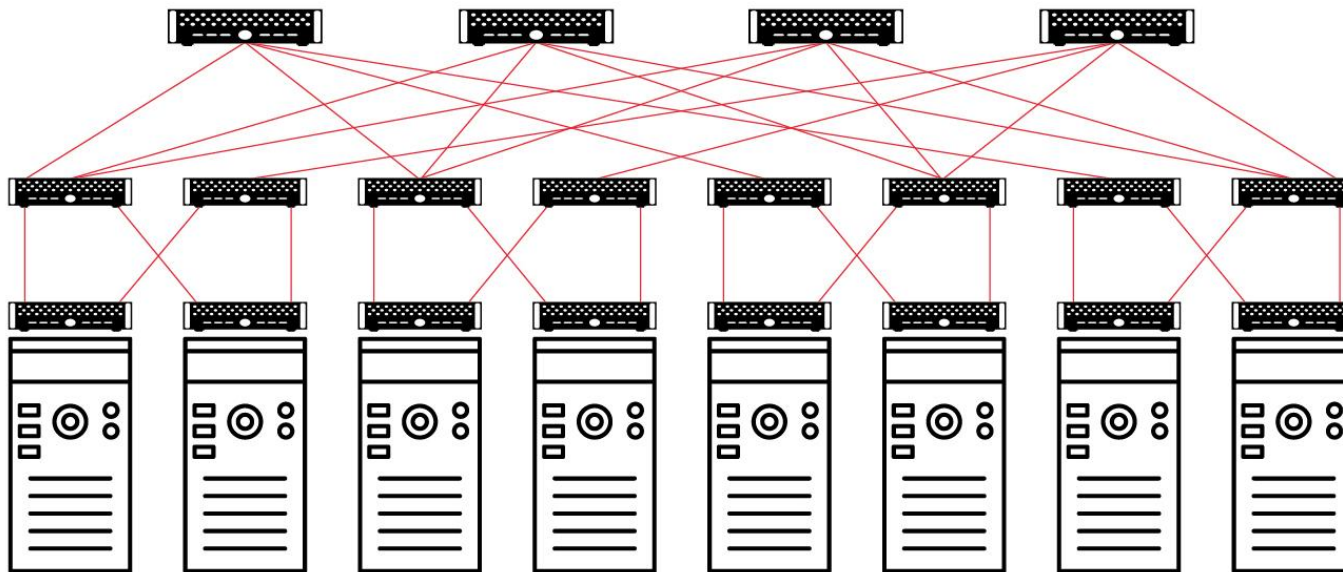
How to interconnect? Focus on this talk: **scale-out network**.



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

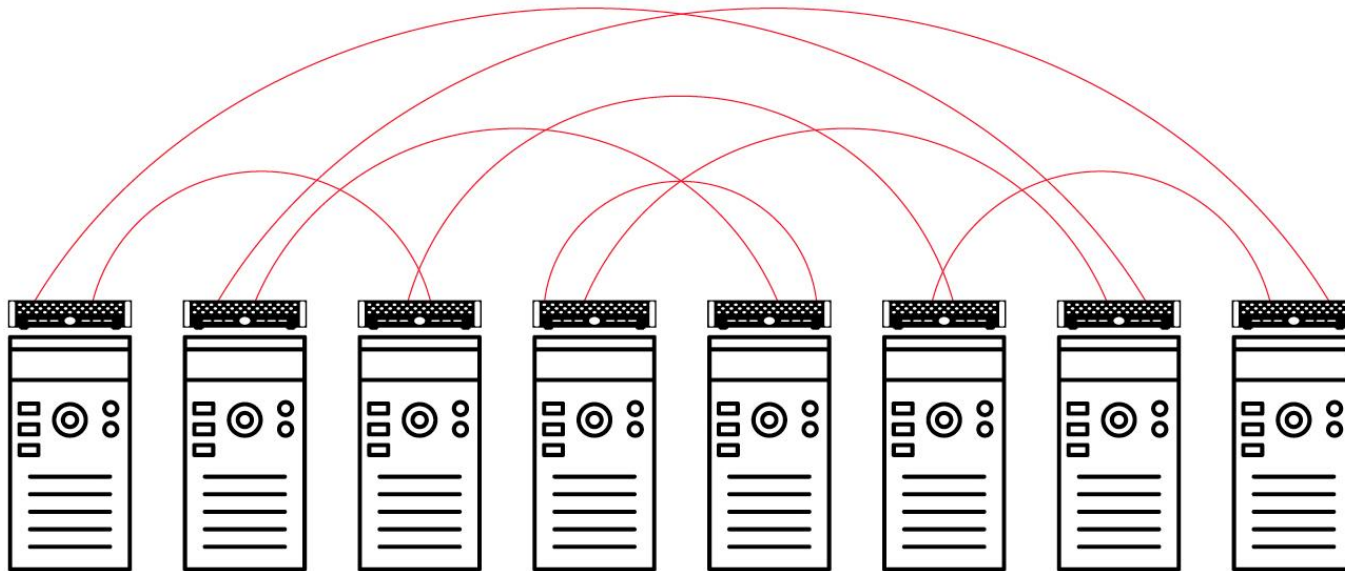


An Inefficiency

Fixed and Demand-Oblivious Topology

→ Example: expander topology (**uni-regular**)

- Only 1 type of switches:
lower installation and management overheads

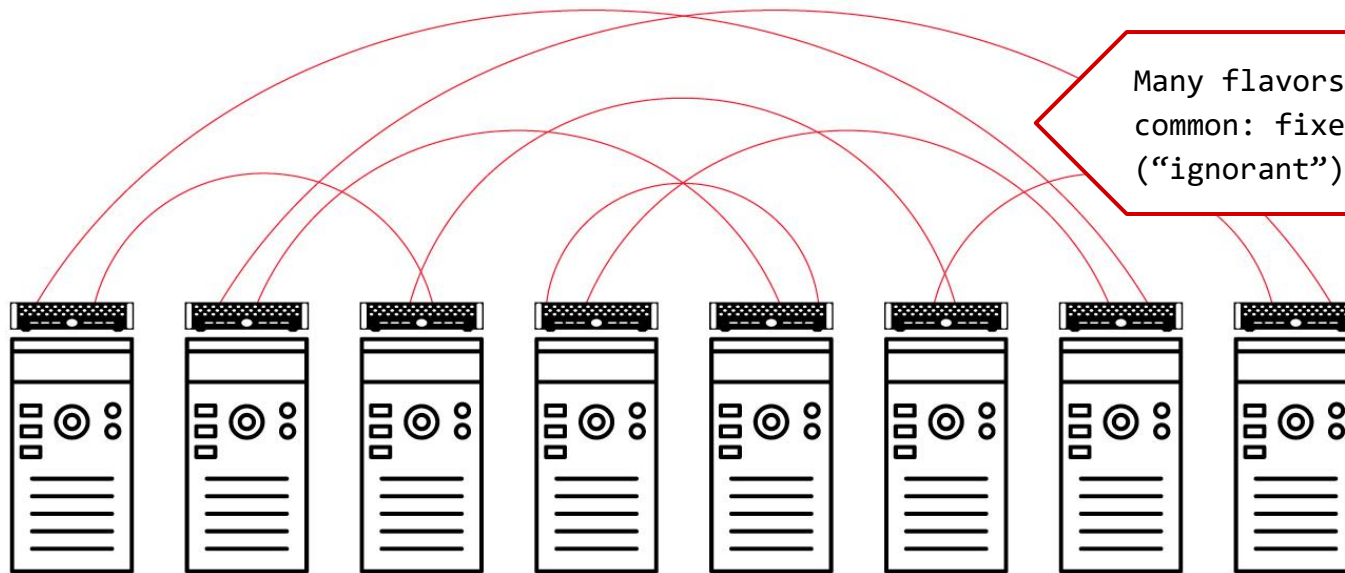


An Inefficiency

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

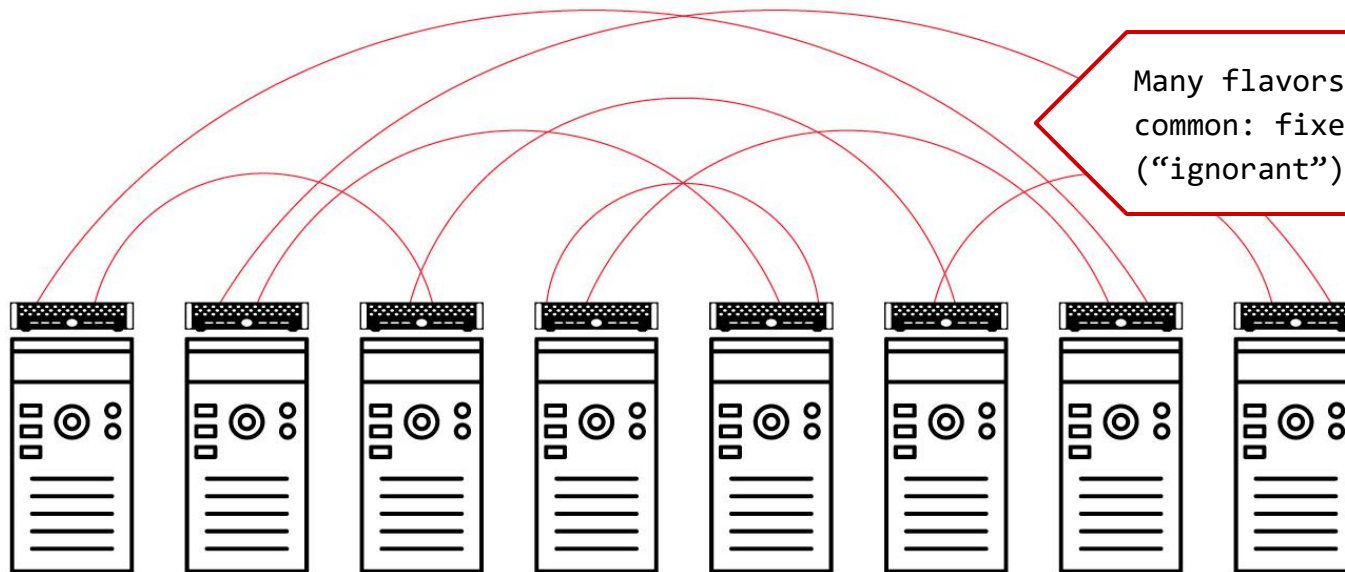
Fixed and Demand-Oblivious Topology



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

→ Example: expander topology (**uni-regular**)

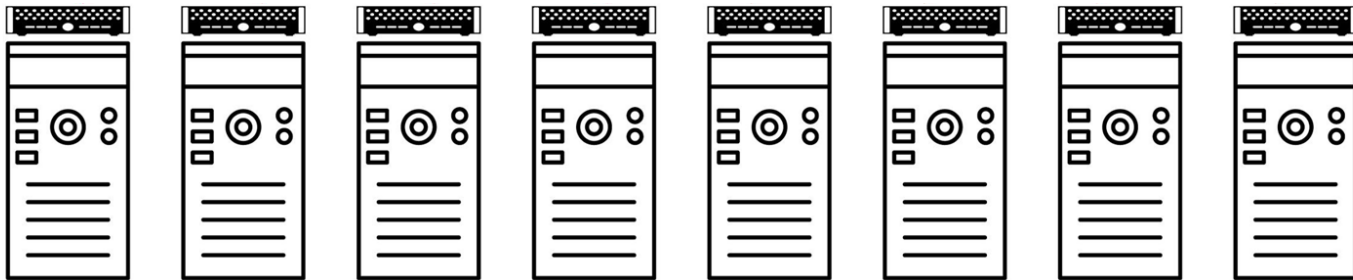
- Only 1 type of switches:
lower installation and management overheads



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

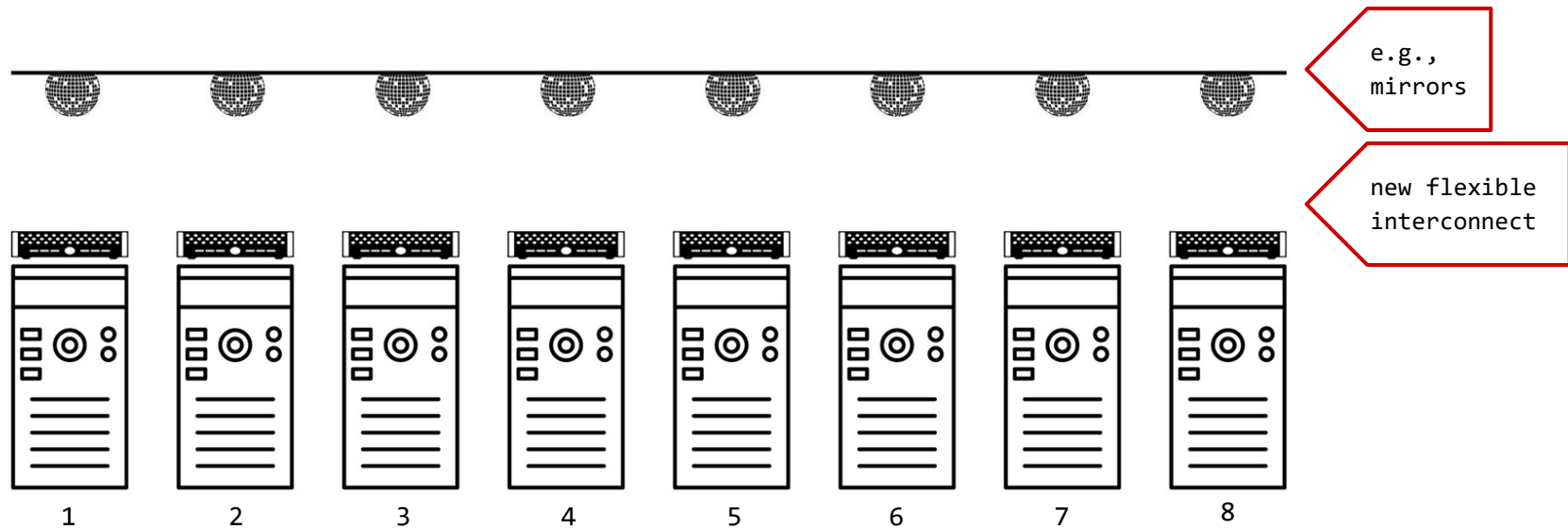
A Vision

Flexible and Demand-Aware Topologies



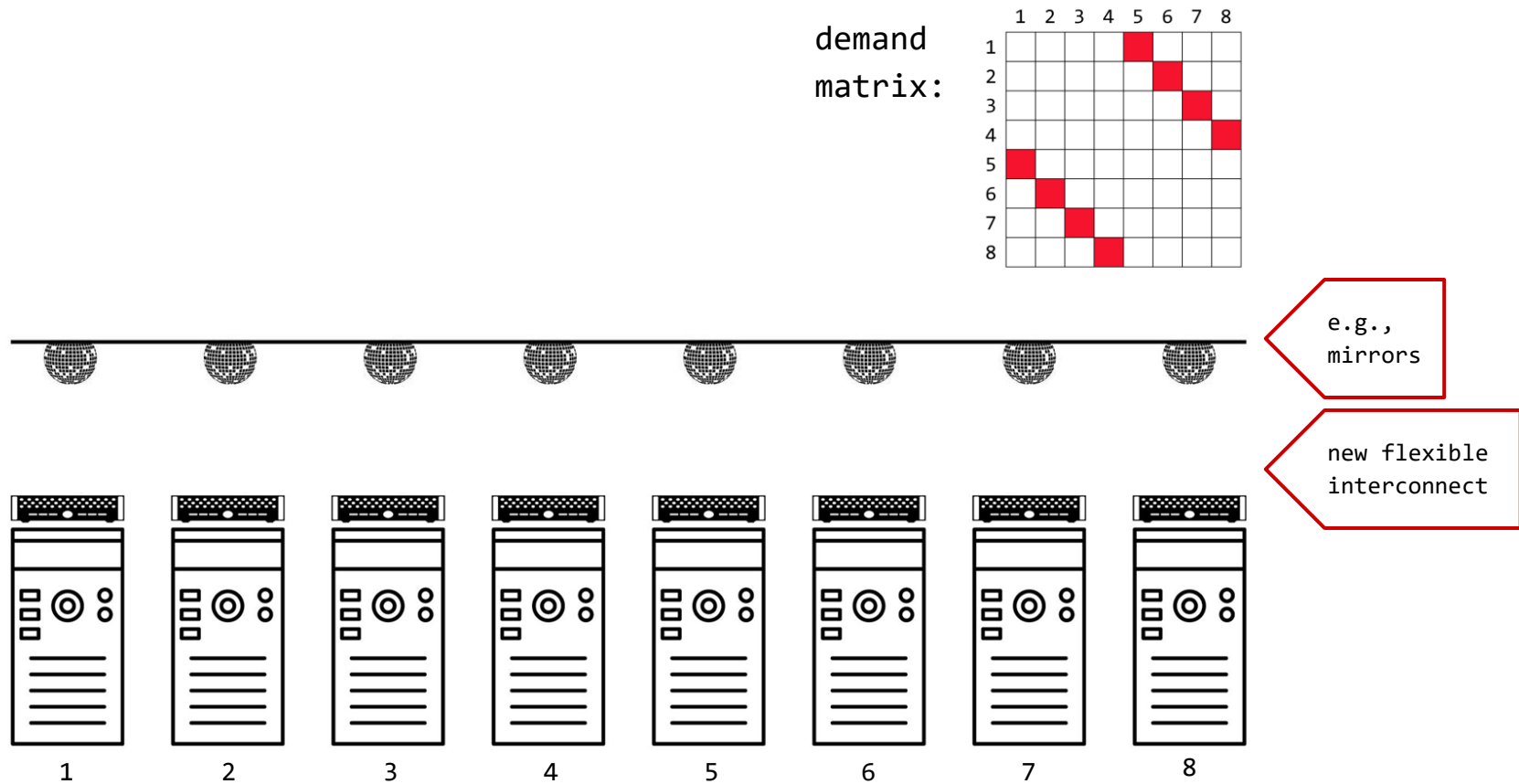
A Vision

Flexible and Demand-Aware Topologies



A Vision

Flexible and Demand-Aware Topologies



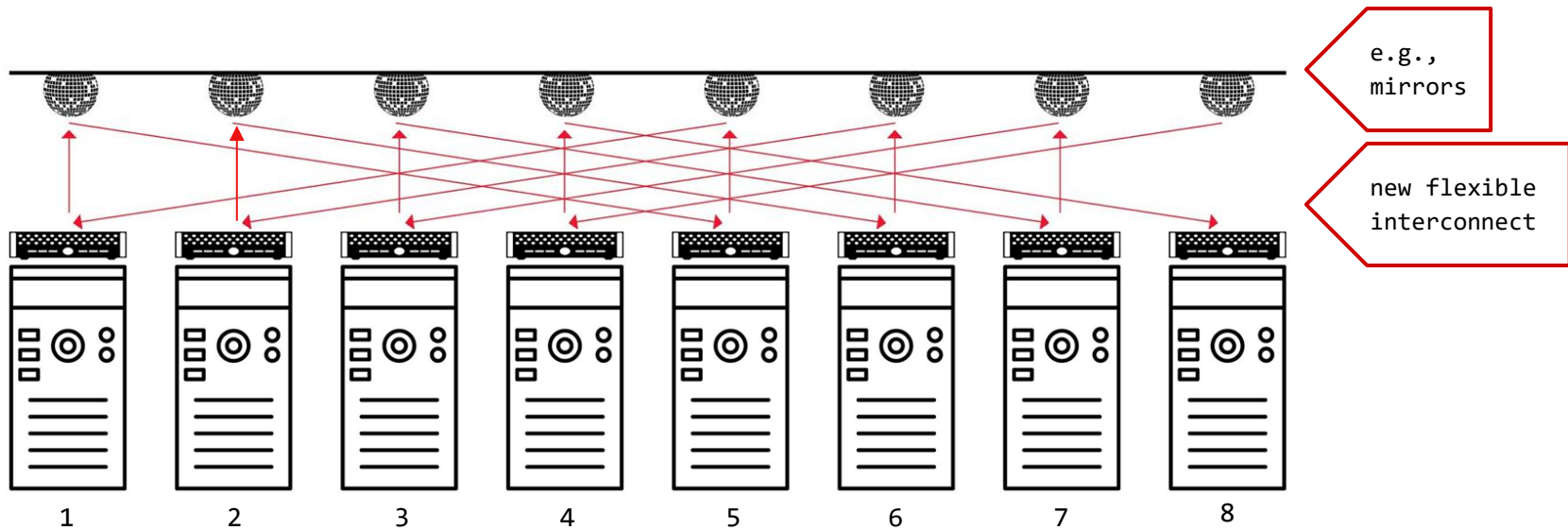
A Vision

Flexible and Demand-Aware Topologies

Matches demand

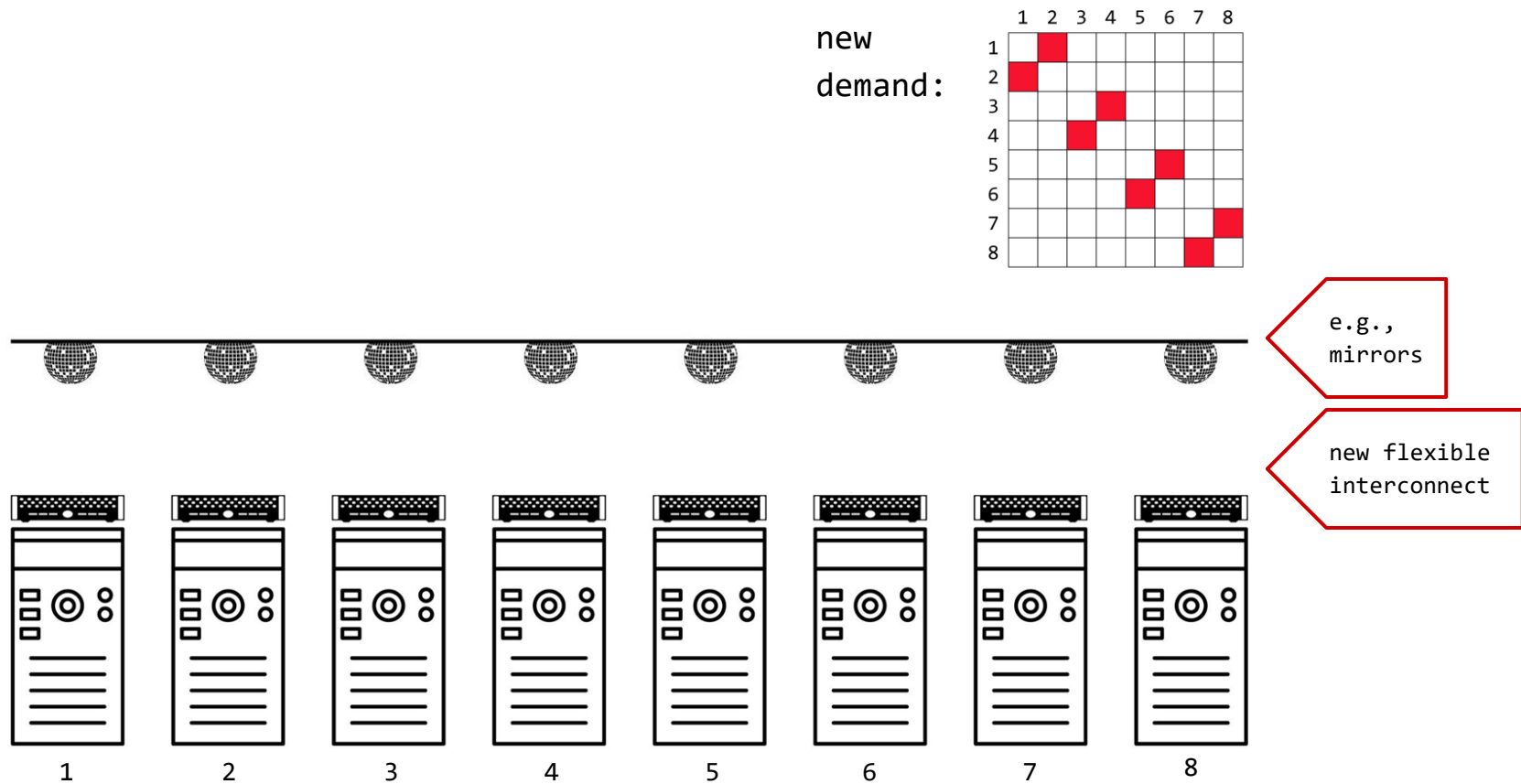
demand
matrix:

	1	2	3	4	5	6	7	8
1					■			
2						■		
3							■	
4								■
5	■							
6		■						
7			■					
8				■				



A Vision

Flexible and Demand-Aware Topologies



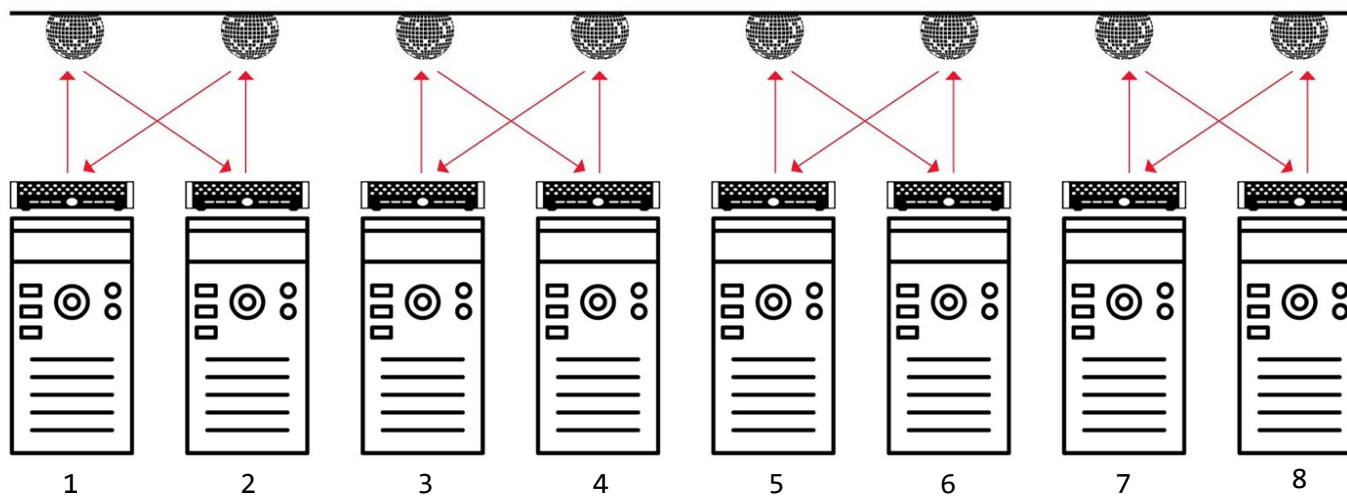
A Vision

Flexible and Demand-Aware Topologies

Matches demand

new
demand:

	1	2	3	4	5	6	7	8
1		■						
2	■							
3				■				
4			■					
5						■		
6					■			
7							■	
8								■



e.g.,
mirrors

new flexible
interconnect

A Vision

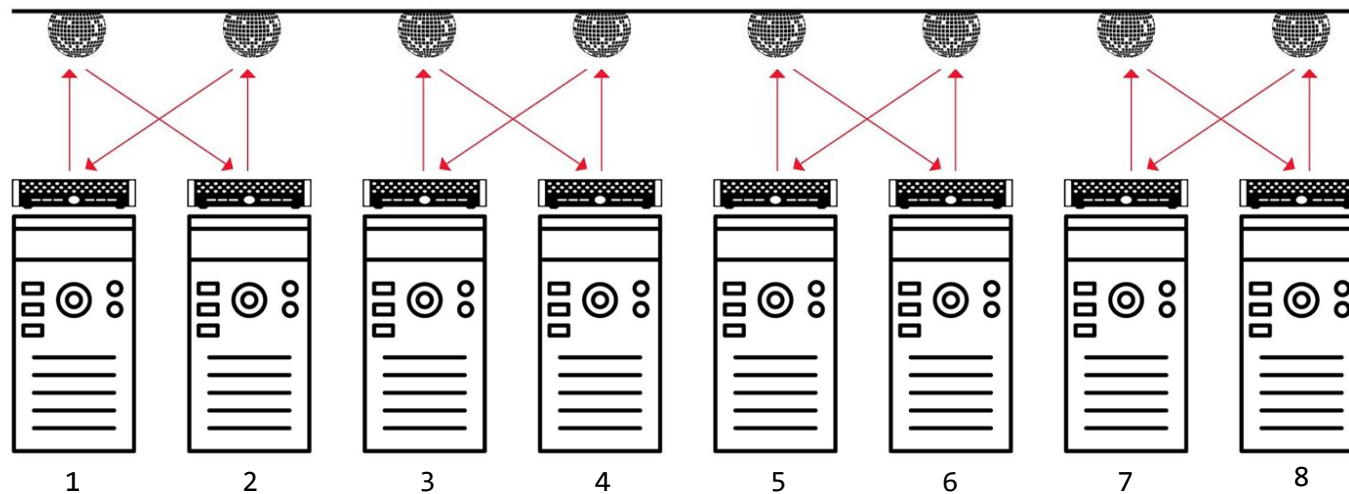
Flexible and Demand-Aware Topologies



Self-Adjusting Networks

new
demand:

	1	2	3	4	5	6	7	8
1		■						
2	■							
3				■				
4			■					
5						■		
6					■			
7							■	
8								■



e.g.,
mirrors

new flexible
interconnect

Analogy



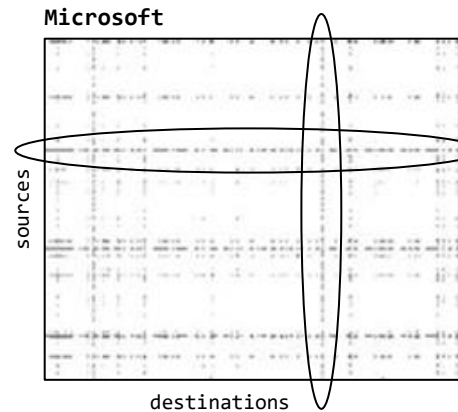
Golden Gate Zipper

Empirical Motivation

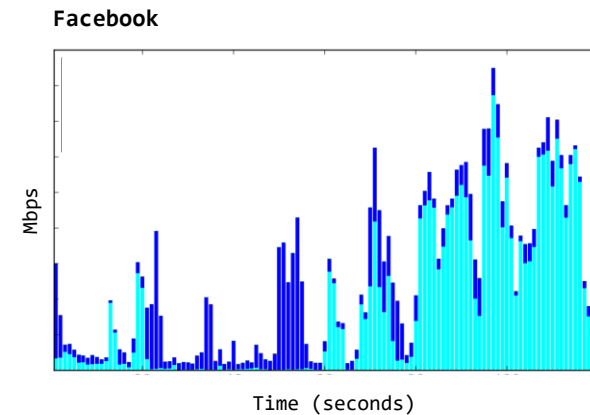
Much Structure in the Demand

Empirical studies:

traffic matrices **sparse** and **skewed**



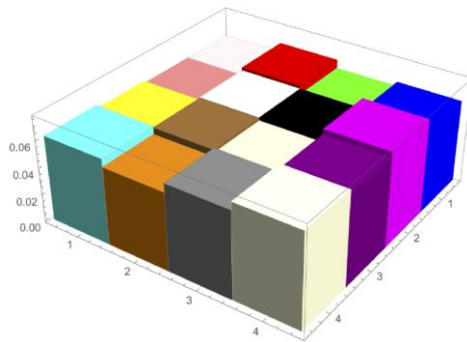
traffic **bursty** over time



The **hypothesis**: can be exploited.

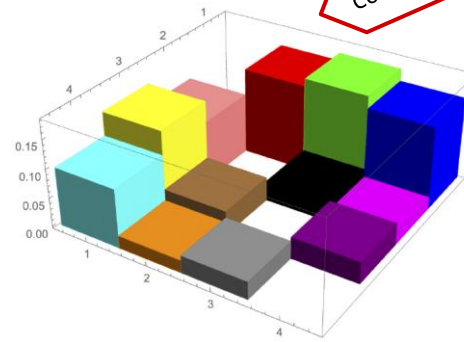
Example: ML Workloads

→ Traffic matrices of two different distributed ML apps
→ 4 GPUs, GPU-to-GPU traffic



ML App 1

VS



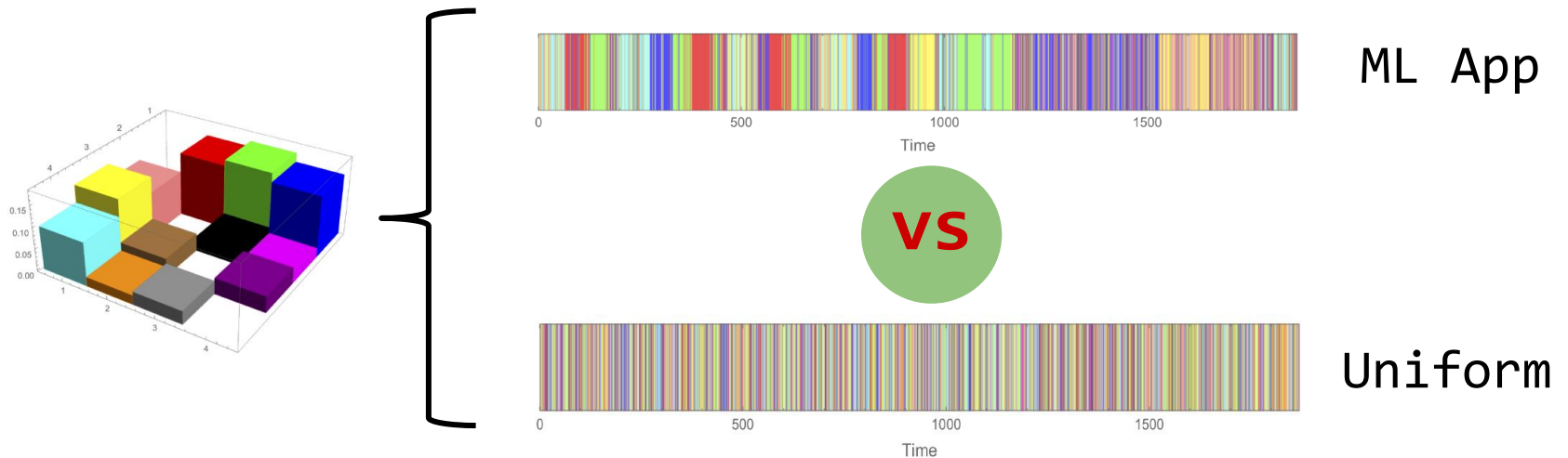
ML App 2

Color = communication pair

Can have significant spatial structure!

Example: ML Workloads

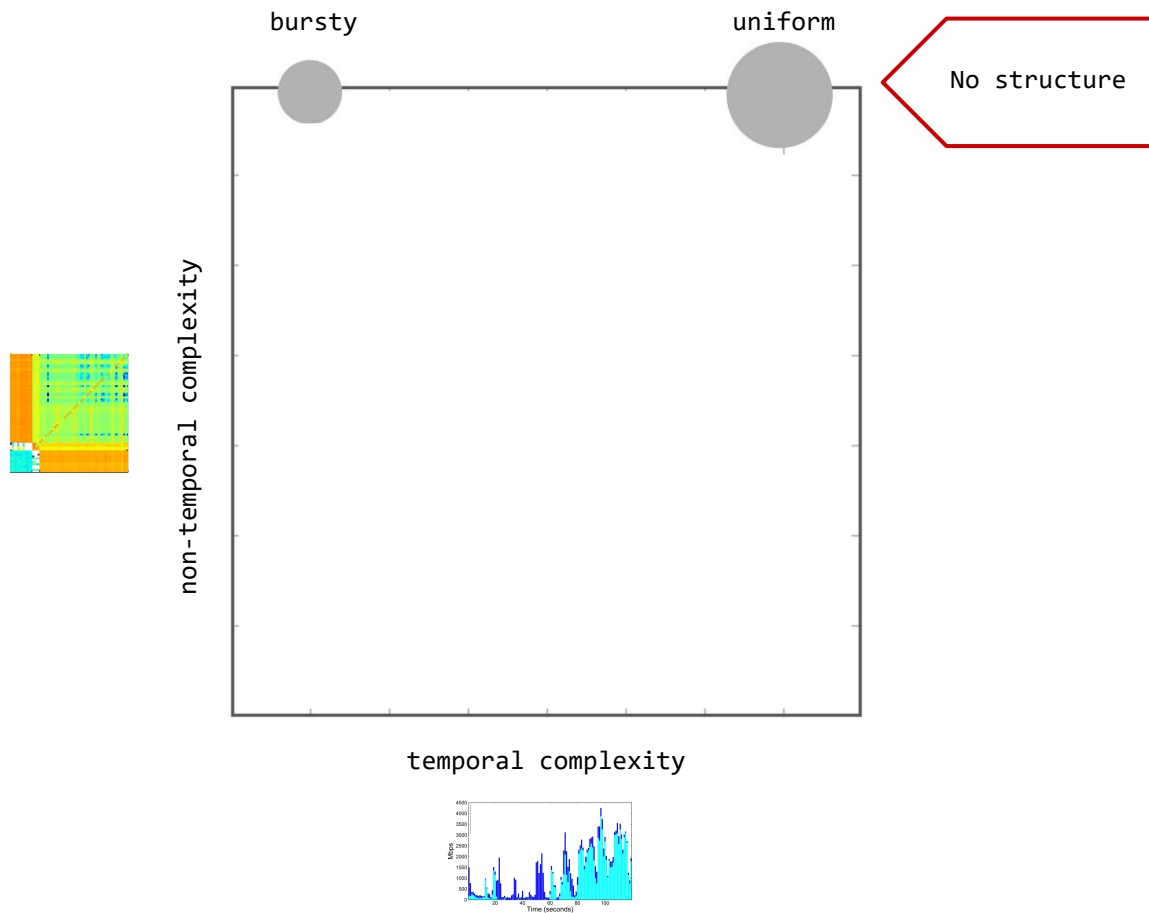
→ Traffic matrices of two different distributed ML apps
→ 4 GPUs, GPU-to-GPU traffic



Can have significant temporal structure!

