HEDERA

DYNAMIC FLOW SCHEDULING FOR DATA CENTER NETWORKS

Authors: Mohammad Al-Fares, Sivasankar Radhakrishnan,

Barath Raghavan, Nelson Huang, Amin Vahdat [2010]

Presenter

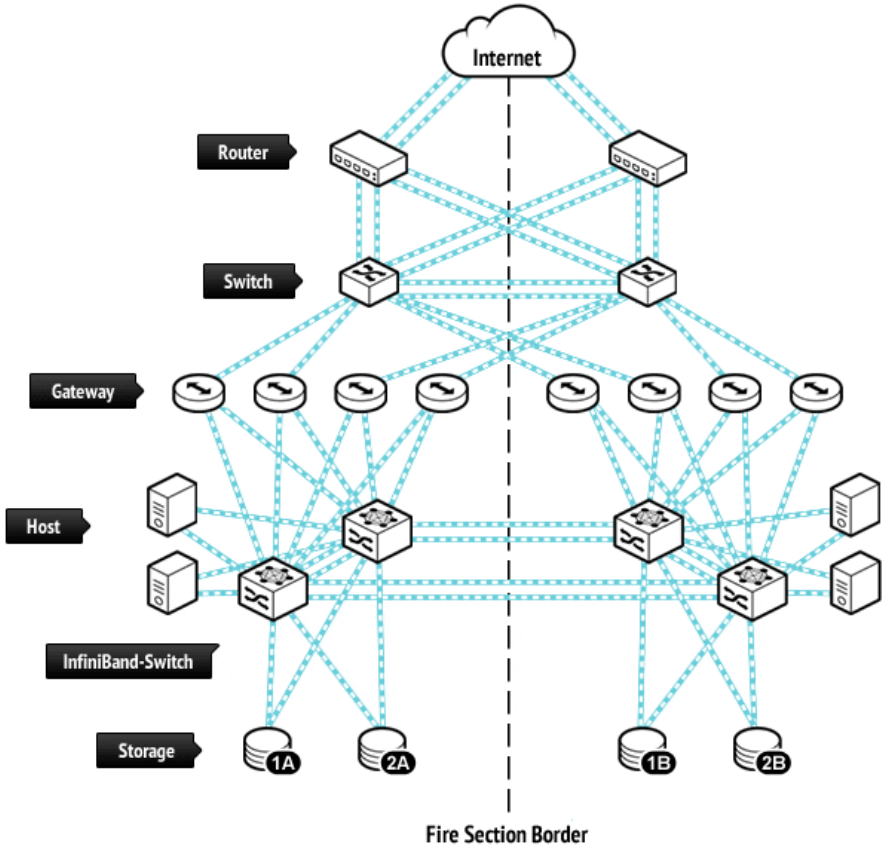
Baran Kaya

OUTLINE

- Introduction
- Data Center Networking
- Fat Tree network architecture
- Static Load Balancers → ECMP and VLB
- What is Hedera?
- Hedera Algorithms
- Simulation and Test
- Results and Evaluation
- SDN
- Conclusion

INTRODUCTION

- Datacenters with thousands of machines
- High bandwidth connections
- Multi-rooted tree networks
- Multi-pathing → Static hashing
- Hedera → Dynamic flow scheduling
- 8192 hosts → 96% optimal and 113% better load balancing
- Non-blocking switch for comparison (support all ports at full forwarding capacity)

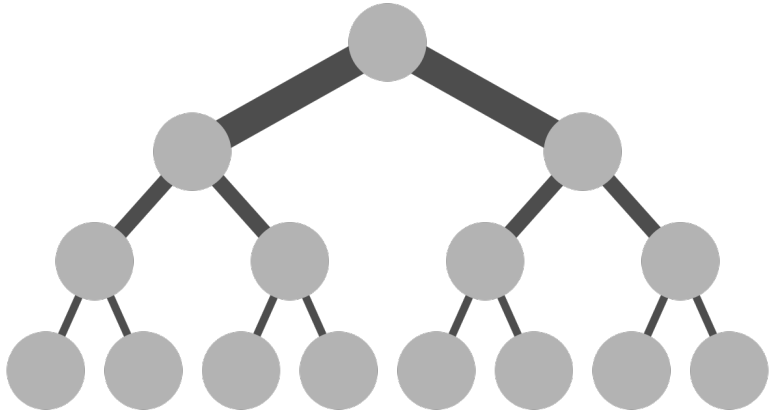


DATACENTER NETWORKING

- Why DC network design is difficult?
- Unknown workloads (time & space)
- Customers applications run on commodity OS, requires high bandwidth
- Virtualization: application may not run on the same rack
- Multi-rooted tree network with higher-speed links
- There are many paths between all hosts
- Minimize flow collision → Higher aggregate bandwidth
- Static forwarding → Single path for the whole flow

FAT-TREE NETWORKS

- Tree data structure → Every branch has the same thickness
- Fat tree → Top branches thicker



