Spineless Datacenters

Vipul Harsh
UIUC

Sangeetha Abdu Jyothi
UC Irvine & VMware research

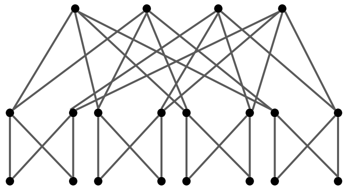
Brighten Godfrey
UIUC & VMware

HotNets 2020

Datacenter (DC) Topology

Hyperscale DC

Standard

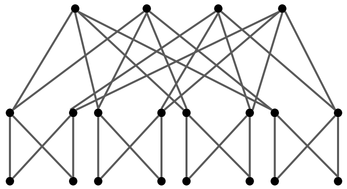


3-tier Fat-tree

Datacenter (DC) Topology

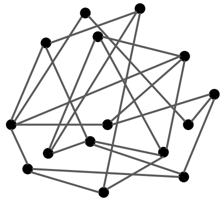
Hyperscale DC

Standard



3-tier Fat-tree

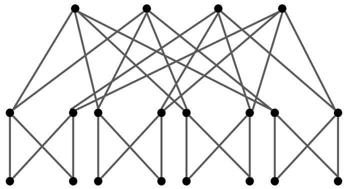
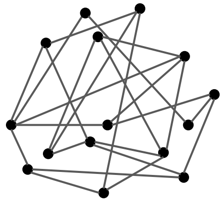
High performance



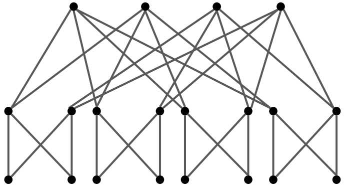
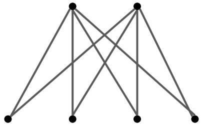
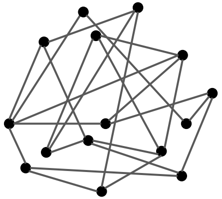
Expanders (e.g. Jellyfish)

Adoption restricted due to management/wiring complexity, non-traditional protocols

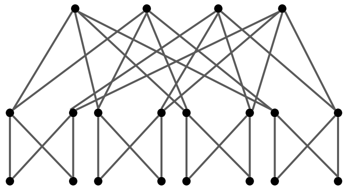
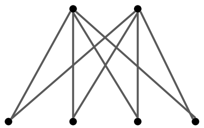
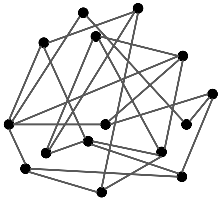
Datacenter (DC) Topology

	Hyperscale DC	Small-medium DC (<100 racks, <10K servers)
Standard	 3-tier Fat-tree	
High performance	 Expanders (e.g. Jellyfish) Adoption restricted due to management/wiring complexity, non-traditional protocols	

Datacenter (DC) Topology

	Hyperscale DC	Small-medium DC (<100 racks, <10K servers)
Standard	 3-tier Fat-tree	 2-tier Leaf-spine
High performance	 Expanders (e.g. Jellyfish) Adoption restricted due to management/wiring complexity, non-traditional protocols	

Datacenter (DC) Topology

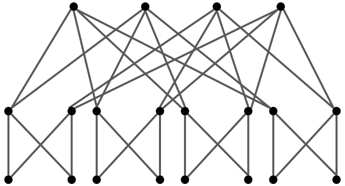
	Hyperscale DC	Small-medium DC (<100 racks, <10K servers)
Standard	 3-tier Fat-tree	 2-tier Leaf-spine
High performance	 Expanders (e.g. Jellyfish) Adoption restricted due to management/wiring complexity, non-traditional protocols	?

Datacenter (DC) Topology

	Hyperscale DC	Small-medium DC (<100 racks, <10K servers)
Standard	3-tier Fat-tree	2-tier Leaf-spine
High performance	Expanders (e.g. Jellyfish)	Our work

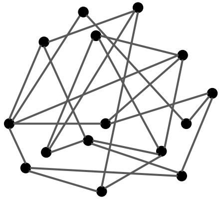
Hyperscale DC

Standard:



3-tier Fat-tree

High performance:

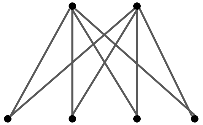


Expanders (e.g. Jellyfish)

Adoption restricted due to management/wiring complexity, non-traditional protocols

Small-medium DC
(<100 racks, <10K servers)

Standard:



2-tier Leaf-spine

High performance:

Our work

- Are there more efficient topologies at small scale?
- Can we make them practical?
  - routing
  - management/wiring complexity

Candidates for efficient topologies at small scale

- Expanders: maximally “connected” graphs
  - High performance, especially at large scale
  - Provably near-optimal as $n \rightarrow \infty$
  - Not obvious if they’re better than leaf-spines (since leaf-spine has shorter path length than 3-tier Clos)

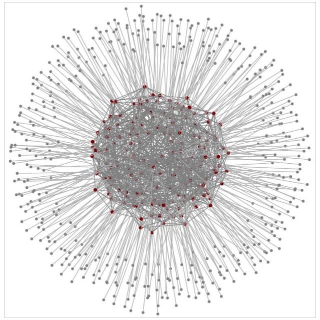
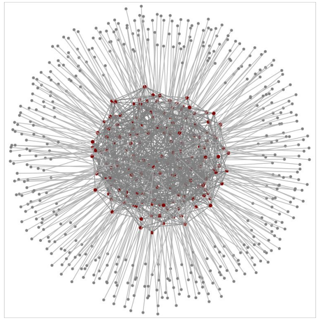


Image borrowed from the Jellyfish talk, NSDI 2012

Candidates for efficient topologies at small scale

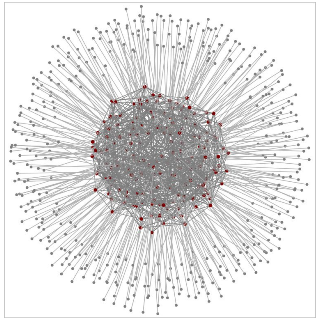
- Expanders: maximally “connected” graphs
    - High performance, especially at large scale
    - Provably near-optimal as $n \to \infty$
    - Not obvious if they’re better than leaf-spines (since leaf-spine has shorter path length than 3-tier Clos)



- **Other candidates?**

Image borrowed from the Jellyfish talk, NSDI 2012

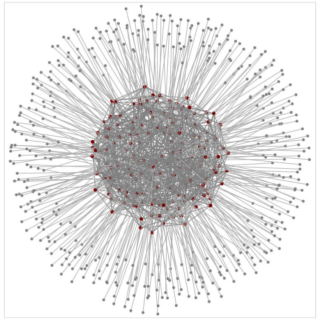
What are the reasons for expanders' high performance?



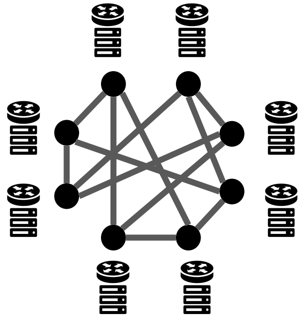
1. **Expansion**: how "well connected" the graph is
   - Results in shorter paths → less resource utilization per unit throughput
   - Helps in keeping traffic well-balanced across the network

First image borrowed from the Jellyfish talk, NSDI 2012

What are the reasons for expanders' high performance?



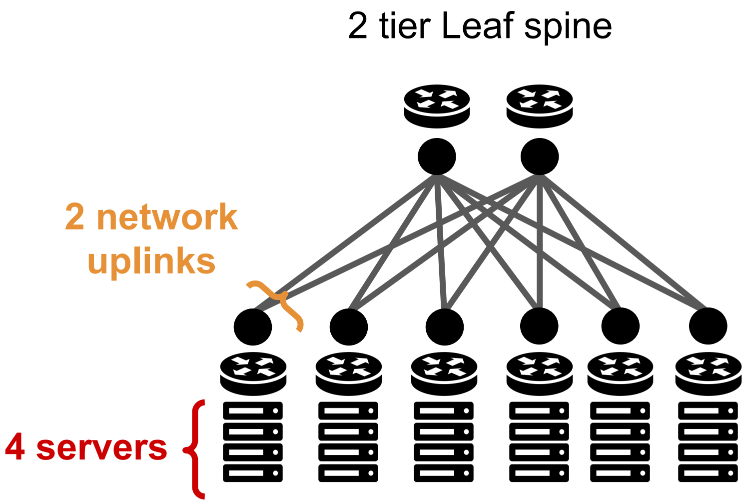
1. **Expansion**: how "well connected" the graph is
   - Results in shorter paths → less resource utilization per unit throughput
   - Helps in keeping traffic well-balanced across the network



2. **Flatness**: servers evenly distributed across all switches
   - Even distribution → Helps in alleviating hotspots

First image borrowed from the Jellyfish talk, NSDI 2012

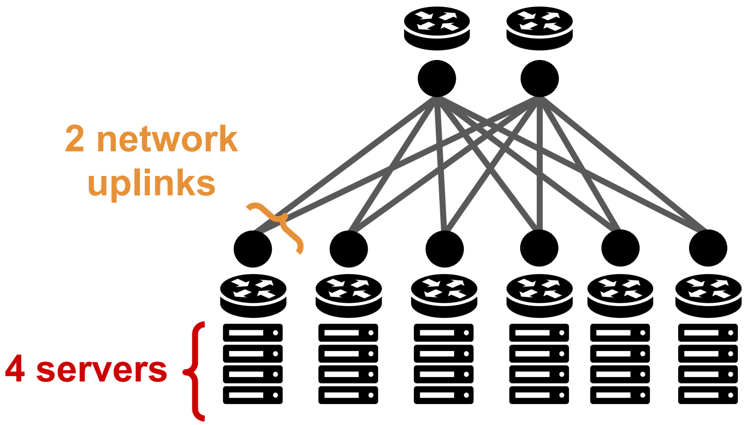
Analyzing benefit of flatness



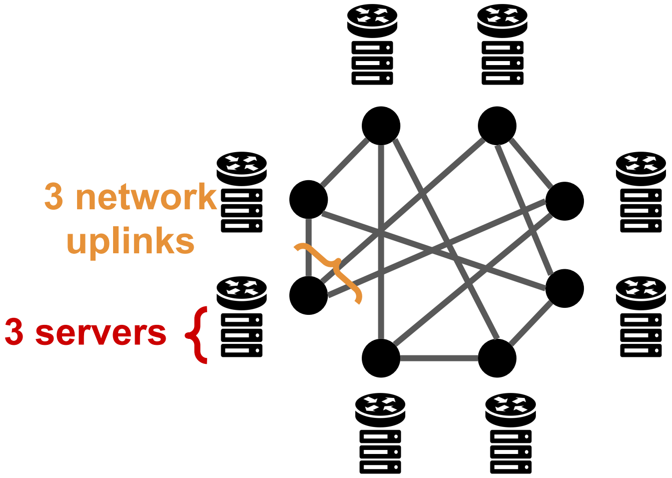
Network uplinks/**S**erver in a rack (**NS Ratio**)
= 2 network links / 4 servers = 0.5

Analyzing benefit of flatness

2 tier Leaf spine



Flat topology: ToRs are directly connected



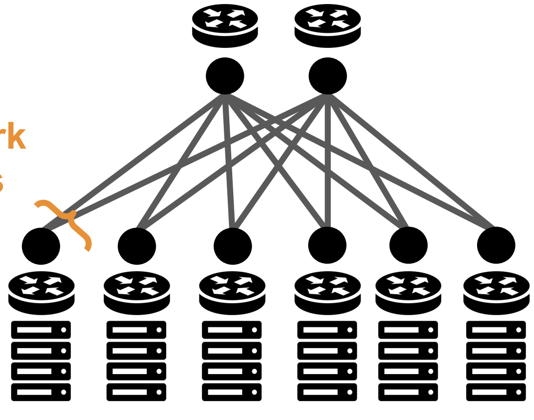
Network uplinks/**S**erver in a rack (**NS Ratio**)
= 2 network links / 4 servers = 0.5

