VeriFlow: Verifying Network-Wide Invariants in Real Time

Krystle Bates
Yenming Chen

ILLINOIS INSTITUTE OF TECHNOLOGY
Paul V. Galvin Library

Agenda

- Introduction
- VeriFlow
- Experiments
- Comparison: Header Space Analysis (HSA)
- Lessons Learned
- Conclusion & Discussion

Introduction

SDN's Refresh

- Software Defined Network (SDN) allows the use of programming network devices

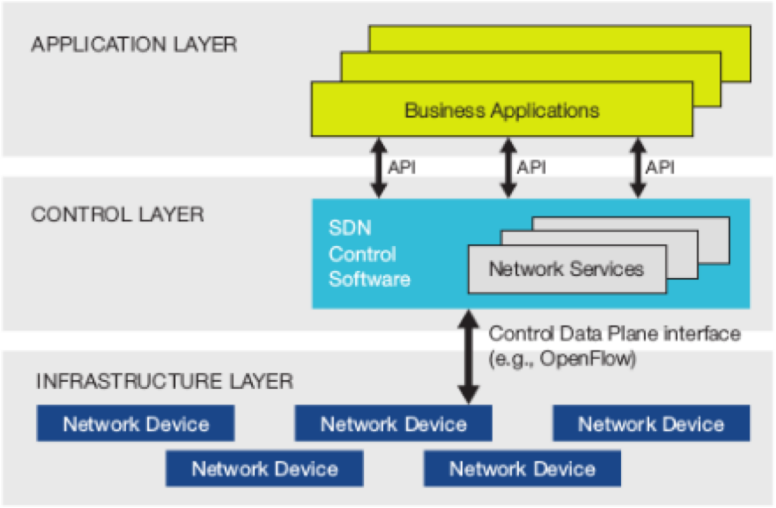


Fig. 1: Software Defined Networking Architecture [1]

Challenges of SDNs

- A logically-centralized network applications allow bugs occur due to the increases in software complexities.
- The use of multiple applications or user's ability to program the same physical network simultaneously, could result in conflicting rules.

Overview Approach

- Uses real-time data plane verification.
- Has a standardized and open interface to read and write the data plane of network devices and a centralized device can run code and is responsible for transmitting commands to network devices.

