Spineless Data Centers

Vipul Harsh
University of Illinois at Urbana-Champaign

Sangeetha Abdu Jyothi
University of California, Irvine and VMware Research

P. Brighten Godfrey
University of Illinois at Urbana-Champaign and VMware

ABSTRACT

In enterprises, CDNs, and increasingly in edge computing, most data centers have moderate scale. Recent research has developed designs such as expander graphs that are highly efficient compared to large-scale, 3-tier Clos networks, but moderate-scale data centers need to be constructed with standard hardware and protocols familiar to network engineers, and are overwhelmingly built with a leaf-spine architecture.

This paper explores whether the performance efficiency that is known to be theoretically possible at large scale can be realized in a practical way for the common leaf-spine data center. First, we find that more efficient topologies indeed exist at moderate scale, showing through simulation and analysis that much of the benefit comes from choosing a "flat" network that uses one type of switch rather than having separate roles for leafs and spines; indeed, even a simple ring-based topology outperforms leaf-spine for a wide range of traffic scenarios. Second, we design and prototype an efficient routing scheme for flat networks that uses entirely standard hardware and protocols. Our work opens new research directions in topology and routing design that can have significant impact for the most common data centers.

CCS CONCEPTS

• **Networks** → **Network design principles**; **Routing protocols**.

ACM Reference Format:
Vipul Harsh, Sangeetha Abdu Jyothi, and P. Brighten Godfrey. 2020. Spineless Data Centers. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks (HotNets '20), November 4–6, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3422604.3425945

1 INTRODUCTION

Over the last decade, public clouds, manifested in hyperscale data centers (DCs), have become key infrastructure run by a handful of top cloud service providers. However, small- and medium-scale DCs, comprised of a few 10s to 100 racks and up to a few thousand servers, are critical as well, and are more numerous. Such DCs form the on-premises, privately-owned infrastructure expected to run half of all enterprise IT workloads as of 2021 [26]. Moderate-scale DCs are also the foundation of Internet exchange points, CDNs, and, increasingly, edge computing, whose market size is projected to grow from $4.5B in 2018 to $16B in 2025 [28].

HotNets '20, November 4–6, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8145-1/20/11…$15.00
https://doi.org/10.1145/3422604.3425945

A line of research has developed modern architectures for hyperscale DC networks, typically based on 3-tier Clos networks [4], and has optimized various aspects of these architectures. Of most interest here, alternative topologies based on expander graphs[1], such as Jellyfish [23] and Xpander [27], have recently been shown to yield higher performance than 3-tier Clos networks. This is partly due to smaller path length in expanders, which helps reduce congestion, an effect that is more pronounced with larger scale [13].

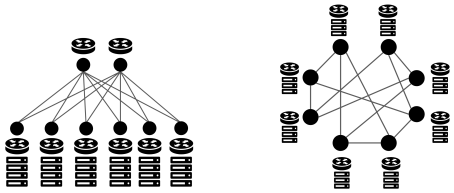
Moderate-scale DCs, however, have so far not realized these benefits. In practice, they are overwhelmingly built with 2-tier leaf-spine networks [1]. As leaf-spine networks have shorter paths than 3-tier Clos networks, it is not clear if the gains of expanders apply to leaf-spine networks. In addition, current expander designs require uncommon or novel transport, routing, or forwarding protocols, like MPTCP with k-shortest path routing [23], or an ECMP/VLB routing hybrid with dynamic switching at flowlet granularity [15]. Using such protocols is always a deployment hurdle, but would be a non-starter for enterprises that depend on standard hardware and IT support, lacking the hyperscale operators' ability to develop and support custom designs.

Our goal is to determine whether advances are possible in the under-explored area of efficiency of moderate-scale DC networks, and in particular, whether more efficient topologies are feasible. As we will show, achieving this goal comes with fundamentally different challenges and opportunities than large-scale DC networks.

We first show that topologies that outperform leaf-spine networks at moderate scale do exist. Importantly, a significant benefit comes from using a "flat" network, by which we mean that switches have only one role: all are top-of-rack (ToR) switches directly connected to servers and to other switches, and servers are distributed evenly across all the switches. To demonstrate this, we design a flat ring-like topology that we call a DRing, which is topologically unlike an expander. We show that the DRing's performance at small scale is comparable to expanders, and both DRings and expanders significantly outperform leaf-spine networks for several broad classes of traffic demand. The DRing's performance deteriorates with increasing scale, thereby showing that new design points exist for small scale DCs that would not be feasible at large scale. These design points may have new and perhaps more advantageous tradeoffs in system design (such as wiring and management complexity, which has been a road block for adoption of large-scale expander DCs [31]).

Intuitively, flatness increases the number of links exiting a rack, relative to the number of servers in the rack. Analytically, we show ToR over-subscription is 2× less in flat networks than leaf-spines built with the same hardware (§ 3.1). This does not mean flat networks always get 2× higher throughput. But when two factors are combined – (1) over-subscription causing a bottleneck exiting

[1]Expanders are an extensively studied class of graphs that are in a sense maximally or near-maximally well-connected.



