Distributed cloud optimization: It's all about the model

Danny Raz
Technion

Part of the work done in Nokia Bell Labs with many coauthors

The Network

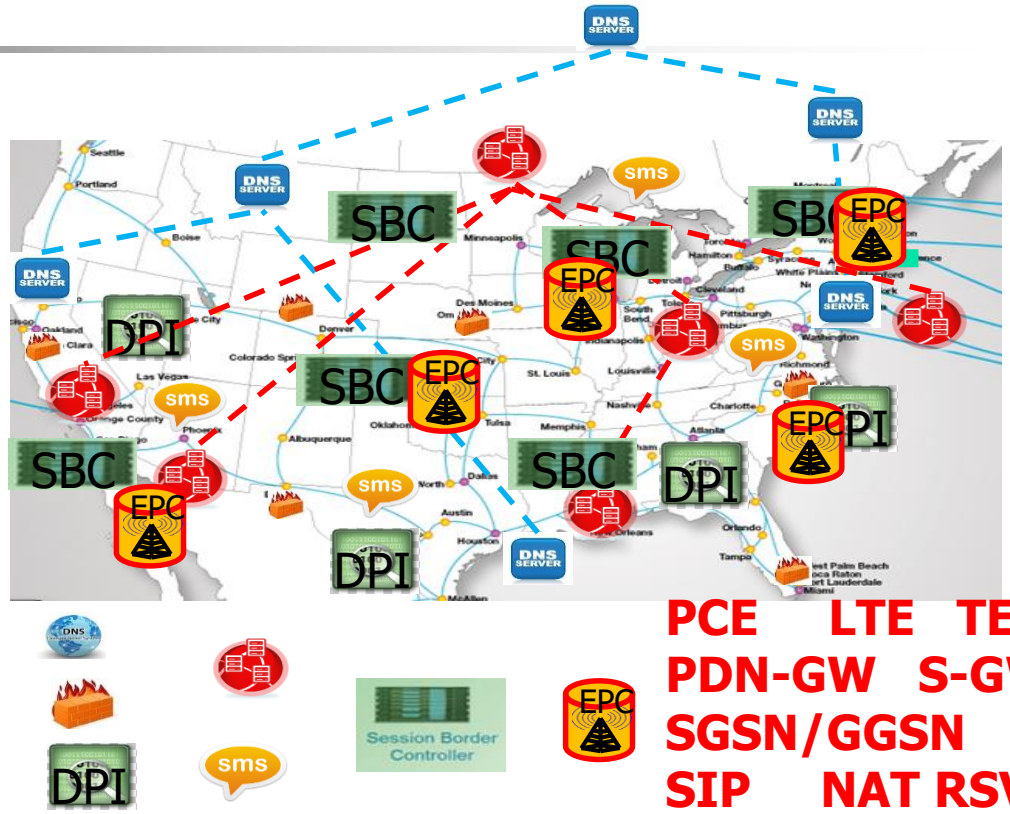
- Basically
  - Transport information from place to place
  - Transport bits from place to place
  - Transport packets from place to place

The Network

- Basically
  - Transport information from place to place
  - Transport bits from place to place
  - Transport packets from place to place
- Actually
  - People can talk (video-conf)
  - People can text (or Whatsapp)
  - Communities can be formed
  - Machines can share state
  - Applications can …. (real time traffic, public transportation, ….)

The Network

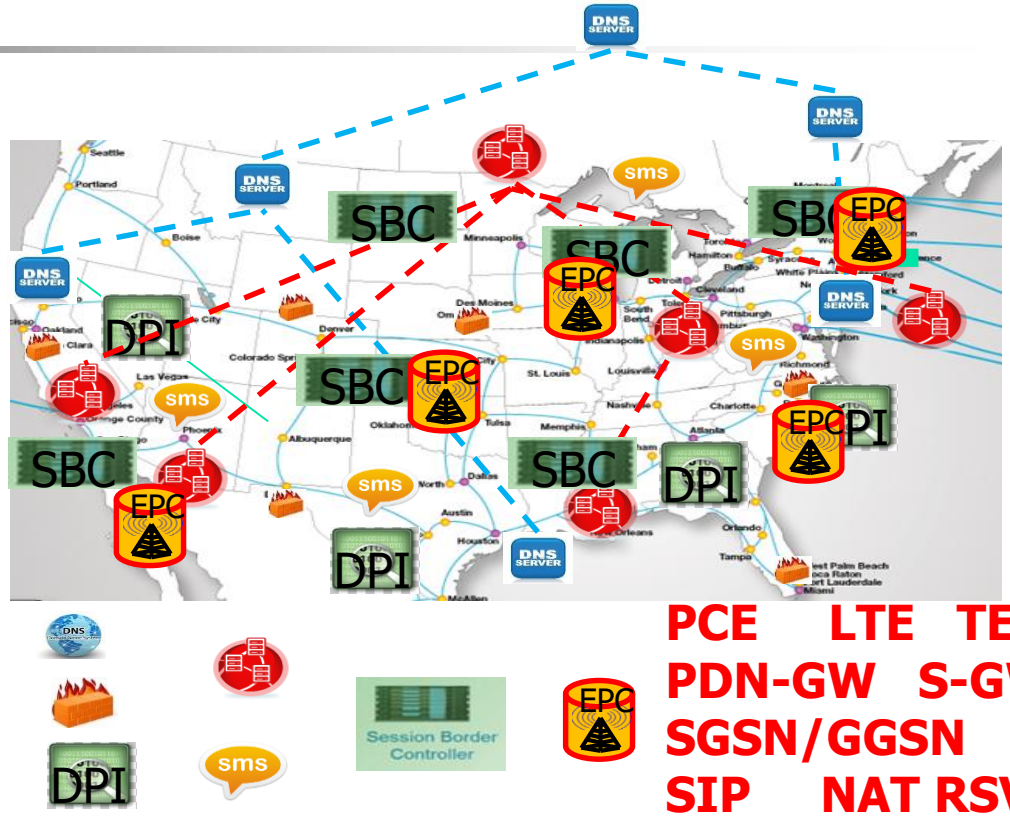
- Much more than just
  - Transport packets from place to place
- Actually
  - People can talk (video-conf)
  - People can text (or whatsup)
  - Communities can be formed
  - Machines can share state
  - Applications can …. (real time traffic, public transportation, ….)



PCE LTE TE
PDN-GW S-GW
SGSN/GGSN
SIP NAT RSVP

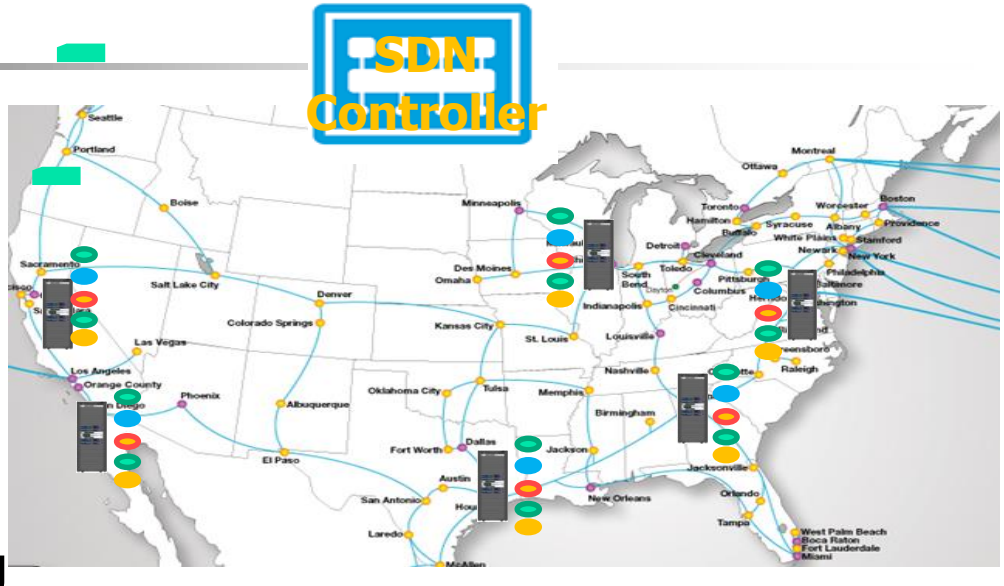
The Network is a Service

- A Network Service
  - Composed of one or more network functions
  - Service function chaining
- Currently
  - Functions (and services) are implemented via dedicated hardware located on the flow path



The Network is a Service

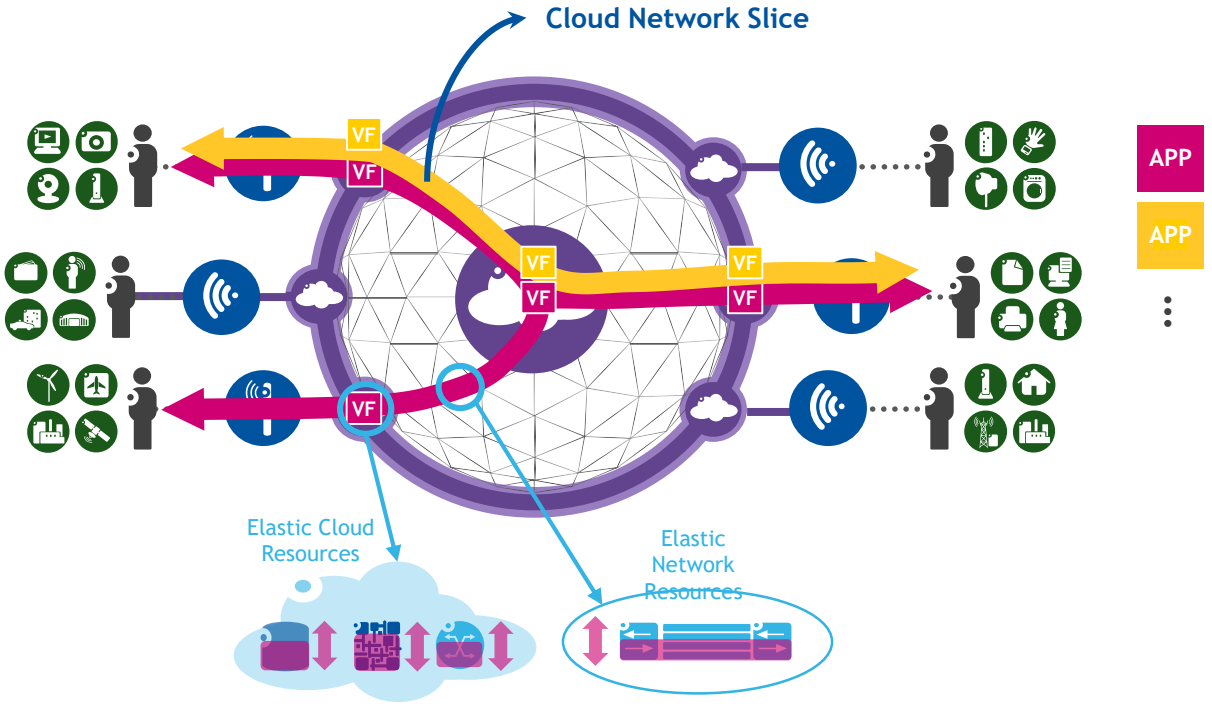
- A Network Service
  - Composed of one or more network functions
  - Service function chaining
- Distributed Cloud Networking
  - Functions (and services) are implemented on COTS servers located at mini) data centers distributed within the network
  - Traffic is send to these servers using the control mechanism of SDN



PCE LTE TE
PDN-GW S-GW
SGSN/GGSN
SIP NAT RSVP

Distributed cloud networking = NFV + SDN

- Key enablers
  - Network function virtualization (NFV)
  - Software defined networking (SDN)

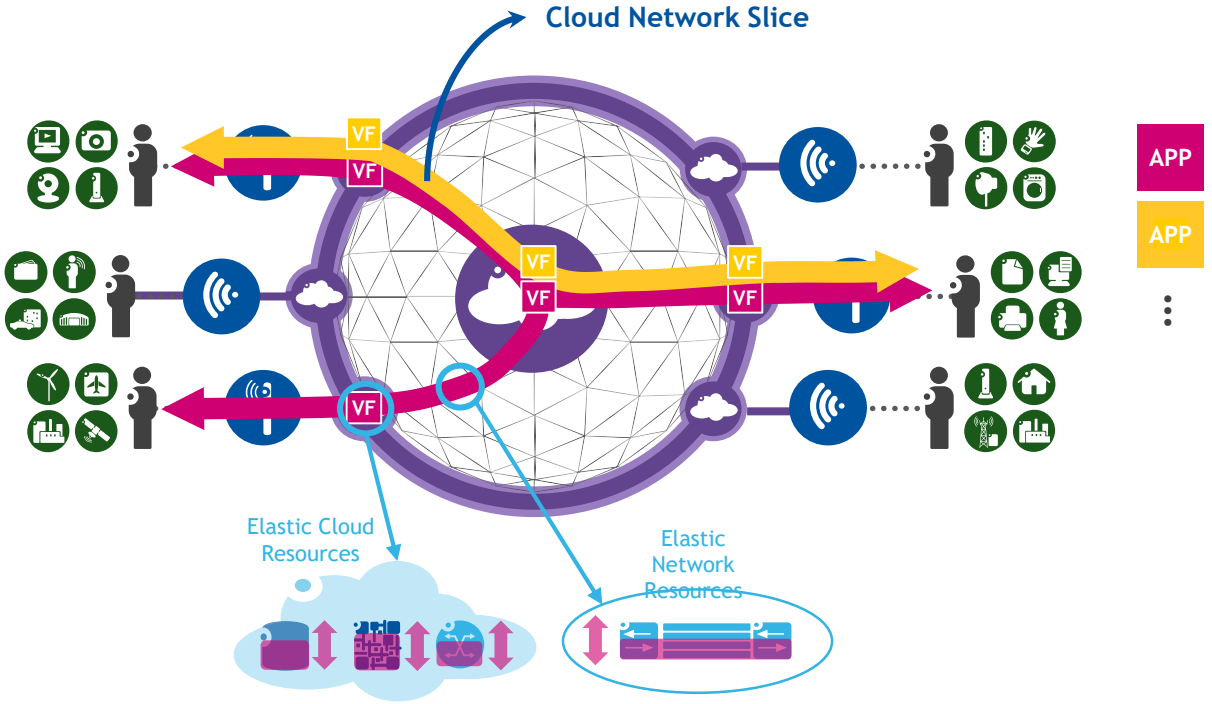


Bell Labs, "The Future X Network," CRC PRESS, October 2015.