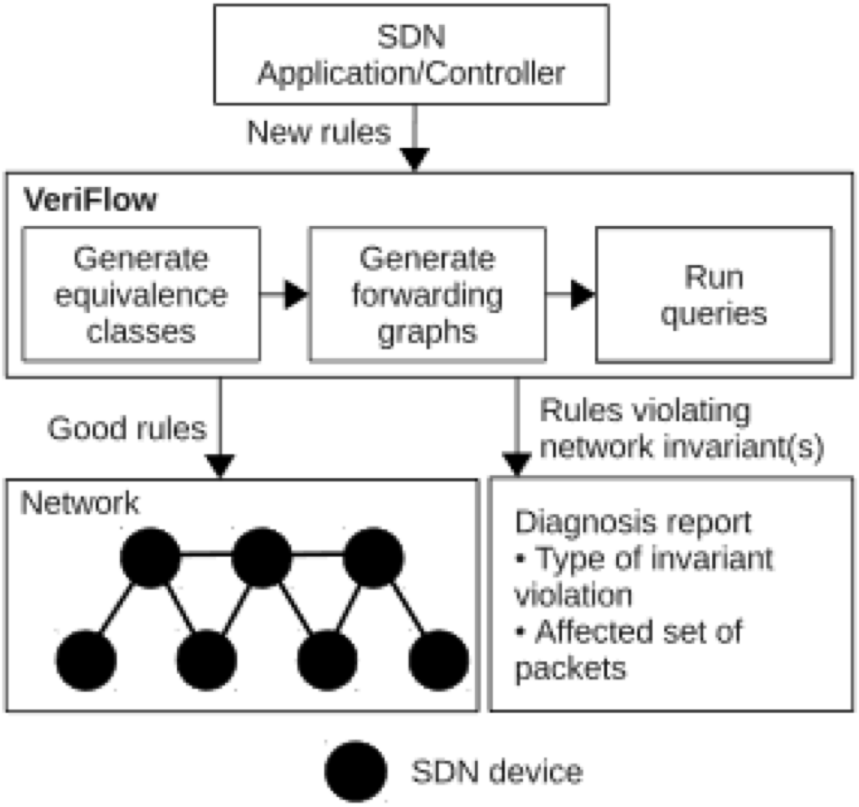
Challenges

- Obtaining real time view of network
- Verification speed

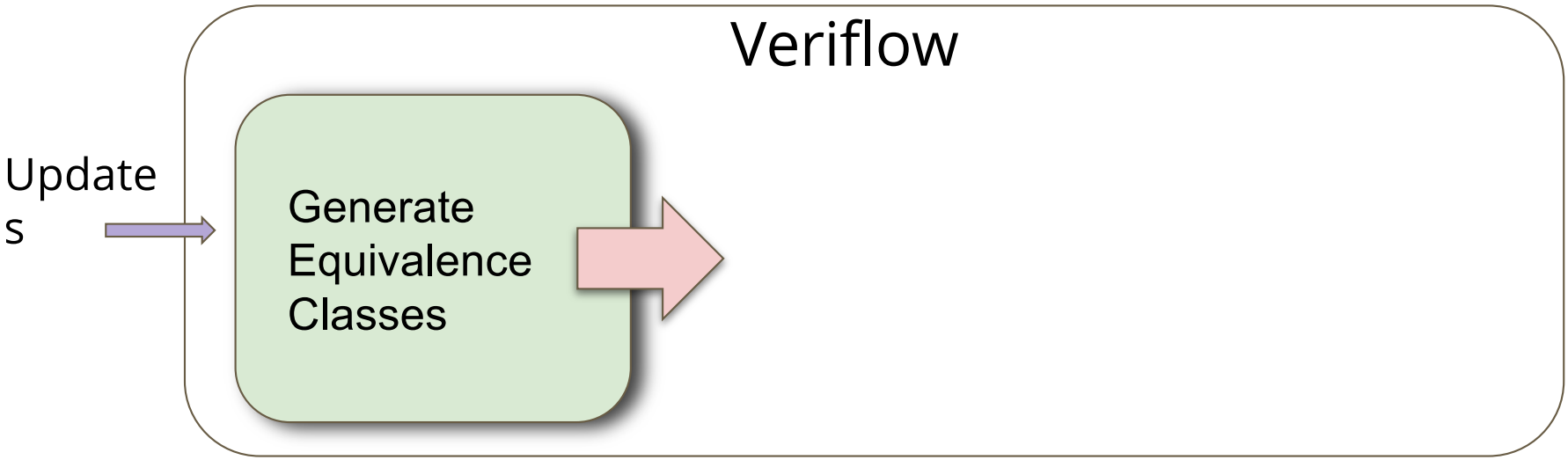
VeriFlow

Structure of VeriFlow

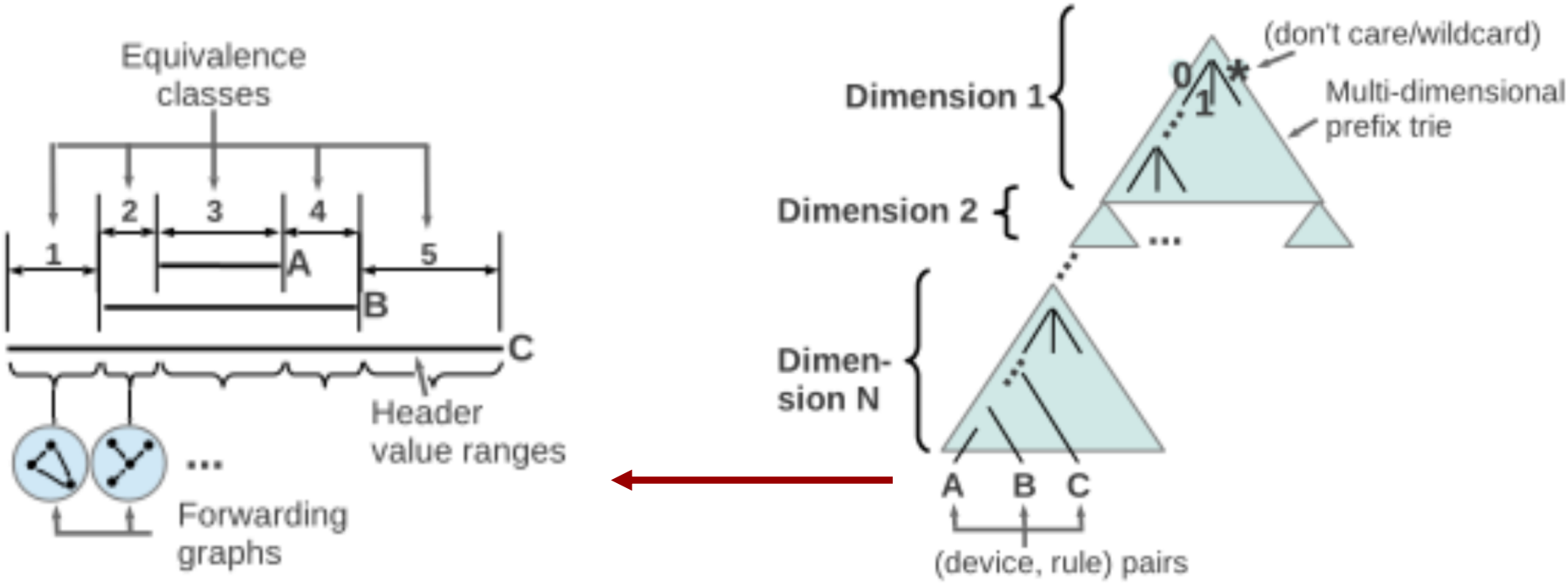
- Real - Time analysis
- Function by
  - Dynamic Monitor
  - Model Behavior
  - Custom Algorithms for Error Detection



