FPO Talk

PRINCETON UNIVERSITY

Building Efficient and Reliable Software-Defined Networks

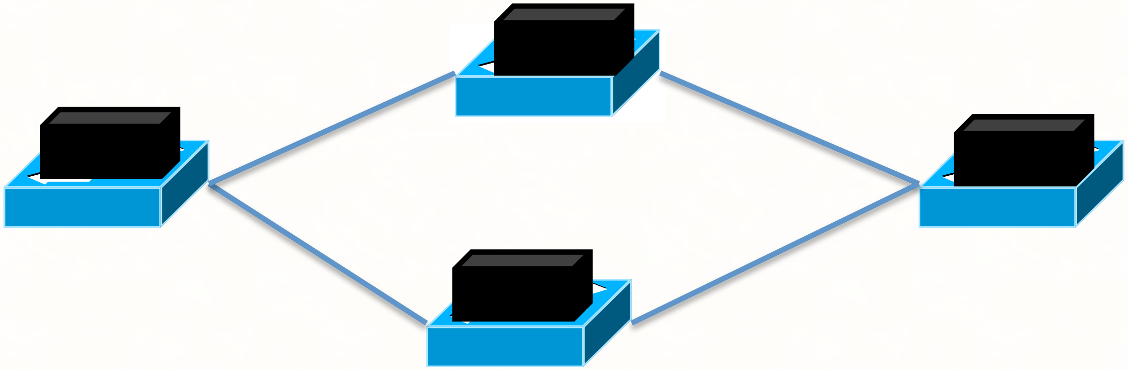
Naga Katta

Jennifer Rexford (Advisor)

Readers: Mike Freedman, David Walker

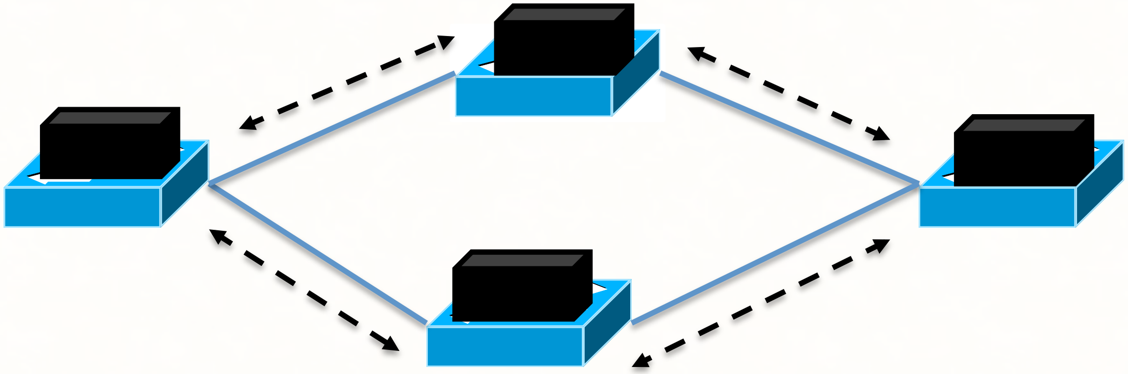
Examiners: Nick Feamster, Aarti Gupta

Traditional Networking



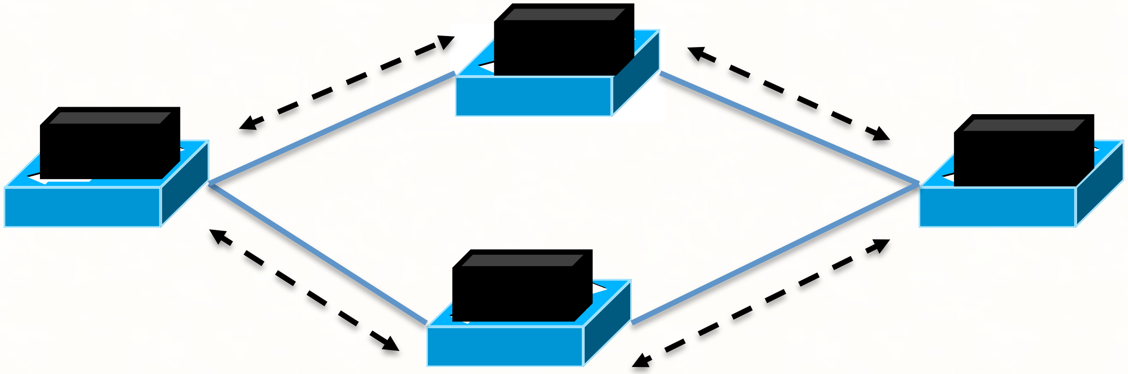
Traditional Networking

- Distributed Network Protocols

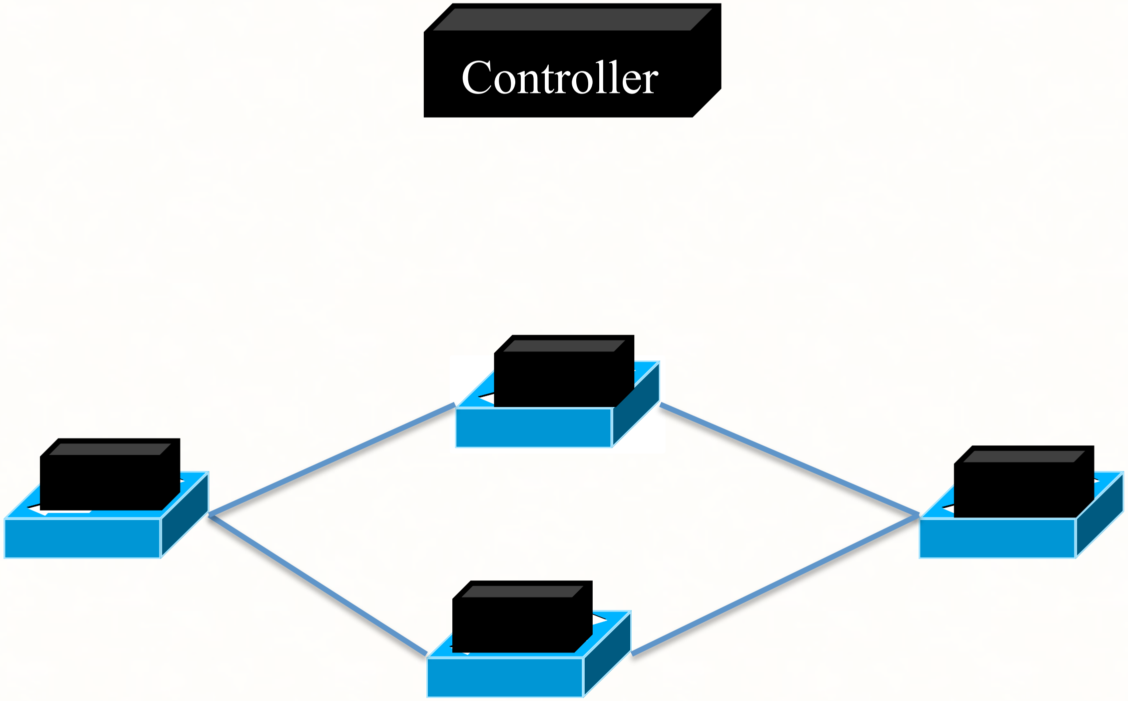


Traditional Networking

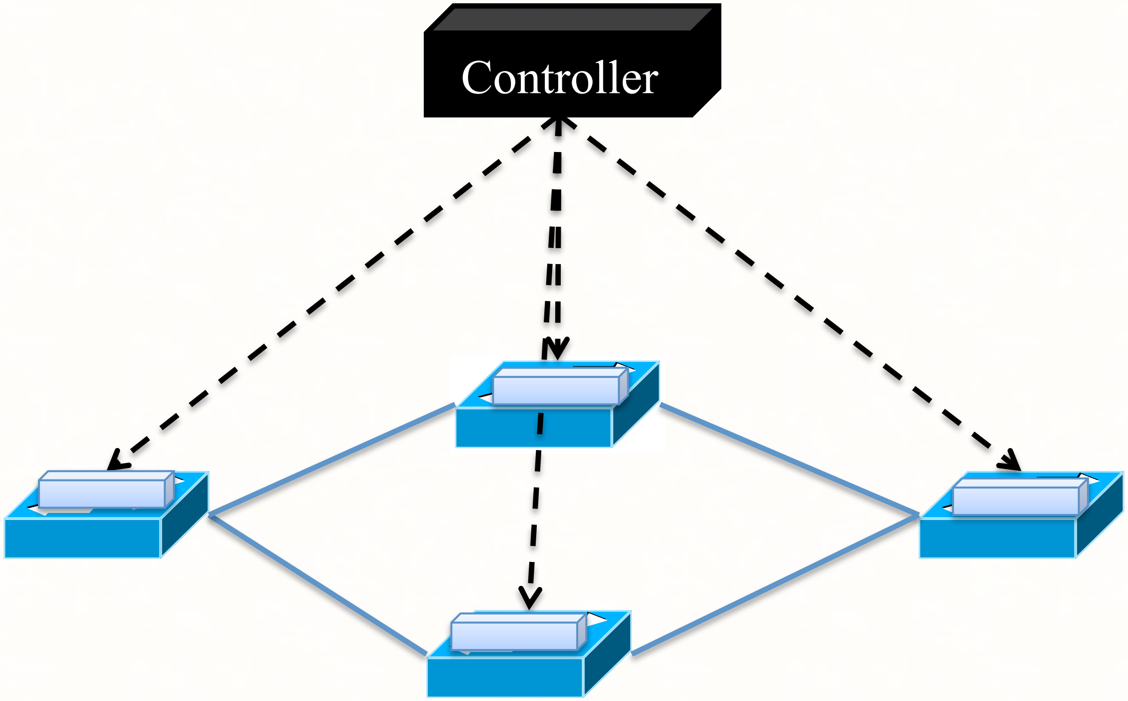
- Distributed Network Protocols
  - Reliable routing
  - Inflexible network control



