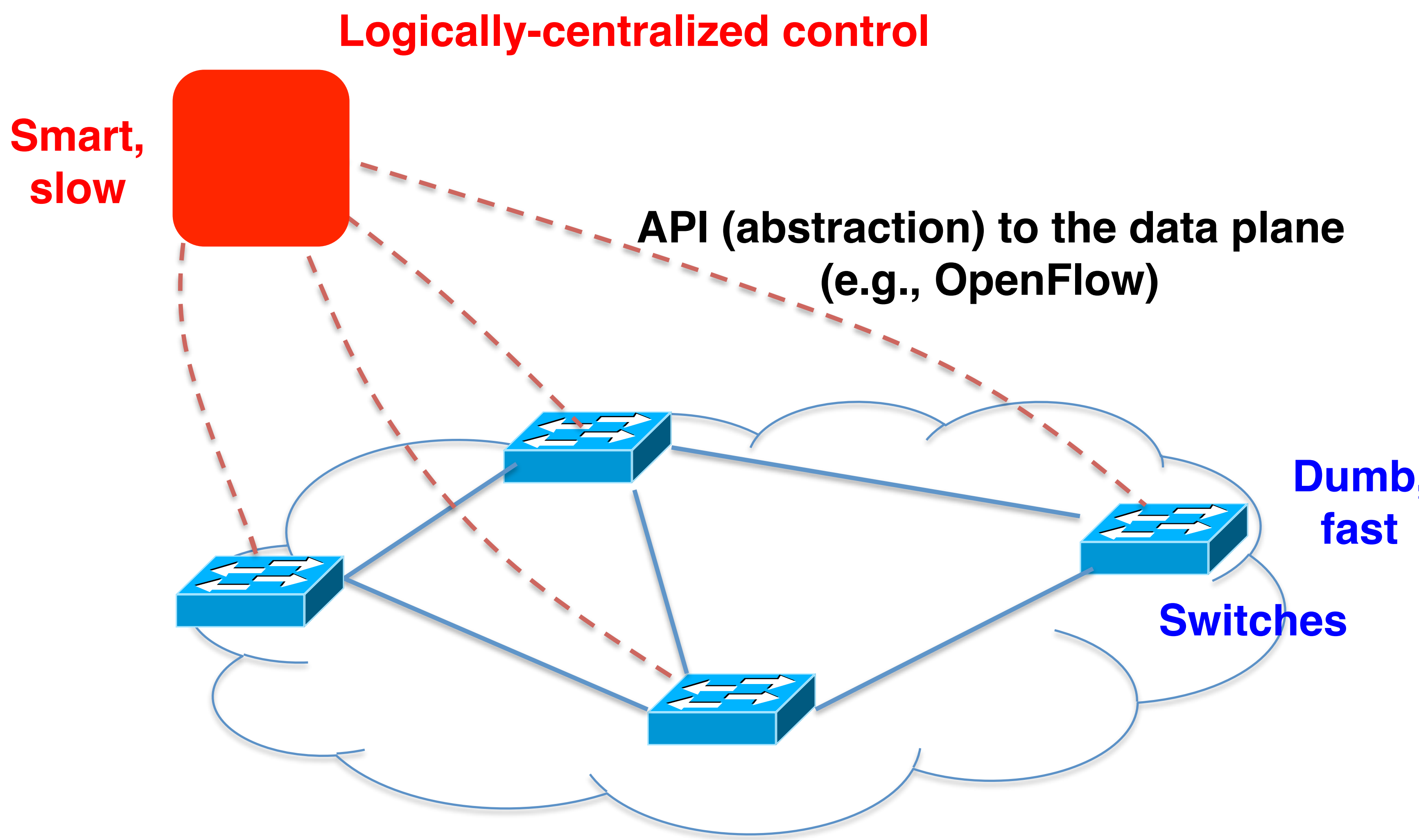
Rethinking the SDN Abstraction

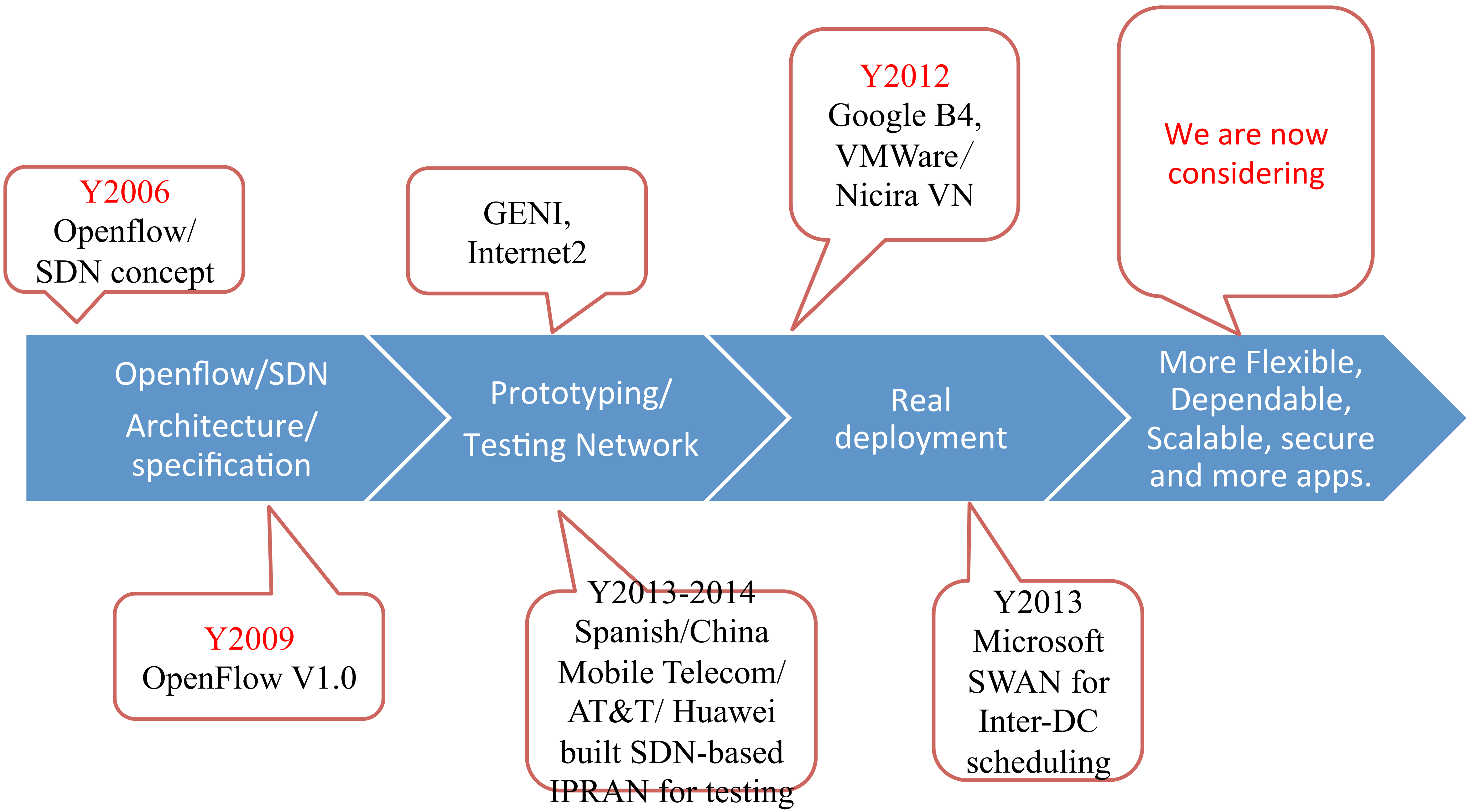
Chengchen Hu
XJTU
Oct. 15, 2016

西安交通大學
XI'AN JIAOTONG UNIVERSITY

- Separate Control and Data
- Abstraction
- Global view



History


Y2006
Openflow/
SDN concept
GENI,
Internet2
Y2012
Google B4,
VMWare/
Nicira VN
We are now
considering
Openflow/SDN
Architecture/
specification
Prototyping/
Testing Network
Real
deployment
More Flexible,
Dependable,
Scalable, secure
and more apps.
Y2009
OpenFlow V1.0
Y2013-2014
Spanish/China
Mobile Telecom/
AT&T/ Huawei
built SDN-based
IPRAN for testing
Y2013
Microsoft
SWAN for
Inter-DC
scheduling