NS Ratio = 3 network links/ 3 servers = 1

Quantifying benefit of flatness

2 tier Leaf spine

2 network uplinks

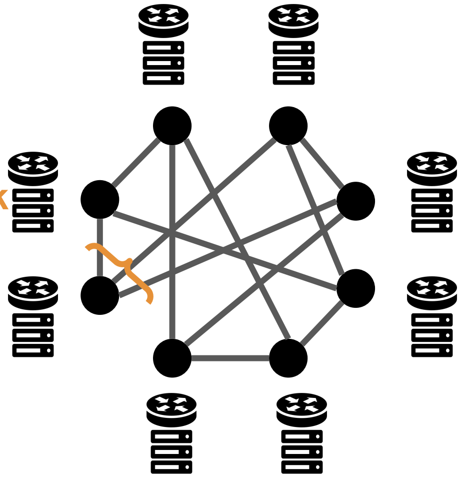


4 servers

Network uplinks/Server in a rack (**NS Ratio**)
= 2 network links / 4 servers = 0.5

Flat topology: ToRs are directly connected

3 network uplinks



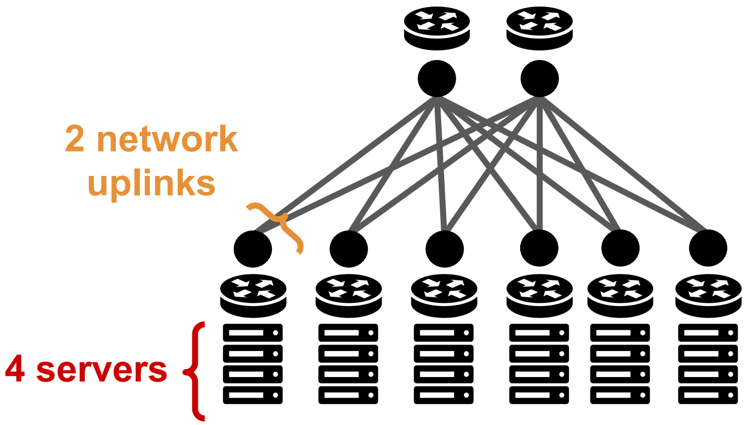
3 servers

NS Ratio = 3 network links/ 3 servers = 1

2 times more network uplinks per server
(vs any leaf-spine, x leafs y spines)

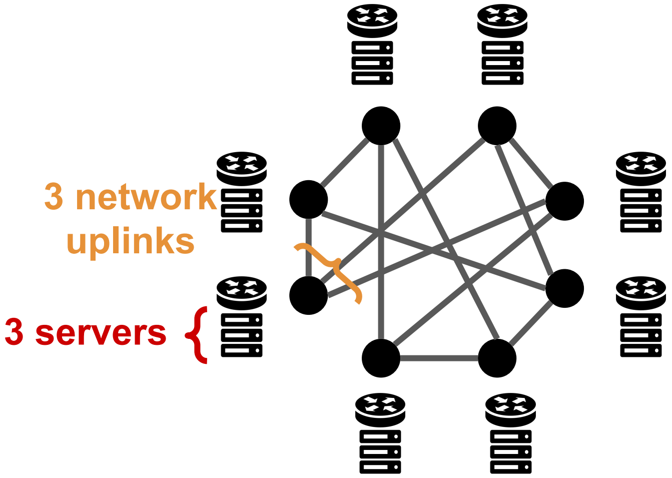
Analyzing benefit of flatness

2 tier Leaf spine



Network uplinks/Server in a rack (NS Ratio)
= 2 network links / 4 servers = 0.5

Flat topology: ToRs are directly connected



NS Ratio = 3 network links/ 3 servers = 1

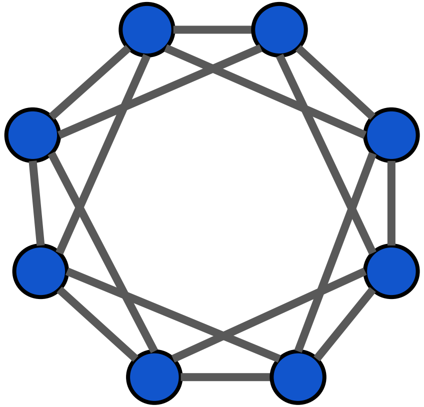
2 times more network uplinks per server

(vs any leaf-spine, x leafs y spines)

Flat networks effectively mask oversubscription when bottleneck is at ToR network links

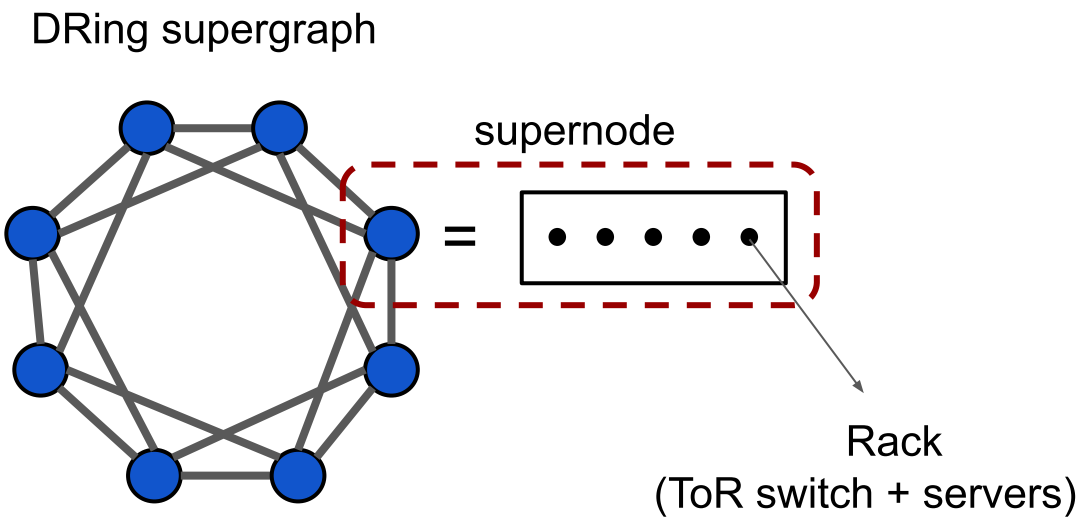
DRing: a simple flat network

DRing supergraph

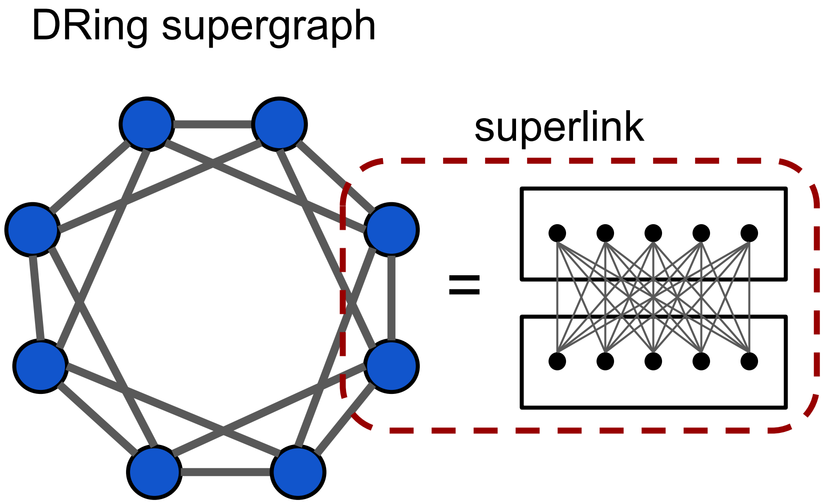


supernode (i) is connected to
supernodes (i+1) and (i+2)

DRing: a simple flat network

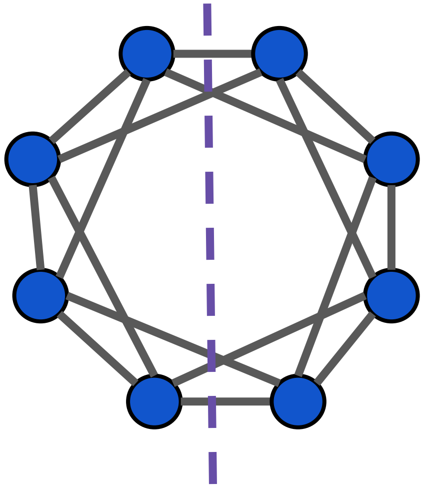


DRing: a simple flat network



DRing: a simple flat network

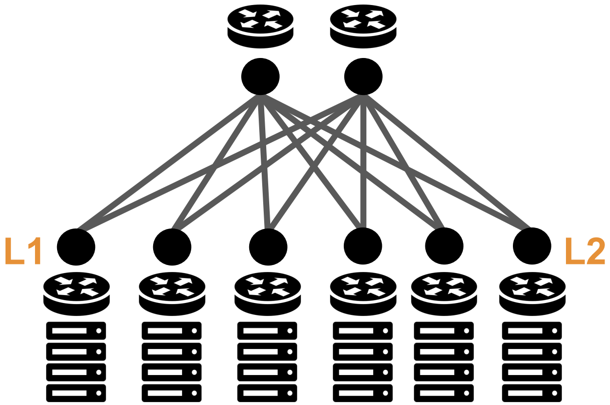
DRing supergraph



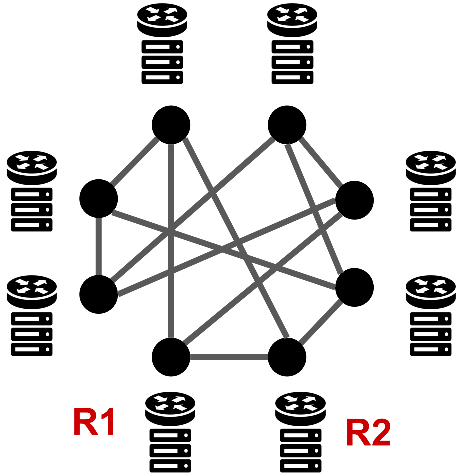
Bisection bandwidth is O(n)
worse than an expander!

