Software-Defined Networking and Application Platforms

Raouf Boutaba

David R. Cheriton School of Computer Science
University of Waterloo - Canada

ICIN, Issy Moulineaux, March 1st, 2016

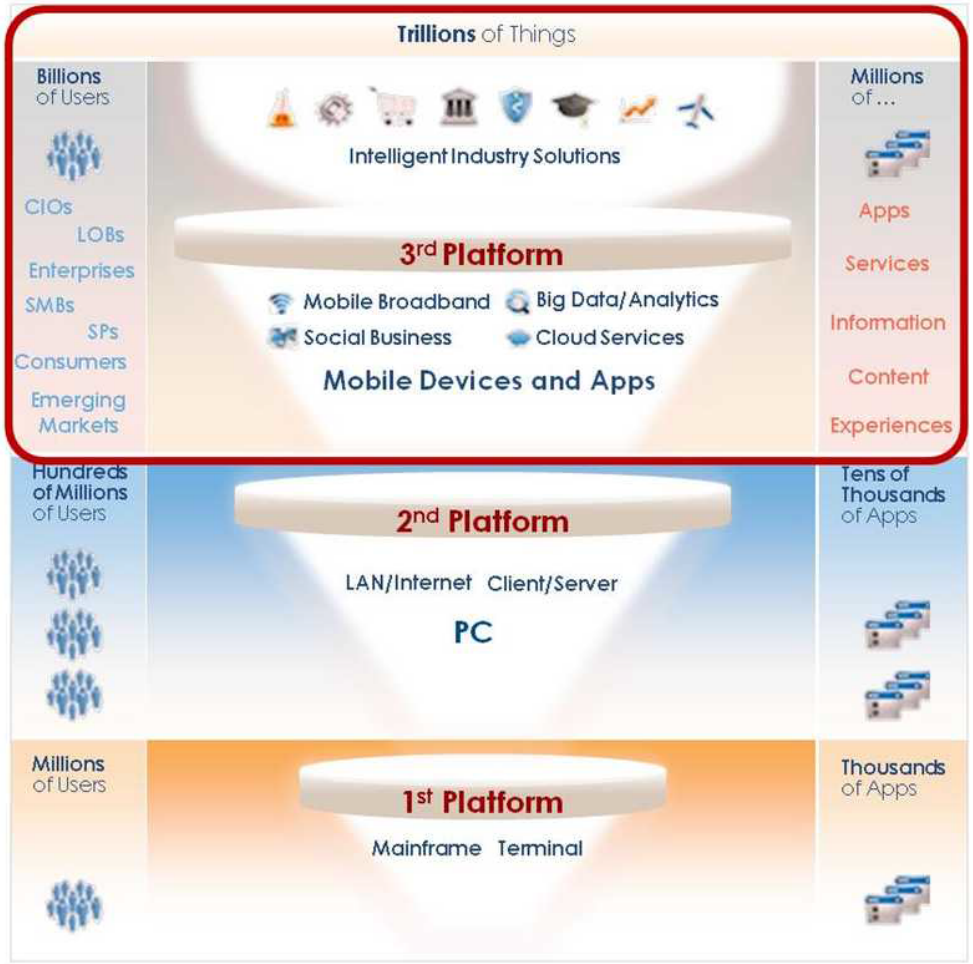
Outline

- Future Application Platforms: trends and challenges
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

ICT Innovation

ICT Innovation
2005–2020+

ICT Innovation
1985–2005



IDC top 10 predictions

3rd Platform's 4 Pillars

- Mobile broadband
- Cloud-based services
- Big Data analytics
- Social media

Mobile Broadband

- High connectivity
- High transmission speed
- Low latency
- 5G ?

Cloud Services

- Economical
- Scalable
- Elastic
- Flexible

Big Data Analytics

- Ability to harness and analyse large and complex sets of data
- The 3 Vs
  - Volume
  - Velocity
  - Variety
- More Vs ?

Social Media

- Social Networking
- Social Business solutions
- Other innovative applications
  - Social media search
  - Social gaming
  - …

Future Application Platform

- Mobile broadband
- Cloud-based services
- Big Data analytics
- Social media

➔ SAVI Project

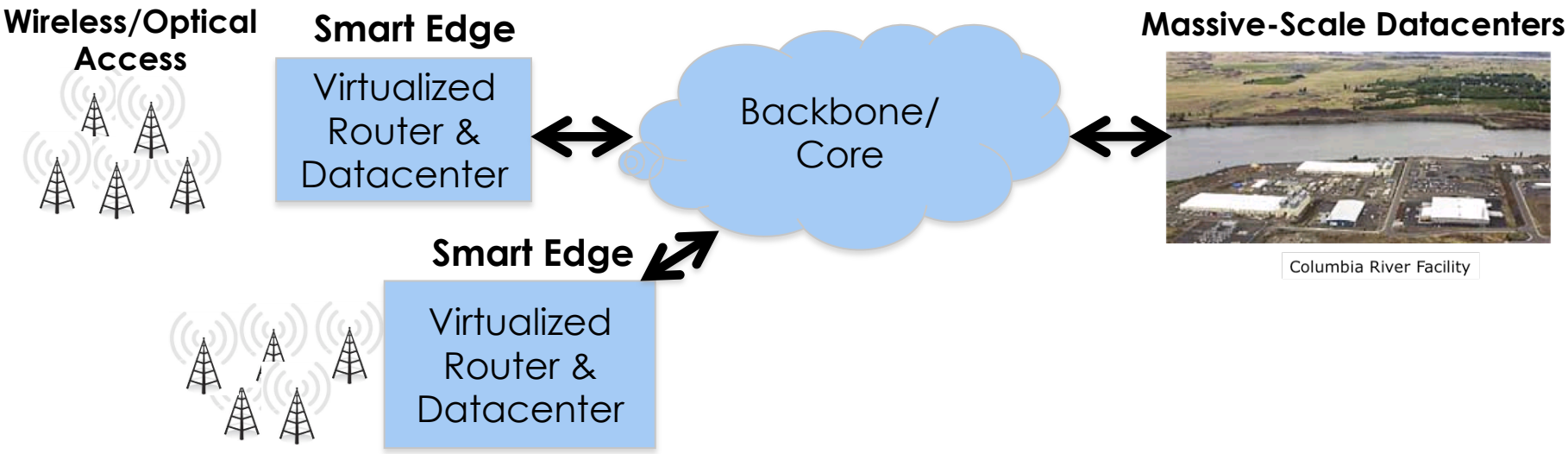
Outline

- Future Application Platform: trends and challenges
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

The SAVI Project

- SAVI : Smart Applications on Virtual Infrastructures
  - An NSERC Strategic Research Network
  - 9 Universities (16 professors & > 80 graduate/postgraduate students)
  - 13 Industry Partners (IBM, Cisco, Ericsson, Juniper, TELUS, …)
- SAVI Research Themes:
  - Smart applications
  - Extended Cloud Computing
  - Smart Converged Edge
  - Integrated Wireless Optical Access
  - SAVI Application Platform Testbed

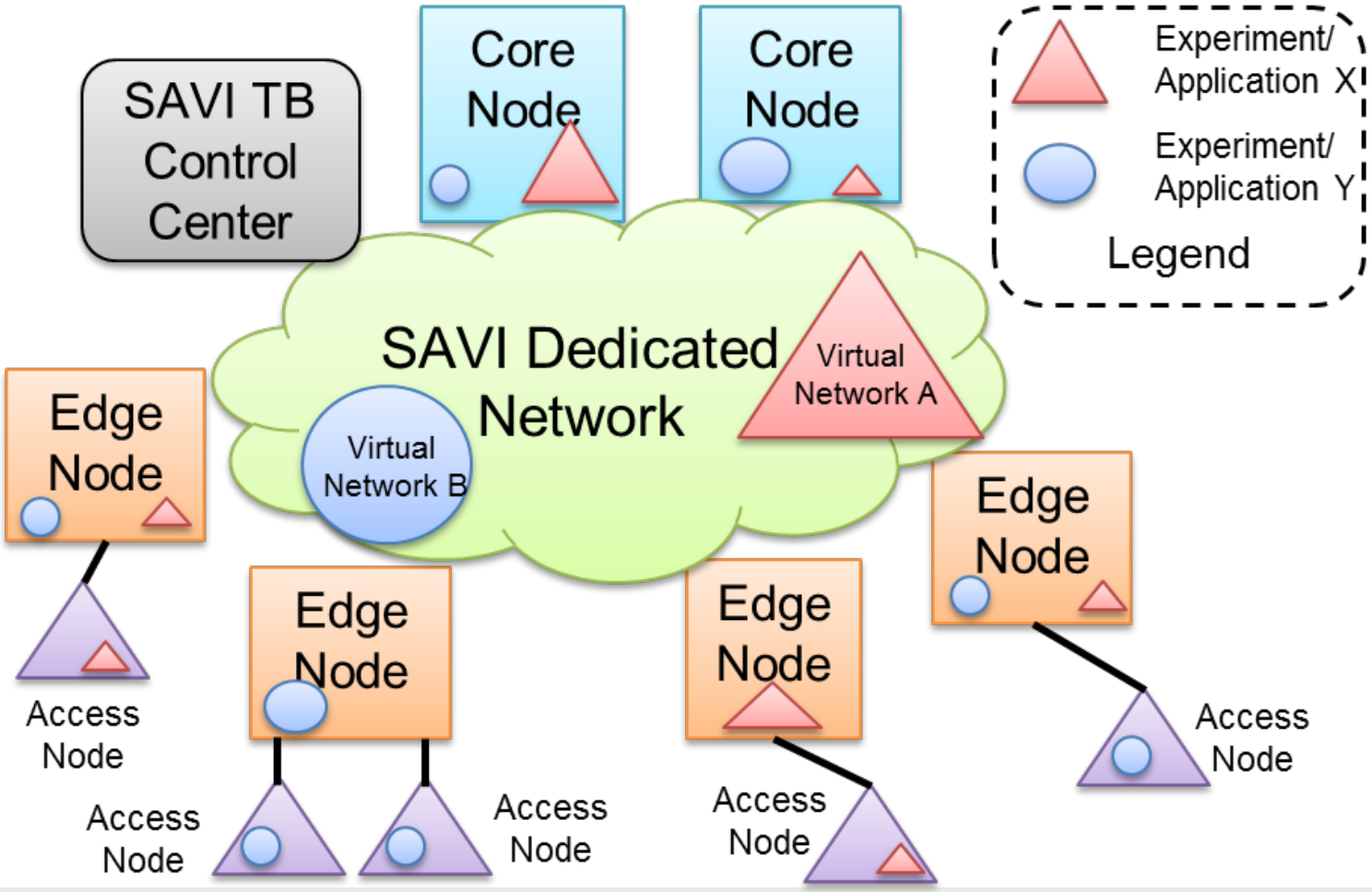
SAVI Vision



- **Research Scope**
  - Extended computing cloud that includes smart edge
  - Application enablement leveraging very-high bandwidth access and low-latency services in the smart edge and massive remote cloud resources
  - Integrated wireless/optical access controlled by the smart edge
  - Control & Management system to enable experimentation with service applications and Future Internet architectures
  - SAVI Testbed

SAVI Testbed

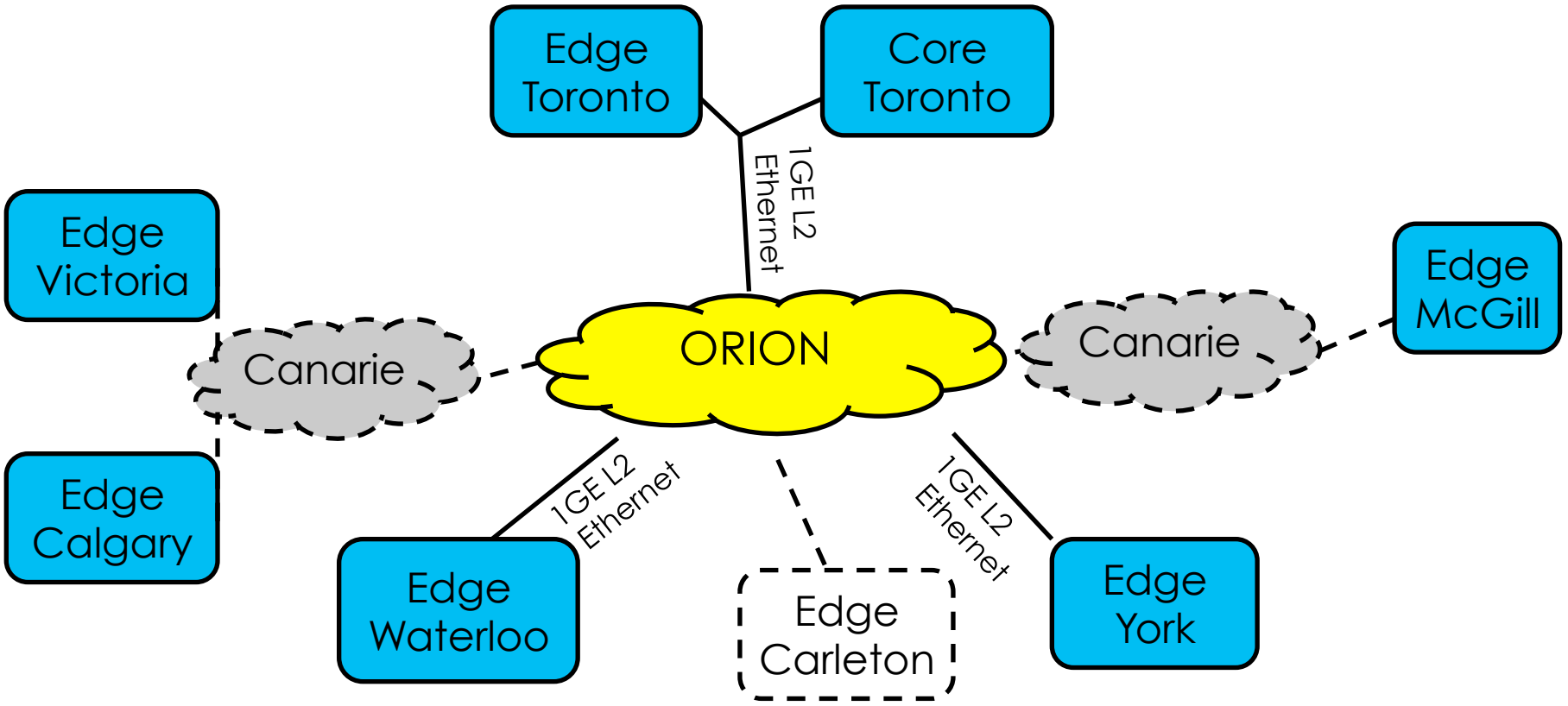
- Experimenters request slices of resources
  - Interconnected to form virtual infrastructures



SAVI Testbed Resources

- 1000+ cores
- 10+ FPGA systems
- 6+ GPU systems
- 100+ TB storage
- 10/1 GE fabrics (OpenFlow)
- 1GE dedicated backbone: ORION

SAVI Testbed Dedicated Network



Outline

- Future Application Platform: trends and challenges
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- The Present and the Future of SDN
- Conclusion

The Smart Edge - Goals

- To develop a virtualized network and computing infrastructure that provides responsive and high-capacity virtualized resources and services close to the user.
- To provide converged computing, networking, programmable hardware processing, and storage that complement the resources provided in remote datacenters
- To implement edge routers using virtual resources with a focus on energy efficiency, scalability and programmability.