OPEN NETWORKING FOUNDATION

OF-PI: A Protocol Independent Layer

Version 1.1
September 5, 2014

ONF TR-505

POF: Protocol Oblivious Forwarding

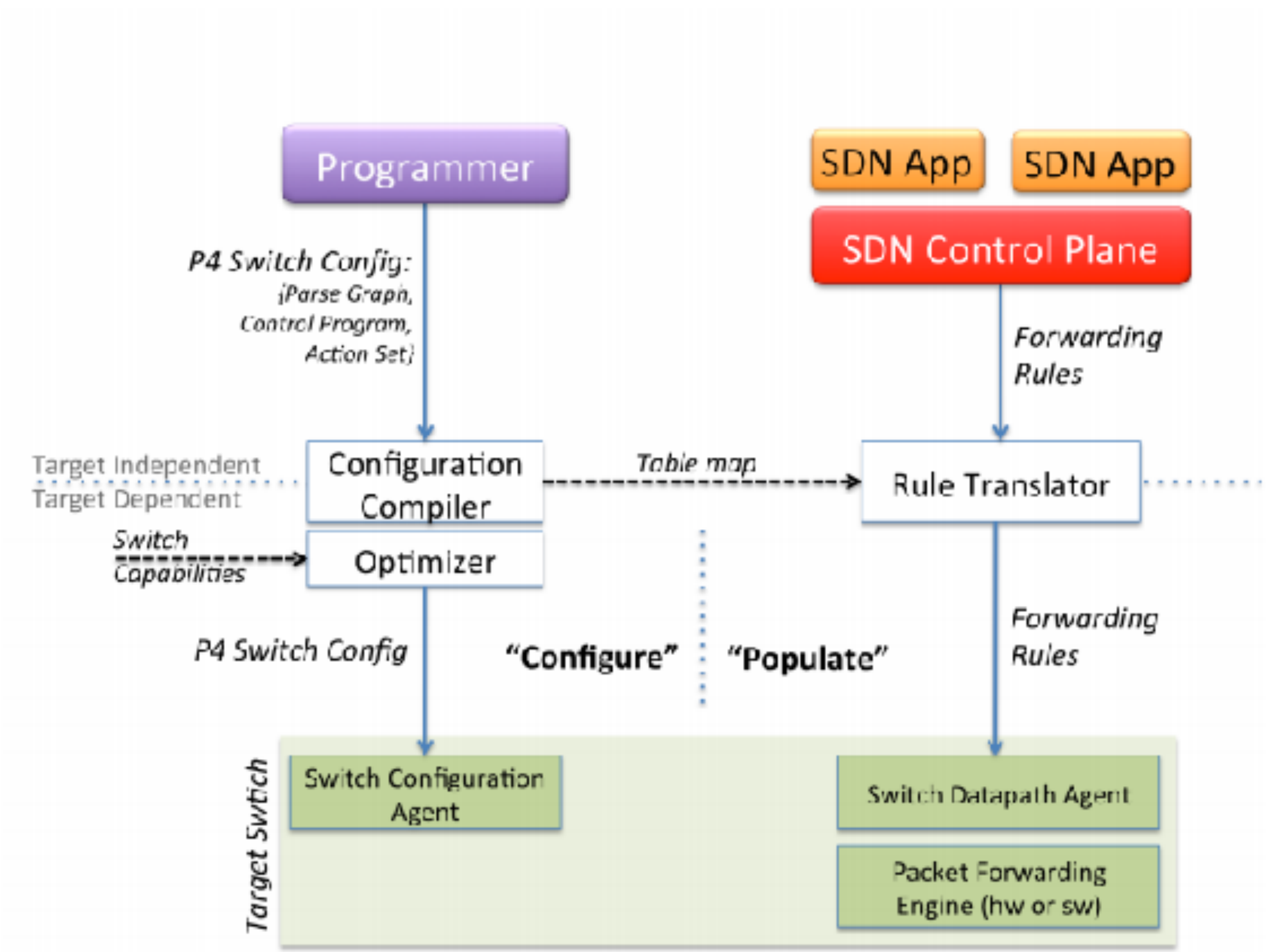
- Table search keys are defined as {offset, length} tuples
- Instructions/Actions access packet data or metadata using {offset, length} tuples
- Include other math, logic, move, branching, and jump instructions

	Current OpenFlow	POF
Match	~40 matching header fields defined yet still many uncovered protocols/headers	{offset, length} covers any frame based formats
Action	OFPAT_COPY_TTL_OUT OFPAT_COPY_TTL_IN OFPAT_SET_MPLS_TTL OFPAT_DEC_MPLS_TTL OFPAT_PUSH_VLAN OFPAT_POP_VLAN OFPAT_PUSH_MPLS OFPAT_POP_MPLS OFPAT_SET_NW_TTL OFPAT_DEC_NW_TTL OFPAT_PUSH_PBB OFPAT_POP_PBB and on and on and on …	POFAT_SET_FIELD POFAT_ADD_FIELD POFAT_DELETE_FIELD POFAT_MOD_FIELD Period.

Packet field parsing and handling are abstracted as generic instructions to enable flexible and future proof forwarding elements. This is simple yet has profound implications to SDN.

Haoyu Song, Protocol-Oblivious Forwarding: Unleash the Power of SDN through a Future-Proof Forwarding Plane @ **HotSDN 2013**

P4: Programming Protocol-Independent Packet Processors

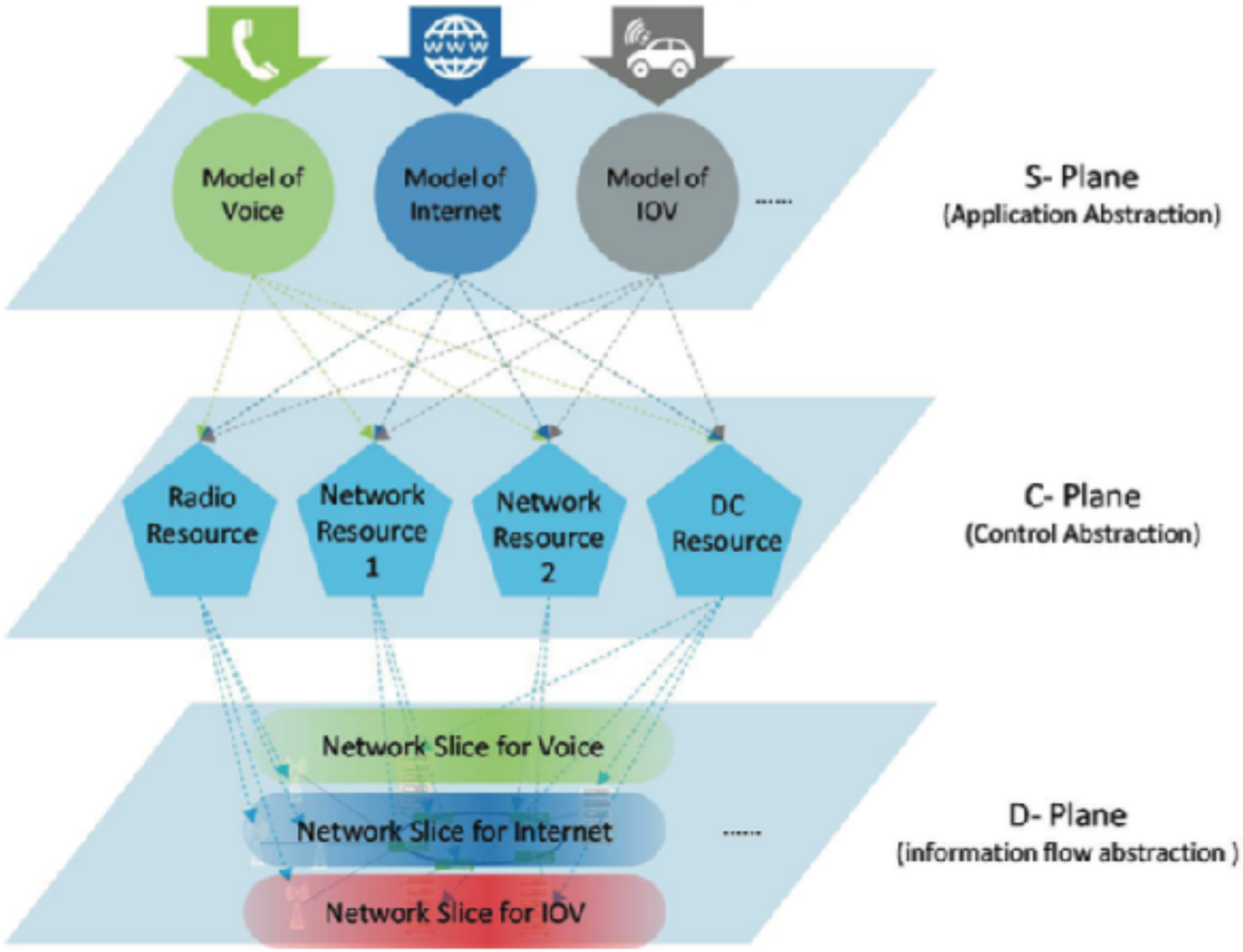


Proposed by
- Nick McKeown
- Jennifer Rexford
- Amin Vahdat
- George Varghese

Goals
- Protocol independence
- Target independence
- Reconfigurability

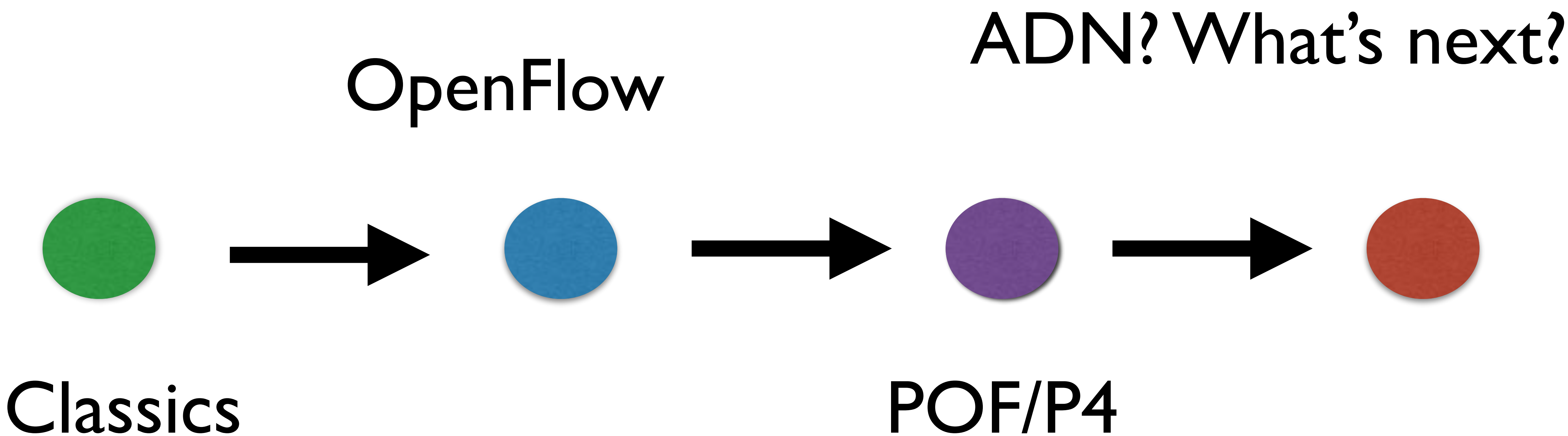
"P4: Programming Protocol-Independent Packet Processors," ***ACM Sigcomm Computer Communications Review (CCR).*** *Volume 44, Issue #3 (July, 2014)*

Application-Driven Network



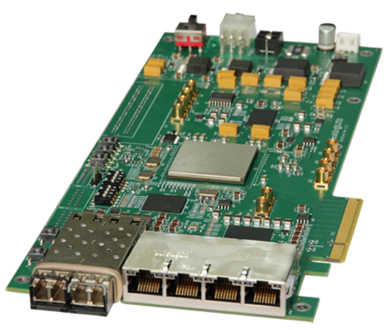
Application Driven Network: providing On-Demand Services for Applications. Demo&Poster @ **SIGCOMM '16** (co-work with Huawei).