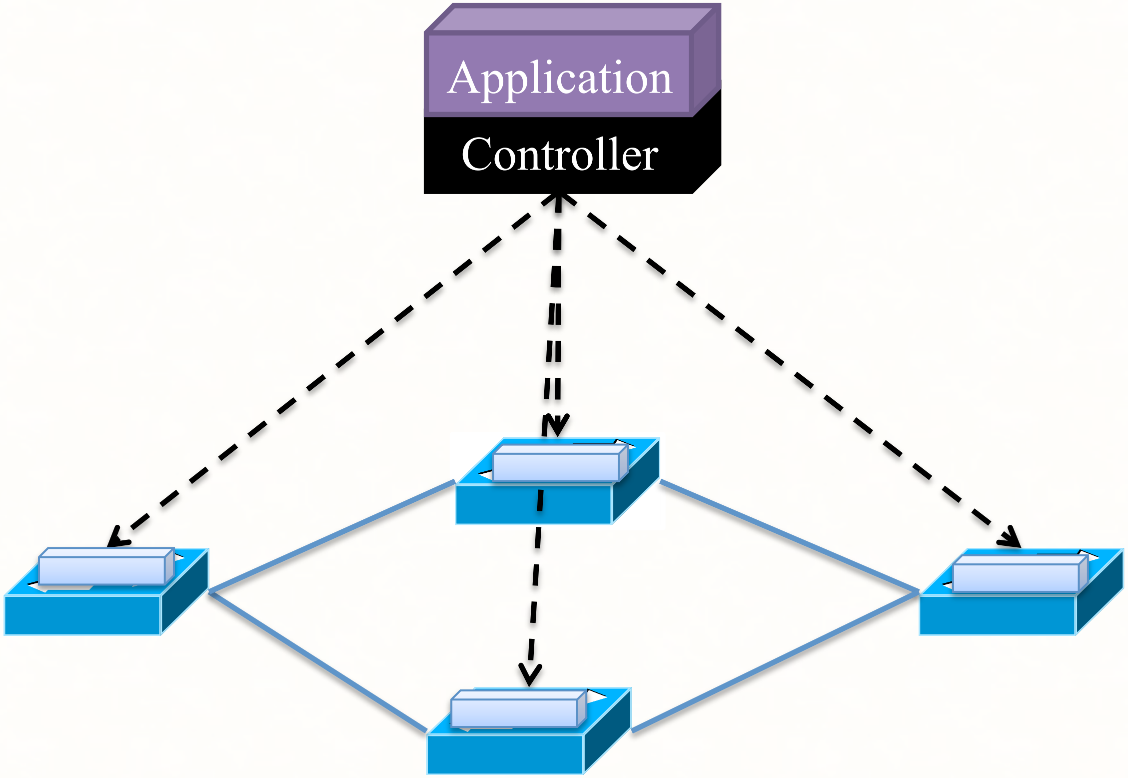
Software-Defined Networking



Software-Defined Networking



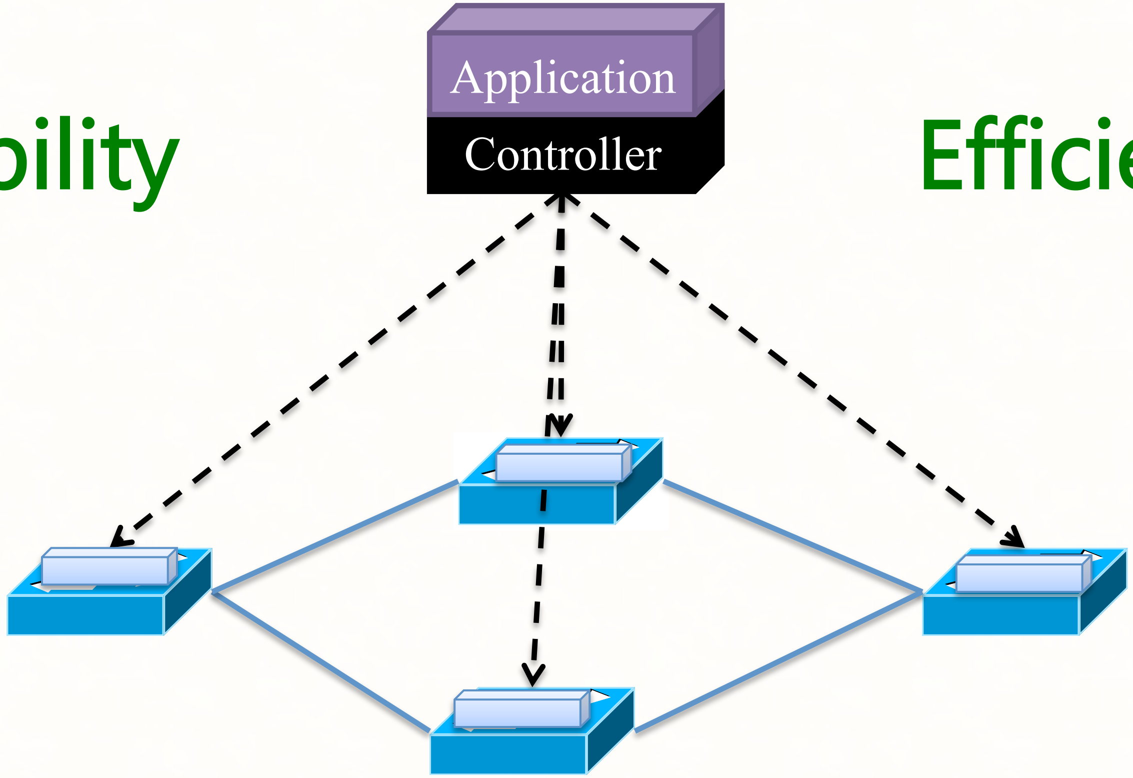
SDN: A Clean Abstraction



SDN Promises

Flexibility

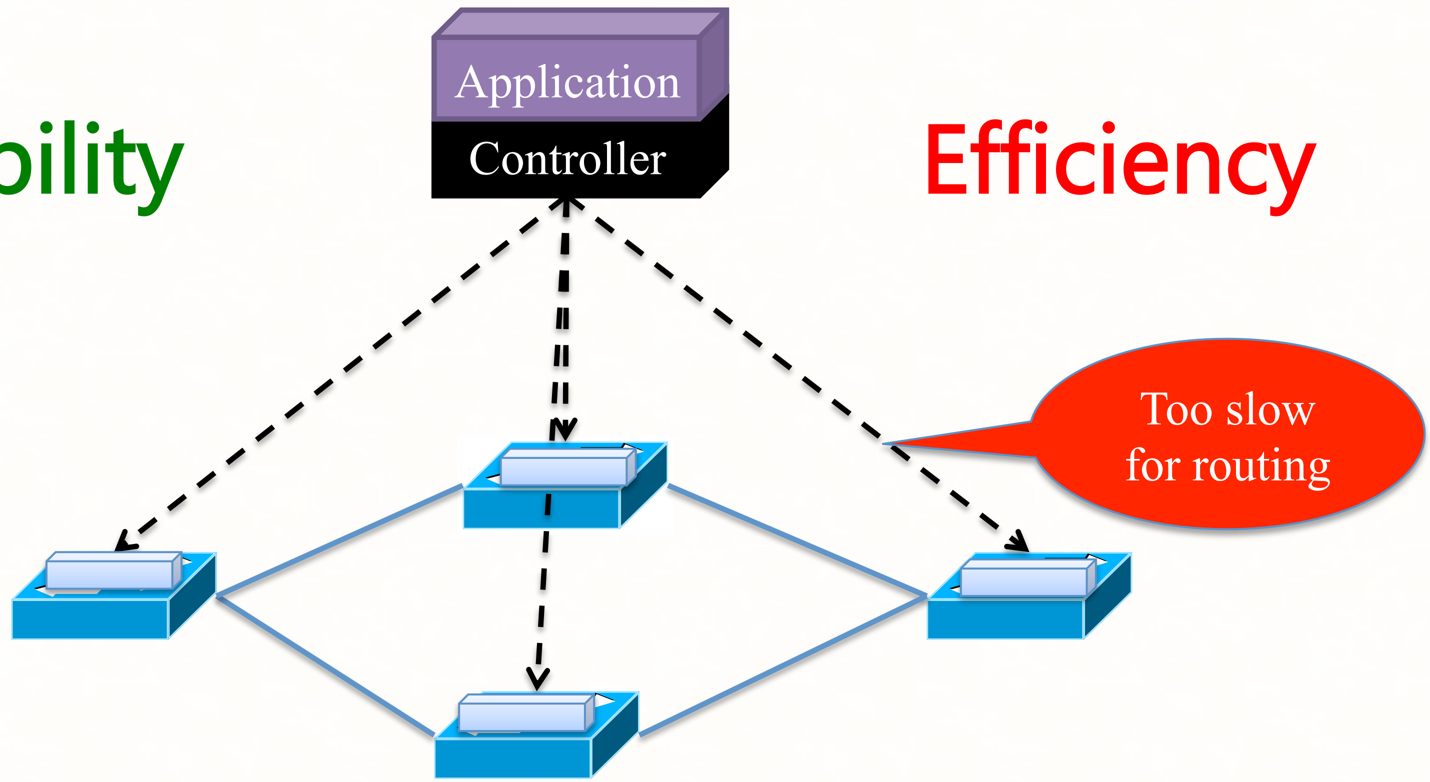
Efficiency



SDN Meets Reality

Flexibility

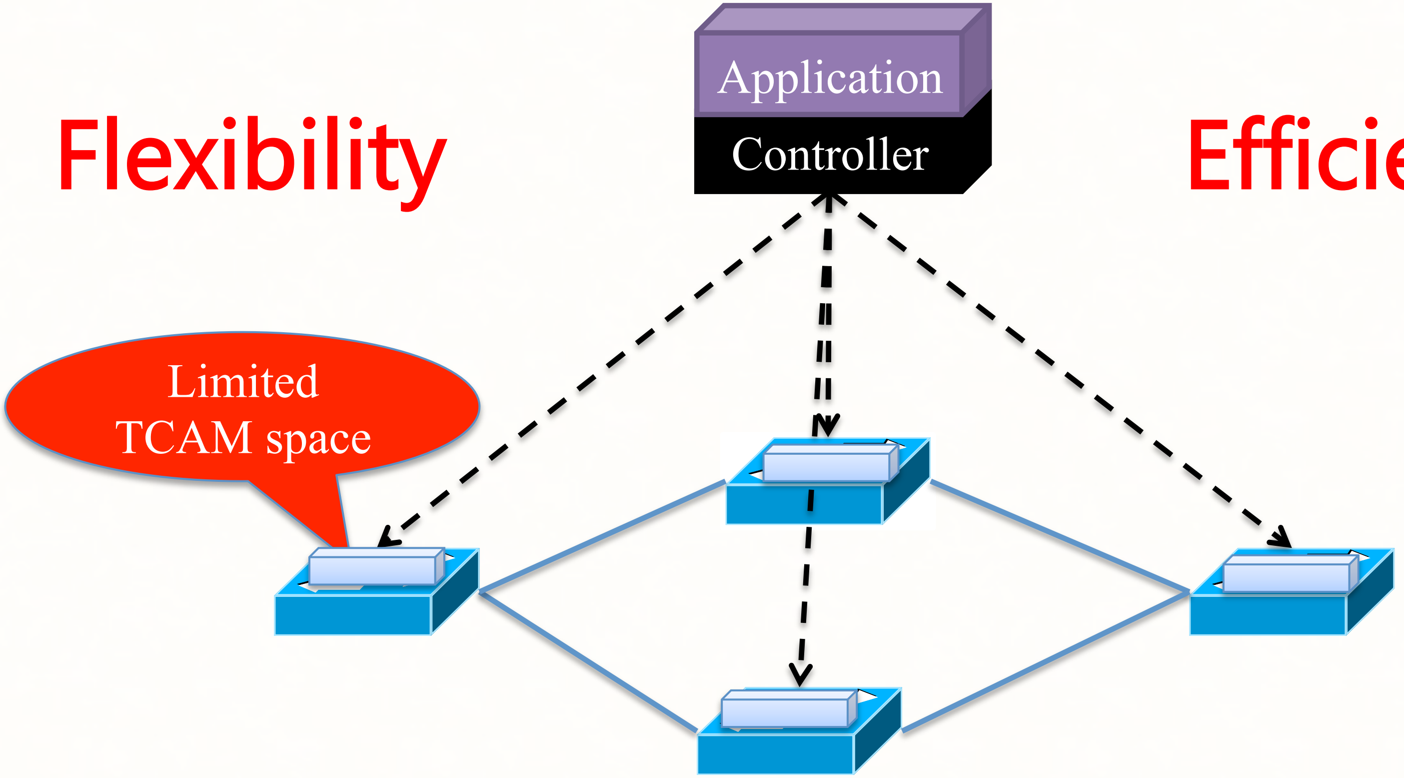
Efficiency



SDN Meets Reality

Flexibility

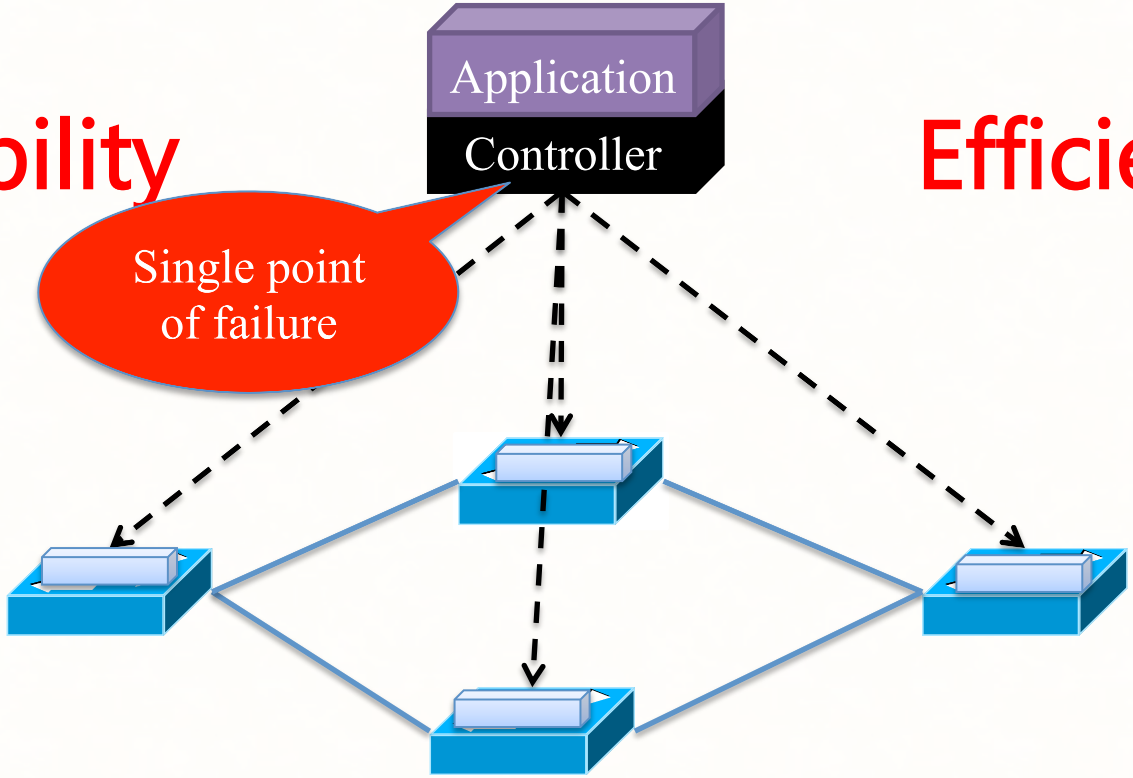
Efficiency



SDN Meets Reality

Flexibility

Efficiency

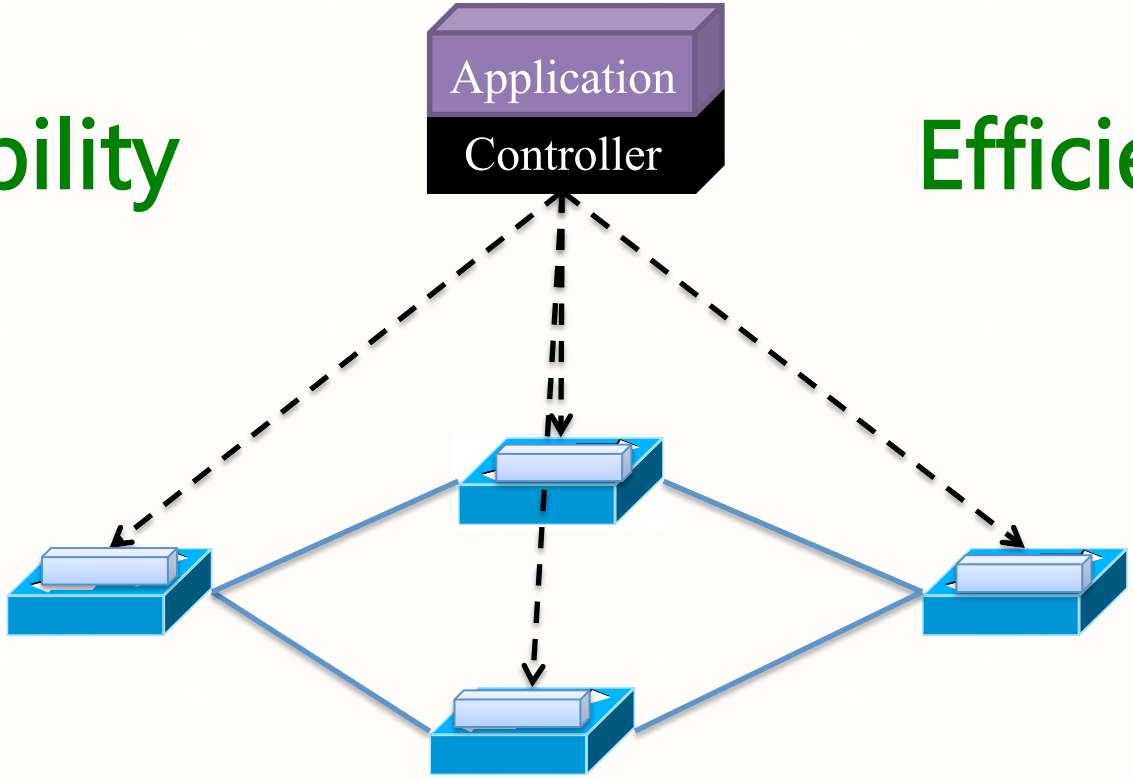


Reliability

My Research

Flexibility

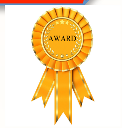
Efficiency



Reliability

Research Contribution

- HULA (SOSR 16) — Efficiency
  - An efficient non-blocking switch
- CacheFlow (SOSR 16) — Flexibility — Best Paper
  - A logical switch with infinite policy space



- Ravana (SOSR 15) — Reliability
  - Reliable logically centralized controller

Research Contribution

- HULA (SOSR 16) — Efficiency
  - An efficient non-blocking switch
- CacheFlow (SOSR 16) — Flexibility — Best Paper
  - A logical switch with infinite policy space

- Ravana (SOSR 15) — Reliability
  - Reliable logically centralized controller

PRINCETON UNIVERSITY

HULA: Scalable Load Balancing Using Programmable Data Planes

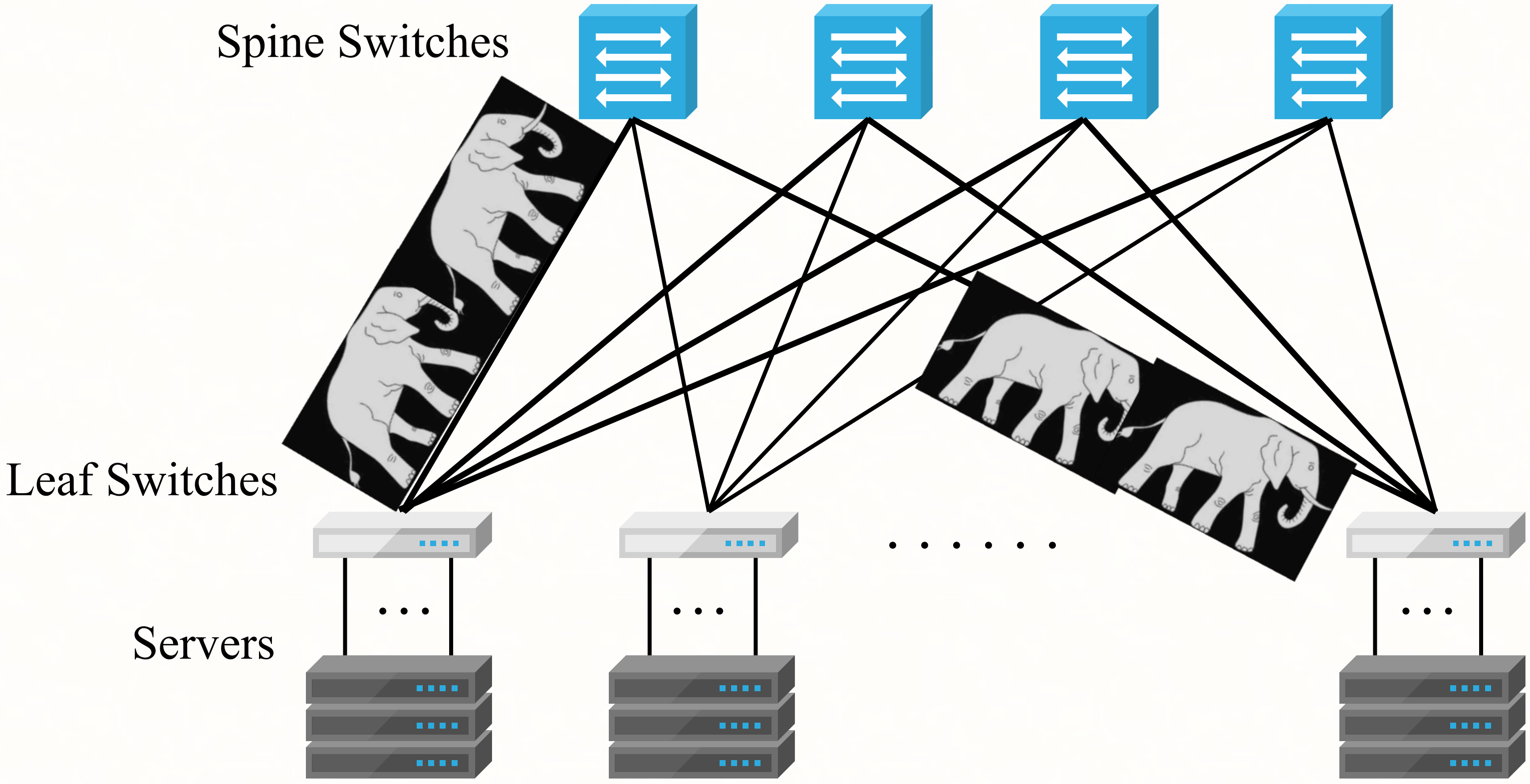
Naga Katta[1]

Mukesh Hira[2], Changhoon Kim[3], Anirudh Sivaraman[4], Jennifer Rexford[1]

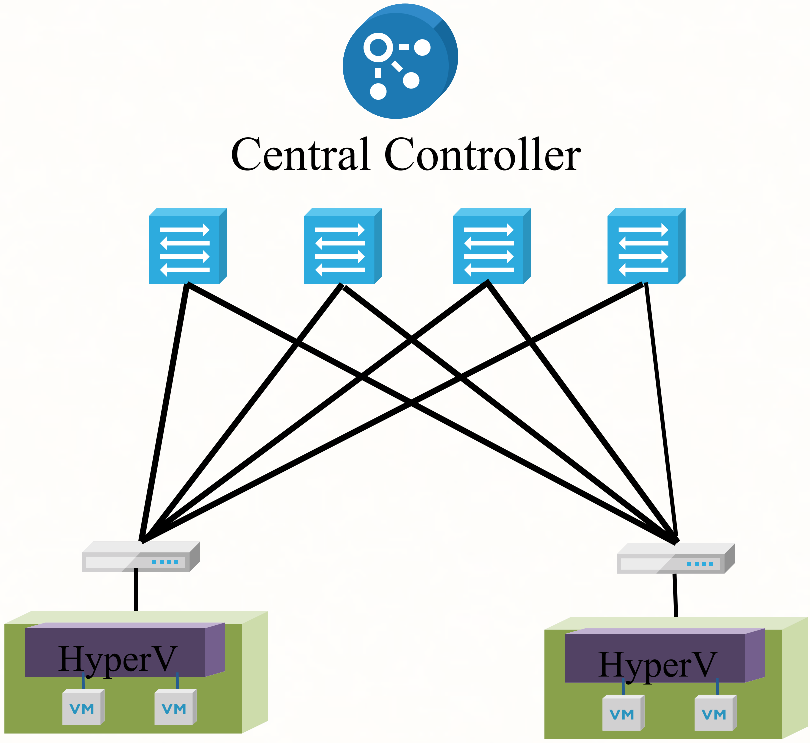
1.Princeton 2.VMware 3.Barefoot Networks 4.MIT