(a) Leaf-spine topology (b) Flat topology

Figure 1: A flat topology can mask oversubscription. The leaf-spine (a) has 4 servers and 2 network links per rack, whereas the flat network (b), built with the same hardware, has 3 servers and 3 network links per rack. Hence, the number of network ports per server in the leaf-spine is 1/2 whereas it is 1 for the flat topology.

the leaf-spine's racks, and (2) a skewed traffic pattern causing this bottleneck at a minority of racks – flat networks are effectively able to mask the over-subscription and behave closer to a non-blocking network (see Figure 1). Oversubscription happens to be the most realistic scenario and was not explicitly explored in past work.

Second, we evaluate practical routing designs for flat DC networks through emulation in multiple classes of traffic demand. We find that simple shortest path routing with ECMP (and standard TCP transport) is sufficient for certain important traffic patterns, having up to 7× lower flow completion times than a leaf-spine for a real-world workload. However, as in larger expanders [15], there are cases where ECMP provides too few paths. In fact, this will be true of any flat topology. We therefore propose a practical and efficient routing scheme that exposes the path diversity of flat networks via simple, familiar features available in essentially all DC switches: BGP, ECMP, and virtual routing and forwarding (VRF). We demonstrate our scheme's viability by prototyping it in the GNS3 emulator [3] with Cisco 7200 switch images. To the best of our knowledge, this is the first implementation of a routing scheme on standard hardware for expanders or flat networks in general.

Overall, these results show that promising new design points are possible for small- to medium-scale DC networks. Our work suggests new research directions- searching for other new small-scale topology design point, evaluating their operational advantages, and improving practical routing designs. Via use of standard protocols, we believe this line of work can have real impact on DC deployments. Our code and routing setup is available open source [2].

2 BACKGROUND

Previous research has shown that expander graphs can yield higher performance than 3-tier Clos topologies. Singla et. al [23] first showed that expanders, embodied in the random-graph-based Jellyfish, can outperform 3-tier fat trees [4], and [22] demonstrated that they flexibly accommodate heterogeneous switch hardware and come close to optimal performance for uniform patterns under a fluid flow routing model. Asaf et al. [27] proposed the Xpander topology as a cabling-friendly alternative to Jellyfish, while matching its performance. Both Jellyfish and Xpander used k-shortest path routing and, in the transport layer, MPTCP [30] for good performance. Kassing et. al [15] demonstrated that expanders can outperform fat trees for skewed traffic patterns using a combination of VLB, ECMP, and flowlet switching [14, 25]. Jyothi et. al [13] showed that under the fluid flow model with ideal routing, the random graph outperforms the fat tree for near-worst case traffic.

However, reliance on non-traditional protocols (MPTCP, k-shortest path routing, flowlet switching, and VLB) restricts the practicality of these proposals. k-shortest path routing requires control and data plane modifications; MPTCP requires operating system support and configuration, often out of the control of the data center operator; and flowlet switching depends on flow size detection and dynamic switching of paths. Some of these mechanisms, such as flowlet switching [5], exist in some data center switches, but they are not common. Designing completely oblivious routing schemes for expanders, or flat networks, that yield high performance and can be realized with common tools available in data center switches has thus far remained an elusive goal.

All of the above proposals target replacing 3-tier Clos topologies, which are suitable for hyperscale datacenters. Most datacenters, however, are small- or medium-sized and in modern realizations are overwhelmingly based on 2-tier Clos topologies, i.e., leaf-spine networks (Figure 1a), running shortest-path routing (BGP or OSPF) with equal cost multipath (ECMP) forwarding. At this small scale, there are different factors in play. First, topologies such as Jellyfish and Xpander are known to be excellent expanders, but as [13, 23] showed, their performance gains come with scale. Small-scale realization of these networks don't necessarily respect these asymptotic characteristics, and networks that are inefficient at large scale may perform well at small scale. Second, it is especially important to stay as close as possible to the protocols that are standard for these data centers and familiar to their network engineers.

3 TOPOLOGY DESIGN

We define a flat network as one in which switches have only one role: all are top-of-rack (ToR) switches directly connected to servers and to other switches (we refer to the latter connections as *network links*). Flat networks mask rack oversubscription since they have more network links per server in a rack (see Figure 1). Note that, in a leaf-spine, the network links of a ToR carry only local traffic (originating from or destined to servers in the rack). In contrast, the network links of a ToR in a flat network carry both local traffic as well as transit traffic (originating and destined elsewhere but routed through that ToR). Nevertheless, all network links of a ToR in a flat network can carry local traffic. This is especially valuable for micro bursts where a rack has a lot of traffic to send in a short period of time and traffic is well-multiplexed at the network links (very few racks are bursting at any given point). The same argument also holds for skewed traffic matrices. In the next section, motivated by the above arguments, we introduce the notion of **Uplink to Downlink Factor** (**UDF**) of a topology, representing how much throughput gains one can expect from a flat topology compared to a baseline topology, when the network links are bottlenecked due to oversubscription.

