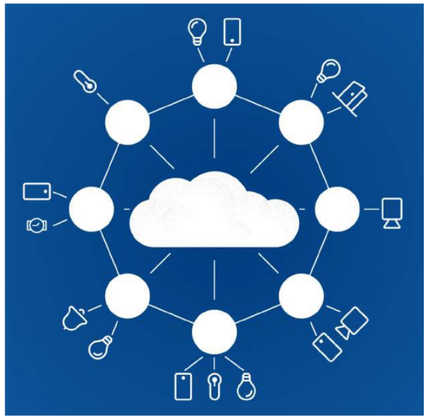
max planck institut informatik

Enabling System Innovation of Optical Data Center Networks with an Open Framework

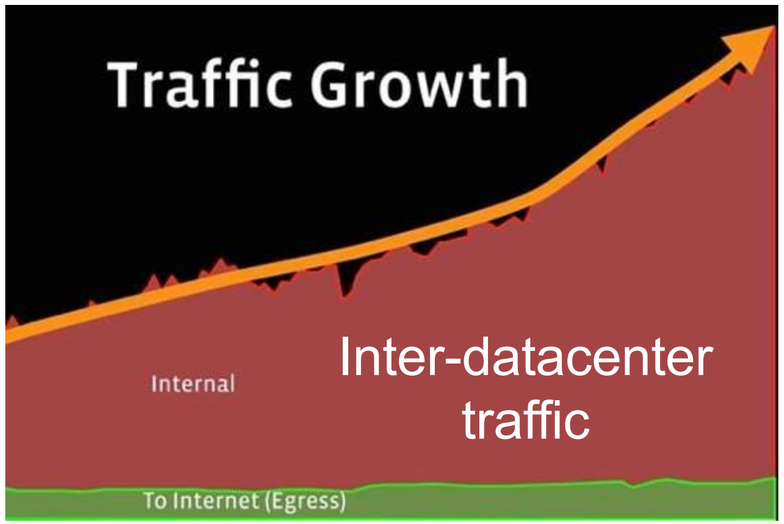
Yiming Lei*, Federico De Marchi*, Raj Joshi#, Jialong Li*, Balakrishnan Chandrasekaran+, **Yiting Xia***

Max Planck Institute for Informatics*, National University of Singapore#, Vrije Universiteit Amsterdam+

Ever Growing Cloud Traffic

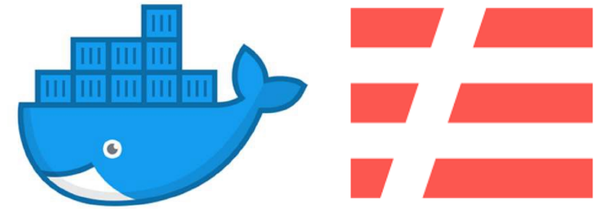


Cloudlet and IoT

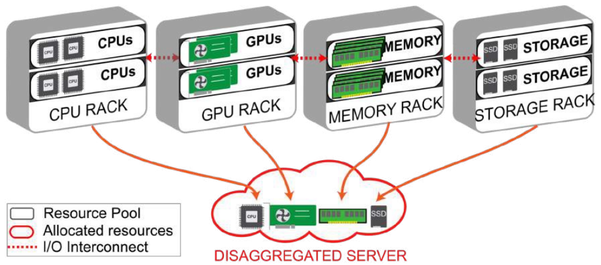


Cloud Data Center Traffic Growth

Source: [Facebook Network Statistics]



Container and Serverless



Disaggregated Data Centers

ML and AI

The Future Network

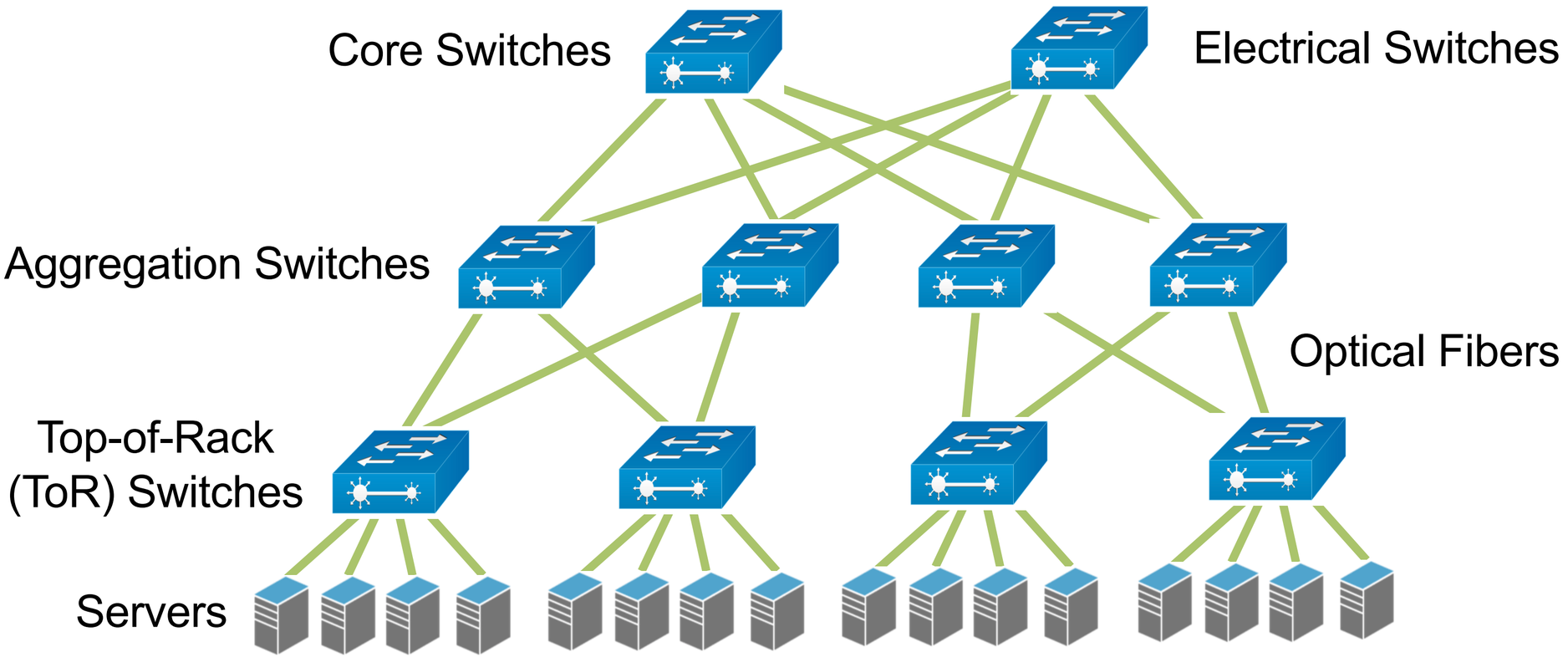
*Networking is the backbone of the **communications infrastructure**, and it is emerging like other civil infrastructures – water, electricity, transportation – in the way it evolves rather than being replaced, because of how it is built over generations, and how its high initial installation cost is followed by even higher ongoing maintenance expenses.*

– Henning Schulzrinne, SIGCOMM 2022 Keynote Speech

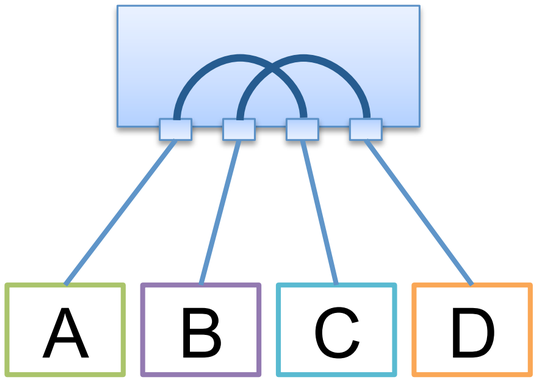
- Connectivity → performance → more
  - Accessibility (low cost)
  - Expandability (bandwidth upgrade)
  - Reliability and availability (24/7 on)
  - Environmental friendliness (low power consumption)

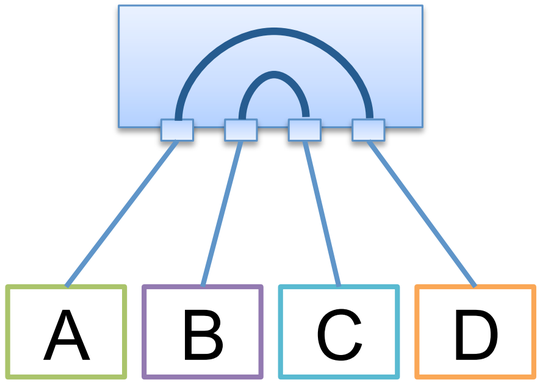
Moore’s Law for Electrical Switches

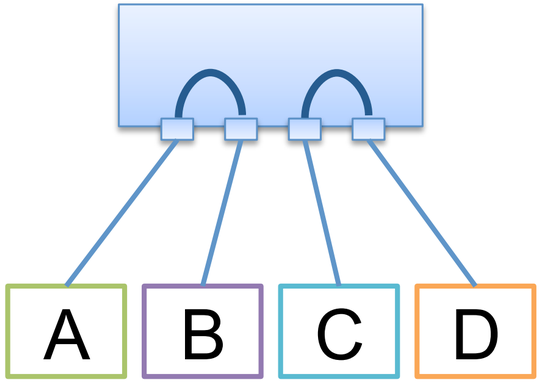
- Electrical switches double their bandwidth every two years at the same power and cost.



Optical Data Center Networks



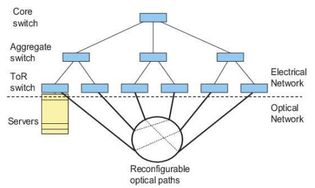




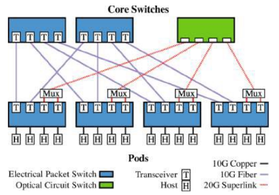
- Optical circuit switches
  - Dedicated links per configuration
  - An accurate transmission “schedule”
- Bufferless
  - No queuing delay
  - Need to buffer packets elsewhere

Optical Data Center Networks

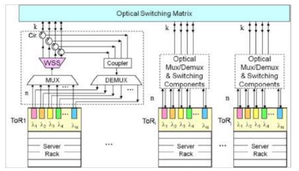
- Power efficient
  - Passive optical switches
- Low cost
  - Lower per-port cost than electrical switches
- Low latency
  - No queuing delay on optical switches
- Future proof
  - High bandwidth limit on optical switches
- Flexible network topology
  - Bring data and computation closer



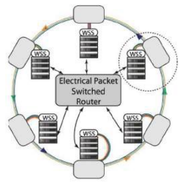
c-Through
[SIGCOMM 2010]



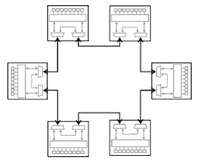
Helios
[SIGCOMM 2010]



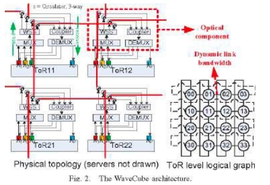
OSA
[NSDI 2012]



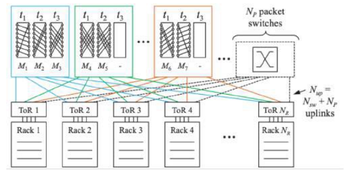
Mordia
[SIGCOMM 2013]



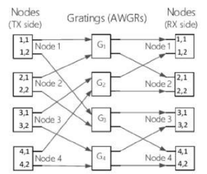
Quartz
[SIGCOMM 2014]