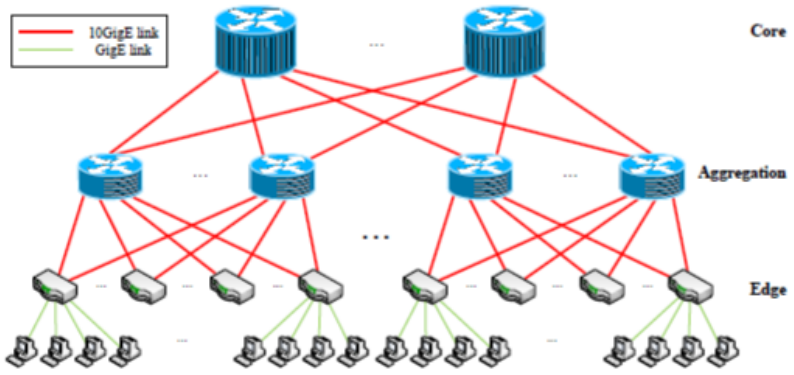
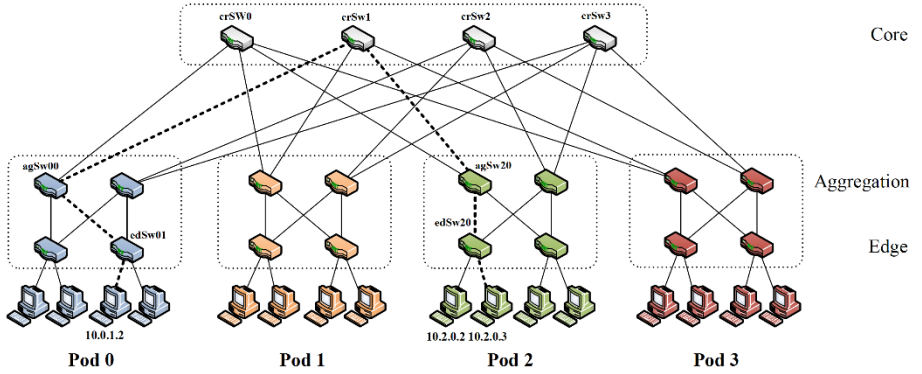
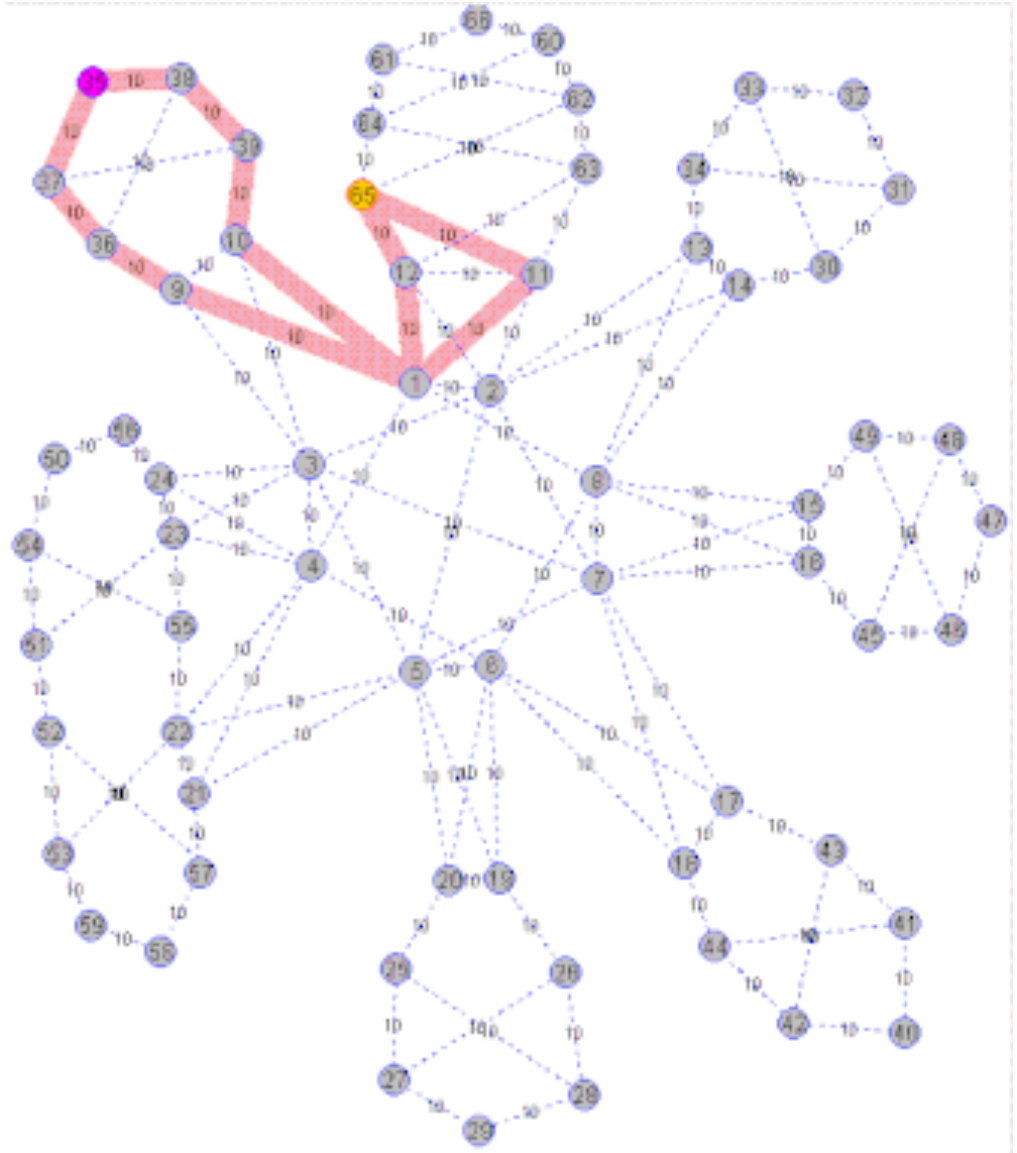


Figure 1: A common multi-rooted hierarchical tree.



