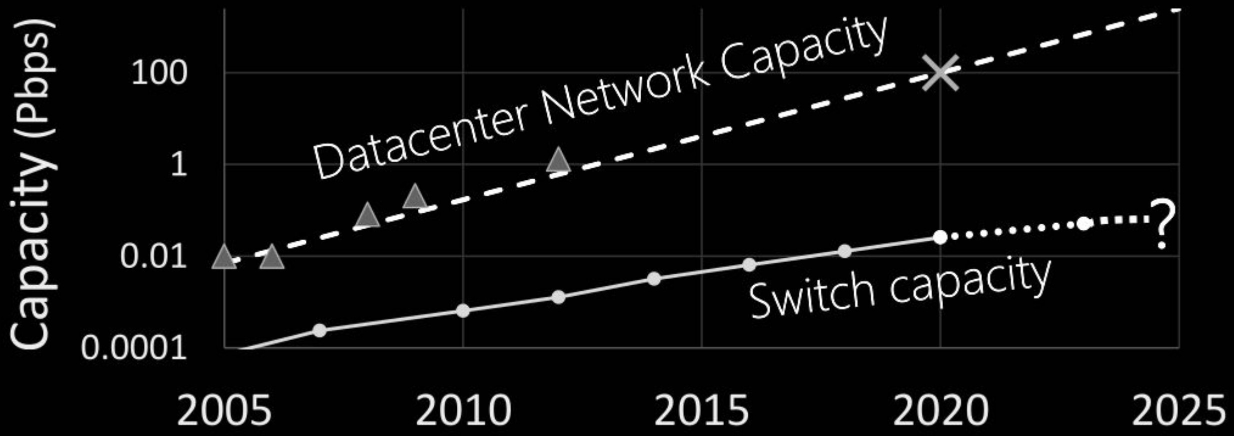
Throughput Bounds of Reconfigurable Networks

Insights & Reflections on Metrics for Collective Communication

Vamsi Addanki

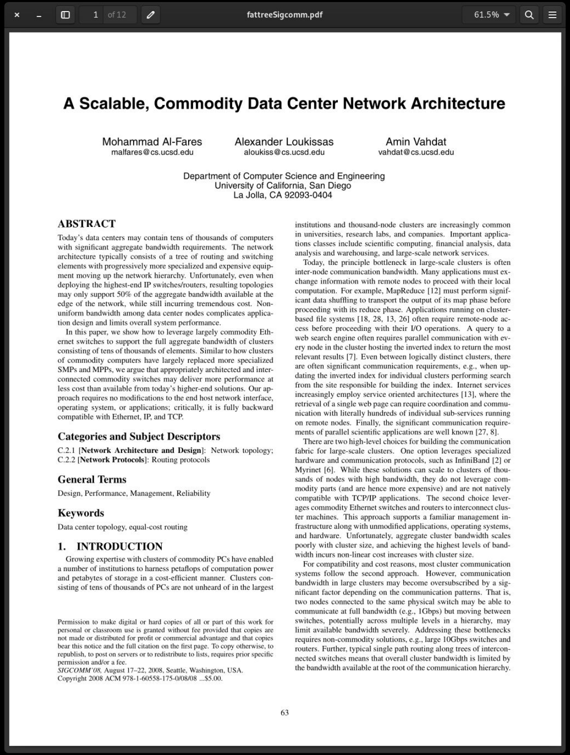
Technische Universität Berlin

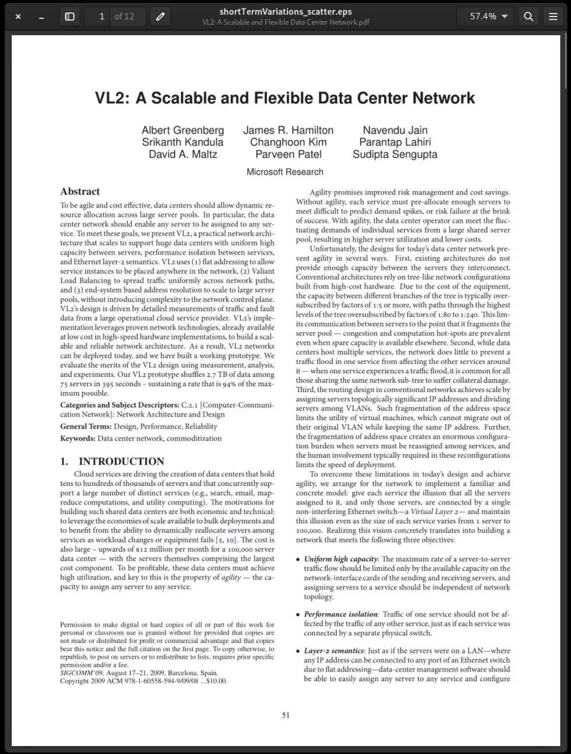
Network Demand vs Capacity Mismatch

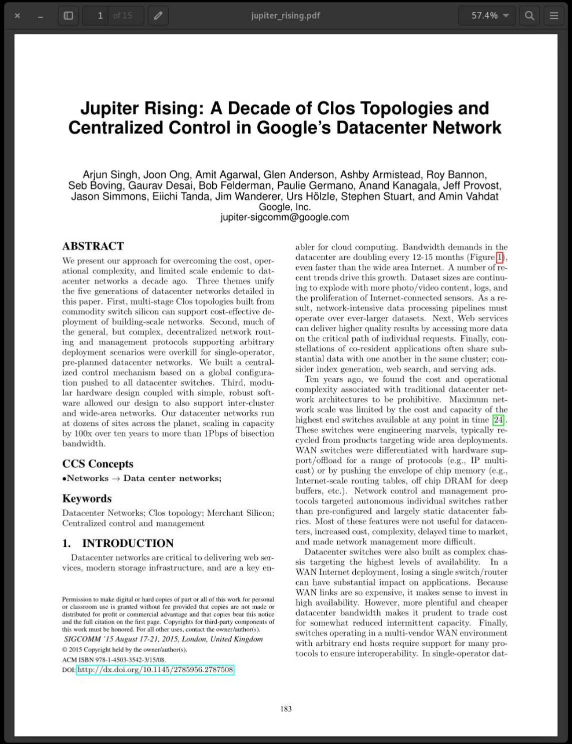


[1] *A flat datacenter network with nanosecond optical switching (SIGCOMM 2020)*

Traditional Approach: Static Datacenter Topologies

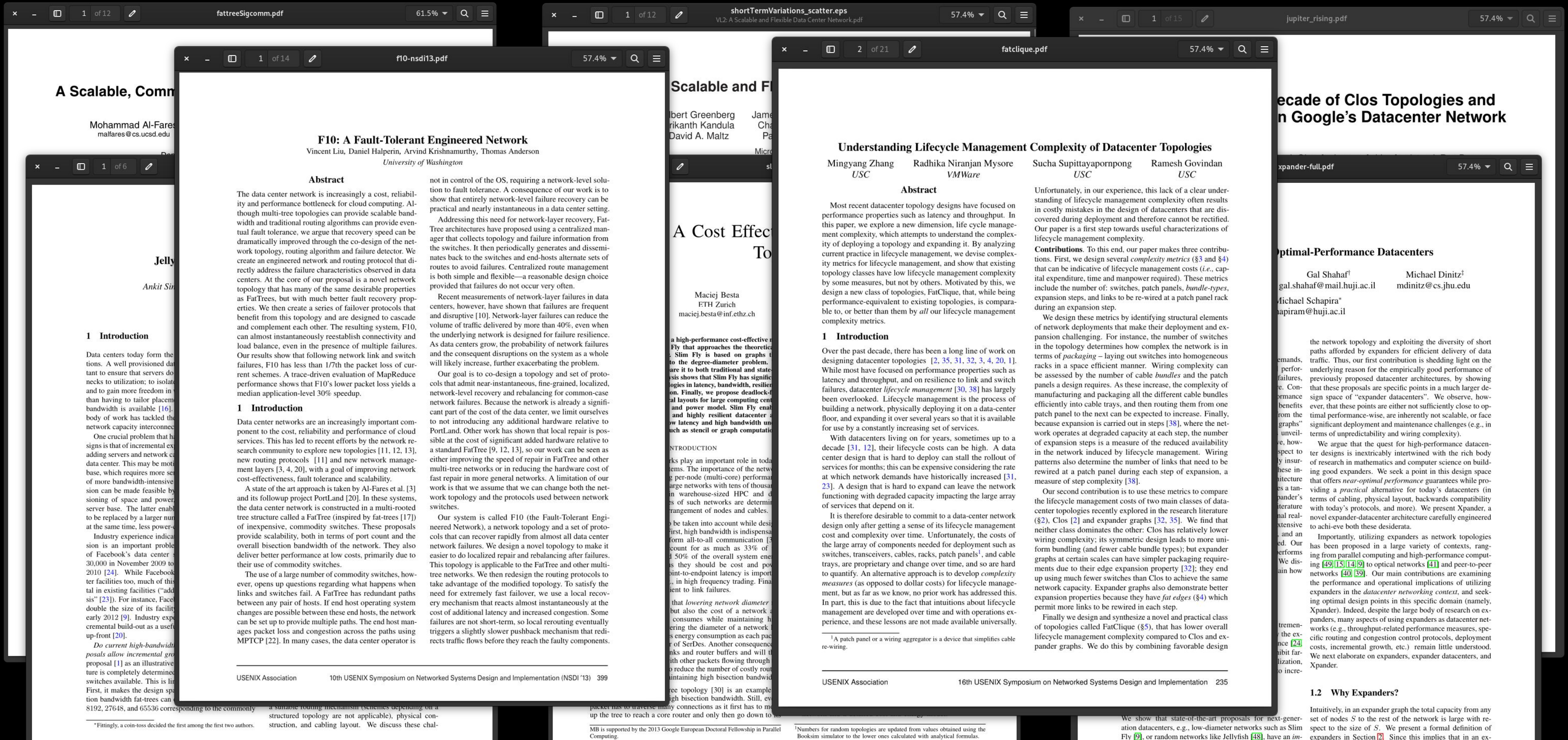






Traditional Approach: Static Datacenter Topologies

Traditional Approach: Static Datacenter Topologies

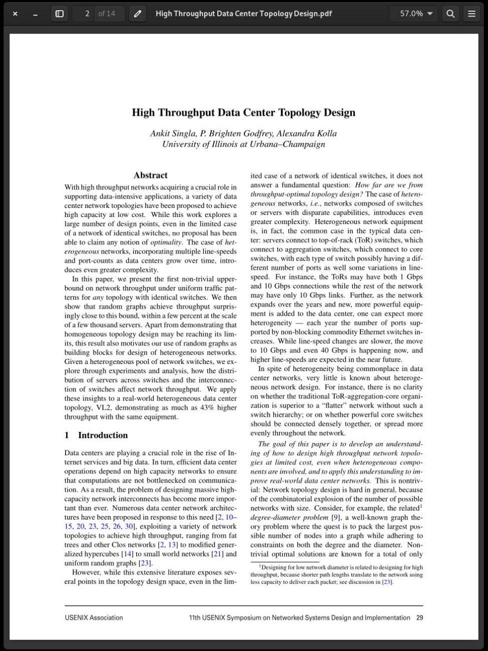


Traditional Approach: Static Datacenter Topologies

Which topology has better throughput?

Traditional Approach: Static Datacenter Topologies

Which topology has better throughput?



Reconfigurable Datacenter Networks

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

Reconfigurable Datacenter Networks

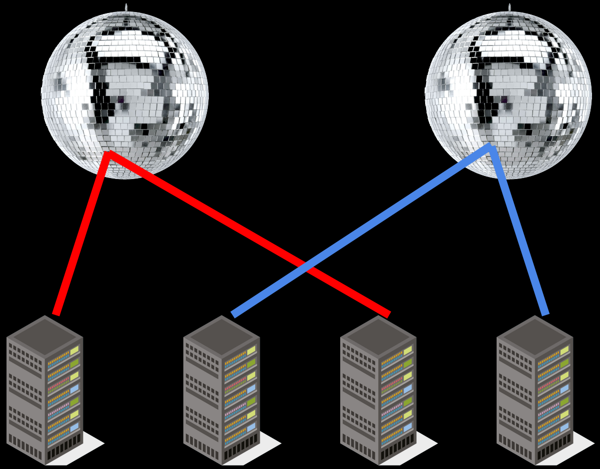
- Generalization of the design space: *Topology can change over time*
- Static networks are a special case

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]



ProjecToR [Sigcomm 2016]

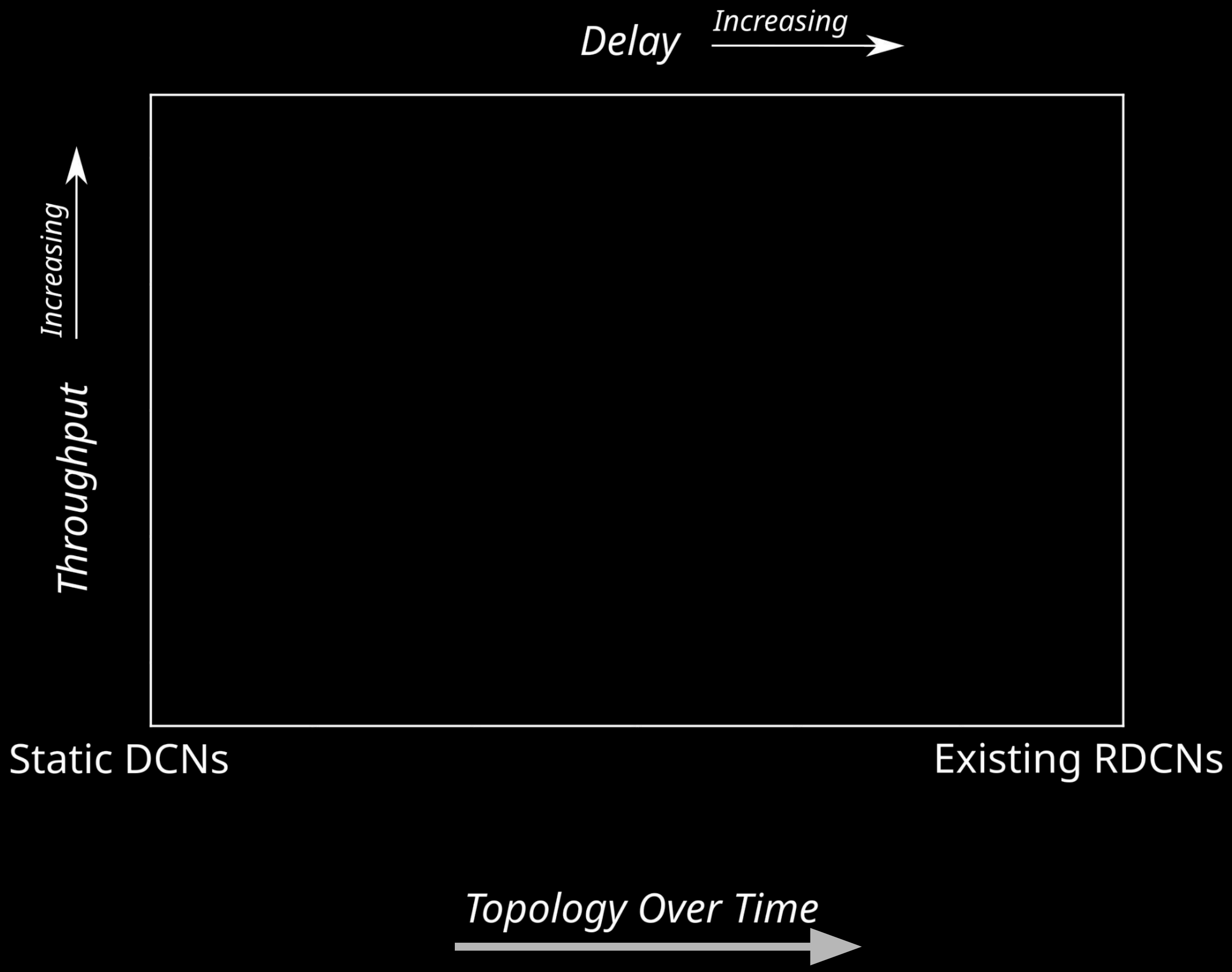
Reconfigurable Datacenter Networks

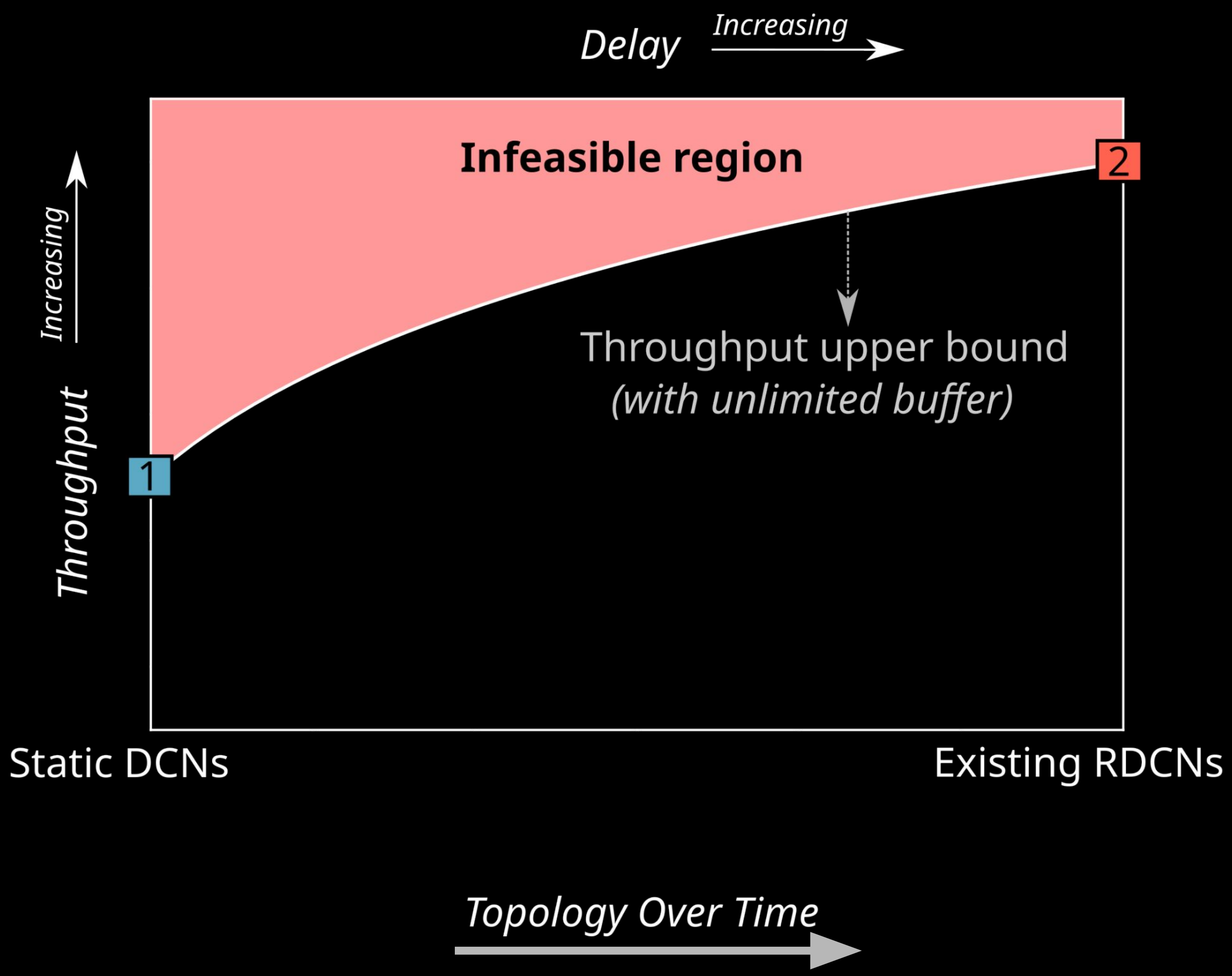
Which topology has better throughput?