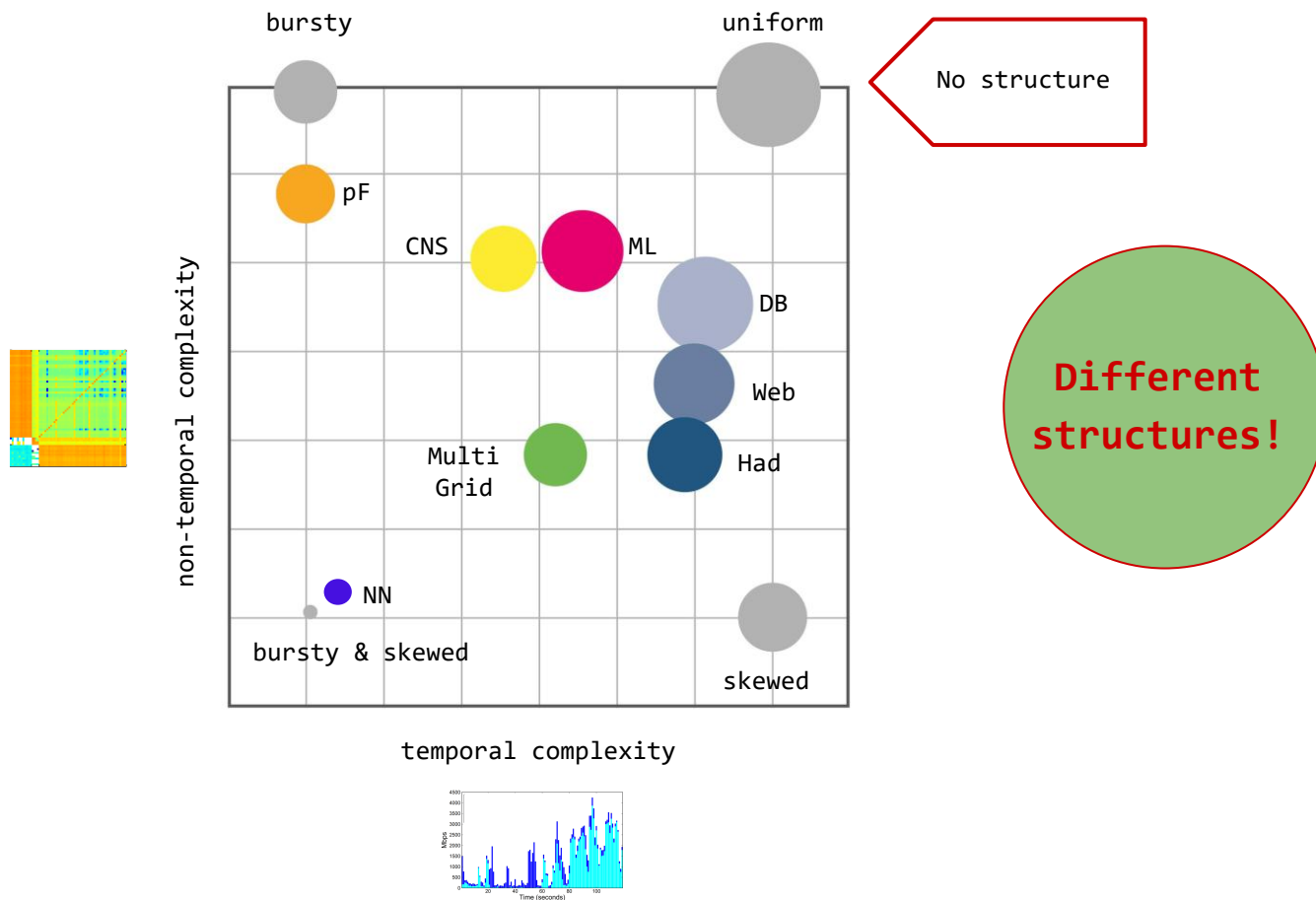
Visualizing Workload Structure

Complexity Map



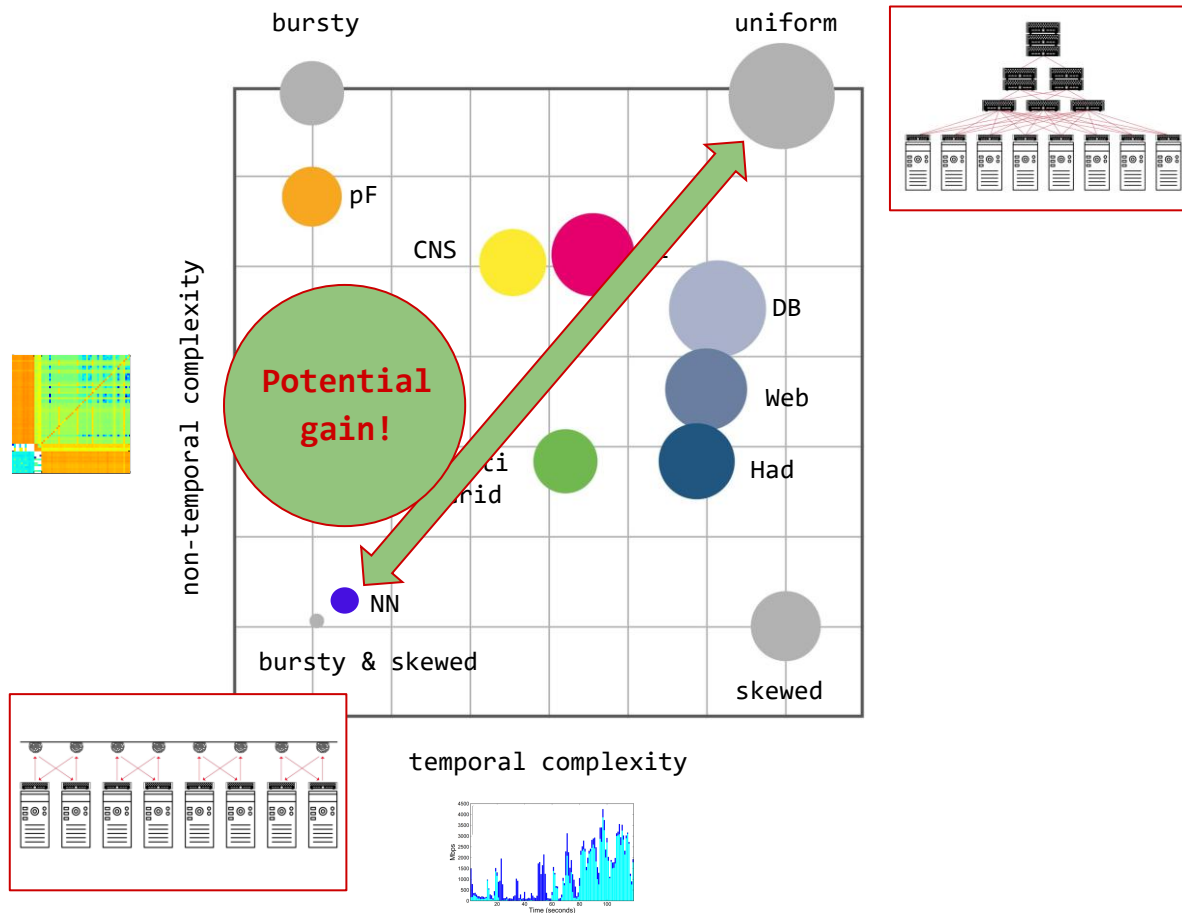
Visualizing Workload Structure

Complexity Map



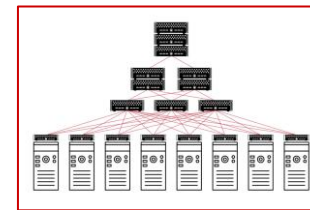
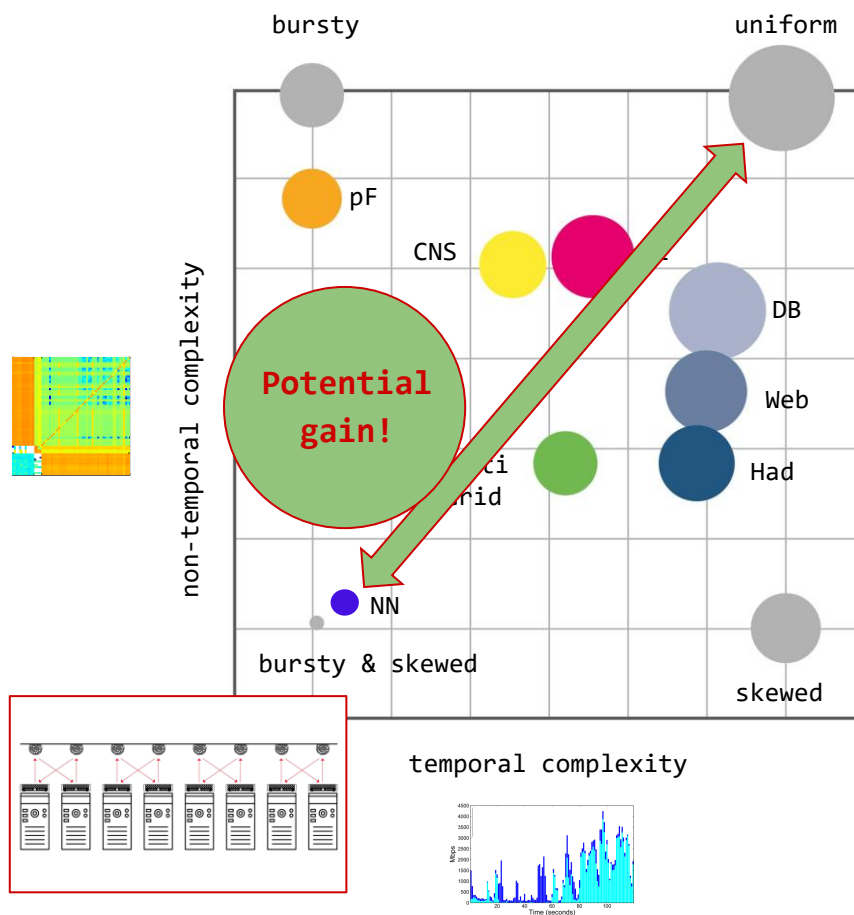
Visualizing Workload Structure

Complexity Map



Visualizing Workload Structure

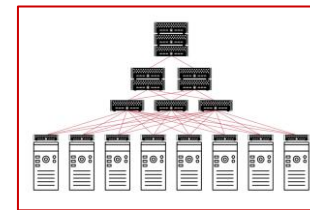
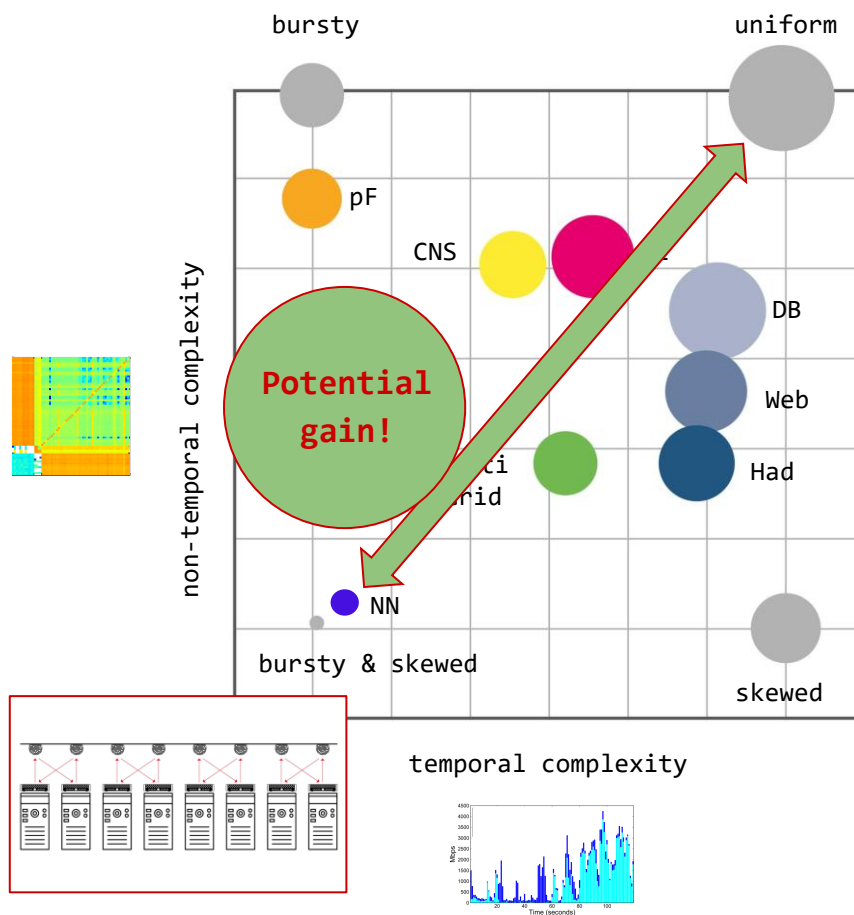
Complexity Map



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

Visualizing Workload Structure

Complexity Map



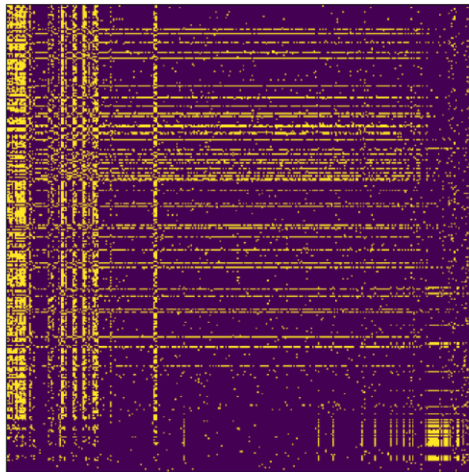
Systematic approach:
iterative **randomization**
and compression of trace
to identify dimensions of
structure.

Further reading:

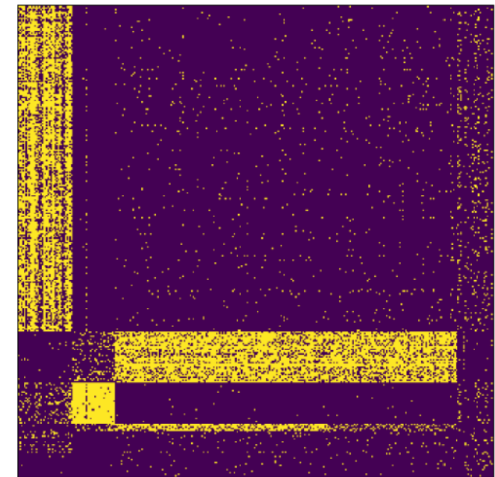
On the Complexity of Traffic Traces and Implications
Chen Avin et al. ACM **Sigmetrics 2020**.

Traffic is also clustered: bi-clustering results

Small Stable Clusters



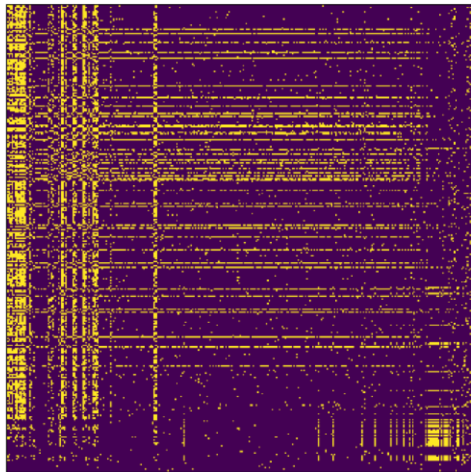
reordering based on
bicluster structure



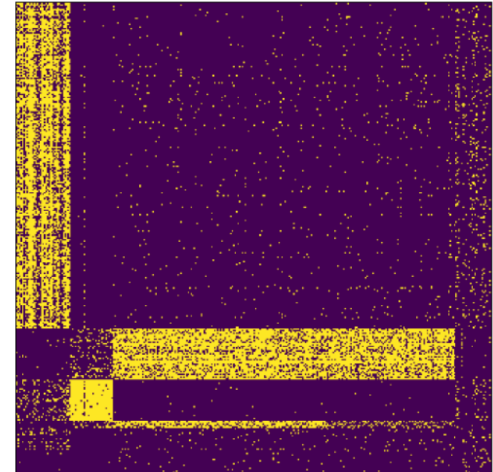
Opportunity: *exploit* with little reconfigurations!

Traffic is also clustered: bi-clustering results

Small Stable Clusters



reordering based on
bicluster structure

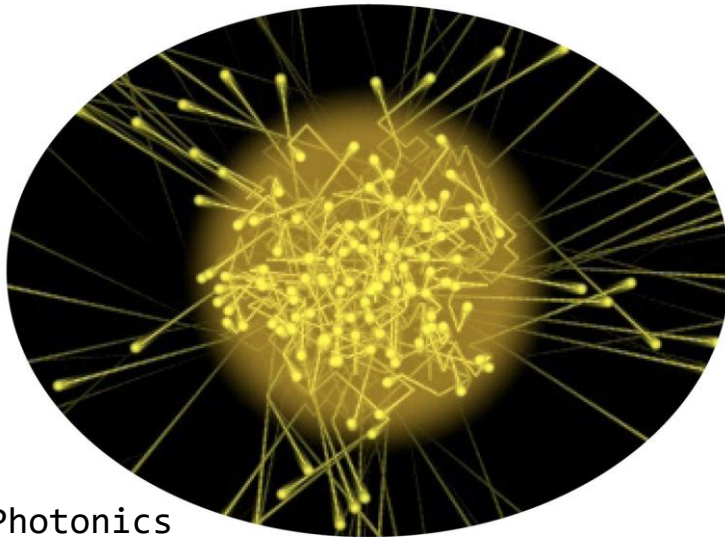


Opportunity: *exploit* with little reconfigurations!

Further reading:

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

Sounds Crazy? Emerging Enabling Technology.



Photonics

H2020:

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

US National Research Council:

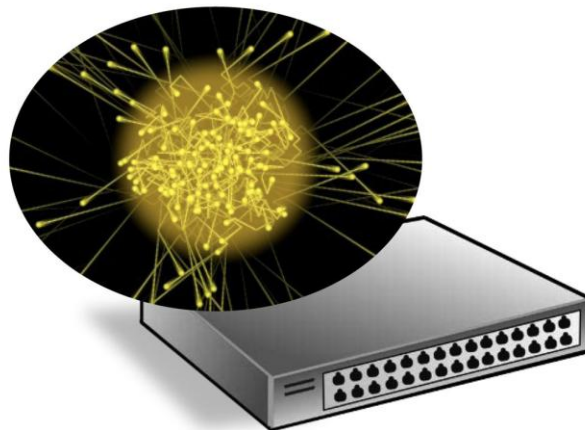
**“Photons are the new
Electrons.”**

Enabler

Novel Reconfigurable Optical Switches

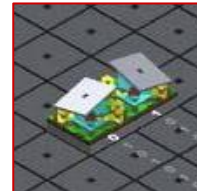
→ **Spectrum** of prototypes

- Different sizes, different reconfiguration times
- From ACM **SIGCOMM** workshop OptSys resp. HotOptics



Prototype 1

Moving antenna (ms)



Prototype 2

Moving mirrors (μs)



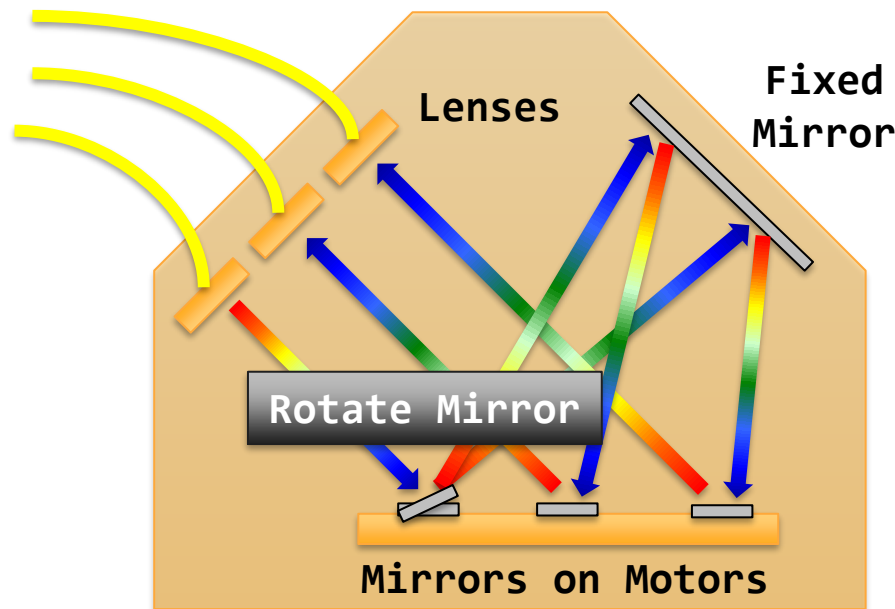
Prototype 3

Changing lambdas (ns)

Example

Optical Circuit Switch

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



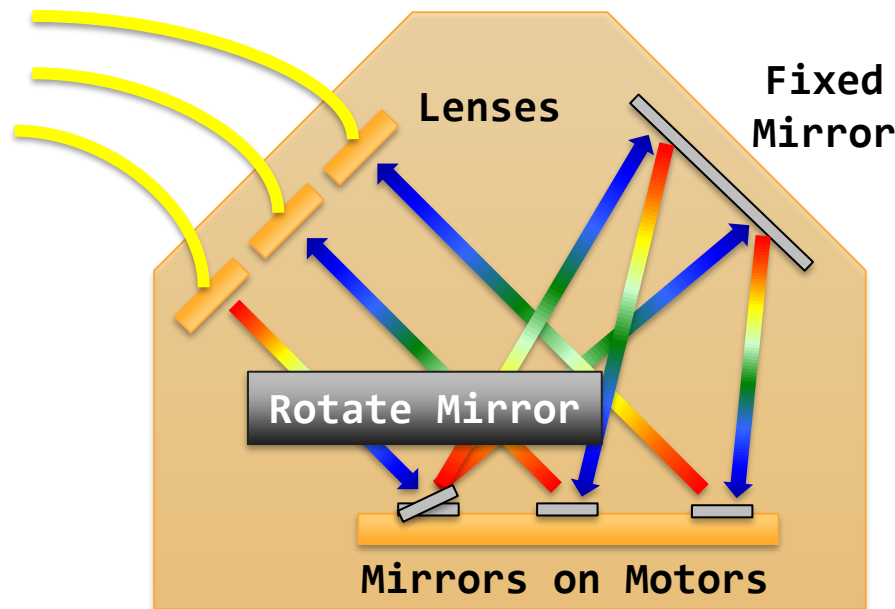
Optical Circuit Switch (OCS)

By Nathan Farrington, SIGCOMM 2010

Example

Optical Circuit Switch

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



**A
Matching!**

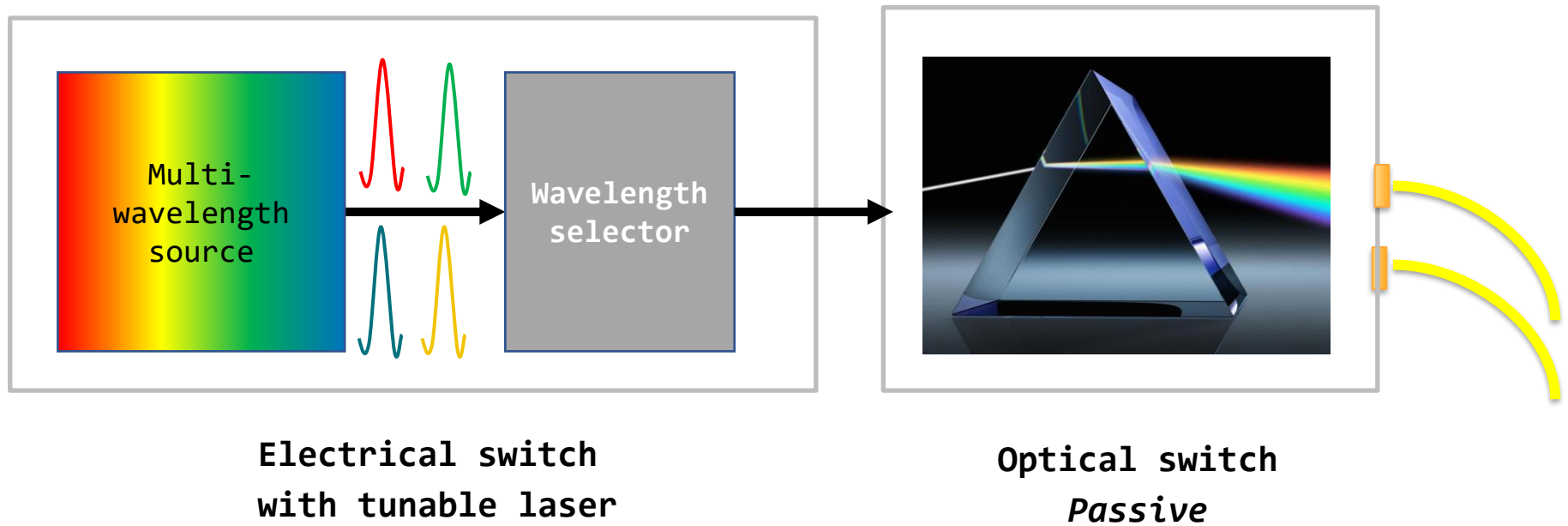
Optical Circuit Switch (OCS)

By Nathan Farrington, SIGCOMM 2010

Another Example

Tunable Lasers

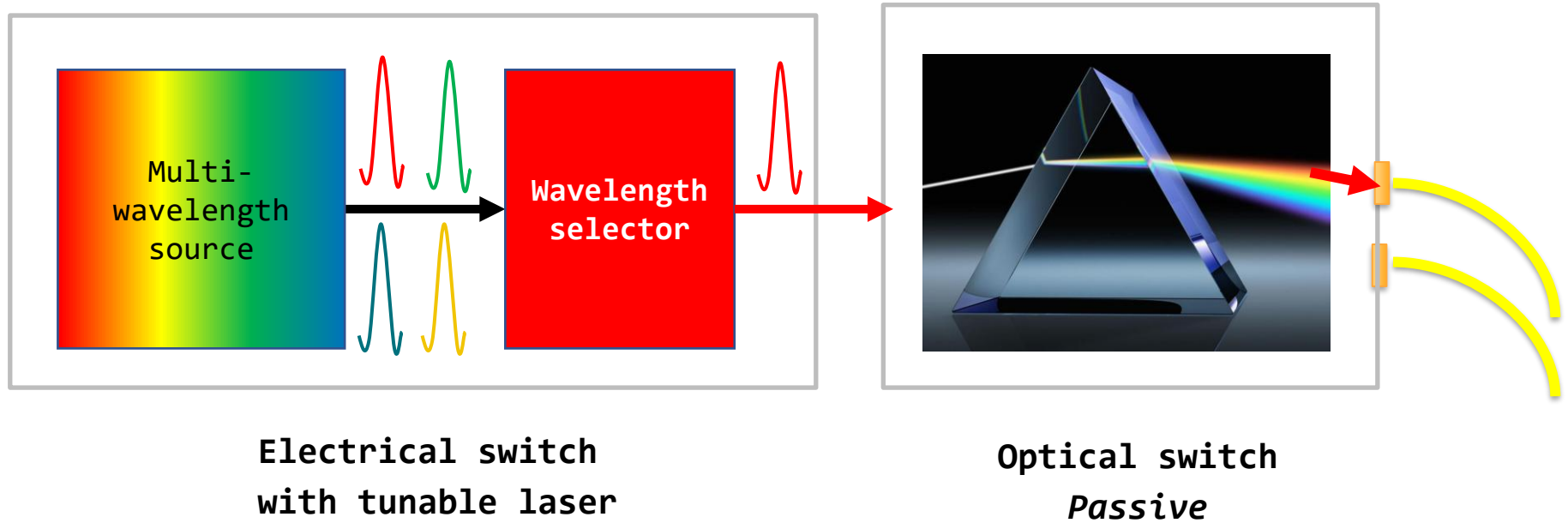
- Depending on wavelength, forwarded differently
- Optical switch is passive



Another Example

Tunable Lasers

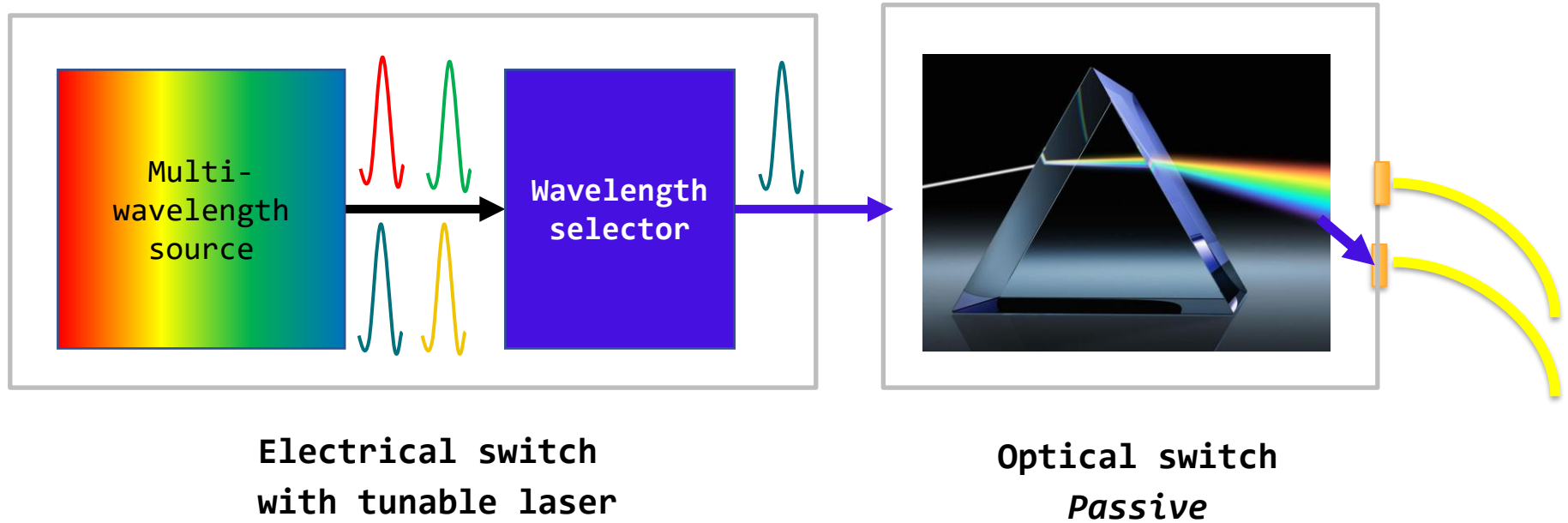
- Depending on wavelength, forwarded differently
- Optical switch is passive



Another Example

Tunable Lasers

- Depending on wavelength, forwarded differently
- Optical switch is passive

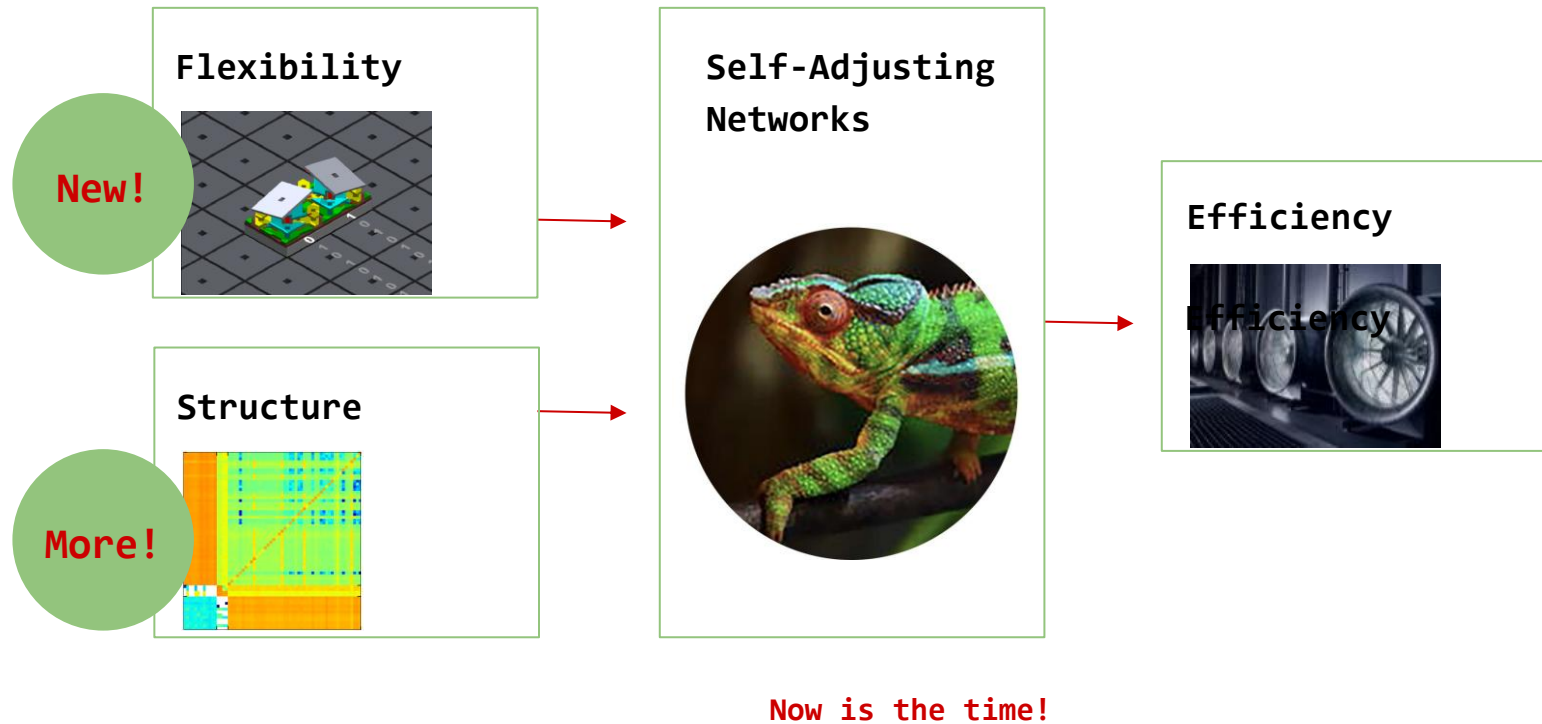


First Deployments

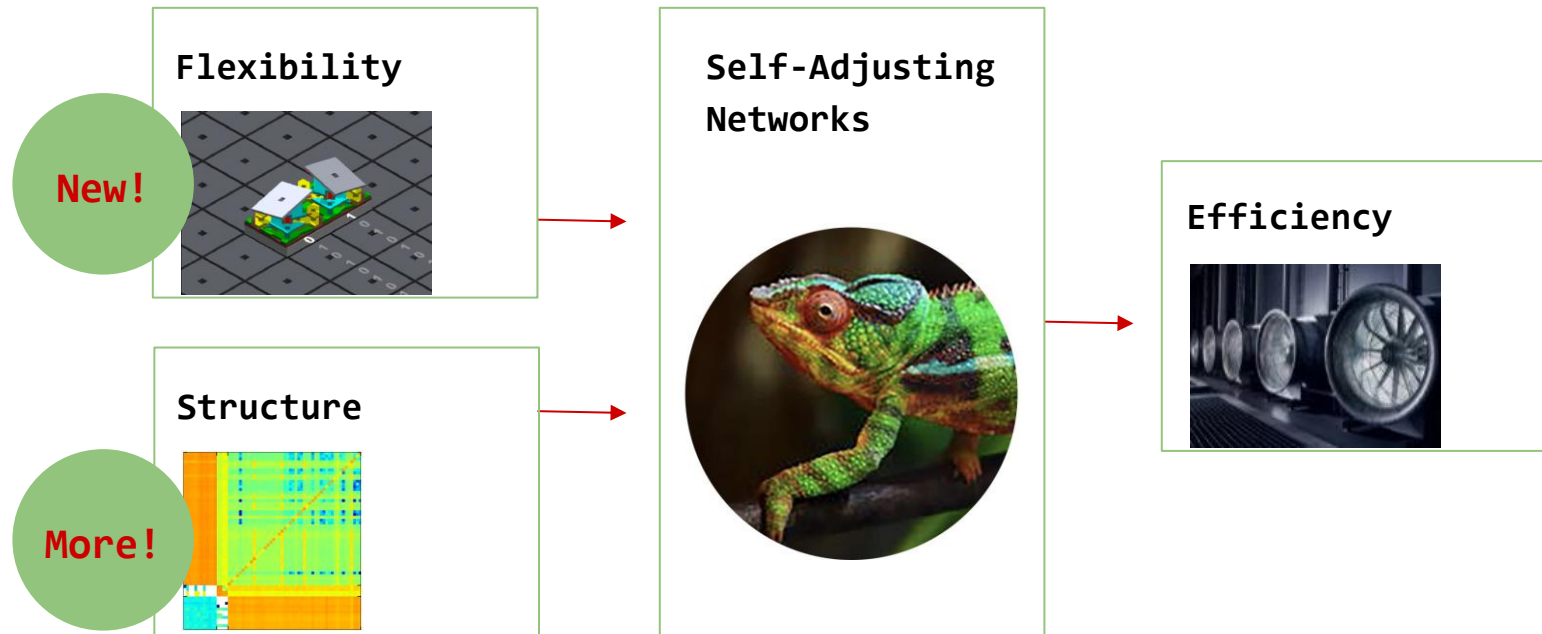
E.g., Google's Datacenter Jupiter



The Big Picture



The Big Picture



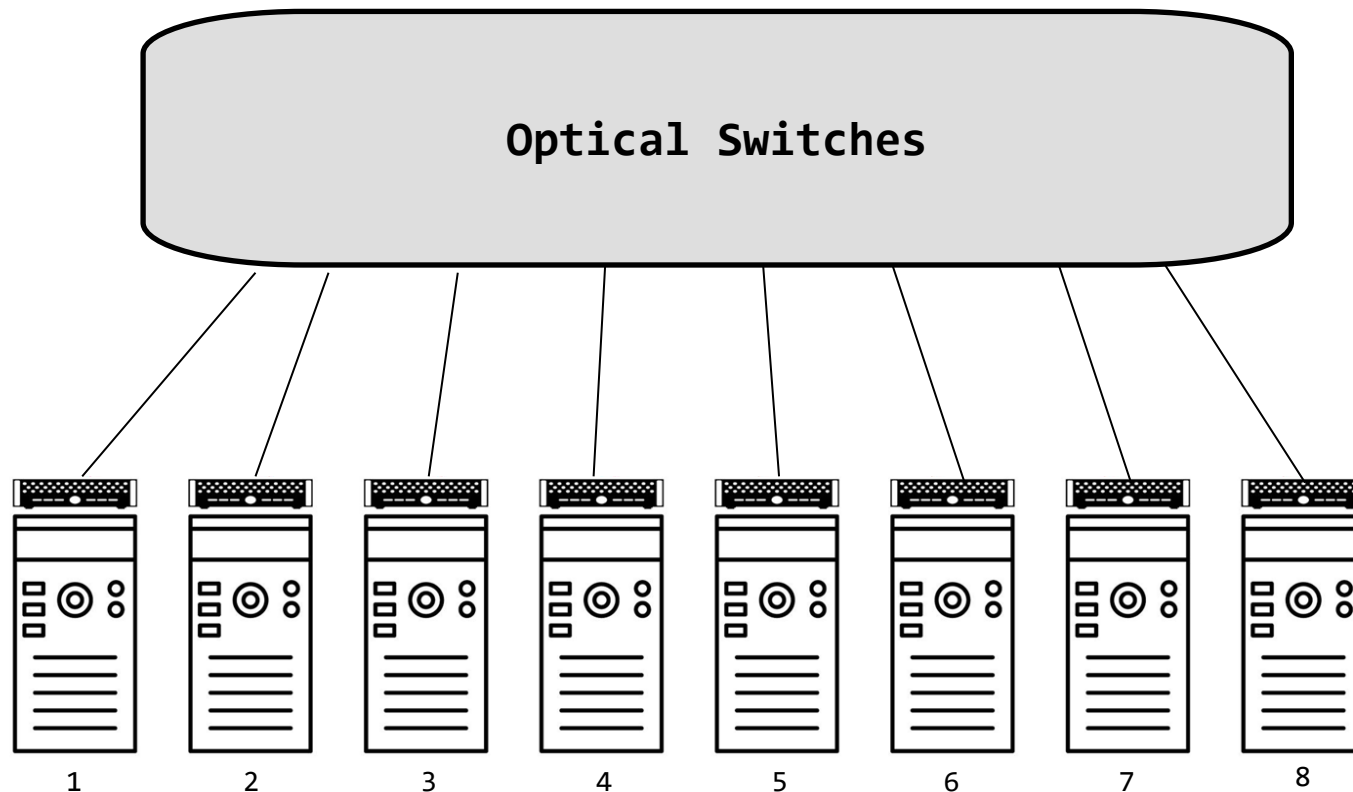
Now is the time!