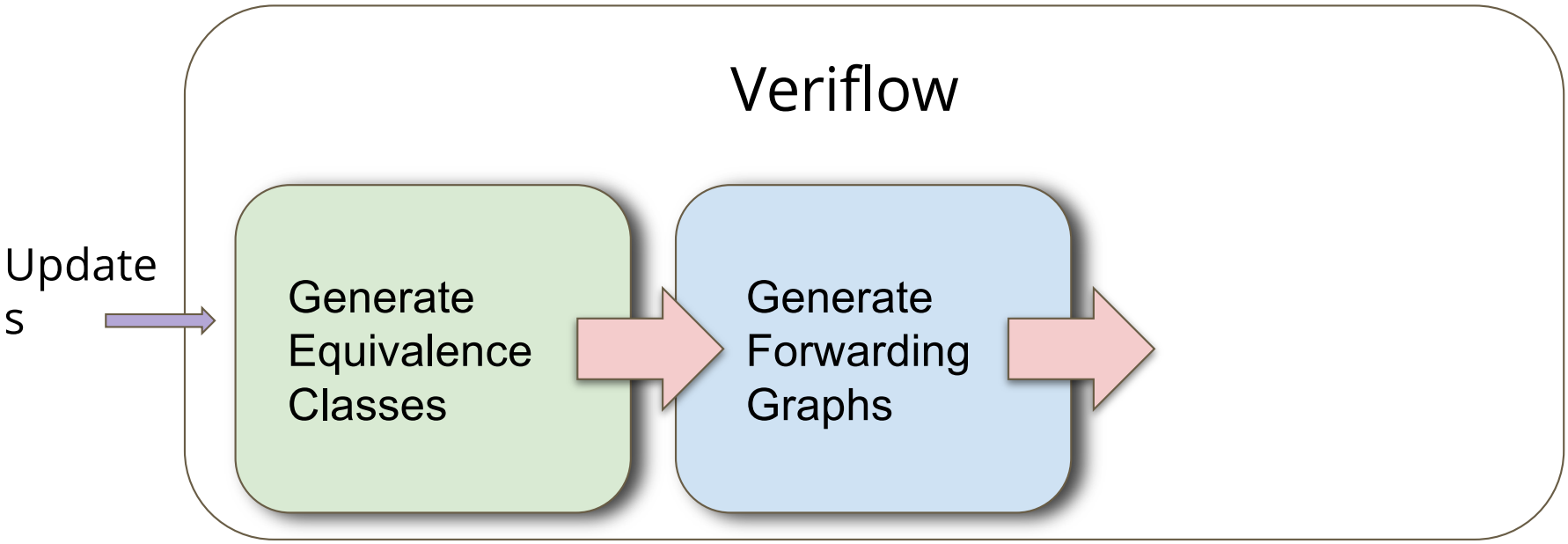
1. Limit the Search Space



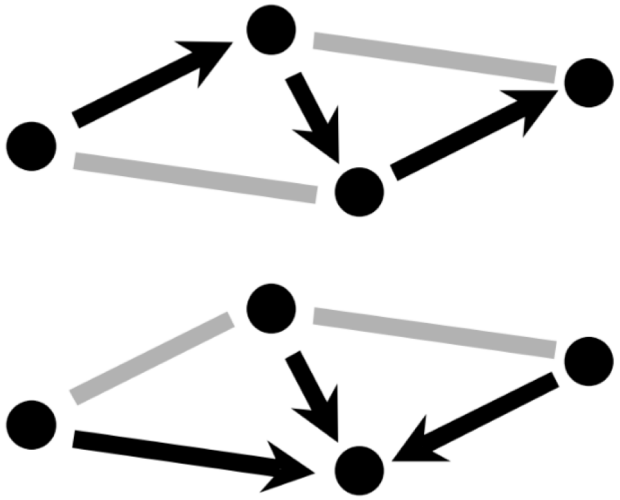
Computing Equivalence Classes (EC)