ECMP & VLB

- Equal Cost Multipath (ECMP)
- Hash based
- Header data (source, destination)
- Valiant Load Balancing (VLB)
- Random “core” switch
- Per-flow → Similar to ECMP
- Per-packet → Requires packet ordering
- Both are static load balancers



ECMP animation using 802.1aq protocol

ECMP

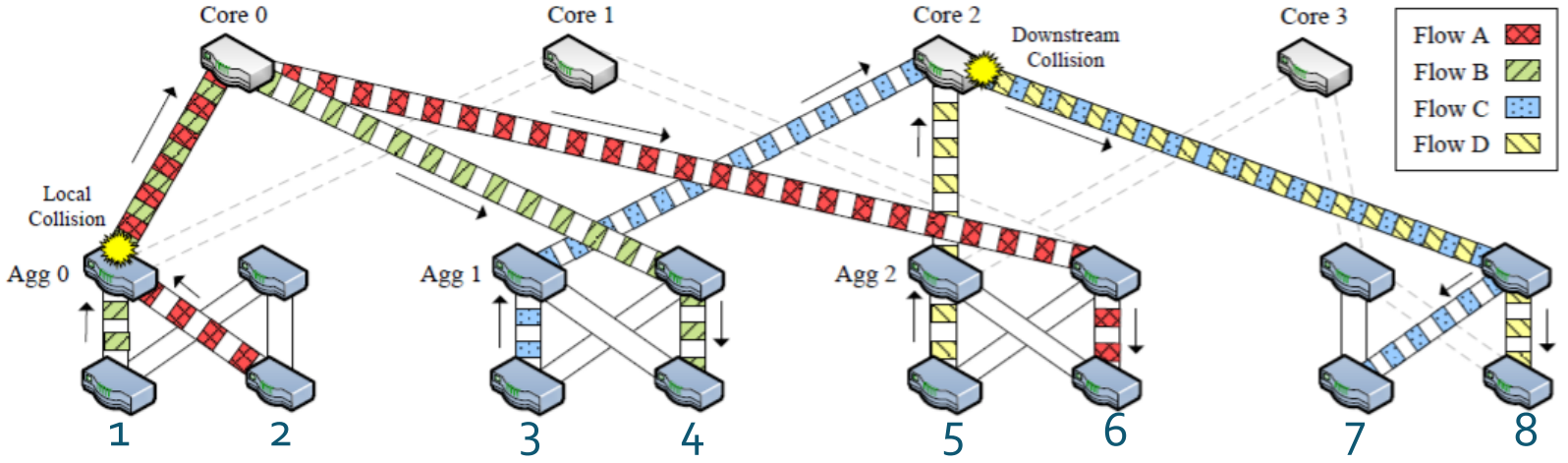


Figure 2: Examples of ECMP collisions resulting in reduced bisection bandwidth. Unused links omitted for clarity.

- 2 or more large long-lived flows can collide → Bottleneck
- A and B collision at Agg 0, C and D collision at Core 2
- 1 Gbps → 500 Mbps
- A to Core 1, D to Core 3

ECMP

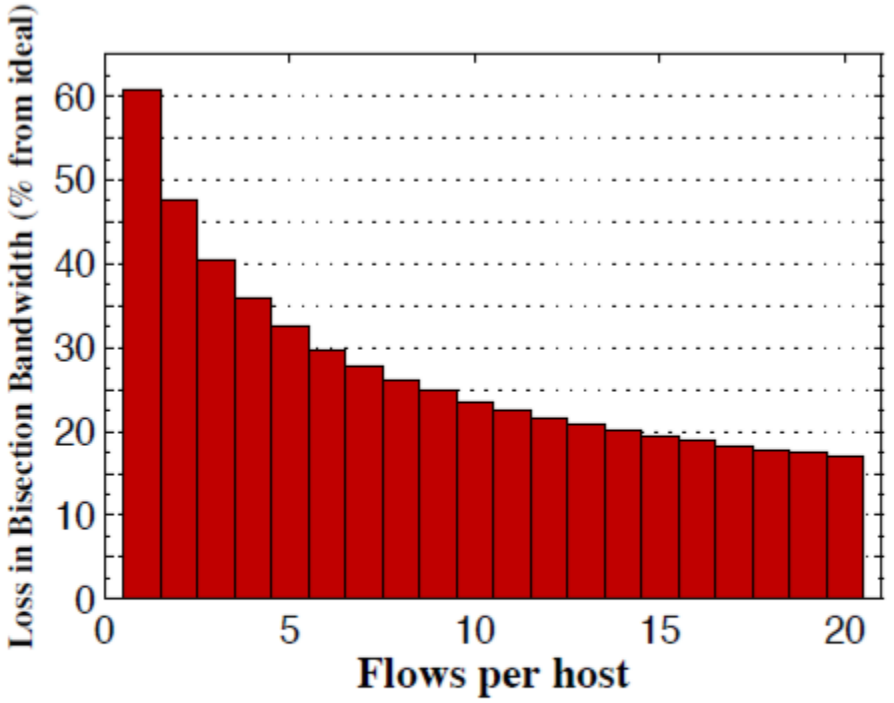


Figure 3: Example of ECMP bisection bandwidth losses vs. number of TCP flows per host for a k=48 fat-tree.

