Comparing the performance of state-of-the-art software switches in the context of NFV

Tianzhu Zhang

About me

- 2020, 8 – now Research Engineer @ Nokia Bell Labs
- 2017,10 – 2019,11 PostDoc @ Telecom ParisTech
- 2017, 6 – 2017, 9 Intern @ Nokia Bell Labs

About the project

- Joint work of TPT, NBLF, and PoliTO
- Collaborators:
  - Leonardo Linguaglossa (Telecom Paris, Paris)
  - Massimo Gallo (Nokia Bell Labs, Paris-Saclay)
  - James Roberts (Telecom Paris, Paris)
  - Luigi Iannone (Telecom Paris, Paris)
  - Paolo Giaccone (Politecnico di Torino, Turin)
- Published results:
  - Demo at IEEE Netsoft'19
  - Paper at ACM CoNEXT'19
  - On-going extension

Outline

Background

- Network Function Virtualization (NFV)
- Software switches
- Motivation

Methodology

- Test scenarios
- Testbed settings

Experimental results

Conclusion

Background

Ossification of traditional networks

Specialized hardware appliances, aka. Middleboxes

Very expensive

Difficult to manage

Inflexible to customize

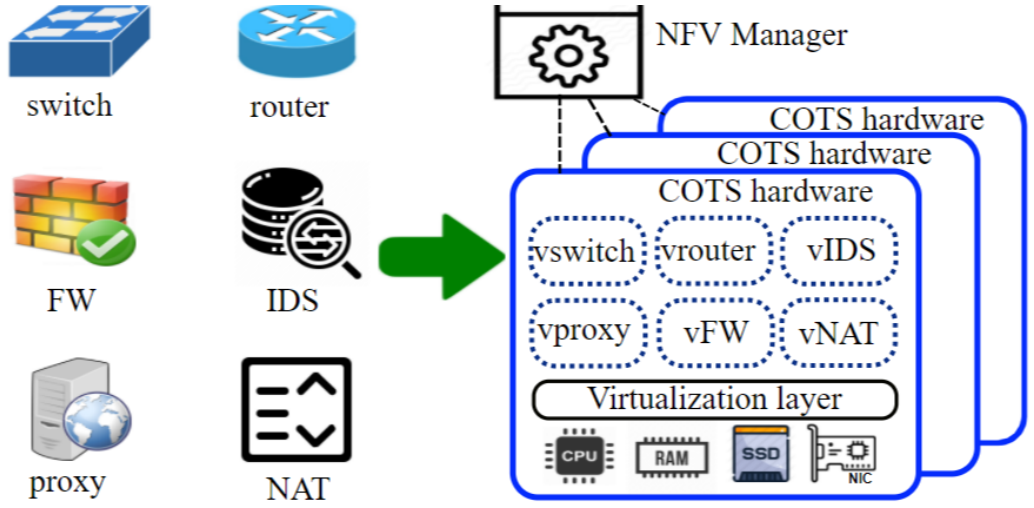
Tightly coupled with hardware production cycle

Tedious service deployment and maintenance

Non-trivial to scale in/out

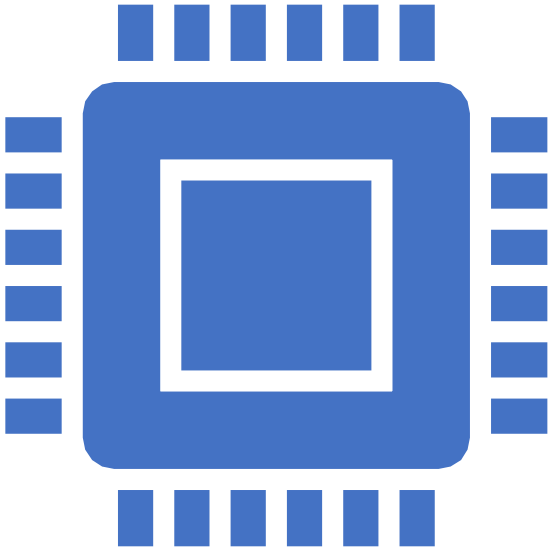
Network Function Virtualization (NFV)

- Shift towards agile and open service provisioning
  - Virtual network functions (VNFs) on Commodity Off-The-Shelf (COTS) hardware



Software switches

- Virtual switch implemented in software
- High flexibility
- Low performance
- Previously only used for fast-prototyping and function testing
- Benefit from high-speed packet I/O techniques
  - DPDK, netmap, PF_ring, eBPF, VMA, etc.
- Widely adopted by existing NFV frameworks
  - Traffic steering between VNFs and network interface cards (NICs)



	Architecture		Programming Paradigm	Model		Virtual Interface	Runtime Reprogrammability	Programming Language	Main Purpose
	Self-contained	Modular		RTC	Pipeline				
BESS		✓	Structured	✓	✓	vhost-user	High	C, Python	Programmable NIC
Snabb		✓	Structured		✓	vhost-user	High	Lua, C	VM-to-VM
OVS-DPDK	✓		Match/action	✓		vhost-user	High	C	SDN switch
FastClick		✓	Structured	✓		vhost-user	Medium	C++	Modular router
VPP	✓		Structured	✓		vhost-user	Medium	C	Full router
VALE	✓		Structured	✓		ptnet	Low	C	Virtual L2 Ethernet
t4p4s	✓		Match/action	✓		vhost-user	Low	C, Python	P4 switch