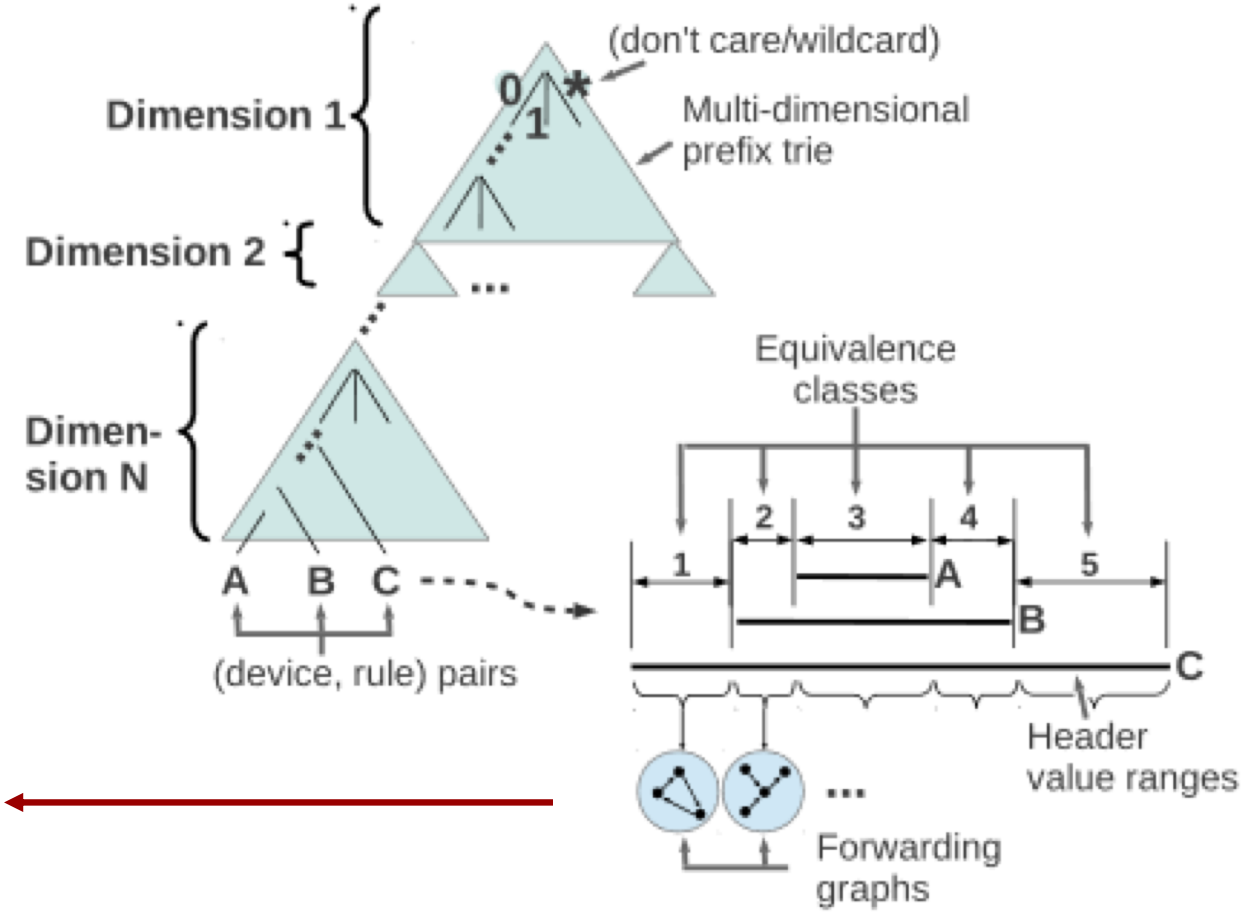


2. Represent Forwarding Behavior

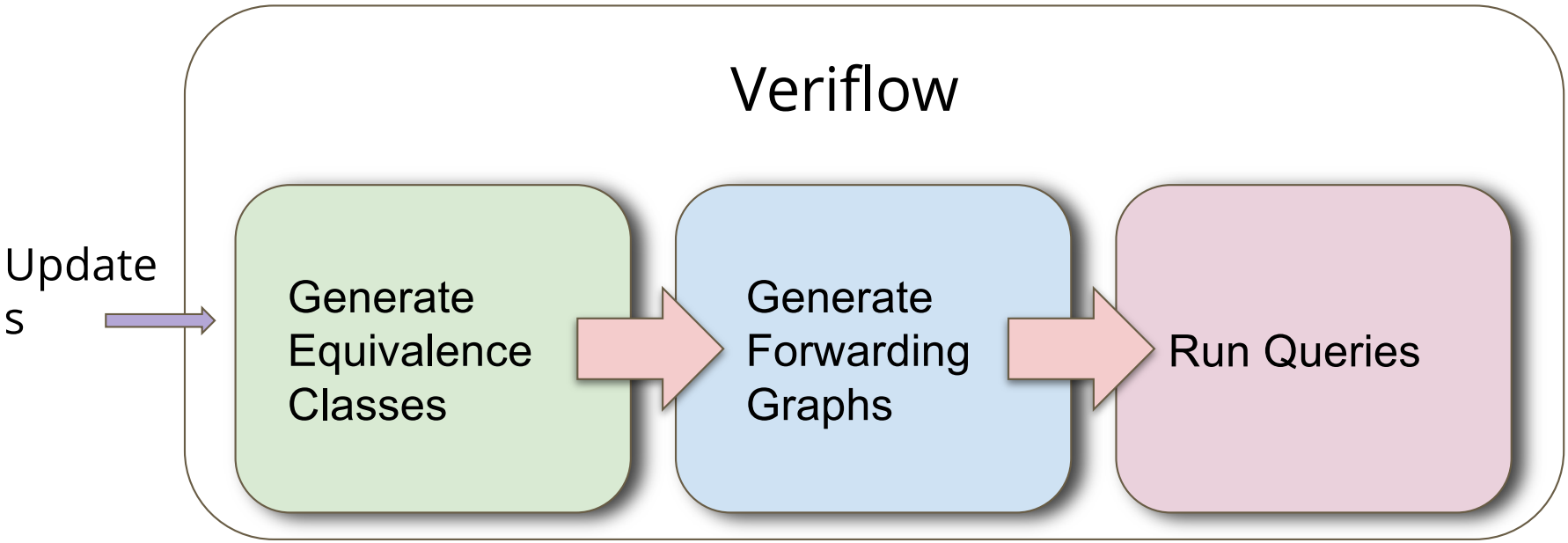


Forwarding Graph





3. Run Query to Check Invariants