3.1 Quantifying benefit of flatness

Consider a topology T and a flat topology $F(T)$ built with the same equipment but with servers distributed among all switches. For every ToR that contains servers, we define the Network Server Ratio (**NSR**) as the ratio of network ports to server ports (to simplify,

we assume this is the same for all ToRs with servers). We define UDF(T) as

$$UDF(T) = \frac{NSR(F(T))}{NSR(T)}.$$

Intuitively, NSR represents the outgoing network capacity per server in a rack. The UDF represents the expected performance gains with a flat network as compared to the baseline topology, when traffic is bottlenecked at ToRs. It represents the best case scenario for a flat graph when the network links of a ToR carry only traffic originated from or destined to the rack.

Define **leaf-spine (x, y)**, for arbitrary (positive integer) parameters x and y, as the following network with switch degree $(x + y)$:

- There are y spines, each connected to all leafs.
- There are $(x + y)$ leafs, each connected to all spines.
- Each leaf is connected to x servers.

We can compute the UDF for leaf-spine networks for arbitrary x and y. We have,

$$NSR(T = LeafSpine(x, y)) = \frac{y}{x}$$

For the corresponding flat network $F(T)$ built with the same equipment,

$$NSR(F(T)) = \frac{(x+y) - \text{server ports per switch}}{\text{server ports per switch}}$$
$$= \frac{(x+y) - \big(x(x+y)/(x+2y)\big)}{x(x+y)/(x+2y)} = \frac{2y}{x}$$

$$\text{Thus,} \quad UDF\big(T = LeafSpine(x, y)\big) = \frac{NSR(F(T))}{NSR(T)} = 2.$$

The UDF of a leaf-spine being 2 implies that a flat network can achieve up to 2 times the throughput of a leaf-spine when the bottleneck is at the ToRs (and hence masking the oversubscription to a large extent). We later show via experiments (§ 6.2) that in some cases, a flat network comes close to having 2x throughput as the leaf-spine.

Note that the UDF of a leaf-spine network is independent of the number of leaf and spine switches. Keeping the number of servers constant, if a network has fewer spines and more leaves, the number of servers per rack are fewer but the aggregate uplink bandwidth at the ToRs is also smaller. These two factors cancel each other and hence, the UDF remains constant.

3.2 A simple flat topology

We propose a simple flat topology, we call DRing. The key idea is to have a "supergraph" consisting of m vertices (referred to as supernodes) numbered cyclically, where vertex(i) is connected to vertex($i + 1$) and vertex($i + 2$) (Figure 2c). Each supernode consists of n ToRs and every pair of ToR switches which lie in adjacent supernodes have a direct link in the topology. All switches in the DRing topology are symmetric to each other and play the exact same role in the network. DRing is also easily incrementally expandable, by adding supernodes in the ring supergraph.

Our choice of DRing as a flat topology is to demonstrate the existence of flat topologies (other than expanders) that outperform leaf-spine networks. The DRing is intentionally dramatically different than an expander – asymptotically, its bisection bandwidth is $O(n)$ worse! Finding the best flat topology at small scale, across multiple design axes (manageability, performance, and complexity) remains an open question.

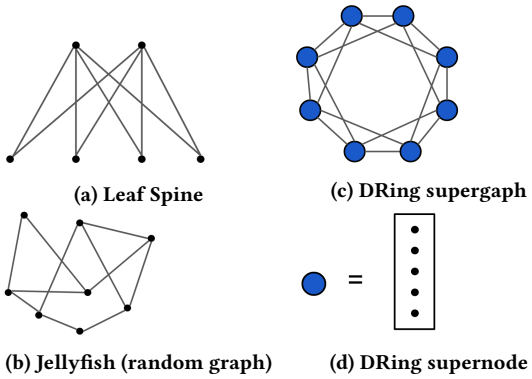


Figure 2: Topologies: each vertex in (a) and (b) represents a router. Each node (referred to as a supernode) in supergraph (c) consists of multiple ToRs (shown in (d)). Any pair of ToRs which lie in neighboring supernodes are directly connected in the topology. (b) and (c) are flat topologies.

4 ROUTING DESIGN

We consider the following two schemes, both of which are implementable in today's common DC hardware.

ECMP: Standard shortest path routing, commonly available on datacenter switches.

Shortest-Union(K): Between two ToR switches R_1 and R_2, use all paths that satisfy either of the following conditions:

- the path is a shortest path between R_1 and R_2
- the length of the path is less than or equal to K

Shortest paths do not suffice for taking full advantage of path diversity in a flat network, as previous works have shown [15, 23] for expander graphs. This is true of all flat networks because there is only one shortest path between two racks that happen to be directly connected; hence, shortest paths cannot exploit the path diversity for adjacent racks (unlike leaf-spine networks, where racks are never directly connected). In general, the closer two racks are to each other, the fewer shortest paths are between them.

