A Tale of Two Topologies: Exploring Convertible Data Center Network Architectures with *Flat-tree*

Yiting Xia, *Xiaoye Steven Sun,*
Simbarashe Dzinamarira, Dingming Wu,
Xin Sunny Huang, T. S. Eugene Ng

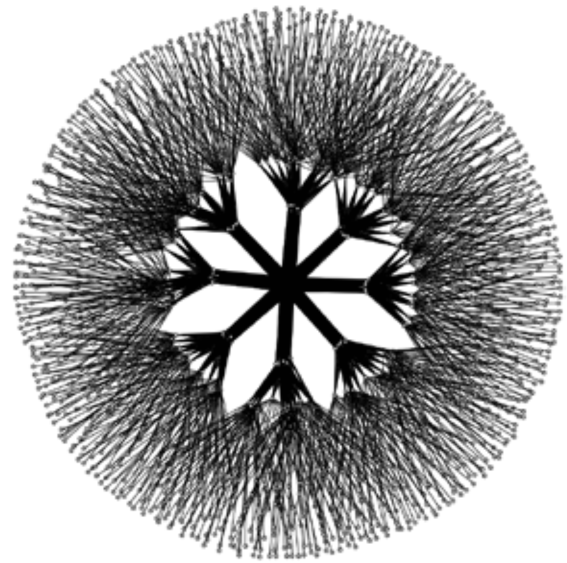
Rice University

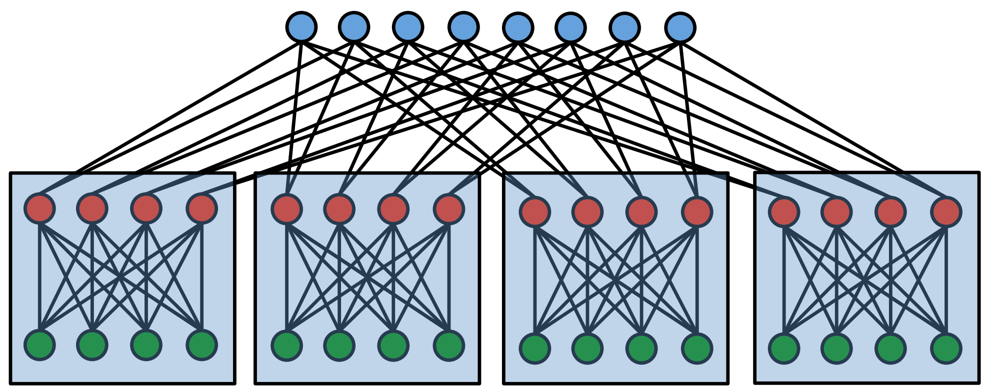
Convertible Network

- Convertibility
  - *A network's ability to change between multiple topologies with different characteristics*
  - *Managed by software, no human labor for rewiring*
- Combine benefits of different worlds

Clos Topology

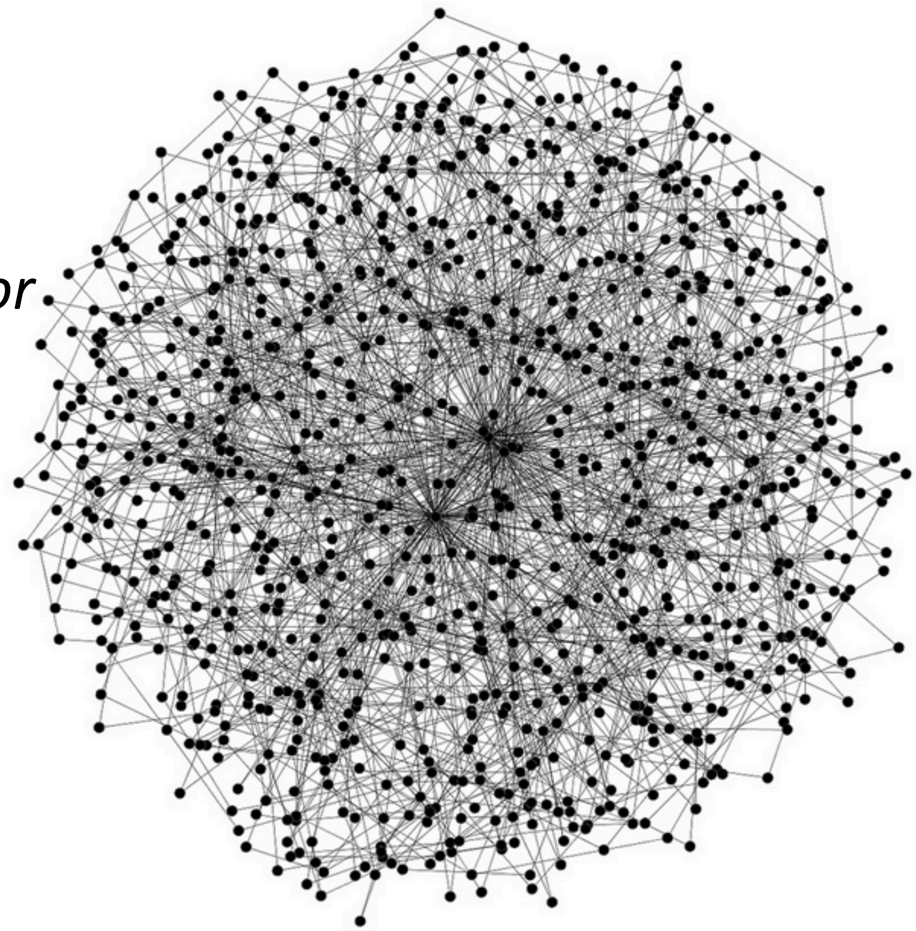
- Implementation friendly
  - *- Central wiring*
  - *- Flexible scale and oversubscription*
  - *- Pod modular design*
- Suboptimal performance
  - *- Long paths*
  - *- Congested network core*





Random Graph

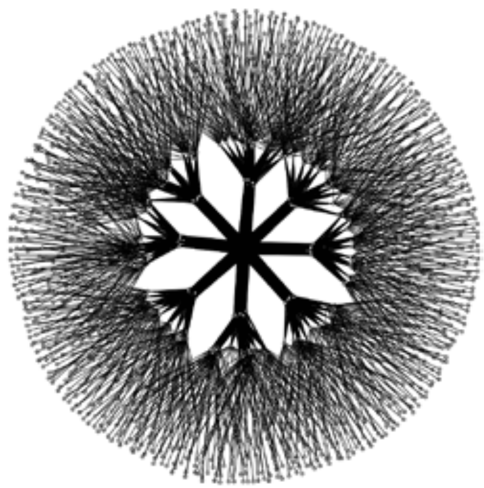
- Good performance
  - *- Low average path length*
  - *- Rich bandwidth*
  - *- Near optimal throughput for uniform traffic*
- Hard to implement
  - *- Neighbor-to-neighbor wiring complicated*



[*Jellyfish NSDI'12*]

The Case for Convertibility

Tree Network



vs.

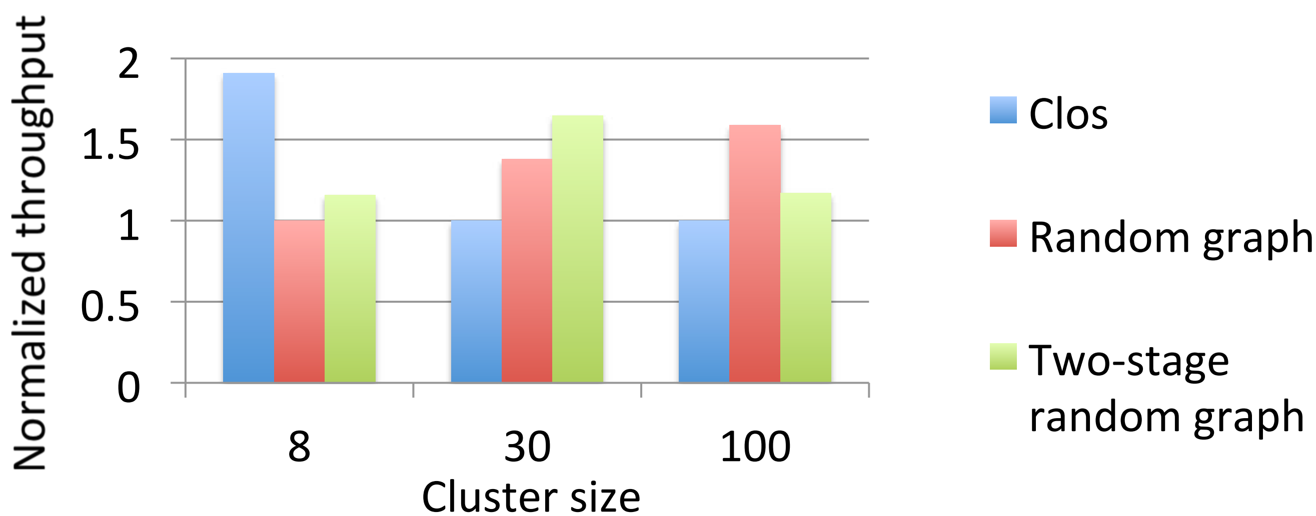
Flat Network

Easy implementation

Good performance

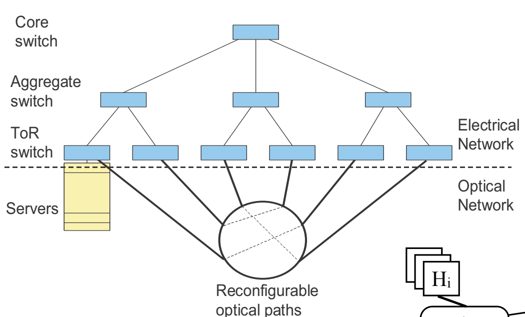
The Case for Convertibility

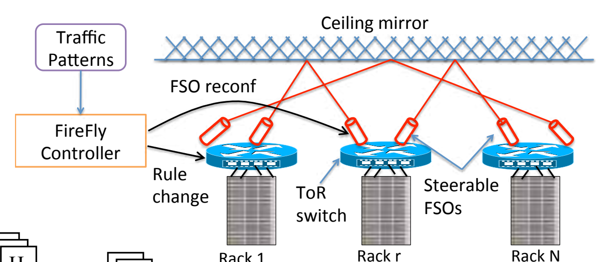
- Data center traffic: different locality and cluster size
  - *Random graph (global) for global traffic*
  - *Two-stage (local) random graph for in-Pod traffic*
  - *Clos for in-rack traffic*
- Motivating example
  - *8 servers per rack, 64 servers per Pod, all-to-all traffic*

