From SDN Network Hypervisor Measurements to Fast Virtual Network Provisioning: A Long Story Short

Andreas Blenk

*Technical University of Munich, Germany

University of Vienna, 2018-10-10
Motivation for Flexibility

Promising technologies and techniques:
- Network Virtualization (NV)
- Software-Defined Networking (SDN)
- Artificial Intelligence (AI)

Frequently changing demands need flexible adaptation
Simply Combining Technologies and Still Predictable?
Network Virtualization

Flexible? Adaptive? Programmable?

Combine with Software-Defined Networking
Software-Defined Networking

- Split control from data plane
- Centralized control
- Flexible control of forwarding (networking) resources

Legacy Network

Software-defined Network

Control Plane

Open Interfaces and Protocols (OpenFlow)
Combining Network Virtualization and Software-Defined Networking

- Virtual Networks according to service and application demands
- Flexible control of virtual networking resources
- Programmable virtual software-defined networks (vSDNs) [1]

And what is the problem now?

The Challenges

- Virtualization itself can introduce overhead
- Interference due to sharing
- Sources of unpredictability
- Good understanding (models) of virtualization layer design needed for correct provisioning

Architecture Design, Measurements

Function Placement, Optimization

FlowVisor: A Network Virtualization Layer

Rob Sherwood, Glen Gibb, Kok-Kiong Yap, Guido Appenzeller
Martin Casado, Nick McKeown, Guru Parulkar
* Deutsche Télékom Inc, R&D Lab, Los Altos, CA USA
† Stanford University, Palo Alto, CA USA
‡ Nokia Networks, Palo Alto, CA USA

State of the art: Measurements and Models

ABSTRACT

Network virtualization has long been a goal of the network research community. With it, multiple isolated logical networks can be shared among multiple logical networks, each with distinct forwarding logic. We use this switch-level virtualization to build a research platform which allows multiple network experiments to run side-by-side with production traffic while still providing isolation and hardware forwarding speeds. We also show that this approach is compatible with commodity switching chipsets and does not require the use of programmable hardware such as FPGAs or network processors.

We build and deploy this virtualization platform on our own production network and demonstrate its use in practice by running two experiments simultaneously within a campus network. Further, we quantify the overhead of our approach and evaluate the correctness of the isolation between virtual slices.

1. INTRODUCTION

This paper explores how to virtualize a network, and describes a particular system that we prototyped called FlowVisor - that we have deployed to slice our own production network. Similar to computer virtualization [22, 1, 21, 17], network virtualization promises to improve network utilization, permit operators to checkpoint their network before changes, and allow competing customers to share the same equipment in a controlled and isolated fashion. Critically, virtual networks also promise to provide a safe and realistic environment for measuring the performance and behavior of new networking protocols.

For many years, the GNS [14] framework, an off-the-shelf version of a virtual network slice, and two distant virtual networks on the same physical hardware slice.

No detailed performance study? Why? No Tool available!
Measurement Procedure
From non-virtualized SDN networks to virtualized SDN networks

- **Challenge**: Coordination and emulation complexity
- **Goal**: One tool emulating single tenant, single switch, multi-tenant, multi-switch
Perfbench [4,5]

- Multi-tenant/multi-switch emulation
- Traffic modeling: inter-arrival time, burstiness
- Modular measurements: either controller(s), switch(es), or both entities


Open Source!
https://github.com/tum-lkn/perfbench
Multi-Tenancy Measurement Setup

- Hypervisor: FlowVisor
- OpenFlow Message: FLOWMOD
- Key performance indicator:
  - Latency [milliseconds]
  - CPU [%] (100% = 1 Core)

- Multi-tenancy impact on CPU consumption
- Impact on control plane latency?
Multi-Tenancy Latency Results

HyperFlex: An SDN virtualization architecture with flexible hypervisor function allocation," 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, 2015, pp. 397-405

Already 10 tenants show a notable latency gap of 6ms (high variance)

The more switches and controllers, the less predictable
Fast and Efficient (Virtual) Network Provisioning
New Opportunities Introduce New Problems and Challenges

- **Optimal Algorithm:**
  - Improves solution quality given more flexibilities
  - Expensive, exponential runtime

- **Heuristic Algorithm:**
  - Can exploit flexibility
  - But cannot achieve optimal solution

- **Machine Learning/Neural Computation:**
  - Improves solution quality
  - Impact of learning time? Computational overhead? RESEARCH!
How some people see AI and Machine Learning!
If you work a bit with it … it should be your friend and helper!
Machine Learning for Networking