Unclear performance

- Disparate design choices and purposes
- Performance depend on traffic patterns
- Too many hardware and software parameters to tune

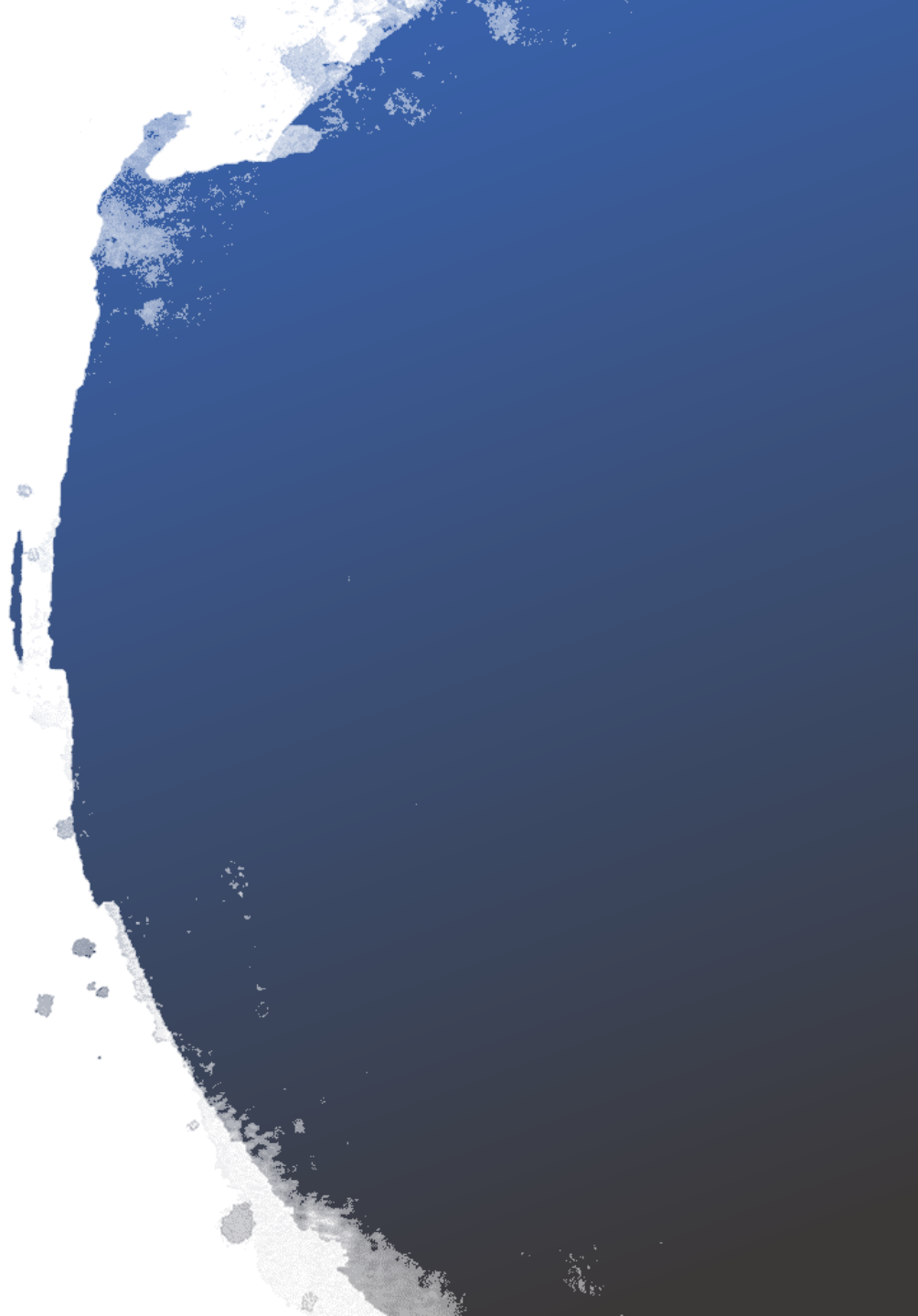
Ref.	Software switches under test	Bare-metal	VNF environment		Inter-VNF forwarding	SFC	Throughput		Latency
			VM	Container			Uni.	Bi.	
[34]	**BESS**			✓		✓	✓		✓
[71]	OVS, **OVS-DPDK**	✓	✓	✓(*)		✓	✓		✓
[72]	Linux bridge, **VALE**			✓	✓	✓	✓		
[73]	ClickOS, **BESS**					✓	✓		✓
[19]	Linux bridge, OVS, **OVS-DPDK**	✓	✓		✓		✓		✓
[25]	OVS, PISCES, **t4p4s**	✓					✓		
[74, 75]	**OVS-DPDK, VPP**		✓				✓		
[18]	**BESS, VPP, OVS-DPDK**	✓					✓		
[76]	**OVS-DPDK, Snabb, VALE**		✓		✓		✓		
[21]	**Snabb**, OVS, **OVS-DPDK**, Linux bridge		✓		✓		✓		
[77, 30]	**OVS-DPDK**, OVS, Linux bridge, Lagopus, xDPd-DPDK	✓	✓				✓		✓