HUAWEI



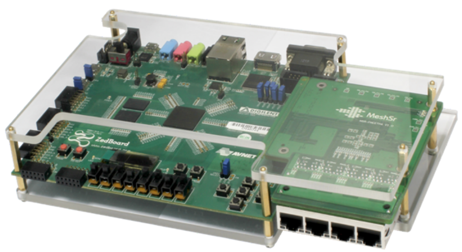
POF/P4 is flexible enough?

Case 0: programmable SDN Data Plane

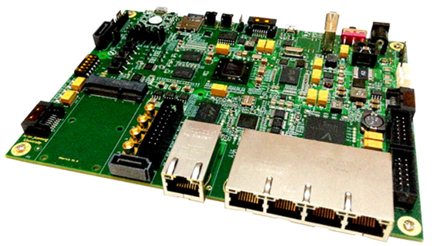


ONetCard
2012 Aug
PCIe Card

ONetSwitch 45
4*10G, 4*GE, wifi
2013 Aug



ONetSwitch 20
4*GE, with ZEDboard
2013 Dec



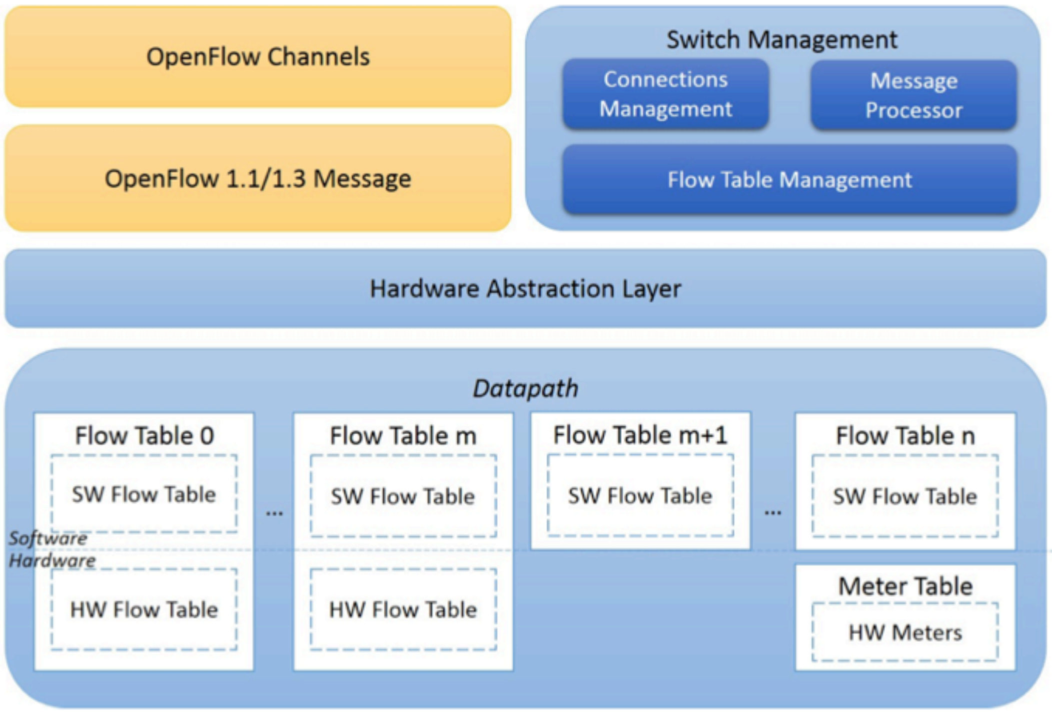
ONetSwitch 30
wifi/storage, 5*GE
2014 Dec.

Sponsored by Xilinx

http://onetswitch.org

ONetSwitch: All programmable SDN Switch

Chengchen Hu, Ji Yang, Hongbo Zhao, and Jiahua Lu. "Design of all programmable innovation platform for software defined networking". **Open Networking Summit (ONS) 2014**, Santa Clara, CA, US, 2014

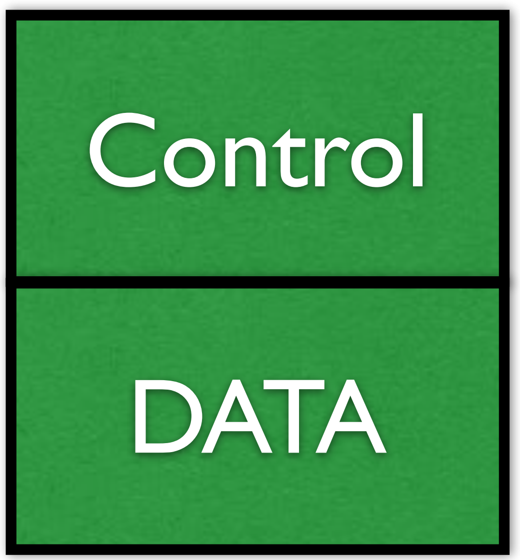


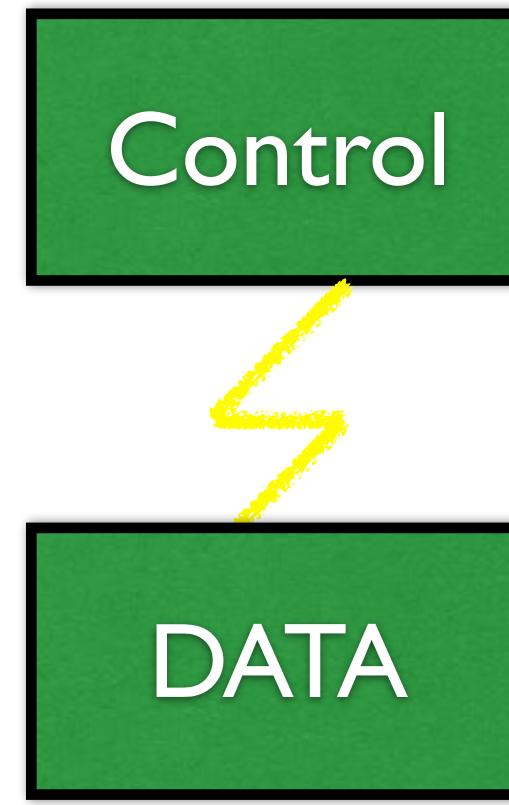
Chengchen Hu, Ji Yang, Zhimin Gong, Shuoling Deng, Hongbo Zhao. "DesktopDC: Setting All Programmable Data Center Networking Testbed on Desk", **Poster&Demo at SIGCOMM 2014**, Chicago, IL, US, 2014