Load Balancing Today

Equal Cost Multi-Path (ECMP) – hashing

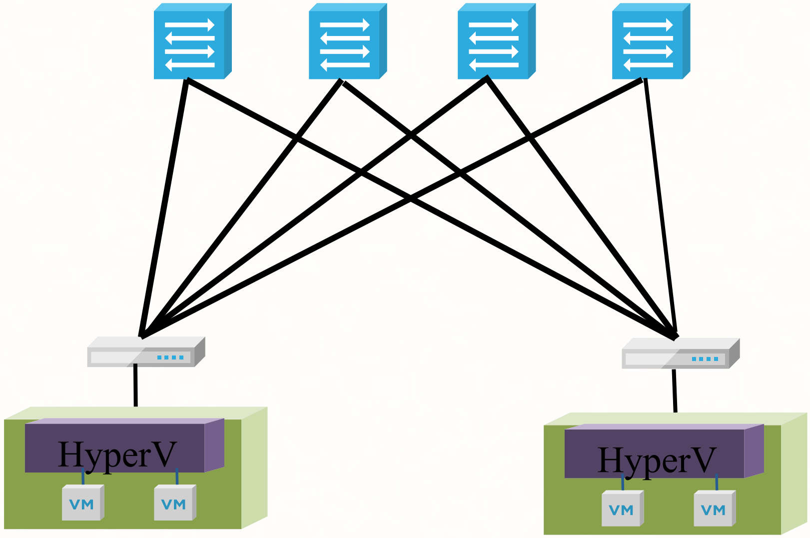


Alternatives Proposed



Slow reaction time

Congestion-Aware Fabric



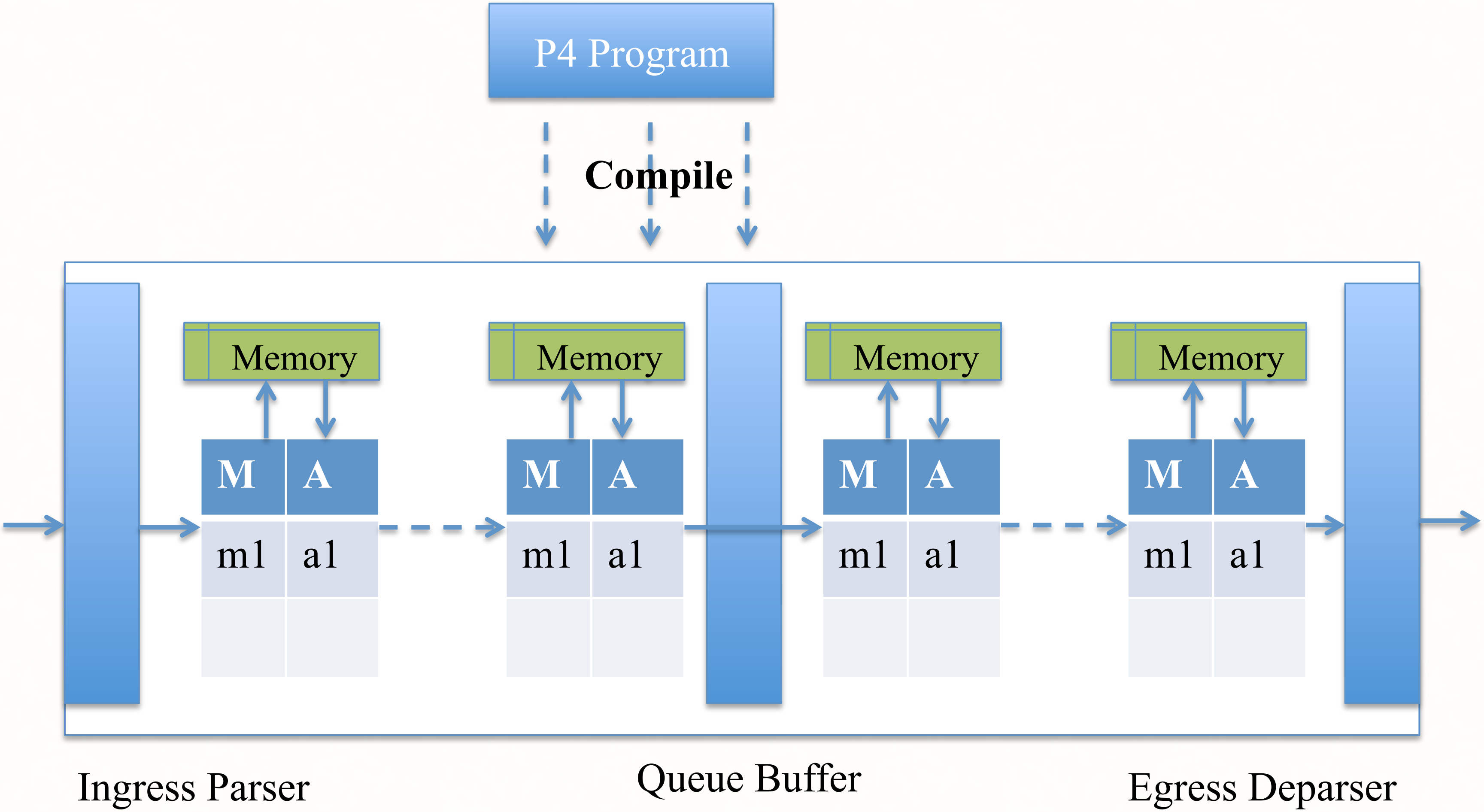
Congestion-aware Load Balancing
CONGA – Cisco

Designed for 2-tier topologies

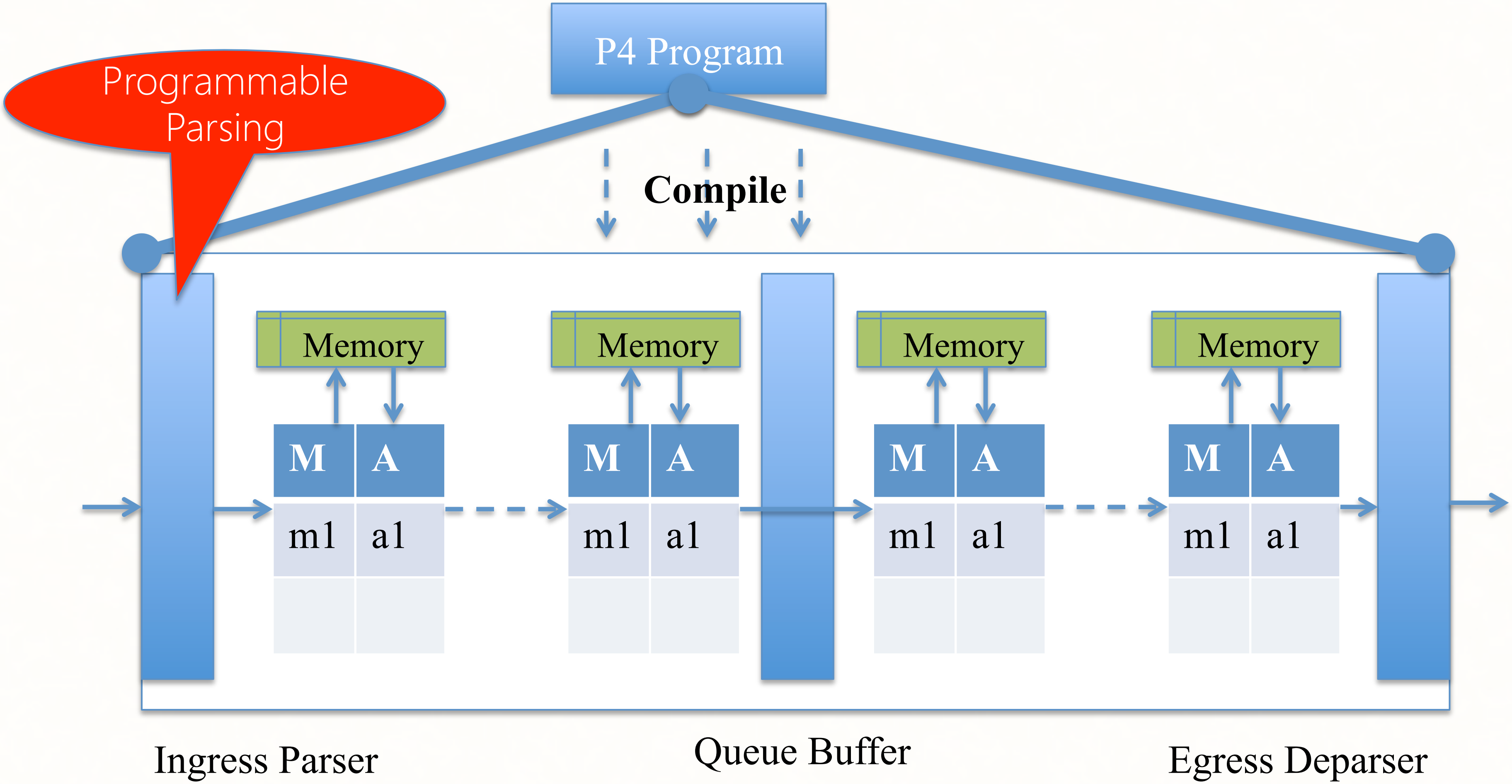
Programmable Dataplanes

- Advanced switch architectures (P4 model)
  - Programmable packet headers
  - Stateful packet processing
- Applications
  - In-band Network Telemetry (INT)
  - HULA load balancer
- Examples
  - Barefoot RMT, Intel Flexpipe, etc.

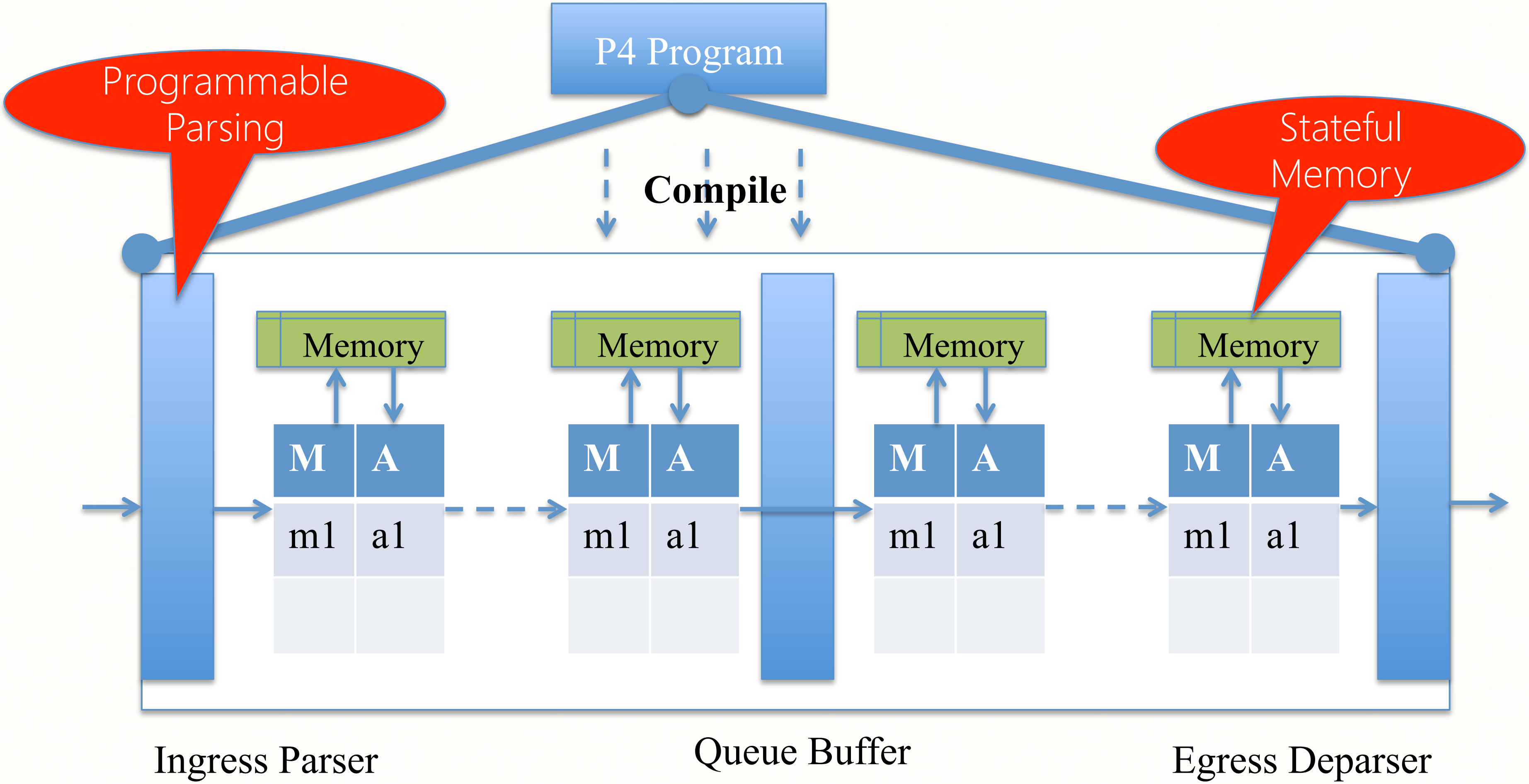
Programmable Switches - Capabilities



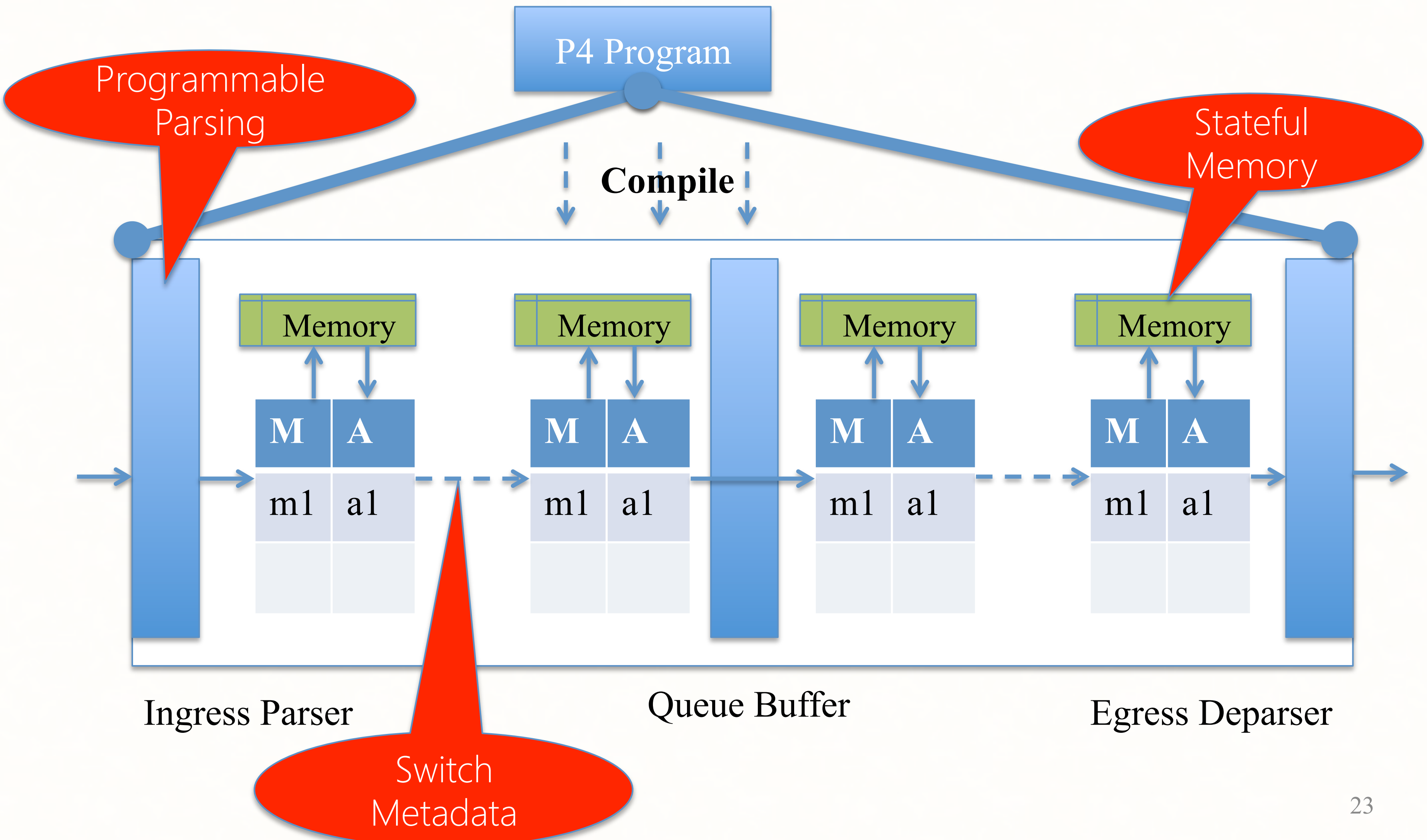
Programmable Switches - Capabilities



Programmable Switches - Capabilities



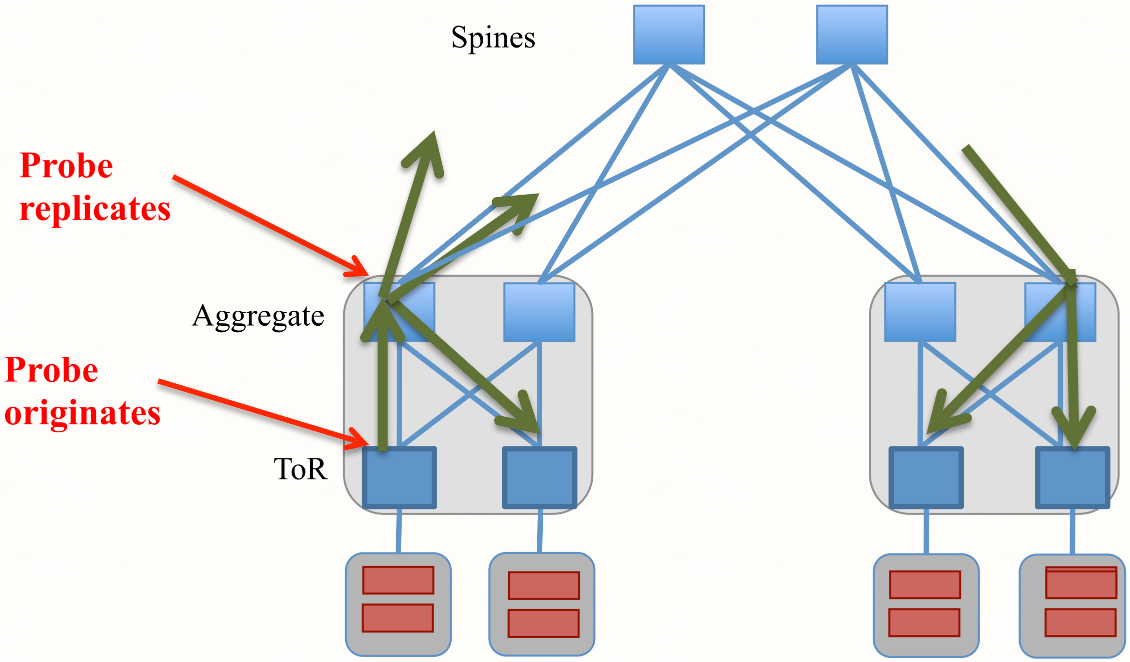
Programmable Switches - Capabilities



Hop-by-hop Utilization-aware Load-balancing Architecture

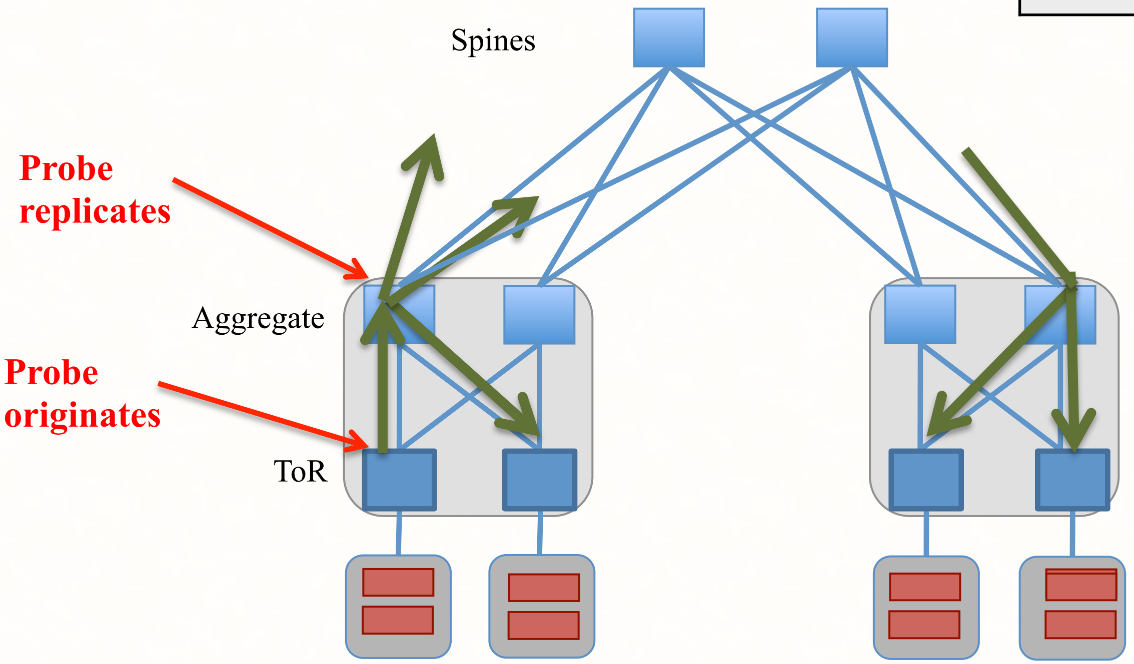
1. HULA probes propagate path utilization
   – Congestion-aware switches
2. Each switch remembers best next hop
   – Scalable and topology-oblivious
3. Split elephants to mice flows (flowlets)
   – Fine-grained load balancing