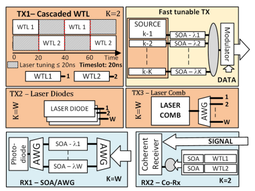
WaveCube
[INFOCOM 2015]



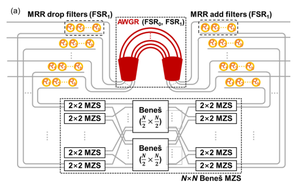
RoterNet
[SIGCOMM 2017]



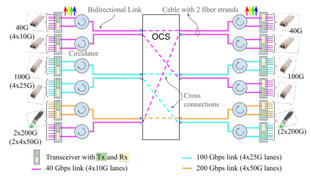
Sirius
[SIGCOMM 2020]



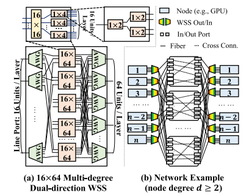
PULSE
[JoLT 2020]



Flex-LIONS
[JoLT 2021]



Jupiter
[SIGCOMM 2022]



Modoru
[JOCN 2024]

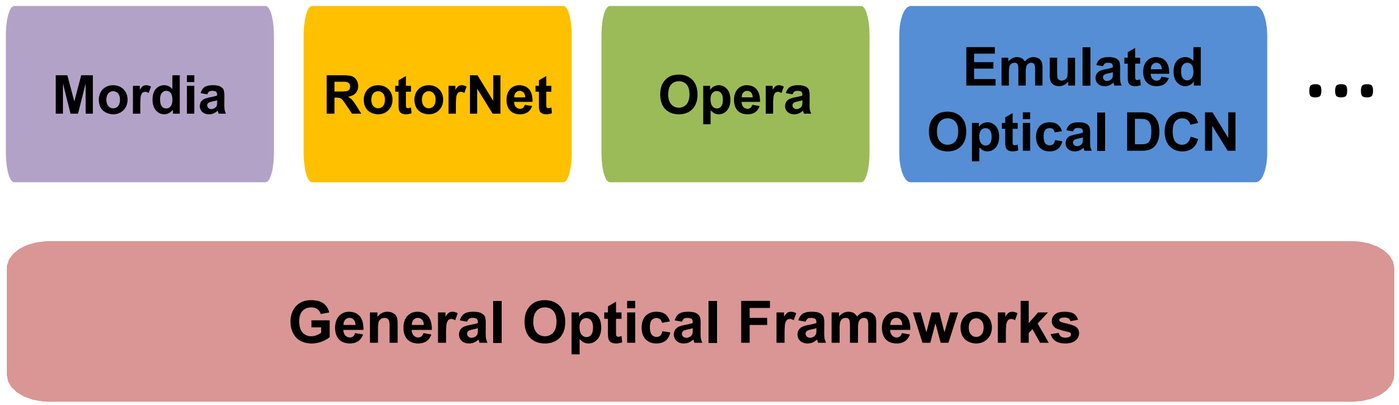
Optical Systems Lagging Behind Hardware

- Closed ecosystem
  - Specialized optical hardware + customized software stack
  - Home-grown simulation + small-scale testbed
- Siloed research and innovation
  - Software systems tied to the underlying hardware
  - Lack of a unified platform for implementation and evaluation

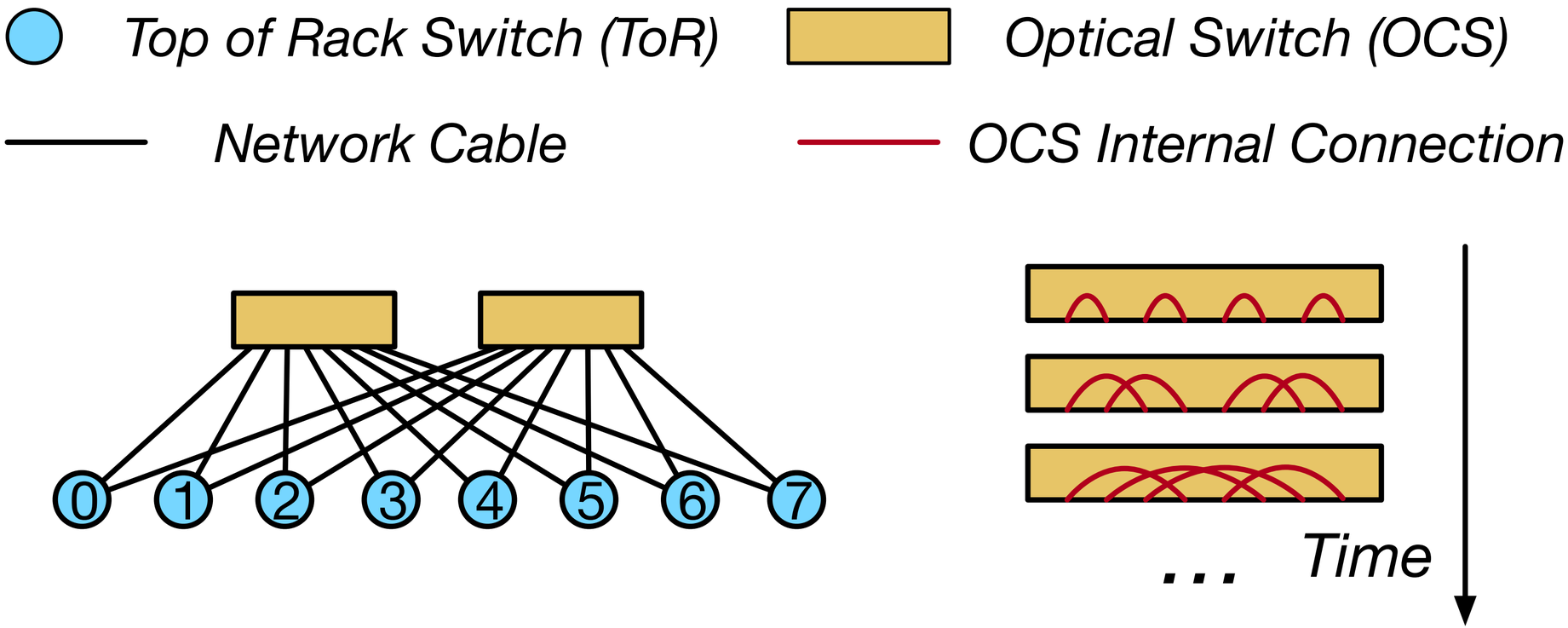
Decouple software from hardware → evolve independently

A General Framework

- OpenFlow for optical DCNs
  - A simple abstraction + user API
  - Transparent to cloud applications
  - Optical DCN architectures work in a plug-and-play manner
  - System intact as optical hardware evolves
  - To enable real-world testing, improvement, education, etc.



Generalized Network Architecture



(a) Optical Data Center Network (b) OCS Configuration

Time-Flow Table

- Time slice
  - Duration of an optical circuit
- Arrival time slice
  - When a packet enters the Top-of-Rack (ToR) switch
- Departure time slice
  - When a packet exits the ToR switch
- Backward compatible to flow tables
  - Wildcard arrival and departure slices

Arrival Slice	Source	Destination	Egress Port	Departure Slice
1	10.0.1.0/24	10.0.4.0/24	5	1

(a) *Forwarding in the current slice.*

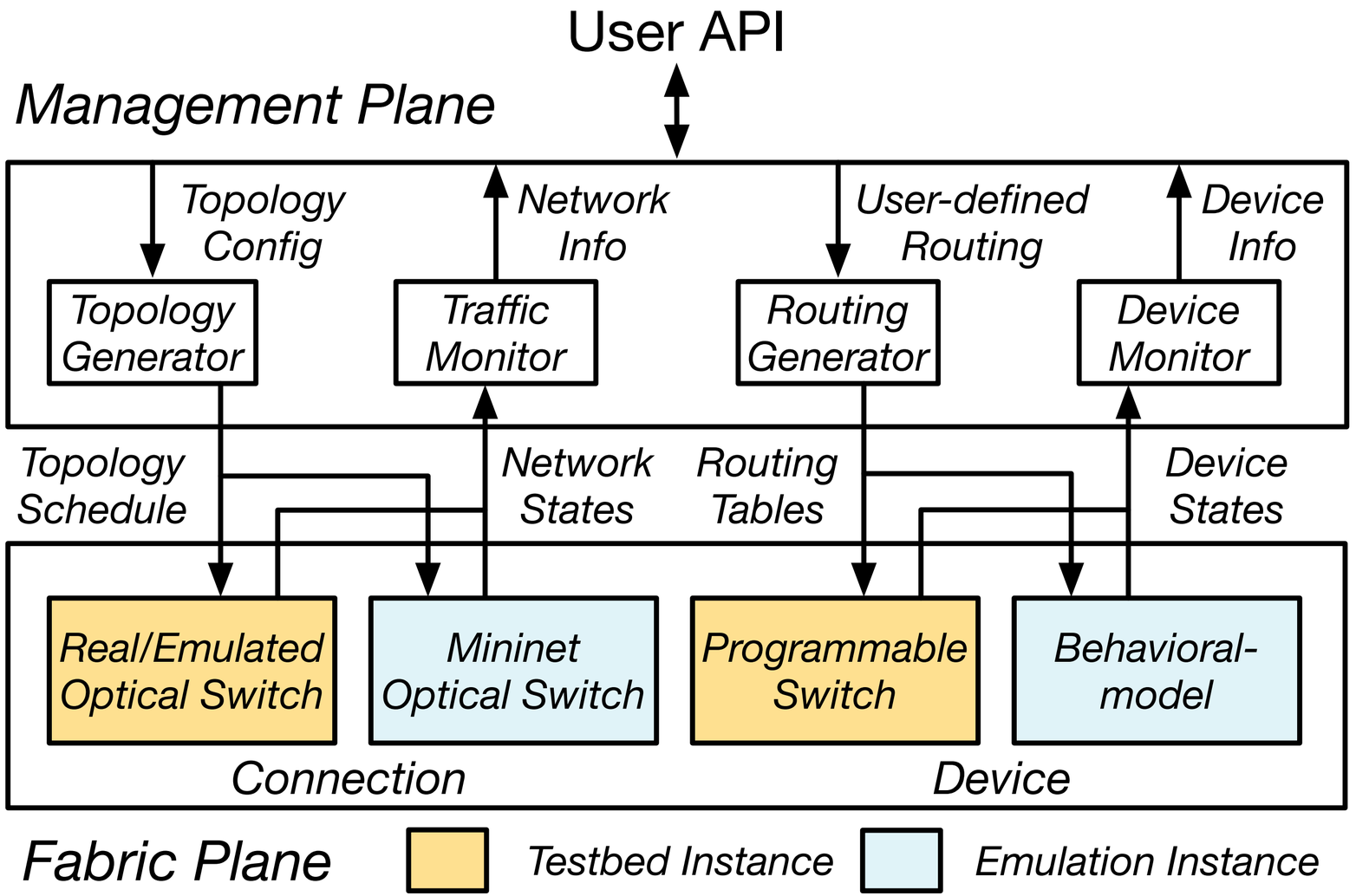
Arrival Slice	Source	Destination	Egress Port	Departure Slice
1	10.0.1.0/24	10.0.4.0/24	5	2

(b) *Forwarding in a future slice.*

Arrival Slice	Source	Destination	Egress Port	Departure Slice
*	10.0.1.0/24	10.0.4.0/24	5	*