Users

400+ ONetSwitches deployed in China, Europe, US

Case1: Frequent protocols

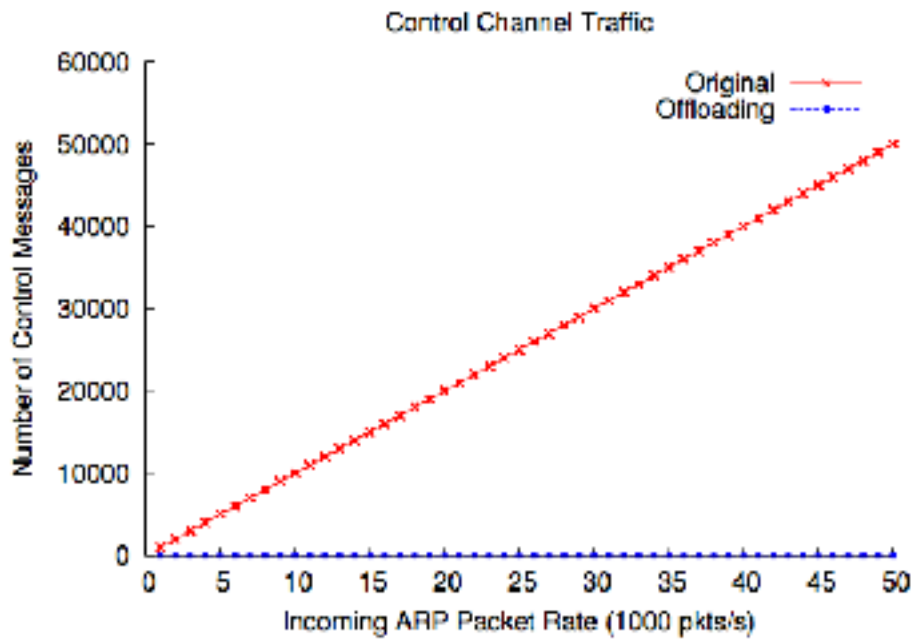


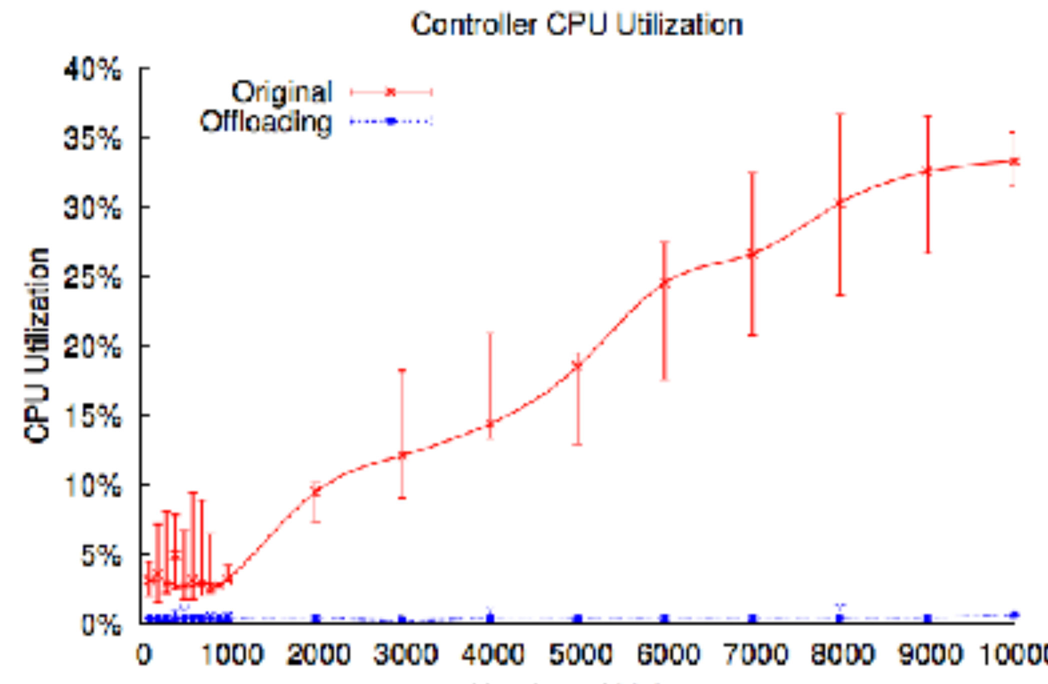


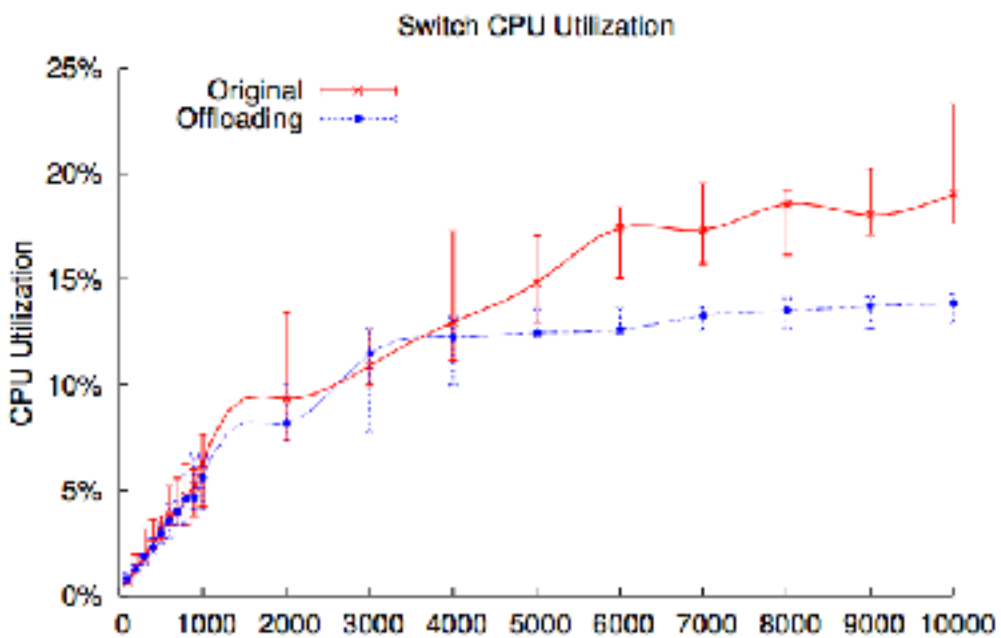
Case 1: ARP Gateway
- Physical Gateway usually doesn’t exist in SDN
- Controller reply to server end hosts
- Data plane queries controller repeatedly

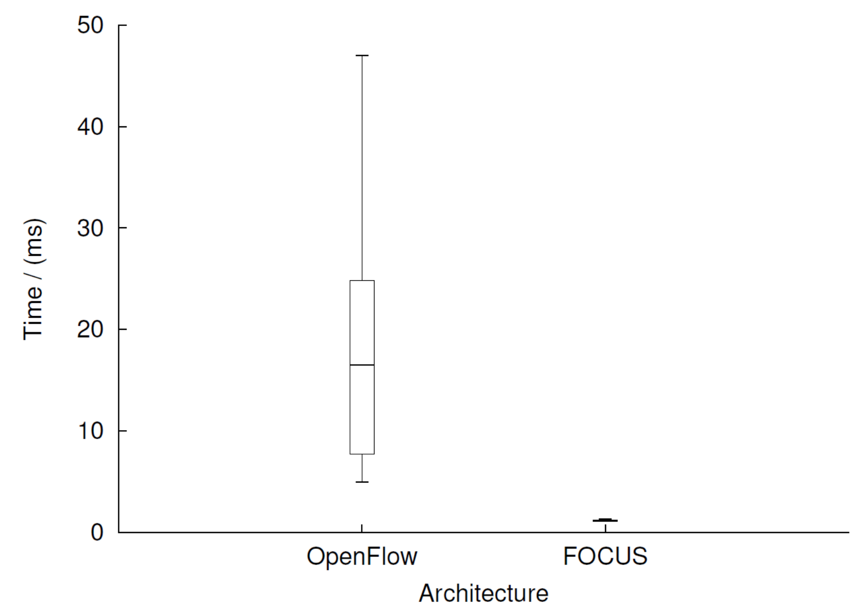
Case 2: LLDP (Link Layer Discovery Protocol)
- Get neighbor info. then keep watching
- Data plane packet goes to controller repeatedly even topology is stable
- Usually 2-5s per port per packet

Case 3: LACP (Link aggregation Control Protocol)
- State maintenance
- Repeat the same work again and again after topology is stable
- Usually 8 packets to converge
- Usually 1/30s per port per packet after convergence