Routing design

Shortest paths not enough for flat topologies

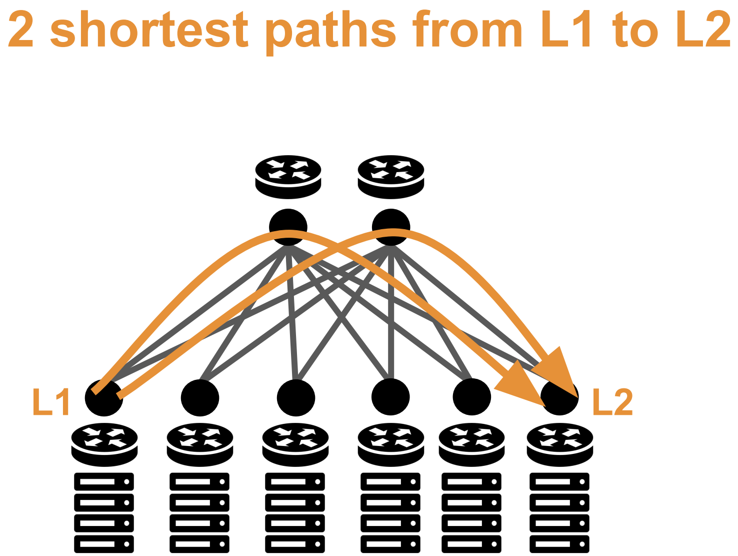


2 tier leaf-spine

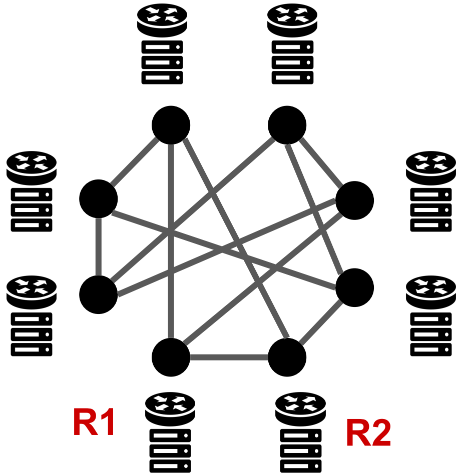


Flat topology

Shortest paths not enough for flat topologies

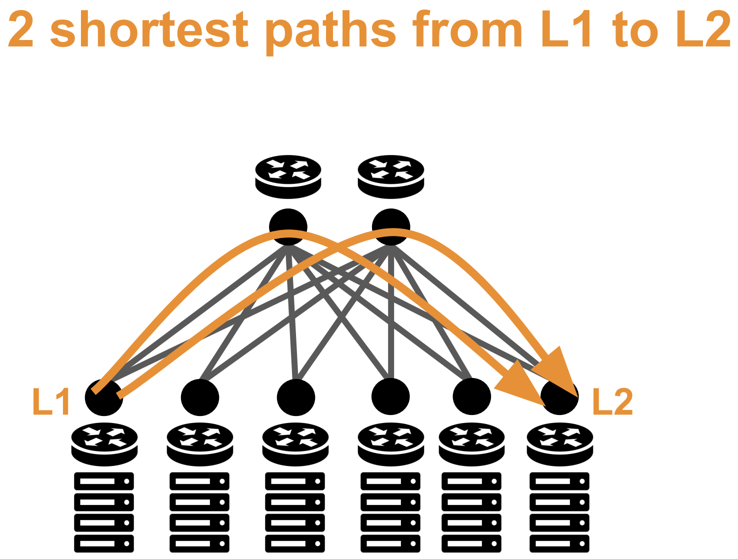


2 tier leaf spine

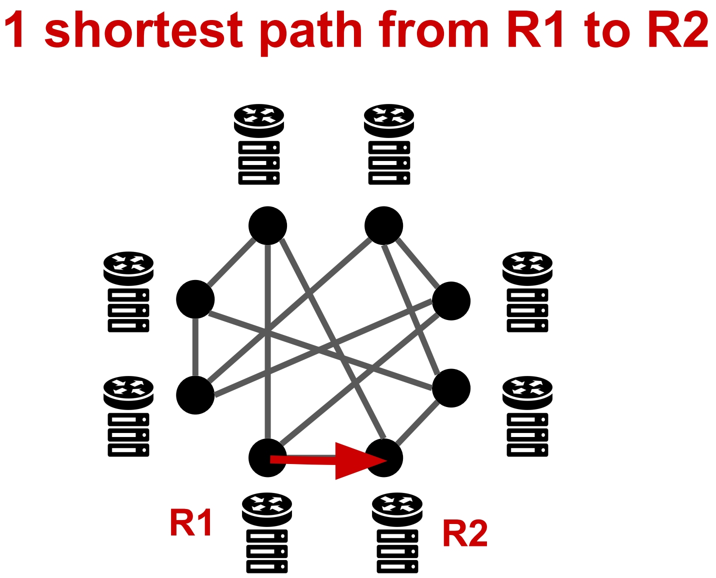


Flat topology

Shortest paths not enough for flat topologies

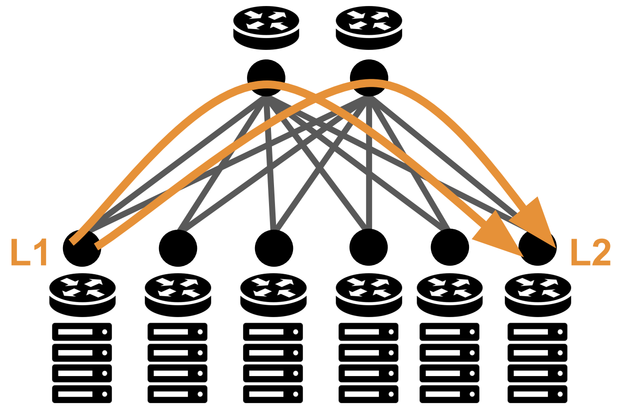


2 tier leaf-spine

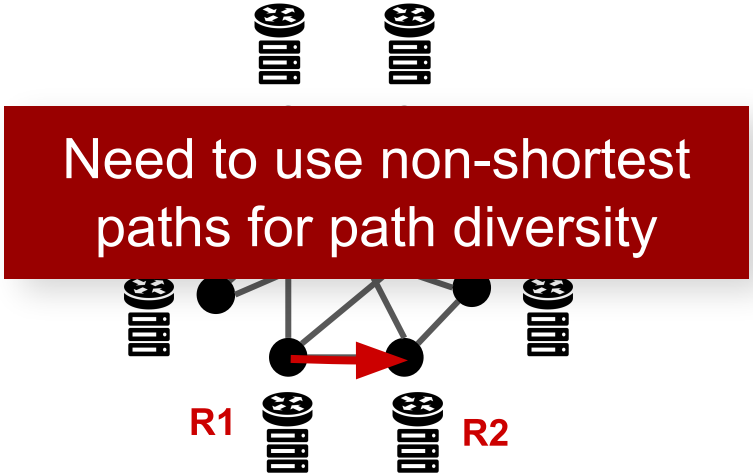


Flat topology

Shortest paths not enough for flat topologies

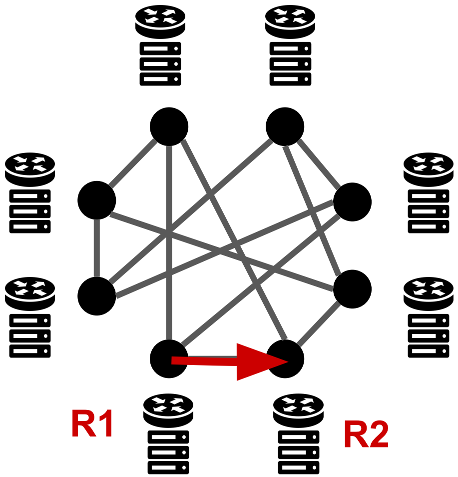


2 tier leaf-spine



Flat topology

Past routing schemes for flat networks

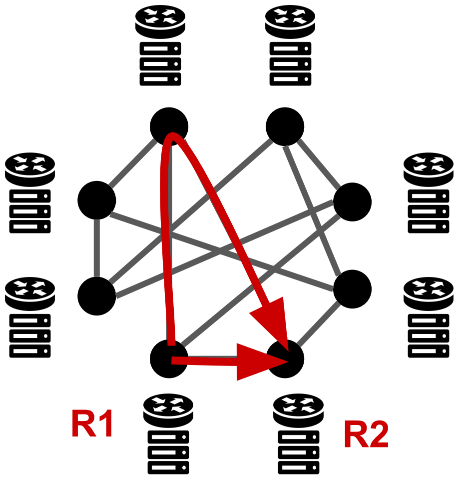


- K-shortest paths + MPTCP [1,2]
- Valiant routing + ECMP + flowlet switching [3]
- Dynamic fluid routing [4]

Require changes to hardware or control/data plane or endpoint OS

[1] Singla et. al., Jellyfish, NSDI 2012
[2] Valadarsky et. al., Xpander, CoNext 2016
[3] Kassing et. al., Beyond fat-trees without antennae, mirrors, and disco-balls, SIGCOMM 2017
[4] Jyothi et. al., Measuring and Understanding Throughput of Network Topologies, SC 2016

Our proposal: Shortest-Union(K) routing



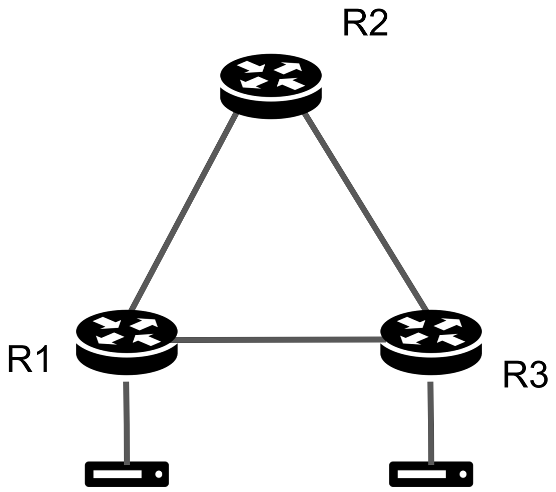
Shortest-Union(2)

Use all paths from R1 to R2 which are either

(a) Shortest paths
(b) or length(path) <= K

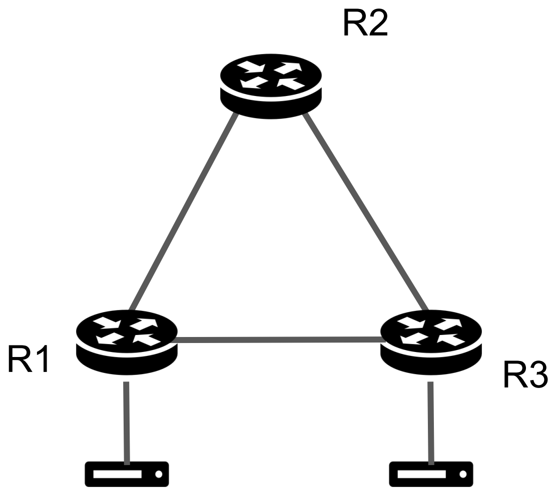
Prototype implementation on GNS3 on emulated Cisco 7200 routers, with BGP and VRFs

Shortest-Union(2): Implementation with BGP and VRFs

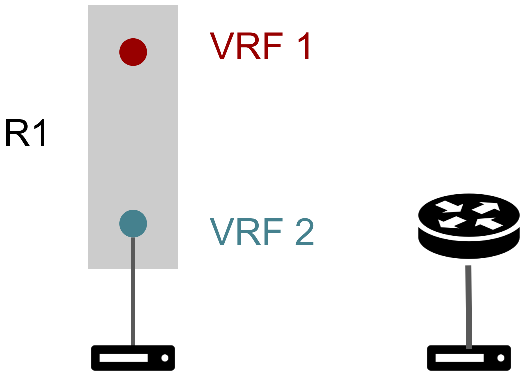


Route traffic from R1 to R3

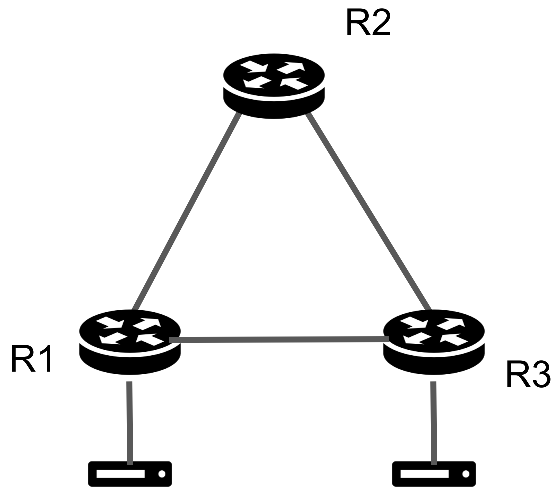
Shortest-Union(2): Implementation with BGP and VRFs