(c) *Forwarding in a flow table.*

System Overview



User API

Category	APIs	
Topology	**connect**(ToR1, port1, ToR2, port2, slice)	
	round_robin(#ToRs, #ports)	
	round_robin_offset(#ToRs, #ports)	
	round_robin_dimension(#ToRs, #ports, h)	
Routing	**routing**(**routing_fn**)	**fn_direct**(src, dst, slice)
	neighbors(ToR, slice)	**fn_vlb**(src, dst, slice)
	earliest_path(src, dst, slice, max_hop)	**fn_ebs**(src, dst, slice)
		fn_opera(src, dst, slice)
	entries(paths, lookup_type, multipath_policy)	**fn_hoho**(src, dst, slice)
		fn_ucmp(src, dst, slice)
Monitoring	**buffer_usage**(ToR)	
	bandwidth_usage(ToR, port)	
	drop_rate(ToR)	

API Example

```
net = Network()
net.topology_random(tors=8, ports=2)
paths = net.routing(routing_fn = net.fn_vlb)
entries(paths, lookup_type=SOURCE,
        multipath_policy=RANDOM)
```

```
def topology_random(self, tor_num, port_num):
    slice_num = tor_num * port_num - 1
    for slice_id in range(slice_num):
        ports = list((tor,port) for tor in range(tor_num)
            for port in range(port_num))
        random.shuffle(ports)
        while ports: #ports not empty
            tor1, port1 = ports.pop()
            tor2, port2 = ports.pop()
            self.connect(tor1, port1, tor2, port2, slice_id)
```

```
def fn_vlb (src, dst, time_slice):
    paths = []; slice1 = time_slice
    for h1 in neighbors(src, slice1):
        (slice2, dst) = earliest_path(src=h1,
        dst=dst, time_slice=slice1, max_hop=1)
        paths.append([(slice1, h1),(slice2, dst)])
    return paths
```

Switch System

Paths (1) $\text{ToR}_1 \xrightarrow{t=1} \text{ToR}_2 \xrightarrow{t=2} \text{ToR}_4$ (2) $\text{ToR}_1 \xrightarrow{t=2} \text{ToR}_4$ (3) $\text{ToR}_1 \xrightarrow{t=4} \text{ToR}_3 \xrightarrow{t=5} \text{ToR}_4$

Switch System

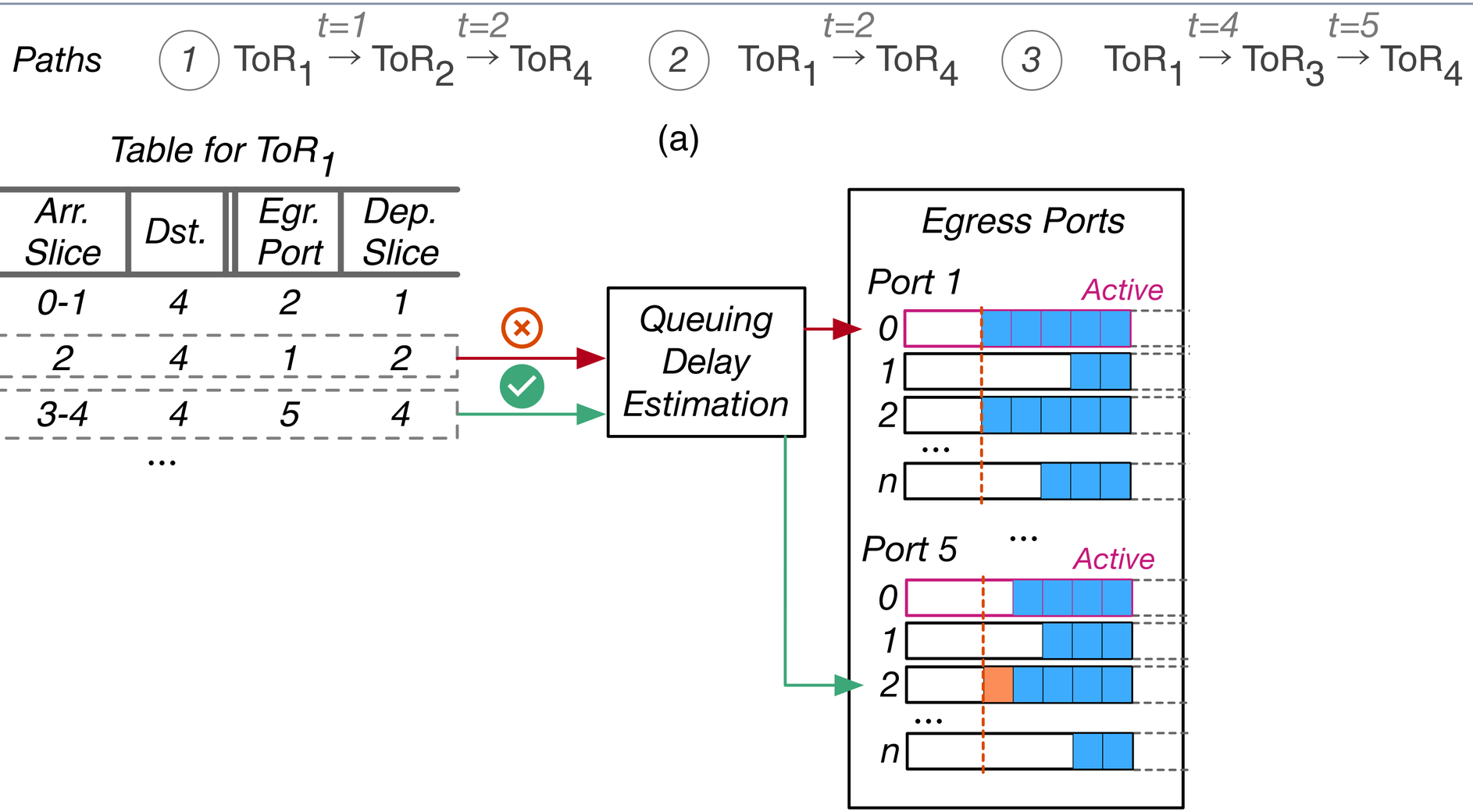
Paths (1) $\text{ToR}_1 \xrightarrow{t=1} \text{ToR}_2 \xrightarrow{t=2} \text{ToR}_4$ (2) $\text{ToR}_1 \xrightarrow{t=2} \text{ToR}_4$ (3) $\text{ToR}_1 \xrightarrow{t=4} \text{ToR}_3 \xrightarrow{t=5} \text{ToR}_4$

Table for ToR_1

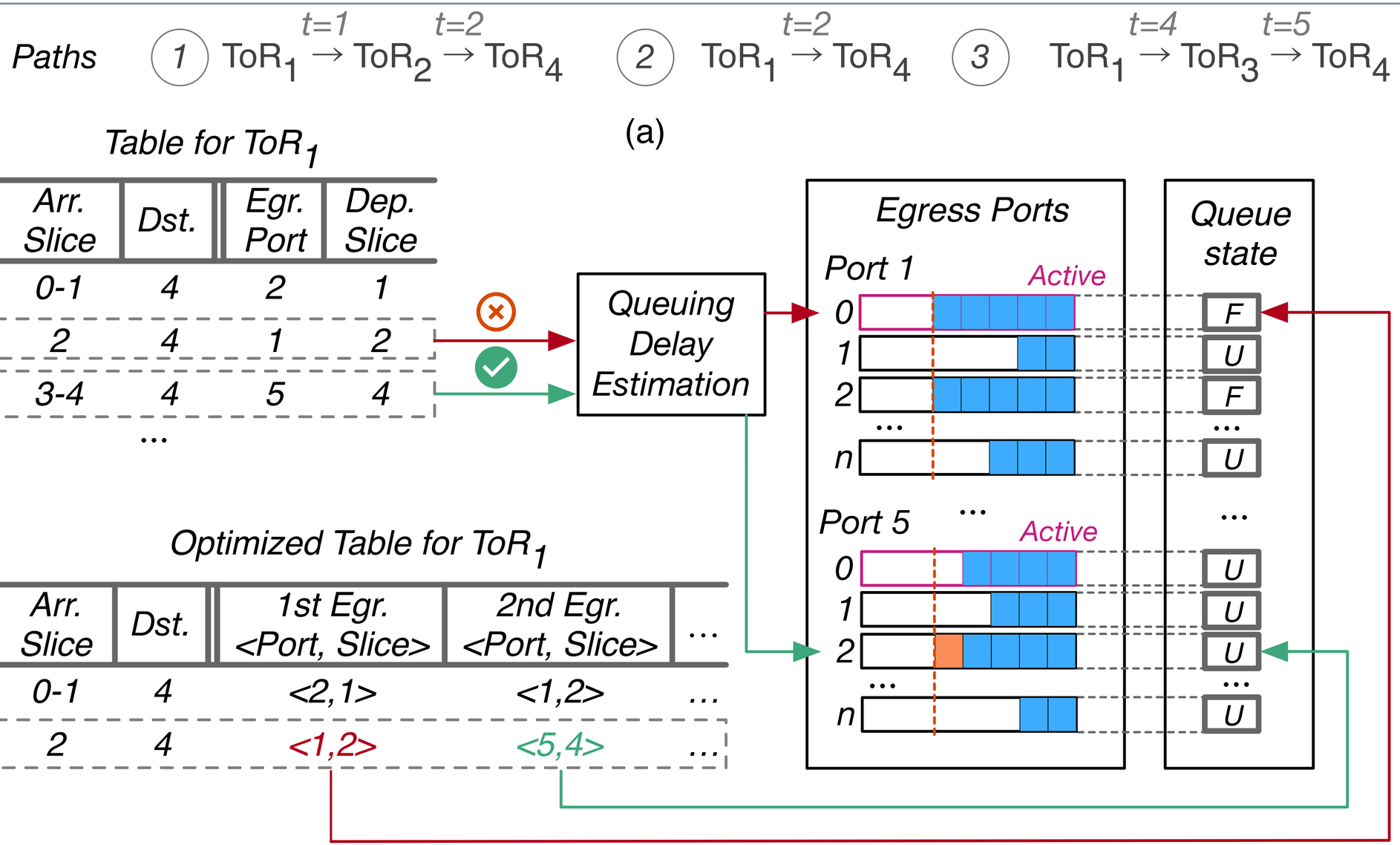
Arr. Slice	*Dst.*	*Egr. Port*	*Dep. Slice*
0-1	*4*	*2*	*1*
2	*4*	*1*	*2*
3-4	*4*	*5*	*4*

...