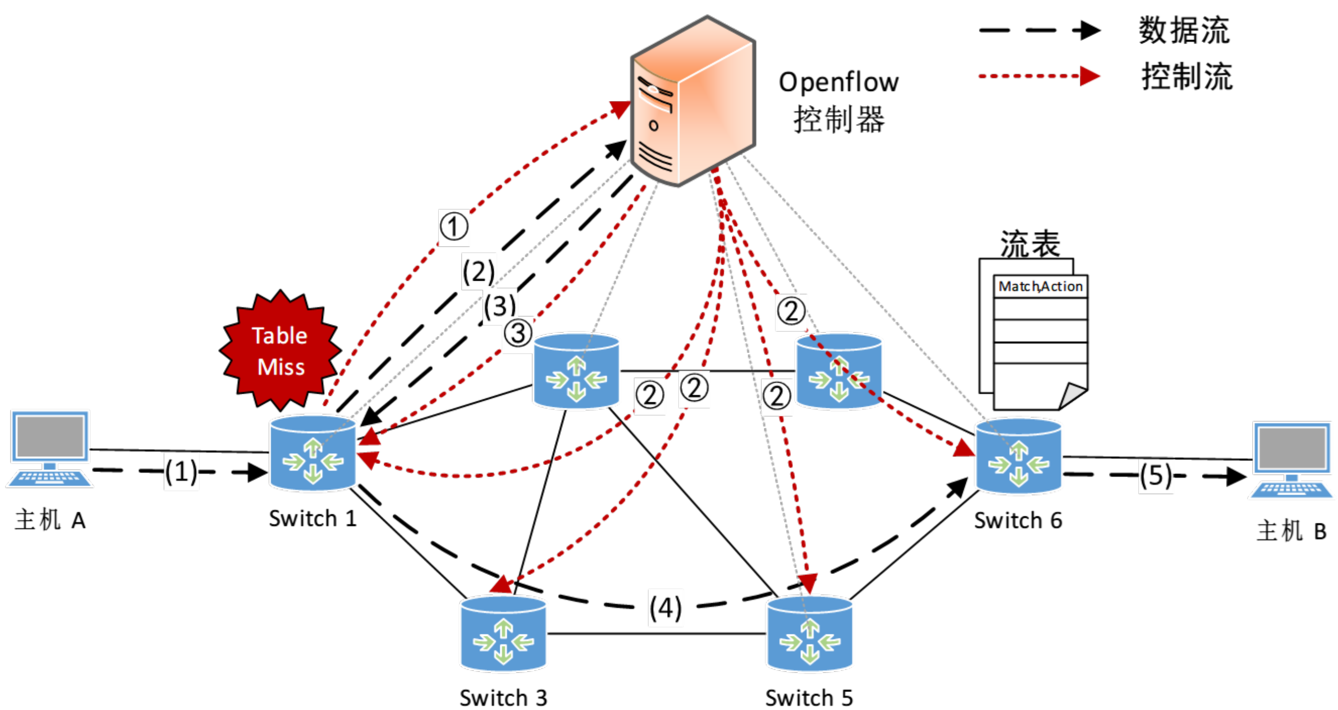


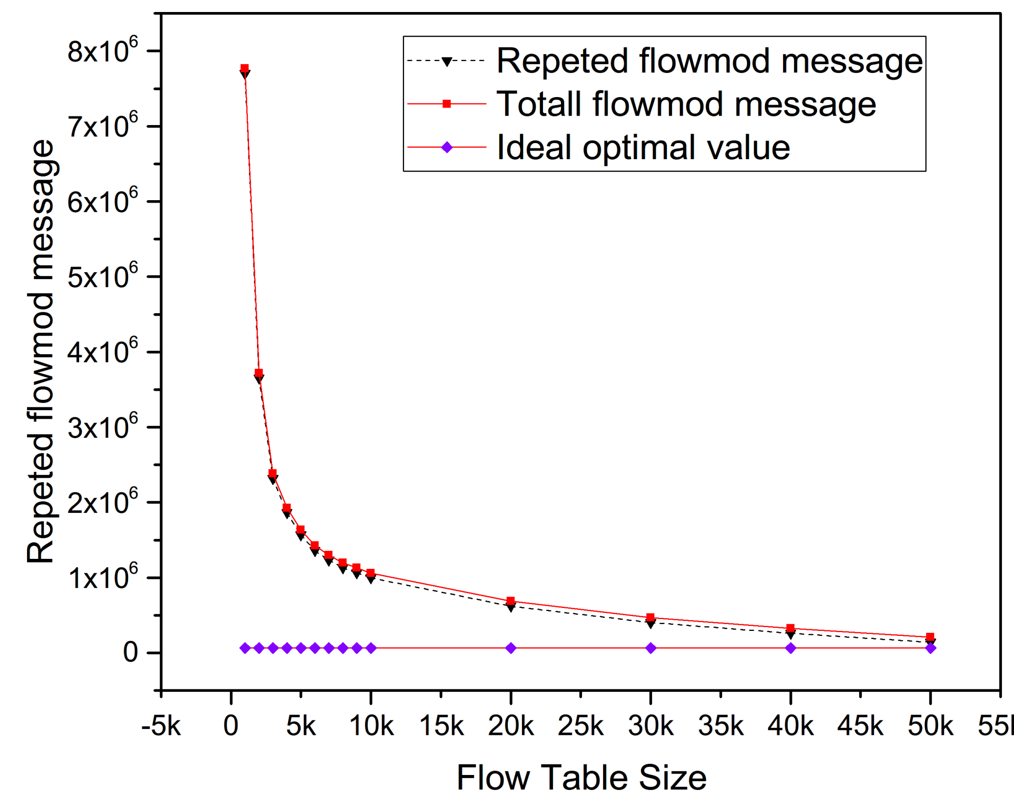


Communication overhead reduced: 50%-100%
Controller CPU work load reduced: 80%-98%
ARP response time: from 10+ms to us

FOCUS: Function Offloading from a Controller to Utilize Switch Power.
Poster @ NSDI'16 and @ SDN/NFV workshop'16

Case2: Table-miss

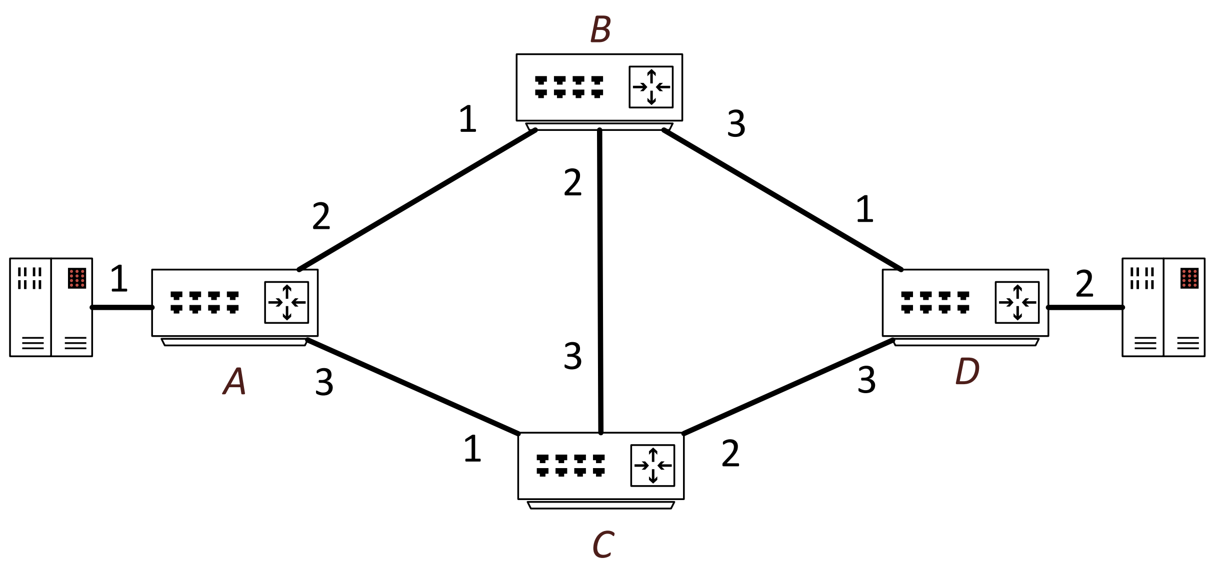




Repeated flowmod message: 68%-98% when flow table with 1k-50k entries.
Eat either fast path memory or bandwidth between switch and controller
Becomes bottleneck between Slow path and fast path in switch
Small flows make the problem worse.

CoSwitch: A Cooperative Switching Design for Software Defined Data Center Networking
@HotData, 2014 (best paper award)
Co-Work with IBM Research Lab
Taming the Flow Table Overflow in OpenFlow Switch. **Poster at SIGCOMM '16**.

Case 3: Rules Conflicts



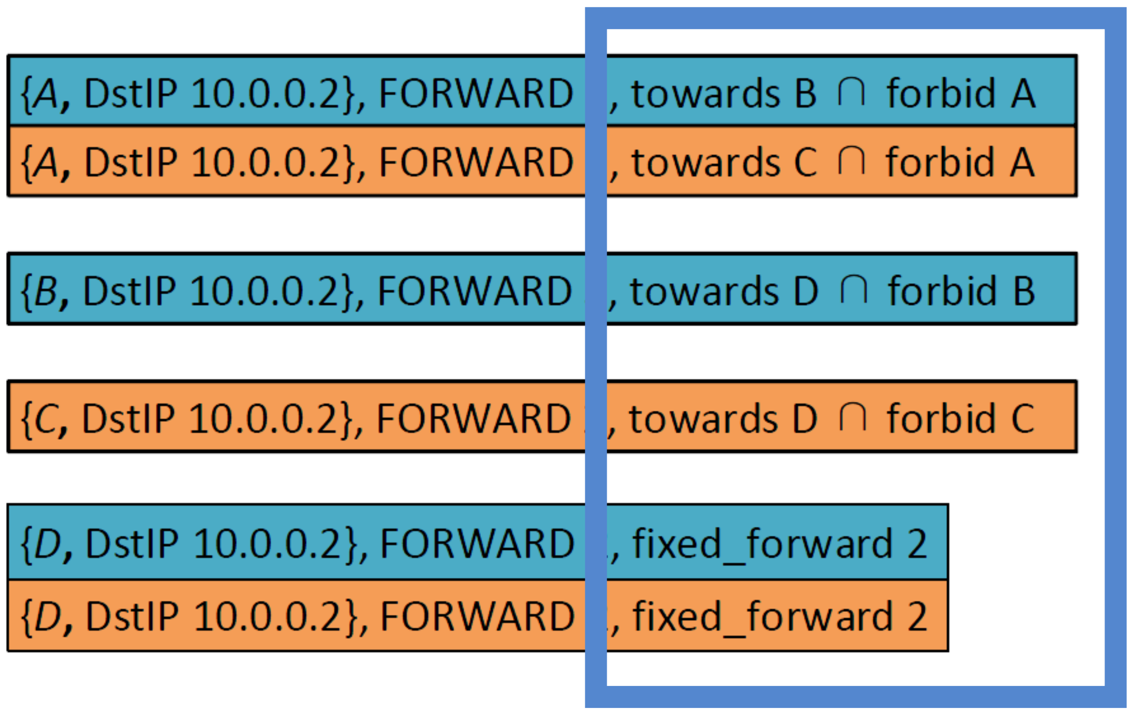
A	DstIP 10.0.0.2, FORWARD 2
B	DstIP 10.0.0.2, FORWARD 3
D	DstIP 10.0.0.2, FORWARD 2

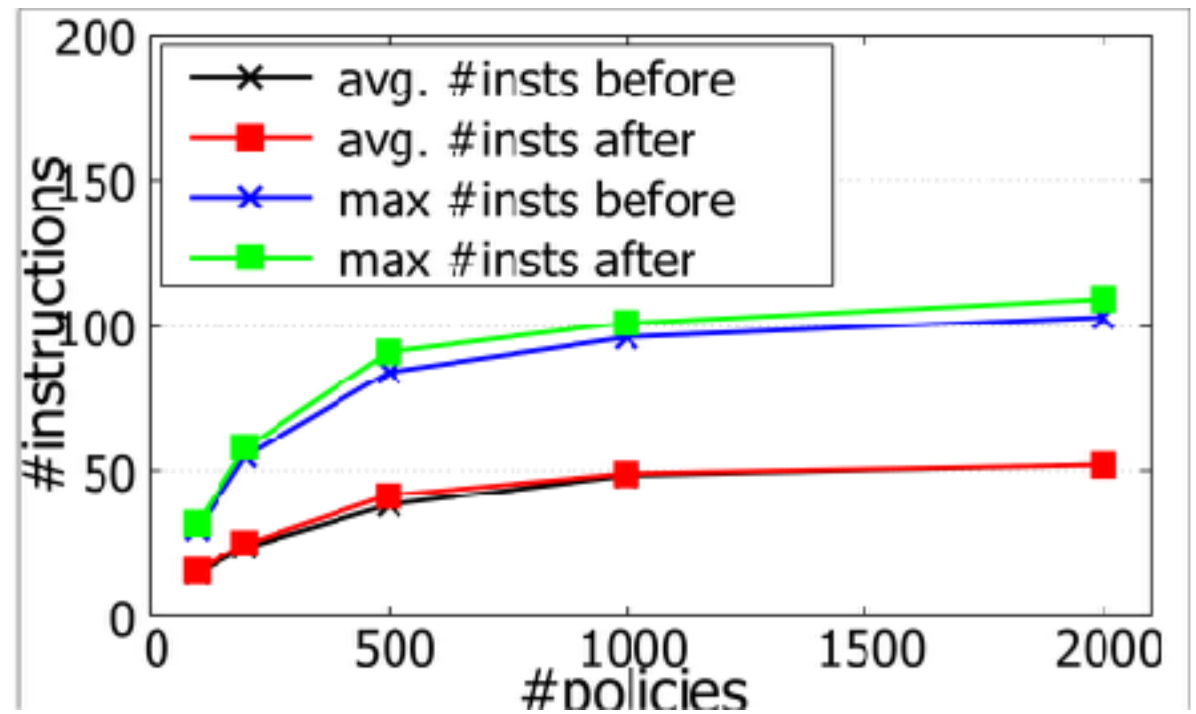
A	DstIP 10.0.0.2, FORWARD 3
C	DstIP 10.0.0.2, FORWARD 2
D	DstIP 10.0.0.2, FORWARD 2

