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

(Folklore)
Today’s Datacenters

Fixed and Demand-Oblivious Topology

Many flavors, but in common: fixed and oblivious to actual demand.
Today’s Datacenters

Fixed and Demand-Oblivious Topology

Highway which ignores actual traffic: frustrating!

Many flavors, but in common: fixed and oblivious to actual demand.
Vision
Demand-Aware Networks

Flexibility
Demand Structure

OptSys

Demand-Aware Networks

Efficiency

Now is the time!
But how much does it help? As usual in computer science: it depends! We need metrics for demand structure and for possible efficiency.
Our Perspective
Information Theory and Entropy

Demand entropy:
Spatial and temporal structure of traffic

Entropy: A tight metric for the achievable route lengths in demand-aware networks
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
Intuition
Which demand has more structure?

→ Traffic matrices of two different distributed ML applications
→ GPU-to-GPU

More uniform  VS  More structure
Intuition

Spatial vs Temporal Structure

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

→ Which one has more structure?
Intuition

Spatial vs Temporal Structure

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

→ Which one has more structure?

Systematically?
Trace Complexity
Information-Theoretic Approach
“Shuffle&Compress”
Trace Complexity

Information-Theoretic Approach

“Shuffle&Compress”

Increasing complexity (systematically randomized)

More structure (compresses better)
Trace Complexity
Information-Theoretic Approach
“Shuffle&Compress”
Trace Complexity
Information-Theoretic Approach
“Shuffle&Compress”

Original → Randomize rows → Uniform

Shuffle

Compress

Difference in size (entropy)?

Difference in size (entropy)?
Trace Complexity

Information-Theoretic Approach
“Shuffle&Compress”

Can be used to define 2-dimensional complexity map!
Trace Complexity

Complexity Map

bursty & skewed

skewed

bursty & skewed

temporal complexity

non-temporal complexity

uniform

No structure
Trace Complexity

Complexity Map

Different structures!
Trace Complexity

Complexity Map

- Non-temporal complexity
- Temporal complexity
- Potential gain!
- Different structures!
On the Complexity of Traffic Traces and Implications

CHEN AVIN, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel
MANYA GHOBADI, Computer Science and Artificial Intelligence Laboratory, MIT, USA
CHEN GRINER, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel
STEFAN SCHMID, Faculty of Computer Science, University of Vienna, Austria

This paper presents a systematic approach to identify and quantify the types of structures featured by packet traces in communication networks. Our approach leverages an information-theoretic methodology, based on iterative randomization and compression of the packet trace, which allows us to systematically remove and measure dimensions of structure in the trace. In particular, we introduce the notion of trace complexity which approximates the entropy rate of a packet trace. Considering several real-world traces, we show that trace complexity can provide unique insights into the characteristics of various applications. Based on our approach, we also propose a traffic generator model able to produce a synthetic trace that matches the complexity levels of its corresponding real-world trace. Using a case study in the context of datacenters, we show that insights into the structure of packet traces can lead to improved demand-aware network designs: datacenter topologies that are optimized for specific traffic patterns.

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

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

ACM Reference Format:

1 INTRODUCTION
Packet traces collected from networking applications, such as datacenter traffic, have been shown to feature much structure: datacenter traffic matrices are sparse and skewed [16, 39], exhibit
Question 2:

How to Exploit Structure Algorithmically? Metrics for Achievable Efficiency?

Insight: Information-theoretic perspective useful here as well!
Models and Connection to Datastructures & Coding

Traditional networks
(worst-case traffic)
Models and Connection to Datastructures & Coding

Traditional networks
(worst-case traffic)  
Demand-aware networks
(spatial structure)
Models and Connection to Datastructures & Coding

Traditional networks (worst-case traffic)  Demand-aware networks (spatial structure)  Self-adjusting networks (temporal structure)
Models and Connection to Datastructures & Coding

Traditional networks (worst-case traffic)  Demand-aware networks (spatial structure)  Self-adjusting networks (temporal structure)

More structure: lower routing cost
Models and Connection to Datastructures & Coding

Traditional networks (worst-case traffic)  
Demand-aware networks (spatial structure)  
Self-adjusting networks (temporal structure)

More structure: lower routing cost

Traditional BST (Worst-case coding)  
Demand-aware BST (Huffman coding)  
Self-adjusting BST (Dynamic Huffman coding)

More structure: improved access cost / shorter codes
Models and Connection to Datastructures & Coding

Traditional networks (worst-case traffic)

Demand-aware networks (spatial structure)

Self-adjusting networks (temporal structure)

More structure: lower routing cost

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.

More structure: improved access cost / shorter codes
Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

\[
\text{ERL}(\mathcal{D}, N) = \sum_{(u, v) \in \mathcal{D}} p(u, v) \cdot d_N(u, v)
\]
Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

<table>
<thead>
<tr>
<th>Sources</th>
<th>Destinations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2/65</td>
</tr>
<tr>
<td>3</td>
<td>1/13</td>
</tr>
<tr>
<td>4</td>
<td>1/65</td>
</tr>
<tr>
<td>5</td>
<td>1/65</td>
</tr>
<tr>
<td>6</td>
<td>2/65</td>
</tr>
<tr>
<td>7</td>
<td>3/65</td>
</tr>
</tbody>
</table>

\[
\text{ERL}(\mathcal{D}, N) = \sum_{(u, v) \in \mathcal{D}} p(u, v) \cdot d_N(u, v)
\]
Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

\[
\text{ERL}(\mathcal{D}, N) = \sum_{(u, v) \in \mathcal{D}} p(u, v) \cdot d_N(u, v)
\]
Case Study “Route Lengths”