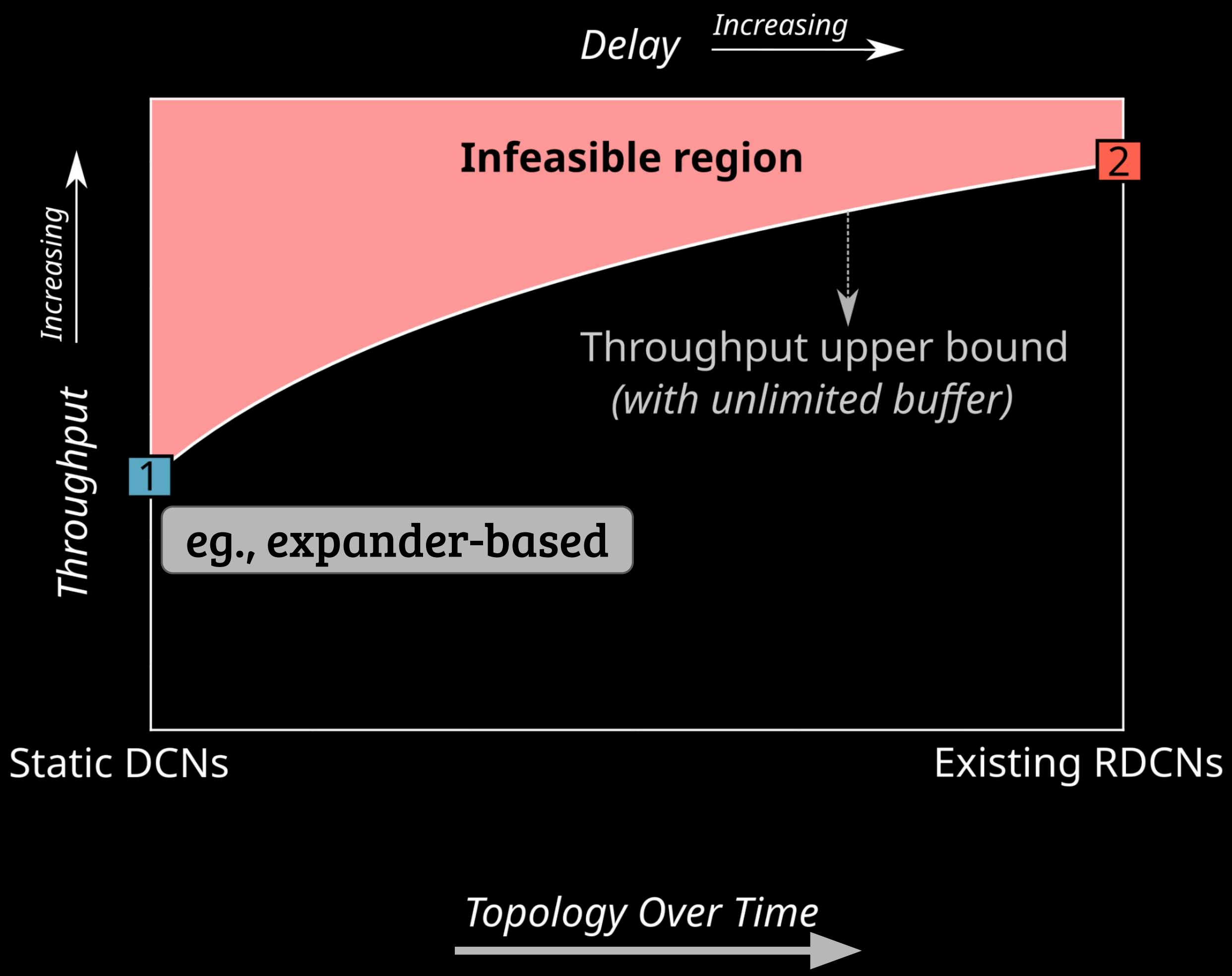
Static DCNs

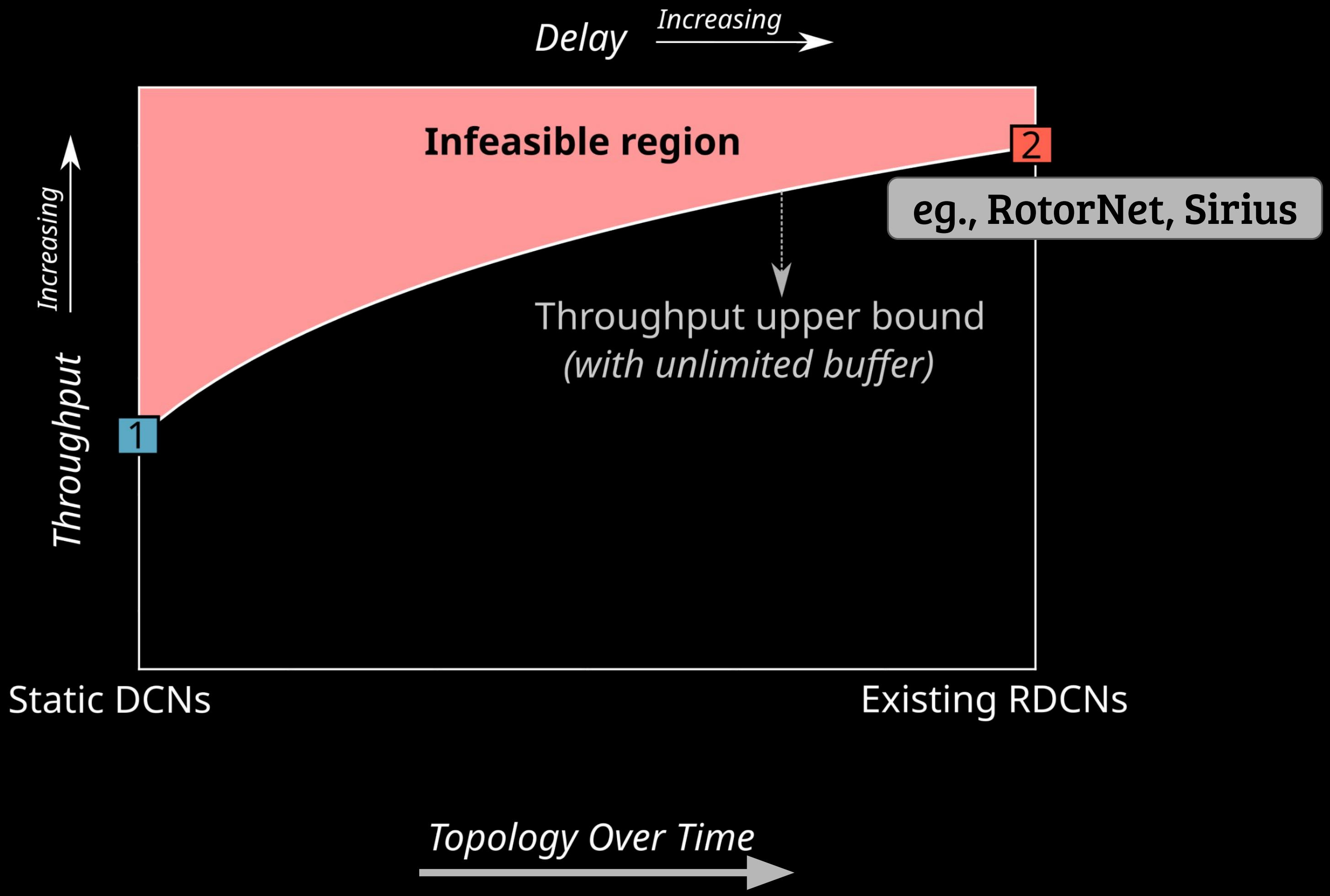
Existing RDCNs

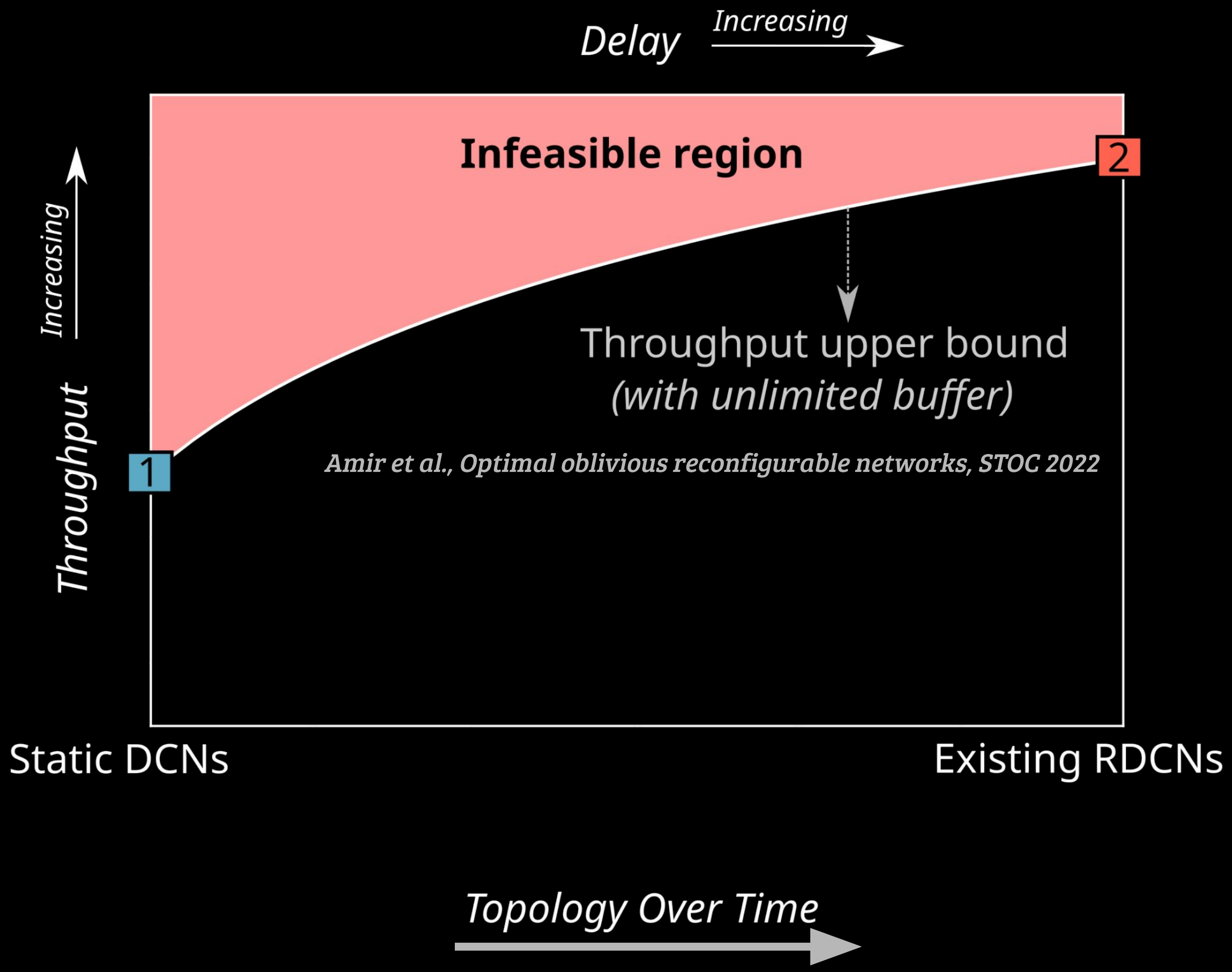
Topology Over Time

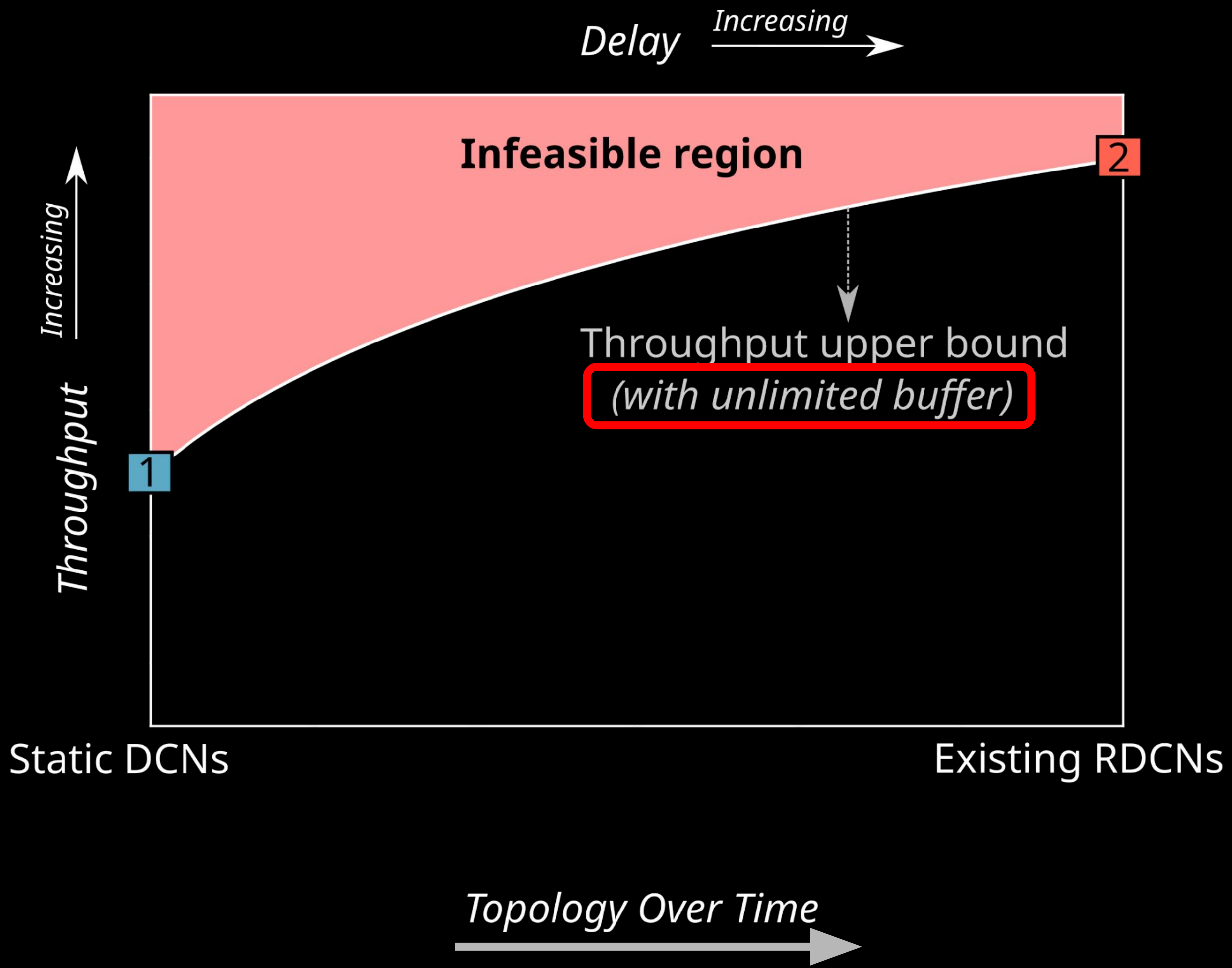

Delay
Increasing
Increasing
Throughput
Static DCNs
Existing RDCNs
Topology Over Time

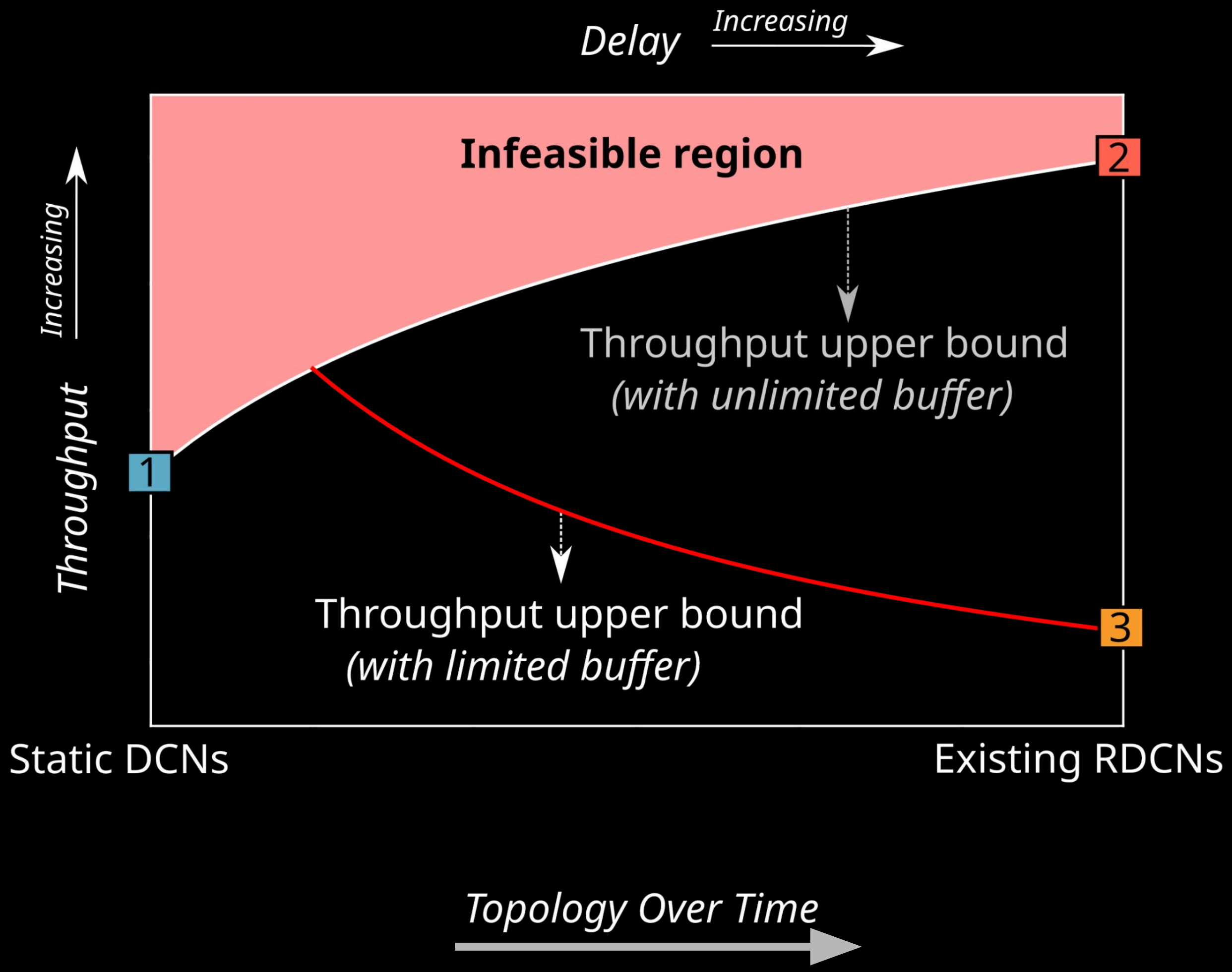

Delay
Increasing
Infeasible region
2
Increasing
Throughput
1
Throughput upper bound
(with unlimited buffer)
Static DCNs
Existing RDCNs
Topology Over Time

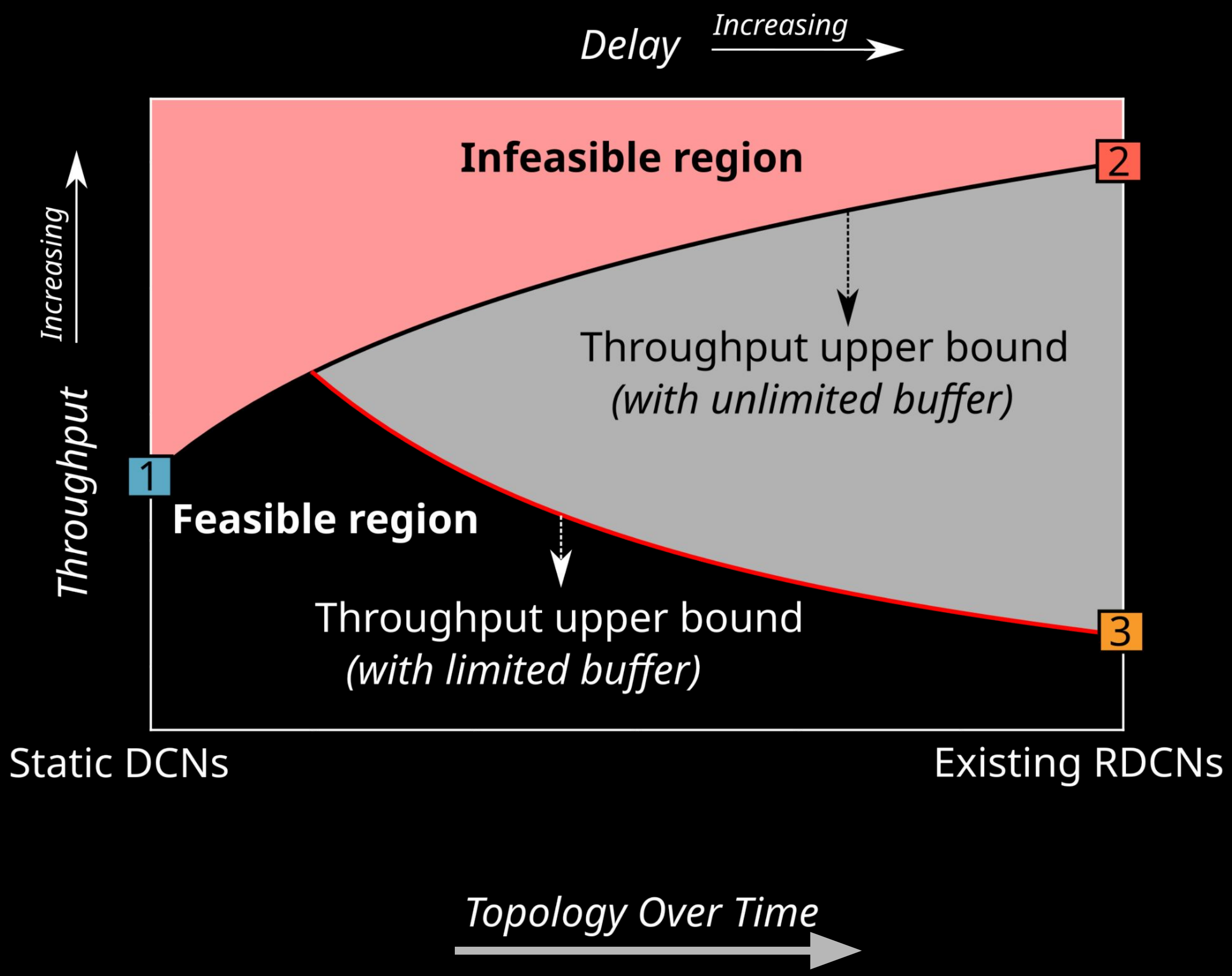

Delay
Increasing
Infeasible region
2
Increasing
Throughput
Throughput upper bound
(with unlimited buffer)
1
eg., expander-based
Static DCNs
Existing RDCNs
Topology Over Time


Delay
Increasing
Infeasible region
2
eg., RotorNet, Sirius
Increasing
Throughput
Throughput upper bound
(with unlimited buffer)
1
Static DCNs
Existing RDCNs
Topology Over Time

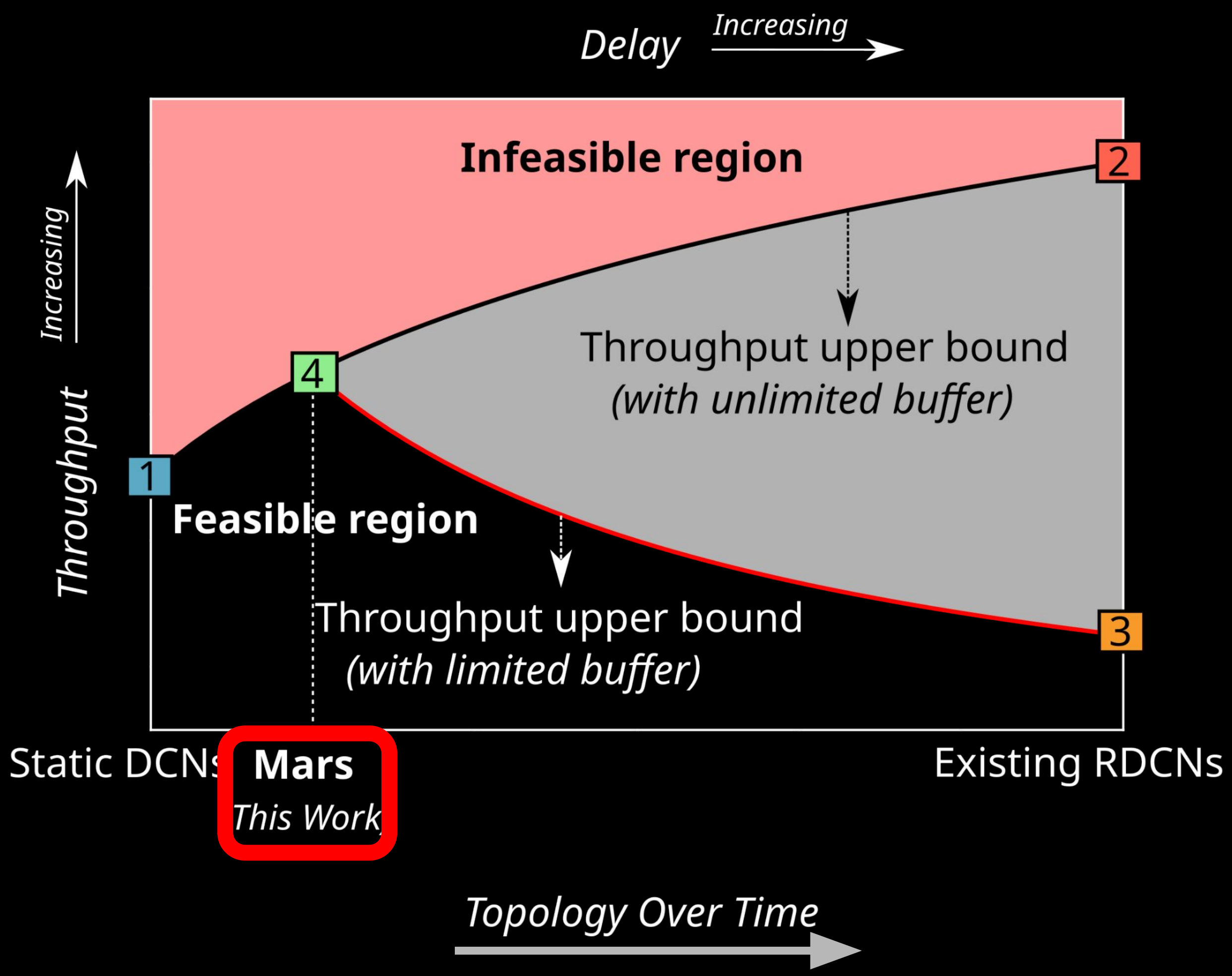

Delay
Increasing
Infeasible region
2
Increasing
Throughput
Throughput upper bound
(with unlimited buffer)
1
Amir et al., Optimal oblivious reconfigurable networks, STOC 2022
Static DCNs
Existing RDCNs
Topology Over Time

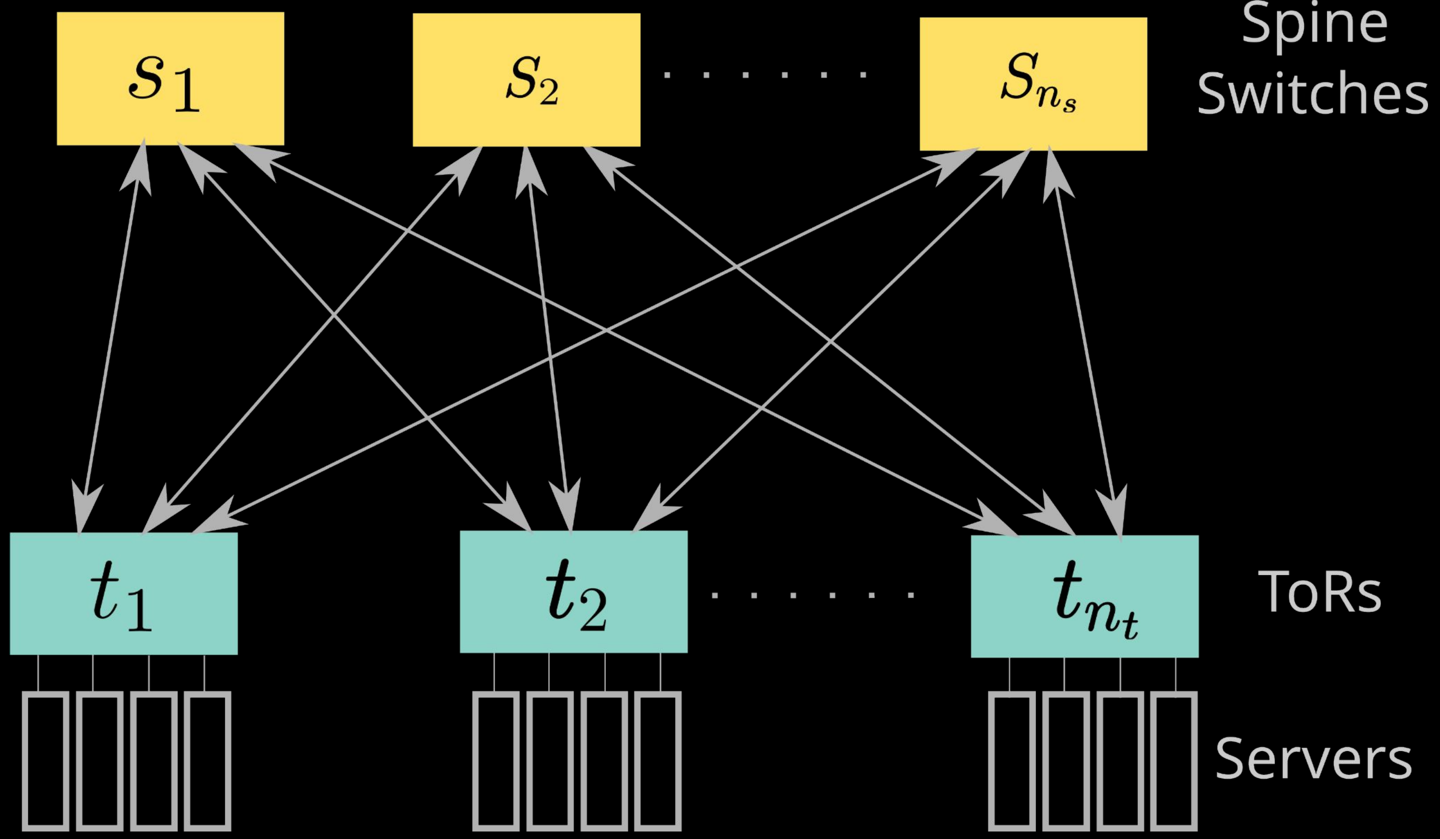

Delay
Increasing
Infeasible region
Increasing
Throughput
2
1
Throughput upper bound
(with unlimited buffer)
Static DCNs
Existing RDCNs
Topology Over Time


Delay
Increasing
Infeasible region
2
Increasing
Throughput
1
Throughput upper bound
(with unlimited buffer)
Throughput upper bound
(with limited buffer)
3
Static DCNs
Existing RDCNs
Topology Over Time