Switch System

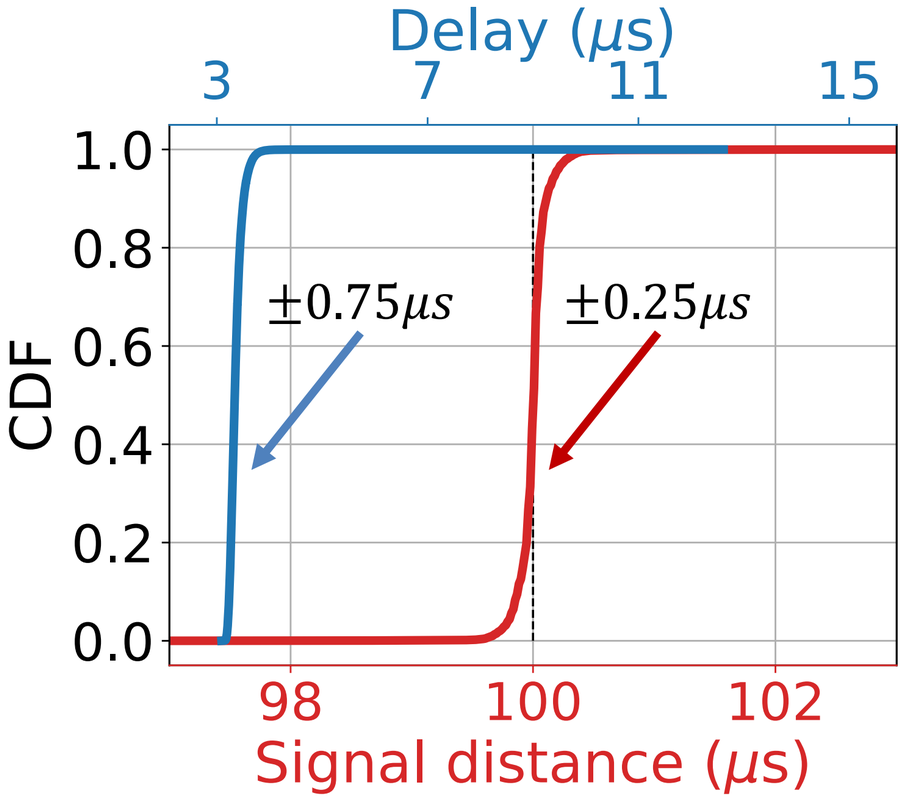


Switch System

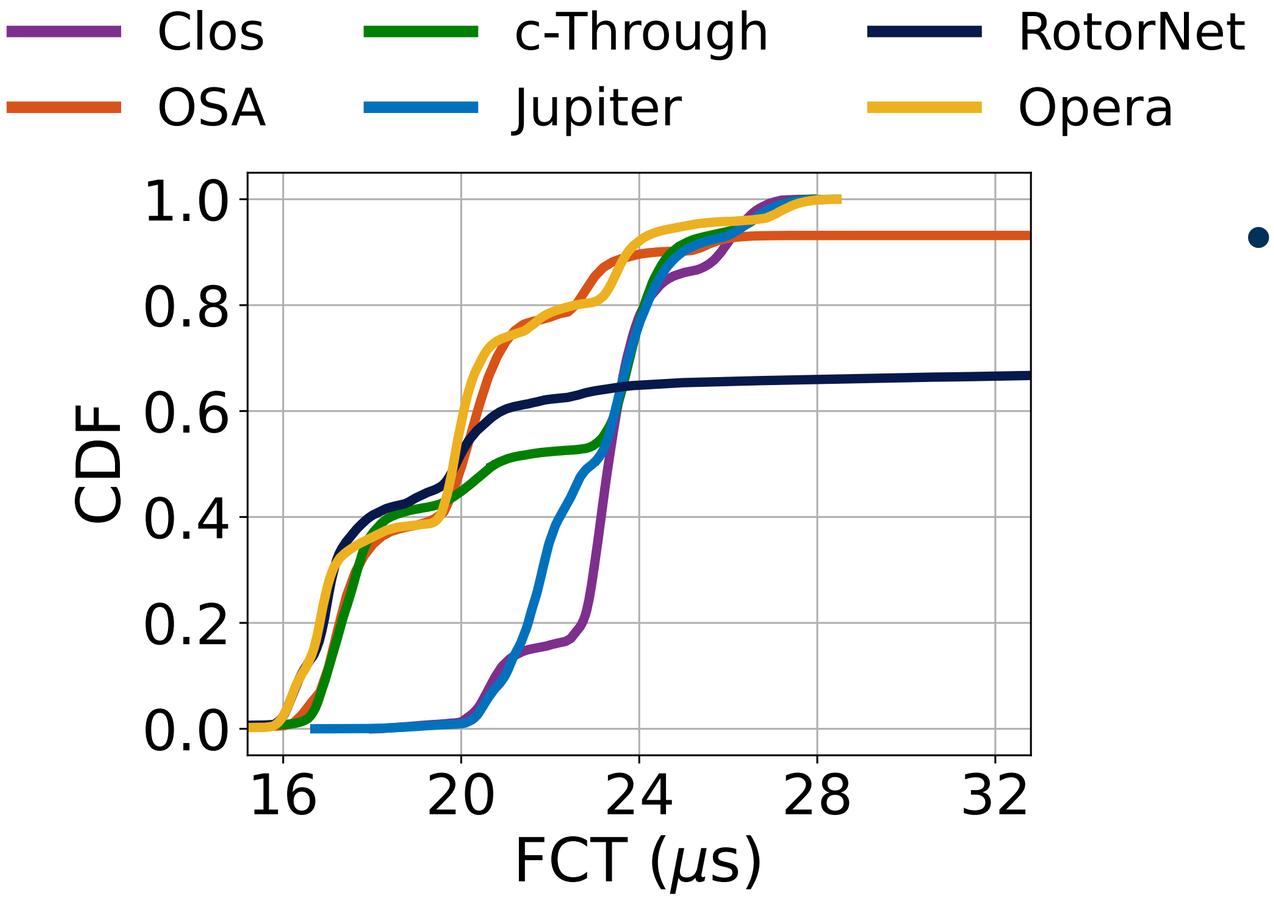


Host System

- VMA-based
  - LWIP TCP/UDP stack
- Signal between host and ToR
  - Reliable delay
  - Orders of magnitude lower accuracy than ToR
- Flow pausing
  - Pause elephant flows with flow aging
- Backpressure for circuit flow control
  - What if packet does not fit into a time slot?
  - Rerouting is not enough, ask source to stop sending

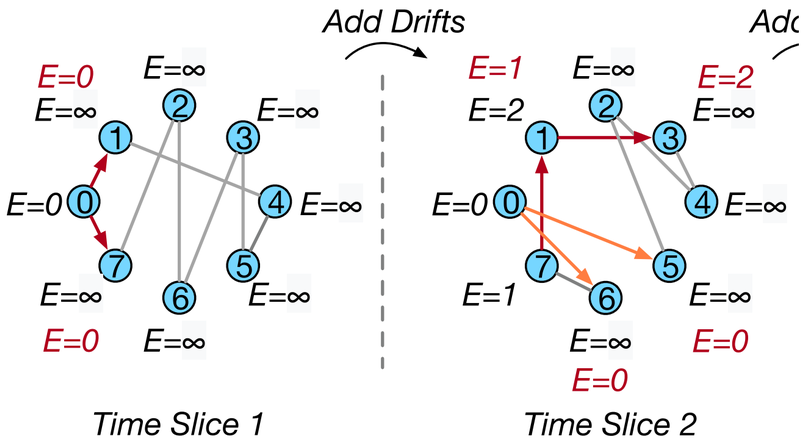


Across-Architecture Performance

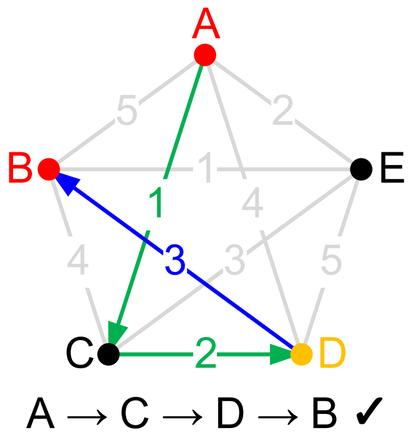


- Memcached traffic
  - 8 ToRs, each with 4 100Gbps uplinks
  - 8 hosts, each under a ToR
  - 2 servers 6 clients each on a host
  - Different architectures and their native routing

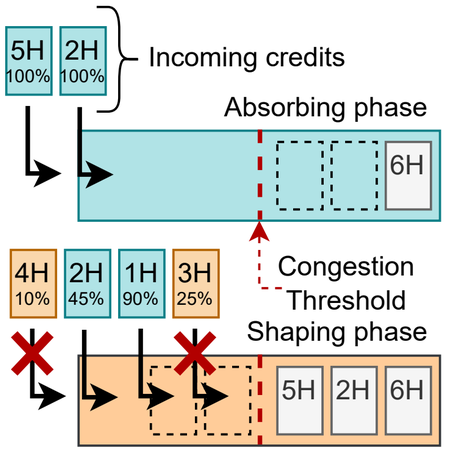
A Research Vehicle



Time synchronization



Optical routing [1]

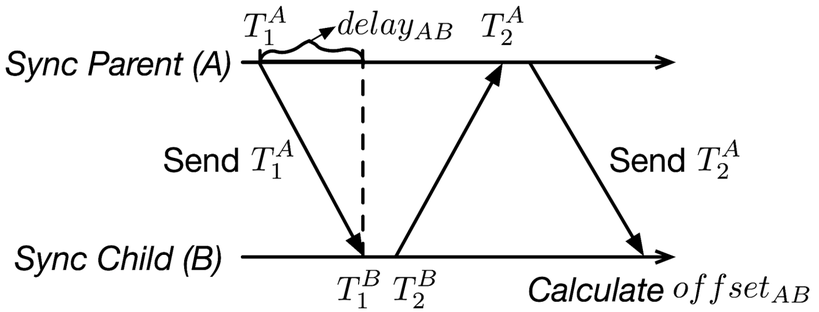


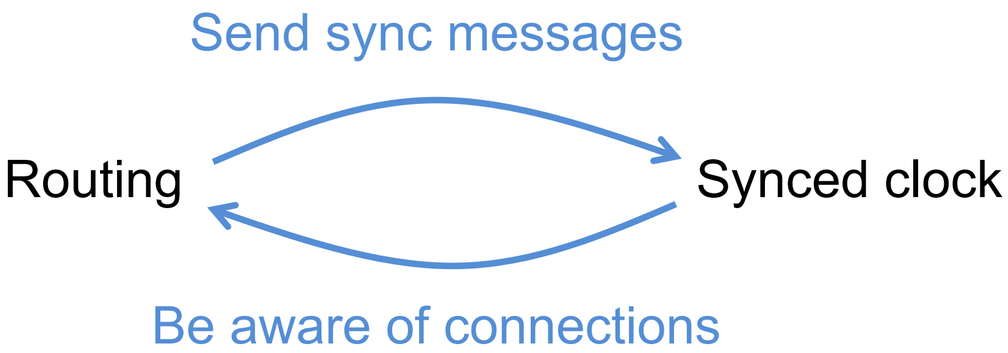
Optical transport [2]

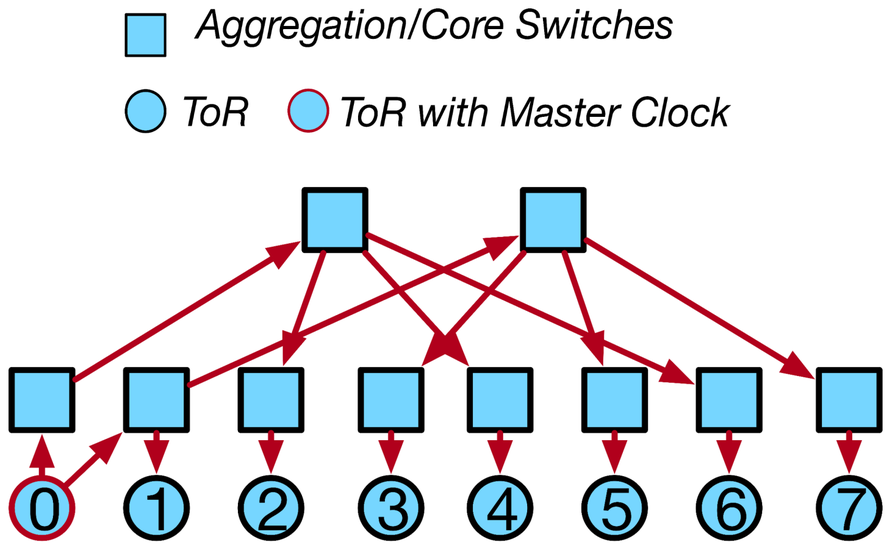
[1] Li et al., *Uniform-Cost Multi-Path Routing for Reconfigurable Data Center Networks*, SIGCOMM 2024.
[2] De Marchi et at., *Opportunistic Credit-Based Transport for Reconfigurable Data Center Networks with Tidal*, Poster, SIGCOMM 2024