Shortest-Union(K) routing employs non-shortest paths for pairs of racks that are close and hence, don't have enough shortest paths between them. Two racks which are distant to each other have sufficiently many shortest paths available between them to effectively load balance traffic and hence no extra paths are required. We use $K = 2$ in our experiments since it offers a good tradeoff between path diversity and path length. It offers more paths than ECMP but also uses paths that are not too much longer than shortest paths (important for high performance for uniform traffic). For DRing, Shortest-Union(2) provides at least $(n + 1)$ disjoint paths between any two racks (n = number of racks in one supernode).

Next, we show how to implement the Shortest-Union(K) scheme with BGP and VRFs, as these are available in essentially all datacenter switches. We've prototyped Shortest-Union(2) scheme in the GNS3 network emulator [3] on emulated Cisco 7200 routers. VRFs gives us the power to virtualize a switch and partition the switch interfaces across the VRFs. We partition each router into K VRFs-

(VRF 1, VRF 2 ... VRF K). The host interfaces are assigned to VRF K. We use a unique AS number for each router and all VRFs on one router have that same AS number. For Shortest-Union(K), K VRFs need to be configured at each router.

For every directed physical connection from switch R1 to R2 in the topology (treating an undirected link as two directed links in opposite directions), we create the following virtual connections in the VRF graph:

(1) (VRF K, R1) $\rightarrow$ (VRF i, R2) of cost i, for all i
(2) (VRF (i-1), R1)[2] $\rightarrow$ (VRF i, R2) of cost 1
(3) (VRF 1, R1) $\rightarrow$ (VRF 1, R2) of cost 1

Note that the cost can be different in the two directions for the same link. The cost of any other link not listed above is ∞. The costs can be set via path prepending in BGP. This design is also illustrated in Figure 3. We simply use shortest path routing in this VRF graph, which can be done via BGP (specifically eBGP). There are no loops in the routing paths, at the router level, since BGP does not admit any path that contains multiple nodes belonging to the same AS.

Theorem 1. *For two routers in the topology R1 and R2 separated by distance L, the shortest path in VRF graph between (VRF K, R1) to (VRF K, R2) has length = max(L, K).*

Proof. Let's say the shortest path between R1 and R2 in the topology is (R1, $A_1, A_2 \ldots A_{L-1}$, R2).

Case 1: $L \geq K$: consider the path ((VRF K, R1), (VRF 1, A_1), (VRF 1, A_2) ... (VRF 1, A_{L-k+1}), (VRF 2, A_{L-K+2}) ... (VRF K-1, A_{L-1}) (VRF K, R2)), which has cost L. Since all links have cost ≥ 1, any other shortest path between (VRF K, R1) and (VRF K, R2), which will have at least L hops, will also have cost at least L. Hence, the shortest path length in the VRF graph is L.

Case 2: $L < K$: The path ((VRF K, R1), (VRF $L-1$, A_1), (VRF $L-2$, A_2) ... (VRF K-1, A_{L-1}), (VRF K, R2)) has cost K (the first link has cost $K - L$, all other links have cost 1). Hence, the distance between (VRF K, R1) and (VRF K, R2) is at most K. Next we show that any other path between (VRF K, R1) and (VRF K, R2) length at least K. If the second hop in the path belongs to VRF i of an adjacent node of R1, then the length of the path is $\geq i + (K - i) = K$ since at least $K - i$ hops are needed to reach VRF K of any node from VRF i of any node. □

In order to reach a destination host h2 in rack R2 from a source host h1 in rack R1, a flow needs to reach (VRF k, R2) from (VRF k, R1). If the shortest path between R1 and R2 is < k, then this design ensures that all paths of length $\leq K$ in the physical topology can be used since they all have cost K in the VRF graph.

We note that a simple change to BGP's path selection process would simplify the above routing design, removing the need of configuring VRFs. Currently, in popular implementations, BGP does not support multipath route selection with different AS lengths. This can be done easily by allowing the two commands "bgp ignore-as-path" and "bgp maximum-paths" to be configured simultaneously, which is currently disallowed in common vendor implementations. We also note that the routing configurations at each router can be generated by a simple script to avoid errors.

[2]An earlier version incorrectly mentioned (VRF (i+1), R1)

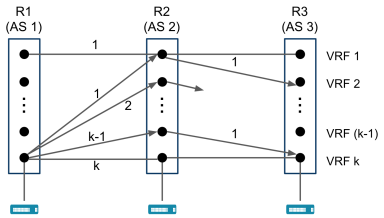


Figure 3: Shortest path routing in the VRF graph. Links are annotated with their costs. Links with arrows have different weights for forward and reverse links. Each router constitutes one AS for BGP. Not all connections are shown.

5 EXPERIMENTAL SETUP

5.1 Topologies

Leaf-spine(x,y): As per recommended industry practices [1], we choose an oversubscription ratio = $x/y = 3$ with $x = 48, y = 16$ (see § 3.1 for definition), matching an actual industry-recommended configuration [1], leading to 64 racks and 3072 servers. We chose this recommended configuration in part because it uses leaf and spine switches with the same line speed, making comparisons more straightforward; we leave heterogeneous configurations to future work, but expect similar results.