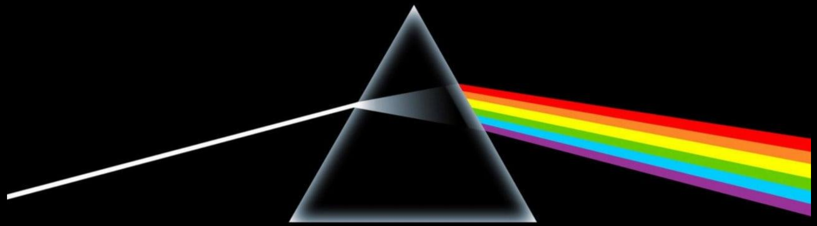


Delay
Increasing
Infeasible region
2
Increasing
Throughput
Throughput upper bound
(with unlimited buffer)
1
Feasible region
Throughput upper bound
(with limited buffer)
3
Static DCNs
Existing RDCNs
Topology Over Time

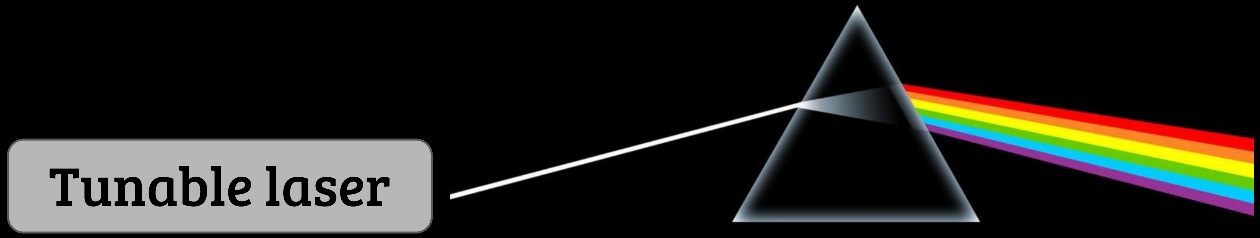

Delay
Increasing
Infeasible region
2
Increasing
Throughput
4
Throughput upper bound
(with unlimited buffer)
1
Feasible region
Throughput upper bound
(with limited buffer)
3
Static DCNs
Mars
This Work
Existing RDCNs
Topology Over Time


s_1
s_2
s_{n_s}
Spine
Switches
t_1
t_2
t_{n_t}
ToRs
Servers

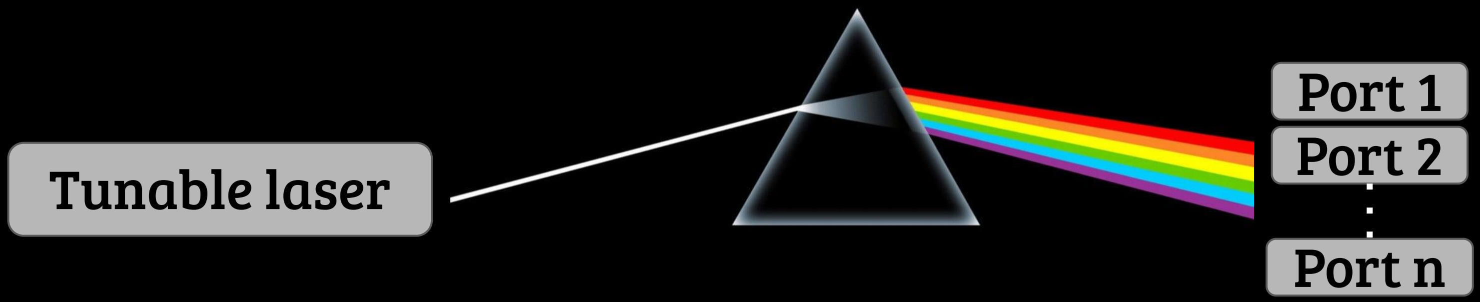
Emerging Technologies - Optical Networks on the Rise



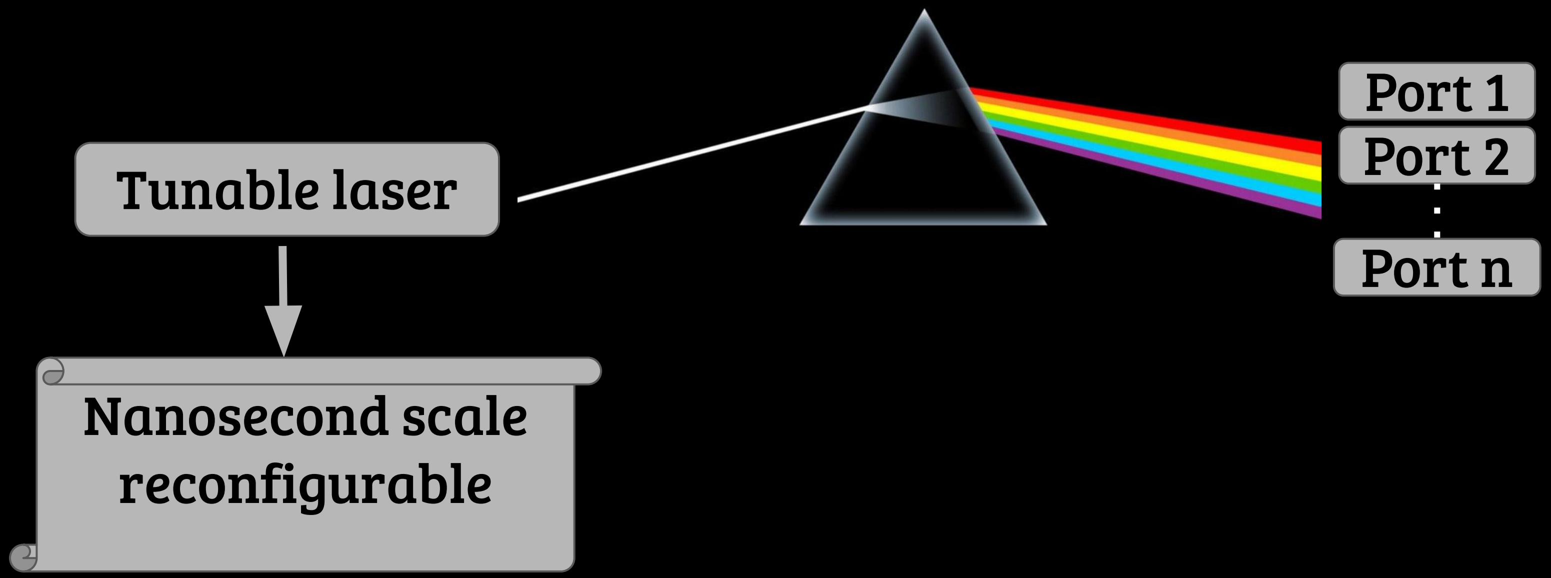
Emerging Technologies - Optical Networks on the Rise



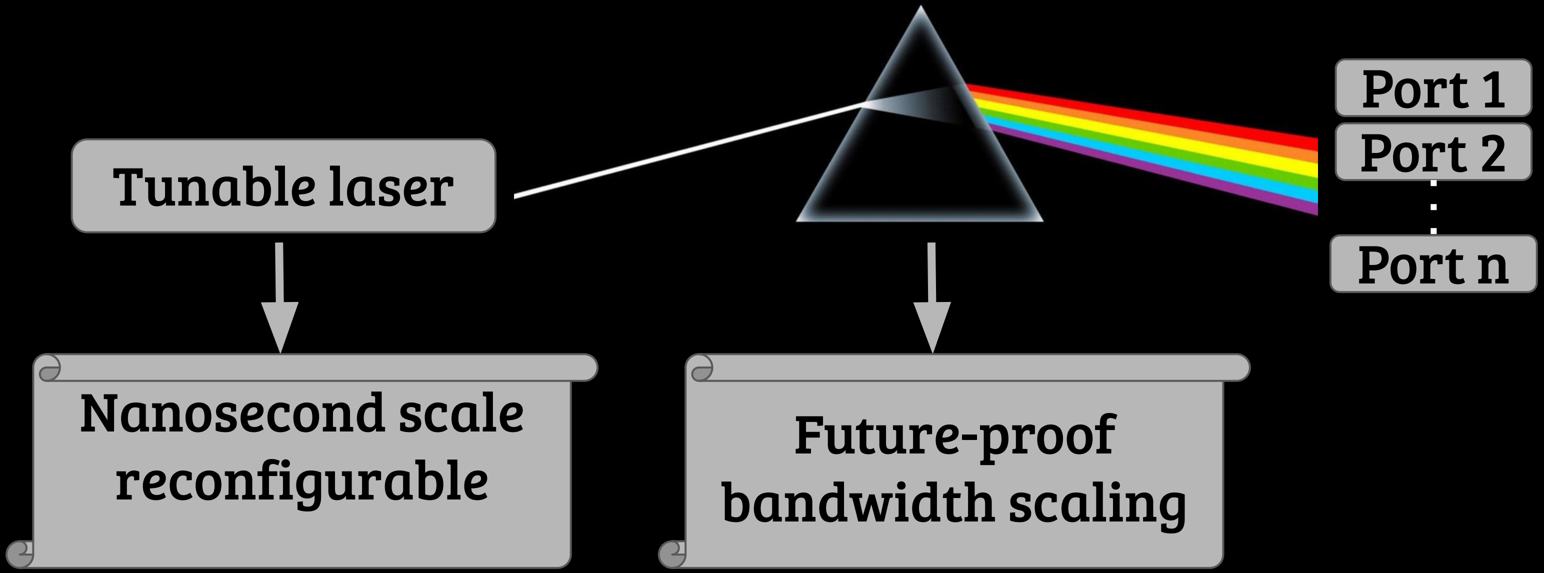
Emerging Technologies - Optical Networks on the Rise



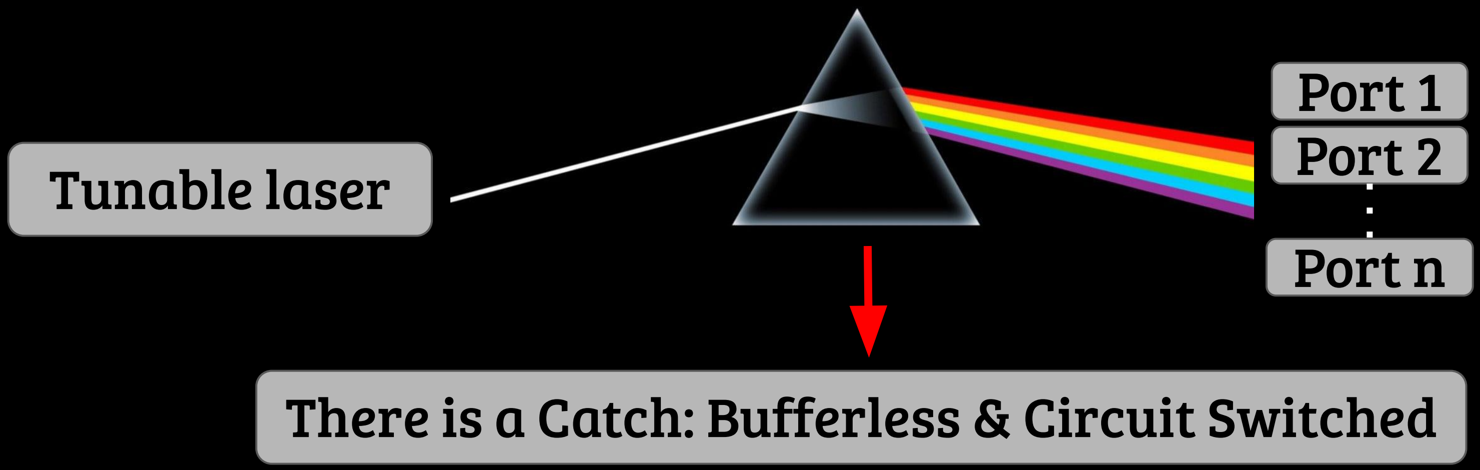
Emerging Technologies - Optical Networks on the Rise

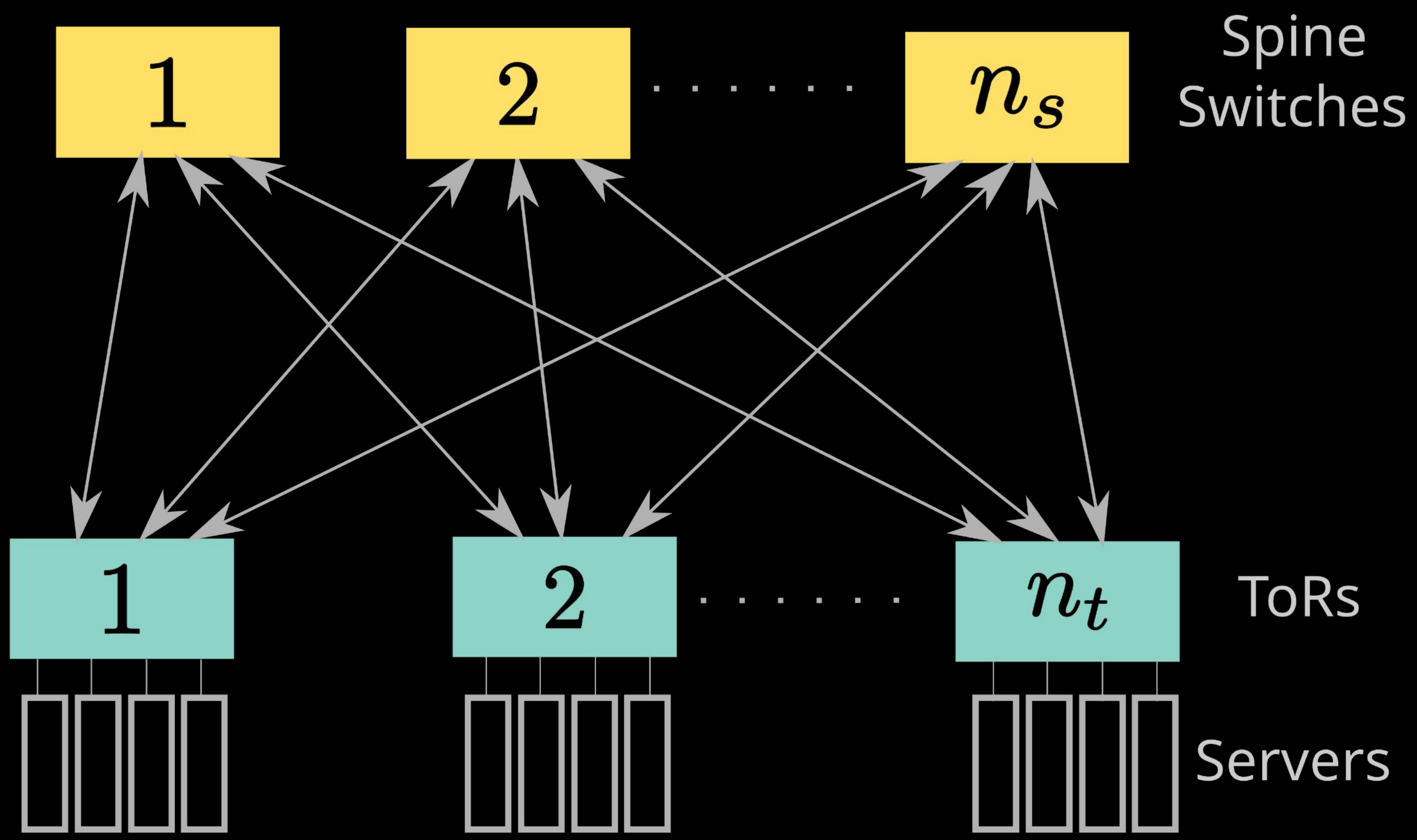


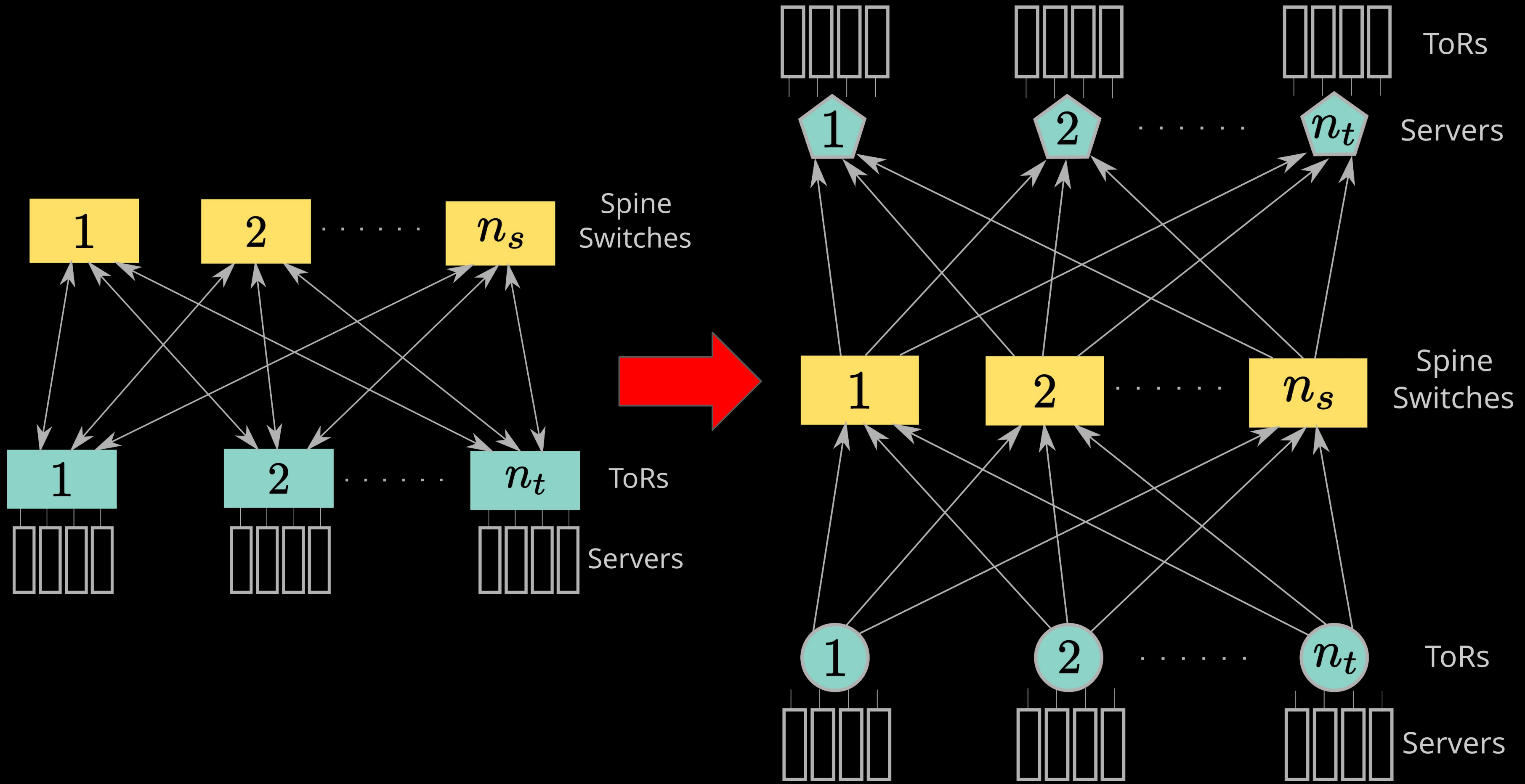
Emerging Technologies - Optical Networks on the Rise



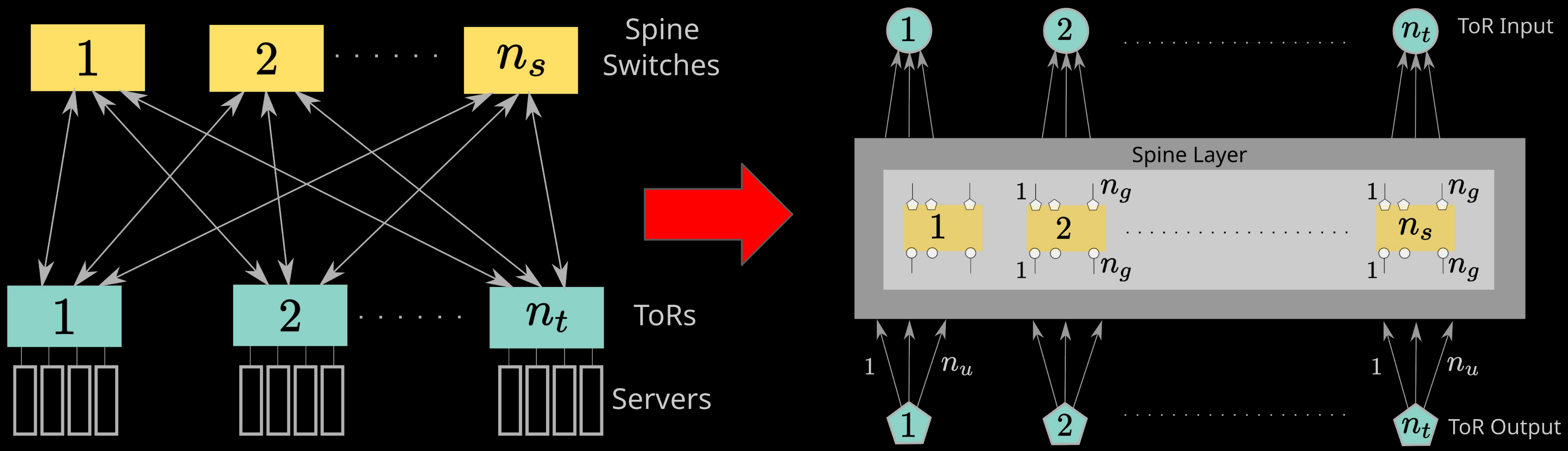
Emerging Technologies - Optical Networks on the Rise

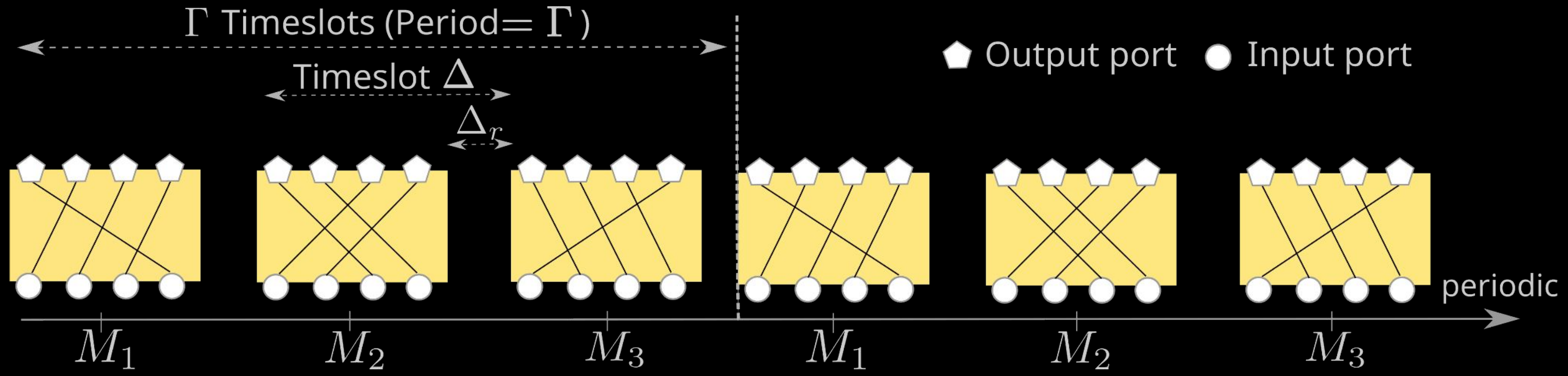


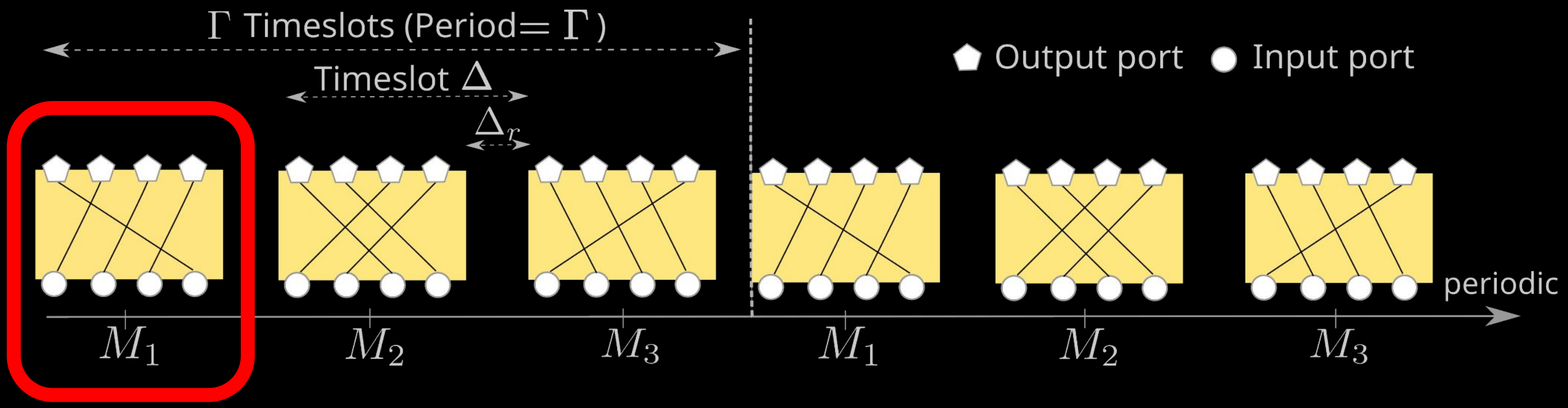

1
2
n_s
Spine Switches
1
2
n_t
ToRs
Servers



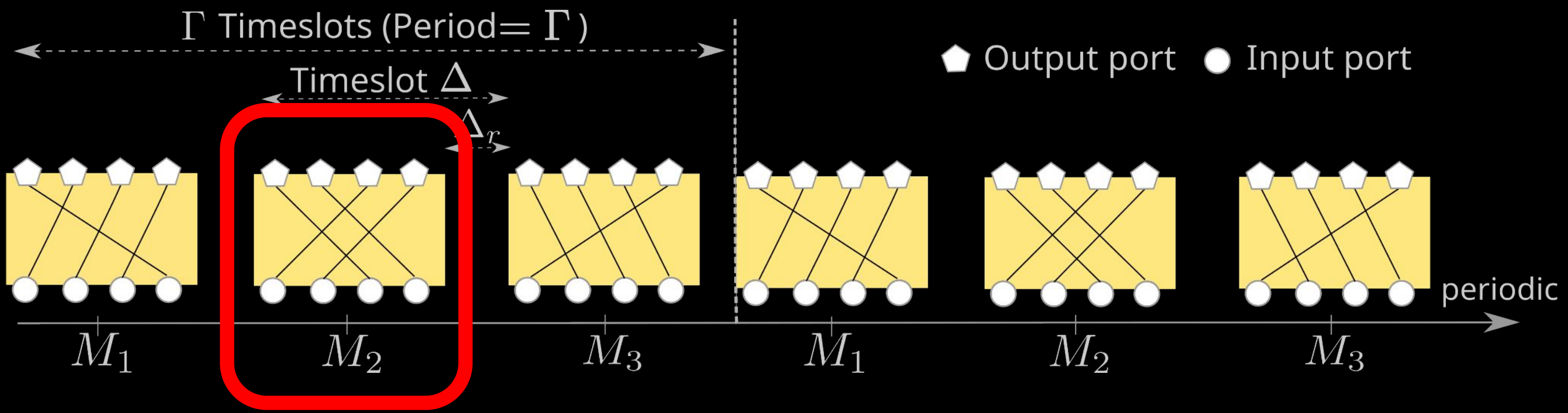
1
2
n_s
Spine Switches
1
2
n_t
ToRs
Servers
ToRs
1
2
n_t
Servers
1
2
n_s
Spine Switches
1
2
n_t
ToRs
Servers