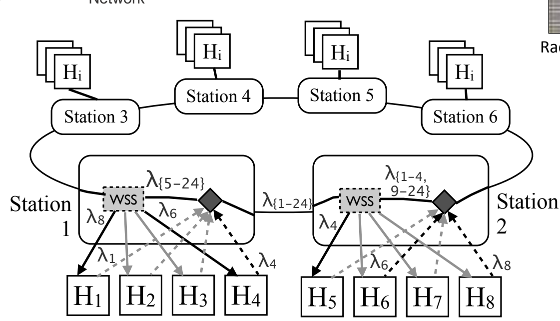


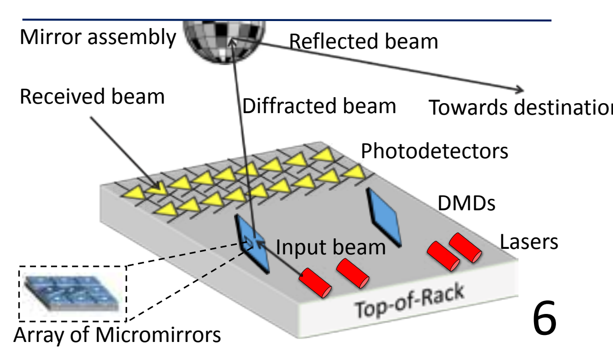
Topology can be configurable

- Helios, c-Through, Flyways, OSA, 3DBeam, Mordia, FireFly, Quartz, WaveCube, ProjecToR, etc
- Create ad-hoc links on the fly
- Technology available
  - *3D MEMS, WSS, WDM, DMD, free-space optics, 60GHz wireless, wireless beamforming*







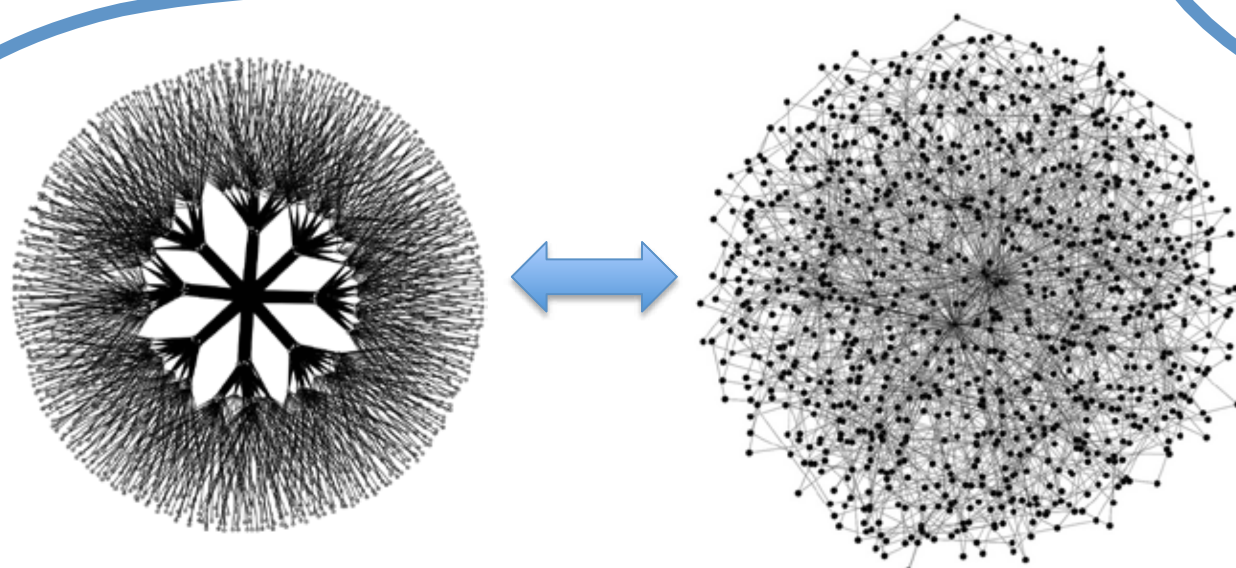


Flat-tree Prototype Architecture

- Start from Clos
- Flatten tree structure
- Approximate random graphs

Flat-tree

Tree Network



Flat Network

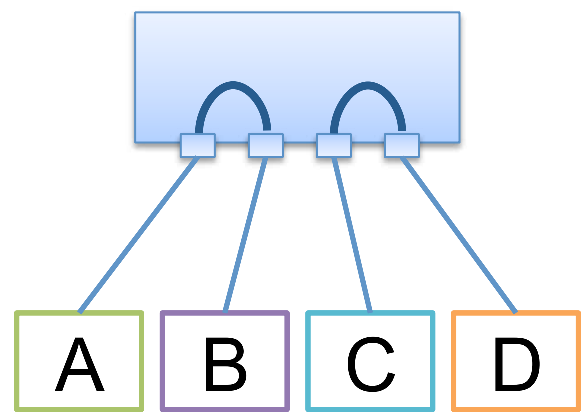
Flatten the Tree

- How to flatten the tree structure?

Difference	Clos	Random graph	Solution
Server distribution	Edge switches	All switches	Relocate servers
Wiring	Central	Neighbor-to-neighbor	Diversify connections

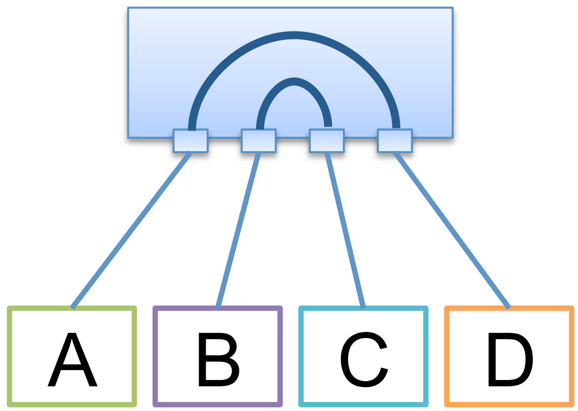
Converter Switch

- Small port-count
- Low cost
  - *Optical Fibers in data center*
    - ∗ Small optical switch
    - ∗ $10 per port
  - *DAC in data center*
    - ∗ Crosspoint switch
    - ∗ $3 per port
- Physical layer device



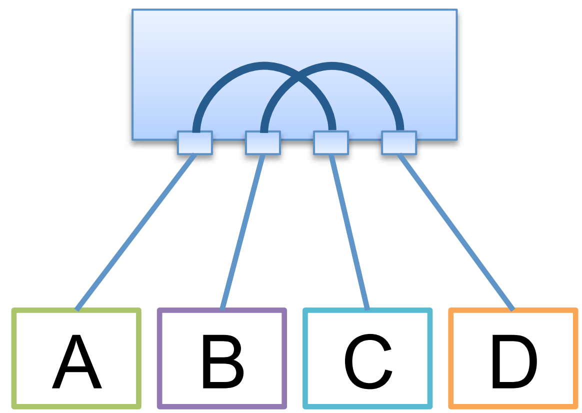
Converter Switch

- Small port-count
- Low cost
  - *- Optical Fibers in data center*
    - ∗ Small optical switch
    - ∗ $10 per port
  - *- DAC in data center*
    - ∗ Crosspoint switch
    - ∗ $3 per port
- Physical layer device



Converter Switch

- Small port-count
- Low cost
  - *Optical Fibers in data center*
    - Small optical switch
    - $10 per port
  - *DAC in data center*
    - Crosspoint switch
    - $3 per port
- Physical layer device