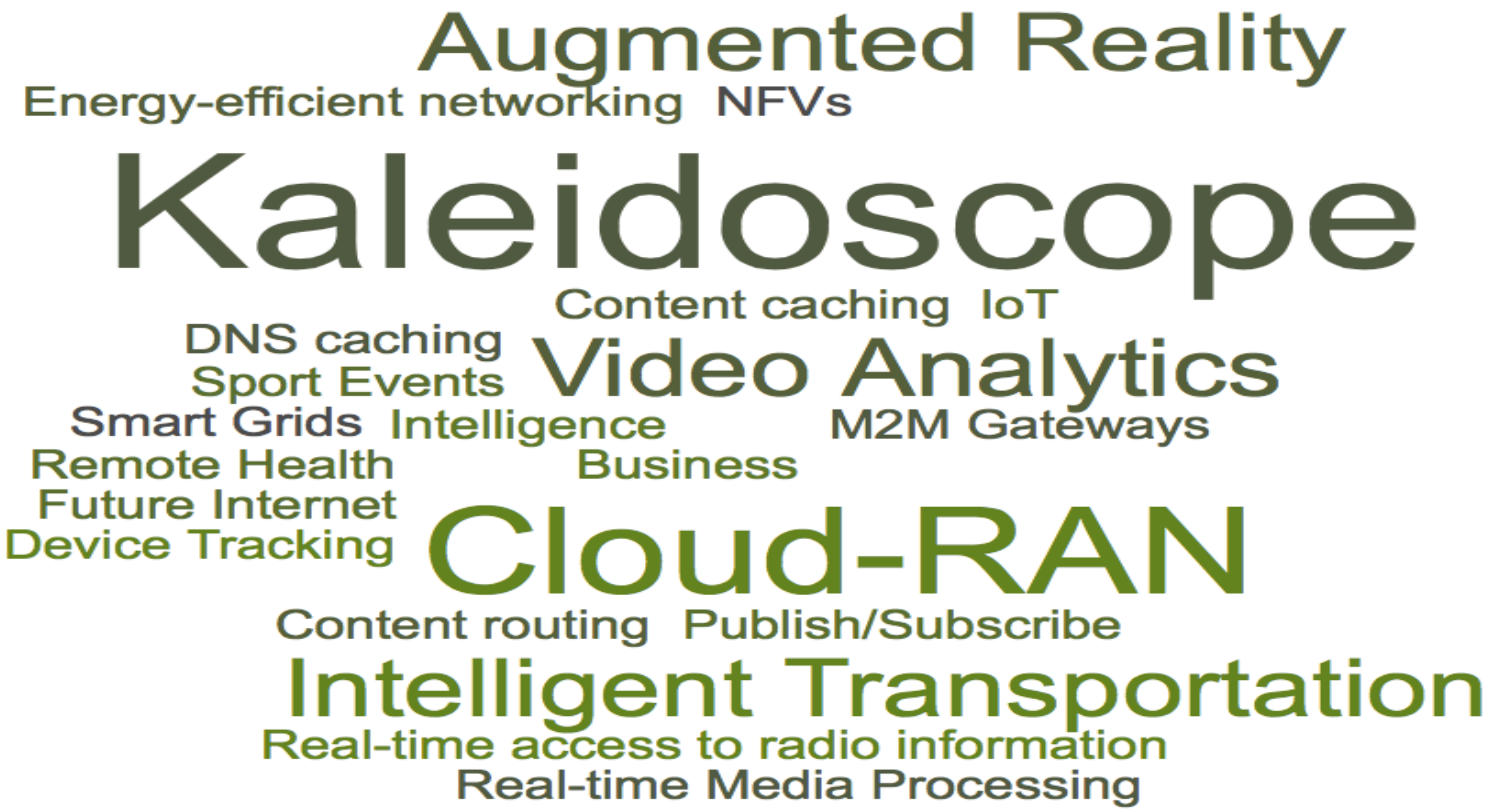
Smart Edge Characteristics

- Converged
  - Provides access to heterogeneous resources
    - Computing, storage and networking (data center, WAN)
    - Programmable hardware (GPUs, FPGAs, NetFPGAs, BEE boards)
- On-Premises
  - Can be isolated from rest of network while accessing local resources
  - e.g., security/safety systems
- Proximity
  - Close to source of information, can capture key information for Big Data Analytics
  - Direct access to devices, e.g., can be leveraged by business specific apps

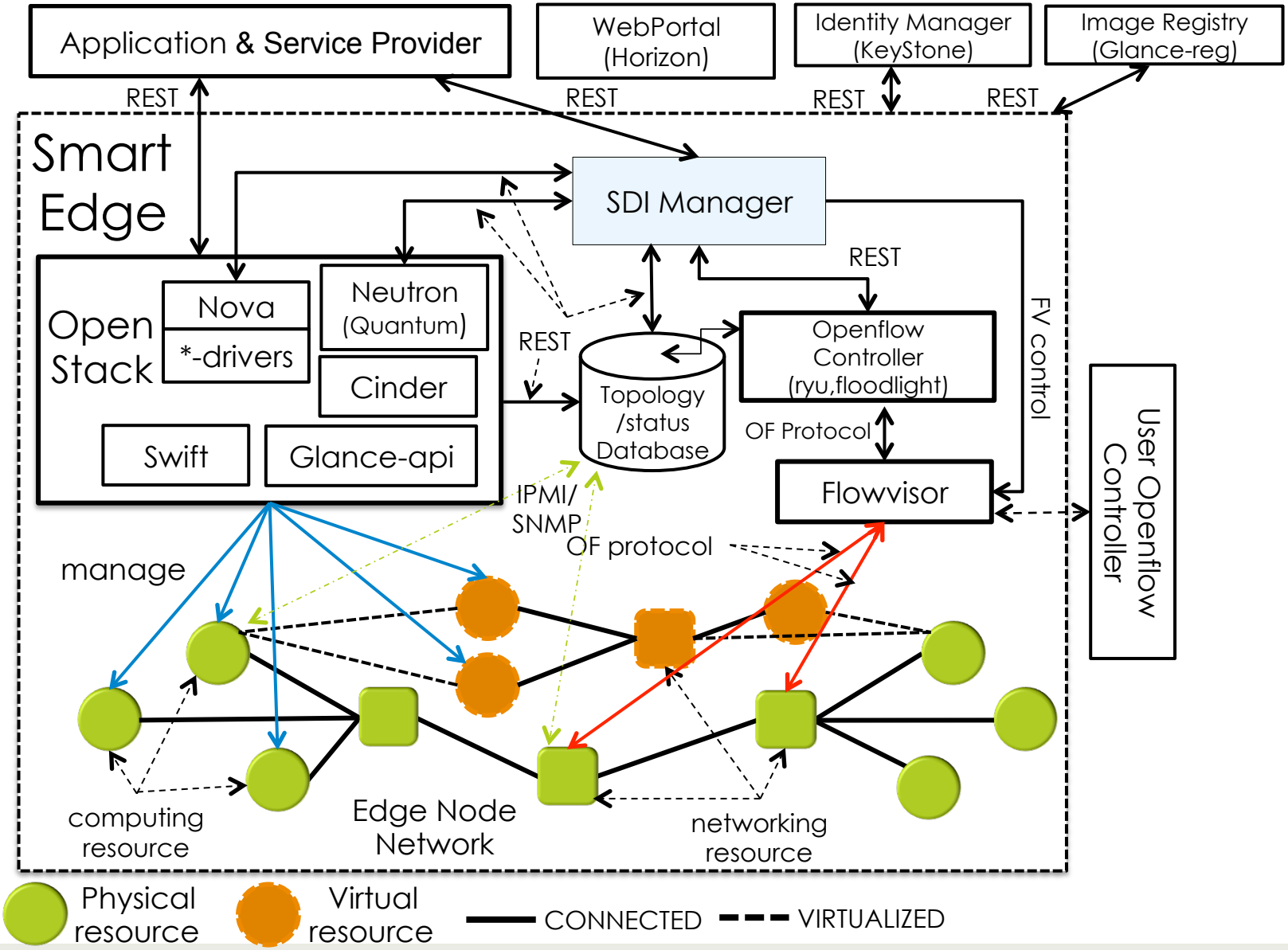
Smart Edge Characteristics (cont)

- Low-latency
  - Close to end user, reducing latency considerably
  - React faster, improve QoE, minimize congestion elsewhere
- Location awareness
  - User/device tracking
  - Location-based services (local points of interest), analytics, etc.
- Network context information
  - Real-time network data (e.g., radio condition, network stats)
  - Context-aware services/apps for improved QoE
- Programmability
  - Software-Defined Infrastructure: Combines cloud computing technologies and software-defined networking under a single management system

Use Cases


Augmented Reality
Energy-efficient networking
NFVs
Kaleidoscope
Content caching
IoT
DNS caching
Sport Events
Video Analytics
Smart Grids
Intelligence
M2M Gateways
Remote Health
Business
Future Internet
Device Tracking
Cloud-RAN
Content routing
Publish/Subscribe
Intelligent Transportation
Real-time access to radio information
Real-time Media Processing

Smart Edge Integrated C&M

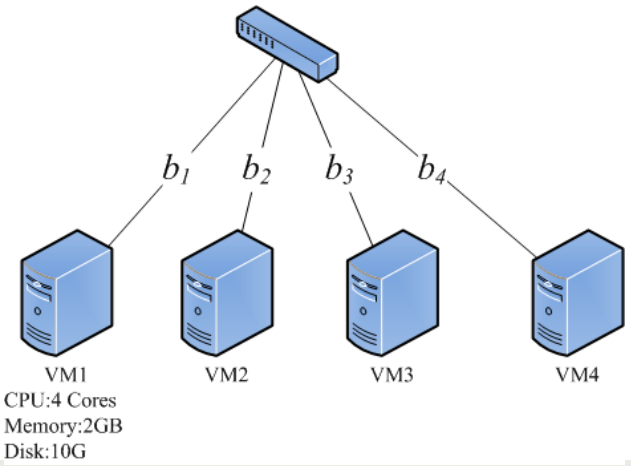


Outline

- Future Application Platforms: trends and challenges
- Convergence of IT and Telecommunication Infrastructure
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

Virtual Data Centers

- Currently cloud providers provide only computing resources but do no provide guaranteed network resources
- Goal: Provide both guaranteed computing and network resources
  - Virtual Data Centers (VDCs): virtual machines, routers, switches and links



VDC Embedding

- Objectives
  - Map VDCs onto physical infrastructure (Computing + networking resources)
  - Maximize acceptance ratio/revenue
  - Minimize energy costs
  - Minimize the scheduling delay
  - Achieve all of the above objectives dynamically over-time
- VDC Planner*
  - A migration-aware virtual data center embedding framework
  - VDC embedding, VDC scaling
  - Dynamic VDC consolidation.

*M. F. Zhani, Q. Zhang, G. Simon, R. Boutaba. VDC Planner: Dynamic Migration-Aware Virtual Data Center Embedding for Clouds. IFIP/IEEE IM'13. Ghent (Belgium), May 27-31, 2013.

VDC Planner Architecture

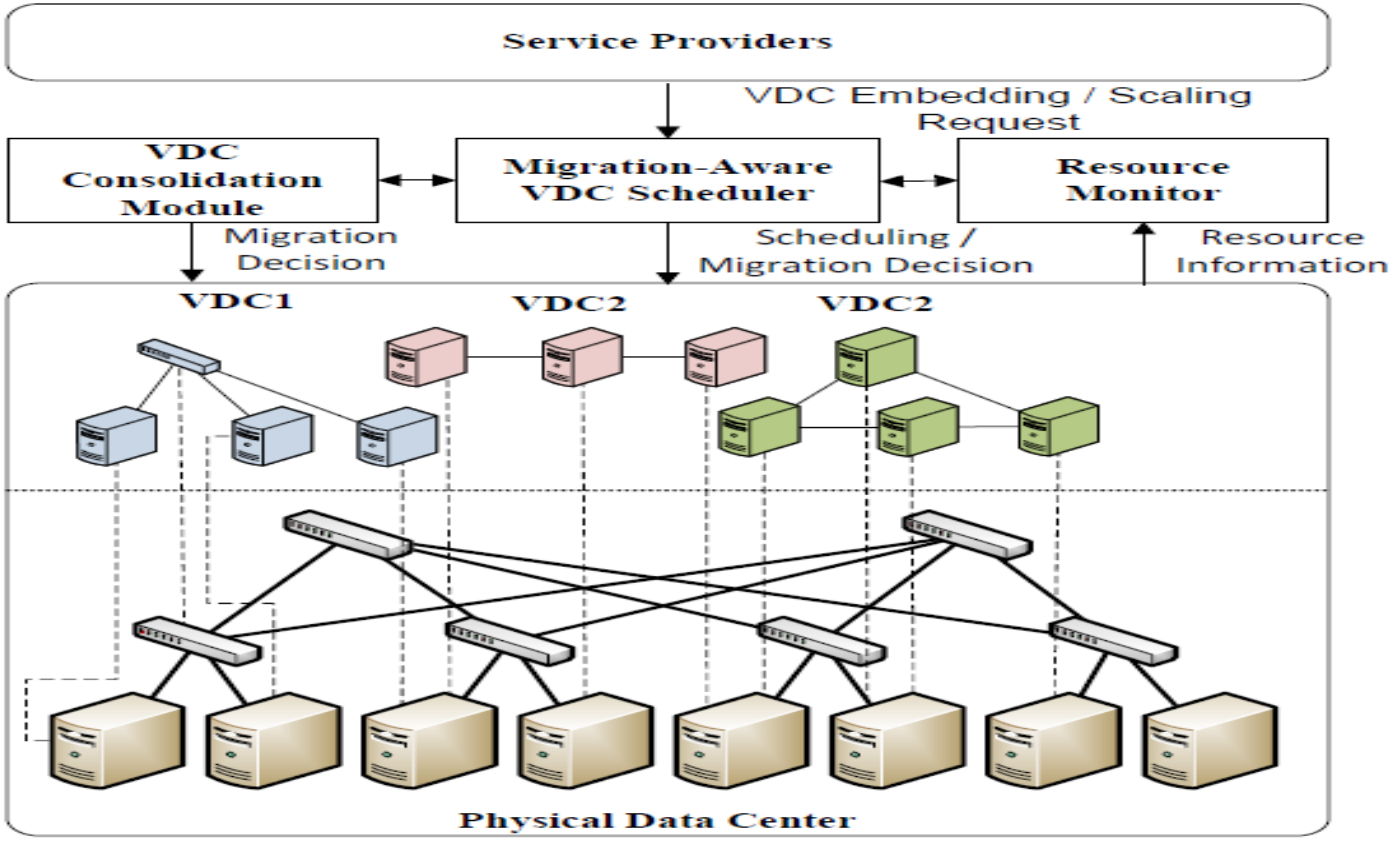
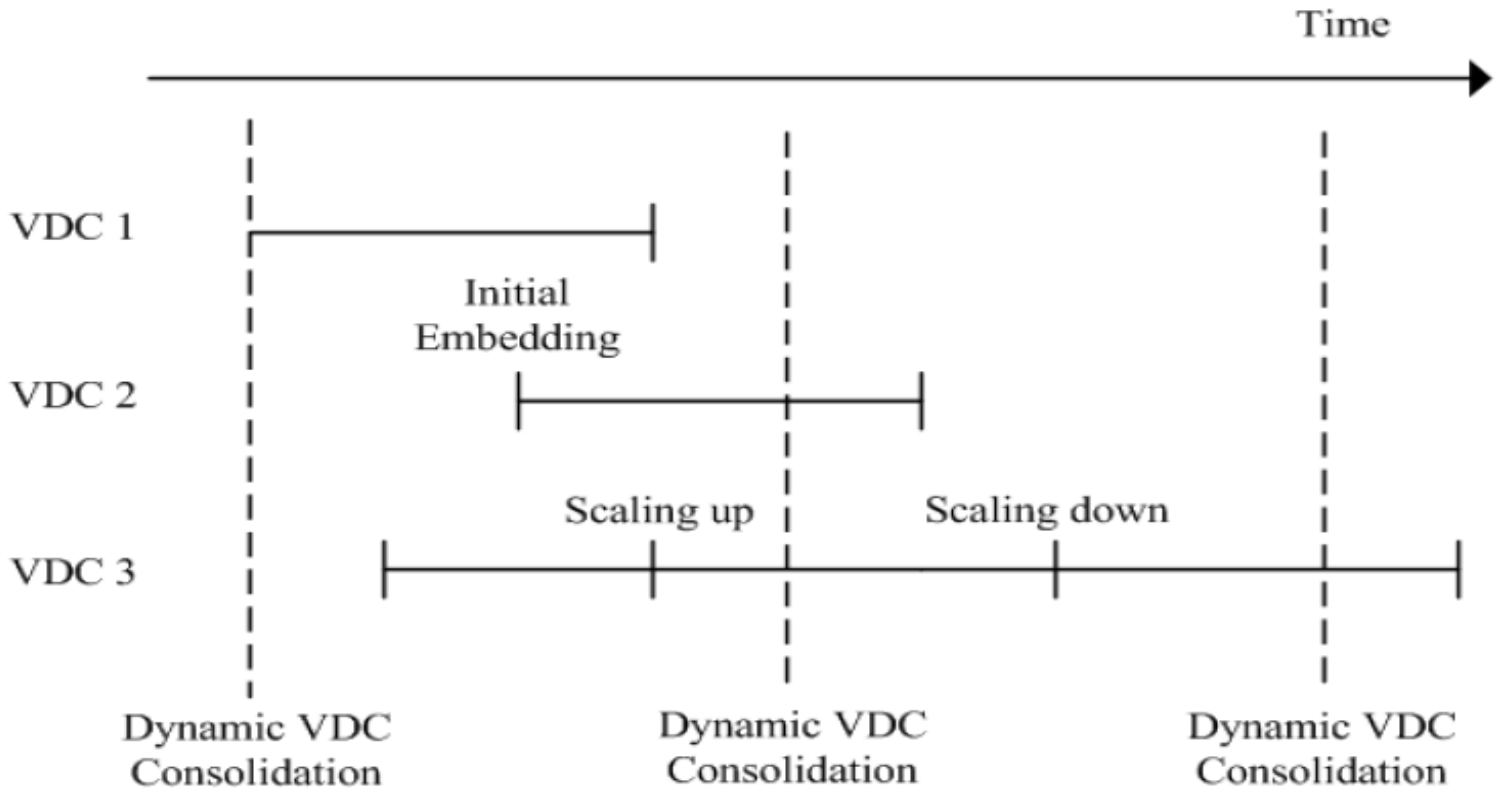


Figure 1: VDC Planner Architecture

VM Migration: Usage Scenarios



Problem formulation

- Objective function

$$\min \sum_{\bar{n} \in \bar{N}} y_{\bar{n}} p_{\bar{n}} + \sum_{i \in I} \sum_{n \in N^i} \sum_{\bar{n} \in \bar{N}} \gamma_n x^i_{n\bar{n}} g^i_{n\bar{n}}$$

Operational costs

embedding cost

$y_{\bar{n}}$ a Boolean that indicates that $\bar{n}$ is active
$x^i_{n\bar{n}}$ a Boolean that indicates that n is embedded in $\bar{n}$

- Embedding cost

$$g^i_{n\bar{n}} = \begin{cases} mig(n, \bar{m}, \bar{n}) & \text{if } \bar{n} \neq \bar{m} \\ 0 & \text{if } \bar{n} = \bar{m} \\ 0 & \text{if } n \text{ is currently not embedded} \end{cases}$$

Migration-Aware VDC Embedding

- Sort the VMs by their size

$$size_n^i = \sum_{r \in R} w^r c_n^{ir}$$

- Compute the embedding cost (for each VM and physical node)

$$cost^i(n, \bar{n}) = \gamma_n(mig(n, \bar{m}, \bar{n}) + MigOther(n, \bar{n})) + \sum_{n' \in N^i : (n', n) \in L^i} d(\bar{n'}, \bar{n}) \cdot b_{(n', n)} \quad (19)$$

- Embed the VM in the physical machine with the minimal embedding cost

Dynamic VDC Consolidation

- Sort the physical nodes in increasing order of their utilizations

$$U_{\bar{n}} = \sum_{r \in R} \sum_{i \in I} \sum_{n \in N^1 : n \in loc(\bar{n})} \frac{w^r c_n^{ir}}{c_{\bar{n}}^r},$$

- Migrate the VMs hosted in low-utilization machines (using Migration-Aware VDC Embedding Algorithm)
- If all VMs are successfully migrated, the machine is turned off.