Distributed cloud networking = NFV + SDN

- Key enablers
  - Network function virtualization (NFV)
  - Software defined networking (SDN)

- Ideal for next generation services
  1) Network services
     - NFV
  2) Automation services
     - Smart X, IoT
  3) Augmented experience
     - Virtual X, Augmented X



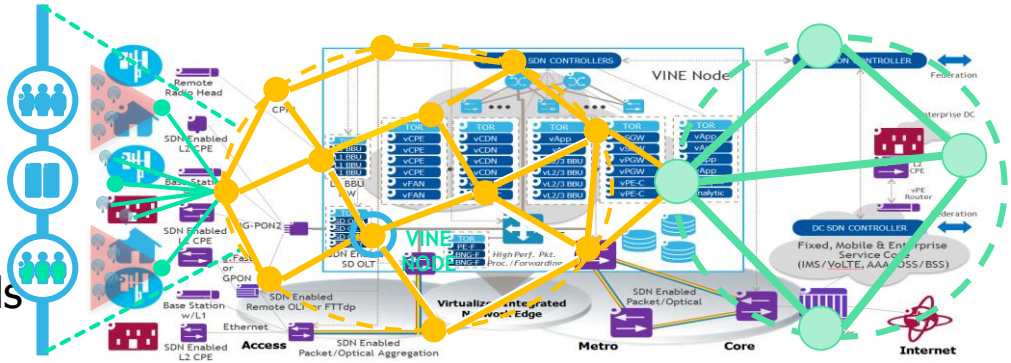
Bell Labs, "The Future X Network," CRC PRESS, October 2015.

NVF + SDN

- Lots to gain
  - Use COTS silicon - Reduced Capex
  - Easy provisioning - reducing time to market
  - Easier operation – reduced Opex
- Not so simple
  - Can we get the performance we (want) need
  - Can we get the reliability we (want) need
  - Isn't this too complex (to operate)
  - Can we achieve agility despite of:
    - Vendors and operators
    - multi vendors environment
  - Full, end to end, carrier-grade telco NFV at a reasonable cost

Placement of Network Functions

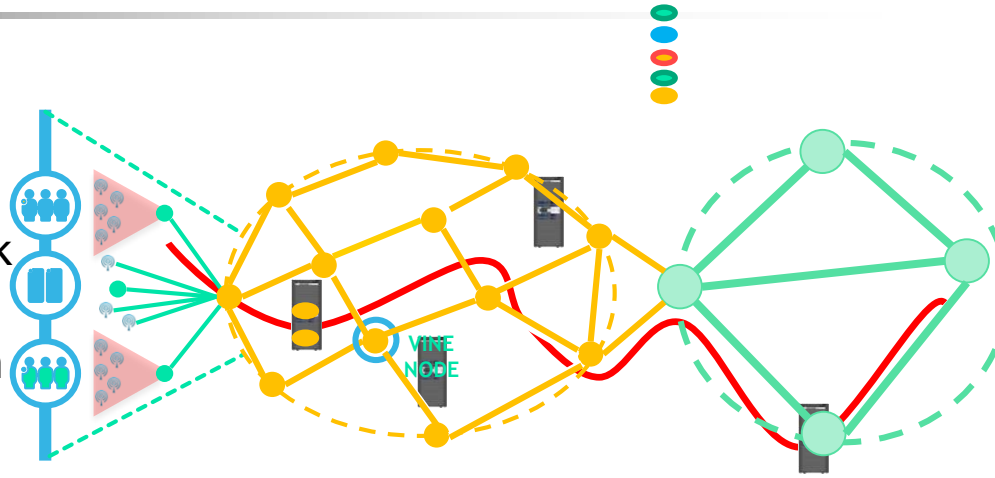
- Where to place each function
  - One place (globally)
  - In each location
  - Statically – network planning
  - Dynamically (as needed) depends on demand
- What exactly is
  - The demand
  - The cost (of placing network functions)
  - The constraints (what can be put where)
  - A good placement (objective function)



A network optimization problem

Placement of Network Functions - A Model

- Input
  - A set of flows, each with a path and a demand for each of the possible network functions.
  - A set of datacenters locations, each with a size.
  - A set of network functions realizations , each with capacity (amount of clients to be served), size, and establishment cost .
- Output
  - A placement of copies of the realization of the network functions and a rerouting of the flow into the DCs



Such that: The demand for each flow and for each function is satisfied, the size constraints are met, and the overall cost is minimal

- Input
  - A set of flows, each with a path and a demand for each of the possible network functions.
  - A set of datacenters locations with a size.
  - A set of network functions realizations , each with capacity (amount of clients to be served), size, and establishment cost .
- OUTPUT
  - A placement of realization of the network functions and a rerouting of the flow into the DCs

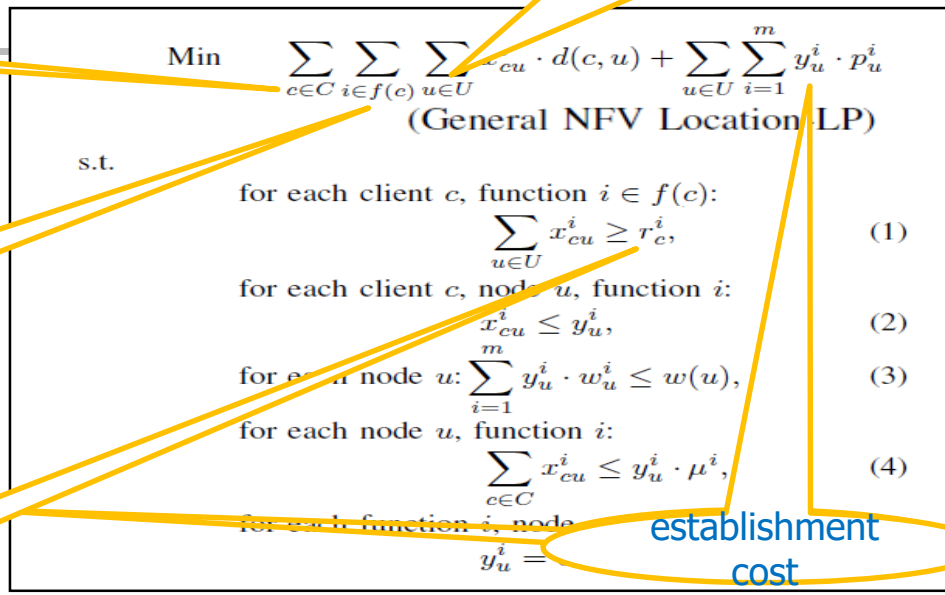
flows=clients

node = DC location

i is a network function

flow's demand for function *i*

establishment cost



$$\text{Min} \quad \sum_{c \in C} \sum_{i \in f(c)} \sum_{u \in U} x^i_{cu} \cdot d(c,u) + \sum_{u \in U} \sum_{i=1}^{m} y^i_u \cdot p^i_u$$

(General NFV Location LP)

s.t.

for each client c, function $i \in f(c)$:

$$\sum_{u \in U} x^i_{cu} \geq r^i_c, \qquad (1)$$

for each client c, node u, function i:

$$x^i_{cu} \leq y^i_u, \qquad (2)$$

for each node u:

$$\sum_{i=1}^{m} y^i_u \cdot w^i_u \leq w(u), \qquad (3)$$

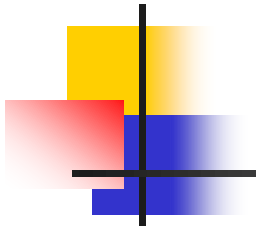
for each node u, function i:

$$\sum_{c \in C} x^i_{cu} \leq y^i_u \cdot \mu^i, \qquad (4)$$

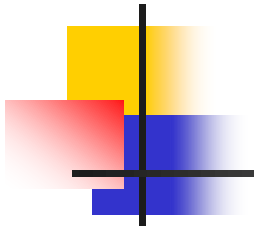
for each function i, node

$$y^i_u =$$

Such that: The demand for each flow and for each function is satisfied, the size constraints are met, and the overall cost is minimal



This talk is about the MODEL



This talk is about the

MODEL

where the goal is to

optimize REAL SYSTEMS

Why modeling?

- **Real systems are very complex**
  - Different parameters that affect the result
  - Many configuration options
  - In the Network Function placement case:
    - depends on the actual VNF (vCPE, vCDN, …)
    - on the underlying infrastructure (VM, container, …)
    - many more ….

- **Need to capture the important (and only the important) aspects**
  - What is important?
  - How to quantify the affect of these (important) parameters
  - What are the criteria for success (optimization objective)

Addressing an optimization problem

- **Model the problem**
  - must select the "right" perspective
  - this is the most difficult part
- **Find an optimization scheme for the "theoretical" problem**
  - not always so easy
  - most problems are NP-hard
  - approximation or heuristics
- **Apply the solution to the original (real) problem**
  - need to modify the "theoretical" approximation algorithm
- **Evaluate expected performance**
  - in many cases difficult for lack of data (NFV)

Main Theoretical Result

- notes
  - If there is only one network function then this problem is actually the well known facility location problem .
  - If there are no network distances this problem reduces to the well known generalized assignment problem (GAP).

$$\text{Min} \quad \sum_{c\in C}\sum_{i\in f(c)}\sum_{u\in U} x^i_{cu}\cdot d(c,u) + \sum_{u\in U}\sum_{i=1}^{m} y^i_u \cdot p^i_u$$

(General NFV Location-LP)

s.t.

for each client c, function $i \in f(c)$:

$$\sum_{u\in U} x^i_{cu} \geq r^i_c, \qquad (1)$$

for each client c, node u, function i:

$$x^i_{cu} \leq y^i_u, \qquad (2)$$

for each node u: $\sum_{i=1}^{m} y^i_u \cdot w^i_u \leq w(u), \qquad (3)$

for each node u, function i:

$$\sum_{c\in C} x^i_{cu} \leq y^i_u \cdot \mu^i, \qquad (4)$$

for each function i, node u:

$$y^i_u = 0 \quad \text{if } w^i_u > w(u). \qquad (5)$$

Theorem:

There exists a bi-criteria (O(1), O(1)) approximation algorithm for the General NFV location problem

Lewin-Eytan et. al., "Near Optimal Placement of Virtual Network Functions," IEEE INFOCOM, 2015.

Experimental evaluation

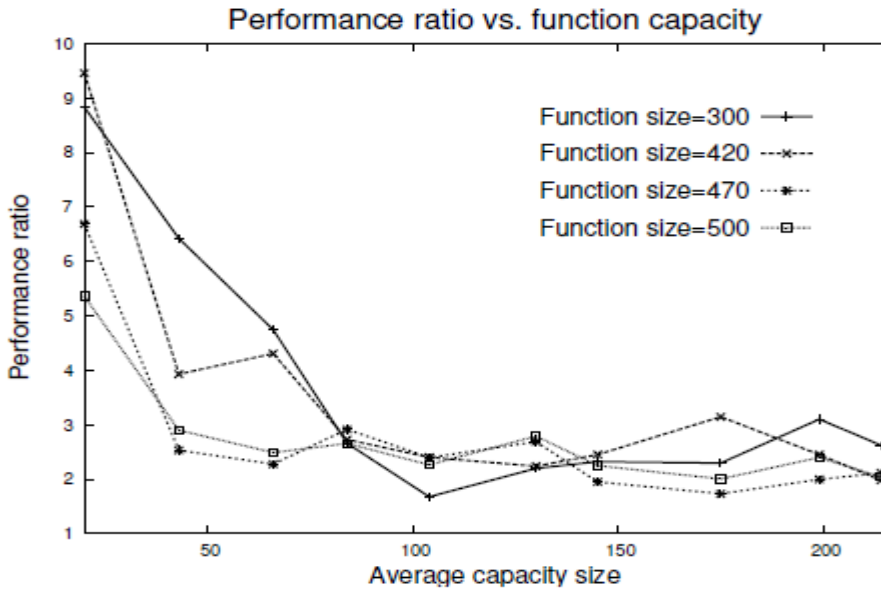
This network covers:

- 195 access locations (mostly within Europe and North America), about 260 links and almost 40 data centers

- Input
  - A set of flows, each with a path and a demand for each of the possible network functions.
  - A set of datacenters locations, each with a size.
  - A set of network functions realizations , each with capacity (amount of clients to be served), size, and establishment cost .

- selected 400 random pairs of (source, destination), and determined a shortest path between each source and destination, unit demand per flow.
- Each such flow is associated with 1-4 network functions that were chosen randomly from a set of 30.
- The size of a network function varies.
- The size of data center was randomly selected in the range 200-500.
- Opening cost was constant.

Experimental evaluation

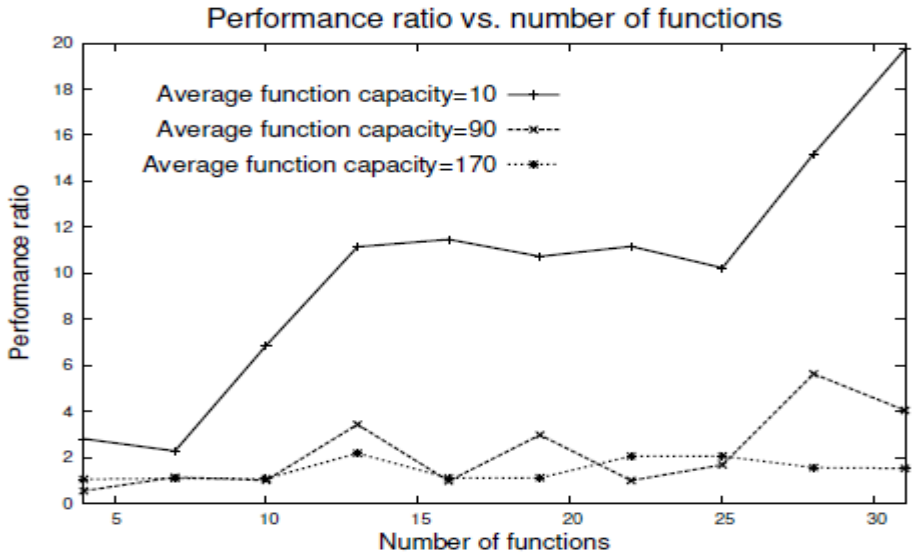


- Greedy
  - Go over all network function in an arbitrary order
  - For each such function
  - Find in a greedy way the best placement to satisfy the flows' demand

randomly from a set of 30.

