Fat-Free Topologies

... without antennae, mirrors, and disco-balls?

Ankit Singla (Department of Computer Science, ETH Zürich)

ABSTRACT

With the growing size of data center networks, full-bandwidth connectivity between all pairs of servers is becoming difficult and expensive to scale. Thus, numerous recent topology proposals incorporate reconfigurable wireless and optical connectivity, allowing the topology to adapt to the traffic demands — only servers that require bandwidth at any given time receive such dynamic connections. Implicitly, this work has suggested that statically wired topologies are fundamentally inflexible, and would need to be built at full capacity to handle unpredictably skewed traffic.

This work shows the reports of inflexibility of statically wired networks to be greatly exaggerated — if traffic engineering were efficient, certain statically wired networks could achieve performance and cost comparable to topology-adaptive designs, even for skewed workloads. Thus, alongside the development of reconfigurable topologies, the community should also invest in developing superior traffic engineering over static networks other than fat-trees as an alternate path forward. These results also call for a rigorous quantification of the difference between the power of two techniques for handling dynamic, unpredictable traffic with limited network resources: traffic engineering over suitable static networks, and changing the topology itself dynamically.

1 Introduction

Virtually every popular Web service today is backed by data center infrastructure. With the growth of these services, the supporting infrastructure has also acquired massive scale, with data centers being designed with as many as 100,000 servers [8]. For such large facilities, engineering full-bandwidth connectivity between all pairs of servers entails significant effort and expense. Further, for many facilities, at any given time, only a fraction of servers may require high-bandwidth connectivity [6, 10], making such a design seem wasteful. However, traditional topologies like the fat-tree [2] present network designers with only two hard choices: (a) either build a rearrangeably non-blocking network at a large expense; or (b) build a cheaper, oversubscribed network, which does not provide high-bandwidth connectivity even to small (albeit arbitrary) subsets of the server pool, thereby needing aggressive workload management.

Over the past few years, numerous data center network designs have tackled this problem, based on the general idea of adapting the topology itself to the traffic demands [4–7, 10, 11, 15, 16, 21, 23]. The key insight in this literature is that if only a fraction of servers demands high bandwidth at any given time, reconfigurable wireless or optical elements can dynamically set up connectivity to meet these demands. For certain workloads, such a network's performance can be similar to a much more expensive interconnect that provides full bandwidth between all pairs of servers at all times. Each proposal follows broadly similar contours: estimate the traffic demands, calculate a network topology and routes fitting these demands, configure the wireless or optical elements according to the topology, and configure end-hosts or top-of-rack switches to use the routes made available. As the traffic demand changes, this process is continually repeated. The ingenuity of different proposals lies in the varied capabilities of the reconfigurable elements and how they are connected together (including movable wireless antennae [10], ceiling mirrors [11, 23], and disco-balls [6]), and the algorithms for computing the dynamic topology and the routes.

This general approach has obvious, intuitive appeal, and indeed, some of the evaluations show performance similar to a full-bandwidth fat-tree at 25-40% lower cost for certain workloads [6, 11]. Not only are such results impressive, it is also unclear whether there any other viable options beyond full-bandwidth topologies and such topology adaptation. Before fleshing out their reconfigurable optics-based architecture, Helios [5], its authors summarized the (2010) state of data center network topology design as follows:

> *"Unfortunately, given current data center network architectures, the only way to provision required bandwidth*

HotNets-XV, November 09 - 10, 2016, Atlanta, GA, USA

 ISBN 978-1-4503-4661-0/16/11…$15.00

DOI: http://dx.doi.org/10.1145/3005745.3005747

between dynamically changing sets of nodes is to build a non-blocking switch fabric at the scale of an entire data center, with potentially hundreds of thousands of ports."

More recently, the 3D-beamforming proposal [23] (2012) made a similar assessment, and Firefly [11] (2014) explicitly considered only two design possibilities: a full bisection-bandwidth network, and topology adaptation. This literature can leave one with a grim outlook on static networks, prodding the conclusion that topology adaptation is the only alternative to expensive full-bandwidth fabrics like fat-trees.

This work contests this presumption by showing that the inflexibility of static topologies is vastly over-rated. While the claimed limitation is indeed fundamental when considering fat-trees and similar designs, it is not fundamental of statically-wired networks in general: *if* traffic engineering (TE) were efficient, known statically-wired topologies would achieve results comparable to a restricted model of topology adaptation. Given that at least 8 proposals on topology adaptation, starting with the premise of statically-wired networks being inflexible, have appeared in recent iterations of SIGCOMM alone, this result should be surprising: while topology adaptation clearly offers greater flexibility, "just" TE could realize a lot of its potential gains.