- 3 stage fat-tree with ECMP (Flows hashed)
- 1GigE 48-port switches, 27k hosts
- Each host transfers equal amount of data to all remote hosts one at a time
- Hash collision → Reduce bisection bandwidth with → Average of 60.8%
- Each host communicate ~~the~~ remote hosts in ~~a~~ parallel across 1000 simultaneous flows
- Hash collision bandwidth → 2.5%

HEDERA

- With efficient forwarding protocol, multi-rooted tree topology can deliver high bisection bandwidth.
- Move flows from highly utilized links to less utilized links
- Which flow?
- Measure flows' bandwidth and transfer the flow that the alternate ~~flow~~ can carry.
- Flow's bandwidth may not reflect the natural demand.
- 1. Detect large flows at the edge switches
- 2. Estimate natural demand and compute better path
- 3. Install new paths to edge switches

HEDERA

- Spread the traffic as evenly as possible among all core switches.
- In Hedera packet path is non-deterministic
- Chosen on its way up to the core
- But deterministic → Going down
- There is exactly one path from any core switch to any destination host
- Fixed IP address in core switches to any destination pods

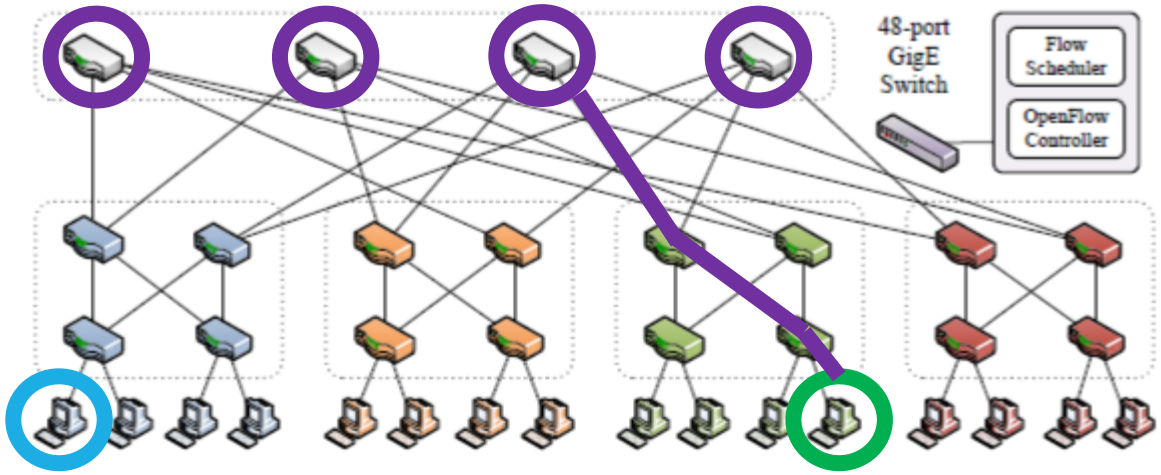


Figure 8: System Architecture. The interconnect shows the data-plane network, with GigE links throughout.

HEDERA

- New flow → Use hash based forwarding (similar to ECMP)
- Use this path until flow grows past a threshold rate
- After threshold → Hedera dynamically calculates alternative path (placement)
- Threshold for test & simulation → 100Mbps (10% of 1GigE link)

HEDERA

- Hedera has a central scheduler for fail-overs and scalability
- Dynamically changes the edge and aggregation switches forwarding tables
- Uses current network-wide communication demand data
- Uses one of the scheduling algorithms:
  - Global First Fit
  - Simulated Annealing
- Both algorithms searches new paths to increase bisection bandwidth
- Paper compares these 2 algorithms

HEDERA ALGORITHMS

- Flows: network-limited (bandwidth) or host-limited (RAM/disk)
- Demand estimation algorithm → O(|F|)

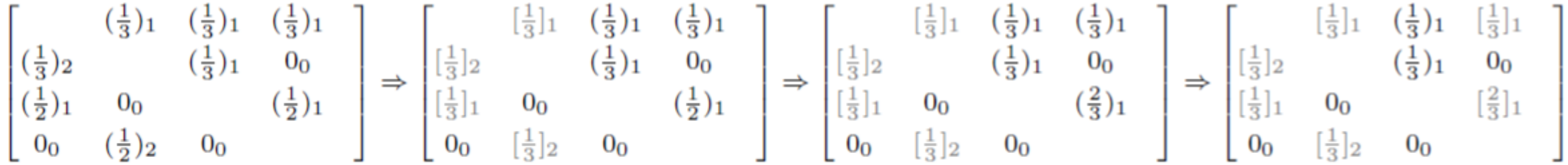


Figure 4: An example of estimating demands in a network of 4 hosts. Each matrix element denotes demand per flow as a fraction of the NIC bandwidth. Subscripts denote the number of flows from that source (rows) to destination (columns). Entries in parentheses are yet to converge. Grayed out entries in square brackets have converged.

- Ho → H1, Ho → H2, Ho → H3, H1 2 flow→ Ho, …

HEDERA ALGORITHMS

Global First Fit

- Searches all possible paths to find a path whose link can carry that flow
- Higher network load → Hard to find a suitable path
- Does not guarantee that all flows will be accommodated

Simulated Annealing

- Efficient probabilistic search, requires large search space, uses iterations
- They assign a single core switch for each destination rather than for each flow
- Searches through a solution space to find a near-optimal solution

HEDERA ALGORITHMS

Global First Fit
- Re-route immediately upon detection

Simulated Annealing
- Waits next tick and uses previously computed flow to optimize

Algorithm Complexity	Time	Space
Global First-Fit	$O((k/2)^2)$	$O(k^3 + \|F\|)$
Simulated Annealing	$O(f_{avg})$	$O(k^3 + \|F\|)$

Table 1: Time and Space Complexity of Global First Fit and Simulated Annealing. k is the number of switch ports, $|F|$ is the total number of large flows, and f_{avg} is the average number of large flows to a host. The k^3 factor is due to in-memory link-state structures, and the $|F|$ factor is due to the flows' state.

- Both are simple algorithms.
- More complex → More time to calculate → Performance decrease
- Simulated Annealing is close to optimal and better than Global First Fit.
- It is more complex but better calculation results tolerate that.