- The size of a network function varies.
- The size of data center was randomly selected in the range 200-500.
- Opening cost was constant.

Experimental evaluation



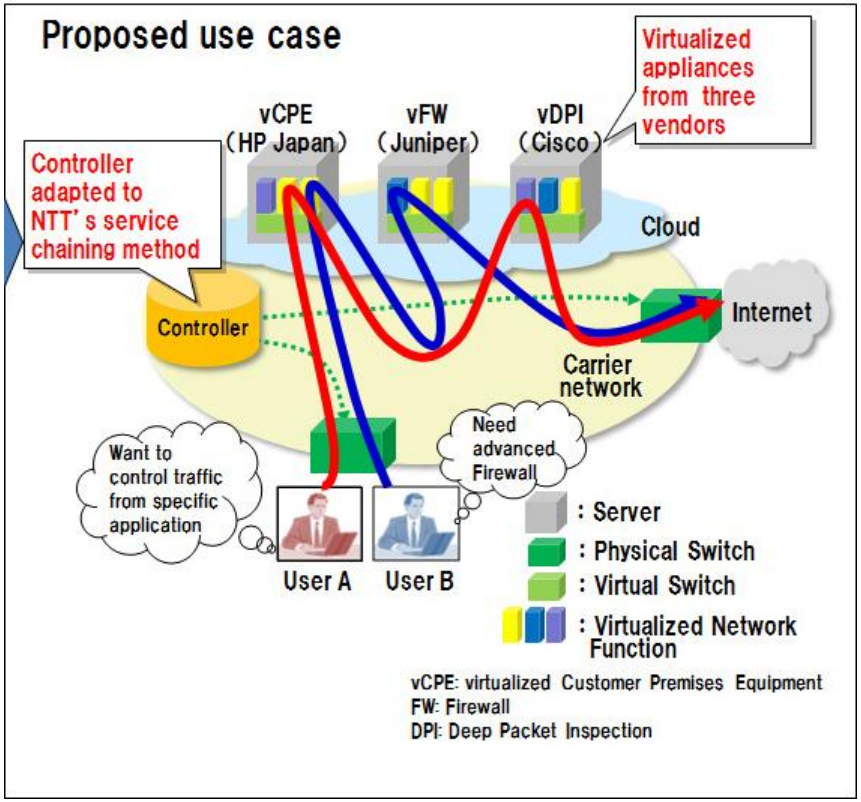
- Greedy
  - Go over all network function in an arbitrary order
  - For each such function
  - Find in a greedy way the best placement to satisfy the flows' demand

randomly from a set of 30.

- The size of a network function varies.
- The size of data center was randomly selected in the range 200-500.
- Opening cost was constant.

How good is this model?

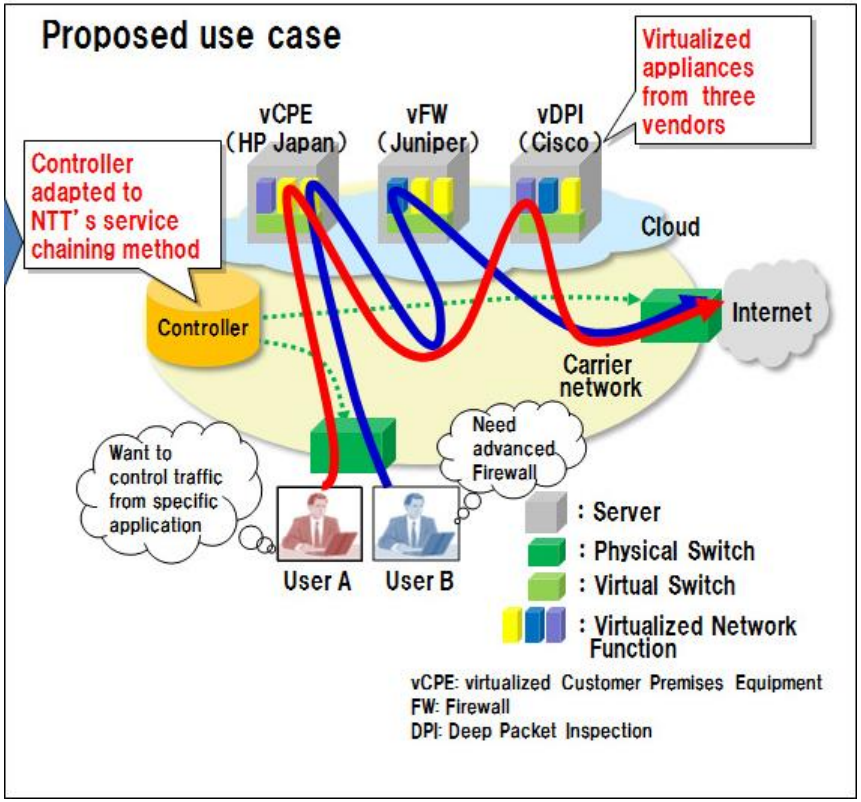
- **Service chaining example**
  - CPE – FW – DPI
- **Can we use the previous model for function placement in this case?**
- **Can we find a better model?**



Source: ETSI Ongoing PoC

How good is this model?

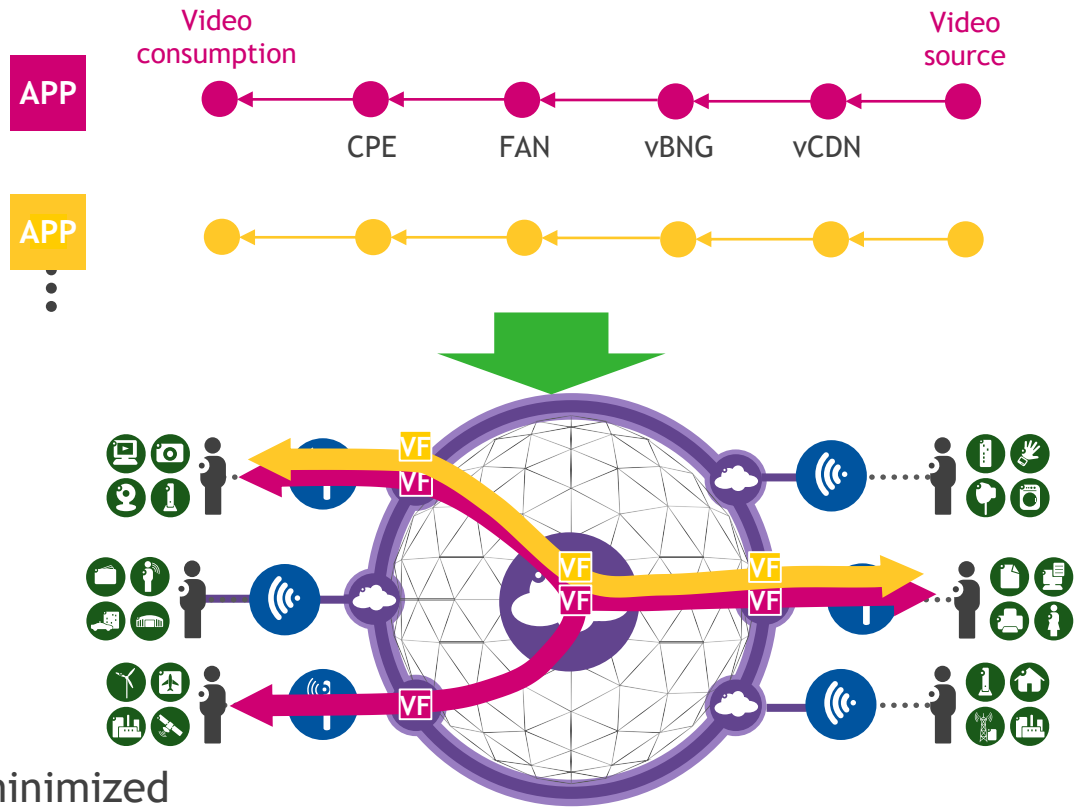
- **The order of the functions (per flow) is given**
- **No pre-defined paths**



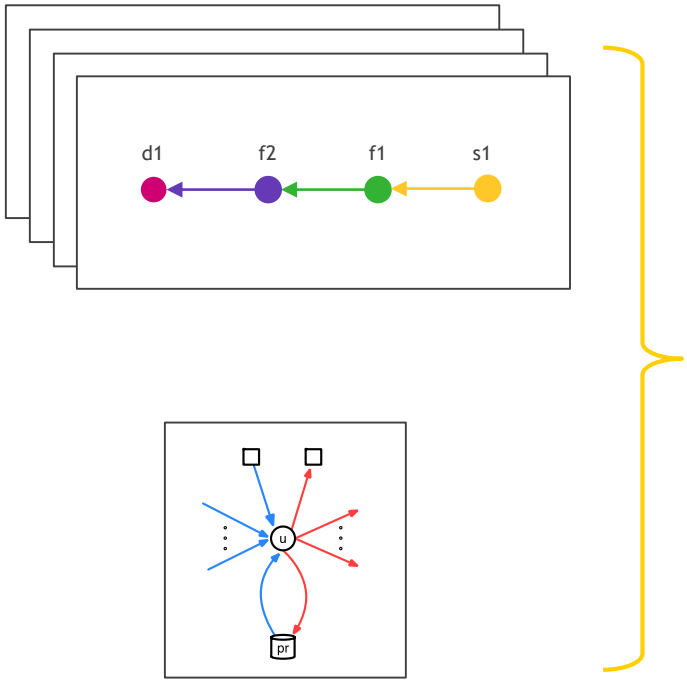
Source: ETSI Ongoing PoC

Service chain model – take 2

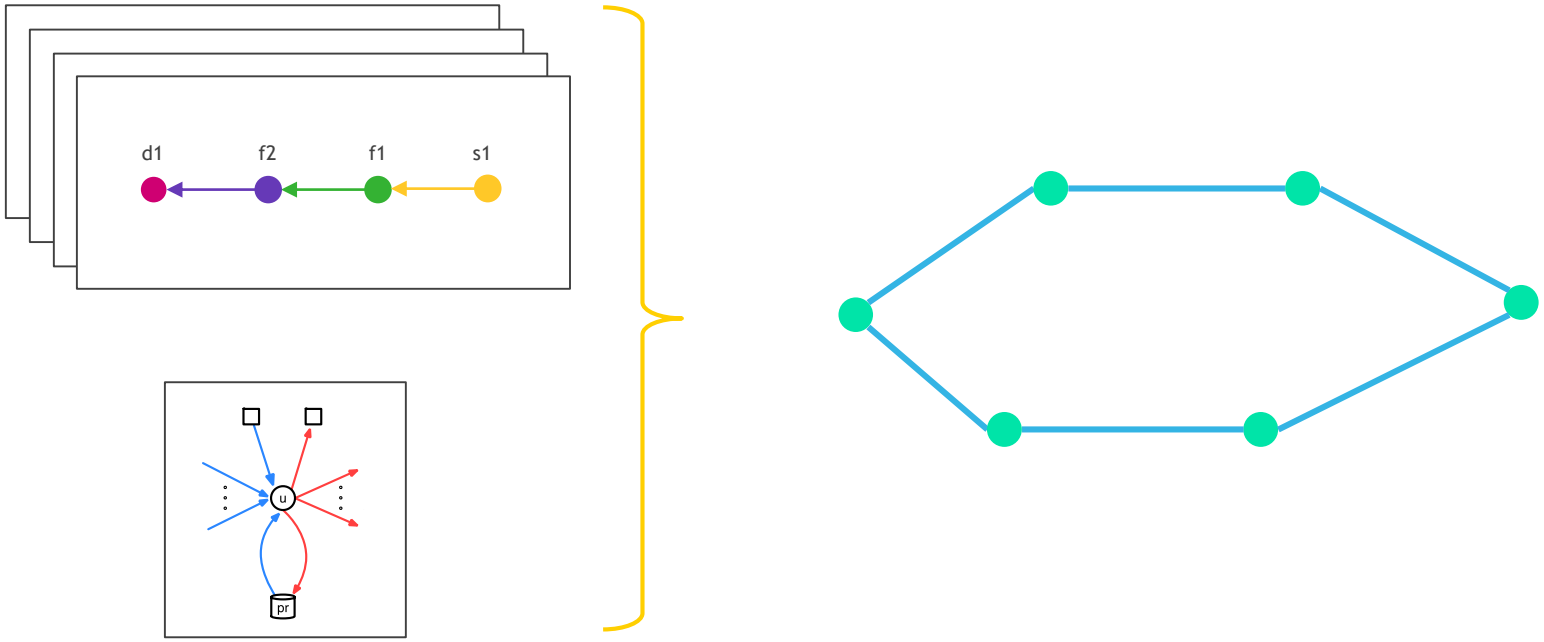
- Given
  - Set of services
  - Set of demands
- Find
  - Function placement
  - Flow routing
  - Cloud resource allocation
  - Network resource allocation
- Such that
  - Demands are satisfied
  - Overall operational cost is minimized