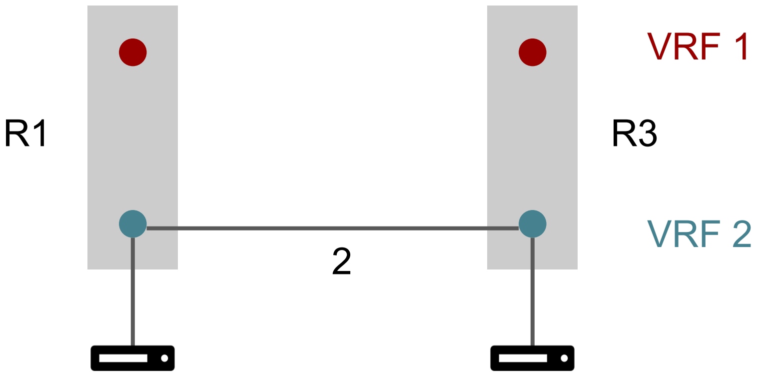
Route traffic from R1 to R3



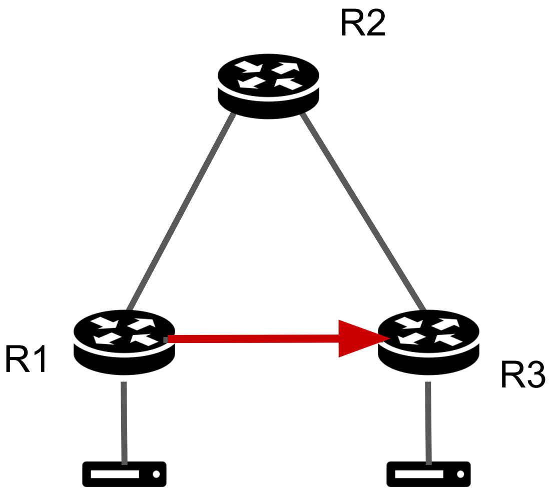
Shortest-Union(2): Implementation with BGP and VRFs



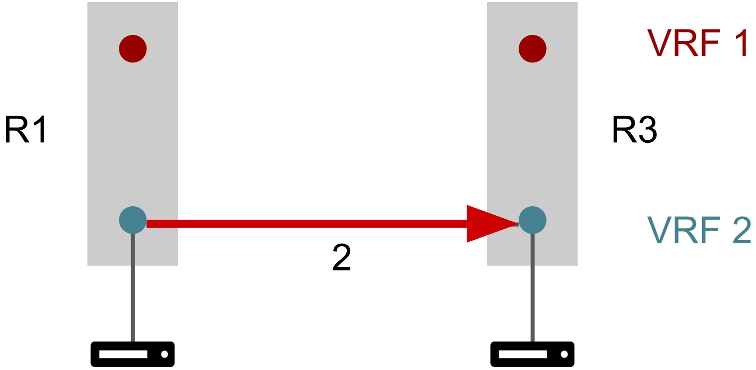
Route traffic from R1 to R3



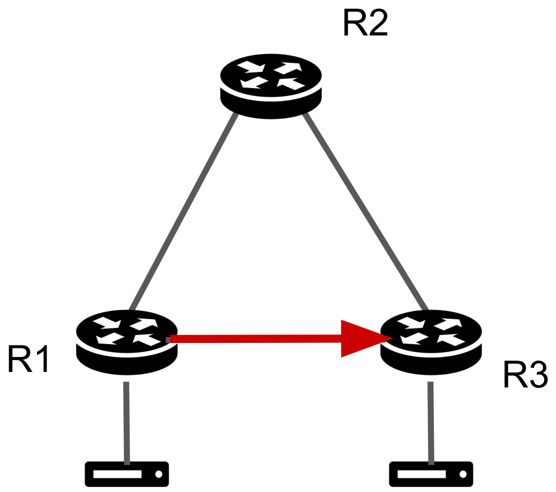
Shortest-Union(2): Implementation with BGP and VRFs



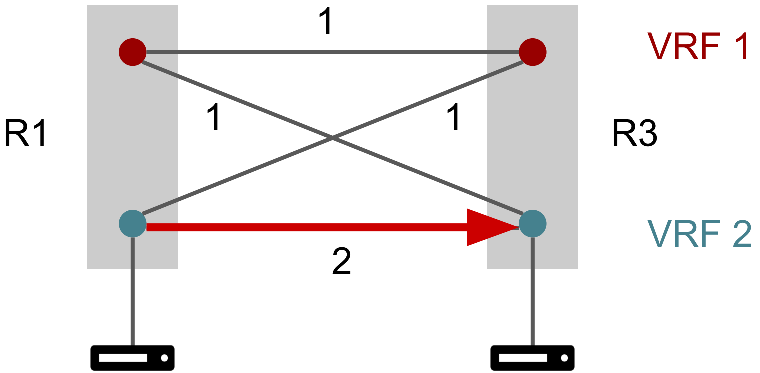
Route traffic from R1 to R3



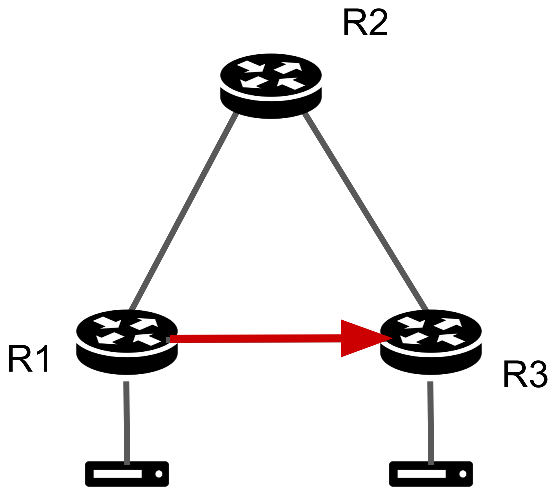
Shortest-Union(2): Implementation with BGP and VRFs



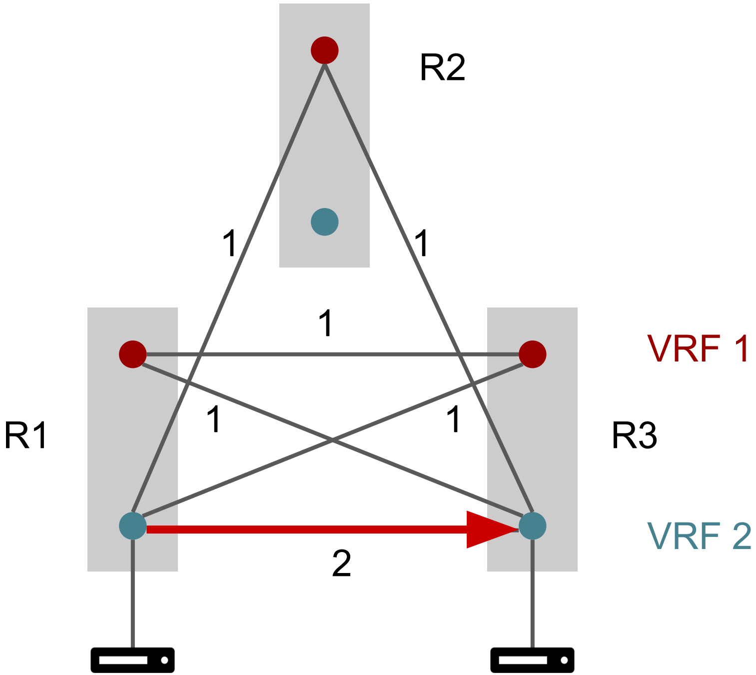
Route traffic from R1 to R3



Shortest-Union(2): Implementation with BGP and VRFs

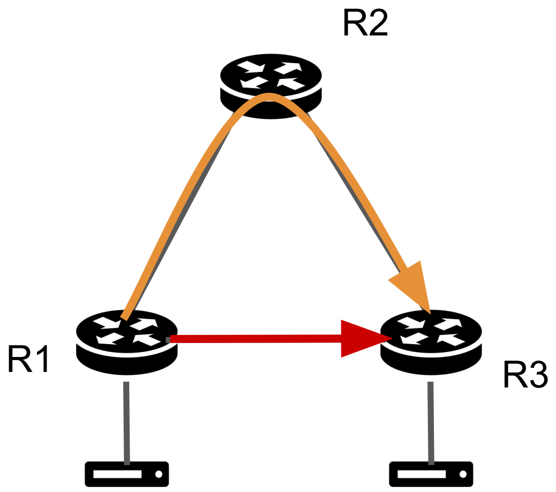


Route traffic from R1 to R3

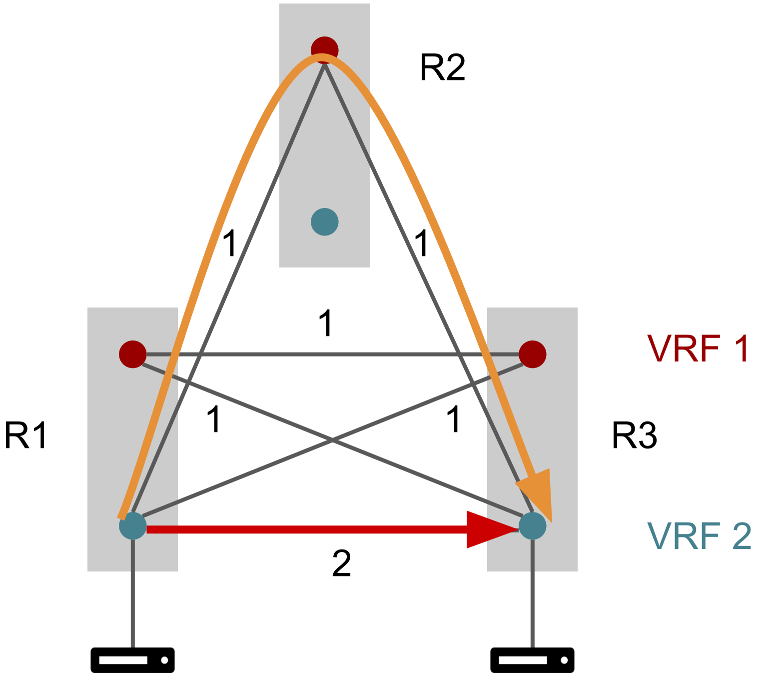


Not all connections are shown.

Shortest-Union(2): Implementation with BGP and VRFs



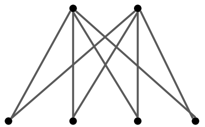
Route traffic from R1 to R3



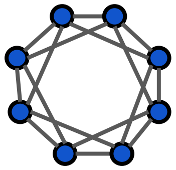
Not all connections are shown.