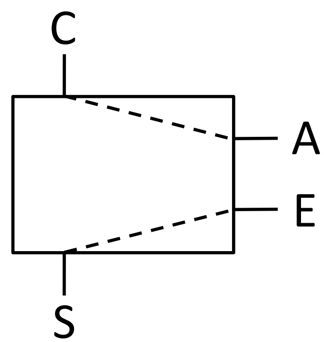
Converter Switch

C: core switch

A: aggregation
switch

E: edge switch

S: server



4-port Converter Switch

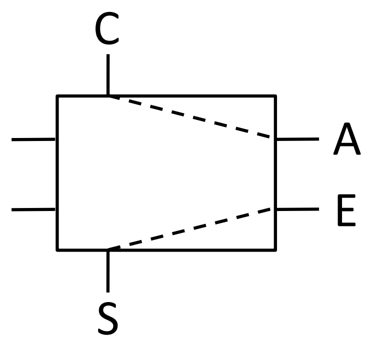
Converter Switch

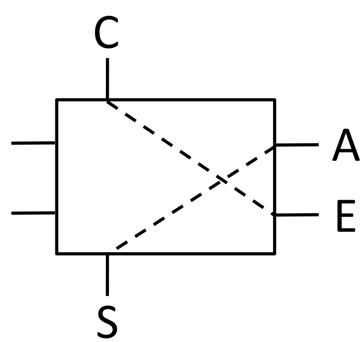
C: core switch

A: aggregation switch

E: edge switch

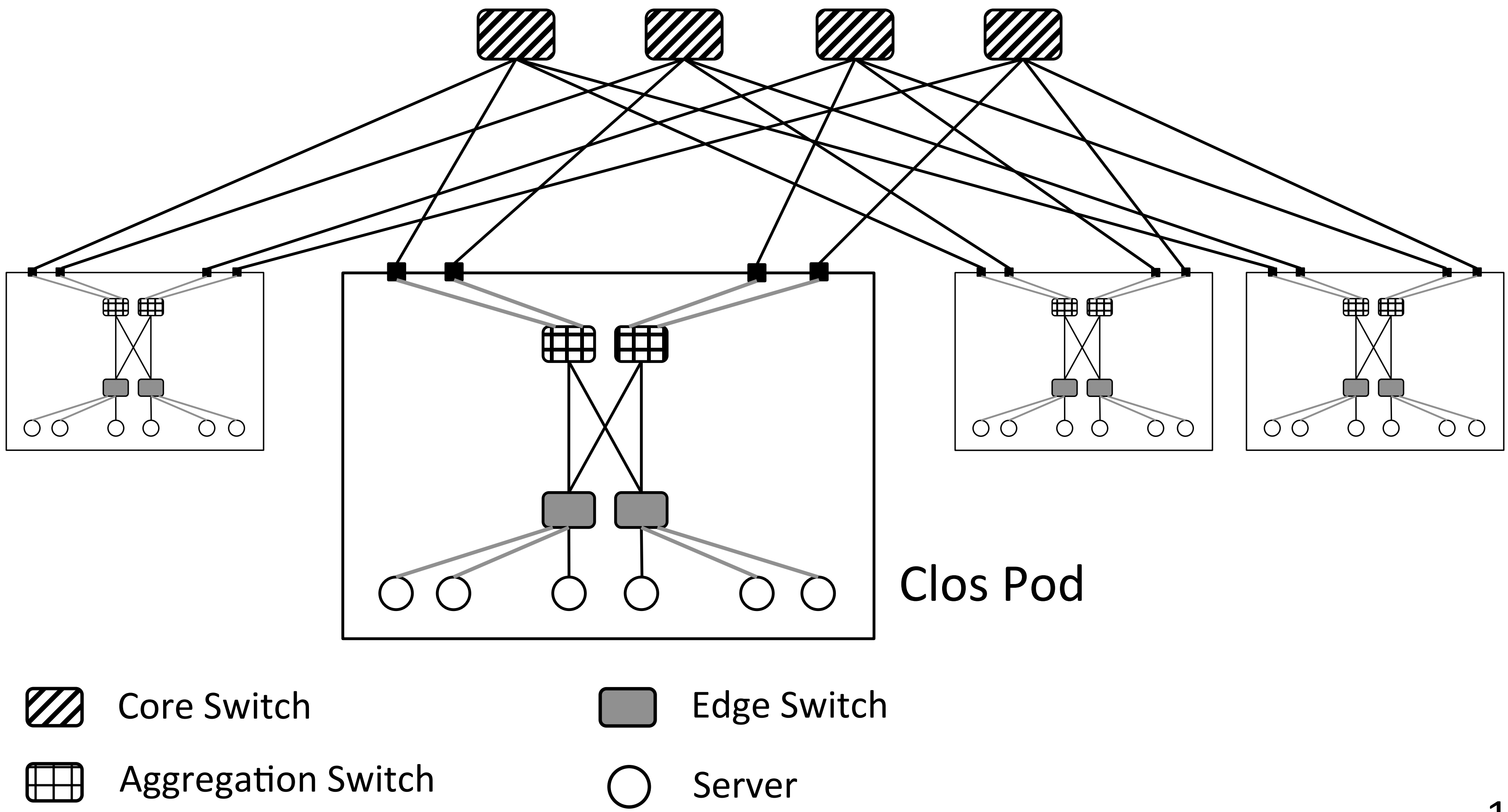
S: server



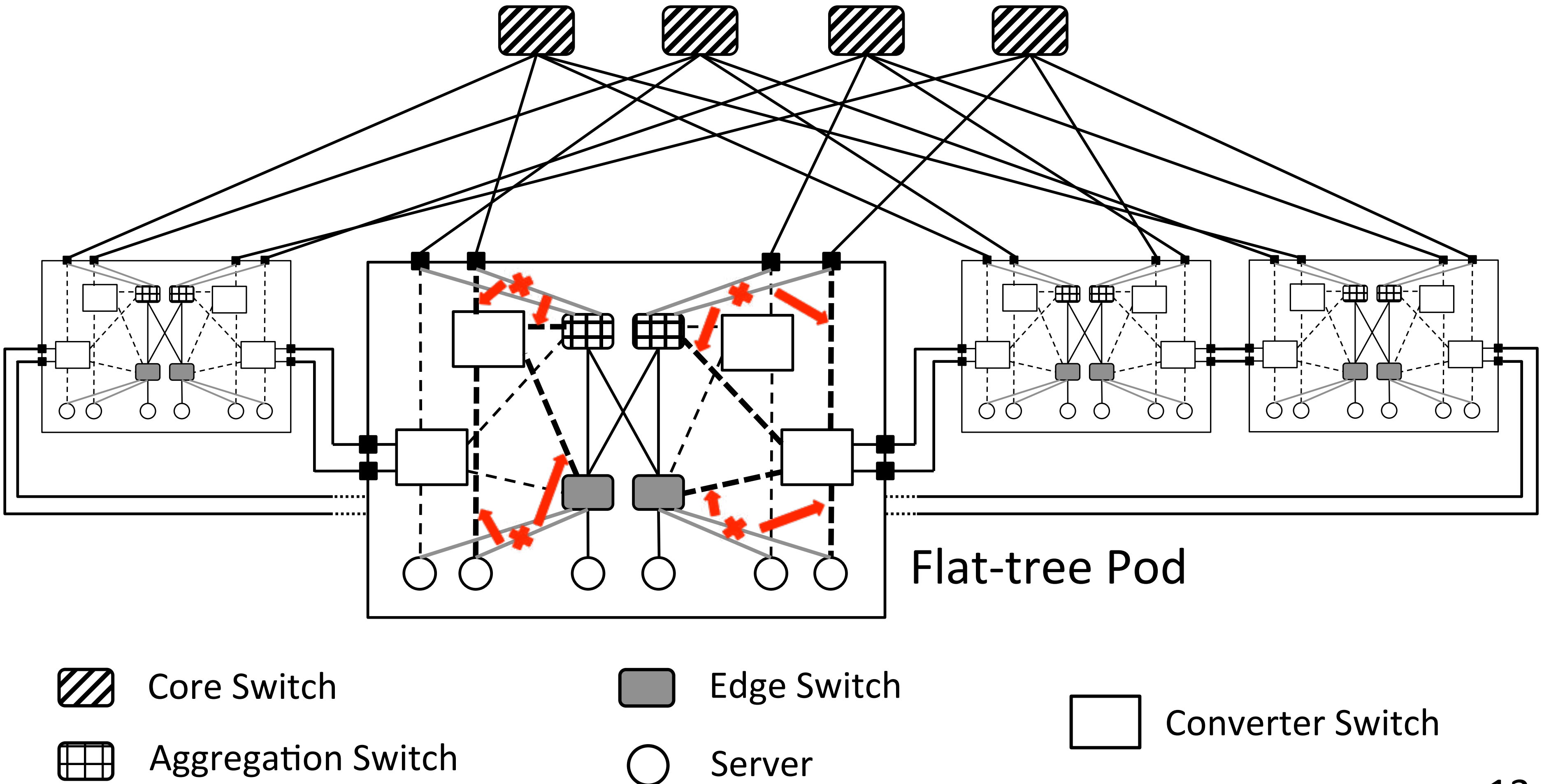


6-port Converter Switch

