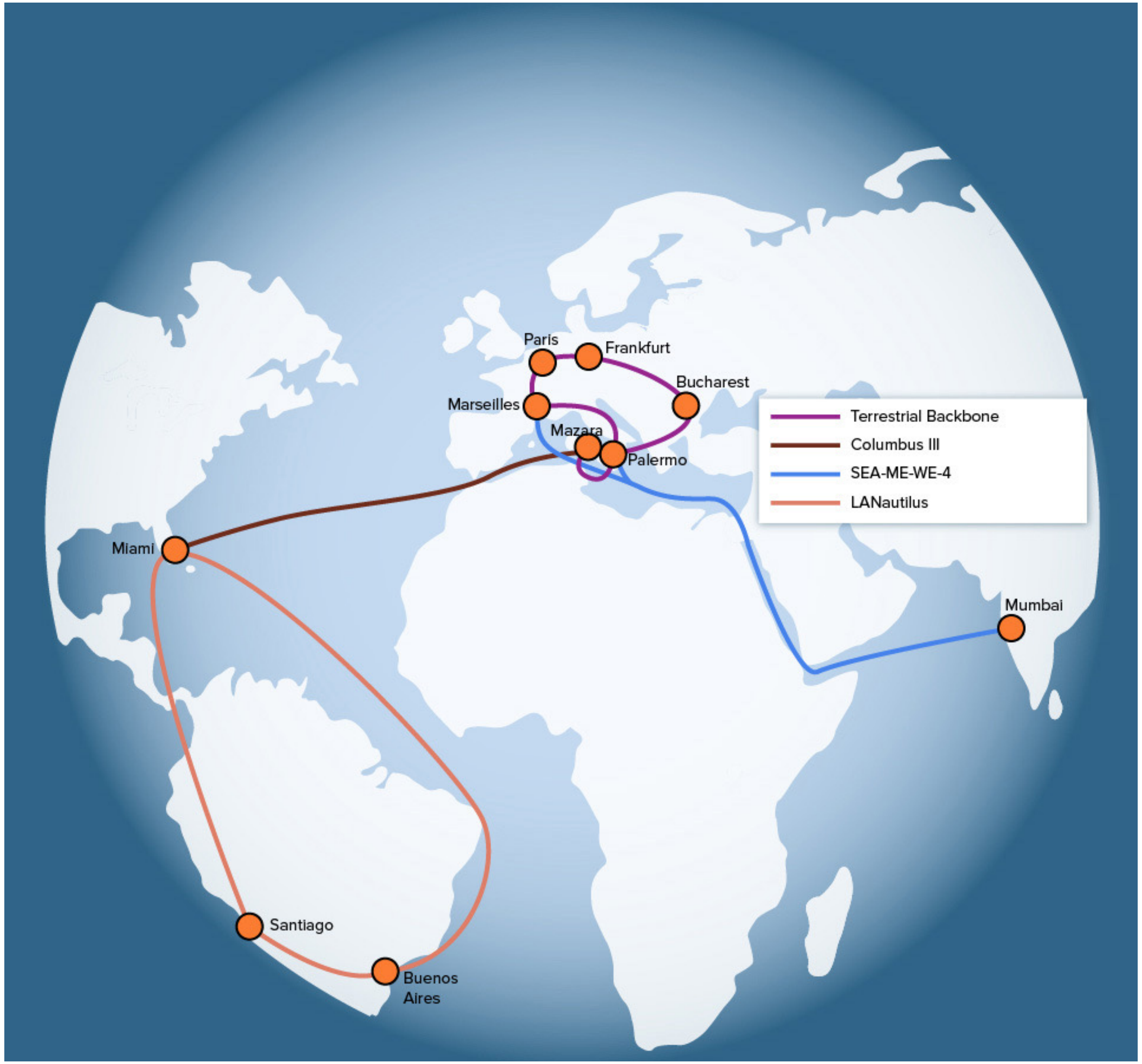
Verification

Networks are complex

- Many control plane protocols
    - Protocols interact in complex ways, cascading effects
- Protocols must often work across administrative boundaries
- Significant outages often due to avoidable reasons
    - Human errors cause >50% of outages
- Responding to physical connectivity disruptions
- Network is in a constant state of change