SIMULATION & IMPLEMENTATION TEST

- Due to privacy and security concerns, DC traffic rates are not available
- Synthetic communication pattern
- k=4 pods
- Pod lower layer → Edge switches
- Pod upper layer → Aggregation switch
- Edges ~~has~~ (k/2) hosts → 2
- Pods are interconnected by $(k/2)^2$ core switches → 4

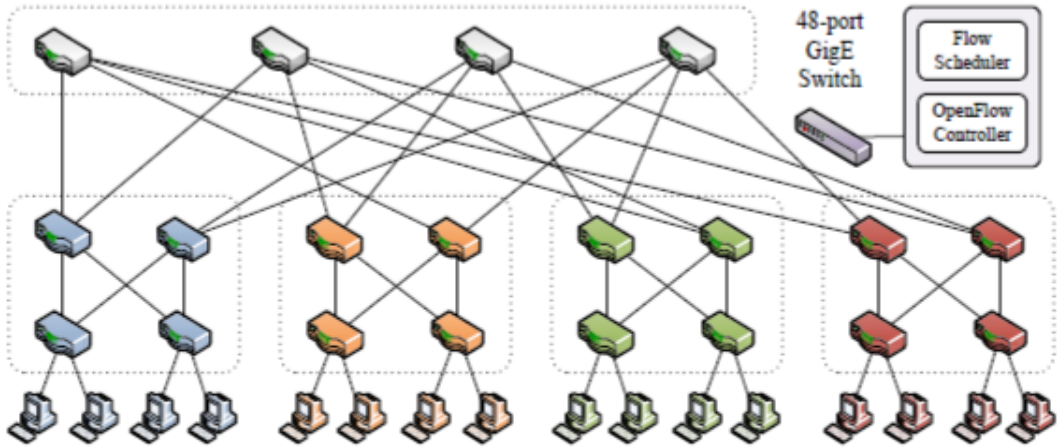


Figure 8: System Architecture. The interconnect shows the data-plane network, with GigE links throughout.

SIMULATION & TEST

- Test scheduler updates every 5 seconds

- Simulator with 8,192 hosts
- Each simulation takes 60 seconds
- Each host sending 1Gbps → 2.5×10^{11} packets for 60 seconds run
- Existing per-packet simulator with 1M packet per sec → 71 hours to simulate
- Network graph with directed edges and accepts communication patterns
- Measured the average bisection bandwidth during middle 40 seconds

RESULTS & EVALUATION

- # of hosts: 16, 1024 and 8192 with corresponding k: 4, 16, 32
- Each test lasts 60 seconds and uses TCP flows

(1) Stride(i): A host with index x sends to the host with index $(x + i) mod (num\_hosts)$.

(2) Staggered Prob *(EdgeP, PodP)*: A host sends to another host in the same edge switch with probability *EdgeP*, and to its same pod with probability *PodP*, and to the rest of the network with probability *1-EdgeP - PodP*.

(3) Random: A host sends to any other host in the network with uniform probability. We include bijective mappings and ones where hotspots are present.

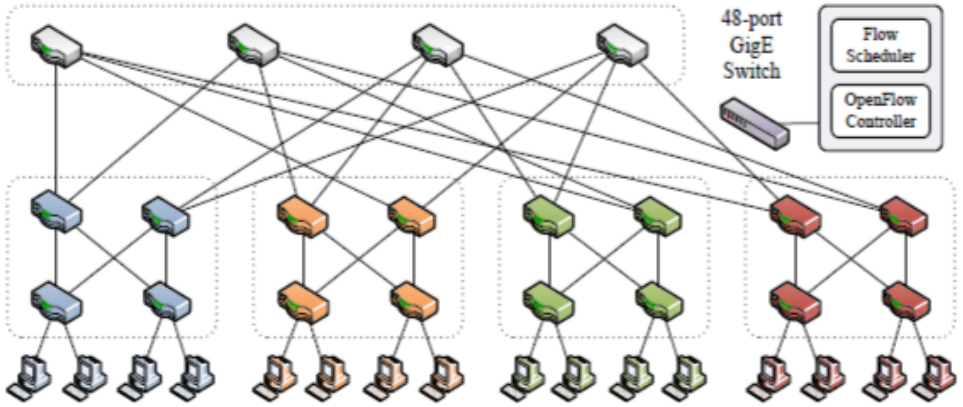


Figure 8: System Architecture. The interconnect shows the data-plane network, with GigE links throughout.