Constant-Degree Demand-Aware Network

**ERL(Ω, Δ)** = \[ \sum_{(u, v) \in \Omega} p(u, v) \cdot d_\Delta(u, v) \]
Entropy Lower Bound

Huffman tree: “ego-tree”
Entropy Lower Bound

\[ \text{ERL} = \Omega(H_\Delta(Y|X)) \]
Idea for algorithm:
- union of trees
- reduce degree
- but keep distances

Ok for sparse demands
- not everyone gets tree
- helper nodes

What about dynamic case?
Dynamic Setting

→ Dynamic the same:
   → union of dynamic ego-trees

→ E.g., SplayNets

→ Online algorithms
Dynamic Objectives

Demand-Aware

Reconfigurable

Offline

ON

Static Optimality

Dynamic Optimality

Working Set
Overview: Models

Toward Demand-Aware Networking: A Theory for Self-Adjusting Networks

Chen Avin
Ben Gurion University, Israel
avin@cse.bgu.ac.il

Stefan Schmid
University of Vienna, Austria
stefan.schmid@univie.ac.at

ABSTRACT

This paper studies the design of self-adjusting networks whose topology dynamically adapts to the actual demand (i.e., workload or communication pattern) they currently serve. Rather, they are designed for all-to-all communication patterns, by ensuring properties such as full broadcast bandwidth or O(log n) route lengths between any node pair in a constant-degree n-node network. However, demand-oblivious networks can be inefficient for more specific demand patterns, as they usually arise in

Dynamic DAN

Overview: Demand-Aware Network Designs of Bounded Degree

Chen Avin
Koushik Mondal
Stefan Schmid

Abstract Traditionally, networks such as datacenter internetworks are designed to optimize worst-case performance under arbitrary traffic patterns. Such network designs can however be far from optimal when considering the actual workloads and traffic patterns which they serve. This insight led to the development of demand-aware datacenter interconnects which can be reconfigured depending on the workload. Motivated by these trends, this paper initiates the 1st revision of the algorithmic study of demand-aware networks (DAN), and in particular the design of bounded-degree networks. The inputs to the network design problem are a discrete communication request distribution, \( \mathcal{D} \), defined over communicating pairs from the node set \( V \), and a bound, \( \Delta \), on the maximum degree. In turn, the objective is to design an (undirected) demand-aware network \( \mathcal{N} = (V, E) \) of bounded-degree \( \Delta \), which provides short routing paths between frequently communicating nodes distributed across \( V \). In particular, the designed network should minimize the expected path length on \( V \) with respect to \( \mathcal{D} \), which is a basic measure of the

Further Reading

Static DAN

Demand-Aware Network Designs of Bounded Degree

Overview: Dynamic DAN

SpaylNet: Towards Locally Self-Adjusting Networks

Stefan Schmid1, Chen Avin2, Christian Scholské2, Michael Brandtke3, Bernard Hampl3, Zvi Lotker

Abstract This paper initiates the study of locally self-adjusting networks whose topology adapts dynamically to the actual demand, i.e., to the communication pattern \( \mathcal{D} \). This can be seen as a distributed generalization of the self-adjusting architectures introduced by Stone and Tanenbaum [22]. In contrast to these works, which specifically approach the design of distributed hash tables, we are interested in the design of a protocol that allows an arbitrary communication pattern to be distributed over a network. To achieve this, we study the problem of maintaining a list of nodes that are reachable from a given node (BSST), while supporting operations that efficiently trade off between the benefits and costs of self-adjusting networks. We present the SpaylNet algorithms and formally analyse its performance. Based on the algorithm, we introduce a novel technique that balances the dynamics of the network with the costs of maintaining the list of reachable nodes. The main contribution is the development of an efficient method for maintaining a list of reachable nodes in a distributed manner.

1 Introduction

In the 1990s, Stone and Tanenbaum [22] proposed an alternative new paradigm to design efficient Binary Search Tree (BST) datastructures, rather than optimizing traditional protocols such as

Static Optimality

ReNets: Toward Static Optimality in Self-Adjusting Networks

Chen Avin1, Stefan Schmid2
1 Ben Gurion University, Israel
2 University of Vienna, Austria

Abstract

This paper studies the design of self-adjusting networks whose topology dynamically adapts to the workload, in an online and demand-aware manner. This problem is motivated by emerging optical technologies which allow to reconfigure the datacenter topology at runtime. Our main contribution is ReNets, a self-adjusting network which maintains a balance between the benefits and costs of reconfigurations. In particular, we show that ReNets are statistically optimal for arbitrary sparse communication demands, i.e., perform at least as good as any fixed demand-aware network designed with a perfect knowledge of the future demand. Furthermore, ReNets provide compact and local routing, by leveraging ideas from self-adjusting datastructures.
Notion of self-adjusting networks opens a **large uncharted field** with many questions:

→ Metrics and algorithms: by how much can load be lowered, **energy** reduced, quality-of-service improved, etc. in demand-aware networks? Even for **route length** not clear!

→ How to model reconfiguration costs?

→ Impact on **other layers**?

**Future Work:**

Models, Metrics, Algos

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

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

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

Toward Demand-Aware Networking: A Theory for Self-Adjusting Networks (Editorial)
Chen Avin and Stefan Schmid.

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.

Distributed Self-Adjusting Tree Networks
Bruna Peres, Otavio Augusto de Oliveira Souza, Olga Goussevskaia, Chen Avin, and Stefan Schmid.

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.

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.

Characterizing the Algorithmic Complexity of Reconfigurable Data Center Architectures
Klaus-Tycho Foerster, Monia Ghobadi, and Stefan Schmid.