Time Synchronization

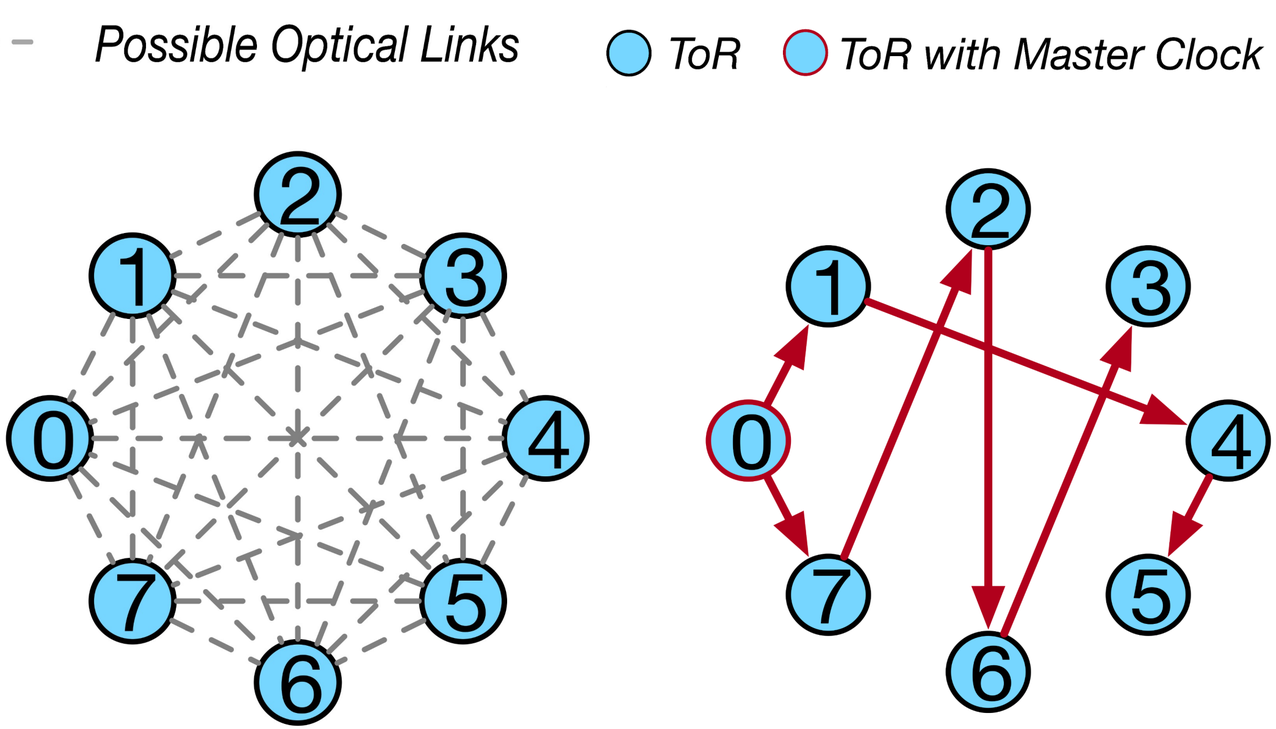
- Nanosecond accuracy
  - (Sub)microsecond-scale time slices
  - Only send when circuits ON
- Sync protocols for static networks don't work
  - Assume reliable routing
  - Unaware of optical circuits







Bootstrap Clocks



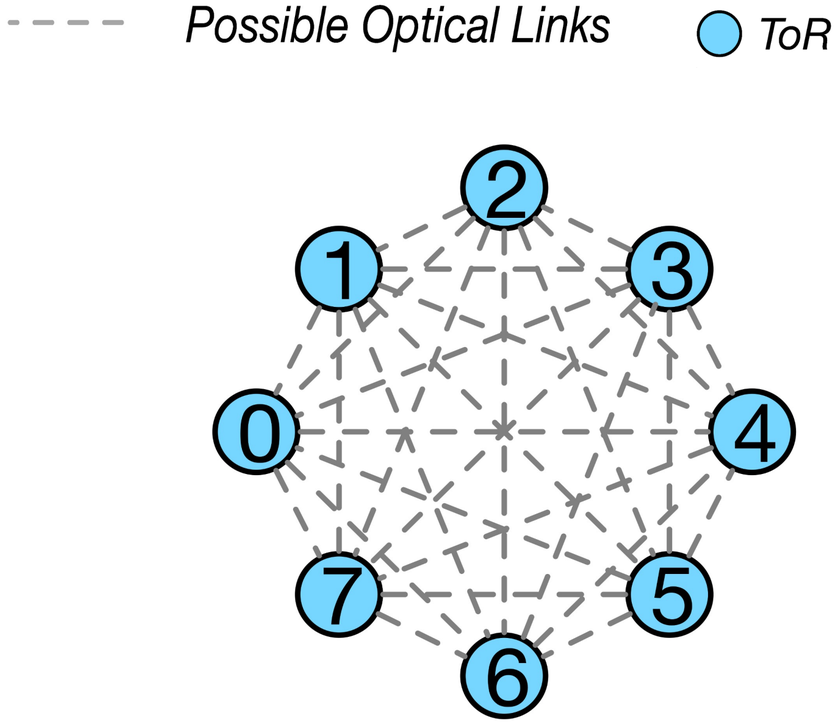
Send sync messages

Routing ⇄ Synced clock ✅

Be aware of connections

Drift-Aware Synchronization

- Drift awareness for accuracy
  - Drifts dominate sync errors
  - Different devices have different drifts [3]
  - Median stable, with high variance [3]
- Profile drifts
  - Connect each ToR to master
  - Calculate median of N drift measurements



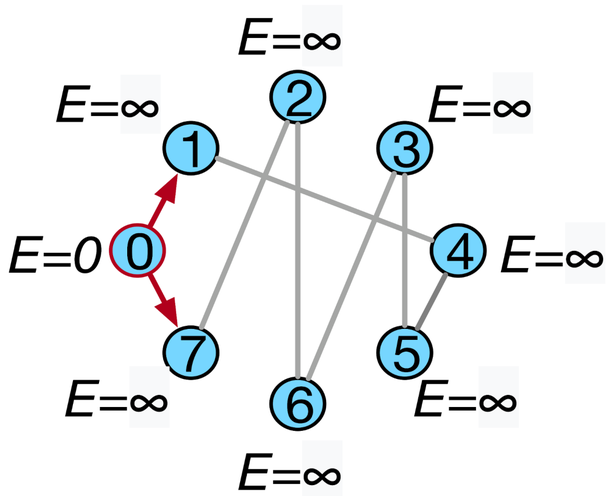
[3] Najafi et al., *Graham: Synchronizing clocks by leveraging local clock properties*, NSDI 2022

Drift-Aware Synchronization

ToR ID	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
Drift (ns/slice)	*0*	*2*	*1*	*2*	*-1*	*1*	*1*	*1*