RESULTS & EVALUATION

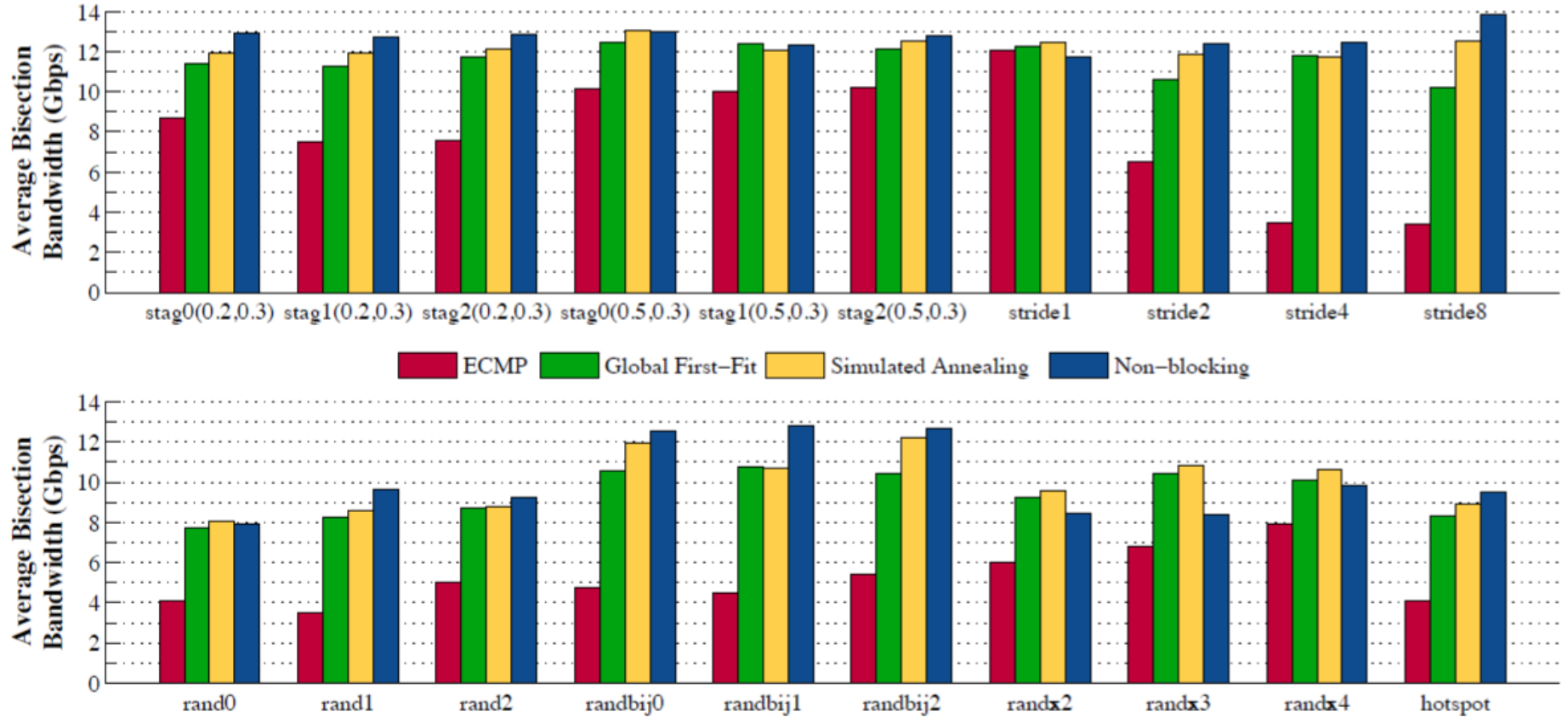


Figure 9: Physical testbed benchmark suite results for the three routing methods vs. a non-blocking switch. Figures indicate network bisection bandwidth achieved for staggered, stride, and randomized communication patterns.

RESULTS & EVALUATION

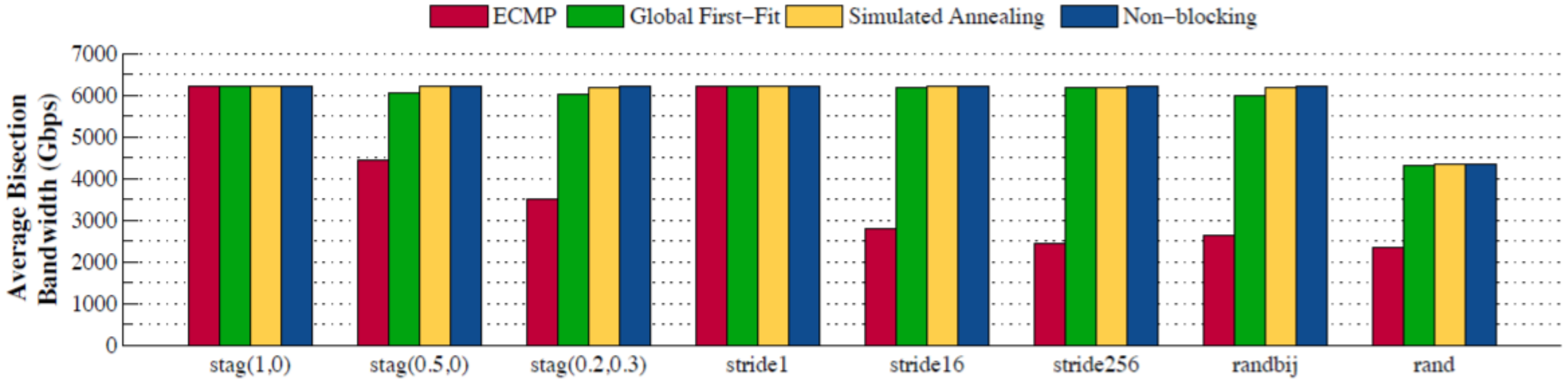


Figure 10: Comparison of scheduling algorithms for different traffic patterns on a fat-tree topology of 8,192-hosts.