1. Probes carry path utilization

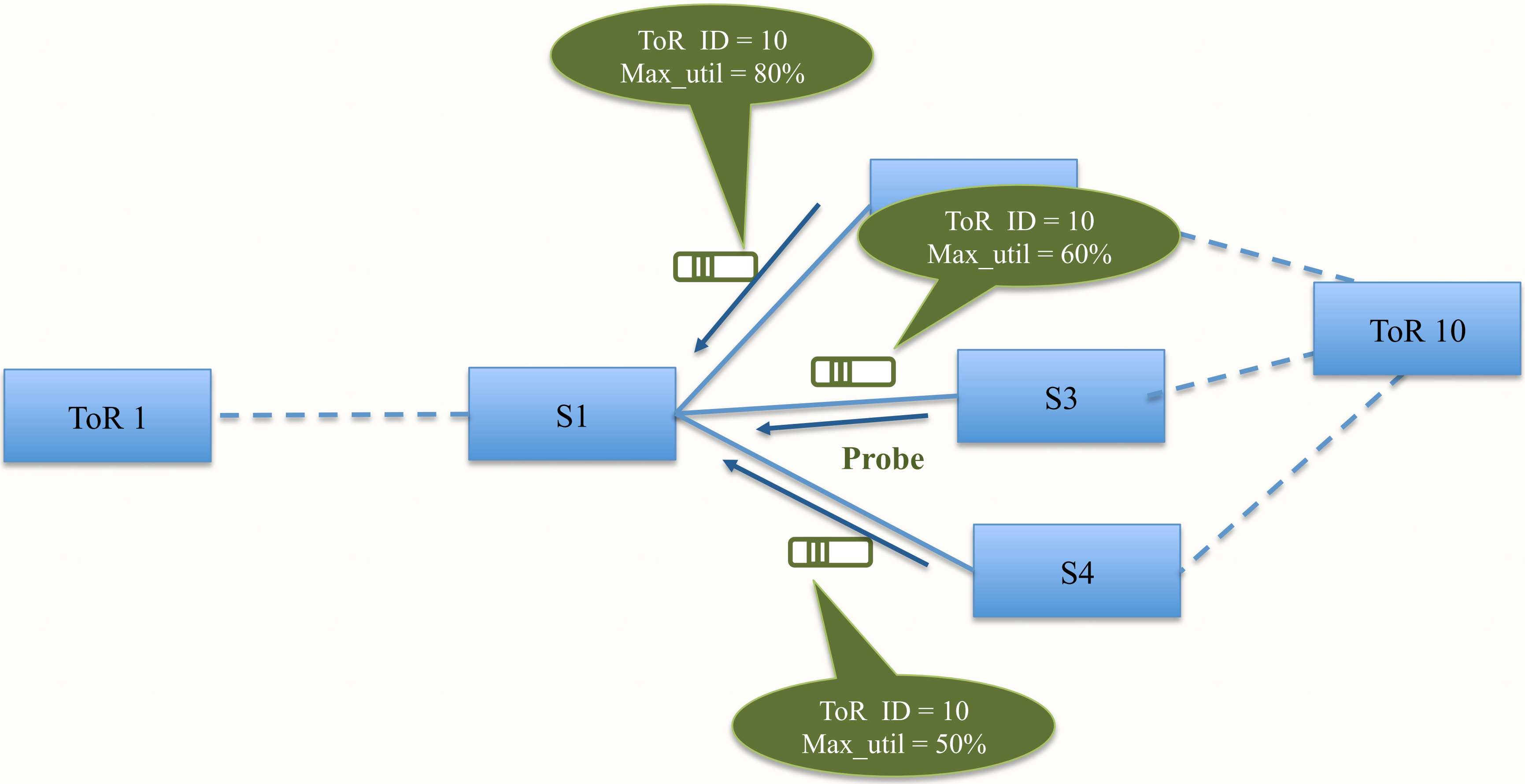


1. Probes carry path utilization

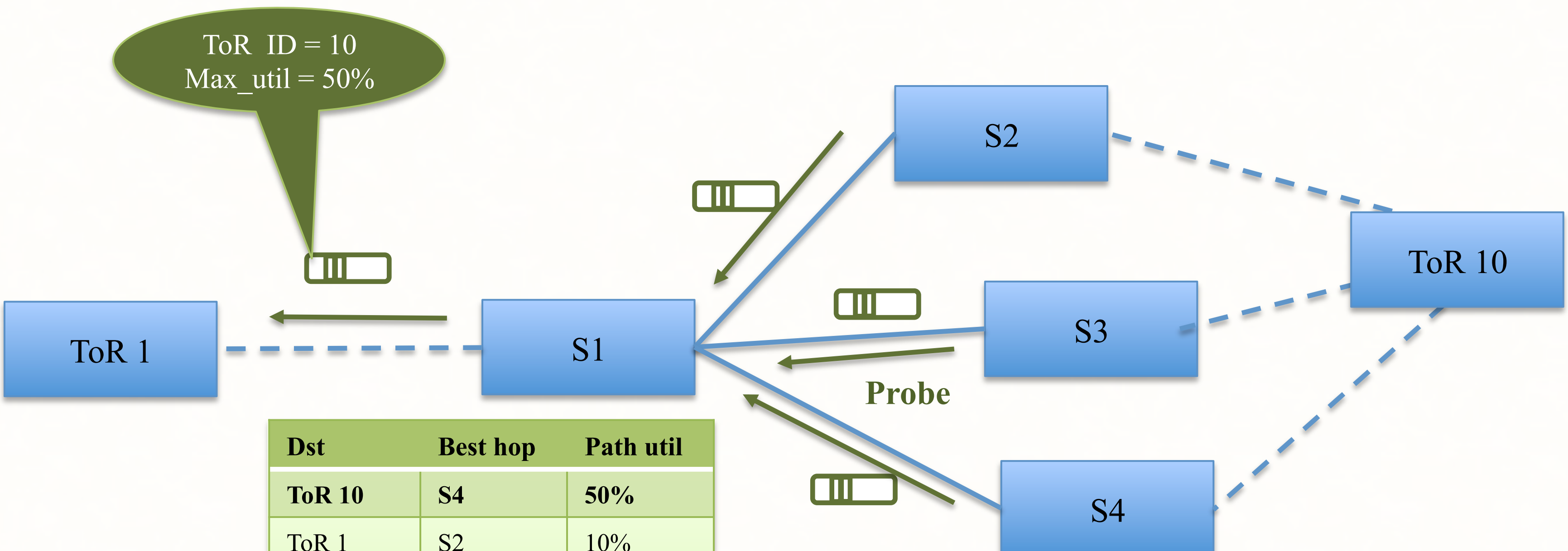
P4 primitives
New header format
Programmable Parsing
Switch metadata



1. Probes carry path utilization



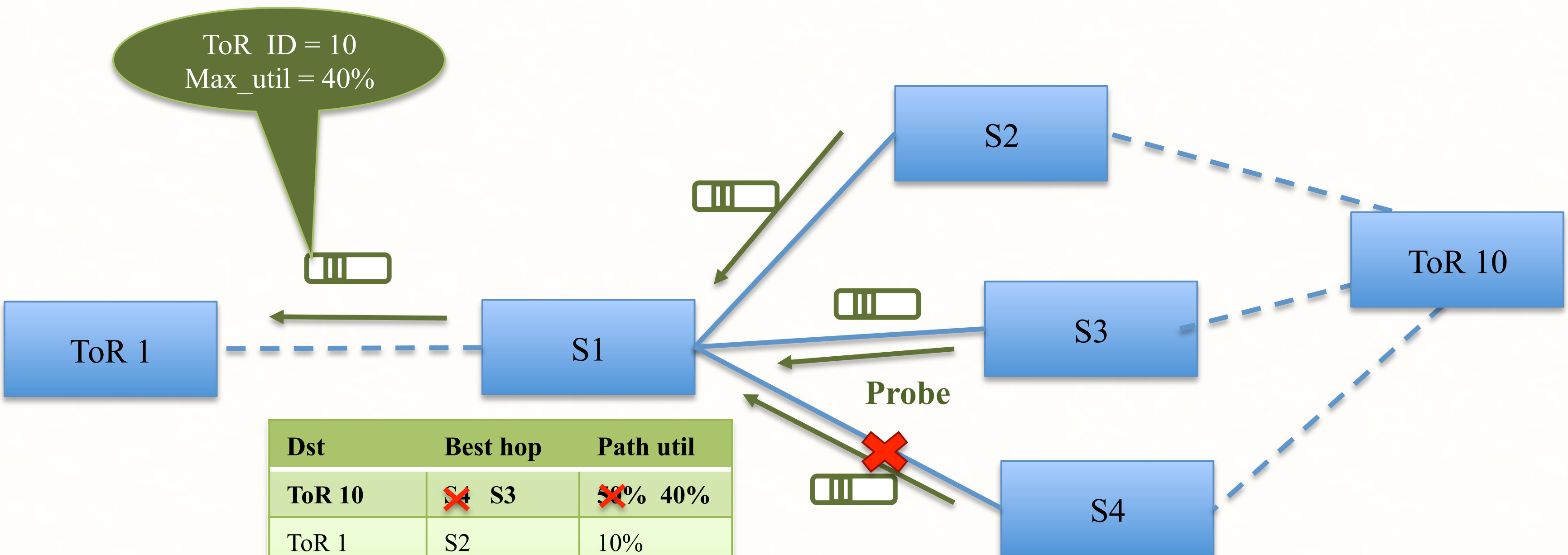
2. Switch identifies best downstream path



Dst	Best hop	Path util
ToR 10	**S4**	**50%**
ToR 1	S2	10%
…	…	

Best hop table

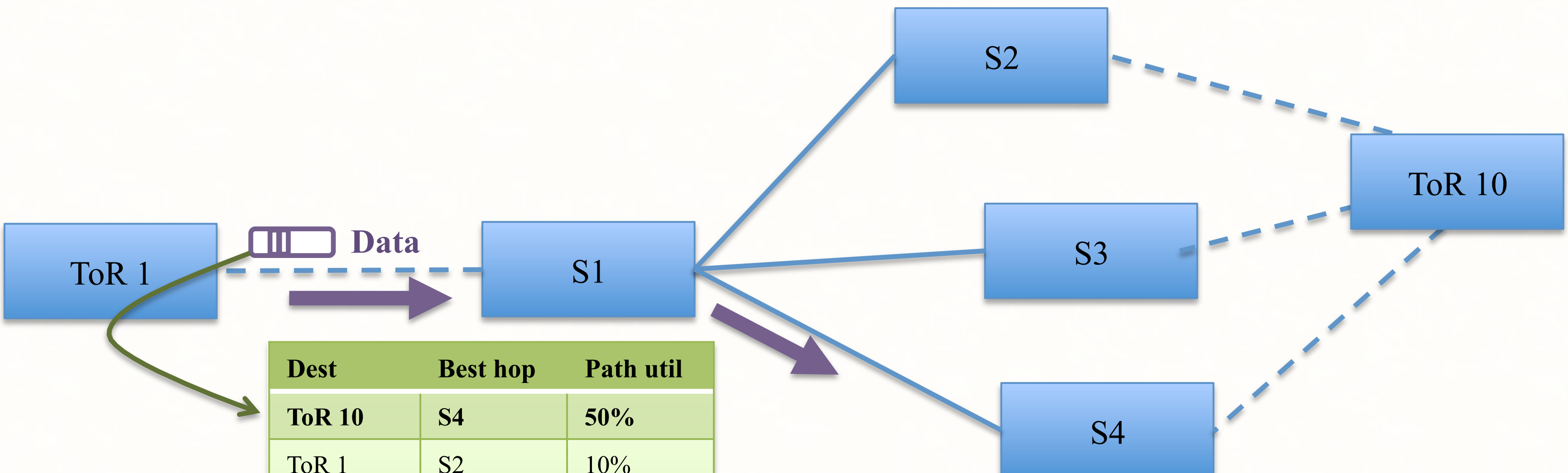
2. Switch identifies best downstream path



Dst	Best hop	Path util
ToR 10	**~~S4~~ S3**	**~~50%~~ 40%**
ToR 1	S2	10%
…	…	

Best hop table

3. Switches load balance flowlets



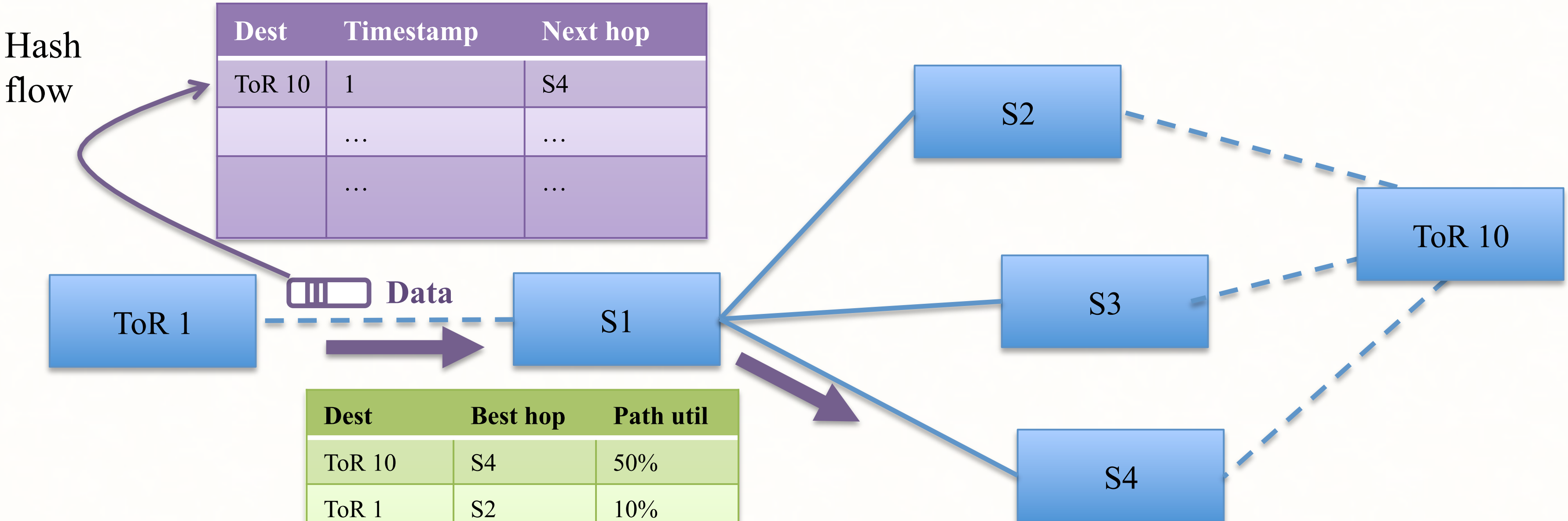
Dest	Best hop	Path util
ToR 10	**S4**	**50%**
ToR 1	S2	10%
…	…	

