

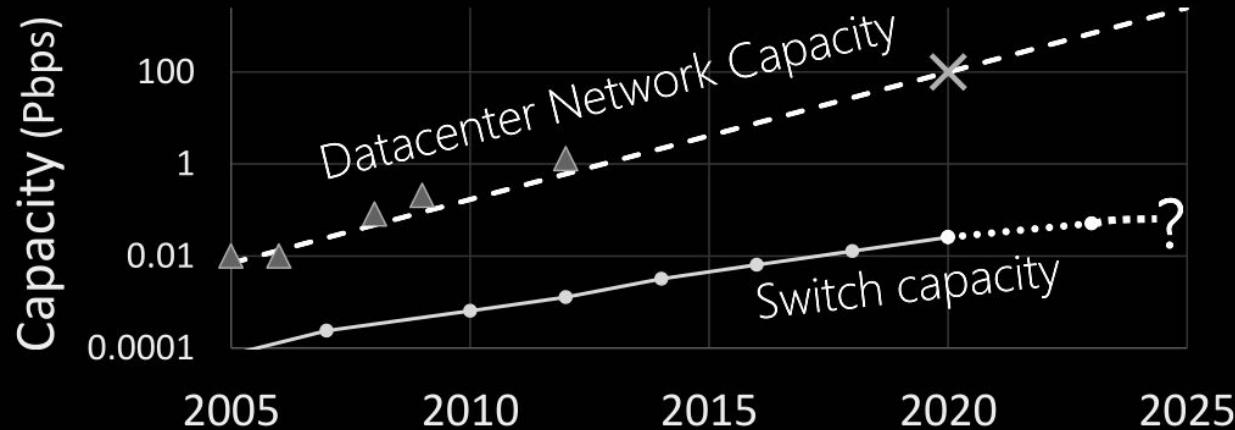
# Throughput Bounds of Reconfigurable Networks

*Insights & Reflections on Metrics for Collective Communication*

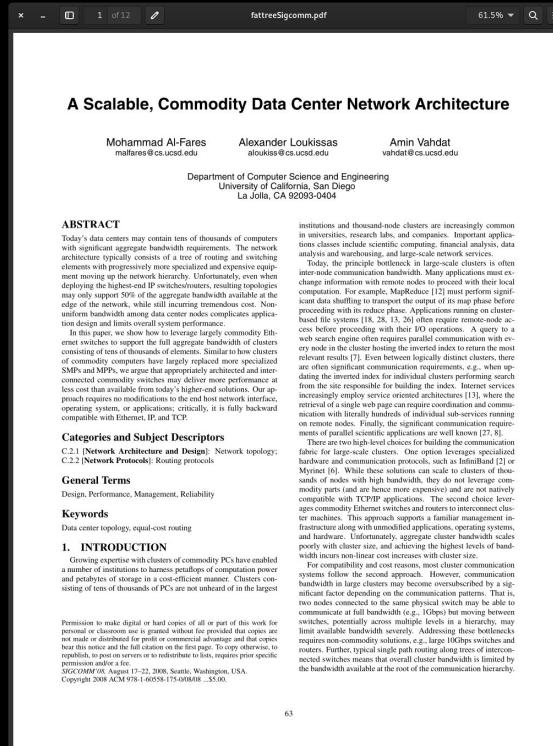
Vamsi Addanki



# Network Demand vs Capacity **Mismatch**



# Traditional Approach: Static Datacenter Topologies



## ABSTRACT

Today's data centers contain tens of thousands of computers with significant aggregate bandwidth requirements. The network architecture typically consists of a tree of routing and switching elements with progressively more specialized and expensive components moving from the edge to the core. Deploying the highest-PF IP switches, resulting topologies may only support 50% of the aggregate bandwidth available at the edge of the network, while still incurring tremendous cost at the server side. This paper shows how to design a commodity application design and limits overall system performance.

In this paper, we show how to leverage largely commodity Ethernet switches and commodity servers with commodity clusters consisting of tens of thousands of elements. Similar to how clusters of commodity computers have largely replaced more specialized routers and MPLS switches, commodity switches and commodity connected commodity switches can deliver more performance at less cost than available from today's higher-end solutions. Our application design is built to be load and host network interface operating system, or application, critic. It is fully backbone-compatible with Ethernet, TCP, and TCP.

## Categories and Subject Descriptors

C.2 [Network Architecture and Design]: Network topology; C.2.2 [Network Protocols]: Routing protocols

## General Terms

Design, Performance, Management, Reliability

## Keywords

Data center topology, equal-cost routing

## 1. INTRODUCTION

Growing expertise with clusters of commodity PC's has enabled a number of institutions to harness petabytes of computation power and petabytes of storage in a cost-efficient manner. Clusters consisting of tens of thousands of PCs are not unique at the largest

institutions, and thousand node clusters are increasingly common in universities, research labs, and companies. Important applications classes include scientific computing, financial analysis, data analysis and warehousing, and large-scale network services.

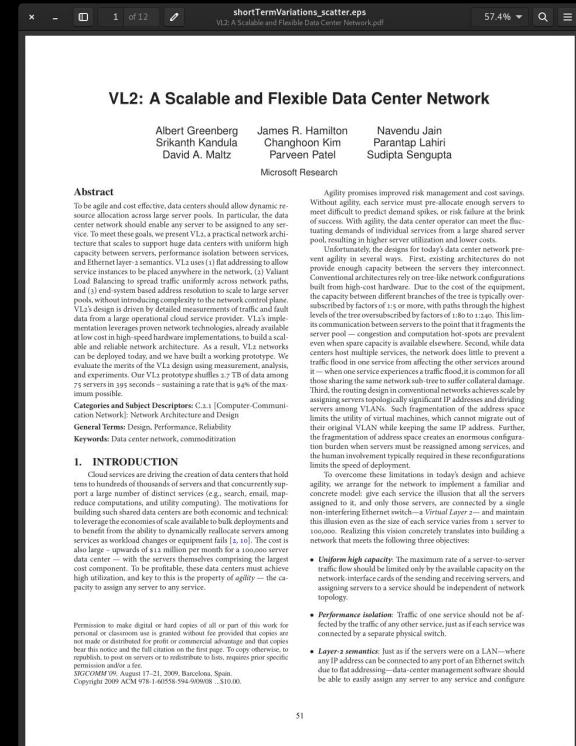
Today, the principle bottleneck in large-scale clusters is often interconnection bandwidth between nodes. Many applications exchange information with remote nodes to proceed with their local computations. For example, MapReduce [12] must perform significant inter-node communication to process the data it receives, proceeding with its reduce phase.

Applications running on cluster-based file systems [18, 28, 13, 26] often require remote-node access to store and retrieve data. For example, a search engine or a web search engine often requires parallel communication with every node in the cluster hierarchy to return the most relevant results. In addition, many distributed systems clusters are often significant communication requirements, e.g., when updating the inverted index for individual clusters performing search functions. In addition, clusters are often used as data centers, increasingly employ service oriented architectures [13], where the retrieval of a single web page can trigger coordination and communication between multiple clusters and multiple servers running on remote nodes. Finally, the significant communication requirement of parallel scientific applications will be well-known [27].

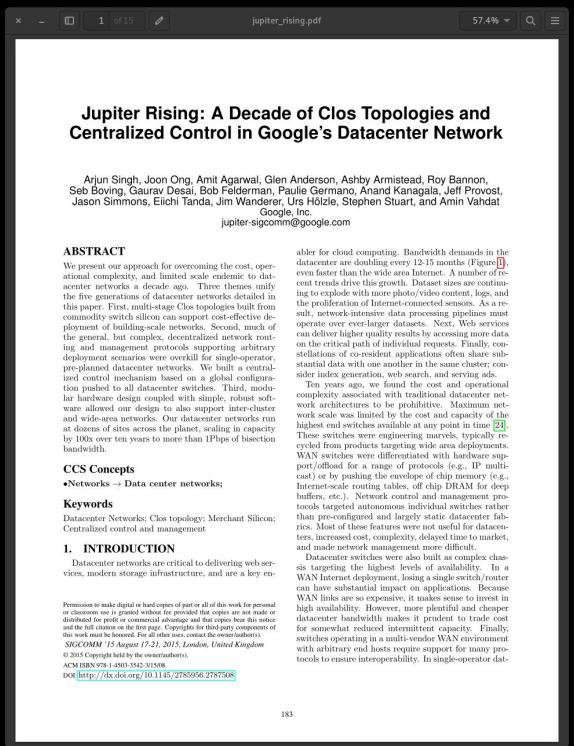
The second approach to building a datacenter is to use a commodity fabric for large-scale clusters. One option leverages specialized hardware and communication protocols, such as Infiniband [2] or Myrinet [10]. These fabrics can scale to clusters of thousands of nodes with high bandwidth, low latency, and very commodity parts (and are hence more expensive) and are not naturally compatible with TCP/IP applications.

The second choice leverages standard commodity hardware and commodity switches and router machines. This approach supports a familiar management infrastructure along with commodity applications, operating systems, and management tools. The main challenge is that commodity nodes poorly fit cluster size, and achieving the highest levels of bandwidth incurs non-linear cost increases with cluster size.

For commodity clusters, the traditional approach of communication systems follows the second approach. However, communication bandwidth in large clusters may become oversubscribed by a significant amount. For example, consider a cluster with two nodes. If two nodes connected to the same physical switch may be able to communicate at full bandwidth (e.g., 1Gb/s) but moving between servers in different clusters may be limited to 1Gb/s, then the cluster may limit available bandwidth severely. Addressing these bottlenecks requires non-commodity solutions, e.g., large 1Gb/s switches and routers. The problem is that the cost of these non-commodity connected switches means that overall cluster bandwidth is limited by the bandwidth available at the root of the communication hierarchy.



permissions to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for third-party components of this work must be honored. For more information about the use of this material, please contact the publisher. ACM 978-1-4503-3542-3/10/08 \$10.00.



Datacenter Networks; Clos topology; Merchant Silicon; Centralized control and management

## 1. INTRODUCTION

Datacenter networks are critical to delivering web services, modern storage infrastructure, and are a key en-

abler for cloud computing. Bandwidth demands in the datacenter are doubling every 12-15 months [Figure 1], even faster than the wide area Internet. A number of recent trends drive this growth. Dataset sizes are continuing to explode with more processing/video/medical, logs, and sensor data. The Internet of things is also a factor. In result, network-intensive data processing pipelines must operate over ever-larger datasets. New web services can deliver higher-quality results by accessing numerous datasets co-resident on the same cluster. Finally, substantial data with one another in the same cluster; content indexing, generation, web search, and serving logs.

To meet these challenges, the architectural and operational complexity associated with traditional datacenter network architectures is becoming untenable. Maintaining a single fabric is no longer feasible due to the density and variety of the hardware and switches available at any point in time [2].

These switches were engineering marvels, typically re-

quired to be built with high-performance components.

WAN traffic was often forwarded via a single switch port/offload for a range of protocols (e.g., IP multicast) or by pushing the envelope of chip memory (e.g., internet-scale routing tables, off-chip DRAM for deep buffering). Nevertheless, these protocols and their associated management overheads were not well-suited for autonomous individual switches rather than pre-configured and largely static datacenter fabrics.

Most of these features were useful for datacenter management, but were not delivered in a timely manner,

and made network management more difficult.

Datacenter switches were also built as complex ex-

# Traditional Approach: Static Datacenter Topologies

A Scalable, Commodity Data Center Network Architecture

Mohammad Al-Fares  
malfares@cs.ucsd.edu

Alexander Loukissas  
aloukissas@cs.ucsd.edu

Amin Vahdat  
vahdat@cs.ucsd.edu

Jellyfish: Networking Data Centers, Randomly

Ankit Singla<sup>1</sup>, Chi-Xiao Hong<sup>1</sup>, Lucian Popet<sup>2</sup>, Brighten Godfrey<sup>3</sup>

<sup>1</sup>University of Illinois at Urbana-Champaign

<sup>2</sup>University of California, Berkeley

## 1 Introduction

Data centers today form the backbone of cloud operations. A well-provisioned data center network is important to ensure that servers do not face bandwidth bottlenecks to utilization, to isolate services from each other, and to gain performance in workload placement. Rather than having to add more bandwidth to increase bandwidth availability [16]. As a result, a significant body of work has tackled the problem of building high network connectivity while minimizing costs.

One crucial problem that has been ignored in prior design is that of incremental expansion of the network, i.e., adding servers and network capacity incrementally to the data center. This is motivated by growth of the user base, but also by the need to support deployment of more bandwidth-intensive applications. Such expansion can be made feasible by either planned overprovisioning of speed and power, or by an original design that allows for a random topology to be built with the inherent slope gains of random walks. The potential of being significantly more flexible than past designs. Additional components: racks of servers or switches, and impact of server placement on network traffic and power consumption [8]. Key properties of such networks are determined by their underlying graph structure.

Indeed, network expansion is an important problem. Consider the network expansion of Facebook's data center server population from ~30,000 in November 2009 to more than 60,000 by June 2010 [24]. In fact, Facebook's data center server footprint is one of the few that is more than incremental in existing facilities ("adding capacity on a daily basis" [23]). For instance, Facebook announced that it will double the size of its facility at Princeton University by early 2011 [9]. Interestingly, it was also identified in incremental growth [1] as a useful technique to reduce cap-up-front [20].

Do we need high-bandwidth data center networks to support incremental growth? Consider the first network proposal [1] as an illustrative example. The entire structure is defined by the number  $k$  of the number of switches available. This is done in at least two ways. First, it makes the design space very sparse: full bandwidth bandwidth faults can only be built at sizes 3456, 8192, 27648, and 65536 corresponding to the commonly

"Finally, a coin-toss decided the first among the first two authors."

VL2: A Scalable and Flexible Data Center Network

Albert Greenberg  
Srikanth Kandula  
David A. Maltz  
James R. Hamilton  
Changhoon Kim  
Navendu Jain  
Parveen Patel  
Sudipta Sengupta  
Microsoft Research

Slim Fly: A Cost Effective Low-Diameter Network Topology

Maciej Besta  
ETH Zurich  
maciej.besta@inf.ethz.ch

Torsten Hoefer  
ETH Zurich  
hoefer@inf.ethz.ch

**Abstract**—We introduce a high-performance cost-effective network topology called Slim Fly that approaches the theoretically optimal network diameter. Slim Fly is based on graphs that approach the theoretical lower bound of  $\sqrt{N}$ . We analyze the performance of this design and compare it to both traditional and state-of-the-art datacenter topologies. We show that Slim Fly has several advantages over other topologies in latency, bandwidth, cost, and power consumption. Finally, we propose deadlock-free routing for this topology. Our results indicate that Slim Fly requires only 1.1 times the cost of the best topology in the literature, as well as a detailed cost and power model. Slim Fly enables constructing a cost effective and highly resilient datacenter and provides a low-latency solution for datacenters under different HPC workloads such as stencil or graph computations.

1 INTRODUCTION

Interconnection networks play an important role in today's large-scale computing systems. The importance of the network grows with increasing per-node (multi-core) performance and network density. As a result, the total number of nodes deployed in a warehoused HPC and data centers [8]. Key properties of such networks are determined by their underlying graph structure.

Since it seems that the more-hubs incremental expansion, we propose the opposite: a random network interconnect. The approach we call **Jellyfish**, consists of a random graph topology built with random walks. The inherent slope gains of random walks make the potential of being significantly more flexible than past designs. Additional components: racks of servers or switches, and impact of server placement on network traffic and power consumption [8]. Key properties of such networks are determined by their underlying graph structure.

In this paper we show that *lowering network diameter* is indispensable as many applications perform all-to-all communication [38]. Second, networks can account for as much as 33% of the total power consumption of a datacenter [2], and power consumption [2] and they should be cost and power efficient. Thus, low endpoint-to-endpoint latency is important for applications that require frequent data trading. Finally, topologies should be robust to link failures.

In this paper we show that *lowering network diameter* is indispensable as many applications perform all-to-all communication [38]. Second, networks can account for as much as 33% of the total power consumption of a datacenter [2], and power consumption [2] and they should be cost and power efficient. Thus, low endpoint-to-endpoint latency is important for applications that require frequent data trading. Finally, topologies should be robust to link failures.

This work is supported by the 2013 European Doctoral Fellowship in Parallel Computing.

Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network

Xpander: Towards Optimal-Performance Datacenters

Asaf Valadarsky<sup>\*</sup>  
asaf.valadarsky@mail.huji.ac.il

Gal Shahaf<sup>†</sup>  
gal.shahaf@mail.huji.ac.il

Michael Schapira<sup>\*</sup>  
schapiram@mail.huji.ac.il

Michael Dinitz<sup>‡</sup>  
mdinitz@cs.huji.edu

**Abstract** Despite extensive efforts to meet ever-growing demands, today's datacenters often exhibit far-from-optimal performance in terms of network latency and bandwidth. Resiliency to failures is another critical challenge. In this work, we show that the benefits of the most popular datacenter topologies are, in fact, derived from the fact that they are highly predictable ("expander-like" or (aka expanders) as their network topologies, thus unveiling a unifying theme of these proposals. We observe, however, that these proposals are either with respect to latency or bandwidth, or offer from seemingly insurmountable deployment challenges. We leverage these insights to present Xpander, a novel datacenter architecture that achieves near-optimal performance and provides a tangible alternative to existing datacenter topologies. Xpander design ideas from the rich graph-theoretic literature on constructing optimal expanders into an operational reality. We evaluate Xpander via extensive analytical, experimental, and real-world evaluations, and our implementation on an SDN-capable network testbed. Our results demonstrate that Xpander significantly outperforms both traditional and proposed datacenter designs. We discuss challenges to real-world deployment and explain how these can be resolved.

We argue that the quest for high-performance datacenter design is inextricably intertwined with the quest for optimal performance in mathematical and physical sciences on building good expanders. We seek a point in this design space that offers *near-optimal performance* guarantees while providing *practical* alternatives for today's datacenters (in terms of cost of capital, physical layout, bandwidth requirements, etc.) with their pros and cons. We introduce Xpander, a novel expander-datacenter architecture carefully engineered to achieve both these desiderata.

Implementing, utilizing and evaluating new network topologies has been proven in a large variety of contexts, ranging from parallel computing and high-performance computing [49, 15, 14, 19] to optical networks [41] and peer-to-peer networks [35, 36]. Our work is the first to study the performance and operational implications of utilizing expanders in the datacenter networking context, and seeking optimal design points in this specific domain (namely, Xpander). Indeed, the large body of research on expanders (e.g., construction, properties, and applications) in networks (e.g., throughput-related performance measures, explicit routing and congestion control protocols, deployment cost, incremental growth, etc.) remain little understood. We hope to elaborate on expanders, datacenter architectures, and Xpander.

1 Introduction

The rapid growth of Internet services is placing tremendous demands on datacenter networks. Yet, as evidenced by the extensive research on improving datacenter performance [2, 25, 6, 18, 2, 22, 2, 23], today's datacenters often exhibit far-from-optimal performance in terms of network utilization, resilience to failures, efficiency, amenability to incremental growth, and beyond.

1.1 The Secret to High Performance

We show that state-of-the-art proposals for next-generation datacenters, e.g., low-diameter networks such as Slim Fly [9], or random networks like Jellyfish [18], have an interesting property: the total capacity of the network is the total capacity of the nodes. So if the rest of the network is large with respect to the size of  $E$ , we present a formal definition of expanders in Section 2. Since this implies that an ex-

Network Size	Slim Fly	Torus 3D	Torus 2D	Tree 3D	Tree 2D	Full Tree	Random	Diamond	Grid	Dragonfly	Hypercube	Hypergrid
1000	~1.5	~1.5	~1.5	~1.5	~1.5	~1.5	~1.5	~1.5	~1.5	~1.5	~1.5	~1.5
2000	~2.5	~2.5	~2.5	~2.5	~2.5	~2.5	~2.5	~2.5	~2.5	~2.5	~2.5	~2.5
3000	~3.5	~3.5	~3.5	~3.5	~3.5	~3.5	~3.5	~3.5	~3.5	~3.5	~3.5	~3.5
4000	~4.5	~4.5	~4.5	~4.5	~4.5	~4.5	~4.5	~4.5	~4.5	~4.5	~4.5	~4.5
5000	~5.5	~5.5	~5.5	~5.5	~5.5	~5.5	~5.5	~5.5	~5.5	~5.5	~5.5	~5.5
6000	~6.5	~6.5	~6.5	~6.5	~6.5	~6.5	~6.5	~6.5	~6.5	~6.5	~6.5	~6.5
7000	~7.5	~7.5	~7.5	~7.5	~7.5	~7.5	~7.5	~7.5	~7.5	~7.5	~7.5	~7.5
8000	~8.5	~8.5	~8.5	~8.5	~8.5	~8.5	~8.5	~8.5	~8.5	~8.5	~8.5	~8.5
9000	~9.5	~9.5	~9.5	~9.5	~9.5	~9.5	~9.5	~9.5	~9.5	~9.5	~9.5	~9.5
10000	~10.5	~10.5	~10.5	~10.5	~10.5	~10.5	~10.5	~10.5	~10.5	~10.5	~10.5	~10.5

Fig. 1: Comparison of average number of hops (uniform matrix) in Slim Fly and other networks. Topology size is balanced or used to build balanced configurations (explained in Section III), allowing for highest global bandwidth.<sup>1</sup>

Slim Fly enables us to construct cost-efficient, full-bandwidth networks with small diameter and low diameter. It is the first topology to allow for a low-diameter datacenter using readily available high radix routers (e.g., 64-port Black Widow [35] or Mellanes 10Gb Port Director [5]). Larger HPC workloads such as stencil or graph computations can be constructed with diameter three as discussed in HIC-A. The main contributions of this work are:

- We design and analyze a new class of cost effective low-diameter network topologies based on random walks.
- We propose a new methodology to build datacenter topologies that minimize the number of nodes and links while maintaining low mean path length.
- We show that our approach is superior to the first intuition. Slim Fly using fewer cables and routers, is more tolerant towards link failures than comparable Dragonfly.
- We show a physical layout for a datacenter or an HPC center using standard components and low-power energy model.

<sup>1</sup>Number of random topologies are higher than shown due to the number of nodes similar to the lower values obtained using the Bounding Similarity to the lower values obtained using analytical formulas.

# Traditional Approach: Static Datacenter Topologies

The figure displays three overlapping windows from academic papers, illustrating the traditional approach to static datacenter topologies:

- f10-nsdi13.pdf**: A Scalable, Commodity-based Fault-tolerant Engineered Network (F10). This paper discusses F10, a fault-tolerant engineered network designed for cloud computing. It features a multi-tree topology where switches are not controlled by the OS, allowing for fast failover and recovery.
- shortTermVariations\_scatter.aps**: Scalable and Flexible Datacenter Topology. This work proposes a centralized manager that collects topology and failure information from switches to support fast failover and routing decisions.
- fatclique.pdf**: Understanding Lifecycle Management Complexity of Datacenter Topologies. This paper explores the complexity of lifecycle management in datacenters, noting that while some metrics like latency and bandwidth are well-understood, others like patch panel expansion steps and link re-wiring are less so.

**fatreeSigcomm.pdf**: A decade of Clos Topologies and Google's Datacenter Network. This paper provides a historical overview of Clos topology usage in Google's datacenters, highlighting its evolution and benefits.

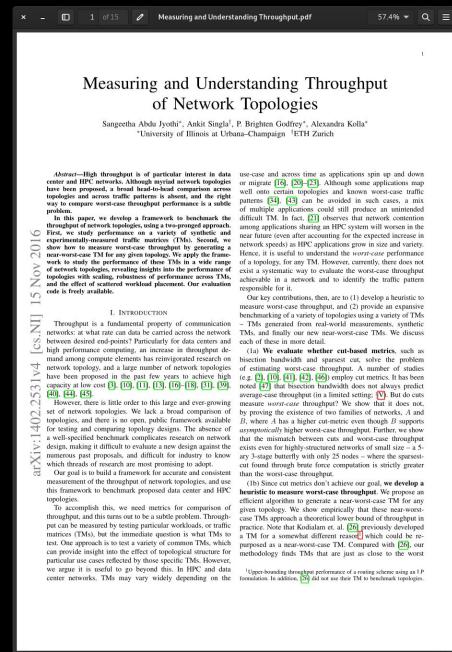
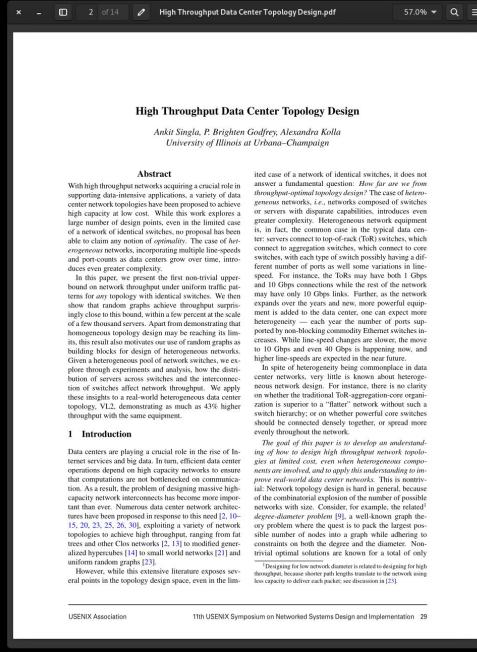
**jupiter\_rising.pdf**: optimal-Performance Datacenters. This work discusses the challenges of building high-performance datacenters, focusing on the need for efficient delivery of short bursts of traffic and the complexity of lifecycle management.

Traditional Approach: Static Datacenter Topologies

**Which topology has better throughput?**

# Traditional Approach: Static Datacenter Topologies

## Which topology has better throughput?



### High Throughput Data Center Topology Design

Ankit Singla<sup>1</sup>, P. Brighten Godfrey<sup>2</sup>, Alexandra Kolla<sup>1</sup>  
University of Illinois at Urbana-Champaign

#### Abstract

With high-performance computing becoming more important for supporting data-intensive applications, a variety of new data center network topologies have been proposed to achieve high capacity at low cost. Thus far, this work explores a large number of these topologies, but to the limit of our knowledge, no one has been able to claim any notion of optimality. The case of heterogeneous topologies is particularly interesting because of the need for port-constraints as data centers grow over time, increasing degrees even greater complexity.

In this paper, we propose the first non-trivial upper-bound on network throughput under uniform traffic patterns for all topologies with identical switches. We then show that this bound is very close to the bound, with a few percent at the scale of a few thousand servers. As the network grows, if a new server is added to the data center, one can expect more heterogeneity – each year the number of ports supported by each switch increases, while the number of switches creates. While line-speed changes are slower, the move to 10 Gbps and even 40 Gbps is happening now, and to 100 Gbps is not far off.

In spite of heterogeneity being commonplace in data center networks, very little is known about heterogeneous network topologies. For this reason, we also study whether the traditional Tor-aggregation-core organization is superior to a “flatter” network without such a strict hierarchy. Our results show that nodes in a core should be connected densely together, or spread more evenly throughout the network.

The rest of this paper develops an understanding of how to design high throughput network topologies at limited costs even when heterogeneous components are present. We believe that this work will help improve real-world data center networks. This is non-trivial. Network topology design is hard in general, particularly when it comes to data centers, which have unique constraints with respect to size and placement of servers. For example, consider the related *degree-diameter problem* [1], a well-known graph problem that asks for the maximum number of nodes in a graph after allowing to set constraints on both the degree and the diameter. This is a well-studied problem that has been proposed to this need [2, 10–14, 20].

Our methodology is to propose a new class of topologies to achieve high throughput, ranging from fat trees and other Class II networks [2, 13] to modified generalized hypercubes [14] and world networks [21] and uniform random graphs [23].

However, while this extensive literature exposes several points in the topology design space, even the limi-

tated case of a network of identical switches, it does not answer a fundamental question: *How are we to know that one topology is better than another?* The answer is that no generic networks, i.e., networks composed of switches with separate capabilities, introduce even more complexity. In this paper, we propose a novel methodology to claim any notion of optimality. The case of heterogeneous topologies is particularly interesting because of the need for port-constraints as data centers grow over time, increasing degrees even greater complexity.

In this paper, we propose the first non-trivial upper-bound on network throughput under uniform traffic patterns for all topologies with identical switches. We then show that this bound is very close to the bound, with a few percent at the scale of a few thousand servers. As the network grows, if a new server is added to the data center, one can expect more heterogeneity – each year the number of ports supported by each switch increases, while the number of switches creates. While line-speed changes are slower, the move to 10 Gbps and even 40 Gbps is happening now, and to 100 Gbps is not far off.