Physical faults

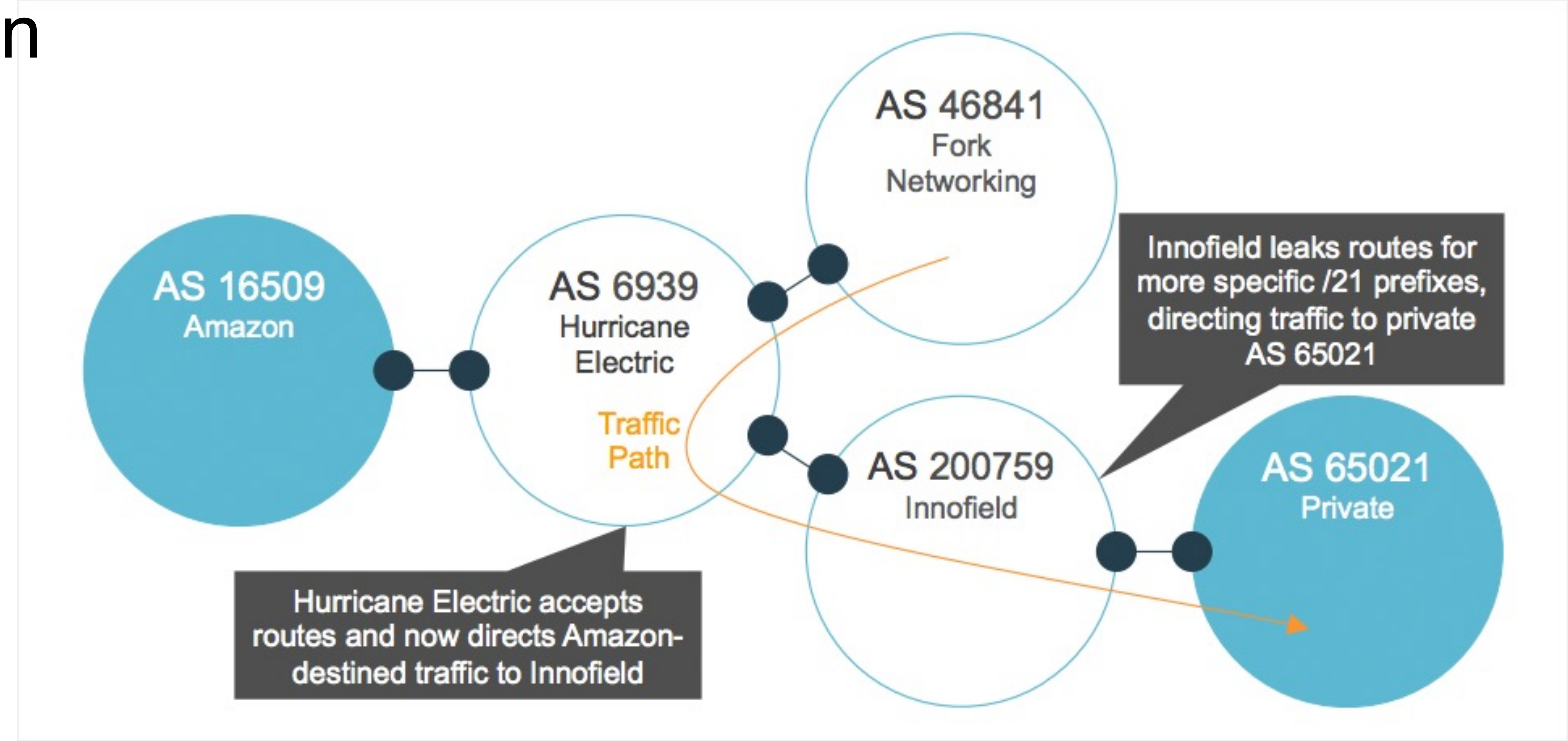
- Single cable fault or break can take out multiple ISP paths
- Cascading effects due to load on other links
  - New inter-dom paths taken
  - Peering points overloaded
  - Drop traffic worldwide



https://www.thousandeyes.com/blog/smw-4-cable-fault-ripple-effects-across-networks

Route misconfigurations

- Leaks: e.g., ISP announces more specific routes to a destination
  - Prefix “hijacking”
- Likely to be misconfigurations
  - e.g., Youtube08
- But can also be deliberate: MITM
  - E.g., Belarus



https://blog.thousandeyes.com/nanog-68-decoding-performance-data-internet-outages/

Some operator requirements

- Know answers to simple questions
    - Can A talk to B?
        - Reachability
        - A and B can be hosts, IP prefixes, “slices”
    - Are there loops, blackholes? Do slices leak?