- Deep Reinforcement Learning
- Neural Computation
- Towards Self-Driving and Intelligent Networks
- Empowering Networks
- Probabilistic Modeling

Key Contributions:
- IEEE INFOCOM 2018 Main conference paper: NeuroViNE
- ACM SIGCOMM 2017 Big-DAMA Workshop, 2017: o zapft'is
- IEEE ICNP 2017: Data-Driven Algorithm Improvement
- IEEE NetwSoft 2016: Performance Prediction for VNFs
- IFIP/IEEE CNSM, 2016: Boost VNE
- IEEE AnNet Workshop 2017: Generating Topologies with SBMs
- ACM SIGCOMM Poster 2018: Weighted SBMs
- IEEE INFOCOM Poster 2018: Botnet Detection
- IFIP Networking 2018. Data-Driven Admission Control
- ACM SIGCOMM Self-DN Workshop 2018: Empowerment

- Alpha GO inspired (Monte Carlo Tree Search)
- XXX (submitted) Routing (confidential)
Overview in this talk: NeuroViNE

IEEE INFOCOM 2018 Main conference paper: NeuroViNE

ACM SIGCOMM 2017 Big-DAMA Workshop, 2017: o zapft'is

IFIP/IEEE CNSM, 2016: Boost VNE

ACM SIGCOMM Poster 2018: Weighted SBMs

IEEE INFOCOM Poster 2018: Botnet Detection

ACM SIGCOMM Self-DN Workshop 2018: Empowerment

Neural Computation

Towards Self-Driving and Intelligent Networks

Prediction

Probabilistic Modeling

Empowering Networks
Use case study: Online Virtual Network Embedding (VNE) problem

Hard Problem

(1) Optimal solutions do not scale

Vs.

(2) Heuristics may result in large footprints

Neural Preprocessing to achieve
(1) scalability and (2) quality
The Idea: Subgraph Extraction

→ Reduce embedding cost of heuristics (search on close substrate nodes)
→ Improve runtime of optimal algorithms (shrink search space)

But how do we find good subgraphs?!
Contribution: NeuroViNE [6]

Neural Computation
Parallel Computation
Implementable on hardware
Reuse existing VNE algorithms

Hopfield network solution provides nodes with high capacity close to each other

“Neural” computation of decisions in optimization problems
JJ Hopfield, DW Tank - Biological cybernetics, 1985 - Springer

Abstract Highly-interconnected networks of nonlinear analog neurons are shown to be extremely effective in computing. The networks can rapidly provide a collectively-computed solution (a digital output) to a problem on the basis of analog input information. The …
Hopfield Network

An Artificial Recurrent Neural Network (which can be used for optimization)

- Number of neurons and states $V$
- Input bias vector $I$
- Connection weights $T$
- Energy of network

\[ E = -\frac{1}{2} V^T TV - V^T I \]

- Fullfills Lyapunov function property
- Convergence to local (global) optima guaranteed

**Hopfield Network Properties**

How to map Virtual Network Embedding problem?
Hopfield Network
How to use for optimization ...

1. Optimization problem: find subgraph with low resource footprint and high probability for accepting virtual network
2. VNE problem energy function
   \[ E = V^T \left( \Psi(t) + \alpha \cdot T^{\text{constraint}} \right) V + V^T \left( \Xi(t) + \alpha \cdot I^{\text{constraint}} \right) \]
3. Derive: \( \Psi(t), T^{\text{constraint}}, \Xi(t), I^{\text{constraint}} \)
4. Execute network: solve
5. After execution \( \rightarrow \) Neuron states (values) indicate subgraph nodes

Hopfield Optimization Procedure

We do not solve VNE directly ...
But show Hopfield‘s preprocessing capabilities
NeuroViNE’s Hopfield Network Energy Function

\[ E = V^T (\Psi(t) + \alpha \cdot T_{\text{constraint}}) V + V^T (E(t) + \alpha \cdot I_{\text{constraint}}) \]

Select paths with low costs = low energy
Satisfying constraint = low energy
Select physical nodes with high CPU ratio = low energy
NeuroViNE’s Hopfield Network Construction

Example for 3-Node Substrate and 2-Node Virtual Network

$\mathcal{E}_i(t)$

3 substrate nodes with CPU resource

3 neurons - Input bias vector considers CPU

Node ranking

$$\mathcal{E}_i(t) = \frac{\max_{N j \in N} C_j(t) - C_i(t)}{\max_{N j \in N} C_j(t)}$$
Path Ranking
NeuroViNE’s Hopfield Network Construction

