Self-Adjusting Networks: Vision, Solutions, Challenges
Stefan Schmid

“We cannot direct the wind, but we can adjust the sails.”
(Folklore)
It's a Great Time to Be a Networking Researcher!

Credits: George Varghese
It's a Great Time to Be a Networking Researcher!

Innovation

Credits: George Varghese

Enables and motivates self-adjusting networks!
It’s High Time!

Explosive Traffic

Datacenters ("hyper-scale")

Interconnecting networks: a critical infrastructure of our digital society.
It’s High Time!

Explosive Traffic

Datacenters ("hyper-scale")

Interconnecting networks: a critical infrastructure of our digital society.

Credits: Marco Chiesa
It’s High Time!

Reality vs Requirements

Today, dependability requirements stand in contrast with reality:

Countries disconnected

Google routing blunder sent Japan’s Internet dark on Friday
Another big BGP blunder
By Richard Chirgwin 27 Aug 2017 at 22:35

Last Friday, someone in Google fat-thumbed a border gateway protocol (BGP) advertisement and sent Japanese Internet traffic into a black hole.

The trouble began when The Chocolate Factory “leaked” a big route table to Verizon, the result of which was traffic from Japanese giants like NTT and KDDI was sent to Google on the expectation it would be treated as transit.

Passengers stranded

British Airways’ latest Total Inability To Support Upwardness of Planes* caused by Amadeus system outage
Stuck on the ground awaiting a load sheet? Here’s why
By Sasset F. 14 Jul 2018 at 11:54

Even 911 affected

Officials: Human error to blame in Minn. 911 outage
According to a press release, CenturyLink told department of public safety that human error by an employee of a third party vendor was to blame for the outage Aug 15, 2018

Even tech-savvy companies struggle:

Credits: Laurent Vanbever, Nate Foster
It’s High Time!

Reality vs Requirements

Today, dependability requirements stand in contrast with reality:

Countries disconnected

Data Centre › Networks

Google routing blunder sent Japan’s Internet dark on Friday

Another big BGP blunder

By Richard Chirgwin 27 Aug 2017 at 22:35

Last Friday, someone in Google fat-thumbed a border gateway protocol (BGP) advertisement and sent Japanese Internet traffic into a black hole.

The trouble began when The Chocolate Factory “leaked” a big route table to Verizon, the result of which was traffic from Japanese giants like NTT and KDDI was sent to Google on the expectation it would be treated as transit.

Passengers stranded

British Airways’ latest Total Inability To Support Upwardness of Planes* caused by Amadeus system outage

Stuck on the ground awaiting a load sheet? Here’s why

By Sarah Corkett 19 Jul 2018 at 11:54

Officials: Human error to blame in Minn. 911 outage

According to a press release, CenturyLink told department of public safety that human error by an employee of a third party vendor was to blame for the outage

Aug 15, 2018

Duluth News Tribune

*SNIP* Filed: Mpls — The Minnesota Department of Public Safety-Emergency Communication Networks division was told by its 911 provider that an Aug. 1 outage was caused by human error.

Even 911 affected

Even tech-savvy companies struggle:

Mainly: human errors!

Credits: Laurent Vanbever, Nate Foster
Agenda

Three Use Cases

Passau, Germany
Agenda

Three Use Cases

Formal methods as a tool!
Motivation: Complexity
Especially Under Failures (Policy Compliance)

Example: BGP in Microsoft datacenter

Credits: Ratul Mahajan
Motivation: Complexity
Especially Under Failures (Policy Compliance)

Example: BGP in Microsoft datacenter

Cluster with globally reachable services
Cluster with internally accessible services

Credits: Ratul Mahajan
Motivation: Complexity
Especially Under Failures (Policy Compliance)

Example: BGP in Microsoft datacenter

Internet

Datacenter

Cluster with globally reachable services
Cluster with internally accessible services

Credits: Ratul Mahajan
Motivation: Complexity
Especially Under Failures (Policy Compliance)

Example: BGP in Microsoft datacenter

What can go wrong?

Cluster with globally reachable services

Cluster with internally accessible services

Credits: Ratul Mahajan
Motivation: Complexity
Especially Under Failures (Policy Compliance)

Example: BGP in Microsoft datacenter

What can go wrong?

Credits: Ratul Mahajan
Motivation: Complexity
Especially Under Failures (Policy Compliance)

Example: BGP in Microsoft datacenter

If link (G,X) fails and traffic from G is rerouted via Y and C to X: X announces (does not block) G and H as it comes from C. (Note: BGP.)

What can go wrong?

Credits: Ratul Mahajan
Admin’s Responsibilities

Forwarding and failover rules

Reachability: Can traffic from ingress port A reach B?