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

Self-Adjusting Networks: Mental Model and Taxonomy

Reconfigurable Datacenters Based on OCSs:

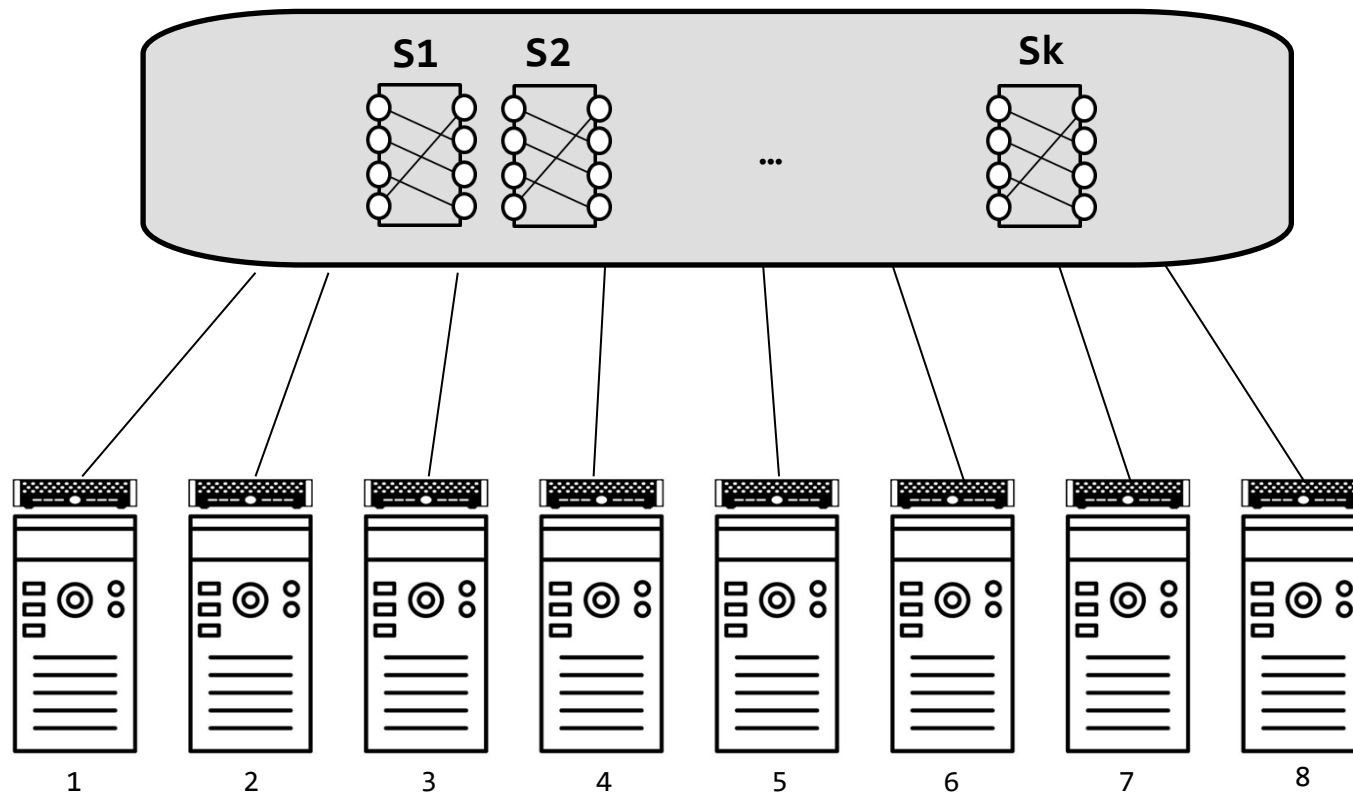
It's about Dynamic Matchings



→ Racks connected to **optical fabric**
(often in addition to an electrical fabric: **hybrid**)

Reconfigurable Datacenters Based on OCSs:

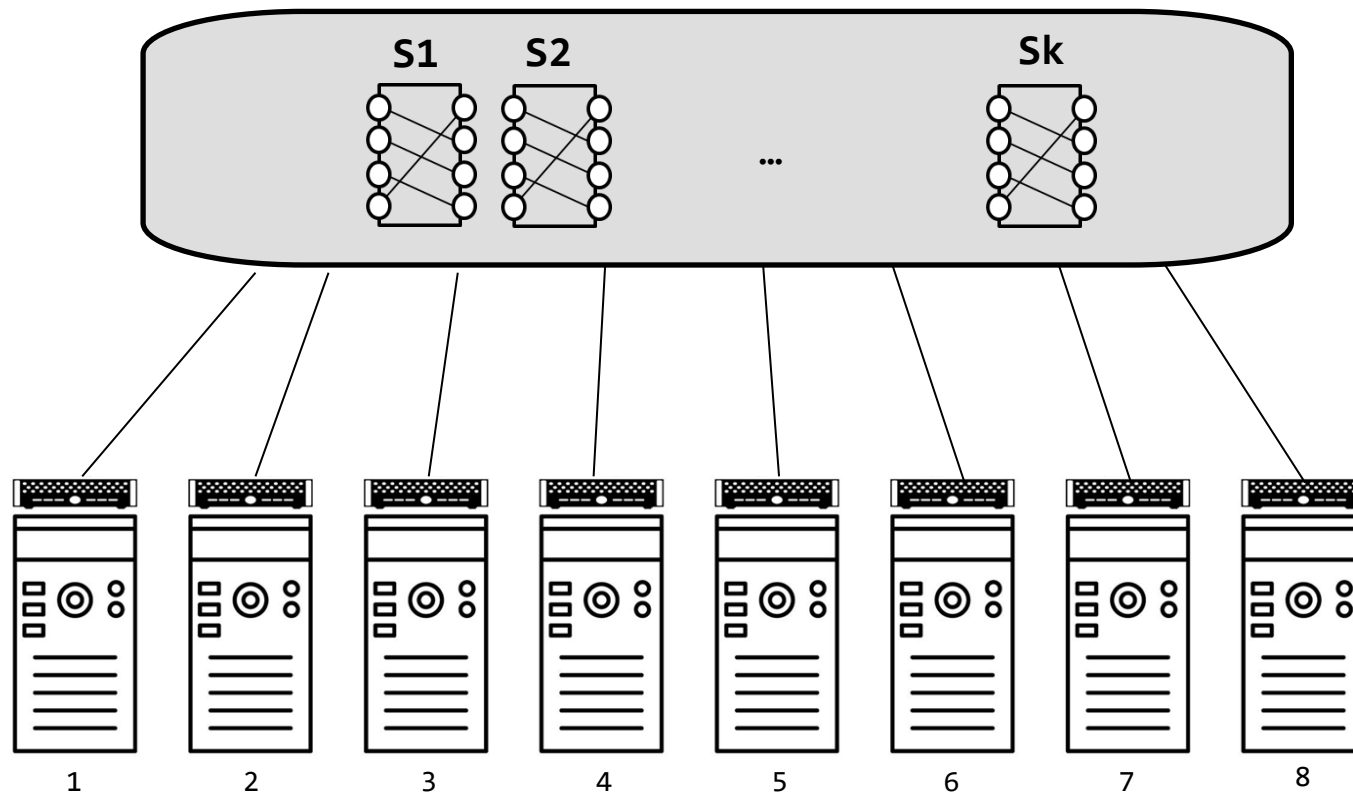
It's about Dynamic Matchings



→ Specifically: ToR switches connected to **set of k OCSs**
(often in addition to an electrical switches if **hybrid**)

Reconfigurable Datacenters Based on OCSs:

It's about Dynamic Matchings

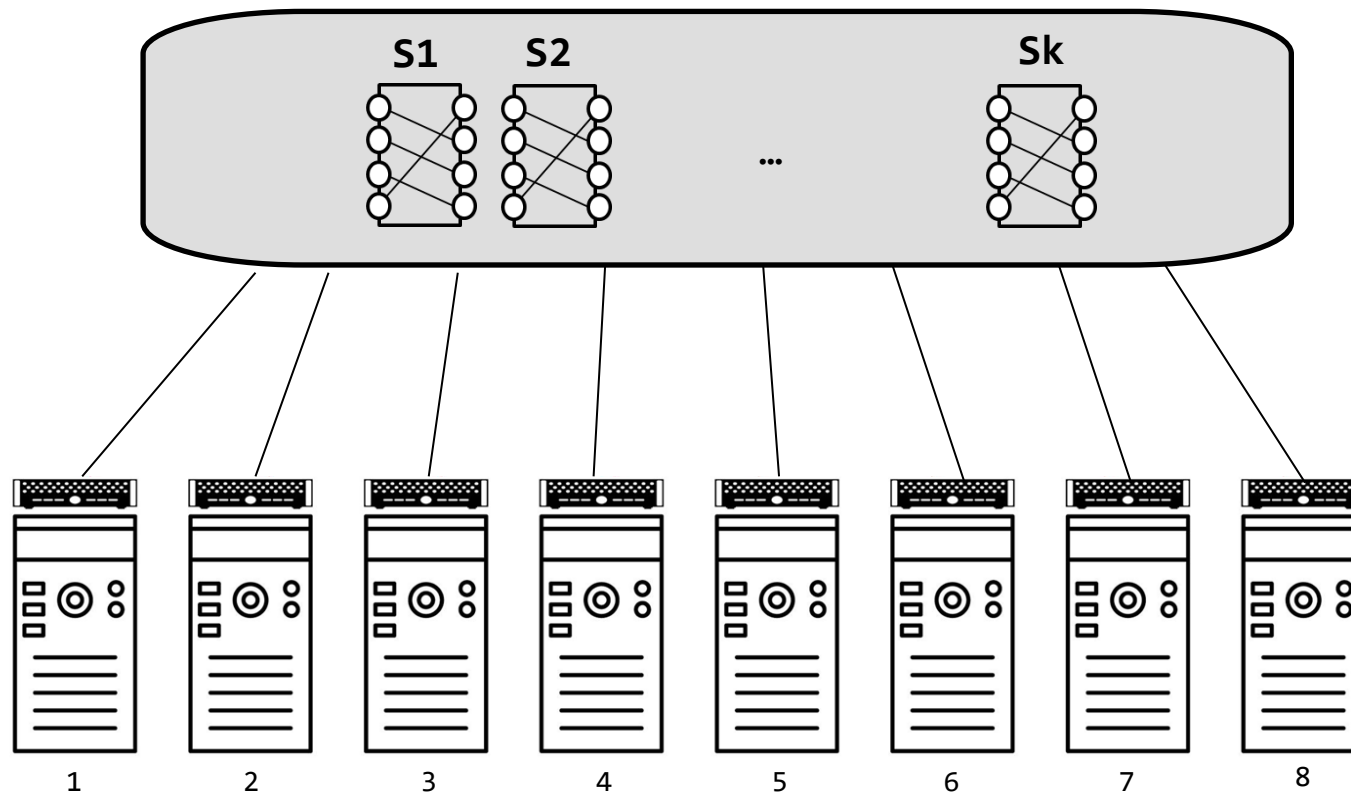


→ Recall: each OCS provides a *matching*

→ aka. *ToR-Matching-ToR (TMT)* model

Reconfigurable Datacenters Based on OCSs:

It's about Dynamic Matchings

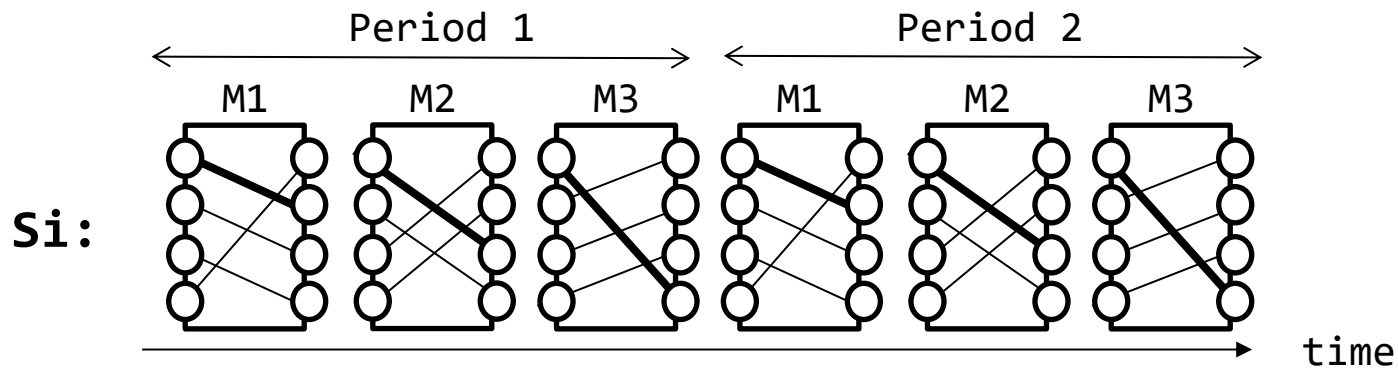


- It's **dynamic**: Each OCS S_i can **reconfigure** over time
- Two flavors: **demand-aware** vs **demand-oblivious** (periodic)

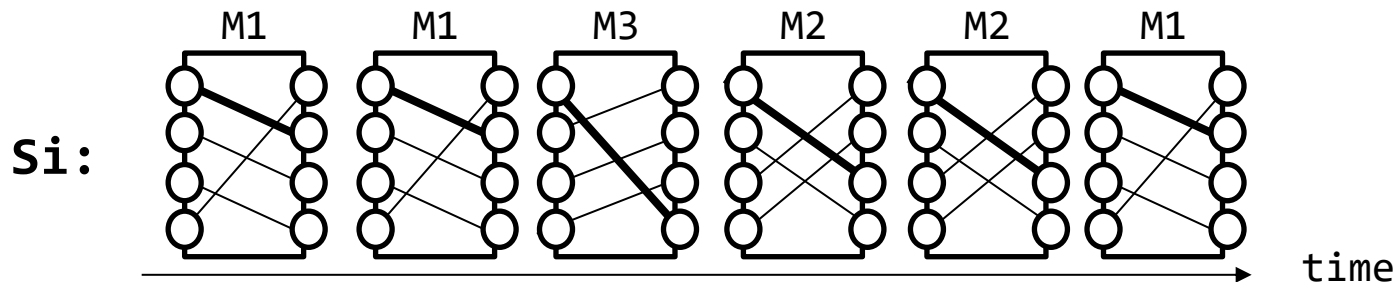
Two Types of Reconfigurable Optical Switches

Demand-Aware Demand-Oblivious

Demand-oblivious: OCS provides **periodic** matchings



Demand-aware switch: **arbitrary/optimized** matchings



Taxonomy and Opportunity

Orthogonal topology choices:

→ demand-**oblivious** and
demand-**aware**

Demand-
oblivious

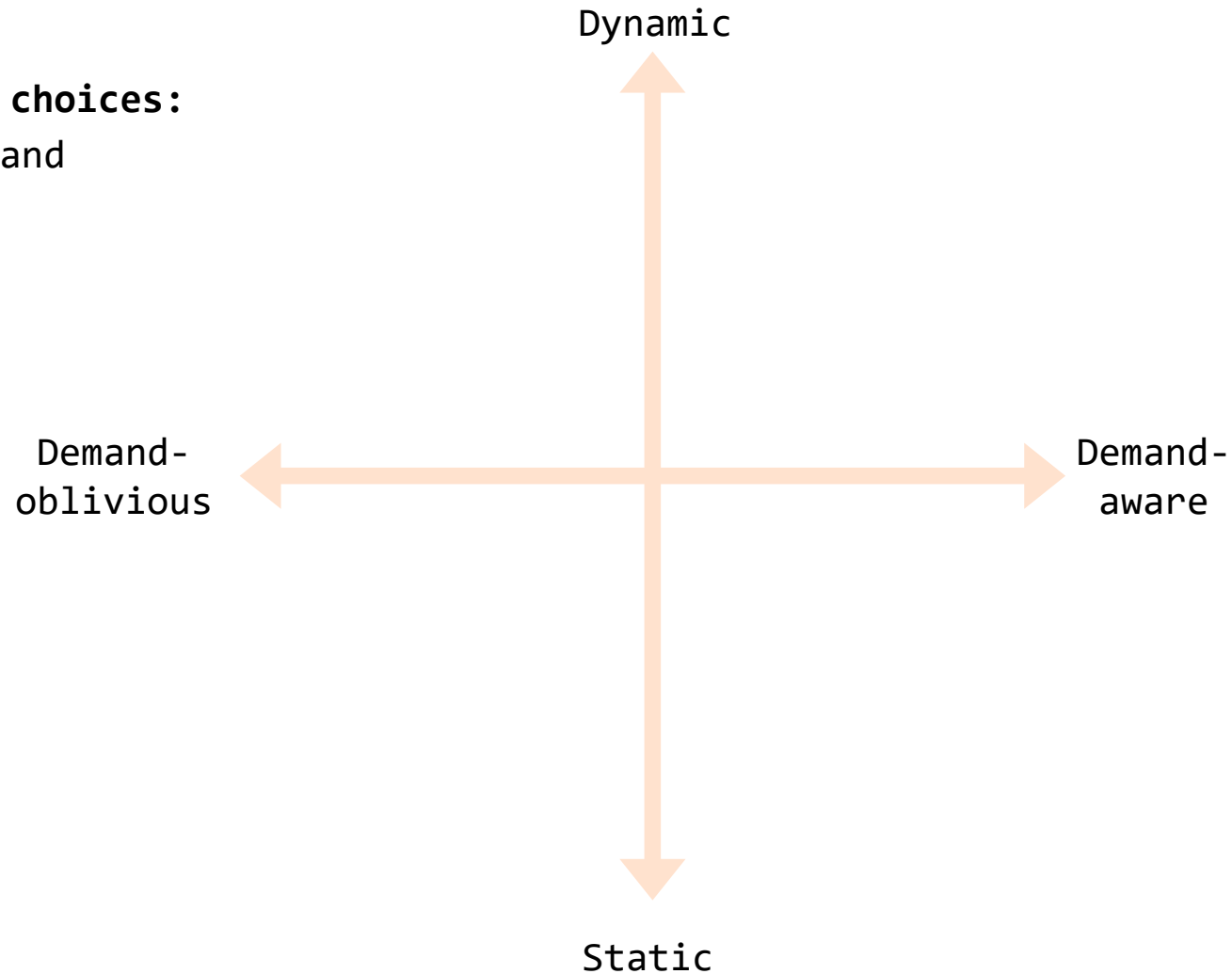


Demand-
aware

Taxonomy and Opportunity

Orthogonal topology choices:

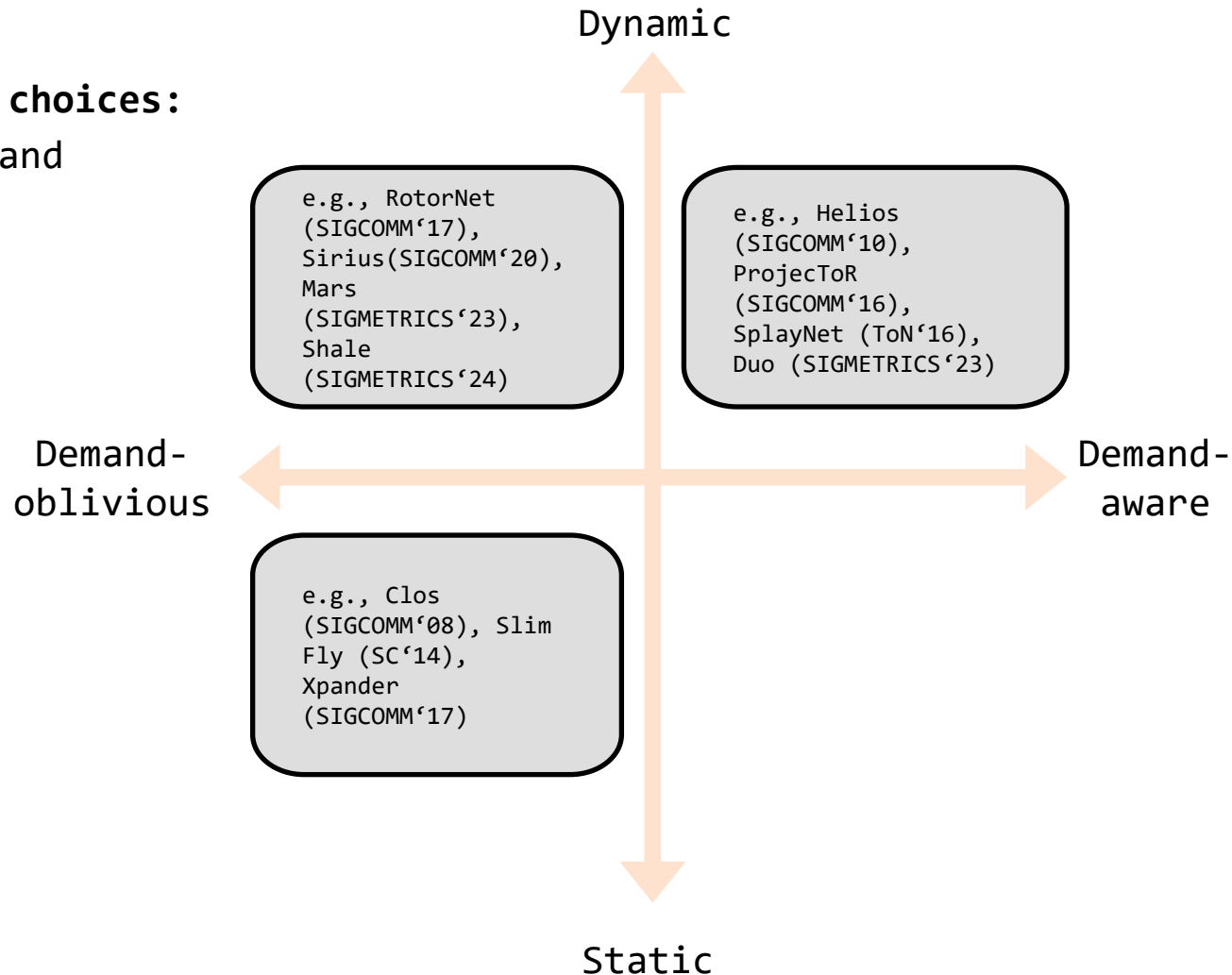
- demand-**oblivious** and demand-**aware**
- static vs dynamic



Taxonomy and Opportunity

Orthogonal topology choices:

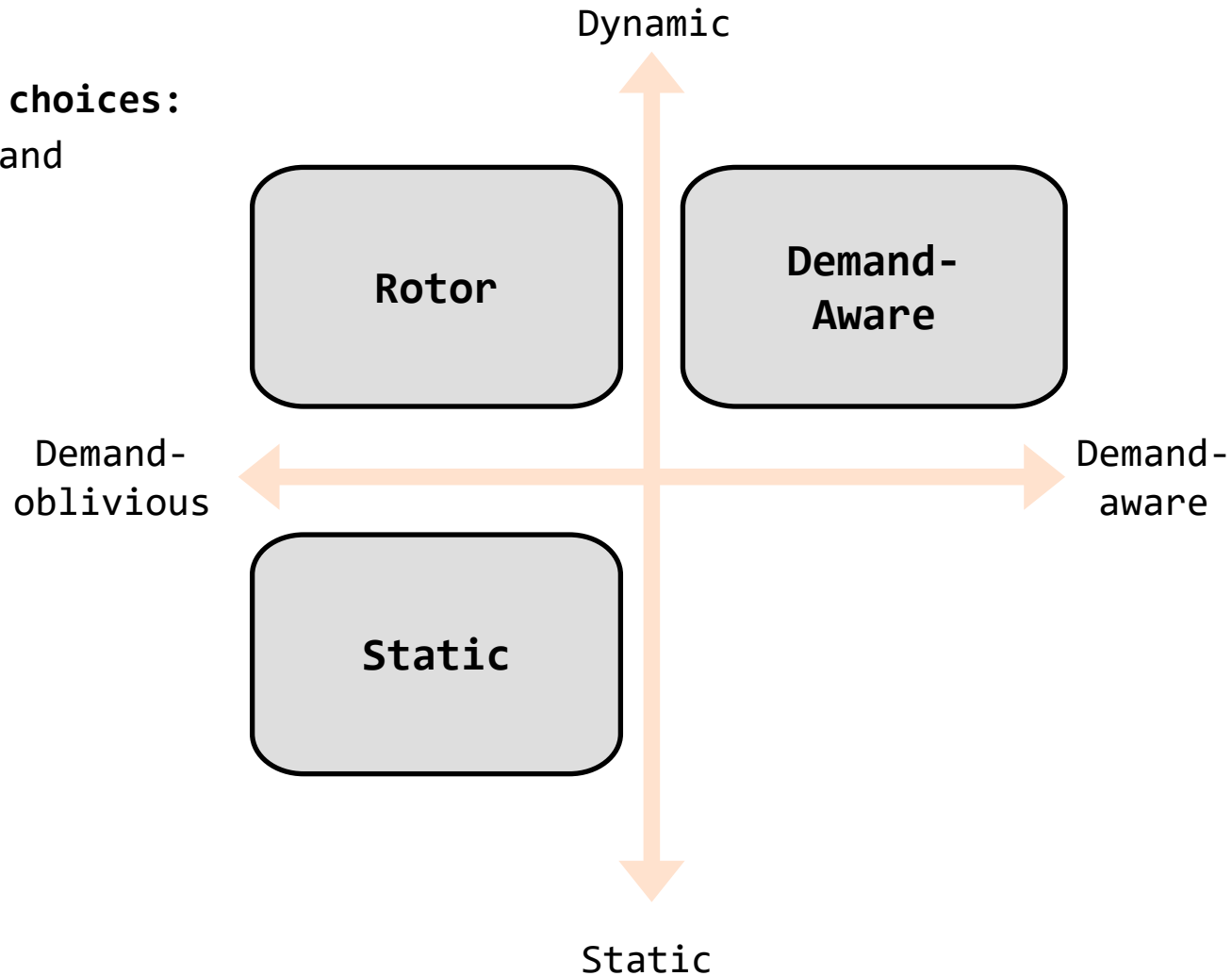
- demand-**oblivious** and demand-**aware**
- static vs dynamic



Taxonomy and Opportunity

Orthogonal topology choices:

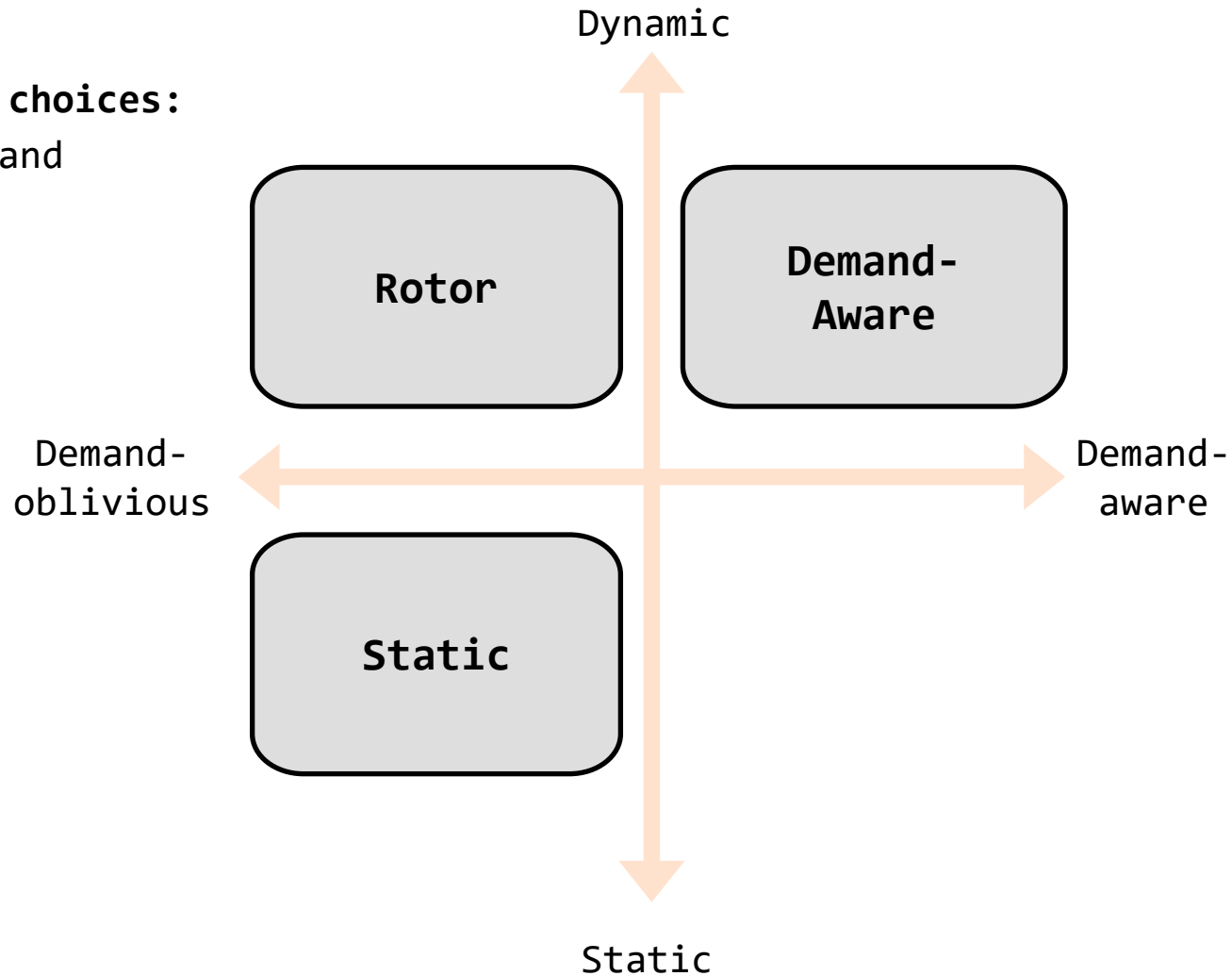
- demand-**oblivious** and demand-**aware**
- static vs dynamic



Taxonomy and Opportunity

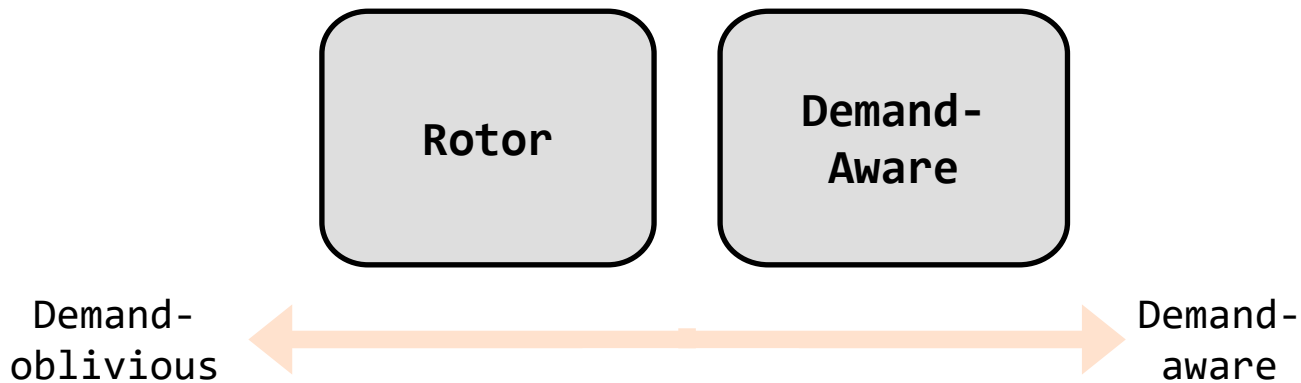
Orthogonal topology choices:

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



Which approach
is best?

A Tradeoff?

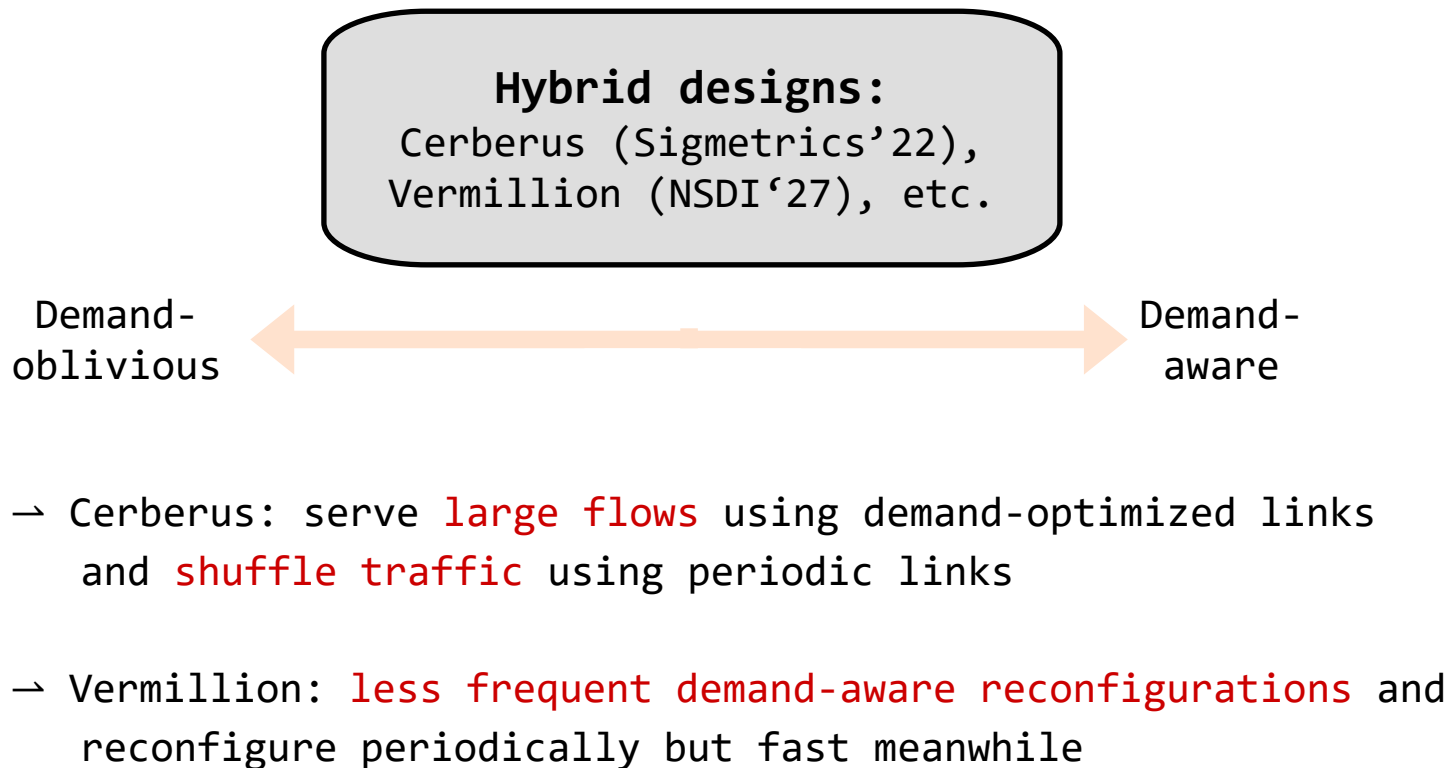


- **Faster** to reconfigure
- Not optimized

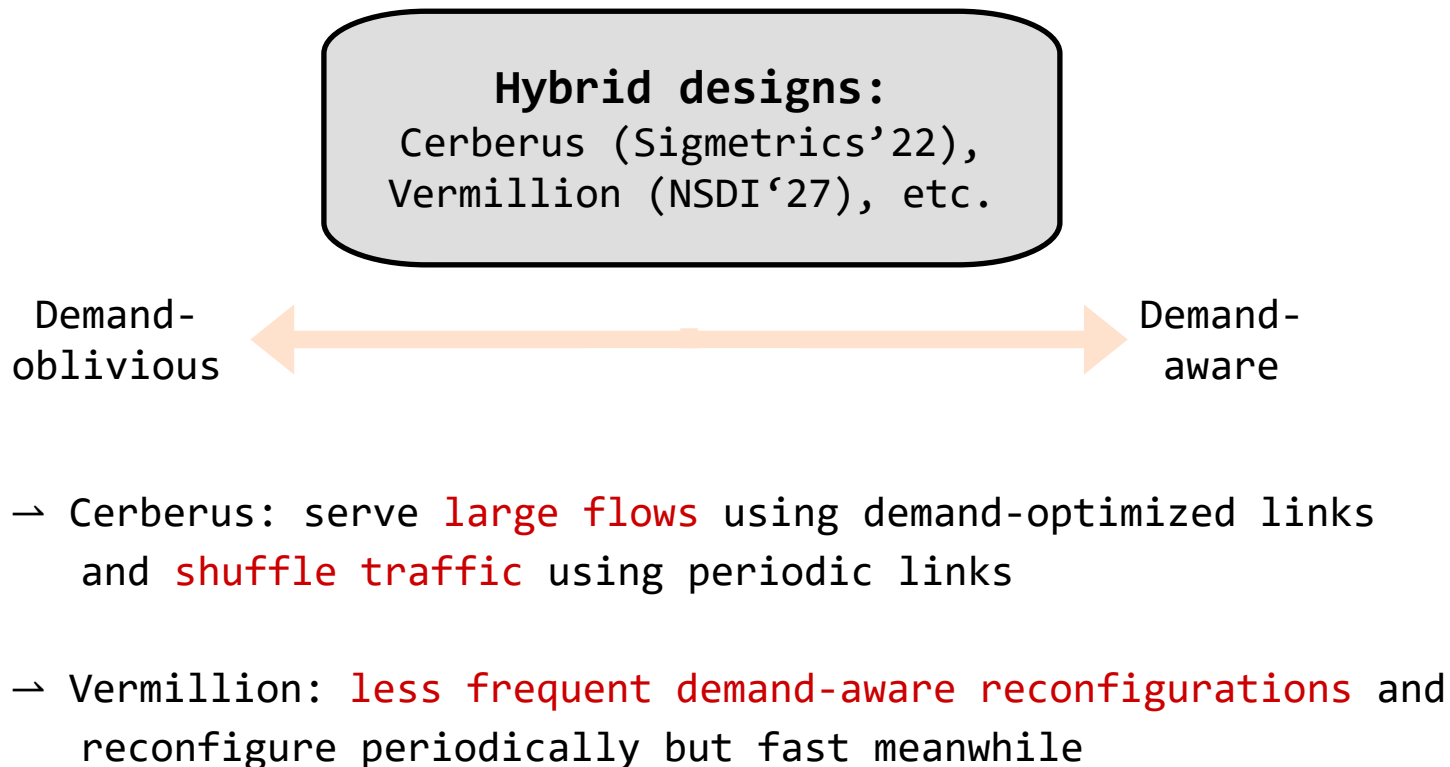
VS

- Slower: optimized
- **Optimized**
- Need **info about demand** (?)