Experiments

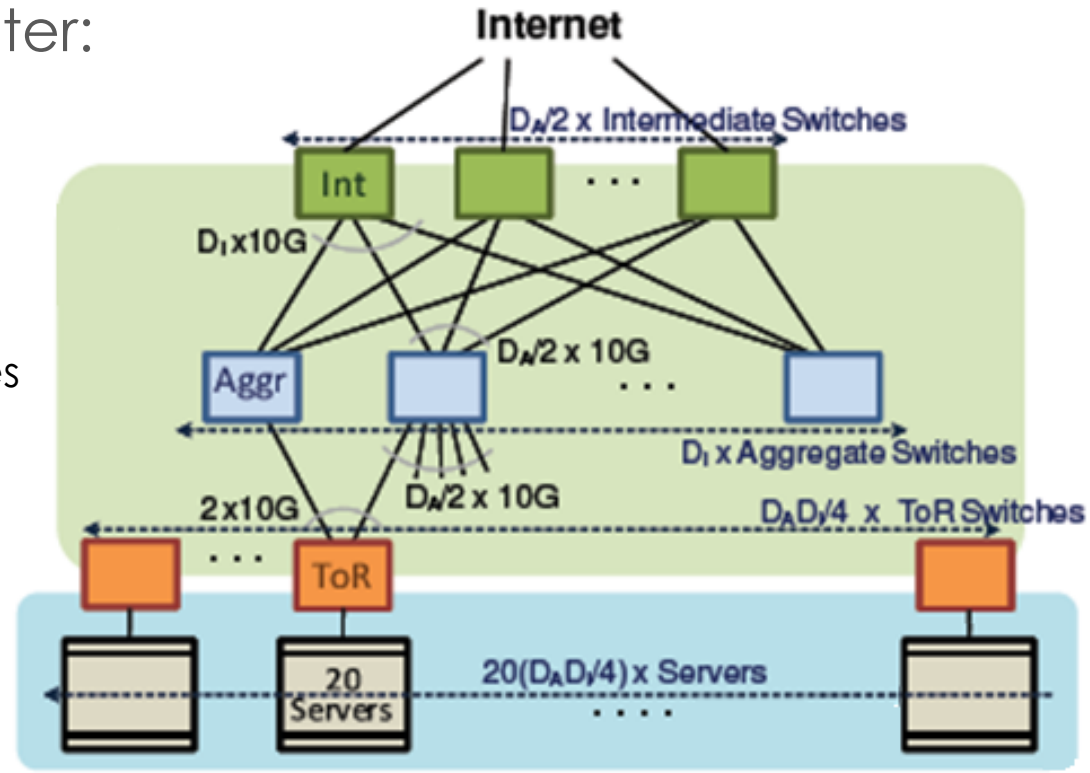
- Physical data center:

4 core switches

4 aggregation switches

4 top-of-rack switches

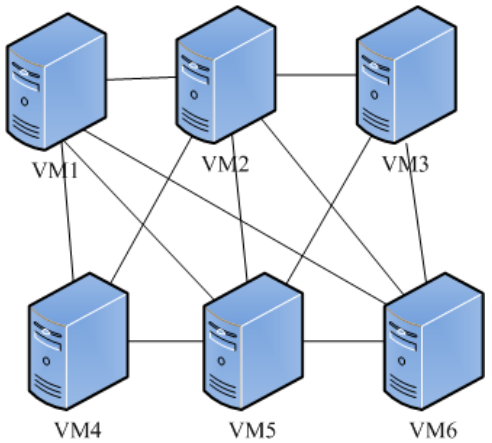
400 physical machines
(8 Cores, 8GB, 100 GB disk).



VL2 Topology

Experiment Set up

- VDC requests:
  - Number of VMs/VDC: [1-20]
  - VM requirements:
    - 1 – 4 cores
    - 1 – 2GB of RAM
    - 1 – 10GB of disk space
  - Virtual link capacity: [1-10 Mbps]
  - Arrival: Poisson distribution
    - 0.01 request/second during night time
    - 0.02 request/second during day time
  - VDC lifetime: exponential distribution (~3 hours)
  - Maximum waiting time: 1 hour



Performance Metrics

- Gain in acceptance Ratio

$$A_{m/n} = 100 \times \frac{A_m}{A_n} - 100$$

- Gain in revenue

$$G_{m/n} = 100 \times \frac{R_m}{R_n} - 100$$

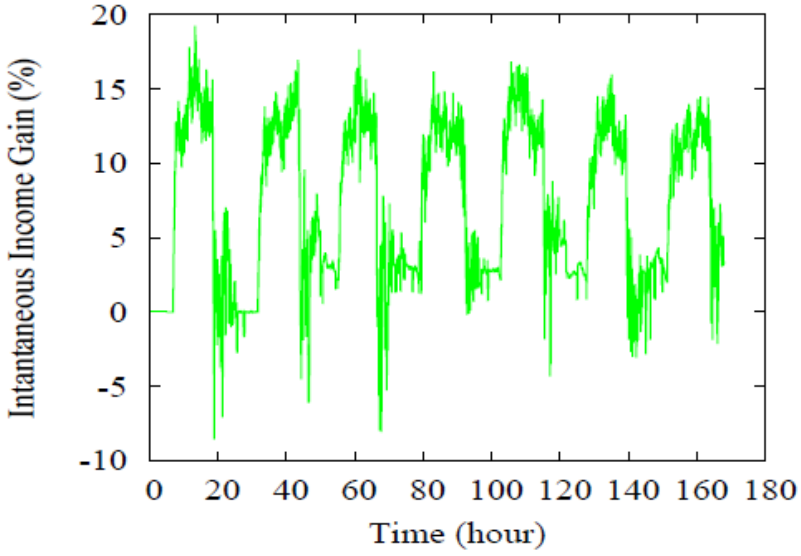
- Gain in number of active machines

$$M_{m/n} = 100 \times \frac{M_m}{M_n} - 100$$

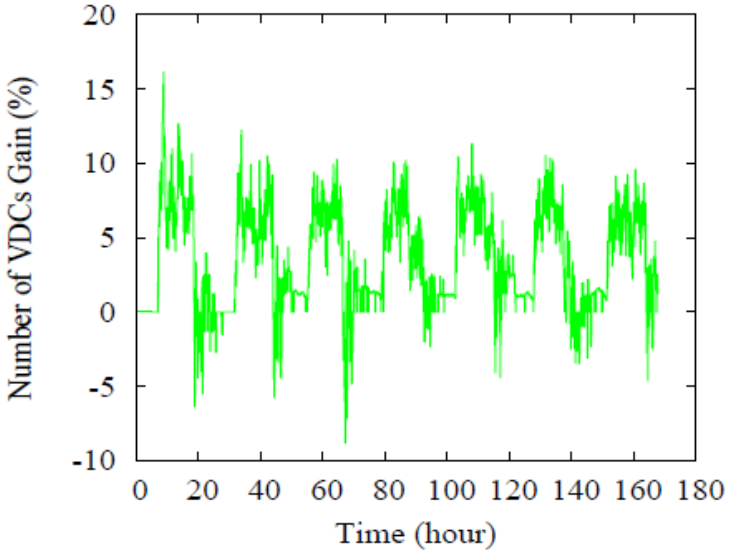
- Request scheduling delay

Results: Revenue & Acceptance Ratio

- Migration-aware Embedding vs. Baseline



(a) Instantaneous income gains

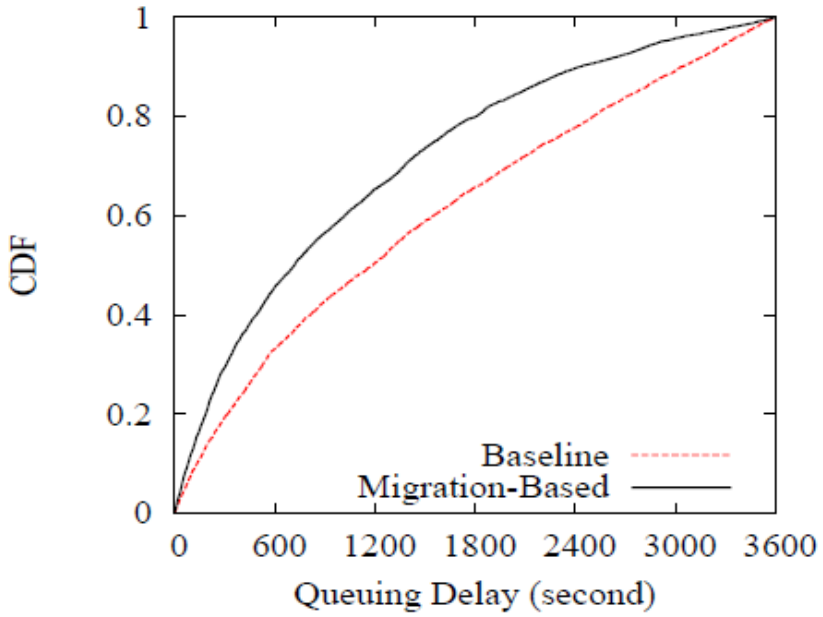


(b) Gain in acceptance ratio

- VDC planner achieves up to 17% gain in revenue and up to 10% gain in acceptance ratio.

Results: Queuing Delay

- Migration-aware Embedding vs. Baseline

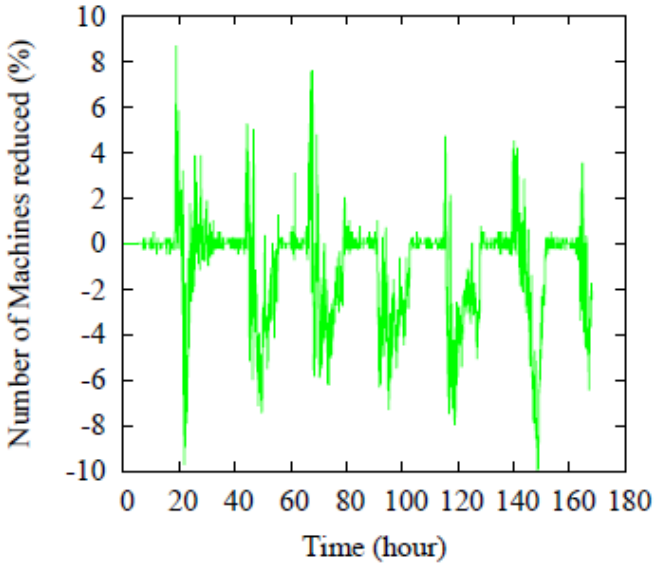


(c) Gain in terms of queuing delay

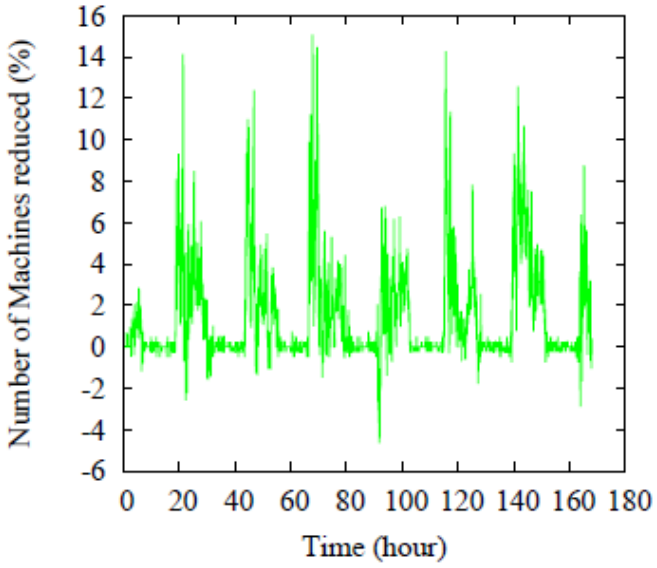
- VDC planner reduces the scheduling delay by up to 25%.

Results: With Consolidation

- Migration-Aware embedding + Consolidation



(a) Migration-aware algorithm



(b) Migration-aware embedding + consolidation

- VDC planner uses up to 14% less machines than the Baseline.

Outline

- Future Application Platforms: trends and challenges
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

VDC Reliability/Availability

- Reliability is a major concern of service providers
  - A service outage can potentially incur high penalty in terms of revenue and customer satisfaction
- Availability is a common reliability metric specified in SLAs
- VDC availability is dependent on
  - Service priority
  - VDC topology and replication groups
  - Hardware availability

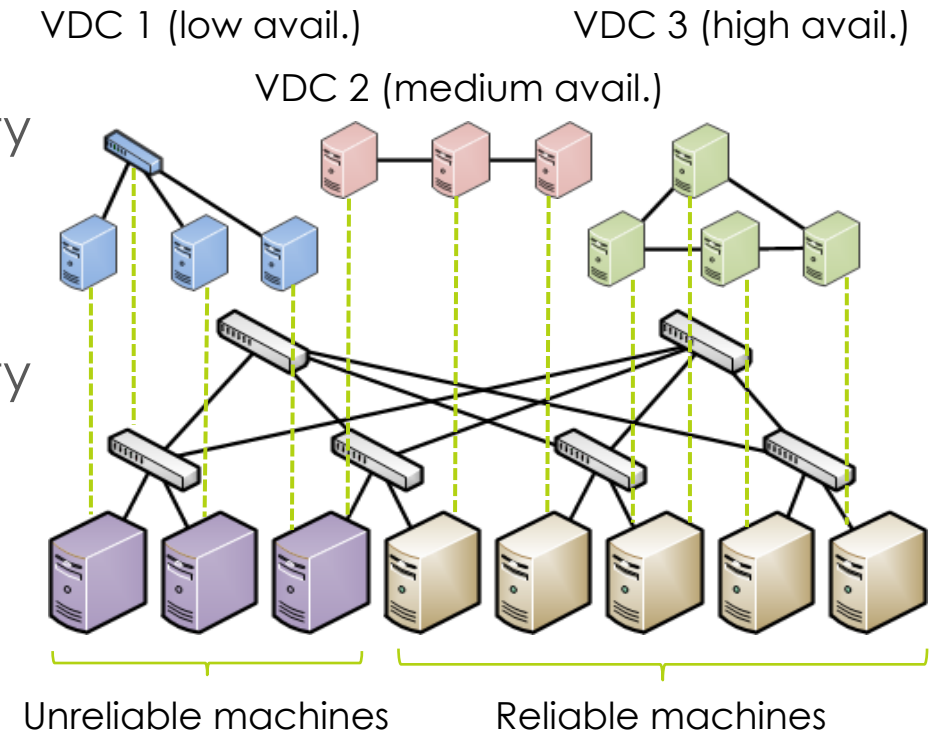
Understanding Data Center Failures

- Heterogeneous server failure rates*
  - Server that has experienced a failure is likely to fail again in the near future
- Network failure characteristics **
  - Failure rates of network equipment is type-dependent
    - Load balancers have high probability of failure (≥20%),
    - Switches often have low failure probability (≤5%).
  - Number of failures are unevenly distributed across equipment of the same type
    - E.g. Load balancer failures dominated by few failure prone devices
  - Correlated network failures are rare
    - More than 50% of link failures are single link failures, and more than 90% of link failures involve less than 5 links

* Vishwanath et al. "Characterizing Cloud Computing Hardware Reliability", SoCC 2010.
** Gill et. al. "Understanding network failures in data centers: measurement, analysis, and implications", SIGCOMM 2011.

