Self-Adjusting Networks

Stefan Schmid

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

(Folklore)

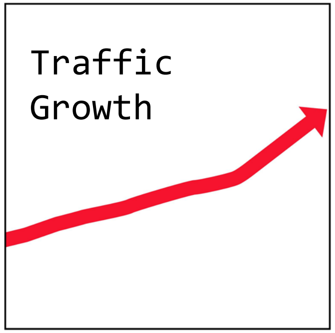
Trend:

Data-Centric Applications

Datacenters ("hyper-scale")

+network

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



Source: Facebook

The Problem:

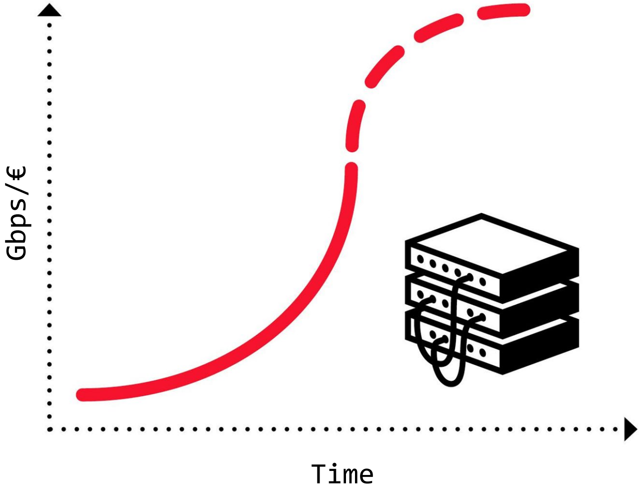
Huge Infrastructure, Inefficient Use

⇢ Network equipment reaching capacity limits

- ⇀ Transistor density rates stalling
- ⇀ “End of **Moore‘s Law** in networking” [1]

⇢ Hence: more equipment, larger networks

⇢ Resource intensive and: **inefficient**



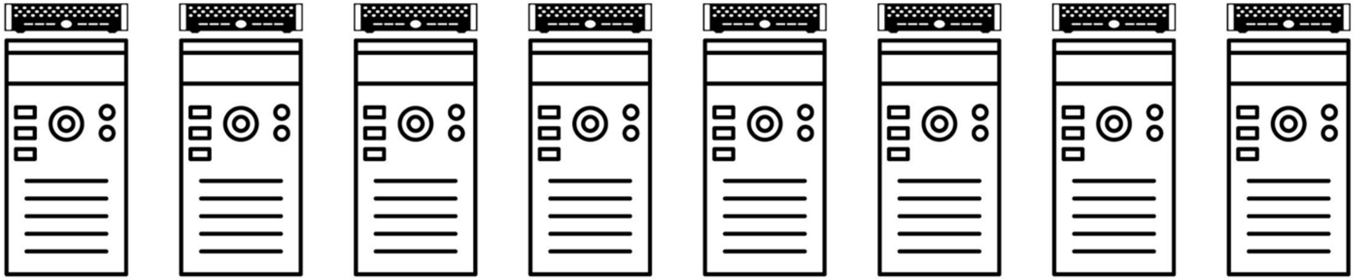
[1] Source: Microsoft, 2019

Annoying for companies, **opportunity** for researchers

Root Cause:

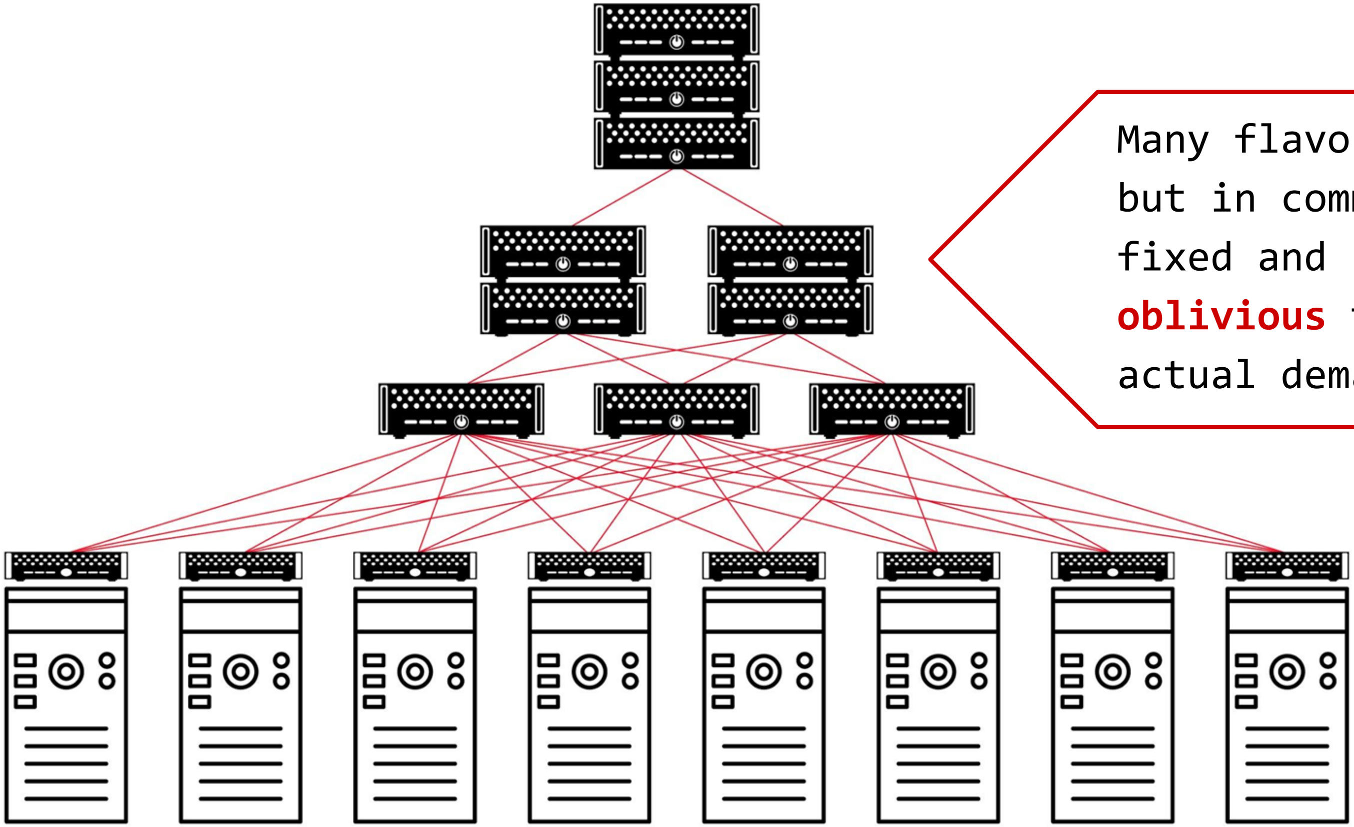
Fixed and Demand-Oblivious Topology

How to interconnect?



Root Cause:

Fixed and Demand-Oblivious Topology

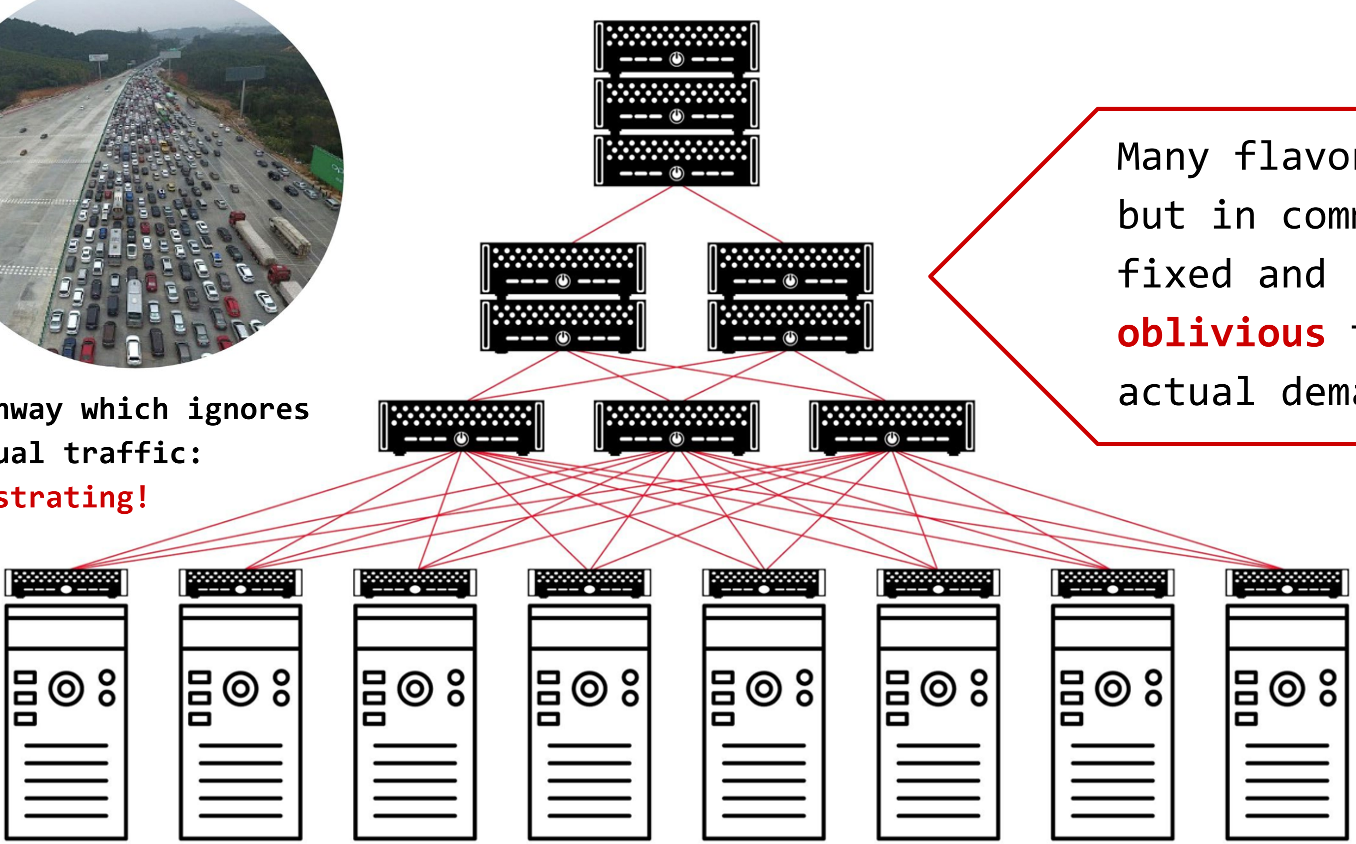


Many flavors,
but in common:
fixed and
oblivious to
actual demand.