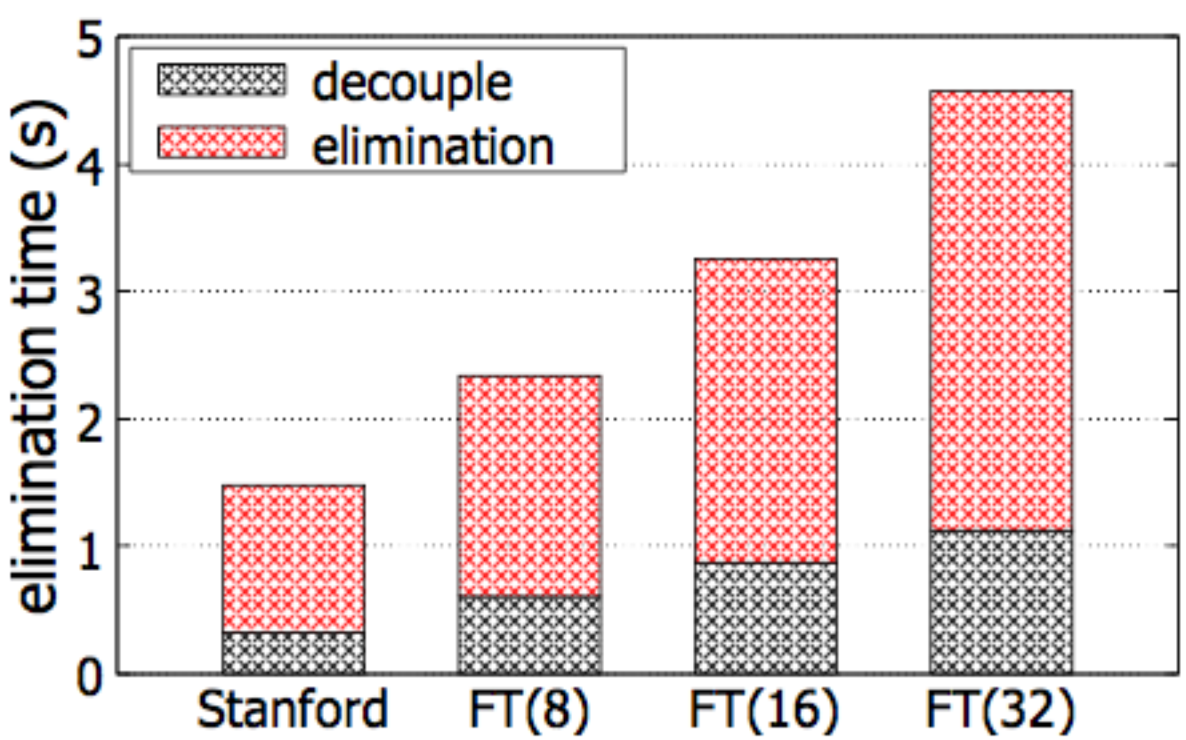
Hard to eliminate the conflicts without the high-level intents

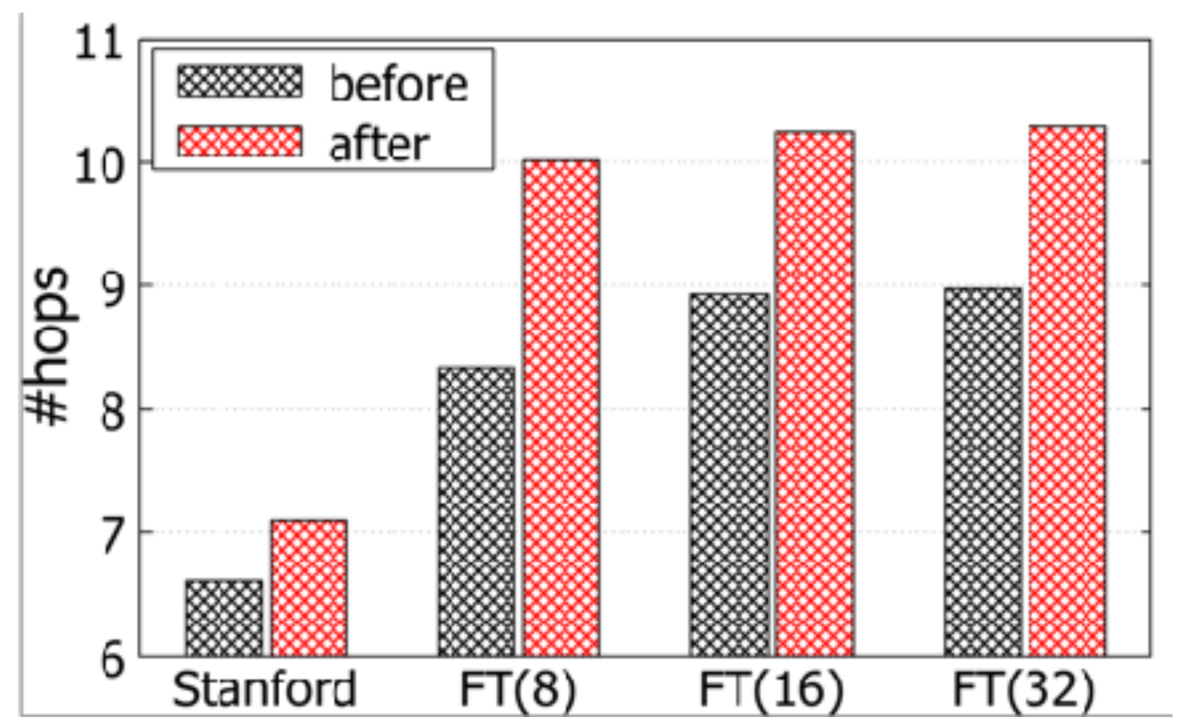
- blue and orange policies are only to connect A and D → random overwriting
- blue and orange policies critically specify the current path → unsolvable
- blue/orange policy is to connect A and with B/C in path → A to B to C to D

Modular SDN Compiler Design with Intermediate Representation. **poster @ SIGCOMM '16**.


{A, DstIP 10.0.0.2}, FORWARD , towards B ∩ forbid A
{A, DstIP 10.0.0.2}, FORWARD , towards C ∩ forbid A
{B, DstIP 10.0.0.2}, FORWARD , towards D ∩ forbid B
{C, DstIP 10.0.0.2}, FORWARD , towards D ∩ forbid C
{D, DstIP 10.0.0.2}, FORWARD , fixed_forward 2
{D, DstIP 10.0.0.2}, FORWARD , fixed_forward 2


#instructions
#policies
avg. #insts before
avg. #insts after
max #insts before
max #insts after
0
50
100
150
200
500
1000
1500
2000


elimination time (s)
decouple
elimination
0
1
2
3
4
5
Stanford
FT(8)
FT(16)
FT(32)


#hops
before
after
6
7
8
9
10
11
Stanford
FT(8)
FT(16)
FT(32)

Case 4: Rule update

- Flowtable update bottleneck
  - 10s to 100s of rule edits per second
  - Full refresh of 5K entries takes minutes