- Know the effects of a change, preferably before it happens
    - What-if analyses
    - Link failures, protocol messages accepted from peers

- Answer these Qs fast to keep up with change in the network

Verification (software)

- For a program P, property X, input I, asking:
  - Will P satisfy X on all inputs I?
  - For what inputs I will P satisfy property X?
- Example:
  - Can a program f(x) := return x + 1 guarantee output f(x) > x?
  - Does g(x, y) := return x/y execute without exceptions always?
- Related problem: Synthesis
  - Can I generate a program P that satisfies X on all inputs I?
- Example:
  - Using binary & arithmetic operators alone (+, -, &, …) produce a program that determines whether input (int) x is a power of 2

Network Verification

SDN verification

Program + data plane should satisfy desired properties

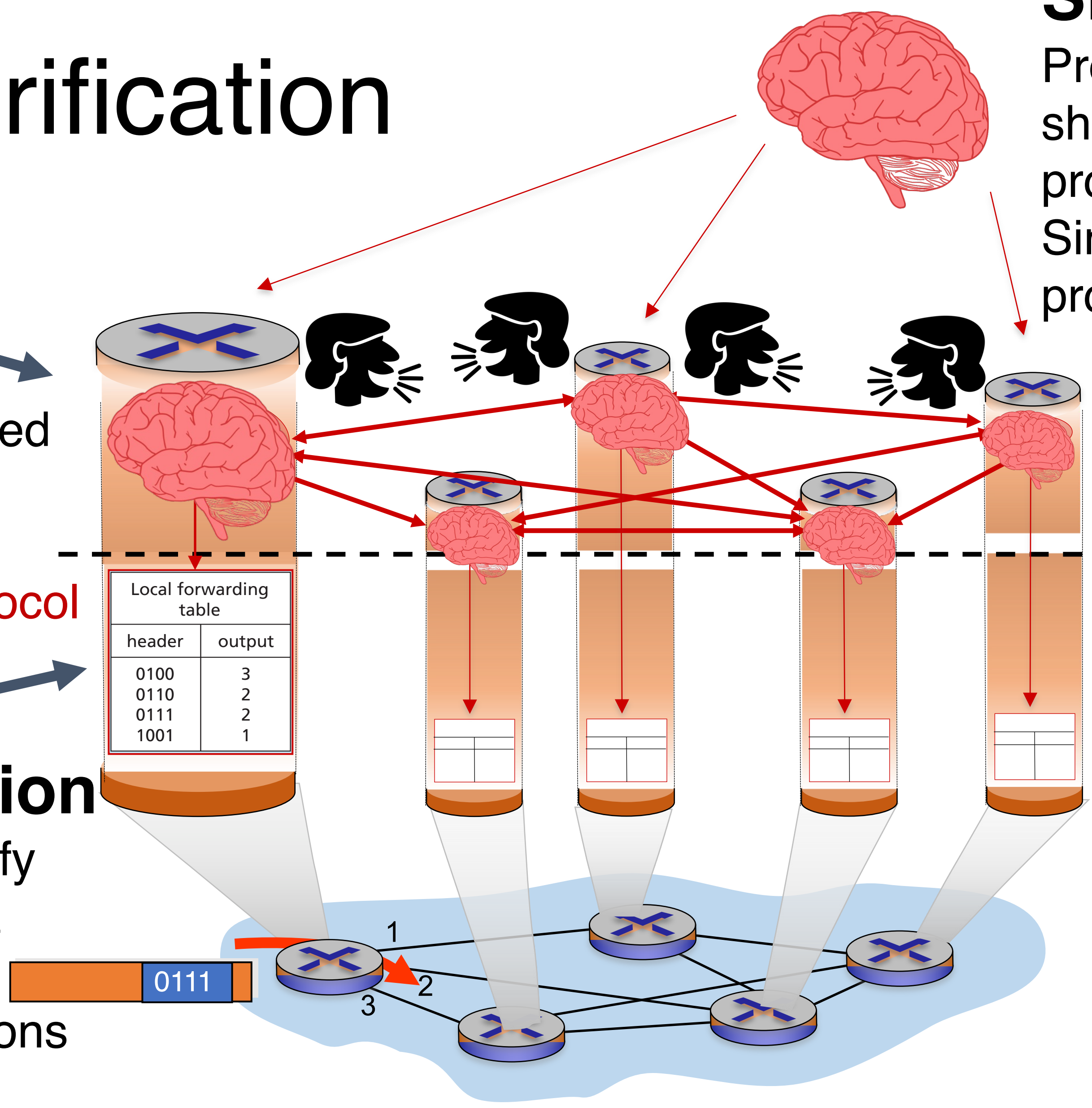
Single box state + program

Control plane verification

Do routing protocol configurations result in desired data plane properties?

Compute fixed points

Consider failures, route protocol message injections



Data plane verification

Do the devices together satisfy desired properties in the data plane?

Compute logical representations

Loop-free, blackhole-free, waypoints

Decision Procedure: An efficient algorithm that answers yes/no

Ask the question under assumptions about network change:
static, incremental, or dynamic

for all M, does N satisfy P?