In spite of heterogeneity being commonplace in data center networks, very little is known about heterogeneous network topologies. For this reason, we also study whether the traditional Tor-aggregation-core organization is superior to a “flatter” network without such a strict hierarchy. Our results show that nodes in a core should be connected densely together, or spread more evenly throughout the network.

The rest of this paper develops an understanding of how to design high throughput network topologies at limited costs even when heterogeneous components are present. We believe that this work will help improve real-world data center networks. This is non-trivial. Network topology design is hard in general, particularly when it comes to data centers, which have unique constraints with respect to size and placement of servers. For example, consider the related *degree-diameter problem* [1], a well-known graph problem that asks for the maximum number of nodes in a graph after allowing to set constraints on both the degree and the diameter. This is a well-studied problem that has been proposed to this need [2, 10–14, 20].

Our methodology is to propose a new class of topologies to achieve high throughput, ranging from fat trees and other Class II networks [2, 13] to modified generalized hypercubes [14] and world networks [21] and uniform random graphs [23].

However, while this extensive literature exposes several points in the topology design space, even the limited case of a network of identical switches, it does not answer a fundamental question: *How are we to know that one topology is better than another?* The answer is that no generic networks, i.e., networks composed of switches with separate capabilities, introduce even more complexity. In this paper, we propose a novel methodology to claim any notion of optimality. The case of heterogeneous topologies is particularly interesting because of the need for port-constraints as data centers grow over time, increasing degrees even greater complexity.

In this paper, we propose the first non-trivial upper-bound on network throughput under uniform traffic patterns for all topologies with identical switches. We then show that this bound is very close to the bound, with a few percent at the scale of a few thousand servers. As the network grows, if a new server is added to the data center, one can expect more heterogeneity – each year the number of ports supported by each switch increases, while the number of switches creates. While line-speed changes are slower, the move to 10 Gbps and even 40 Gbps is happening now, and to 100 Gbps is not far off.

In spite of heterogeneity being commonplace in data center networks, very little is known about heterogeneous network topologies. For this reason, we also study whether the traditional Tor-aggregation-core organization is superior to a “flatter” network without such a strict hierarchy. Our results show that nodes in a core should be connected densely together, or spread more evenly throughout the network.

### 1 Introduction

Data centers are playing a crucial role in the rise of Internet services and big data. In turn, efficient data center operation is critical to the success of these services. That computations are not bottlenecked on communication. As this is often the problem of designing massive high-capacity data centers, it is important to understand what is better than ever. Numerous data center network architectures have been proposed to this need [2, 10–14, 20].

One approach is to propose a new class of topologies to achieve high throughput, ranging from fat trees and other Class II networks [2, 13] to modified generalized hypercubes [14] and world networks [21] and uniform random graphs [23].

However, while this extensive literature exposes several points in the topology design space, even the limi-

tated case of a network of identical switches, it does not answer a fundamental question: *How are we to know that one topology is better than another?* The answer is that no generic networks, i.e., networks composed of switches with separate capabilities, introduce even more complexity. In this paper, we propose a novel methodology to claim any notion of optimality. The case of heterogeneous topologies is particularly interesting because of the need for port-constraints as data centers grow over time, increasing degrees even greater complexity.

In this paper, we propose the first non-trivial upper-bound on network throughput under uniform traffic patterns for all topologies with identical switches. We then show that this bound is very close to the bound, with a few percent at the scale of a few thousand servers. As the network grows, if a new server is added to the data center, one can expect more heterogeneity – each year the number of ports supported by each switch increases, while the number of switches creates. While line-speed changes are slower, the move to 10 Gbps and even 40 Gbps is happening now, and to 100 Gbps is not far off.

In spite of heterogeneity being commonplace in data center networks, very little is known about heterogeneous network topologies. For this reason, we also study whether the traditional Tor-aggregation-core organization is superior to a “flatter” network without such a strict hierarchy. Our results show that nodes in a core should be connected densely together, or spread more evenly throughout the network.

The rest of this paper develops an understanding of how to design high throughput network topologies at limited costs even when heterogeneous components are present. We believe that this work will help improve real-world data center networks. This is non-trivial. Network topology design is hard in general, particularly when it comes to data centers, which have unique constraints with respect to size and placement of servers. For example, consider the related *degree-diameter problem* [1], a well-known graph problem that asks for the maximum number of nodes in a graph after allowing to set constraints on both the degree and the diameter. This is a well-studied problem that has been proposed to this need [2, 10–14, 20].

Our methodology is to propose a new class of topologies to achieve high throughput, ranging from fat trees and other Class II networks [2, 13] to modified generalized hypercubes [14] and world networks [21] and uniform random graphs [23].

However, while this extensive literature exposes several points in the topology design space, even the limited case of a network of identical switches, it does not answer a fundamental question: *How are we to know that one topology is better than another?* The answer is that no generic networks, i.e., networks composed of switches with separate capabilities, introduce even more complexity. In this paper, we propose a novel methodology to claim any notion of optimality. The case of heterogeneous topologies is particularly interesting because of the need for port-constraints as data centers grow over time, increasing degrees even greater complexity.

In this paper, we propose the first non-trivial upper-bound on network throughput under uniform traffic patterns for all topologies with identical switches. We then show that this bound is very close to the bound, with a few percent at the scale of a few thousand servers. As the network grows, if a new server is added to the data center, one can expect more heterogeneity – each year the number of ports supported by each switch increases, while the number of switches creates. While line-speed changes are slower, the move to 10 Gbps and even 40 Gbps is happening now, and to 100 Gbps is not far off.

In spite of heterogeneity being commonplace in data center networks, very little is known about heterogeneous network topologies. For this reason, we also study whether the traditional Tor-aggregation-core organization is superior to a “flatter” network without such a strict hierarchy. Our results show that nodes in a core should be connected densely together, or spread more evenly throughout the network.

# Reconfigurable Datacenter Networks

- Generalization of the design space: *Topology can change over time*
- Static networks are a special case

# Reconfigurable Datacenter Networks

- Generalization of the design space: *Topology can change over time*
- Static networks are a special case



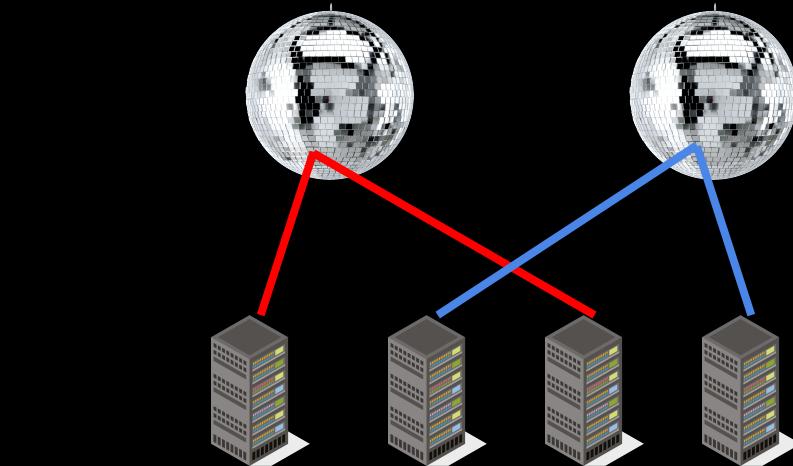
*Sirius [Sigcomm 2020]*

# Reconfigurable Datacenter Networks

- Generalization of the design space: *Topology can change over time*
- Static networks are a special case

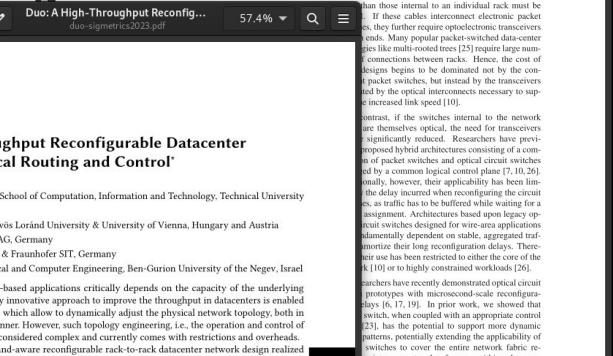
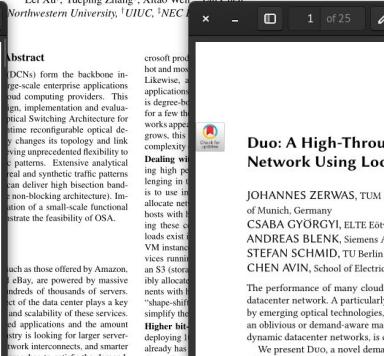
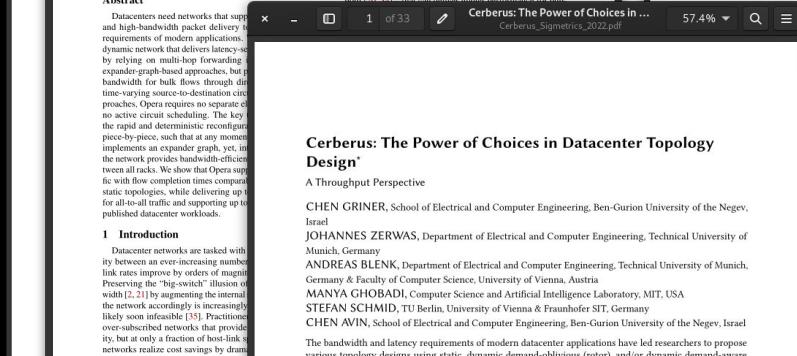
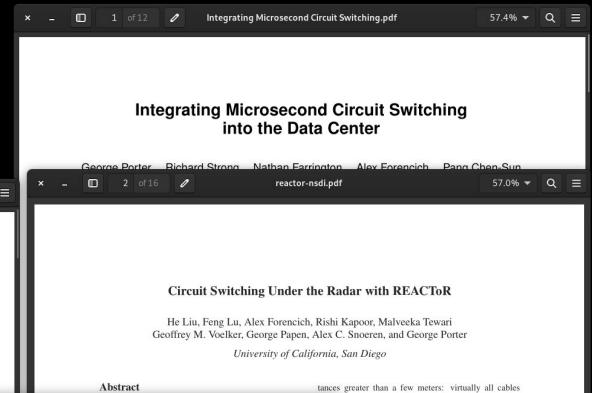
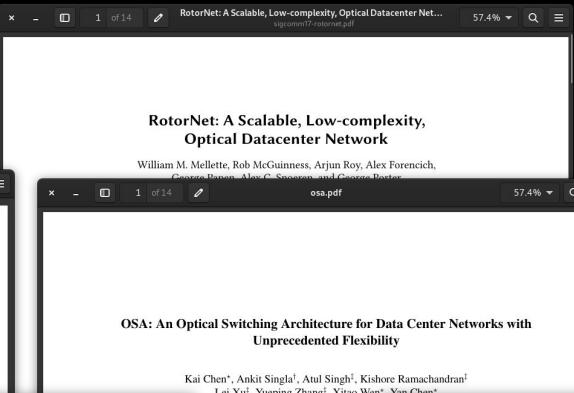
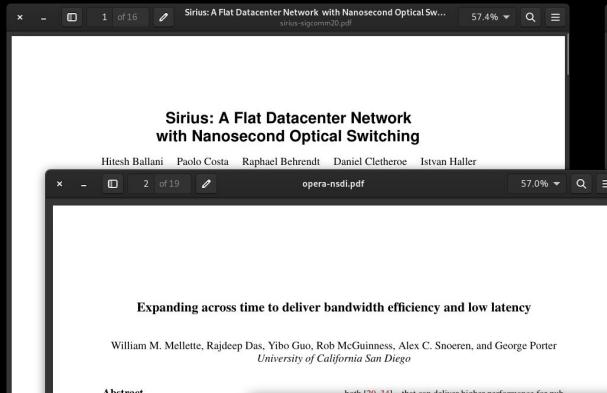


*Sirius [Sigcomm 2020]*



*ProjectToR [Sigcomm 2016]*

# Reconfigurable Datacenter Networks

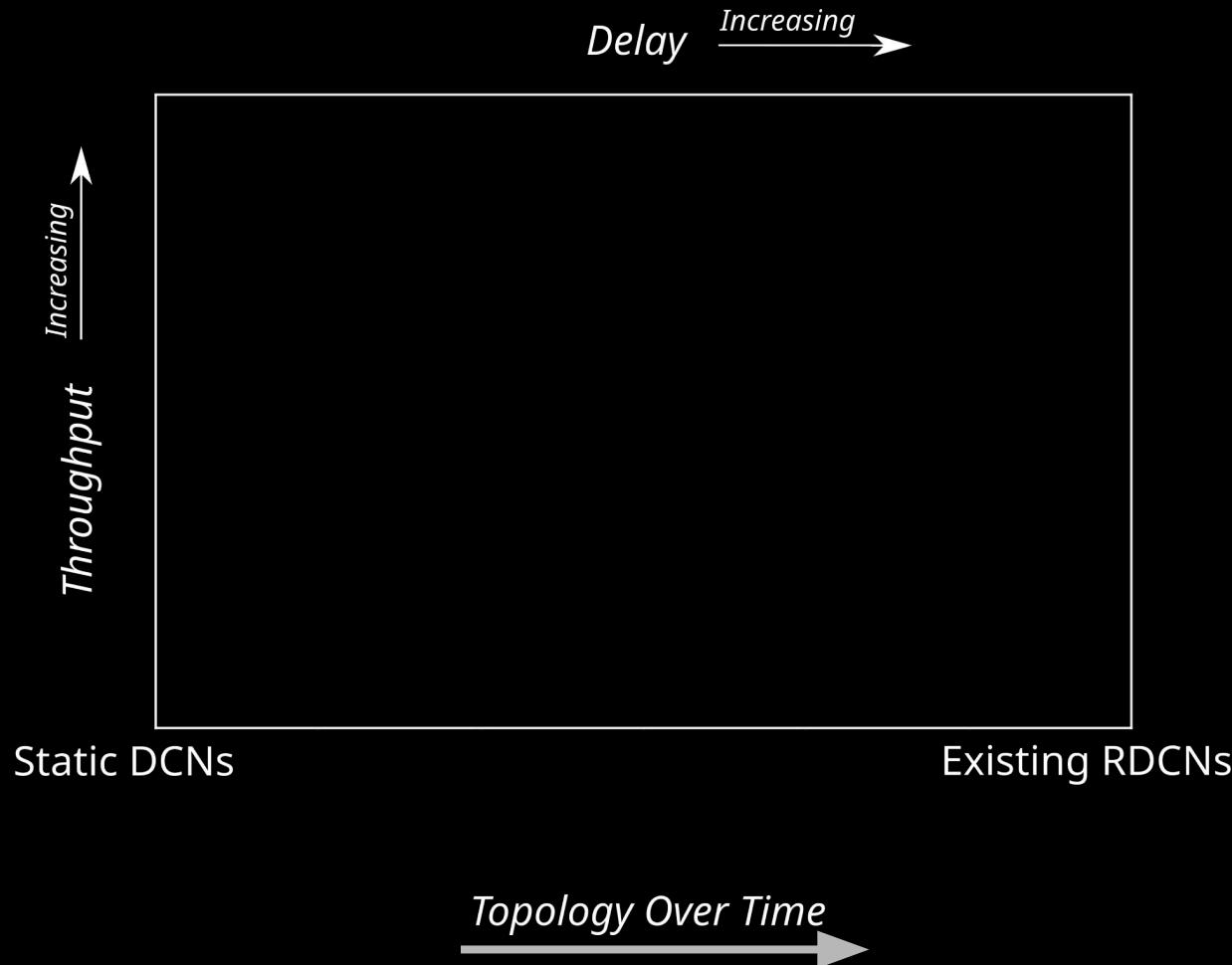


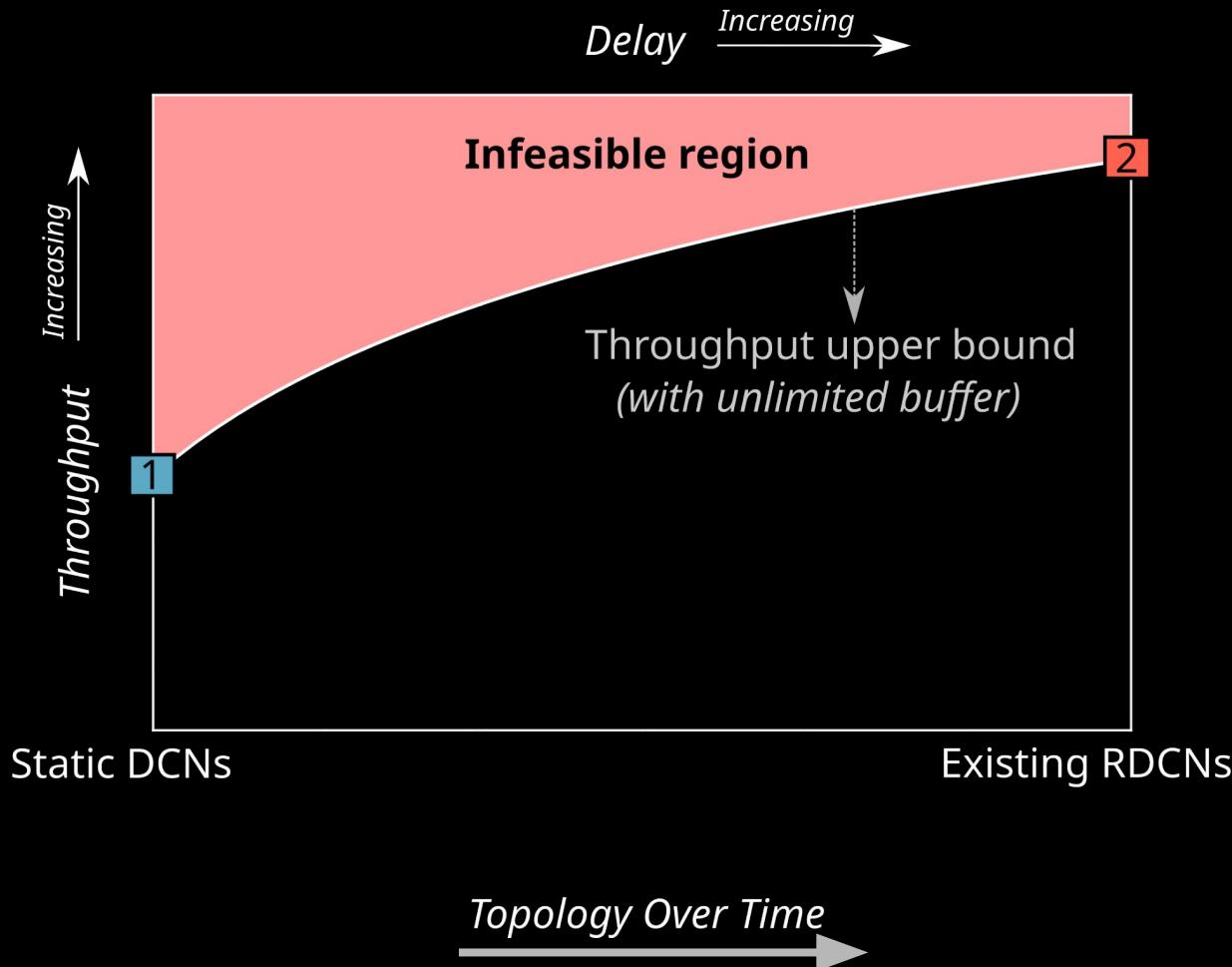
# Reconfigurable Datacenter Networks

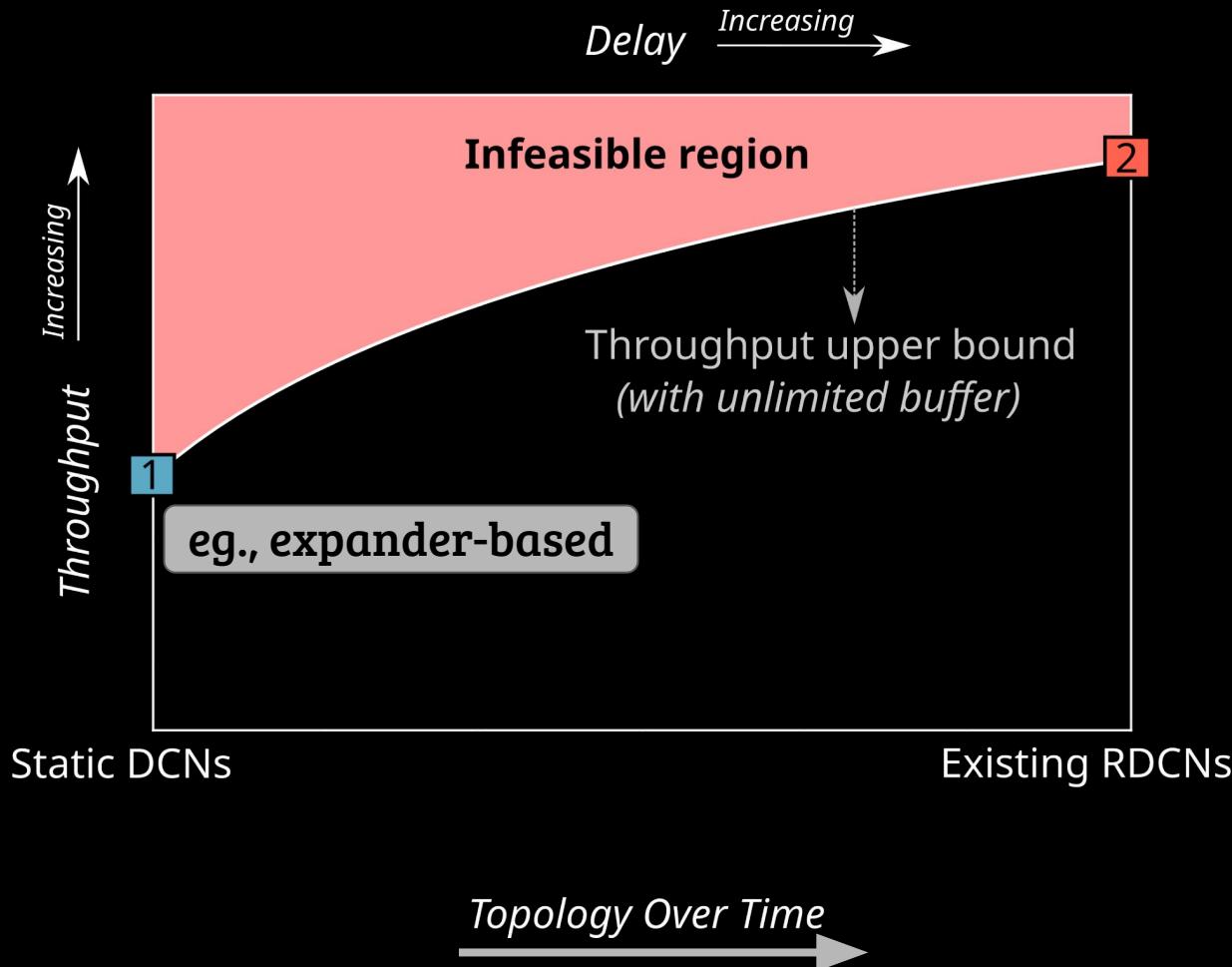
**Which topology has better throughput?**