Venice

- VDCs have heterogeneous availability requirements
- Resources have heterogeneous availability characteristics
- Place VDCs with high availability requirement on reliable machines



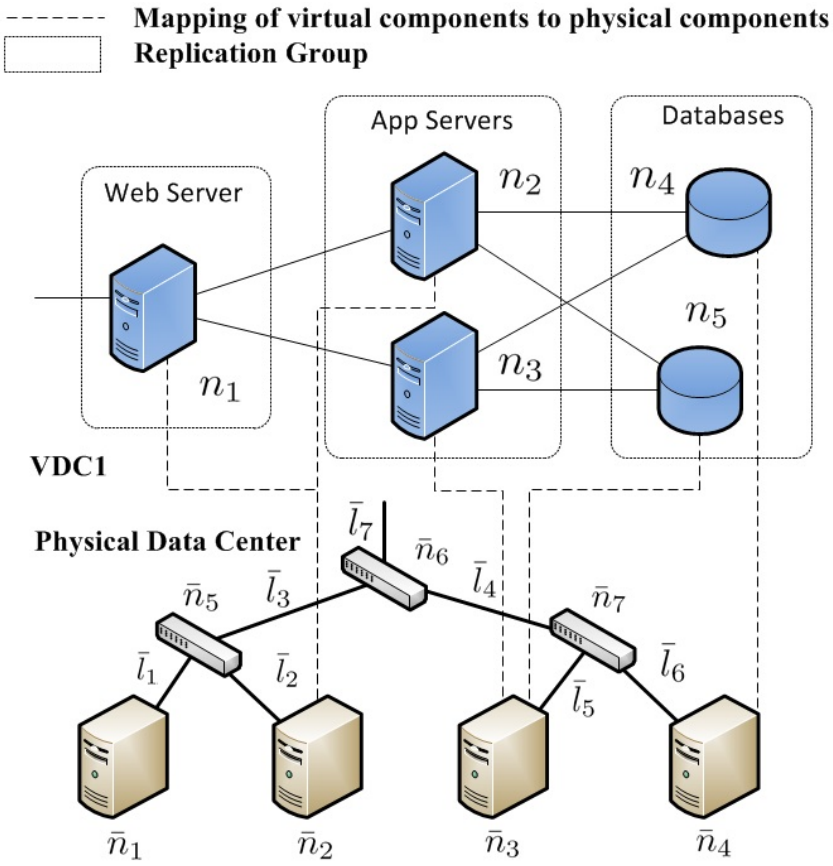
Q. Zhang, M. F. Zhani, M. Jabri, R. Boutaba. Venice: Reliable Virtual Data Center Embedding in Clouds. IEEE INFOCOM'14, Toronto, ON (Canada), April 27 - May 2, 2014.

Computing VDC Availability

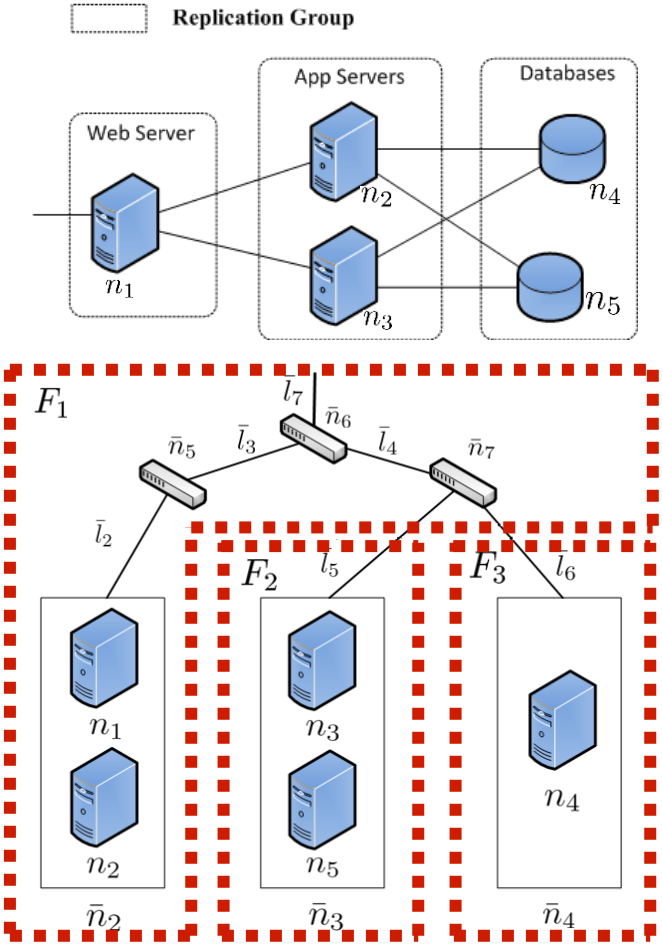
- Example of 3-tier application
- Availability of device *j*:

$$A_j = \frac{MTBF_j}{MTBF_j + MTTR_j}$$

- How to compute the availability of this VDC?



Computing VDC Availability (cont)



- Identify all possible failure scenarios S^k and compute the availability

Case 1: F1 unavailable,

$$A_{F_1} = 0$$

Prob. of occurrence: $P(F_i) = 1 - \prod_{i \in F_1} A_i$

Case 2: F1 available, F2 unavailable

$$A_{F_1} = \prod_{i \in F_2} A_i$$

Prob. of occurrence: $P(F_2) = (\prod_{i \in F_1} A_i)\,(1 - \prod_{i \in F_2} A_i)$

Case 3: F1 available, F2 available

$$A_{F_1} = 1$$

Prob. of occurrence: $P(F_2) = \prod_{i \in F_1 \cup F_2} A_i$

Using conditional probability, the availability of VDC_1 can be computed as:

$$A_{VDC_1} = \sum_{i=1}^{3} P(F_i) A_{F_i}$$

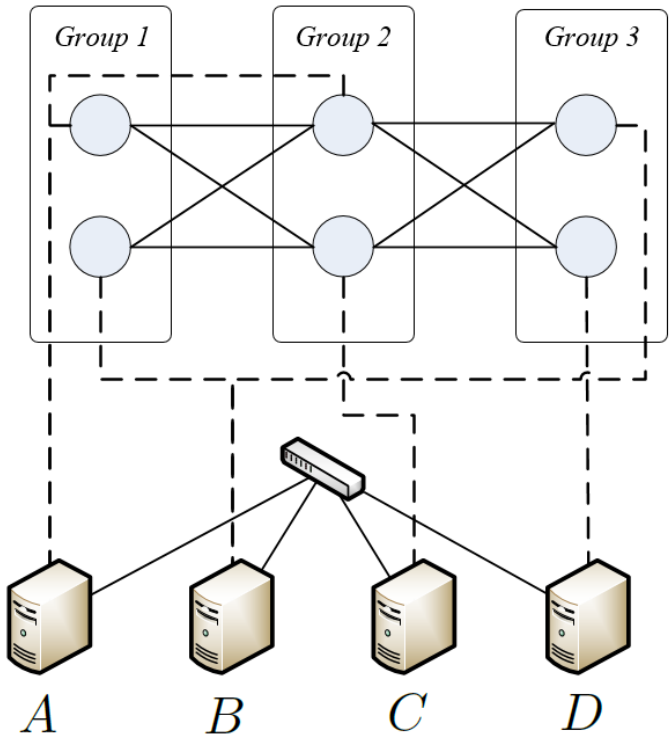
Computing VDC Availability (cont)

Theorem: VDC availability cannot be computed in polynomial time in the general case

Proof: Reduction from the counting monotone 2-Satisfiability problem

… Need to consider an exponential number of scenarios in the worst case!

$$f(A,B,C,D) = (A \vee B) \wedge (A \vee C) \wedge (B \vee D)$$



Computing VDC Availability (cont)

- Observation: it is unlikely to see large simultaneous failures
  - Given 3 nodes, each with availability > 95%, the probability of seeing all 3 nodes fail simultaneously is at most $(1-0.95)^3<0.00013$
- A fast heuristic:
  - Compute availability using scenarios S^k that involve at most 3 simultaneous failures
- Fast heuristic provides a lower bound on VDC availability

Problem Formulation

- Objective function:

$$\min C_E + C_M + C_A$$

- Where

$$C_E = \sum_{\bar{n} \in \bar{N}} y_{\bar{n}} p_{\bar{n}}$$ (Resource cost)

$$C_M = \sum_{i \in I} \sum_{n \in N^i} \sum_{\bar{n} \in \bar{N}} \gamma_n x^i_{n\bar{n}} g^i_{n\bar{n}}$$ (Migration cost)

$$\mathcal{C}_A = \sum_{i \in I} (1 - A_i)\pi_i + \sum_{\bar{n} \in \bar{N}} F_{\bar{n}} C^{restore}_{\bar{n}} + \sum_{\bar{l} \in \bar{L}} F_{\bar{l}} C^{restore}_{\bar{l}}$$ (Failure cost)

Greedy Scheduling Algorithm

- For each received VDC request
  - Initial embedding: embed one node from each replication group.
  - Repeat
    - For each remaining component compute a score as the availability improvement - resource cost
    - Embed the component with the highest score
  - Until the VDC availability is achieved or all nodes are embedded
  - Embed the remaining components greedily based solely on resource cost

Experiments

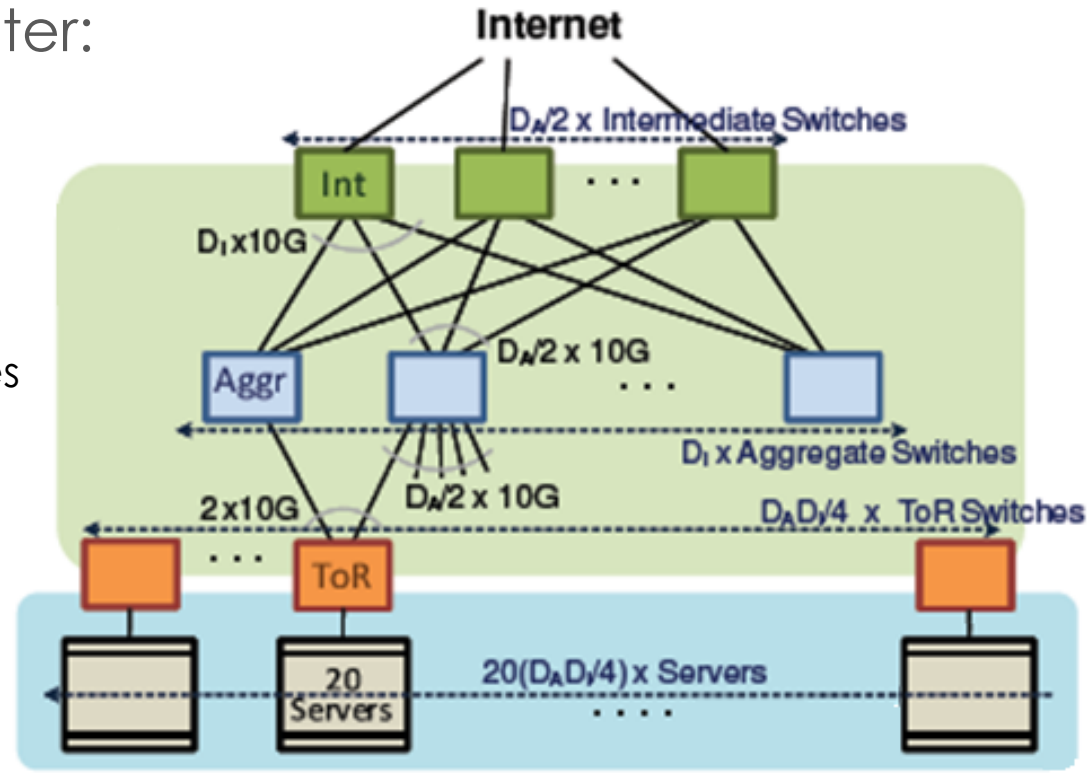
- Physical data center:

4 core switches

4 aggregation switches

4 top-of-rack switches

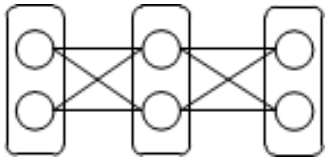
400 physical machines
(8 Cores, 8GB, 100 GB disk).



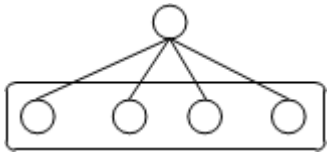
VL2 Topology

Experiment Setup

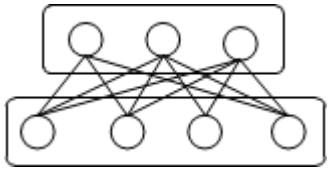
- VDC request formats
  - From 1 to 10 VMs per group
  - Different availability requirements
- VDC Planner used as a baseline for comparison



(a) Multi-tiered



(b) Partition-Aggregate

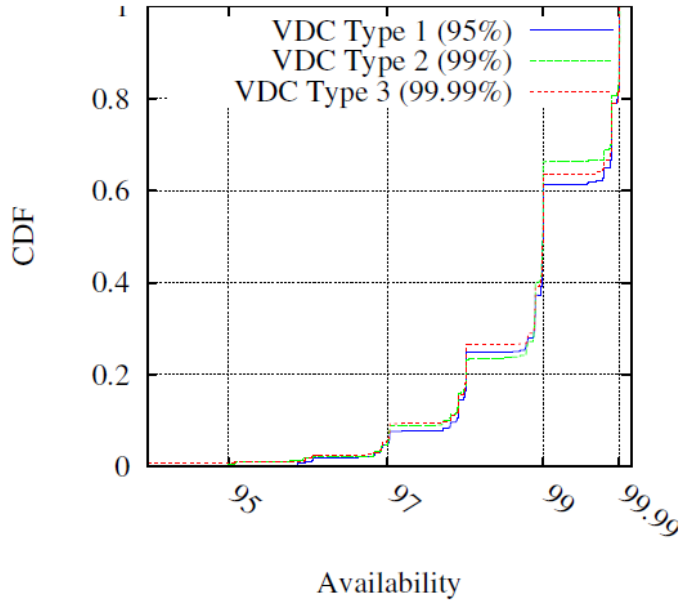


(c) Bipartite

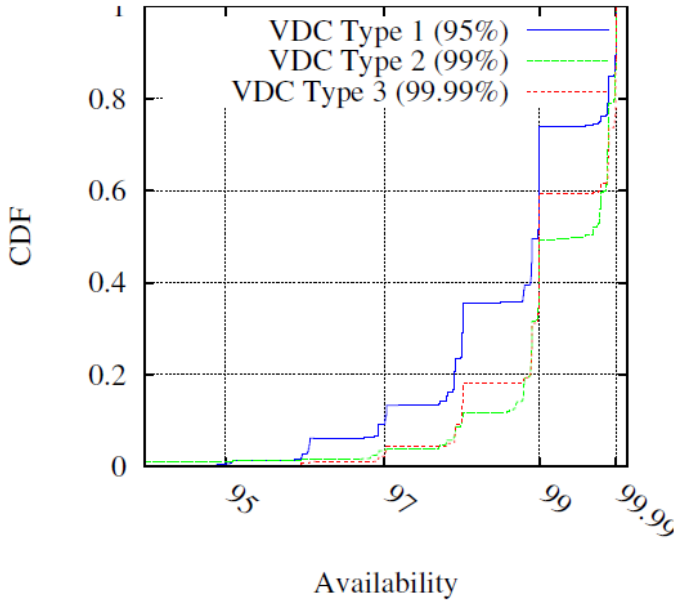
TABLE I: VDC Availability requirements

VDC Type	Minimum Required Availability (%)	Acceptable daily downtime
1	95.00	1h:12mn
2	99.00	14mn:2s
3	99.99	08.64s

Results: Availability



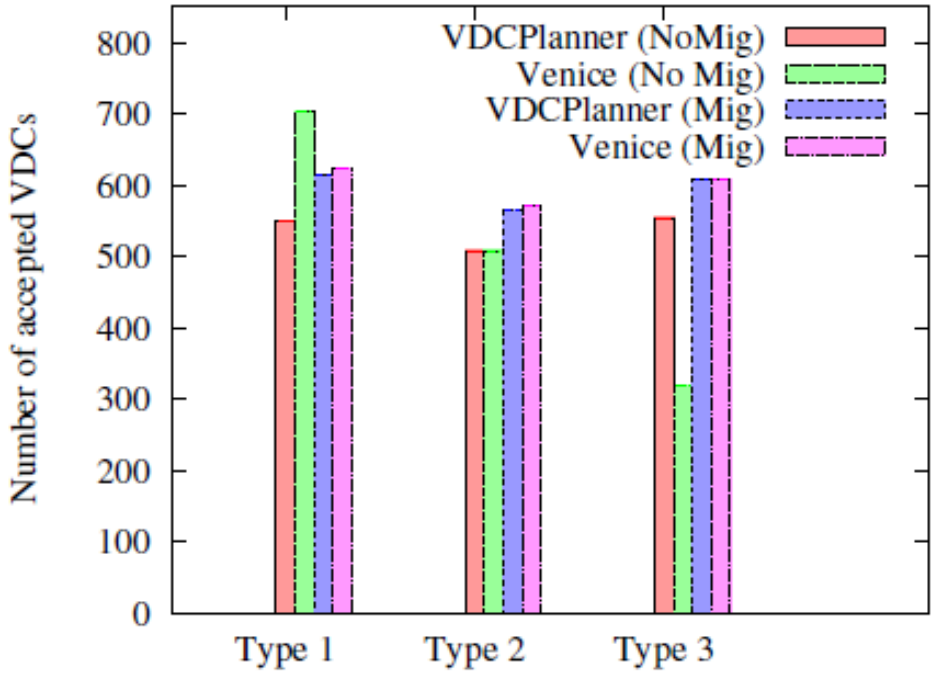
(a) VDC Planner (using migration)