Existing works

Our contribution

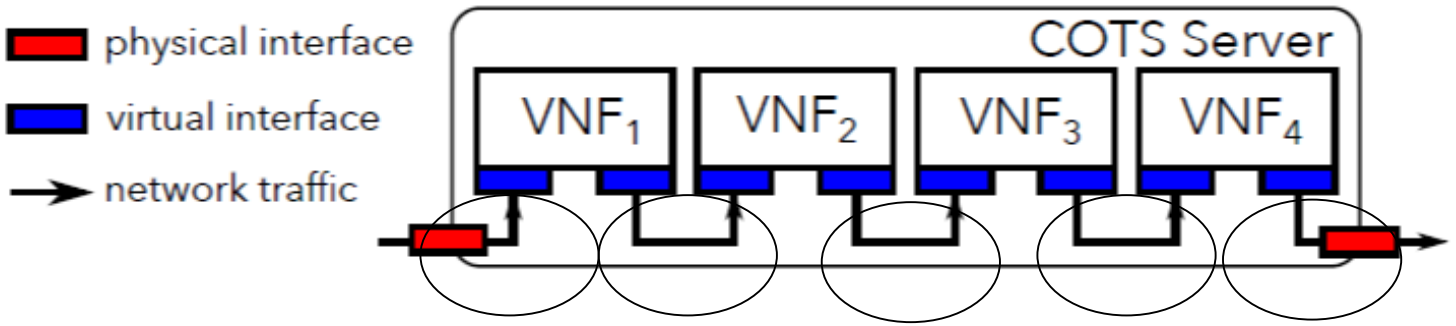
- Experimental methodology to thoroughly evaluate the performance of software switches for NFV
- Compare the performance of 7 state-of-the-art high-speed software switches, quantitative-> qualitative
- Define the best use case for each tested software switch

Methodology



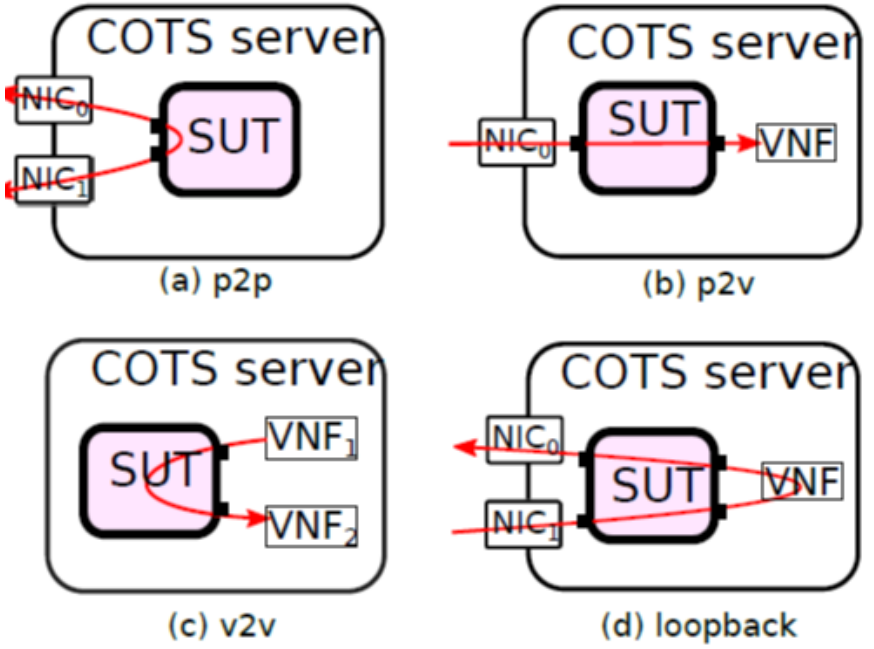
Typical NFV scenario

- Service function chain (SFC)



Test scenarios

- Physical-to-physical (p2p)
- Physical-to-virtual (p2v)
- Virtual-to-virtual (v2v)
- Loopback



Software switches under test

OVS-DPDK

FastClick

Berkley Extensible Software Switch (BESS)

Vector Packet Processing (VPP)