Expander graph: We use a regular random graph (RRG) [23] as it's a high-end expander [27]. Other expanders have similar performance characteristics to the random graph [27] and hence we expect our results to apply to all high-end expanders. We build a random graph with the exact same equipment as the leaf-spine, by rewiring the baseline leaf-spine topology, redistributing servers equally across all switches (including switches that previously served as spines) and applying a random graph to the remaining ports.

DRing supergraph: We use a DRing supergraph with 12 supernodes (see § 3.2), that consists of 80 racks and 2988 servers overall, which was closest to the leaf-spine config we picked and has about 2.8% fewer servers.

5.2 Traffic workload

Uniform/A2A: Standard uniform traffic where each flow is assigned a random source and destination, similar to sampled All-to-all(A2A).

Rack-to-rack (R2R): All servers in one rack send to all servers in another rack.

Real world TMs: We use two real-world traffic workloads (FB skewed and FB uniform) from two 64 rack-clusters at Facebook presented in [21], one from a Hadoop cluster comprising of largely uniform traffic and one from a front-end cluster with significant skew. The raw data on the traffic weights between any two racks were obtained as in [13]. Flows are chosen between a pair of racks in the leaf-spine network as per the rack-level weights obtained from the Facebook data, yielding a server-level traffic matrix (TM).

FB skewed/uniform Random Placement (RP): We take the server level TM as described in the previous entry and randomly shuffle the servers across the datacenter. This helps in evaluating topologies with random placement of VMs which has been shown to be beneficial [13].

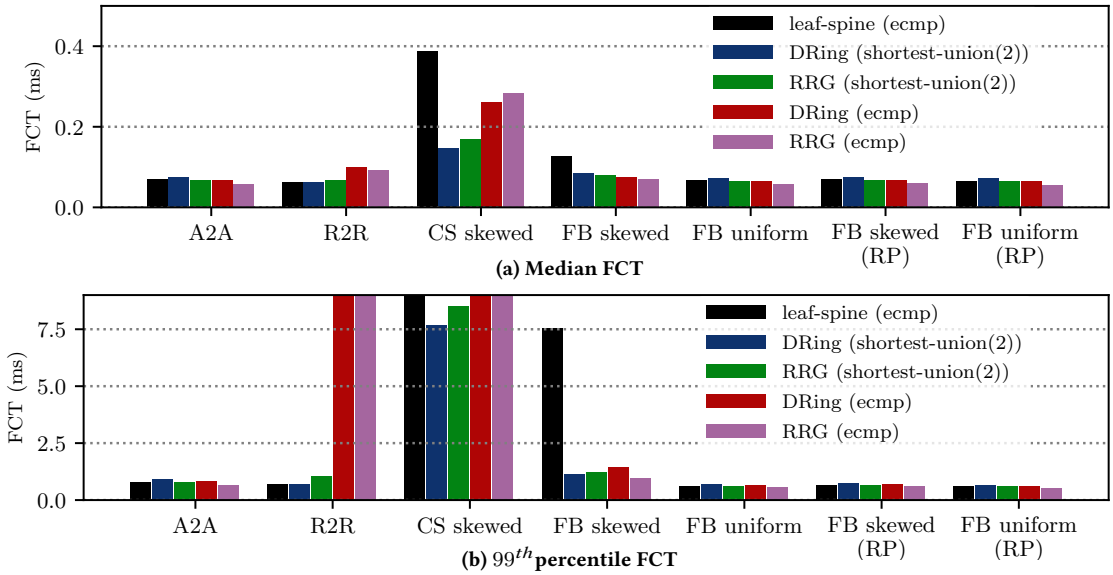


Figure 4: Flow completion times for various traffic matrices: both flat topologies namely DRing and RRG show significant improvement over leafspine for skewed traffic and have comparable performance for uniform traffic matrices. The routing for each scheme is shown in paranthesis. In (b), bars touching the top are all > 30 ms. C-S skewed traffic represents C=n/4, S=n/16 in the C-S model where n represents the total number of hosts in the DC.

Flow size distribution: Flow sizes are picked from a standard Pareto distribution with mean 100KB and scale=1.05 to mimic irregular flow sizes in a typical datacenter [6]. The number of flows are determined according to the weights of the TM and flow start times are chosen uniformly at random across the simulation window.

C-S model: To capture a wide range of scenarios, we pick a subset C of hosts to act as clients and pack these clients into the fewest number of racks while randomly choosing the racks in the DC. Similarly, we pick a subset S of hosts to act as servers and pack them into the fewest number of racks possible (avoiding racks used for C). We wish to measure the network capacity between the sets C and S for all possible sizes of C and S. By varying the sizes of C and S, this model, which we call the C-S model, captures a wide range of patterns that commonly occur in applications, including: (i) incast/outcast for $C = 1/S = 1$, (ii) rack to rack, (iii) a range of skewed traffic for $|C| << |S|$, and (iv) uniform traffic for $|C| = |S| = n/2$, where n is the total number of hosts.

5.3 Simulation setup

We use the htsim-based packet level simulator used in [18], configured with TCP and 10Gbps links. We evaluate ECMP and Shortest-Union(2) routing for the flat topologies and ECMP for leaf-spine.