Best hop table

3. Switches load balance flowlets

Flowlet table

Dest	Timestamp	Next hop
ToR 10	1	S4
	…	…
	…	…



Dest	Best hop	Path util
ToR 10	S4	50%
ToR 1	S2	10%
…	…	

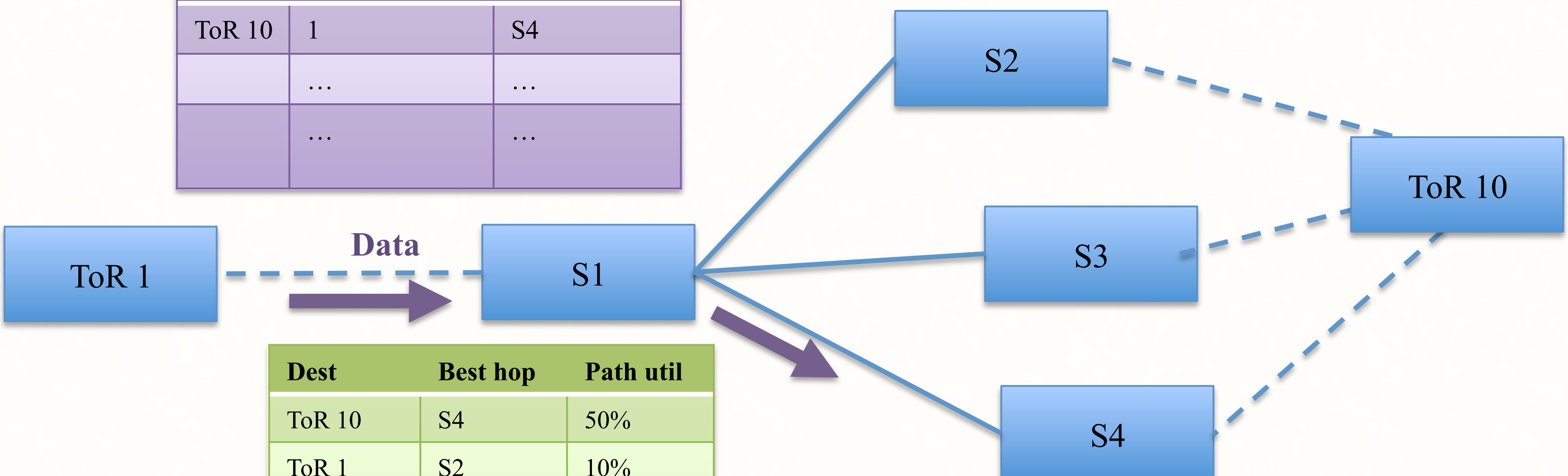
Best hop table

3. Switches load balance flowlets

P4 primitives
RW access to stateful memory
Comparison/arithmetic operators

Flowlet table

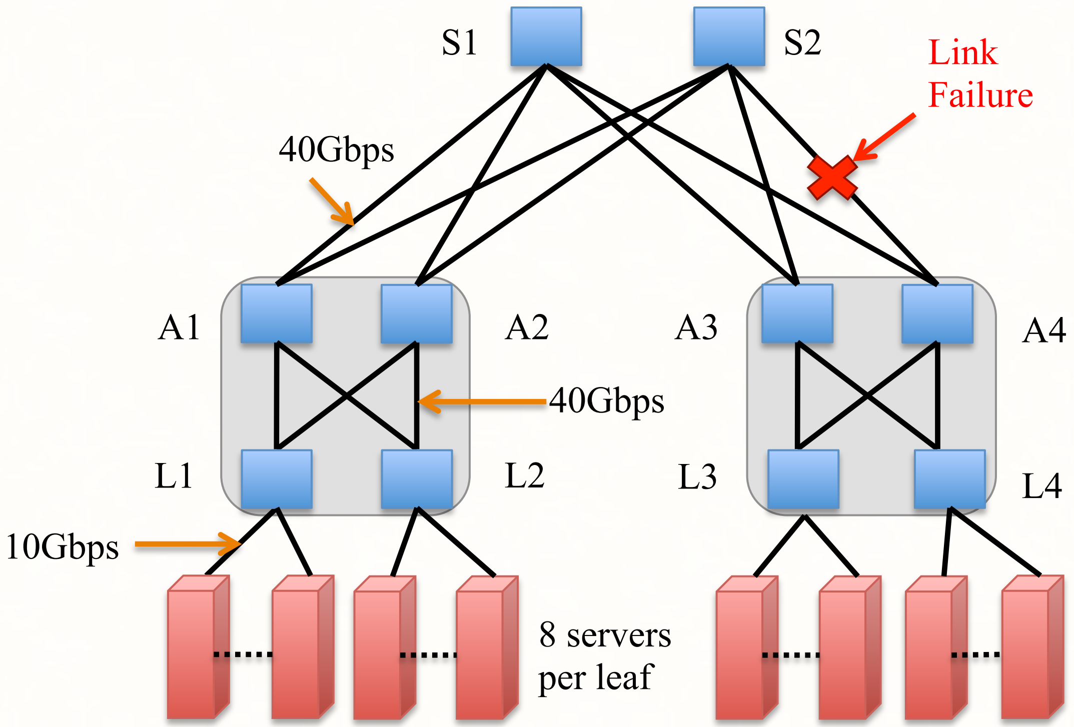
Dest	Timestamp	Next hop
ToR 10	1	S4
	…	…
	…	…



Dest	Best hop	Path util
ToR 10	S4	50%
ToR 1	S2	10%
…	…	

Best hop table

Evaluated Topology



Evaluation Setup

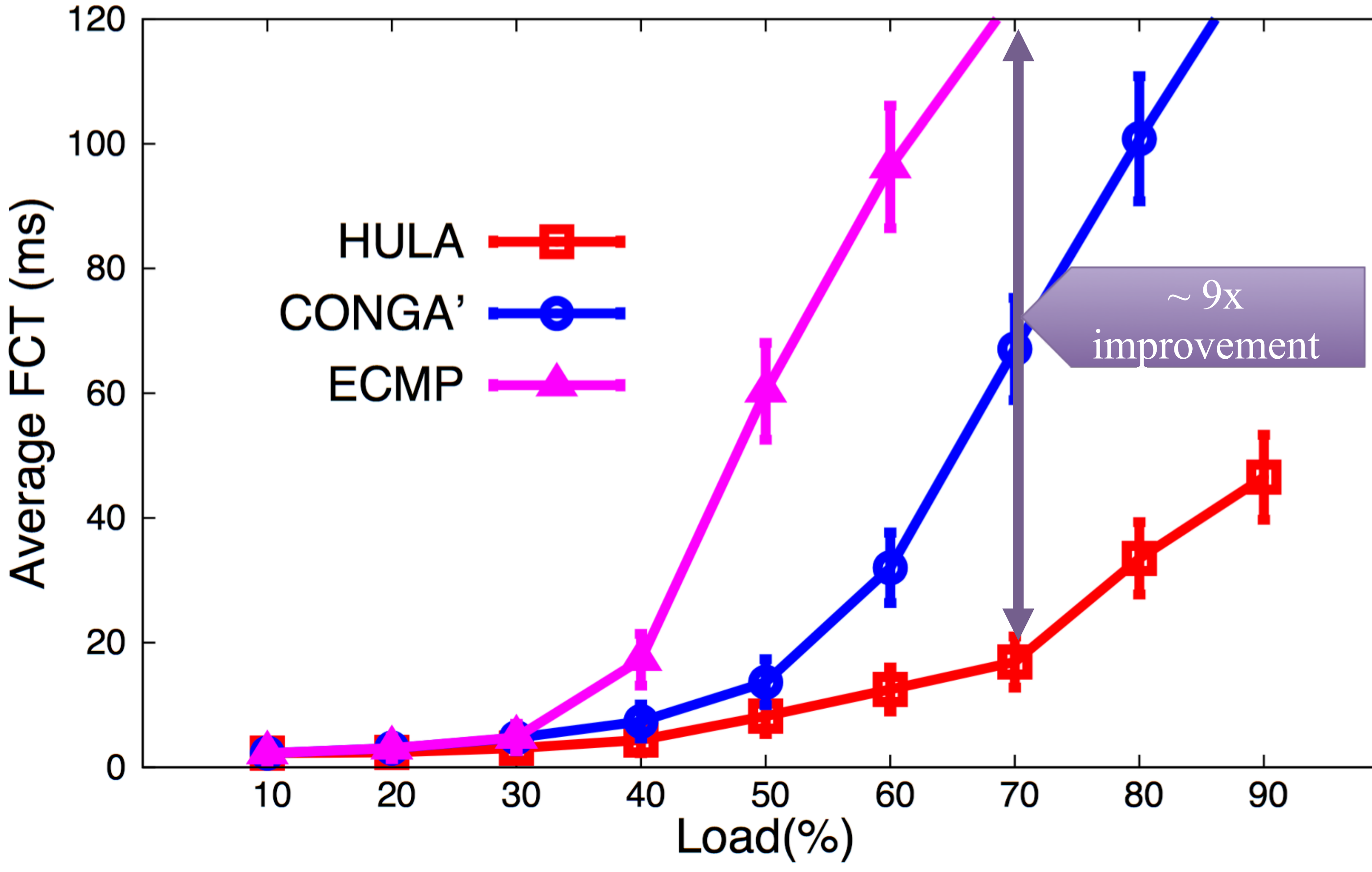
- NS2 packet-level simulator
- RPC-based workload generator
  - Empirical flow size distributions
  - Websearch and Datamining
- End-to-end metric
  - Average Flow Completion Time (FCT)

Compared with

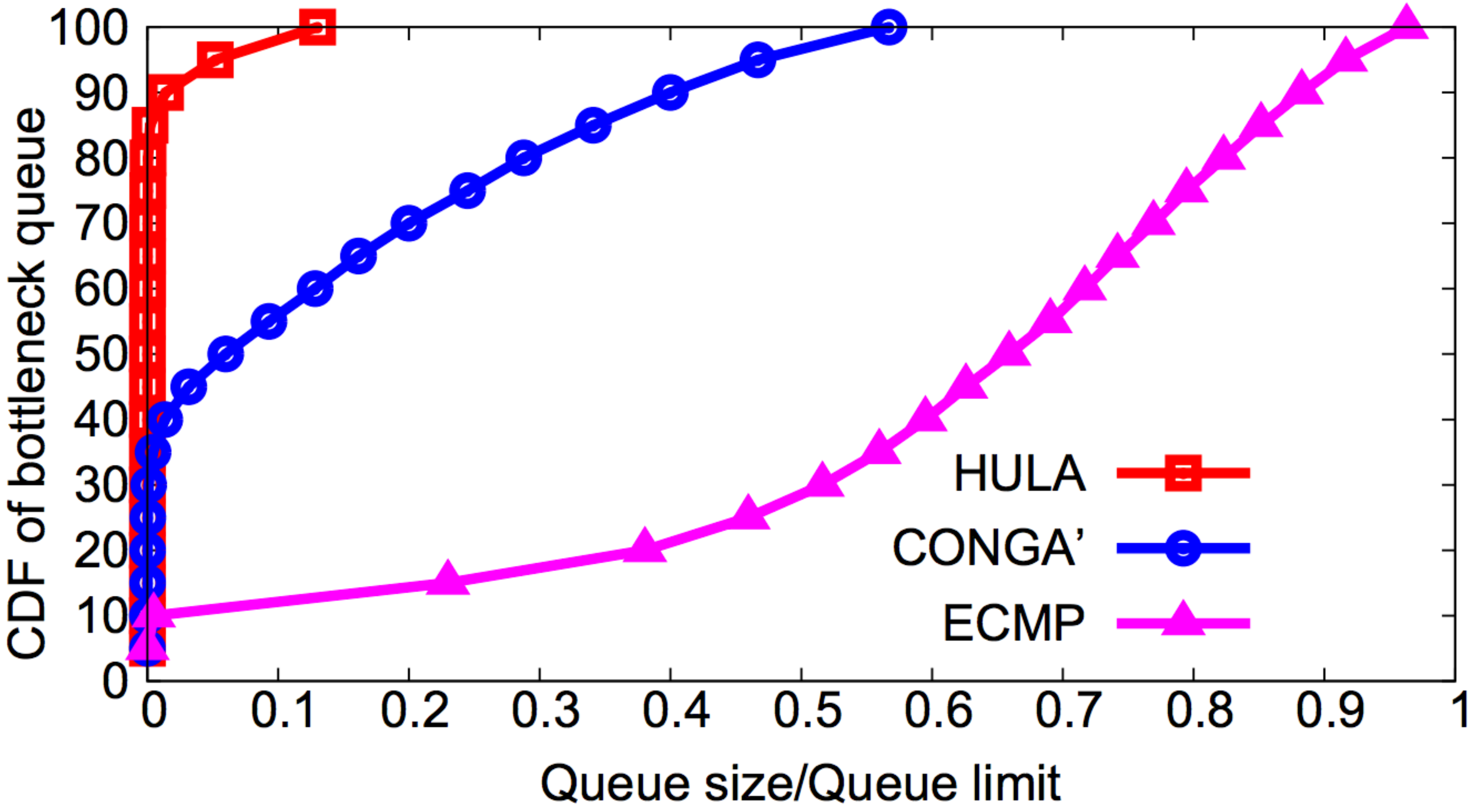
- ECMP
  - Flow level hashing at each switch
- CONGA'
  - CONGA within each leaf-spine pod
  - ECMP on flowlets for traffic across pods[1]

1. Based on communication with the authors

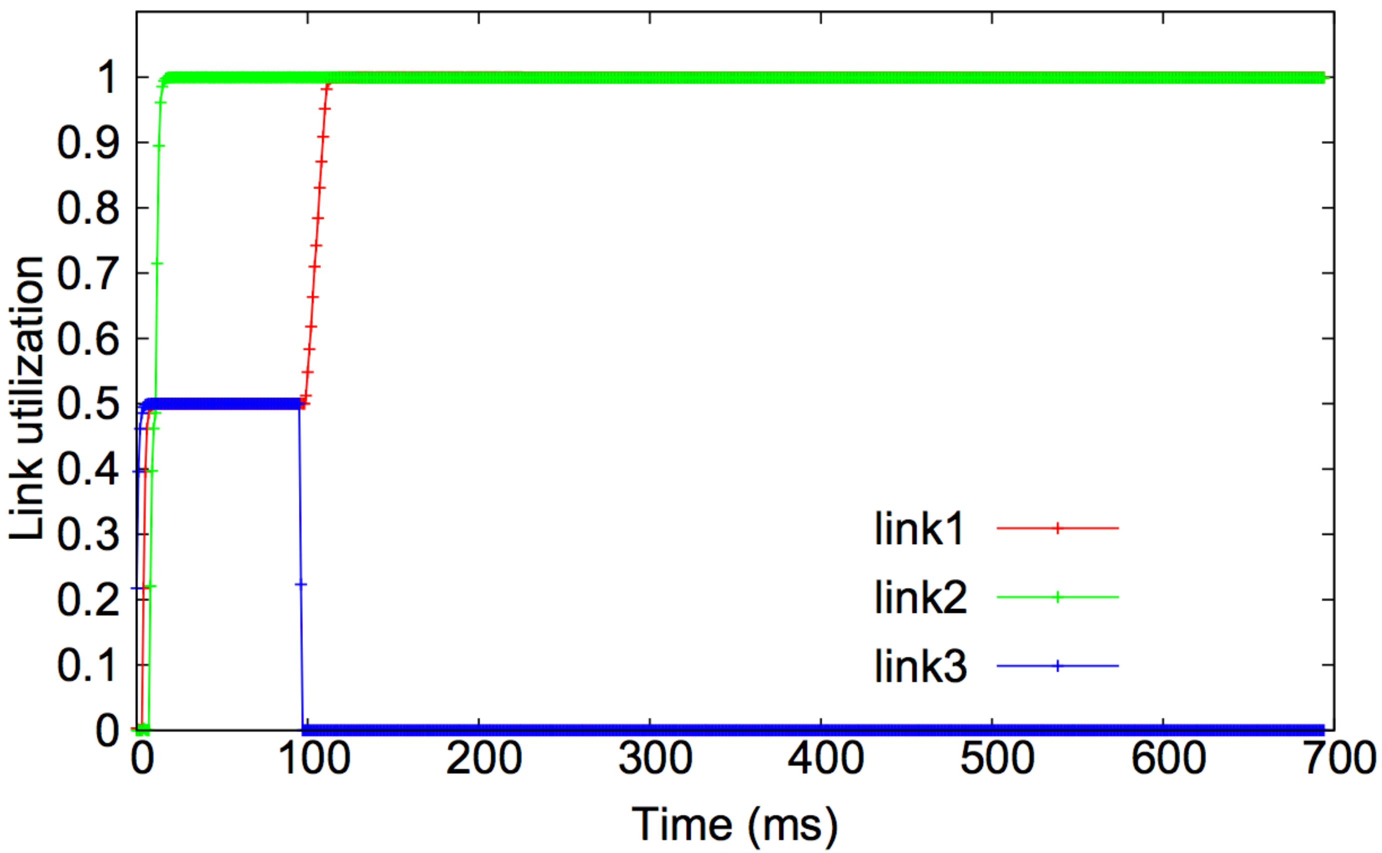
HULA handles high load much better



HULA keeps queue occupancy low



HULA is stable on link failure

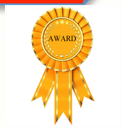


HULA: An Efficient Non-Blocking Switch

- **Scalable** to large topologies
- **Adaptive** to network congestion
- **Reliable** in the face of failures
- Bonus: **Programmable** in P4!

Research Contribution

- HULA (SOSR 16) — Efficiency
  - One big efficient non-blocking switch
- CacheFlow (SOSR 16) — Flexibility — Best Paper
  - A logical switch with infinite policy space
- Ravana (SOSR 15) — Reliability
  - Reliable logically centralized controller



Flexibility

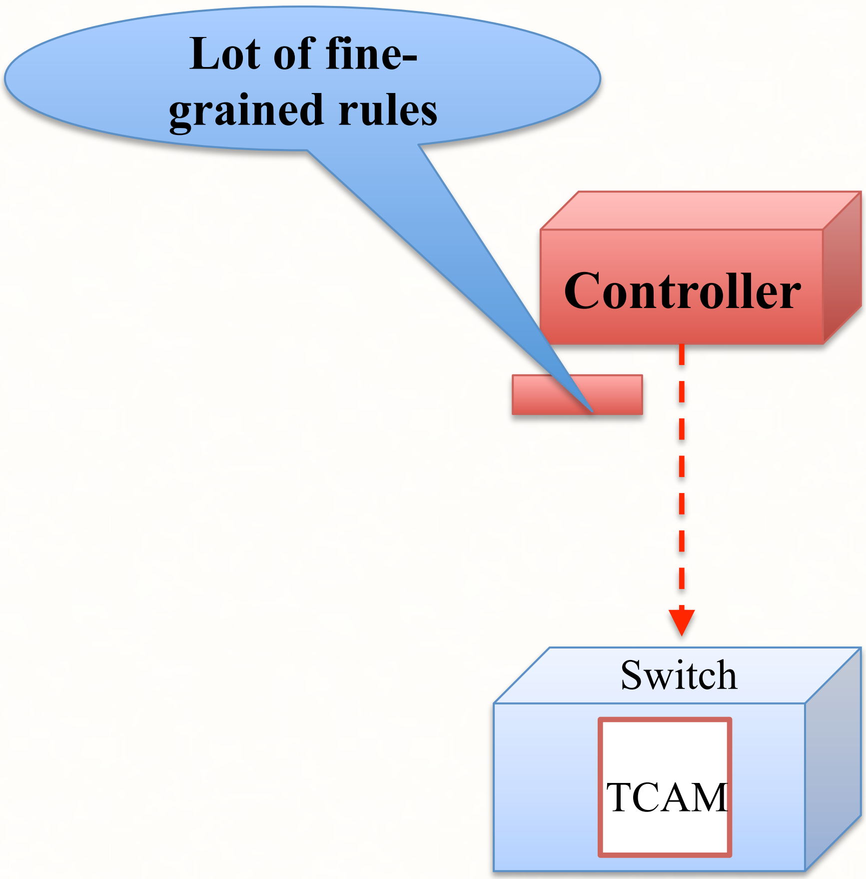
2. CacheFlow: Dependency-Aware Rule-Caching for Software-Defined Networks