Drift-Aware Synchronization

ToR ID	0	1	2	3	4	5	6	7
Drift (ns/slice)	0	2	1	2	-1	1	1	1

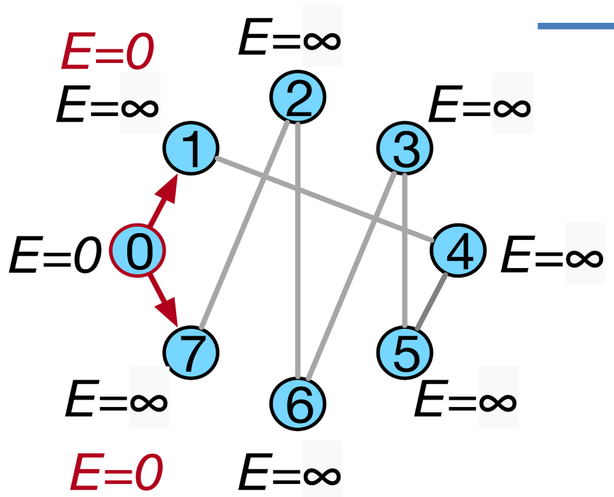


Time slice 1

Drift-Aware Synchronization

ToR ID	0	1	2	3	4	5	6	7
Drift (ns/slice)	0	2	1	2	-1	1	1	1

Add drifts

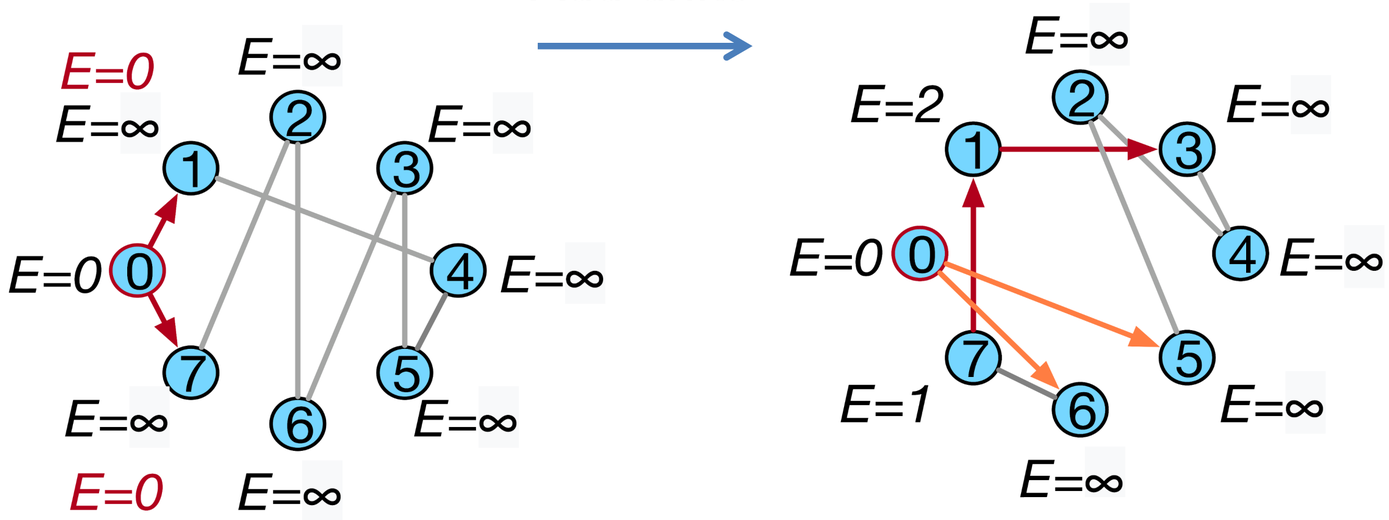


Time slice 1

Drift-Aware Synchronization

ToR ID	0	1	2	3	4	5	6	7
Drift (ns/slice)	0	2	1	2	-1	1	1	1

Add drifts

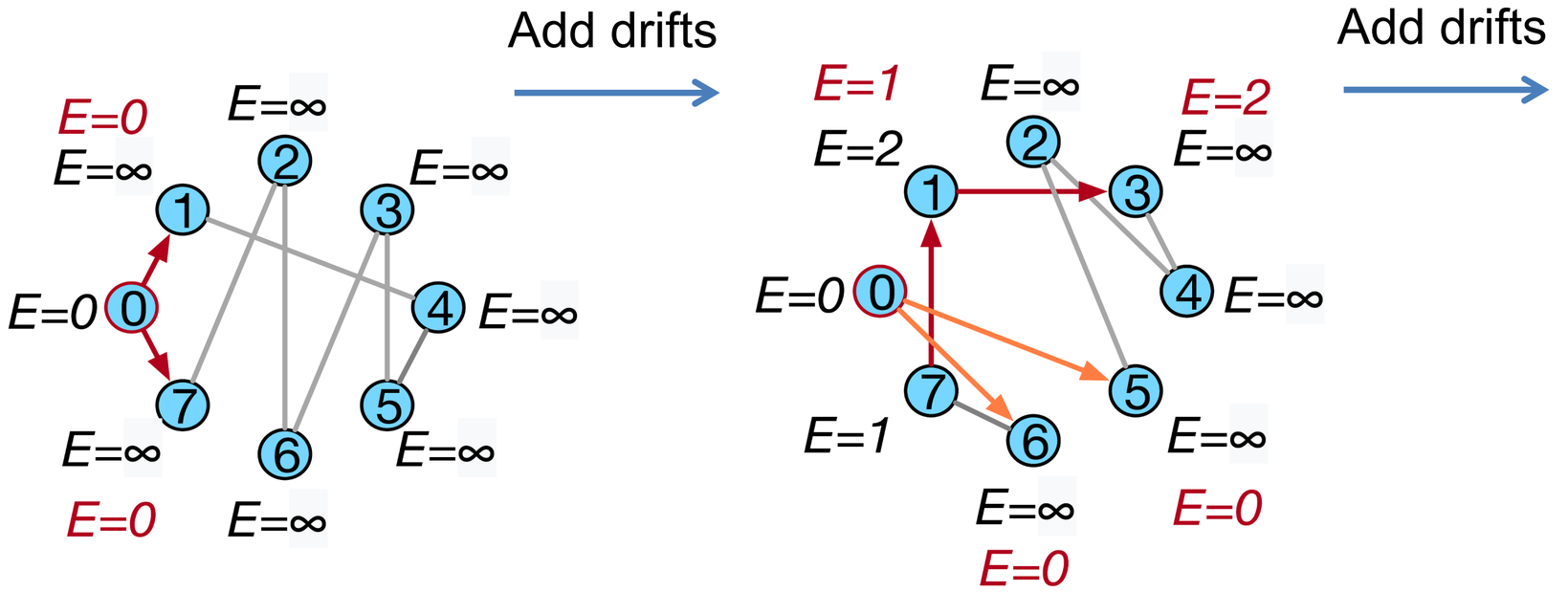


Time slice 1

Time slice 2

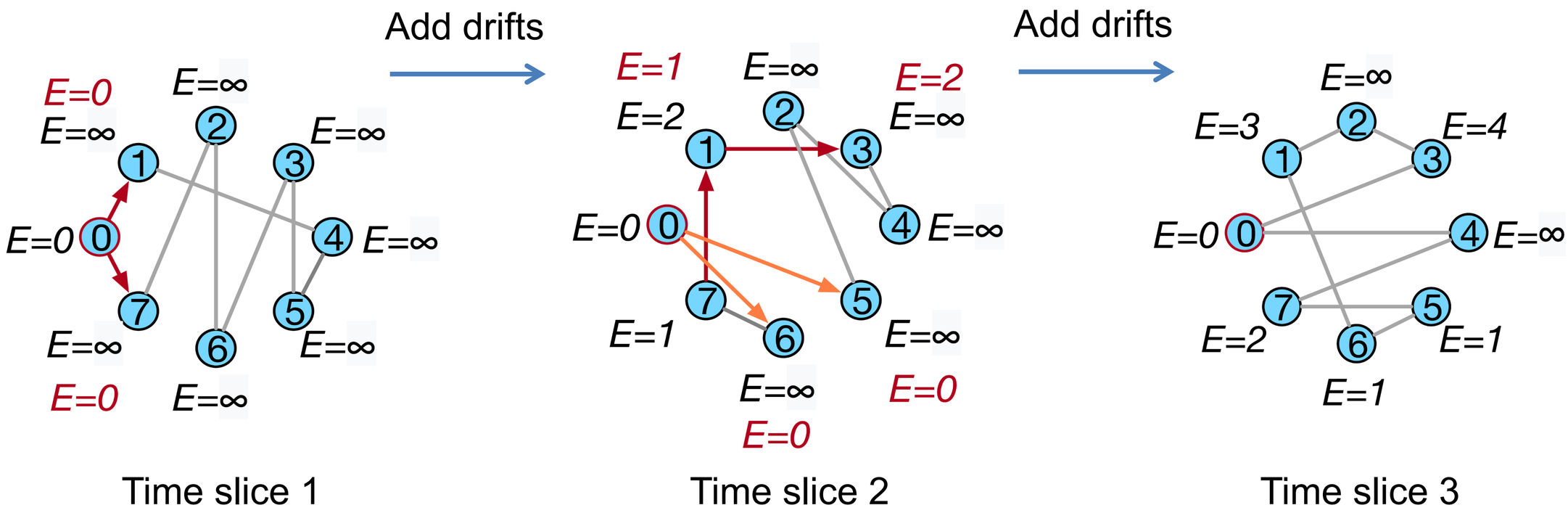
Drift-Aware Synchronization

ToR ID	0	1	2	3	4	5	6	7
Drift (ns/slice)	0	2	1	2	-1	1	1	1



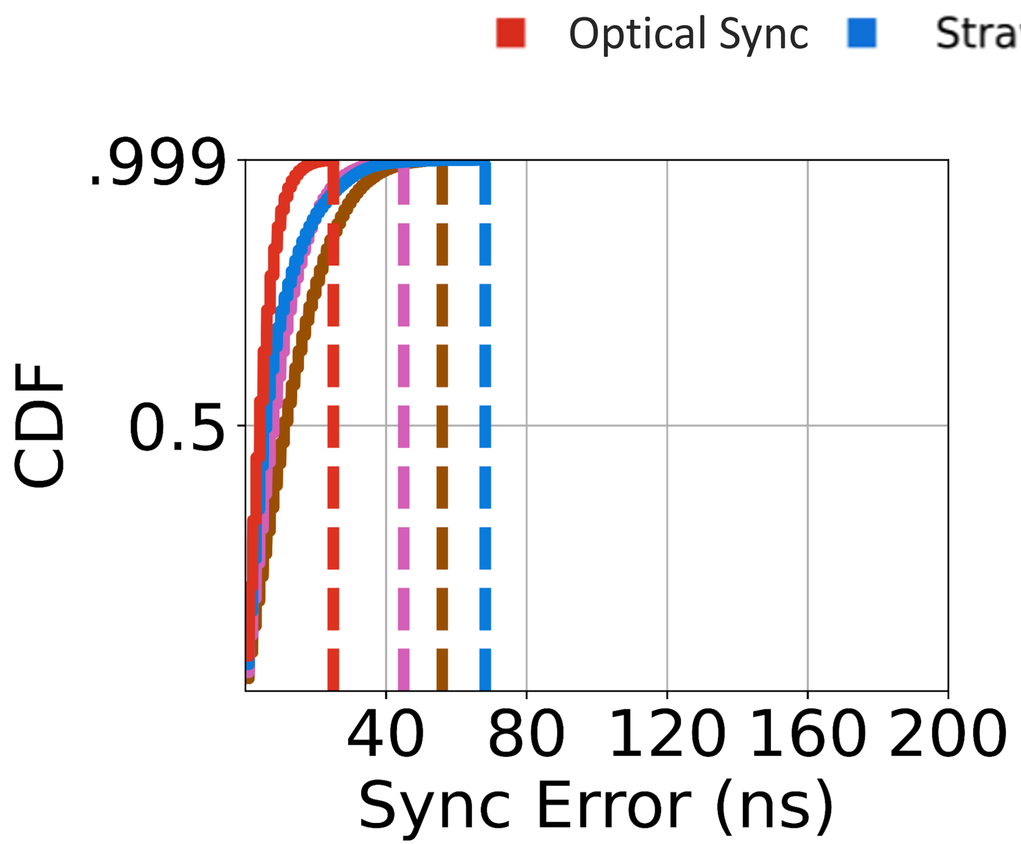
Drift-Aware Synchronization

ToR ID	0	1	2	3	4	5	6	7
Drift (ns/slice)	0	2	1	2	-1	1	1	1

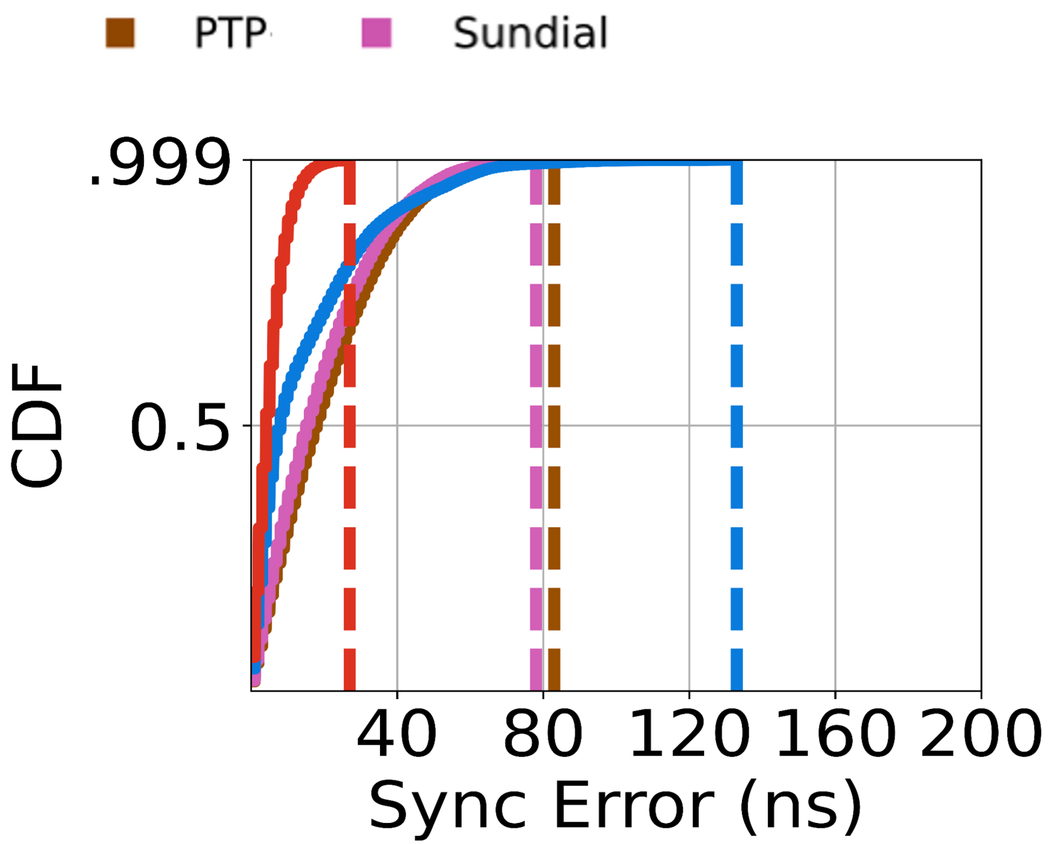


Time slice 1 | Time slice 2 | Time slice 3

Nanosecond Sync Accuracy



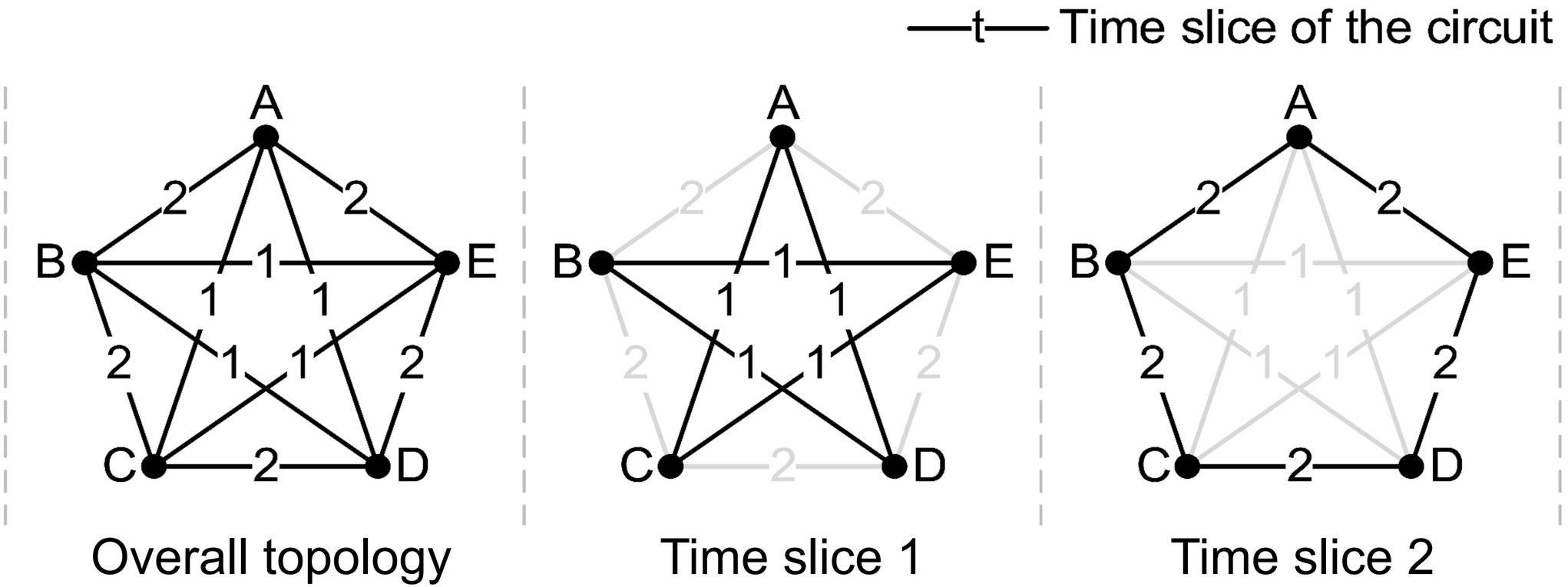
12 uplinks, 300us time slices



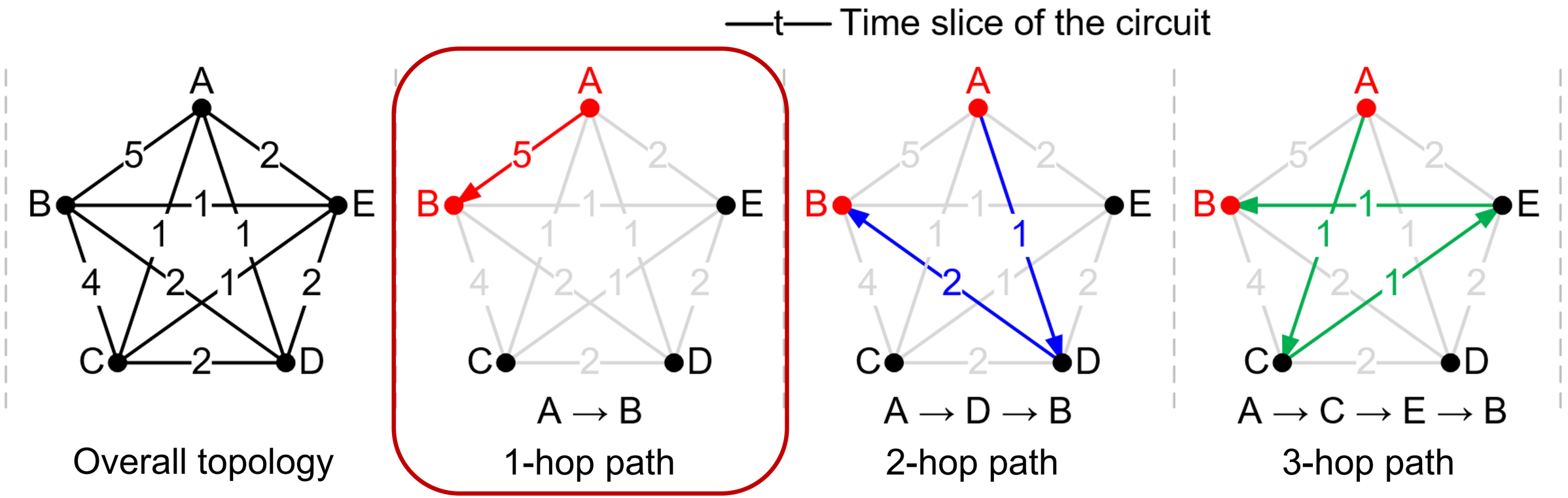
6 uplinks, 300us time slices

Optical Routing

Time-Varying Graph

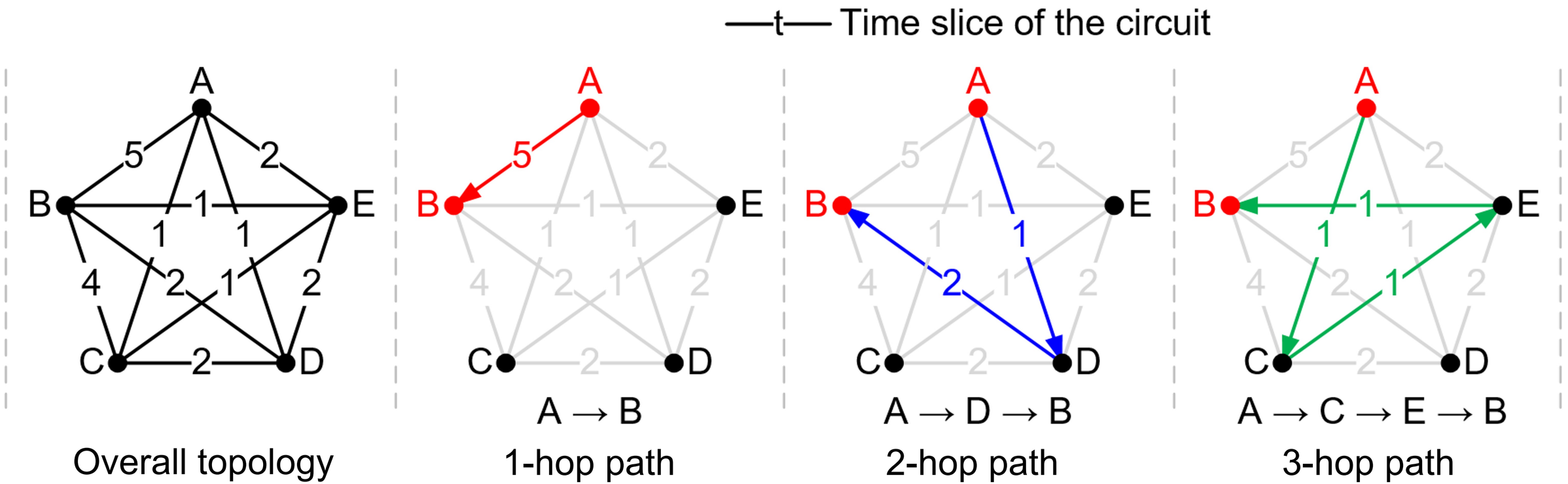