Evaluation

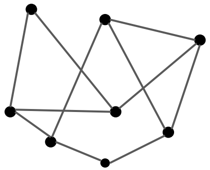
Topologies



Leaf-spine
16 spines, 64 racks, 3072 servers
(a recommended config from Arista)



DRing
80 racks, 2988 servers



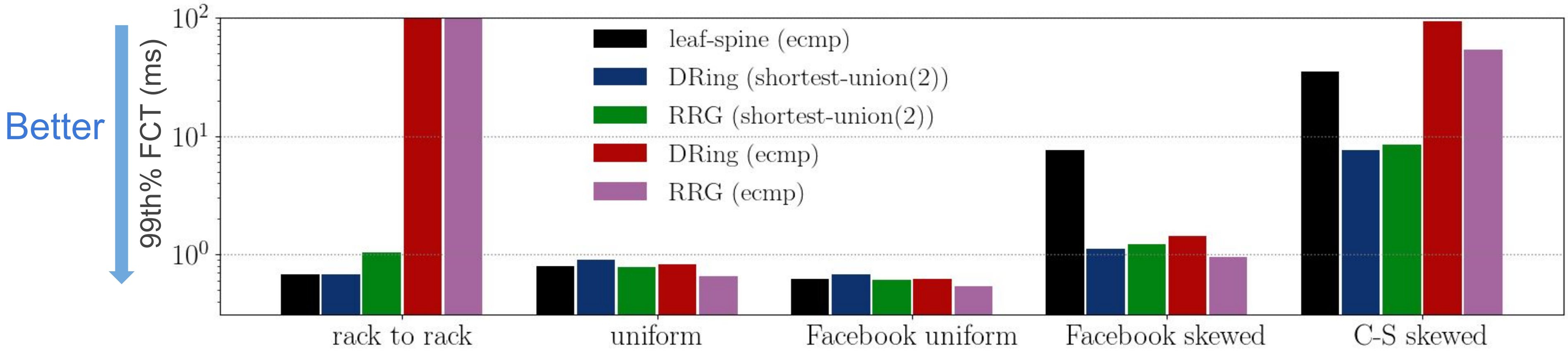
Expander: Random regular graph (RRG)
80 racks, 3072 servers

Evaluation goals

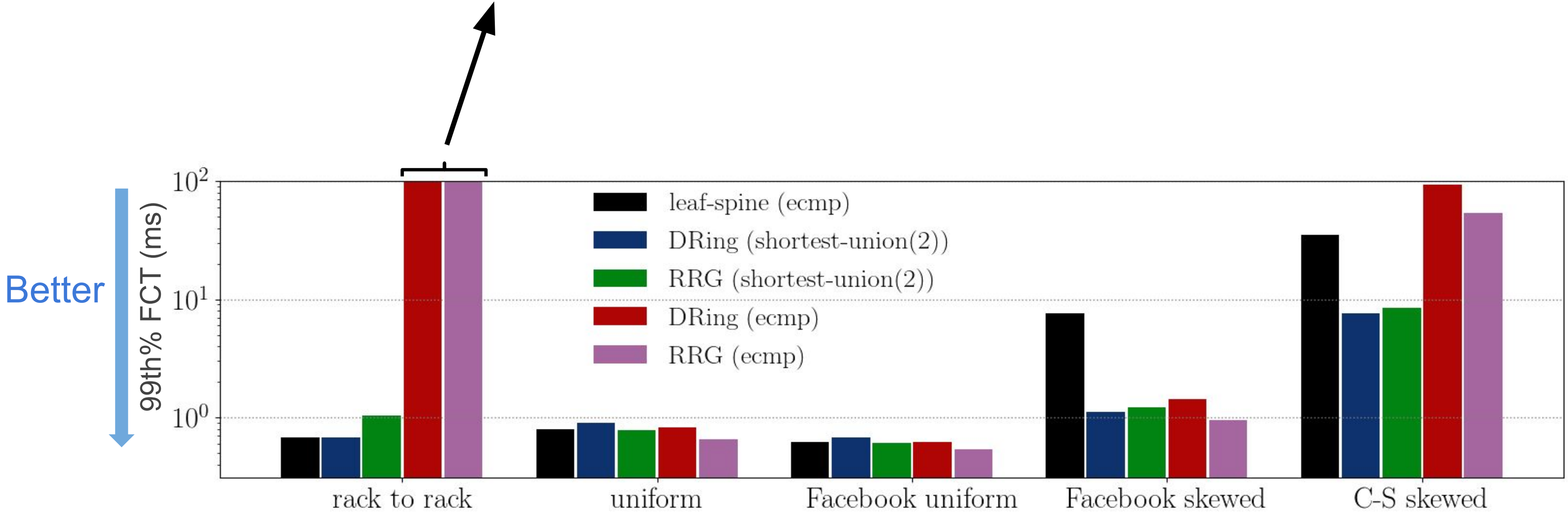
Can flat topologies (DRing, RRG) outperform leaf-spine?

Are there classes of topologies, besides expanders, that work well at small scale?

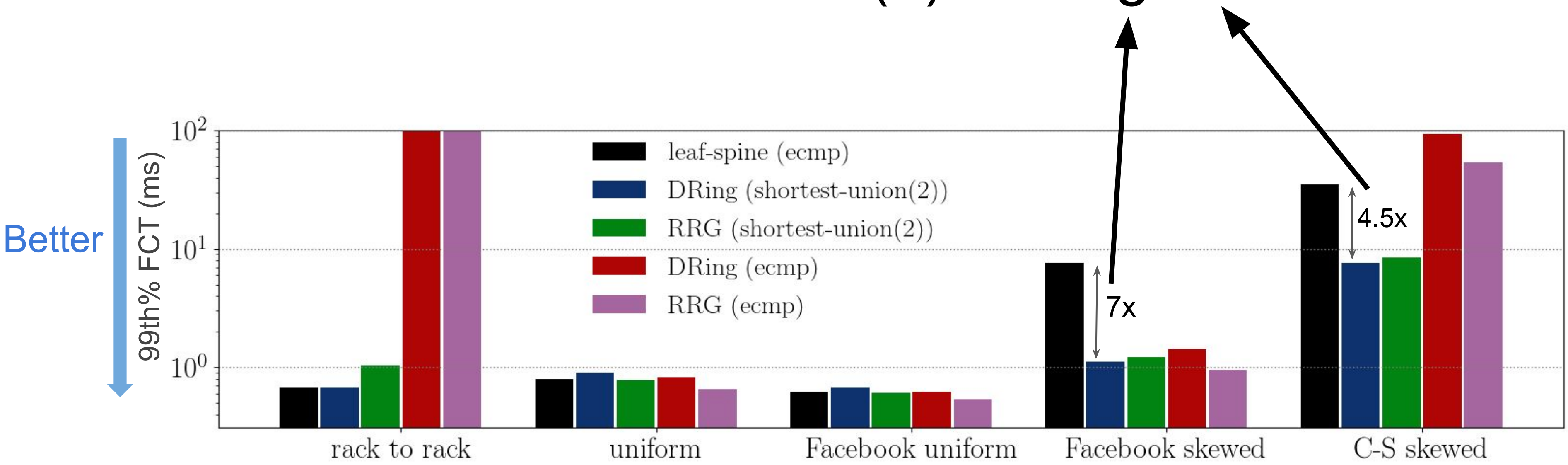
Can flat topologies (DRing, RRG) outperform leaf-spine?


Better
99th% FCT (ms)
10^2
10^1
10^0
leaf-spine (ecmp)
DRing (shortest-union(2))
RRG (shortest-union(2))
DRing (ecmp)
RRG (ecmp)
rack to rack
uniform
Facebook uniform
Facebook skewed
C-S skewed

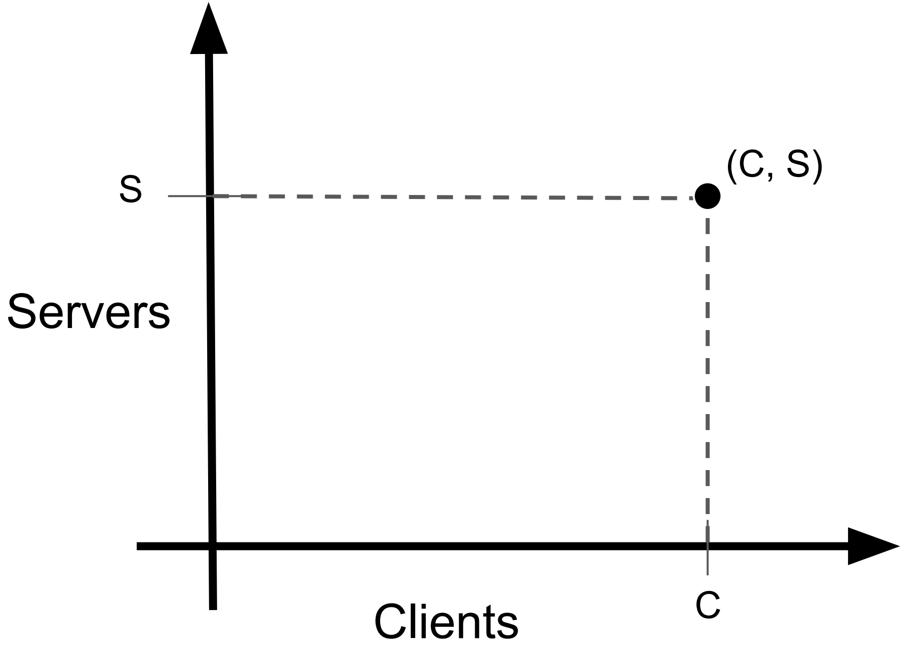
(Flat networks + ECMP) don't work in some cases