Sequence of messages:
Packets,
Routing protocol
Link failures

Network representation:
Data plane
Control plane
Need good abstractions

Property of interest:
Loop freedom
Blackholes
Waypoints
Equivalence

A simple example: Modeling firewall rules

- Assume packets just have 2 bits; there are only 2 ports
- Firewall config: 10 -> fwd(2); x1 -> fwd(1). All others dropped
- Boolean representation of the network:
  - N: (d1 & ~d0) | ((d1 | ~d1) & d0)
- Property: only the packets from 00 are dropped
  - P: (~d1 & ~d0)
- Messages (M): all combinations of Boolean variables d0, d1
- Verification question: for all d0, d1, is formula P <=> ~N valid? i.e.,
  - Is P <=> N a tautology?
- Decision procedure: SAT solver

Verification, testing, synthesis, eq checks

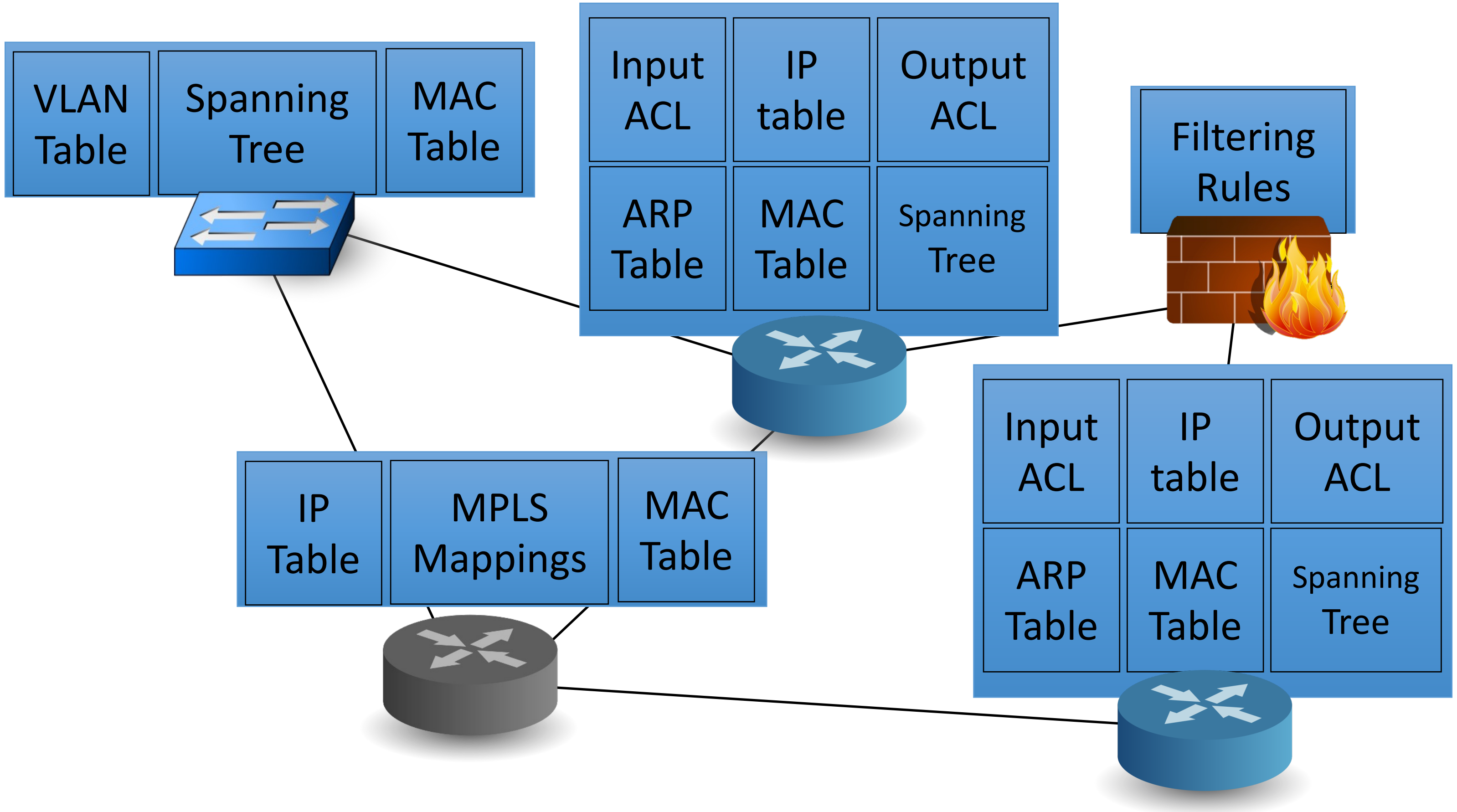
- Verification: for all M, does N satisfy P?
- Testing: For the given M, does N satisfy P?
- Synthesis: Given P, can you produce an N that satisfies it
  - For a given set of M? (including for all M)
- Let N' be another network representation
- Equivalence checking: For all M, do N and N' behave in the same way with respect to P?, i.e.,
  - i.e., either both satisfy P or both violate it