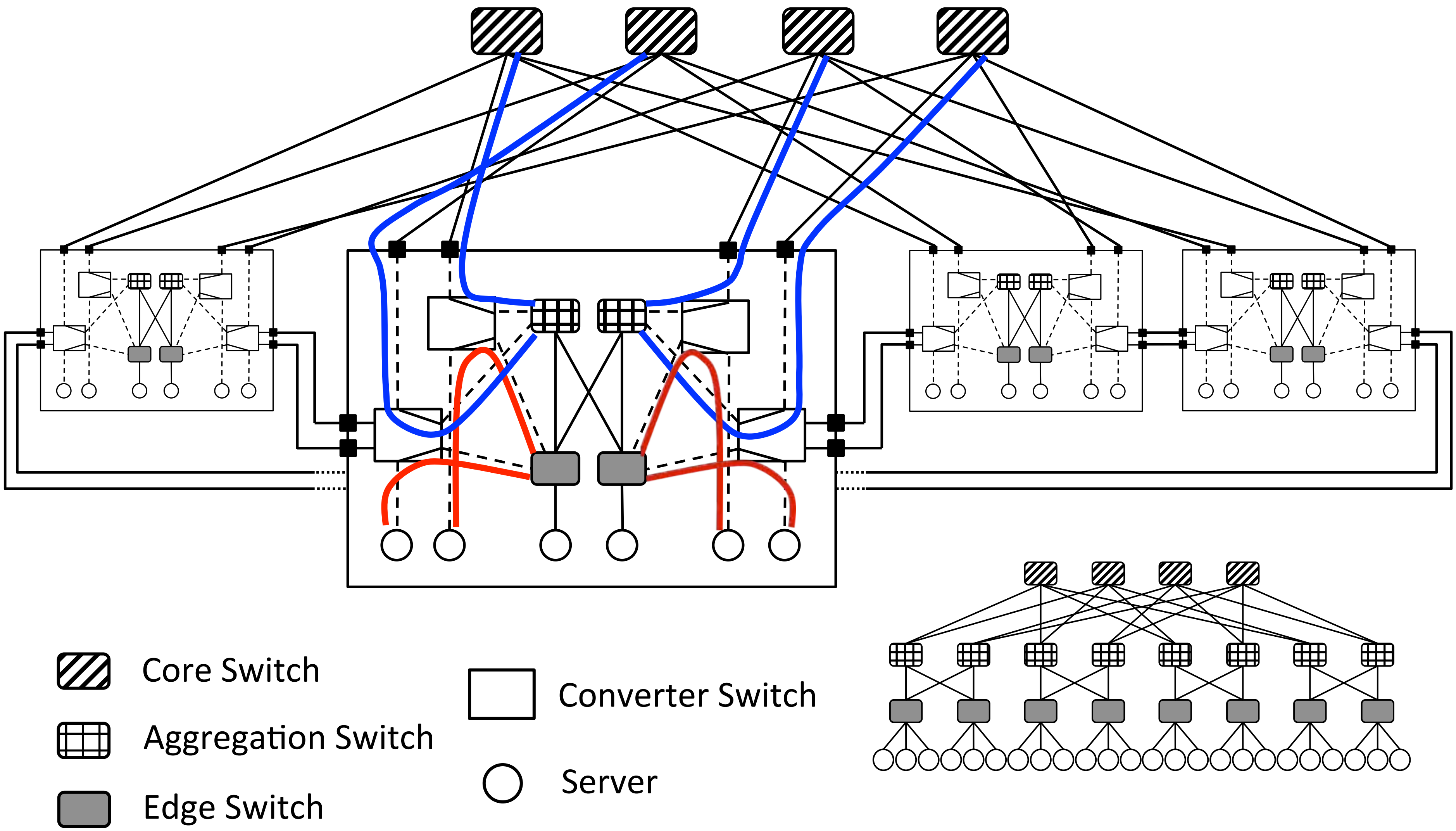
Flat-tree Example



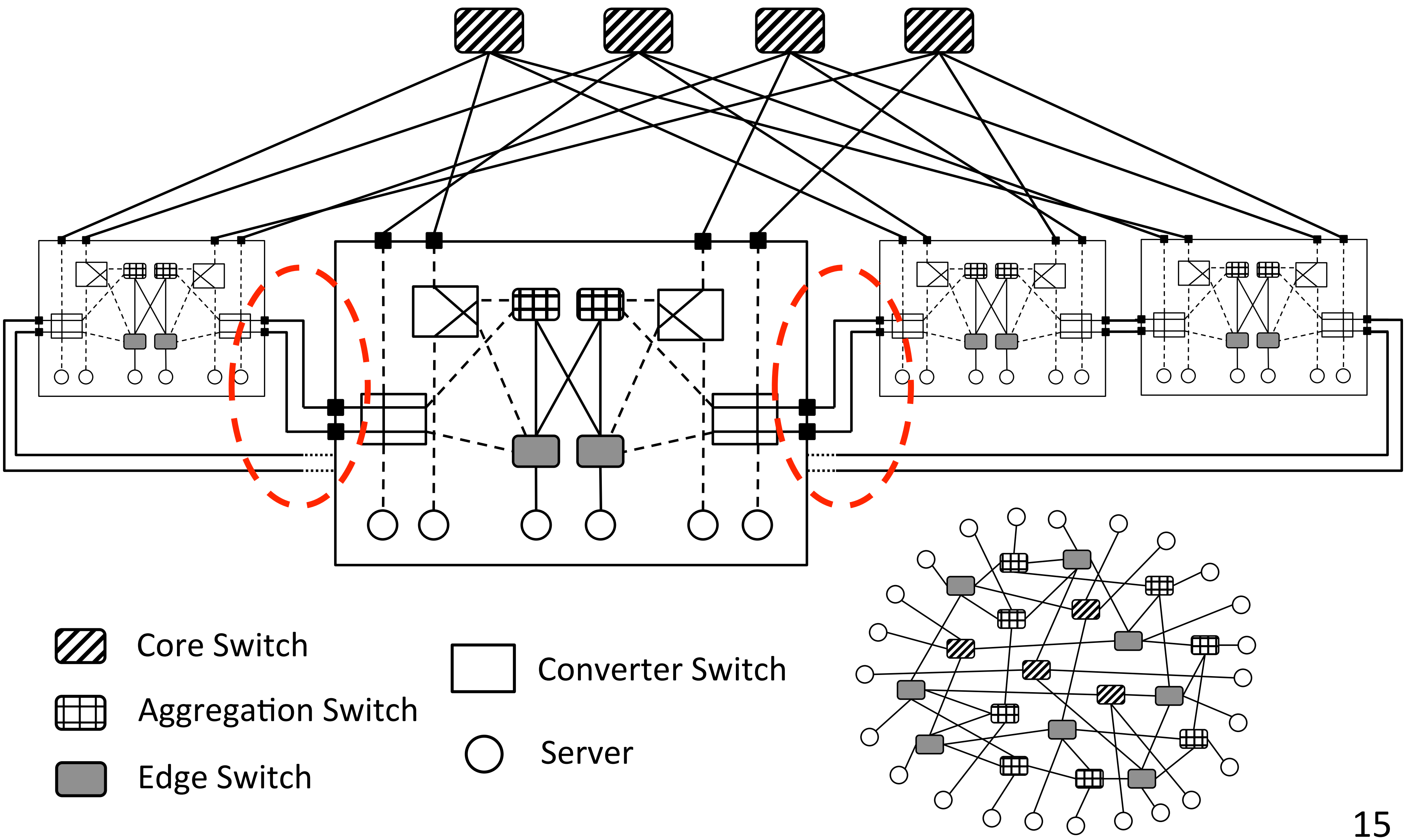
Flat-tree Example



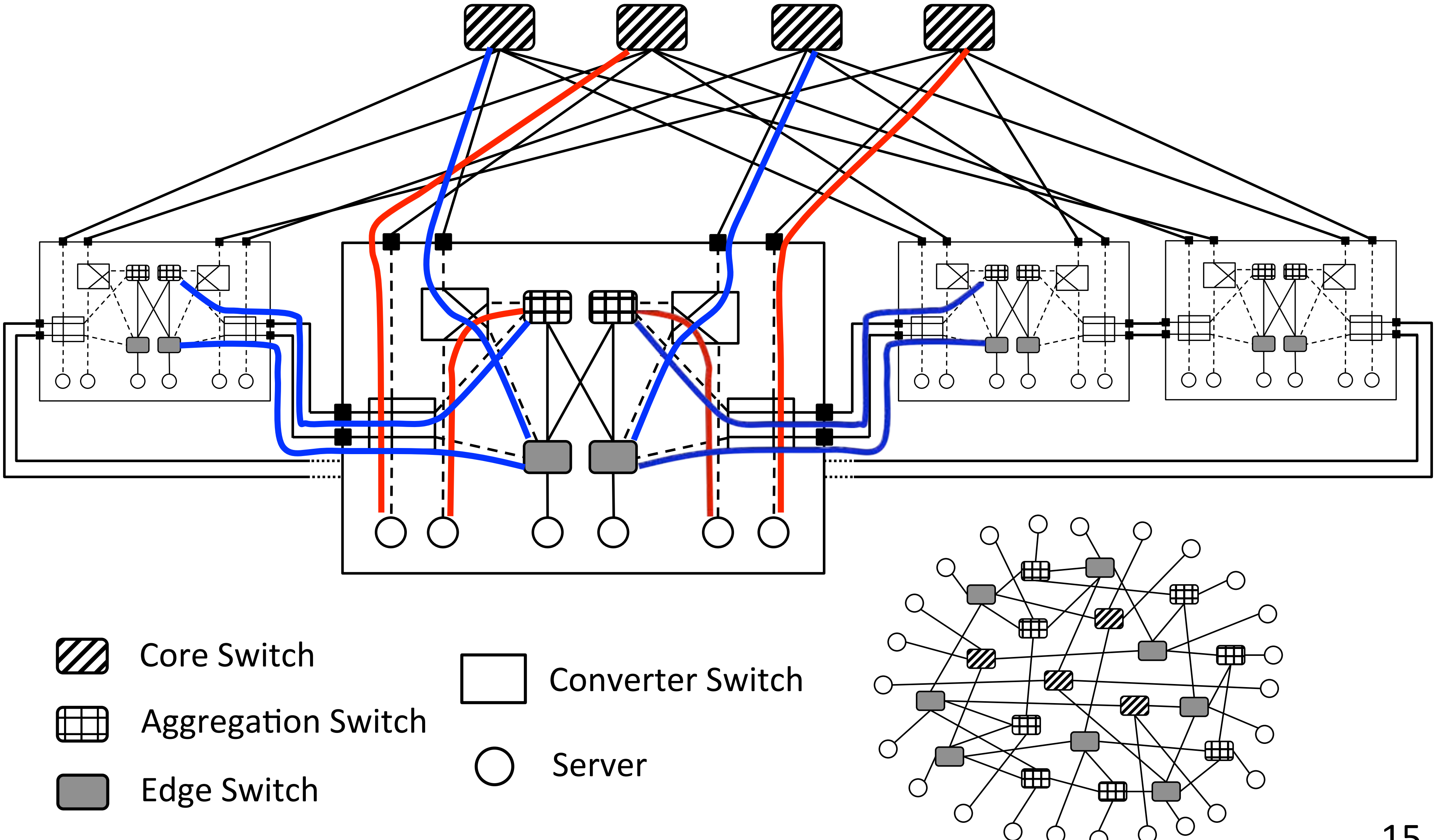
Clos Network



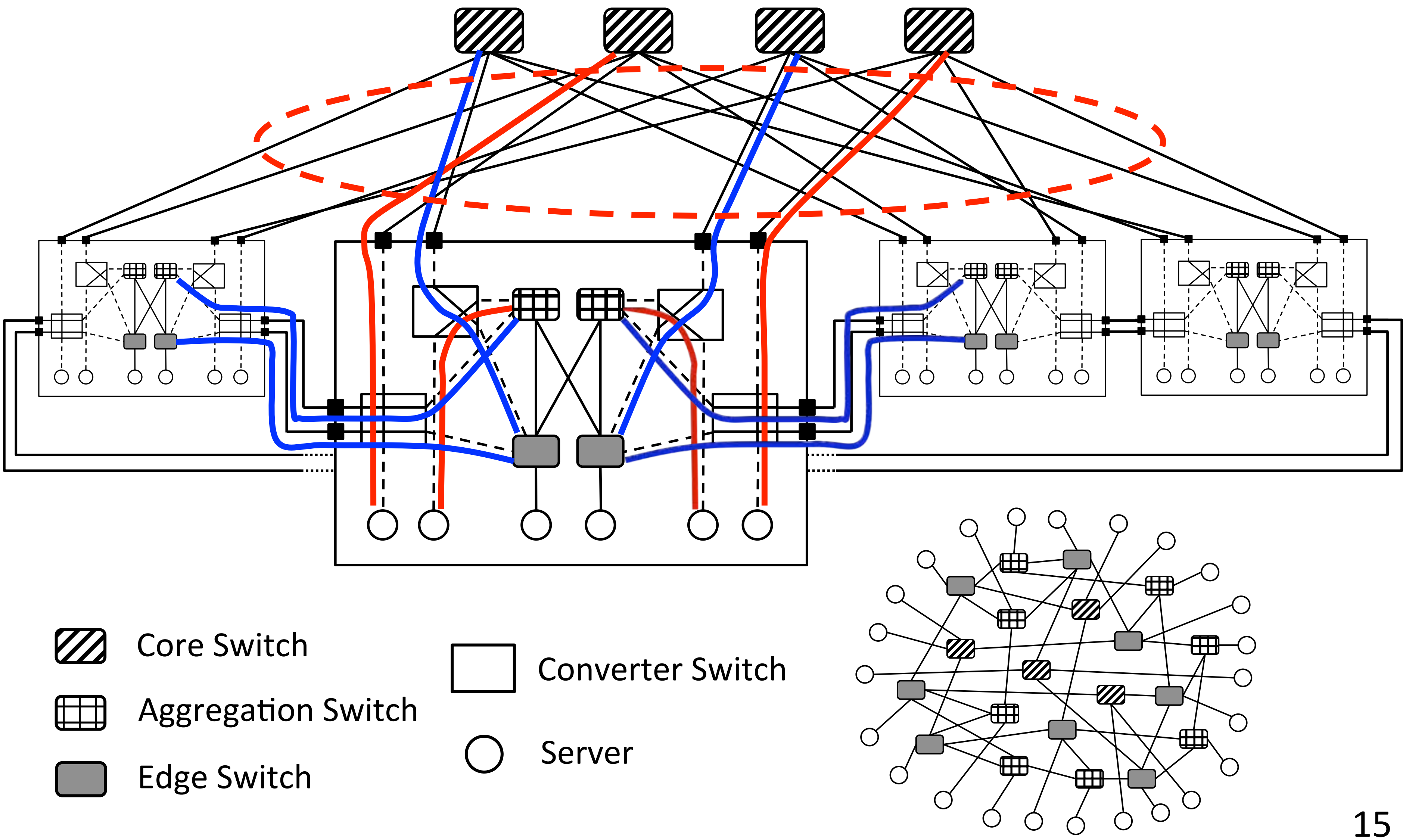
Approximate Random Graph



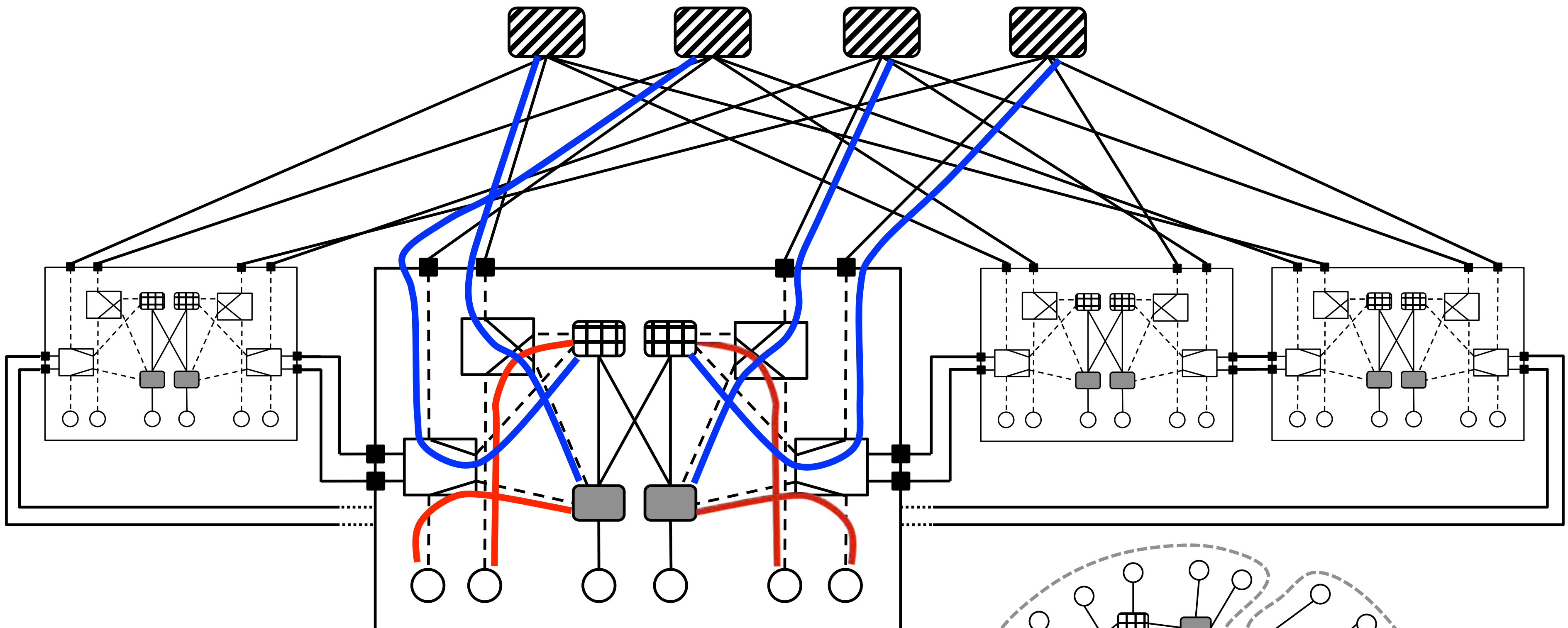
Approximate Random Graph

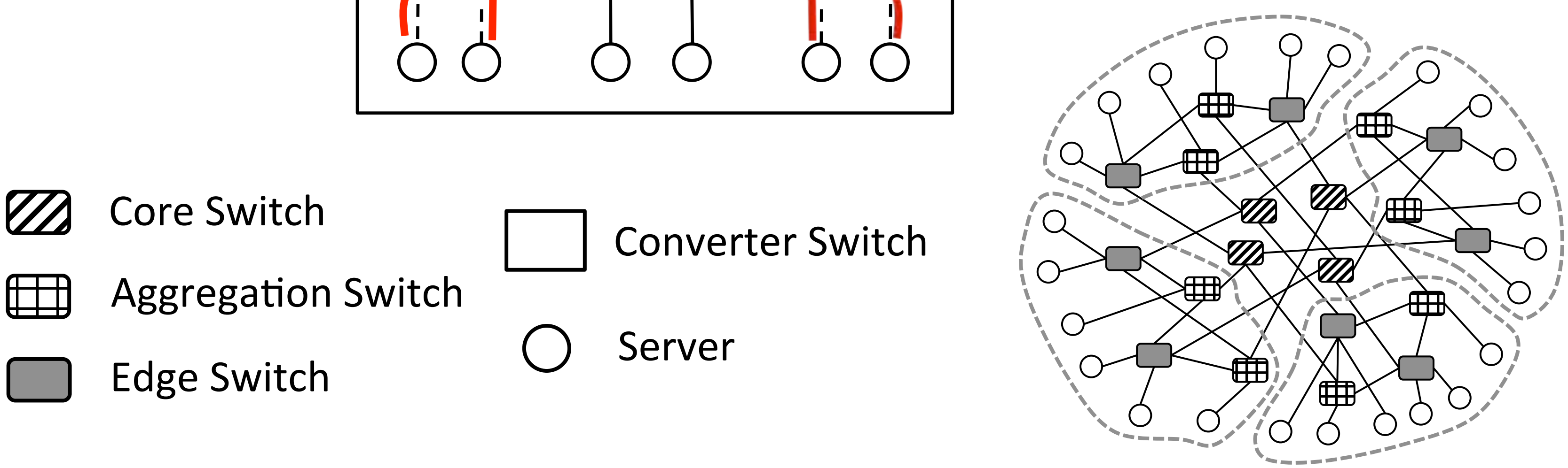


Approximate Random Graph



Approximate Local Random Graph