(b) Venice (using migration)

- Venice increases the number of VDCs satisfying availability requirements by up to 35%

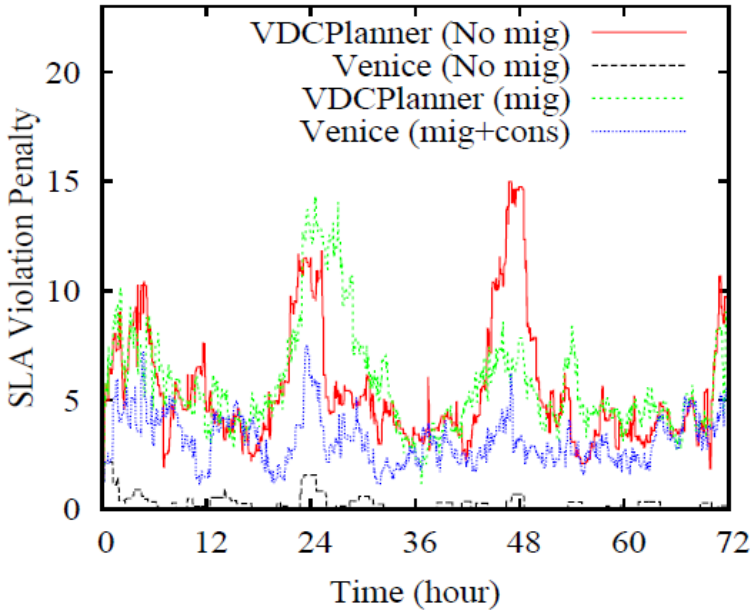
Results: Acceptance Ratio



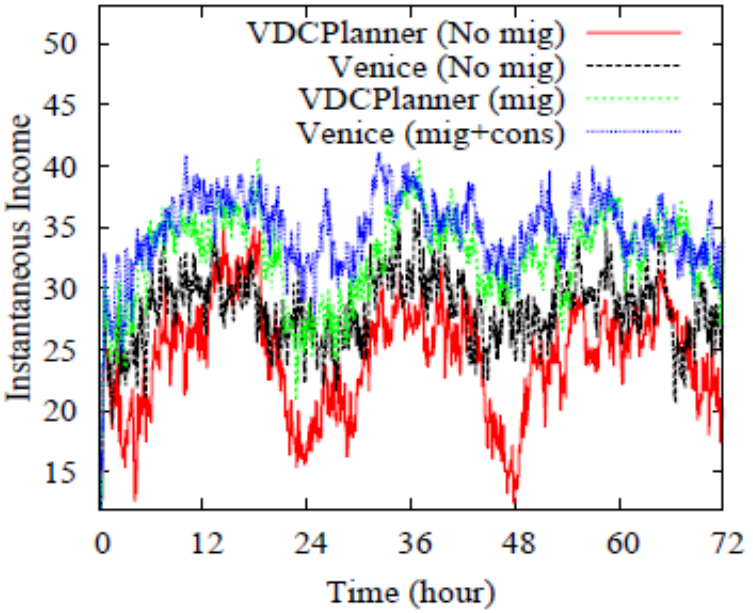
Number of accepted VDCs

- With migration, the number of accepted VDCs is comparable to that of VDC Planner

Results: Revenue



SLA Violation Cost



Instantaneous Income Rate

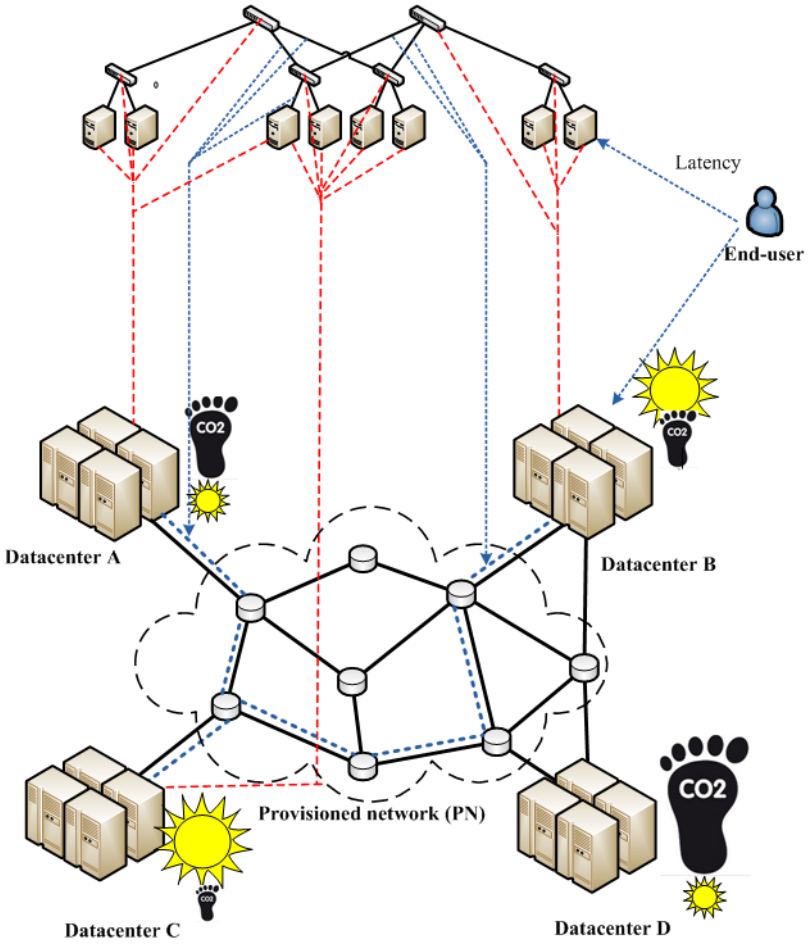
- Venice achieves 15% increase in revenue compared to VDC Planner

Outline

- Future Application Platforms: trends and challenges
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

Business Model

- Infrastructure Providers (InPs) build geographically distributed data centers
- InPs rent resources in the form of Virtual Data Centers (VDCs)
- VDCs geo-distributed to reduce energy cost and carbon footprint and/or to satisfy location constraints



InP Operational Objectives

- Satisfy performance requirements (e.g., latency)
- Reduce energy cost
  - Use data centers with low electricity prices
  - Reduce use of power from the grid
- Reduce carbon footprint
  - Use local renewable energy available at the data centers
  - Use source of power with the minimal carbon emission
- Reduce traffic in backbone network

Challenges

- Sources of clean energy for data centers:
  - Locally-available renewable energy sources (e.g., solar, wind)
  - Limited and dependent on location and weather conditions
- Electric grid:
  - Fluctuating prices of electricity and high carbon footprint
- Price and carbon footprint differ from one location to another

Greenhead Problem Formulation

- ILP decision variables:

$$x_{ik}^{j} = \begin{cases} 1 & \text{If the VM } k \text{ of the VDC } j \text{ is assigned to data center } i \\ 0 & \text{Otherwise.} \end{cases}$$

$$f_{e,e'} = \begin{cases} 1 & \text{If the physical link } e \in E \text{ is used to embed the virtual link } e' \in E^{j} \\ 0 & \text{Otherwise.} \end{cases}$$

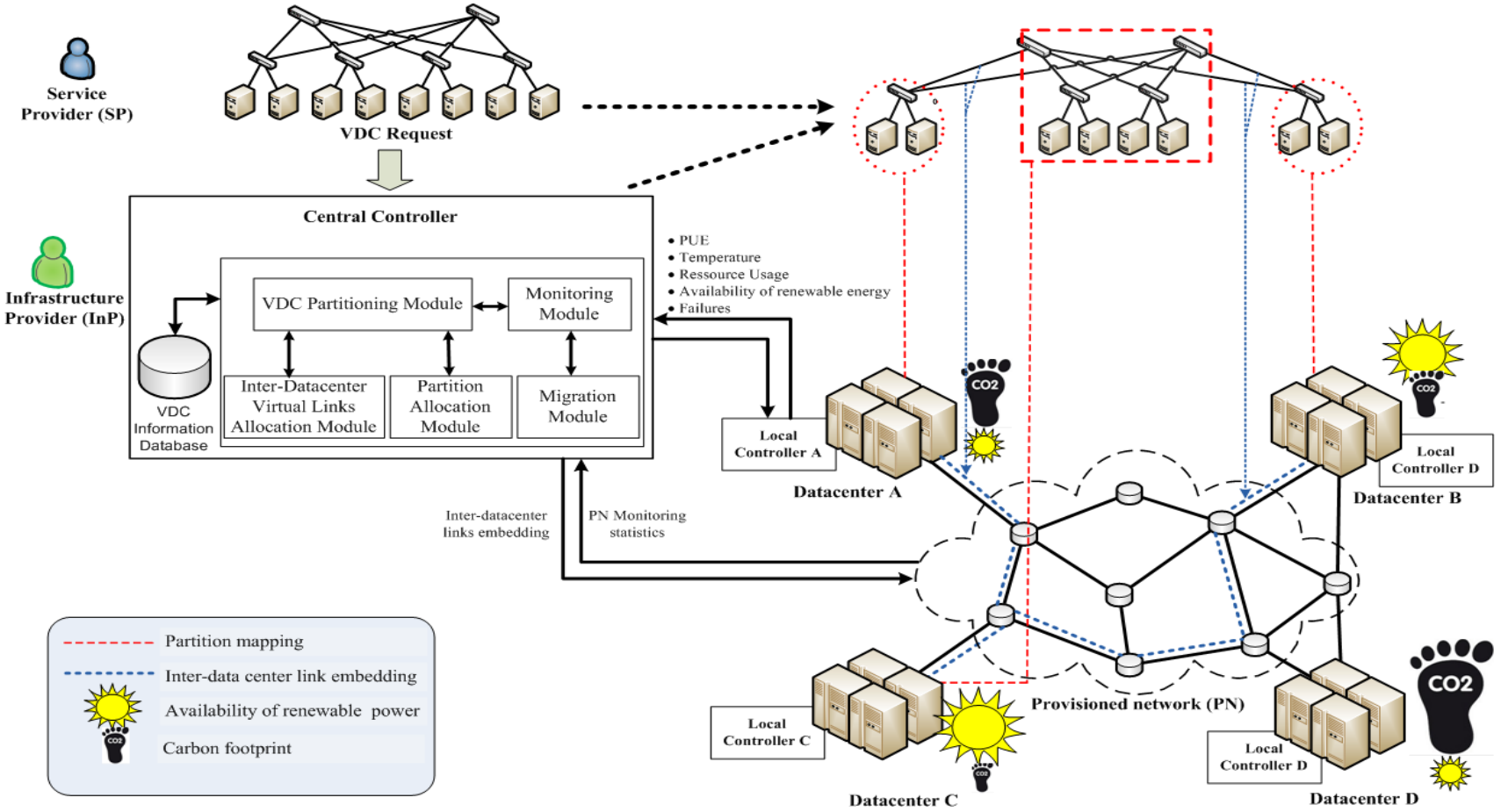
- Objective Function:

Maximize $R^j - (D^j + P^j)$

  - R^j: Revenue for VDC request j
  - D^j: Embedding cost for request j in data centers
  - P^j: Embedding cost for request j in the backbone network

A. Amokrane, M. F. Zhani, R. Langar, R. Boutaba, G. Pujolle. Greenhead: Virtual Data Center Embedding Across Distributed Infrastructures. IEEE Transactions on Cloud Computing (TCC), Vol 1(1), pp. 36-49, September 2013

Greenhead Framework

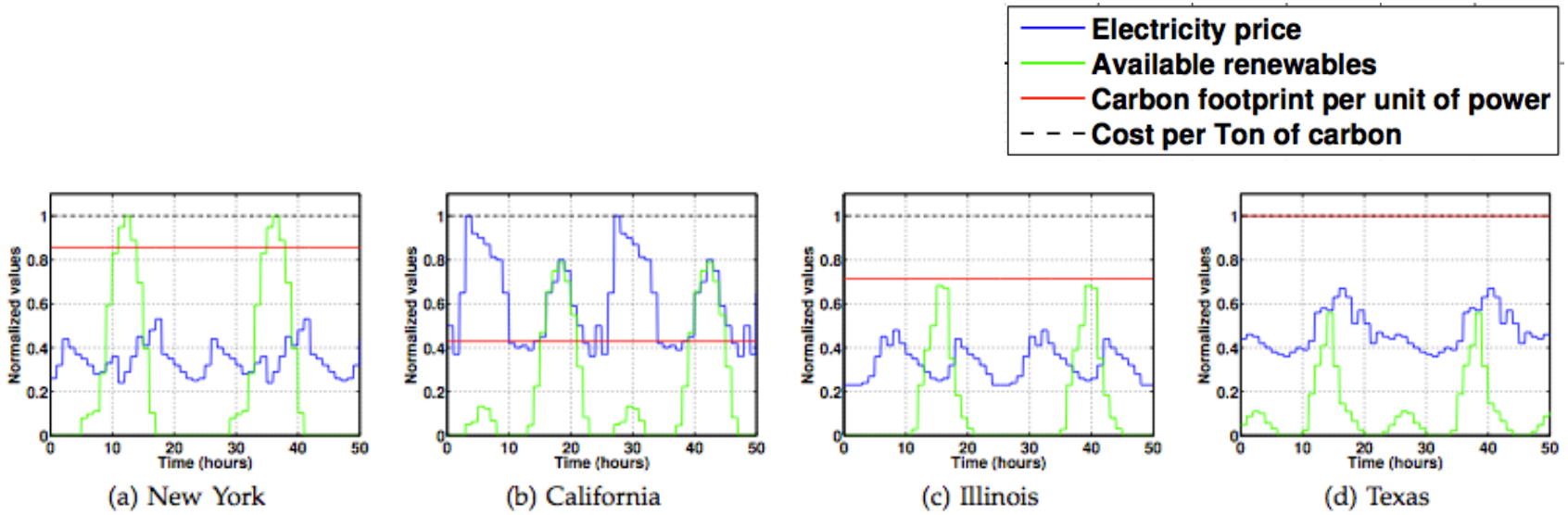


Greedy Algorithm

- Step 1 - VDC Partitioning: Location Aware Louvain Algorithm
  - Split the VDC requests into partitions with high intra-partition bandwidth demand and low inter-partition bandwidth demand
  - Minimize inter-data center traffic
- Step 2 – Partition embedding: Greedy partition assignment to data centers
  - Assign each partition to the data center that:
    - Satisfies location and delay constraints
    - Minimizes electricity costs and carbon footprint

Experiments

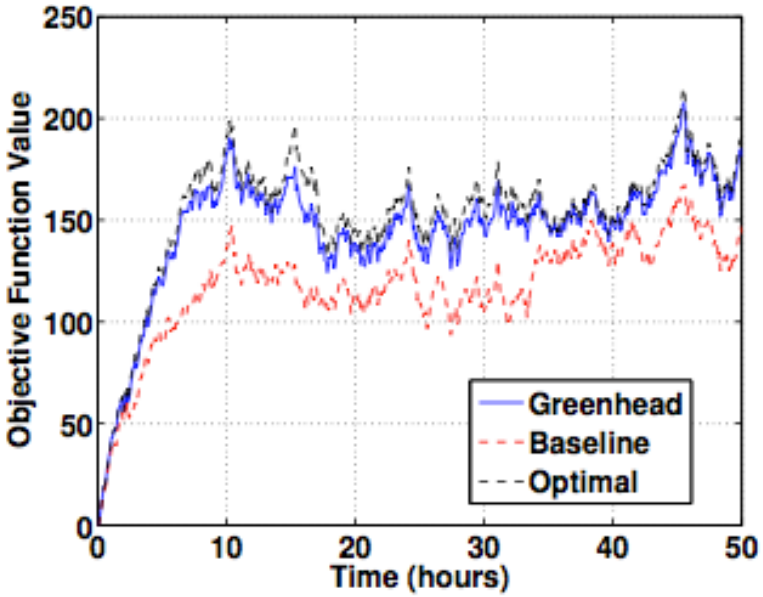
- Physical infrastructure:
  - 4 data centers located in New York, California, Illinois and Texas
  - Backbone network: NSFNet network



Experiments Setup

- VDC requests:
  - Number of VMs randomly generated between 5 and 10 for small-sized VDCs and between 20 and 100 for large-sized VDCs
  - Virtual links randomly created with a bandwidth demand between 10 and 50 Mbps
  - Poisson arrivals (8 requests per hour) and exponential lifetime (average 6 hours)
  - 15 % of the VMs have location constraint
- Baseline approach: Greenhead without partitioning