Header Space Analysis

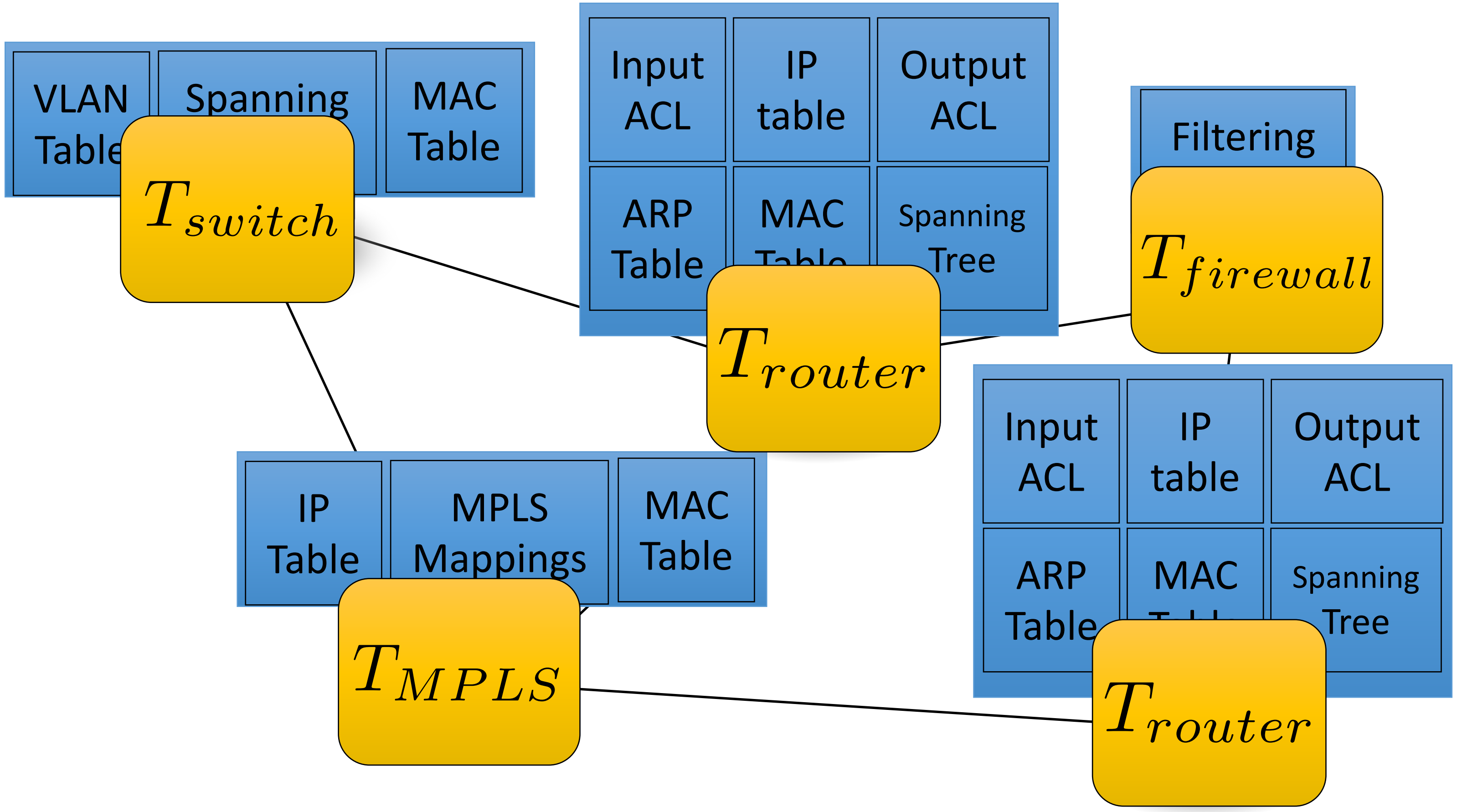
Ack: Thanks to slides from Peyman Kazemian!