Root Cause:

Fixed and Demand-Oblivious Topology

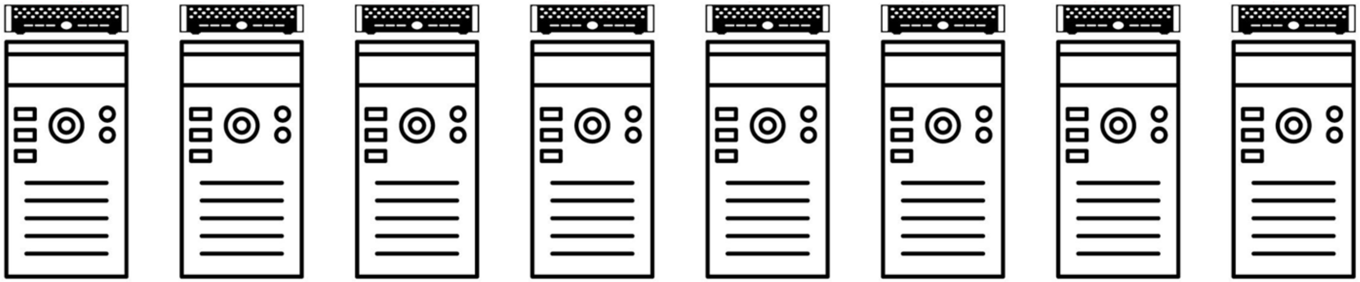
Highway which ignores actual traffic: frustrating!



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

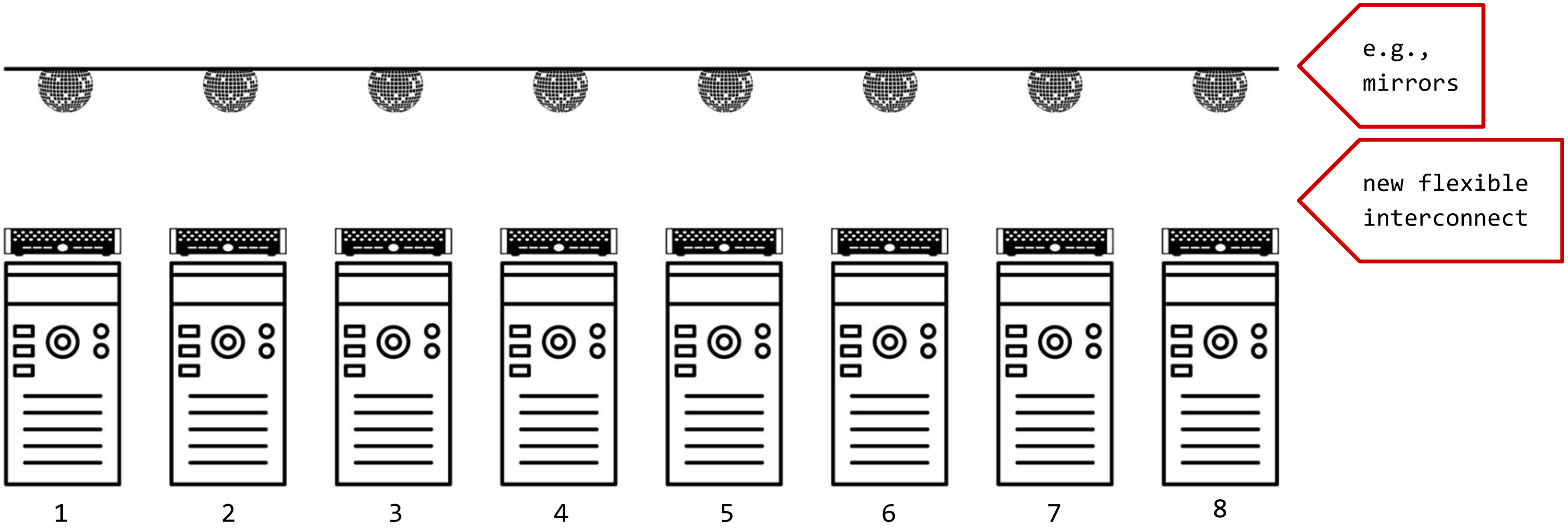
Our Vision:

Flexible and Demand-Aware Topologies



Our Vision:

Flexible and Demand-Aware Topologies

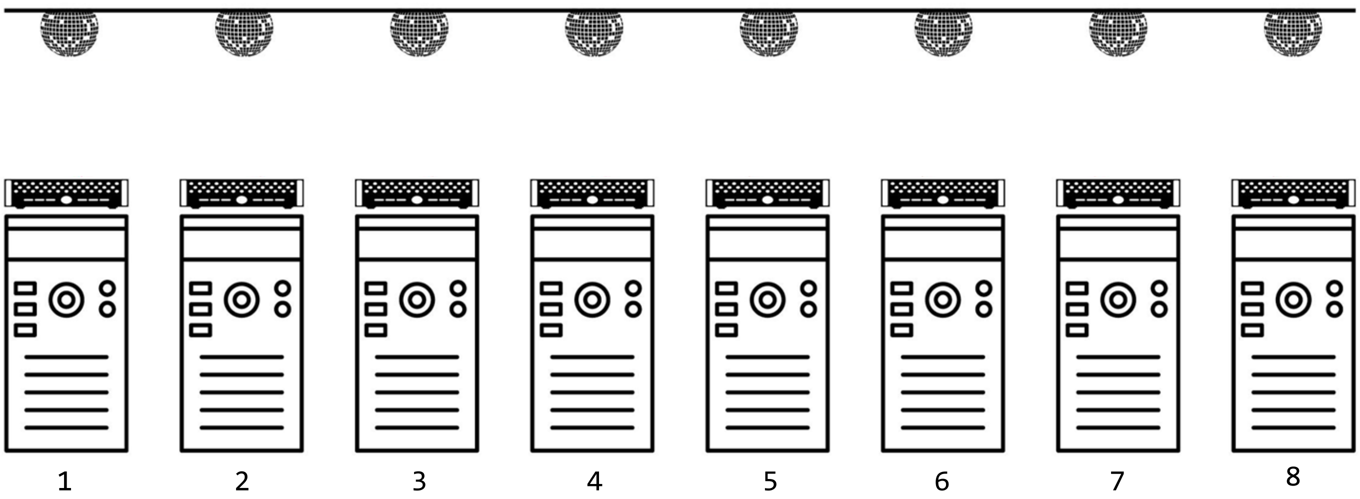


Our Vision:

Flexible and Demand-Aware Topologies

demand matrix:

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



e.g., mirrors

new flexible interconnect

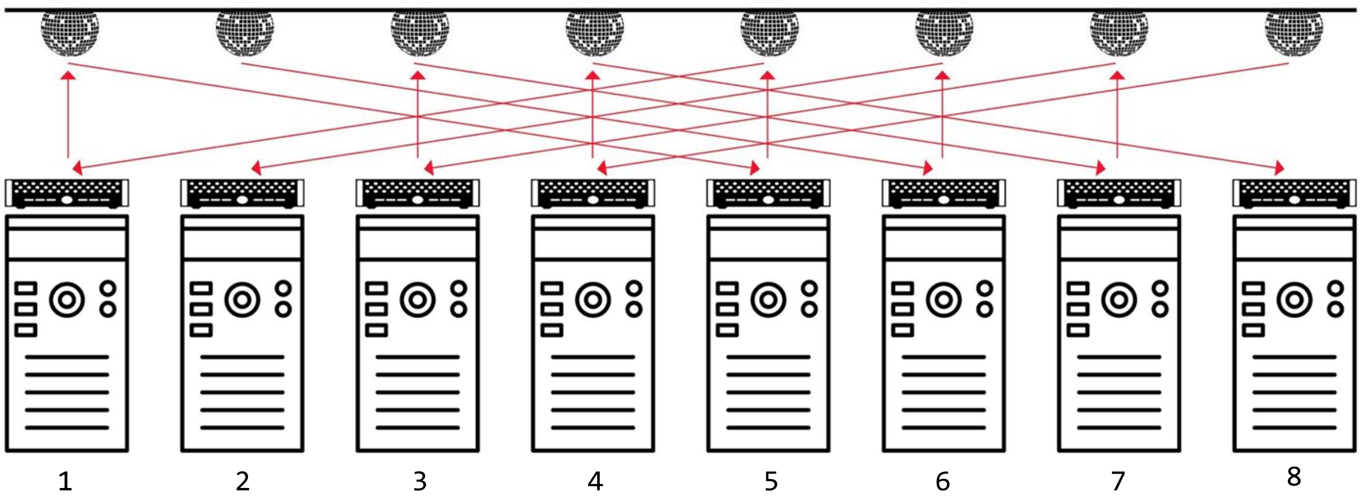
Our Vision:

Flexible and Demand-Aware Topologies

Matches demand

demand matrix:

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



e.g., mirrors

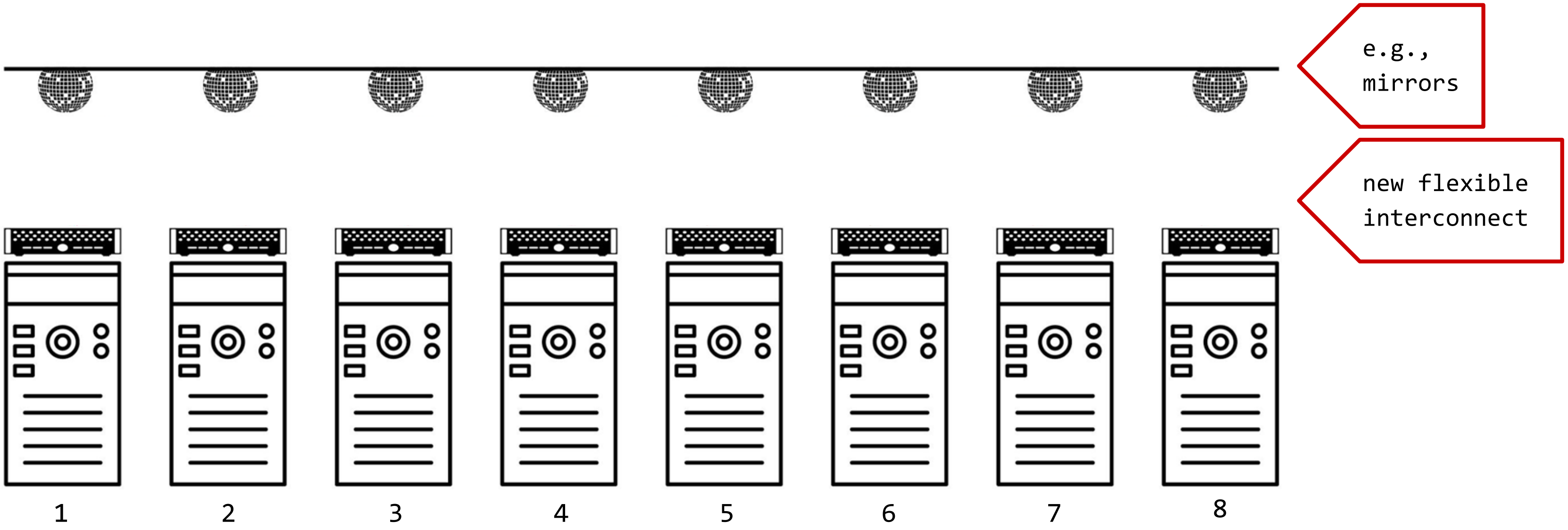
new flexible interconnect

Our Vision:

Flexible and Demand-Aware Topologies

new demand:

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



e.g., mirrors

new flexible interconnect

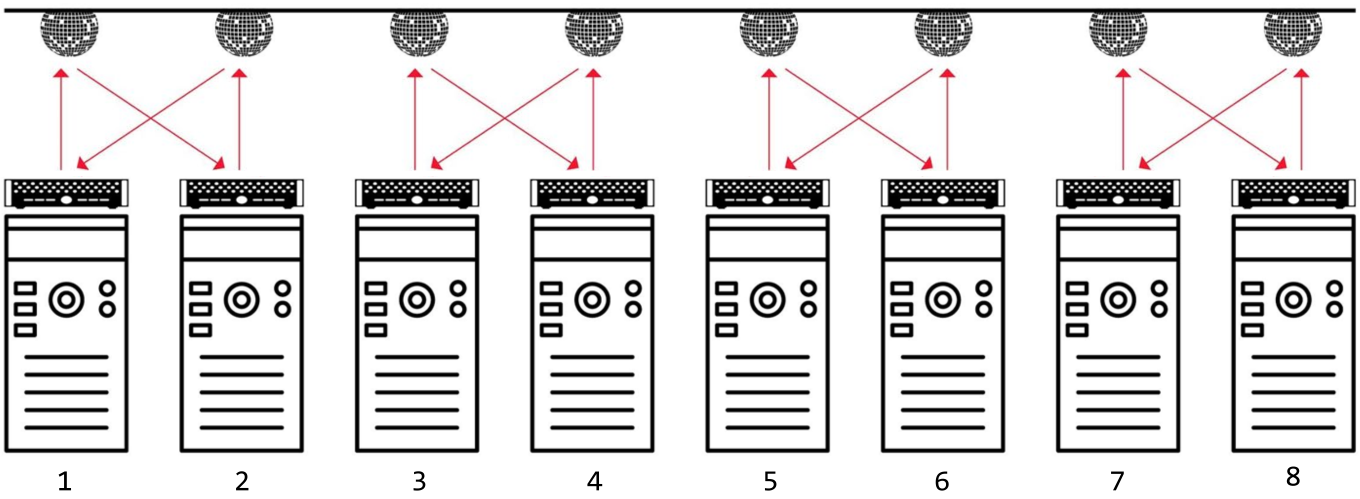
Our Vision:

Flexible and Demand-Aware Topologies

Matches demand

new demand:

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



e.g., mirrors

new flexible interconnect

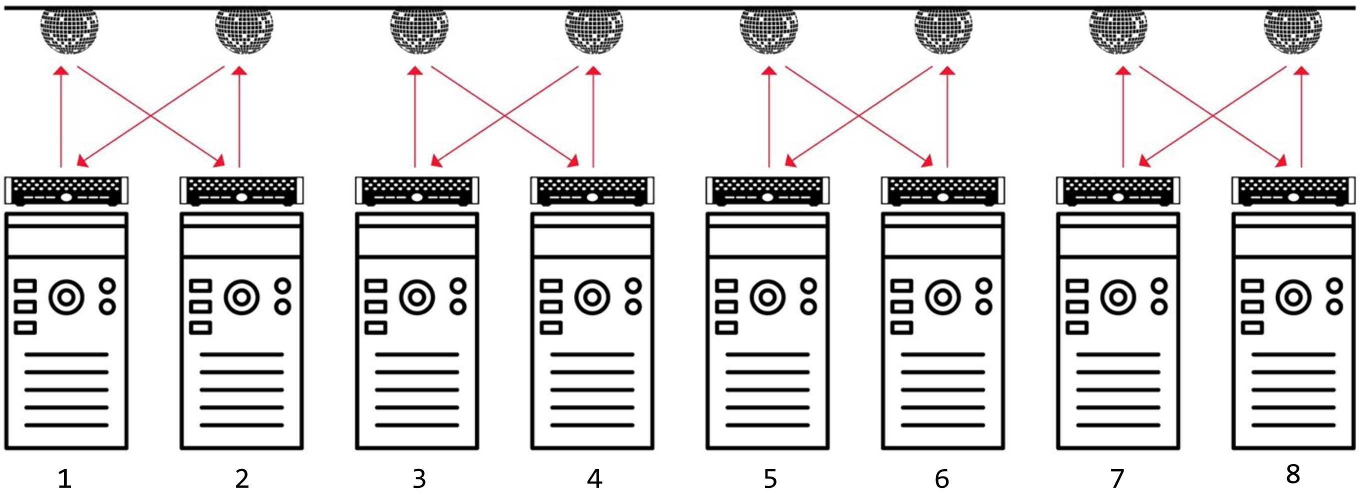
Our Vision:

Flexible and Demand-Aware Topologies

Self-Adjusting Networks

new demand:

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



e.g., mirrors

new flexible interconnect

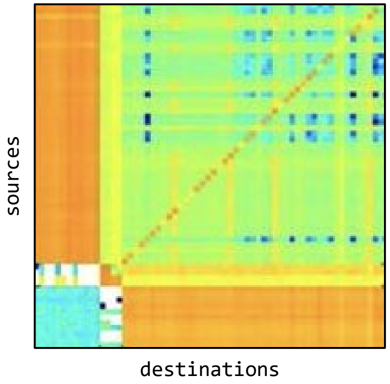
Our Motivation:

Much Structure in the Demand

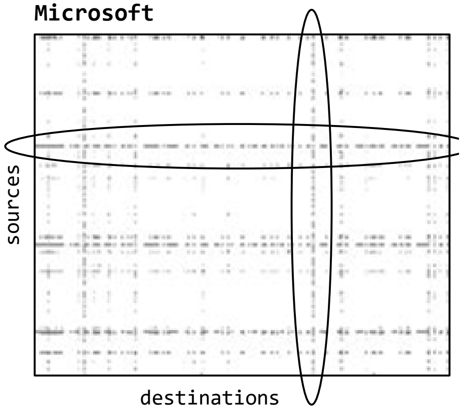
Empirical studies:

traffic matrices sparse and skewed

Facebook

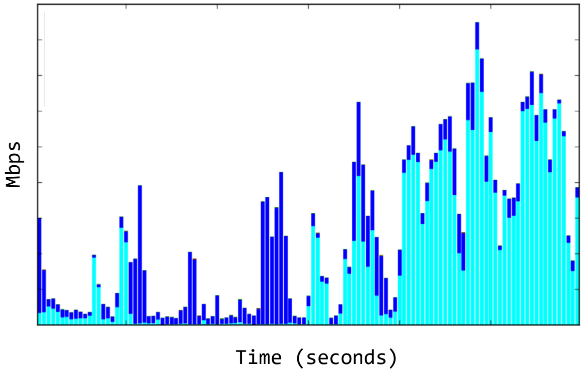


Microsoft



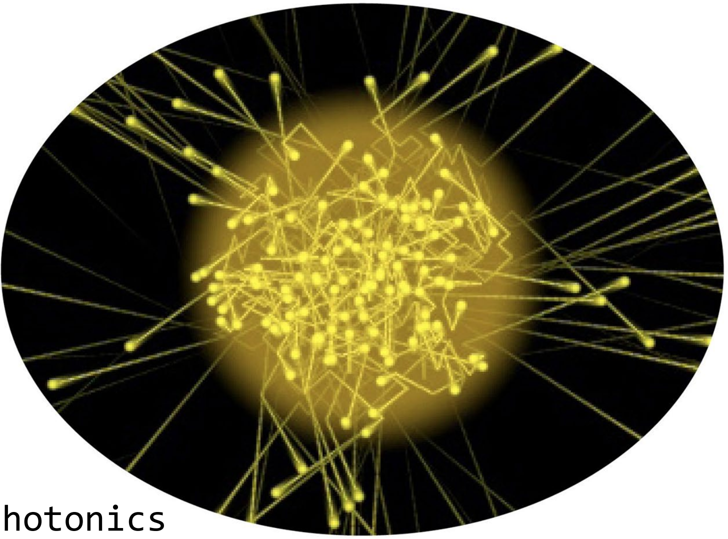
traffic bursty over time

Facebook



My hypothesis: can be exploited.

Sounds Crazy? Emerging Enabling Technology.



Photonics

H2020:

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

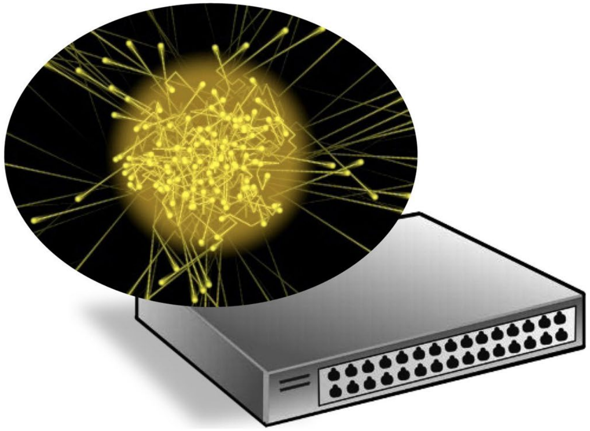
US National Research Council:

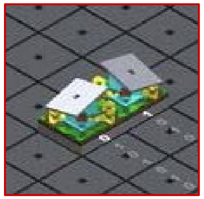
“Photons are the new Electrons.”

Enabler:

Novel Reconfigurable Optical Switches

- **Spectrum** of prototypes
  - Different sizes, different reconfiguration times
  - From our last month‘s ACM **SIGCOMM** workshop





Prototype 1

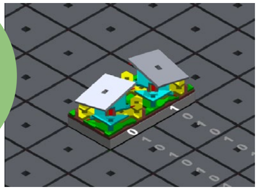
Prototype 2

Prototype 3

The Big Picture

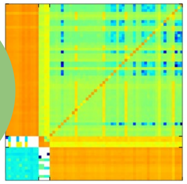
New!

Flexibility



More!

Structure



Self-Adjusting Networks

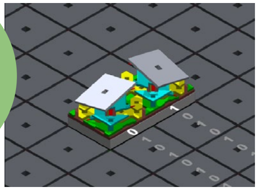
Efficiency

Now is the time!