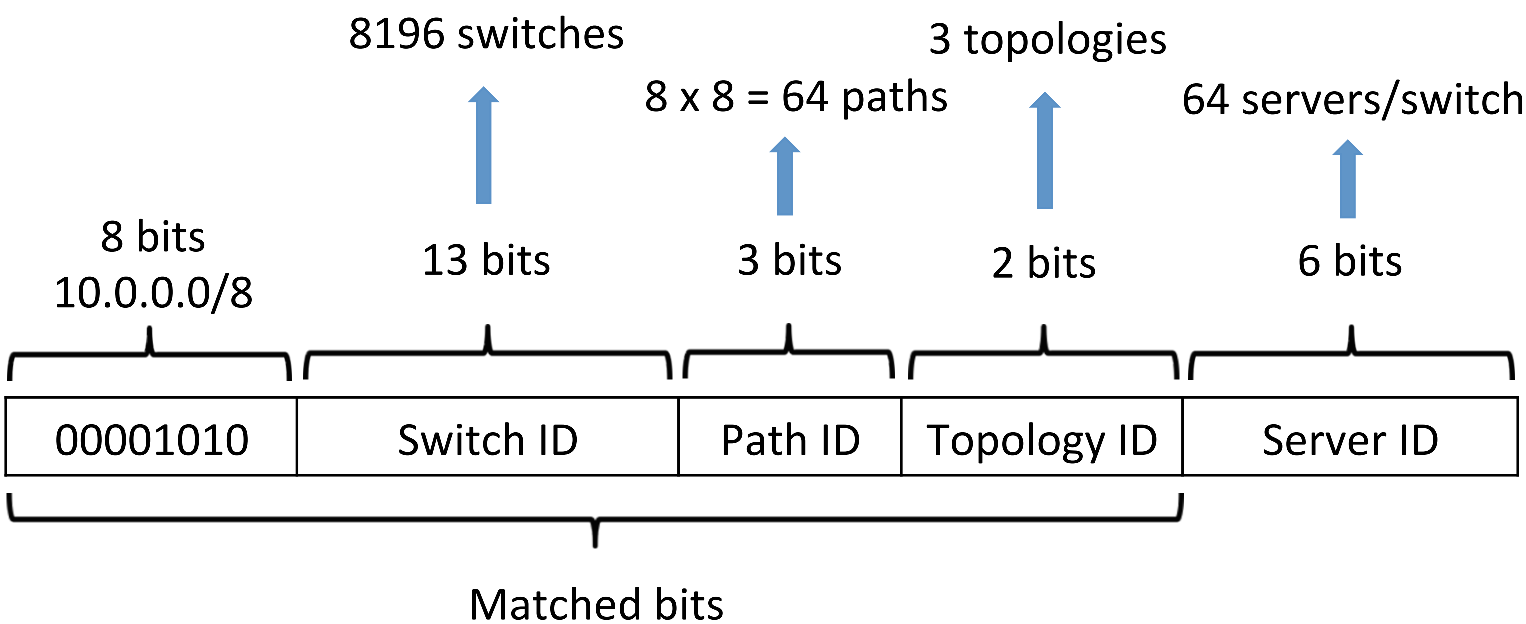




Control Plane

- k-shortest-path routing + MPTCP
  - *k paths for every sever pairs*
  - *Enormous number of states → exceed switch capacity*
  - *No solution from random graph networks*
  - *Scalability concern!!!*
- Aggregation
  - *Addressing: prefix matching of ingress/egress switch*
  - *Source routing*
- Highly challenging in flat-tree
  - *Server mobility to different switches*