https://web.cs.ucla.edu/~varghese/NETWORK_VERIFICATION_COURSE/Lecture4.pptm

Abstracting across devices

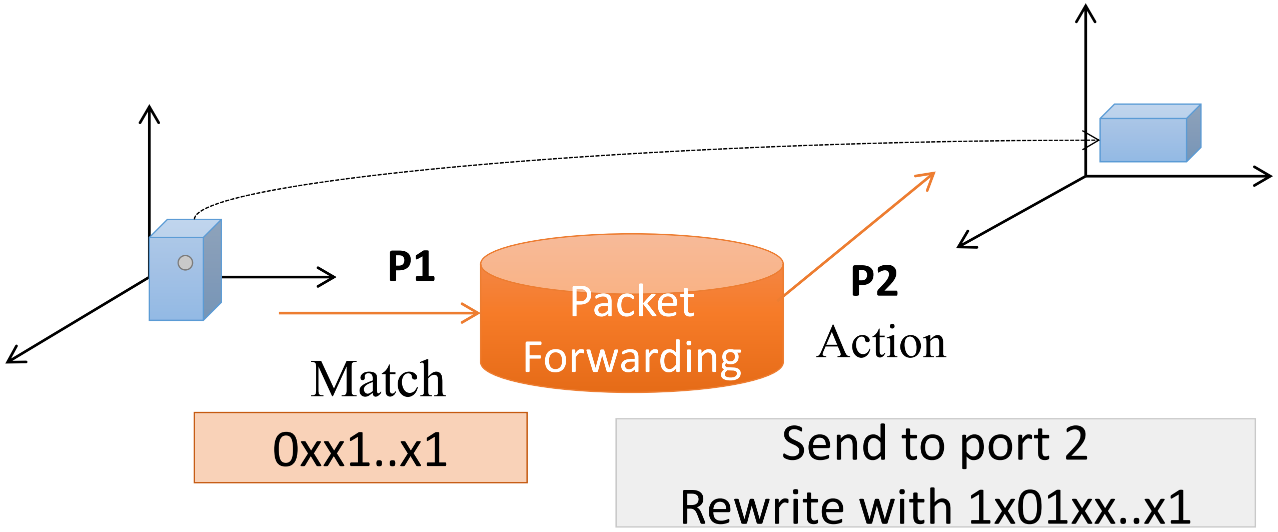


Vision for Network Verification



Networks as geometric transformers

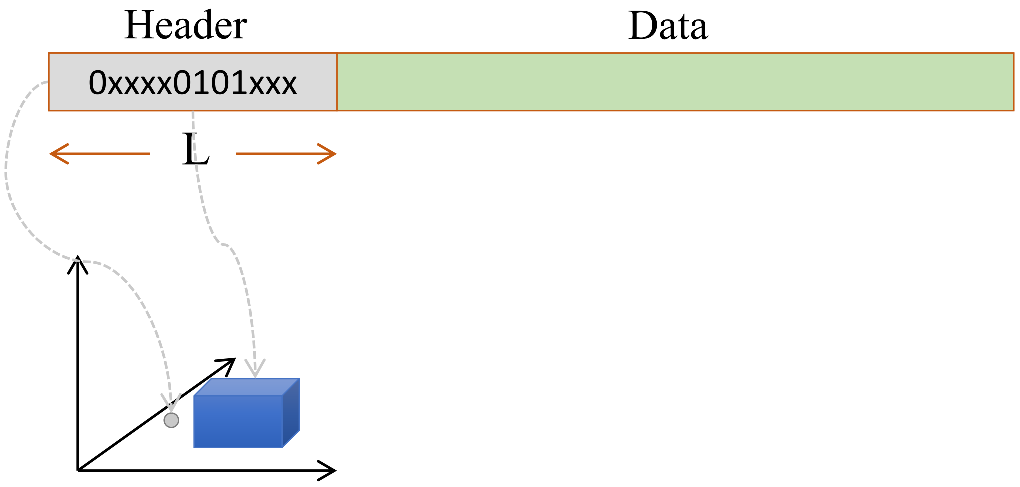
- Model header as point in high dimensional space and all networking boxes as transformers of header space