An Opportunity! Hybrid.



An Opportunity! Hybrid.



Further reading:

Vermillion: A Traffic-Aware Reconfigurable Optical Interconnect with Formal Throughput Guarantees
Addanki et al. USENIX **NSDI 2027**.

Vermillion: A Traffic-Aware Reconfigurable Optical Interconnect with Formal Throughput Guarantees
Addanki et al. USENIX **NSDI 2027**.

Why am I telling all this? I believe that

**Self-Adjusting Networks Are
Especially Useful for AI
Workloads**

Collective Communications

- Distributed HPC and ML applications frequently perform very **specific and predictable** communication operations:
AllReduce, Broadcast, All-to-All, ...
- 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 frequently perform very **specific and predictable** communication operations:
AllReduce, Broadcast, All-to-All, ...
- Known as ***Collective Communications***
- Examples: **Nvidia's NCCL**, AMD's RCCL,....
 - Basically reincarnations of MPI
 - "CCL" stands for Collective Communication **Library**

Importantly:

- Collective algorithms realize primitives in **communication phases**
- And communication is **pair-wise** ("***matching***")

Collective Communications

especially relevant for MoE parallelism,
for training and inference

- Distributed HPC and ML applications frequently perform very **specific and predictable** communication operations:
AllReduce, Broadcast, All-to-All, ...

Key Collective Communications

More for data parallelism
and tensor parallelism

, AMD's RCCL, ...

- Basically reincarnations of MPI
- "CCL" stands for Collective Communication **Library**

Importantly:

- Collective algorithms realize primitives in **communication phases**
- And communication is **pair-wise** ("**matching**")

Self-Adjusting Networks and AI: It's a Match!

Demand-aware self-adjusting networks are good when:

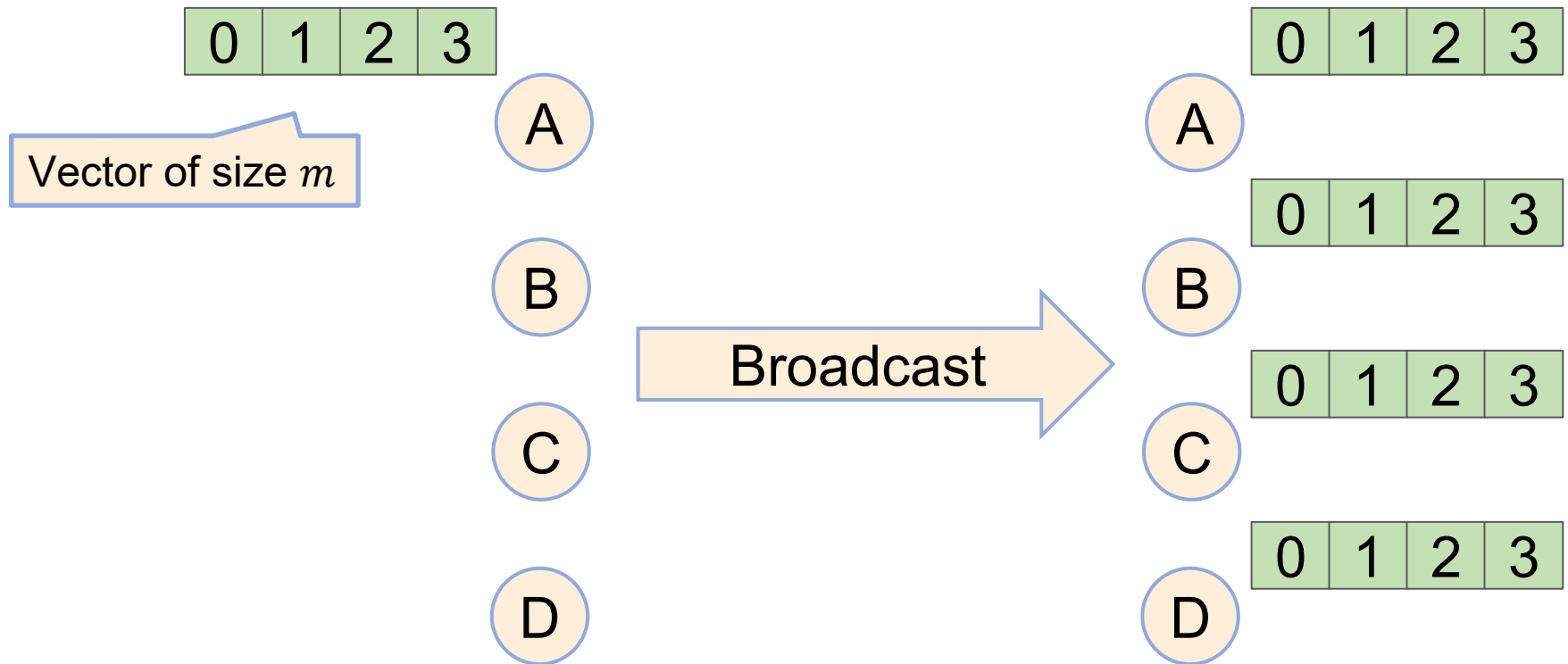
- Demand is **structured** (sparse, skewed, large flows, ...)
- Demand is repetitive or **predictable** (can optimize for it offline)

This is often the case of AI/ML workloads:

- **Collective communication** patterns form **matchings**!
Fits OCSs which provide such matchings.
- Mixture-of-Expert workloads: skewed

Example of a Collective Communication

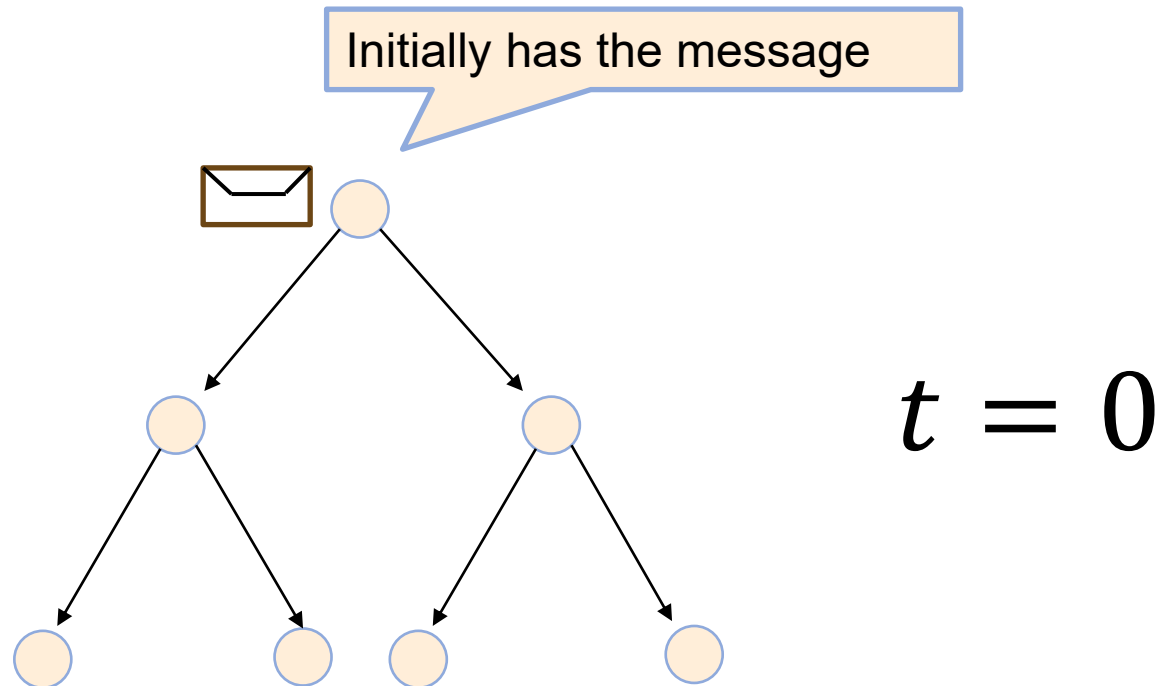
Broadcast



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

Example of a Collective Communication

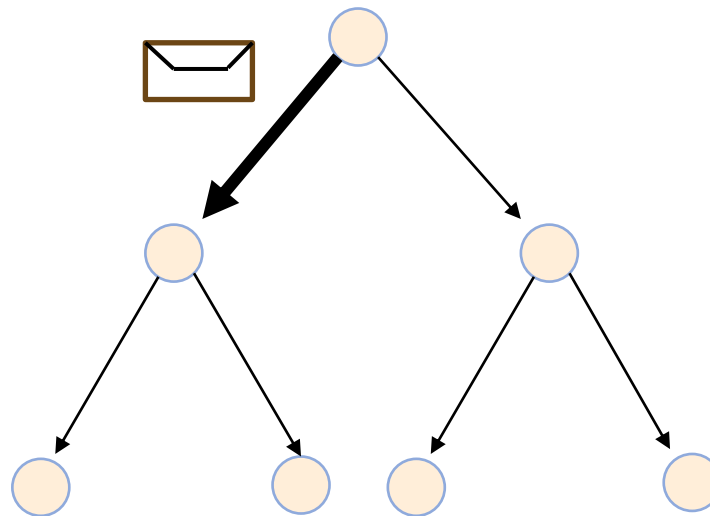
Broadcast Algorithm



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

Example of a Collective Communication

Broadcast Algorithm

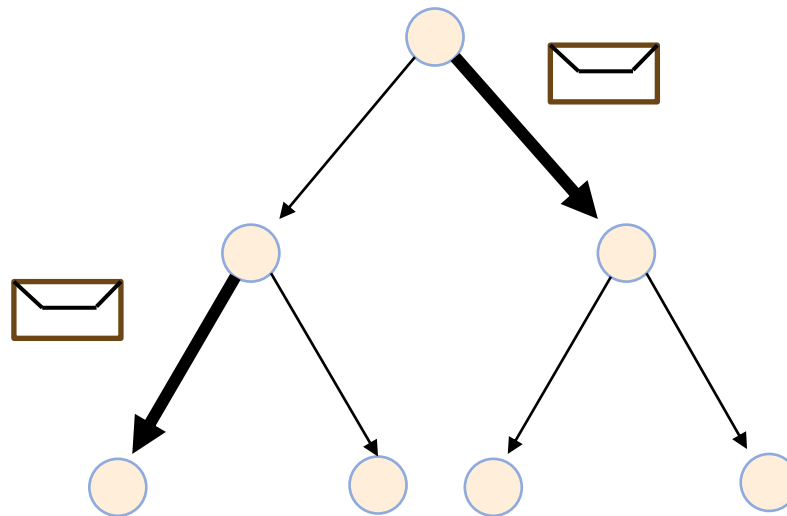


$t = 1$

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

Example of a Collective Communication

Broadcast Algorithm

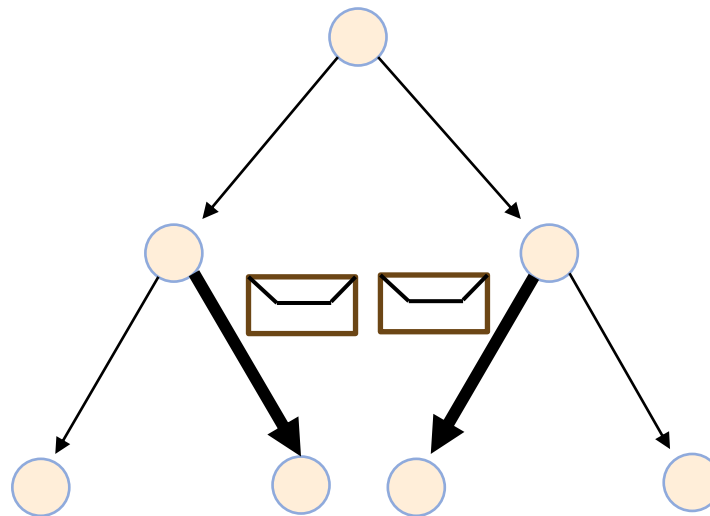


$t = 2$

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

Example of a Collective Communication

Broadcast Algorithm

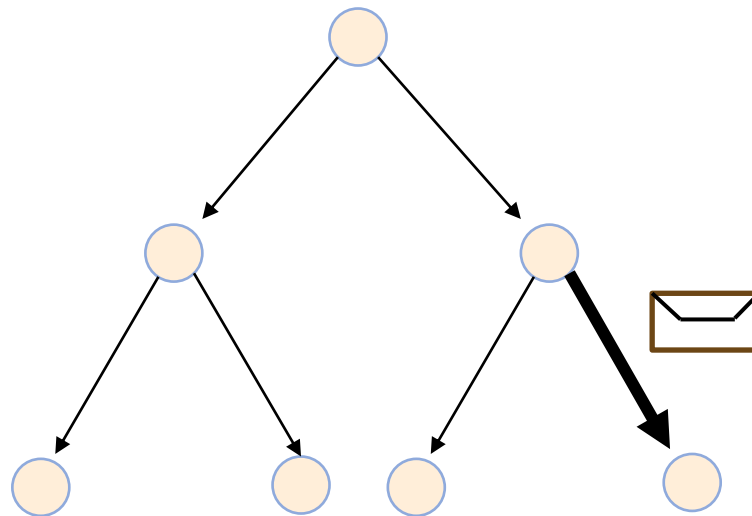


$t = 3$

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

Example of a Collective Communication

Broadcast Algorithm



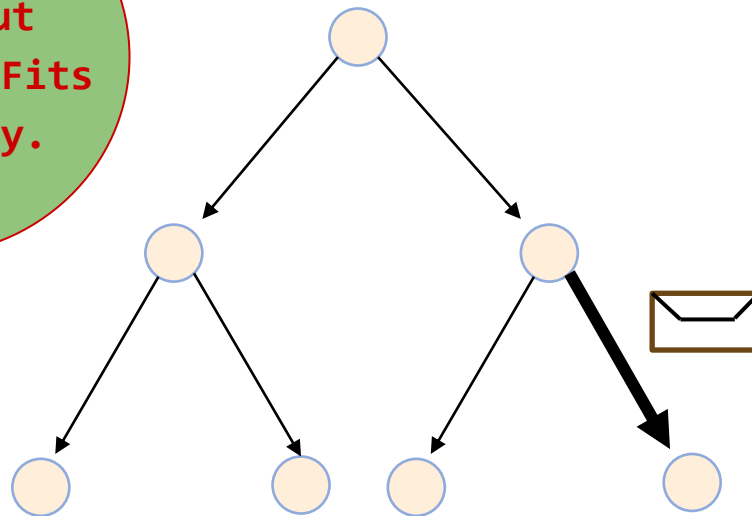
$t = 4$

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

Example of a Collective Communication

Broadcast Algorithm

It's about
matchings! Fits
OCS nicely.



$t = 4$

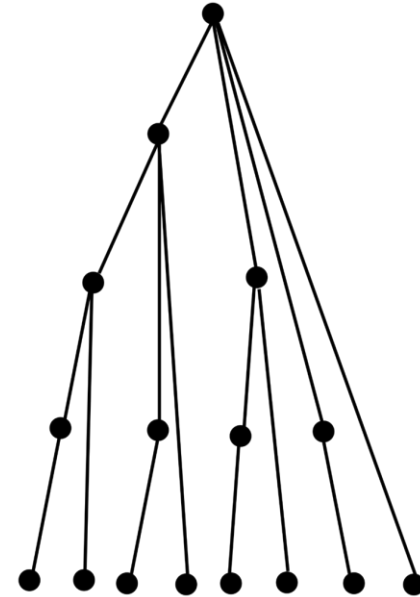
Realization in *collectives algorithm*: *pair-wise* matchings!

Example of a Collective Communication

Broadcast Algorithm

Keep helping in future too!

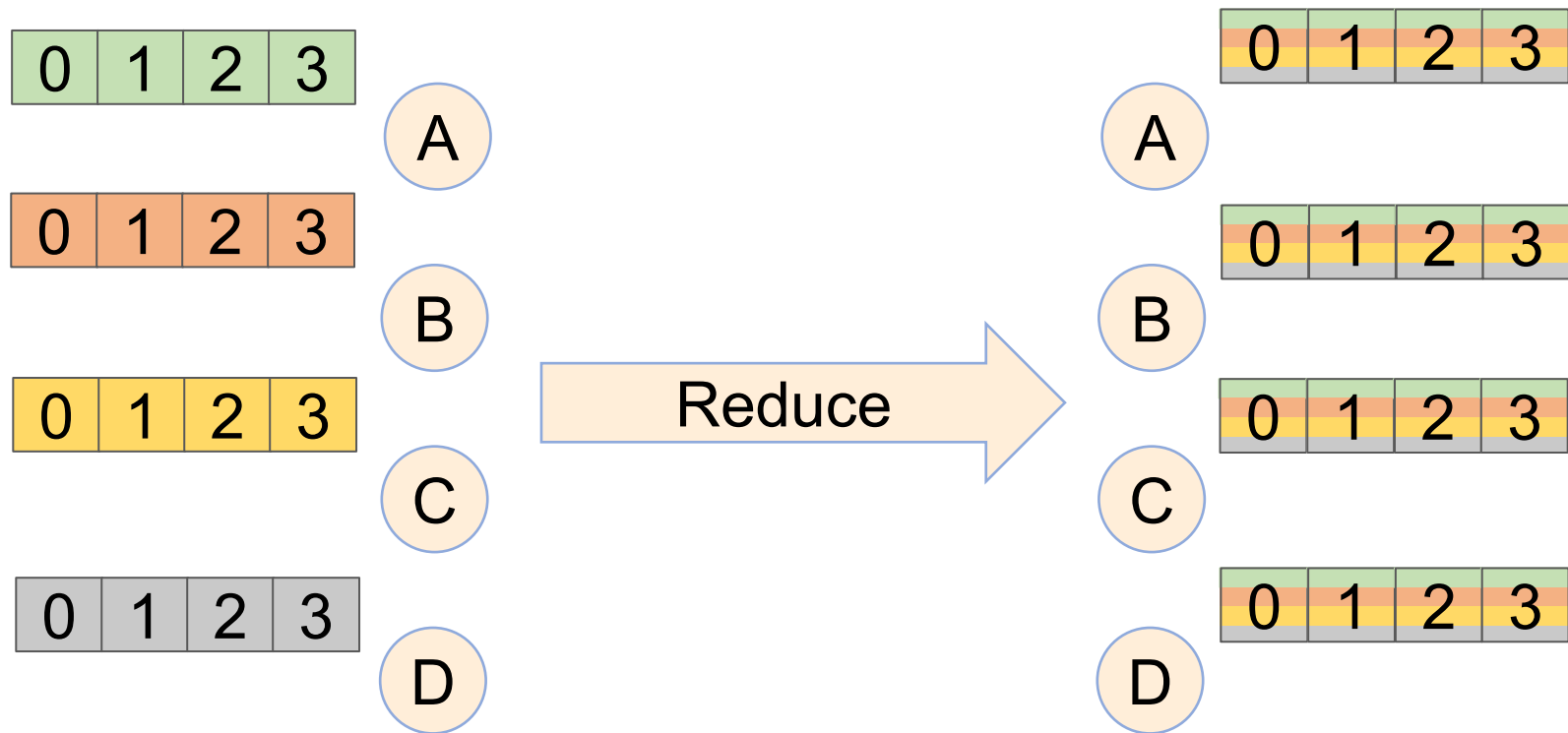
Binomial tree



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

Example of a Collective Communication

AllReduce

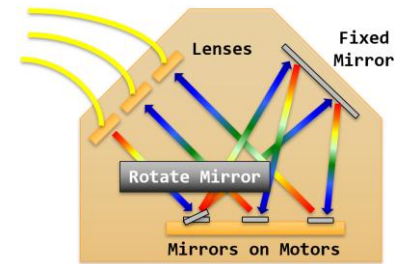
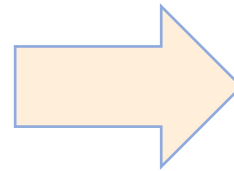
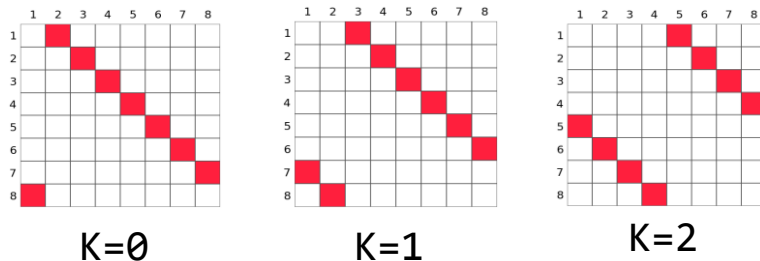


Every node transmits its vector *to every other node*, and *aggregation* (“reduction”) is performed (e.g, sum, min, max).

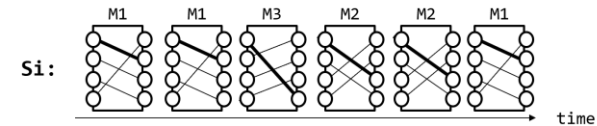
Example of a Collective Communication

All-to-All

- Each node sends a *unique* vector to *every* other node
- Algorithmic realization in *phases=matchings*: *Bruck's algorithm*



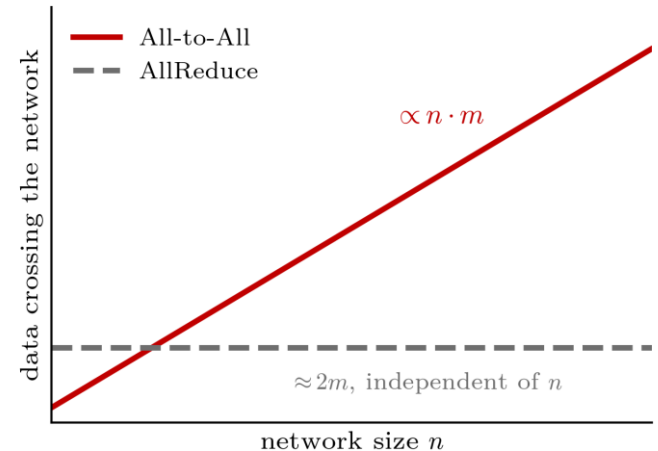
Demand-aware switch: *optimized* matchings



- Related algorithms: e.g., *Birckhoff von Neumann* decompositions: how to serve a demand (matrix) with a sequence of matching?

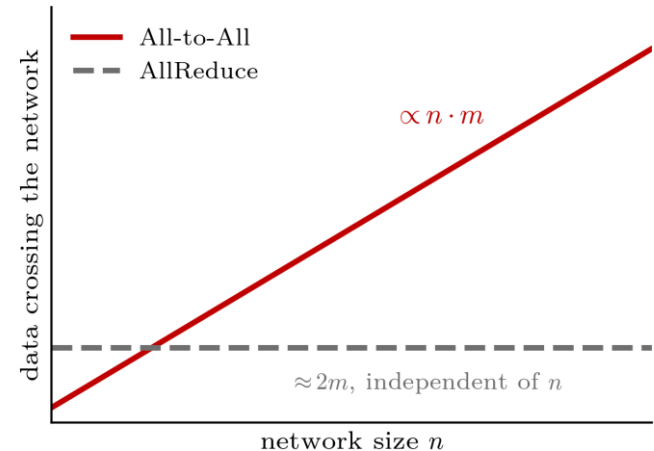
Bruck and OCS: A Match!

- On ring topologies All-to-All communication cost grows linearly in network size while constant for, e.g., AllReduce
- So optical shortcuts beneficial



Bruck and OCS: A Match!

- On ring topologies All-to-All communication cost grows linearly in network size while constant for, e.g., AllReduce
- So optical shortcuts beneficial

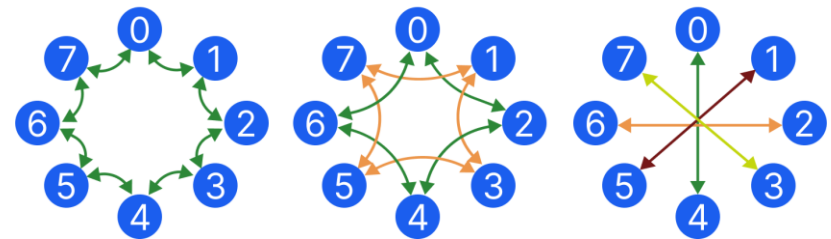


Note: Optimally scheduling collective communications *over a network* is still an open question *even for many fixed topologies* as well! Classic HPC work on scheduling under assumption of *complete network*.

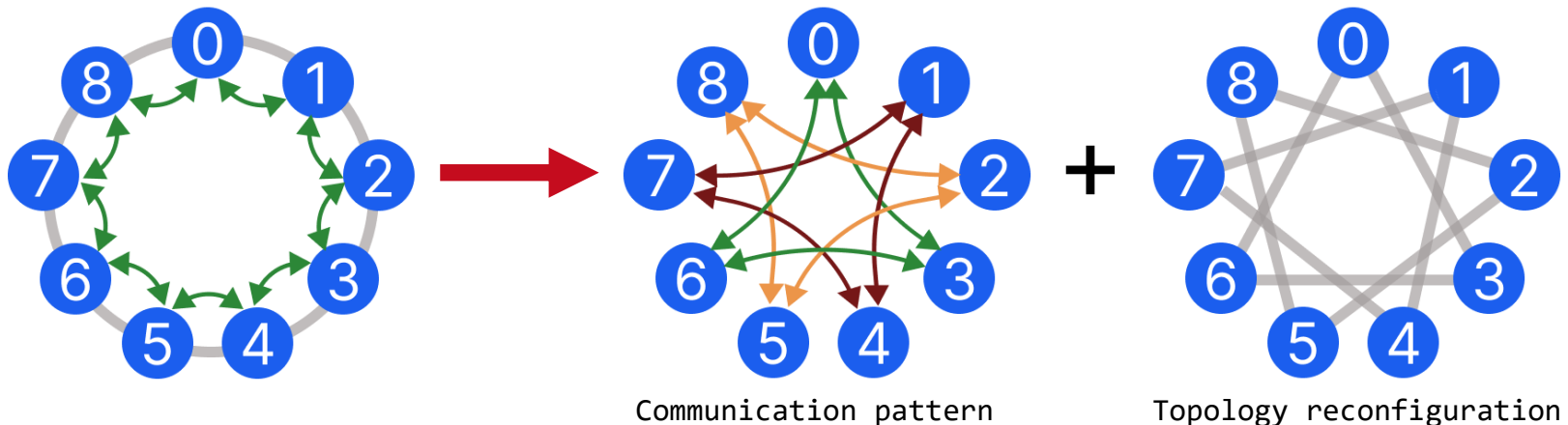
Especially Powerful: Combining the Two Techniques

Reconfigurable+Multiport

- **ReTri**: Reconfigurable Trivance (coming next)
- **Multi-port extension of Bruck** proceeds in **subring** phases



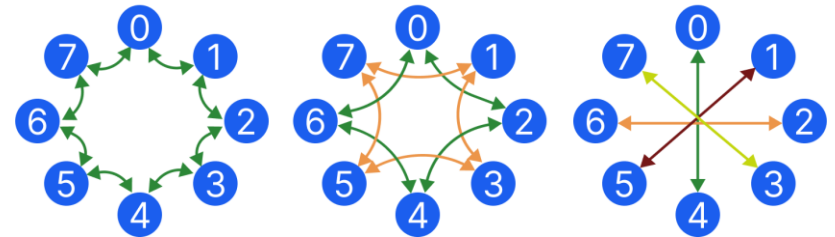
- In **reconfigurable** networks, can build **direct connections** for these demands:



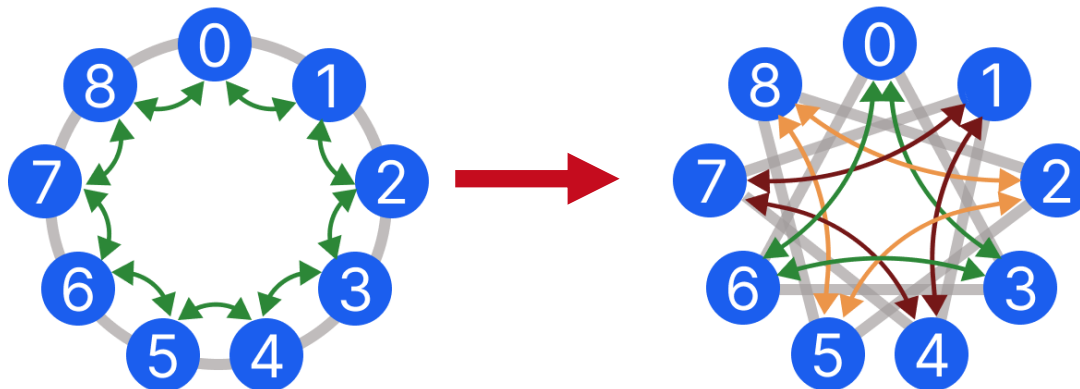
Especially Powerful: Combining the Two Techniques

Reconfigurable+Multiport

- **ReTri**: Reconfigurable Trivance (coming next)
- **Multi-port extension of Bruck** proceeds in **subring** phases



- In **reconfigurable** networks, can build **direct connections** for these demands:

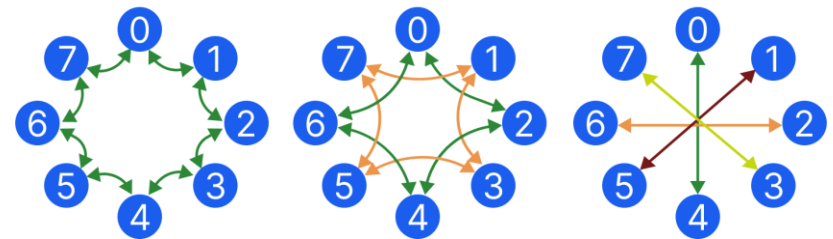


Communication pattern
= Topology

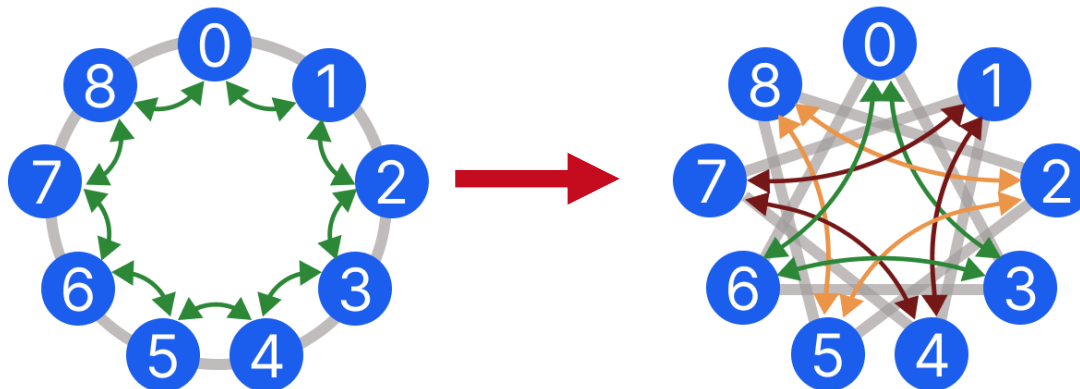
Especially Powerful: Combining the Two Techniques

Reconfigurable+Multiport

- **ReTri**: Reconfigurable Trivance (coming next)
- **Multi-port extension of Bruck** proceeds in **subring** phases



- In **reconfigurable** networks, can build **direct connections** for these demands:



Communication pattern
= Topology

*It's a
Match!*

HotOptics@Sigcomm 2026

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

Anton Jueress
Weizenbaum Institute & TU Berlin
Berlin, Germany
anton.jueress@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 $10\times$ 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-

**The ReTri
paper!**

Particularly Powerful in Hybrid Networks

ACM Sigmetrics 2027

Optimizing Hybrid Reconfigurable Datacenter Networks for All-to-All Collective Communications

ANTON JUERSS, Weizenbaum Institute, Germany and TU Berlin, Germany

MITJA KREBS, TU Berlin, Germany

DARYA MELNYK, TU Berlin, Germany

STEFAN SCHMID, TU Berlin, Germany

All-to-All collective communication is a fundamental and costly primitive that has recently started gaining importance in hybrid-parallel training, where it is applied in models such as Deep Learning Recommendation and Mixture-of-Experts. Advancements in optical networks allow us to tailor network topologies for collective operations. In this work, we study how a reconfigurable optical switch—providing a matching among nodes—can be leveraged to accelerate All-to-All communication on ring topologies. Prior work for hybrid network settings has largely settled on two extremes: either never reconfiguring the topology; or reconfiguring before every communication step, where the topology is optimized for one communication step only. We explore the design space between these extremes, in which the optical matching is reconfigured opportunistically during the collective—infrequently enough to amortize the reconfiguration overhead, yet strategically enough to reduce communication distance for future steps.

We present two complementary reconfiguration approaches, one dynamic, one static (one-shot), both of which run Bruck’s algorithm—a classical $\lceil \log_2 n \rceil$ -step All-to-All collective algorithm—on the augmented

Reconfigurations Also Help Other Collectives

ACM MobiHoc 2026

BRIDGE: Optimizing Collective Communication Schedules in Reconfigurable Networks with Reusable Subrings

Anton Jueress

Weizenbaum Institute & TU Berlin
Germany

Stefan Schmid

TU Berlin & Weizenbaum Institute
Germany

Abstract

Optical circuit-switched networks have emerged as an appealing alternative to electrical fabrics as they can reconfigure the network topology at runtime, reducing communication cost and improving bandwidth utilization. Yet exploiting optical reconfigurable networks for collective communication comes with a fundamental trade-off: each reconfiguration incurs non-negligible delay, communication must pause while the fabric reconfigures, and the benefit of a new topology depends on future traffic. The central question is therefore when reconfiguration is worth its cost. While prior work has demonstrated the benefits of reconfiguration, existing

depends critically on collective communication among accelerators. Inter-node communication becomes increasingly important, making network interconnects a major bottleneck for communication-intensive collectives such as AllReduce and All-to-All [17].

The design of datacenter interconnect fabrics therefore plays a central role in the scalability and efficiency of large-scale ML systems [10, 14, 24]. Whereas electrically switched networks are power-intensive and may lead to performance bottlenecks, optical networks have emerged as a promising opportunity to reconfigure network topology at runtime, enabling direct-connect topologies that significantly

Another attractive fabric capability:

Leveraging simultaneous
communication ports for AI

Motivation: Consider

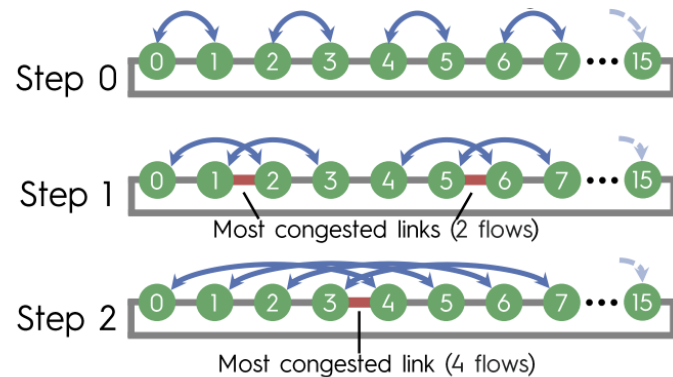
Recursive Doubling (RD)

- **Recursive doubling**: a fundamental **algorithmic technique** for several collective communication operations
- Idea: processes exchange their current data with partners at **exponentially increasing distances**
- Known for low **latency**: for n processes, takes **$\log n$ rounds**
- Can be used for: **AllReduce**, AllGather, etc.