Naga Katta

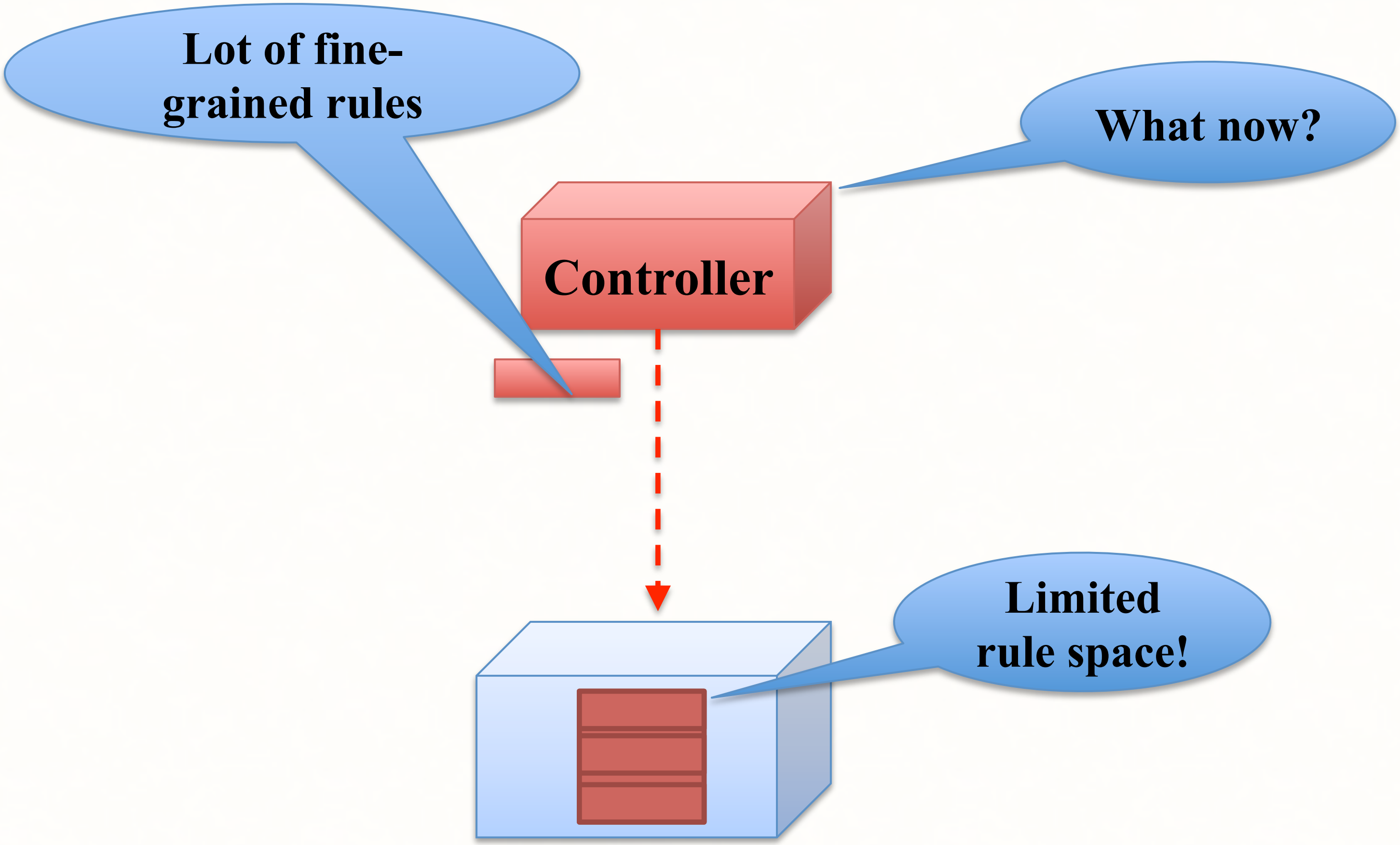
Omid Alipourfard, Jennifer Rexford, David Walker

Princeton University

SDN Promises Flexible Policies



SDN Promises Flexible Policies

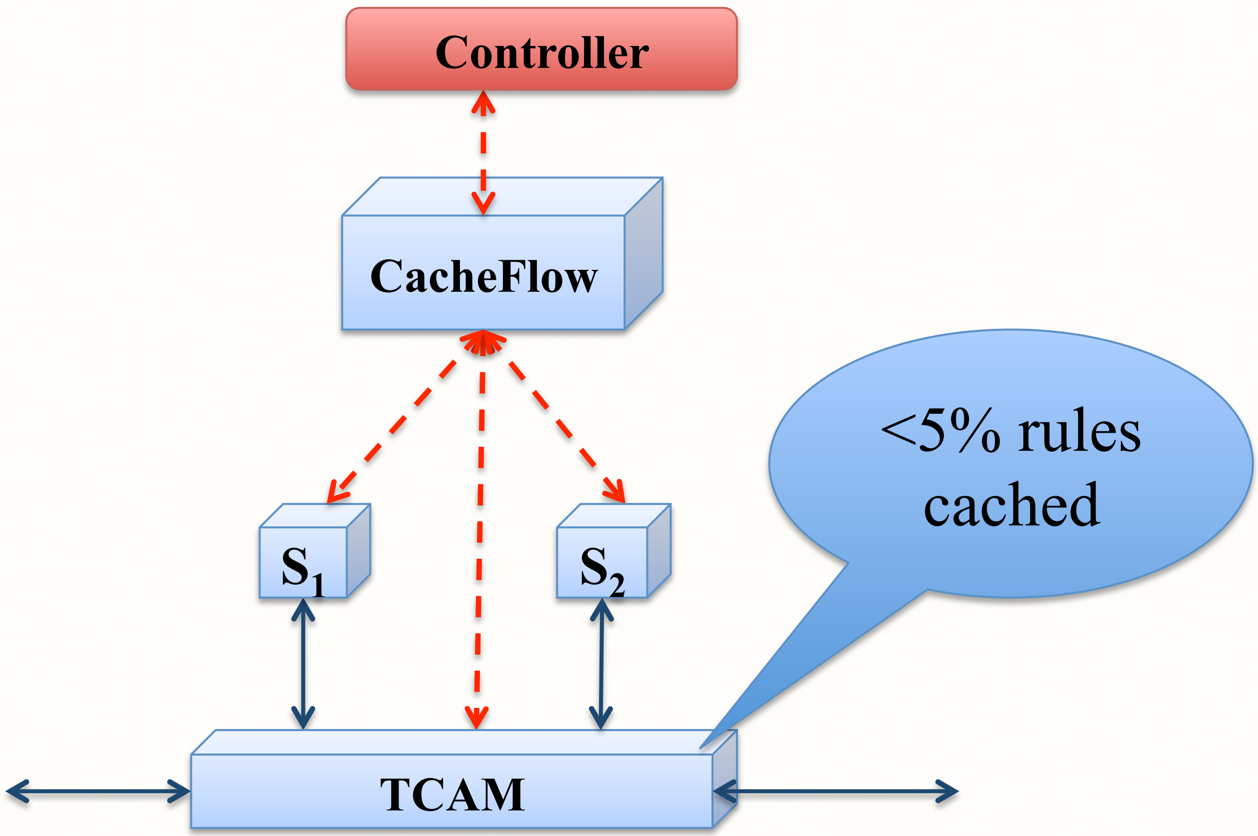


State of the Art

	Hardware Switch	Software Switch
Rule Capacity	Low (~2K-4K)	High
Lookup Throughput	High (>400Gbps)	Low (~40Gbps)
Port Density	High	Low
Cost	Expensive	Relatively cheap

TCAM as cache

- High throughput + high rule space



Caching Ternary Rules

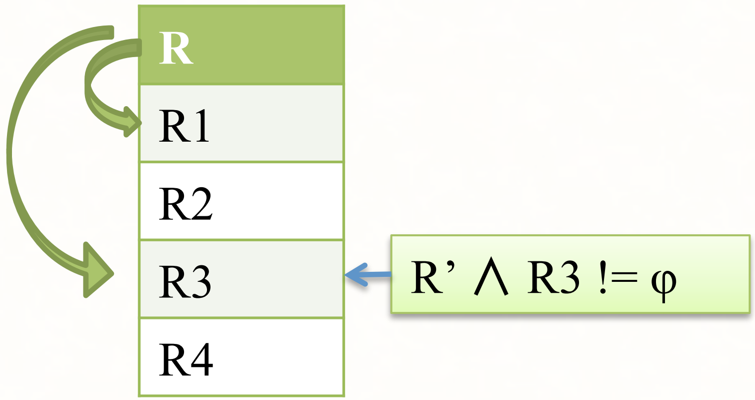
Rule	Match	Action	Priority	Traffic
R1	11*	Fwd 1	3	10
R2	1*0	Fwd 2	2	60
R3	10*	Fwd 3	1	30

Partial Overlaps!

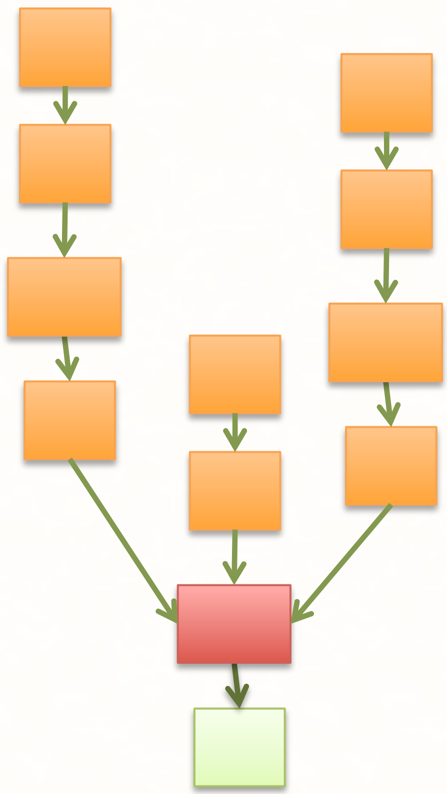
- Greedy strategy breaks rule-table semantics

The dependency graph

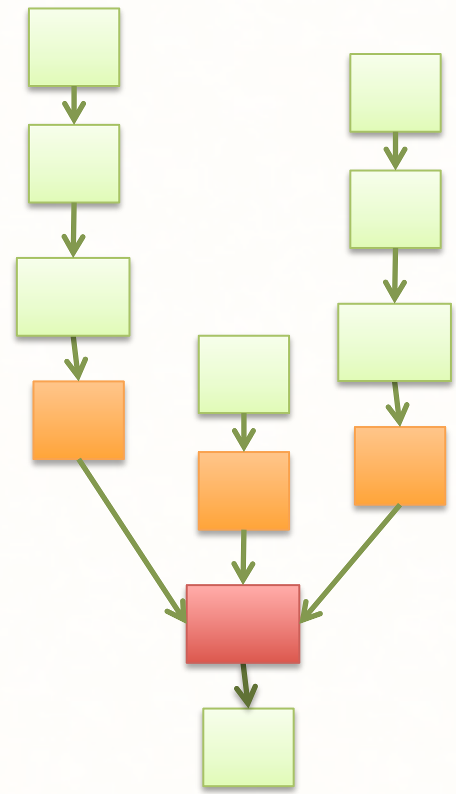
- For a given rule R
  - **Find all the rules that its packets may hit if R is removed**



Splice Dependents for Efficiency



Dependent-Set



Cover-Set

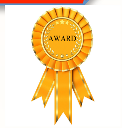
Rule Space Cost

CacheFlow: Enforcing Flexible Policies

- A switch with logically infinite policy space
  - Dependency analysis for correctness
  - Splicing dependency chains for Efficiency
  - Transparent design

Research Contribution

- HULA (SOSR 16) — Efficiency
  - One big efficient non-blocking switch
- CacheFlow (SOSR 16) — Flexibility — Best Paper
  - A logical switch with infinite policy space



- Ravana (SOSR 15) — Reliability
  - Reliable logically centralized controller

Reliability

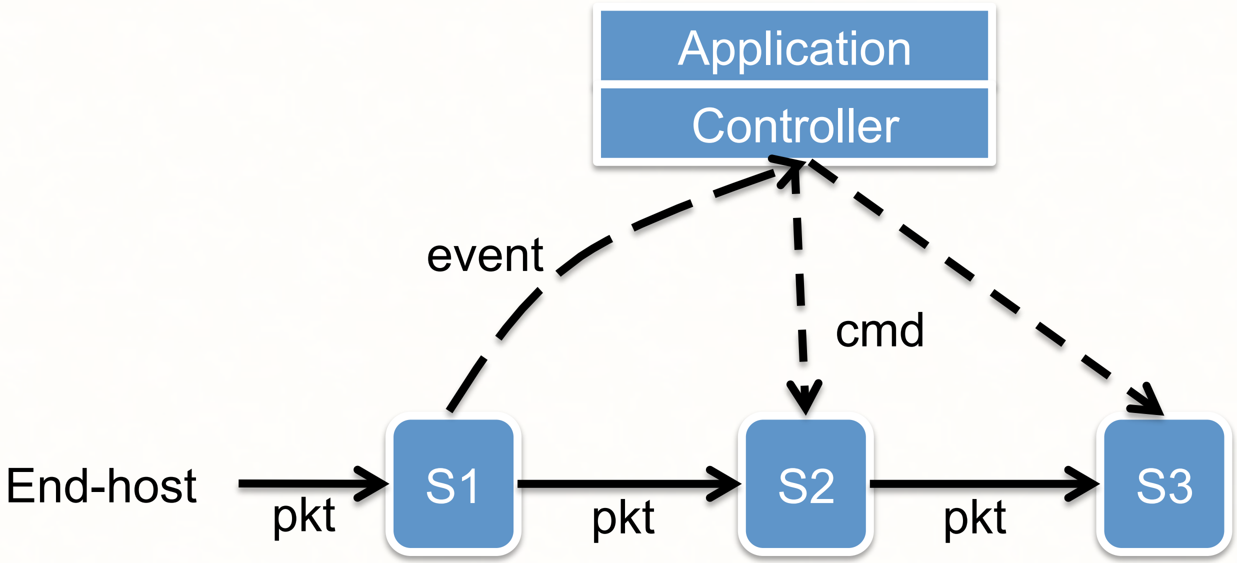
3. Ravana: Controller Fault-Tolerance in Software-Defined Networking

Naga Katta

Haoyu Zhang, Michael Freedman, Jennifer Rexford

PRINCETON UNIVERSITY

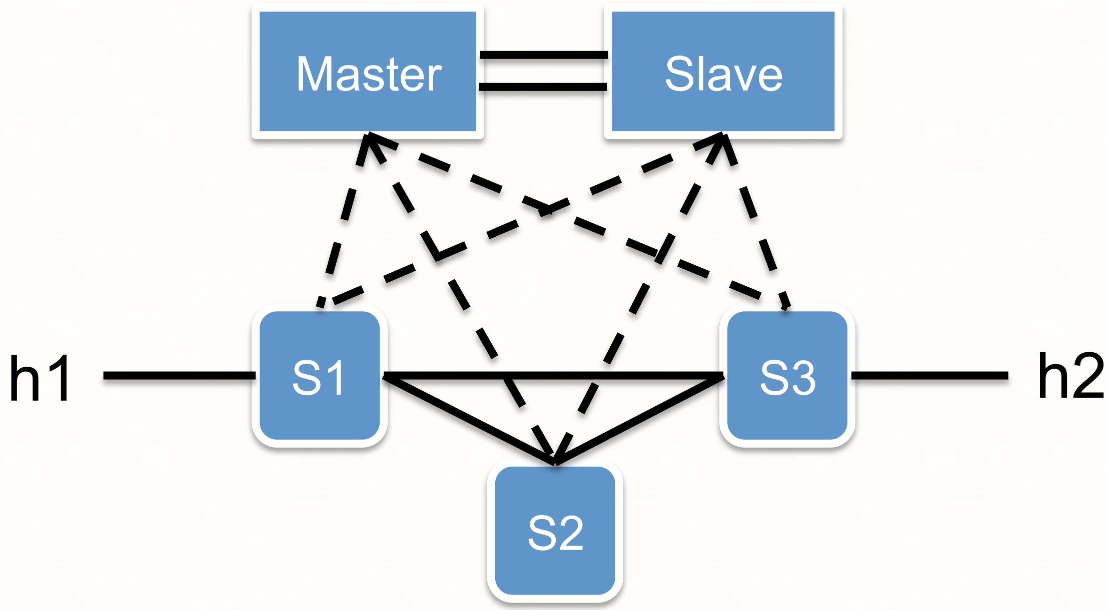
SDN controller: single point of failure



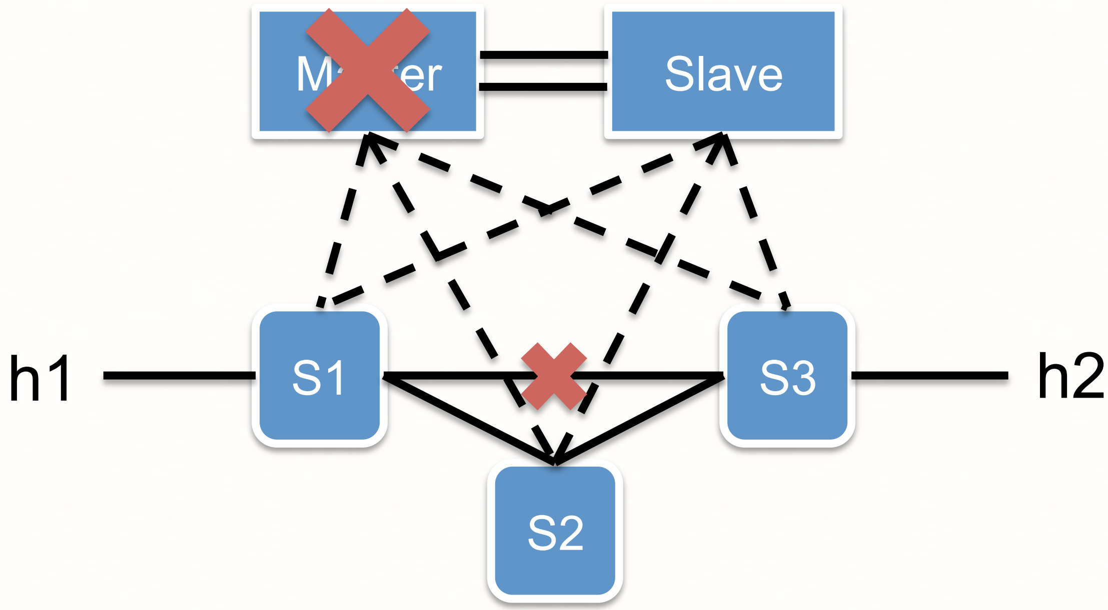
Failure leads to

- Service disruption
- Incorrect network behavior

Replicate Controller State?