Redefine Cost



Hop Count as cost → High latency cause by circuit waiting delay

Redefine Cost

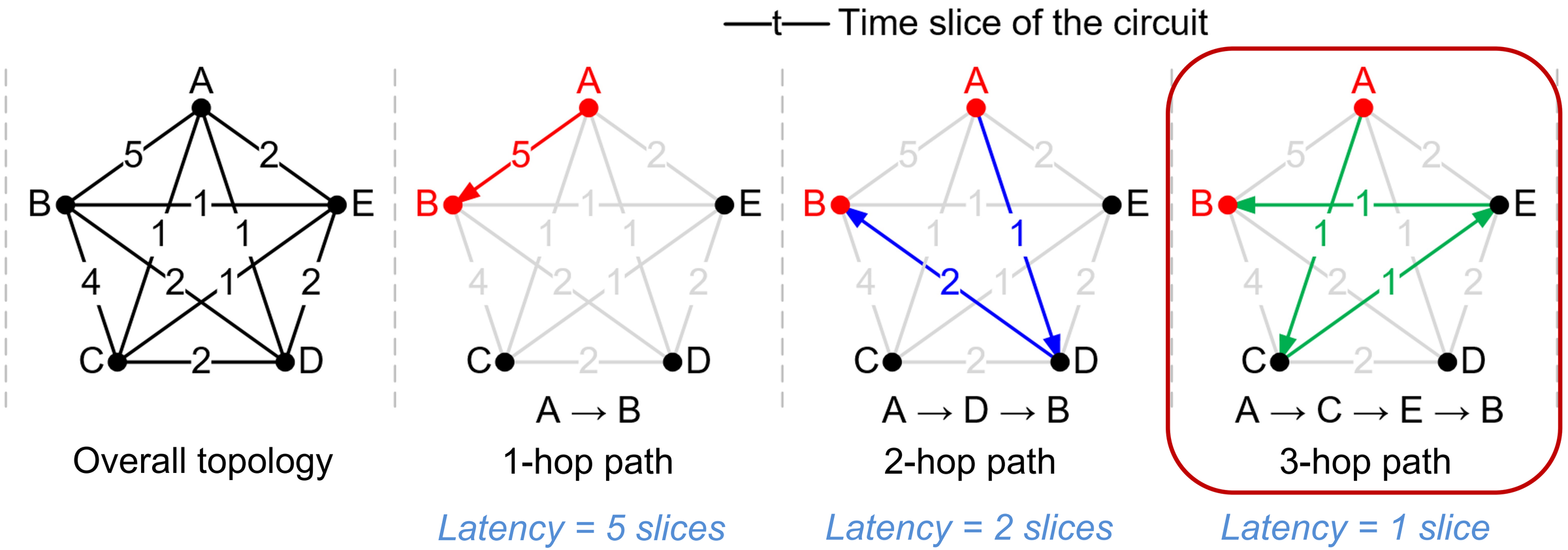


Overall topology | 1-hop path | 2-hop path | 3-hop path

Latency as cost: # time slices a packet experiences

*Latency = (t_{end} - t_{start} + 1) * time slice duration*

Redefine Cost



Latency as cost → High bandwidth cost (congestion)

Uniform Cost

$$C(p,f) = latency(p) + \alpha \times hop(p) \times size(f)/B$$

Uniform Cost — Latency — Bandwidth Tax

- ***p***: A path in optical DCN
- ***f***: A flow
- ***latency (p)*** = *(t_{end} - t_{start} + 1) * time slice duration*
- ***B***: Link bandwidth
- ***α***: Weight factor, relative importance between latency and hop count term

Uniform Cost

$$C(p,f) = latency(p) + \alpha \times hop(p) \times size(f)/B$$

Uniform Cost

Latency

Bandwidth Tax

- Uniform cost
  - Represents the “price” that a flow ***f*** travers a path ***p***
  - Is specific to both paths and flows
  - A flow should choose a path with the minimum uniform cost