Experiment

Evaluation #1- Microbenchmarking VeriFlow run time

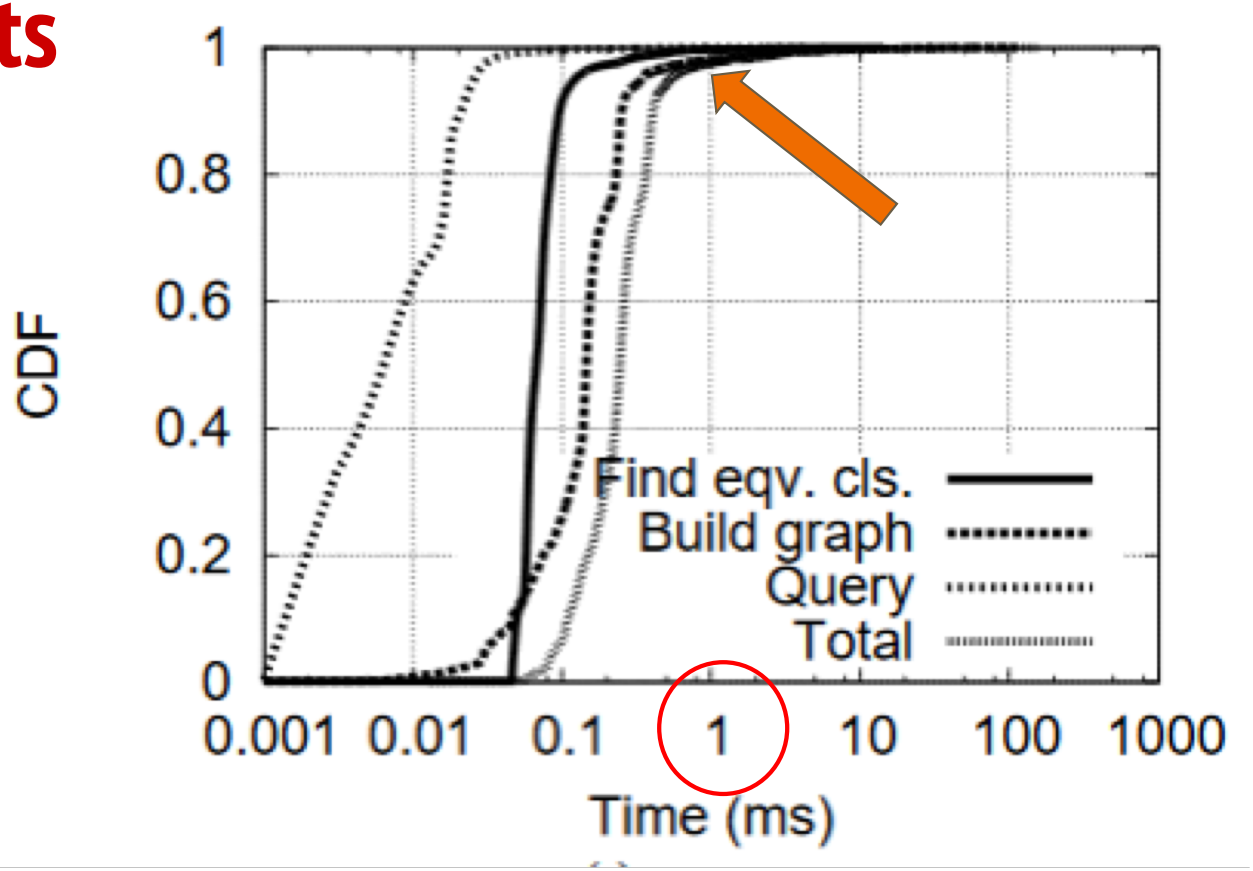
Goal - Observe Veriflow's different phases contributions to the overall run time