Results: VDC Embedding



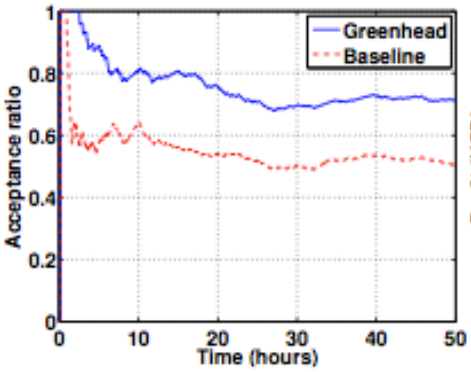
Objective function

VDC size	Greenhead			Baseline	ILP Solver
	Partitioning	Embedding	Total		
5-10 VMs	0.214	0.061	0.079	0.275	13540
20-100 VMs	31.41	0.28	31.69	2.2	-

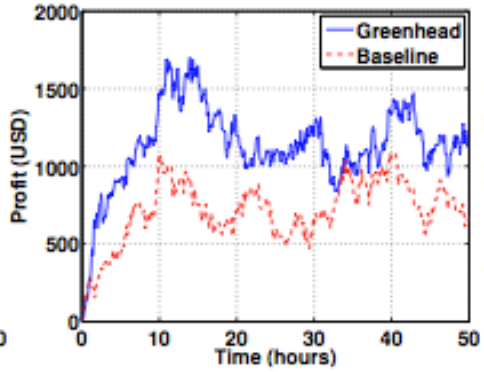
Computation Time (ms)

- Greenhead provides near-optimal solution within a reasonable time frame

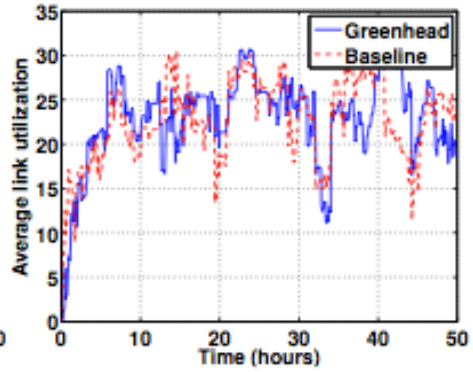
Results: Acceptance, Revenue, Utilization



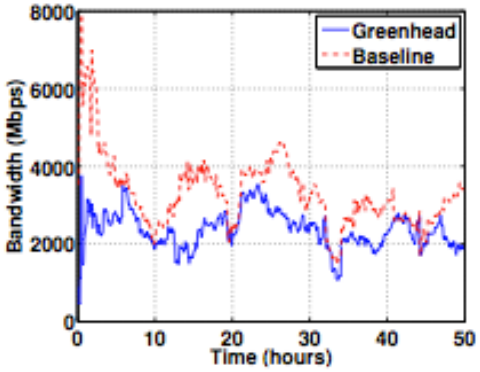
(a) Acceptance ratio



(b) Instantaneous profit



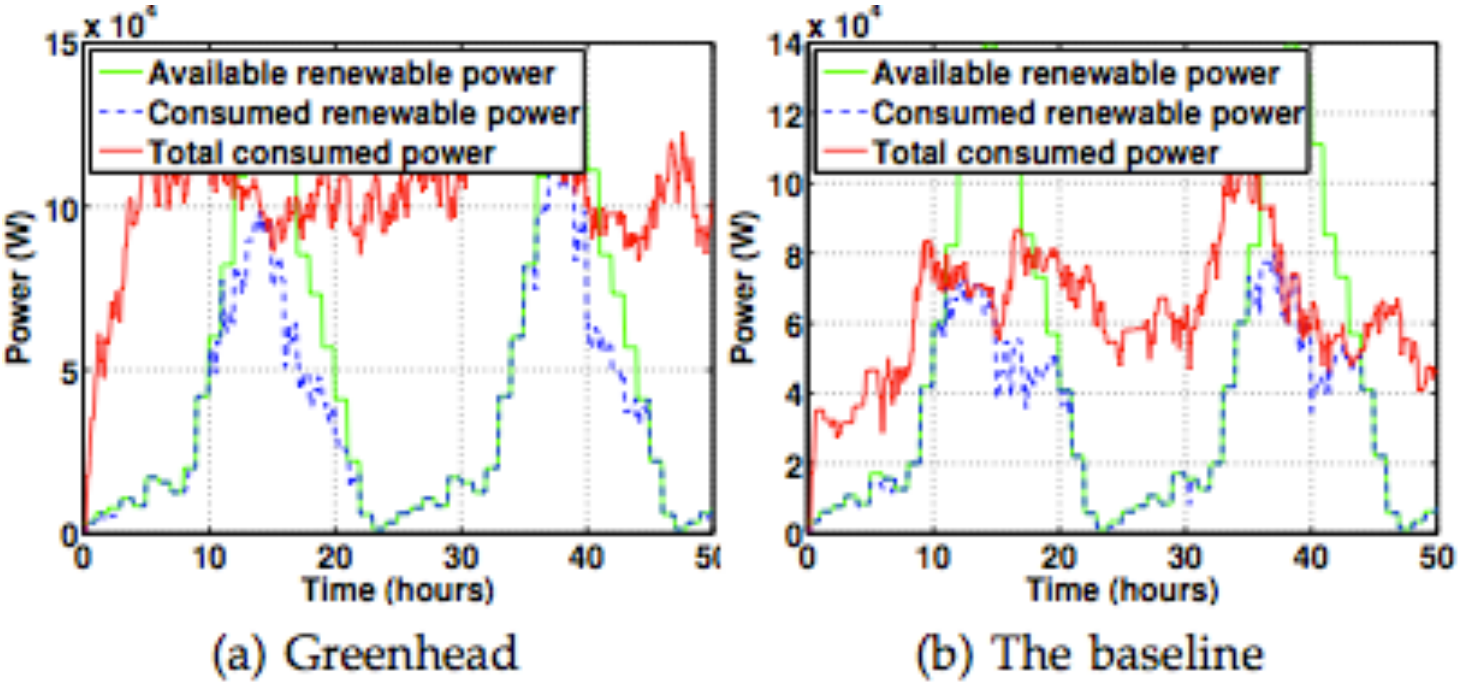
(c) Average link utilization in the backbone network



(d) Used bandwidth per request in the backbone network

- Greenhead achieves higher acceptance ratio, higher revenue and better backbone network utilization

Results: Renewable Energy



- Greenhead maximizes the utilization of available renewable energy

Outline

- Future Application Platforms: trends and challenges
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

NFV Orchestration

- Middleboxes have become an integral part of modern networks
- Traditional hardware middleboxes are:
  - Expensive
  - Proprietary
  - Vertically integrated
- Difficult to compose Service Function Chains
  - In a service function chain traffic flows through an ordered sequence of middleboxes
    - Example:
      - Firewall → IDS → Proxy
      - Traffic Analyzer → Firewall → Video Optimizer

Network Function Virtualization

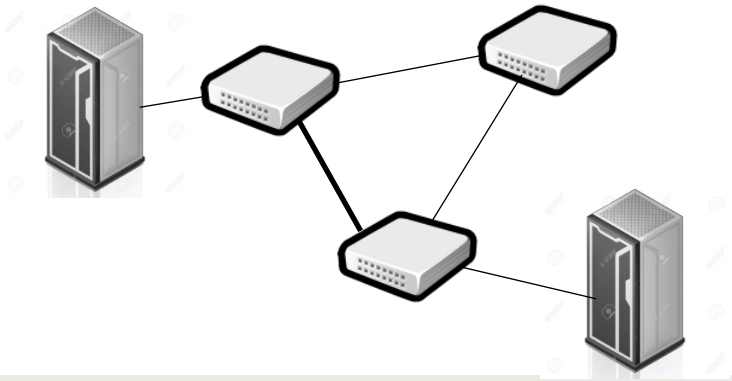
- introduced to overcome problems with HW middleboxes
- Basic idea:
  - Packet processing by software middleboxes or Virtualized Network Functions (VNFs)
  - VNFs can be deployed on commodity servers
    - E.g., x86 based systems
  - VNFs are no longer constrained to fixed network locations
  - Service function chains can be composed on the fly
- These features are expected to facilitate network optimization

NFV Orchestration Contributions

- Two important aspects of NFV Management and Orchestration:
  - Service chain orchestration
  - API for NFV management and Orchestration

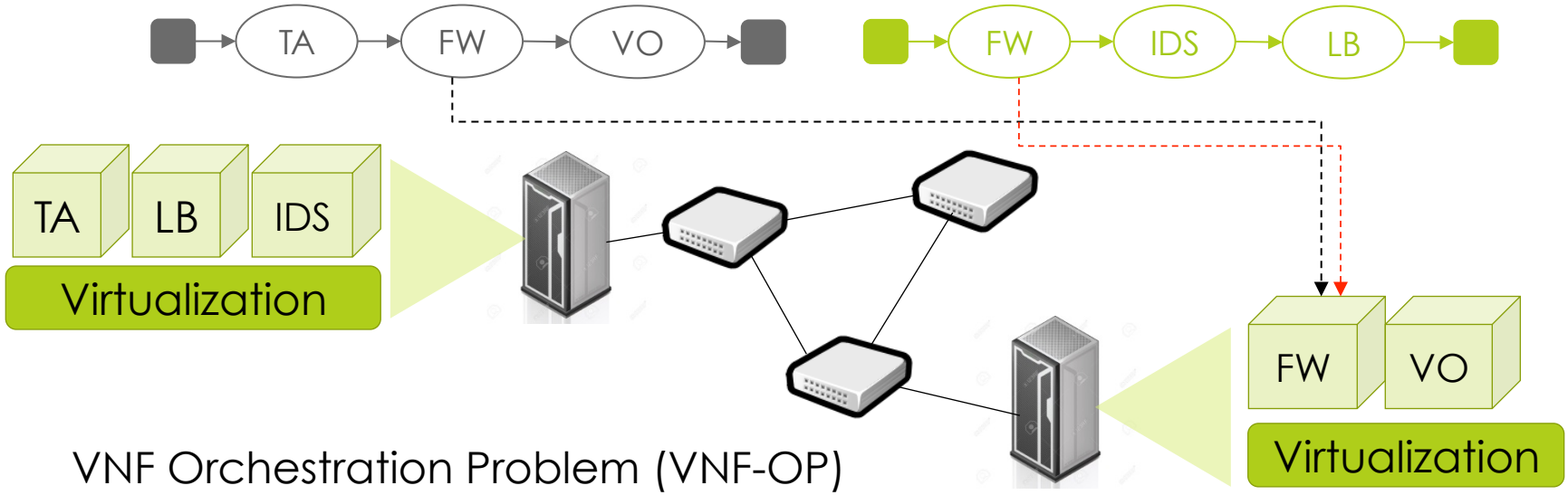
Service Chain Orchestration: *Problem Statement*

- Given
  - A set of VNF chain requests
  - Physical infrastructure status



Problem Statement (Cont.)

- We need to decide
  - How many VNF instances (VM, container) to deploy?
  - Where to place them?
  - Which VNF (from chain) should be assigned where?
  - How to route traffic between the VNFs?



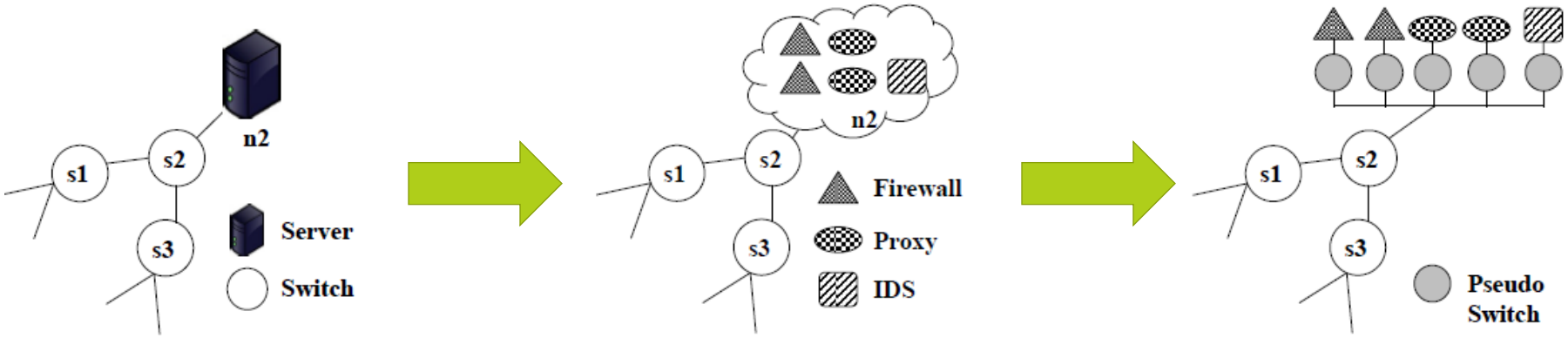
VNF Orchestration Problem (VNF-OP)

VNF Orchestration Problem

- VNF-OP is a combination of three problems:
  - Allocating resource for VMs/containers
  - Assigning chain VNFs to these VMs
  - Finding routing paths for the chains
- Mathematical formulation is difficult:
  - Joint optimization results in quadratic constraints
  - Takes a long time to solve even for small problem instances

Optimal Solution

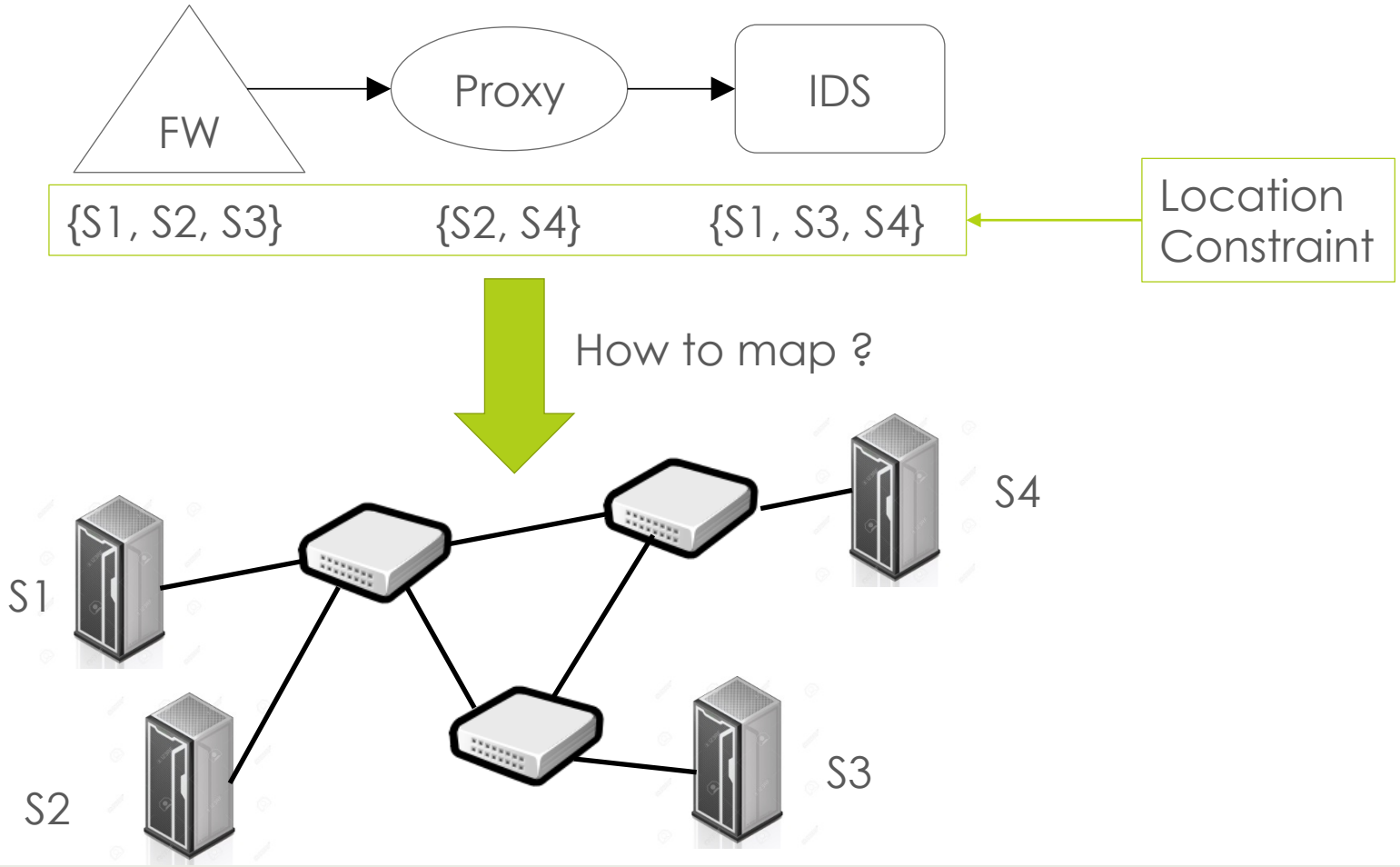
- Proposed approach:
  - Transform physical network



  - Can be formulated as an ILP
  - Much faster than implementation with quadratic constraints