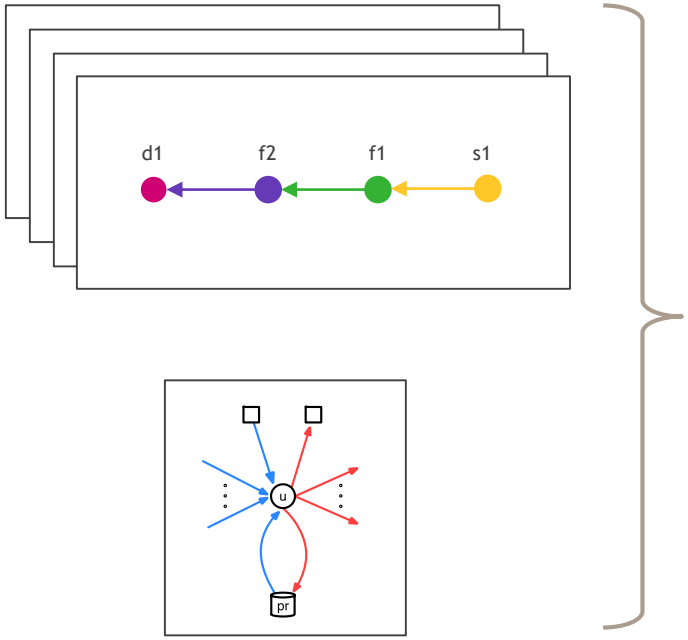
cloud network flow

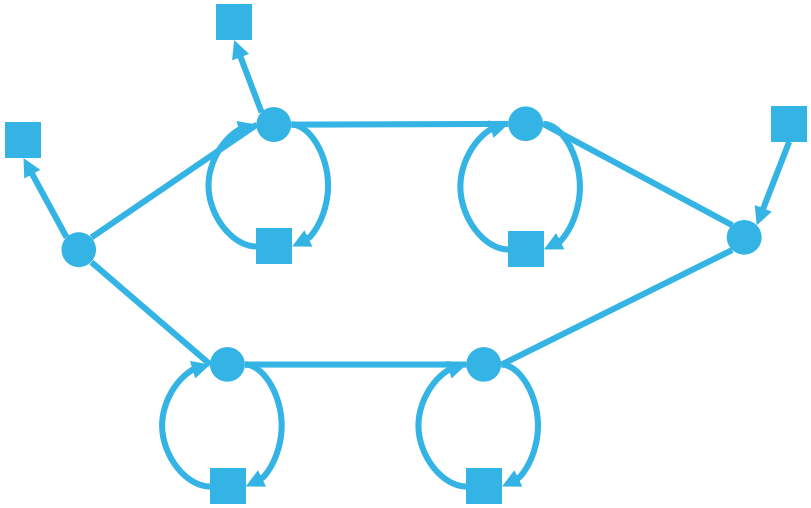


cloud network flow

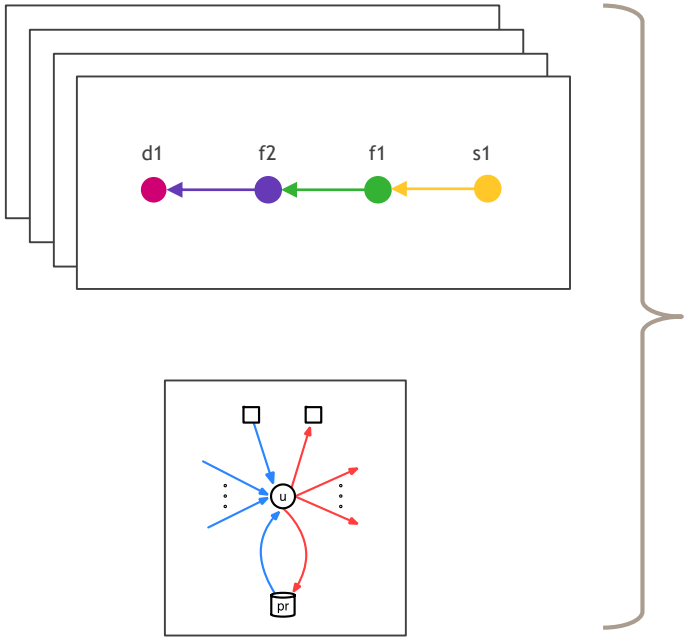


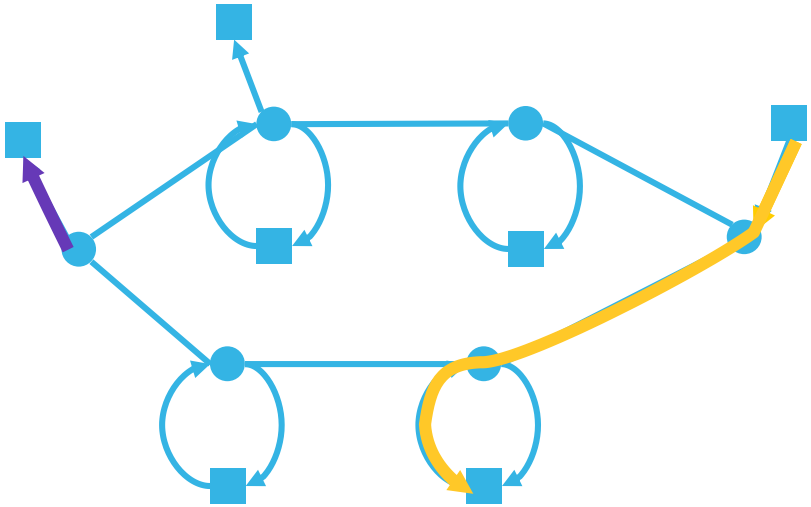
cloud network flow



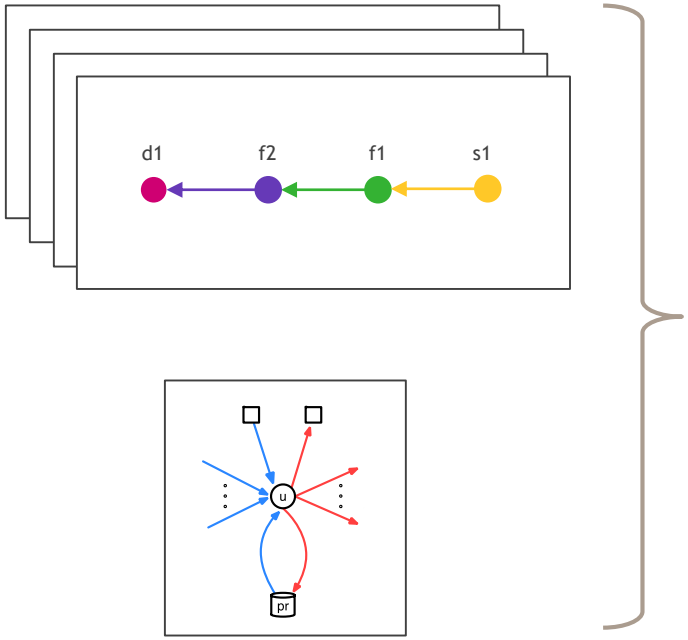


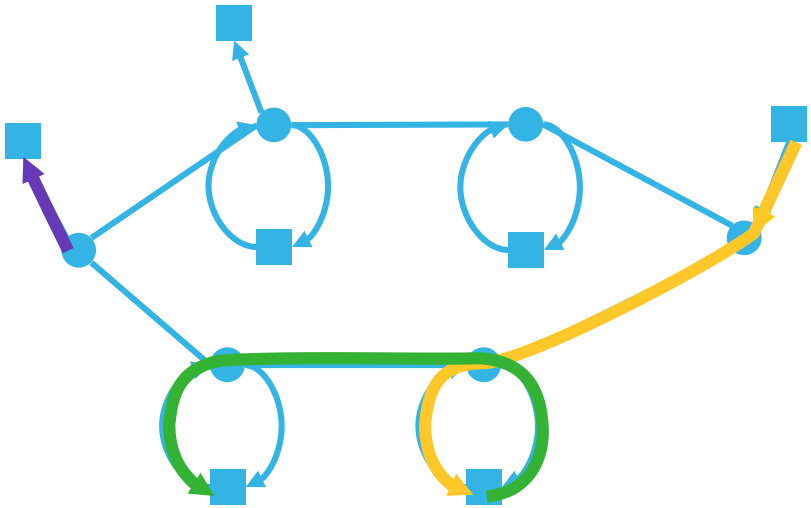
cloud network flow



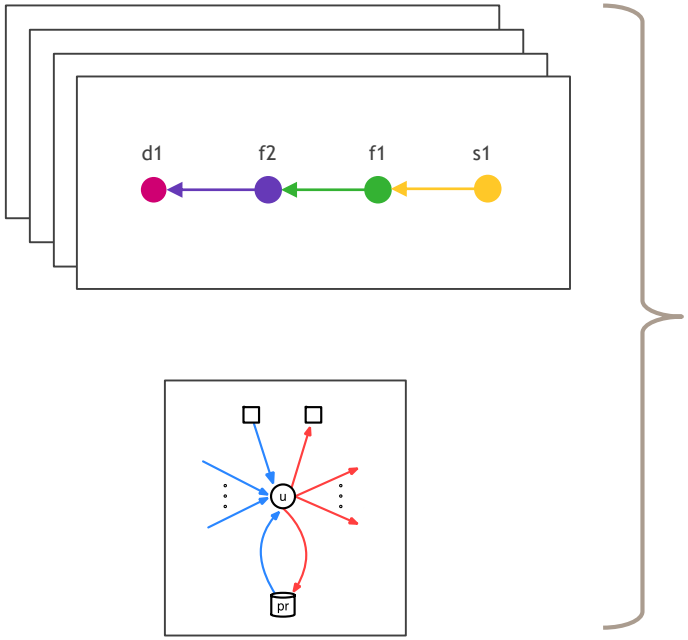


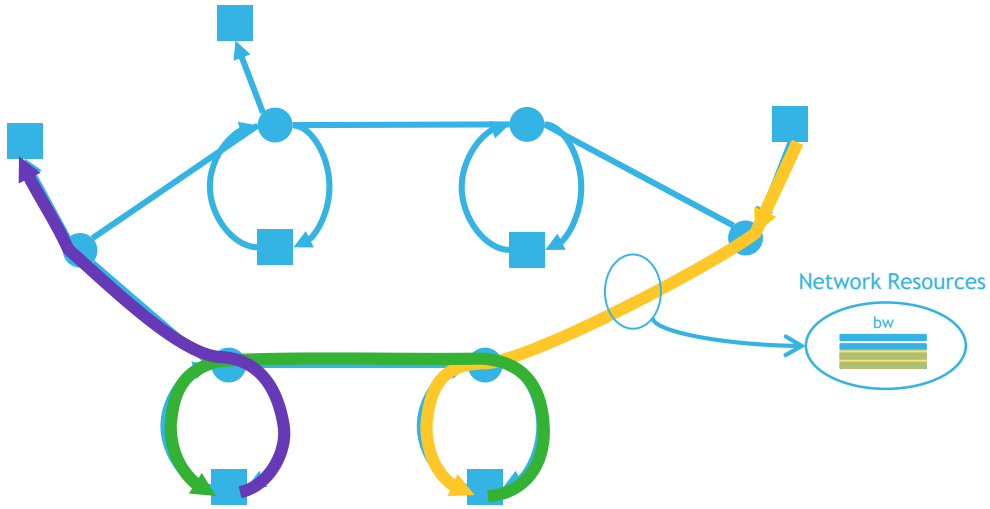
cloud network flow



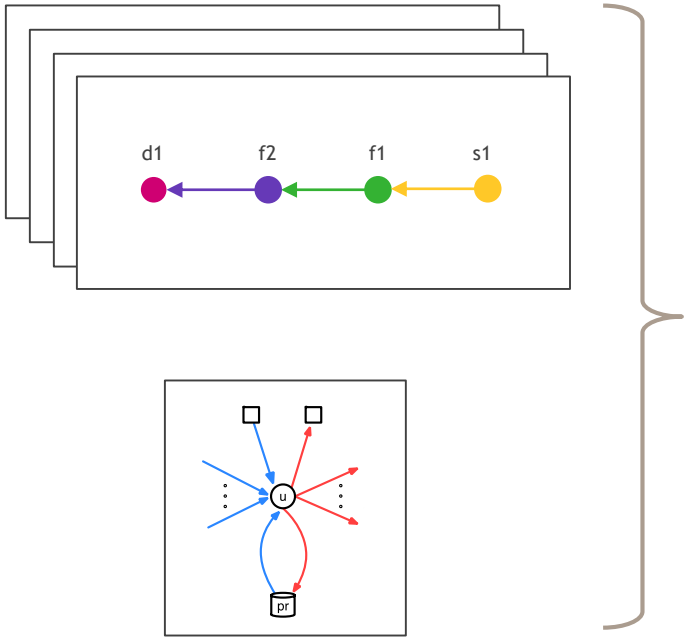


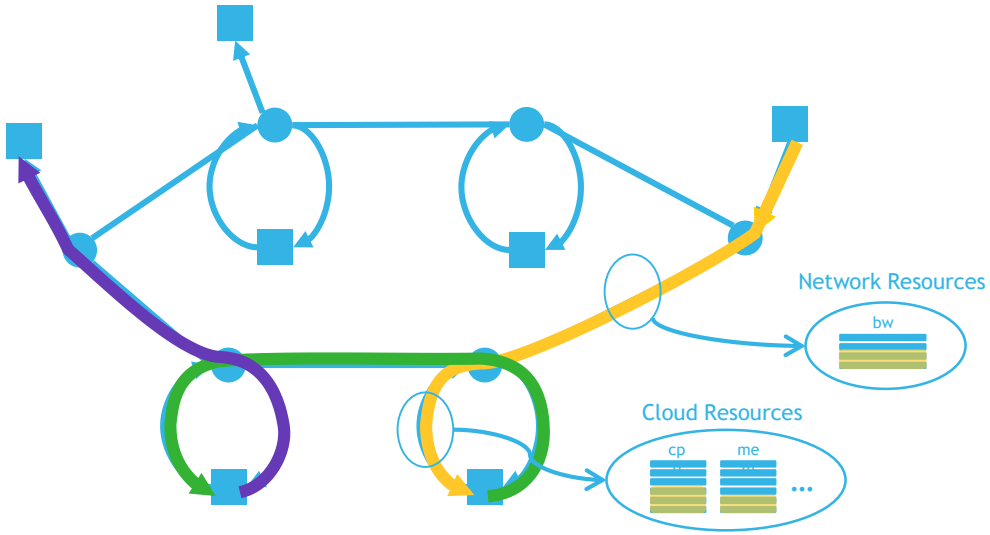
cloud network flow



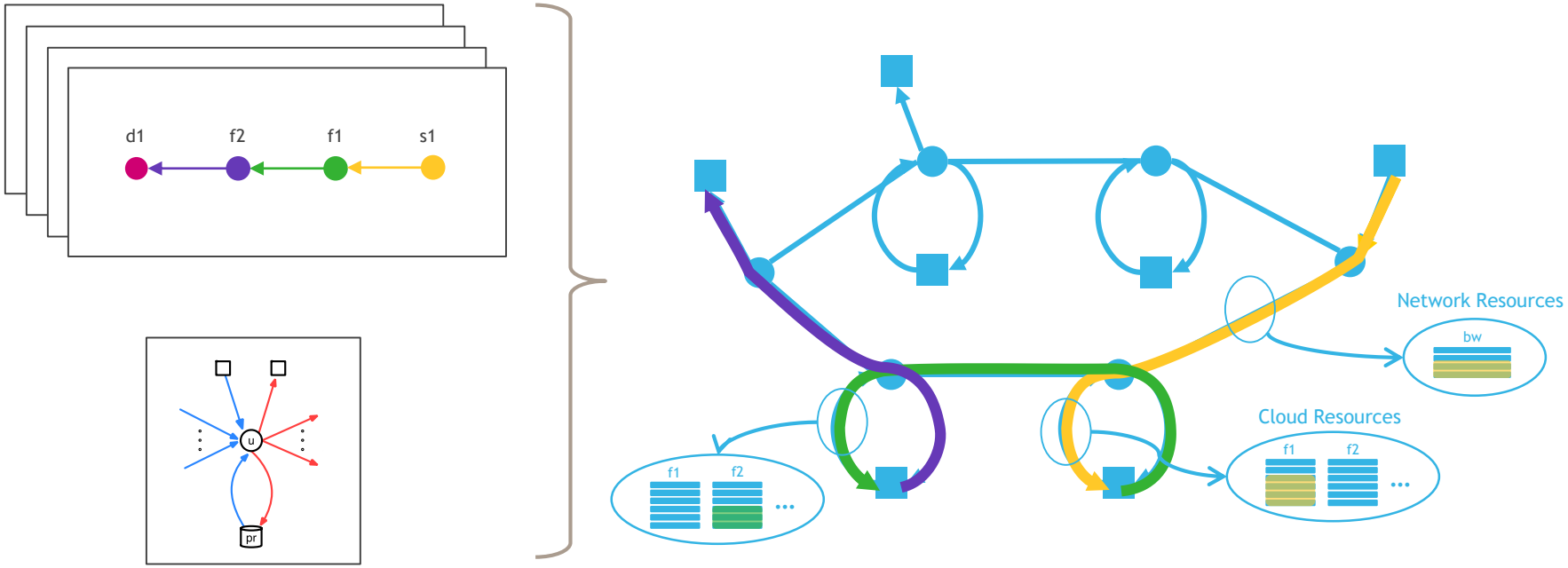


cloud network flow

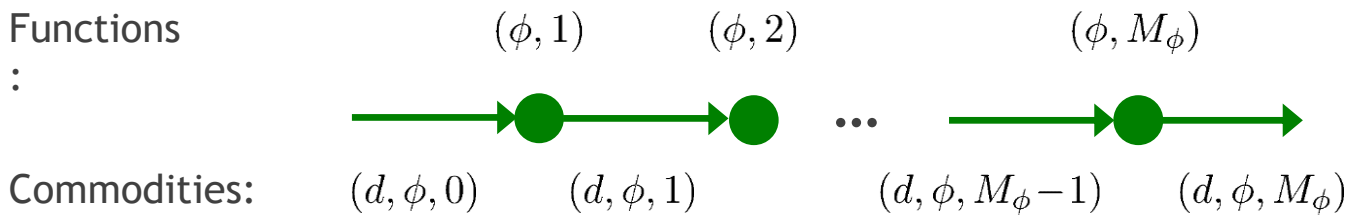




cloud network flow



Service Model



- A network service $\phi \in \Phi$ is described by a chain of M_ϕ virtual network functions (VNFs)
- (ϕ, i) denotes the i-th function of service ϕ
- (d, ϕ, i) denotes the output of the i-th function of service ϕ for destination d
- Function (ϕ, i) has resource requirement $r^{(\phi,i)}$ processing resource units per flow unit, scaling factor $\xi^{(\phi,i)}$ output flow units per input flow unit

Service chain model – take 2

$$\min \quad \sum_{(u,v)} w_{uv}\, y_{uv}$$

Cost Function

$$\text{s.t.} \quad \sum_{v \in \delta^-(u)} f_{vu}^{(d,\phi,i)} = \sum_{v \in \delta^+(u)} f_{uv}^{(d,\phi,i)} \qquad \forall u, d, \phi, i$$

Combined Flow Conservation

$$f_{p(u),u}^{(d,\phi,i)} = \xi^{(\phi,i)} f_{u,p(u)}^{(d,\phi,i-1)} \qquad \forall u, d, \phi, i$$

Service Chaining

$$\sum_{(d,\phi,i)} f_{uv}^{(d,\phi,i)} r_{uv}^{(\phi,i+1)} \le y_{uv} \le c_{uv} \qquad \forall (u,v)$$

Capacity

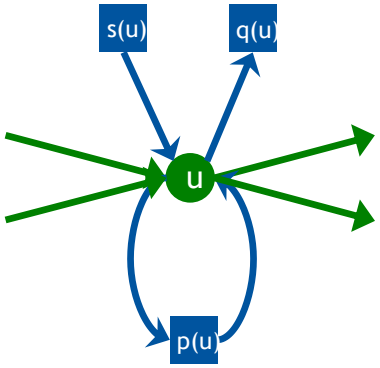
$$f_{s(u),u}^{(d,\phi,0)} = \lambda_u^{(d,\phi)} \qquad \forall u, d, \phi$$

$$f_{u,q(u)}^{(d,\phi,M_\phi)} = 0 \qquad \forall d, \phi, u \neq d$$