Simulated an IP network with 172 routers

Replayed BGP traces, with 5 million RIB entries and 90k BGP updates

Performance Results

97.8% of the updates were verified within 1 millisecond



Evaluation #2 - Effect on TCP Connection Setup Latency

Goal - Understand the impact of Veriflow on TCP connection setup latency

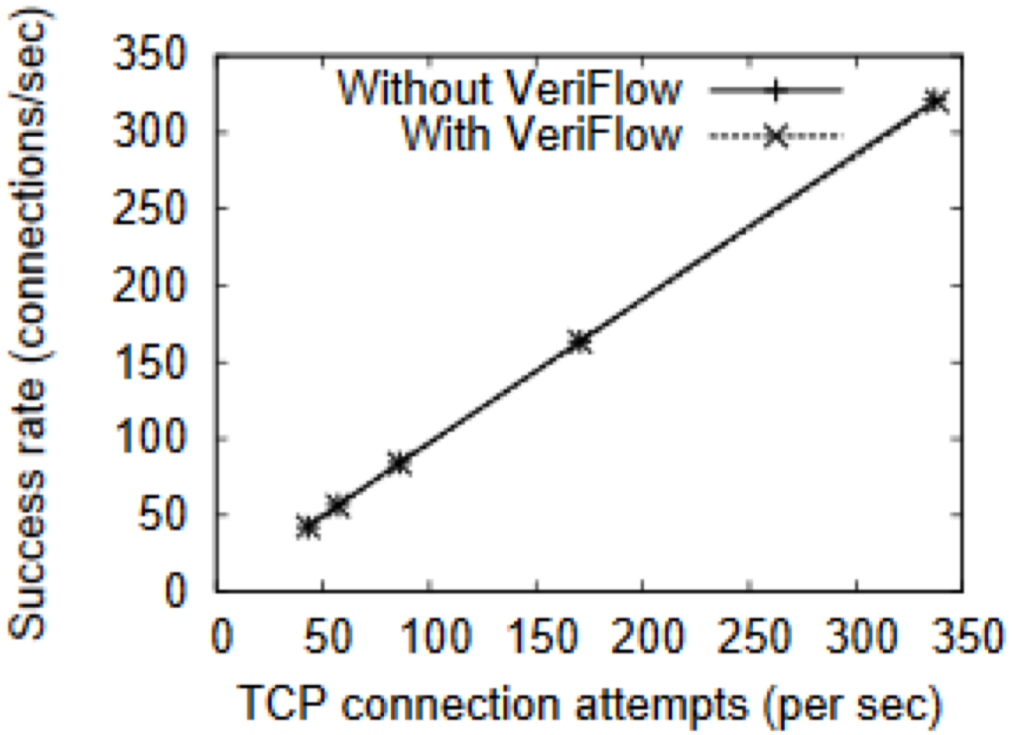
Mininet OpenFlow network

- 10 switched arranged in chain-like topology
- A host connect to every switch

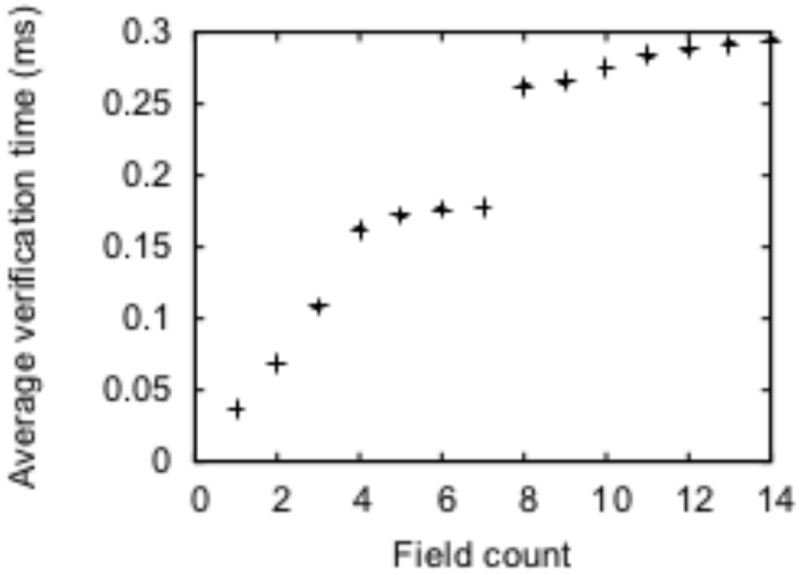
Nox controller running “learning switch” app

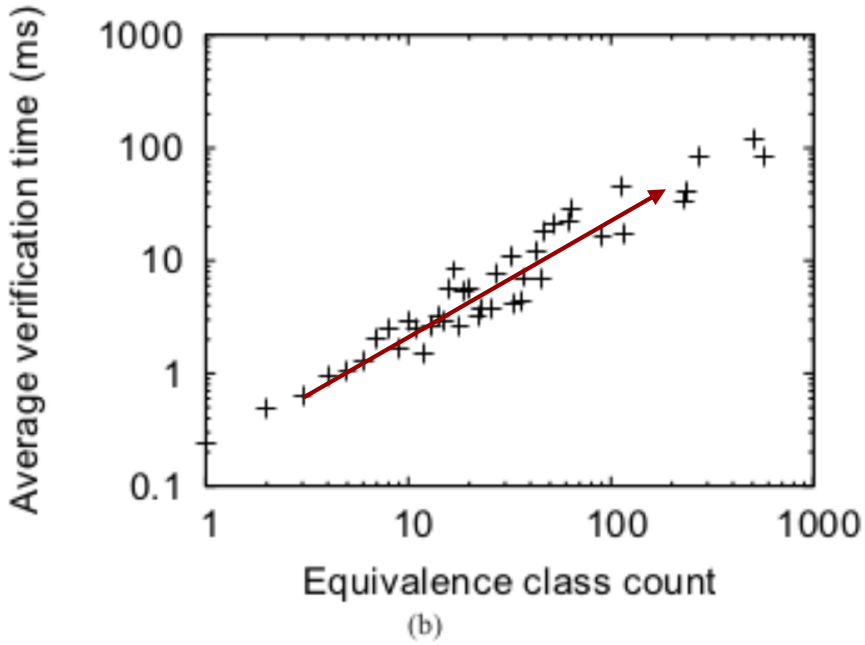
TCP connections between random pairs of hosts

Evaluation Results



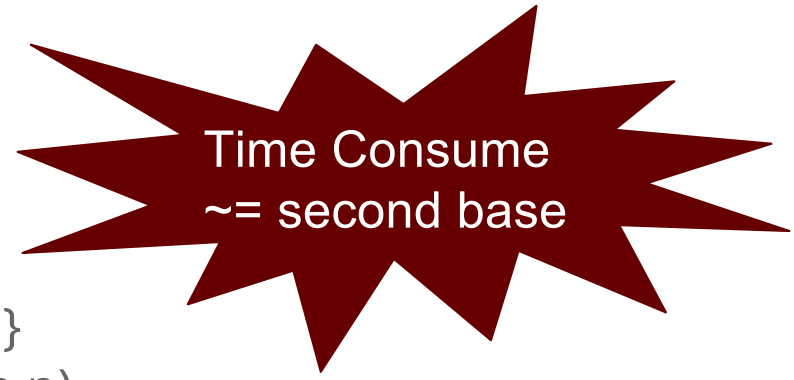
Time Consume





Comparison: HSA

Header Space Analysis



Operating overview

- Extract header from packets in binary {0,1}
- Construct forwarding transfer function T(h,p)
- Mathematical computation for verification