Pattern	Priority
<1, 2>	3
<*, 2>	2
<*, *>	1

Old

Pattern	Priority
<1, 2>	5
<2, *>	4
<1, *>	3
<*, 2>	3
<3, *>	2
<*, *>	1

New

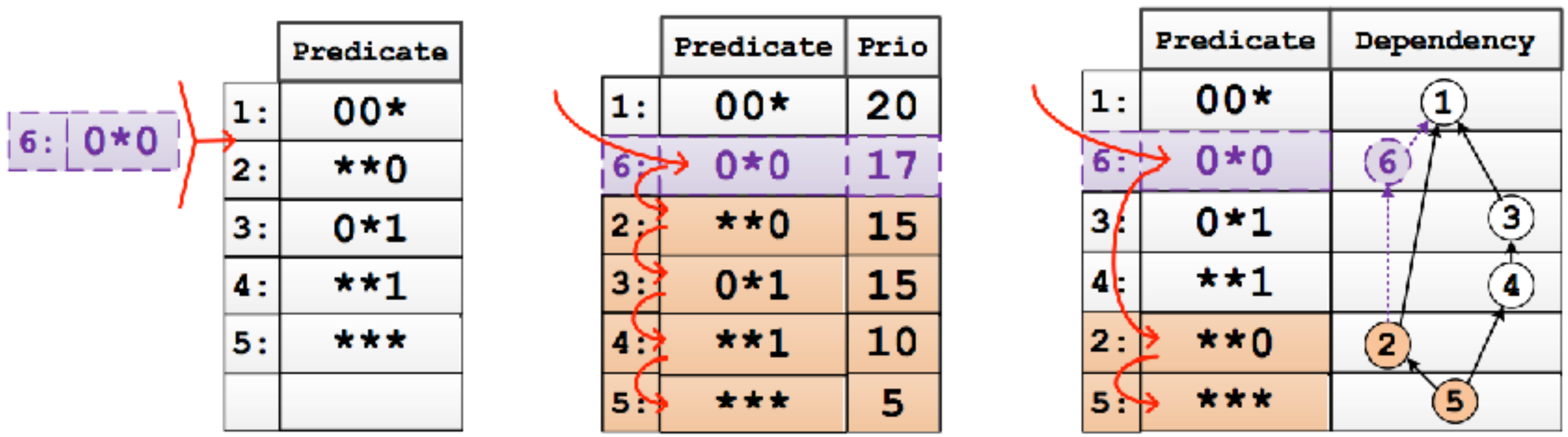
Priority Updates

3 rule adds + 2 priority updates

Unmodified fields

Modified fields

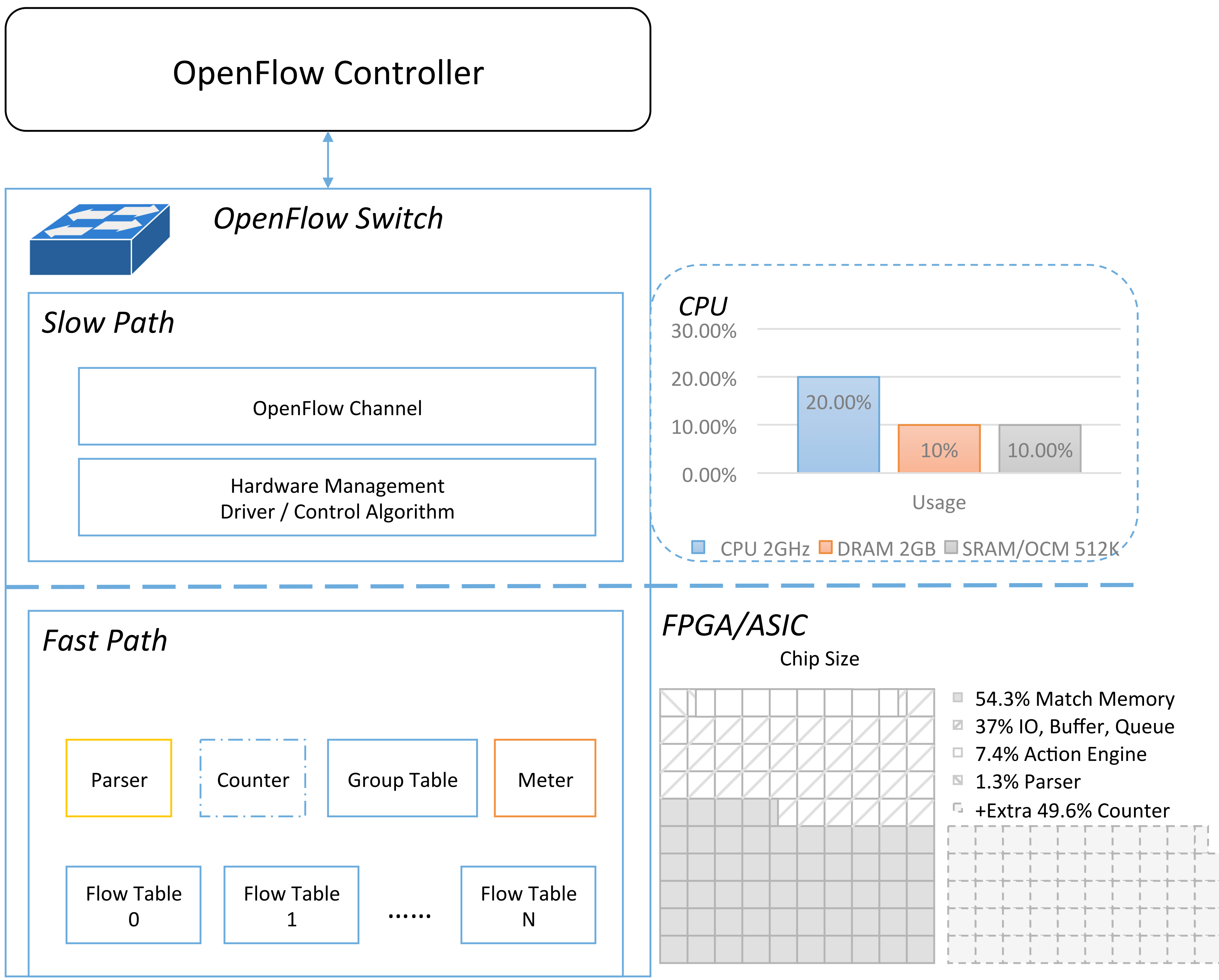
Try to minimize the update



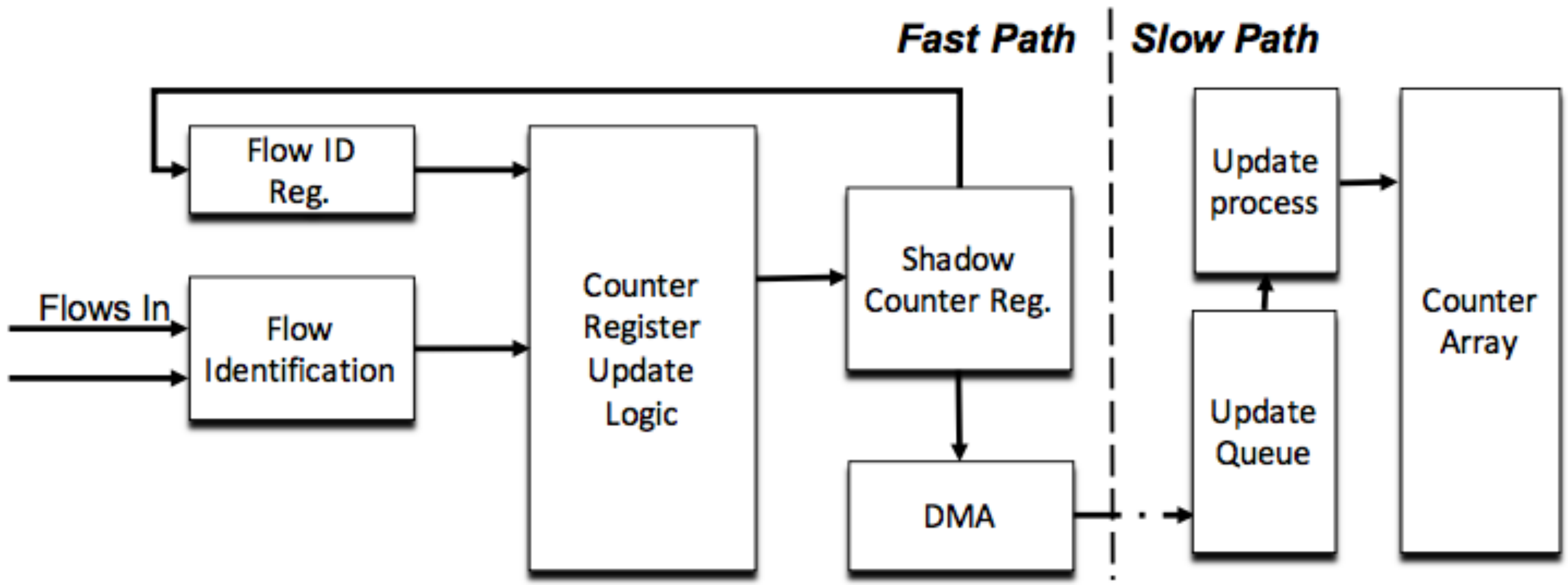
(a) An example of rule update. (b) Update with priority. (c) Update with dependency graph.

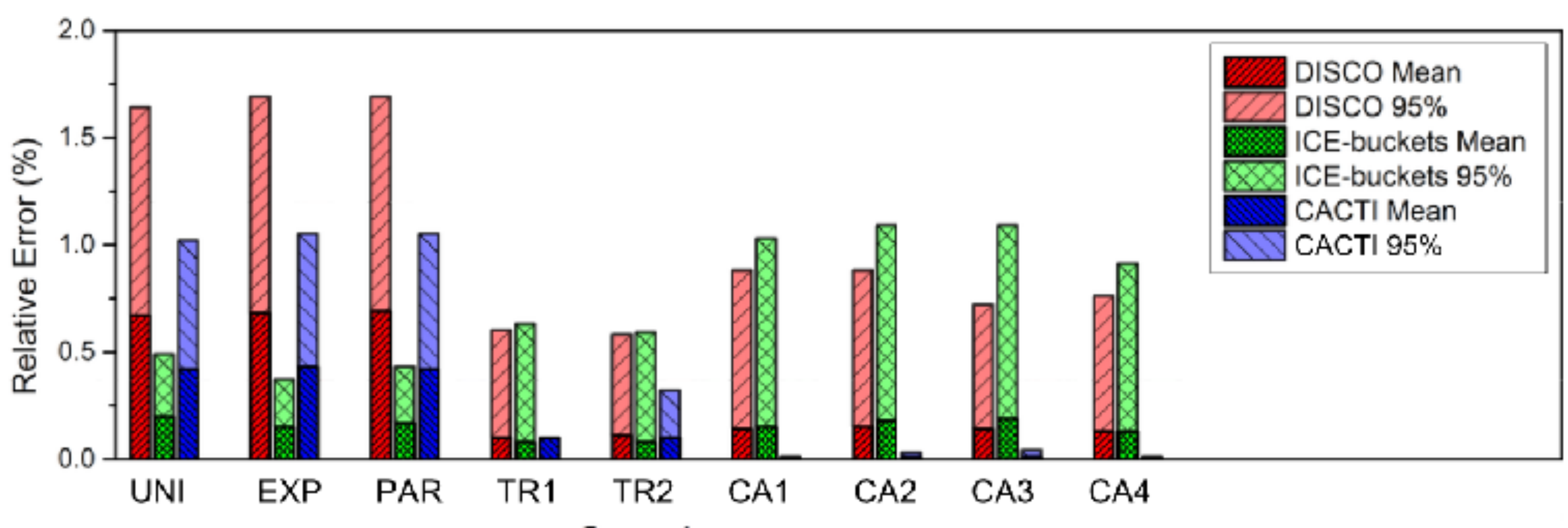
Xitao Wen, Bo Yang, Yan Chen, Li Erran Li, Kai Bu, Peng Zheng, Yang Yang, Chengchen Hu, RuleTris: Minimizing Rule Update Latency for TCAM-based SDN Switches, **ICDCS 2016**, Nara, Japan, June 27 – June 30, 2016.