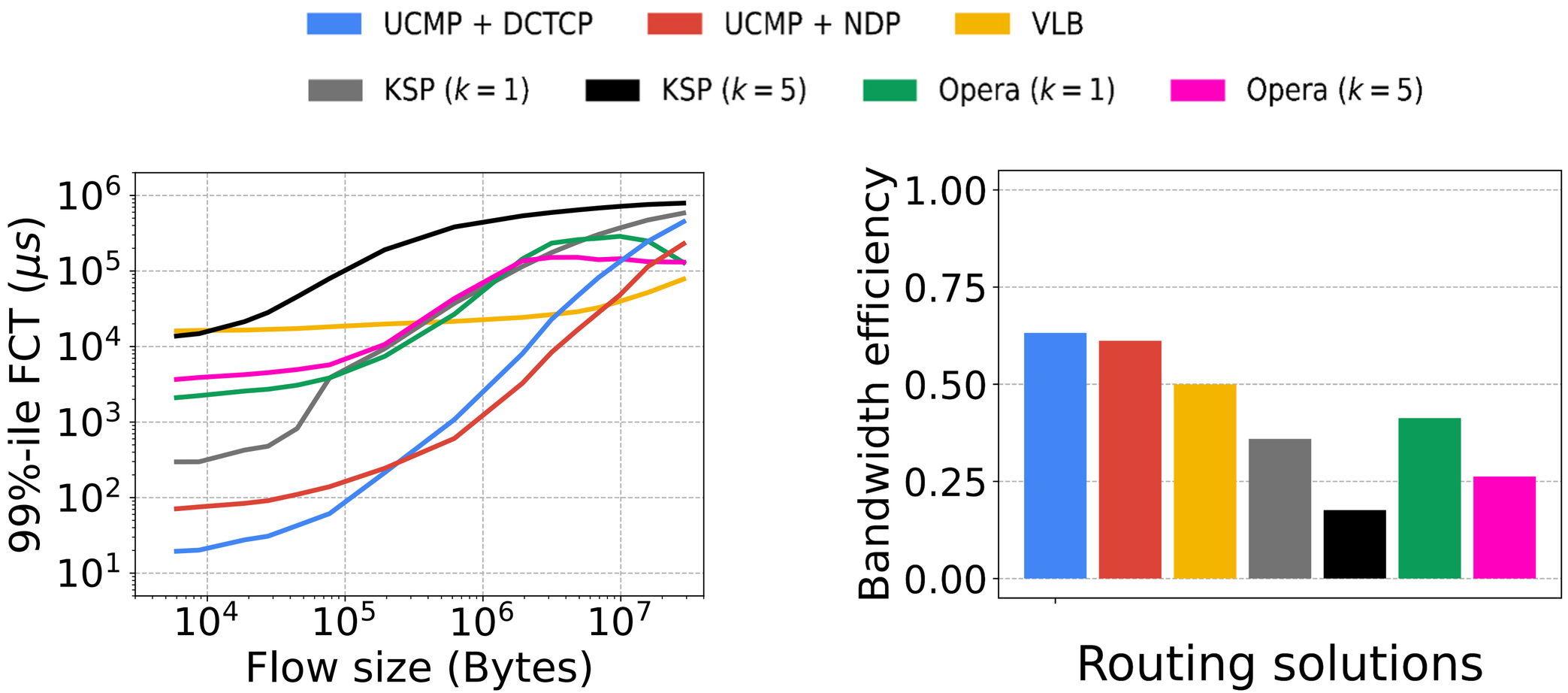
UCMP: ECMP Equivalent for Optical DCN

offline path calculation + online path assignment

hop(p)	latency(p)	C(p,f_1=1 MB)	C(p,f_2=100 KB)	C(p,f_3=10 KB)
1-hop	60 µs	140	68	60.8
2-hop	15 µs	175	31	16.6
3-hop	10 µs	250	34	12.4
4-hop	8 µs	328	40	11.2

$$C(p, f) = latency(p) + \alpha \times hop(p) \times size(f)/B$$

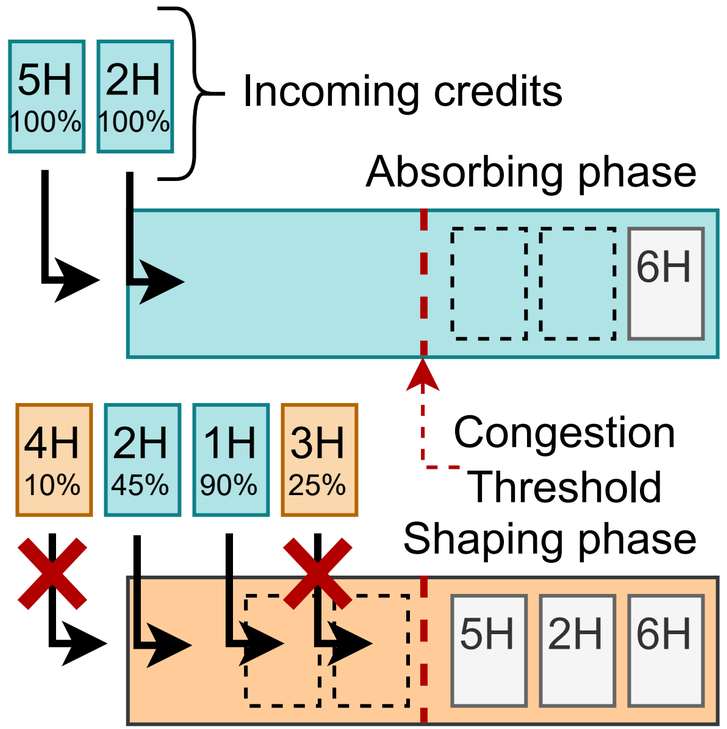
UCMP Performance



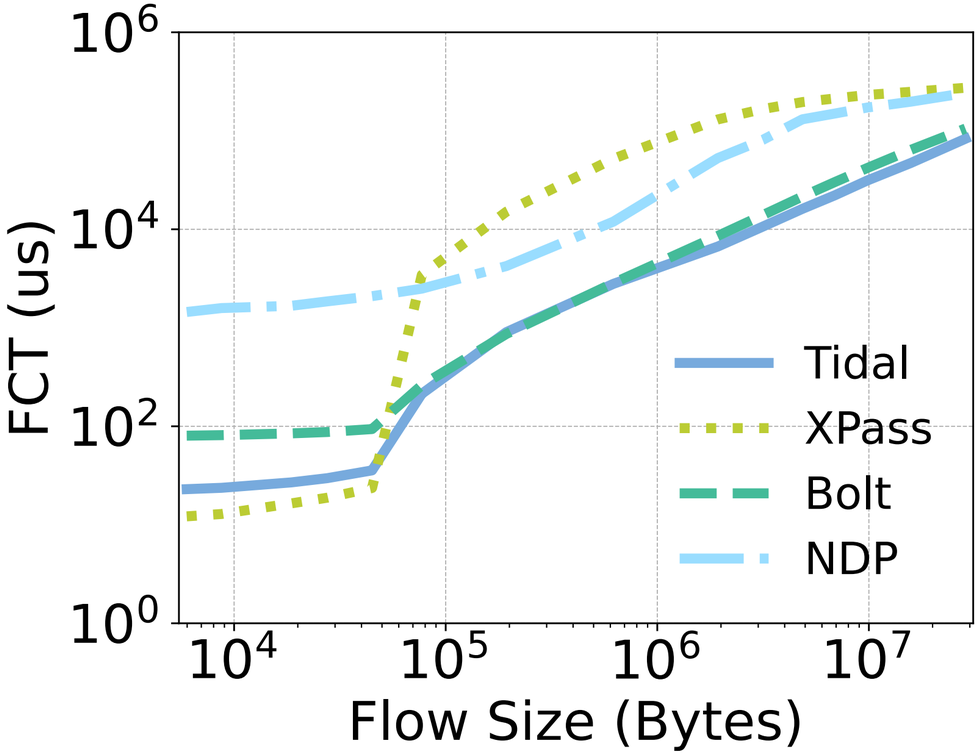
53%-98% FCT reduction, upto 55% bandwidth efficiency improvement

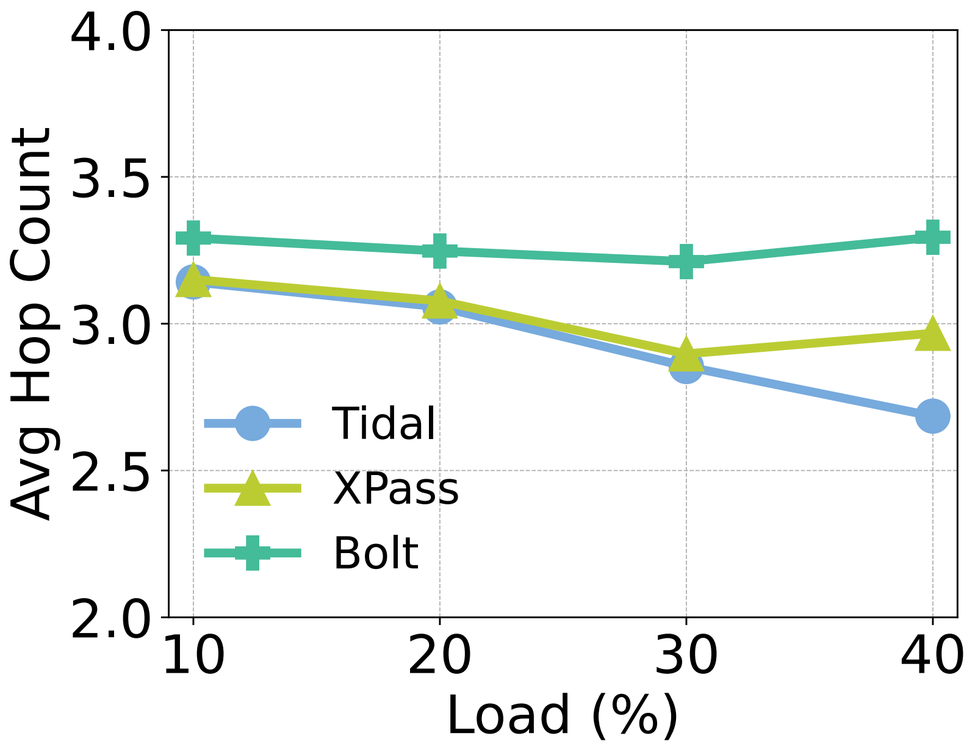
Optical Transport

- Exiting transport protocols fail to adapt to optical DCNs
  - NDP cannot tame core congestion
  - Feedback-based solutions have slow reaction
  - TDTCP tuned for electrical-optical hybrid DCNs
- Opportunistic credit-based transport
  - Proactive in assigning the amount of traffic
  - Fairness can be relaxed with the changing paths
  - More credits to packets traversing fewer hops



Preliminary Performance





Low FCT thanks to relaxation of fairness

Summary

- Optical data center network
  - Promising for future cloud
  - Many proposed hardware architectures
- System research in stagnation
  - Tightly coupled with optical hardware
  - Lack implementation and evaluation platform
- The first general platform
  - ToR-centrice design for generality and performance
  - Host system for necessary functionalities
  - Enables upper-layer research

Event Highlights

- August 4th 3:25pm, HotOptics Workshop
  - Rethinking Transport Protocols for Reconfigurable Data Centers: An Empirical Study
- August 5th 4:30pm, Main conference
  - Non-paper session: SIGCOMM Reviewing Q&A
- August 6th 11:05am, Main conference
  - Reconfigurable network topologies session
  - Uniform-Cost Multipath Routing for Reconfigurable Data Center Networks
- August 7th 11:05am, Poster & Demo
  - Poster: Opportunistic Credit-Based Transport for Reconfigurable Data Center Networks with Tidal
  - Demo: An Open Research Framework for Optical Data Center Networks