Achievements

- Reachability analysis
- Loop detection
- Slice isolation

P. Kazemian, G. Varghese and N. McKeown, “Header space analysis: static checking for networks”, NSDI'12 Proceedings of USENIX conference on Networked Systems Design and Implementation, 2012

NetPlumber

Features

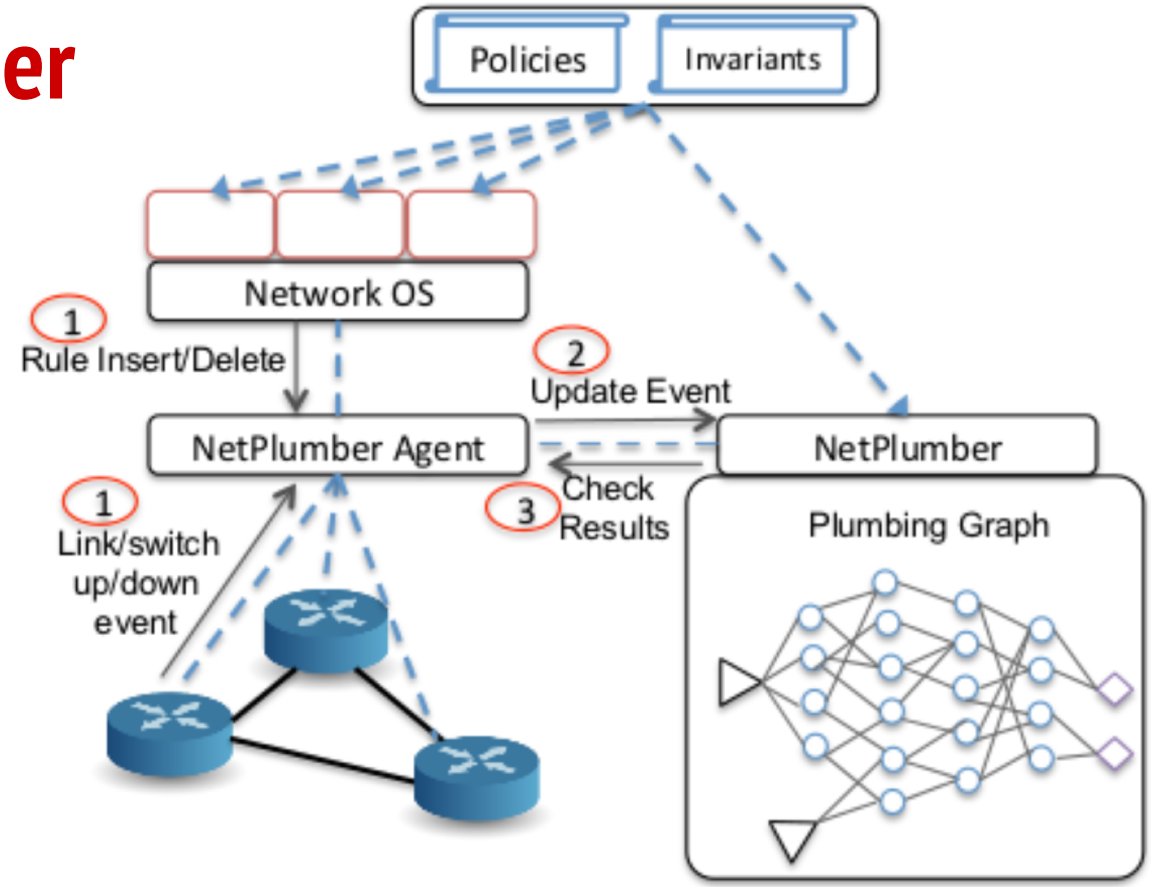
- Based on HSA
- Built dependency graph

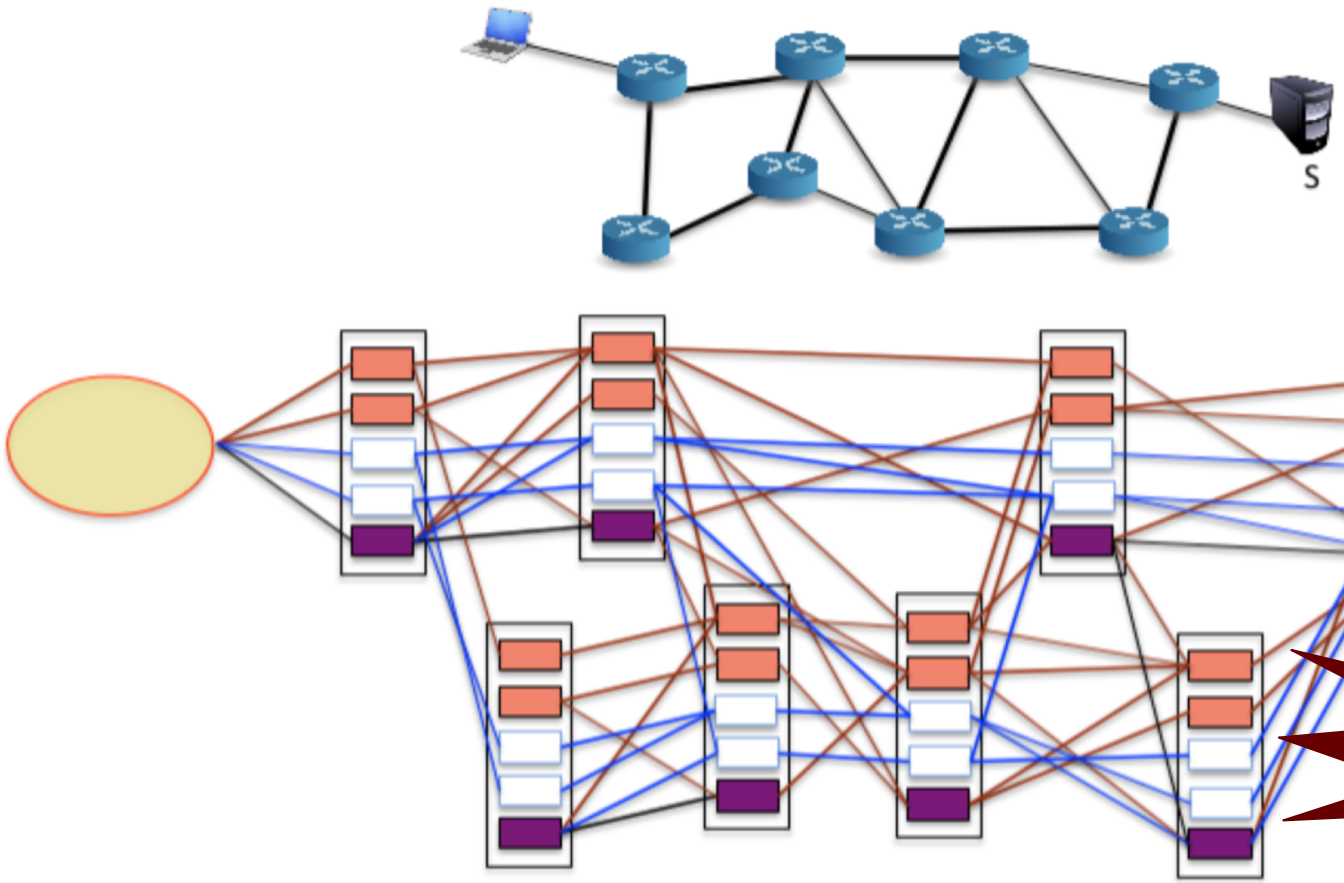
P. Kazemian, M.l Chang, H. Zeng, G. Varghese , N. McKeown, S. Whyte, “Real Time Network Policy Checking using Header Space Analysis”, NSDI'12 Proceedings of USENIX conference on Networked Systems Design and Implementation, 2013

Improvements of HSA

- Incremental update rules (achieve real-time)
- Without ad hoc code required by HSA (generalize to probe nodes)
- Cluster graph and reduce inner-edges (parallelization)

conti. NetPlumber





Time Consume
~= millisecond

Lessons Learned

VeriFlow vs NetPlumber (HSA)

	VeriFlow	NetPlumber(HSA)
History	2013 by UIUC	2013(2012) by Stanford
Apply Layer	Data-Plane	Data-Plane
Data Structure	Tree	Graph
Time Consume	millisecond	millisecond
Steps	Class / Flow / Queries	Space / Topology / Algebra
Verification	Custom Query Procedure	Algebra Operation

Both support forwarding actions and verification

Conclusion & Discussion

Conclusion

VeriFlow achieves real-time verification

- A layer between SDN controller and network elements