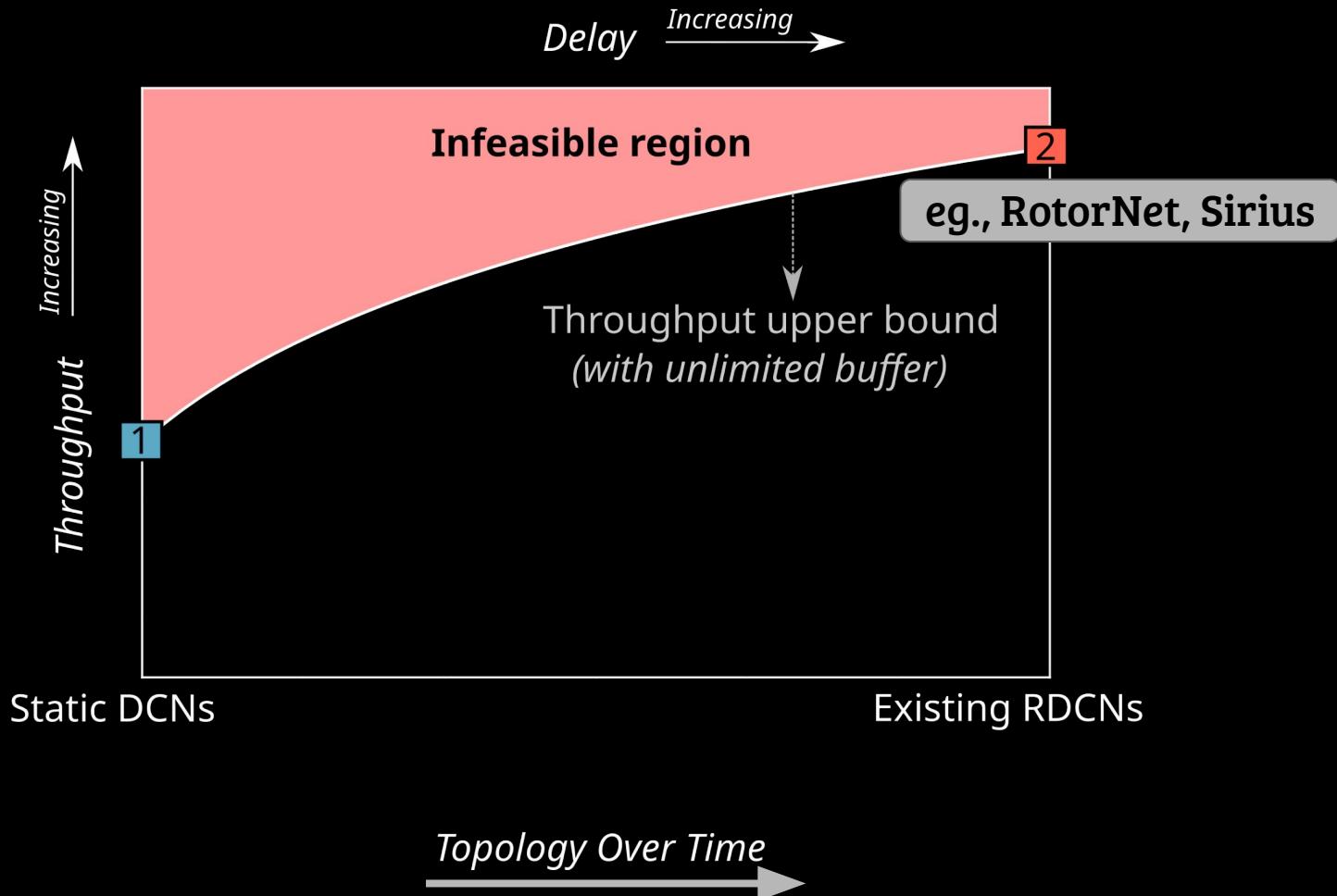


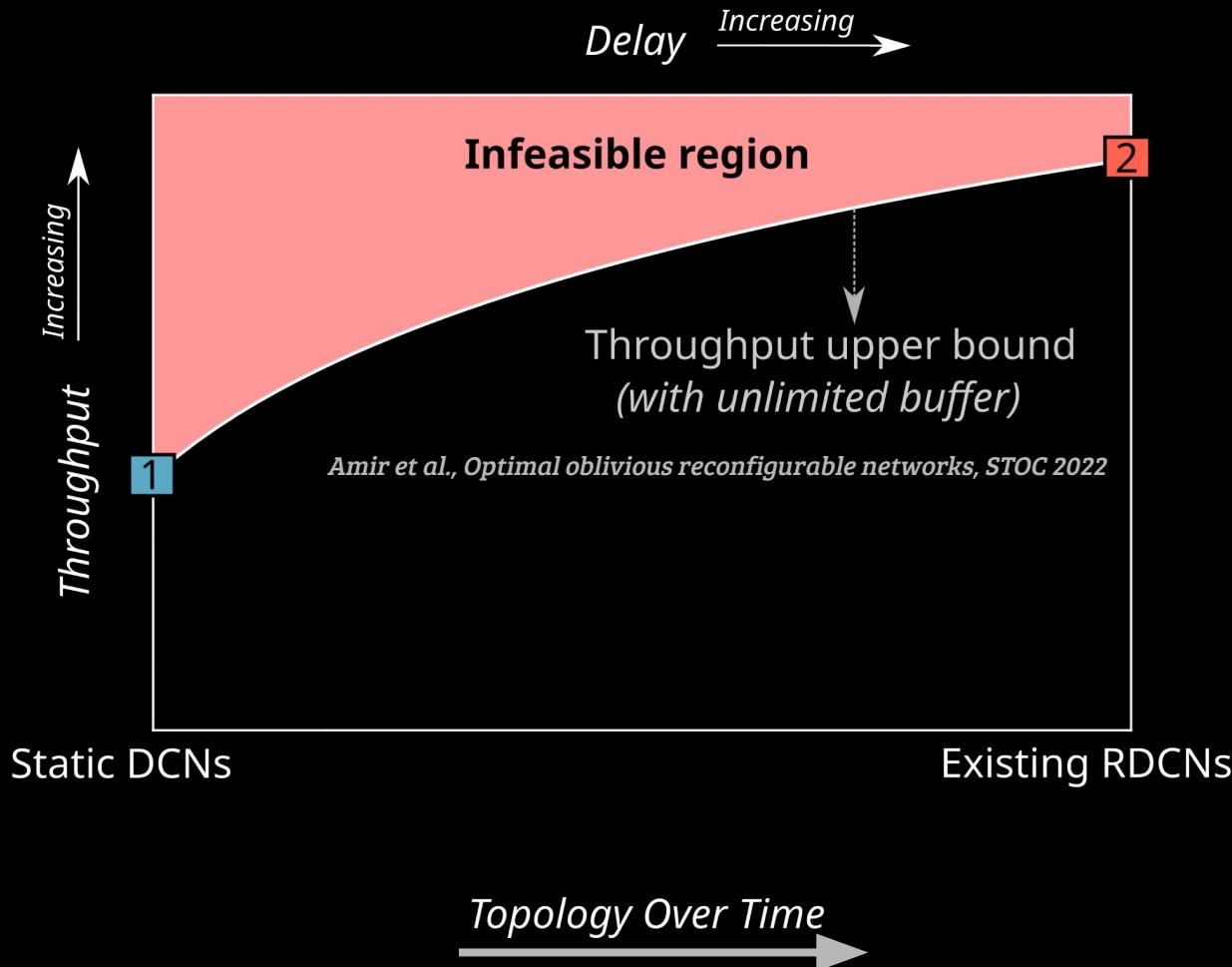
*Topology Over Time*

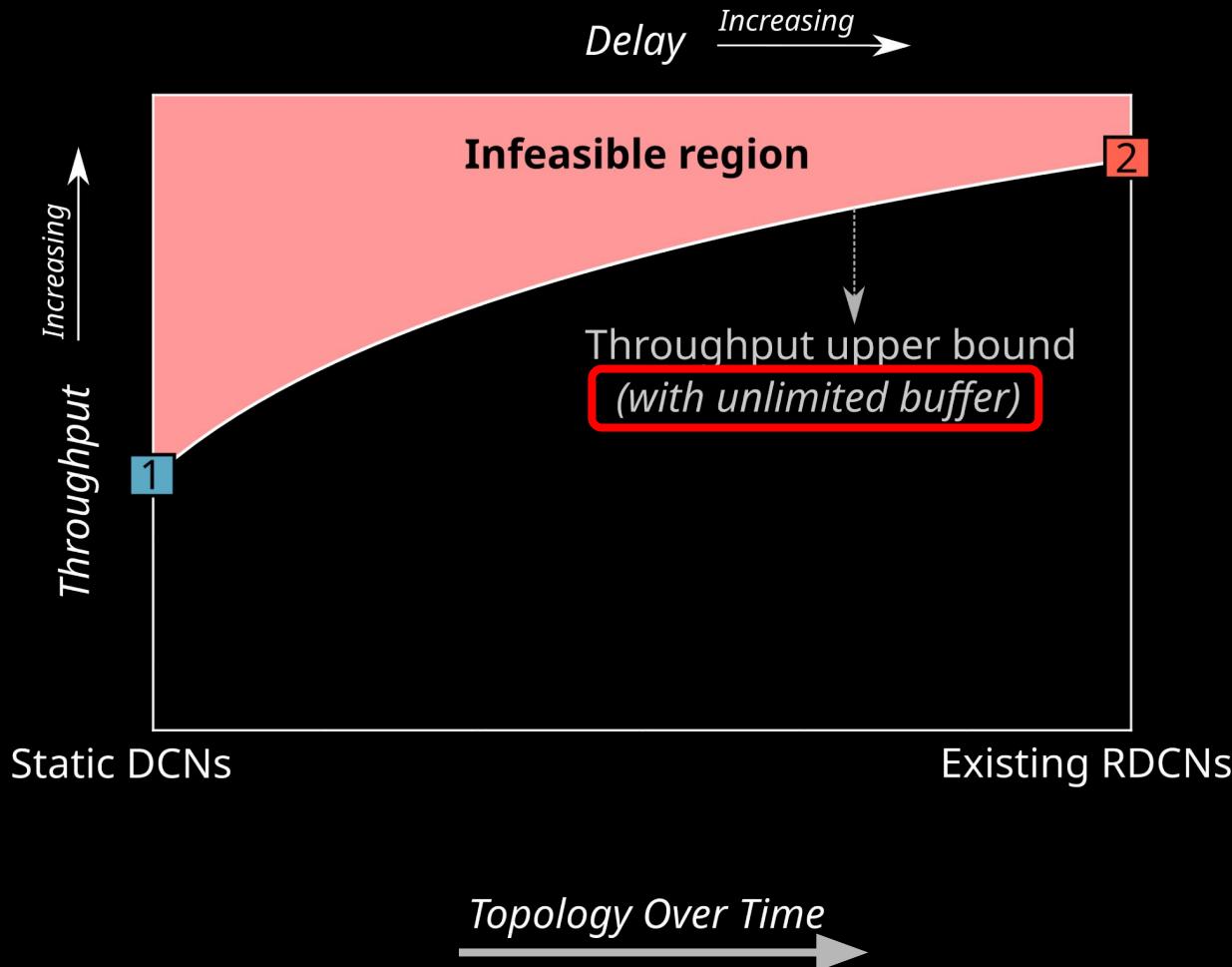


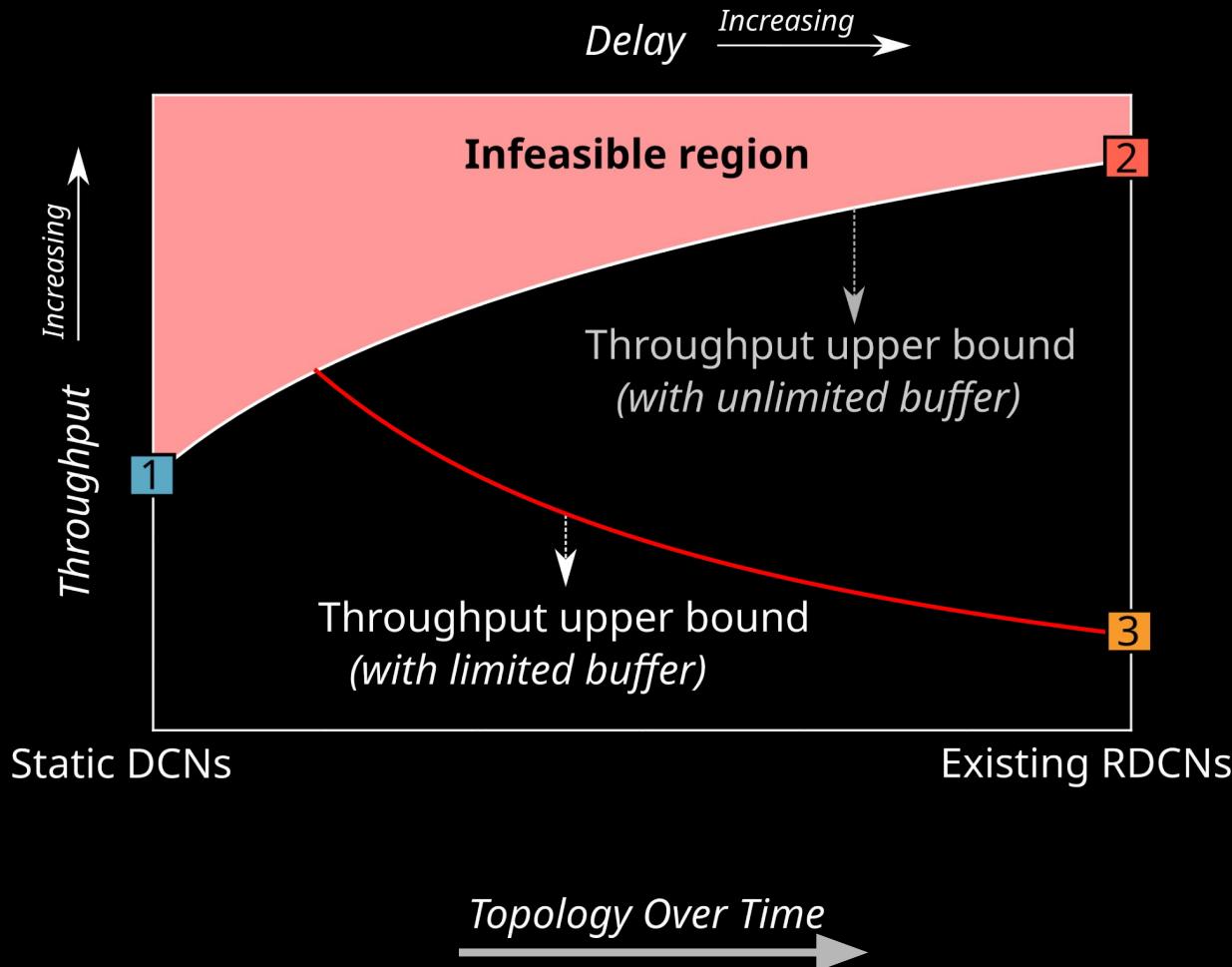


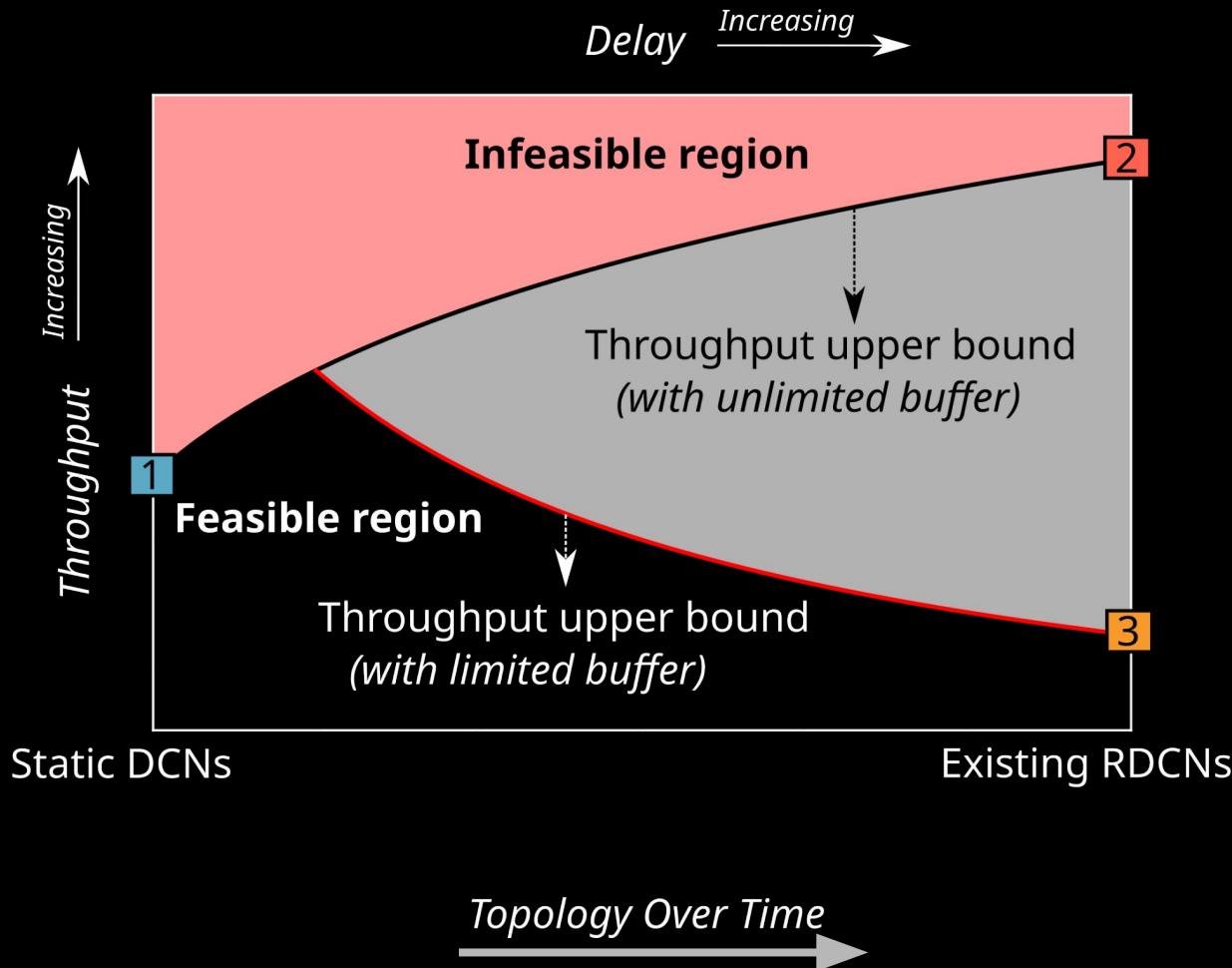


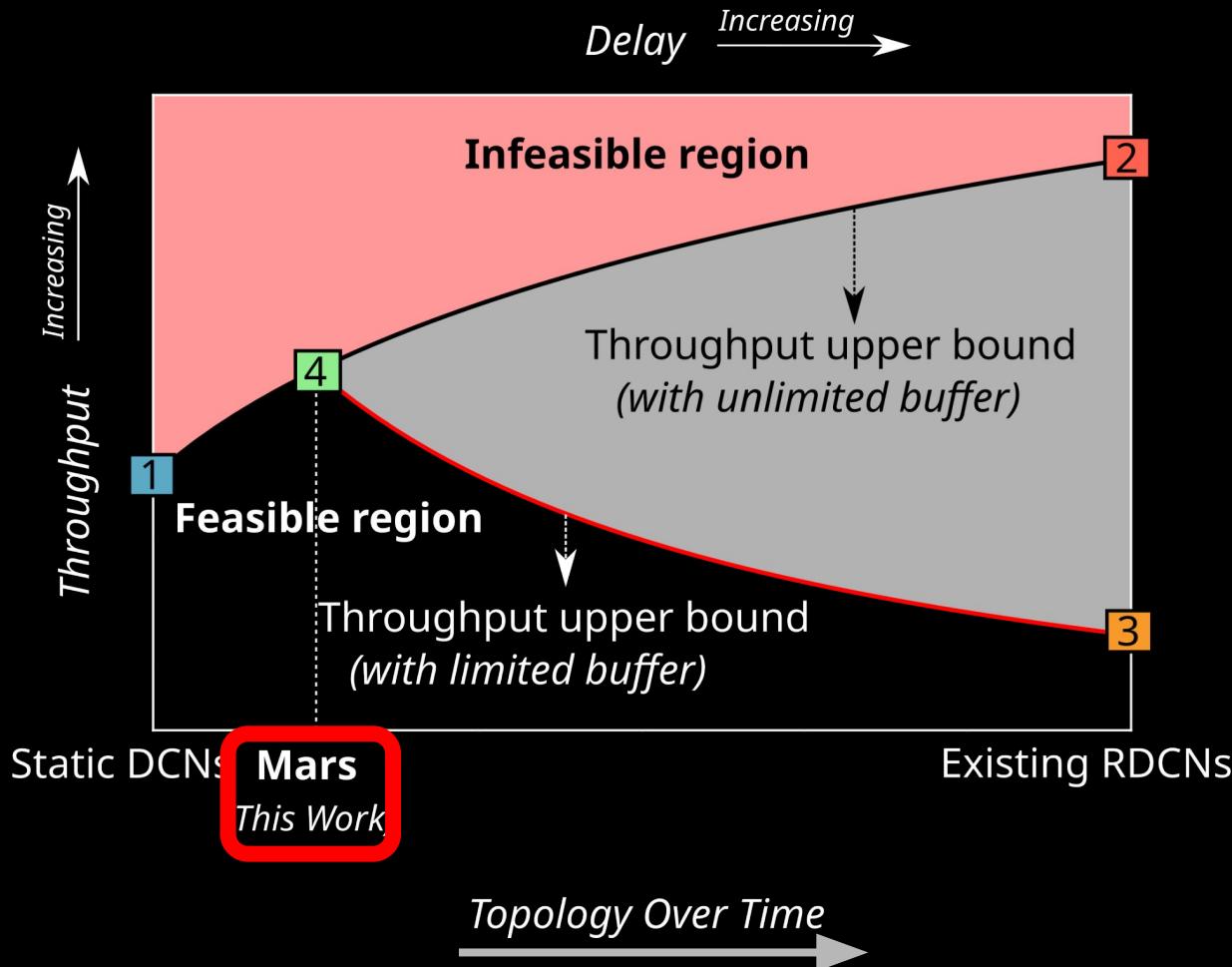


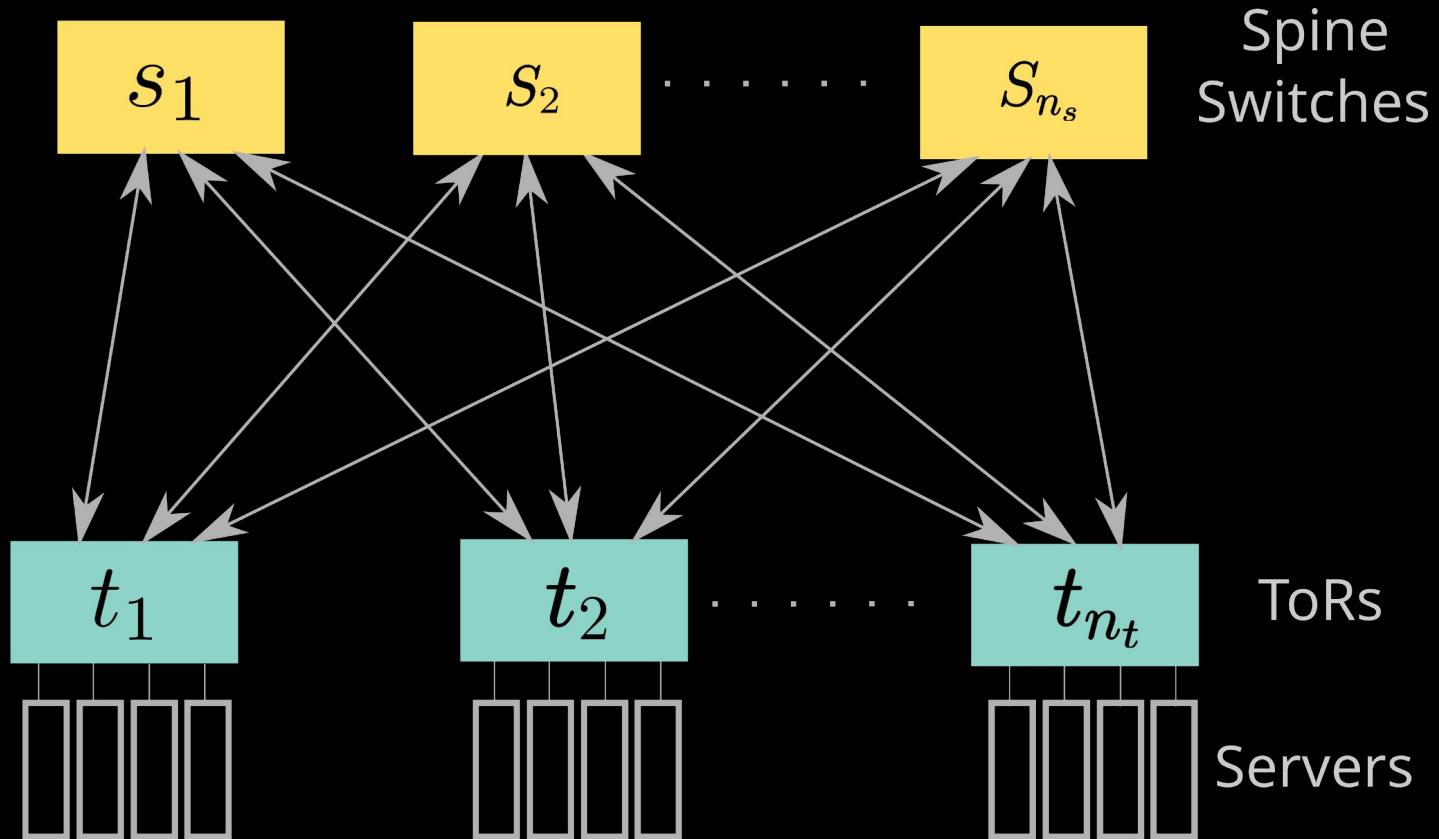




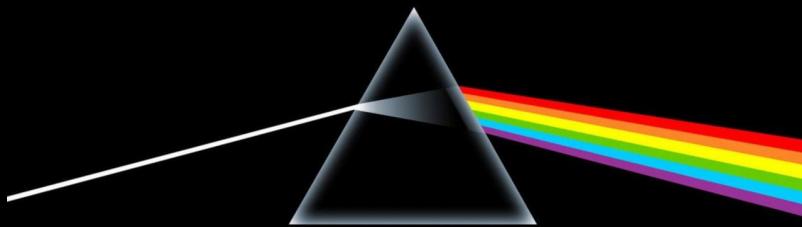






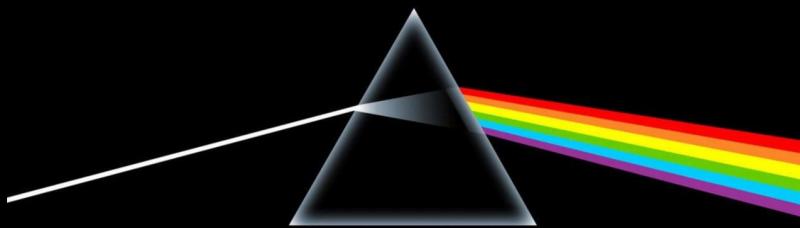


# Emerging Technologies - Optical Networks on the Rise

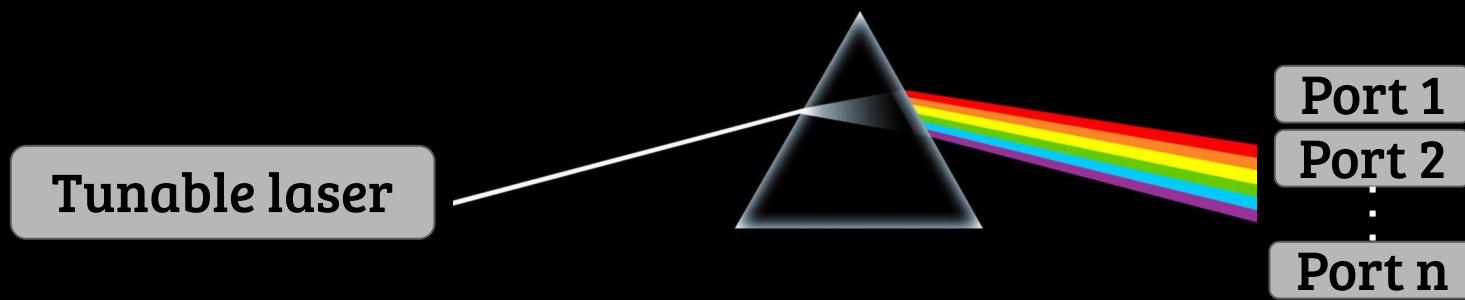


# Emerging Technologies - Optical Networks on the Rise

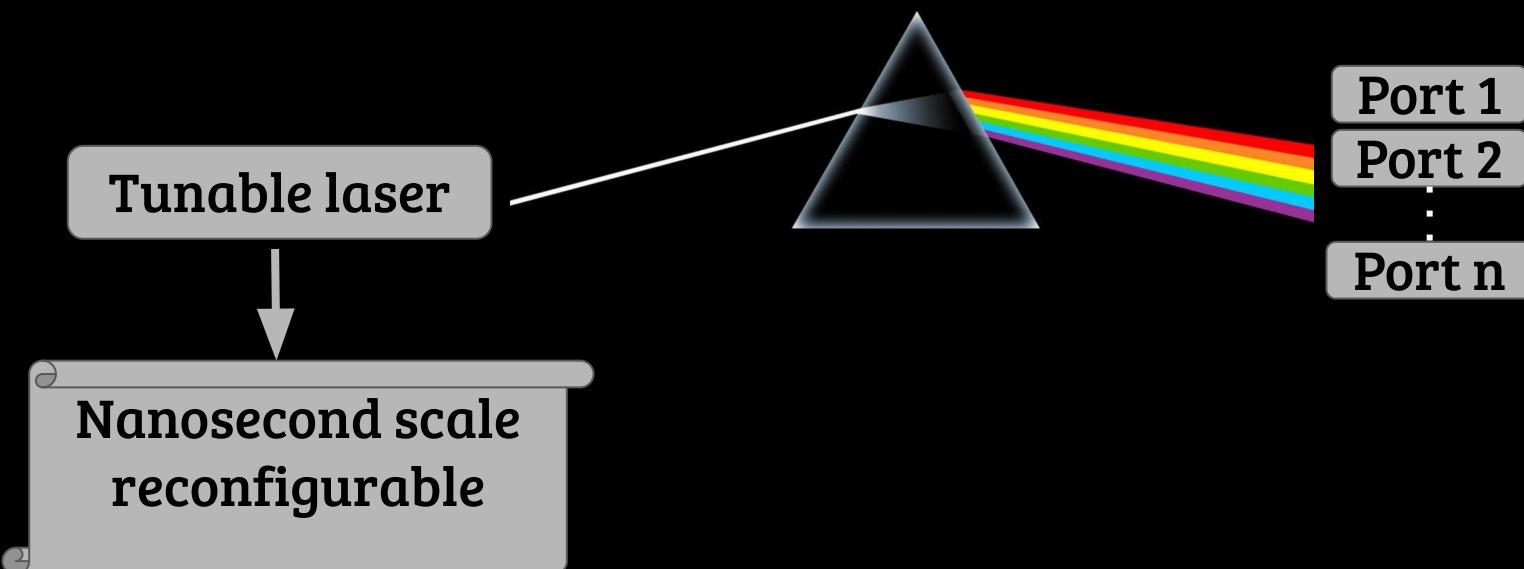
Tunable laser



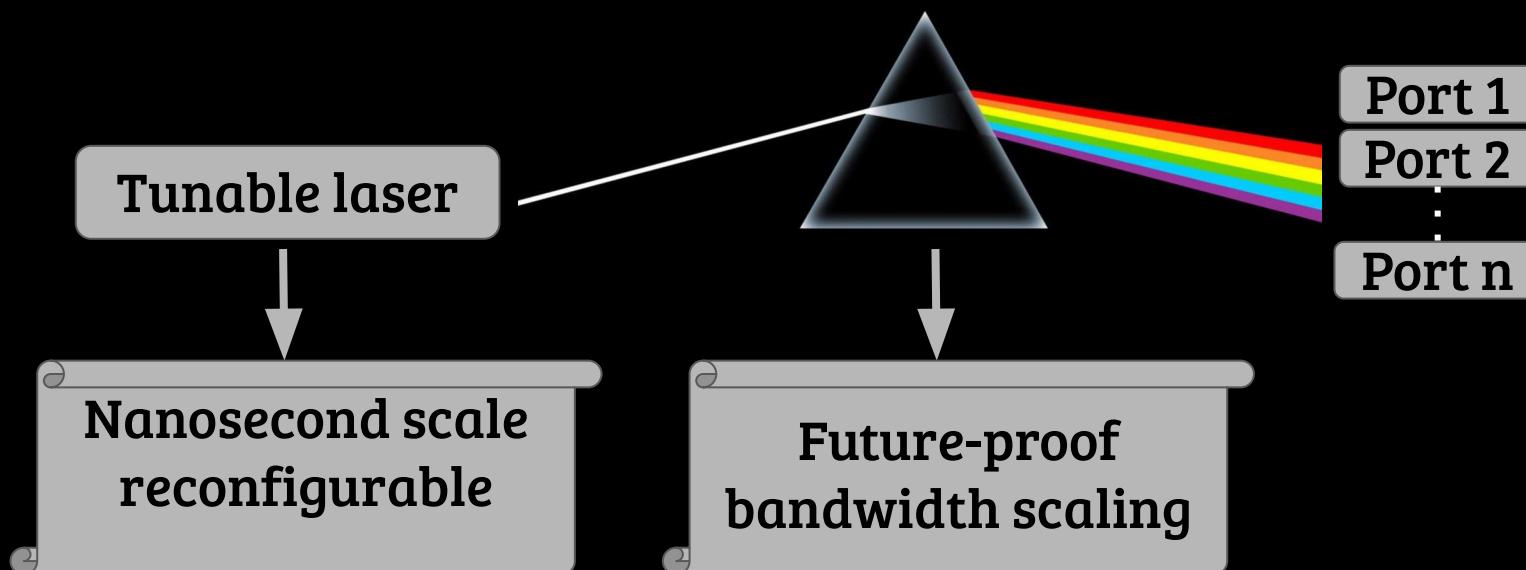
# Emerging Technologies - Optical Networks on the Rise