Addressing


8196 switches
3 topologies
8 x 8 = 64 paths
64 servers/switch
8 bits
10.0.0.0/8
13 bits
3 bits
2 bits
6 bits
00001010
Switch ID
Path ID
Topology ID
Server ID
Matched bits

Addressing Example

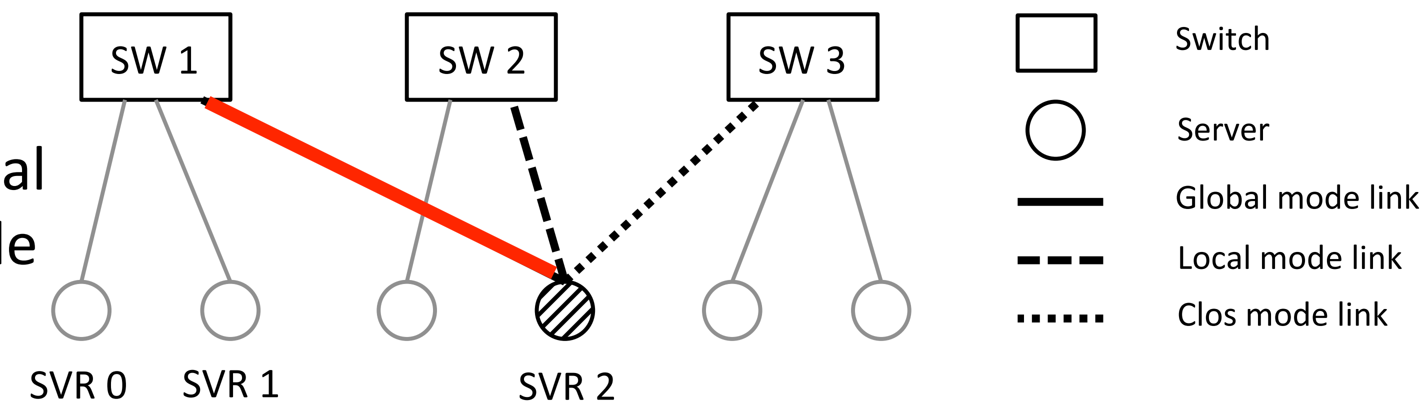
8 bits 10.0.0.0/8	13 bits	3 bits	2 bits	6 bits
00001010	Switch ID	Path ID	Topology ID	Server ID

IP addresses

10	**1**	**0**	**0**	**2**
10	**1**	**1**	**0**	**2**
10	2	0	1	1
10	2	1	1	1
10	3	0	2	0
10	3	1	2	0

4 paths

Global mode

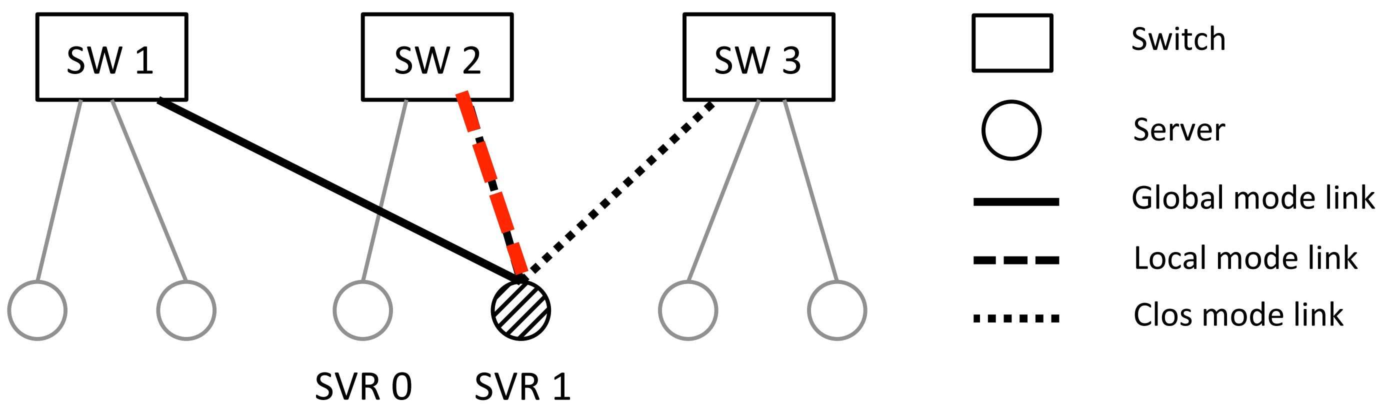


Addressing Example

8 bits 10.0.0.0/8	13 bits	3 bits	2 bits	6 bits
00001010	Switch ID	Path ID	Topology ID	Server ID

IP addresses	10	1	0	0	2
	10	1	1	0	2
	10	**2**	**0**	**1**	**1**
	10	**2**	**1**	**1**	**1**
	10	3	0	2	0
	10	3	1	2	0

Local mode



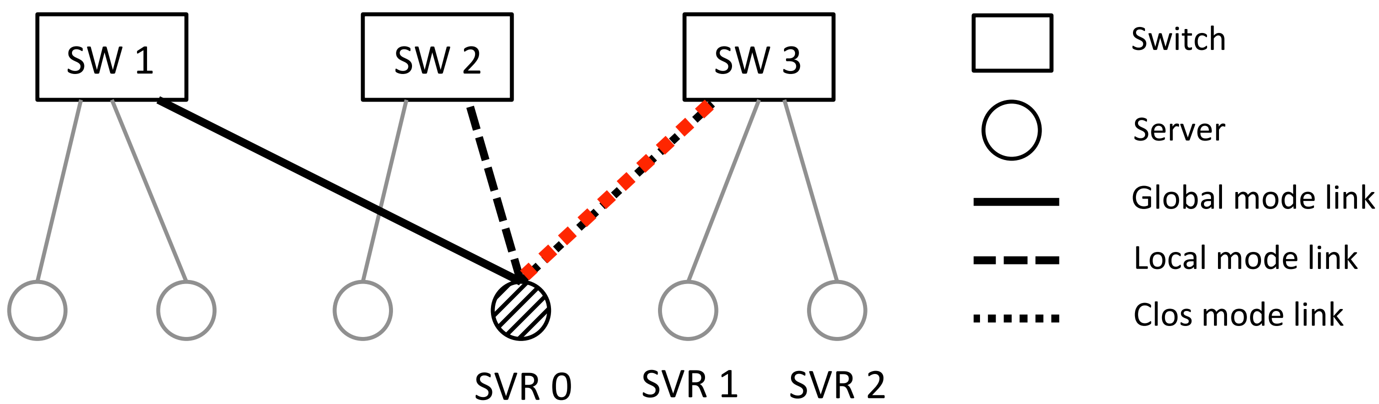
Addressing Example

8 bits 10.0.0.0/8	13 bits	3 bits	2 bits	6 bits
00001010	Switch ID	Path ID	Topology ID	Server ID

IP addresses

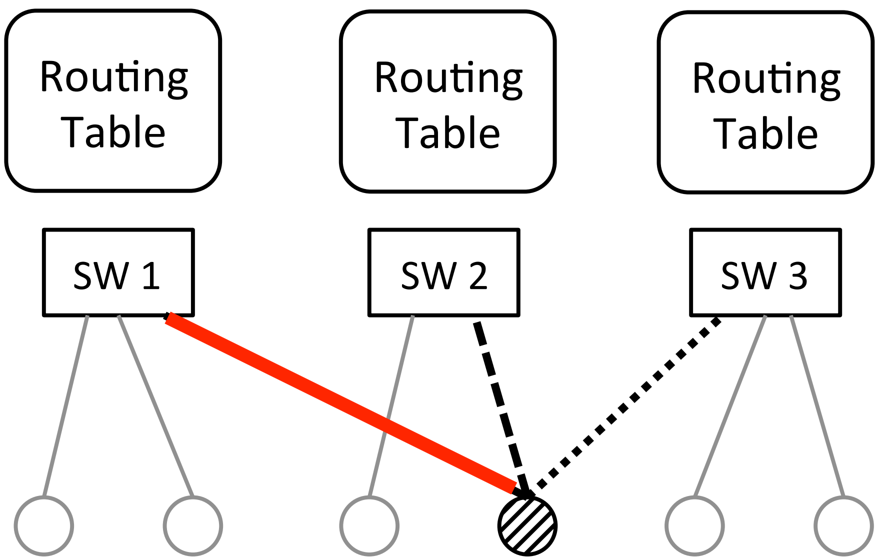
10	1	0	0	2
10	1	1	0	2
10	2	0	1	1
10	2	1	1	1
10	**3**	**0**	**2**	**0**
10	**3**	**1**	**2**	**0**

Clos mode



Addressing Example

	8 bits 10.0.0.0/8	13 bits	3 bits	2 bits	6 bits
	00001010	Switch ID	Path ID	Topology ID	Server ID
Global mode	**10**	**1**	**0**	**0**	**2**
	10	**1**	**1**	**0**	**2**

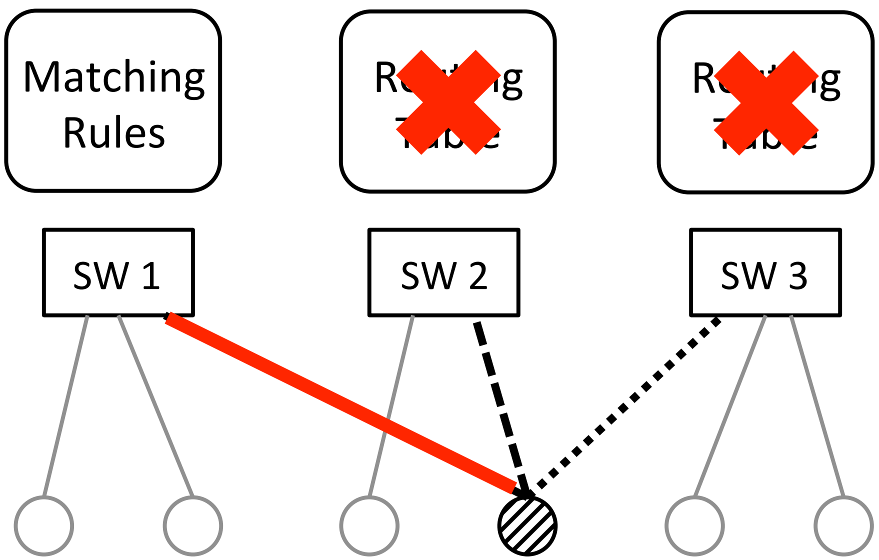


Routing Table

src prefix	dest prefix	next hop
src_sw + path0	dest_sw + path0	port0
src_sw + path0	dest_sw + path1	port1
src_sw + path1	dest_sw + path0	port2
src_sw + path1	dest_sw + path1	port3