Sources and Demands

$$f_{uv}^{(d,\phi,i)} \ge 0,\ y_{uv} \in \mathbb{Z}^+ \qquad \forall (u,v), d, \phi, i$$

Fractional flows Integer resources



Service chain model – take 2

$$\min \quad \sum_{(u,v)} w_{uv}\, y_{uv}$$

Cost Function

s.t.

$$\sum_{v \in \delta^-(u)} f_{vu}^{(d,\phi,i)} = \sum_{v \in \delta^+(u)} f_{uv}^{(d,\phi,i)} \qquad \forall u, d, \phi, i$$

Combined Flow Conservation

$$f_{p(u),u}^{(d,\phi,i)} = \xi^{(\phi,i)} f_{u,p(u)}^{(d,\phi,i-1)} \qquad \forall u, d, \phi, i$$

Service Chaining

$$\sum_{(d,\phi,i)} f_{uv}^{(d,\phi,i)} r_{uv}^{(\phi,i+1)} \le y_{uv} \le c_{uv} \qquad \forall (u,v)$$

Capacity

$$f_{s(u),u}^{(d,\phi,0)} = \lambda_u^{(d,\phi)} \qquad \forall u, d, \phi$$

$$f_{u,q(u)}^{(d,\phi,M_\phi)} = 0 \qquad \forall d, \phi, u \neq d$$

Sources and Demands

$$f_{uv}^{(d,\phi,i)} \ge 0,\ y_{uv} \in \mathbb{Z}^+ \qquad \forall (u,v), d, \phi, i$$

Fractional flows Integer resources

Smaller Resource Granularity

Dedicated boxes

Commodity servers

Virtual machines

Containers

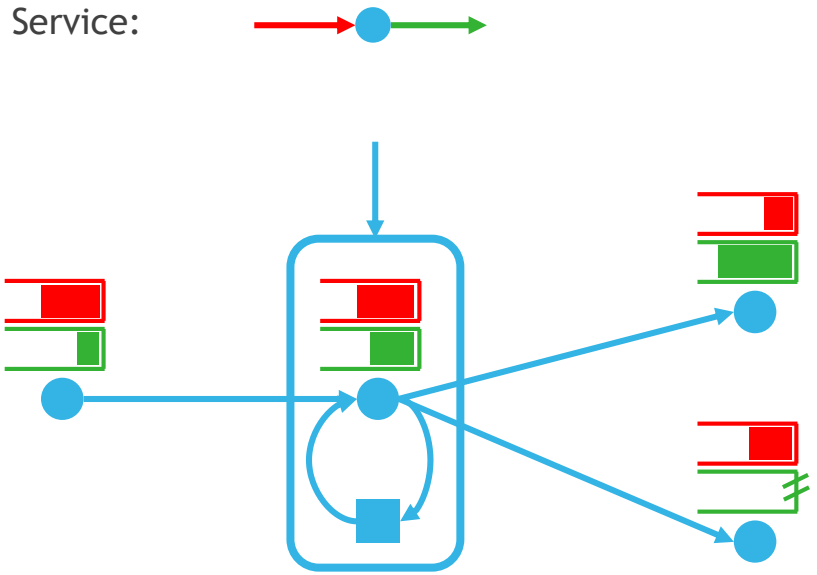
Grains

Main Theoretical Result

There is a fast approximation algorithm for the fractional NSDP

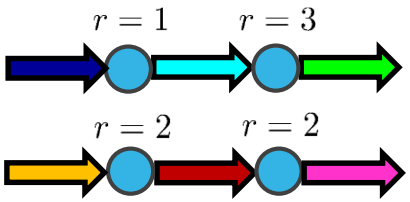
that produces an ε approximation solution in time $O(m^2nL/\varepsilon)$

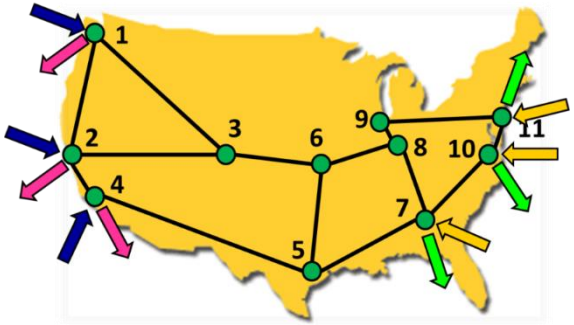
Use dynamic evolution of underlying queuing system to construct an iterative approximation to original static problem

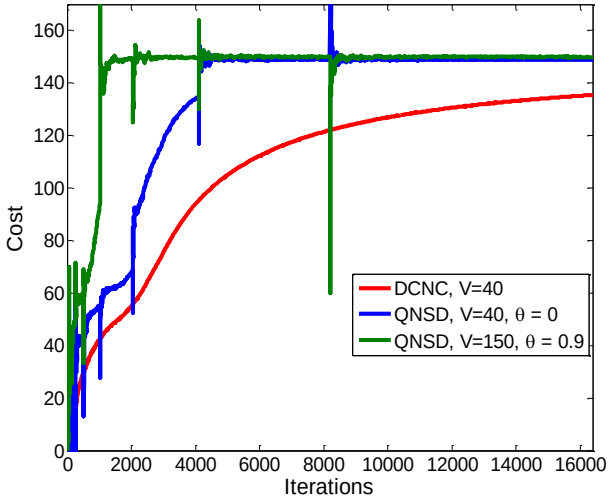


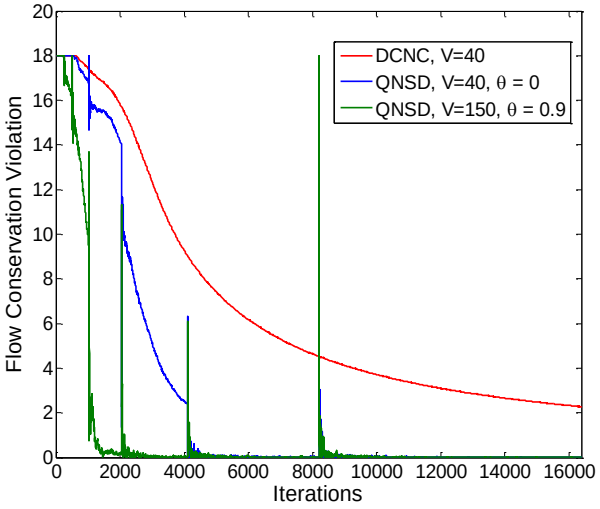
Feng, Llorca, Tulino, Raz and Molischl, "Approximation Algorithms for the NFV Service Distribution Problem"," IEEE INFOCOM 2017.

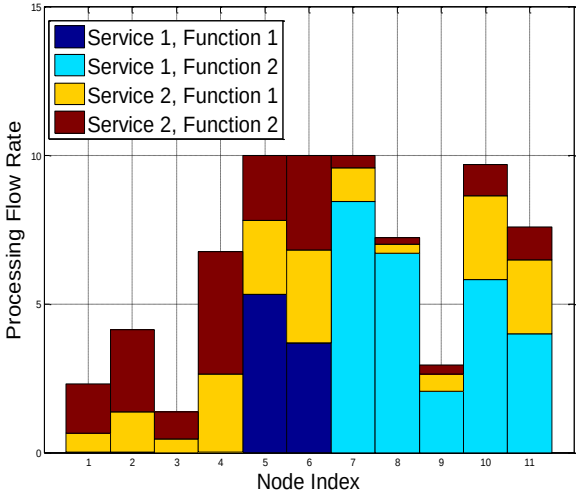
Performance





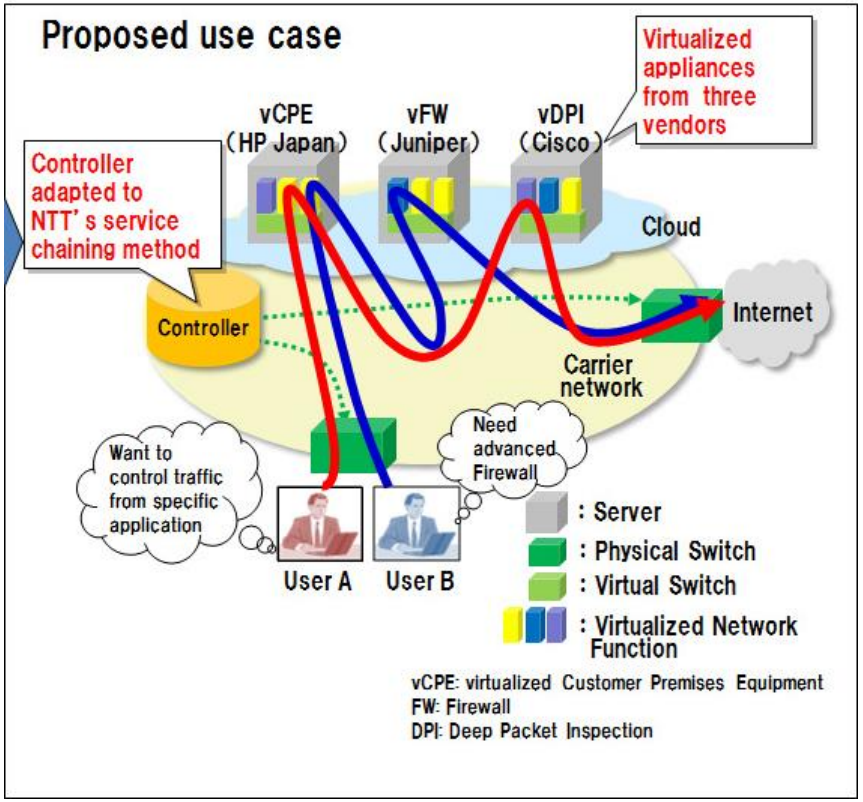






How good is this model?