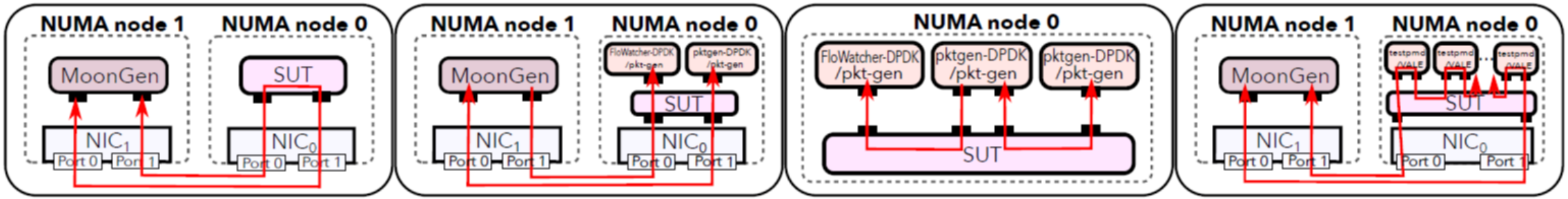
t4p4s

Snabbswitch

Netmap/VALE

Tools

- Traffic generators and monitors @ 10Gbps
  - MoonGen
  - pktgen-DPDK
  - pkt-gen
  - FloWatcher-DPDK
- Hypervisors
  - QEMU/KVM + CentOS 7
  - Docker
- Virtual network functions
  - Traffic monitors
  - testpmd/VALE