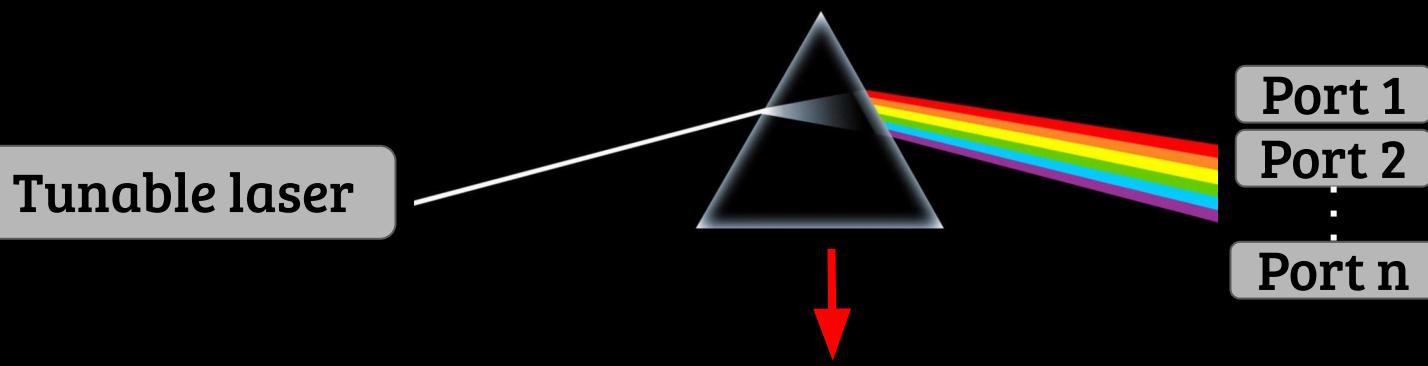
# Emerging Technologies - Optical Networks on the Rise

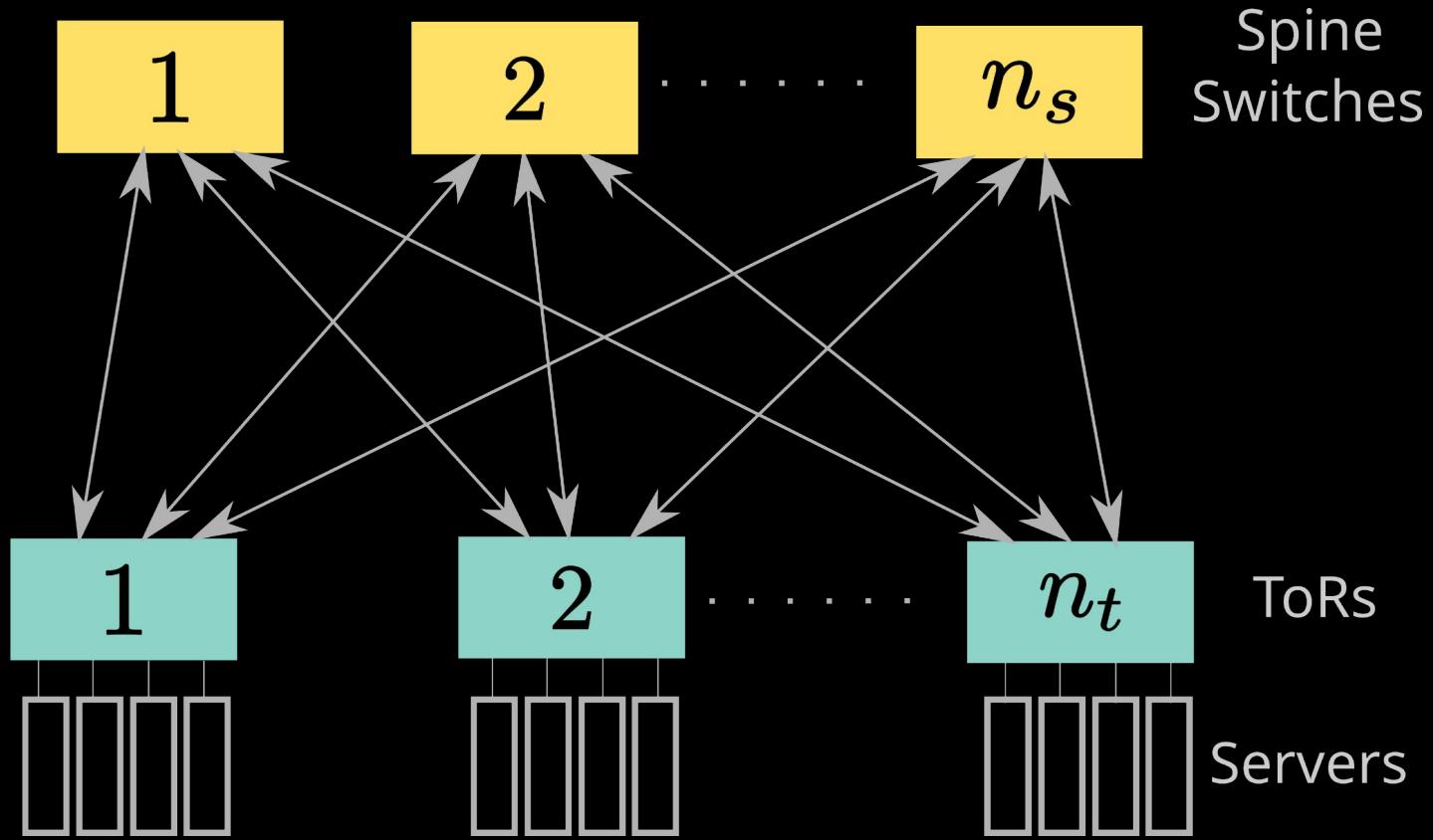


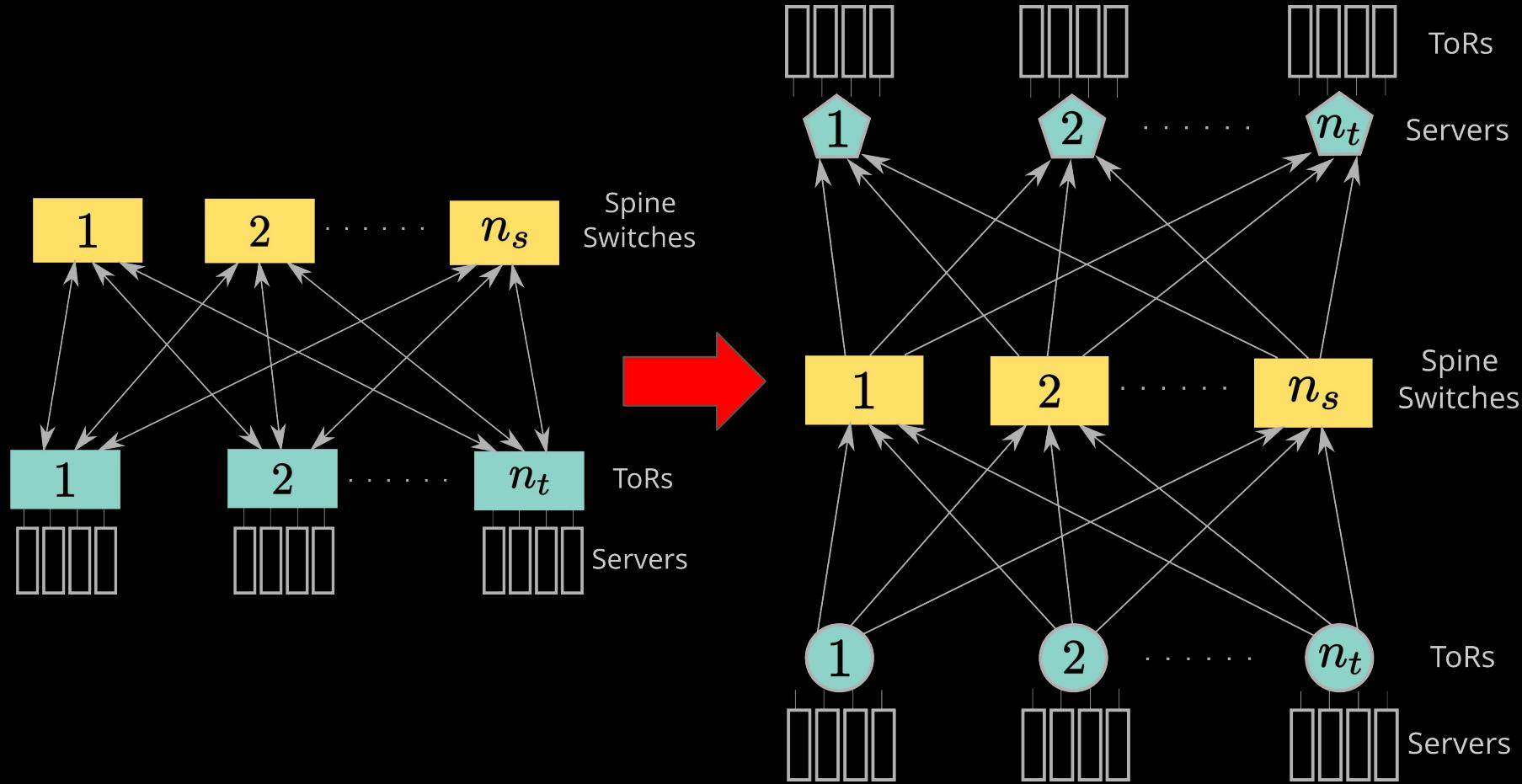
# Emerging Technologies - Optical Networks on the Rise

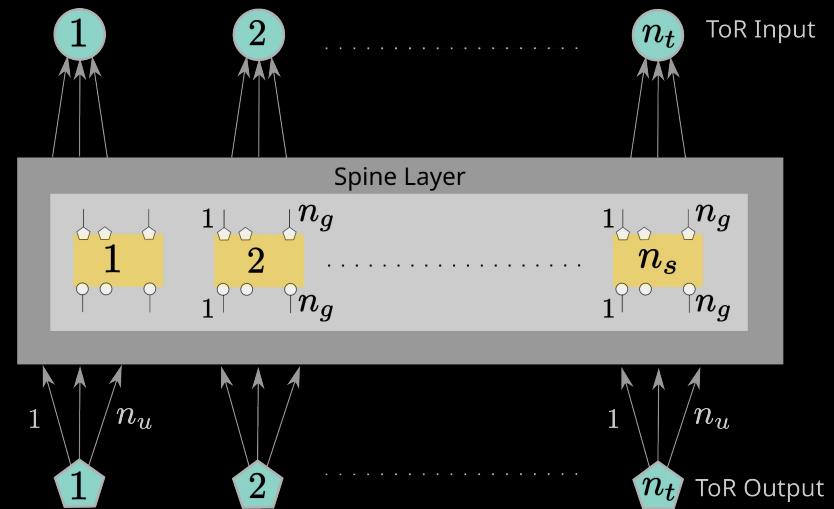
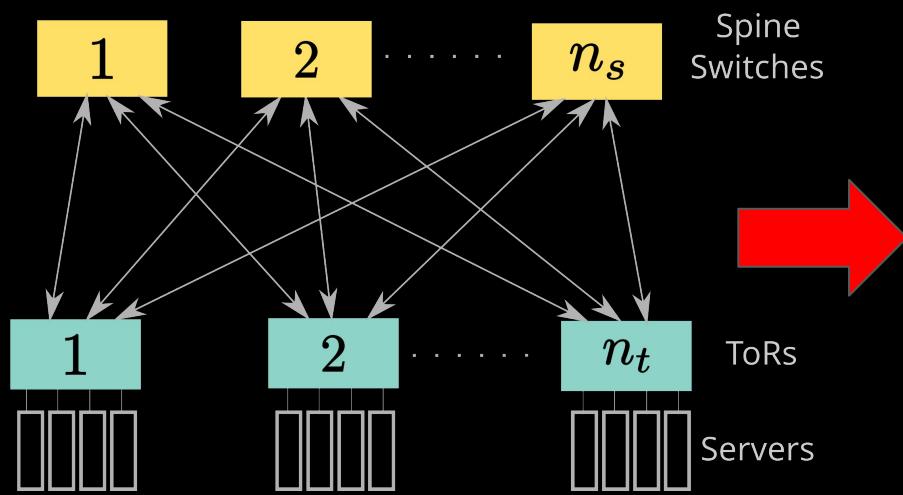


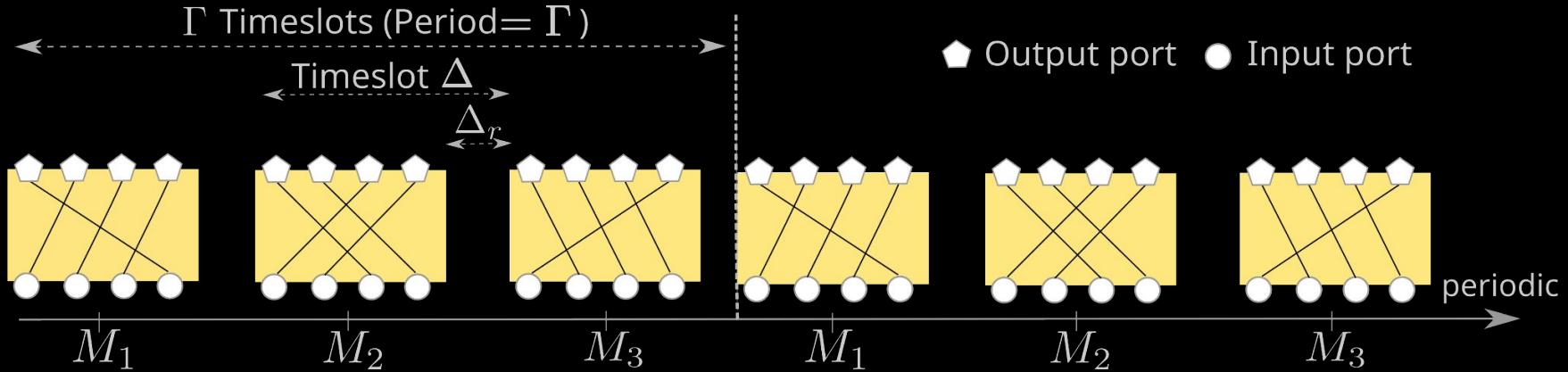
# Emerging Technologies - Optical Networks on the Rise