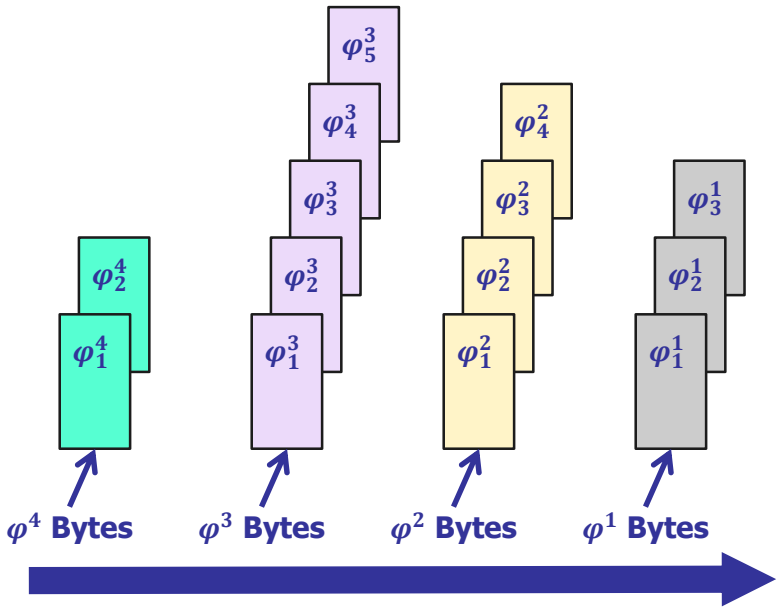
- **Previous models address placement in node (DC) granularity**
- **How about physical host granularity?**
- **Placement of VNF VMs in the physical hosts**



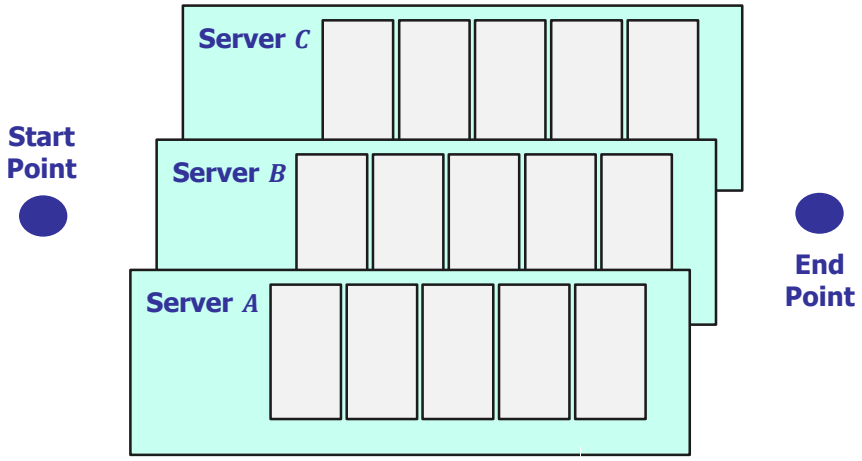
Source: ETSI Ongoing PoC

NFV Chaining Placement (scheduling)

Consider the following sequence of service chaining (a), each with a specified amount of traffic to be processed, and a set of physical servers (b):



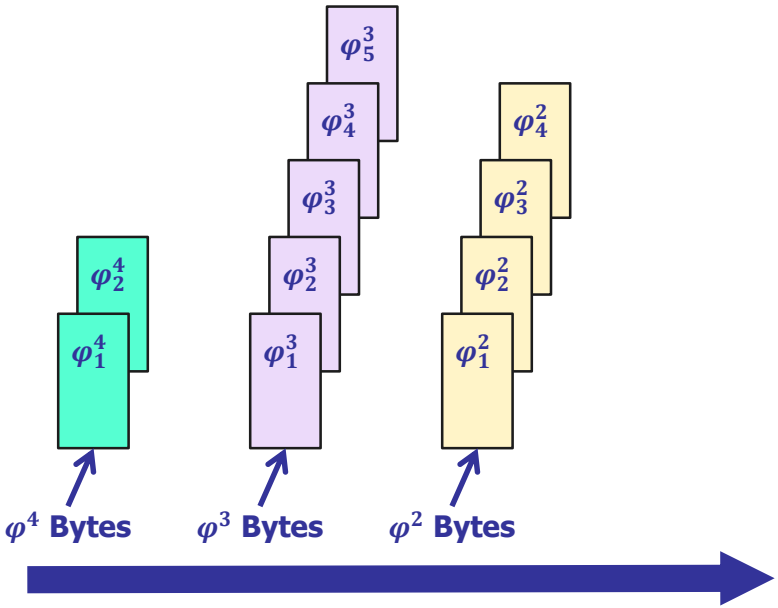
(a) Sequence of service chaining



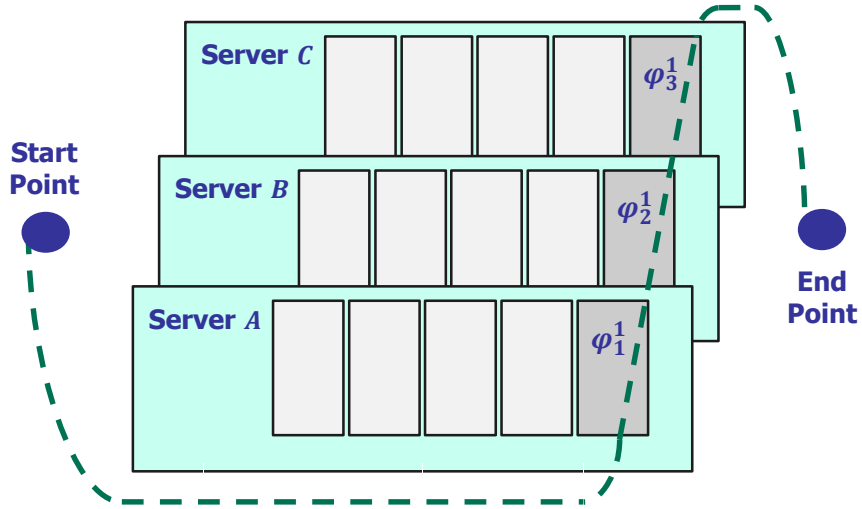
(b) Set of servers

NFV Chaining Placement (scheduling)

Consider the following sequence of service chaining (a), each with a specified amount of traffic to be processed, and a set of physical servers (b):



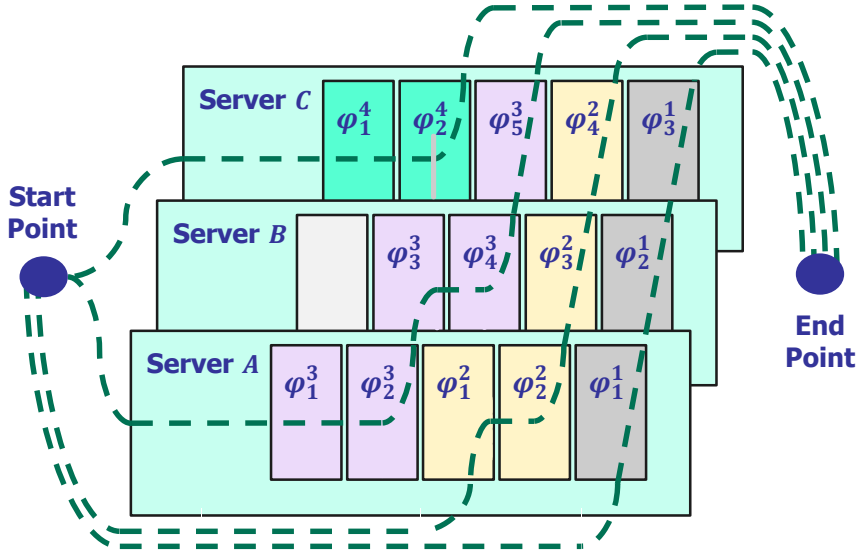
(a) Sequence of service chaining



(b) Set of servers

NFV Chaining Placement (scheduling)

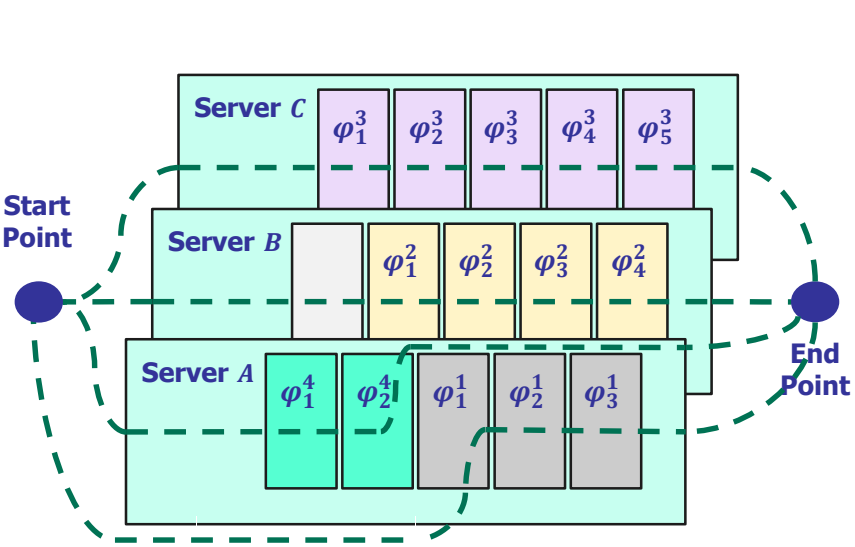
Consider the following sequence of service chaining (a), each with a specified amount of traffic to be processed, and a set of physical servers (b):

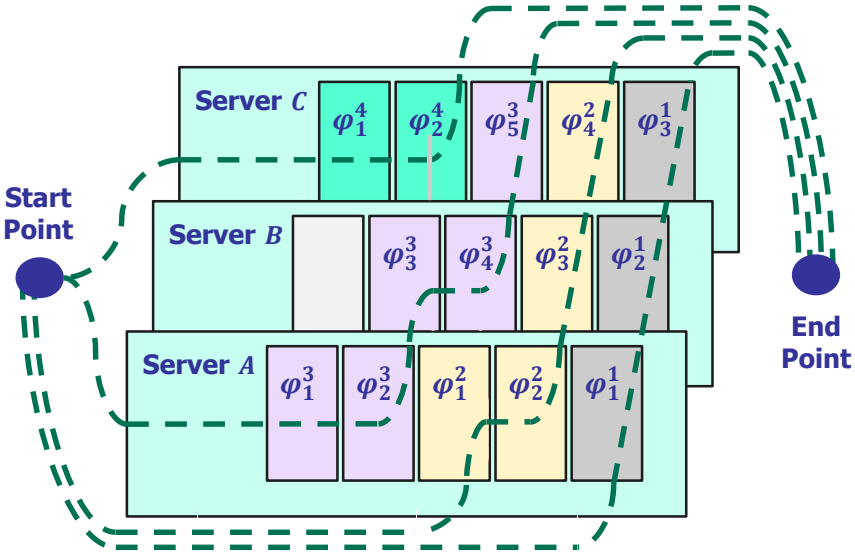


(a) Sequence of service chaining

NFV Chaining Placement (scheduling)

But, how should they be placed?



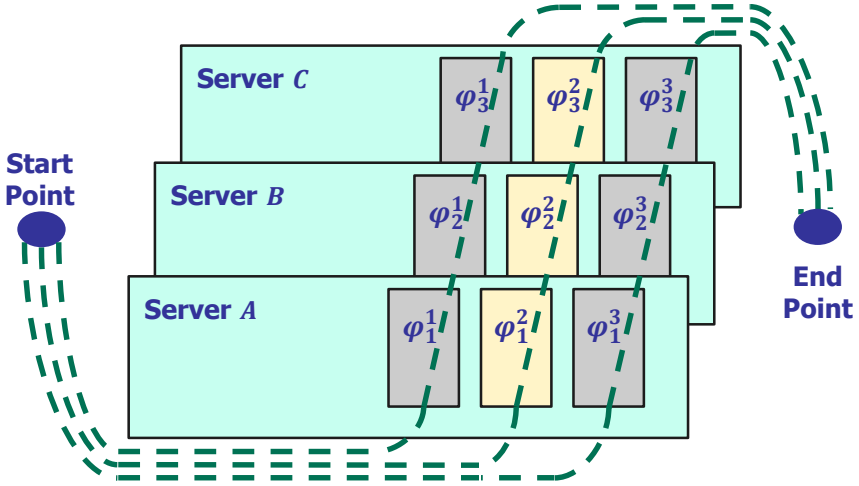


NFV Chaining Placement (scheduling)

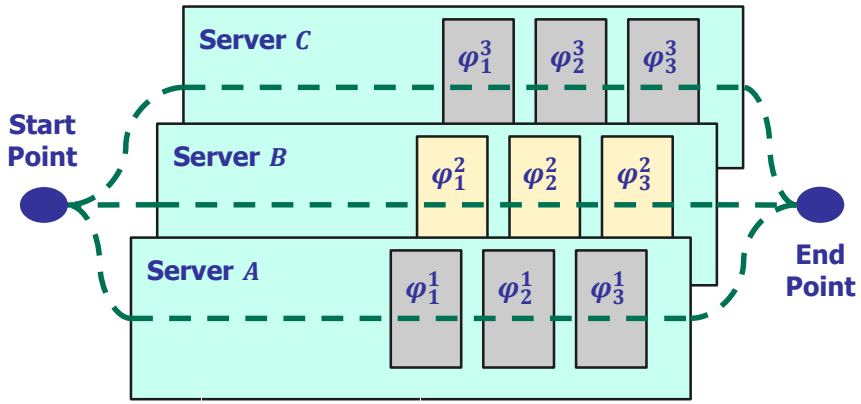
But, how should they be placed?