MATHEMATICAL FRAMEWORK TO REASON ABOUT WHICH SET OF POINTS ENTERING CAN EXIT NETWORK

Header Space Framework

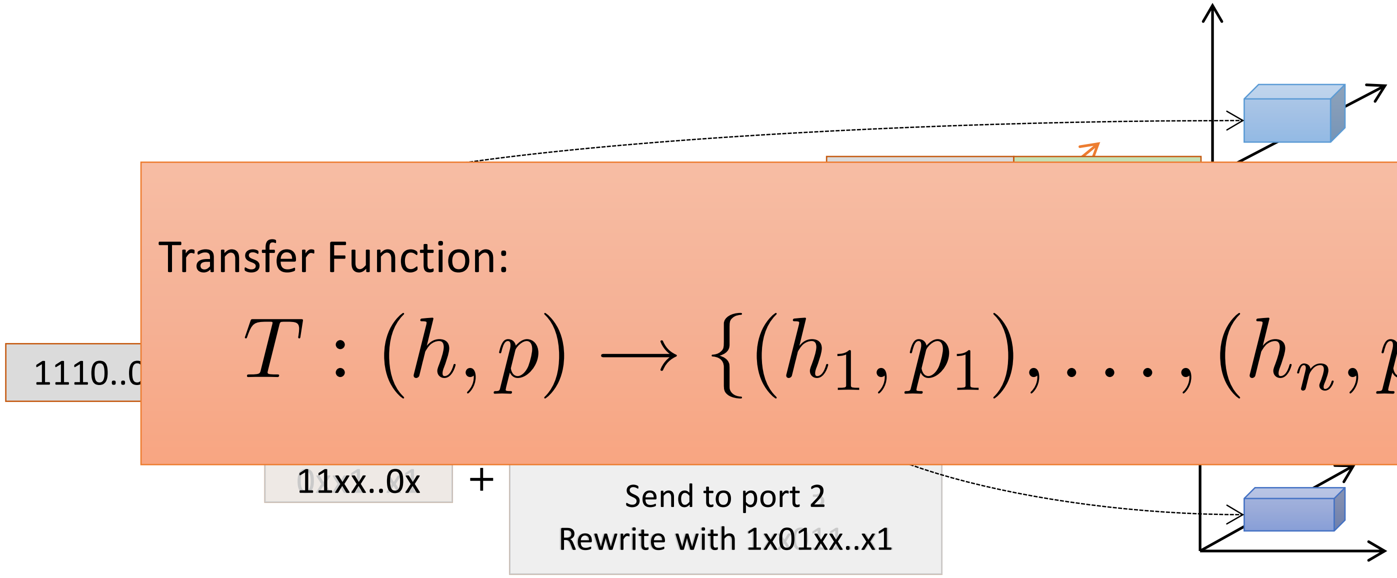
- Step 1 - Model a packet, based on its header bits, as a point in $\{0,1\}^L$ space – The Header Space



- Sets of packets (ternary expressions) are hypercubes in $\{0,1\}^L$ space

Header Space Framework

- Step 2 – Model all networking boxes as transformer of header space

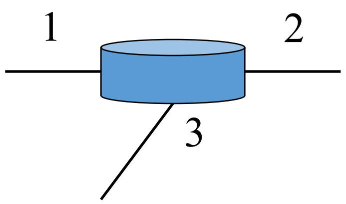


Transfer Function:

$$T : (h, p) \rightarrow \{(h_1, p_1), \ldots, (h_n, p_n)\}$$

Transfer Function Example

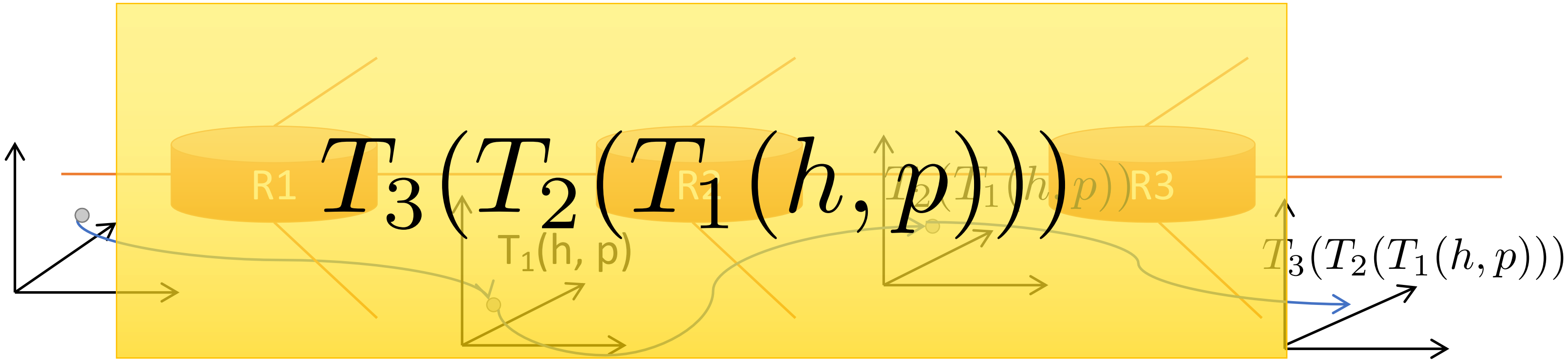
- IPv4 Router – Forwarding Behavior
  - 172.24.74.x Port1
  - 172.24.128.x Port2
  - 171.67.x.x Port3



$$T(h, p) = \begin{cases} (h,1) & \text{if } dst\_ip(h) = 172.24.74.x \\ (h,2) & \text{if } dst\_ip(h) = 172.24.128.x \\ (h,3) & \text{if } dst\_ip(h) = 171.67.x.x \end{cases}$$