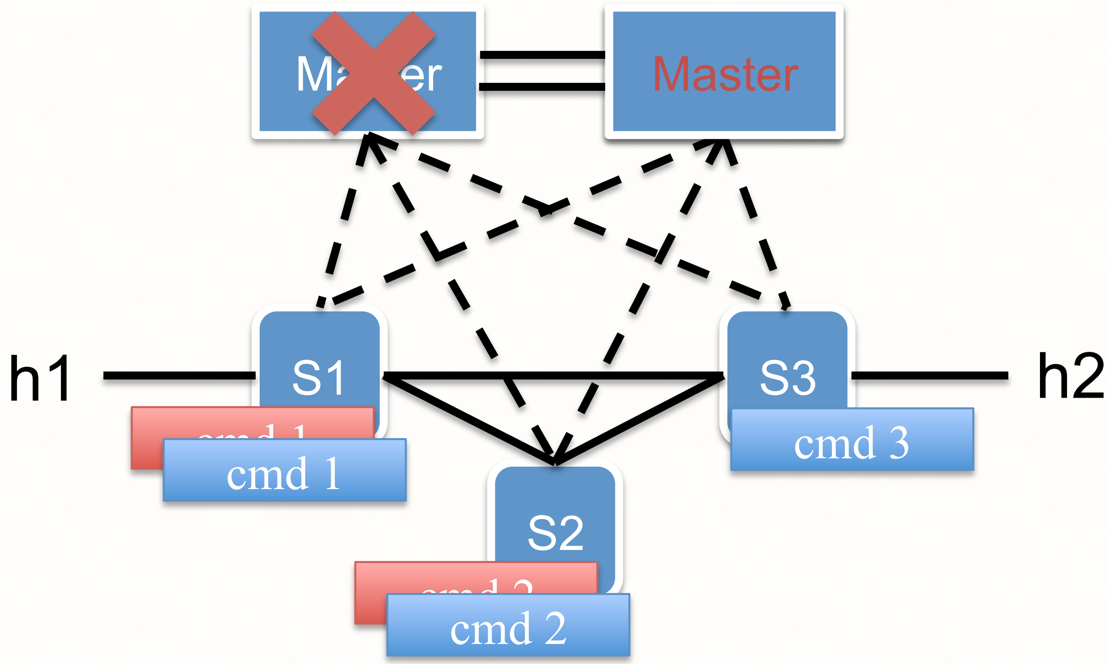


State External to Controllers: Events



- During master failover…
- Linkdown event is generated
  → event loss!

State External to Controllers: Commands

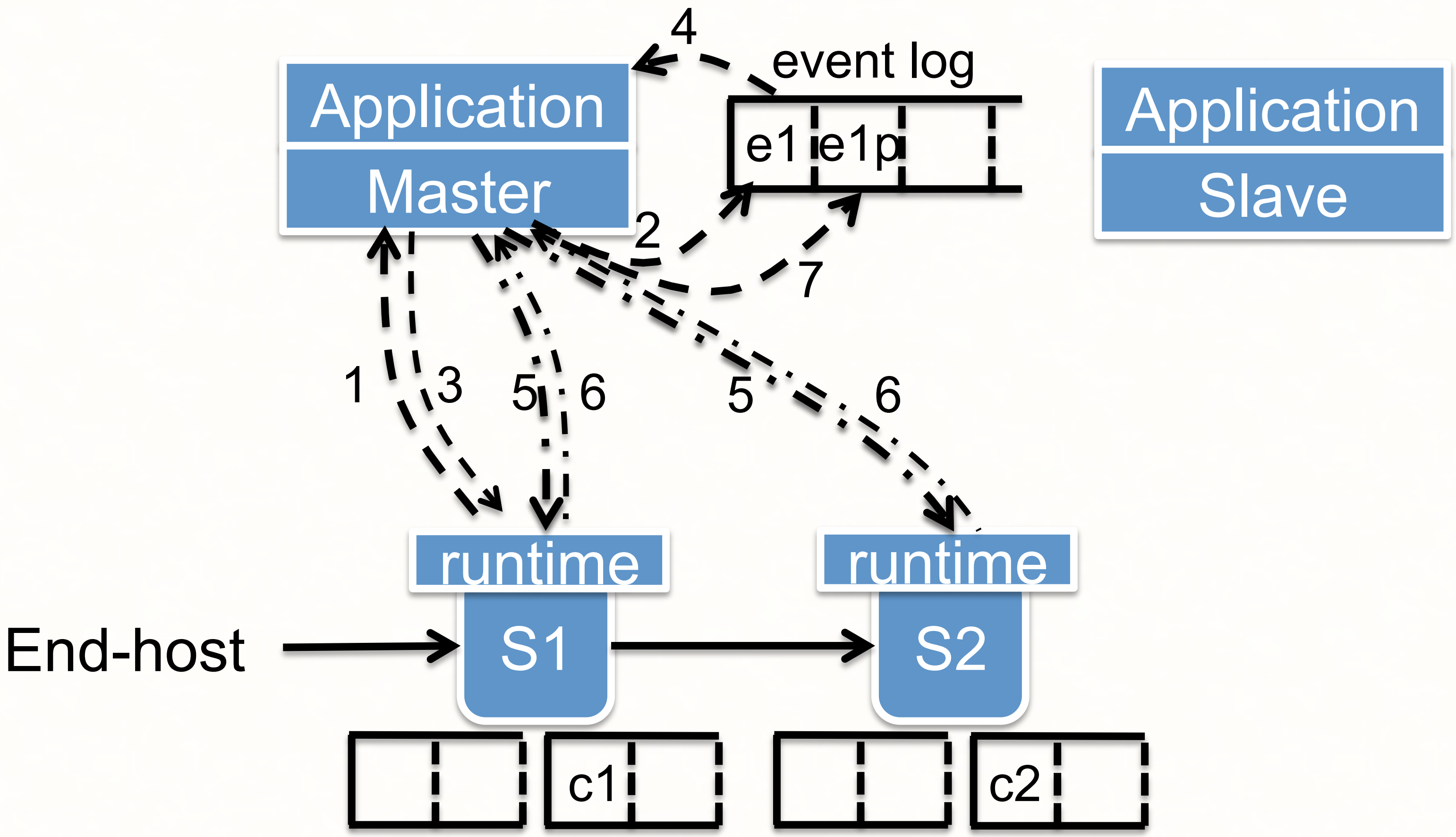


- Master crashes while sending commands…
- New master will process and send commands again → repeated commands!

Ravana: A Fault-Tolerant Control Protocol

- Goal: Ordered Event Transactions
  - Exactly-once events
  - Totally ordered events
  - Exactly once commands
- Two stage replication protocol
  - Enhances RSM
  - Acknowledgements, Retransmission, Filtering

Exactly Once Event Processing

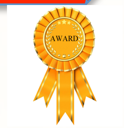


Conclusion

- **Reliable** control plane
- **Efficient** runtime
- **Transparent** programming abstraction

Research Contribution

- HULA (SOSR 16) — Efficiency
  - One big efficient non-blocking switch
- CacheFlow (SOSR 16) — Flexibility — Best Paper
  - A logical switch with infinite policy space
- Ravana (SOSR 15) — Reliability
  - Reliable logically centralized controller



Other Work

- Flog: Logic Programming for Controllers
  - XLDI 2012

  Control plane
- Incremental Consistent Updates
  - HotSDN 2014

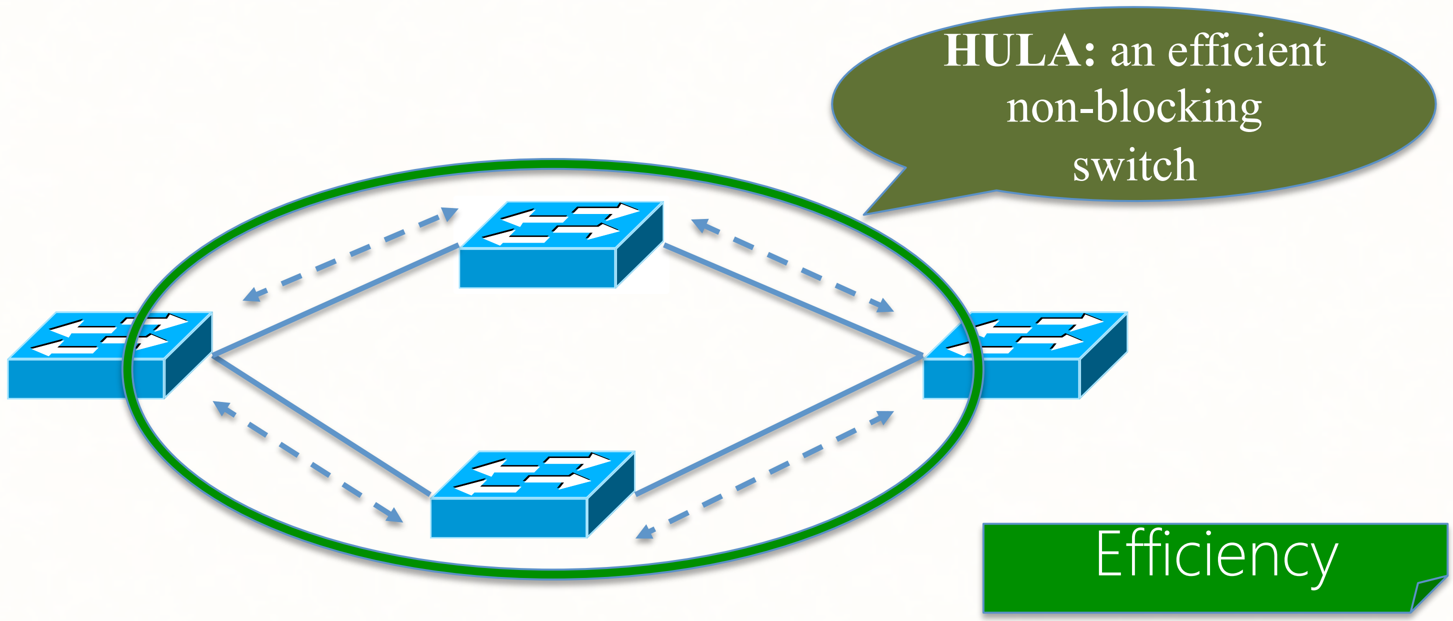
  Middle layer
- In-band Network Telemetry
  - SIGCOMM Demo 2015