Big improvement for skewed traffic with shortest-union(2) routing



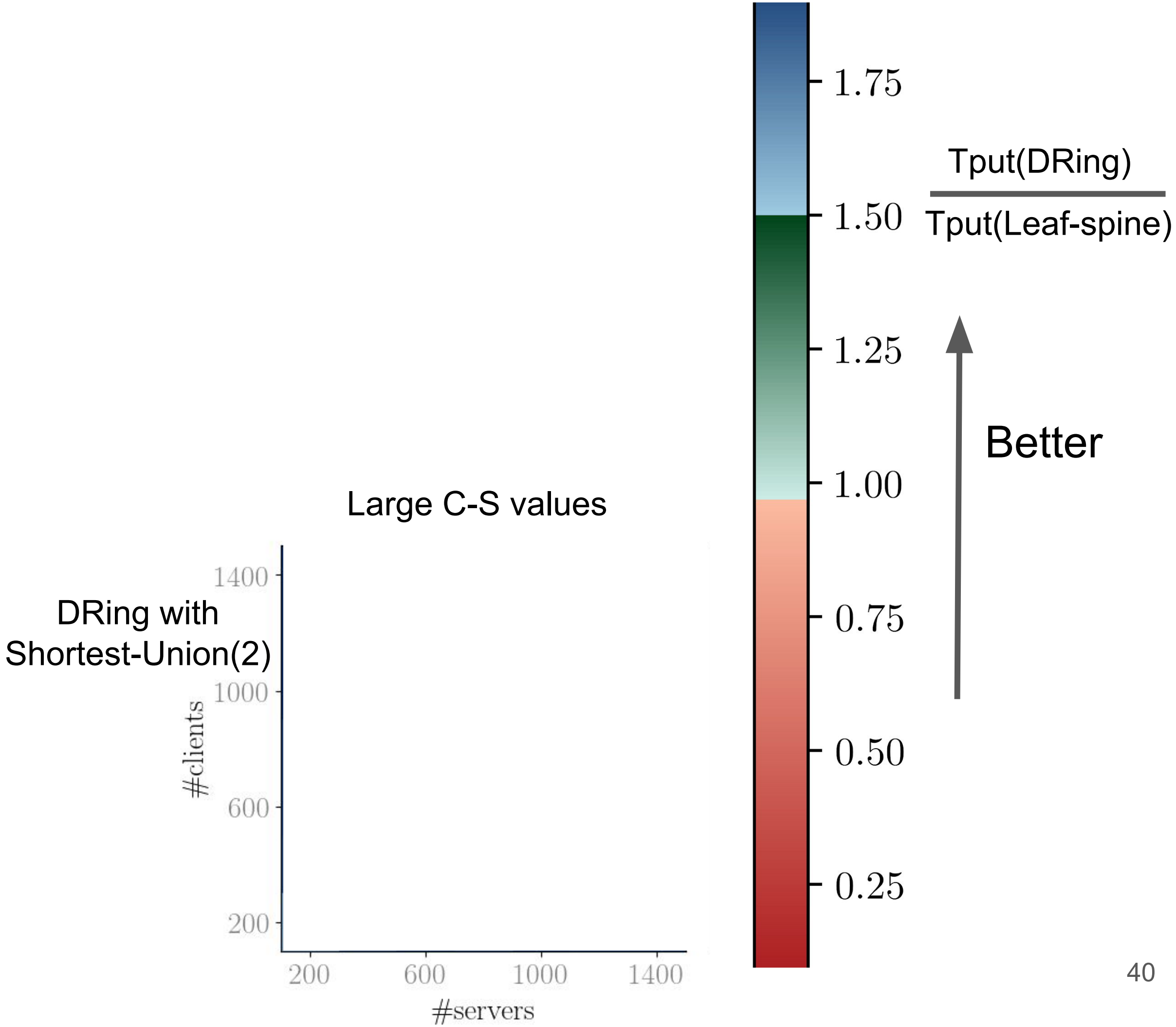
Throughput in the C-S model

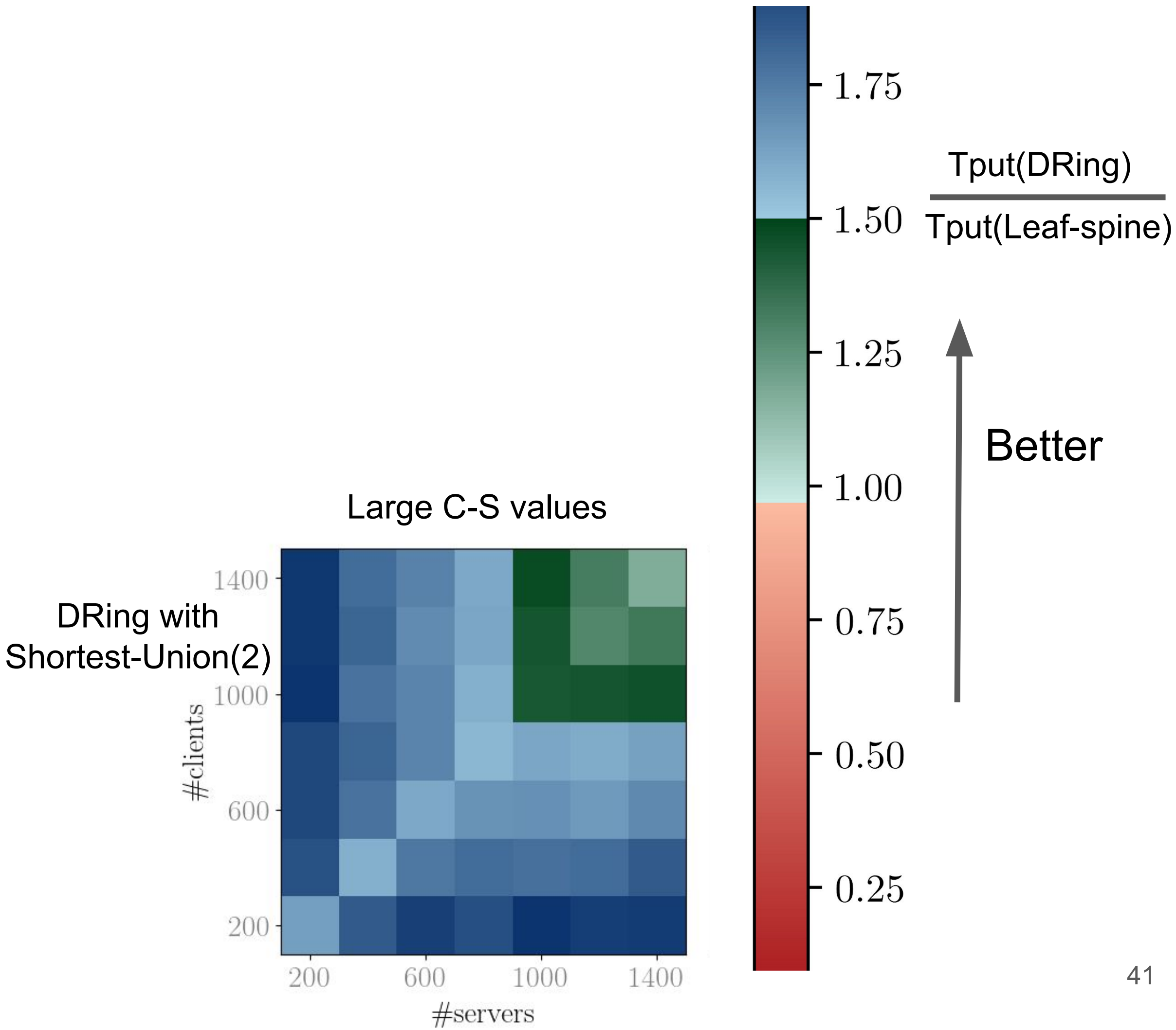


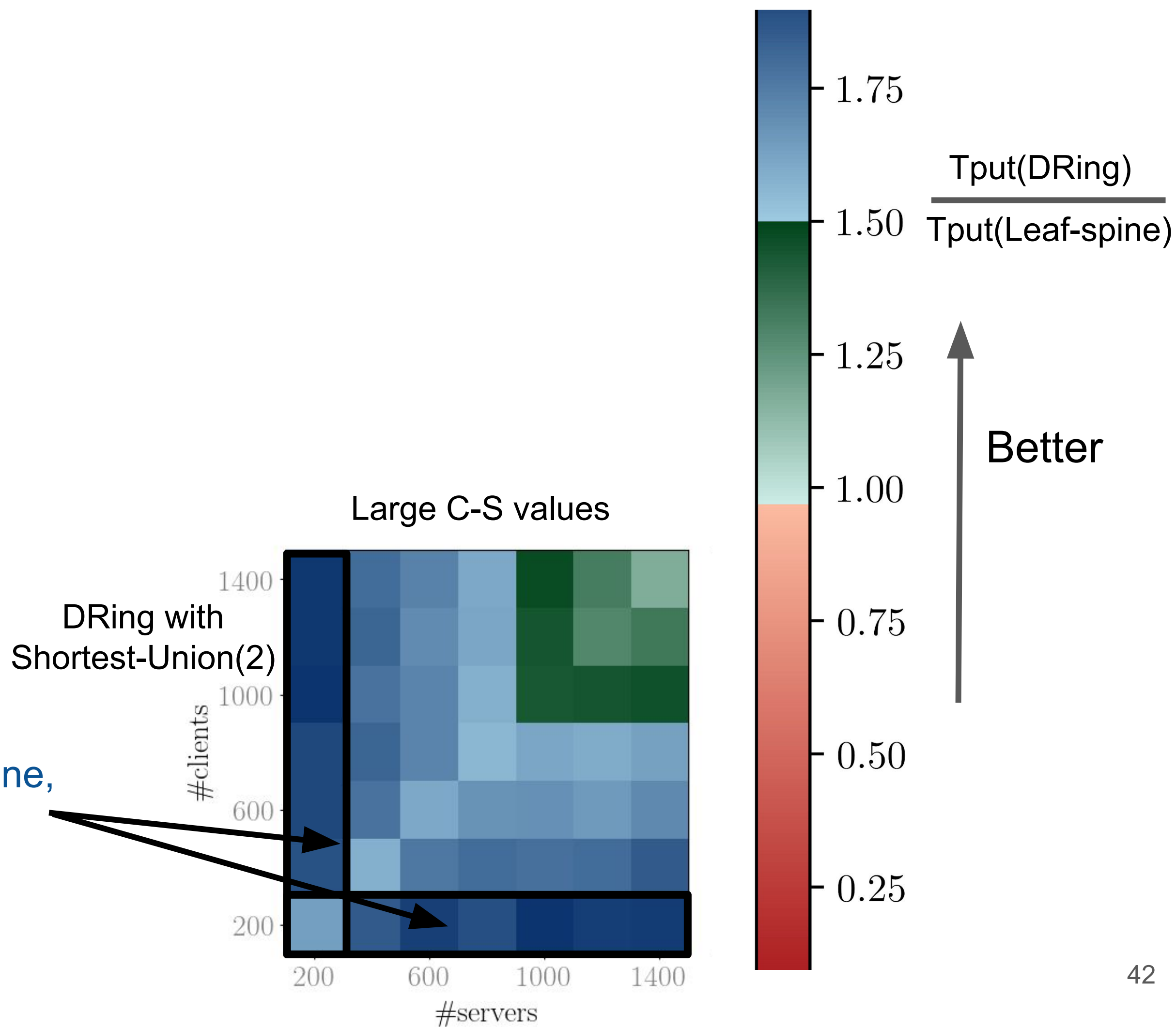
C-S traffic pattern

C client hosts send to S server hosts

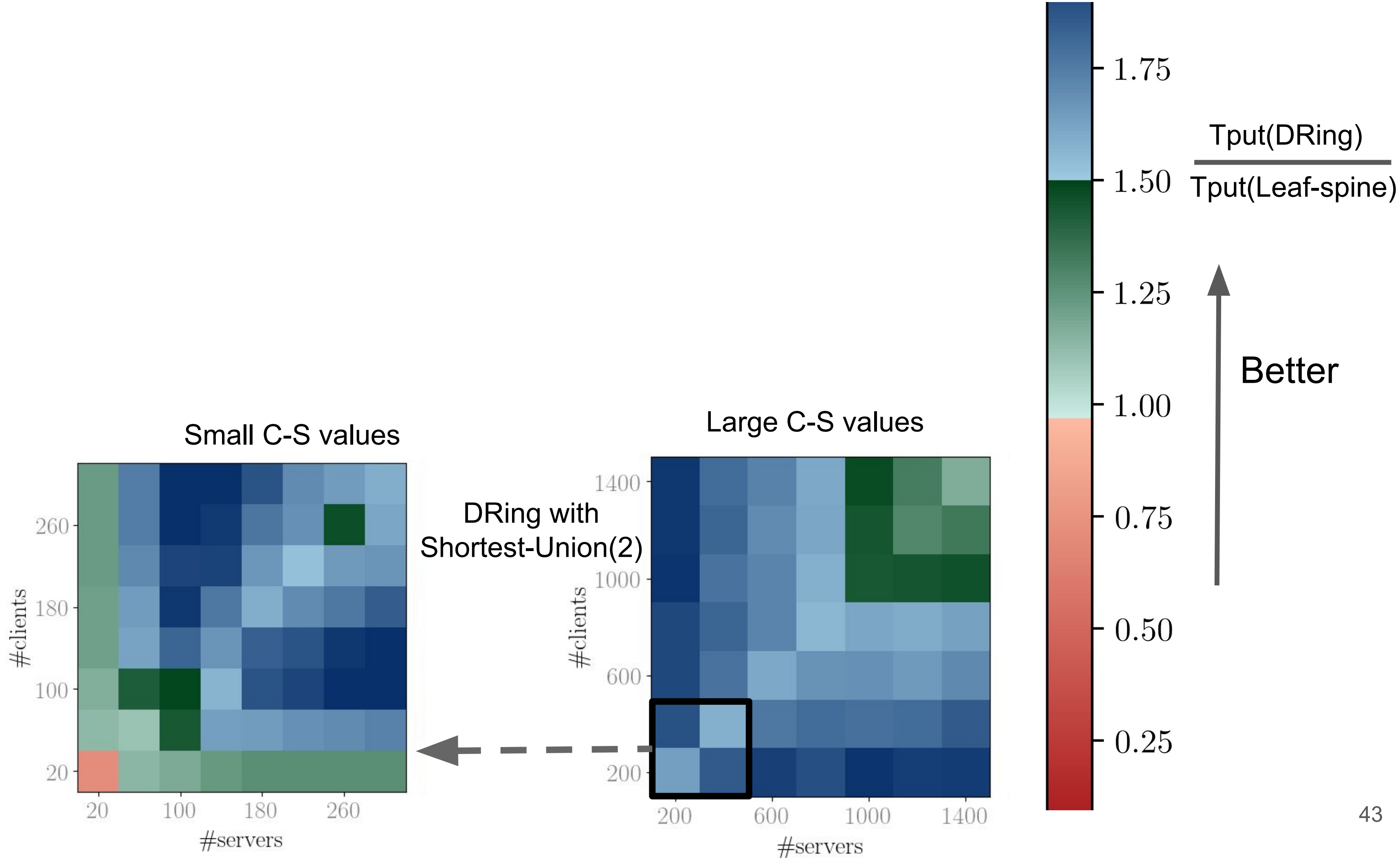
- Incast: C>>1, S=1
- Outcast: C=1, S>>1
- Uniform traffic: C = n/2, S = n/2
- Skewed: C >> S (or vice-versa)
- Rack-to-rack: C = S = #hosts in a rack