Composing Transfer Functions

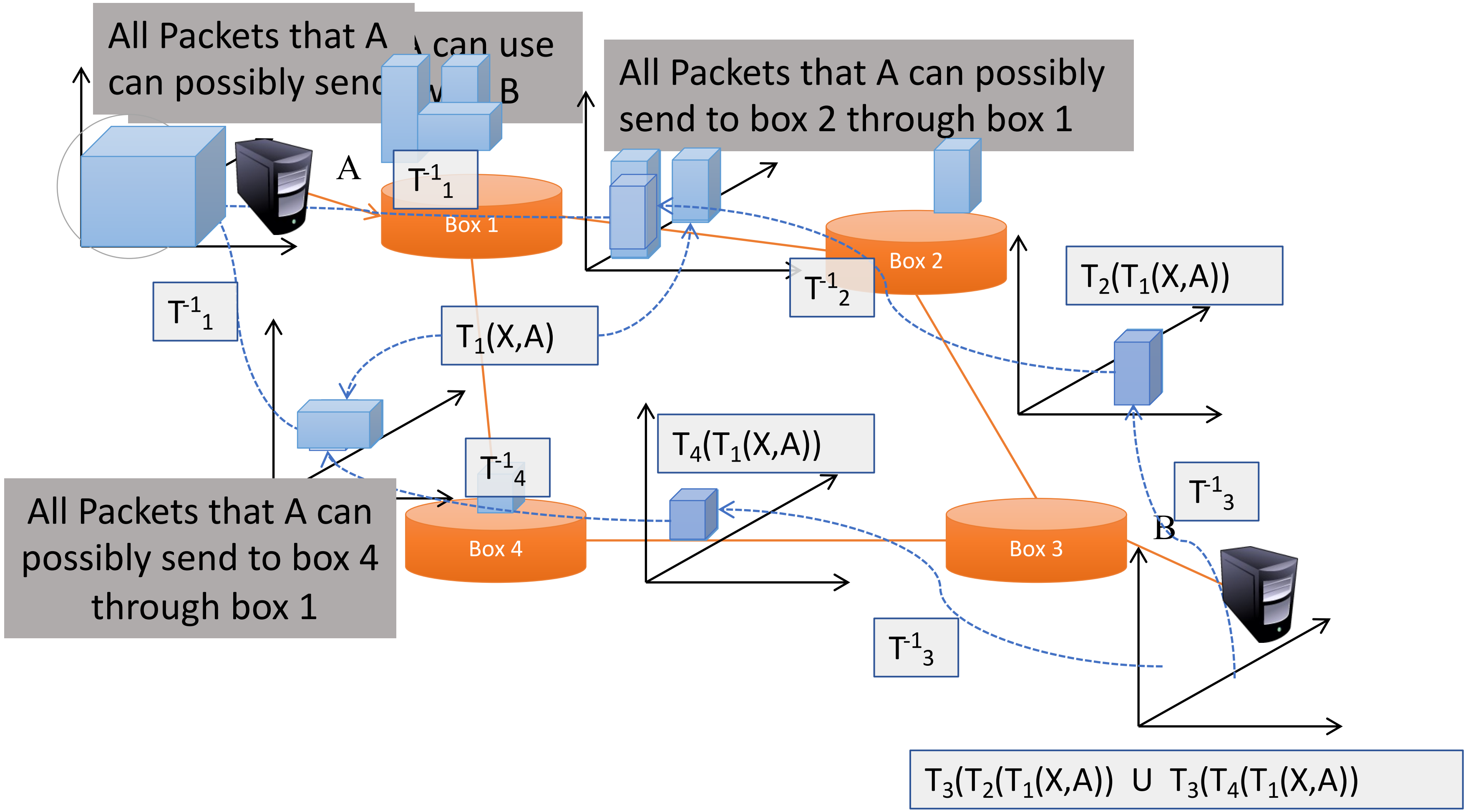
- By composing transfer functions, we can find the end to end behavior of networks.



Header Space Framework

- Step 3- Header Space Set Algebra.
  - Intersection
  - Complementation
  - Difference
  - Check subset and equality condition.
- Every region of Header Space, can be described by union of Wildcard Expressions. (example: 10xx U 011x)
- **Goal**: do set operations on wildcard expressions.

Computing Reachability



PROPAGATE MILLIONS OF SETS OF HEADERS IF DONE NAIVELY

Computer Networks

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Outro

Computer Networks

- Abstractions and algorithms for communication among interconnected machines
  - End to end and per-node
- A research experiment that escaped the lab and became fundamental to the world as we know it
- Architecture; transport; network data plane; network control plane; verification

What next?

Thanks, and all the best!

- Live with deeper appreciation of the Internet and networking
  - Does more bandwidth matter? Why do Internet apps run slowly? Why am I seeing outages in connectivity?
- Put your programming knowledge to good use
  - Significant tech work builds on the fundamentals in this course
  - Build, debug, optimize networked applications
  - Principles to organize communication across machines
- Go deeper
  - Research projects, independent study, theses, …