  Data plane
- Edge-Based Load-Balancing
  - To appear in HotNets 2016

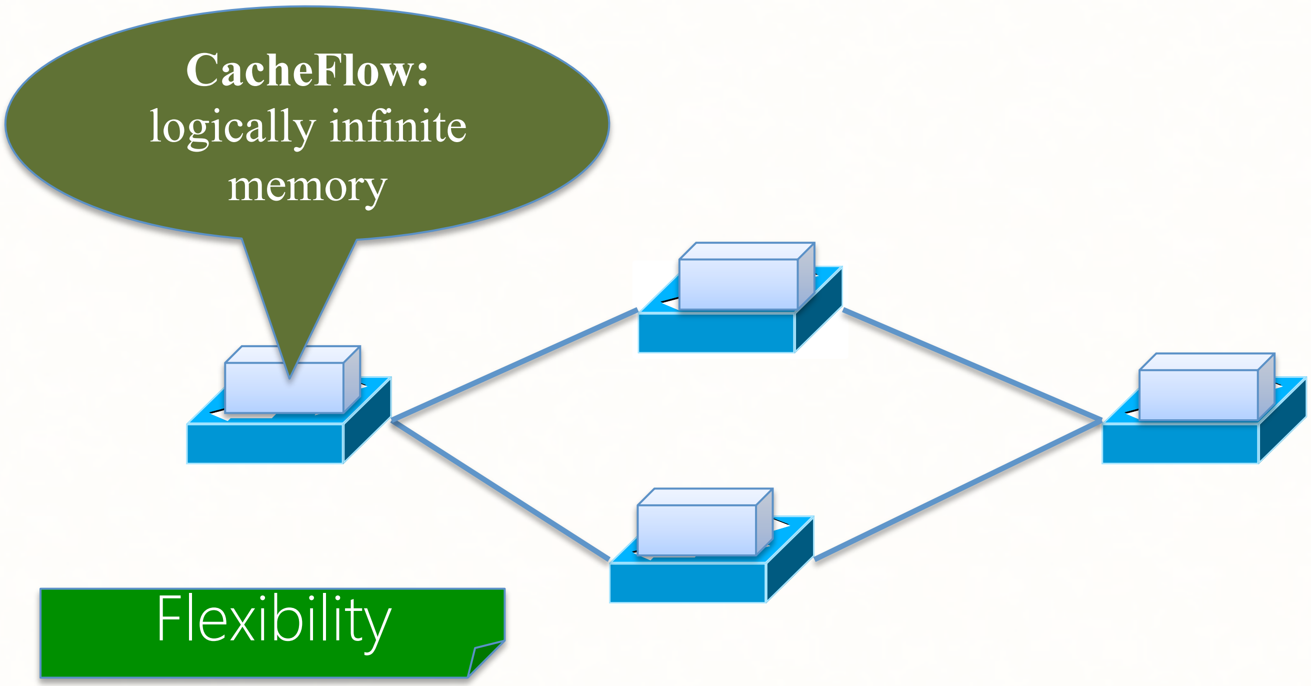
  Data plane

Thesis: Summary

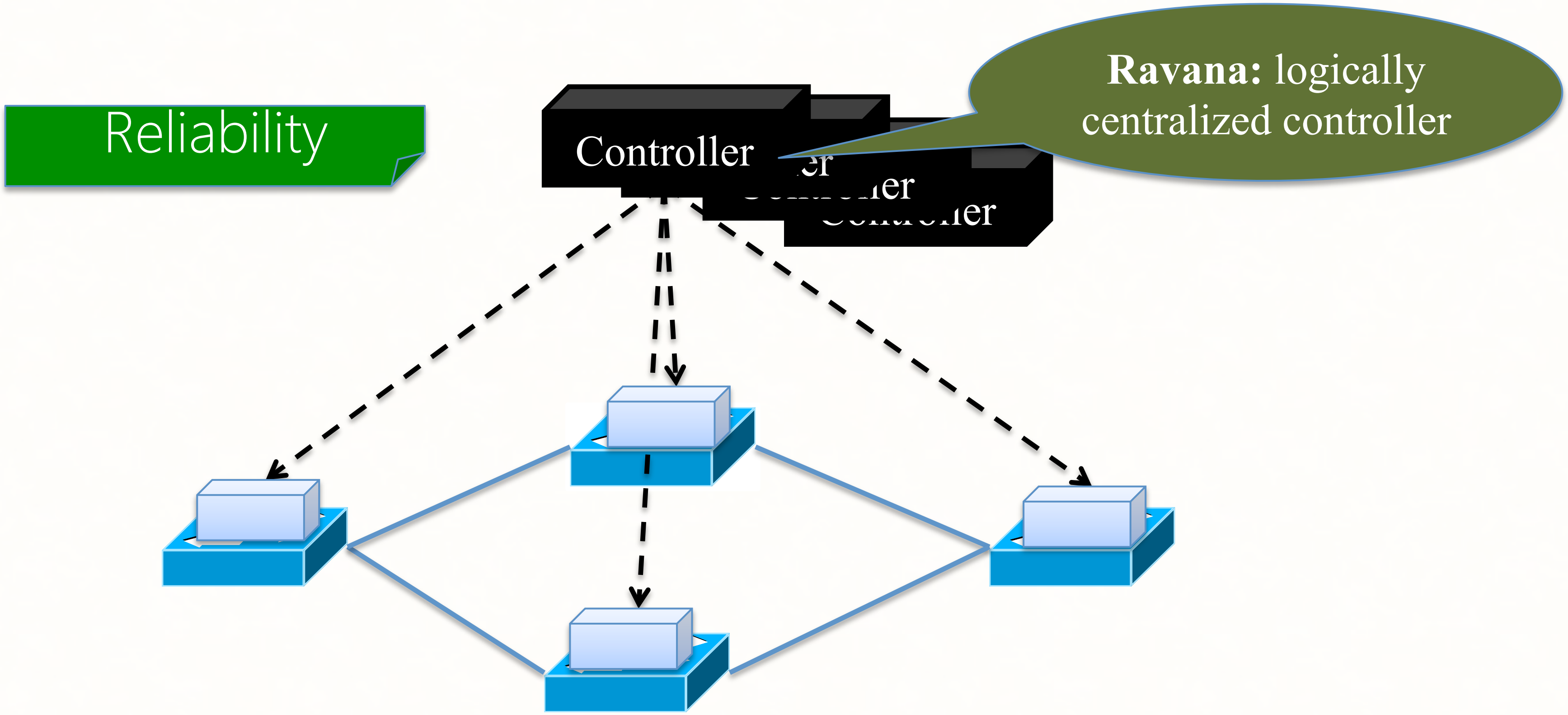
Thesis: Summary



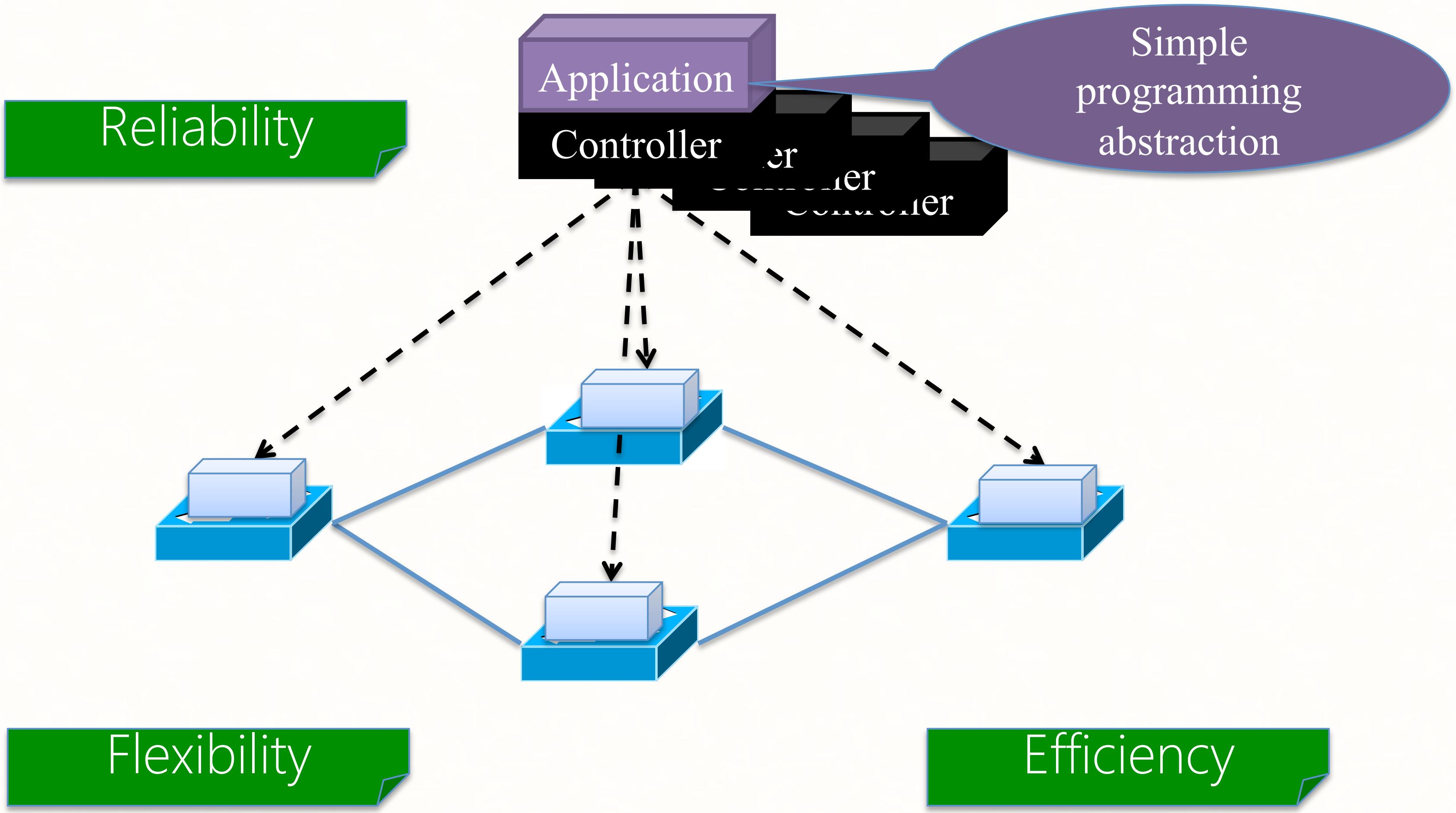
Thesis: Summary



Thesis: Summary



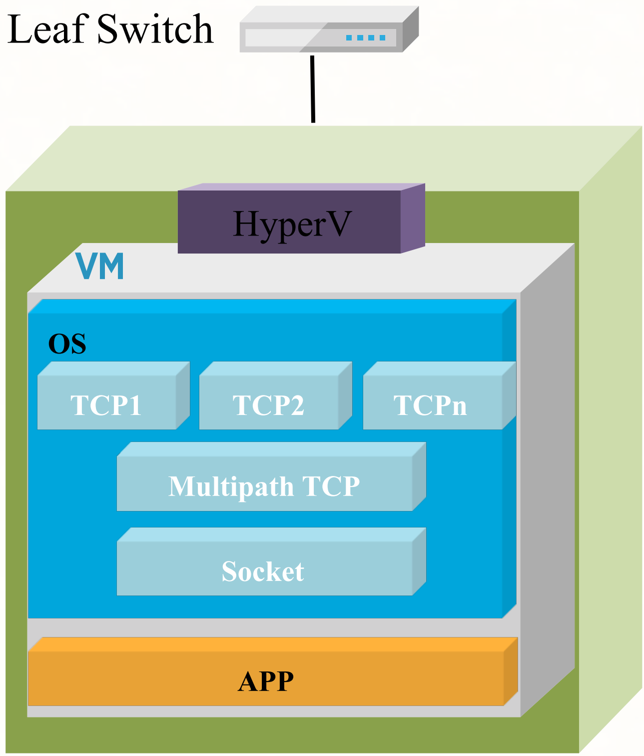
A Desirable SDN



Thank You!

Backup slides

Transport Layer (MPTCP)



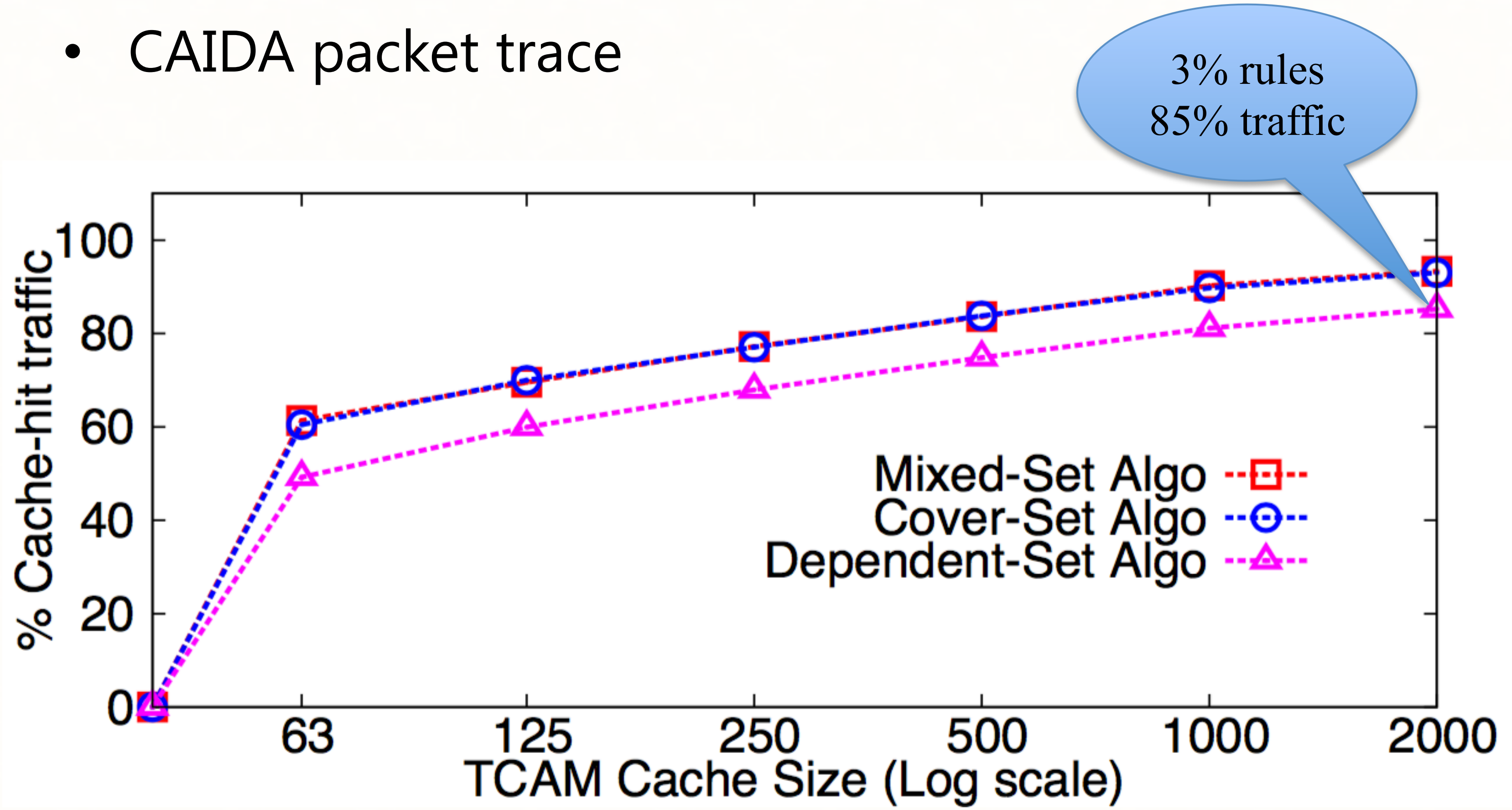
Guest VM Changes

HULA: Scalable, Adaptable, Programmable

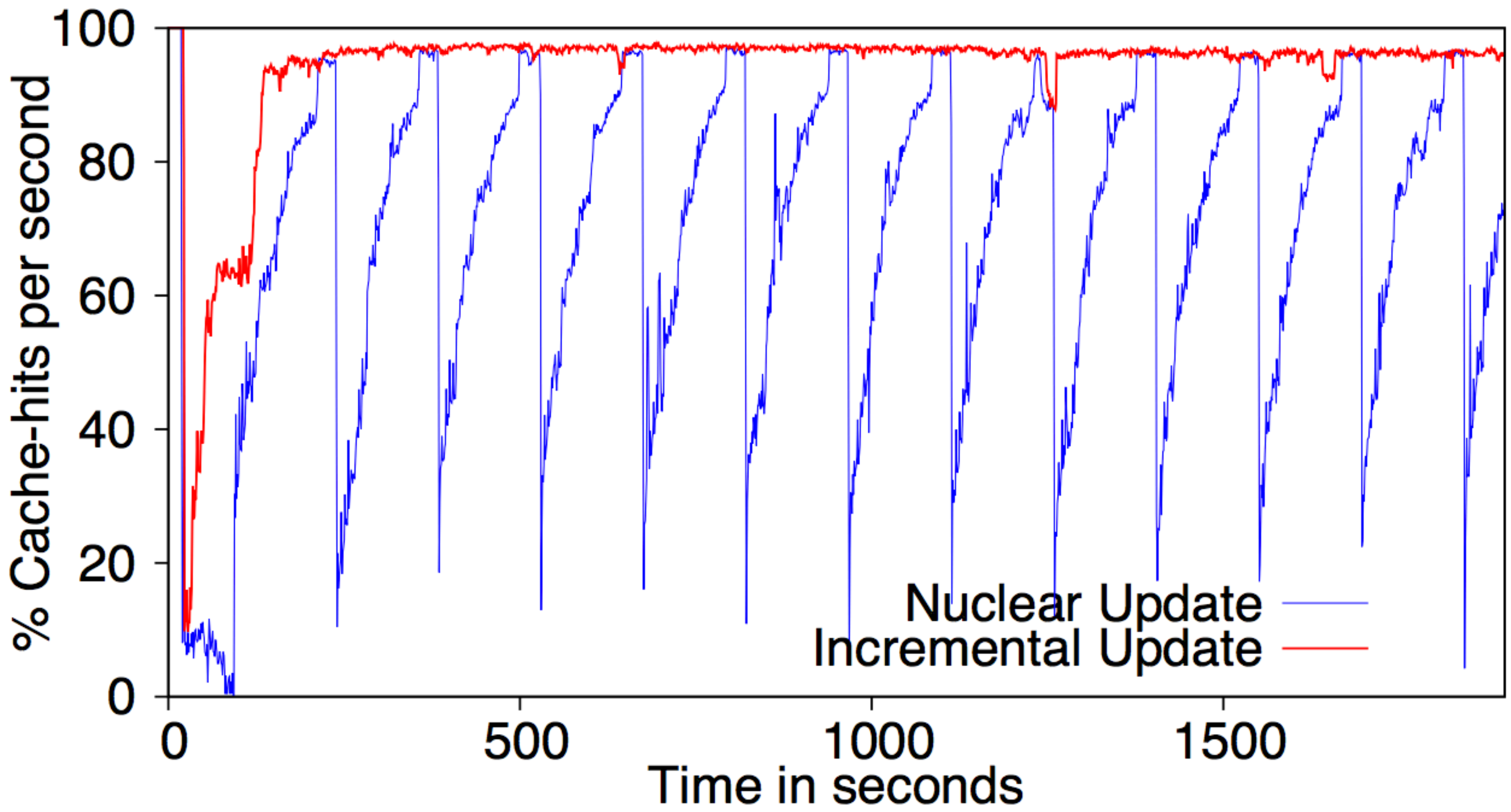
LB Scheme	Congestion aware	Application agnostic	Dataplane timescale	Scalable	Programmable dataplanes
ECMP					
SWAN, B4					
MPTCP					
CONGA					
HULA					

Dependency Chains – Clear Gain

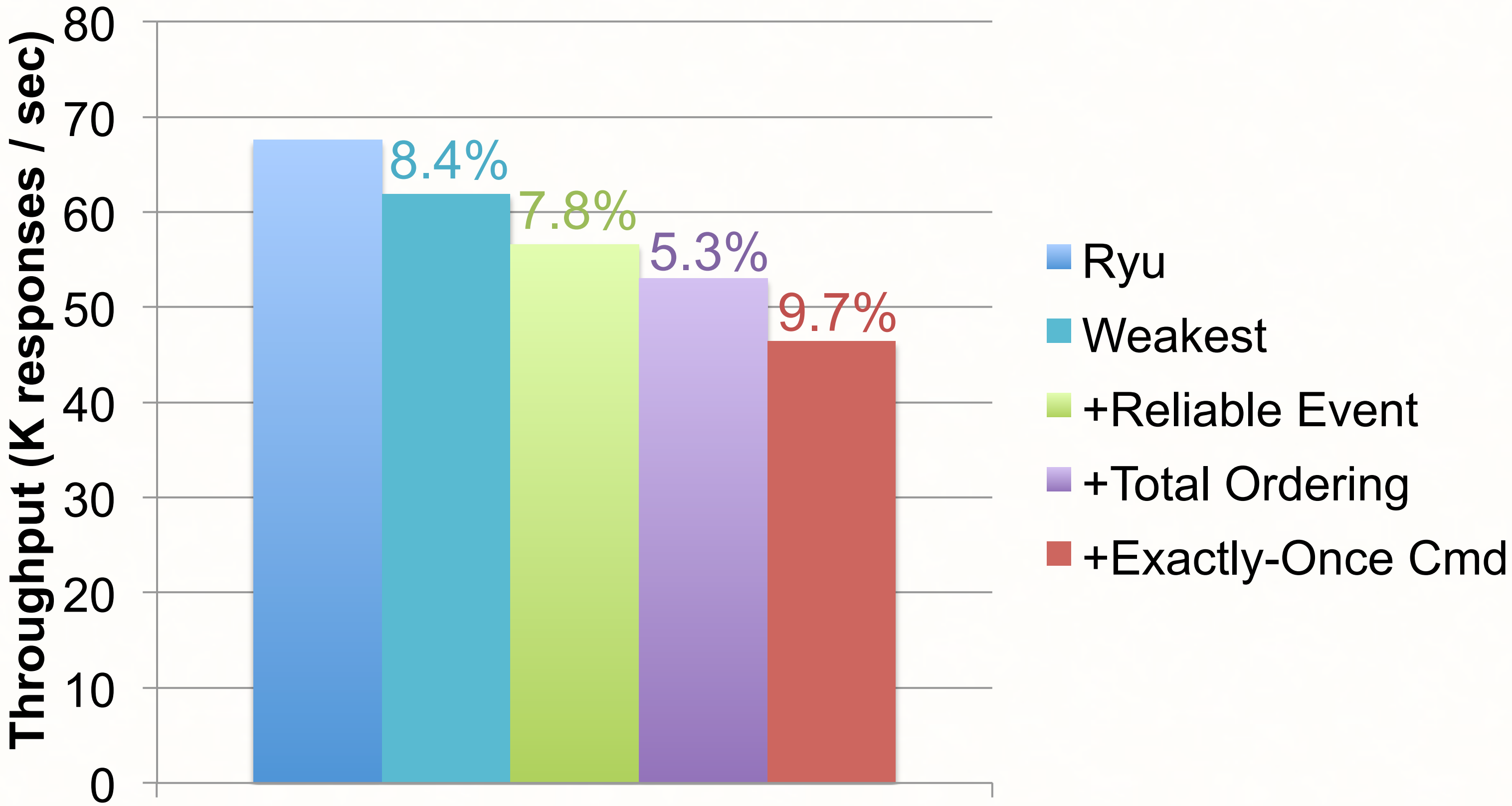
- CAIDA packet trace



Incremental update is more stable

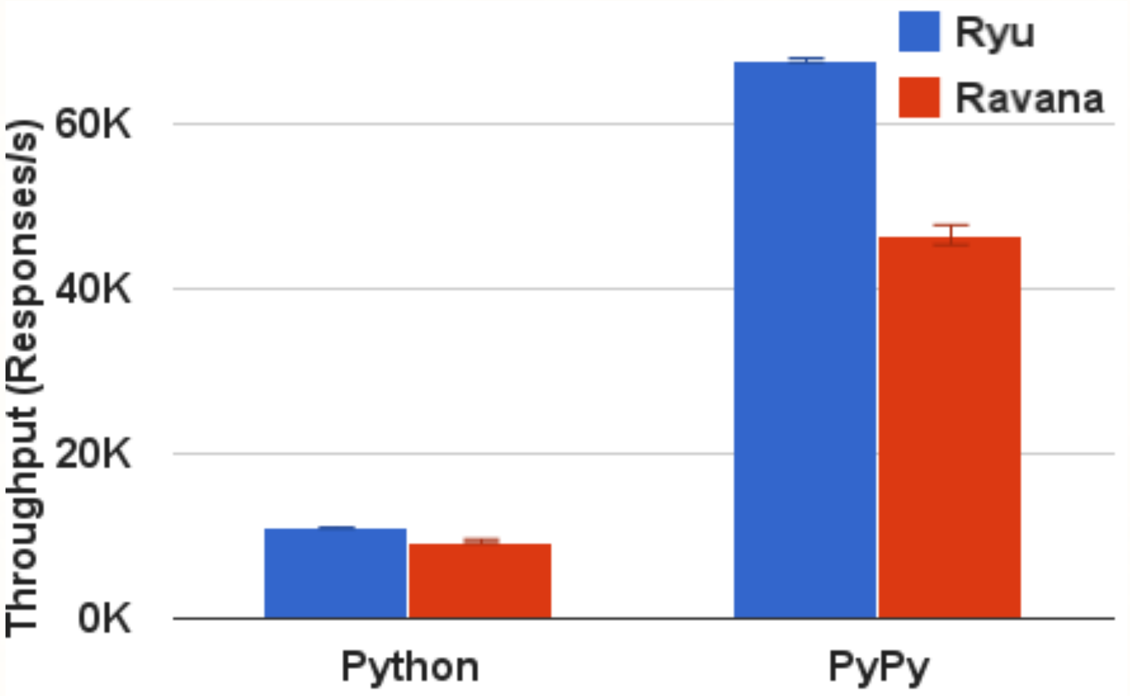


What causes the overhead?



- Factor analysis: overhead for each component

Ravana Throughput Overhead



- Measured with cbench test suite
- Event-processing throughput: 31.4% overhead

Controller Failover Time