Consider the following simplified placement extremes:

Distribute each chain between servers



Gather each chain on a specific server



NFV Chaining Placement (scheduling)

But, how should they be placed?

Metrics	Distributing VNFs 	Gather VNFs 
Networking traffic	Same subnet traffic (aggregate)	Reduced subnet traffic (split)
Availability level (independent server failures)	0% available	66% available
Hardware utilization	Better utilization of specialized hardware (if such exists)	Balanced usage of common HW
Migration/state management (stateful VNFs)	External state transfer	Internal state transfer
Switching	Network switching capabilities	Virtual switching cost

NFV Chaining Placement (scheduling)

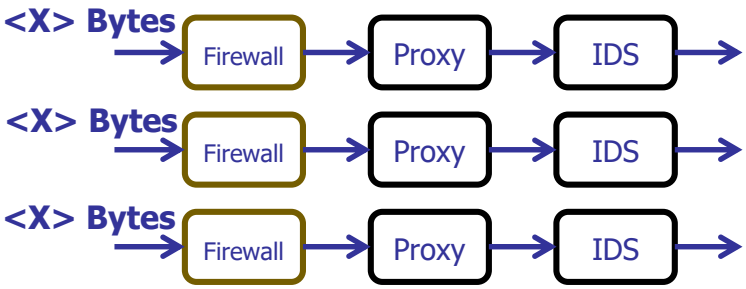
But, how should they be placed?

Optimizing the cost of virtual switching for service chaining

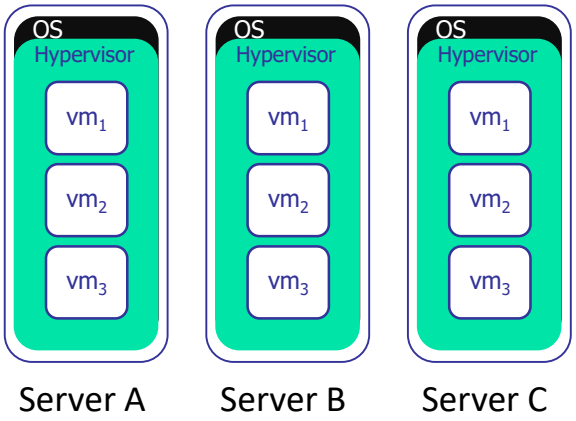
Metrics	Distributing VNFs 	Gather VNFs 
Networking traffic	Same subnet traffic (aggregate)	Reduced subnet traffic (split)
Availability level (independent server failures)	0% available	66% available
Hardware utilization	Better utilization of specialized hardware (if such exists)	Balanced usage of common HW
Migration/state management (stateful VNFs)	External state transfer	Internal state transfer
Switching	Network switching capabilities	Virtual switching cost

NFV Chaining Placement (scheduling)

Consider the following set of service chaining (a), each with a specified amount of traffic to be processed, and a set of physical servers (b):



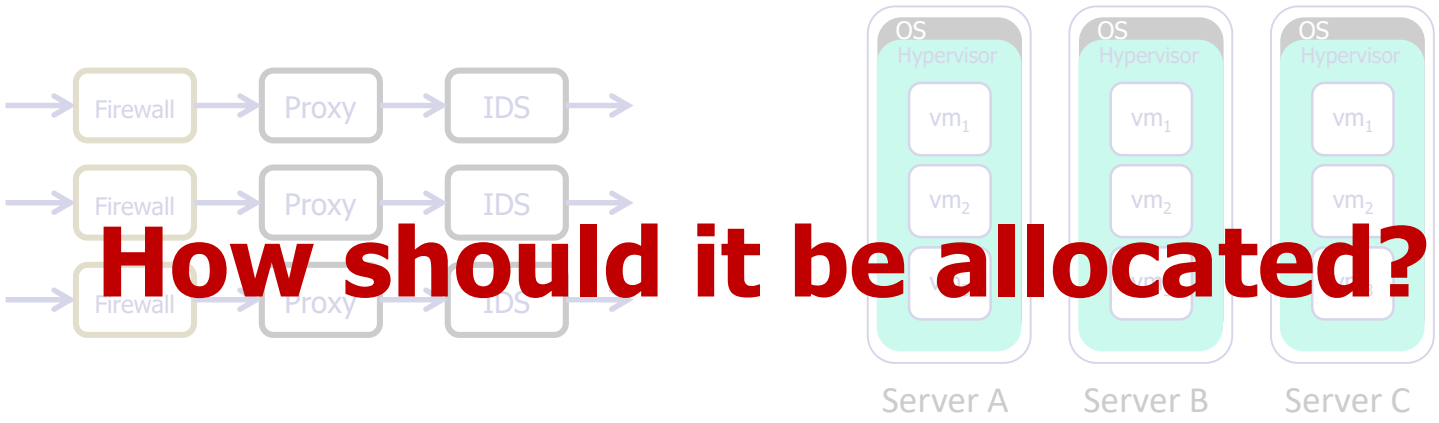
(a) Set of service chaining



(b) Set of servers

NFV Chaining Placement (scheduling)

Consider the following set of service chaining (a), each with a specified amount of traffic to be processed, and a set of physical servers (b):



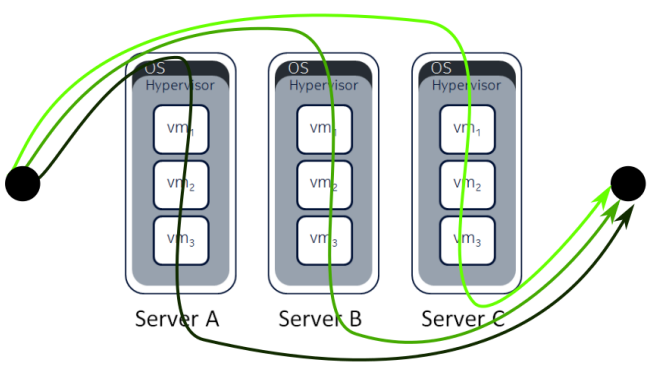
How should it be allocated?

(a) Set of service chaining

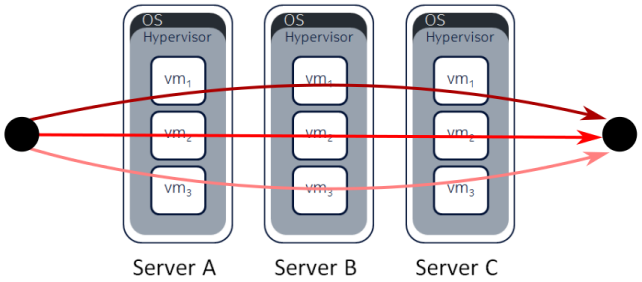
(b) Set of servers

NFV Chaining Placement (scheduling)

Consider the following set of service chaining (a), each with a specified amount of traffic to be processed, and a set of physical servers (b):



gather



distributed

Caggiani Luizelli, Raz, Saar and Yallouz, "The Actual Cost of Software Switching for NFV Chaining", IM '17.

BW – A Single Server

Environment – hardware

- Server: ProLiant DL380p Gen8
  - 2 sockets: each Xeon(R) CPU E5-2697 (12 cores)
  - Intel 82599ES 10-Gigabit NIC
  - 2 NUMA of 12 banks (each is 16GB – total 384GB)
- Hyperthreading & turboboost: disabled
- Isolation: 4-12 (HV), 20-12 (VMs)

Environment – software

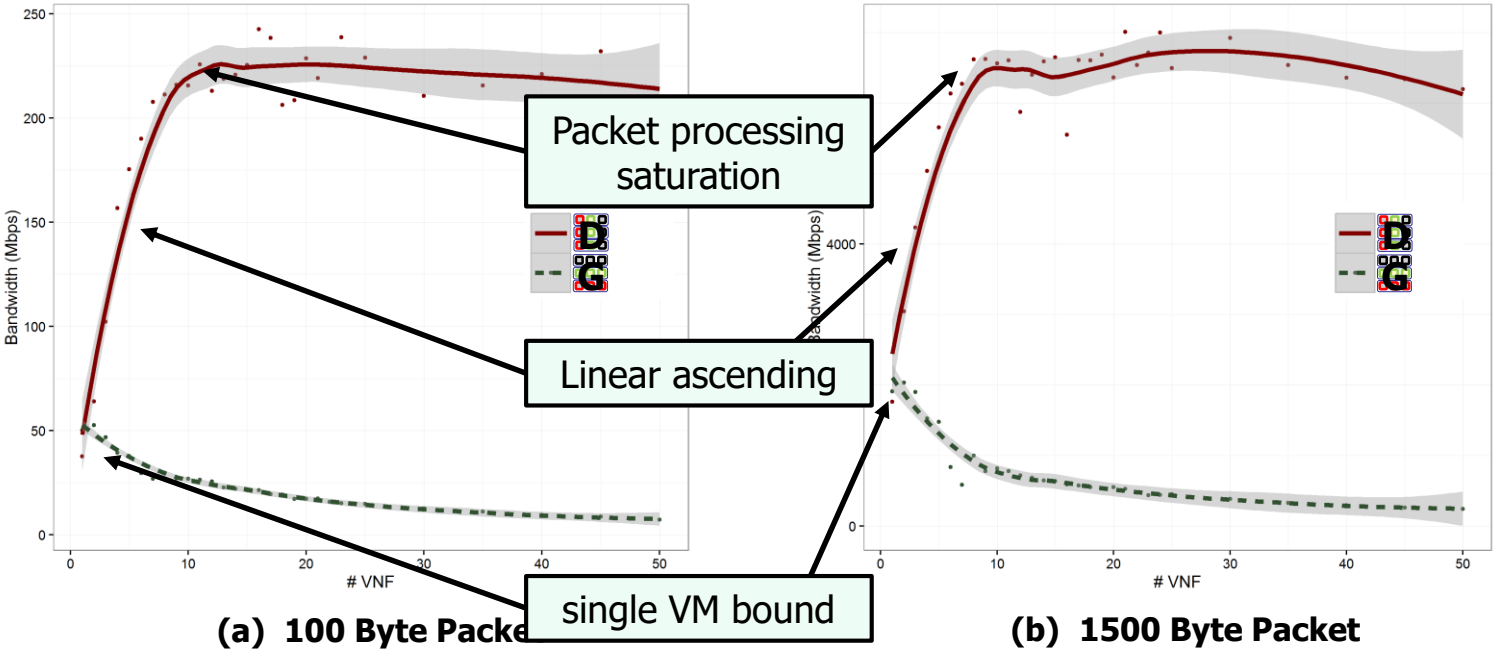
- Host (CentOS 3.10); Guest (Fedora 4.0.4)
- VM pinning
- Open vSwitch: 2.3.1
- TCP optimizations (offloading): disabled
- RSS and irqbalance:
  - queues set up according to kernel CPU

Metrics and tools

- Evaluated metrics: bandwidth, CPU utilization, packet processing capabilities, and response time
- Tools: sar, ping, sockperf (traffic generator), Linux counters

Parameters					
Experiment	**Gather**	**Distribute**	CPU isolation	**10**	
Flows	**Distributed (50 flows)**		VM pinning	**On (10-23)**	Off
Packet size	**100**	**1500**	NIC offloading	**Off**	**Off**
OVS mode	user	**kernel**	Tuning RSS	**On**	**On**
			Number of VMs		**{1 … 50}**

Note that up until a certain point, the VM's BW is the bottleneck. From that point on, the OVS is the main factor.