Loop-freedom: Do forwarding rules imply loop-free routes?

Policy: Does traffic from A to B never go via C?

Waypoint enforcement: Is traffic from A to B always routed via a node C (e.g., an IDS)?
Admin’s Responsibilities

Reachable?

Reachability: Can traffic from ingress port A reach B?

Loop-freedom: Do forwarding rules imply loop-free routes?

Policy: Does traffic from A to B never go via C?

Waypoint enforcement: Is traffic from A to B always routed via a node C (e.g., an IDS)?
Admin’s Responsibilities

→ Reachability: Can traffic from ingress port A reach B?
→ Loop-freedom: Do forwarding rules imply loop-free routes?
Admin’s Responsibilities

→ Reachability: Can traffic from ingress port A reach B?

→ Loop-freedom: Do forwarding rules imply loop-free routes?

→ Policy: Does traffic from A to B never go via C?

E.g. NORDUnet: no traffic via Iceland (expensive!).
Admin’s Responsibilities

→ Reachability: Can traffic from ingress port A reach B?

→ Loop-freedom: Do forwarding rules imply loop-free routes?

→ Policy: Does traffic from A to B never go via C?

→ Waypoint enforcement: Is traffic from A to B always routed via a node C (e.g., an IDS)?
Admin’s Responsibilities

- Reachability: Can traffic from ingress port A reach B?
- Loop-freedom: Do forwarding rules imply loop-free routes?
- Policy: Does traffic from A to B never go via C?
- Waypoint enforcement: Is traffic from A to B always routed via a node C (e.g., an IDS)?

... and everything even under failures?!
A Modern Approach: Automated Whatif Analysis

Router configurations (Cisco, Juniper, etc.)

Formal language which supports automated analysis
A Modern Approach: Automated Whatif Analysis

Router configurations (Cisco, Juniper, etc.)

Formal language which supports automated analysis

Compilation

On request or regularly.

Interpretation

pX ⇒ qXX
pX ⇒ qYX
qY ⇒ rYY
rY ⇒ r
rX ⇒ pX
A Modern Approach: Automated Whatif Analysis

Router configurations (Cisco, Juniper, etc.)

Compilation
On request or regularly.
Fix/synthesize

Formal language which supports automated analysis

<table>
<thead>
<tr>
<th>FF</th>
<th>In1</th>
<th>In2</th>
<th>Out-1</th>
<th>op</th>
</tr>
</thead>
<tbody>
<tr>
<td>rX</td>
<td>1</td>
<td>1</td>
<td>pX</td>
<td>push(30)</td>
</tr>
<tr>
<td>rY</td>
<td>1</td>
<td>1</td>
<td>qY</td>
<td>push(30)</td>
</tr>
<tr>
<td>rZ</td>
<td>1</td>
<td>1</td>
<td>rZ</td>
<td>push(40)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Local FFT</th>
<th>Out-1</th>
<th>In-Label</th>
<th>Out-1</th>
<th>op</th>
</tr>
</thead>
<tbody>
<tr>
<td>pX ⇒ qXX</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pX ⇒ qYX</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>qY ⇒ rYY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rY ⇒ r</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rX ⇒ pX</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A Modern Approach: Automated Whatif Analysis

Router configurations (Cisco, Juniper, etc.)

Formal language which supports automated analysis

Compilation

Fix/synthesize

On request or regularly.

What if?!

pX ⇒ qXX
pX ⇒ qYX
qY ⇒ rYY
rY ⇒ r
rX ⇒ pX

Router configurations

(Cisco, Juniper, etc.)

Formal language supports automated analysis
Challenge:

Hard Even for Computers?

→ NORDUnet: provider for Nordic countries

→ 24 MPLS routers, running Juniper OS, >30,000 labels!
Challenge: Hard Even for Computers?

→ NORDUnet: provider for Nordic countries

→ 24 MPLS routers, running Juniper OS, >30,000 labels!

For specific networks such as MPLS: feasible and fast! Tools such as P-Rex or AalWiNes do it in secs for MPLS: reduction to automata theory, polynomial-time.
Challenge:
Hard Even for Computers?

→ NORDUnet: provider for Nordic countries

→ 24 MPLS routers, running Juniper OS, >30,000 labels!

For specific networks such as MPLS: feasible and fast!
Tools such as P-Rex or AalWiNes do it in secs for MPLS: reduction to automata theory, polynomial-time.
But general networks more challenging.
Fixing & Synthesis: Harder

• Approaches: Petri games, Stackelberg games, UPPAAL Stratego…

• But synthesis slower than verification
Fixing & Synthesis: Harder

- Approaches: Petri games, Stackelberg games, UPPAAL Stratego...

- But synthesis slower than verification

- An opportunity for using AI!