However, RD inefficient on high-diameter networks:

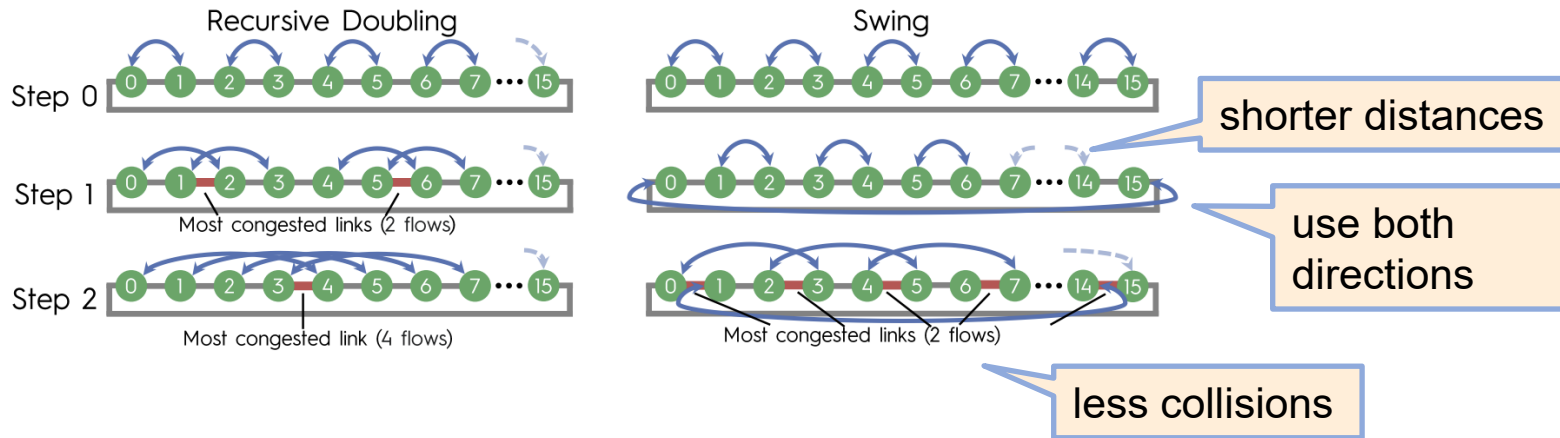
RD on the Ring

Recursive doubling on ring:
Communicate at distances
1, 2, 4, 8, etc.



- **Inefficiency 1:** Multiple pairs share physical links in same step: **congestion**
- **Inefficiency 2:** While some other physical links **not used** for propagation

De Sensi et al. at NSDI 2024 presented an Improvement: Swing*



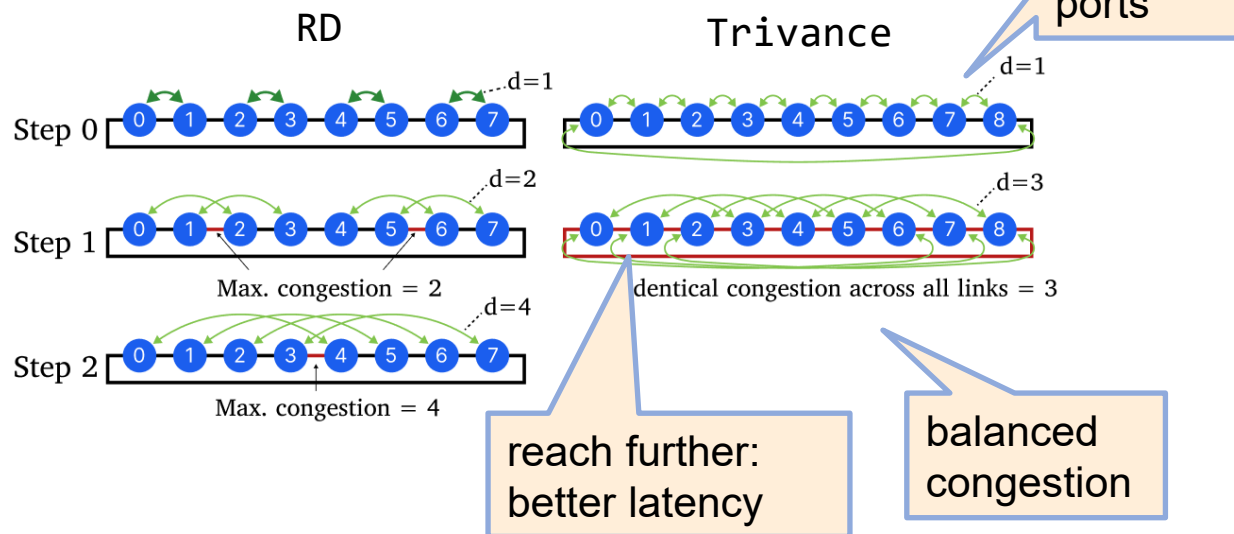
- Idea to **reduce collisions**: Communication in both directions at **shorter distance** (focus: AllReduce)
- But still **same Latency**: identical number of steps and transferred data

*Swing: Short-cutting Rings for Higher Bandwidth Allreduce

Can we reduce number of steps further?

Latency-Optimal: Trivance

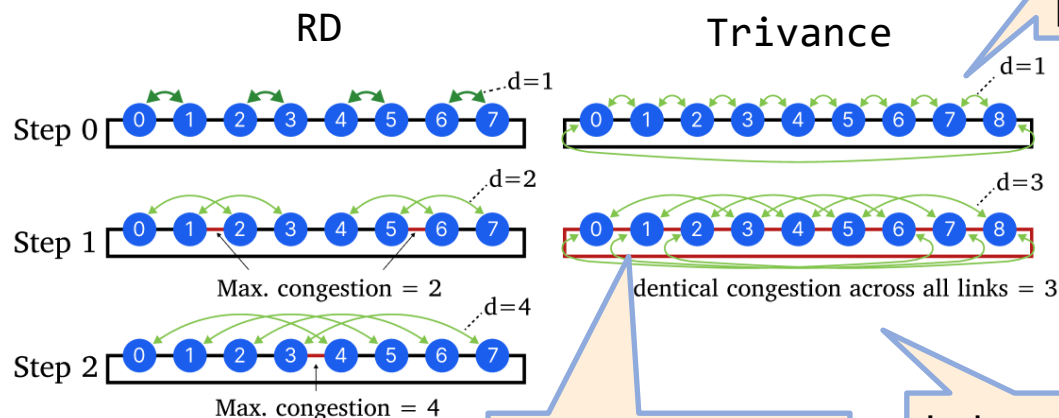
- Idea: **avoid idle** ports/links, **use ports simultaneously** and communicate **bidirectionally** in each step
- Each node reduces data **from both incoming communications**
- Data can be **tripled** per step, latency reduces from $\log_2(n)$ to **$\log_3(n)$** (58% less)



Can we reduce number of steps further?

Latency-Optimal: Trivance

- Idea: **avoid idle** ports/links, **use ports simultaneously** and communicate **bidirectionally** in each step
- Each node reduces data **from both incoming communications**
- Data can be **tripled** per step, latency reduces from $\log_2(n)$ to **$\log_3(n)$** (58% less)



use both ports

Completes one **round faster!**

And **less congestion**

reach further:
better latency

balanced
congestion

TRIVANCE: Latency-Optimal AllReduce by Shortcutting Multiport Networks

Anton Jueress
Weizenbaum Institute & TU Berlin

Vamsi Addanki
Purdue University

Stefan Schmid
TU Berlin & Weizenbaum Institute

Abstract

AllReduce is a fundamental collective communication operation in distributed computing and a key performance bottleneck for large-scale training and inference. Its completion time is determined by the number of communication steps, which dominate latency-sensitive workloads, and the communication distance affecting both latency- and bandwidth-bound regimes. Direct-connect topologies, such as Google's TPUv4 tori, are particularly prone to large communication distances due to limited bisection bandwidth.

In this paper, we present TRIVANCE, a novel AllReduce algorithm that completes within $\log_3 n$ steps—a 50% improvement in comparison to Swing and Recursive Doubling, while reducing congestion compared to Bruck's algorithm by a factor of three and preserving bandwidth-optimality. TRIVANCE exploits both transmission ports of a bidirectional ring within each step to *triple* the communication distance along both directions simultaneously. By performing joint reductions, TRIVANCE improves both the number of steps and network congestion. We further show that TRIVANCE extends naturally to multidimensional torus networks, retaining its latency advantage while achieving performance comparable to bandwidth-optimal algorithms for large AllReduce sizes.

Our packet-level SST simulation shows that TRIVANCE improves state-of-the-art approaches by 5-30% for AllReduce sizes up to

1 Introduction

Collective communication lies at the heart of many high-performance computing applications and machine learning (ML), both for training and inference. As ML model sizes continue to grow [3, 23, 31], efficient collective communication primitives are critical for minimizing training time and maximizing hardware utilization [5, 23]. Recent research has focused on improving collective communication performance in real networks, spanning general algorithm design [10, 28, 29], synthesizing custom algorithms for specific topologies [5, 22, 30], and building scalable network topologies for large-scale training [19, 36].

AllReduce stands out as a fundamental collective primitive, aggregating and disseminating data such as gradients across nodes during parallel training [9, 18, 29]. AllReduce often dominates in both HPC and deep learning environments where the latter heavily relies on AllReduce for gradient synchronization. In production HPC workloads, measurements on large-scale systems have shown that more than 40% of the total Message Passing Interface (MPI) time can be spent in MPI_AllReduce and MPI_Reduce combined [8, 25, 34].

With the adoption of torus topologies in large-scale GPU clusters [19, 36], optimizing AllReduce for multiport, bidirectional networks is key to minimizing communication overheads [29]. In such

Many Exciting Open Problems

Challenge: Computing Heavy Matchings?

- How to schedule a **maximum amount of traffic/matchings** on optical network?
- Online algorithms, offline algorithms, dynamic algorithms
- Several first results known

DistroMatch: Distributed Disjoint Weighted Matchings in Demand-Aware Reconfigurable Optical Datacenters

Kathrin Hanauer
kathrin.hanauer@univie.ac.at
University of Vienna
Faculty of Computer Science
Research Network Data Science
Vienna, Austria

Sophia Heck
sophia.heck@univie.ac.at
University of Vienna
Faculty
Doctoral !

Stefan Schmid
stefan.schmid@tu-berlin.de
TU Berlin
Berlin, Germany
Fraunhofer SIT

Abstract

Reconfigurable optical circuit switches revolutionize data networks by allowing to adjust the physical topology in a demand-aware manner. These switches directly manage and frequently communicating racks, reducing bandwidth and hence improving throughput. The underlying problem is essentially the NP-hard WEIGHTED k -DISJOINT MATCHINGS problem. Existing efficient solutions to this problem require a centralized controller, which constitutes a scalability

Fast and Heavy Disjoint Weighted Matchings for Demand-Aware Datacenter Topologies

Kathrin Hanauer
Faculty of Computer Science, University of Vienna, Austria

Stefan Schmid
Faculty of Computer Science, University of Vienna, Austria
and TU Berlin, Berlin, Germany

Abstract—Reconfigurable optical topologies promise to improve the performance in datacenters by dynamically optimizing the physical network in a demand-aware manner. State-of-the-art optical technologies allow to establish and update direct connectivity (in the form of *edge-disjoint matchings*) between top-of-rack switches within microseconds or less. However, to fully exploit temporal structure in the demand, such fine-grained

Monika Henzinger
Faculty of Computer Science, University of Vienna, Austria

Jonathan Trummer
Faculty of Computer Science, University of Vienna, Austria

[8]; these technologies allow to enhance existing datacenter networks with reconfigurable optical matchings, e.g., one disjoint matching per optical circuit switch [9]–[15]. These matchings can be adapted towards the traffic demand, exploiting temporal and spatial structure [16]. State-of-the-art technologies allow in principle to change such matchings within

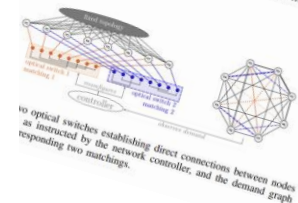
Dynamic Demand-Aware Link Scheduling for Reconfigurable Datacenters

Kathrin Hanauer
Faculty of Computer Science
University of Vienna
Vienna, Austria

Monika Henzinger
Faculty of Computer Science
University of Vienna
Vienna, Austria

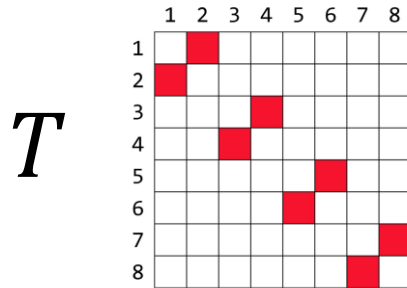
Lara Ost
Faculty of Computer Science
University of Vienna
Vienna, Austria

Stefan Schmid
TU Berlin
Berlin, Germany



Demand-Aware Throughput?

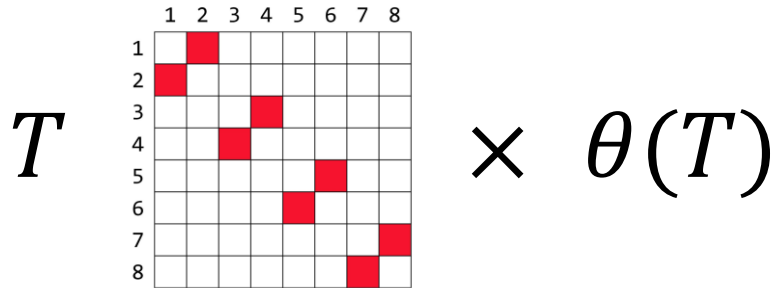
Demand Matrix



Metric: throughput
of a demand matrix...

Demand-Aware Throughput?

Demand Matrix

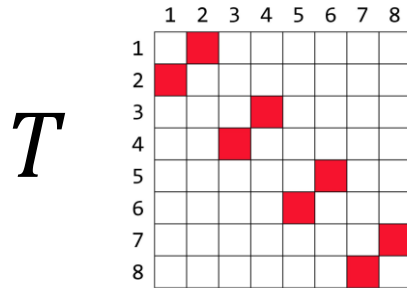


Metric: throughput
of a demand matrix...

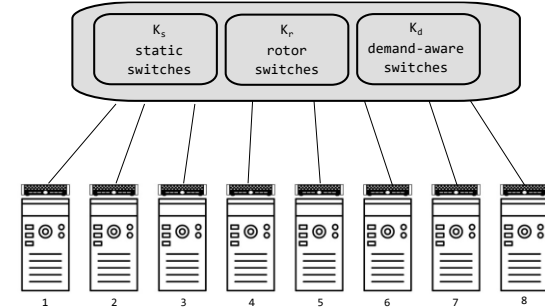
... is the maximal scale
down **factor** by which
traffic is **feasible**
 $0 \leq \theta(T) \leq 1$.

Demand-Aware Throughput?

Demand Matrix



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



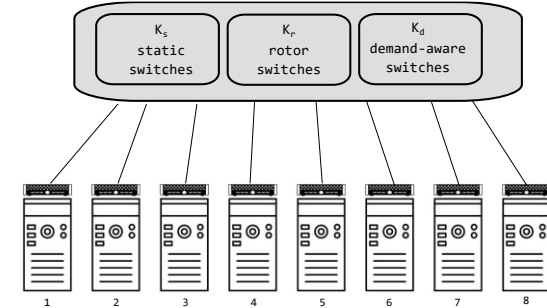
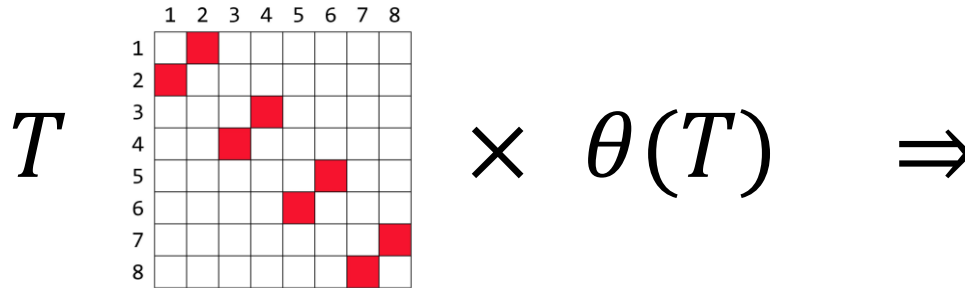
Metric: throughput
of a demand matrix...

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

Throughput of network θ^* :
worst case T

Demand-Aware Throughput?

Demand Matrix



Metric: throughput
of a demand matrix...

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

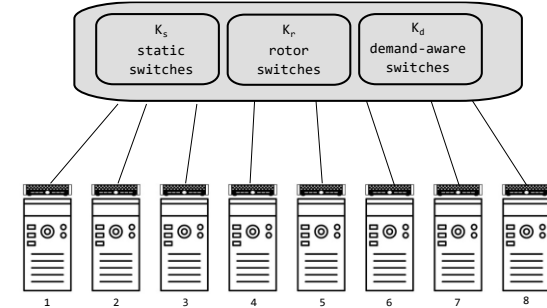
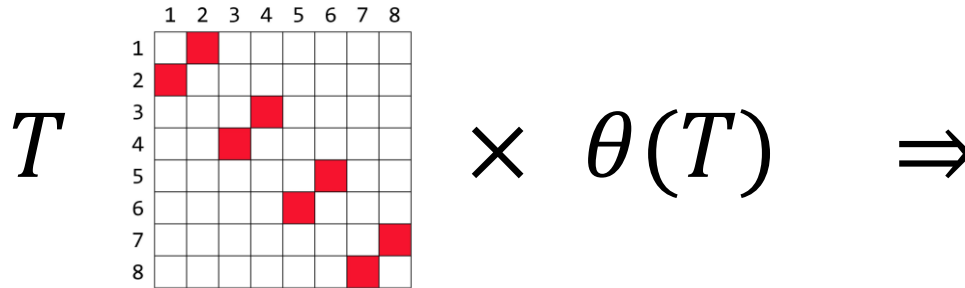
Throughput of network θ^* :
worst case T

**Definition
extends to
dynamic
networks!**

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

Demand-Aware Throughput?

Demand Matrix



Metric: throughput
of a demand matrix...

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

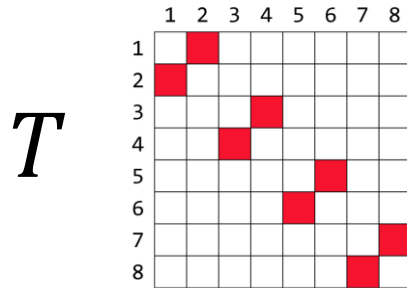
Throughput of network θ^* :
worst case T

**Definition
extends to
dynamic
networks!**

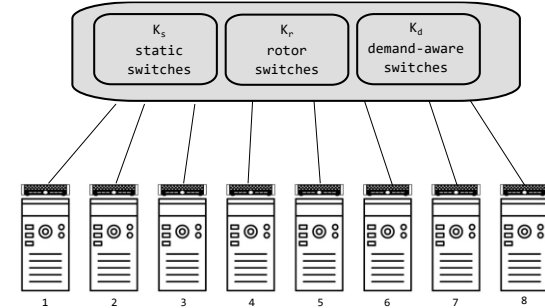
**T typically assumed to be doubly
stochastic (entries sum to 1).**

Demand-Aware Throughput?

Demand Matrix



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



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 θ^* :
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

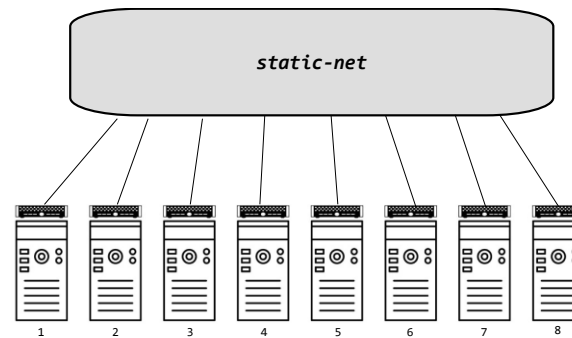
→ E.g., **worst T** for (oblivious) expander?

Demand Matrix

T

	1	2	3	4	5	6	7	8
1		■						
2	■							
3				■				
4			■					
5						■		
6					■			
7								■
8							■	

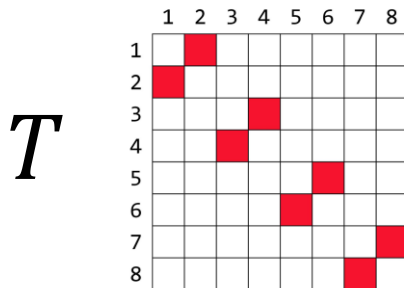
Permutation matrix
is the **worst demand**



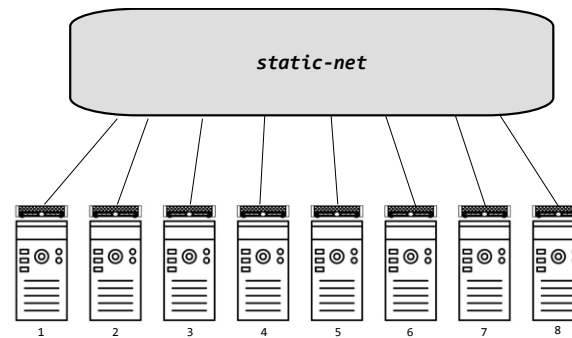
Worst T for Oblivious

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

Demand Matrix



Permutation matrix
is the **worst demand**

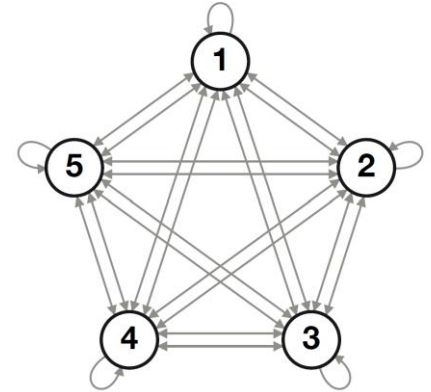


→ Actually the **best T** for demand-aware!

Challenge: Demand-Aware Throughput

Constructive Separation?

- **Known:** worst case throughput θ^* of **demand-oblivious** networks at most $\frac{1}{2}$ (tight)
 - Due to Keslassy, Chang, McKeown, Lee.
 - They consider **$(2n-1)$ -regular** networks
 - **Best oblivious network:**
two links to every other node

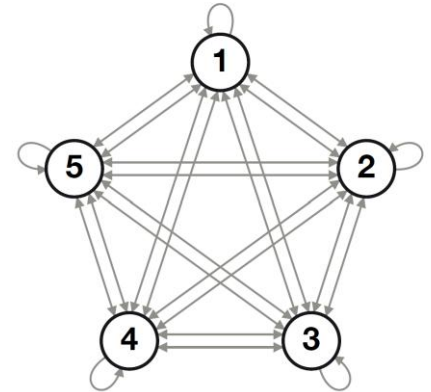


* Same bound also holds for **dynamic** oblivious networks

Challenge: Demand-Aware Throughput

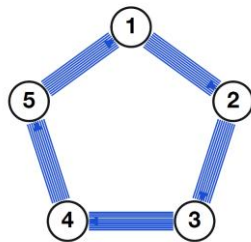
Constructive Separation?

- **Known:** worst case throughput θ^* of **demand-oblivious** networks at most $\frac{1}{2}$ (tight)
 - Due to Keslassy, Chang, McKeown, Lee.
 - They consider **$(2n-1)$ -regular** networks
 - **Best oblivious network:**
two links to every other node



SAME DEMAND

	1	2	3	4	5
1		■			
2			■		
3				■	
4					■
5	■				



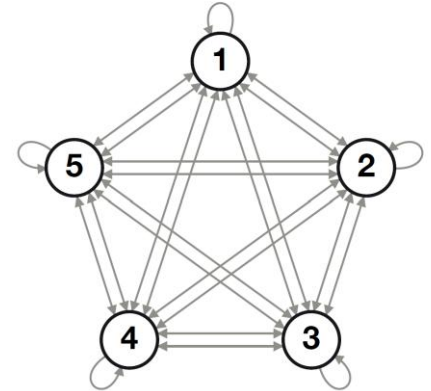
- For this graph, worst T is permutation
- **Best demand-aware network:**
Knowing the demand, each node **pointing all** $(2n-1)$ links at its successor gives perfect **throughput 1**

So demand-aware network **much better** in this **bad case** for oblivious.
Question: But is it better also under **worst-case demand**?

Challenge: Demand-Aware Throughput

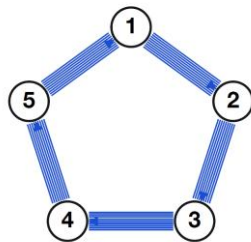
Constructive Separation?

- **Known:** worst case throughput θ^* of **demand-oblivious** networks at most $\frac{1}{2}$ (tight)
 - Due to Keslassy, Chang, McKeown, Lee.
 - They consider **$(2n-1)$ -regular** networks
 - **Best oblivious network:**
two links to every other node



SAME DEMAND

	1	2	3	4	5
1		■			
2			■		
3				■	
4					■
5	■				



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

So demand-aware network **much better** in this **bad case** for oblivious.
Question: But is it better also under **worst-case demand**? **Subtle!**

Challenge: Demand-Aware Throughput

Constructive Separation?

- Unknown: **Demand-aware** throughput can never be worse than demand-oblivious, but is it always strictly better in the worst case?

Input: A doubly stochastic demand matrix M and a number $0 \leq \theta \leq 1$.

Question: Is there a graph G such that the throughput of G with respect to M is at least θ ?

DEMAND-OBVIOUS

$$\theta_{\text{OB}}^* = \max_G \min_M \theta(G, M)$$

Fix G first — then face the worst demand.

DEMAND-AWARE

$$\theta_{\text{DA}}^* = \min_M \max_G \theta(G, M)$$

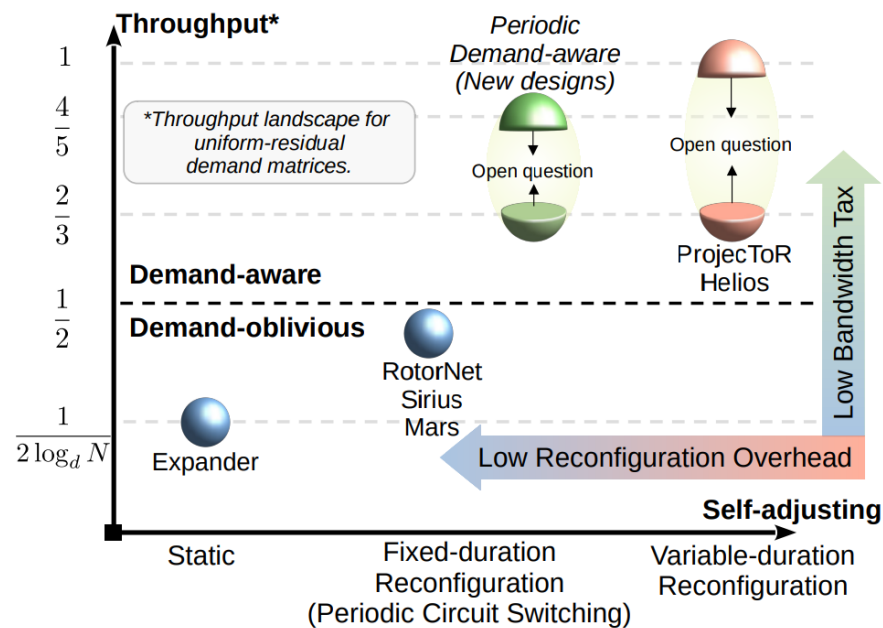
See M first — then choose the best G .

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

Many More Open Questions, f. ex.

Hybrid Throughput?

→ More open questions, for dynamic hybrid datacenter networks



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

Addanki et al., Vermillion:
USENIX NSDI 2026

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

Matthias Bentert¹, Chen Avin², and Stefan Schmid¹

¹Technische Universität Berlin, Germany

²Ben-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.

Challenge: Impact of Reconfigurations on 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

Challenge: Impact of Reconfigurations on Buffer Management?

ABM: Active Buffer Management in Datacenters

Vamsi Addanki*
TU Berlin

Stefan Schmid
TU Berlin

Maria Apostolaki
Princeton University

Laurent Vanbever
ETH Zurich

Manya Ghobadi
MIT

ABSTRACT

ABSTRACT

Today's network devices share buffer across queues to avoid drops during transient congestion and buffer bloats. The need for optimal buffer-handling action in datacenter decreases, the need for an optimal buffer utilization becomes more urgent. Typical devices use a hierarchical packet admission control scheme. First, a Buffer Management (BM) scheme decides the maximum length per queue at the local BM scheme and then an Active Queue Management (AQM) scheme decides which packets will be admitted at the queue level. Unfortunately, lack of coordination between the two control schemes results in: (i) harmful interference across queues, due to the obliviousness; (ii) increased congesting delay; and (iii) thus aggressive buffer loss tolerance. We propose an Active Queue Management (AQM) scheme that is aware of the per-queue buffer drain time and achieves predictable buffer loss tolerance. To overcome these limitations, we propose ABM improves the buffer drain time and achieves predictable buffer loss tolerance. To overcome these limitations, we propose ABM improves the buffer drain time and achieves predictable buffer loss tolerance. To overcome these limitations, we propose ABM improves the buffer drain time and achieves predictable buffer loss tolerance.

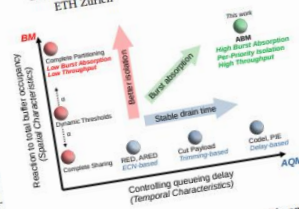


Figure 1: BM and AQM are orthogonal in their goals, and the hierarchical scheme fundamentally limits the burst absorption capabilities of the buffer.

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

Vamsi Addanki
TU Berlin

Giannis Patronas
NVIDIA

Paraskevas Bakopoulos
NVIDIA

Chen Avin
Ben-Gurion University of the Negev

Dimitris Syrivelis
NVIDIA

Ilias Marinos
NVIDIA

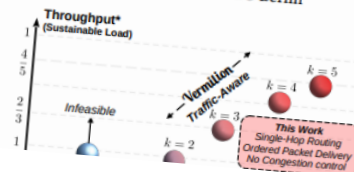
Goran Dario Knabe
TU Berlin

Nikos Terzenidis
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



Challenge: Impact of Reconfigurations on 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 μ s, 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

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

Oliver Michel
University of Vienna
Princeton University

Stefan Schmid
University of Vienna
TU Berlin

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

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

☺ Can achieve near-zero queue equilibrium

☹ Slow reaction

→ 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

☺ Fast reaction

☹ No equilibrium

Case Study:

PowerTCP

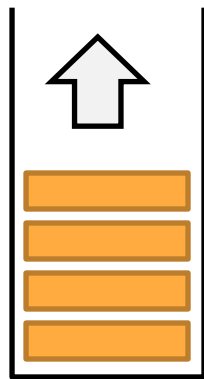
Existing congestion control algorithms based on either

- 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

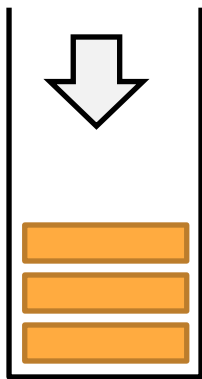
Limitation: using only one of the two
may miss useful information for fine-grained adaptations!

Limitation of SOTA

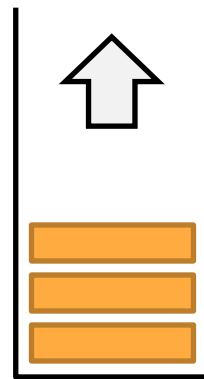
→ Consider a queue which may be in three different states:



1
growing



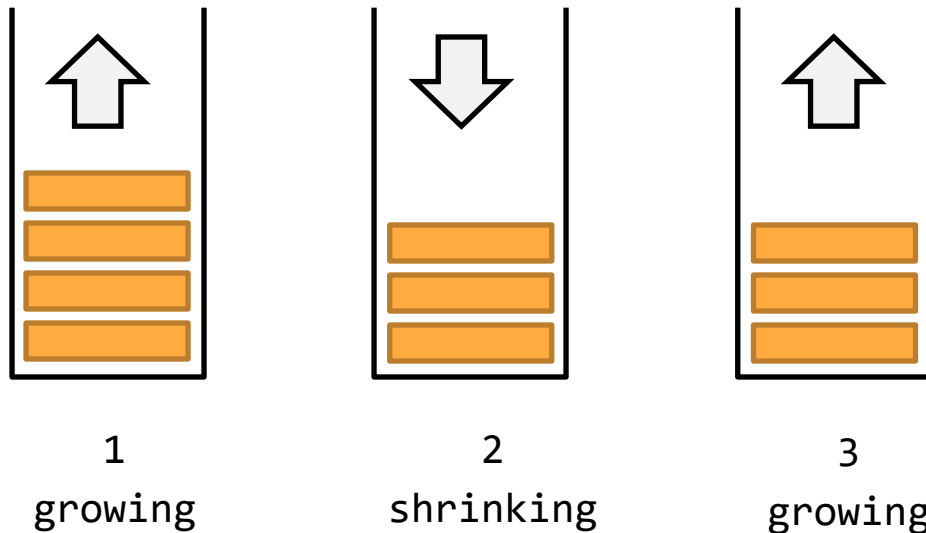
2
shrinking



3
growing

Limitation of SOTA

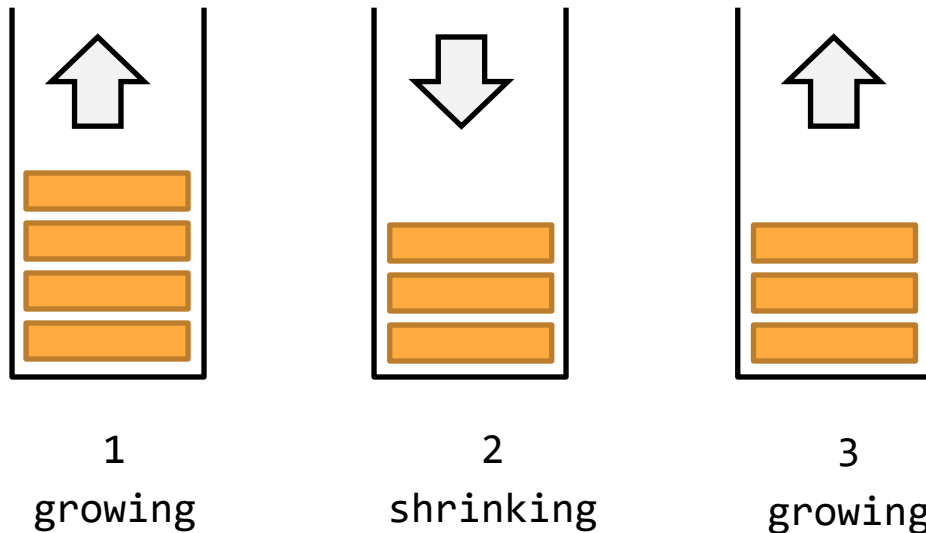
→ Consider a queue which may be in three different states:



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

Limitation of SOTA

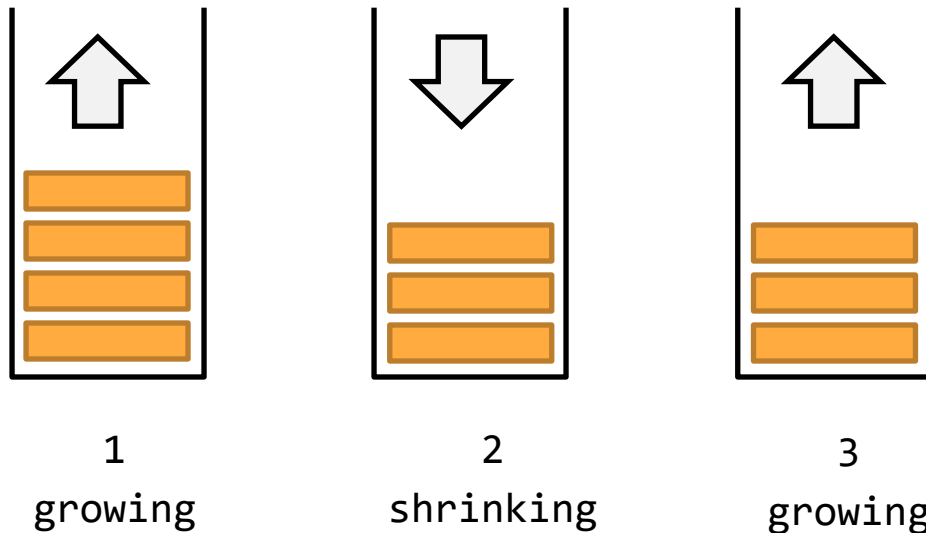
→ Consider a queue which may be in three different states:



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

Limitation of SOTA

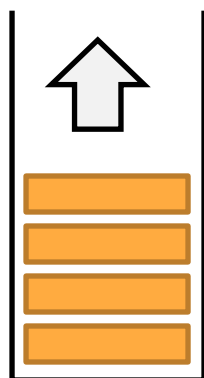
→ Consider a queue which may be in three different states:



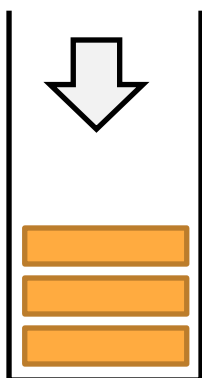
We need both: **Power** (Voltage x Current)

Limitation of SOTA

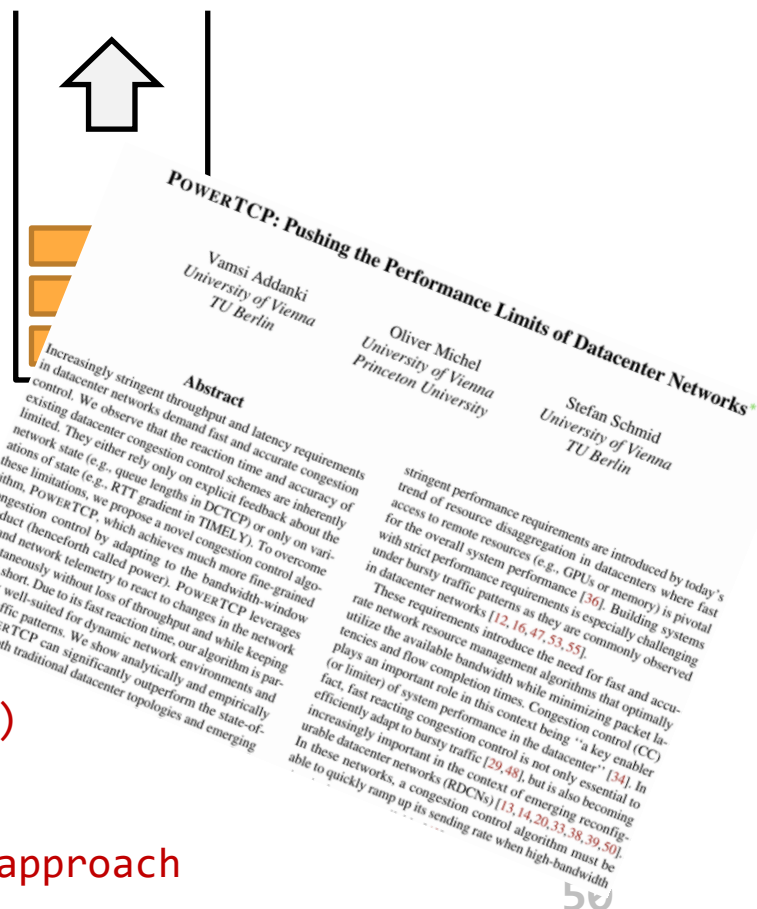
→ Consider a queue which may be in three different states:



1
growing



2
shrinking



We need both: **Power** (Voltage x Current)

Inspired:



A control-theoretic approach

More benefits of optical & reconfigurable switching

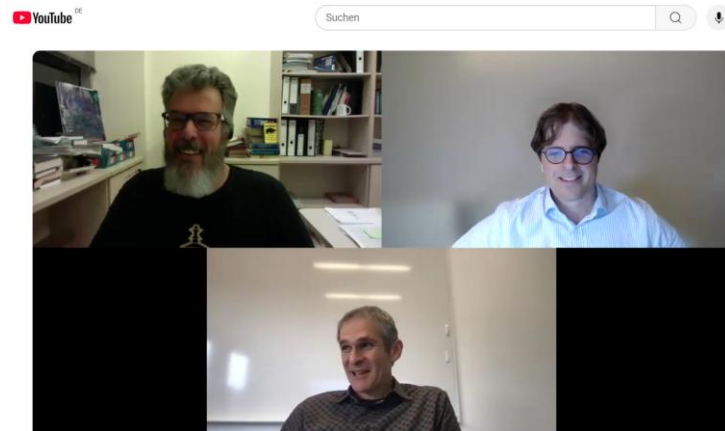
So far: focus on throughput performance.