Large C-S values
DRing with
Shortest-Union(2)
1400
1000
600
200
#clients
200
600
1000
1400
#servers
1.75
1.50
1.25
1.00
0.75
0.50
0.25
Tput(DRing)
Tput(Leaf-spine)
Better

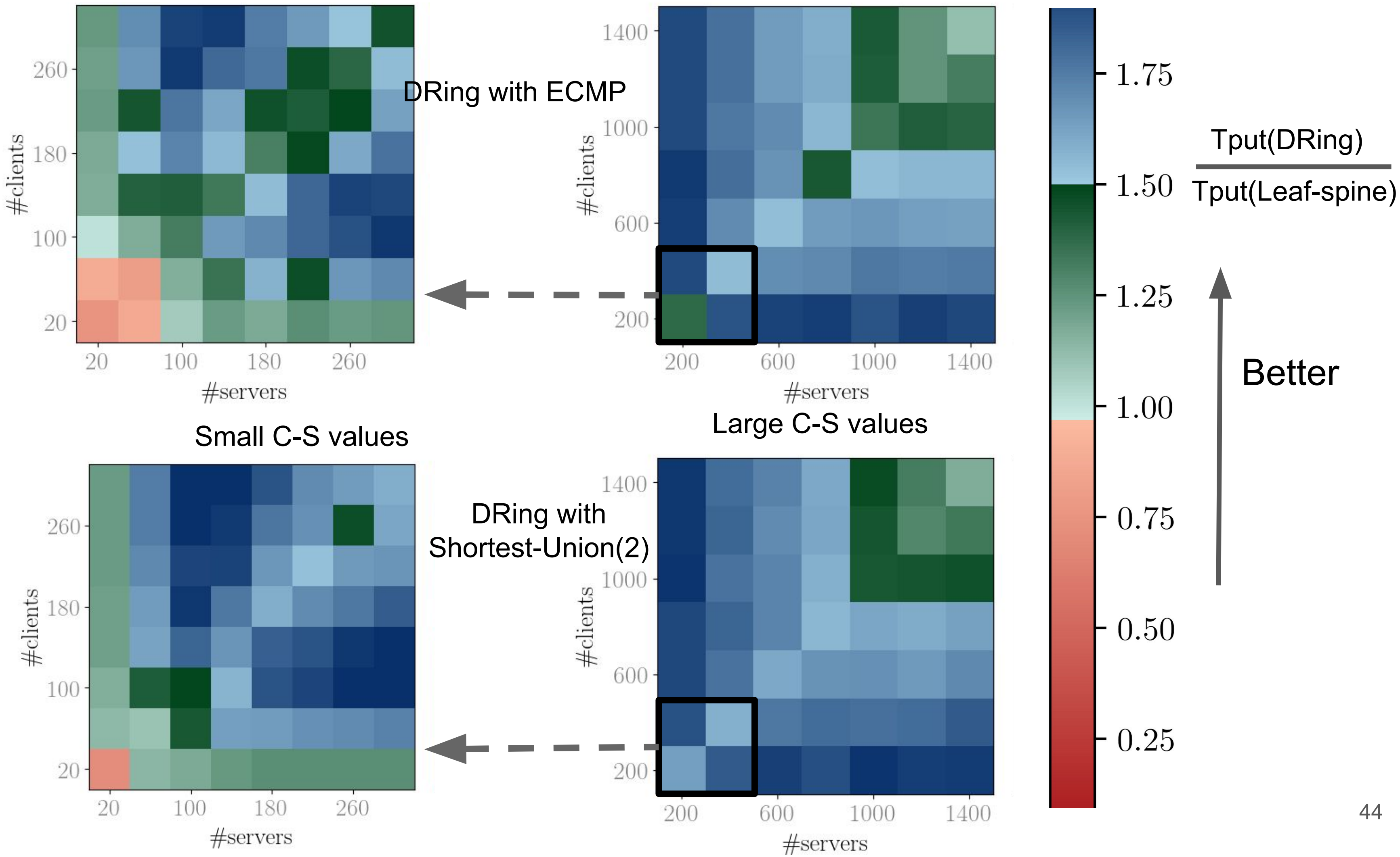

Large C-S values
DRing with
Shortest-Union(2)
#clients
1400
1000
600
200
#servers
200
600
1000
1400
1.75
1.50
1.25
1.00
0.75
0.50
0.25
Tput(DRing)
Tput(Leaf-spine)
Better



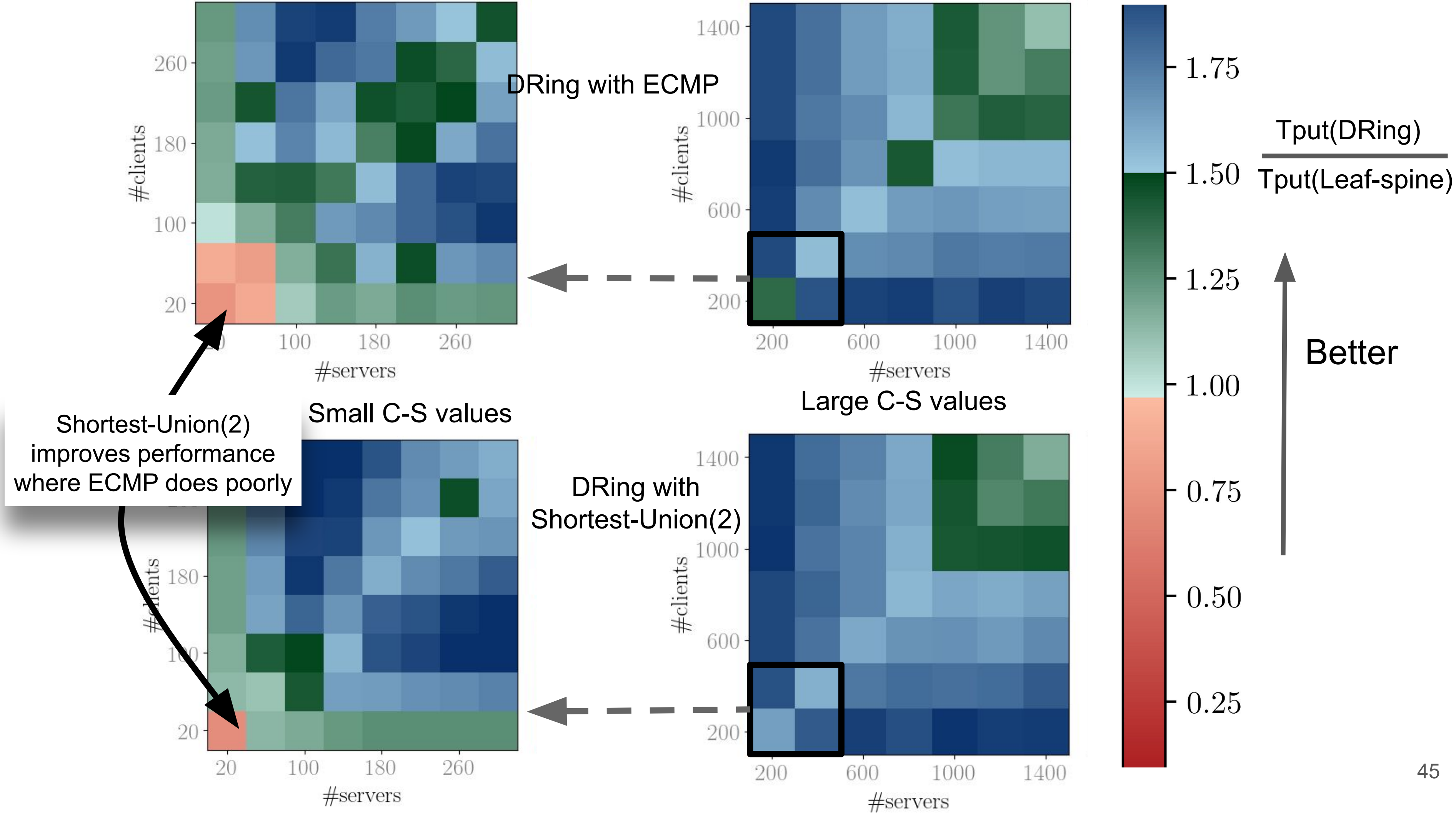
For skewed traffic, C>>S or S>>C, DRing's throughput is ~2x of leaf-spine, (as predicted by our analysis)



Small C-S values
Large C-S values
DRing with
Shortest-Union(2)
#clients
#servers
260
180
100
20
20
100
180
260
1400
1000
600
200
200
600
1000
1400
1.75
1.50
1.25
1.00
0.75
0.50
0.25
Tput(DRing)
Tput(Leaf-spine)
Better



DRing with ECMP
DRing with Shortest-Union(2)
Small C-S values
Large C-S values
#clients
#servers
Tput(DRing)
Tput(Leaf-spine)
Better


DRing with ECMP
Small C-S values
Large C-S values
DRing with
Shortest-Union(2)
Shortest-Union(2)
improves performance
where ECMP does poorly
#clients
#servers
Tput(DRing)
Tput(Leaf-spine)
Better
1.75
1.50
1.25
1.00
0.75
0.50
0.25

Are there classes of topologies, besides expanders, that work well at small scale?