- Finds faulty flows issued by SDN applications
- Verifies network-wide invariants as each flow is inserted

Can prevent a flow from reaching the network

Thanks for your attention!

Problems and Discussion

1) Is there any limitation on Data-Plane verification ?
2) How can we improve the speed of Veriflow ?
3) Within the experimental results, there is a long tail behavior in the CDF. Why do you think that is?
4) Is it possible for VeriFlow to deal with the control logic error?
5) Is SDN pre-requisite for VeriFlow? Can we implement VeriFlow without the SDN's implementation?
6) Can Veriflow replace firewall in a networked system?

References

- A. Khurshid, X. Zou, W. Zhou, M. Caesar, P. Godfrey, VeriFlow: Verifying Network-Wide Invariants in Real Time, (Paper) and “VeriFlow: Verifying Network-Wide Invariants in Real Time”, PPT, http://conferences.sigcomm.org/sigcomm/2012/slides/sdn/session2/03-Veriflow.pdf, 2012
- P. Kazemian, G. Varghese, N. McKeown, “Header Space Analysis: Static Checking For Networks”
- G. N. Nde and R. Khondoker, "SDN testing and debugging tools: A survey," *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*, Dhaka, 2016, pp. 631-635.
- D. Nicol, K. Jin, M. Caesar, B. Sanders, “A Hypothesis Testing Framework for Network Security”, PPT
- Peyman Kazemian, Network Debugging, http://yuba.stanford.edu/~peyman/research.html