Benefit 1:

Evolving Datacenters

- Reconfigurable datacenter networks naturally support *heterogeneous* network elements
- And therefore also *incremental* hardware upgrades

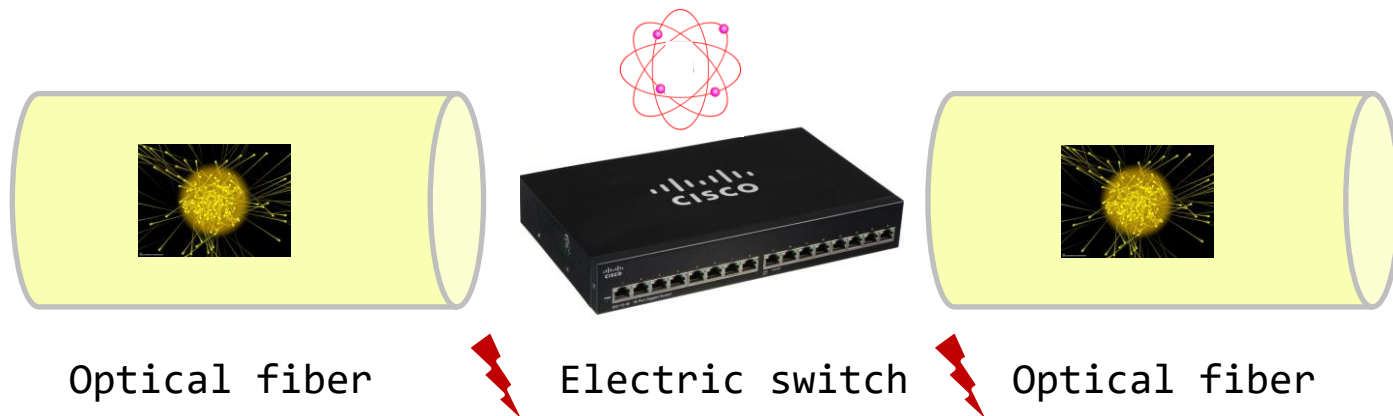
See interview with Amin
Vahdat, Google in CACM:
[https://www.youtube.com/
watch?v=IxcV1gu8ETA](https://www.youtube.com/watch?v=IxcV1gu8ETA)



Benefit 2:

Energy and Latency

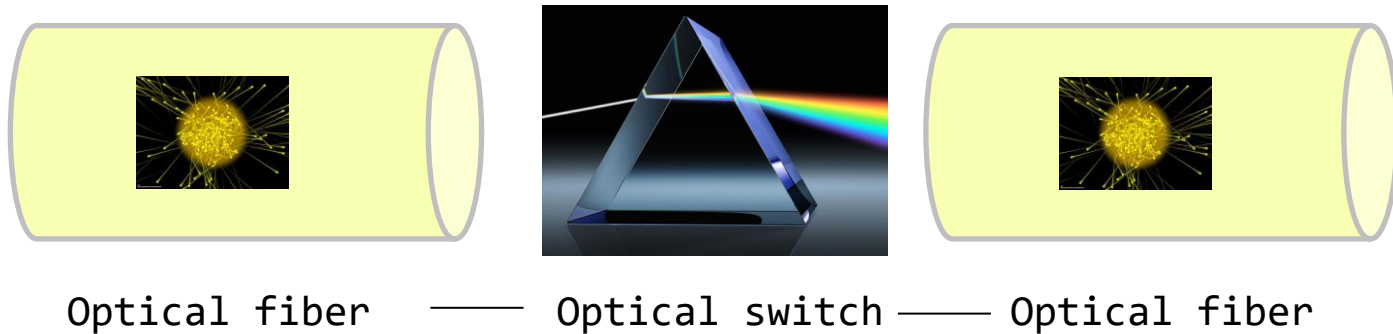
- No need to *convert* photons in fiber to electrons in switch (and back)
- Can save *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)



Benefit 2:

Energy and Latency

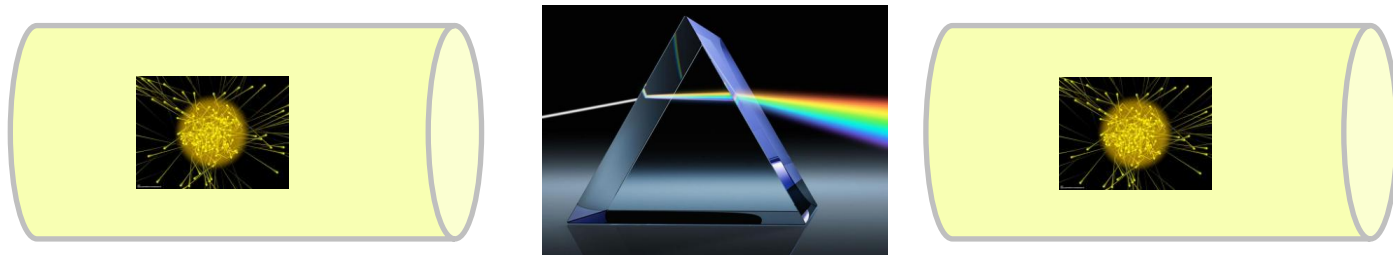
- No need to *convert* photons in fiber to electrons in switch (and back)
- Can save *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)



Benefit 2:

Energy and Latency

- No need to *convert* photons in fiber to electrons in switch (and back)
- Can save *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)



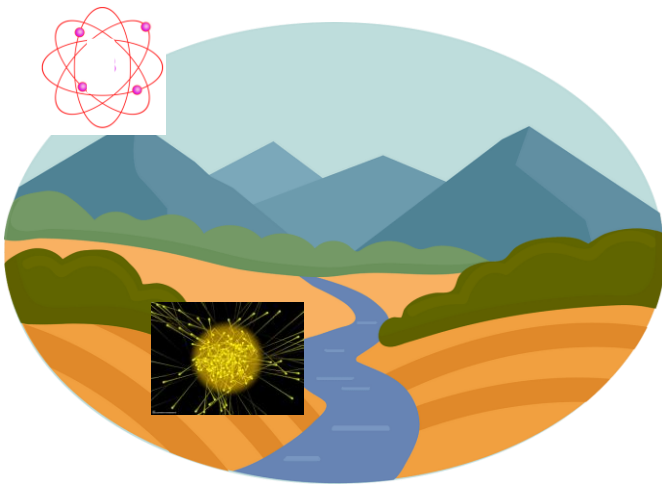
Optical fiber — Optical switch — Optical fiber

- Interesting for emerging *distributed datacenters*!

Benefit 3:

Resilience

Floodings in South Germany destroyed much electrical network infrastructure



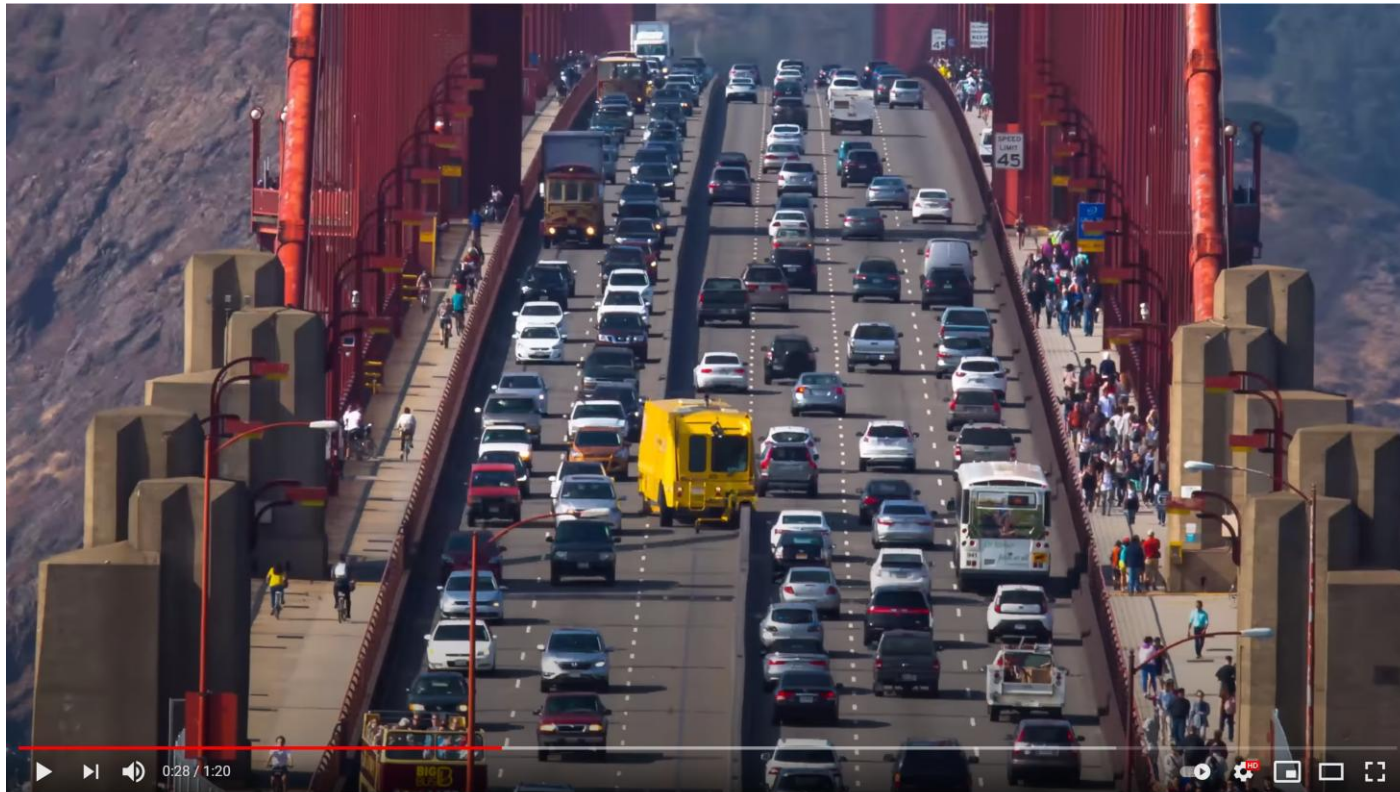
Solution: deploy optical infrastructure (in valleys) and electrical *on hills* where safe?

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?

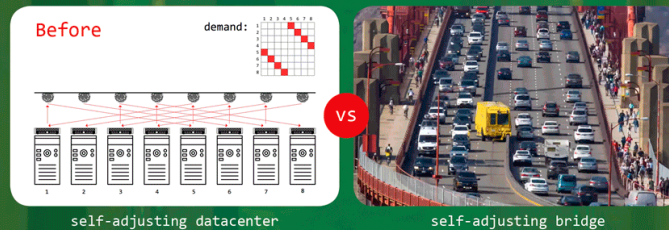


Slides
available
here:



Online Video Course

Invitation to Self-Adjusting Networks A short video course



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



Prof. Chen Avin
(BGU, Israel)



Prof. Stefan Schmid
(TU Berlin, Germany)



<https://self-adjusting.net/course>



YouTube Interview & CACM

Check out our **YouTube interviews**
on Reconfigurable Datacenter Networks:



[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

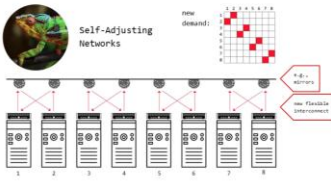
SELF-ADJUSTING NETWORKS
RESEARCH ON SELF-ADJUSTING DEMAND-AWARE NETWORKS

Project Overview Team Publications Contact Us

AdjustNet

Breaking new ground with demand-aware self-adjusting networks

Our Vision:
Flexible and Demand-Aware Topologies



Self-adjusting Networks

new demand

new flexible interconnect

new flexible interconnect

Download Slides

WEBSITE LAUNCHED!
MARCH 12, 2020

This site provides an overview of our ongoing research on the foundations of self-adjusting networks.

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



TRACE COLLECTION
WAN AND DC NETWORK TRACES

Publication Team Download Traces Contact Us



The following table lists the traces used in the publication: **On the Complexity of Traffic Traces and Implications**
To reference this website, please use: bibtex

File Name	Source Information	Type	Lines	Size	Download
exact_BusLib_MultiGnd_C_Large_1024.csv	High Performance Computing Traces	Traces	17,947,800	151.3 MB	Download
exact_BusLib_CNS_NoSpec_Large_1024.csv	High Performance Computing Traces	Traces	1,108,068	9.3 MB	Download
cesar_Nekbone_1024.csv	High Performance Computing Traces	Traces	21,745,229	184.0 MB	Download

<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.

[Vermilion: A Traffic-Aware Reconfigurable Optical Interconnect with Formal Throughput Guarantees](#)

Vamsi Addanki, Deeksha Panduranga Rao, Goran Dario Knabe, Chen Avin, and Stefan Schmid.
24th USENIX Symposium on Networked Systems Design and Implementation (**NSDI**), Providence, Rhode Island, USA, May 2027.

[Bridge: Optimizing Collective Communication Schedules in Reconfigurable Networks with Reusable Subrings](#)

Anton Jürß and Stefan Schmid.
ACM International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (**MobiHoc**), Tokyo, Japan, November 2026.

[Revisiting Bruck: Phase-Efficient All-to-All Collective Communication in Reconfigurable Networks](#)

Anton Jürß and Stefan Schmid.
ACM SIGCOMM Workshop on Hot Topics in Optical Technologies and Applications in Networking (**HotOptics**), Denver, Colorado, USA, August 2026.

[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.

[Trivance: Latency-Optimal AllReduce by Shortcutting Multiport Networks](#)

Anton Jürß, Vamsi Addanki, and Stefan Schmid.
ACM **SIGCOMM**, Denver, Colorado, USA, August 2026.

References (2)

[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.

[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.

[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.

[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.

References (3)

[DistroMatch: Distributed Disjoint Weighted Matchings in Demand-Aware Reconfigurable Optical Datacenters](#)

Sophia Heck, Kathrin Hanauer, and Stefan Schmid.

ACM International Conference on Supercomputing (ICS), Belfast, Northern Ireland, United Kingdom, July 2026.

[Dynamic Demand-Aware Link Scheduling for Reconfigurable Datacenters](#)

Kathrin Hanauer, Monika Henzinger, Lara Ost, and Stefan Schmid.

IEEE Conference on Computer Communications (INFOCOM), New York, USA, May 2023.

[Fast and Heavy Disjoint Weighted Matchings for Demand-Aware Datacenter Topologies](#)

Kathrin Hanauer, Monika Henzinger, Stefan Schmid, and Jonathan Trummer.

IEEE Conference on Computer Communications (INFOCOM), Virtual Conference, May 2022.

[Toward Demand-Aware Networking: A Theory for Self-Adjusting Networks](#) (Editorial)

Chen Avin and Stefan Schmid.

ACM SIGCOMM Computer Communication Review (CCR), October 2018.

Bonus Material



Hogwarts Stair

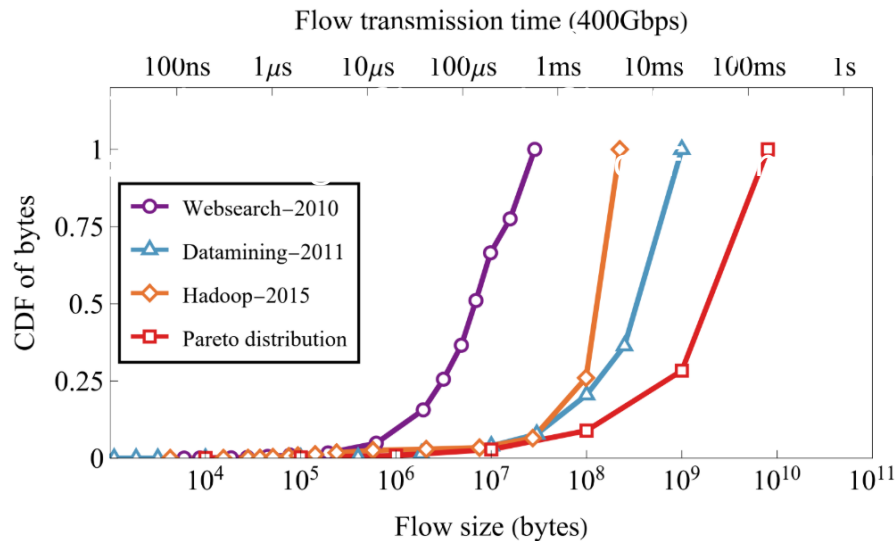
Optimal Hybrid RDCNs

Flow Size Matters

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

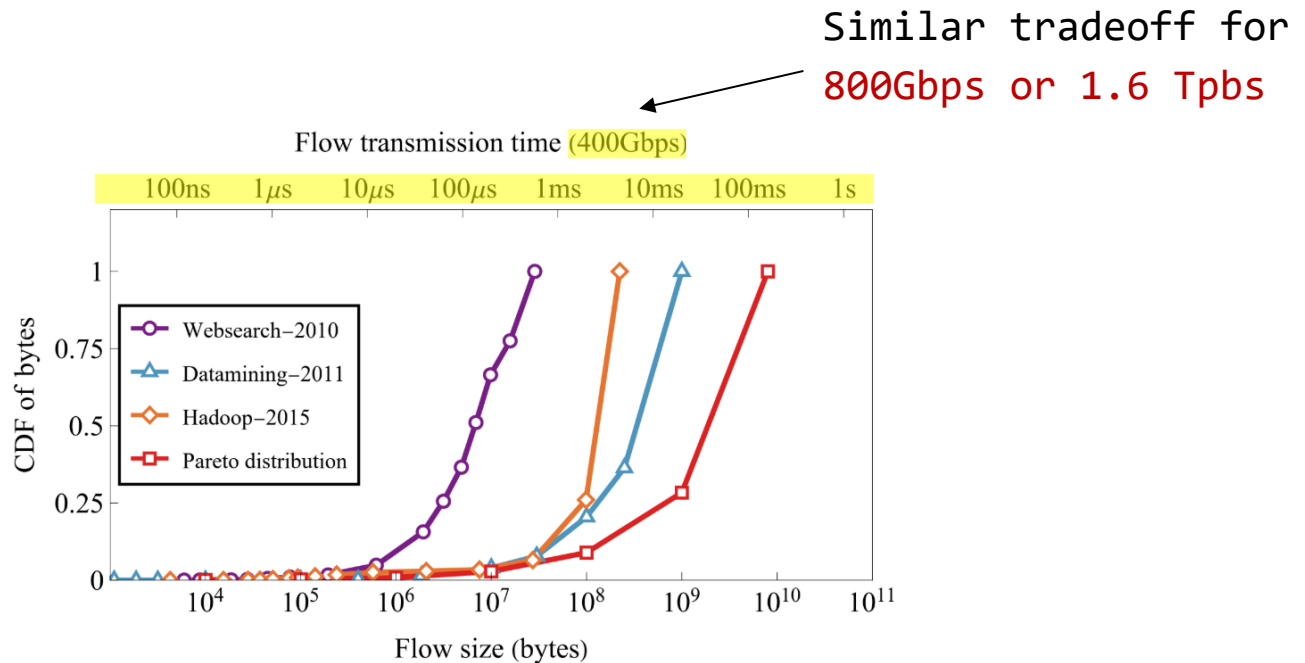
Flow Size Matters

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



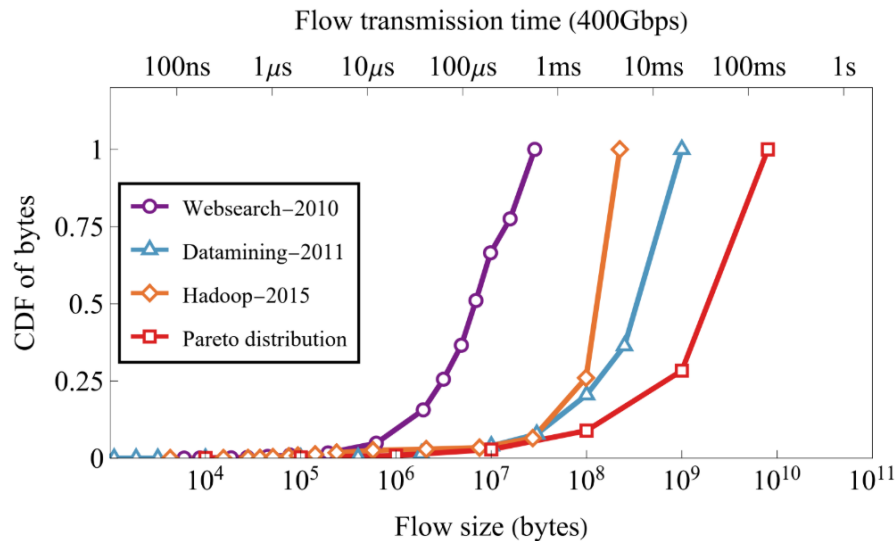
→ **Observation 1:** Different apps have different flow size distributions.

Flow Size Matters



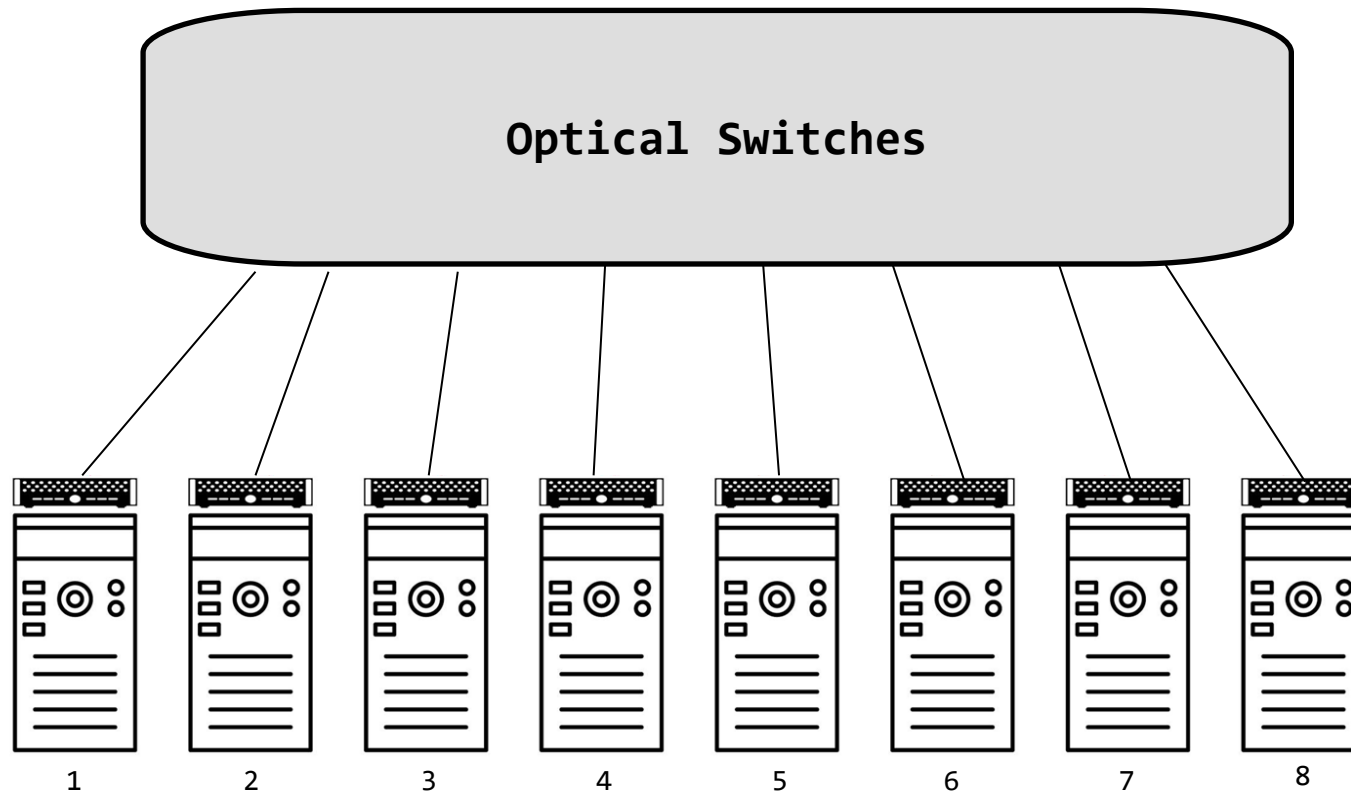
- **Observation 1:** Different apps have different flow size distributions.
- **Observation 2:** The transmission time of a flow depends on its size.

Flow Size Matters

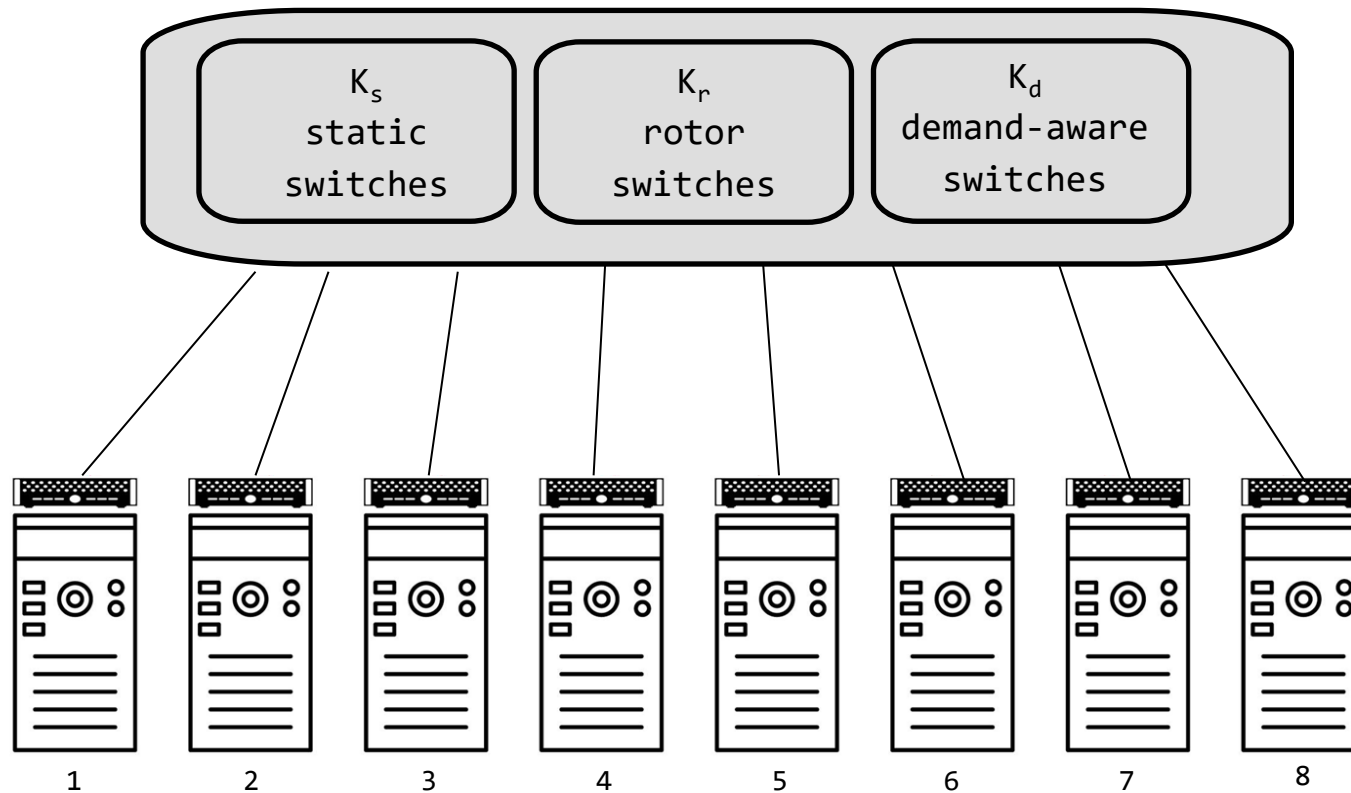


- **Observation 1:** Different apps have different flow size distributions.
- **Observation 2:** The transmission time of a flow depends on its size.
- **Observation 3:** For small flows, flow completion time suffers if network needs to be reconfigured first.
- **Observation 4:** For large flows, reconfiguration time may amortize.

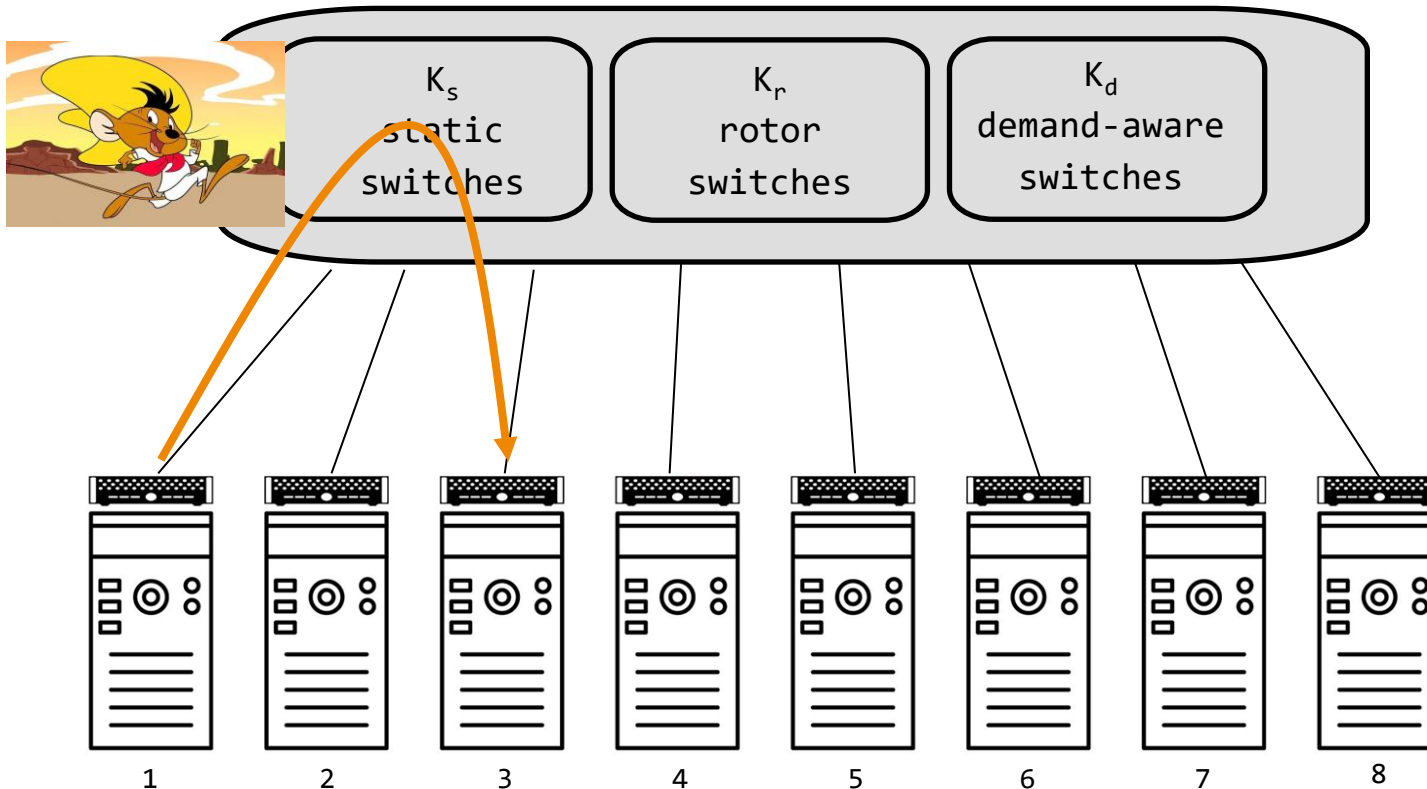
Optimal: Cerberus



Optimal: Cerberus

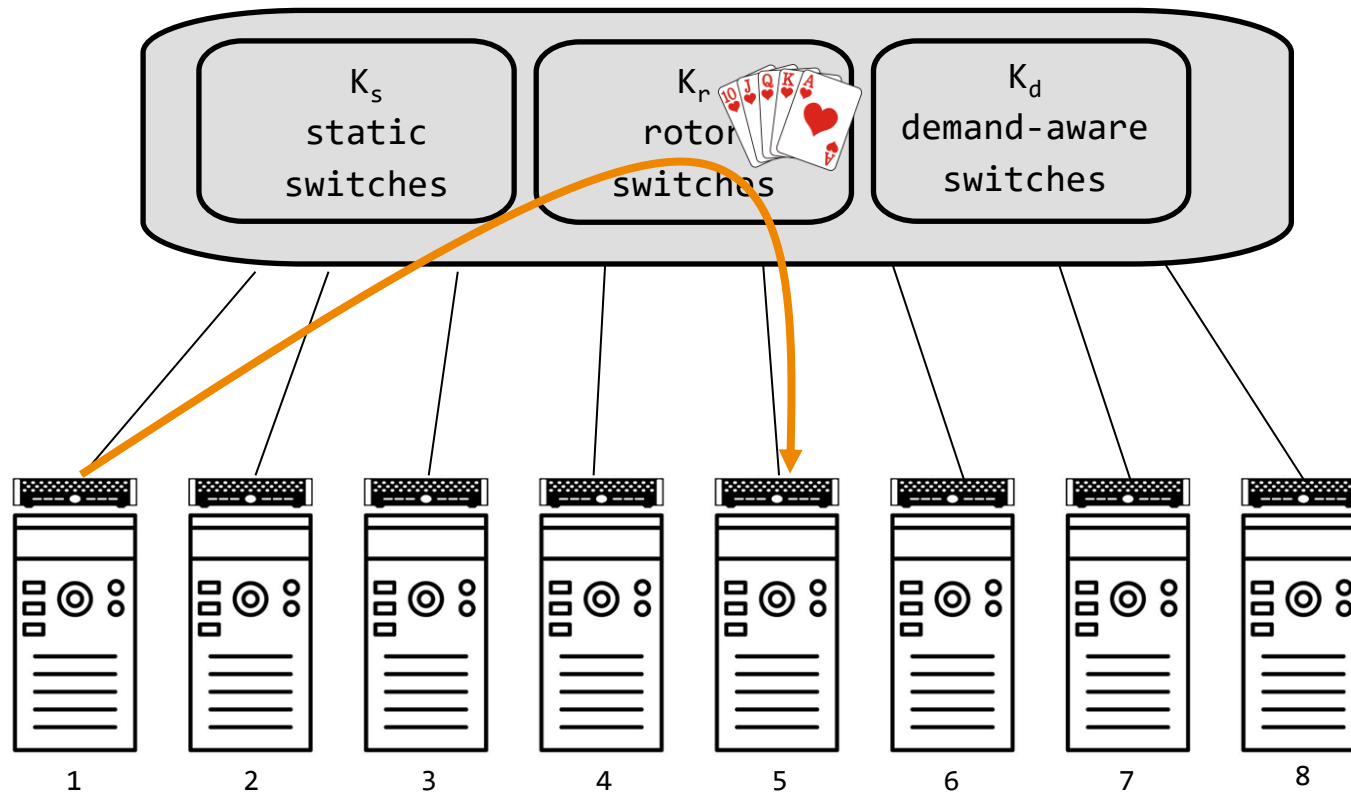


Optimal: Cerberus



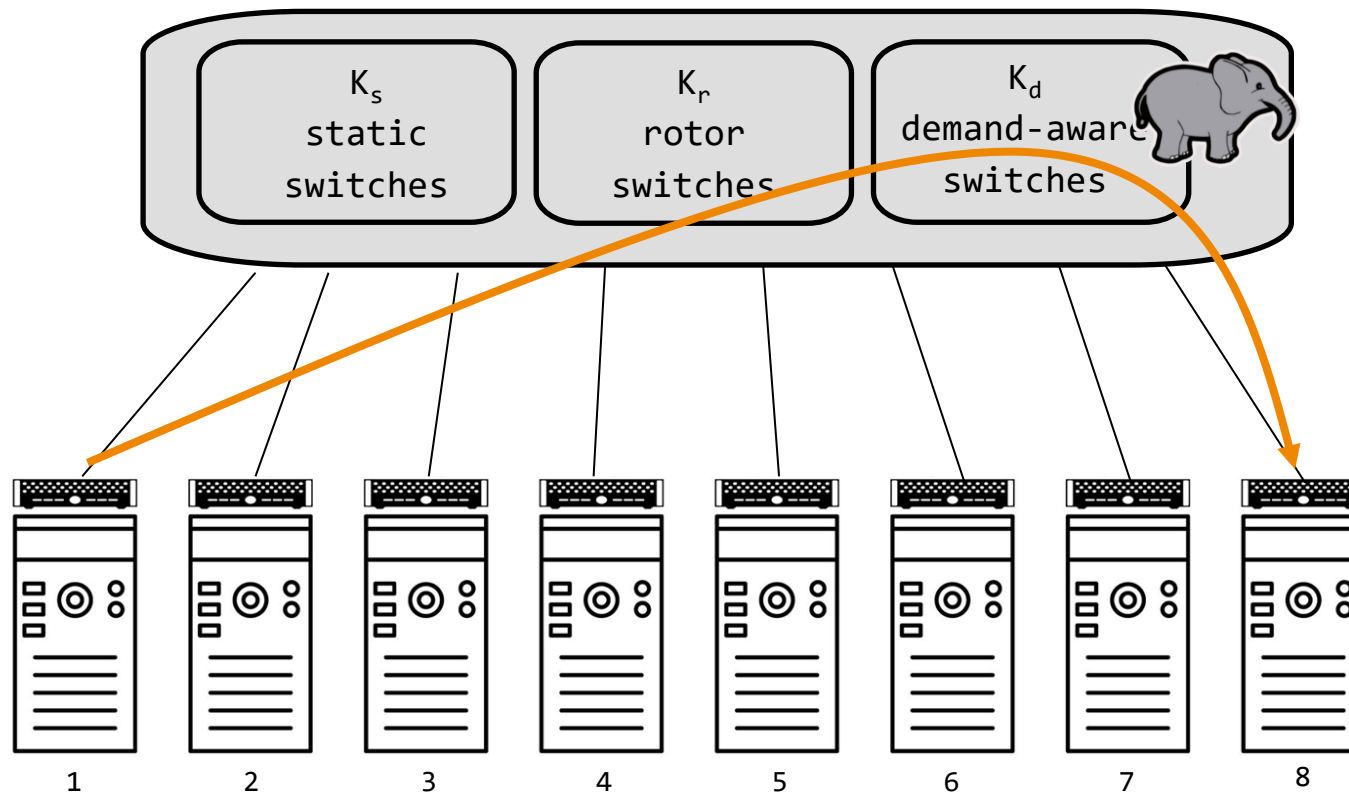
Scheduling: **Small flows** go via static switches...