6 EXPERIMENTAL EVALUATION

6.1 Flow completion times

We compare leaf-spine, DRing and RRG for TMs described in § 5.2. We scale the TMs so that the network utlization in the spine layer is 30%. As only a small subset of the racks participate in the rack-to-rack and C-S traffic patterns, we further scale these TMs down by a factor = number of racks that send traffic / total racks.

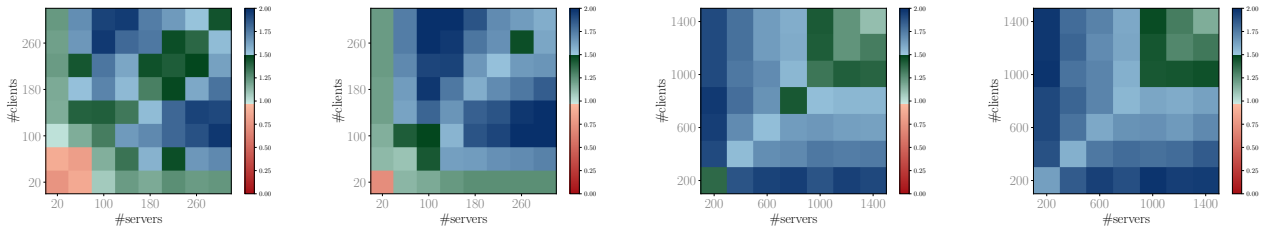
Figure 4 shows the median and 99th percentile FCT. Flat networks show a decent improvement with ECMP for uniform and skewed TMs. However, performance of flat networks with ECMP for rack-to-rack is poor. We see that the Shortest-Union(2) scheme is effective at resolving this problem and also yields better performance than with ECMP for skewed traffic patterns due to the higher path diversity. However, since it uses longer paths than ECMP, in the case of uniform traffic, performance is slightly worse but still comparable to ECMP. Overall, both DRing and RRG with Shortest-Union(2) routing outperform leaf-spine networks for skewed traffic and are comparable for other TMs.

6.2 Throughput in the C-S model

Figure 5 shows the average throughput for a wide range of C-S model TMs. To measure throughput, all flows were long-running (similar to the setup in [23]). DRing with ECMP outperforms leaf-spine significantly for a wide range of scenarios, but is poor at the lower left of Figure 5(a). DRing with Shortest-Union(2) routing fixes that problem and also improves gains across other points. Observe that for skewed TMs ($|C| \ll |S|$ or vice-versa), DRing's throughput almost matches the 2× predicted gains, as showed in § 3.1 for flat networks.

6.3 Effect of scale

As shown in Figure 4, DRing outperforms leaf-spine and even random graphs for multiple classes of traffic matrices at small scale. Figure 6 shows that this is however true only at small scale as DRing's performance deteriorates with increasing scale. Along the x-axis in Figure 6, we add supernodes to obtain a larger topology. Theoretically, the poor performance of DRing at larger scale is



(a) small values, ECMP (b) small values, shortest-union(2) (c) large values, ECMP (d) large values, shortest-union(2)

Figure 5: DRing vs Leaf Spine in the C-S model: Average throughput for small and large values of C, S in the C-S model. Each entry in the heatmap is the ratio: throughput(DRing)/throughput(leaf-spine) for that particular C-S traffic matrix (C clients sending to S servers). The DRing topology used in these experiments had 2988 servers and the leaf-spine had 3072 servers.

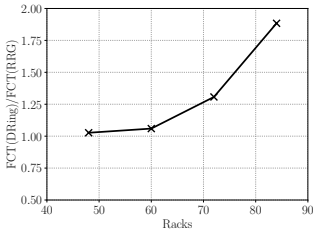


Figure 6: Effect of scale: 99%ile FCT of DRing deteriorates at large scale in comparison to equivalent RRG for uniform traffic. For DRing, we used 6 switches per supernode with 60 ports per switch, 36 of which were server links. Along the x-axis, we add supernodes to obtain a larger topology.

expected since its bisection bandwidth is $O(n)$ worse than the expander. However, this effect only shows up at large scale where the constants don't matter.

6.4 Key takeaways

- Efficient topologies do exist at small scales. Both RRG and DRing provide significant improvements over leaf-spine for many scenarios (Figure 4 and 5).
- There are flat networks (beyond expander graphs) such as DRing that are worth considering for small- and moderate-scale DCs. These topologies might not be good at large scale but can be efficient for small scale.
- The Shortest-Union(2) routing scheme fixes the problems in using flat networks with ECMP (see 4). Further, Shortest-Union(2) is completely oblivious and can be implemented with basic hardware tools.

7 DISCUSSION AND FUTURE WORK

Better small topologies: Performance gains of DRing at small scale suggests that there are high performance networks beyond Clos and expanders. Finding the best topology at small scale along several axes (performance, ease of manageability and wiring, incremental expandability, simple hardware) remains an open question. Recent work has made efforts at topology design along these axes for large scale [31] DCs. But as our work shows, small scale offers new design points that are not feasible at large scale.