1
2
n_s
Spine Switches
1
2
n_t
ToRs
Servers
1
2
n_t
ToR Input
Spine Layer
1
2
n_s
n_g
1
n_u
1
2
n_t
ToR Output

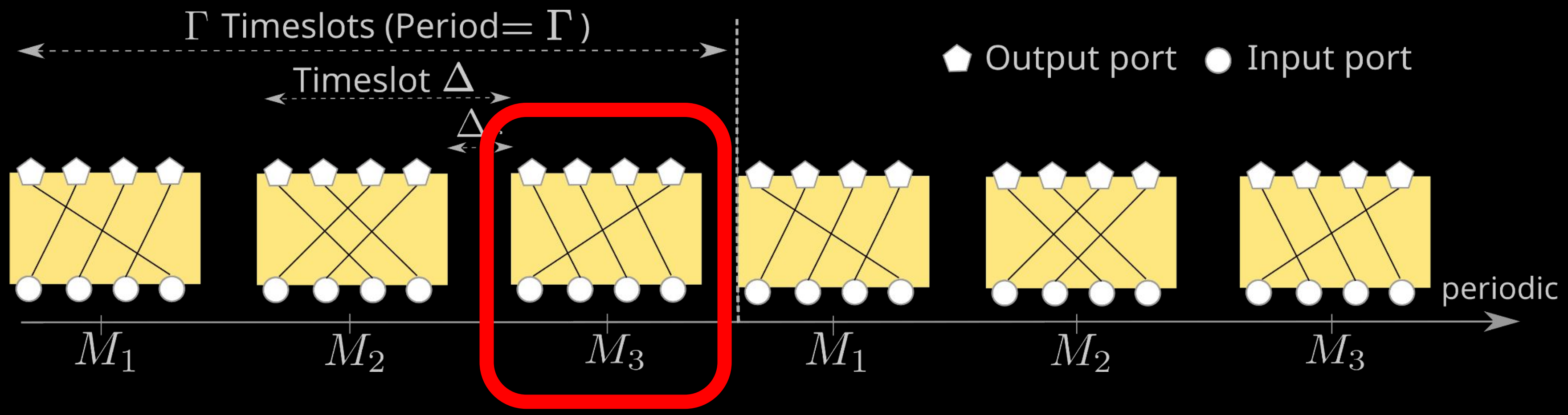

Γ Timeslots (Period$= \Gamma$)
Timeslot Δ
Δ_r
Output port
Input port
periodic
M_1
M_2
M_3
M_1
M_2
M_3



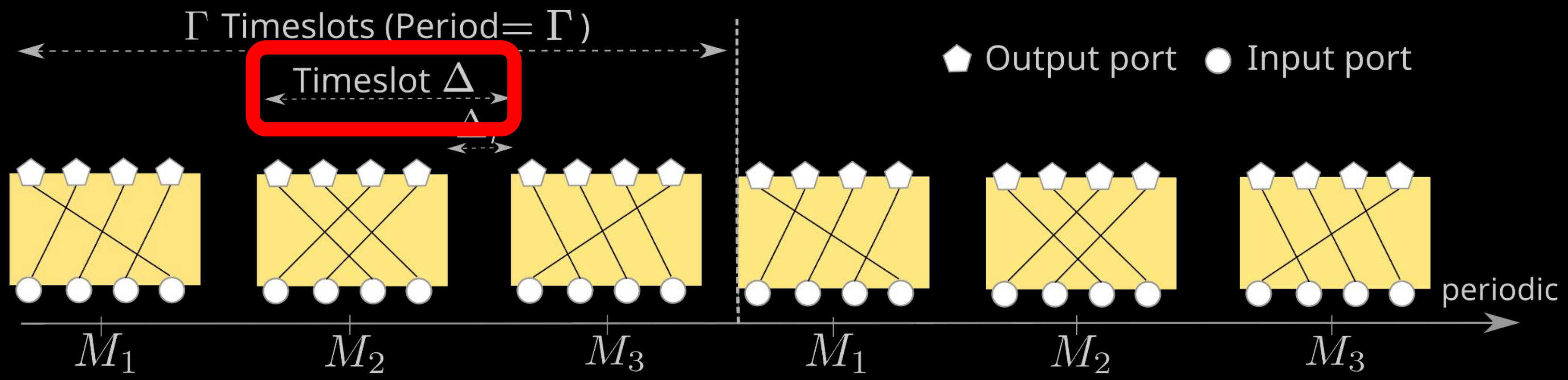
Γ Timeslots (Period$=\Gamma$)
Timeslot Δ
Δ_r
Output port
Input port
periodic
M_1
M_2
M_3
M_1
M_2
M_3



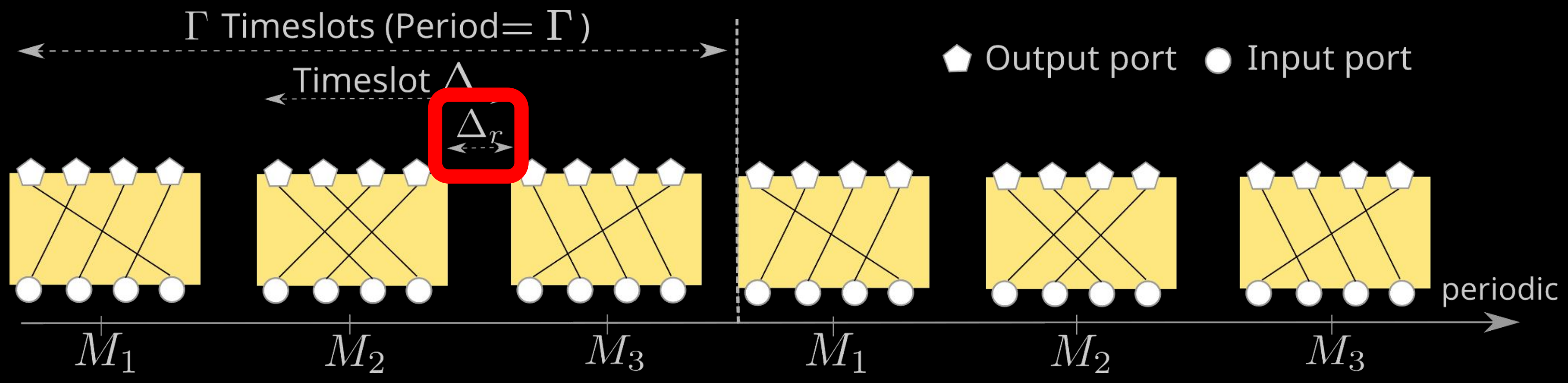
Γ Timeslots (Period= Γ)
Timeslot Δ
Δ_r
Output port
Input port
periodic
M_1
M_2
M_3
M_1
M_2
M_3



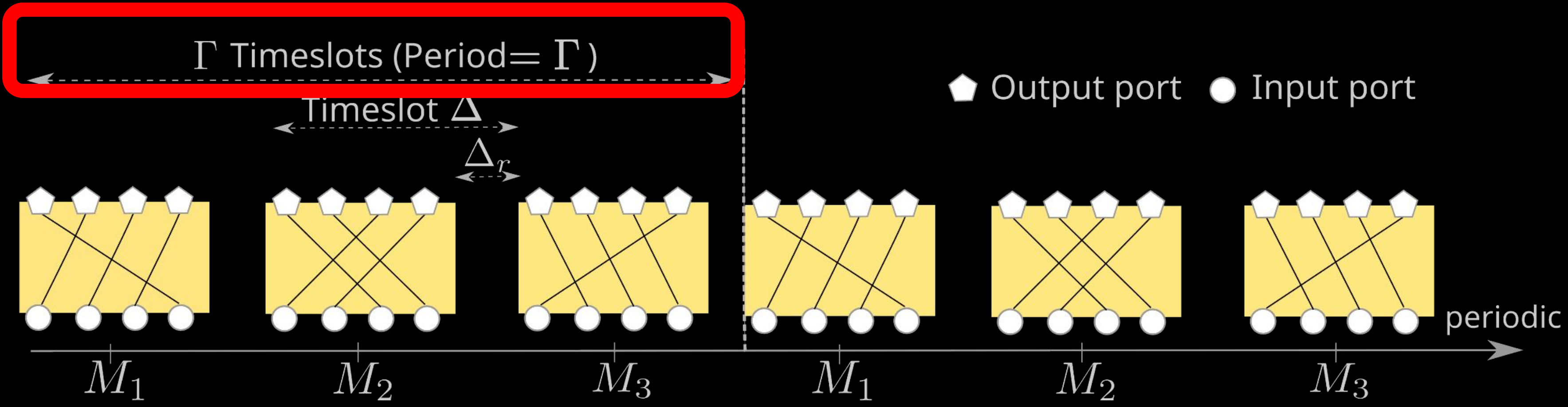
Γ Timeslots (Period= Γ)
Timeslot Δ
Δ
Output port
Input port
periodic
M_1
M_2
M_3
M_1
M_2
M_3



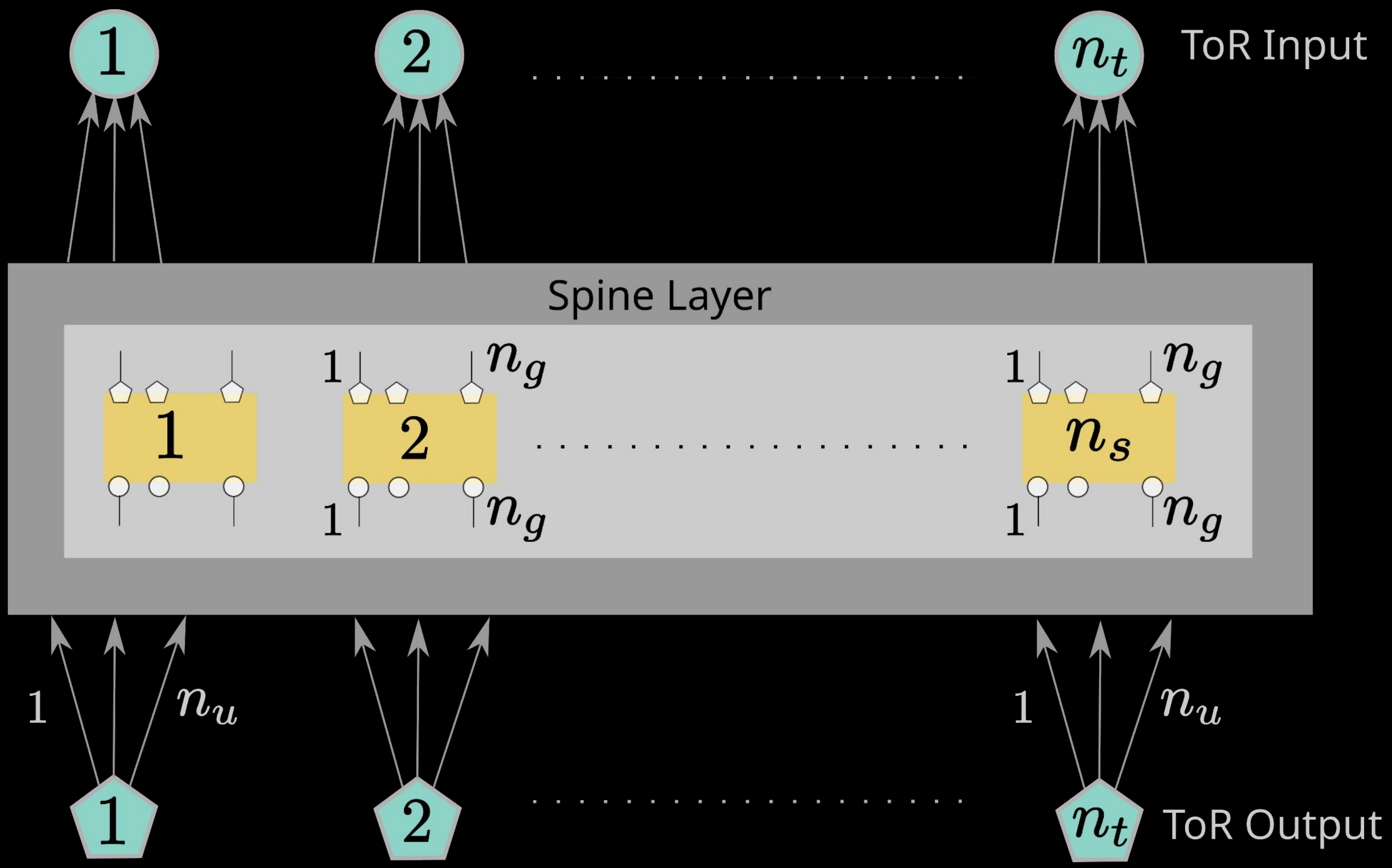
Γ Timeslots (Period= Γ)
Timeslot Δ
Output port
Input port
periodic
M_1
M_2
M_3
M_1
M_2
M_3



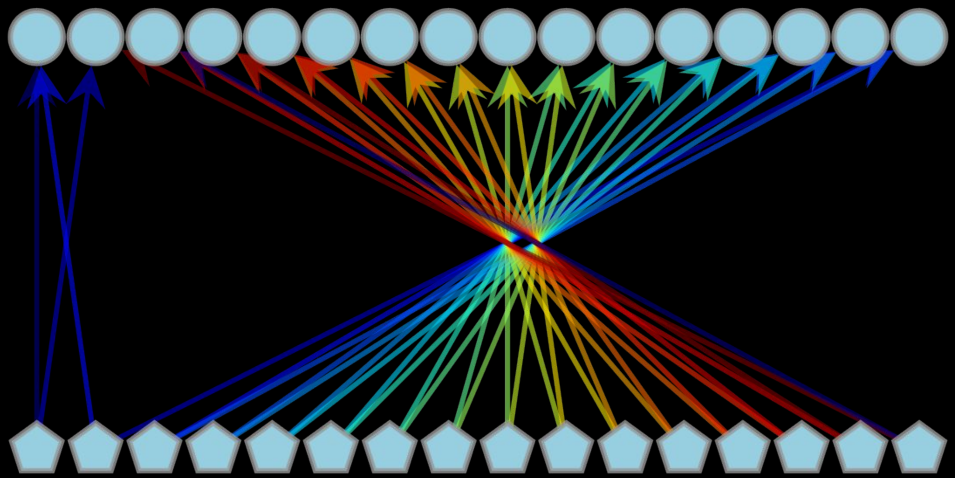
Γ Timeslots (Period= Γ)
Timeslot Δ
Δr
Output port
Input port
periodic
M1
M2
M3
M1
M2
M3



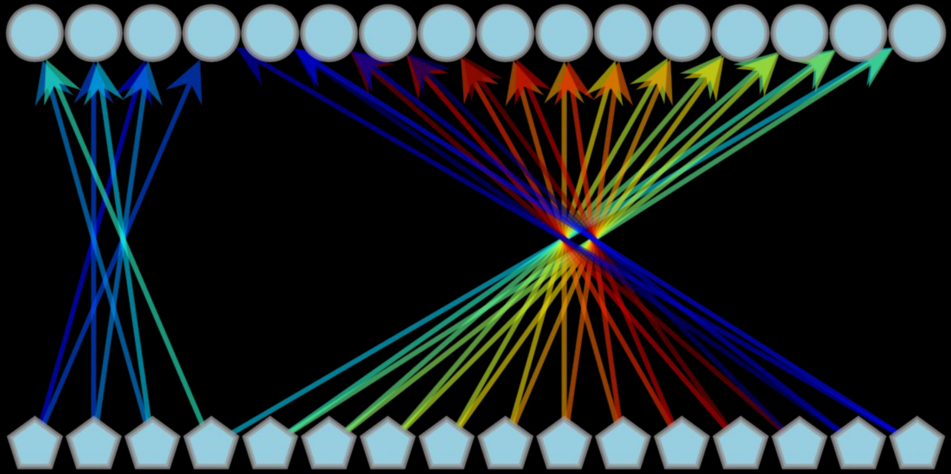
Γ Timeslots (Period= Γ)
Timeslot Δ
Δ_r
Output port
Input port
periodic
M_1
M_2
M_3
M_1
M_2
M_3


ToR Input
Spine Layer
1
2
n_s
n_g
n_u
n_t
ToR Output

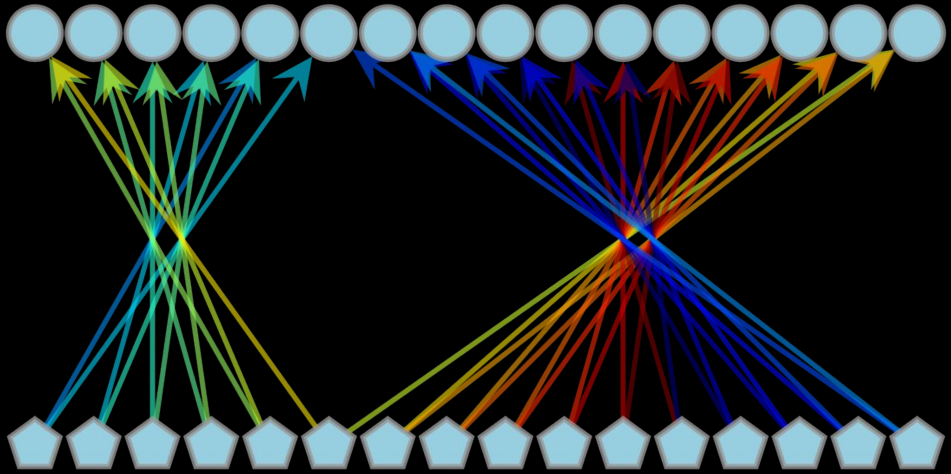
Timeslot 1



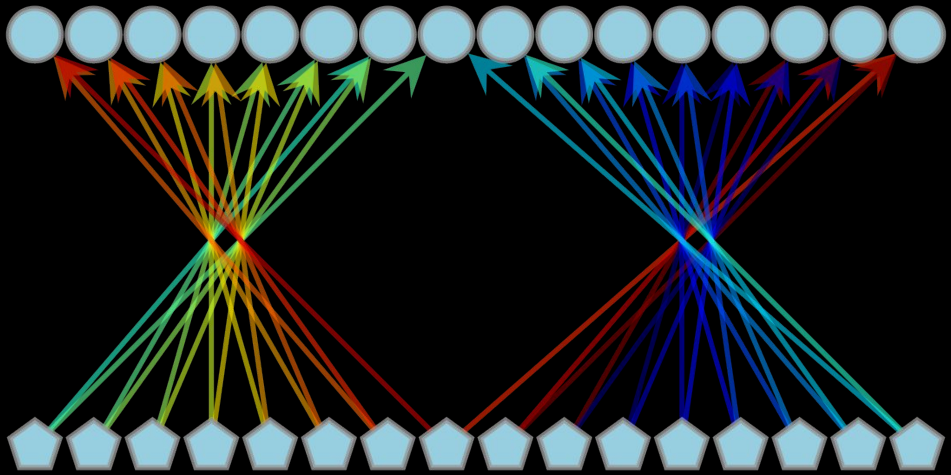
Timeslot 2



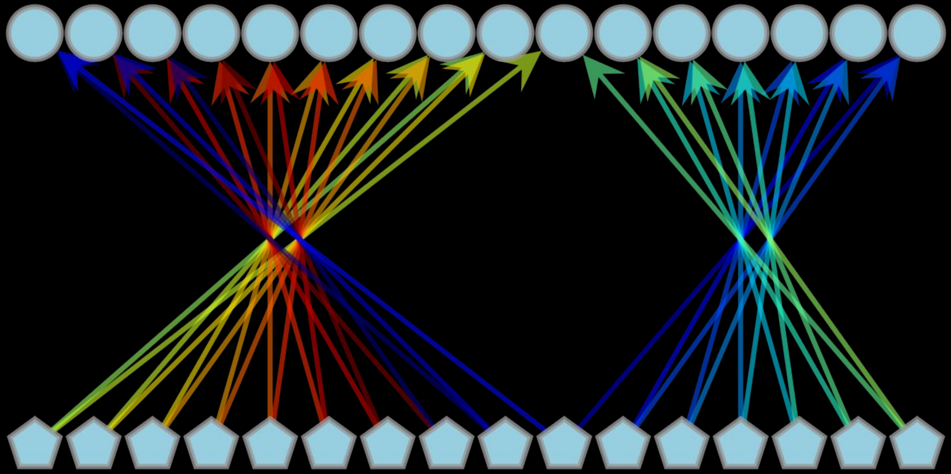
Timeslot 3



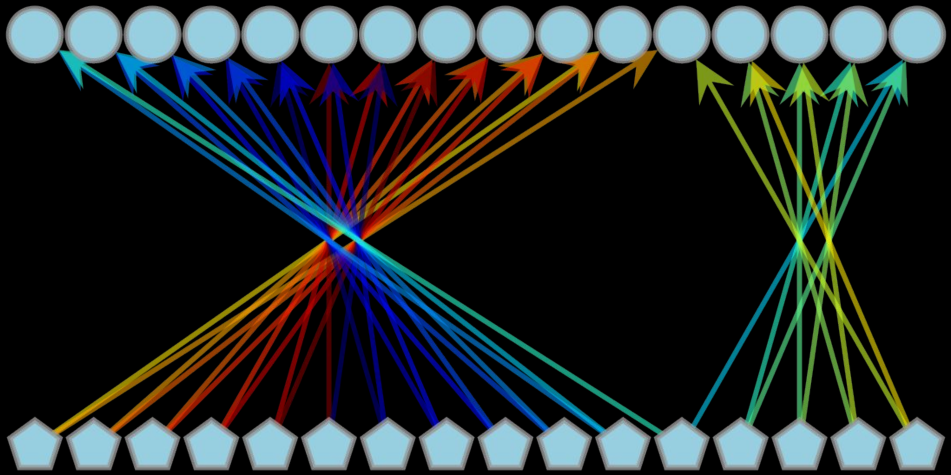
Timeslot 4



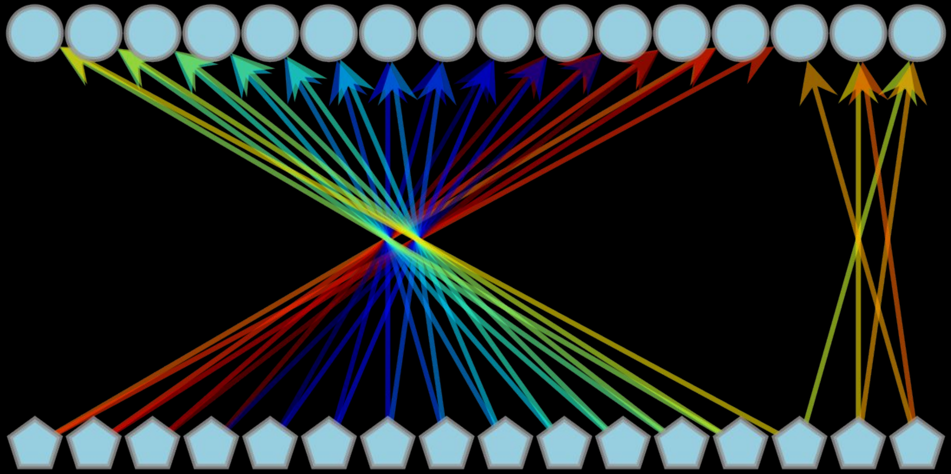
Timeslot 5



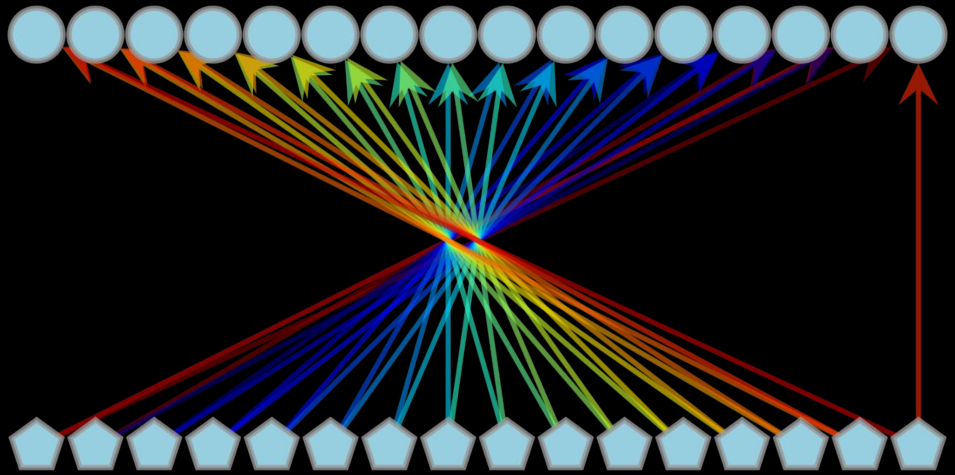
Timeslot 6



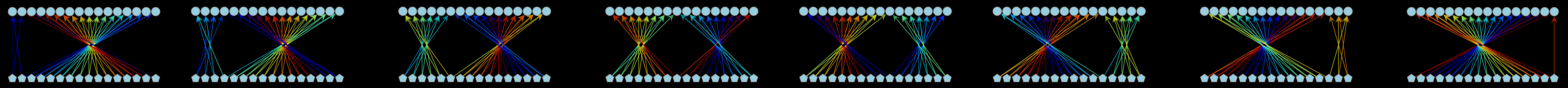
Timeslot 7



Timeslot 8



Periodic Graph



Optimal Topology

Input: Demand Matrix $\mathcal{M}$

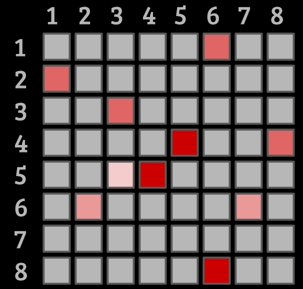
Number of nodes n

Degree bound d' in each timeslot

Output: Periodic graph

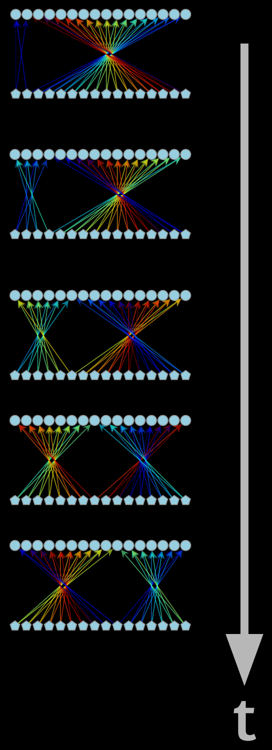
Maximize: Throughput

Throughput



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

Periodic Graph



Demand Matrix

Throughput

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

Throughput of the Periodic Graph

Input: Periodic Graph $\mathcal{G}$

Demand Matrix $\mathcal{M}$

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

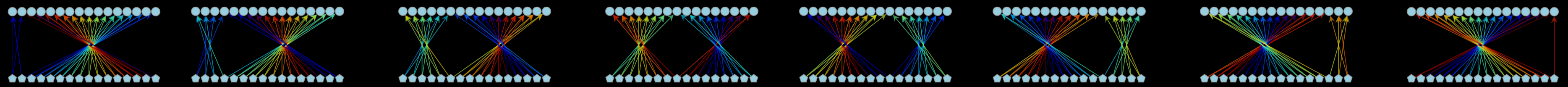
Output: $\theta(\mathcal{M})$ and a feasible *flow**