DRing: Performance deteriorates with scale

Better

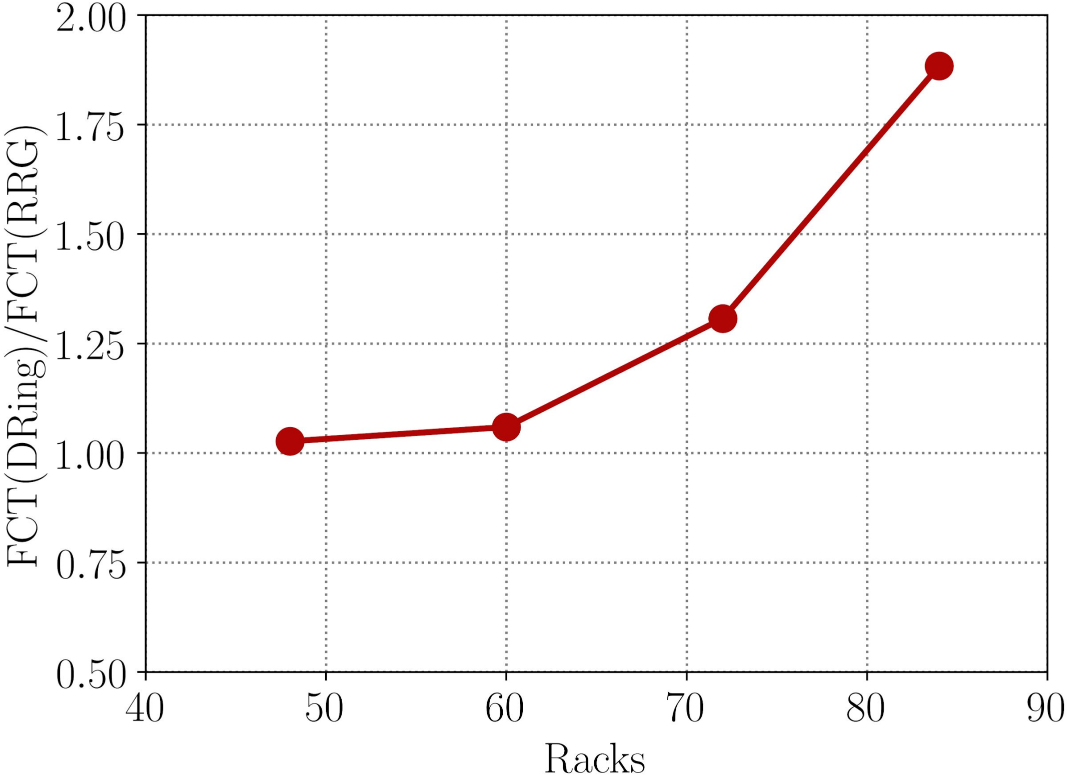
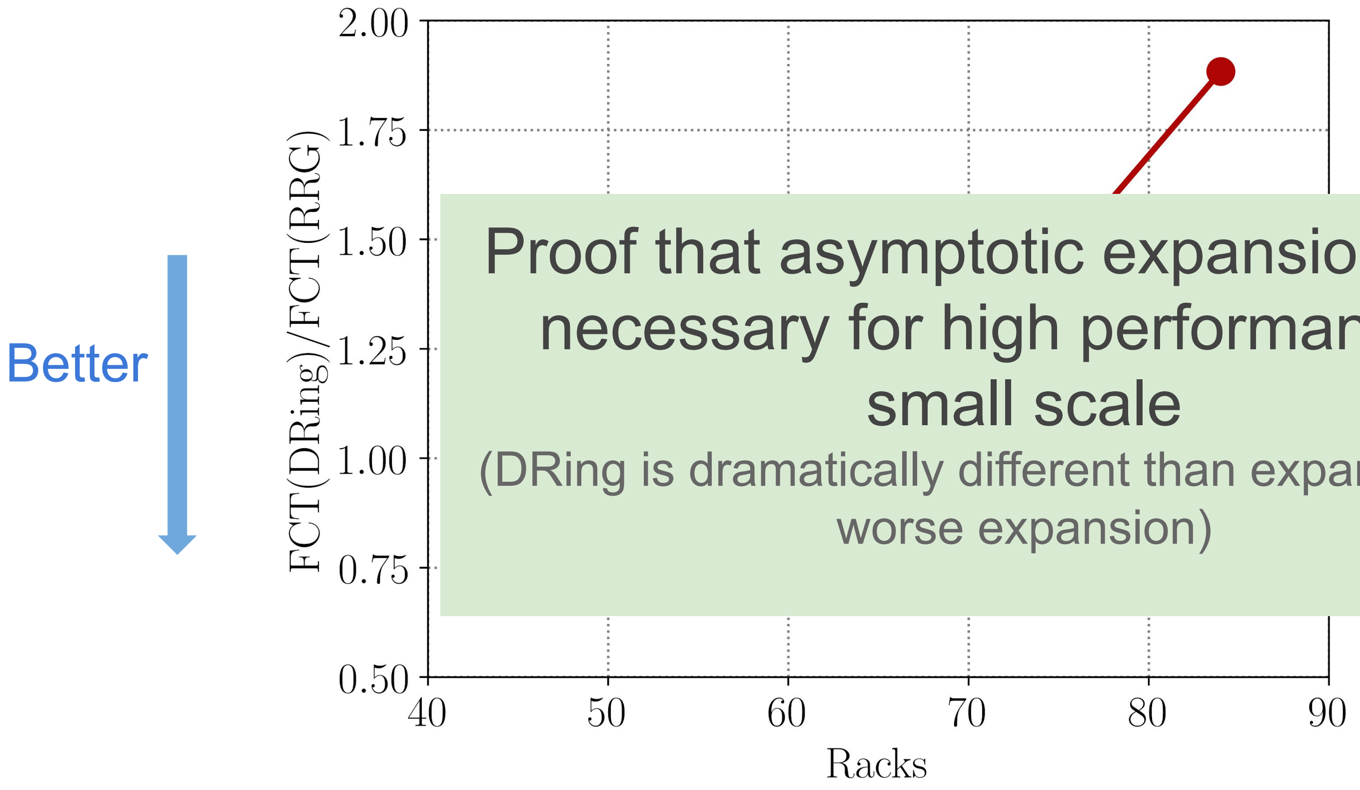


Fig: 99%ile FCT for uniform traffic

DRing: Performance deteriorates with scale



Proof that asymptotic expansion is not necessary for high performance at small scale

(DRing is dramatically different than expander, O(n) worse expansion)

Fig: 99%ile FCT for uniform traffic

Conclusion & future work

- There are more efficient topologies than Leaf-spine
  - A lot of benefit comes from using a flat network (DRing, Expanders)
- Small scale topology design is different than large scale
  - Efficient topologies exist, which aren't good at large scale
  - Can have better trade-offs for wiring/management complexity
- Practical routing for flat topologies with standard router features
  - Shortest-Union(K): Prototype implementation with BGP and VRFs
- Future work
  - Optimal topology for small scale DCs
  - Failure handling in flat networks
  - Adaptive routing/load balancing for flat topologies

Conclusion & future work

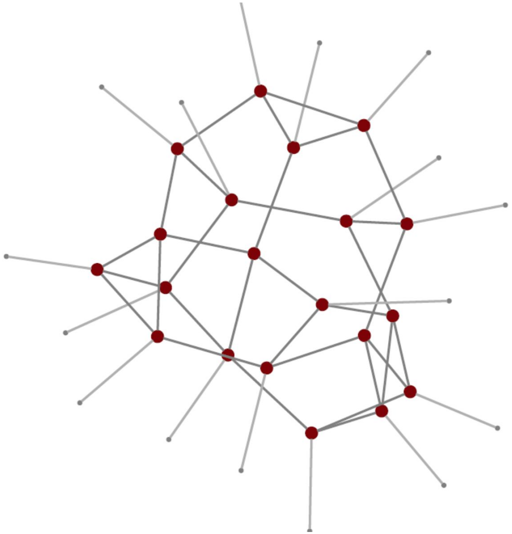
- There are more efficient topologies than Leaf-spine
  - A lot of benefit comes from using a flat network (DRing, Expanders)
- Small scale topology design is different than large scale
  - Efficient topologies exist, which aren't good at large scale
  - Can have better trade-offs for wiring/management complexity
- Practical routing for flat topologies with standard router features
  - Shortest-Union(K): Prototype implementation with BGP and VRFs
- Future work
  - Optimal topology for small scale DCs
  - Failure handling in flat networks
  - Adaptive routing/load balancing for flat topologies

Thank you!

Backup Slides

Troubleshooting in expanders

- Expanders don't have symmetrical structure
  - Unlike tree-like Clos topologies
- Asymmetry good for analysis!
  - We demonstrate it for detecting silent packet drops
  - … using Bayesian network based inference (Flock)



* Image taken from Chi-Yao's slides from Jellyfish, NSDI 2012

Flock system

- Flock: localizes problematic links
  - Using end-to-end flow metrics
    - E.g. retransmits, packets sent, RTT
  - Models problem via Bayesian network
    - No assumption about topology, routes
  - Can accommodate both active, passive information
  - Achieves higher accuracy than other schemes

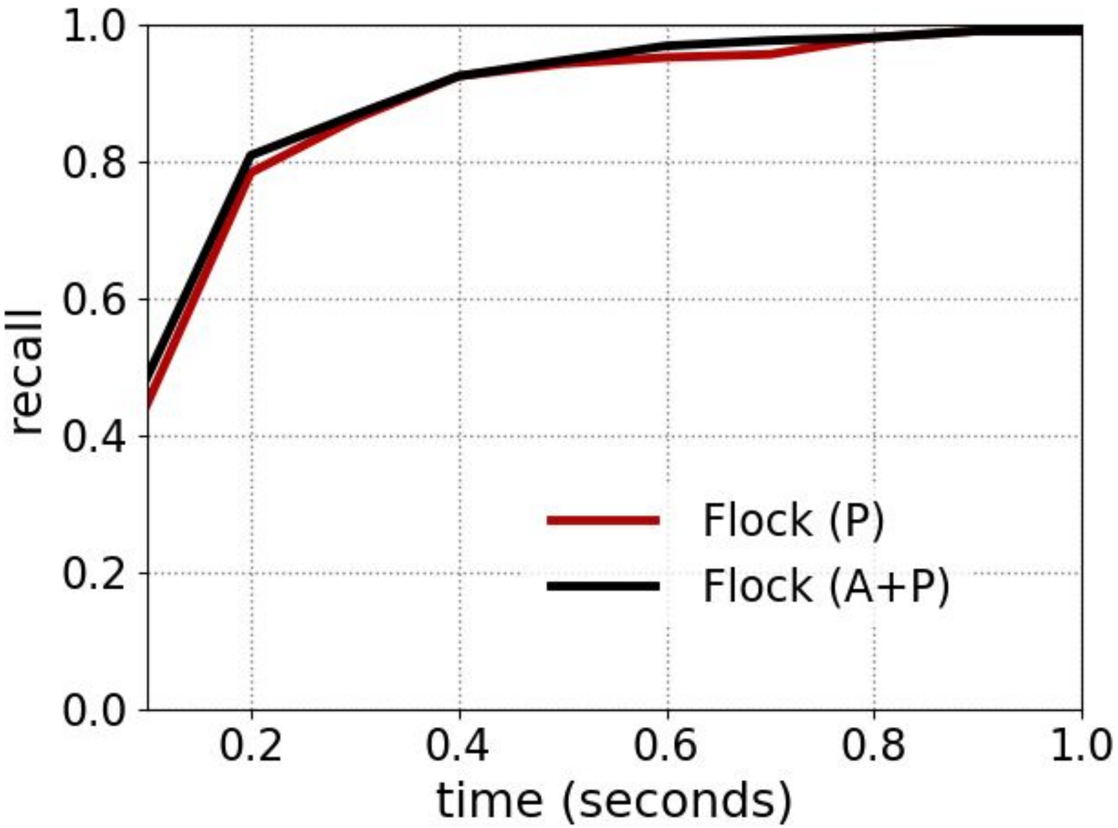
NS3 simulation setup

- Silent packet drops on links
  - 0 - 0.01% on functioning links
  - 0.2 - 2% on failed links
  - Up to 8 failed links

- Jellyfish network with 2500 links@10 Gbps
  - Running ECMP

- Input Information:
  - Active + Passive (A + P)
    - A: application flows with >0 retransmits + their paths
    - P: All other flows, path unknown
  - Passive only (P): All flows, path unknown
  - 300K flows in 1 second monitoring time

Accuracy (recall) for detecting failed links over time



Don’t need active info to localize failures in expander networks

Flock (P) doesn’t work for symmetric Clos networks