RESULTS & EVALUATION

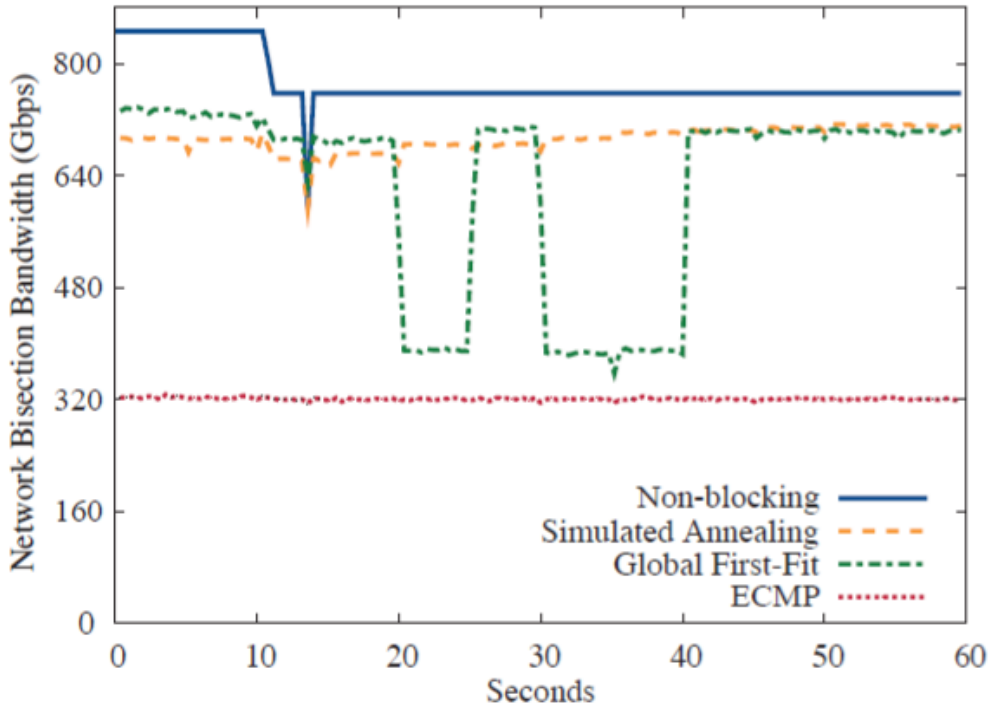


Figure 11: Network bisection bandwidth vs. time for a 1,024 host fat-tree and a random bijective traffic pattern.

RESULTS & EVALUATION

- Quality of ~~the~~ Simulated Annealing
- Few iterations for each ~~scheduling~~ is sufficient for performance
- Even with less iterations outperforms static algorithms

	Number of Hosts		
SA Iterations	16	1,024	8,192
1000	78.73	74.69	72.83
50000	78.93	75.79	74.27
100000	78.62	75.77	75.00
500000	79.35	75.87	74.94
1000000	79.04	75.78	75.03
1500000	78.71	75.82	75.13
2000000	78.17	75.87	75.05
Non-blocking	81.24	78.34	77.63

Table 3: Percentage of final bisection bandwidth by varying the Simulated Annealing iterations, for a case of random destinations, normalized to the full network bisection. Also shown is the same load running on a non-blocking topology.

RESULTS & EVALUATION

k	Hosts	Large flows	Runtime (ms)
16	1024	1024	1.45
16	1024	5000	4.14
32	8192	8192	2.71
32	8192	25000	9.23
32	8192	50000	26.31
48	27648	27648	6.91
48	27648	100000	51.30
48	27648	250000	199.43

Table 4: Demand estimation runtime.

	1,024 hosts		8,192 hosts	
Iterations	f =3,215	f =6,250	f =25k	f =50k
1000	2.997	5.042	6.898	11.573
5000	12.209	20.848	19.091	32.079
10000	23.447	40.255	32.912	55.741

Table 5: Runtime (ms) vs. number of Simulated Annealing iterations for different number of flows f.