**subject to conservation, demand and capacity constraints*

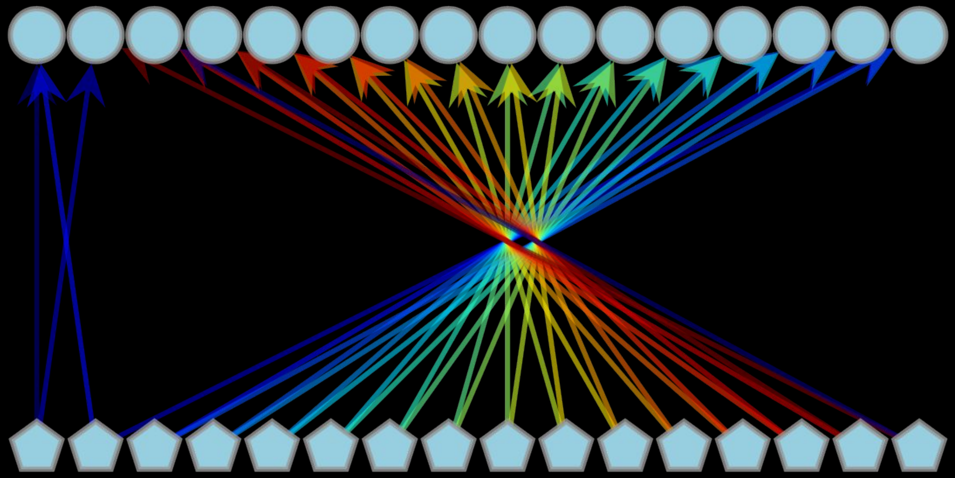
Theorem 1: Periodic Graph ⟷ Static Graph

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

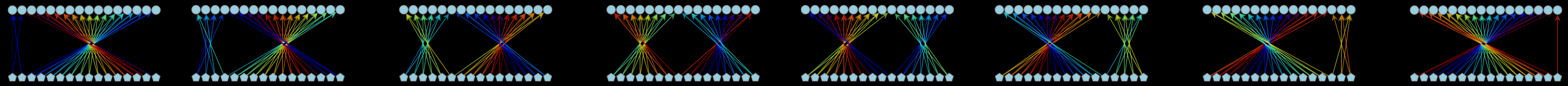
Periodic Graph



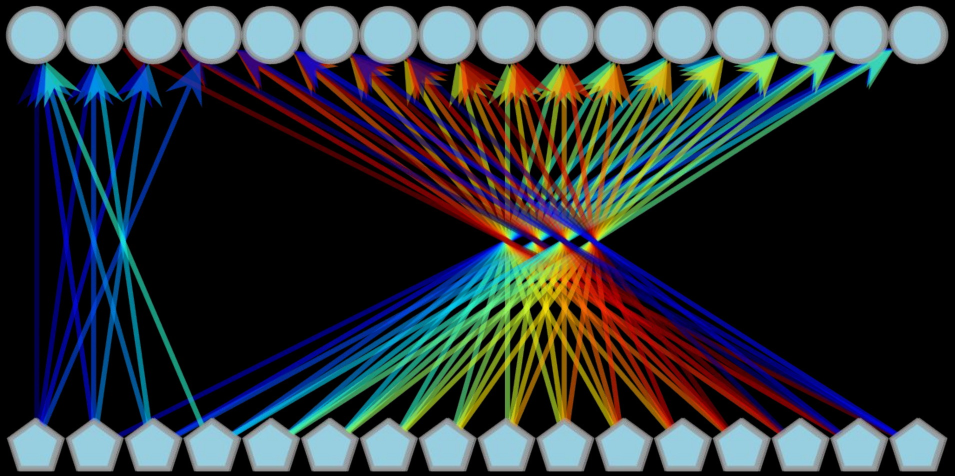
Static Emulated Graph



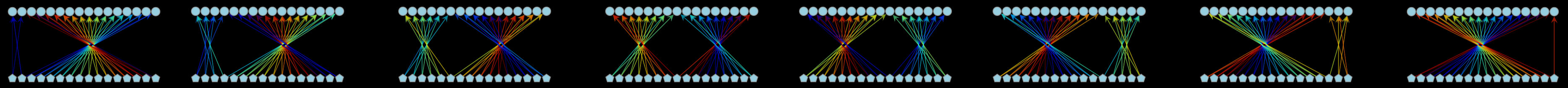
Periodic Graph



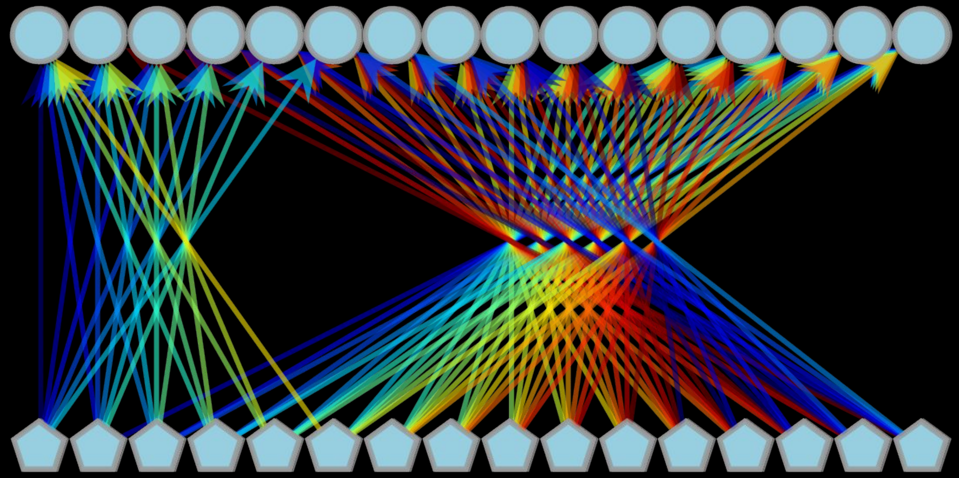
Static Emulated Graph



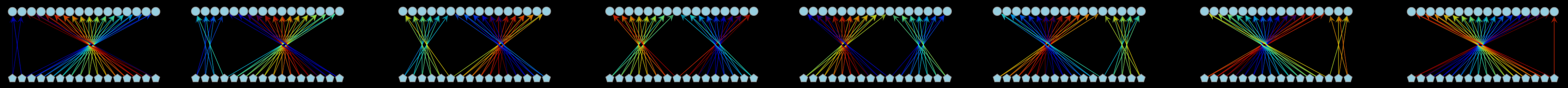
Periodic Graph



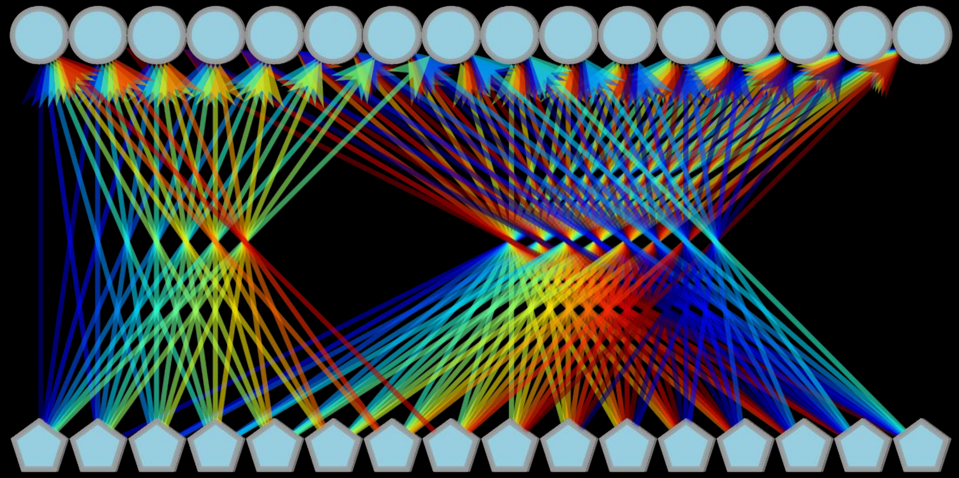
Static Emulated Graph



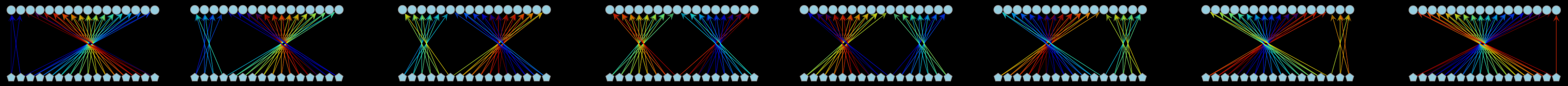
Periodic Graph



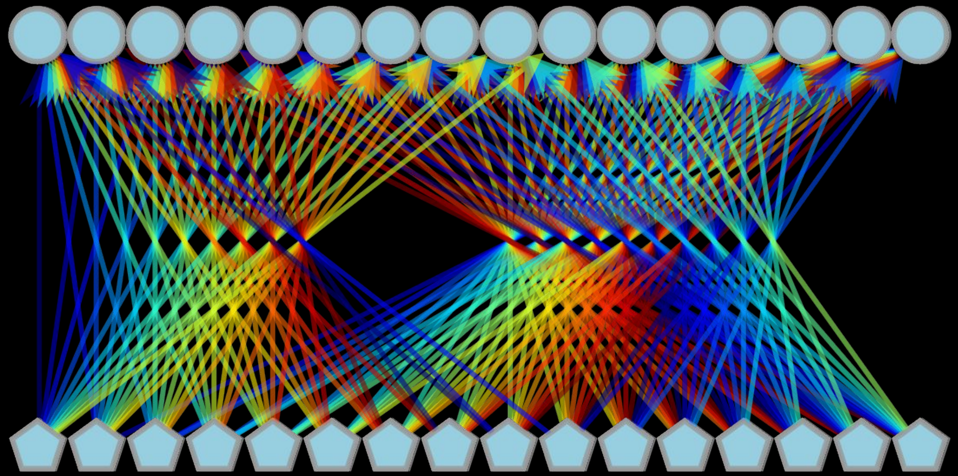
Static Emulated Graph



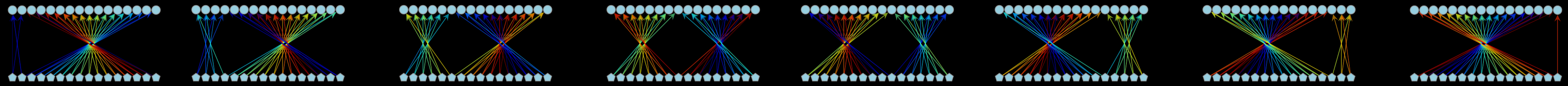
Periodic Graph



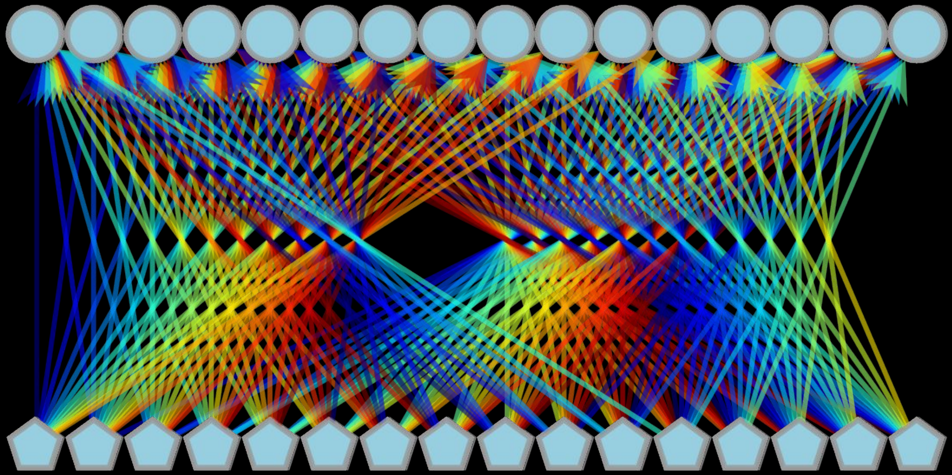
Static Emulated Graph



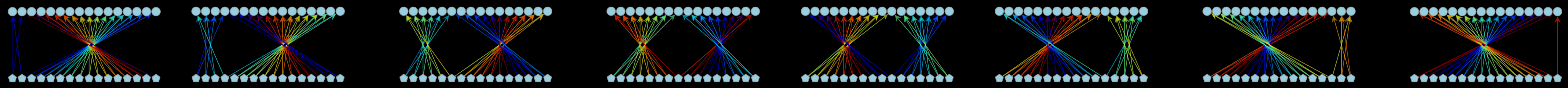
Periodic Graph



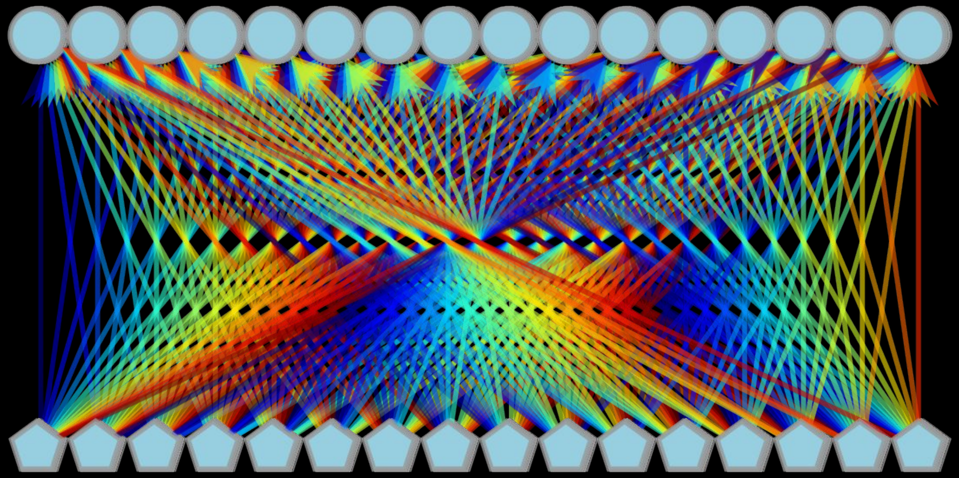
Static Emulated Graph

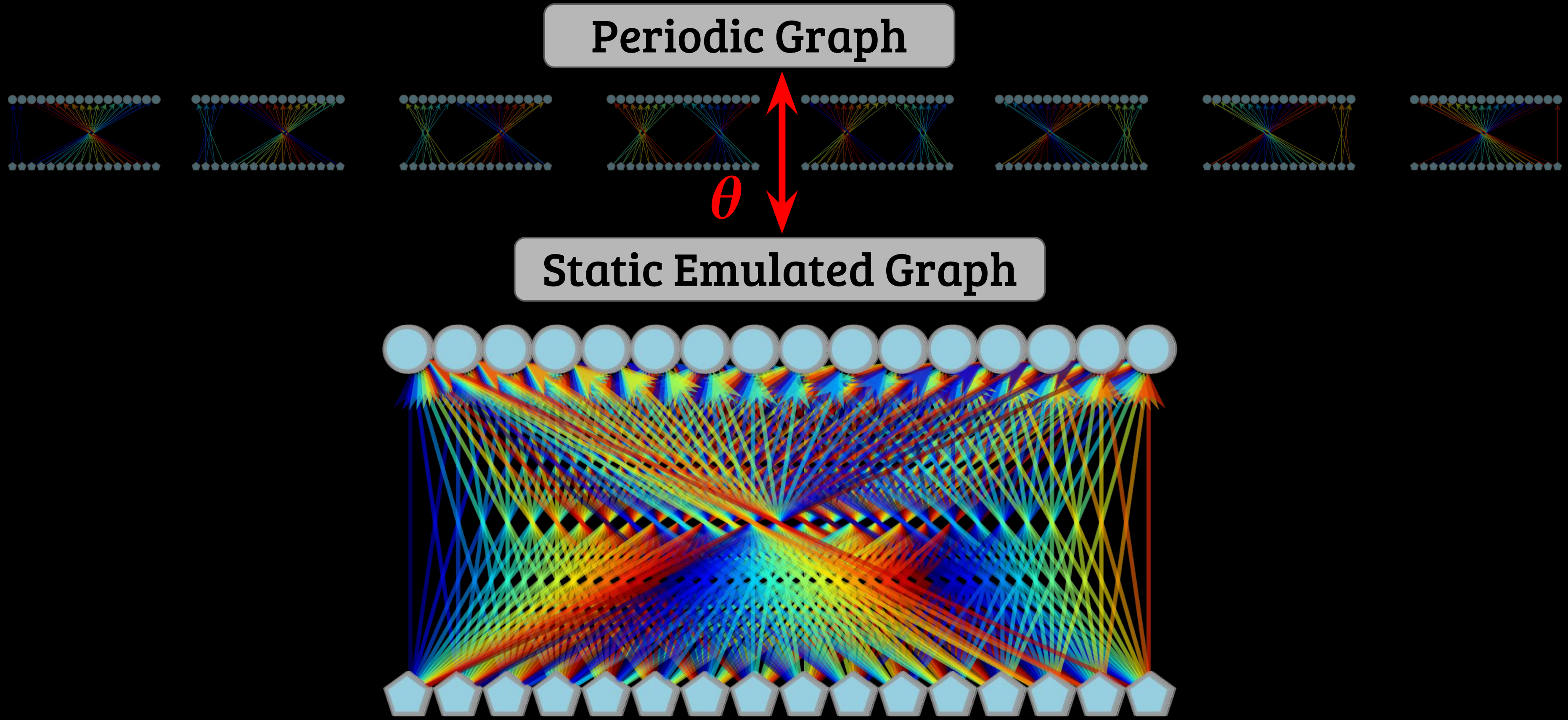


Periodic Graph



Static Emulated Graph




Periodic Graph
θ
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*

Uni-regular

(Static DCNs)

Complete Graph

(Existing RDCNs)

Degree *Increasing* →

Emulated Graph (Topology Over Time)

Theorem 2: Throughput Upper Bound

- Throughput is a function of ***Average Route Length***

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

Theorem 2: Throughput Upper Bound

- Throughput is a function of ***Average Route Length***

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

Theorem 2: Throughput Upper Bound

- Throughput is a function of ***Average Route Length***

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

Theorem 2: Throughput Upper Bound

- Throughput is a function of ***Average Route Length***

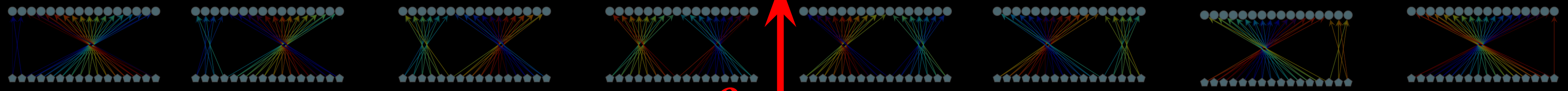
$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{\boxed{M} \ \mathrm{ARL}(\mathcal{M}, F)}$$

Theorem 2: Throughput Upper Bound

- Throughput is a function of ***Average Route Length***

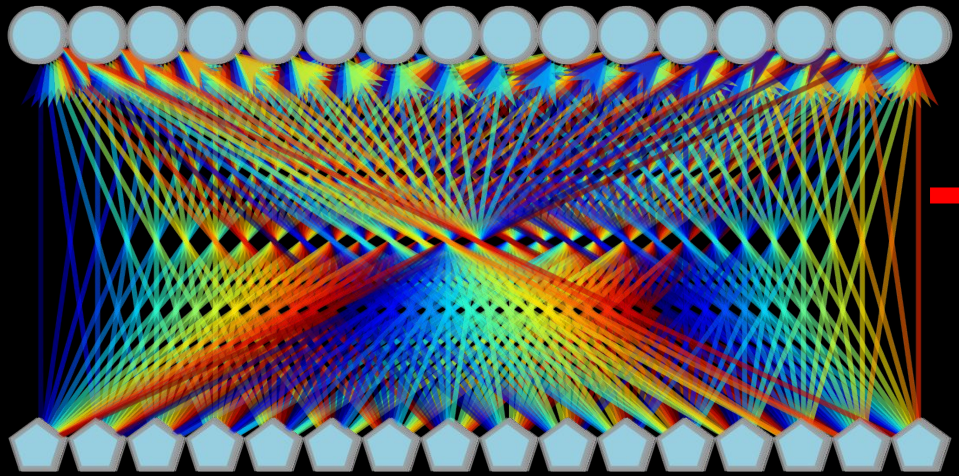
$$\theta(\mathcal{M}, F) \leq \frac{\hat{C}}{M \cdot \boxed{\mathrm{ARL}(\mathcal{M}, F)}}$$