Coarse-grained adaptive routing: As shown in Figure 4, flat networks with ECMP perform poorly for rack-to-rack traffic (because of lack of path diversity between adjacent racks) but perform very well for uniform traffic (because of using shorter paths). The Shortest-Union(2) routing scheme is a good-tradeoff between more paths and shorter paths. However, it is not consistently better than ECMP across all traffic patterns. This suggests that an adaptive routing strategy (e.g. [15, 18]), even at coarse-grained scales based on DC utilization, can provide a further performance improvement using flat networks.

Impact of failures: How quickly can routing converge to alternative paths in the presence of failures in a flat network? What is the impact of failures on network paths and load balancing? We leave these questions for future work.

Dynamic Networks based on flat topologies: Several works have proposed dynamic networks [8, 10, 12, 18–20, 24, 29, 32], where link connections are configured dynamically based on traffic load. However, the overhead of reconfiguring links poses a problem, especially for short flows. Opera [18] attempts to fix that by adapting a hybrid strategy and imposing transient expander graphs with the dynamic links while long flows wait for a direct link. Short flows use whatever paths are available immediately since latency is critical for short flows. Since DRing (and possibly other flat) networks can outperform expanders at smaller scales, it is important to find how much improvement can be gained by reconfiguring links to obtain another flat network instead of an expander.

Other static networks: Flat networks like Slim Fly [7] and Dragonfly [16] which are essentially low-diameter graphs have been shown to have high performance. We expect them to also have high performance at small scales but practicality might be limited since they require non-oblivious routing techniques. We believe that other tree-based designs such as LEGUP [9], F10 [17] and BCube [11] will have similar problems as leaf-spine because of their non-flatness.

8 CONCLUSION

We showed that flat topologies, namely Jellyfish and a new topology we presented called DRing, outperform the standard leaf-spine networks at small scale. Our analysis showed that although DRing's performance is poor at large scale, it outperforms state-of-the-art leaf-spine and even random graphs for multiple TMs at small scale, thus suggesting that small-scale topology design poses different challenges than hyperscale. Finally, we presented an oblivious high performance routing scheme for flat networks that can be implemented with basic tools in current data center switches.

REFERENCES

[1] Cloud networking scale out - arista. https://www.arista.com/assets/data/pdf/Whitepapers/Cloud_Networking__Scaling_Out_Data_Center_Networks.pdf.

[2] Github. https://github.com/netarch/expanders-made-practical.

[3] Gns3. https://gns3.com/.

[4] Al-Fares, M., Loukissas, A., and Vahdat, A. A scalable, commodity data center network architecture. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication* (New York, NY, USA, 2008), SIGCOMM '08, ACM, pp. 63–74.

[5] Alizadeh, M., Edsall, T., Dharmapurikar, S., Vaidyanathan, R., Chu, K., Fingerhut, A., Lam, V. T., Matus, F., Pan, R., Yadav, N., and Varghese, G. Conga: Distributed congestion-aware load balancing for datacenters. *SIGCOMM Comput. Commun. Rev. 44*, 4 (Aug. 2014), 503–514.

[6] Alizadeh, M., Kabbani, A., Edsall, T., Prabhakar, B., Vahdat, A., and Yasuda, M. Less is more: Trading a little bandwidth for ultra-low latency in the data center. In *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)* (San Jose, CA, 2012), USENIX, pp. 253–266.

[7] Besta, M., and Hoefler, T. Slim fly: A cost effective low-diameter network topology. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Piscataway, NJ, USA, 2014), SC '14, IEEE Press, pp. 348–359.

[8] Chen, L., Chen, K., Zhu, Z., Yu, M., Porter, G., Qiao, C., and Zhong, S. Enabling wide-spread communications on optical fabric with megaswitch. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)* (Boston, MA, 2017), USENIX Association, pp. 577–593.

[9] Curtis, A. R., Keshav, S., and Lopez-Ortiz, A. Legup: Using heterogeneity to reduce the cost of data center network upgrades. In *Proceedings of the 6th International COnference* (New York, NY, USA, 2010), Co-NEXT '10, ACM, pp. 14:1–14:12.

[10] Farrington, N., Porter, G., Radhakrishnan, S., Bazzaz, H. H., Subramanya, V., Fainman, Y., Papen, G., and Vahdat, A. Helios: A hybrid electrical/optical switch architecture for modular data centers. In *Proceedings of the ACM SIGCOMM 2010 Conference* (New York, NY, USA, 2010), SIGCOMM '10, ACM, pp. 339–350.

[11] Guo, C., Lu, G., Li, D., Wu, H., Zhang, X., Shi, Y., Tian, C., Zhang, Y., and Lu, S. Bcube: A high performance, server-centric network architecture for modular data centers. In *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication* (New York, NY, USA, 2009), SIGCOMM '09, ACM, pp. 63–74.

[12] Hamedazimi, N., Qazi, Z., Gupta, H., Sekar, V., Das, S. R., Longtin, J. P., Shah, H., and Tanwer, A. Firefly: A reconfigurable wireless data center fabric using free-space optics. In *Proceedings of the 2014 ACM Conference on SIGCOMM* (New York, NY, USA, 2014), SIGCOMM '14, ACM, pp. 319–330.