- Ideally AI+FM: guarantees from formal methods, performance from AI

- For example: synthesize with AI then verify with formal methods

- Examples: DeepMPLS, DeepBGP, ...
Fixing & Synthesis: Harder

→ Approaches: *Petri games*, Stackelberg games, UPPAAL Stratego…

→ But *synthesis slower* than verification

→ An opportunity for using AI!

→ *Ideally AI+FM*: guarantees from formal methods, performance from AI

→ For example: synthesize with AI then verify with formal methods

→ Examples: DeepMPLS, DeepBGP, …
Fixing & Synthesis: Harder

→ Approaches: Petri games, Stackelberg games, UPPAAL Stratego…

→ But synthesis slower than verification

→ An opportunity for using AI!

→ Ideally AI+FM: guarantees from formal methods, performance from AI

→ For example: synthesize with AI then verify with formal methods

→ Examples: DeepMPLS, DeepBGP, …

... and what about quantitative properties?
A Possible Starting Point: The AalWiNes Tool

Online demo: [https://demo.aalwines.cs.aau.dk/](https://demo.aalwines.cs.aau.dk/)
Source code: [https://github.com/DEIS-Tools/AalWiNes](https://github.com/DEIS-Tools/AalWiNes)
Agenda

Three Use Cases

Passau, Germany
Agenda
Three Use Cases

On lower layers!
Recall: explosive growth of demand

Problem: network equipment reaching capacity limits
  → Transistor density rates stalling
  → “End of Moore’s Law in networking”

Hence: more equipment, larger networks

Resource intensive and: inefficient

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.
A Vision
Flexible and Demand-Aware Topologies
A Vision
Flexible and Demand-Aware Topologies

e.g., mirrors
new flexible interconnect
A 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
A Vision
Flexible and Demand-Aware Topologies

Matches demand

demand matrix:

e.g., mirrors
new flexible interconnect
A Vision
Flexible and Demand-Aware Topologies

new demand: e.g., mirrors

new flexible interconnect
A Vision
Flexible and Demand-Aware Topologies

Matches demand

new demand:

e.g., mirrors
new flexible interconnect
A Vision
Flexible and Demand-Aware Topologies

Demand-Aware Networks

e.g., mirrors
new flexible interconnect

demand:

1 2 3 4 5 6 7 8

1 2 3 4 5 6 7 8
The Motivation

Much Structure in the Demand

Empirical studies:

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

traffic **bursty** over time

**Hypothesis**: this can be exploited.
Sounds Crazy?
Emerging Enabling Technology.

H2020:
“Photonics one of only five key enabling technologies for future prosperity.”

US National Research Council:
“Photons are the new Electrons.”
Spectrum of prototypes
→ Different sizes, different reconfiguration times
→ From our last years’ ACM SIGCOMM workshop OptSys
Example

Optical Circuit Switch

→ Optical Circuit Switch rapid adaption of physical layer
  → Based on rotating mirrors

Optical Circuit Switch

By Nathan Farrington, SIGCOMM 2010
The Big Picture

Flexibility

Self-Adjusting Networks

Efficiency

Now is the time!
Unique Position
Demand-Aware, Self-Adjusting Systems

Everywhere, but mainly in software

- Algorithmic trading
- Recommender systems
- Neural networks

Our focus: in hardware
Question 1:
How to Quantify such “Structure” in the Demand?
An Information-Theoretic Approach

Complexity Map

Our approach: iterative randomization and compression of trace to identify dimensions of structure.
An Information-Theoretic Approach

Complexity Map

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

Different structures!
An Information-Theoretic Approach

Complexity Map

Potential gain!

Our approach: iterative randomization and compression of trace to identify dimensions of structure.
An Information-Theoretic Approach

Complexity Map

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

Griner et al., Sigmetrics 2020
Question 2:

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

A first insight: entropy of the demand.
Interesting Perspective:

Connection to Datastructures

Traditional BST  Demand-aware BST  Self-adjusting BST

More structure: improved access cost
Interesting Perspective:

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
Interesting Perspective:

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?
Interesting Perspective:

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?
Interesting Perspective:

Connection to Datastructures & Coding

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

More than an analogy!

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

Results, e.g.: Demand-aware networks of asymptotically optimal route lengths.

Reduced expected route lengths!
Interesting Perspective:
Connection to Datastructures & Coding

Reduced expected route lengths!