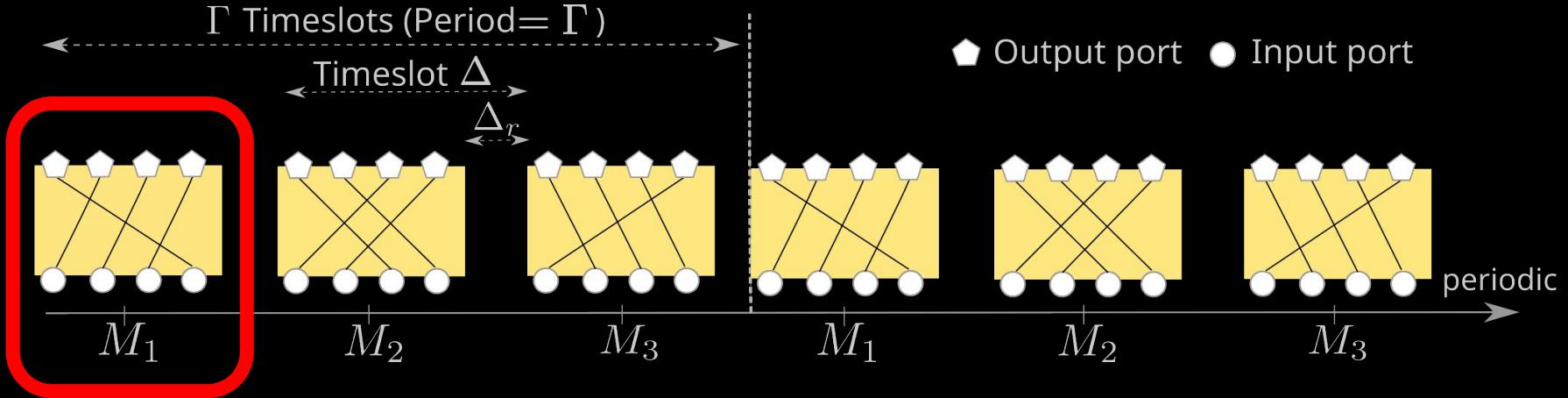


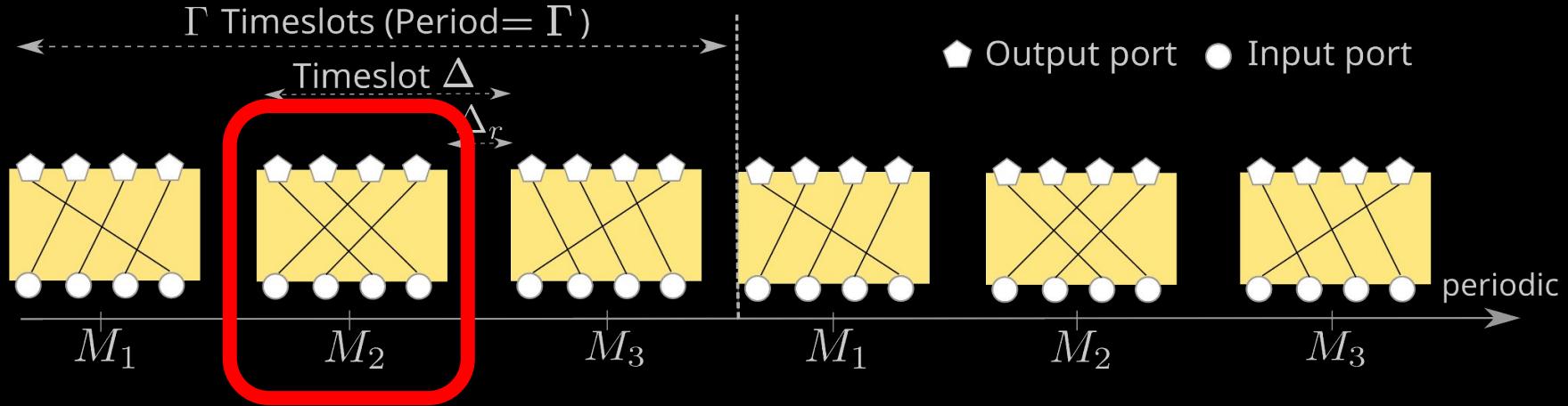


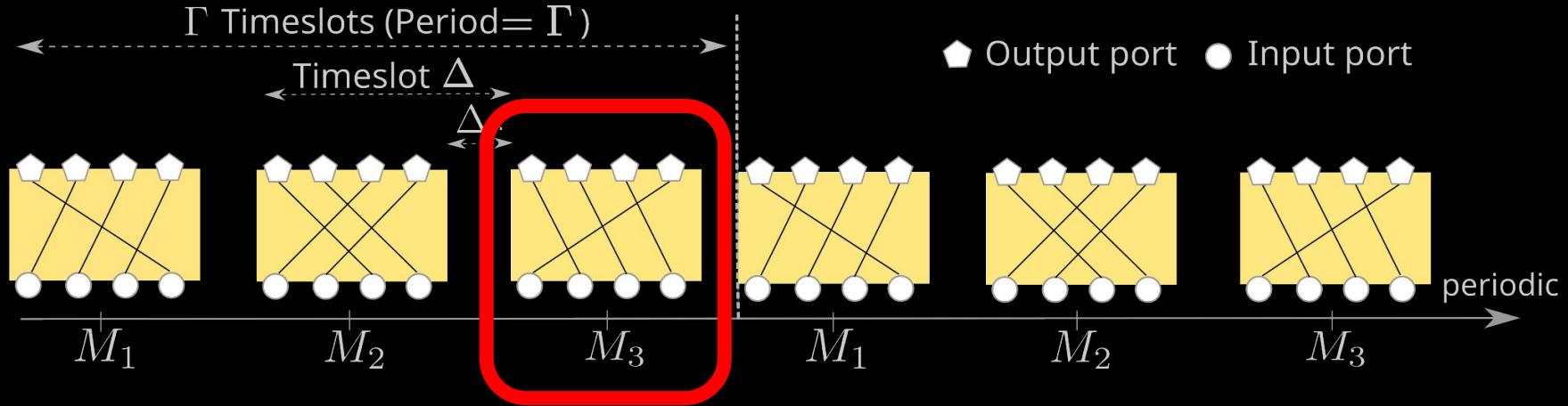


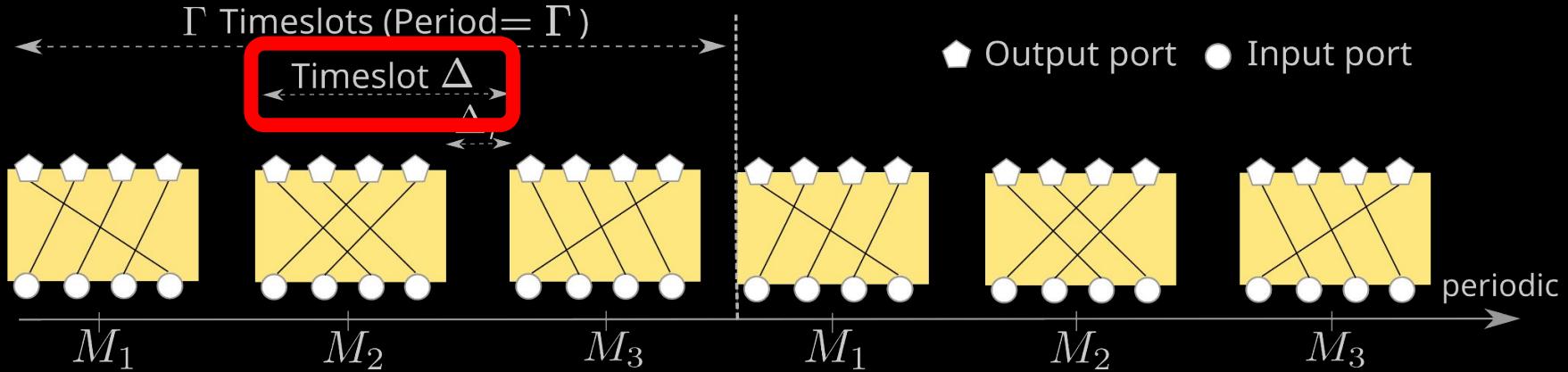


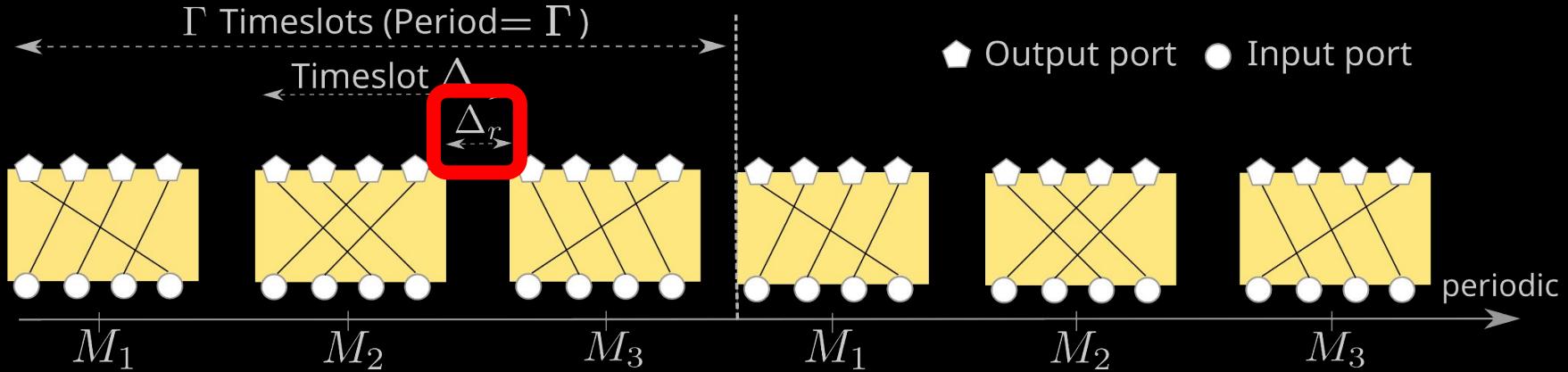


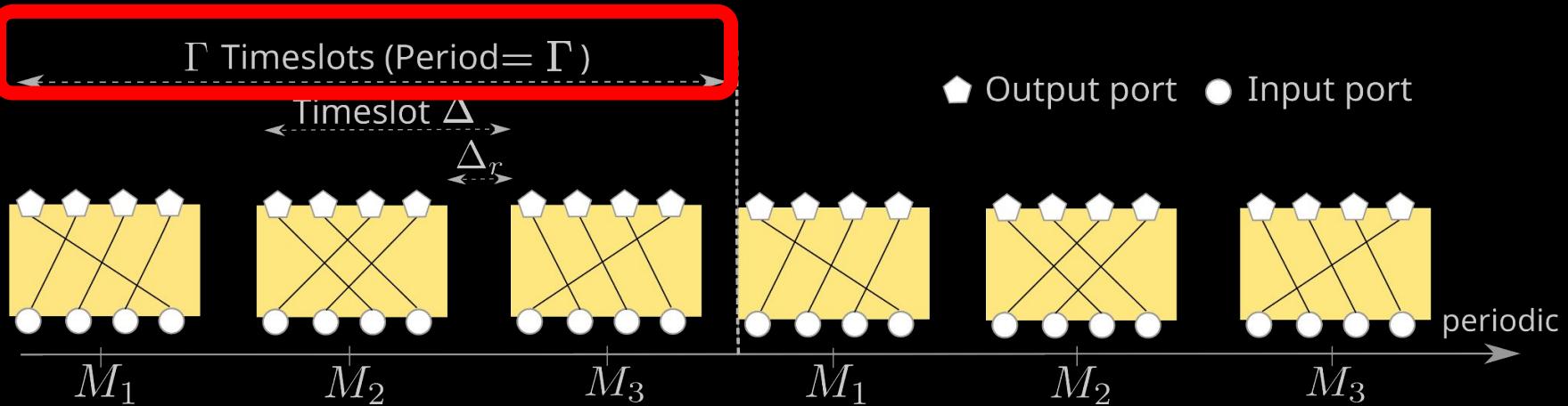


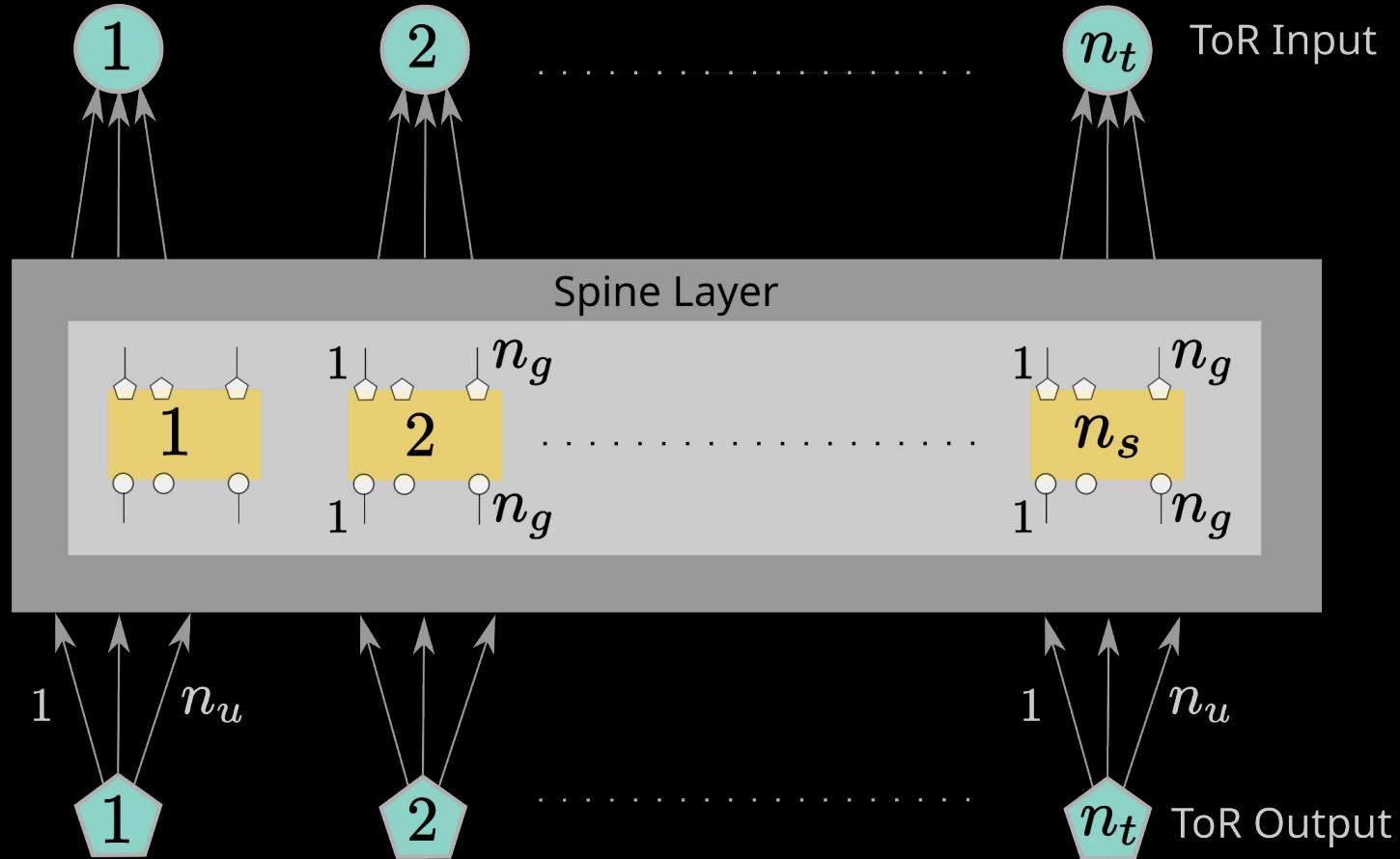




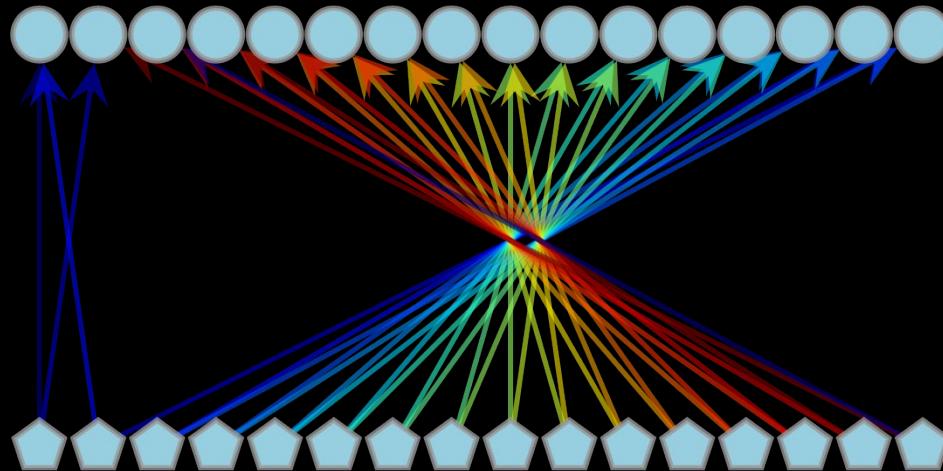




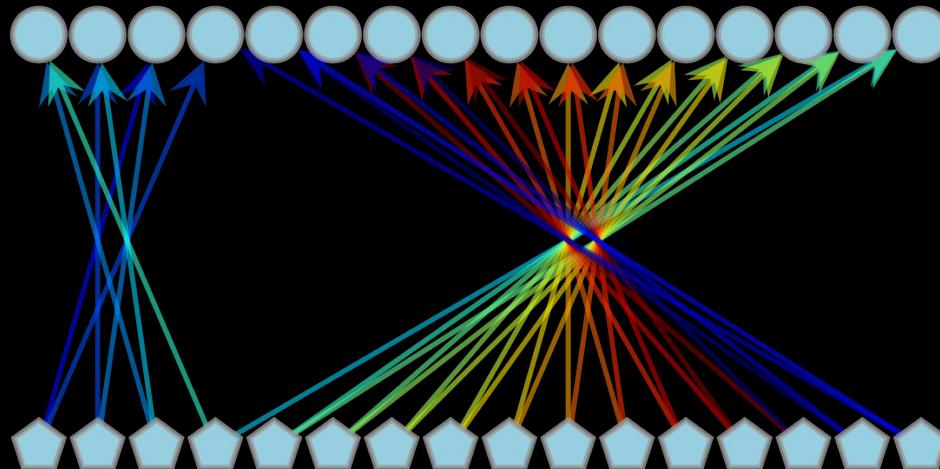




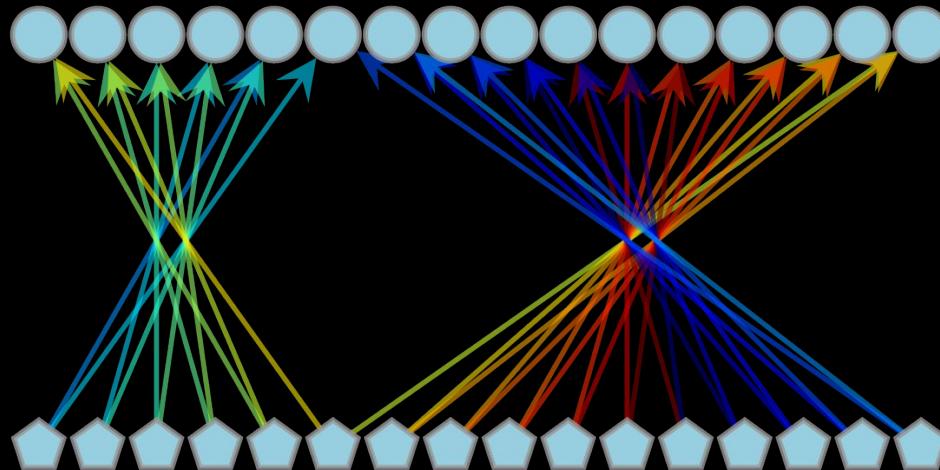
Timeslot 1



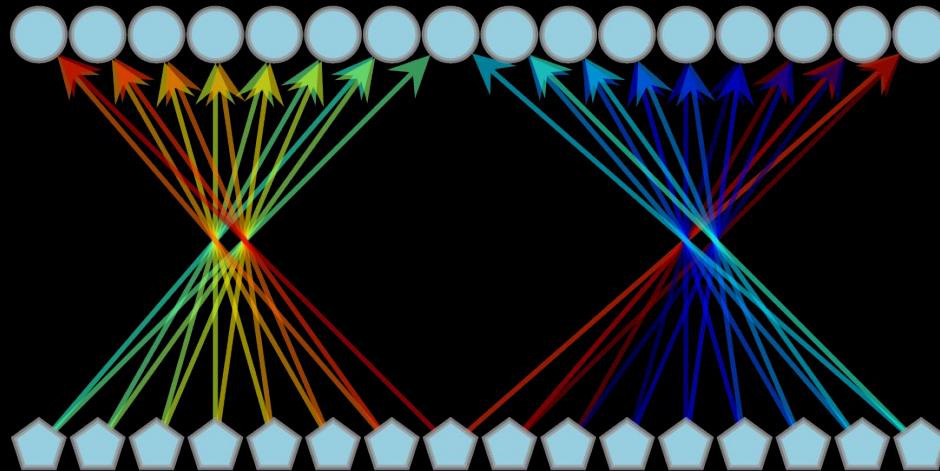
Timeslot 2



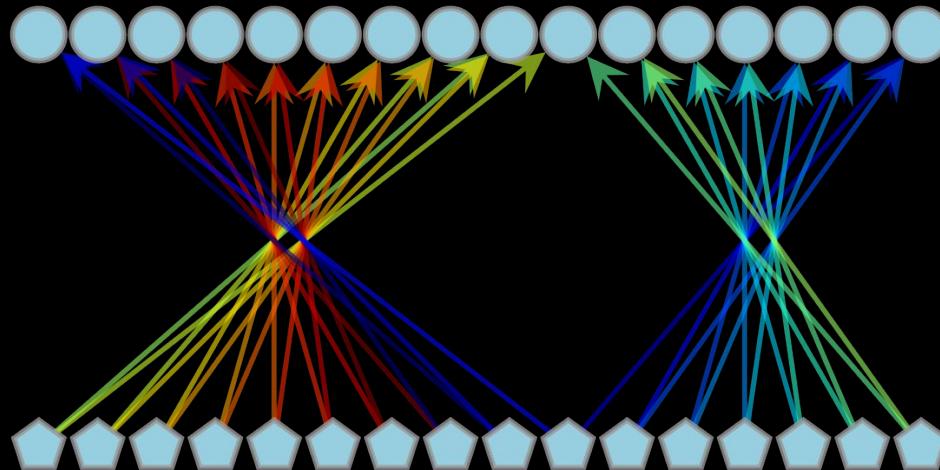
Timeslot 3



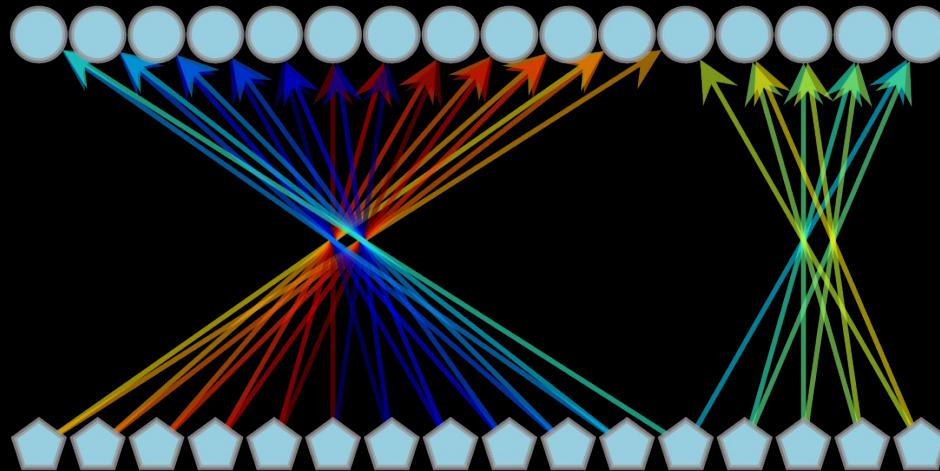
Timeslot 4



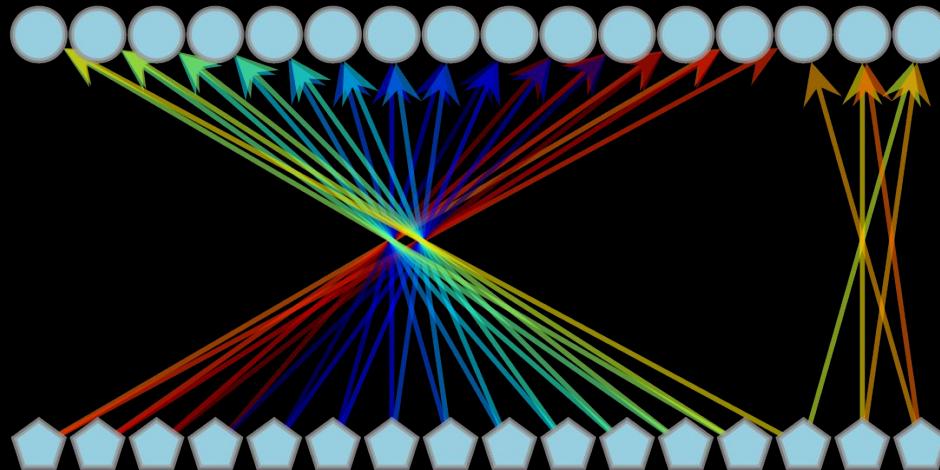
Timeslot 5



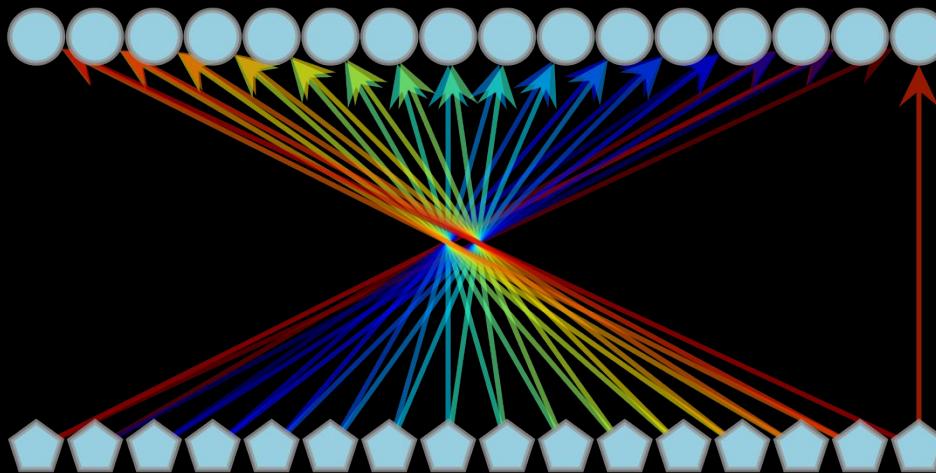
Timeslot 6



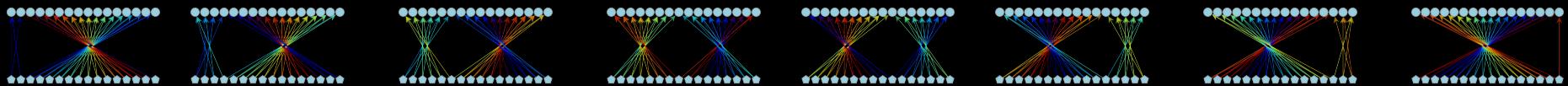
Timeslot 7



Timeslot 8



## Periodic Graph



# Optimal Topology

**Input:** Demand Matrix  $\mathcal{M}$

Number of nodes  $n$

Degree bound  $d'$  in each timeslot

**Output:** Periodic graph

**Maximize:** Throughput

# Throughput

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

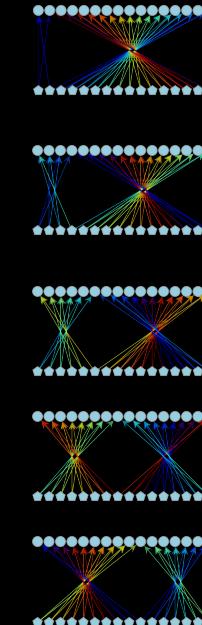
$$\mathcal{M} \quad X \quad \theta(\mathcal{M})$$

Demand Matrix

Throughput

*Highest scaling factor such  
that the scaled demand  $\theta$   
 $(\mathcal{M})$  is feasible in the  
periodic graph*

Periodic Graph



t

# Throughput of the Periodic Graph

**Input:** Periodic Graph  $\mathcal{G}$

Demand Matrix  $\mathcal{M}$

**Objective:** Maximize  $\theta(\mathcal{M})$

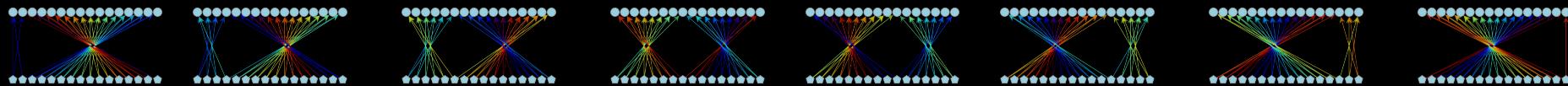
**Output:**  $\theta(\mathcal{M})$  and a feasible flow\*

*\*subject to conservation, demand and capacity constraints*

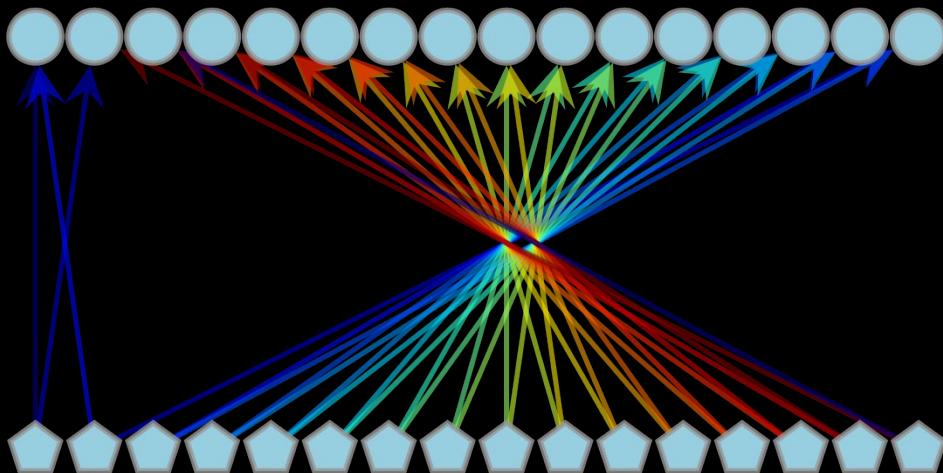
# Theorem 1: Periodic Graph $\longleftrightarrow$ Static Graph

- The periodic graph has the **same throughput** as that of a static graph it emulates - *Static Emulated Graph*