The Big Picture

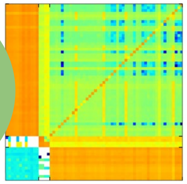
New!

Flexibility



More!

Structure



Self-Adjusting Networks

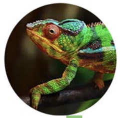
Now is the time!

Efficiency

Our goal: Develop the theoretical **foundations** of demand-aware, self-adjusting networks.

Unique Position:

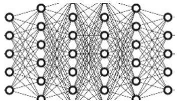
Demand-Aware, Self-Adjusting Systems



Everywhere, but mainly in software

Algorithmic trading

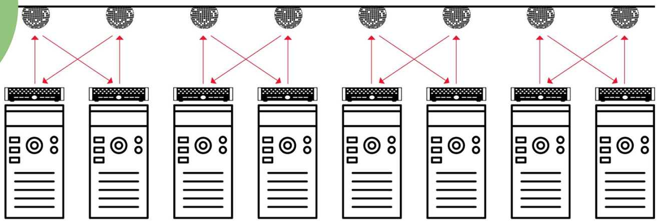
Recommender systems



Neural networks

VS

Our focus: in hardware



Question 1:

How to Quantify such “Structure” in the Demand?

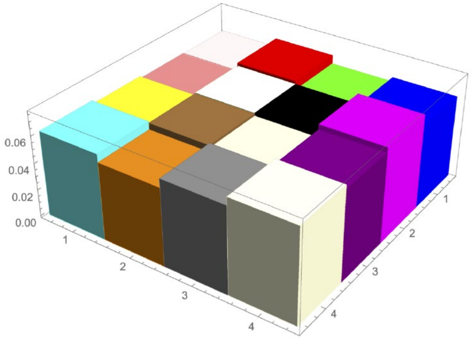
Intuition:

Which demand has more structure?

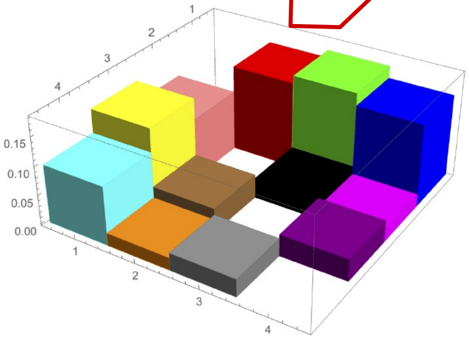
⇢ Traffic matrices of two different distributed ML applications

  ⇀ GPU-to-GPU

Color = communication pair



vs

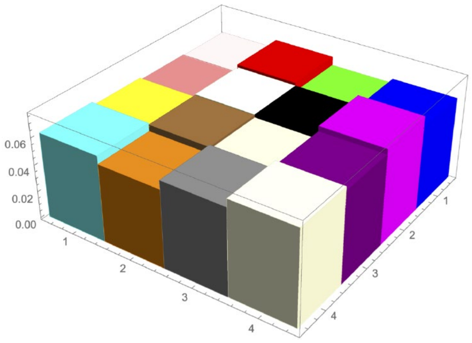


Intuition:

Which demand has more structure?

⇢ Traffic matrices of two different distributed ML applications

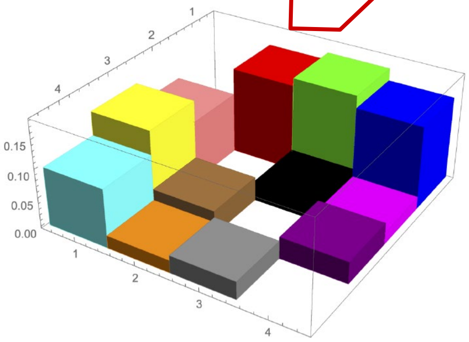
→ GPU-to-GPU



More uniform

VS

Color = communication pair



More structure

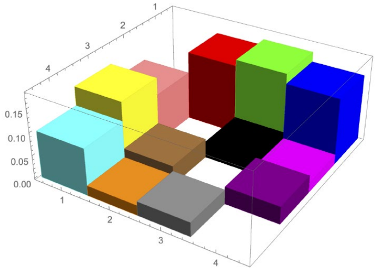
Intuition:

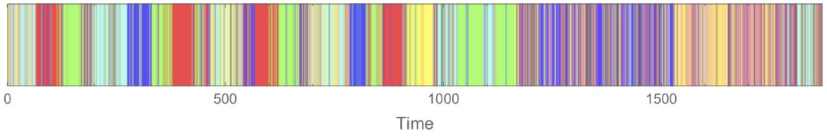
Spatial vs temporal structure

⇢ Two different ways to generate same traffic matrix:

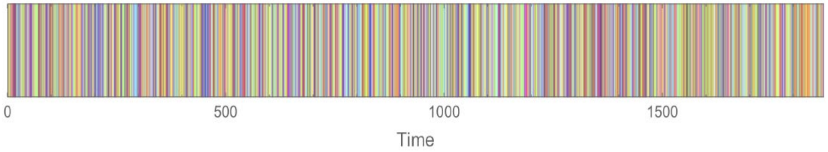
⇀ same non-temporal structure

⇢ Which one has more structure?





VS



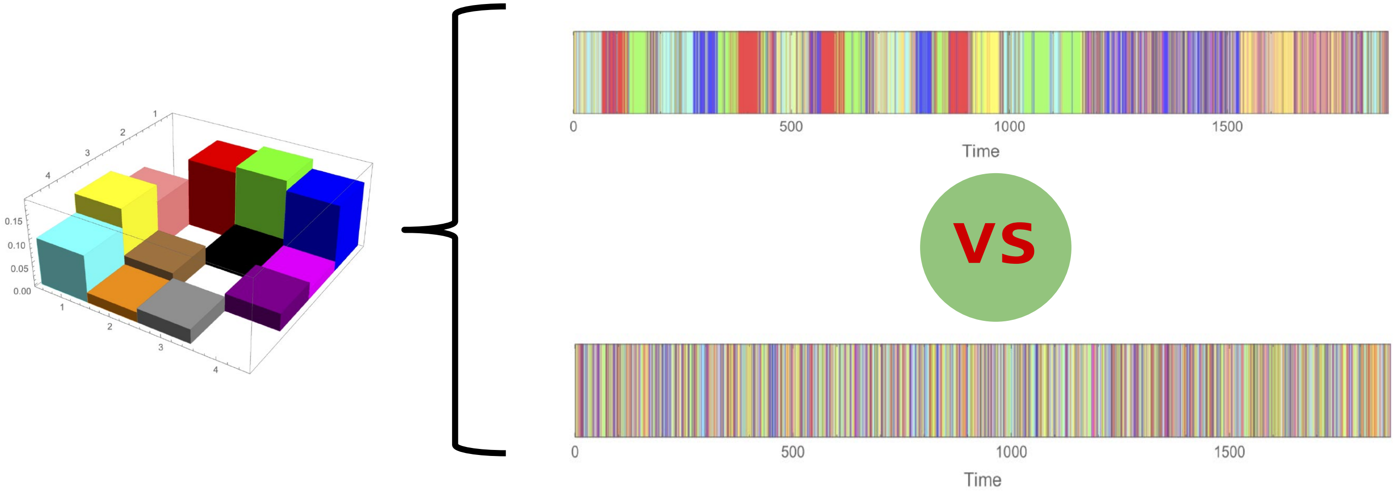
Intuition:

Spatial vs temporal structure

⇢ Two different ways to generate same traffic matrix:

  ⇀ same non-temporal structure

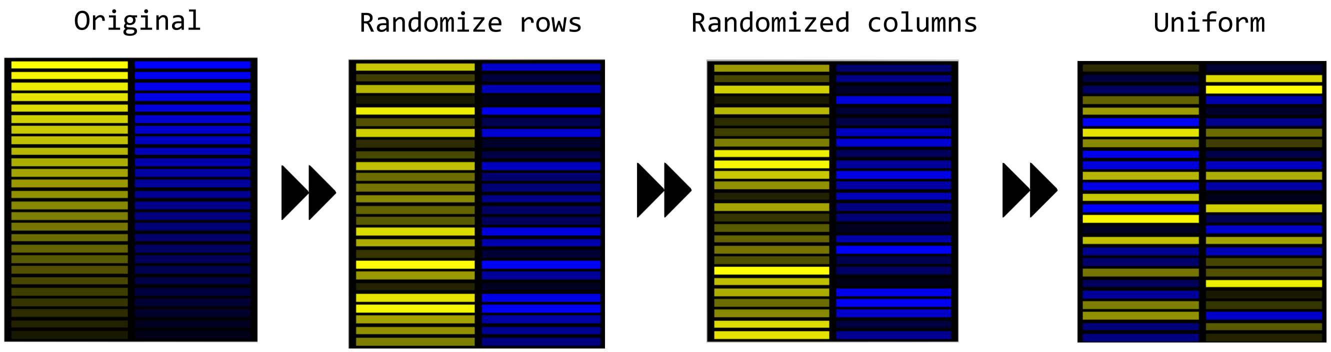
⇢ Which one has more structure?



Systematically?

Trace Complexity:

A Systematic “Shuffle&Compress” Approach



Original

Randomize rows

Randomized columns

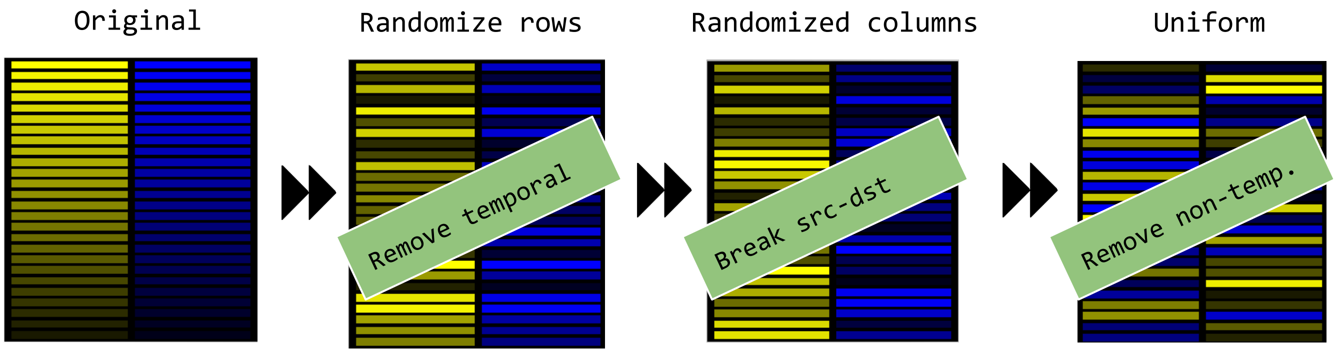
Uniform

Increasing complexity (systematically randomized)

More structure (compresses better)

Trace Complexity:

A Systematic “Shuffle&Compress” Approach

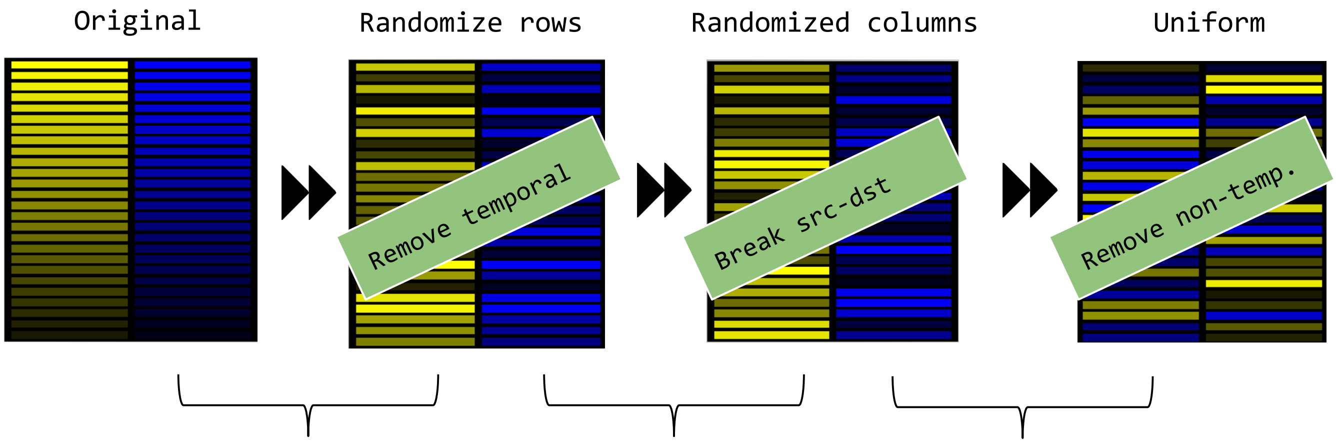


Increasing complexity (systematically randomized)

More structure (compresses better)

Trace Complexity:

A Systematic “Shuffle&Compress” Approach



Original

Randomize rows

Randomized columns

Uniform

Remove temporal

Break src-dst

Remove non-temp.

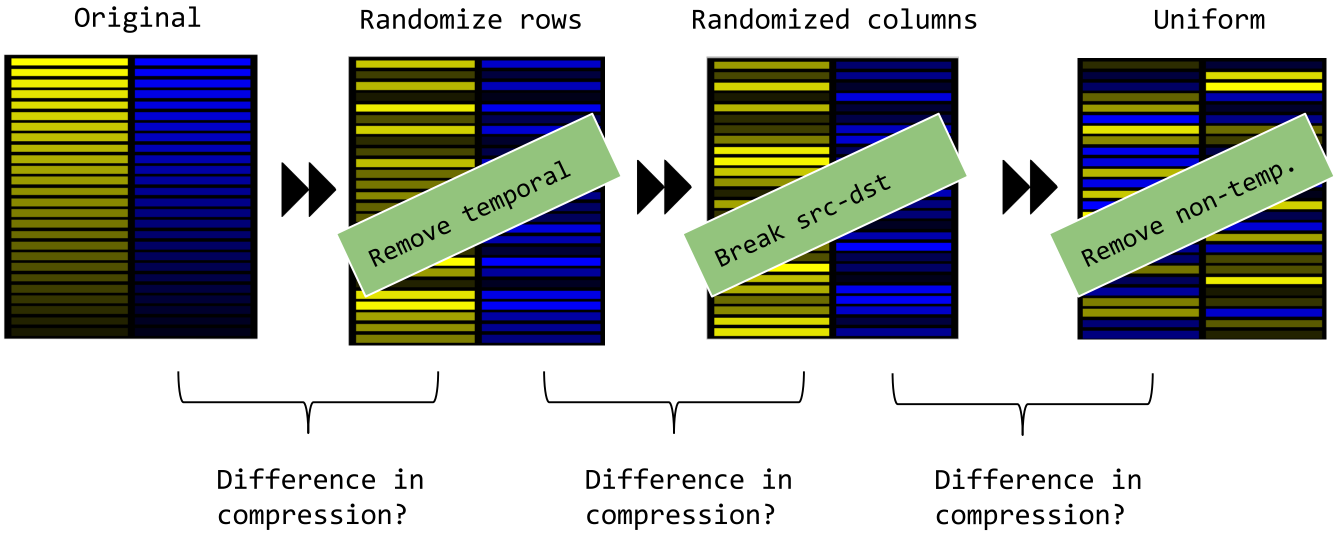
Difference in compression?

Difference in compression?

Difference in compression?

Trace Complexity:

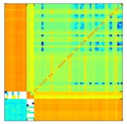
A Systematic “Shuffle&Compress” Approach

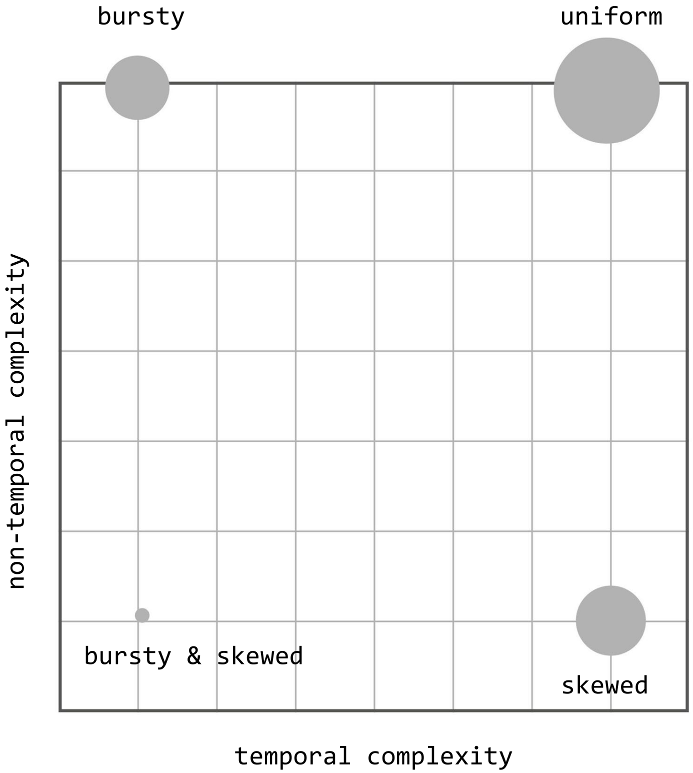


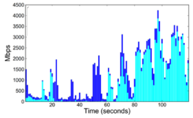
Can be used to define a “Complexity Map”!

Our Methodology:

Complexity Map





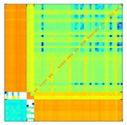


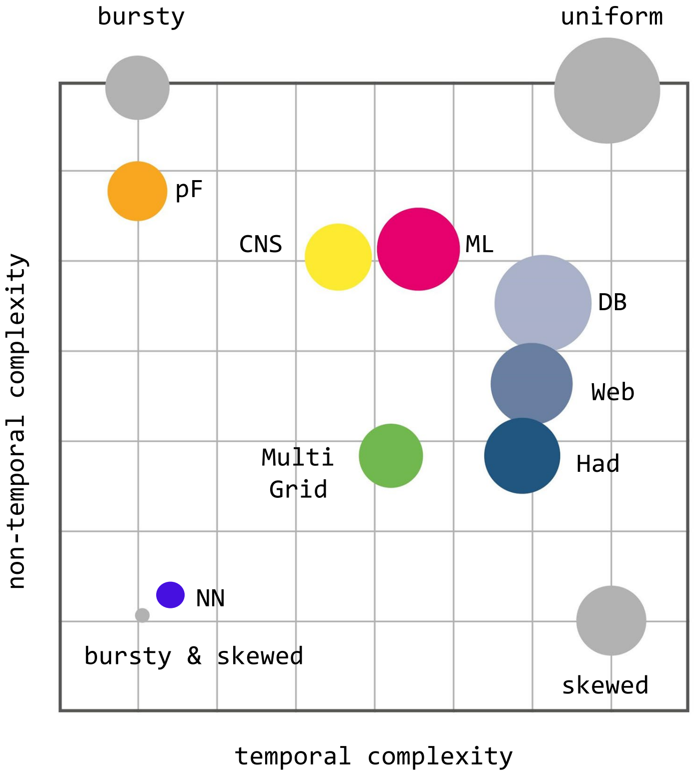
No structure

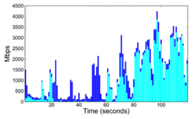
Our **approach**: iterative **randomization and compression** of trace to identify dimensions of structure.

Our Methodology:

Complexity Map







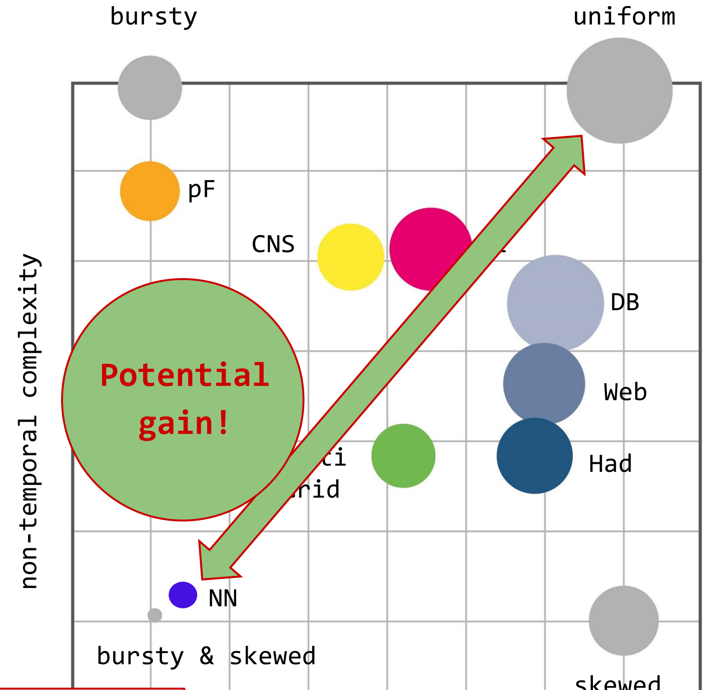
No structure

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

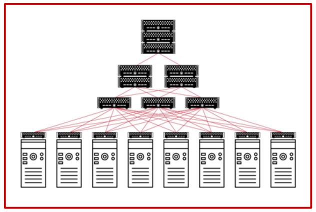
Different structures!

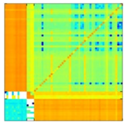
Our Methodology:

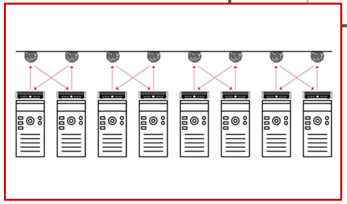
Complexity Map



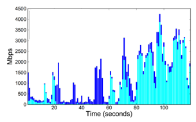
Potential gain!







temporal complexity



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

Different structures!