Source Routing

8 bits 10.0.0.0/8	13 bits	3 bits	2 bits	6 bits
00001010	Switch ID	Path ID	Topology ID	Server ID
Global mode 10	1	0	0	2
10	1	1	0	2



Matching Rules

src prefix	dest prefix	path
path0	dest_sw + path0	port0, port1, …
path0	dest_sw + path1	port1, port2, …
path1	dest_sw + path0	port2, port3, …
path1	dest_sw + path1	port3, port4, …

Control Plane

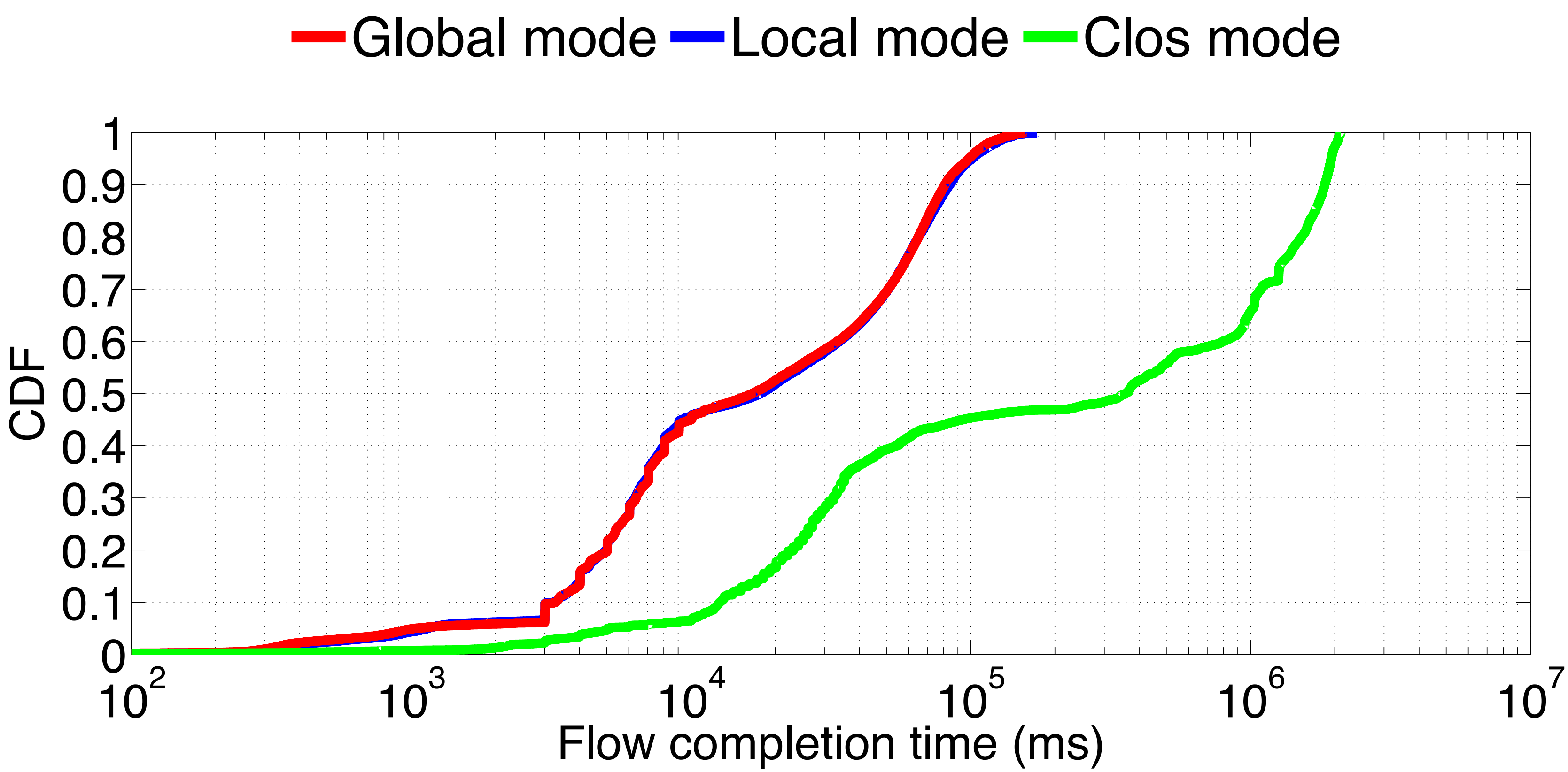
- Addressing
  - *Server-level → switch-level k-shortest path routing*
  - *k paths per server pair → k paths per switch pair*
- Source routing
  - *Transit switches: no states*
  - *Ingress switch: k paths per egress switch*
- Applicable to static random graph networks

Evaluation

- Packet-level simulation
- Traffic traces from 4 Facebook data centers
  - *Hadoop-1: no locality*
  - *Haddop-2 : rack-level locality*
  - *Web: Pod-level locality*
  - *Cache: Pod-level locality*