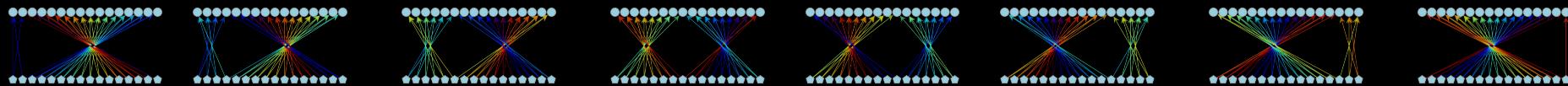
## Periodic Graph



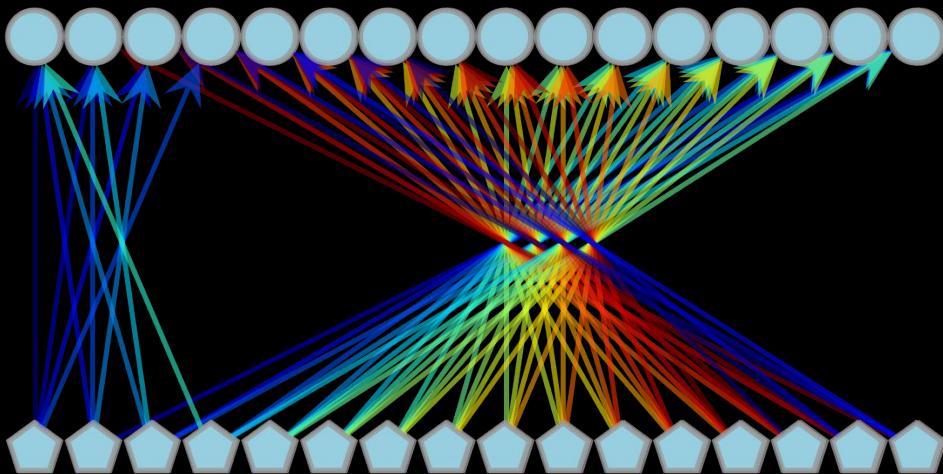
## Static Emulated Graph



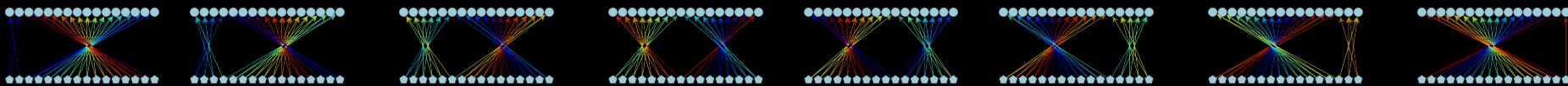
Periodic Graph



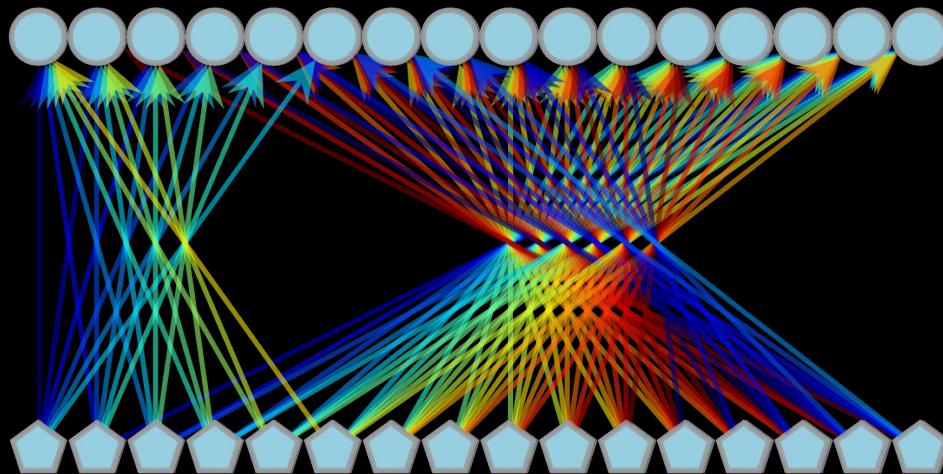
Static Emulated Graph



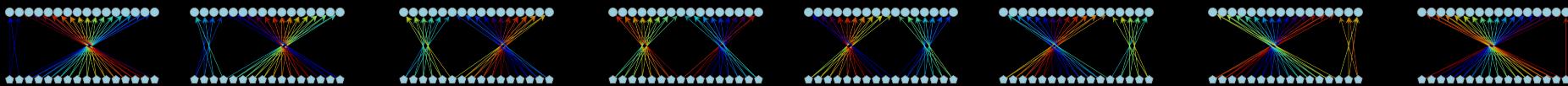
## Periodic Graph



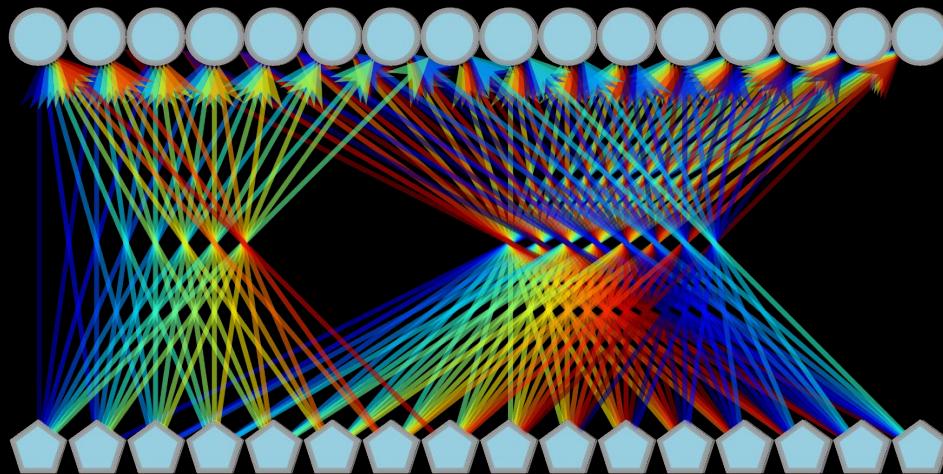
## Static Emulated Graph



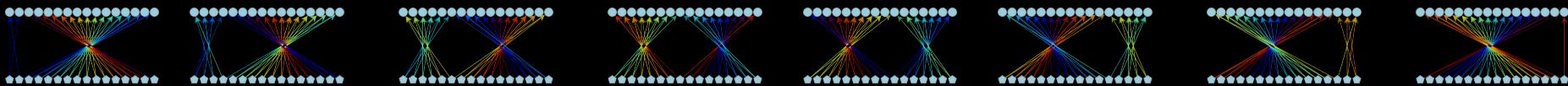
## Periodic Graph



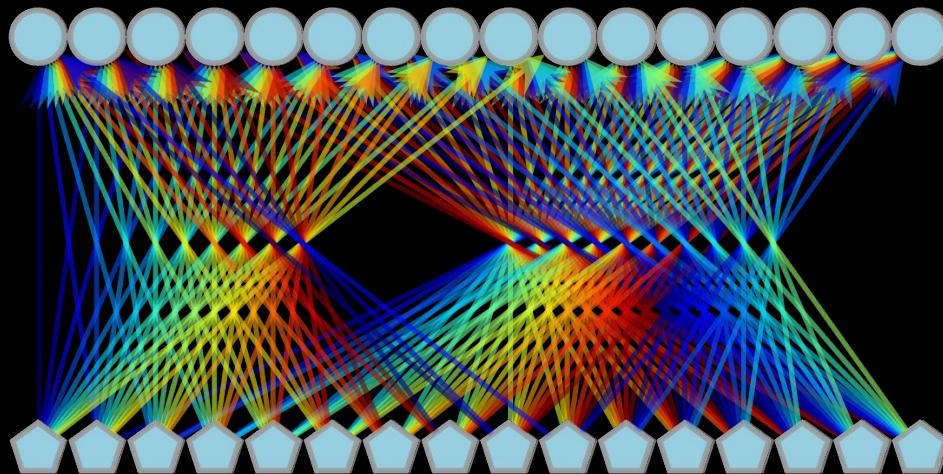
## Static Emulated Graph



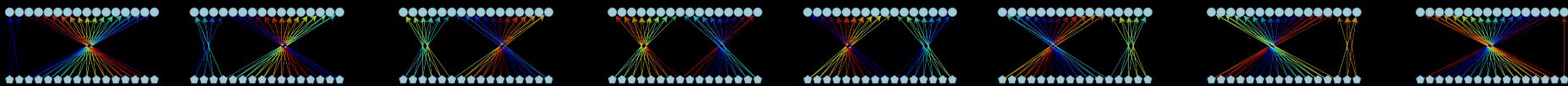
## Periodic Graph



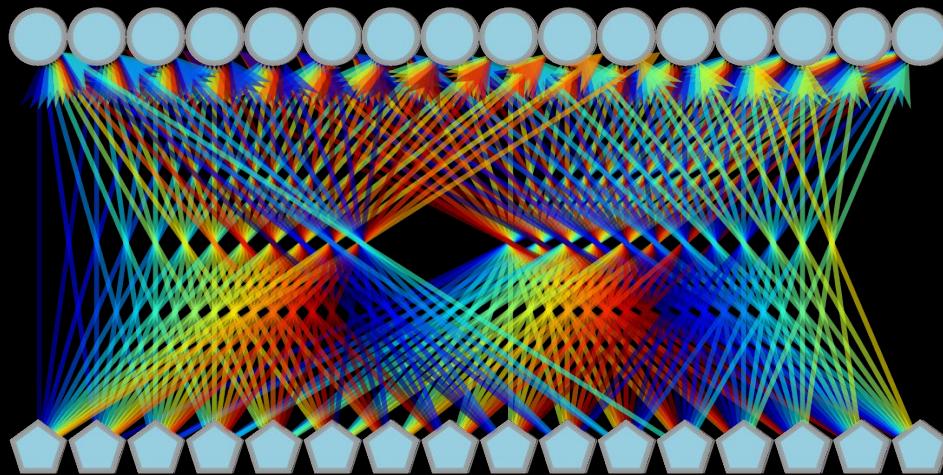
## Static Emulated Graph



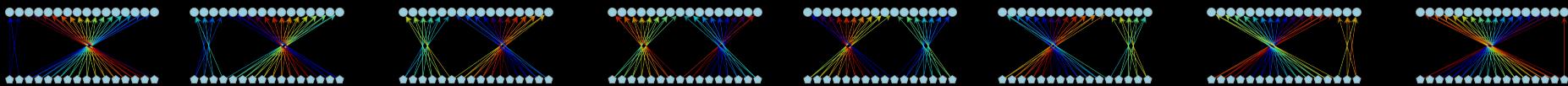
## Periodic Graph



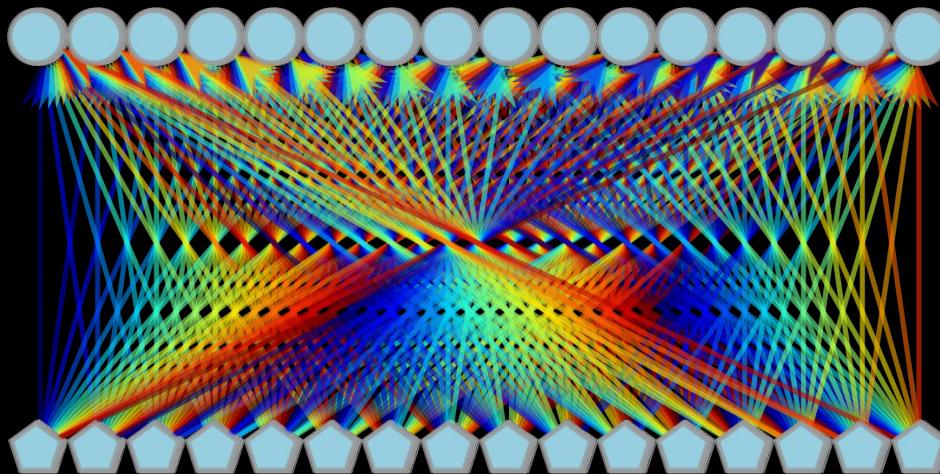
## Static Emulated Graph



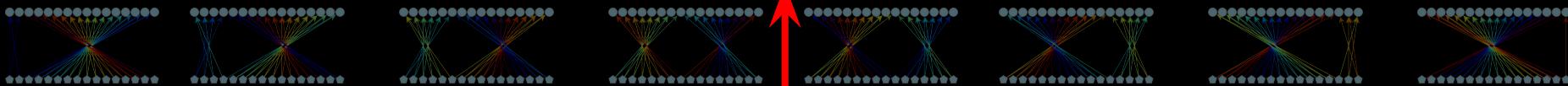
## Periodic Graph



## Static Emulated Graph

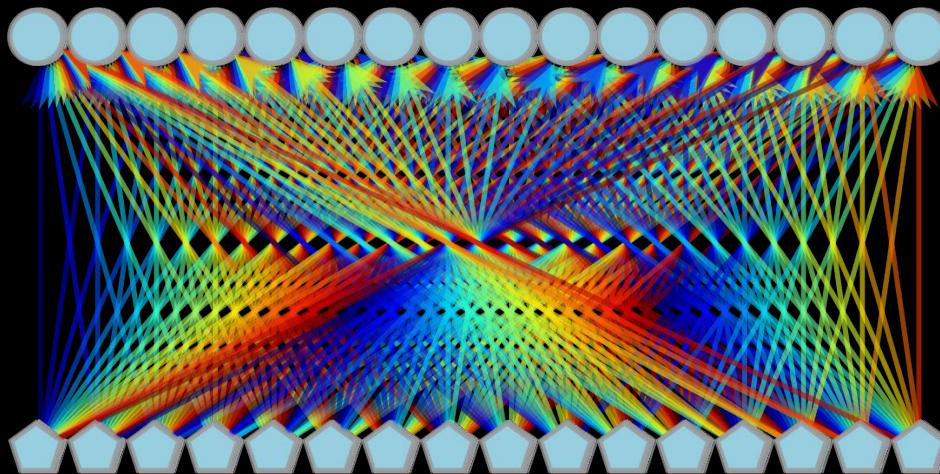


Periodic Graph



$\theta$

Static Emulated Graph



# Throughput of the Periodic Graph

**Input:** Periodic Graph  $\mathcal{G}$

Demand Matrix  $\mathcal{M}$

**Objective:** Maximize  $\theta(\mathcal{M})$

**Output:**  $\theta(\mathcal{M})$  and a feasible flow\*

*\*subject to conservation, demand and capacity constraints*

# Throughput of the Periodic Graph

**Input:** Periodic Graph  $\mathcal{G}$  Static Emulated Graph  $G$

Demand Matrix  $\mathcal{M}$

**Objective:** Maximize  $\theta(\mathcal{M})$

**Output:**  $\theta(\mathcal{M})$  and a feasible flow\*

*\*subject to conservation, demand and capacity constraints*



## Theorem 2: Throughput Upper Bound

- Throughput is a function of *Average Route Length*

$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{M \cdot \text{ARL}(\mathcal{M}, F)}$$

## Theorem 2: Throughput Upper Bound

- Throughput is a function of *Average Route Length*

$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{M \cdot \text{ARL}(\mathcal{M}, F)}$$

## Theorem 2: Throughput Upper Bound

- Throughput is a function of *Average Route Length*

$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{M \cdot \text{ARL}(\mathcal{M}, F)}$$

## Theorem 2: Throughput Upper Bound

- Throughput is a function of *Average Route Length*

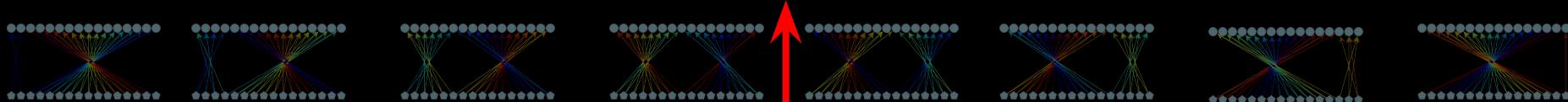
$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{M \text{ARL}(\mathcal{M}, F)}$$

## Theorem 2: Throughput Upper Bound

- Throughput is a function of *Average Route Length*

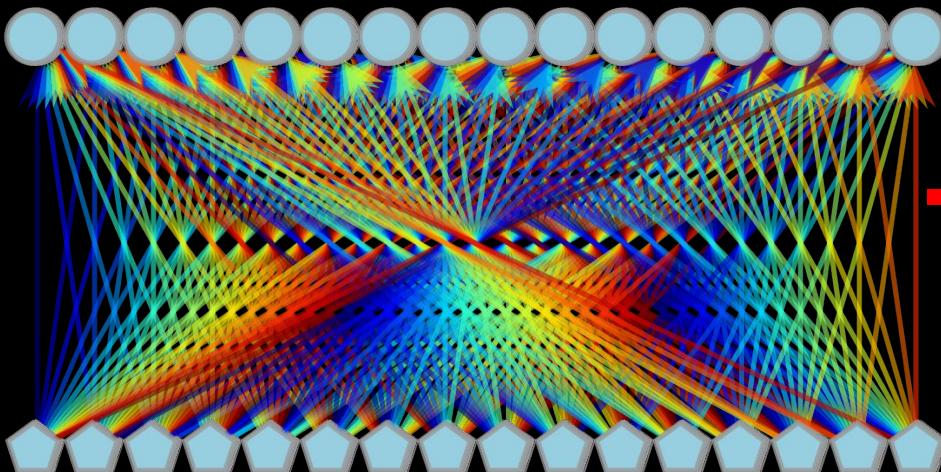
$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{M \cdot \text{ARL}(\mathcal{M}, F)}$$

## Periodic Graph



$\theta$

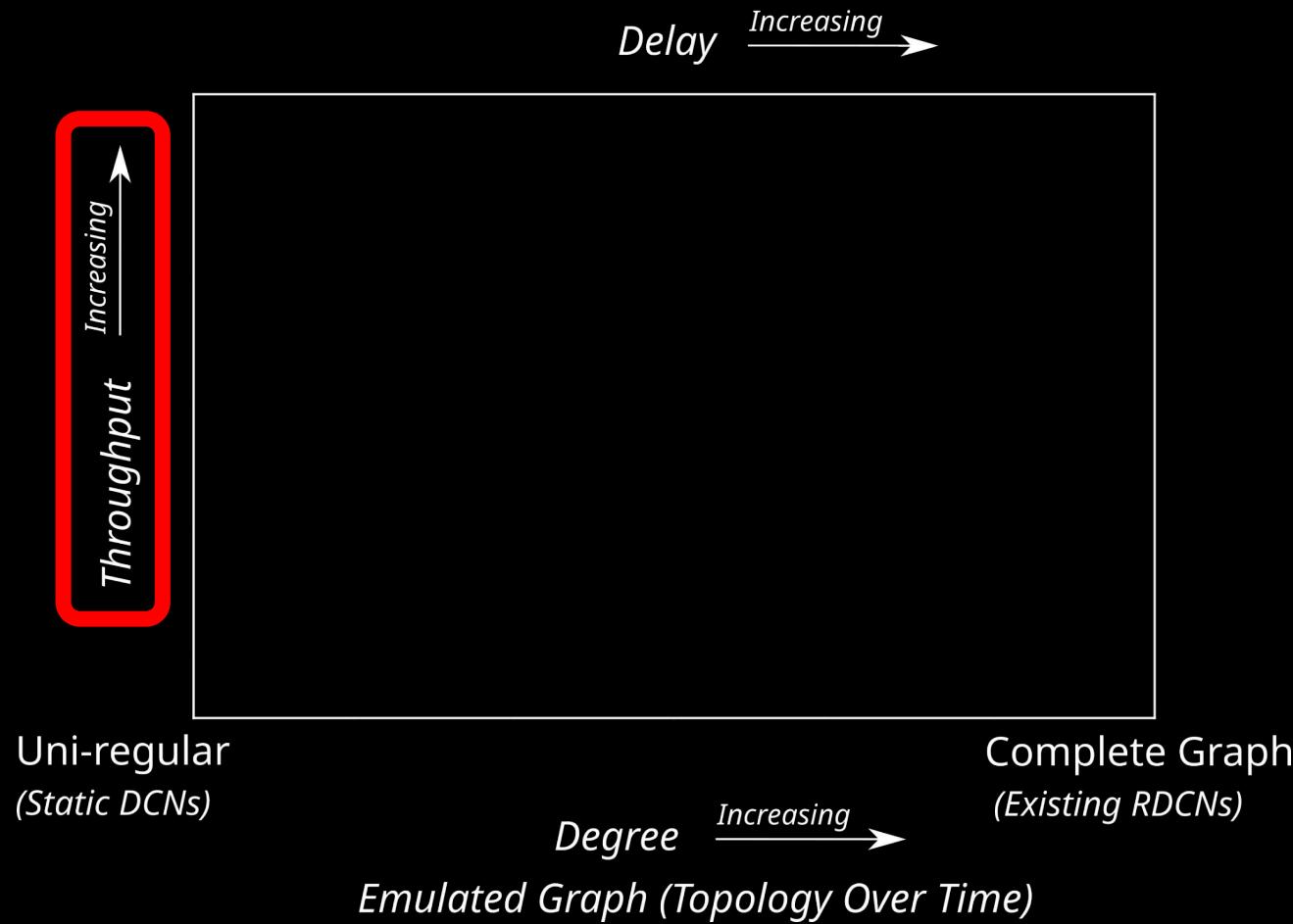
## Static Emulated Graph



$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{M \cdot \text{ARL}(\mathcal{M}, F)}$$



Capacity  
Demand x ARL



## Theorem 3: Delay

- Delay bound is a function of:
  - Degree  $\mathbf{d}$  of the emulated graph
  - Duration of the period  $\Gamma \cdot \Delta$
  - Throughput  $\theta$

$$\begin{aligned} L_{max} &\geq \text{ARD}(\mathcal{M}, \mathcal{F}) = \text{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta \\ &\geq \Omega\left(\frac{d \cdot \Delta}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})}\right) \end{aligned}$$

## Theorem 3: Delay

- Delay bound is a function of:
  - Degree  $d$  of the emulated graph
  - Duration of the period  $\Gamma \cdot \Delta$
  - Throughput  $\theta$

$$\boxed{L_{max}} \geq \text{ARD}(\mathcal{M}, \mathcal{F}) = \text{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta$$
$$\geq \Omega \left( \frac{d \cdot \Delta}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})} \right)$$

## Theorem 3: Delay

- Delay bound is a function of:
  - Degree  $d$  of the emulated graph
  - Duration of the period  $\Gamma \cdot \Delta$
  - Throughput  $\theta$

$$\begin{aligned} L_{max} &\geq \text{ARD}(\mathcal{M}, \mathcal{F}) = \text{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta \\ &\geq \Omega\left(\frac{d \cdot \Delta}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})}\right) \end{aligned}$$

## Theorem 3: Delay

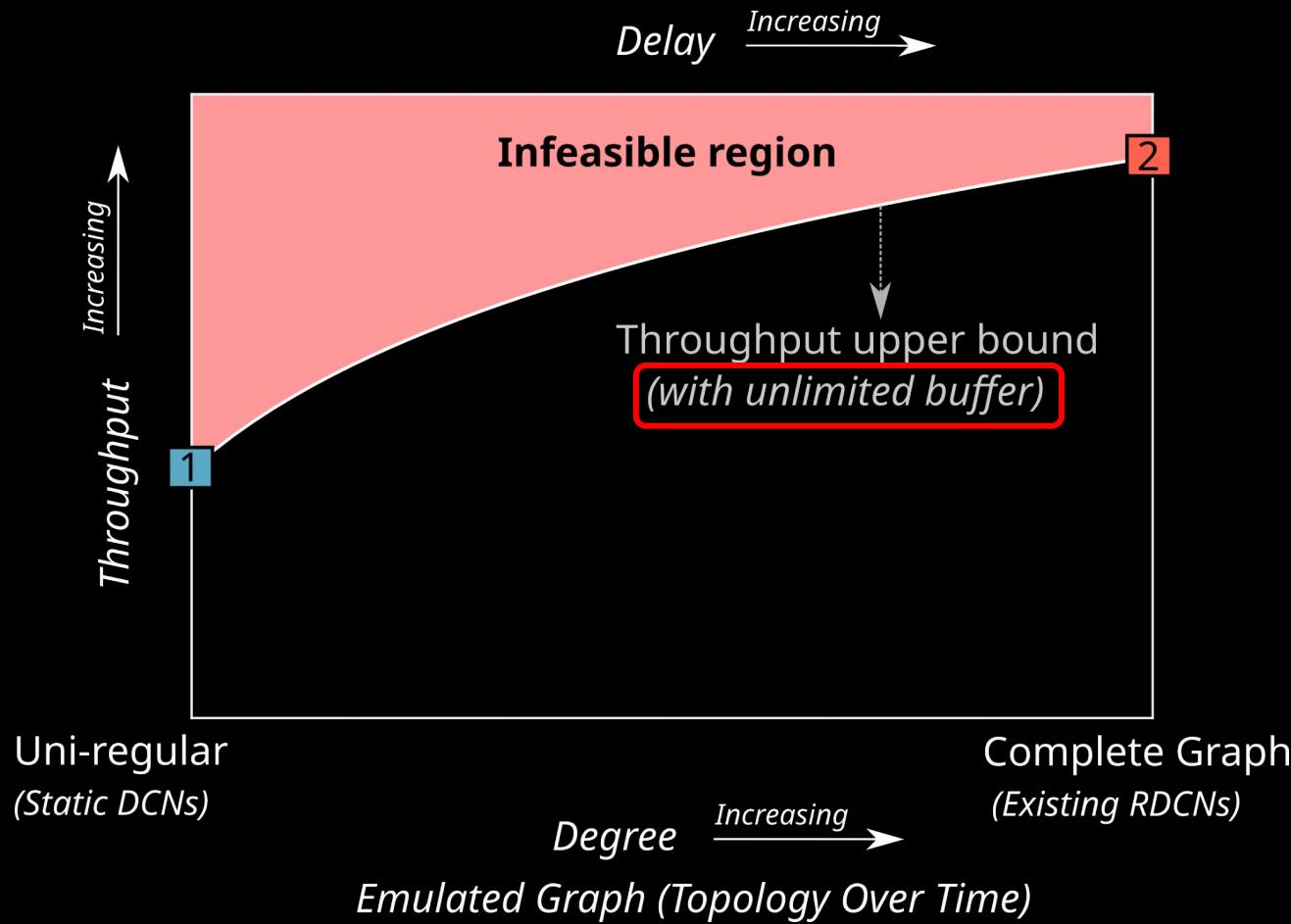
- Delay bound is a function of:
  - Degree  $d$  of the emulated graph
  - Duration of the period  $\Gamma \cdot \Delta$
  - Throughput  $\theta$

$$\begin{aligned} L_{max} &\geq \text{ARD}(\mathcal{M}, \mathcal{F}) = \text{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta \\ &\geq \Omega\left(\frac{d \Delta}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})}\right) \end{aligned}$$

## Theorem 3: Delay

- Delay bound is a function of:
  - Degree  $d$  of the emulated graph
  - Duration of the period  $\Gamma \cdot \Delta$
  - Throughput  $\theta$

$$\begin{aligned} L_{max} &\geq \text{ARD}(\mathcal{M}, \mathcal{F}) = \text{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta \\ &\geq \Omega\left(\frac{d \cdot \Delta}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})}\right) \end{aligned}$$



## Theorem 4: Buffer Requirements

- The required buffer is at least the *throughput delay product*

$$\hat{B} \geq (\theta(\mathcal{M}, \mathcal{F}) \cdot M) \cdot \text{ARD}(\mathcal{M}, \mathcal{F})$$

## Theorem 4: Buffer Requirements

- The required buffer is at least the *throughput delay product*

$$\hat{B} \geq (\theta(\mathcal{M}, \mathcal{F}) \cdot M) \cdot \text{ARD}(\mathcal{M}, \mathcal{F})$$

## Theorem 4: Buffer Requirements

- The required buffer is at least the *throughput delay product*

$$\hat{B} \geq (\theta(\mathcal{M}, \mathcal{F}) \cdot M) \cdot \text{ARD}(\mathcal{M}, \mathcal{F})$$

## Theorem 4: Buffer Requirements

- The required buffer is at least the *throughput delay product*

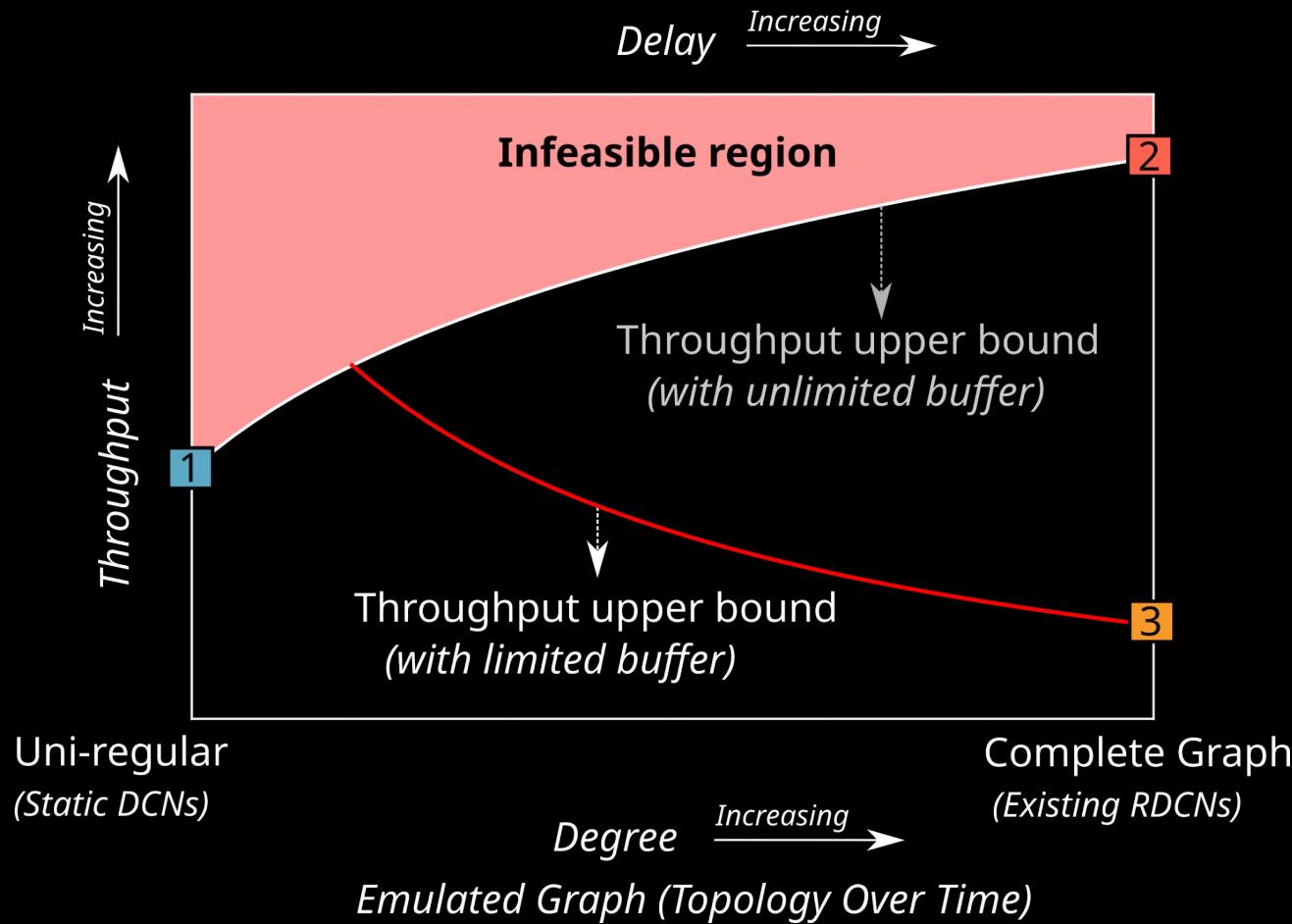
$$\hat{B} \geq (\theta(\mathcal{M}, \mathcal{F}) \cdot M) \cdot \text{ARD}(\mathcal{M}, \mathcal{F})$$

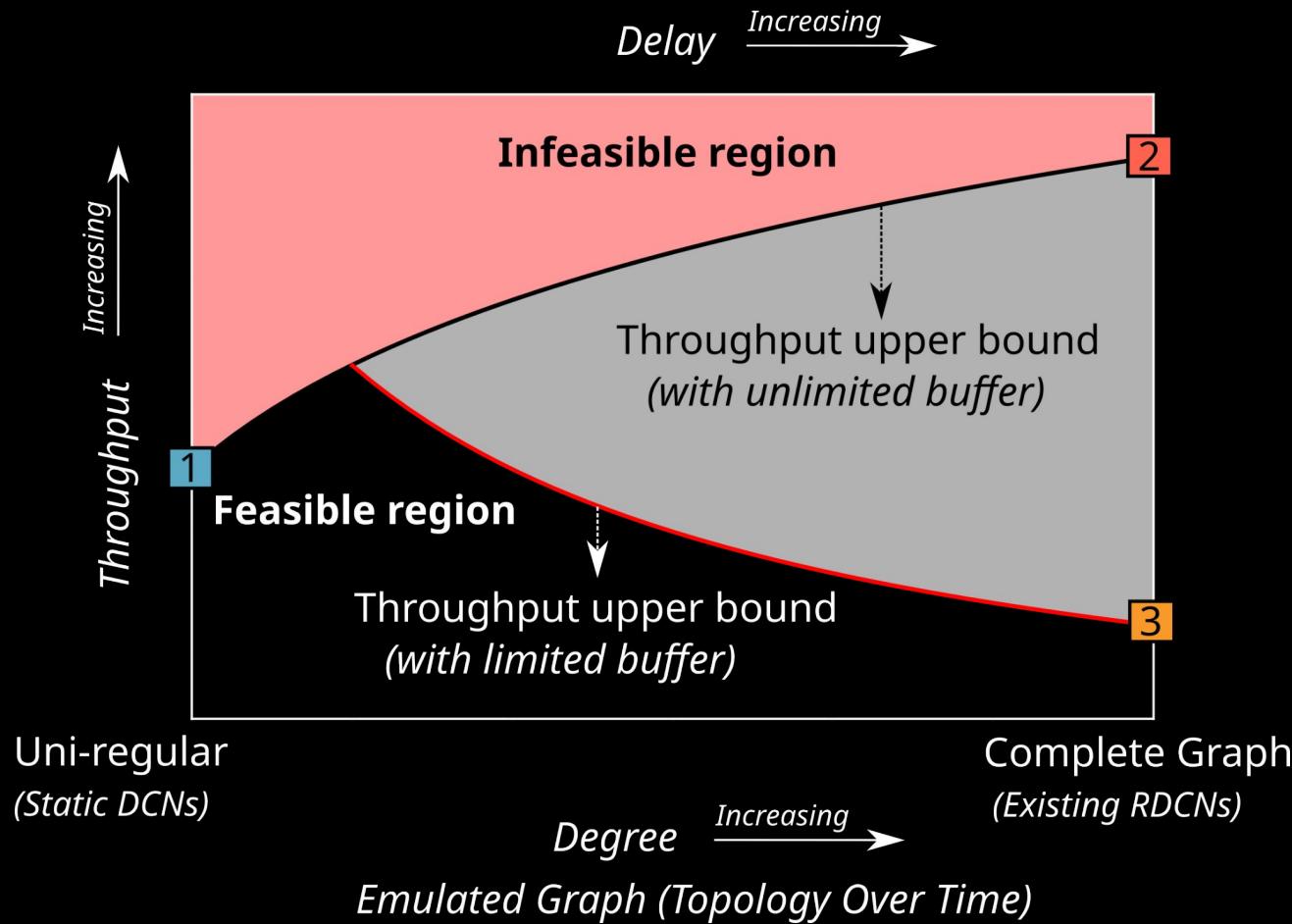
## Theorem 4: Buffer Requirements

- The required buffer is at least the *throughput delay product*

$$\hat{B} \geq (\theta(\mathcal{M}, \mathcal{F}) \cdot M) \cdot \text{ARD}(\mathcal{M}, \mathcal{F})$$

Buffer > Bandwidth x Delay





# Goals

- **Maximize Throughput**
- **Minimize Latency**
- **Minimize Buffer Requirements**