- 4 parallel threads
- Quad-cores, 2.13 GHz
- 250k flows (10 per host) → 200ms

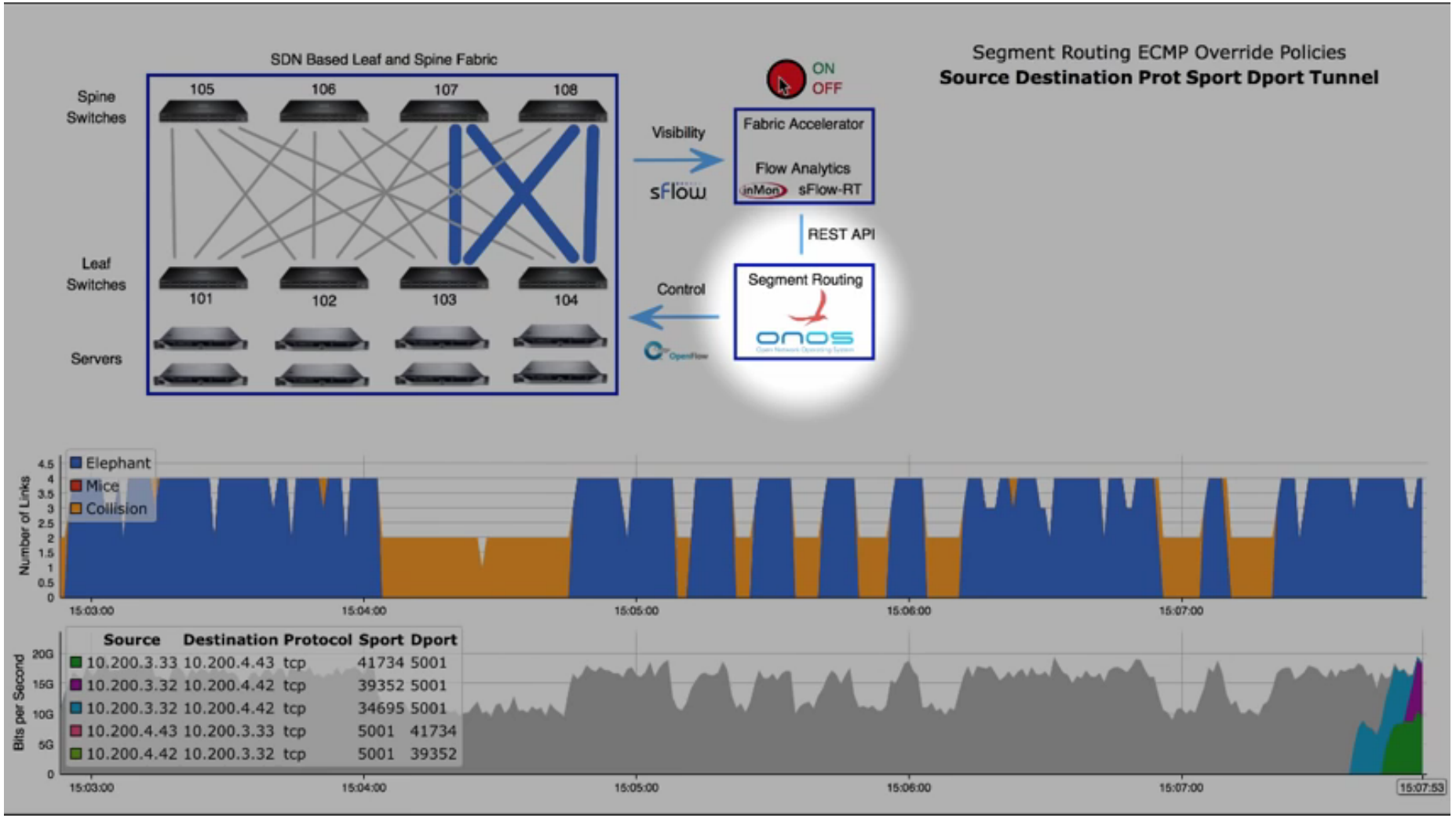
RESULTS & EVALUATION

	Flows per host		
Hosts	1	5	10
1,024	100.2	100.9	101.7
8,192	101.4	106.8	113.5
27,648	104.6	122.8	145.5

Table 6: Length of control loop (ms).

- Control loop:
- 1. Scheduler gets large flow information from all switches
- 2. Scheduler estimates flow demands and calculates their routes
- 3. Scheduler transmits the calculated data to all switches

SDN

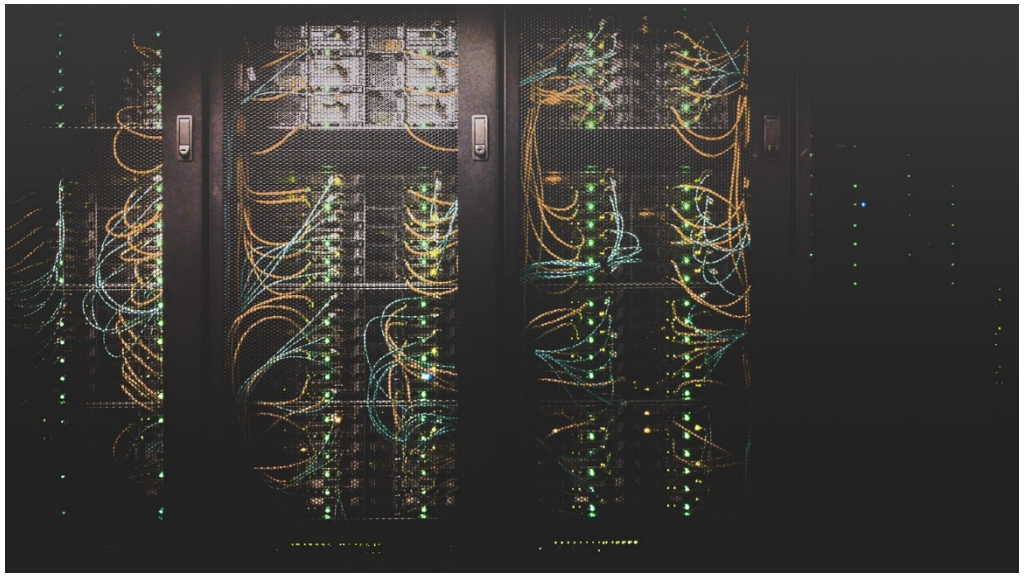

SDN Based Leaf and Spine Fabric
Spine Switches
Leaf Switches
Servers
105
106
107
108
101
102
103
104
Visibility
sFlow
ON
OFF
Fabric Accelerator
Flow Analytics
inMon sFlow-RT
REST API
Control
Segment Routing
onos
Segment Routing ECMP Override Policies
Source Destination Prot Sport Dport Tunnel
Elephant
Mice
Collision
Number of Links
15:03:00
15:04:00
15:05:00
15:06:00
15:07:00
Source Destination Protocol Sport Dport
10.200.3.33 10.200.4.43 tcp 41734 5001
10.200.3.32 10.200.4.42 tcp 39352 5001
10.200.3.32 10.200.4.42 tcp 34695 5001
10.200.4.43 10.200.3.33 tcp 5001 41734
10.200.4.42 10.200.3.32 tcp 5001 39352
Bits per Second
15:07:53

SDN

- Software Defined Networks
- Dynamic network management
- Control plane and data plane
- Programmable control plane → design, build and manage networks
- Control plane → Routing process/calculations
- Data plane → Forwards packages
- 2011 → Decouple control and forwarding planes
- 2012 → First interface for control and data planes

CONCLUSION

- Static load-balancers can cause bisection collisions.
- Central scheduler with global knowledge of active flows can outperform static hash-based ECMP load-balancing.
- In almost every case Simulated Annealing is better than Global First Fit



THANK YOU

Al-Fares, Mohammad, et al. "Hedera: dynamic flow scheduling for data center networks." *Nsdi*. Vol. 10. No. 2010. 2010.

Images References

Outline, Hedera: Photo by Nika Akin on Unsplash

Data Center Networking: Photo by Taylor Vick on Unsplash

Hedera, Data Center: Photo by Stephen Dawson on Unsplash

Conclusion, Server: Photo by Taylor Vick on Unsplash