Question 2:

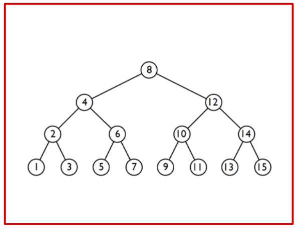
Given This Structure, What Can Be Achieved? Metrics and Algorithms?

A first insight: entropy of the demand.

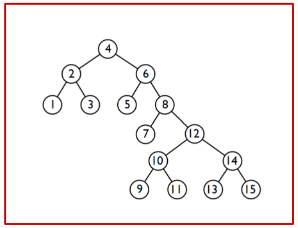
Our Approach:

Connection to Datastructures

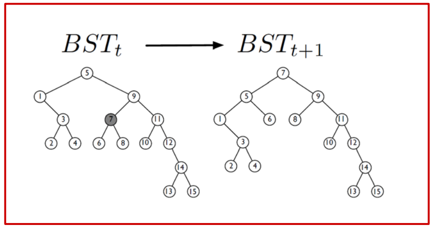
Traditional BST



Demand-aware BST



Self-adjusting BST

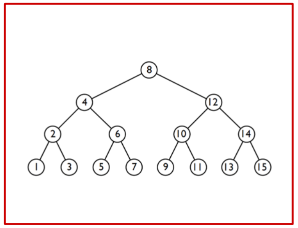


More structure: improved **access cost**

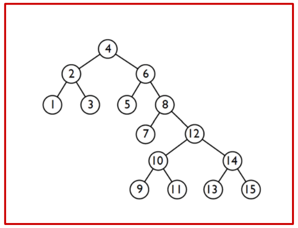
Our Approach:

Connection to Datastructures & Coding

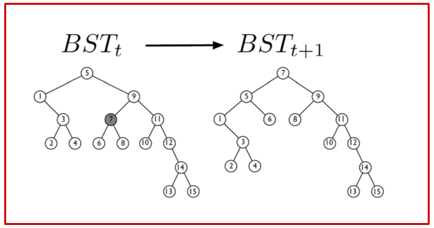
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)



Self-adjusting BST
(Dynamic Huffman coding)



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

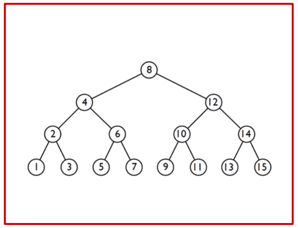
Our Approach:

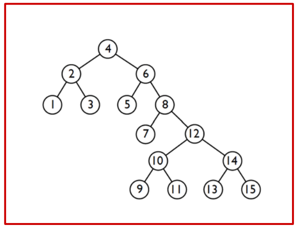
Connection to Datastructures & Coding

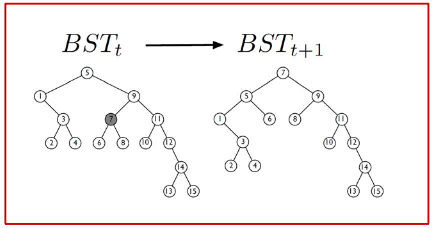
Traditional BST
(Worst-case coding)

Demand-aware BST
(Huffman coding)

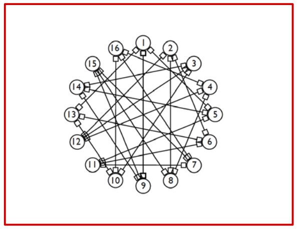
Self-adjusting BST
(Dynamic Huffman coding)

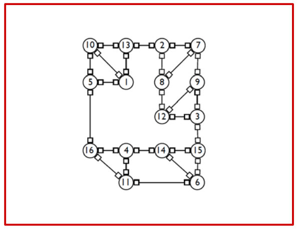


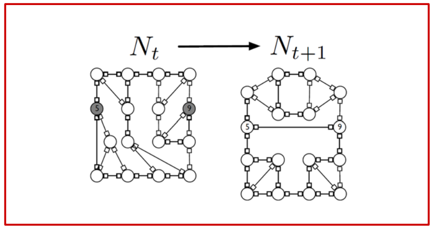




More structure: improved **access cost** / shorter **codes**







Similar **benefits**?

Our Approach:

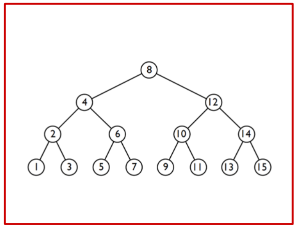
Connection to Datastructures & Coding

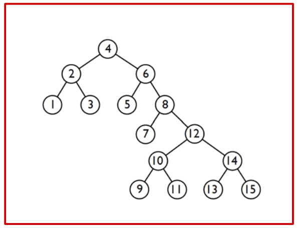
Traditional BST
(Worst-case coding)

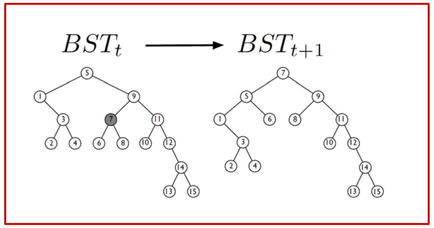
Demand-aware BST
(Huffman coding)

Self-adjusting BST
(Dynamic Huffman coding)

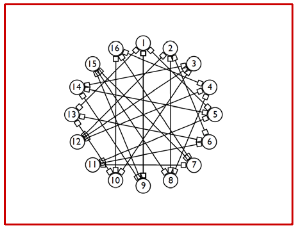
More than
an analogy!

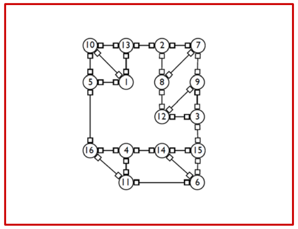


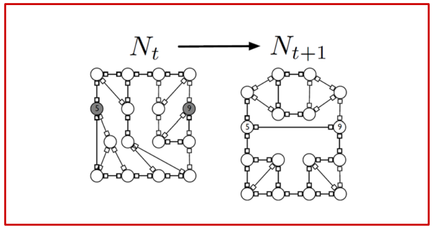




More structure: improved **access cost** / shorter **codes**





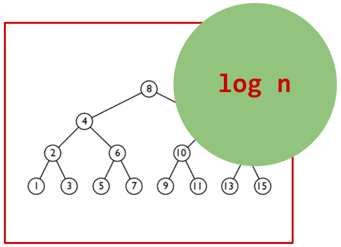


Similar **benefits**?

Our Approach:

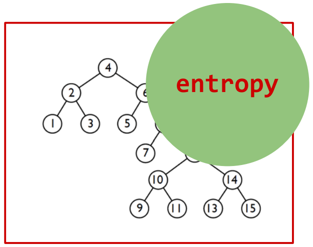
Connection to Datastructures & Coding

Traditional BST
(Worst-case coding)



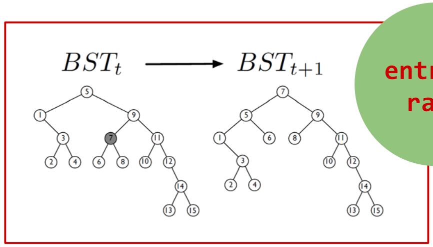
log n

Demand-aware BST
(Huffman coding)

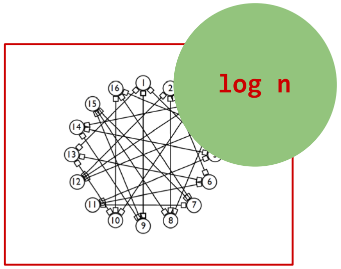


entropy

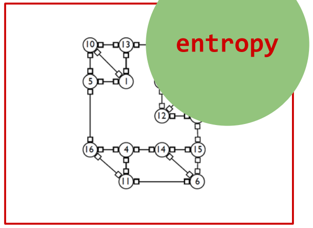
Self-adjusting BST
(Dynamic Huffman coding)



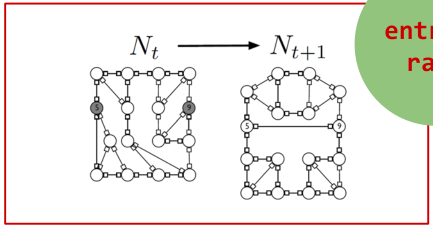
entropy rate



log n



entropy



entropy rate

Reduced expected **route lengths**!

More than an analogy!

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

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

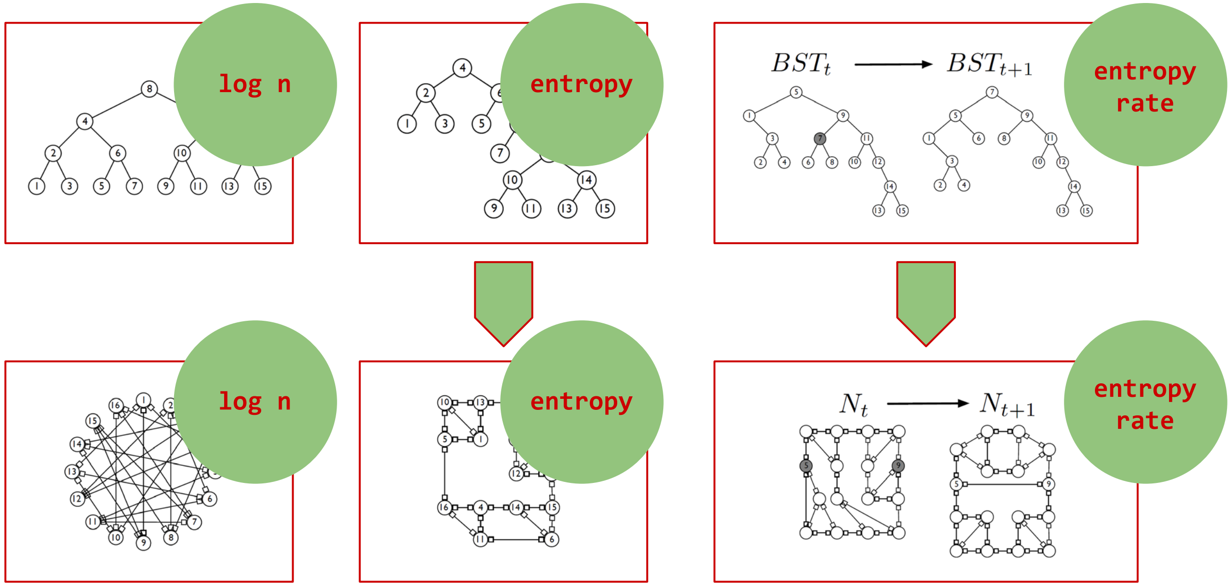
Our Approach:

Connection to Datastructures & Coding

Traditional BST (Worst-case coding)

Demand-aware BST (Huffman coding)

Self-adjusting BST (Dynamic Huffman coding)



log n

entropy

entropy rate

log n

entropy

entropy rate

Reduced expected **route lengths**!