Case 5: SDN Counters



CACTI: CAche CounTIng



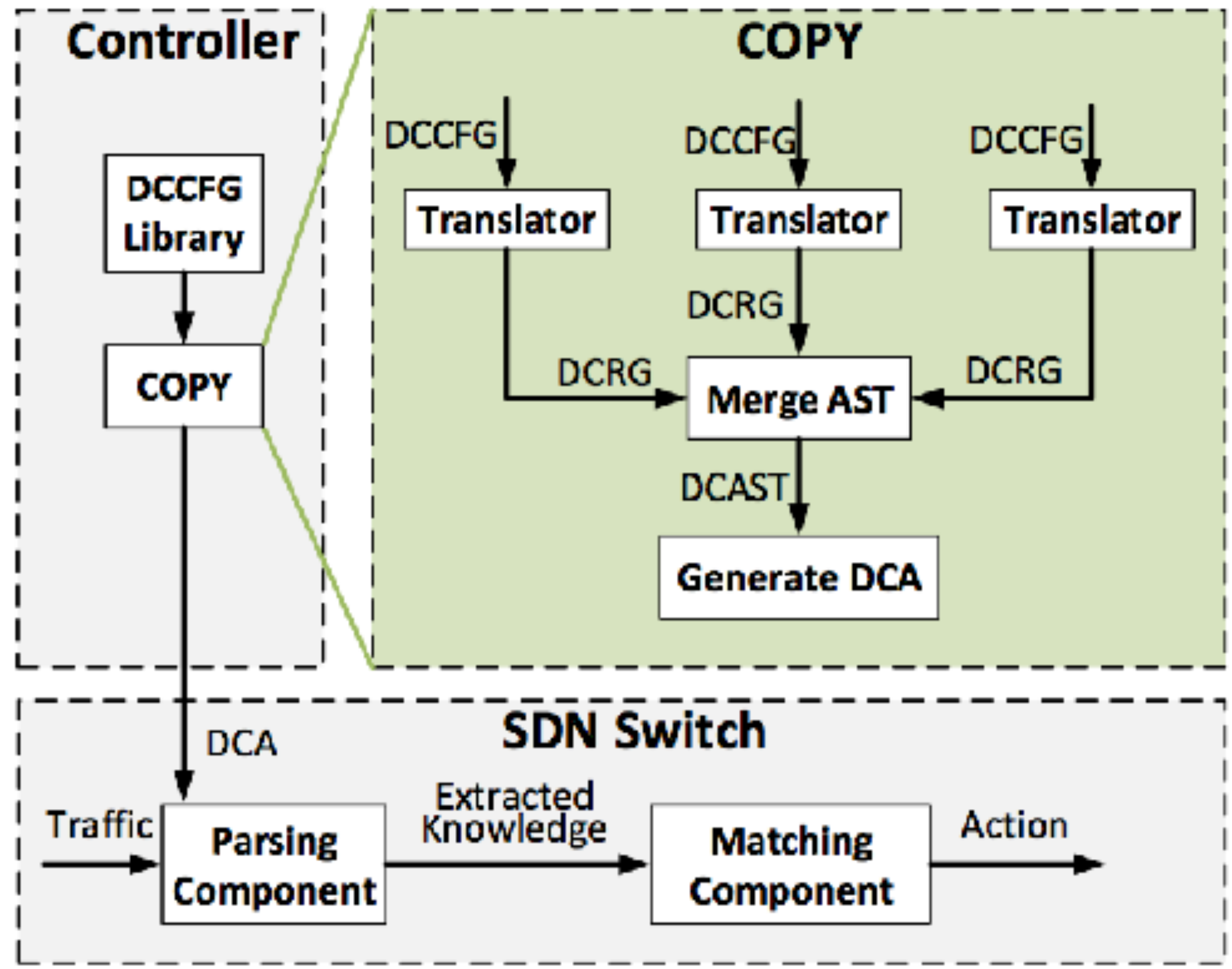


Case 6: Abstract L7

We propose a new specification form DCCFG, which can be formulated as a six-tuple, $\Gamma = (\mathbb{N}, \Sigma, \mathbb{C}, \mathbb{R}, S, \mathbb{E})$, where $\mathbb{N}$, Σ, $\mathbb{C}$, $\mathbb{R}$, S, $\mathbb{E}$ are the finite set of non-terminals, terminals, counters, production rules, start non-terminal, and extraction tokens, respectively. The non-terminals are the symbols where the terminals can be derived. The terminals can be a single character or a RegEx. The production rules can be described as <**guard**>:<**non-terminal**>→<**body**><**action**>.

1	S	→ T L V
2	T	→ <key, 0, “[A-Z]”>
3	L	→ “[0-9]” [len=getnum()]
4	[len>0]V	→ <value,0,“[A-Za-z]*”> [len=reduce()]
5	[len=0]V	→ ϵ

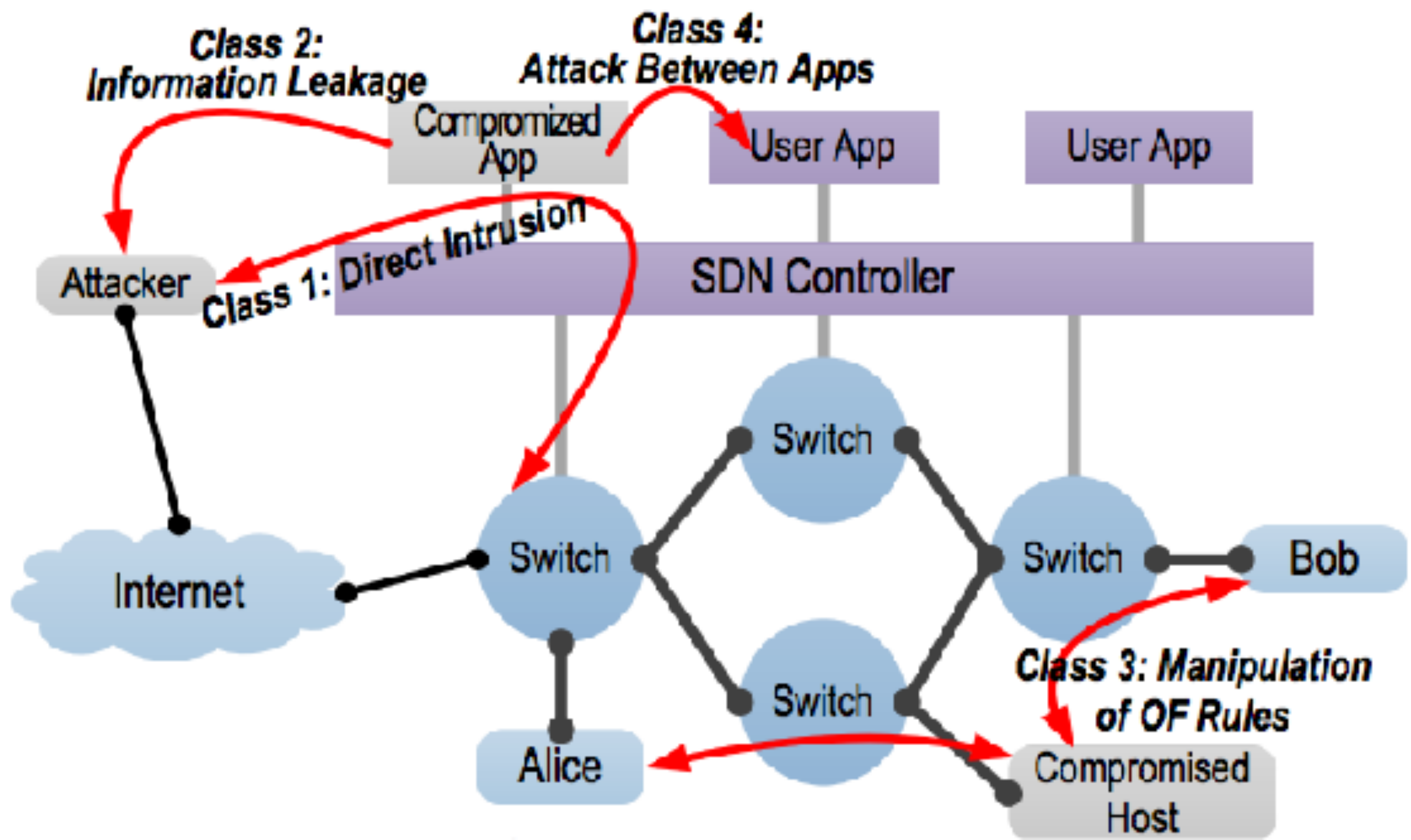
Figure 3: A TLV specification Γ in DCCFG.



Parsing Application Layer Protocol with Commodity Hardware for SDN@**ACM/IEEE ANCS 2015**

Case 7: Vulnerabilities

- Flow table attack
- Secure Channel attack
- Session Hijacking
- Policy bypass
- ….



On Denial of Service Attacks in Software Defined Networks @ **IEEE Network**
SDNShield: Reconciliating Configurable Application Permissions for SDN App Markets @ **DSN 2016**
Mind the Gap: Monitoring the Control-Data Plane Consistency in Software Defined Networks @ **CoNext 2016**

Open Questions

- How to make data plane programable ?
- Flow-action abstraction?
- Forwarding—Switch, Control—Controller?
- Fully Centralization?
- Anything between high level intents and low level rules?
- How to co-design Fast Path & Slow Path in Switch

Thank you