[13] Jyothi, S. A., Singla, A., Godfrey, P. B., and Kolla, A. Measuring and understanding throughput of network topologies. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Piscataway, NJ, USA, 2016), SC '16, IEEE Press, pp. 65:1–65:12.

[14] Kandula, S., Katabi, D., Sinha, S., and Berger, A. Dynamic load balancing without packet reordering. *SIGCOMM Comput. Commun. Rev. 37*, 2 (Mar. 2007), 51–62.

[15] Kassing, S., Valadarsky, A., Shahaf, G., Schapira, M., and Singla, A. Beyond fat-trees without antennae, mirrors, and disco-balls. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication* (New York, NY, USA, 2017), SIGCOMM '17, ACM, pp. 281–294.

[16] Kim, J., Dally, W. J., Scott, S., and Abts, D. Technology-driven, highly-scalable dragonfly topology. In *Proceedings of the 35th International Symposium on Computer Architecture* (Washington, DC USA, 2008), pp. 77–88.

[17] Liu, V., Halperin, D., Krishnamurthy, A., and Anderson, T. F10: A fault-tolerant engineered network. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation* (Berkeley, CA, USA, 2013), nsdi'13, USENIX Association, pp. 399–412.

[18] Mellette, W. M., Das, R., Guo, Y., McGuinness, R., Snoeren, A. C., and Porter, G. Expanding across time to deliver bandwidth efficiency and low latency. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)* (Santa Clara, CA, Feb. 2020), USENIX Association, pp. 1–18.

[19] Mellette, W. M., McGuinness, R., Roy, A., Forencich, A., Papen, G., Snoeren, A. C., and Porter, G. Rotornet: A scalable, low-complexity, optical datacenter network. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication* (New York, NY, USA, 2017), SIGCOMM '17, Association for Computing Machinery, p. 267–280.

[20] Porter, G., Strong, R., Farrington, N., Forencich, A., Chen-Sun, P., Rosing, T., Fainman, Y., Papen, G., and Vahdat, A. Integrating microsecond circuit switching into the data center. *SIGCOMM Comput. Commun. Rev. 43*, 4 (Aug. 2013), 447–458.

[21] Roy, A., Zeng, H., Bagga, J., Porter, G., and Snoeren, A. C. Inside the social network's (datacenter) network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (New York, NY, USA, 2015), SIGCOMM '15, ACM, pp. 123–137.

[22] Singla, A., Godfrey, P. B., and Kolla, A. High throughput data center topology design. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)* (Seattle, WA, 2014), USENIX Association, pp. 29–41.

[23] Singla, A., Hong, C.-Y., Popa, L., and Godfrey, P. B. Jellyfish: Networking data centers randomly. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation* (Berkeley, CA, USA, 2012), NSDI'12, USENIX Association, pp. 17–17.

[24] Singla, A., Singh, A., and Chen, Y. OSA: An optical switching architecture for data center networks with unprecedented flexibility. In *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)* (San Jose, CA, 2012), USENIX, pp. 239–252.

[25] Sinha, S., Kandula, S., and Katabi, D. Harnessing TCPs Burstiness using Flowlet Switching. In *3rd ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)* (San Diego, CA, November 2004).

[26] Uptime Institute. 2019 data center industry survey results. https://uptimeinstitute.com/2019-data-center-industry-survey-results, 2019.

[27] Valadarsky, A., Shahaf, G., Dinitz, M., and Schapira, M. Xpander: Towards optimal-performance datacenters. In *Proceedings of the 12th International on Conference on Emerging Networking EXperiments and Technologies* (New York, NY, USA, 2016), CoNEXT '16, ACM, pp. 205–219.

[28] Wadhwani, P., and Gankar, S. Edge data center market report. https://www.gminsights.com/industry-analysis/edge-data-center-market, October 2019.

[29] Wang, G., Andersen, D. G., Kaminsky, M., Papagiannaki, K., Ng, T. E., Kozuch, M., and Ryan, M. c-through: part-time optics in data centers. *SIGCOMM Comput. Commun. Rev. 41*, 4 (Aug. 2010), –.

[30] Wischik, D., Raiciu, C., Greenhalgh, A., and Handley, M. Design, implementation and evaluation of congestion control for multipath tcp. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation* (Berkeley, CA, USA, 2011), NSDI'11, USENIX Association, pp. 99–112.

[31] Zhang, M., Mysore, R. N., Supittayapornpong, S., and Govindan, R. Understanding lifecycle management complexity of datacenter topologies. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)* (Boston, MA, Feb. 2019), USENIX Association, pp. 235–254.

[32] Zhou, X., Zhang, Z., Zhu, Y., Li, Y., Kumar, S., Vahdat, A., Zhao, B. Y., and Zheng, H. Mirror mirror on the ceiling: Flexible wireless links for data centers. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication* (New York, NY, USA, 2012), SIGCOMM '12, ACM, pp. 443–454.