Optimal: Cerberus



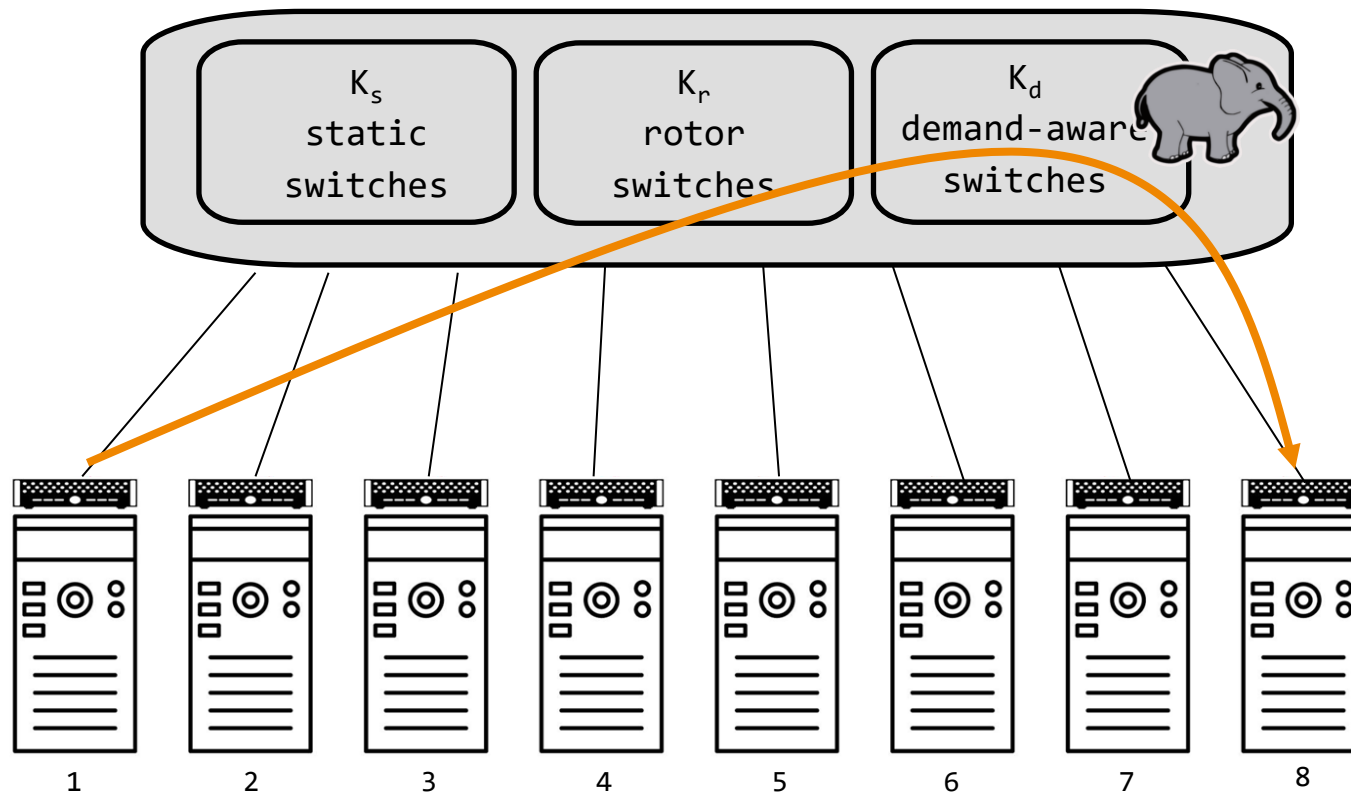
Scheduling: ... medium flows via rotor switches...

Optimal: Cerberus



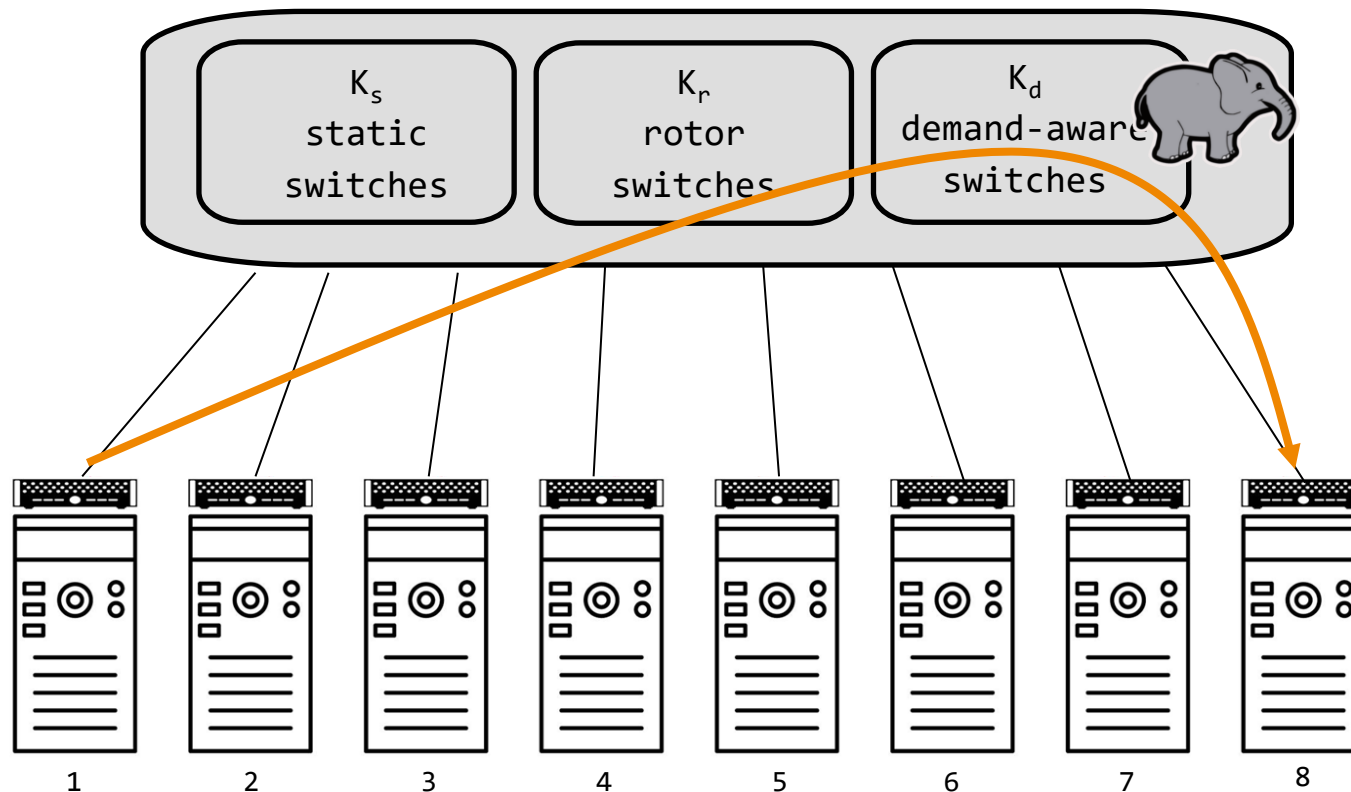
Scheduling: ... and **large flows** via demand-aware switches
(if one available, otherwise via rotor).

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?

Question:

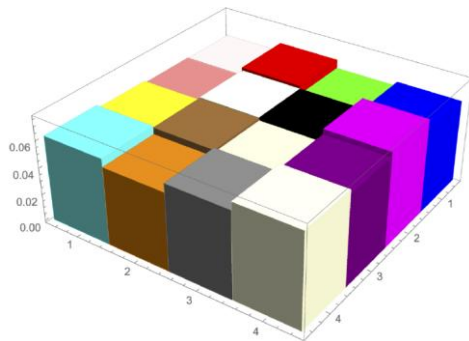
How to Quantify
such “Structure”
in the Demand?

Intuition

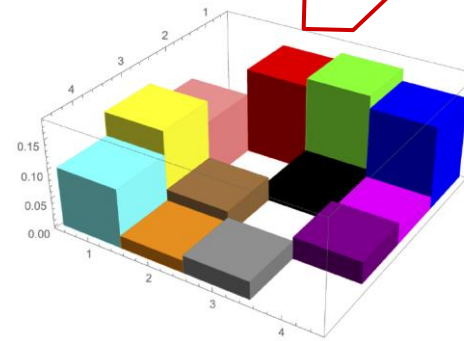
Which demand has more structure?

→ Traffic matrices of two different distributed ML applications

→ GPU-to-GPU



VS



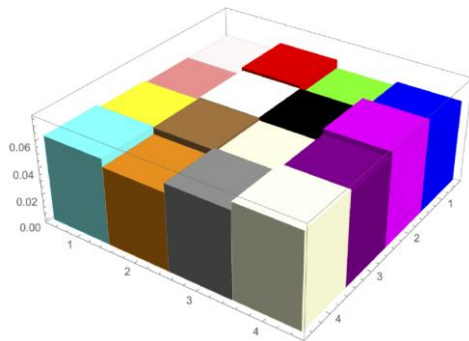
Color = communication pair

Intuition

Which demand has more structure?

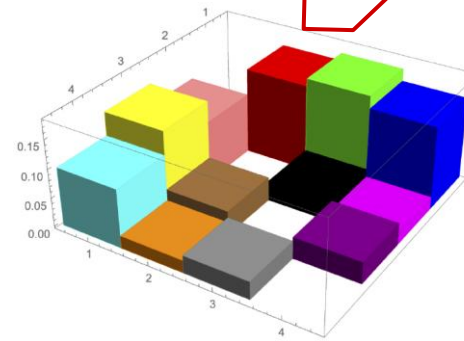
→ Traffic matrices of two different distributed ML applications

→ GPU-to-GPU



More uniform

VS



More structure

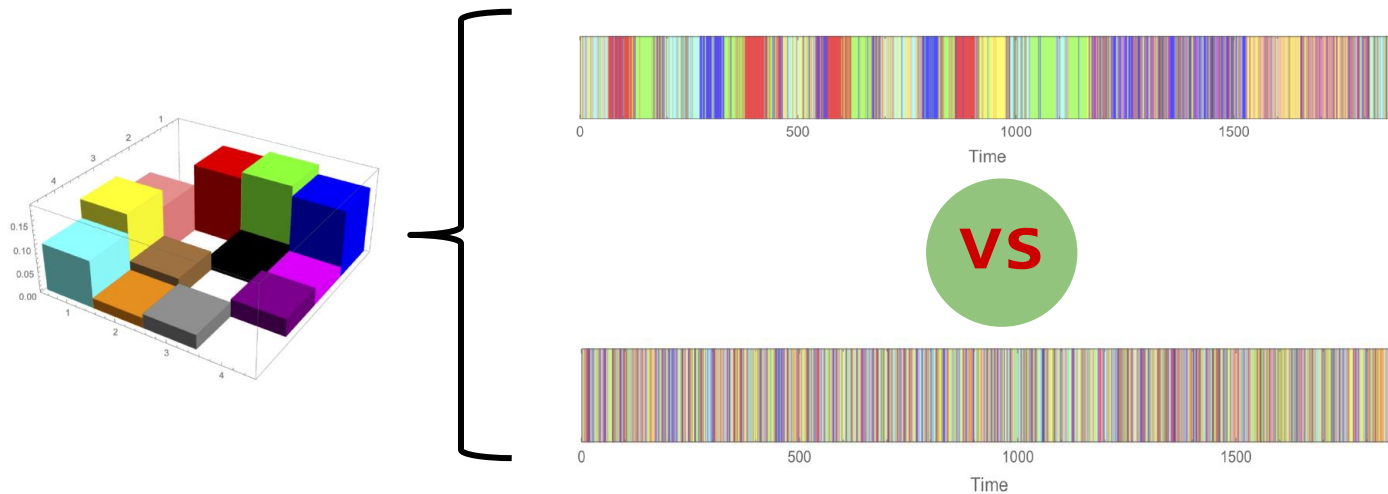
Intuition

Spatial vs temporal structure

→ Two different ways to generate same traffic matrix:

→ Same non-temporal structure

→ Which one has more structure?

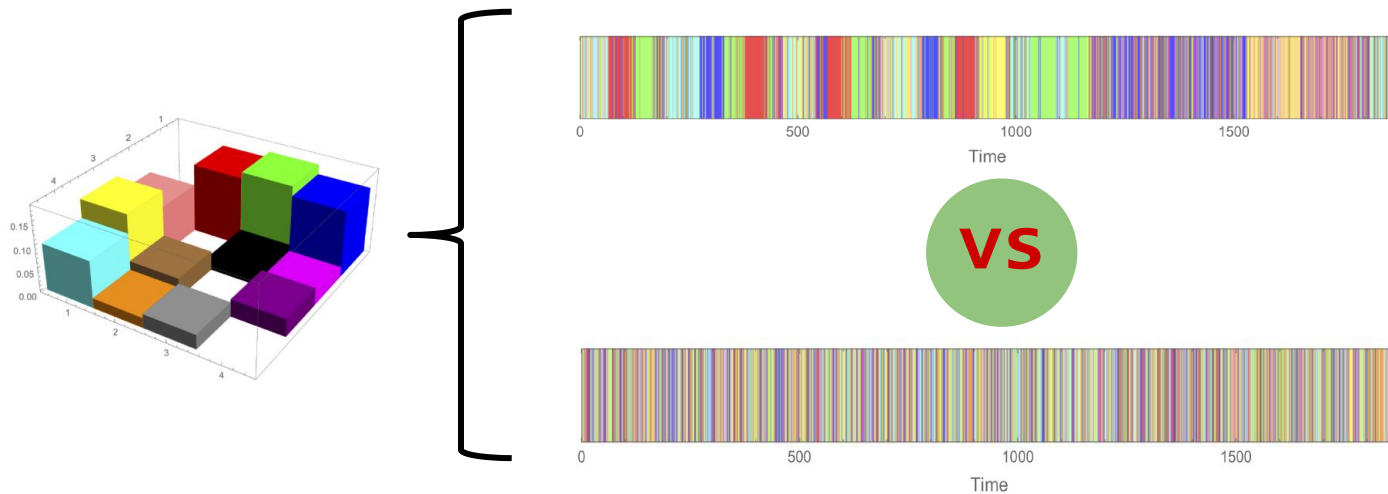


Intuition

Spatial vs temporal structure

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

→ Which one has more structure?

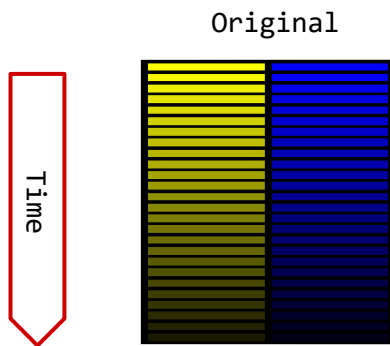


Systematically?

Trace Complexity

Information-Theoretic Approach

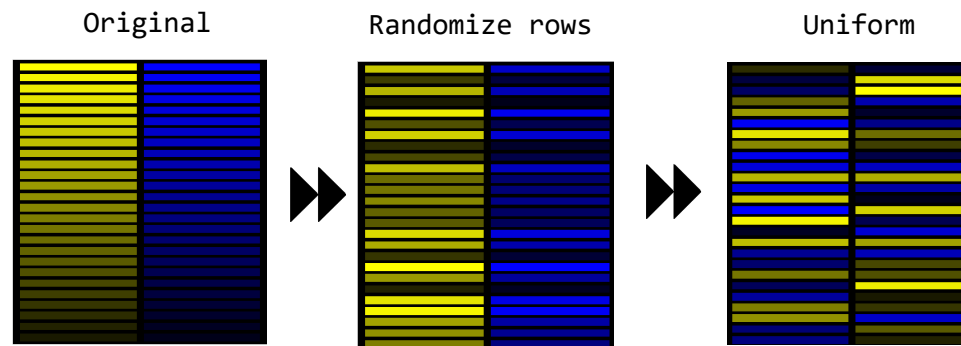
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

“Shuffle&Compress”



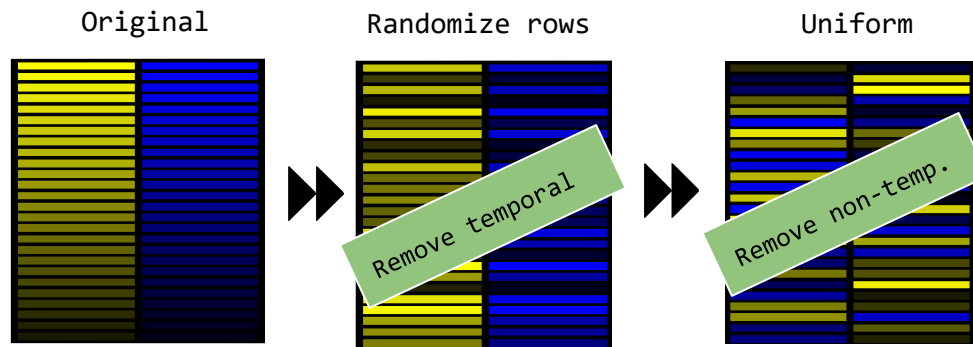
Increasing complexity (systematically randomized)

More structure (compresses better)

Trace Complexity

Information-Theoretic Approach

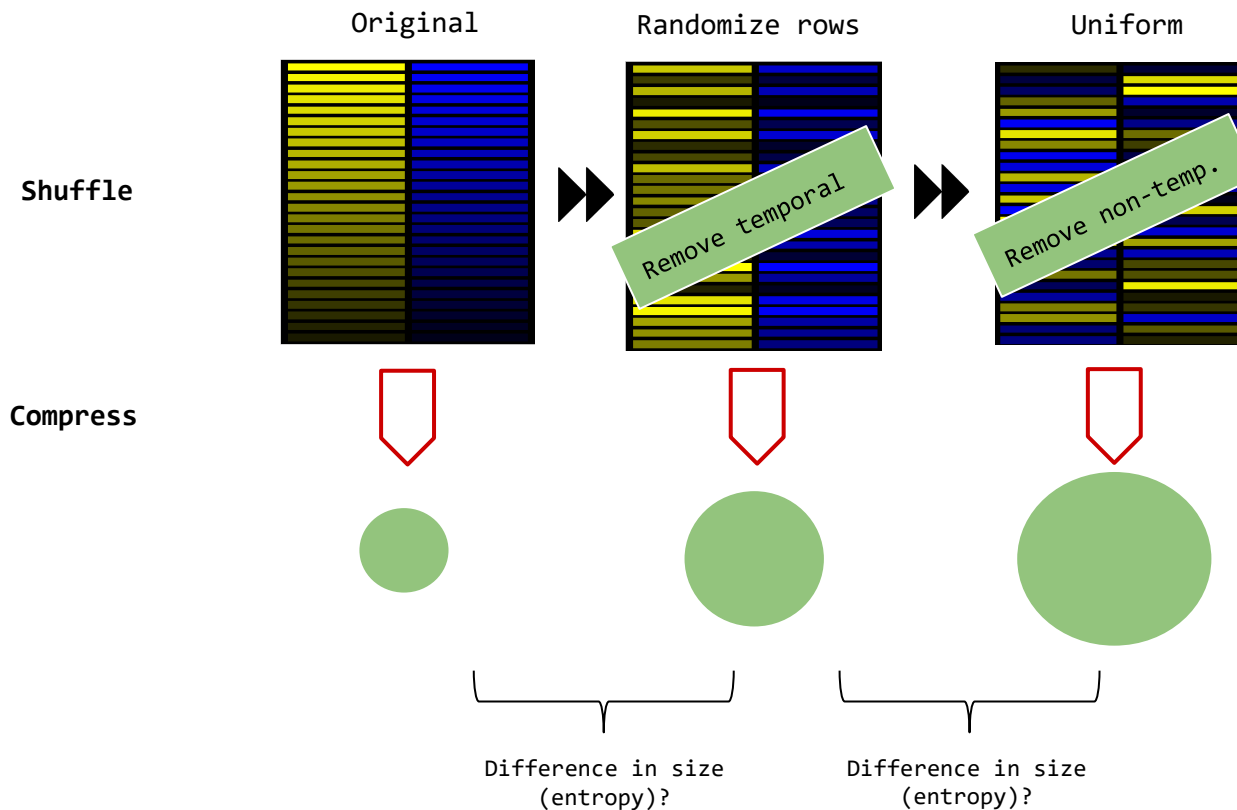
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

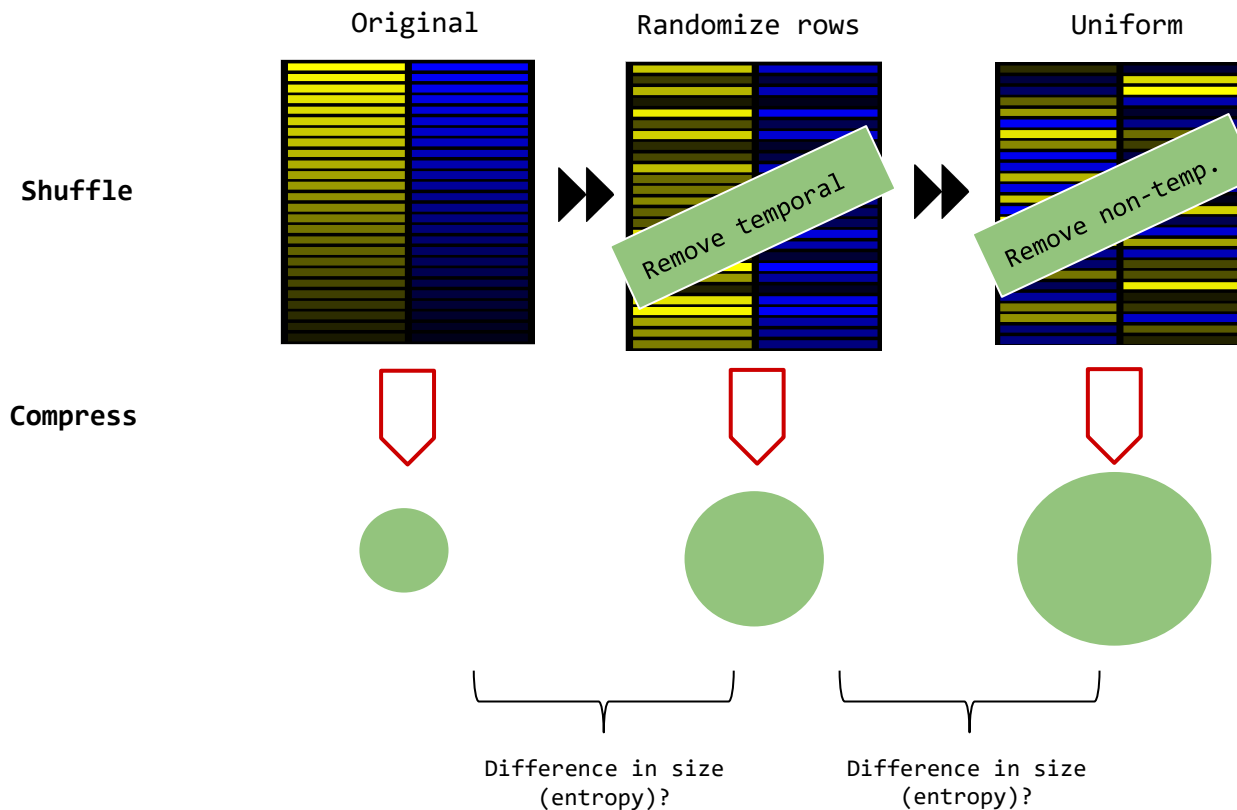
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

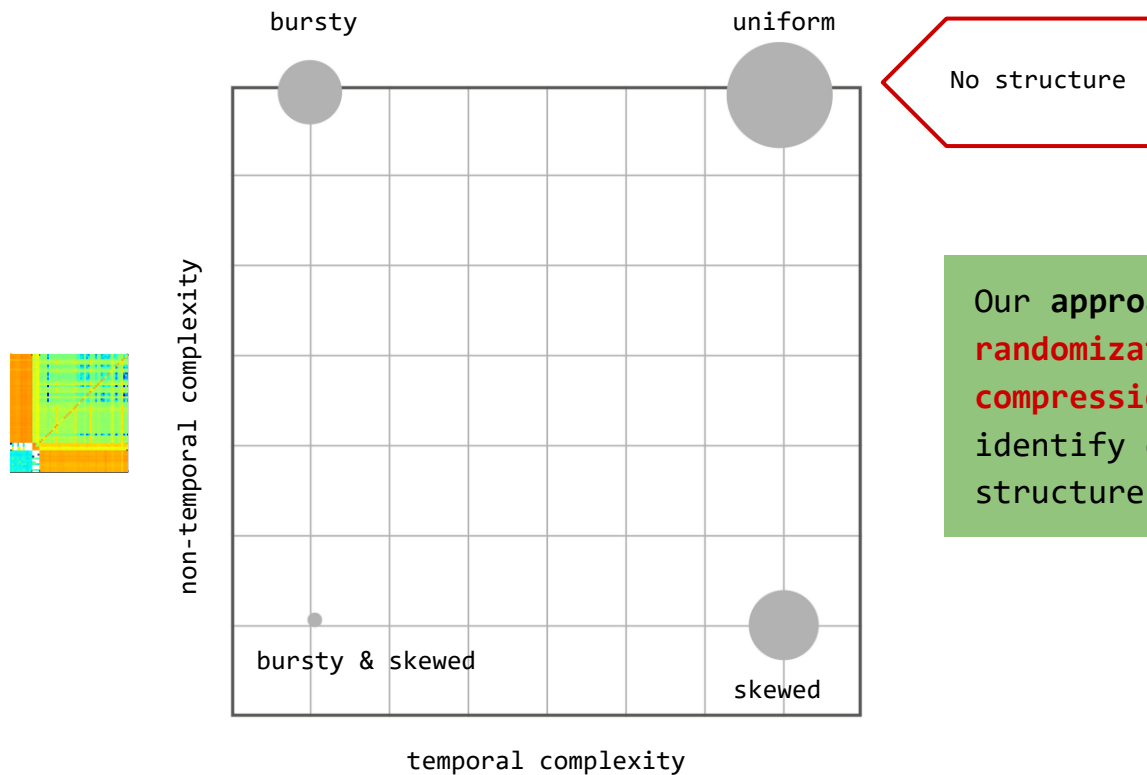
“Shuffle&Compress”



Can be used to define
2-dimensional
complexity map!

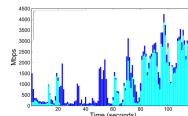
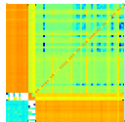
Our Methodology

Complexity Map



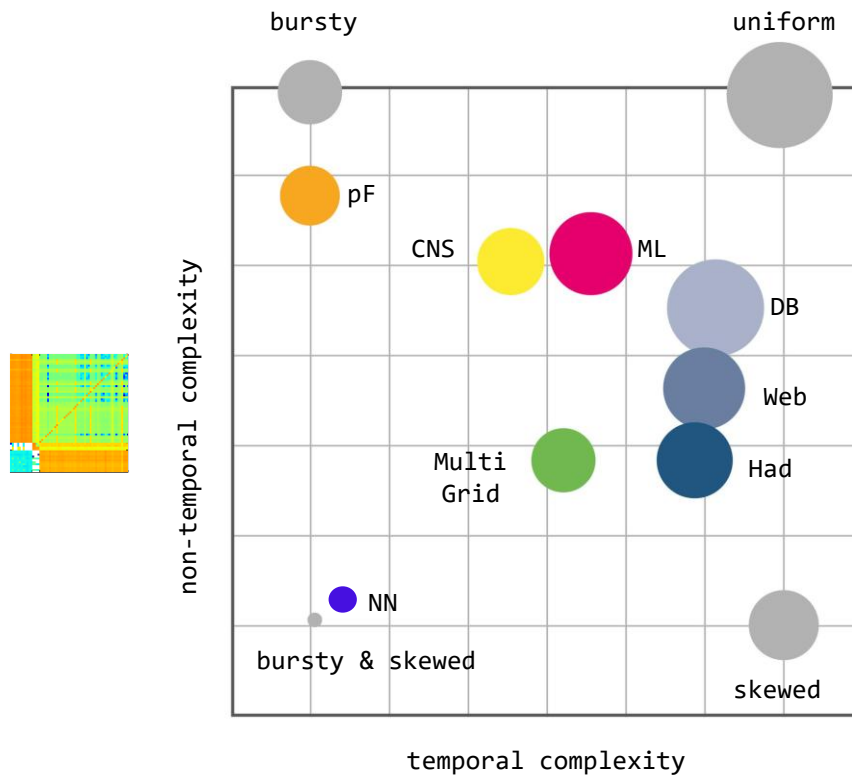
No structure

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



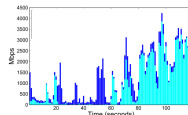
Our Methodology

Complexity Map



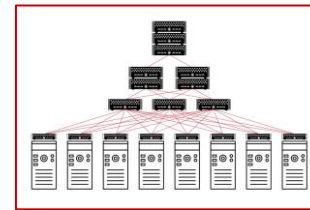
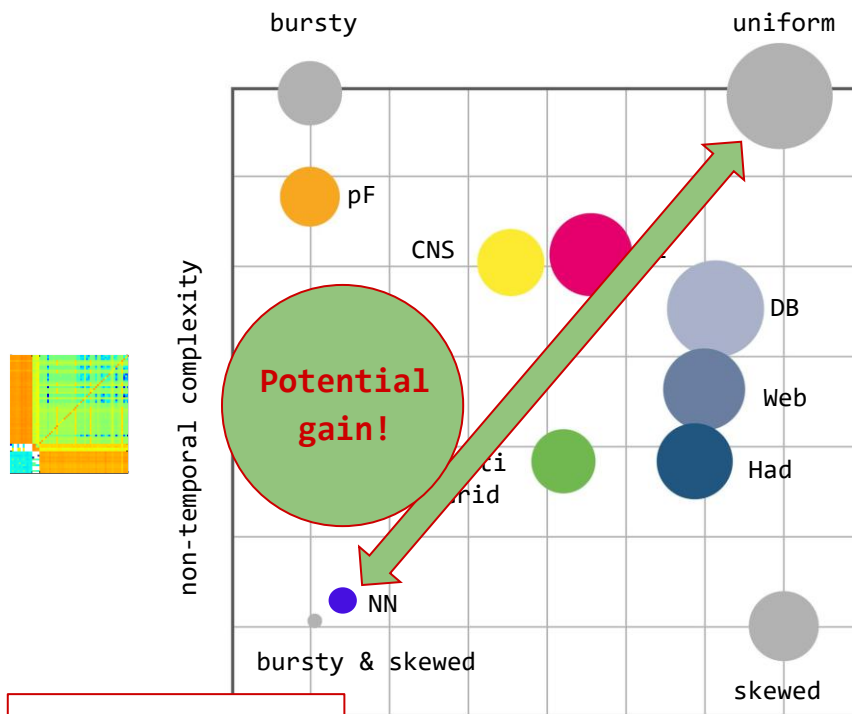
Our approach: iterative **randomization and compression** of trace to identify dimensions of structure.

Different structures!



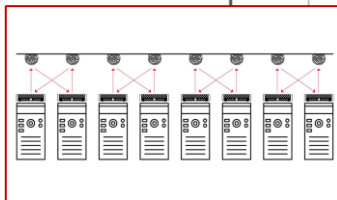
Our Methodology

Complexity Map

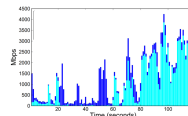


Our approach: iterative **randomization and compression** of trace to identify dimensions of structure.

Different structures!



temporal complexity



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>

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:

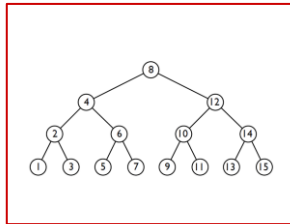
Given This Structure,
What Can Be Achieved?
Metrics and Algorithms?

A first insight: entropy of the demand.

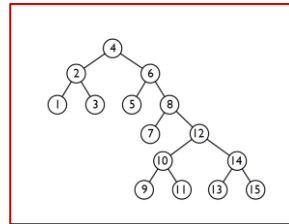
Insight:

Connection to Datastructures

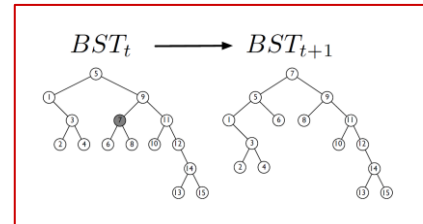
Traditional BST



Demand-aware BST



Self-adjusting BST

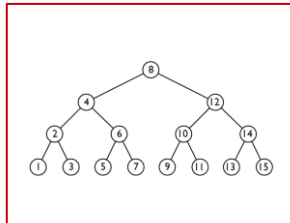


More structure: improved **access cost**

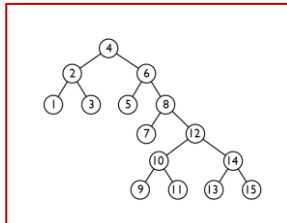
Insight:

Connection to Datastructures & Coding

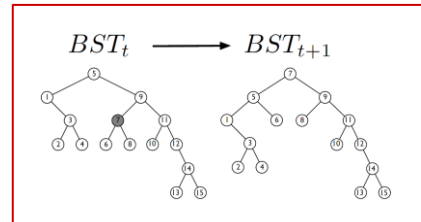
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)



Self-adjusting BST
(Dynamic Huffman coding)

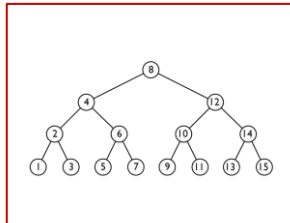


More structure: improved **access cost** / shorter **codes**

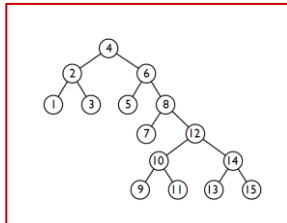
Insight:

Connection to Datastructures & Coding

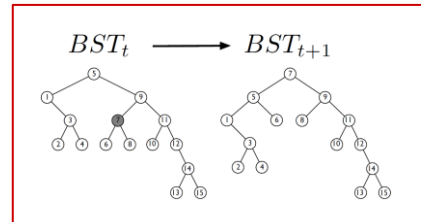
Traditional BST
(Worst-case coding)



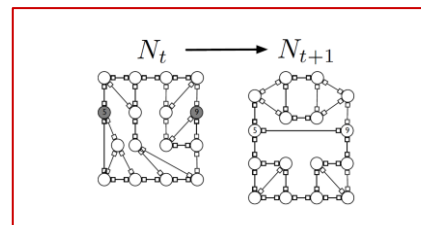
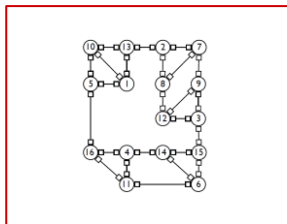
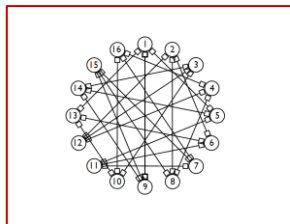
Demand-aware BST
(Huffman coding)



Self-adjusting BST
(Dynamic Huffman coding)



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

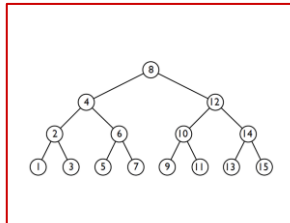


Similar **benefits**?