# Optimal Oblivious Topology with Buffer Constraints

**Input:** Periodic Graph  $\mathcal{G}$

Hose model demand matrix set

Available buffer size  $B$  at each node

**Output:** Degree  $d$  of the emulated graph  $\longrightarrow$  Periodic graph

**Objective:** Maximize the worst-case throughput

# Optimal Oblivious Topology with Buffer Constraints

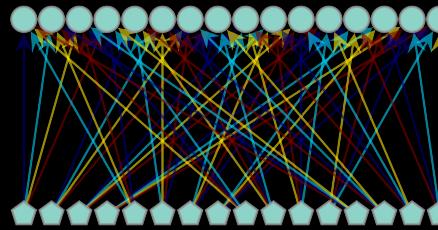
**Output:**  $d = B / (c \cdot \Delta)$

# Optimal Oblivious Topology with Buffer Constraints

Output:  $d = B/(c \cdot \Delta)$



$d$ -regular directed deBruijn graph



# Optimal Oblivious Topology with Buffer Constraints

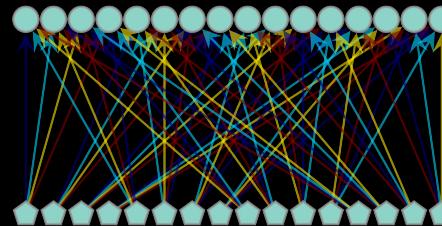
Output:  $d = B/(c \cdot \Delta)$



$d$ -regular directed deBruijn graph



Decomposition to  $d$  matchings



# Optimal Oblivious Topology with Buffer Constraints

Output:  $d = B/(c \cdot \Delta)$



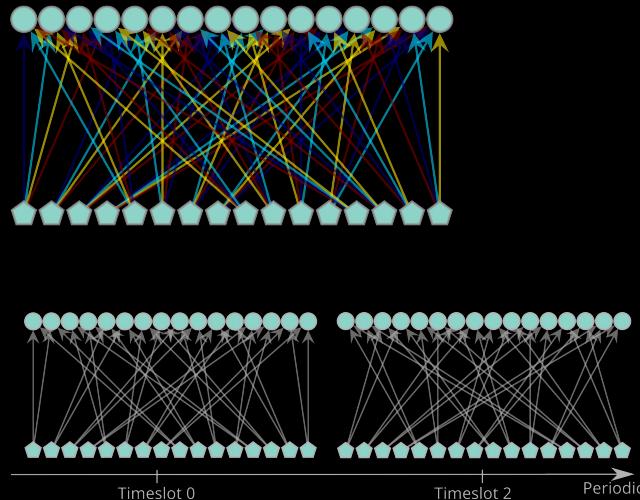
d-regular directed deBruijn graph

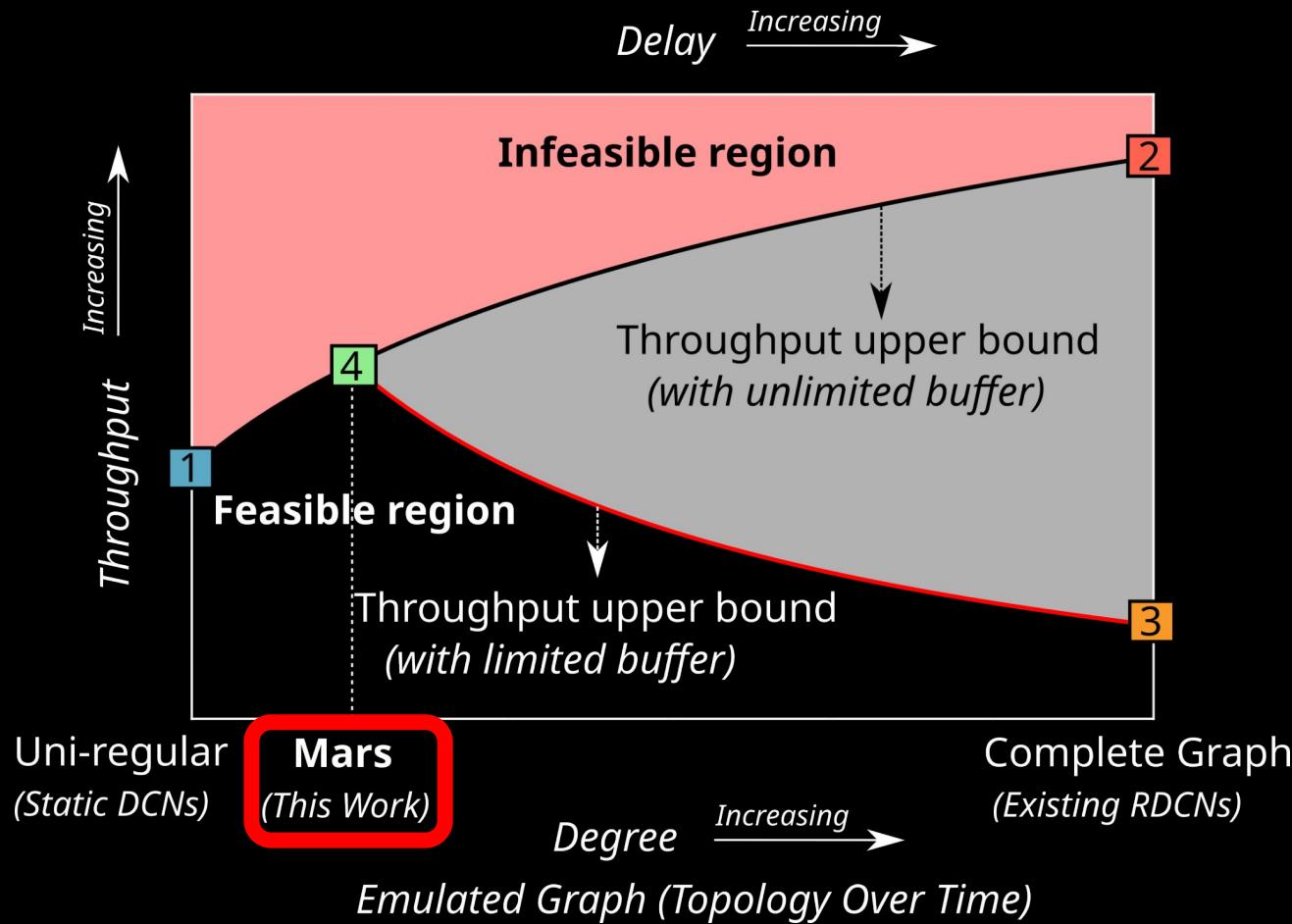


Decomposition to  $d$  matchings



Periodic graph





# Optimal Oblivious Topology Implications & Future Outlook

**Output:**  $d = B / (c \cdot \Delta)$

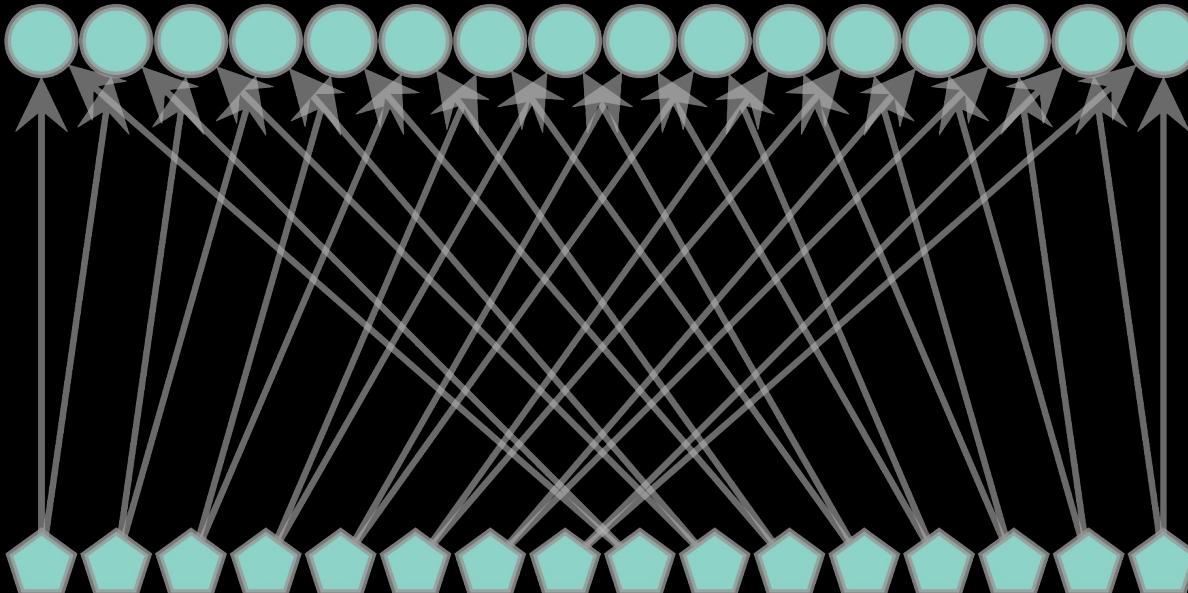
If the reconfiguration technology ( $\Delta$ ) remains same:

- Buffer sizes ( $B$ ) must keep up with the increase in capacity ( $c$ )

If Buffer sizes ( $B$ ) do not keep up:

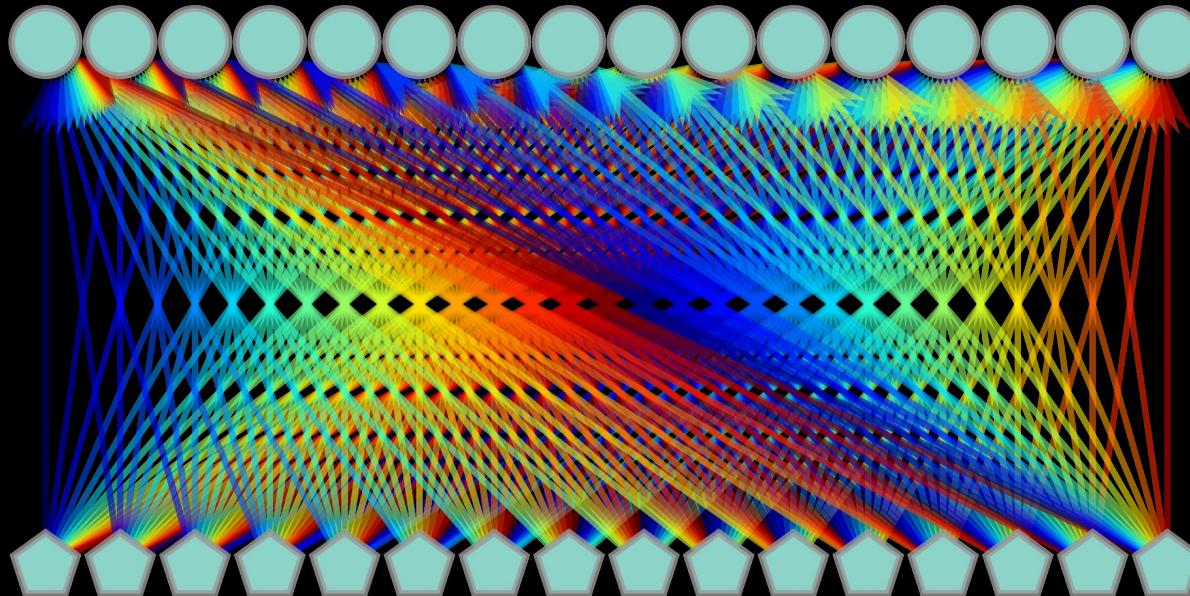
- Increase in capacity ( $c$ ) must be accompanied by decrease in reconfiguration times ( $\Delta$ )
- If not, reducing the degree ( $d$ ) of the emulated graph is inevitable to optimize throughput → eventually reaching the case of static topologies.

## Static DCNs (*uni-regular*)



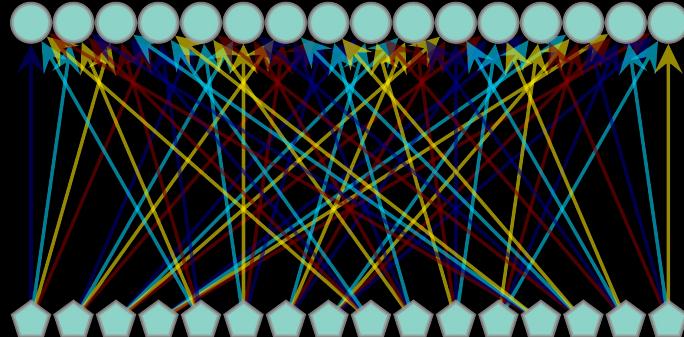
**Low throughput but low delay and buffer requirements**

## Existing RDCN designs (*Emulating a complete graph*)

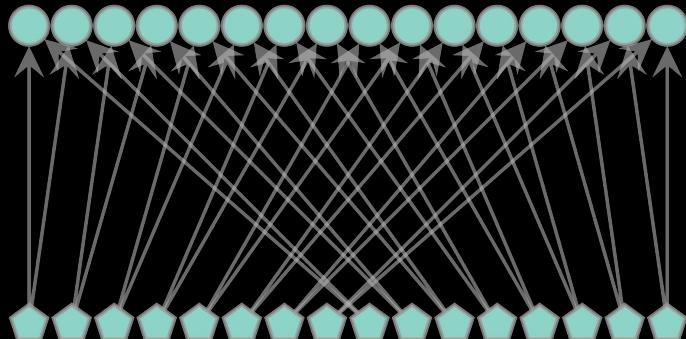


High throughput **but high delay and buffer requirements**

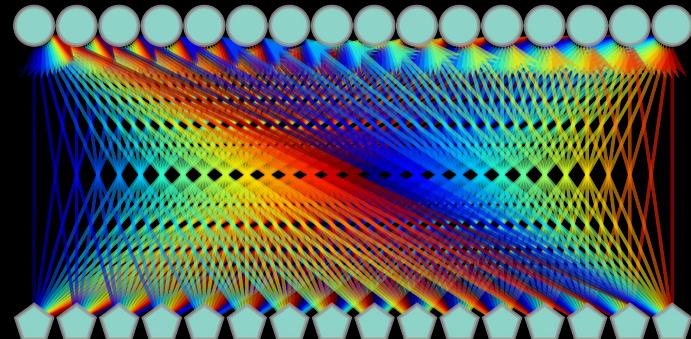
# MARS



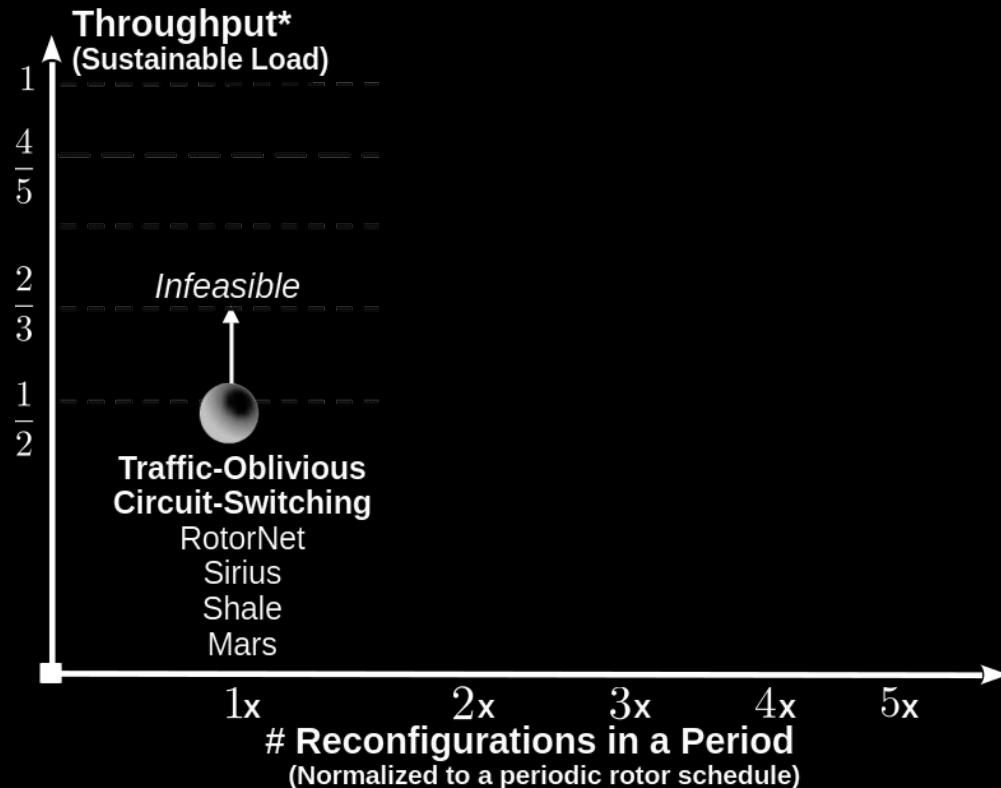
Near-optimal throughput within the available buffer



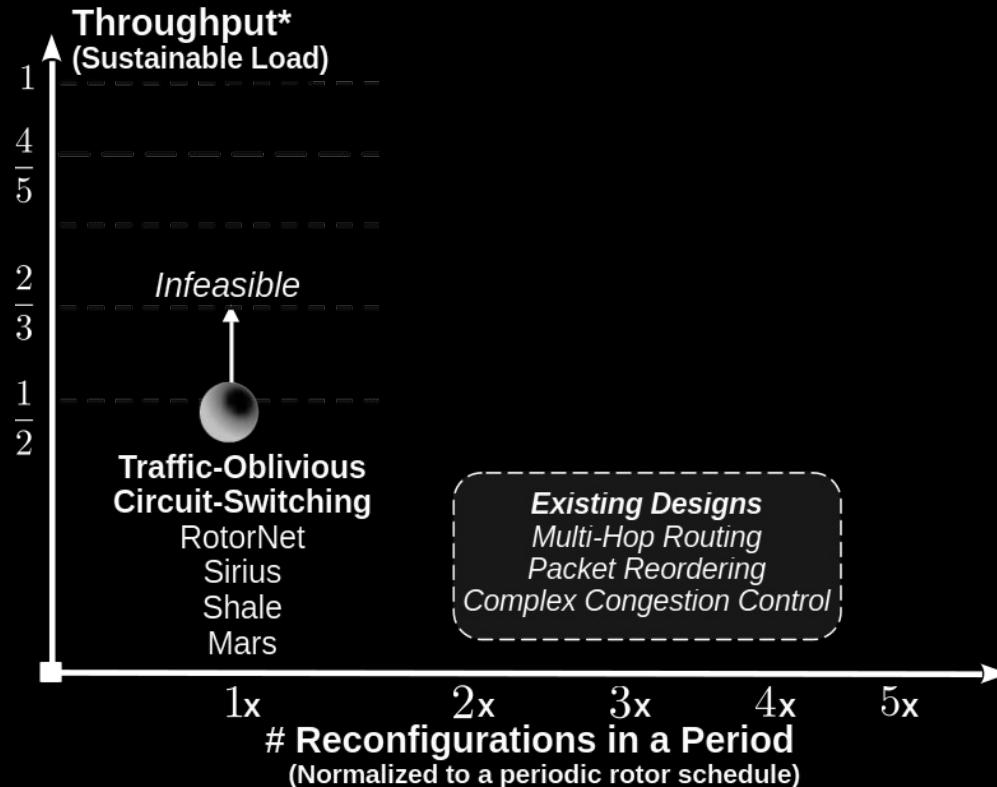
Static DCN: Low Throughput



Existing RDCN: High Delay and buffer

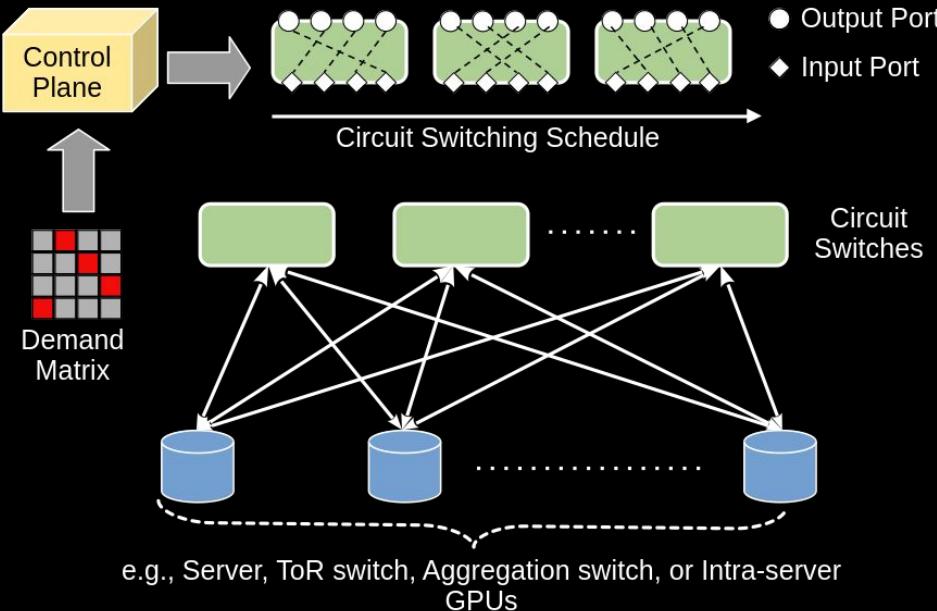


\*worst-case throughput

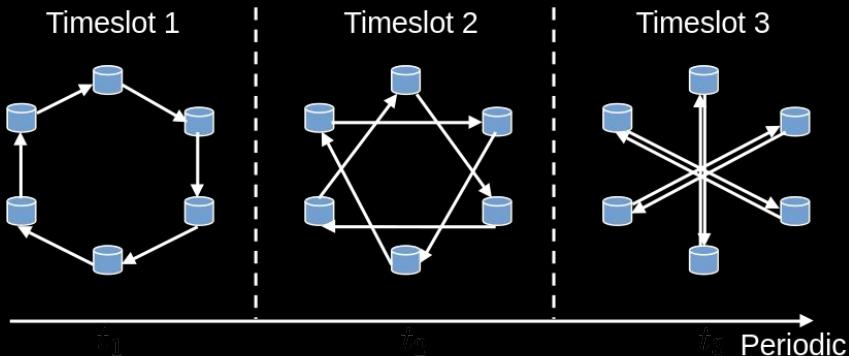


\**worst-case throughput*

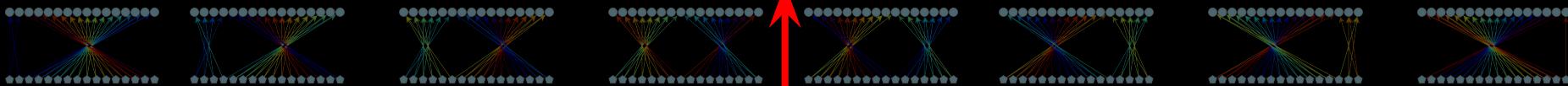
# A Traffic-Aware Approach



# Dynamic Reconfigurable Topology

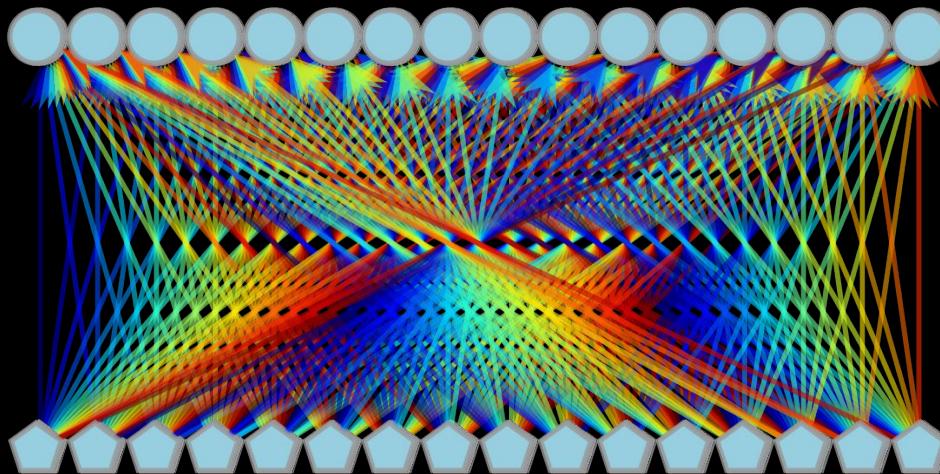


Periodic Graph

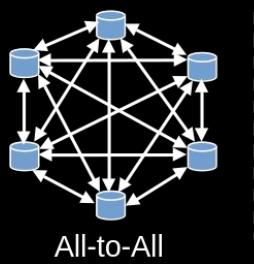


$\theta$

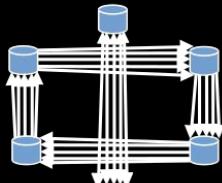
Static Emulated Graph



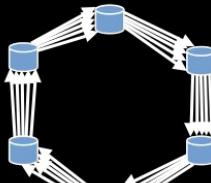
# Emulated Topology



All-to-All

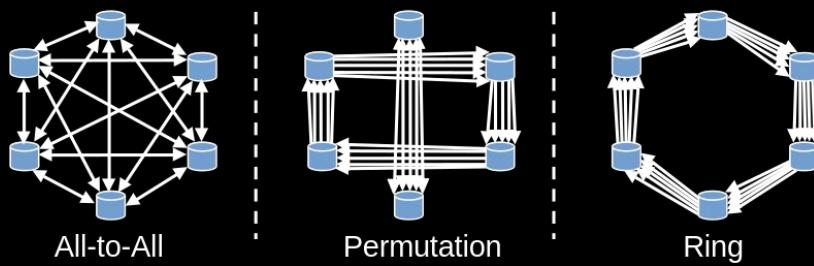


Permutation



Ring

# Emulated Topology



**Which topology is optimal for throughput?**

# Revisiting Optimal Topology Problem

**Input:** Demand Matrix  $\mathcal{M}$

Number of nodes  $n$

Degree bound  $d'$  in each timeslot

**Output:** Periodic graph

**Maximize:** Throughput

# Revisiting Optimal Topology Problem

**Input:** Demand Matrix  $\mathcal{M}$

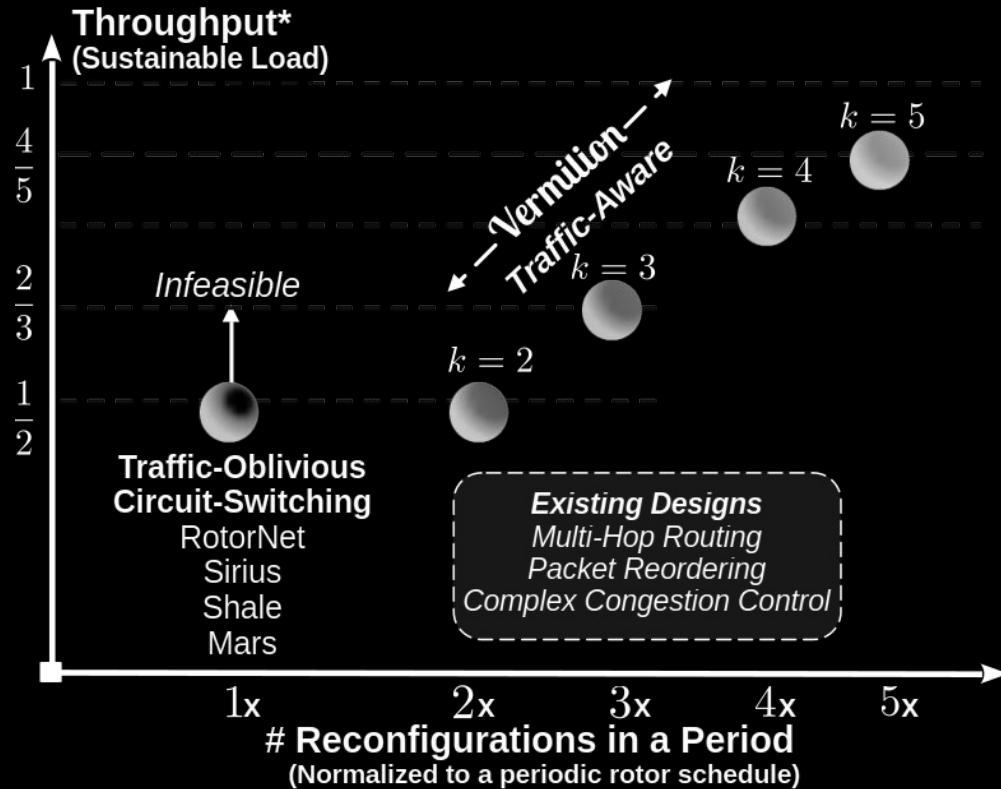
Number of nodes  $n$

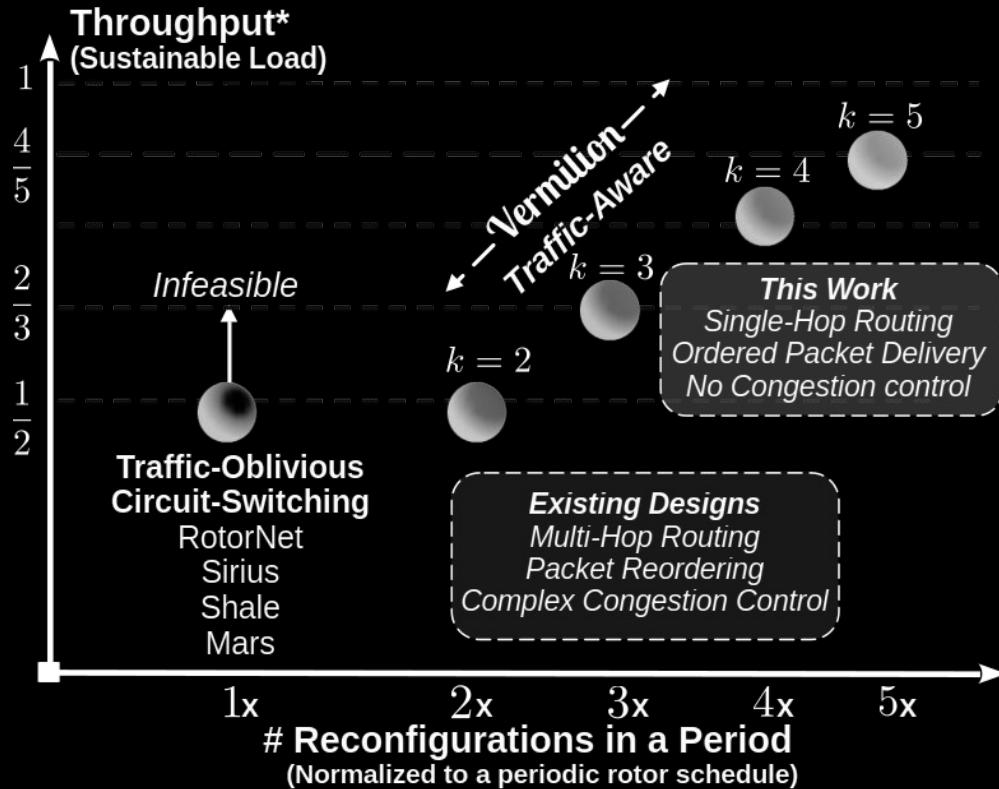
Degree bound  $d'$  in each timeslot

Period bound  $\Gamma$

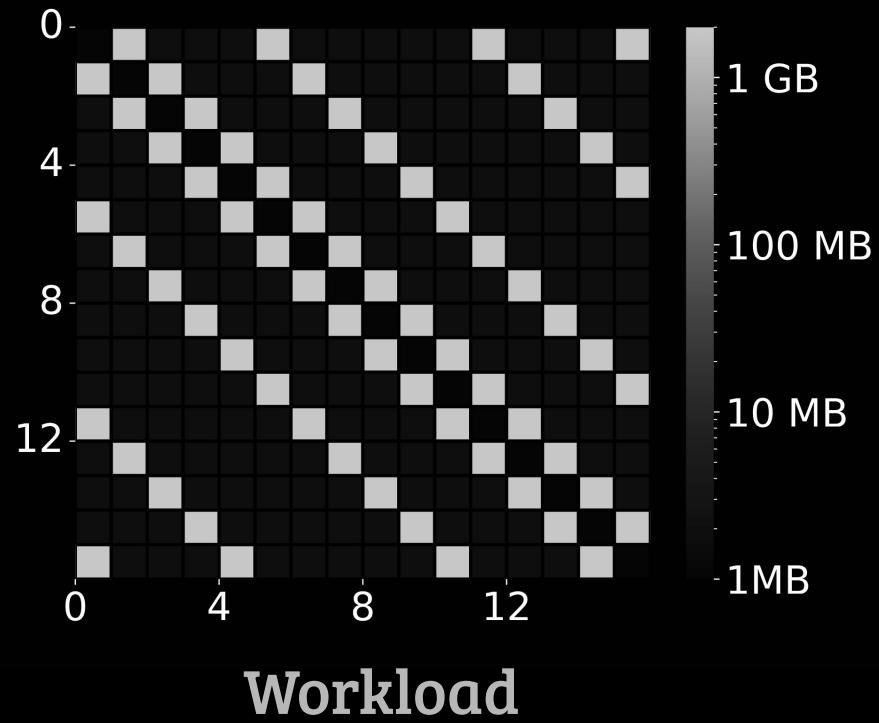
**Output:** Static degree  $d = \Gamma \times d'$  multigraph  $\rightarrow$  Periodic graph