Periodic Graph



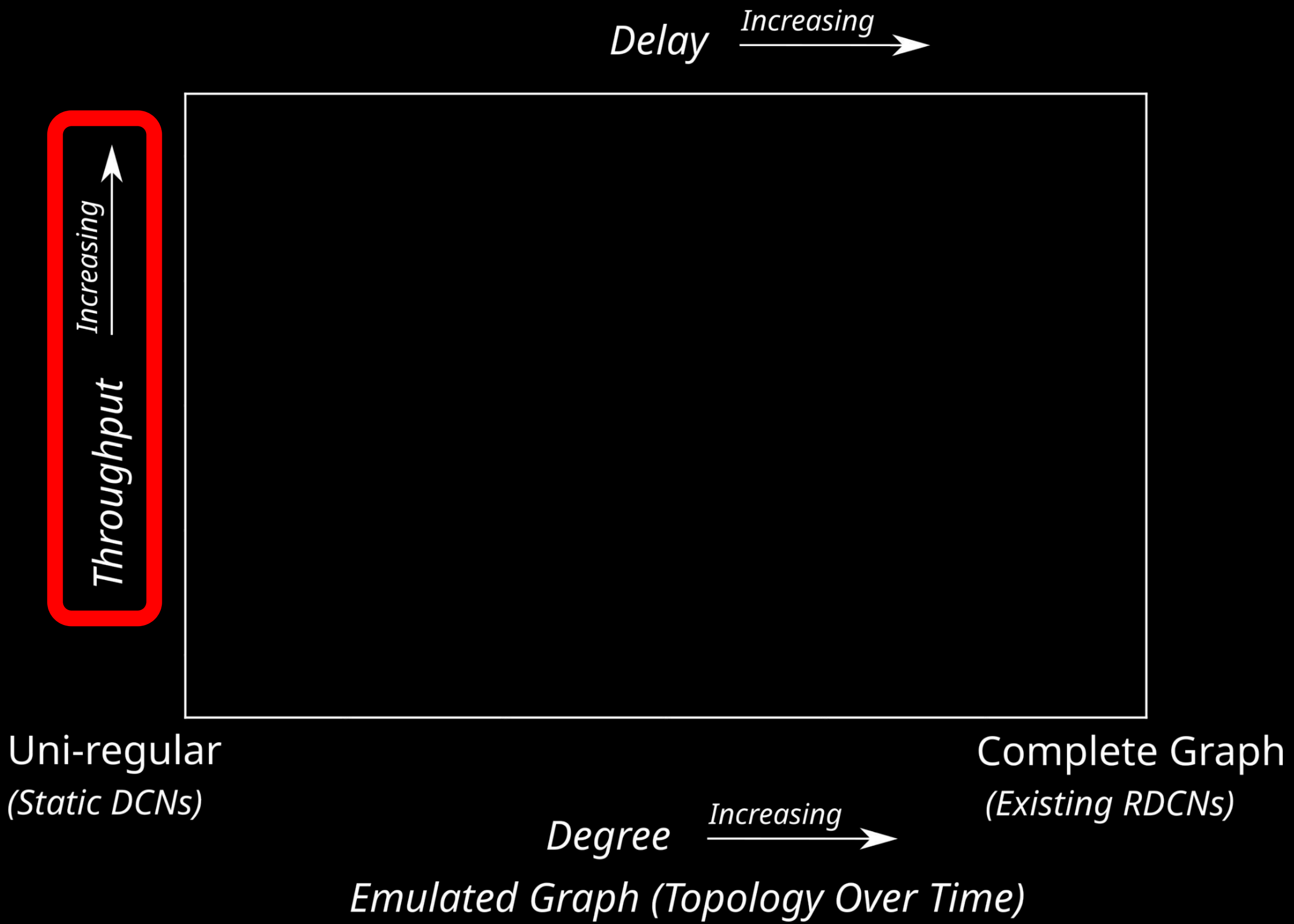
θ

Static Emulated Graph



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

$$\frac{\text{Capacity}}{\text{Demand x ARL}}$$


Delay
Increasing
Increasing
Throughput
Uni-regular
(Static DCNs)
Complete Graph
(Existing RDCNs)
Degree
Increasing
Emulated Graph (Topology Over Time)

Theorem 3: Delay

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

$$L_{max} \geq \mathrm{ARD}(\mathcal{M}, \mathcal{F}) = \mathrm{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta$$

$$\geq \Omega\left(\frac{d \cdot \Delta}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})}\right)$$

Theorem 3: Delay

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

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

Theorem 3: Delay

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

$$L_{max} \geq \mathrm{ARD}(\mathcal{M}, \mathcal{F}) = \mathrm{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta$$

$$\geq \Omega\left(\frac{\boxed{d \cdot} \Delta}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})}\right)$$

Theorem 3: Delay

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

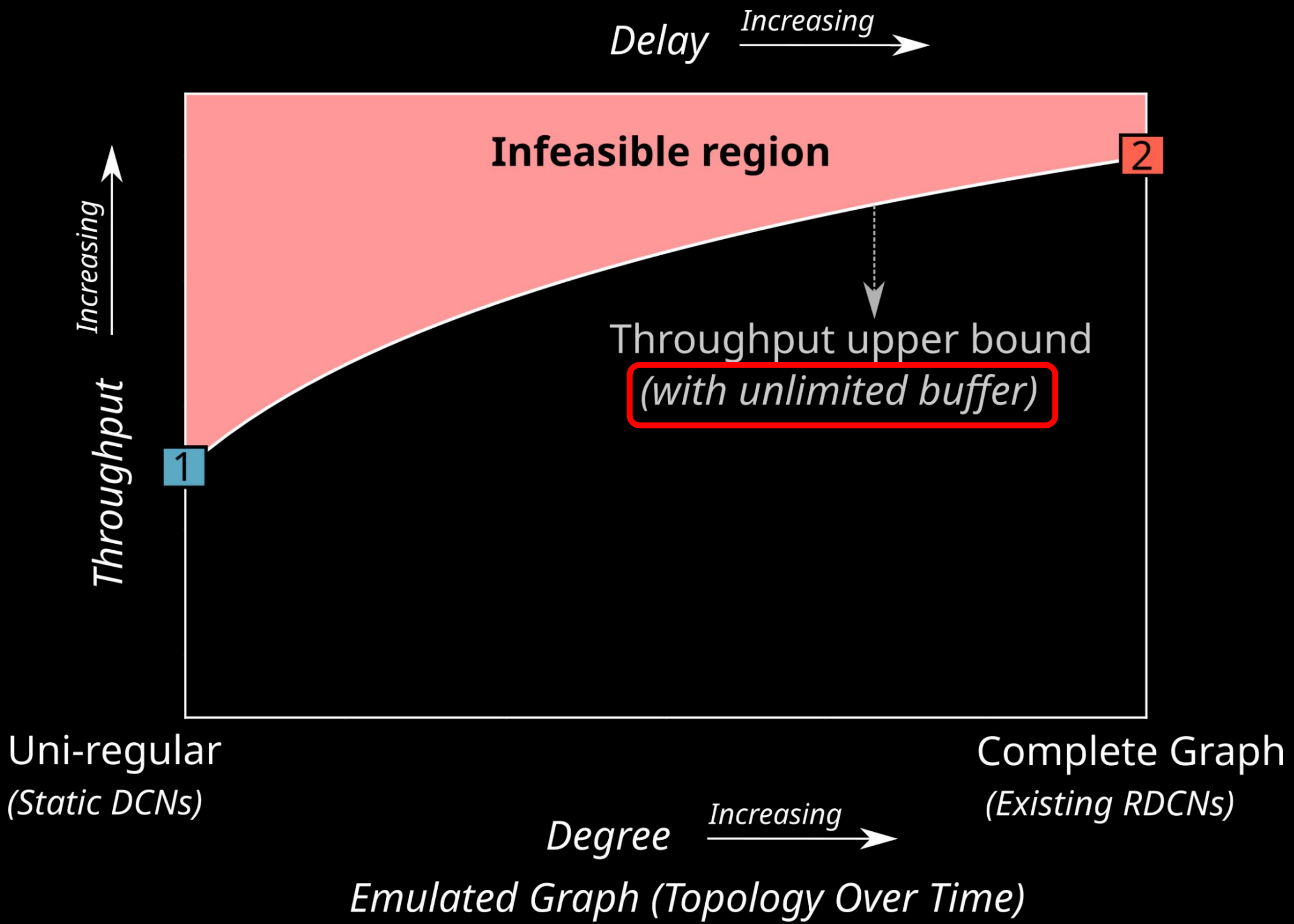
$$L_{max} \geq \mathrm{ARD}(\mathcal{M}, \mathcal{F}) = \mathrm{ARL}(\mathcal{M}, \mathcal{F}) \cdot \Gamma \cdot \Delta$$

$$\geq \Omega\left(\frac{d\,\boxed{\Delta}}{n_u \cdot \theta(\mathcal{M}, \mathcal{F})}\right)$$

Theorem 3: Delay

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

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


Delay
Increasing
Infeasible region
2
Increasing
Throughput
1
Throughput upper bound
(with unlimited buffer)
Uni-regular
(Static DCNs)
Complete Graph
(Existing RDCNs)
Degree
Increasing
Emulated Graph (Topology Over Time)

Theorem 4: Buffer Requirements

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

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

Theorem 4: Buffer Requirements

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

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

Theorem 4: Buffer Requirements

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

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

Theorem 4: Buffer Requirements

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

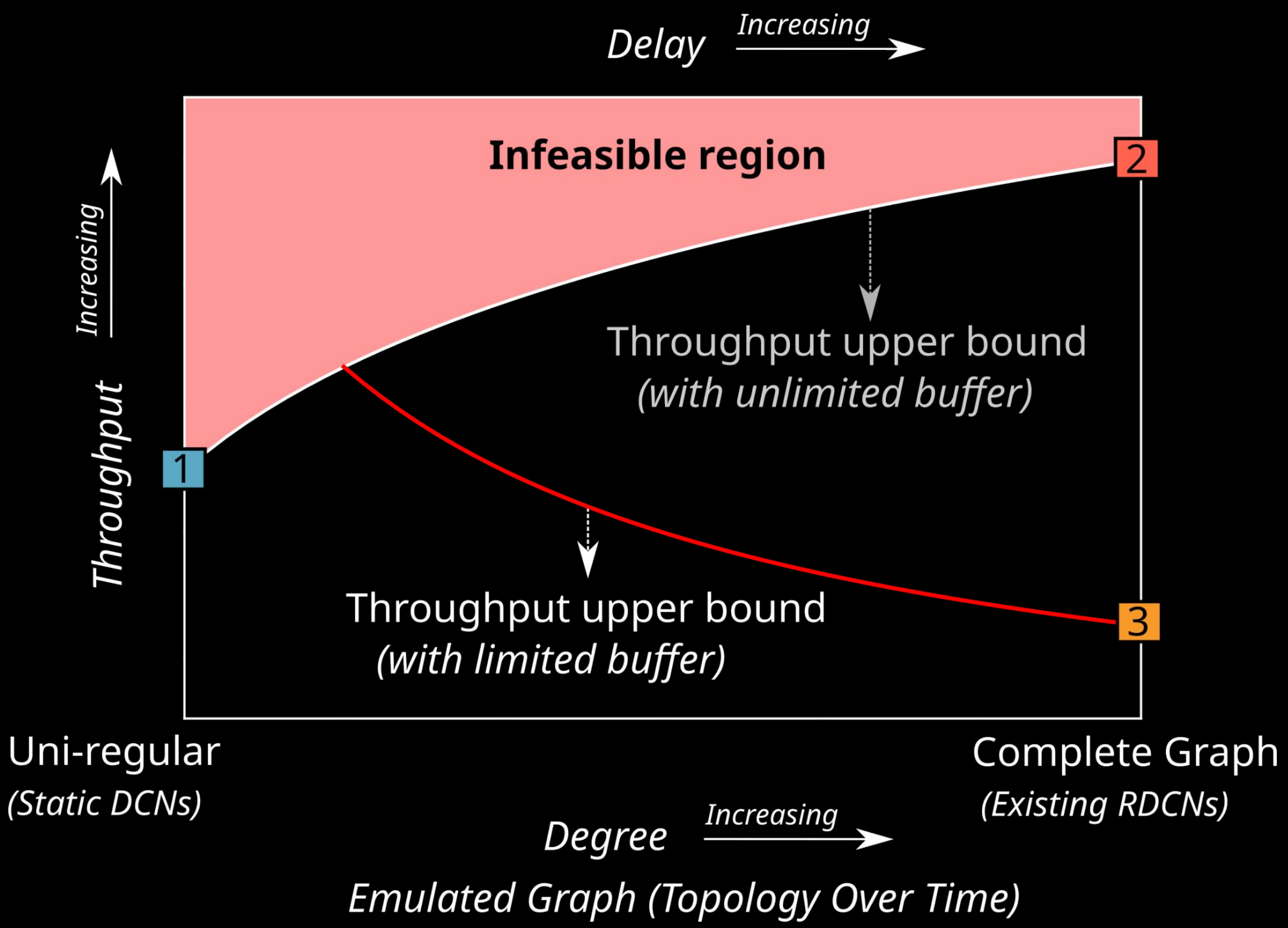
$$\hat{B} \geq (\theta(\mathcal{M}, \mathcal{F}) \cdot M) \cdot \boxed{\mathrm{ARD}(\mathcal{M}, \mathcal{F})}$$

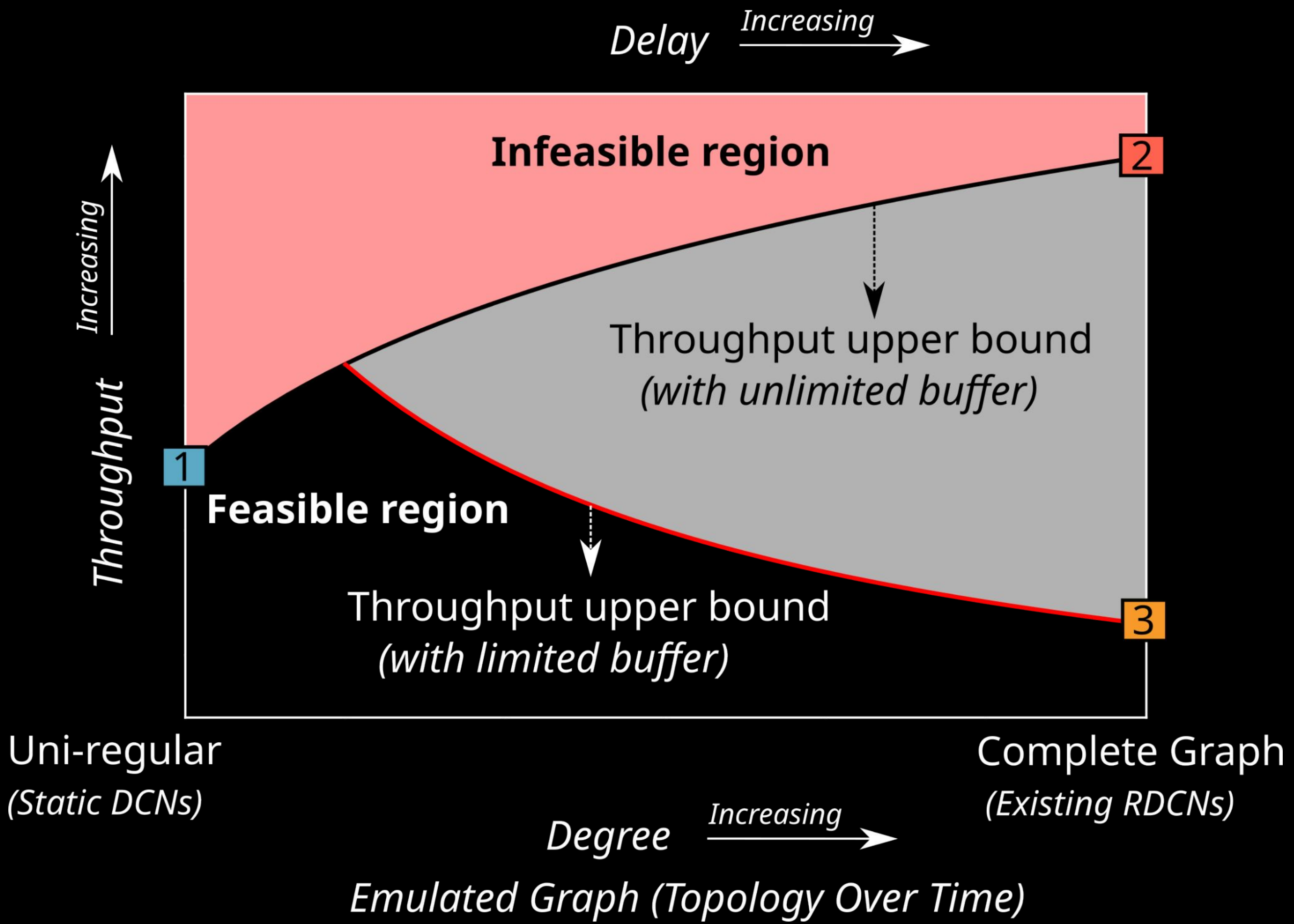
Theorem 4: Buffer Requirements

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

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

Buffer > Bandwidth x Delay


Delay
Increasing
Infeasible region
2
Increasing
Throughput
Throughput upper bound
(with unlimited buffer)
1
Throughput upper bound
(with limited buffer)
3
Uni-regular
(Static DCNs)
Complete Graph
(Existing RDCNs)
Degree
Increasing
Emulated Graph (Topology Over Time)


Delay
Increasing
Infeasible region
2
Increasing
Throughput
Throughput upper bound
(with unlimited buffer)
1
Feasible region
Throughput upper bound
(with limited buffer)
3
Uni-regular
(Static DCNs)
Complete Graph
(Existing RDCNs)
Degree
Increasing
Emulated Graph (Topology Over Time)

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 → Periodic graph

Objective: Maximize the worst-case throughput

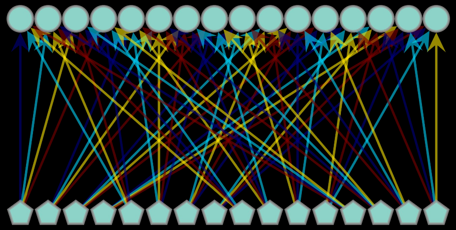
Optimal Oblivious Topology with Buffer Constraints

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

Optimal Oblivious Topology with Buffer Constraints

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

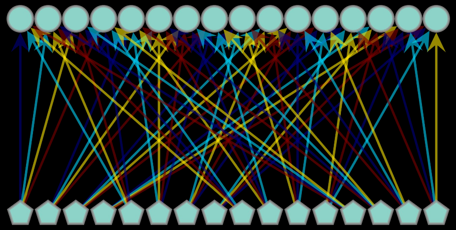
d-regular directed deBruijn graph



Optimal Oblivious Topology with Buffer Constraints

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

d-regular directed deBruijn graph

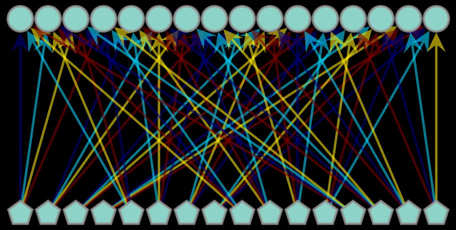


Decomposition to d matchings

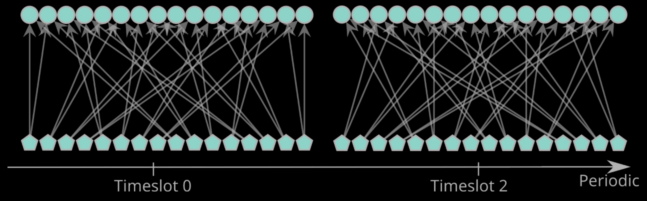
Optimal Oblivious Topology with Buffer Constraints

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

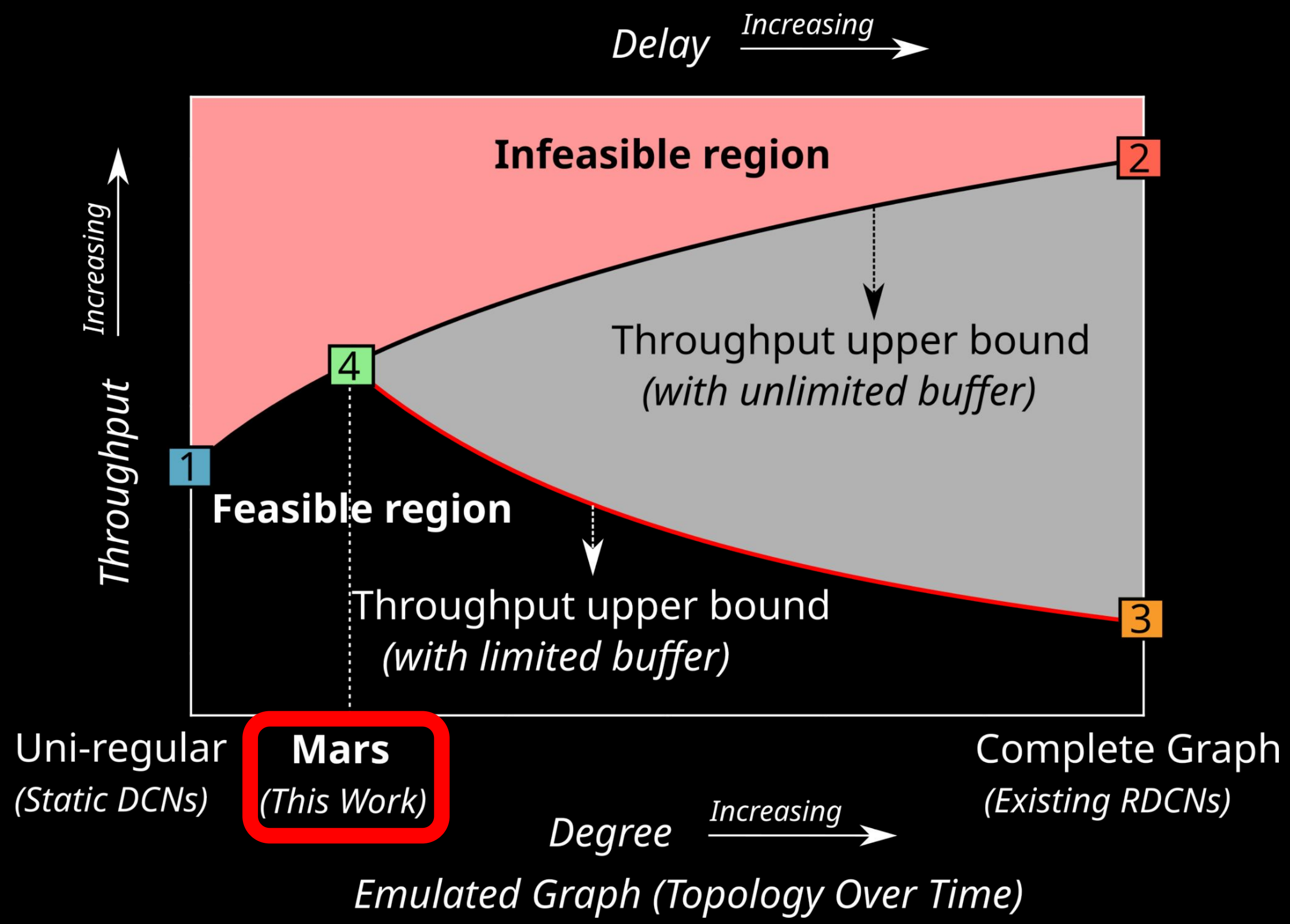
d-regular directed deBruijn graph



Decomposition to d matchings



Periodic graph


Delay
Increasing
Infeasible region
2
Increasing
Throughput
4
Throughput upper bound
(with unlimited buffer)
1
Feasible region
Throughput upper bound
(with limited buffer)
3
Uni-regular
(Static DCNs)
Mars
(This Work)
Complete Graph
(Existing RDCNs)
Degree
Increasing
Emulated Graph (Topology Over Time)

Optimal Oblivious Topology Implications & Future Outlook

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

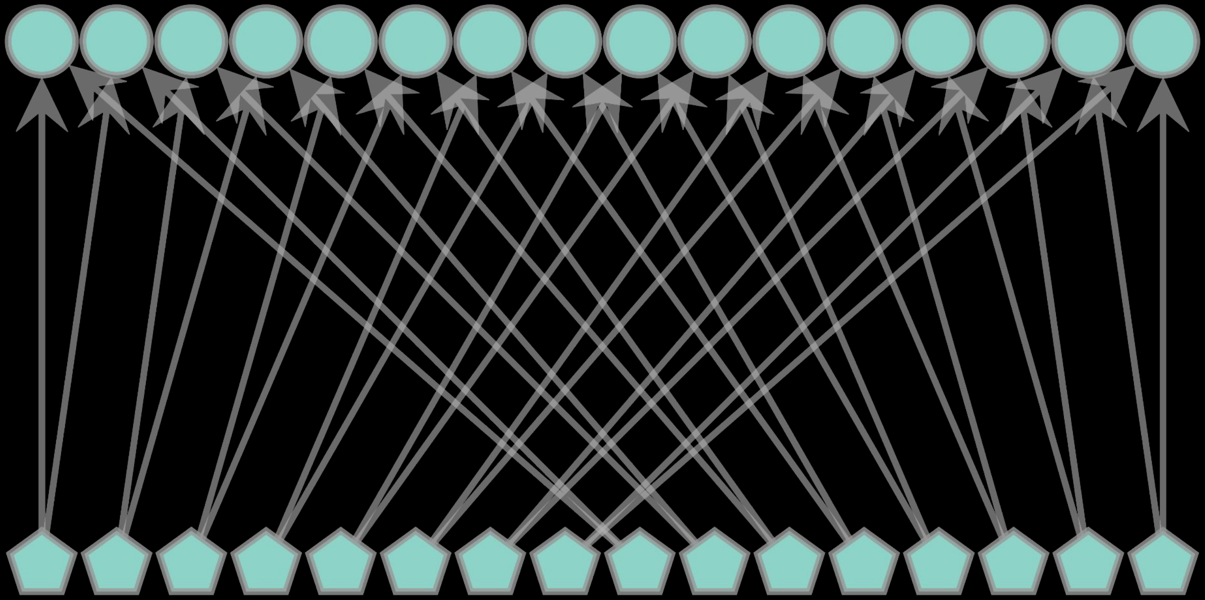
If the reconfiguration technology (Δ) remains same:

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

If Buffer sizes (B) do not keep up:

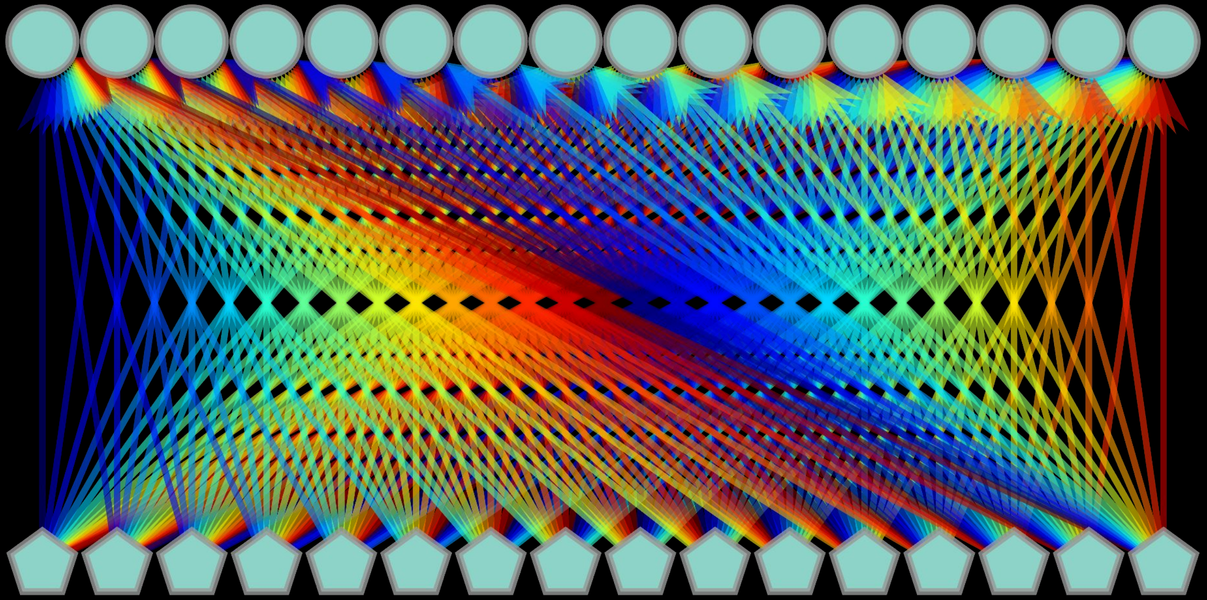
- Increase in capacity (c) $\uparrow$ must be accompanied by decrease in reconfiguration times (Δ) $\downarrow$
- If not, reducing the degree (d) $\downarrow$ of the emulated graph is inevitable to optimize throughput $\rightarrow$ eventually reaching the case of static topologies.