Of course, developing an efficient and responsive traffic engineering mechanism that works at massive scale is a big challenge. But TE is surely a bread-and-butter problem for networking, and topology adaptation also depends, albeit to a lesser extent, on TE. Further, while topology adaptation is an exciting direction, as discussed in §4, it also introduces new and significant challenges.

Thus, the results presented here suggest that we should: (a) actively explore the new-old approach of using traffic engineering over statically-wired topologies other than fat-trees; (b) resolve fundamental questions about the relative power of traffic engineering versus topology adaptation; and (c) re-examine the design space for data center networks, together with the benchmarks used to evaluate it.

2 Network flexibility

This section examines the instructive example of the inflexibility of oversubscribed fat-trees towards skewed traffic matrices (§2.1), introduces a quantitative notion of the desired flexibility (§2.2), and discusses the potential of topology adaptation to achieve it (§2.3).

2.1 Fat-trees are inflexible

Oversubscribed fat-trees are fundamentally limited in their ability to support skewed traffic matrices. In the example shown in Fig. 1, the fat-tree is oversubscribed by removing one root switch. If each server in one pod is communicating with a (different) server in another pod, not all such connections can get full bandwidth. In this example, the network still has more than 75% of its edges, and yet cannot provide full bandwidth connectivity for a traffic matrix involving only 50% of the servers. In general, if any layer in the topology is oversubscribed to $x\%$ of the full capacity (by either removing edges uniformly across all switches, or removing certain switches), then any pair of pods is limited to $x\%$ throughput[1]. Note that for a fat-tree built with k-port devices, a pair of pods comprises only $2/k$ of the network's servers. For $k = 64$, for instance, this would be a mere 3% of servers — if such a fat-tree ($k = 64$) were built at 50% capacity, a pair of pods comprising only 3% of the network's servers could still not achieve full throughput, even while the rest of the network idles. Only if 50% or less of the servers in *each* pod were involved, could such an oversubscribed fat-tree achieve full throughput for these servers. This implies that pod-pod traffic must be carefully managed. Other Clos-network-based designs suffer similar problems.

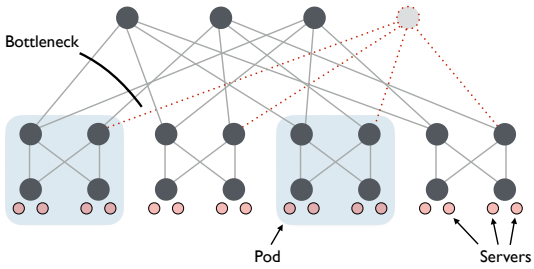


Figure 1: *A $k = 4$ fat-tree. With more than 75% of the network's capacity intact, if all 4 servers in one pod sent traffic to servers in another pod (thus involving 50% of all servers), each would get only 75% of the bandwidth.*

Thus, as literature on topology adaptation rightly points out, oversubscribed fat-trees score low on any metric of network flexibility. But what precisely *is* the metric? So far, there has not been a clearly specified and easily replicable standard for evaluating the flexibility of a topology. In the following, I offer a simple starting point.

2.2 Throughput proportionality

We would like to build oversubscribed networks that can move around their limited capacity to meet traffic demands in a targeted fashion. A network's total capacity is fixed, and defined by the total link capacity of its edges, and for each server, the network expends this capacity on routing its flows. We may hope that as the number of servers participating in the traffic matrix (TM) decreases, we see a proportional increase in the throughput. As the fat-tree example illustrates, network bottlenecks may prevent such proportional increases. But is such proportionality unattainable for other statically wired networks as well? For a generic statically wired network, it is easy to show that for TMs like all-to-all (whereby each participating server communicates with each other participating server) and permutations (whereby participating servers are matched into communicating pairs), per-server-throughput *cannot* improve more than proportionally. This latter observation makes a proportional increase in throughput a useful benchmark for network flexibility – how close can a

[1]If it were known *a priori* which pods might need higher bandwidth, different pods may be oversubscribed differently, but this would be a strong assumption on the predictability and stability of traffic.