3 links with datarate attributes

3 times 3 entries of weight matrix

Path ranking

\[ \Psi_{ij}(t) = \gamma \frac{D_{ij}(t)}{\max_{ij} D_{ij}(t)} \]
Keeping Constraints

NeuroViNE’s Hopfield Network Construction

2 Virtual nodes

2 out of 3 neurons should be chosen

Node number selection constraints

\[ T_{ij}^{\text{constraint}} = \begin{cases} 1 & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \]

\[ I_k^{\text{constraint}} = -(2 \cdot \zeta - 1) \]
NeuroViNE’s Hopfield Network Energy Function

\[ E = V^T \left( \Psi(t) + \alpha \cdot \mathbf{T}_{\text{constraint}} \right)V + V^T \left( \Xi(t) + \alpha \cdot \mathbf{I}_{\text{constraint}} \right) \]

- Select paths with low costs = low energy
- Satisfying constraint = low energy
- Select virtual nodes with high CPU ratio = low energy
NeuroViNE: An Illustrative Example for GRC on 750 nodes ISP network

Selected nodes by GRC lead to long paths!

NeuroViNE selects substrate nodes close to each other!
Same Behavior for Datacenters

Virtual Machines Spread Among Racks

Virtual Machines Close to Each Other

Heuristic

NeuroViNE
NeuroViNE: Efficient also in Datacenters

Uses a datacenter modification (see paper)

NeuroViNE shows similar acceptance ratios...but saves cost
Overview in this talk: o zapft'is [7]

IEEE INFOCOM 2018 Main conference paper: NeuroViNE

ACM SIGCOMM 2017 Big-DAMA Workshop, 2017: o zapft'is

IFIP/IEEE CNSM, 2016: Boost VNE

ACM SIGCOMM Poster 2018: Weighted SBMs

IEEE INFOCOM Poster 2018: Botnet Detection

ACM SIGCOMM Self-DN Workshop 2018: Empowerment

The Limitation – Fire and Forget

The Opportunity – Tap into your Algorithm’s Big Data
Traditional vs. Proposed System

Problems

Instances

Learning

Machine

Optimization

Algorithm

Produce

Solutions

Potentials

Search Space Reduction: reduction/Initial Solutions

Predict Value of Objective Function
Learn and predict the acceptance and embedding cost of a VNR

- Supervised learning
- Offline training!

Learning to Accept and to Predict the Cost

Library:
- Sci-Kit Learn [9]

Graph features:
- Node degree
- Closeness
- Betweenness
- Spectral Features

Measures:
- $R^2$ (goodness of fit for ML models)
- TPR/TNR/Accuracy/…

Classifier/Regressor:
- Recurrent Neural Network (RNN)
- Linear Regression (LR)
- Bayesian Ridge Regressor (BRR)
- Random Forest Regressor (RF)
- Support Vector Regression (SVR)

Supervised Learning & Speed-Up Results [10]

Efficiently filter infeasible and inefficient requests

... save runtime!

And keep the performance …
Conclusion

- Realizing flexible virtualized networks introduces new challenges
  - Overhead and interference
  - New dimensions for optimization

- This talk
  - A tool for measuring virtualization layers
  - Application of neural computation and machine learning to network algorithms

Thank you!
Questions?
Research Trailers
Where did we apply **Artificial Intelligence** (Machine Learning) so far?

- Create network service graphs
- Detect anomalies
- Virtual network provisioning
- Function placement
- Admission control
- Flow routing

**Network Planning and Dimensioning**
- Prepare your network for failures
- Generate network topologies with realistic characteristics

**Network Monitoring**

**Network Resource Allocation Algorithms**

**Research Towards Self-Driving Networks**
Network Monitoring

- (Weighted) Stochastic Block Models
- Machine Learning for training
  (unsupervised)
- Who communicates with whom
  → Create service graph layout
- Plan and benchmarking
  → Generate realistic communication patterns
- Communication pattern changes over time
  → Detect anomalies (abnormal bot communication)

IEEE INFOCOM Poster 2018: Botnet Detection [12]
ACM SIGCOMM Poster 2018: Weighted SBMs [13]

SBM has many applications!

Router Level Graph of the Internet
- 939 nodes, 988 edges

Intelligent grouping based on communication patterns only
Network Planning and Dimensioning

- Empower your network to be prepared
  - For failures
  - For changing traffic
  - For new services
  - ...

ACM SIGCOMM Self-DN Workshop 2018: Empowerment [14]

Empowerment towards Network Intelligence?