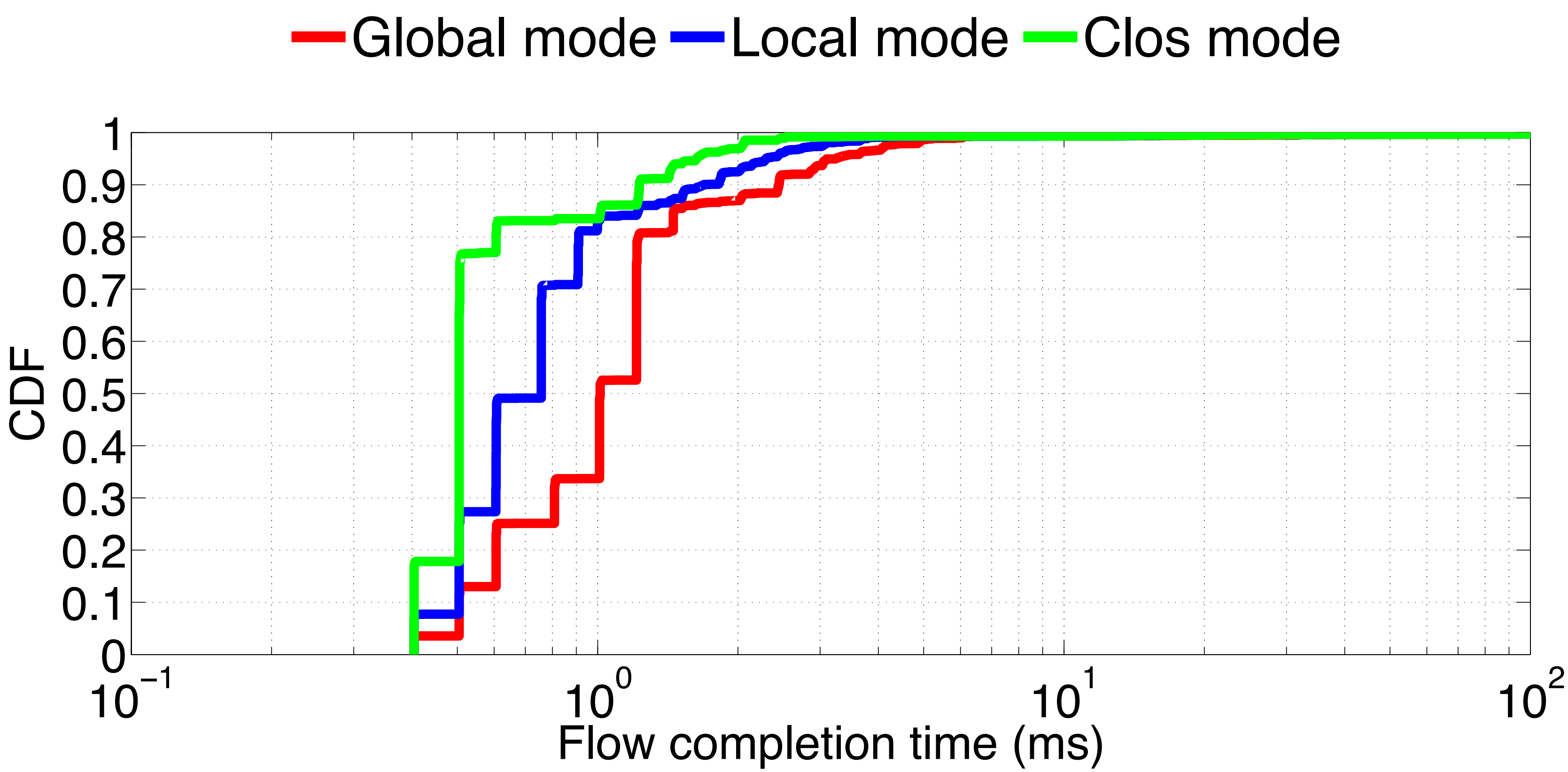
Evaluation

- Hadoop-1: no locality



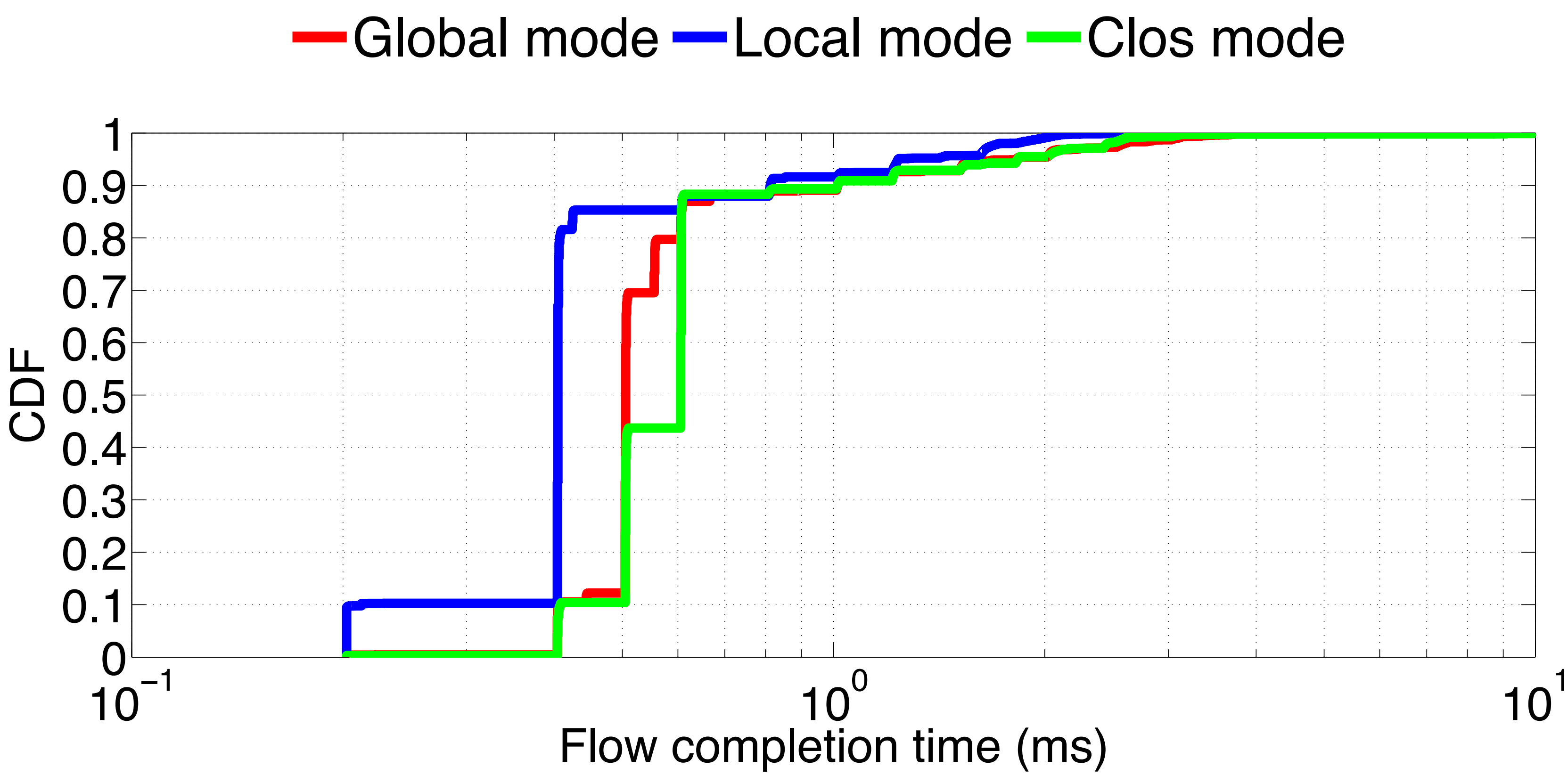
Evaluation

- Hadoop-2: rack-level locality



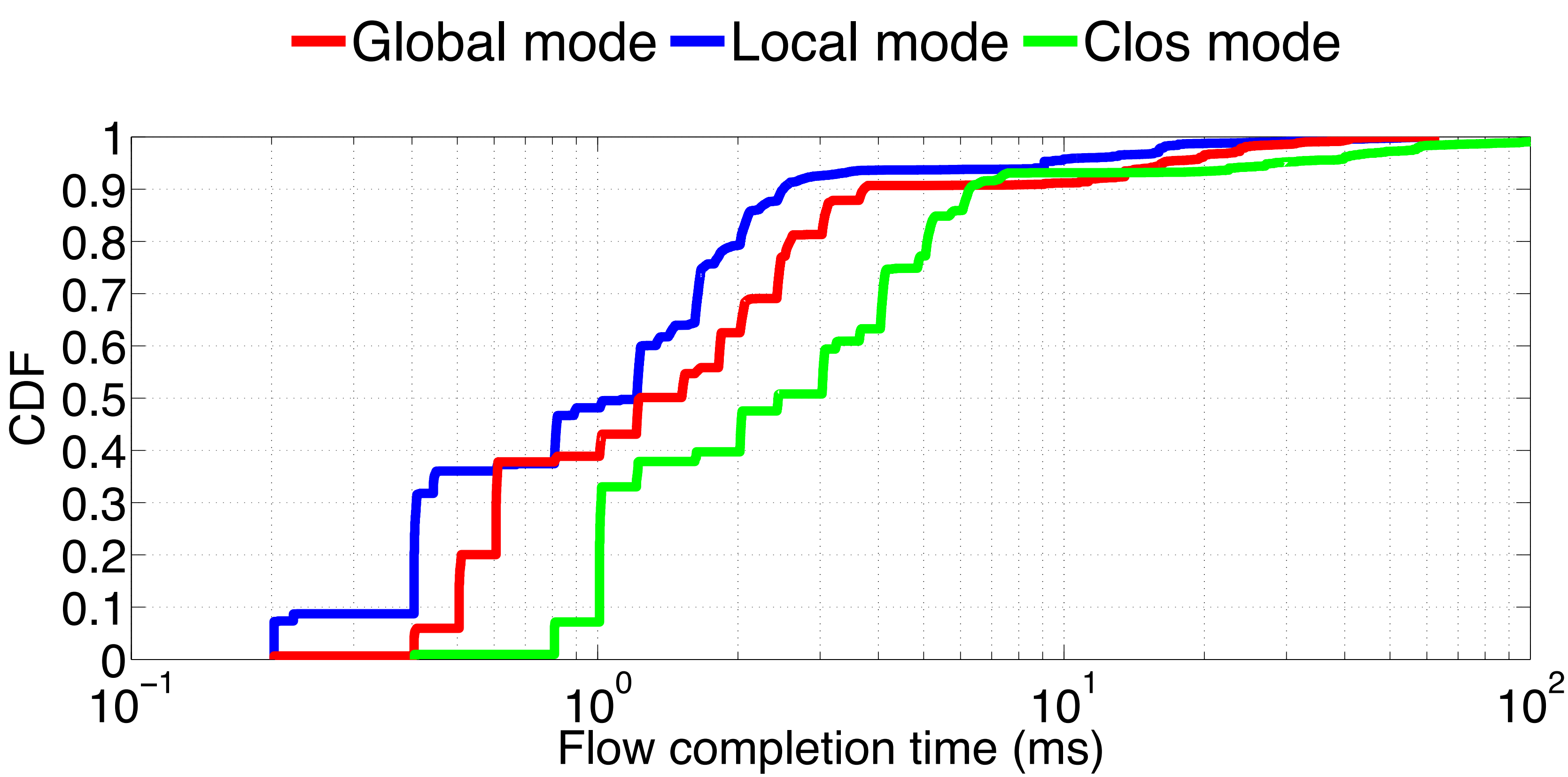
Evaluation

- Web: Pod-level locality



Evaluation

- Cache: Pod-level locality

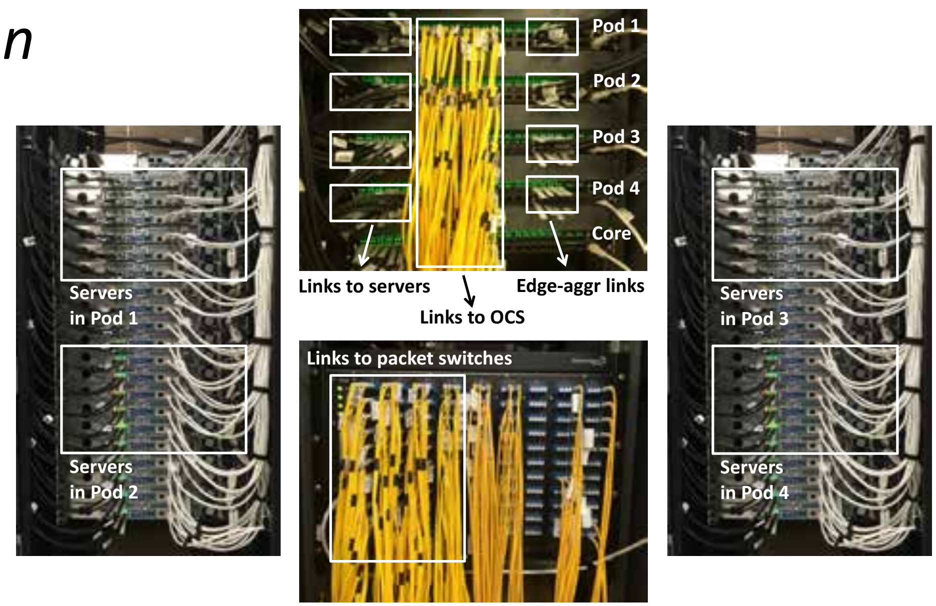


Evaluation

- Convertibility!!!
- Different topology for different workload
- Convert topology as workload changes
- Partition network into zones

Evaluation

- Theoretical performance
  - *Average path length*
  - *Throughput from Linear Programming solver*
- Effectiveness of k-shortest-path routing and MPTCP
  - *Throughput from simulation*
    ≈ LP solver
- Testbed implementation
  - *Hadoop & Spark*
  - *27.6% more bandwidth*
  - *10% less data read time*



Configurability vs. Convertibility

- Helios, c-Through, Flyway, OSA, 3DBeam, Mordia, FireFly, Quartz, WaveCube, ProjecToR, etc
- Different design philosophy

	Configurable network	Convertible network
Traffic to service	Instantaneous flows	Long-lasting workloads
Capacity	Add bandwidth	Better use of bandwidth
Topology change	Incremental & frequent	Network-wide & infrequent

Conclusion

- Convertible data center network architecture
- *Flat-tree* converts between Clos topology and approximate random graphs of different scales
- Complete architecture and control plane design
  - *Inexpensive converter switches*
  - *Distributed converter switches → scalability of architecture*
  - *Addressing + source routing → scalability of control plane*
- Extensive performance evaluation
  - *LP simulations*
  - *Packet-level simulations*
  - *Testbed implementation*