More than an analogy!

Generalize methodology:

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

First result:

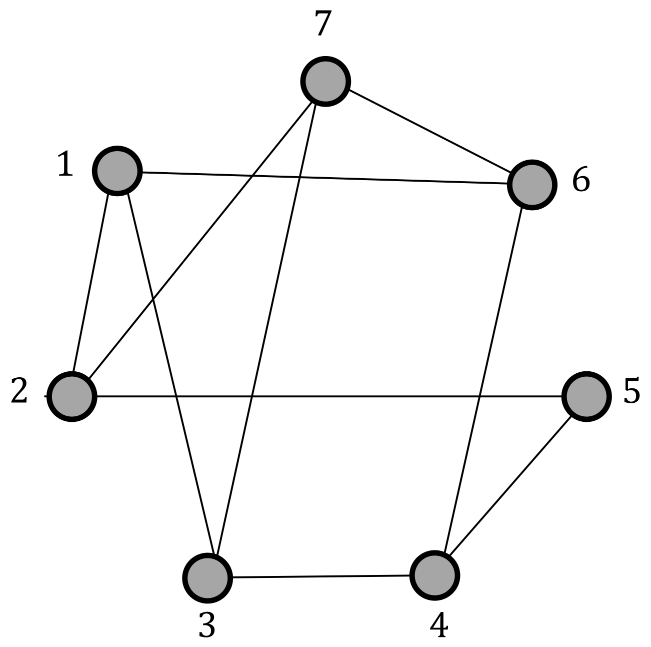
Demand-aware networks of asymptotically optimal route lengths.

An Example

Destinations

Sources

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



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

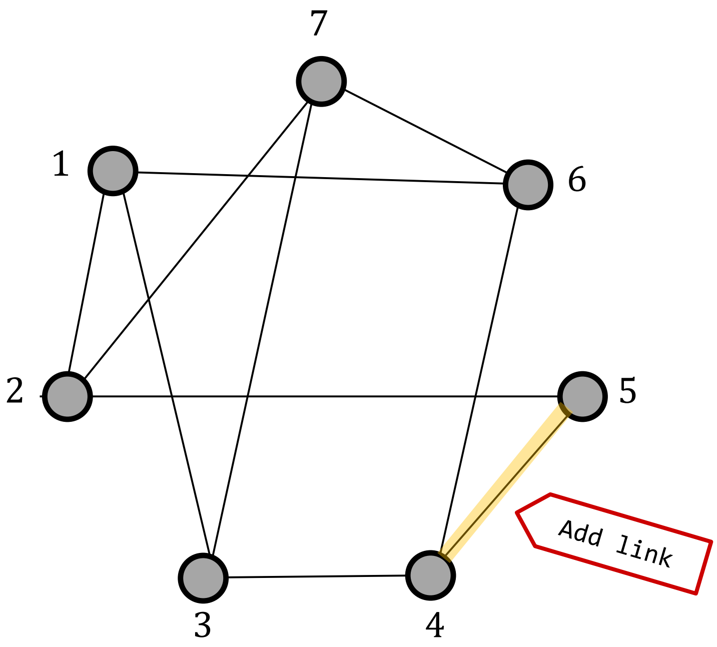
An Example

Destinations

Sources

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

Much from 4 to 5



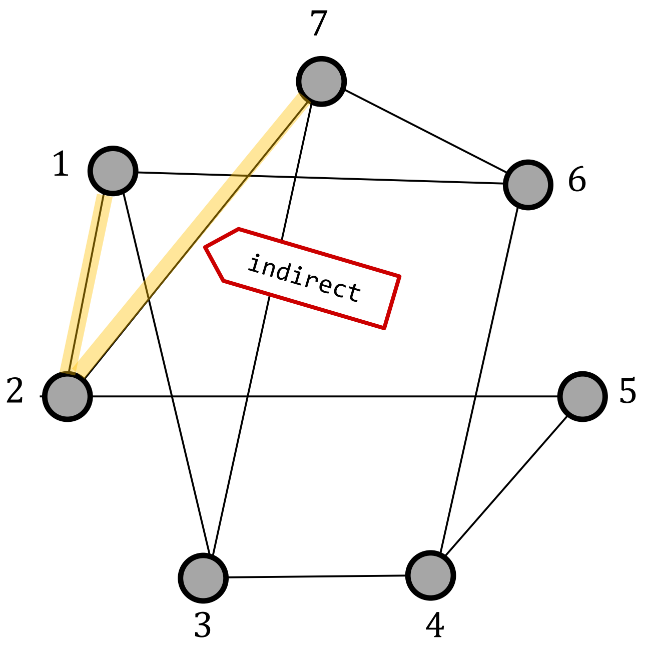
$$\mathrm{ERL}(\mathcal{D},\mathrm{N}) = \sum_{(u,v)\in\mathcal{D}} p(u,v)\cdot d_{N}(u,v)$$

An Example

Communicated with many

Destinations

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



indirect

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

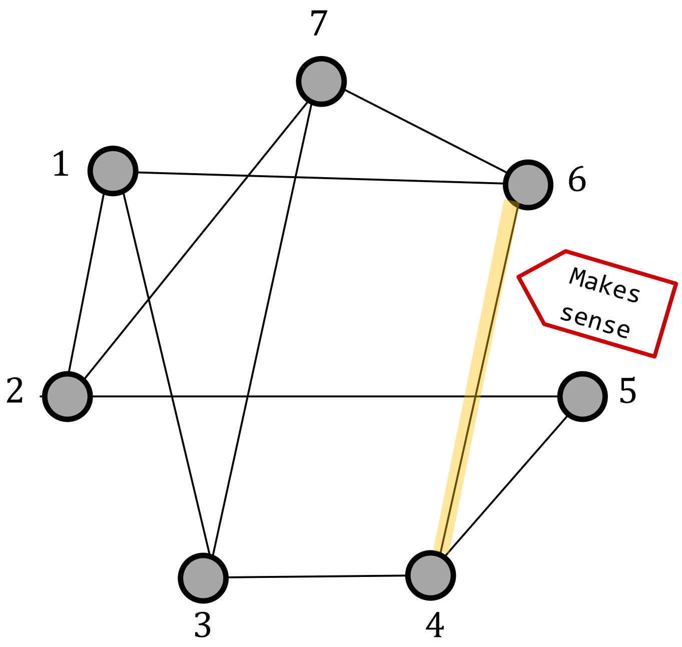
An Example

Destinations

Sources

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

Don't communicate



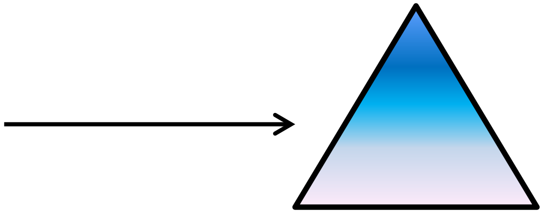
$$\mathrm{ERL}(\mathcal{D},\mathrm{N}) = \sum_{(u,v)\in\mathcal{D}} p(u,v)\cdot d_{N}(u,v)$$

From Static Coding:

Entropy Lower Bound

Destinations

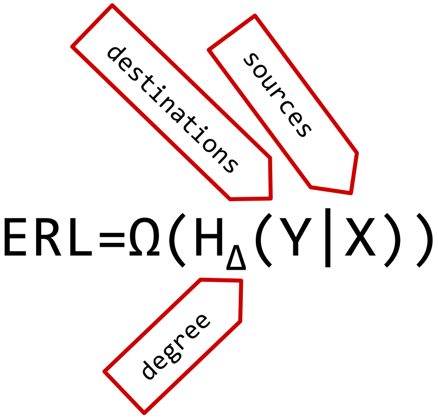
Sources	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$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



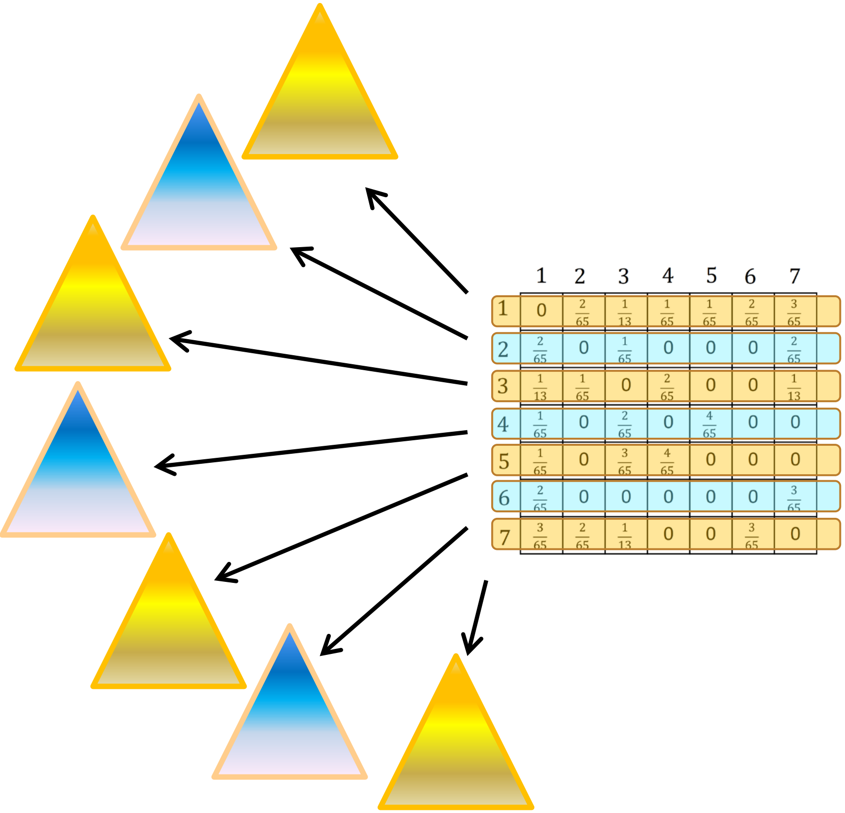
Huffman tree:
“**ego-tree**”

From Static Coding:

Entropy Lower Bound



$$ERL=\Omega(H_{\Delta}(Y|X))$$



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

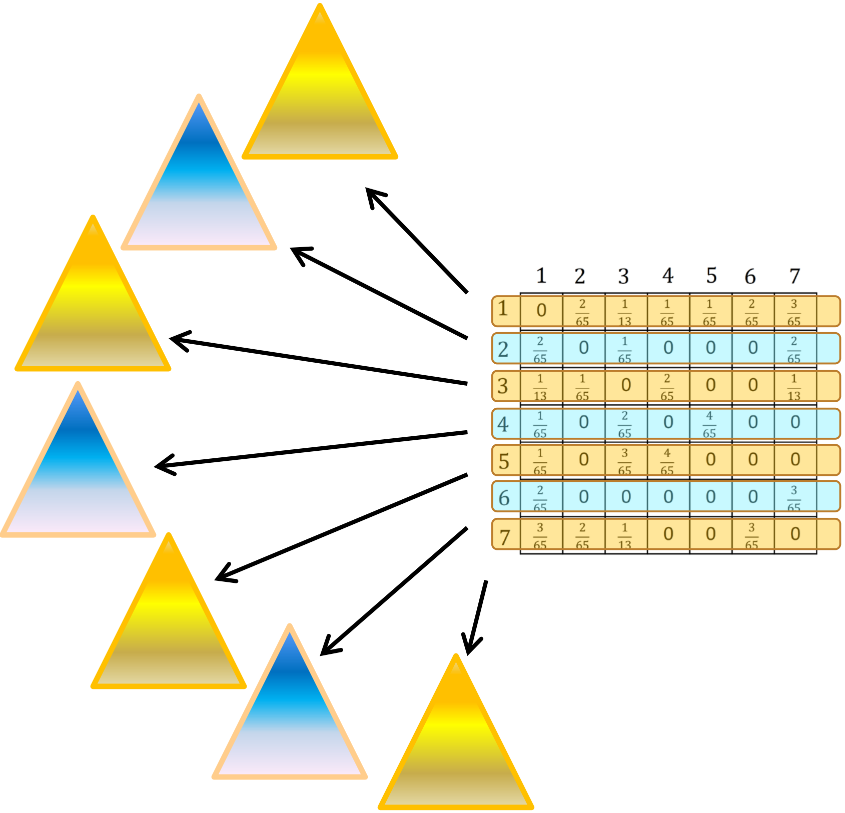
From Static Coding:

Upper Bound and Algo

- Idea for algorithm:
  - union of trees
  - reduce degree

- Ok for sparse demands
  - helper nodes

Static



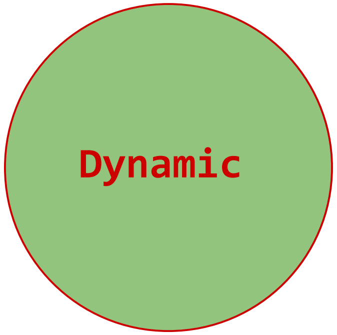
	1	2	3	4	5	6	7
1	0	$\frac{2}{65}$	$\frac{1}{13}$	$\frac{1}{65}$	$\frac{1}{65}$	$\frac{2}{65}$	$\frac{3}{65}$
2	$\frac{2}{65}$	0	$\frac{1}{65}$	0	0	0	$\frac{2}{65}$
3	$\frac{1}{13}$	$\frac{1}{65}$	0	$\frac{2}{65}$	0	0	$\frac{1}{13}$
4	$\frac{1}{65}$	0	$\frac{2}{65}$	0	$\frac{4}{65}$	0	0
5	$\frac{1}{65}$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

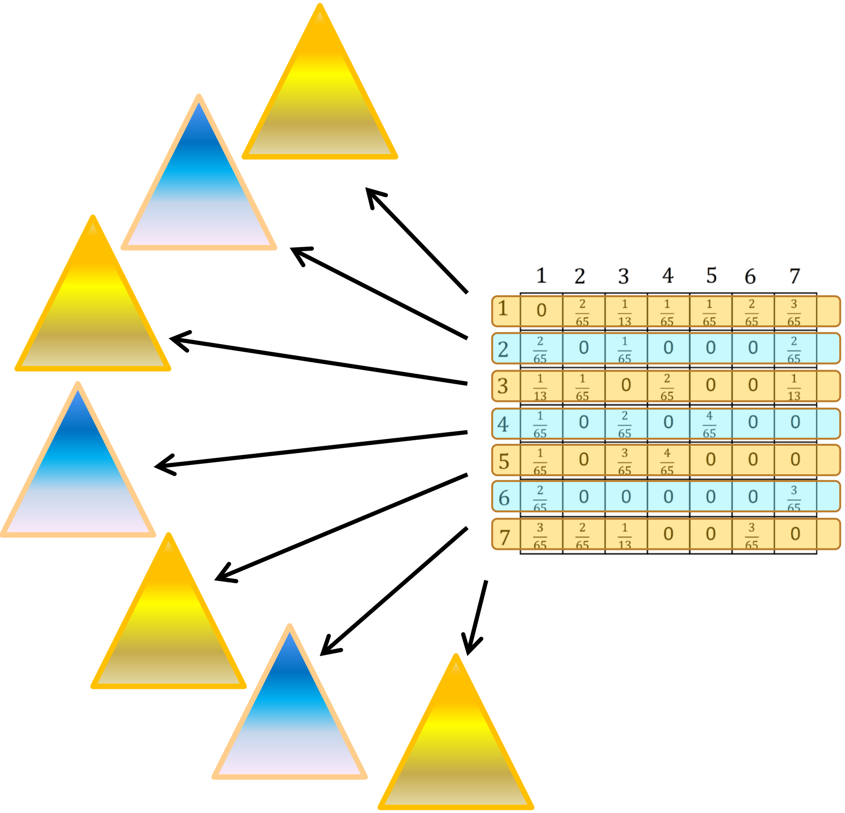
What about dynamic case?

From Dynamic Coding:

Dynamic Setting

- ⇢ Dynamic the same:
  - → union of **dynamic ego-trees**
- ⇢ E.g., SplayNets





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

Future Work

so far

scratched surface

to do ☺

Notion of self-adjusting networks opens a **large uncharted field** with many questions:

→ By how much can load be lowered, **energy** reduced, quality-of-service improved, etc. in demand-aware networks?

→ How to **model** reconfiguration costs?

→ How to render these networks **robust?**

→ Impact on **other layers**?

→ How to design **scalable** control planes?

Challenges:

Domain 1
Models and metrics

Domain 2
Algorithms

Domain 3
Integration

Requires knowledge in networking, distributed systems, algorithms, performance evaluation.

Future Work

so far

scratched surface

to do ☺

Notion of self-adjusting networks opens a **large uncharted field** with many questions:

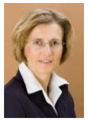
→ By how much can load be lowered, **energy** reduced, quality-of-service improved, etc. in demand-aware networks?

low to model reconfiguration costs?

v to render these networks **robust?**

st on **other layers**?

w to design **scalable** control planes?



Monika Henzinger (University of Vienna)

Challenges:

Domain 1
Models and metrics

Domain 2
Algorithms

Domain 3
Integration

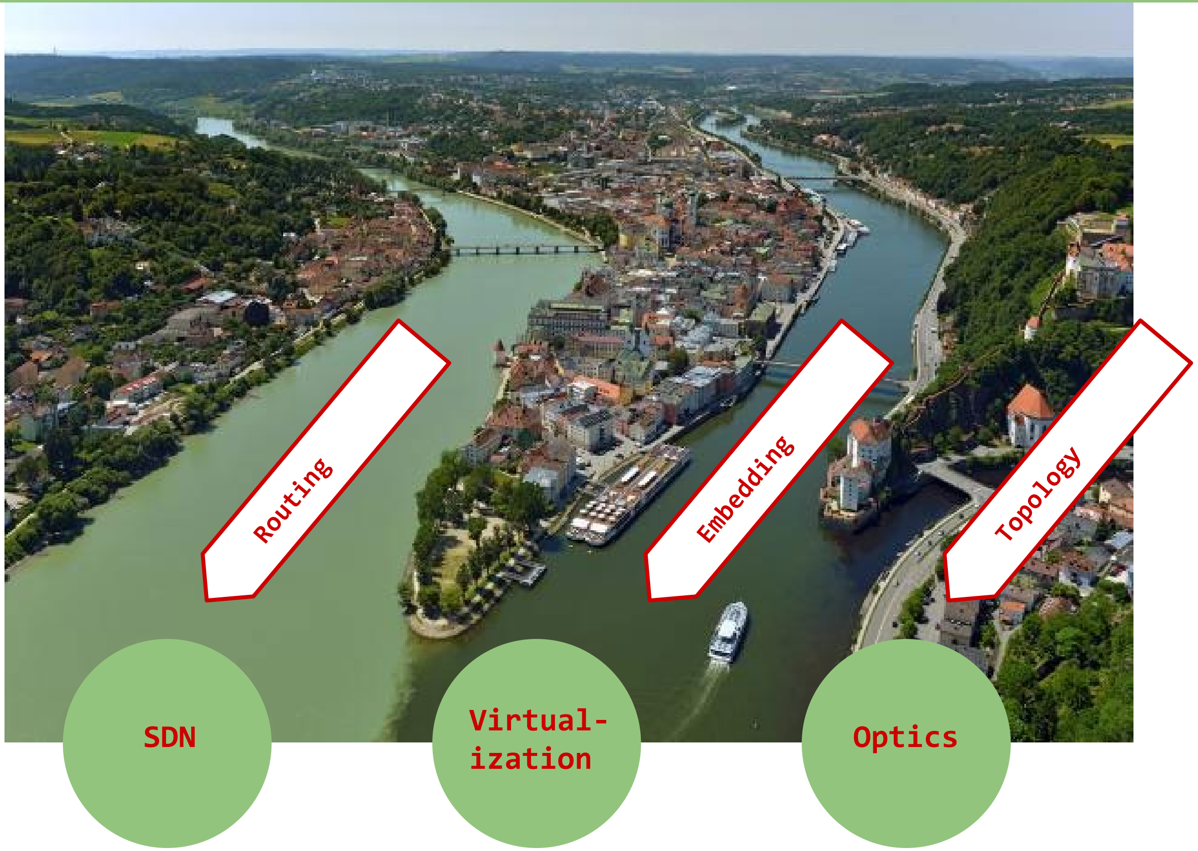
Requires knowledge in networking, distributed systems, algorithms, performance evaluation.

Even bigger picture:

Flexible Networks

Even bigger picture:

Flexible Networks



Routing

Embedding

Topology

SDN

Virtual-ization

Optics

Contributors

Chen Avin

Manya Ghobadi

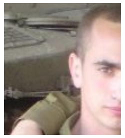
Olga Goussevskaia

Klaus-Tycho Foerster

Kaushik Mondal

Ingo van Duijn

Iosif Salem



Khen Griner

Bruna Peres

et al.!

Funding: GIF German-Israeli Foundation for Scientific Research and Development

References

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

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

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

Measuring the Complexity of Network Traffic Traces
Chen Griner, Chen Avin, Manya Ghobadi, and Stefan Schmid.
arXiv, 2019.

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

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

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

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

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

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

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