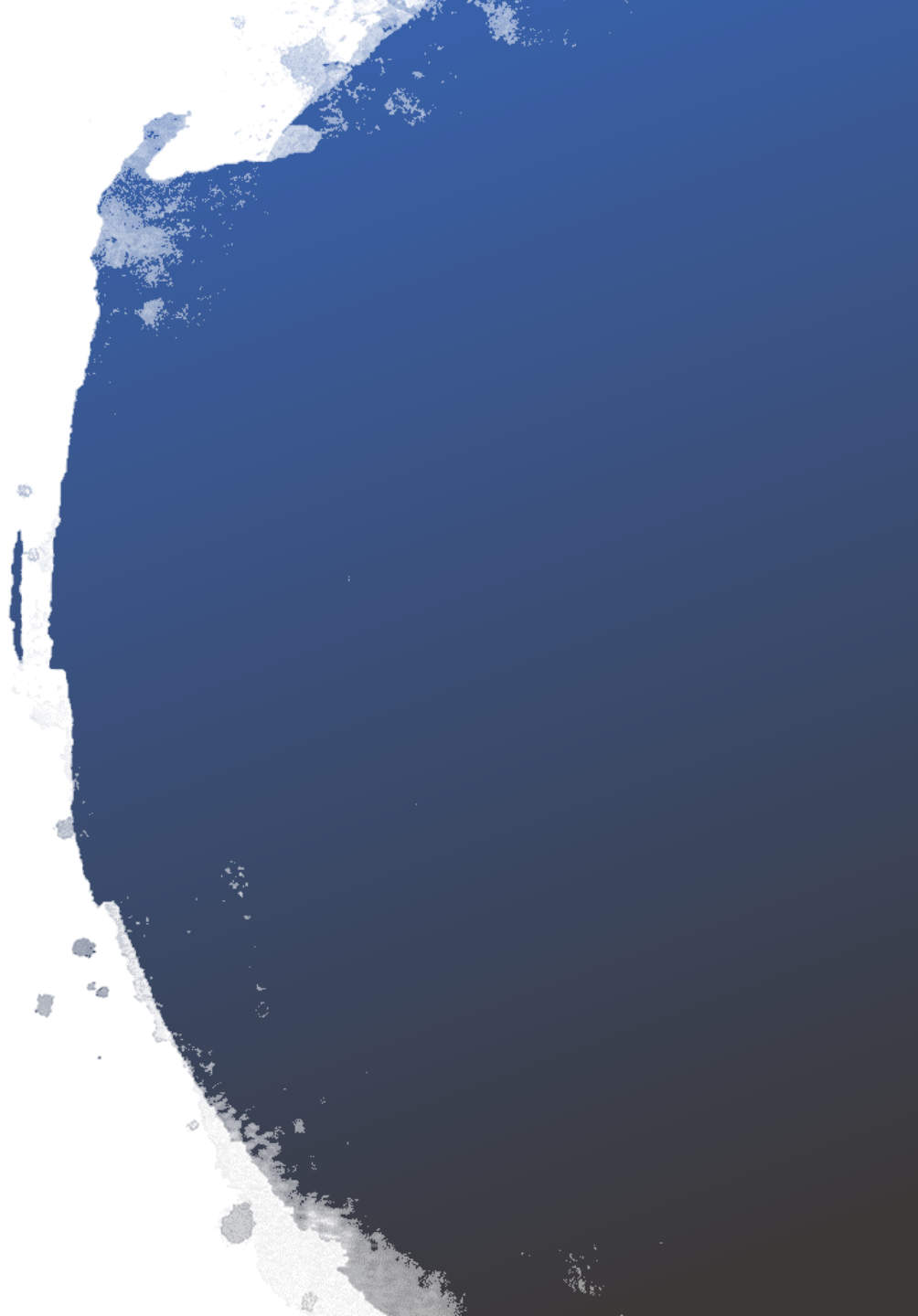
(a) p2p (b) p2v (c) v2v (d) loopback

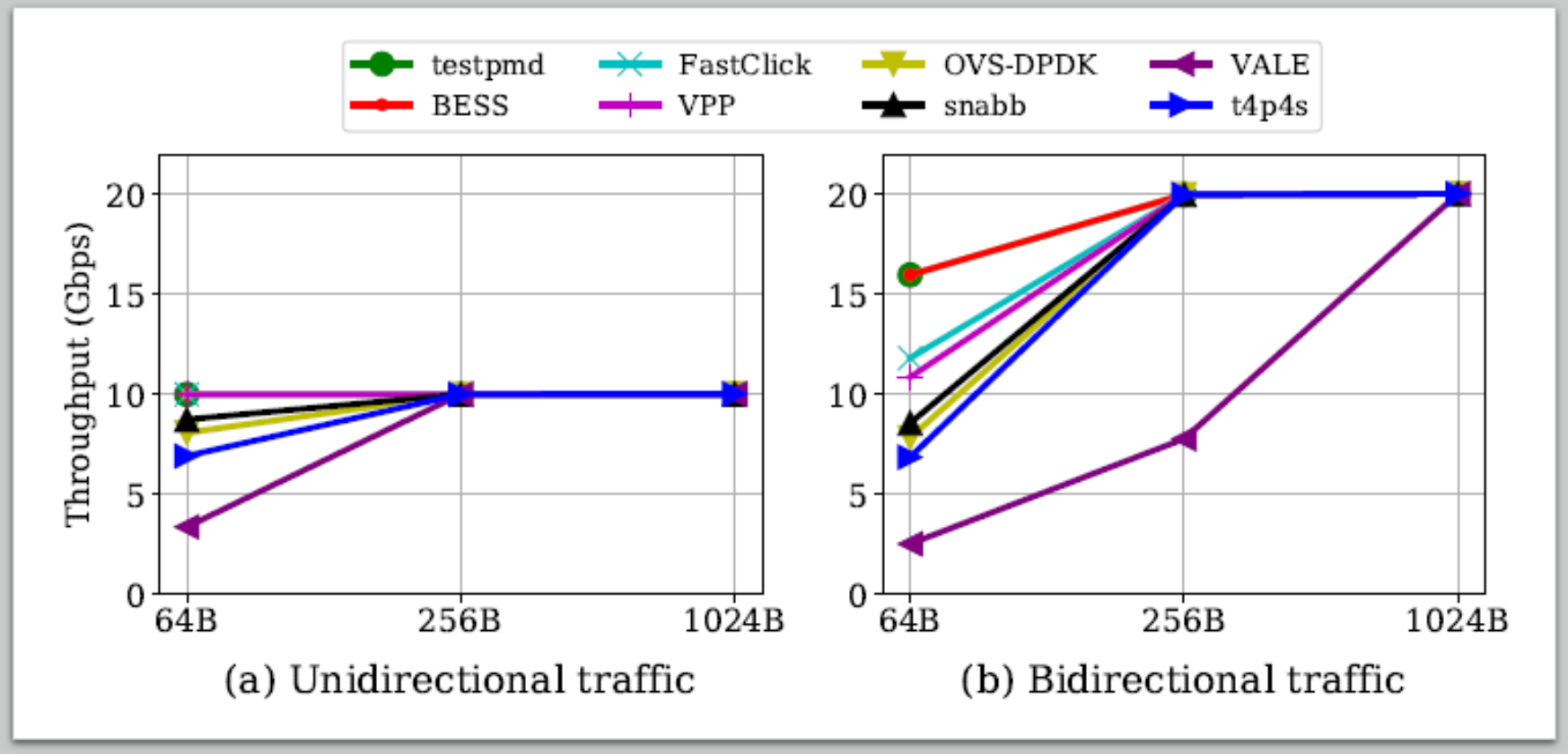
Testbed settings

Experimental parameters

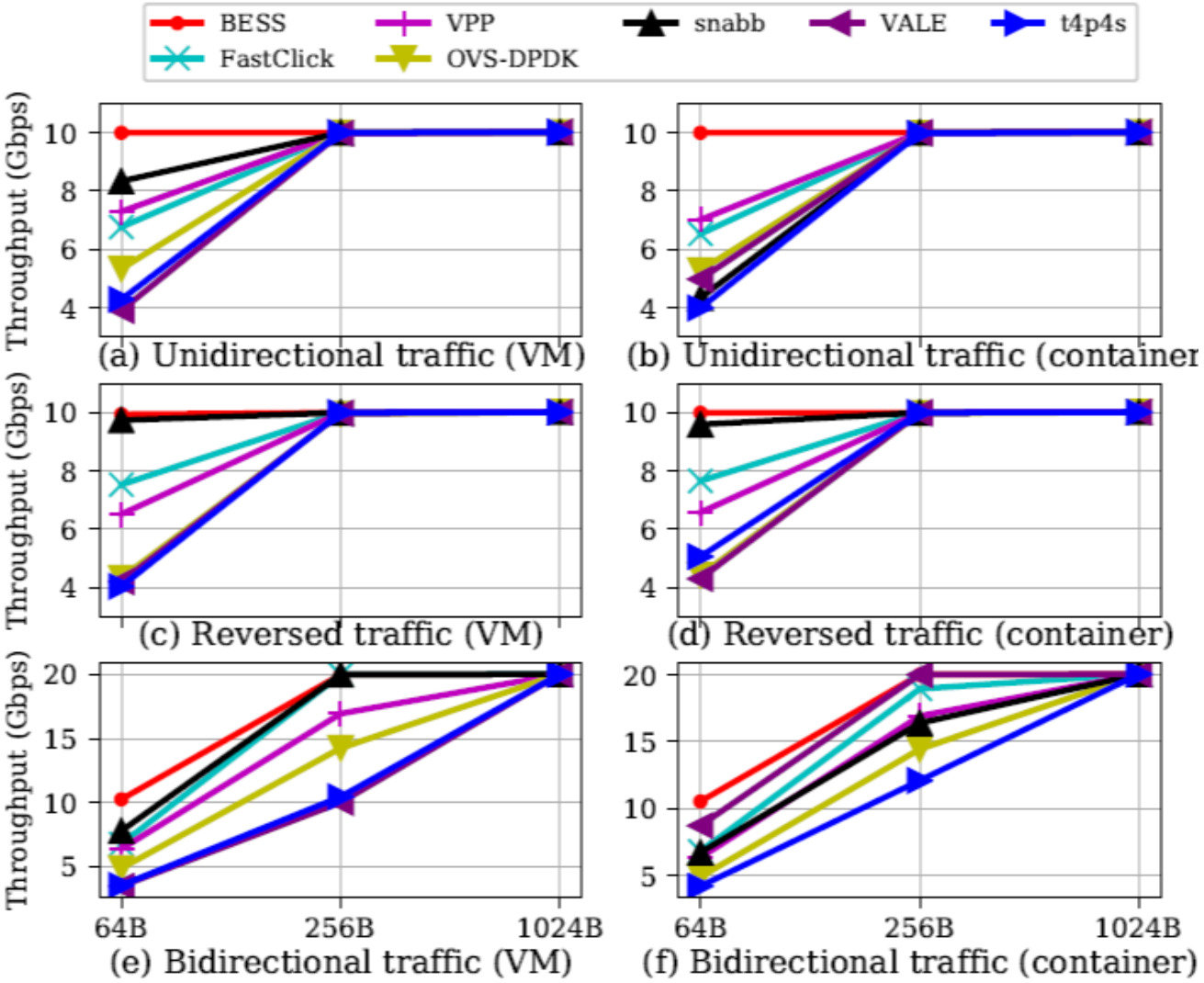
- Traffic patterns
  - Synthetic packets with 64B, 256B, and 1024B
- CPU assignment
  - Single core for software switches
  - Four cores for each VNF
- Virtual environment
  - VM and container
- Performance metrics
  - Throughput: in Gbps, both unidirectional and bidirectional
  - Latency: Round-Trip time under 0.1/0.5/0.99 of the maximal forwarding rate