Static DCNs *(uni-regular)*



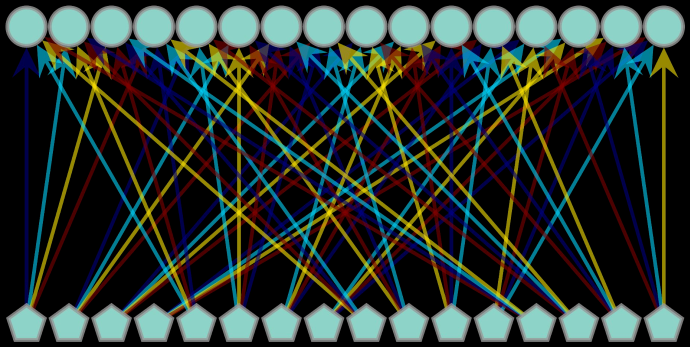
Low throughput but low delay and buffer requirements

Existing RDCN designs *(Emulating a complete graph)*

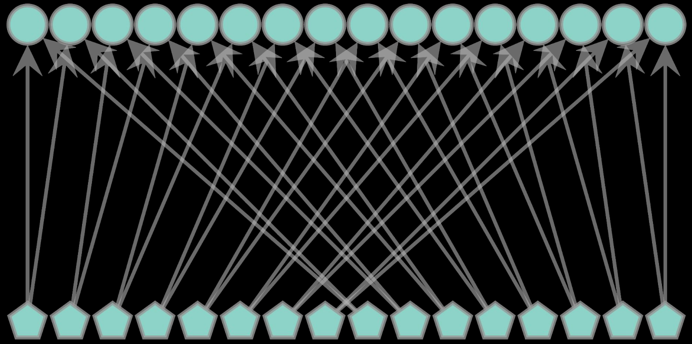


High throughput but high delay and buffer requirements

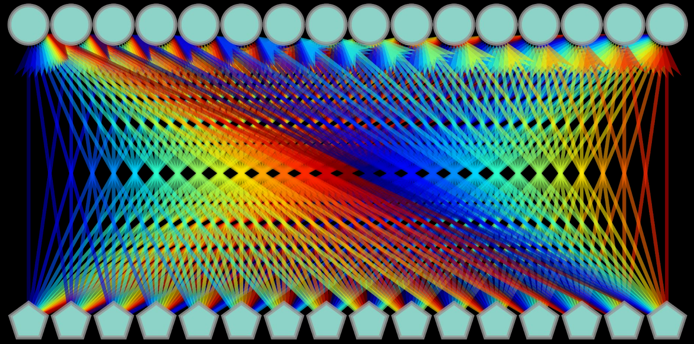
MARS



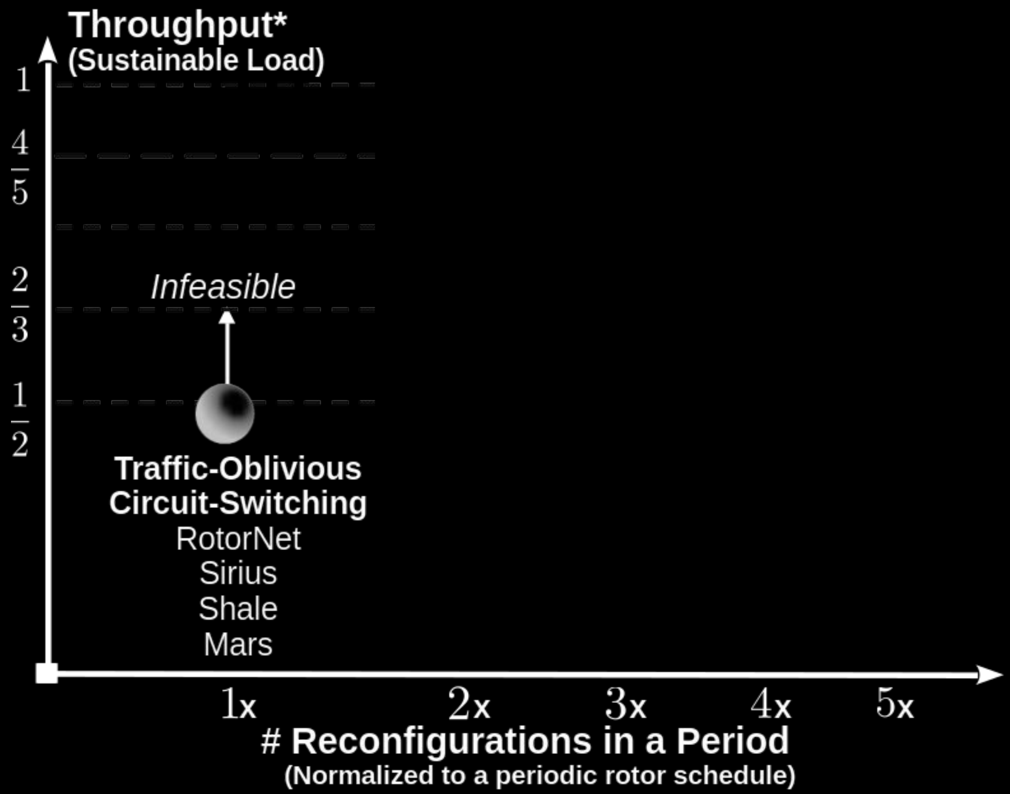
Near-optimal throughput within the available buffer



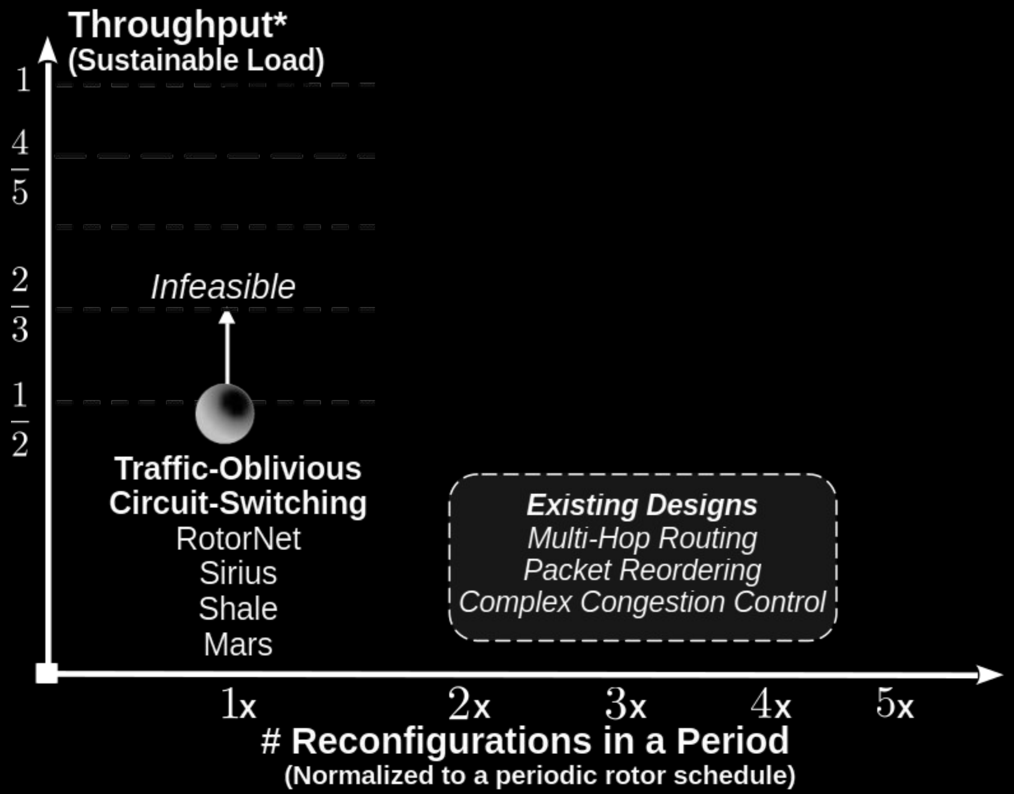
Static DCN: Low Throughput



Existing RDCN: High Delay and buffer

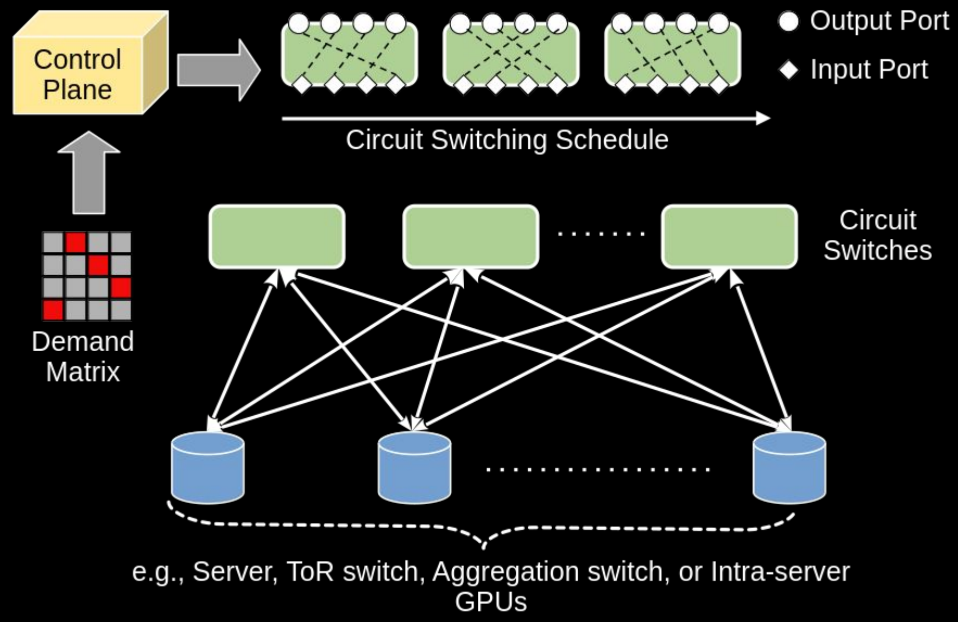


*worst-case throughput

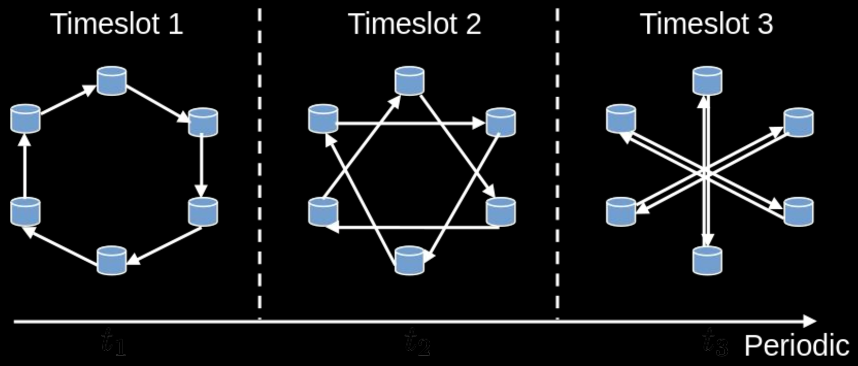


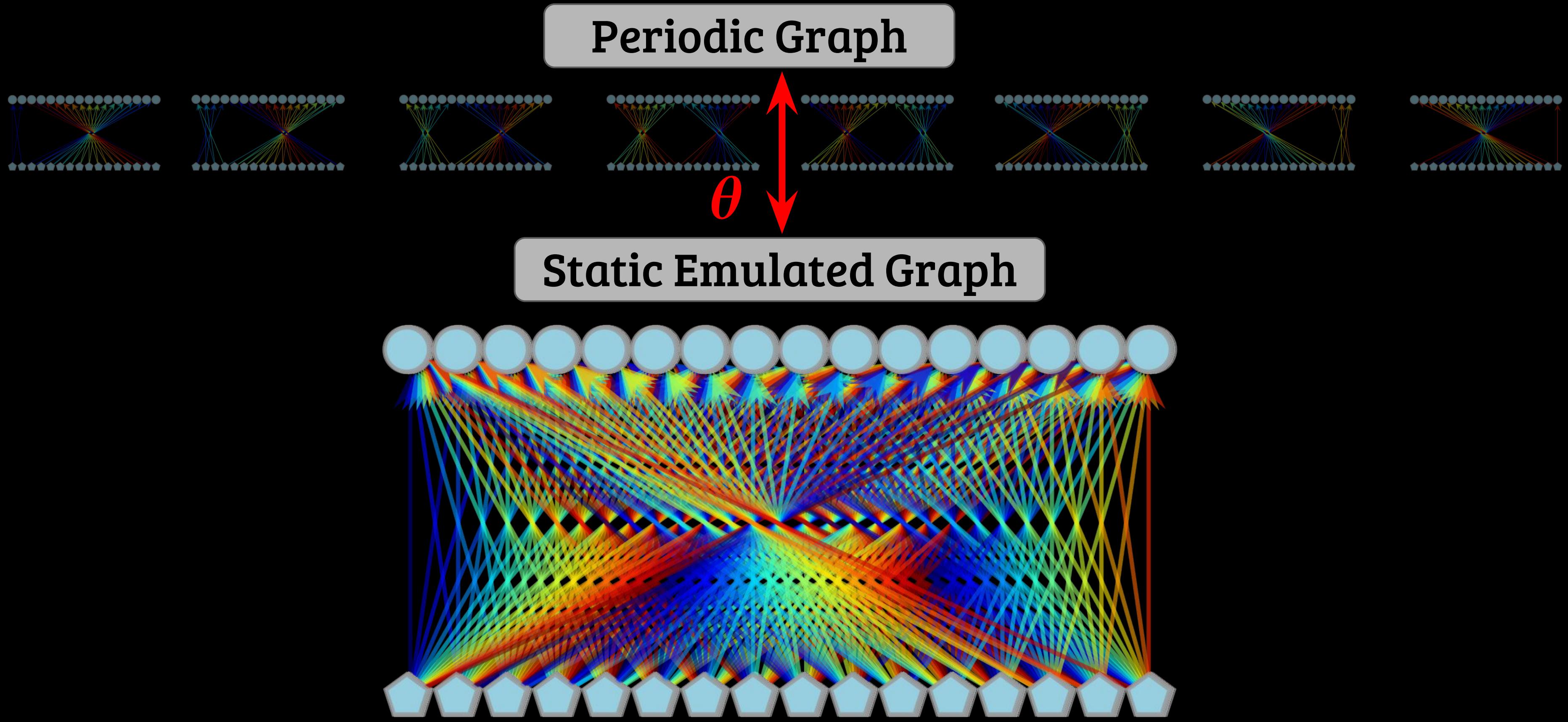
**worst-case throughput*

A Traffic-Aware Approach

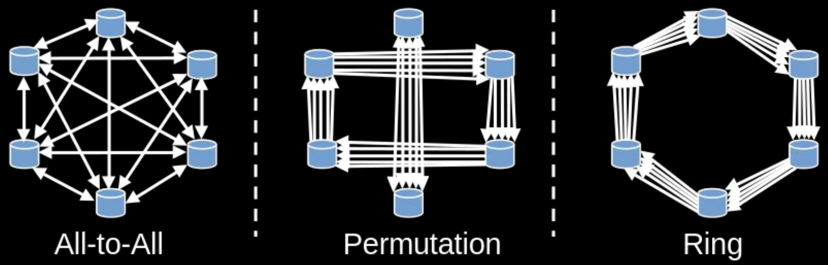


Dynamic Reconfigurable Topology

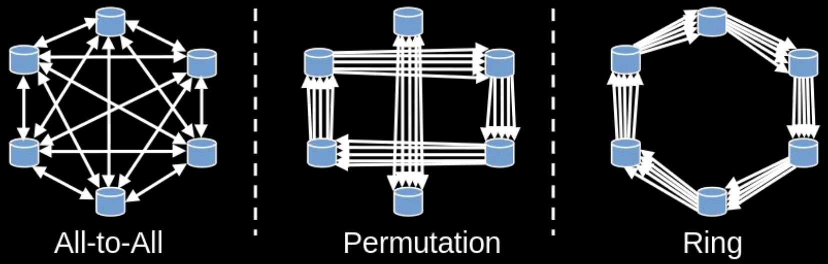



Periodic Graph
θ
Static Emulated Graph

Emulated Topology



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 $\mathcal{M}$

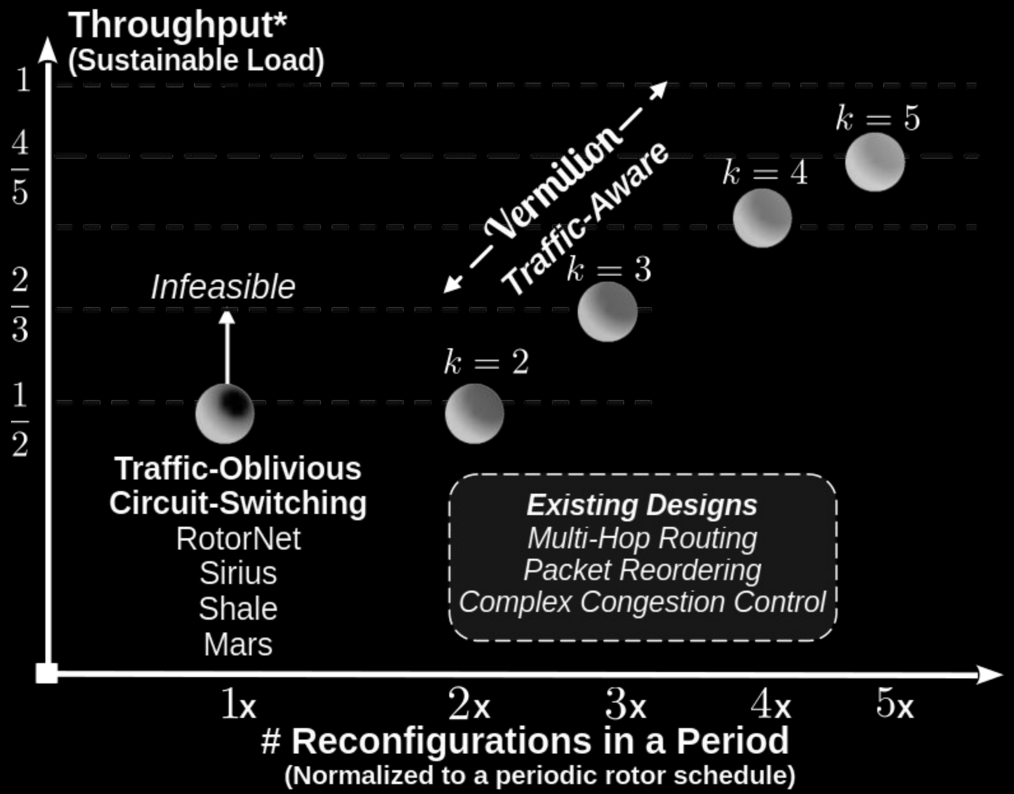
Number of nodes n

Degree bound d' in each timeslot

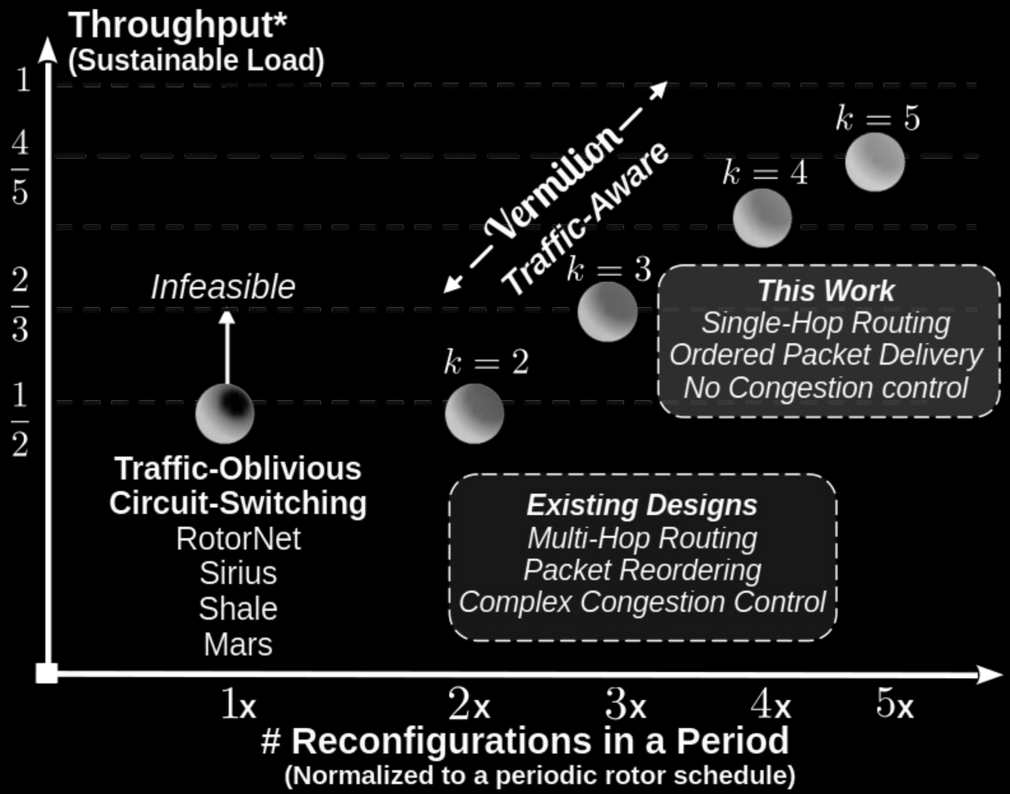
Period bound Γ

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

Maximize: Throughput

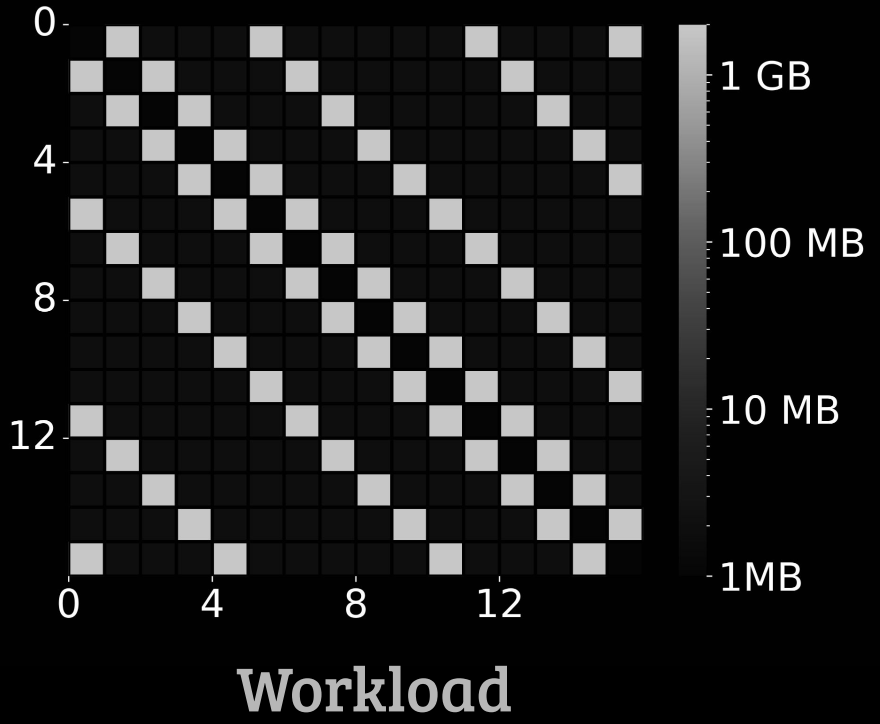


*worst-case throughput

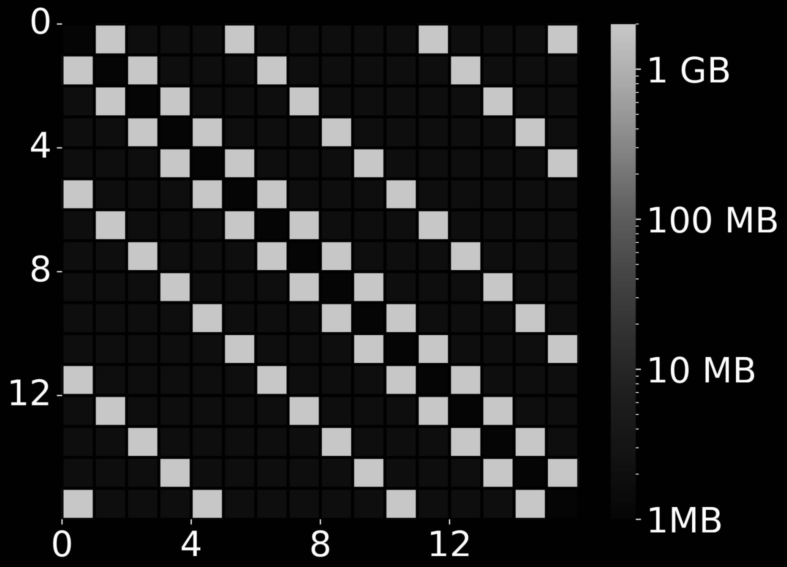


**worst-case throughput*

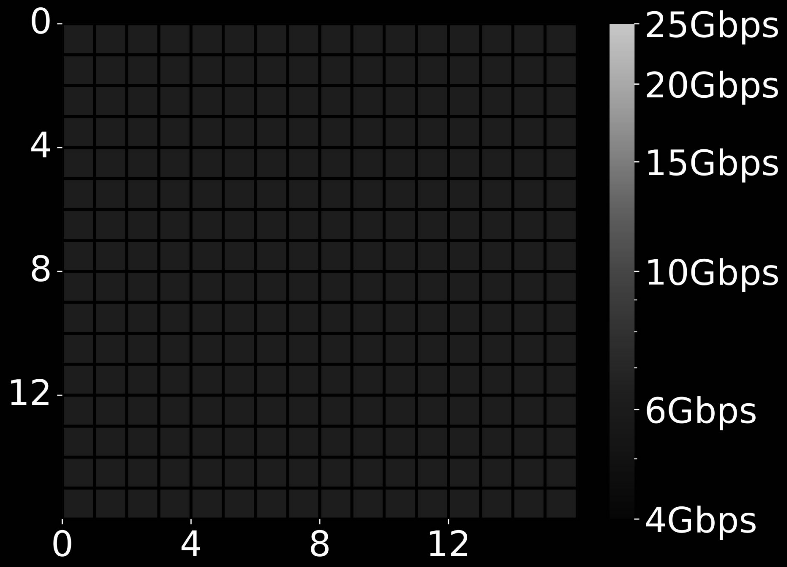
Example: Deep Learning Recommendation Model Workload