(a) 100 Byte Packe

(b) 1500 Byte Packet

BW – Many Servers

Environment – hardware

- Server: ProLiant DL380p Gen8
  - 2 sockets: each Xeon(R) CPU E5-2697 (12 cores)
  - Intel 82599ES 10-Gigabit NIC
  - 2 NUMA of 12 banks (each is 16GB – total 384GB)
- Hyperthreading & turboboost: disabled
- Isolation: 4-12 (HV), 20-12 (VMs)

Environment – software

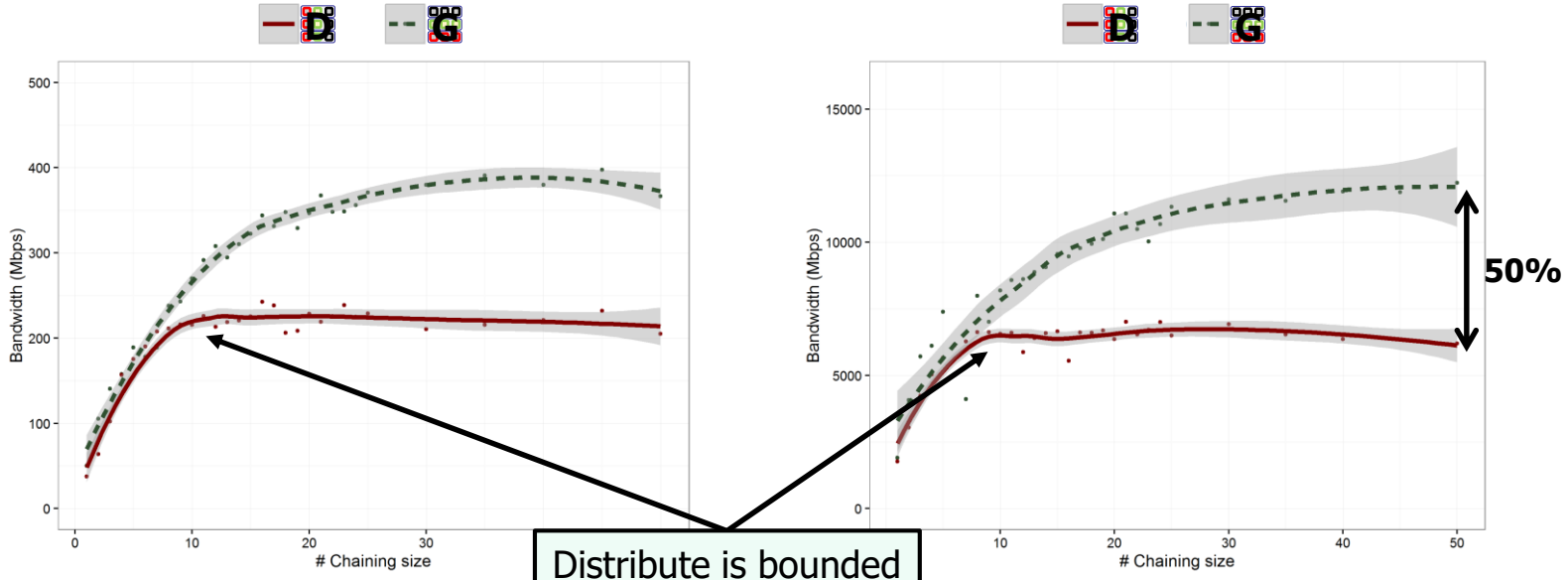
- Host (CentOS 3.10); Guest (Fedora 4.0.4)
- VM pinning
- Open vSwitch: 2.3.1
- TCP optimizations (offloading): disabled
- RSS and irqbalance:
  - queues set up according to kernel CPU

Metrics and tools

- Evaluated metrics: bandwidth, CPU utilization, packet processing capabilities, and response time
- Tools: sar, ping, sockperf (traffic generator), Linux counters

Parameters					
Experiment	**Gather**	**Distribute**	CPU isolation	**10**	
Flows	**Distributed (50 flows)**		VM pinning	**On (10-23)**	Off
Packet size	**100**	**1500**	NIC offloading	**Off**	**Off**
OVS mode	user	**kernel**	Tuning RSS	**On**	**On**
			Number of VMs		**{1 … 50}**

The difference between gather and distribute deployment might be as high as 50%!



(a) 100 Byte Packet **(b) 1500 Byte Packet**

CPU Utilization

Environment – hardware

- Server: ProLiant DL380p Gen8
  - 2 sockets: each Xeon(R) CPU E5-2697 (12 cores)
  - Intel 82599ES 10-Gigabit NIC
  - 2 NUMA of 12 banks (each is 16GB – total 384GB)
- Hyperthreading & turboboost: disabled
- Isolation: 4-12 (HV), 20-12 (VMs)

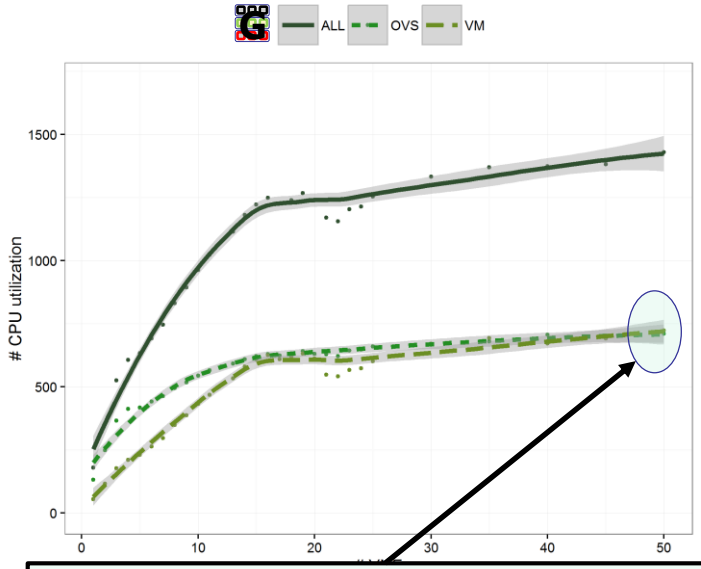
Environment – software

- Host (CentOS 3.10); Guest (Fedora 4.0.4)
- VM pinning
- Open vSwitch: 2.3.1
- TCP optimizations (offloading): disabled
- RSS and irqbalance:
  - queues set up according to kernel CPU

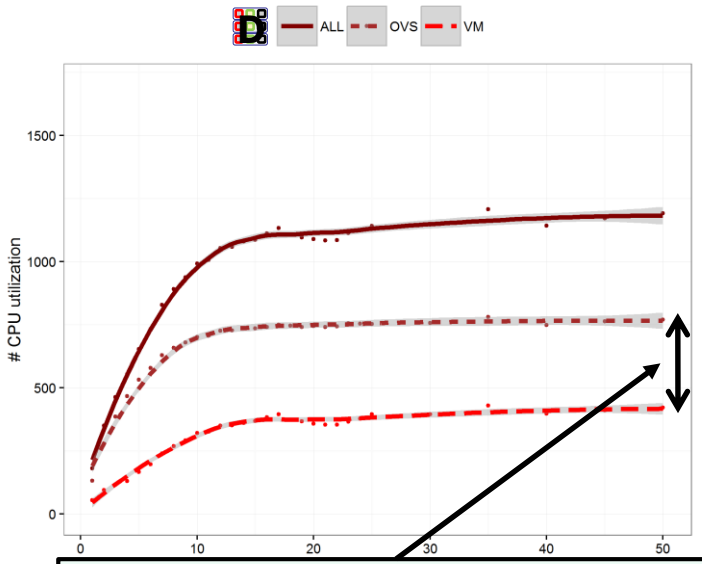
Metrics and tools

- Evaluated metrics: bandwidth, CPU utilization, packet processing capabilities, and response time
- Tools: sar, ping, sockperf (traffic generator), Linux counters

Parameters					
Experiment	**Gather**	**Distribute**	CPU isolation	**8**	
Flows	**Distributed (50 flows)**		VM pinning	**On (10-23)**	Off
Packet size	**100**	1500	NIC offloading	**Off**	**Off**
OVS mode	user	**kernel**	Tuning RSS	**On**	**On**
			Number of VMs		**{1 … 50}**



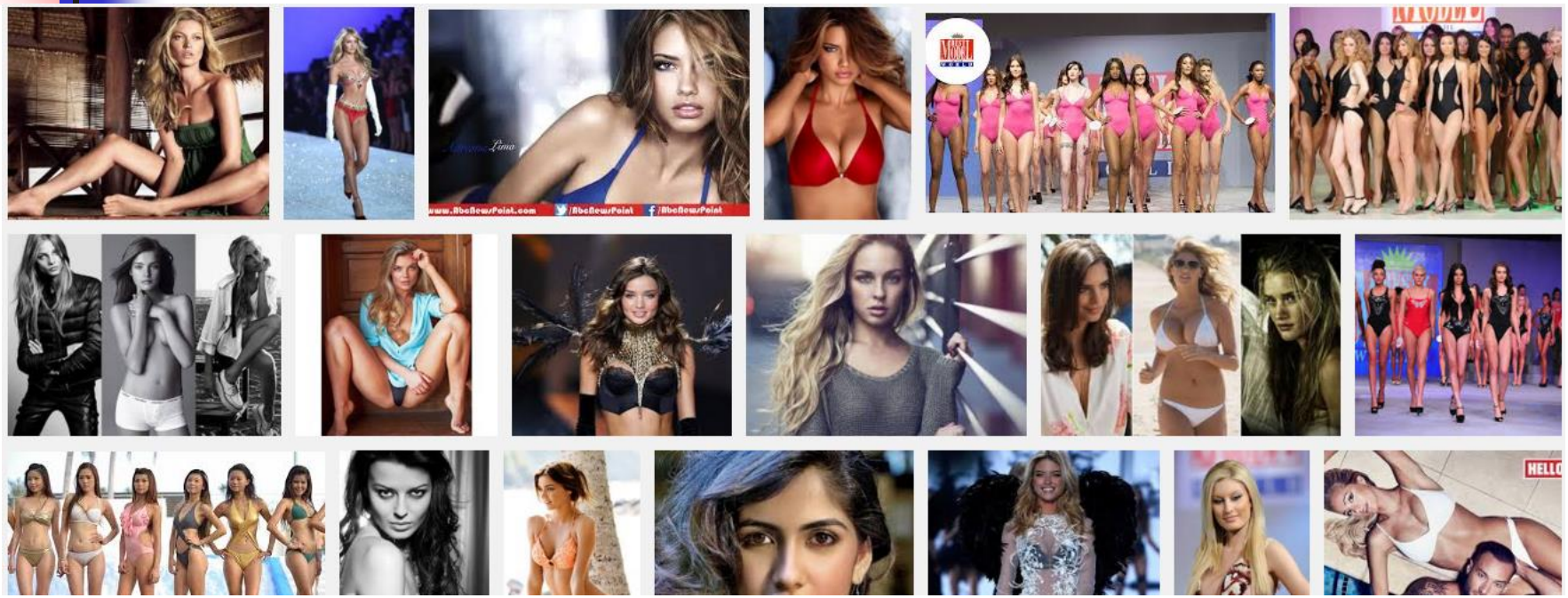
The OVS has a bigger order of packets to process (square) compared to the VM, resulting in competition over resources



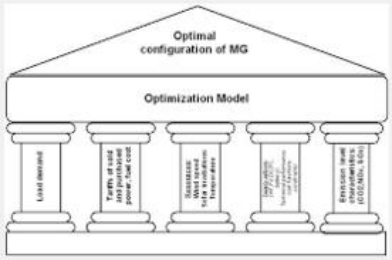
The OVS packet processing workload is smaller, resulting in limitation caused by other resources (no CPU saturation)

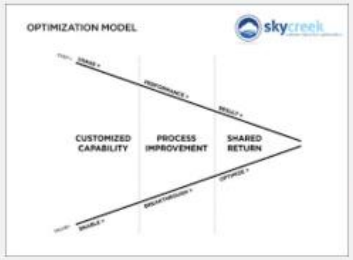
In a nutshell

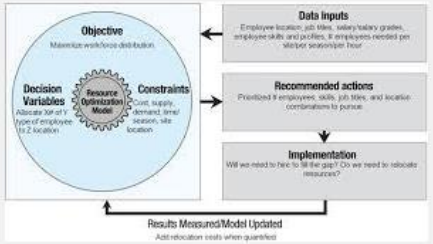
Best Model

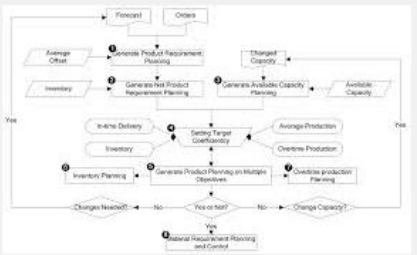


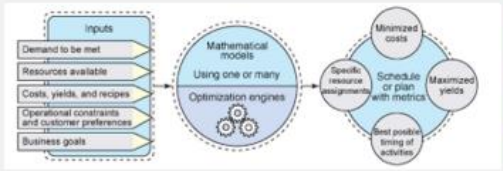
Optimization Model

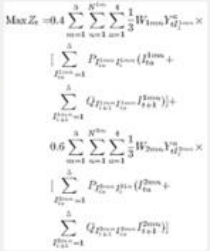


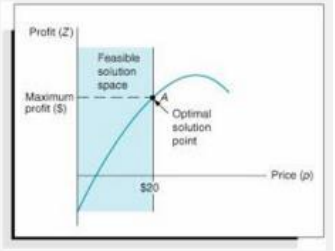


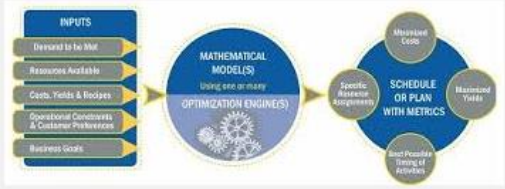


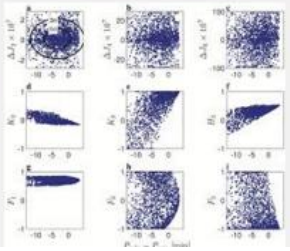


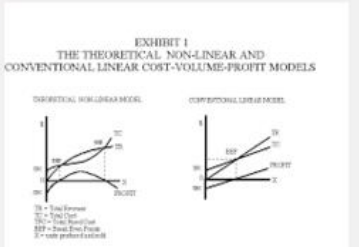


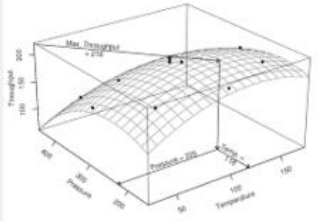


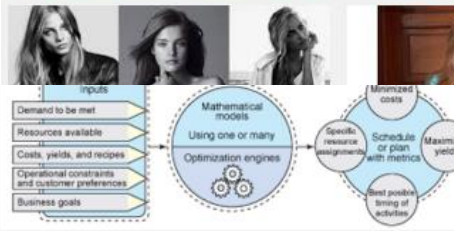


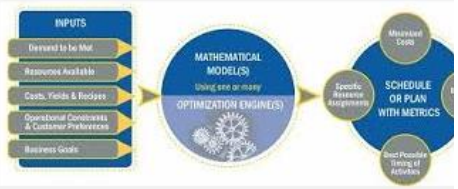




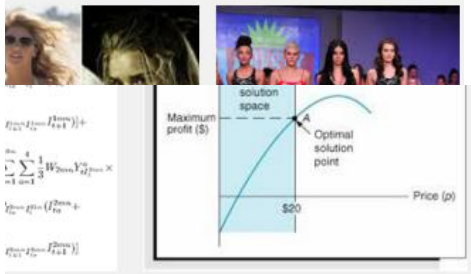


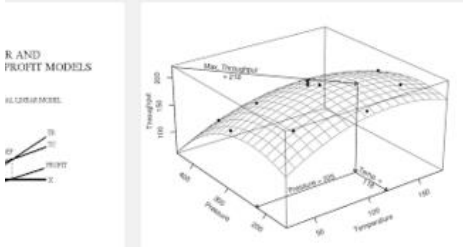






Do Not Fall in LOVE with your MODEL





danny@cs.technion.ac.il