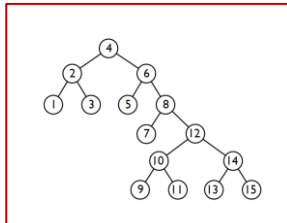
Insight:

Connection to Datastructures & Coding

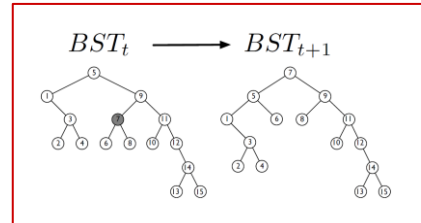
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)

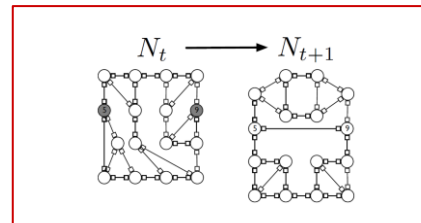
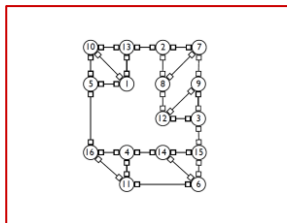
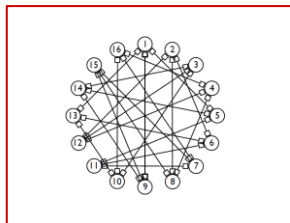


Self-adjusting BST
(Dynamic Huffman coding)



More than
an analogy!

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

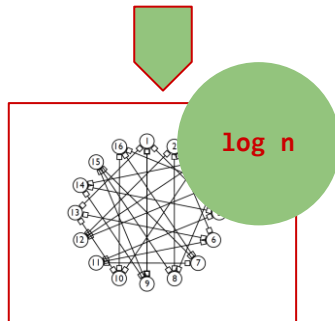
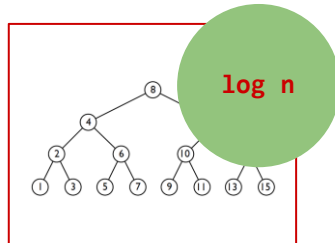


Similar **benefits**?

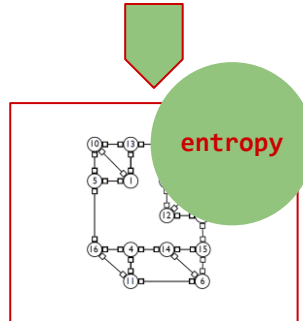
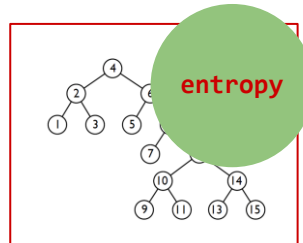
Insight:

Connection to Datastructures & Coding

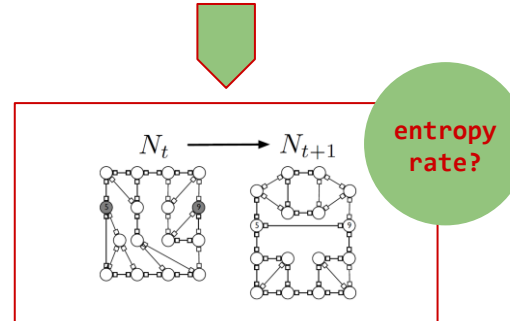
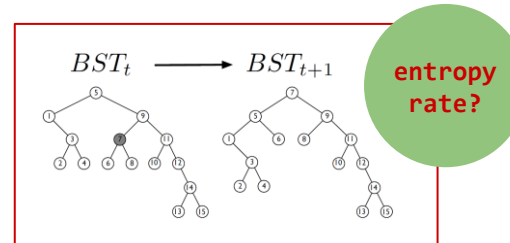
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)



Self-adjusting BST
(Dynamic Huffman coding)



More than
an analogy!

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

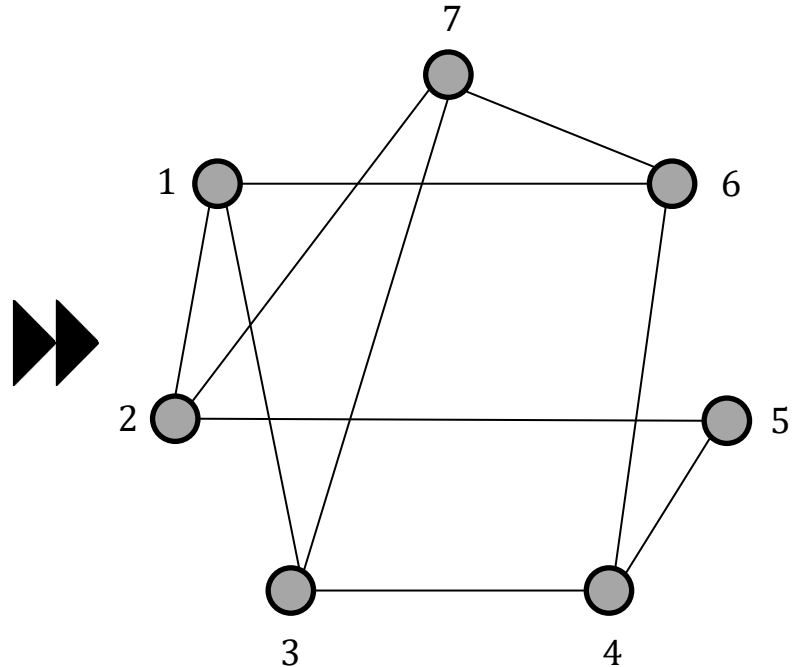
First result:
Demand-aware networks
of asymptotically
optimal route lengths.

Reduced expected **route lengths!**

Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

		Destinations						
		1	2	3	4	5	6	7
Sources	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

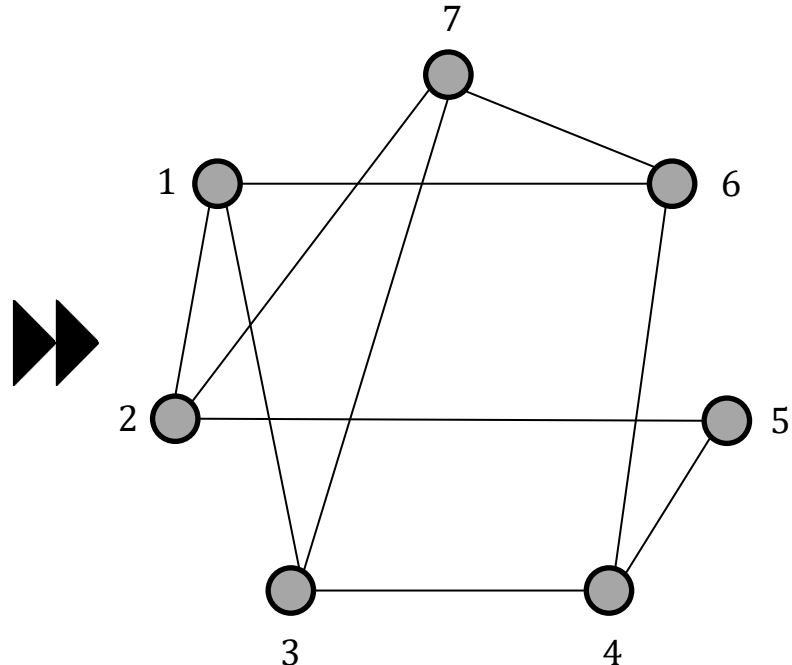


$$\text{ERL}(\mathcal{D}, N) = \sum_{(u,v) \in \mathcal{D}} p(u, v) \cdot d_N(u, v)$$

Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

		Destinations						
		1	2	3	4	5	6	7
Sources	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



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

Expected Route Length

Demand

DAN

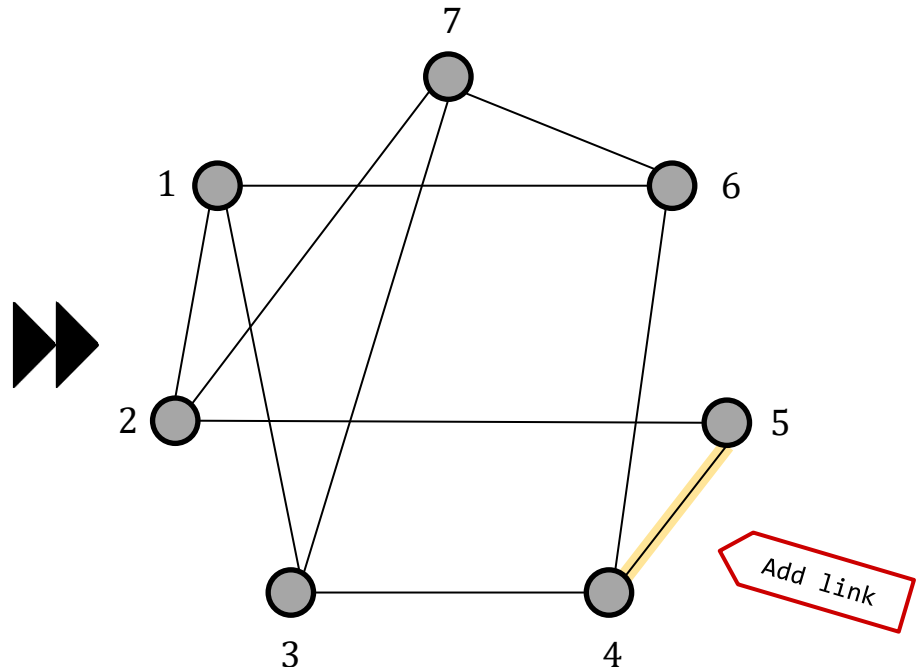
Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

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

Much from 4 to 5



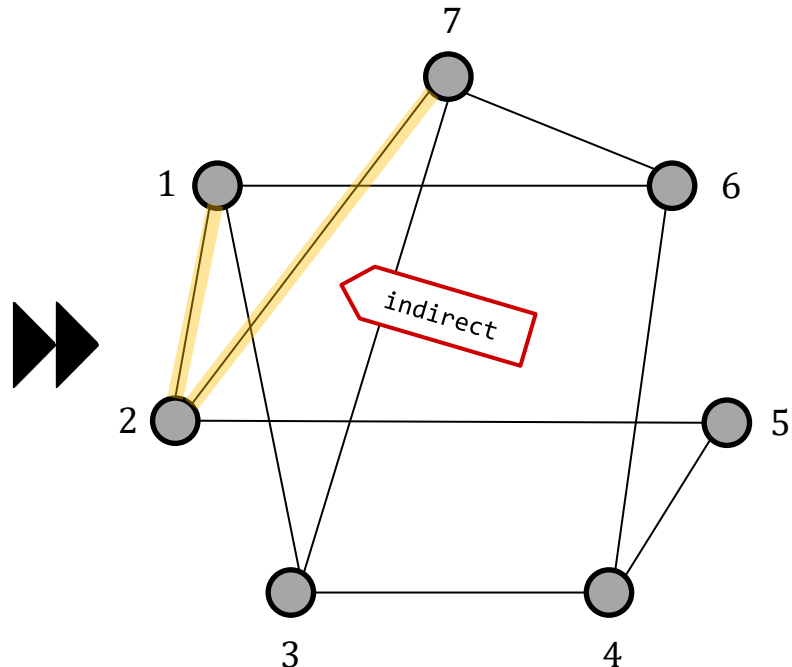
$$\text{ERL}(\mathcal{D}, N) = \sum_{(u,v) \in \mathcal{D}} p(u, v) \cdot d_N(u, v)$$

Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

Communicated with many

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



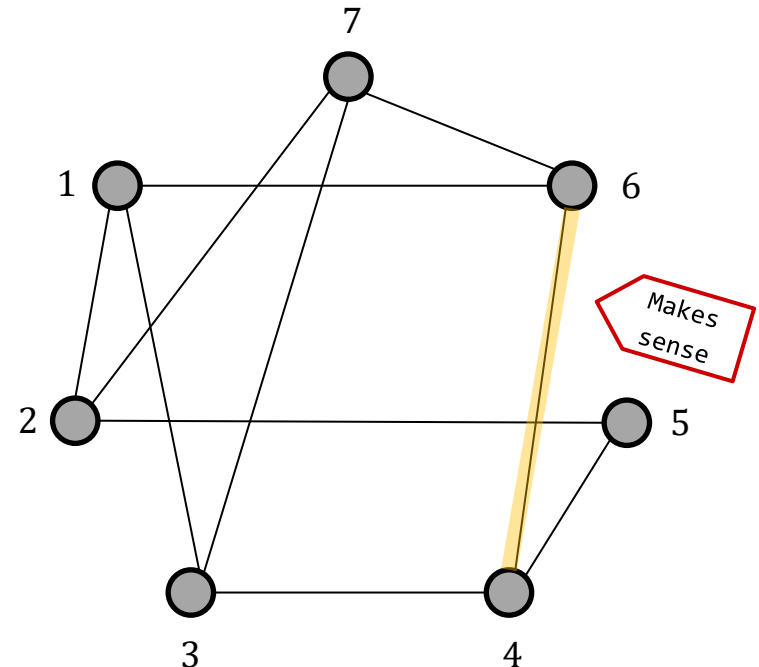
$$\text{ERL}(\mathcal{D}, N) = \sum_{(u,v) \in \mathcal{D}} p(u, v) \cdot d_N(u, v)$$

Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

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

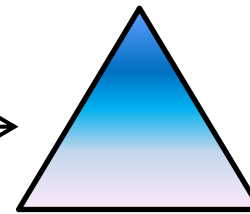
Don't
communicate



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

Algorithm: Idea

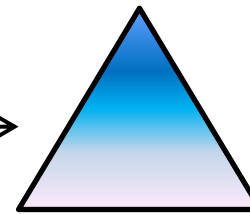
		Destinations						
		1	2	3	4	5	6	7
Sources	1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
	2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
	3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
	4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$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



Huffman tree:
“ego-tree”

Algorithm: Idea

		Destinations						
		1	2	3	4	5	6	7
Sources	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



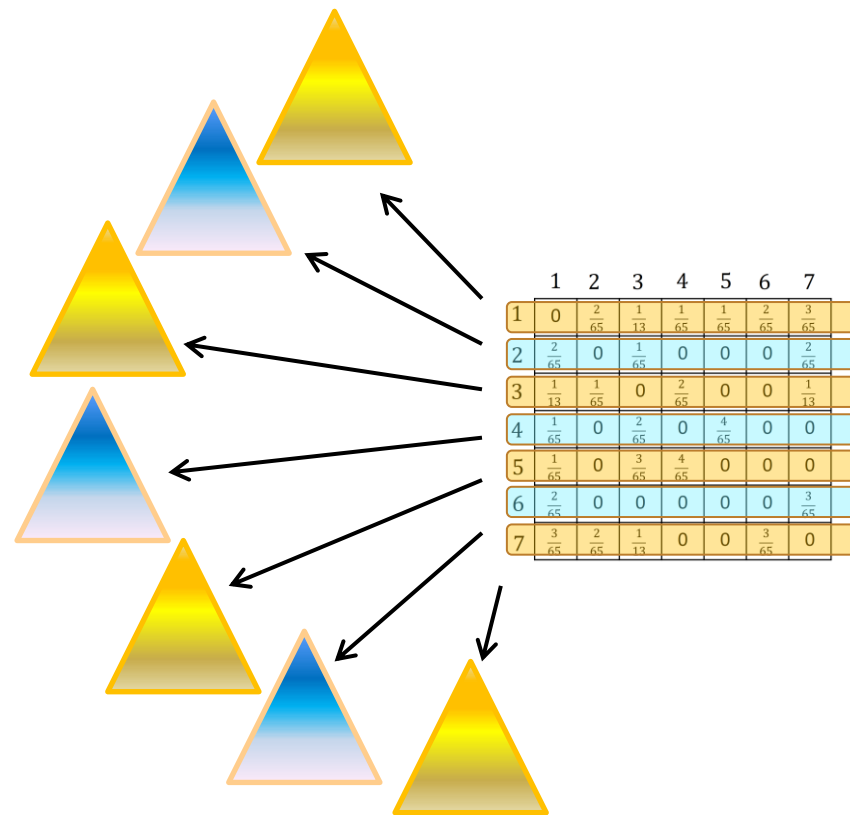
Huffman tree:
“ego-tree”

**Cost:
Entropy!**

Entropy Upper Bound

→ Idea for algorithm:

- Union of trees
- Reduce degree
- But keep distances



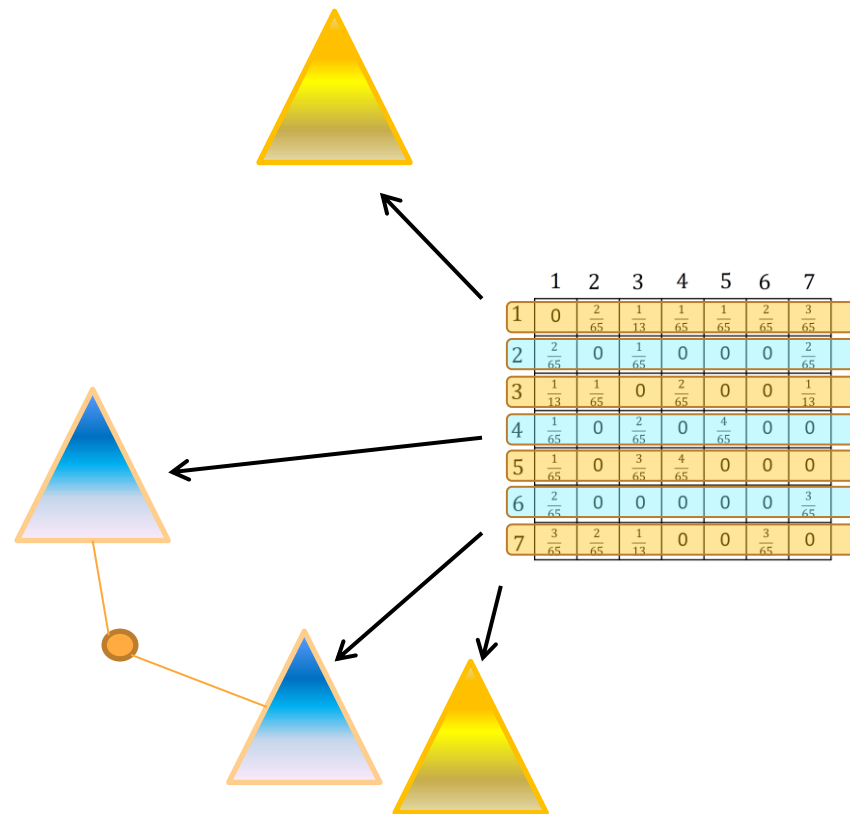
Entropy Upper Bound

→ Idea for algorithm:

- Union of trees
- Reduce degree
- But keep distances

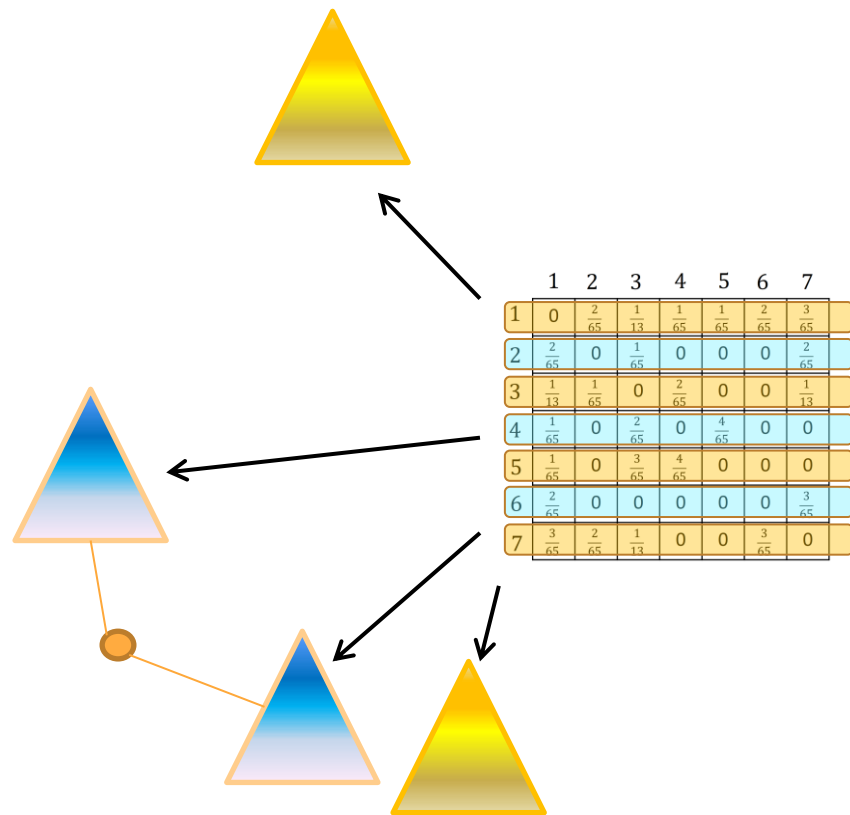
→ Ok for sparse demands

- Not everyone gets tree
- Helper nodes



Entropy Upper Bound

Gives upper
bound of
conditional
entropy*

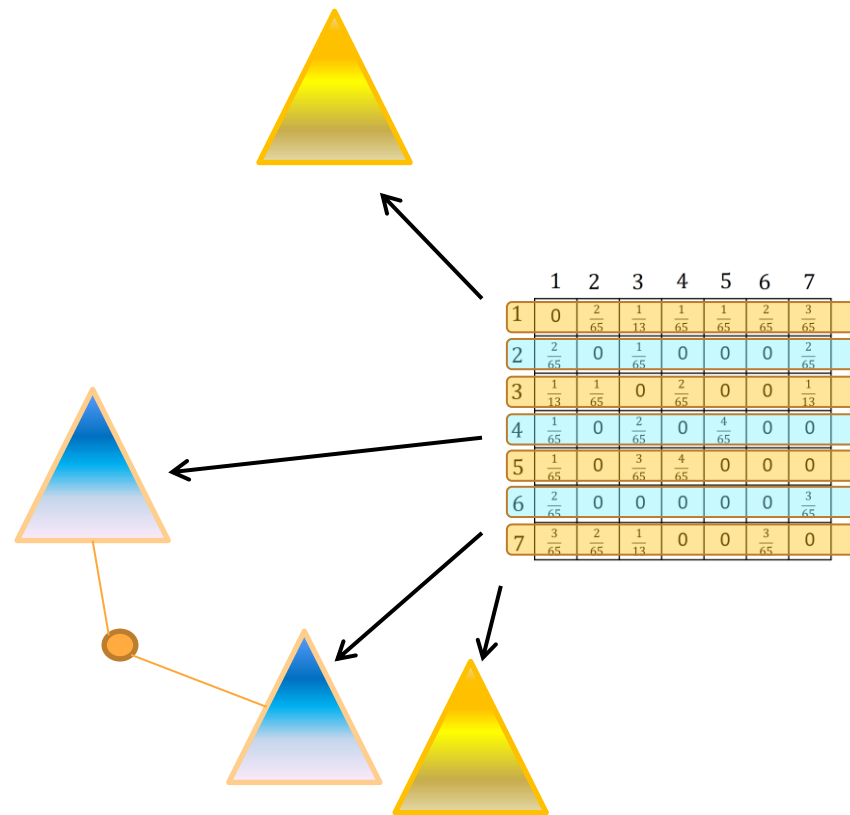


* $O(H(X/Y) + H(Y/X))$

Entropy Upper Bound

Gives upper
bound of
conditional
entropy*

Matching
lower bound
for sparse
demands!

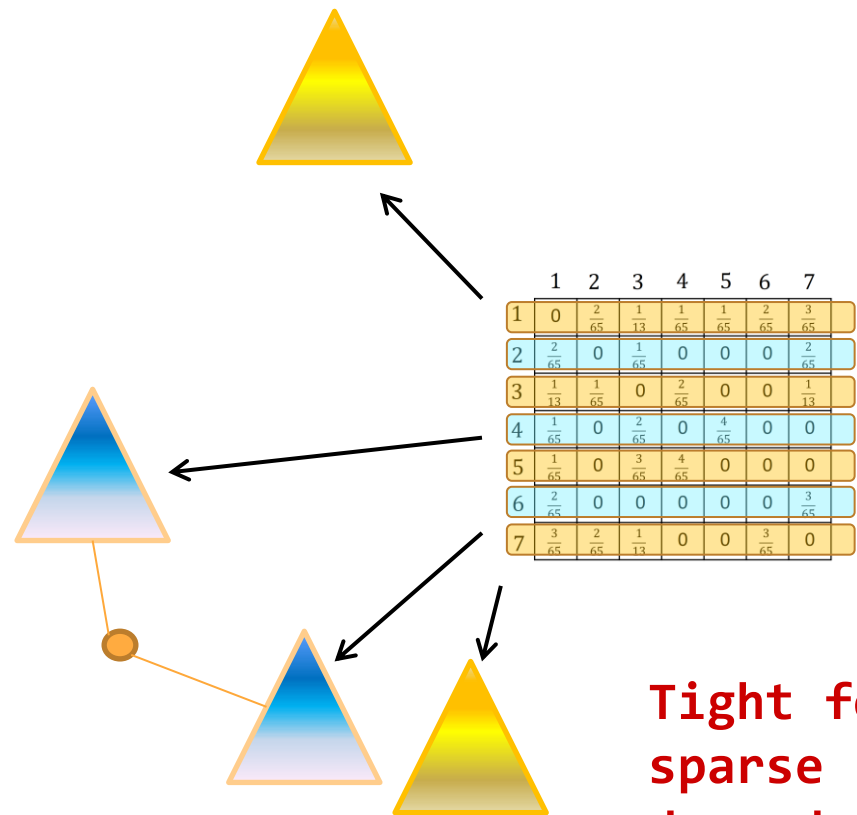


* $O(H(X/Y) + H(Y/X))$

Entropy Upper Bound

Gives upper
bound of
conditional
entropy*

Matching
lower bound
for sparse
demands!



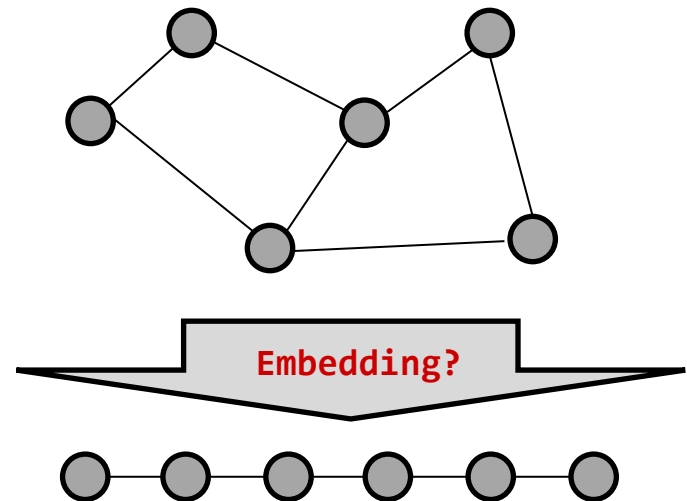
Tight for
sparse
demands!

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

Related Problem:

Virtual Network Embedding Problem (VNEP)

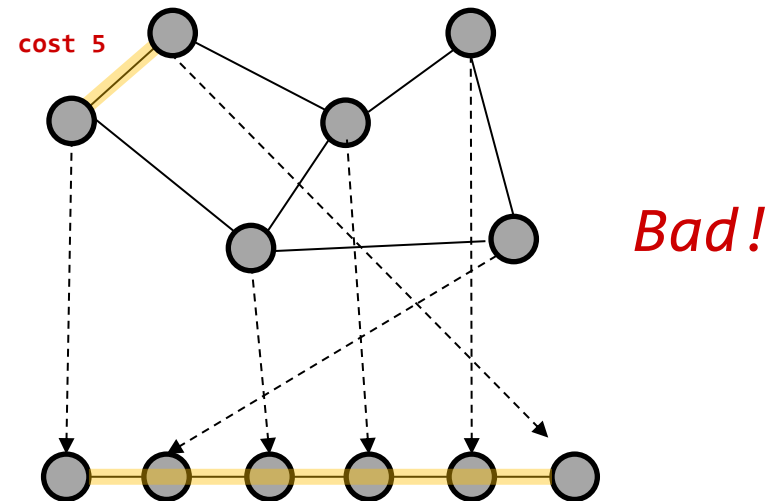
Example $\Delta=2$: A Minimum Linear
Arrangement (MLA) Problem
→ Minimizes sum of virtual
edges



Related Problem:

Virtual Network Embedding Problem (VNEP)

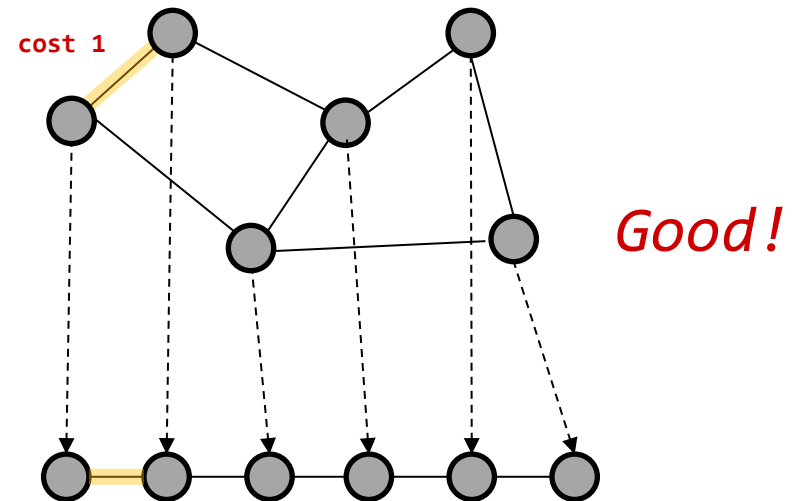
Example $\Delta=2$: A Minimum Linear
Arrangement (MLA) Problem
→ Minimizes sum of virtual
edges



Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\Delta=2$: A Minimum Linear
Arrangement (MLA) Problem
→ Minimizes sum of virtual
edges



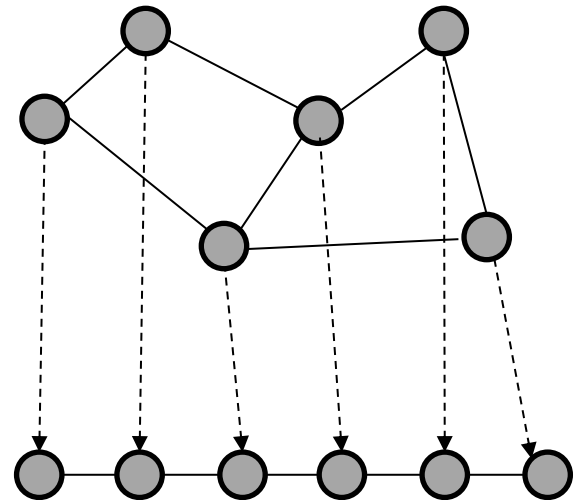
Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\Delta=2$: A Minimum Linear
Arrangement (MLA) Problem
→ Minimizes sum of virtual
edges

MLA is **NP-hard**

→ ... and so is our problem!



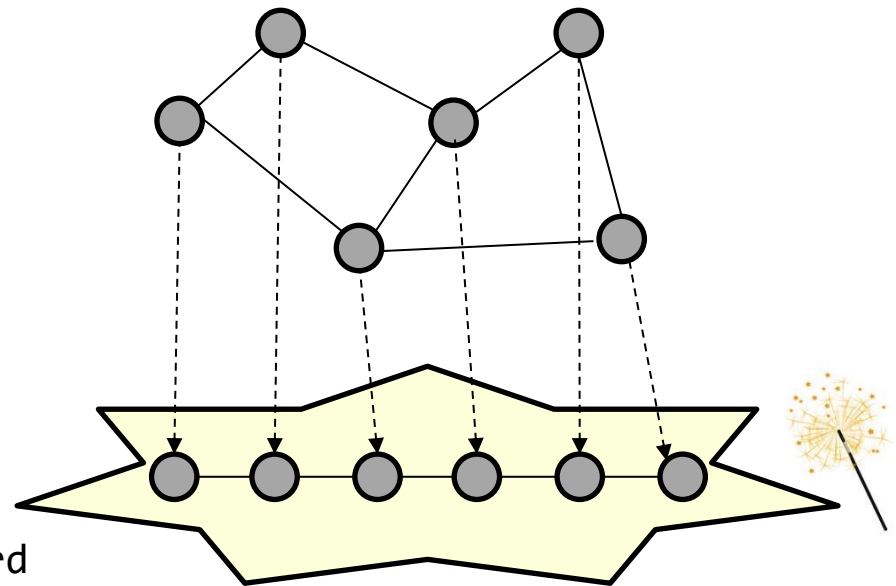
Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\Delta=2$: A Minimum Linear
Arrangement (MLA) Problem
→ Minimizes sum of virtual
edges

MLA is **NP-hard**
→ ... and so is our problem!

But what about $\Delta > 2$?
→ Embedding problem still hard
→ But we have a new **degree of
freedom!**



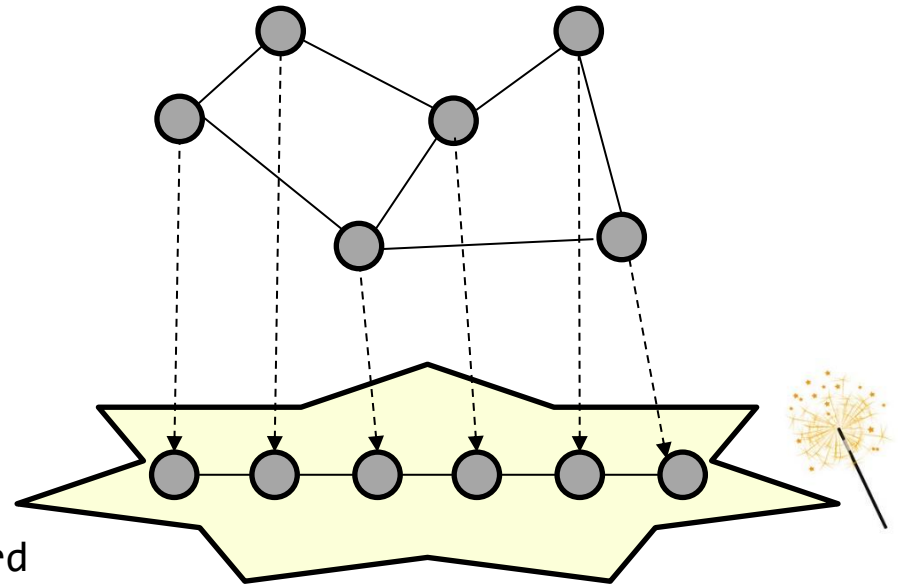
Related Problem:

Virtual Network Embedding Problem (VNEP)

Example $\Delta=2$: A Minimum Linear
Arrangement (MLA) Problem
→ Minimizes sum of virtual
edges

MLA is **NP-hard**
→ ... and so is our problem!

But what about $\Delta > 2$?
→ 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 **$\Omega(\log \log n)$ 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*

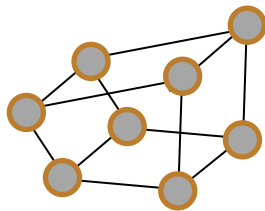
From Spanners to DANs

An Algorithm

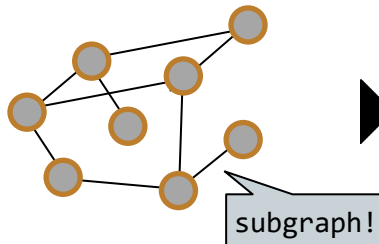
→ Yet, can *leverage the connection* to spanners sometimes!

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., $O(H(X/Y) + H(Y/X))$).

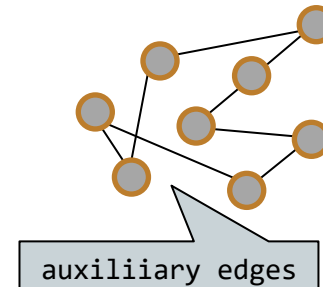
r-regular and
uniform demand:



Sparse, irregular
(constant) spanner:



Constant degree
optimal DAN (ERL
at most $\log r$):



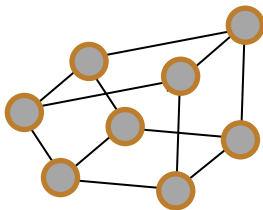
From Spanners to DANs

An Algorithm

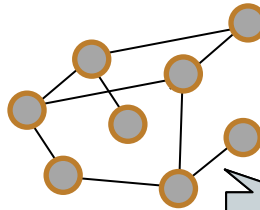
→ Yet, can *Leverage the connection* to spanners sometimes!

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., $O(H(X/Y) + H(Y/X))$).

r-regular and
uniform demand:



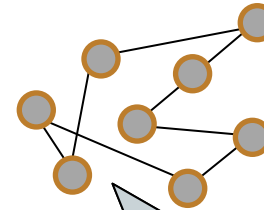
Sparse, irregular
(constant) spanner:



subgraph!



Constant degree
optimal DAN (ERL
at most $\log r$):



auxiliary edges

Our degree reduction
trick again!

Why optimal:
in *r*-regular graphs,
conditional entropy
is $\log r$.

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-*

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 as 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