Avin et al., SIGCOMM CCR 2018.
Agenda

Three Use Cases
Another Benefit of Automation:

More Adaptive Operation

→ Automation and programmability: enables more adaptable networks

→ Attractive for:
  → Fine-grained traffic engineering (e.g., at Google)
  → Accounting for changes in the demand (spatio-temporal structure)
  → Security policy changes
  → Service relocation
  → Maintenance work
  → Link/node failures
Another Benefit of Automation:

More Adaptive Operation

Enabled by SDN, it has become „easy“ to quickly change route to blue route.
Another Benefit of Automation:

More Adaptive Operation

But still need clever algorithms! Updates are asynchronous, may lead to temporal inconsistencies.
Another Benefit of Automation:

More Adaptive Operation

But still need clever algorithms! Updates are asynchronous, may lead to temporal inconsistencies.
Again: Formal Methods for Self-Adjusting Updates

SDN Controller Platform

untrusted hosts  \rightarrow  trusted hosts

Asynchronous: loop!

Vision: self-adjusting networks could synthesize even their algorithms! „Ex machina“: e.g., parametrized.
Examples: NetSynth, Latte

- Already “in the making”!
- NetSynth (PLDI’15): supports any LTL property and hence operator preferences. Then: standard framework to synthesize schedule.
- Latte (PER’20): fast Petri net model and synthesis
- Example: Gadget

If token up here: packets go old path
If token down here: switch updated to new path
Examples: NetSynth, Latte

→ Already “in the making”!

→ NetSynth (PLDI’15): supports any LTL property and hence operator preferences. Then: standard framework to synthesize schedule.

→ Latte (PER’20): fast Petri net model and synthesis

→ Example: Gadget

If token up here: packets go old path

If token down here: switch updated to new path
Examples: NetSynth, Latte

→ Already “in the making”!

→ NetSynth (PLDI’15): supports any LTL property and hence operator preferences. Then: standard framework to synthesize schedule.

→ Latte (PER’20): fast Petri net model and synthesis

→ Example: Gadget

May be enhanced with AI.
Challenges of Self-Adjusting Networks
Challenges

→ **Performance** of formal methods? Opportunity: *algorithm engineering*

→ Use of **AI**: to speed up synthesis and deal with complexity?

→ Limitations of automation: can networks detect themselves, when the need “help from the operator”?

→ **Data**: How to learn about and/or predict demand? Telemetry?

→ Programmability vs **security**?
Enabling technology like SDN often deployed "in software"
E.g., virtual switches in datacenters
Virtual switches reside in the server’s virtualization layer (e.g., Xen’s Dom0). Goal: provide connectivity and isolation.
Complexity: Parsing

Parser directly faces attacker and vSwitch runs with high security privileges.
Parsing must be fast!

Unified packet parser is fast, but complex.
Bears Risks!
Bears Risks!
Bears Risks!

Malformed packet: crashes virtual switch
Bears Risks!
Conclusion

→ A vision: self-adjusting networks

→ Example 1: policy-compliant networks
  → self-verifying
  → self-repairing

→ Example 2: demand-aware topologies

→ On both fronts: tip of the iceberg!

→ E.g., self-adjusting networks further supported by telemetry (data) and AI (e.g., prediction)

Thank you!
References

AalWiNes: A Fast and Quantitative What-If Analysis Tool for MPLS Networks
Peter Gjøl Jensen, Morten Konggaard, Dan Kristiansen, Stefan Schmid, Bernhard Clemens Schrenk, and Jiri Srba.
16th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT), Barcelona, Spain, December 2020.

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

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

Cerberus: The Power of Choices in Datacenter Topology Design (A Throughput Perspective)
Chen Griner, Johannes Zerwas, Andreas Blenk, Manya Ghobadi, Stefan Schmid, and Chen Avin.

Latte: Improving the Latency of Transiently Consistent Network Update Schedules
Mark Glavind, Niels Christensen, Jiri Srba, and Stefan Schmid.

Taking Control of SDN-based Cloud Systems via the Data Plane (Best Paper Award)
ACM Symposium on SDN Research (SOSR), Los Angeles, California, USA, March 2018.
Case Study: MPLS Networks

Original Routing
Case Study:
MPLS Networks

One failure: push 30: route around \((v_2, v_3)\)
Case Study:
MPLS Networks

If \((v_2, v_3)\) failed, push 30 and forward to \(v_6\).

Normal swap

**One failure:** push 30: route around \((v_2, v_3)\)
Case Study:

MPLS Networks

Original Routing

One failure:
push 30: route around \((v_2, v_3)\)

Two failures:
first push 30: route around \((v_2, v_3)\)

Push recursively
40: route around \((v_2, v_6)\)
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

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”

Original ➔ Randomize rows ➔ Uniform

Increasing complexity (systematically randomized)

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

Information-Theoretic Approach

“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!
Bonus Material

Hogwarts Stair
Golden Gate Zipper
Reconfigurable Optical Networks Will Move Supercomputer Data 100X Faster

Newly designed HPC network cards and software that reshapes topologies on-the-fly will be key to success

By Michelle Hampson