**Maximize:** Throughput

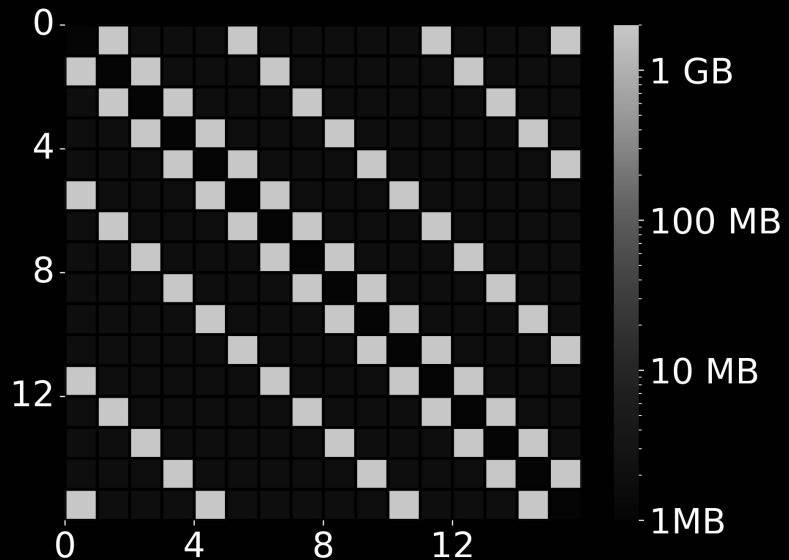




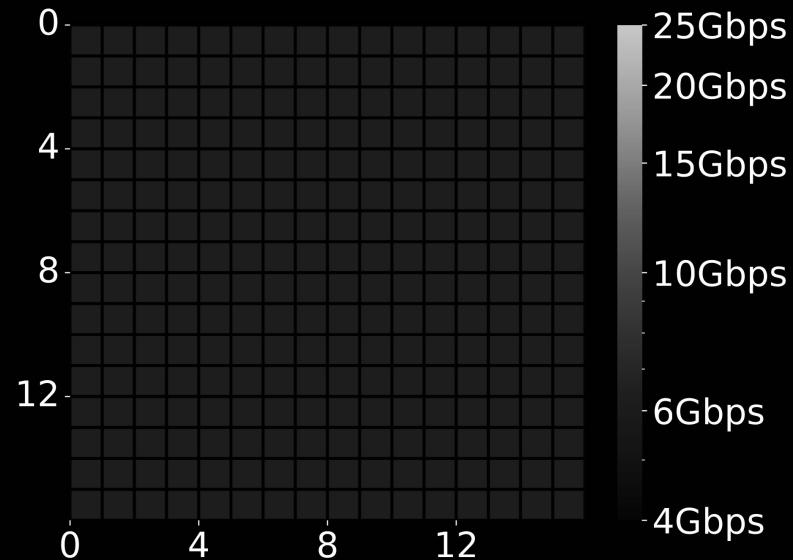
## Example: Deep Learning Recommendation Model Workload



# Example: Deep Learning Recommendation Model Workload

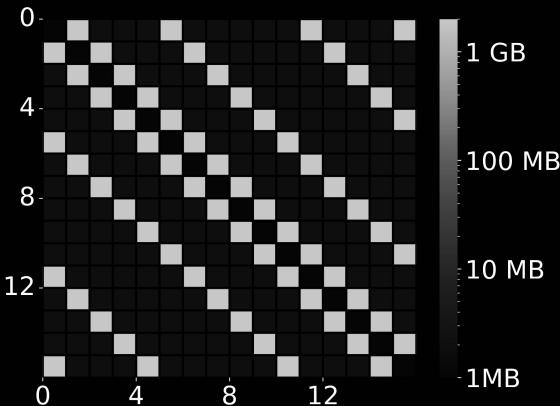


Workload

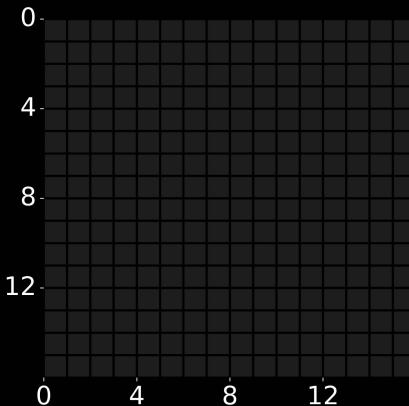


Oblivious Topology

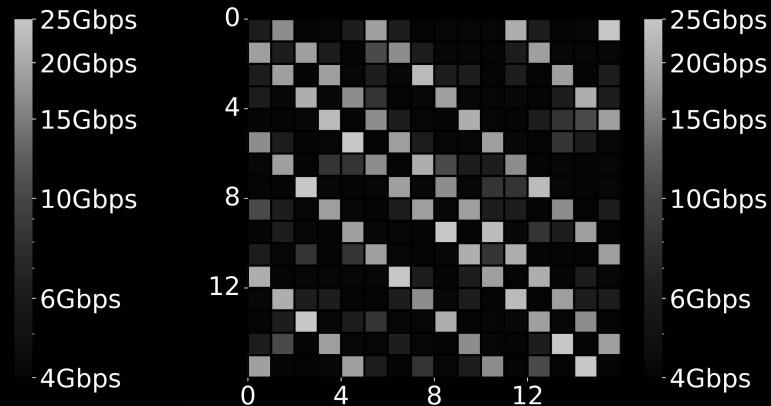
# Example: Deep Learning Recommendation Model Workload



Workload

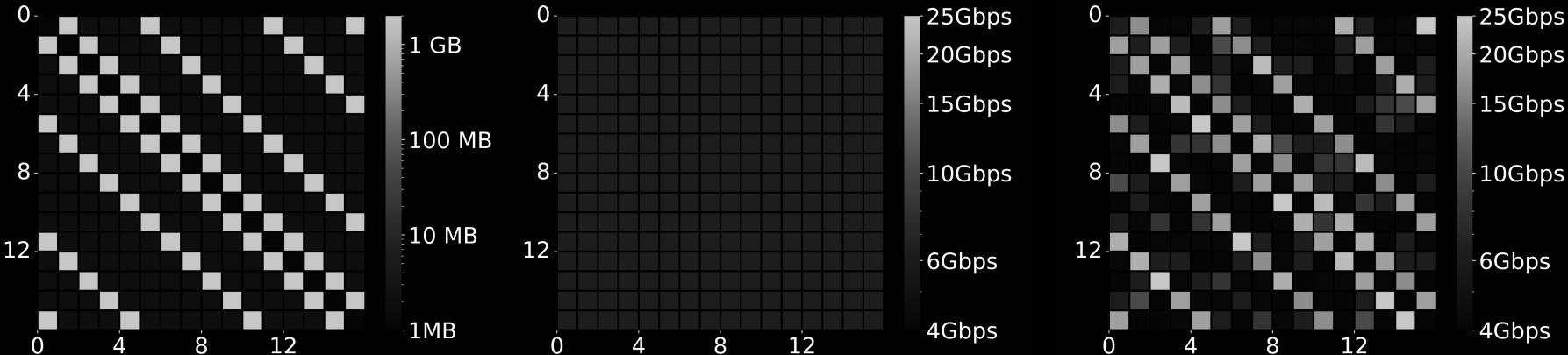


Oblivious



Vermilion

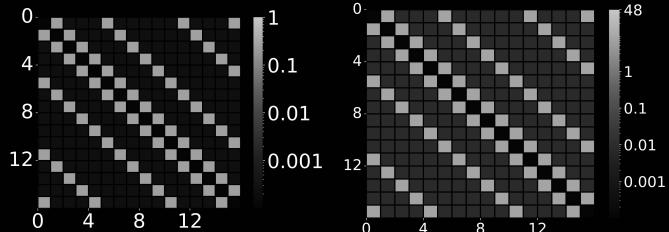
*It's a Match!*



Workload

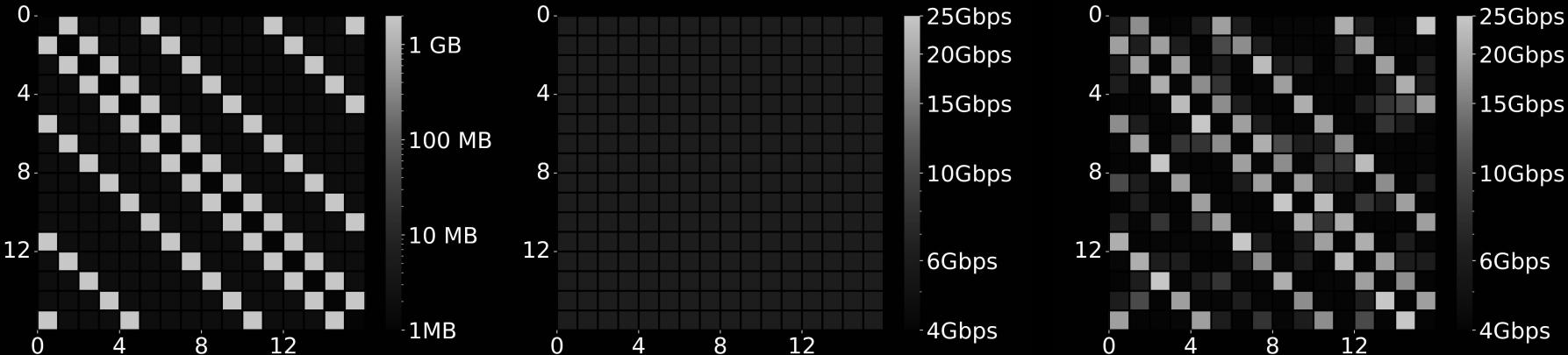
Oblivious

Vermilion



Normalization

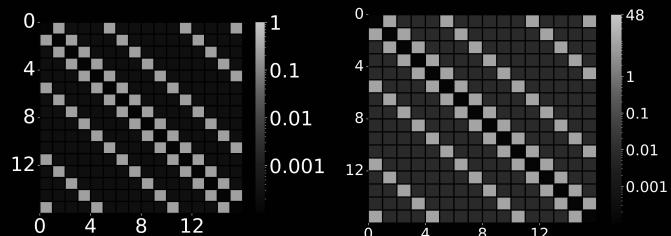
Upscale



**Workload**

**Oblivious**

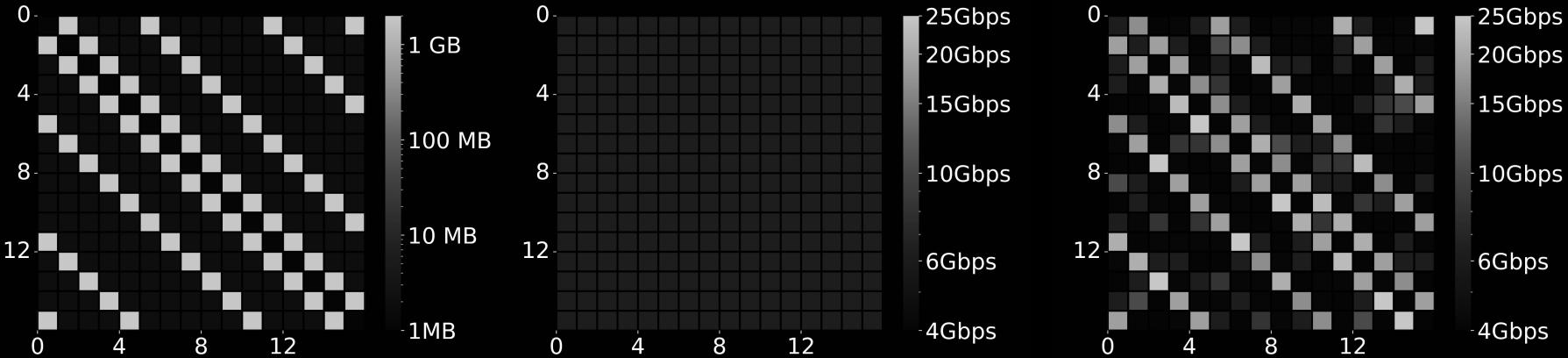
**Vermilion**



**Normalization**

**Upscale**

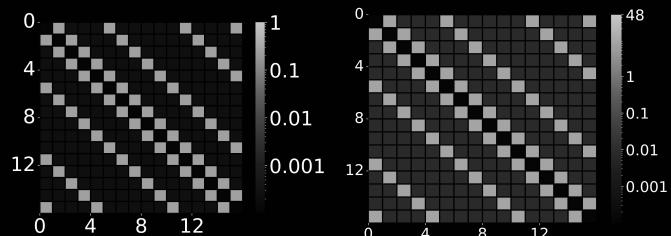
**Matrix  
Rounding**  
**Traffic-aware schedule**



**Workload**

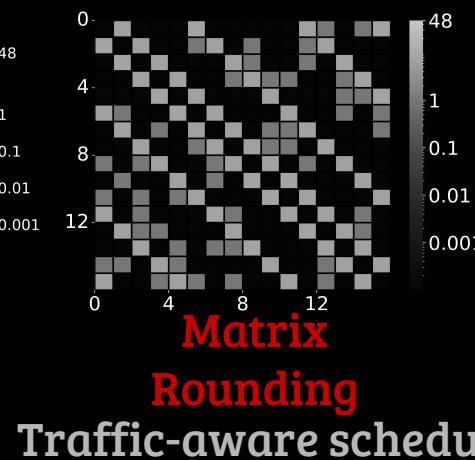
**Oblivious**

**Vermilion**



**Normalization**

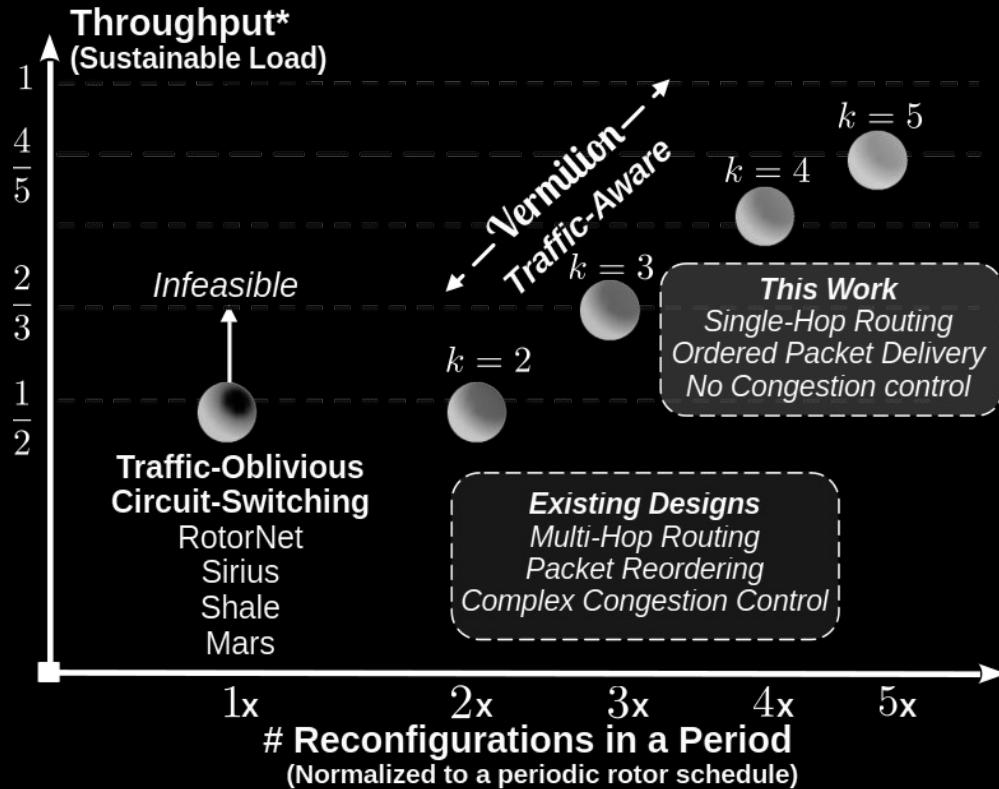
**Upscale**



**Matrix  
Rounding**

**+ Oblivious  
schedule**

**+ Random  
configuration**



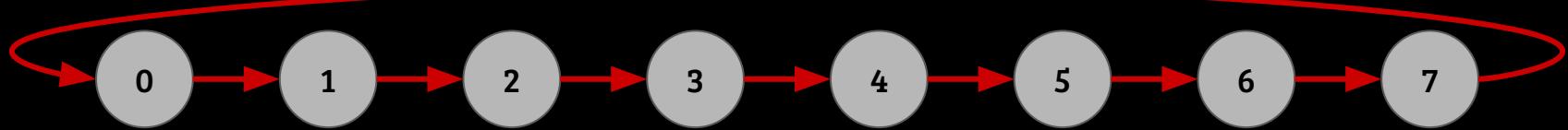
# Insights & Reflection on Metrics for Collective Communication

# Throughput and Reconfiguration Delay

- $\Delta$  = Fraction of time lost in reconfigurations
- Throughput bounds typically scale down by a factor of  $1-\Delta$ 
  - e.g., Throughput bound of periodic switching is  $\frac{1}{2} (1-\Delta)$
- **How does the absolute value of reconfiguration time impact performance?**
  - Is 1 microsecond a satisfactory reconfiguration delay?
  - What about 100 milliseconds?
  - **Note:**  $\Delta$  can be small enough irrespective of the reconfiguration delay

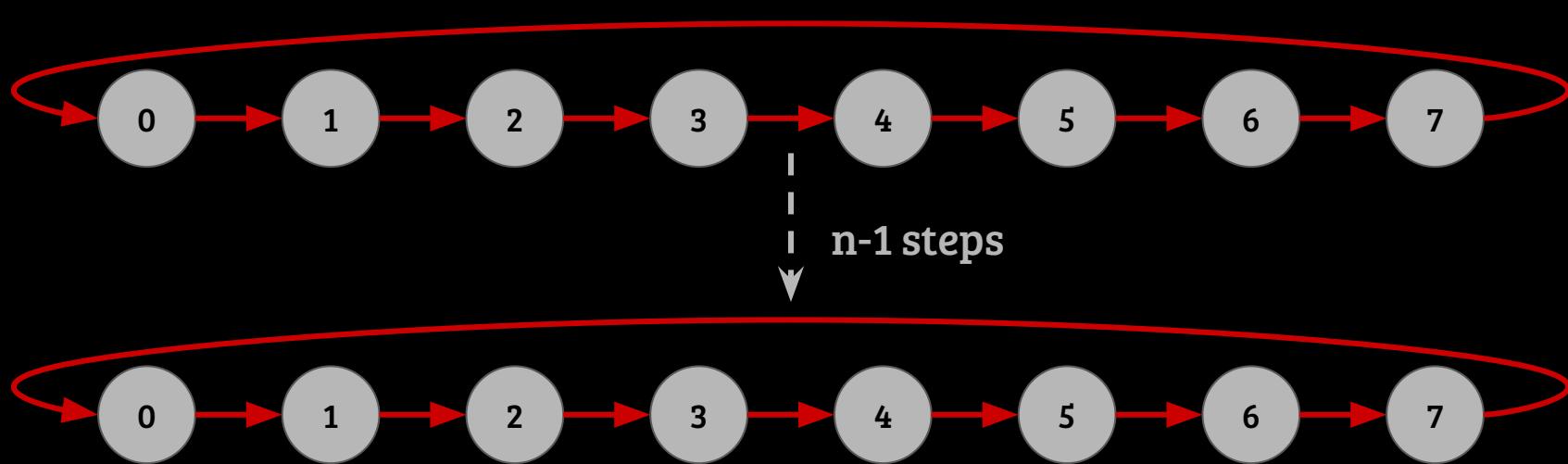
# Ring AllReduce

- The communication pattern is a *matching*



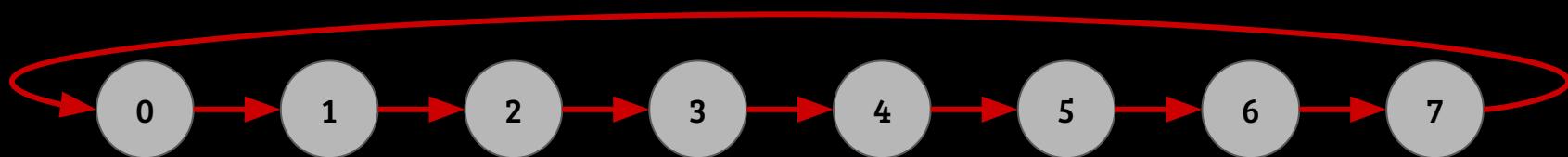
# Ring AllReduce

- The communication pattern is a *matching*



# Ring AllReduce

- The communication pattern is a *matching*



- Periodic circuit switching
  - Prior analysis suggests a throughput of  $1/2$  for the above communication pattern
  - Throughput can in fact be as low as  $1/8$
  - Demand matrix abstraction may be the culprit for these contradictions!
  - Note: New “demand” (next step) only arrives after completing the previous demand

# Key Takeaways

1. Throughput as a metric cannot capture the impact of reconfiguration delay
2. Demand matrix abstraction cannot capture the dependencies in “demand” observed in collective communication

# Modeling the Completion Time using $\alpha, \beta$ Cost Model

$\alpha$  : Initialization time for sending out data

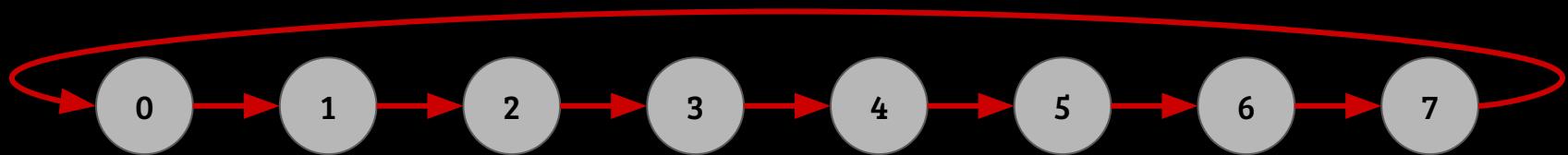
$\beta$  : Transmission delay for sending one bit at line rate

$\lambda$  : Congestion factor (number of flows sharing bandwidth)

$\alpha + m\beta\lambda =$  Time taken to send  $m$  bits in a single step of the algorithm

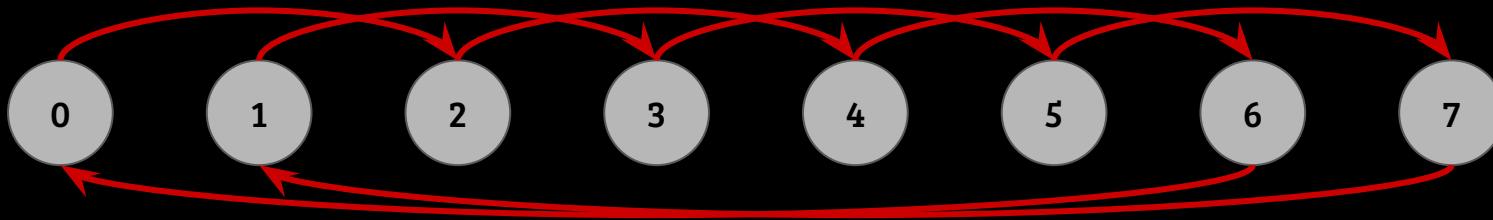
# Recursive Doubling

- Step 1: Matching



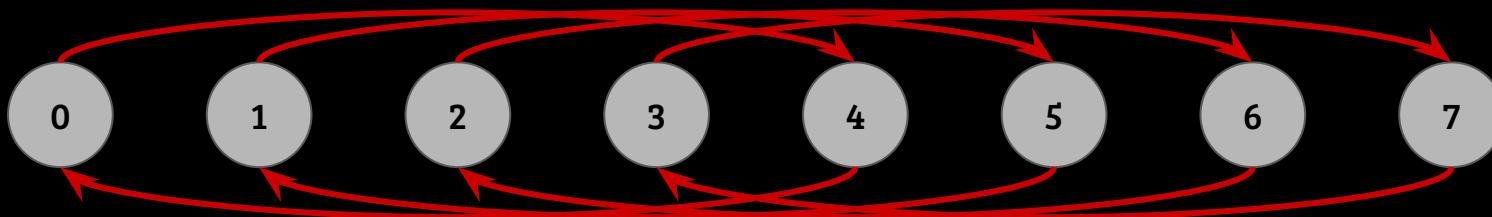
# Recursive Doubling

- Step 2: Matching



# Recursive Doubling

- Step 3: Matching



# Modeling the Completion Time using $\alpha, \beta$ Cost Model

$\alpha$  : Initialization time for sending out data

$\beta$  : Transmission delay for sending one bit

$\lambda$  : Congestion factor (number of flows sharing bandwidth)

$\square$  : Reconfiguration delay

$\alpha + \square + m\beta$  : Reconfigure and align topology to communication matching

$\alpha + m\beta \lambda$  : Mismatch

# Circuit Switching for Collective Communication

- Each step of the algorithm is a matching
- The schedule for circuit switching is given!
- Our goal is to minimize the completion time of the collective
- Decision in each step: Reconfigure or not (a binary variable)

A perfect opportunity for optimization!

More details coming soon :)

# References

- [1] Addanki V, Avin C, Schmid S. [Mars: Near-optimal throughput with shallow buffers in reconfigurable datacenter networks](#). Proceedings of the ACM on Measurement and Analysis of Computing Systems. 2023 Feb 28;7(1):1-43.
- [2] Addanki V, Avin C, Knabe GD, Patronas G, Syrivelis D, Terzenidis N, Bakopoulos P, Marinos I, Schmid S. [Vermilion: A Traffic-Aware Reconfigurable Optical Interconnect with Formal Throughput Guarantees](#). arXiv preprint arXiv:2504.09892. 2025 Apr 14.

# Thank you

Vamsi Addanki

*vamsi@inet.tu-berlin.de*

 @Vamsi\_DT