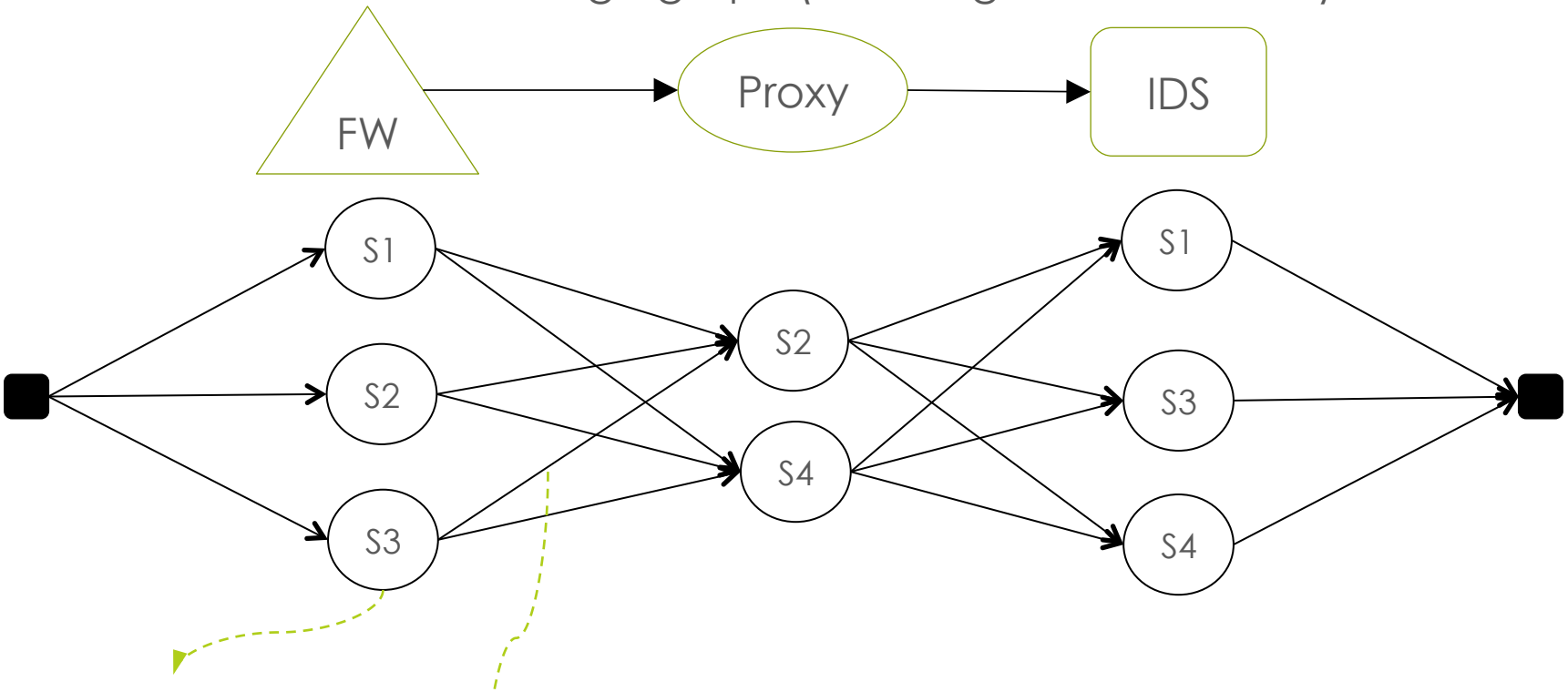
M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba. On Orchestrating Virtual Network Functions. IEEE/ACM/IFIP CNSM, Barcelona (Spain), November 9-13, 2015.

Heuristic


FW
Proxy
IDS
{S1, S2, S3}
{S2, S4}
{S1, S3, S4}
Location Constraint
How to map ?
S1
S2
S3
S4

Proposed Solution: Heuristic (cont.)

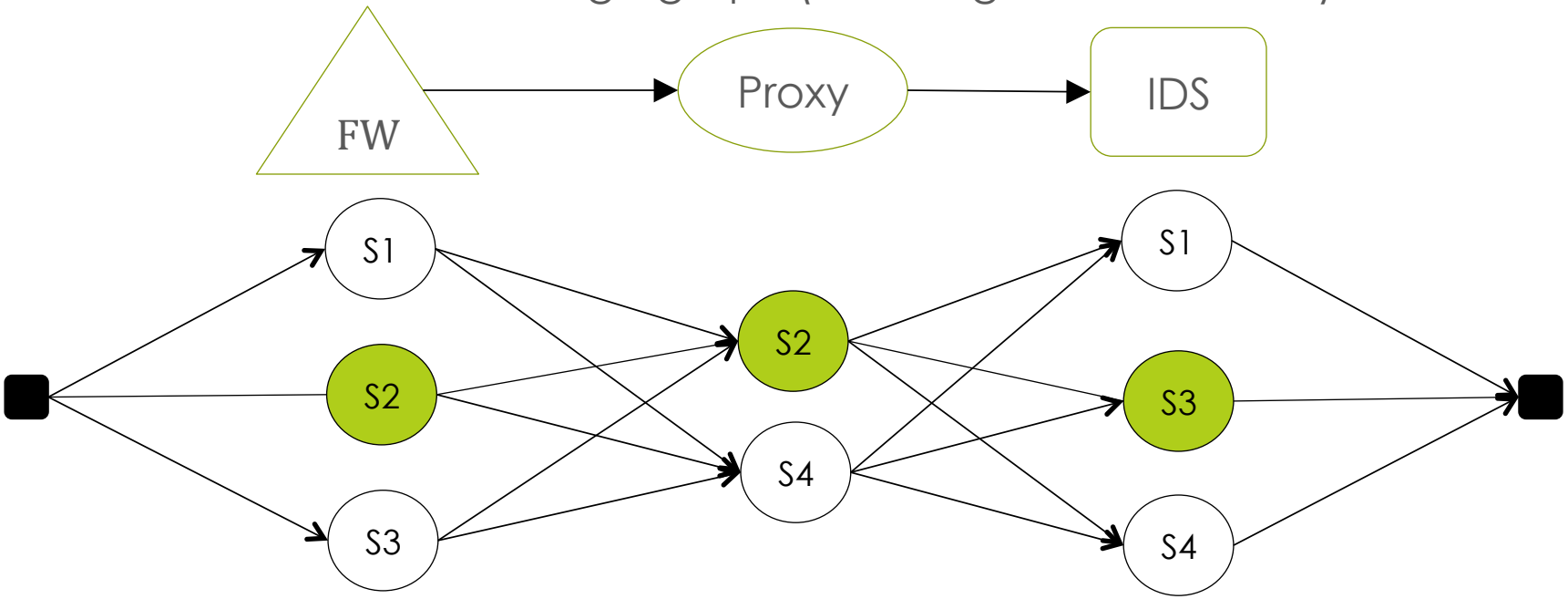
- Create a multi stage graph (one stage for each VNF) as follows:



- Objective: Find a path from left most to right most stage that has minimum cost
  - Select exactly one node at each stage

Heuristic (cont.)

- Create a multi stage graph (one stage for each VNF) as follows:



- Objective: Find a path from left most to right most stage that has minimum cost
  - Select exactly one node at each stage
- Similar to assigning tags to an unknown sequence of observations based on known cost function

Solution: Use Viterbi algorithm to find the minimum cost path

- Viterbi is widely used in pattern recognition to assign tags to unknown sequences of observations.

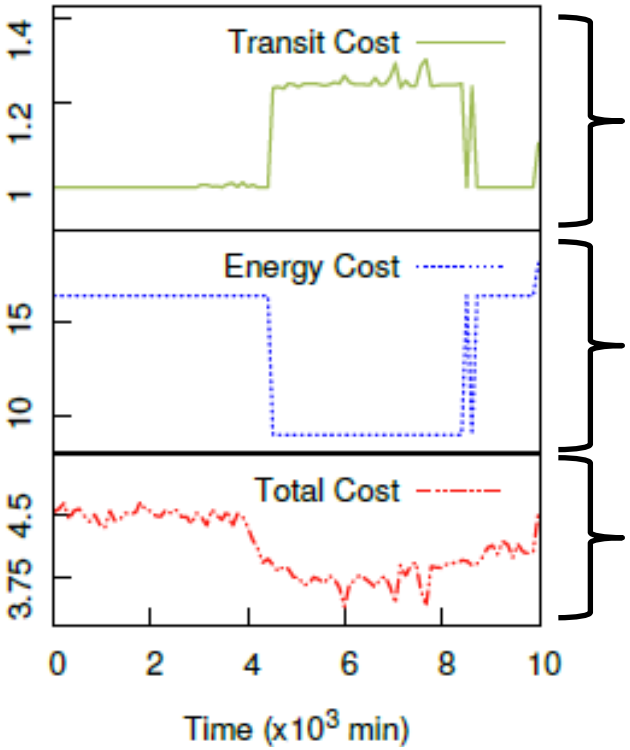
Evaluation: Setup

- Two network topologies:
  - Internet2 research network (12 nodes, 15 links)
  - A university data center topology (23 nodes, 42 links)
- Server energy consumption data collected from Intel datasheet
- Hardware middlebox energy consumption data collected from a manufacturer
- Traffic traces
  - Traffic matrix from Internet2 network
  - Data center traffic trace from *

* T. Benson et al. Network traffic characteristics of data centers in the wild. ACM IMC '10

Evaluation: Results

- Hardware Middlebox vs. VNF



Computed Ratios

$$\frac{Transit\ Cost\ (hardware)}{Transit\ Cost\ (VNF)}$$

$$\frac{Energy\ Cost\ (Hardware)}{Energy\ Cost\ (VNF)}$$

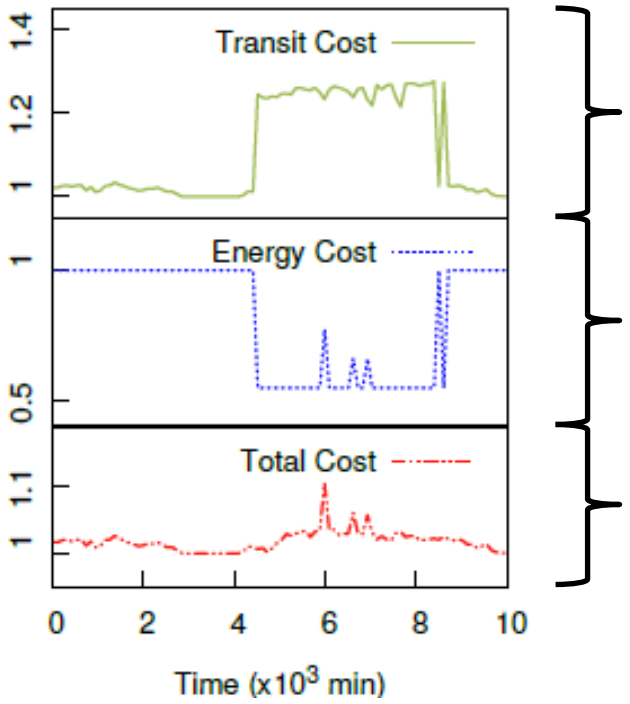
$$\frac{Total\ Cost\ (Hardware)}{Total\ Cost\ (VNF)}$$

$$Total\ cost = Transit\ Cost + Energy\ Cost$$

- VNF provides a 4 x reduction in total cost

Evaluation: Results

- Solution Quality: Heuristic vs. Optimal



Computed Ratios

$$\frac{Transit\ Cost\ (Heuristic)}{Transit\ Cost\ (Optimal)}$$

$$\frac{Energy\ Cost\ (Heuristic)}{Energy\ Cost\ (Optimal)}$$

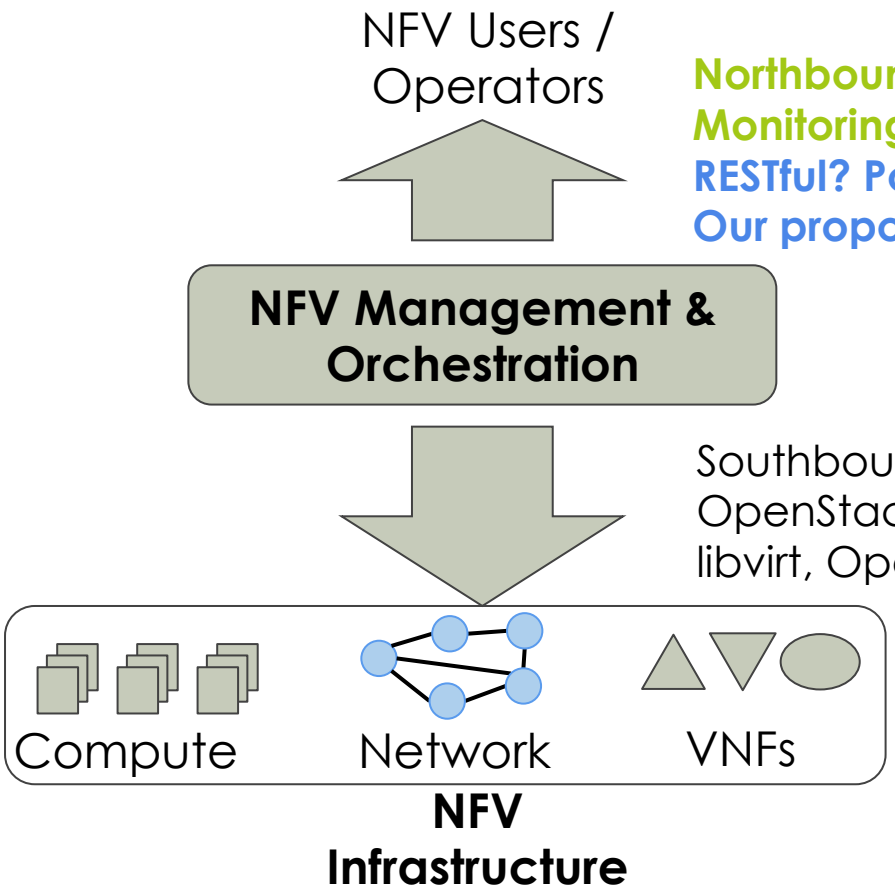
$$\frac{Total\ Cost\ (Heuristic)}{Total\ Cost\ (Optimal)}$$

$$Total\ cost = Transit\ Cost + Energy\ Cost$$

Summary of Results

- 4x OPEX reduction by VNFs compared to hardware middleboxes
- Heuristic produces solutions that are within 1.3x the optimal solution
- Heuristic is faster then the optimal
  - 65x for Internet2
  - 3500x for DC network

NFV Management & Orchestration API



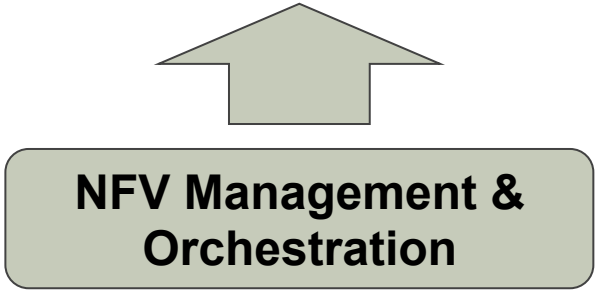
Northbound API for Deploying, Configuring, and Monitoring Virtual Network Functions (VNFs)
RESTful? Policy Language? other?
Our proposal: nfio

Southbound API:
OpenStack nova/neutron, libvirt, OpenFlow etc.