Example: Deep Learning Recommendation Model Workload

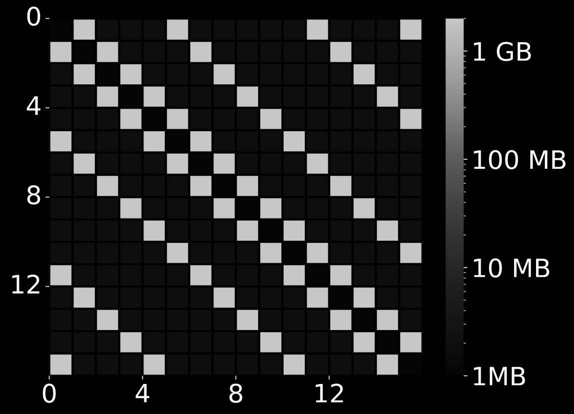


Workload

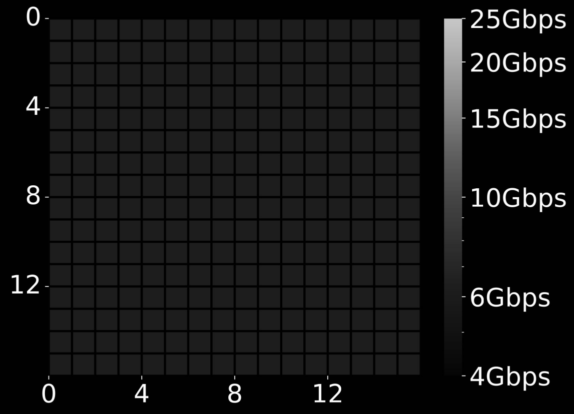


Oblivious Topology

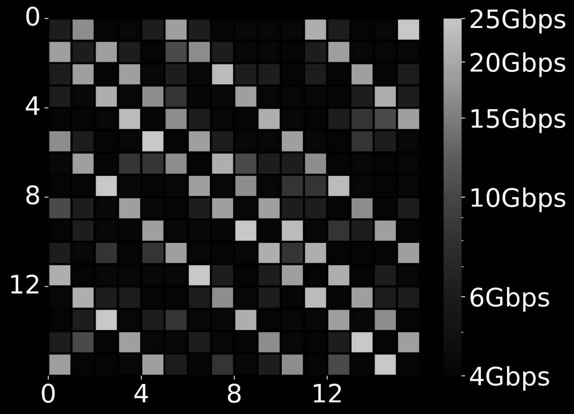
Example: Deep Learning Recommendation Model Workload



Workload

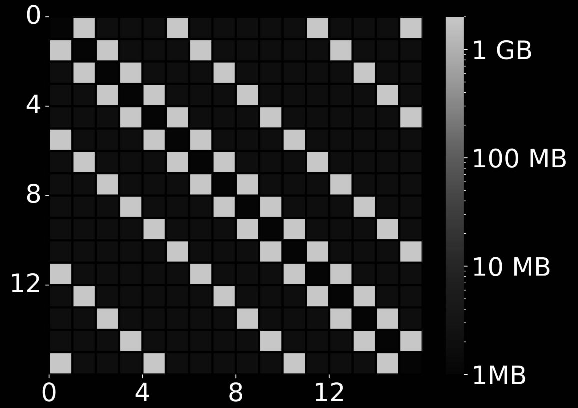


Oblivious

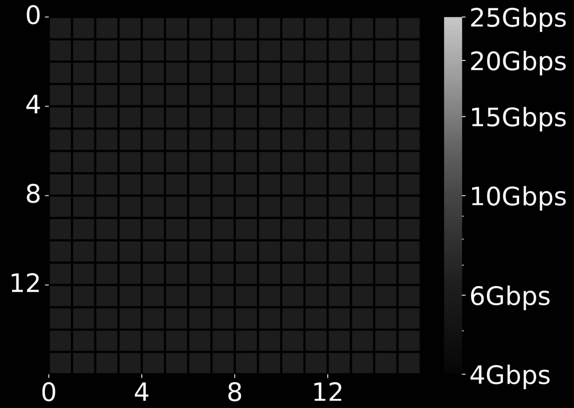


Vermilion

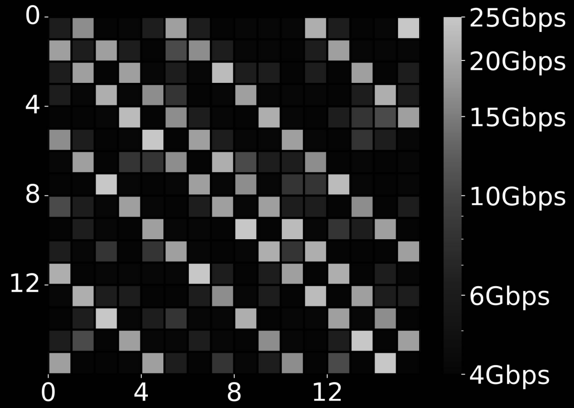
It's a Match!



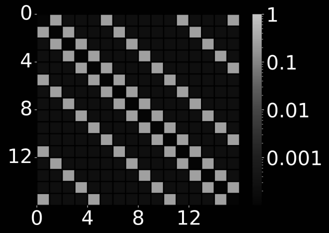
Workload



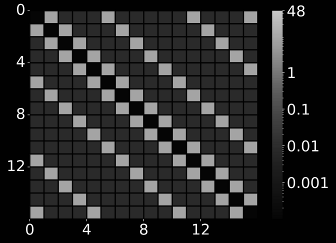
Oblivious



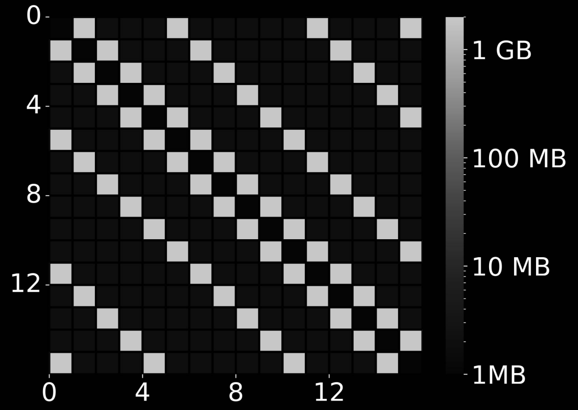
Vermilion



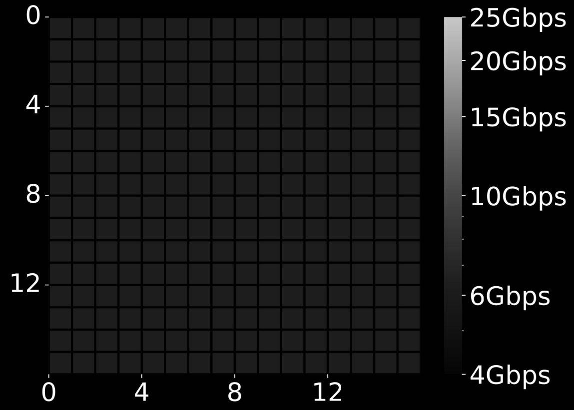
Normalization



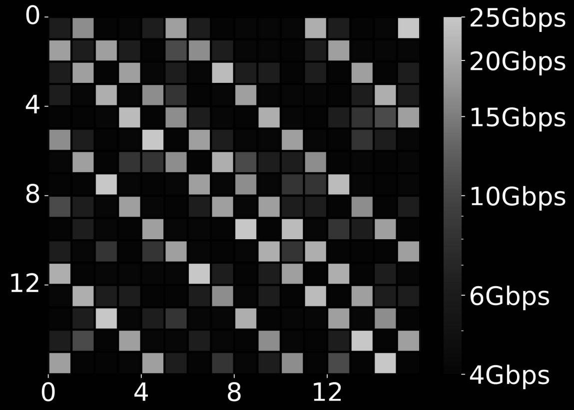
Upscale



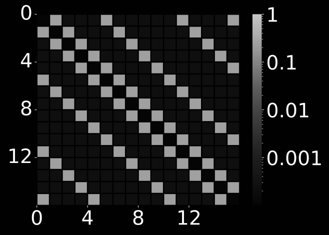
Workload



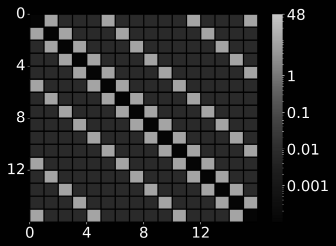
Oblivious



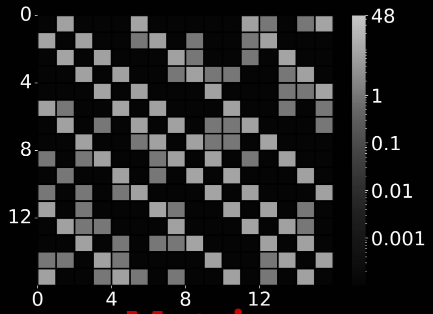
Vermilion



Normalization

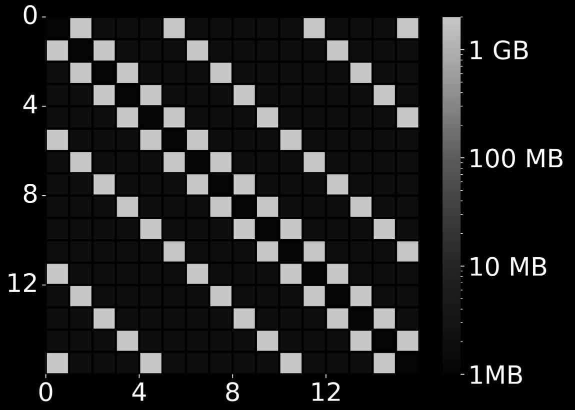


Upscale

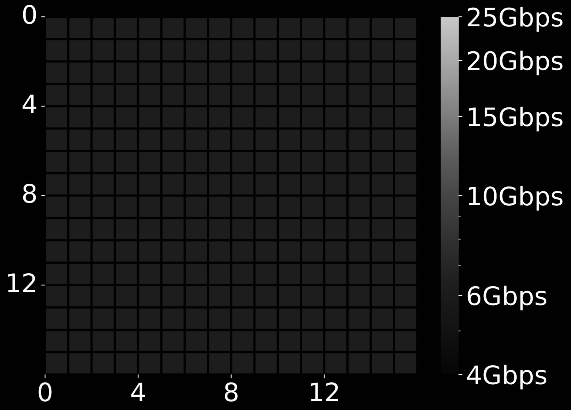


Matrix Rounding

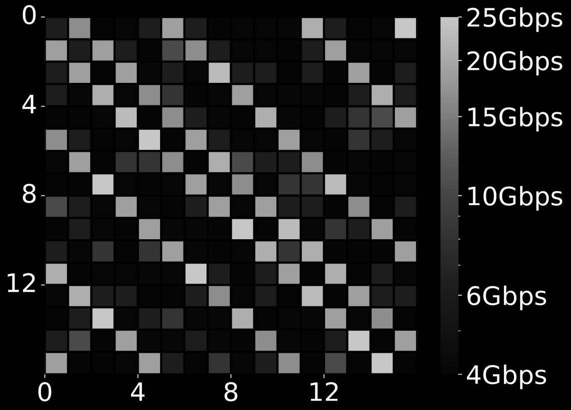
Traffic-aware schedule



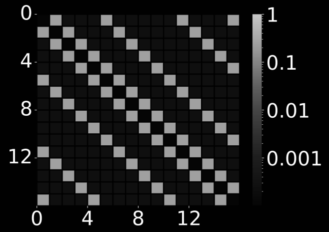
Workload



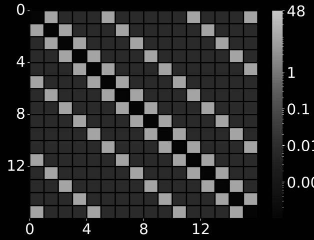
Oblivious



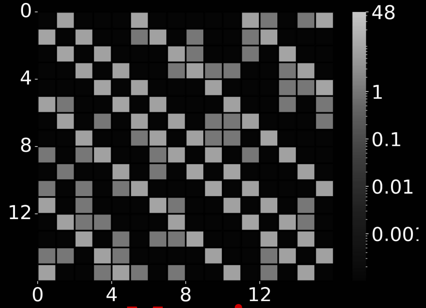
Vermilion



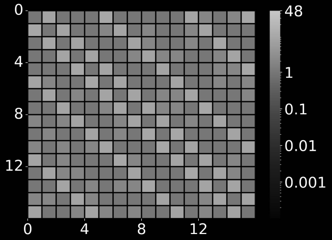
Normalization



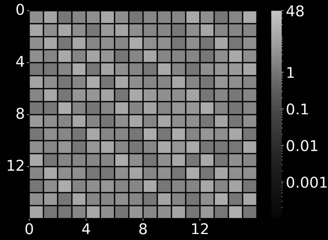
Upscale



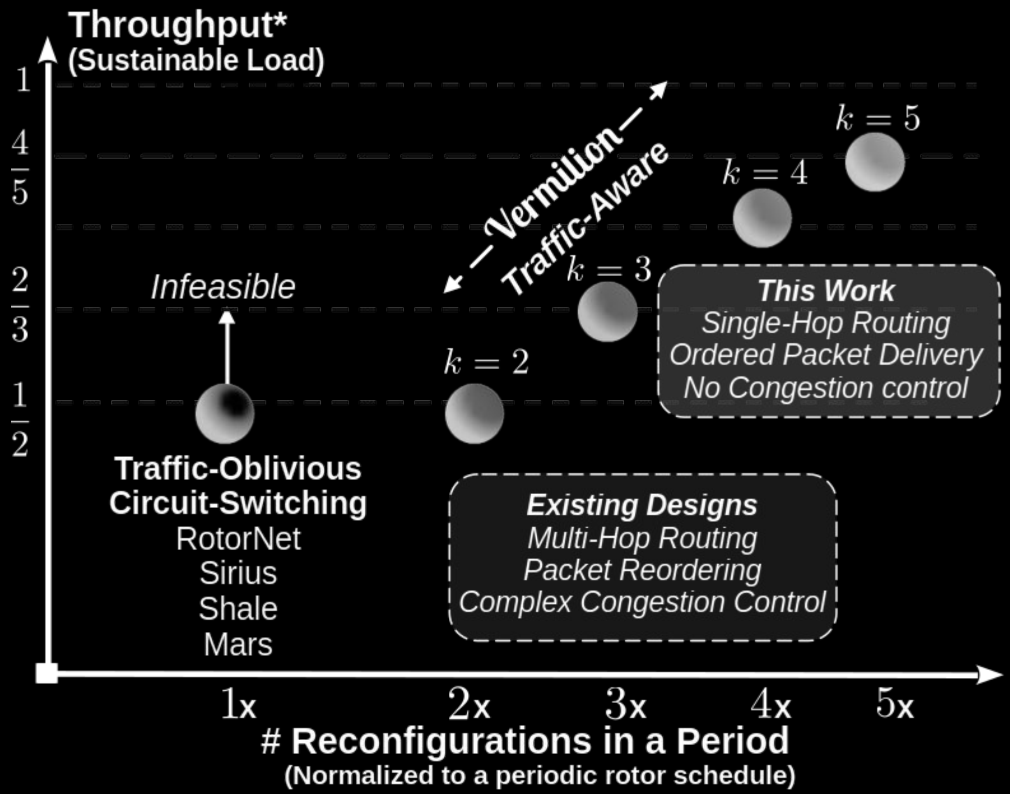
Matrix Rounding
Traffic-aware schedule



+ Oblivious schedule



+ Random configuration



*worst-case throughput

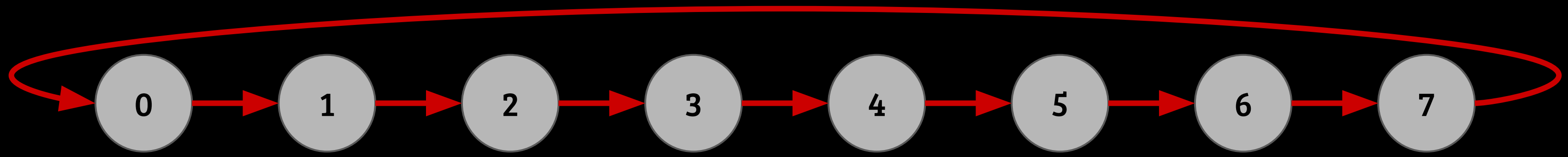
Insights & Reflection on Metrics for Collective Communication

Throughput and Reconfiguration Delay

- Δ = 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: Δ can be small enough irrespective of the reconfiguration delay

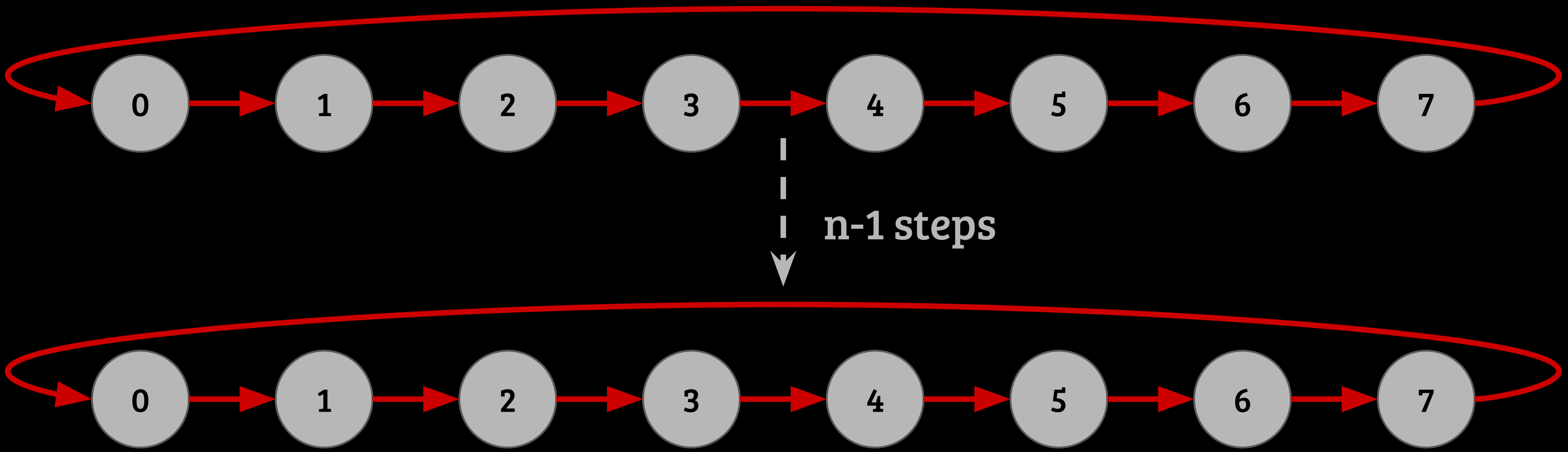
Ring AllReduce

- The communication pattern is a *matching*



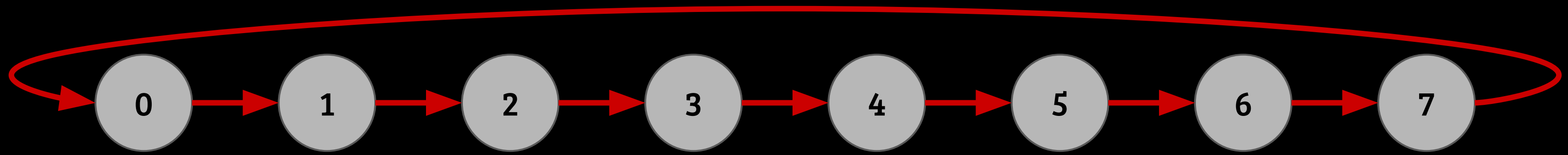
Ring AllReduce

- 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 α,β Cost Model

α : Initialization time for sending out data

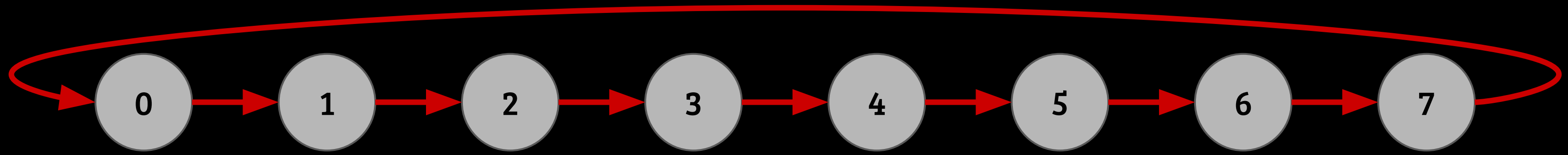
β : Transmission delay for sending one bit at line rate

λ : Congestion factor (number of flows sharing bandwidth)

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

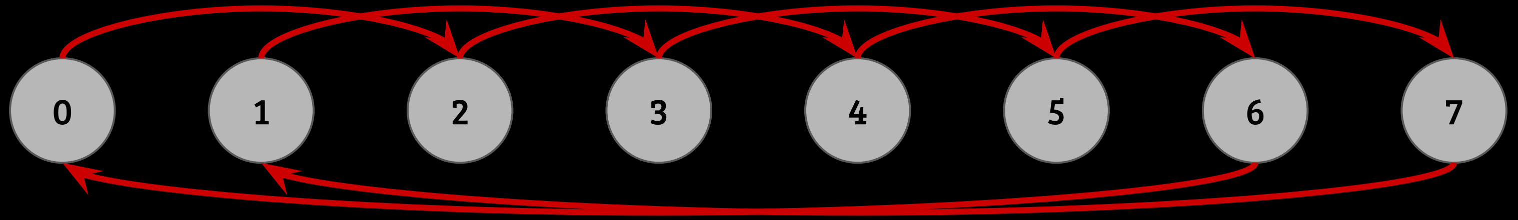
Recursive Doubling

- Step 1: Matching



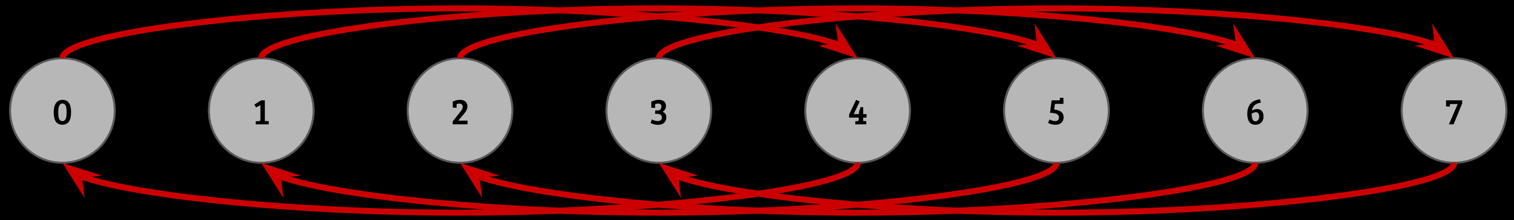
Recursive Doubling

- Step 2: Matching



Recursive Doubling

- Step 3: Matching



Modeling the Completion Time using α,β Cost Model

α : Initialization time for sending out data

β : Transmission delay for sending one bit

λ : Congestion factor (number of flows sharing bandwidth)

□ : Reconfiguration delay

- α + □ + m β : Reconfigure and align topology to communication matching
- α + m β λ : 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