Experimental results

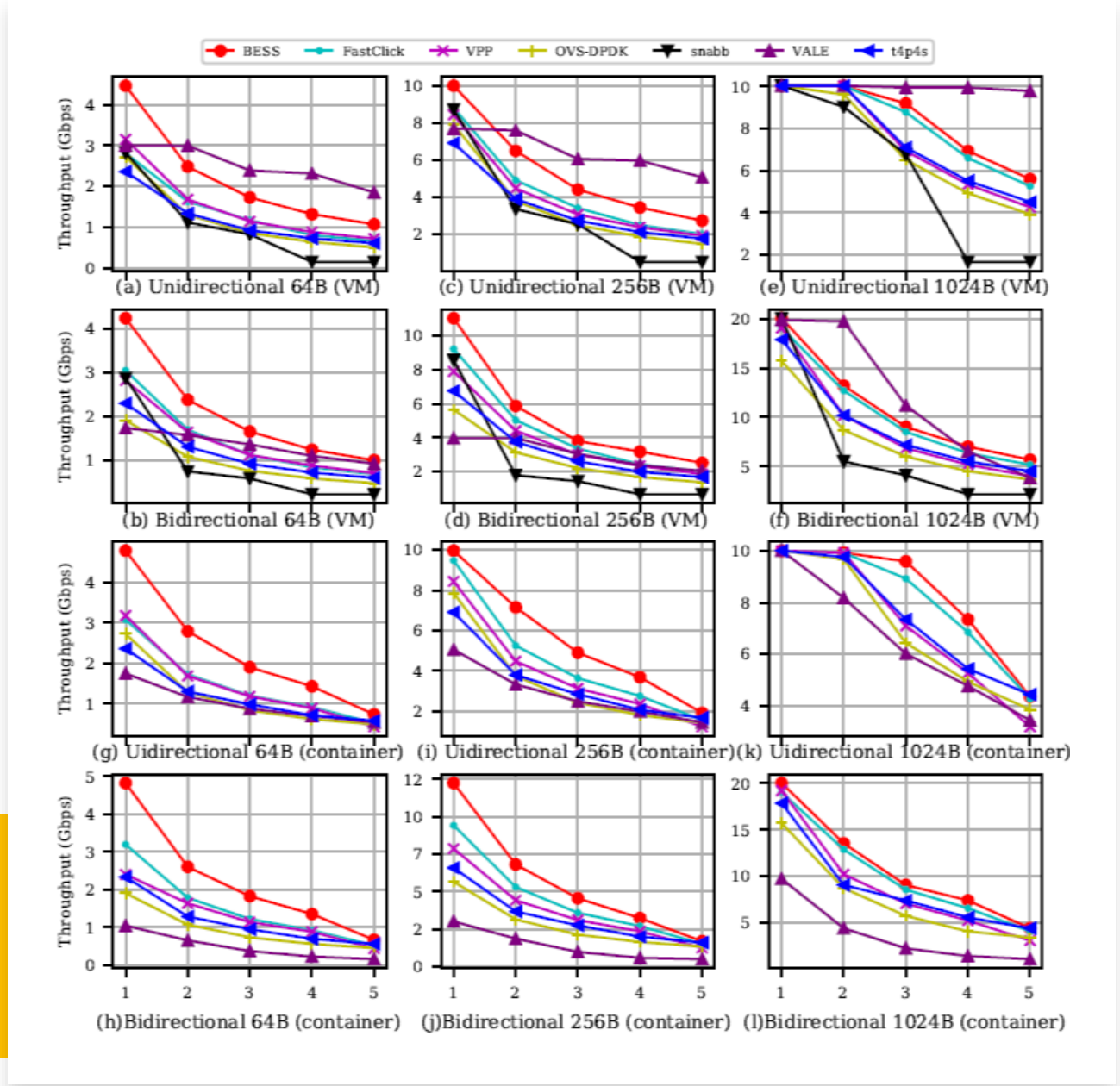




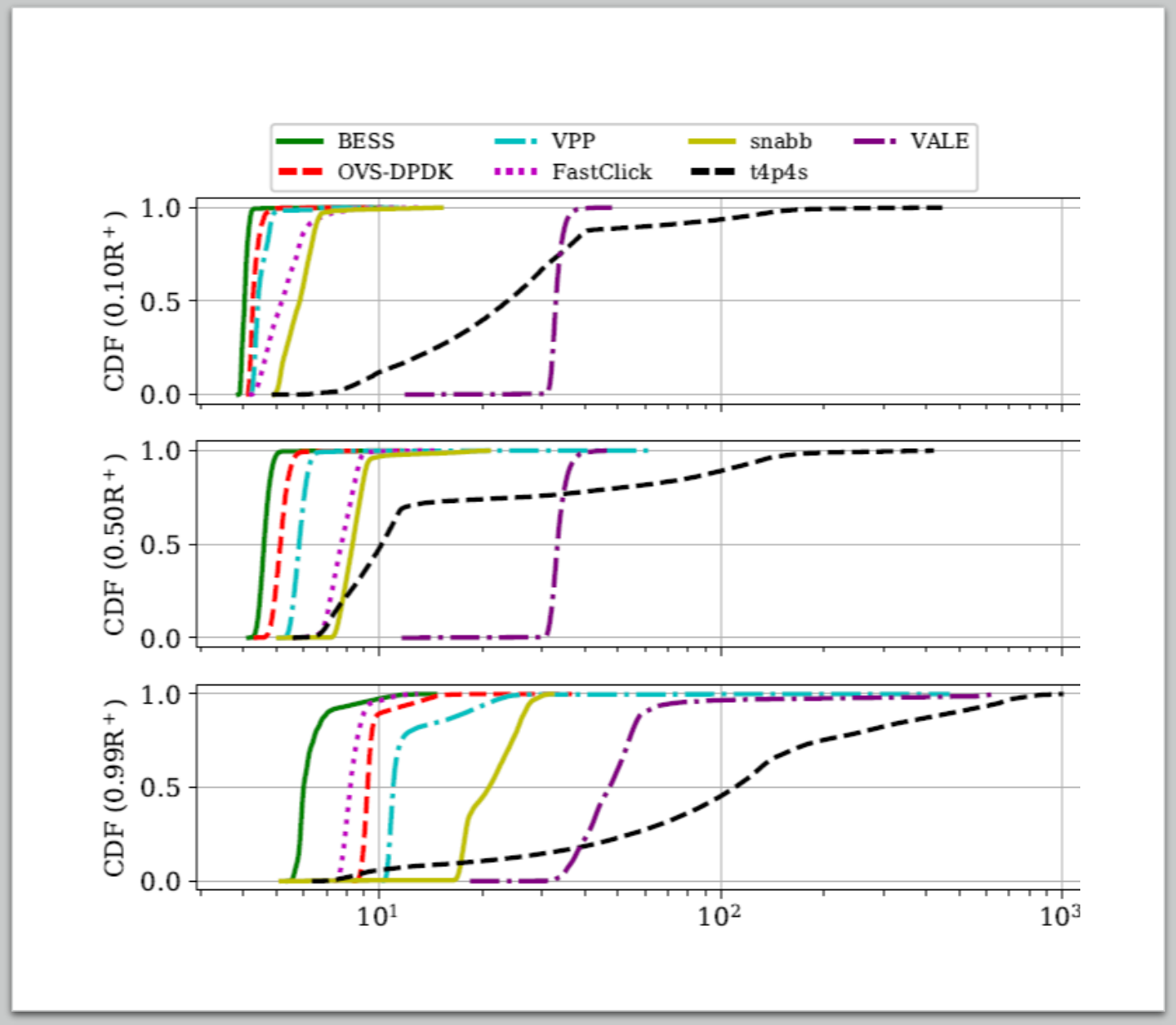
Throughput test for p2p scenario



Throughput test for p2v scenario

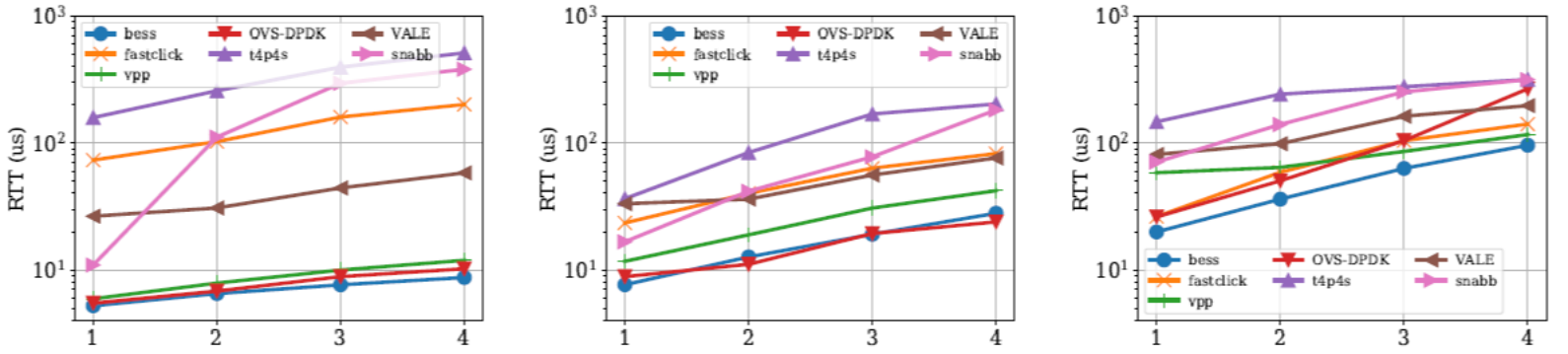


Throughput test for loopback scenario

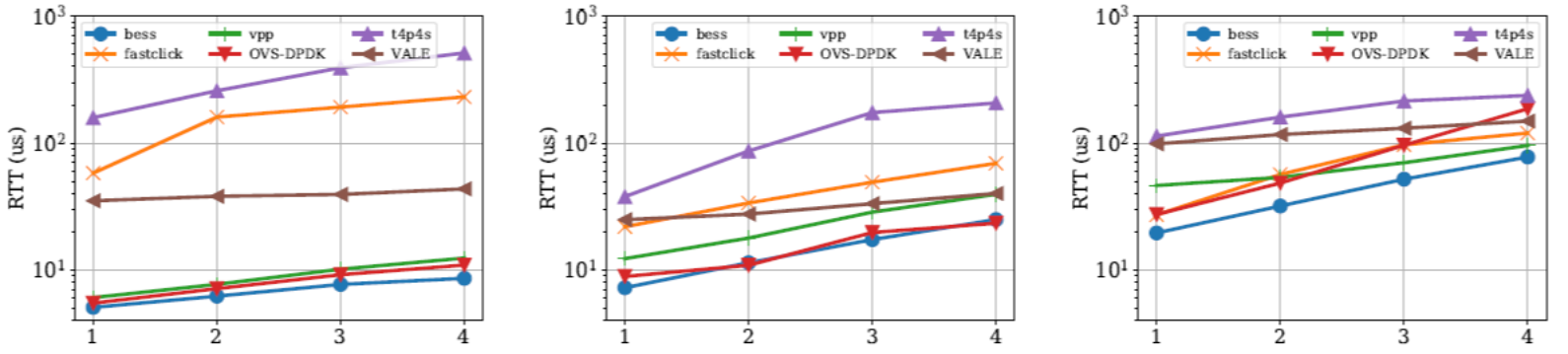


Latency test for p2p scenario

Latency test for loopback scenario



(a) $0.10R^+$ with VNFs deployed in VMs (b) $0.50R^+$ with VNFs deployed in VMs (c) $0.99R^+$ with VNFs deployed in VMs



(d) $0.10R^+$ with containerized VNFs (e) $0.50R^+$ with containerized VNFs (f) $0.99R^+$ with containerized VNFs

Use case summary

	Best use cases	Remarks
BESS	Forwarding between physical NICs and containers	Chaining of containerized VNFs
Snabb	Fast deployment, runtime optimization	Bottlenecked with multiple VNFs
OVS-DPDK	Stateless SDN deployments	Supports OpenFlow protocol
FastClick	VNF chaining	Flexible live migration, high latency at low workload
VPP	VNF chaining	Flexible live migration
VALE	VNF chaining with high workload	Limited traffic classification and live migration capability
t4p4s	Stateful SDN deployments	Supports P4 semantics

Conclusion

- Software switches are widely used by NFV frameworks
- It calls for a benchmarking methodology to assess their performance
- Our approach provides a comprehensive evaluation of software switches in a typical NFV environment
- Quantitative measurement with qualitative evaluation
- Code reproducible and available on GitHub

Thanks!!!