** Simplified view of ETSI Reference NFV Architecture*

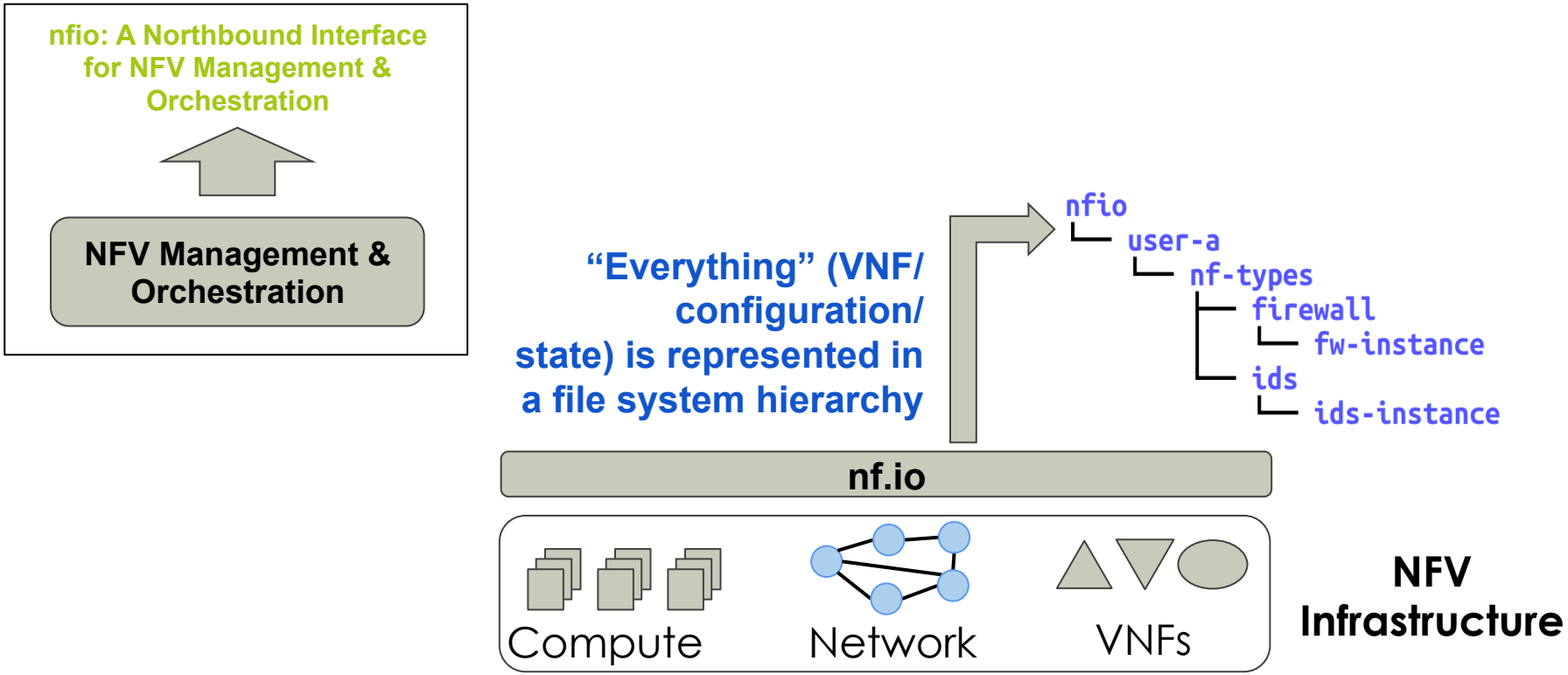
What is nf.io ?

nf.io: A Northbound Interface for NFV Management & Orchestration



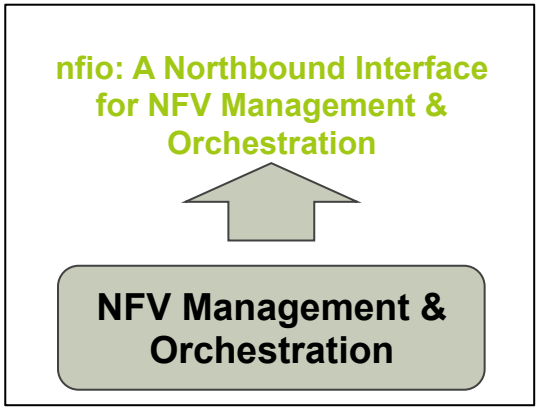
uses Linux File system to abstract the resources.

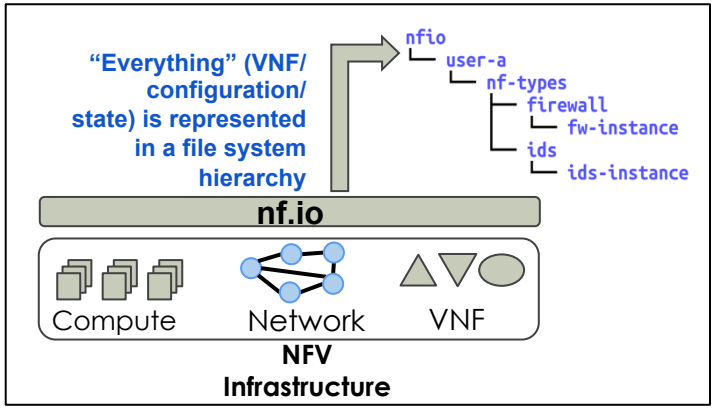
What is nf.io ?

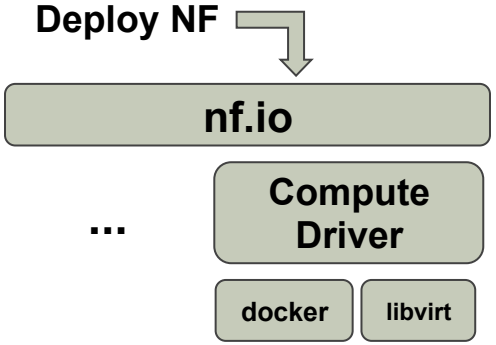


nf.io provides virtual files as placeholders to write VNF configurations as well as read VNF states.

What is nf.io ?



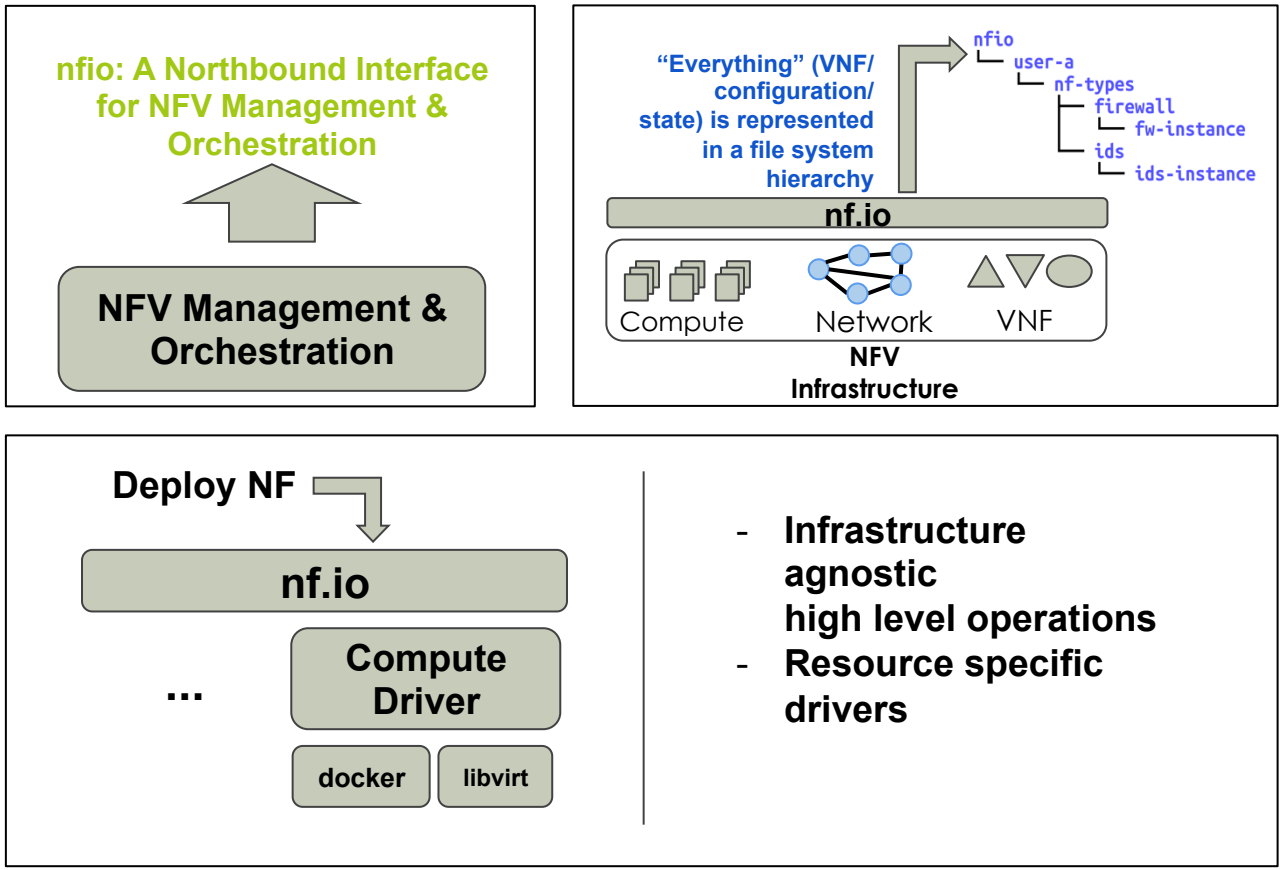




- **Infrastructure agnostic API**
- **high level operations (hides underlying details): file system like API**

- **Resource specific drivers**

What is nf.io ?



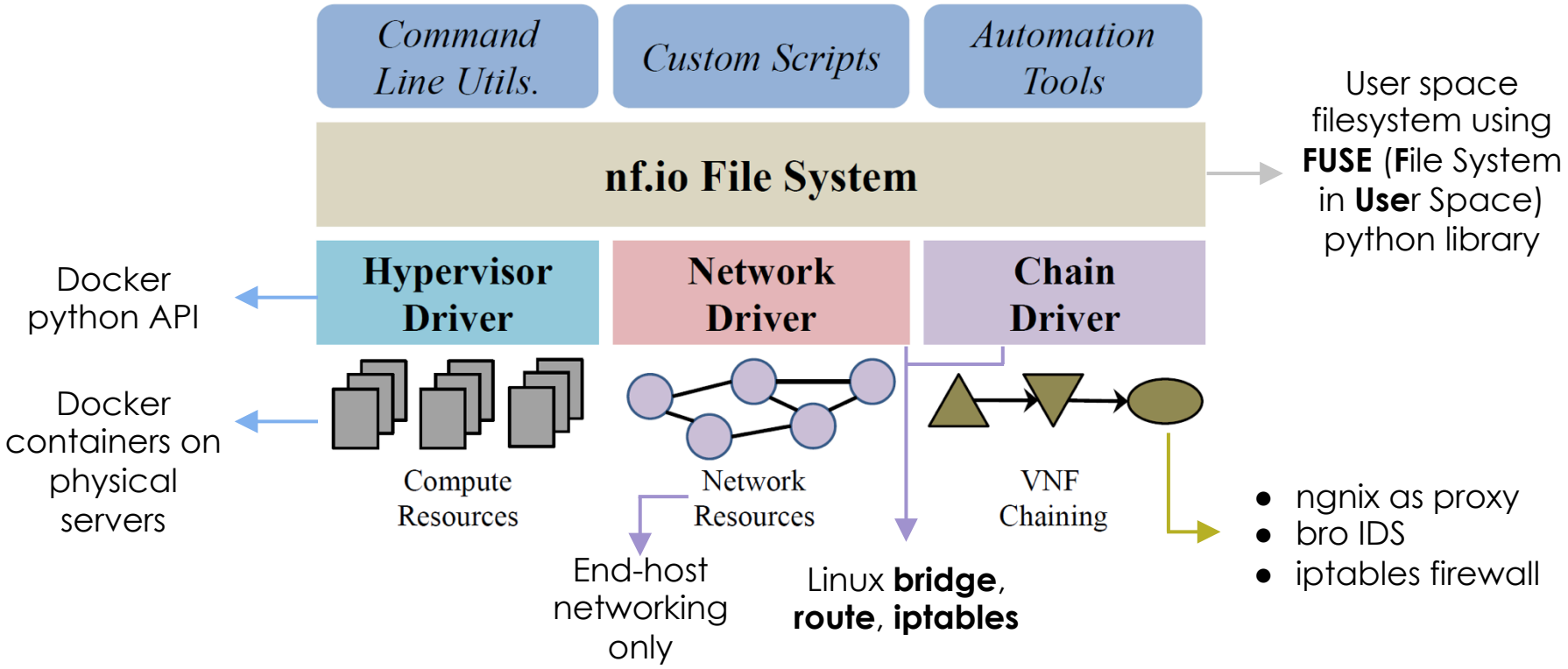
M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba. nf.io: A File System Abstraction for NFV Orchestration. IEEE NFV-SDN, San Francisco (USA), November 18-21, 2015.

Why File System Abstraction ?

- Familiar tools to manage file systems
  - mkdir, cp, move, rm, rsync, etc.
  - grep, sed, awk, tail, etc.
    - e.g., instantiate a new VNF
      - mkdir -p /vnfs/user-a/chain-b/ids
- Rich set of file system management operations offered by configuration management tools such as Chef, Puppet, Salt etc.

System Architecture/Implementation

http://watnfv.github.io/nf.io/



Outline

- Future Application Platforms: trends and challenges
- Convergence of IT and Telecommunication Infrastructure
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

Summary

- Current ICT "Revolution" ?
  - Virtualization and Softwarization as enablers of Future Application Platforms
- SAVI Testbed
  - Canadian Future Application Platform, leveraging Multi-tier Clouds and SDN to provide a fully programmable research testbed
- This presentation
  - Shown how some of the challenges underlying the development of the Smart Edge SDI Manager have been addressed, namely resource management, service availability and green operations
  - VDC Planner, Venice and Greenhead operational in the SAVI Testbed
  - Shown how the smart edge can be leveraged for NFV deployment, orchestration and management

What's Next in SAVI ?

- Smart Edge & SAVI Testbed 2.0:
  - Expand capacity and number of edges
  - Deploy virtualized FPGA resources
  - Spectrum + RF Frontend Virtualization
  - Deploy Wireless Access Manager in Smart Edge
  - Extend to Software Defined Radio and Radio over Fiber access
  - Deploy NFV orchestration module in SDI Manager
  - Extend Monitoring with stream processing and data analysis
  - Cloud-RAN to provide LTE using the Smart Edge
- Kaleidoscope: SAVI Demonstrator App

Kaleidoscope

Other scenario ?

Take Away Message

- Unprecedented challenges facing network operators:
  - Exponential increase in data/video traffic
  - Massive connectivity (mobile devices, things, everything)
  - Proliferation of OTT services
- To meet above challenges and create new revenue streams with innovative services (e.g., content caching, in-network video streaming, ect.):
  - Cost-effectiveness - leveraging economies of scale by constructing infrastructure from a few commodity hardware
  - Elasticity and Agility - ability to rapidly deploy and elastically scale services
  - Efficiency and programmability

➔ Network operators need to reinvent their networks

Reinventing networks @ the Smart Edge

- Why ?
  - Characteristics of smart edge: Converged, On-premises, Proximity, Low-latency, Location-awareness, Network context information, Programmability
- Where ?
  - Central Offices re-architected
- How ?
  - Software-Defined Infrastructure leveraging:
    - Cloud Computing (virtualized platforms, service-oriented architecture, elastic scaling, and scalability)
    - SDN (simpler forwarding devices, programmable control plane)
    - NFV (reduced CAPEX and OPEX, optimized service chaining)

➔ Implications for network operators ?

Outline

- Future Application Platforms: trends and challenges
- Convergence of IT and Telecommunication Infrastructure
- The SAVI Project
- SAVI Smart Edge
- Sample Research Contributions
  - VDC Planner
  - Venice
  - Greenhead
  - NFV Orchestration
- Summary, future work and take away message
- Implications for network operators and challenges ahead

Implications & challenges ahead

- Scalability
  - Early SDN deployments in DCNs revealed excessive flow setup and statistics gathering – controller performance bottleneck
  - Problem exacerbated in WANs: Difficult to maintain acceptable flow setup time and global network view
- Traffic Engineering
  - Leverage SDN abstractions to make traffic engineering decisions and ultimately better utilize network resources
- SDN Support for NFV
  - How SDN can help in steering traffic between dynamically instantiated VNFs, and providing support for NFV chaining?
- Migration from currently deployed hardware to SDN solutions
- Security
- Managing the “S” in “SDN”

Managing the "S" in "SDN"

- SDN control plane is a distributed software system
- Software is prone to bugs
  - Industry Average: "about 15 - 50 errors per 1000 lines of delivered code." [1]
  - SDN control plane is no exception
  - In [2], even for very small scale networks (2~4 switches) and the very basic L2 learning module the authors found the following number of new bugs:
    - Floodlight – 2; NOX – 1; POX - 4
  - Already known issues/bugs
    - NOX - 3 (open), and 24 (closed)
    - POX - 7 (open), and 108 (closed)

[1] Code Complete by Steve McConnell
[2] Scott et al., Troubleshooting SDN Control Software with Minimal Causal Sequences, SIGCOMM 2014.

Debugging the Control Plane

- SDN control plane has to process a sequence of events that involve multiple actors
- In case of an error, only way to debug is to capture and replay traffic trace to execute the same code branch (hopefully in the same sequence)
- It may become very difficult to reproduce an error due to
  - Event timing issues (no clock sync)
  - Controller's internal thread scheduling
  - Non-determinism in the trace (e.g, packet/event ordering in the queue)
  - Parallel processing in the controller
- It is impractical to collect complete network and controller state to reproduce the exact actions for the same traffic

Management Issues

- Managing the policies – We know how to do that, Do we?
- According to SDN proponents, the controller will automate most of the network management tasks
  - How do we enforce some policy manually or that cannot be automated (depends on some external event)?
  - In SDN, human operator's visibility and control is curtailed due to automation
- SDN introduces additional risks, including failure of the controller itself and security vulnerabilities (e.g., heartbleed)
- The possibility of multiple controllers issuing contradictory instructions to switches
- Management functions need to adapt to a flow based model (e.g., monitoring, configuration)

Management Issues (cont)

- Network functions will be translated to controller applications
  - What kind of impact this will have on the existing software infrastructure?
  - How to update one application without disrupting others
    - Controllers + applications are compiled as a single package
    - Currently it is not possible to update/modify the code of one application without stopping the controller (the entire network)
- SDN is supposed to remove vendor lock-in
  - Are we jumping from fire (hardware vendors) to frying pan (software vendors)?
  - Historically speaking, who provides robust products: Cisco vs. Microsoft?
- Transition from communications engineers to computer scientists, from network administrators to app developers


Questions