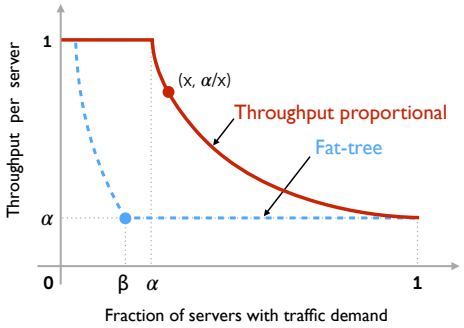


Figure 2: *A throughput-proportional network would be able to distribute its capacity evenly across only the set of servers with traffic demands.*

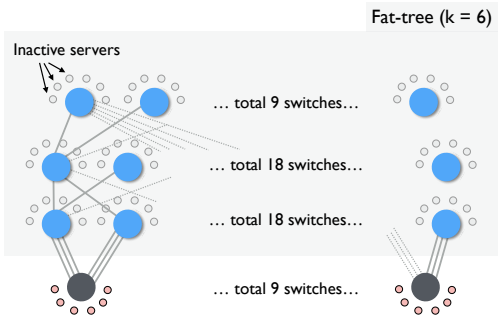


Figure 3: *This topology provides full throughput to all active servers, where any scheme prioritizing direct links between communicating ToRs could not.*

network get to this characteristic?

A network shall be called *throughput-proportional* (TP) **if** when built such that it achieves throughput α (as a fraction of line rate) per server under the hose traffic model[2], then for *any* traffic matrix involving *any* fraction $\leq \alpha$ of servers, it achieves full throughput (subject to constraints at the servers themselves) for each of these servers. Further, for traffic matrices involving an increasing fraction of servers $x \in (\alpha, 1]$, throughput should decrease proportionally[3]. This definition is deliberately simple, but one can easily map a fuzzier classification of servers as "hot" or not, to the strict binary of "involved in the TM" or not.

This notion of a throughput-proportional network is illustrated in Fig. 2, which also contrasts it with the fat-tree's behavior. As discussed earlier in §2.1, for the fat-tree, if a fraction greater than $\beta = 2/k$ of servers are involved, the simple pod-to-pod TM results in throughput per server limited to α. As the fraction of servers involved drops below β, throughput per server increases proportionally, hitting 1 only when α fraction of the pod itself, *i.e.,* $2\alpha/k$ of all servers are involved. The fat-tree's low flexibility to skewed traffic has set off research into topology adaptation.

2.3 Topology adaptation

The general idea behind topology adaptation was discussed in §1. In a generic topology-adaptation proposal, each top-of-rack (ToR) switch may have a certain number of flexible ports, say k, which can be connected to available flexible ports on other ToRs. Clearly, any topology that can be statically wired (without changing the server-switch connections, and subject to the same degree constraints) can also be set up via topology adaptation. But as noted in §2.2, statically-wired networks cannot exceed the TP benchmark. Can topology adaptation use the additional flexibility to reach, or even exceed, TP?

The model of topology adaptation considered here is restrictive in two ways: (a) It only allows direct connections between communicating ToRs, rather than allowing use of idle ToRs as relays. (The numerous topology-adaptation designs referenced, all prioritize direct connections between ToR-pairs with traffic demand.) (b) It imposes the use of multi-hop routes over adaptive links, which is less bandwidth-efficient than buffering and switching-over one-hop links. (Without this restriction, large networks would require extremely large buffers and / or significant redesign at multiple layers.) We shall see below, that under this model, topology adaptation does not achieve TP. Future work will address topology adaptation more broadly.

Intuitively, for such designs, uniform traffic is a difficult case, because it removes opportunities to take advantage of traffic skew. If a subset of the racks are involved in all-to-all traffic among themselves, these racks can each contribute only the limited amount of connectivity they each have to any interconnection between them — ultimately, each "flexible" connection (*e.g.,* a wireless antenna) is plugged into an electrical port on the top-of-rack switch.

Consider a network with 54 switches, each with 12 ports, 6 of which are attached to servers. The devices and links necessary for this discussion are shown in Fig. 3. Now assume that the TM involves servers on only the 9 racks at the bottom. Thus, servers on each of the other 45 switches are inactive and irrelevant, and one can think of these switches as just 6-port devices. These 45 switches can then be connected in a standard fat-tree topology with $k = 6$, while exposing 54 of their ports to traffic sources and sinks between which they provide full bandwidth. These 54 ports can be connected in any convenient manner to the 9 switches with active servers, thus providing full bandwidth between all active servers.

Notice that this topology does not connect any of the 9 switches with traffic demands directly, unlike direct-connection heuristics. In fact, direct-connection heuristics would favor only the edges between these switches. For the 9-rack all-to-all TM in this example, the upper bound (computed in the manner described in [17]) on the throughput of *any* topology built with only links between these switches (*i.e.,* any topology with 9 switches, each with 6 servers and 6 network ports) is 80% of the full throughput. In hindsight, this observation on the limited nature of direct-connection heuristics may seem obvious, but I have not seen

[2]This covers all possible TMs constrained only at the servers.

[3]I acknowledge the parallels with energy proportional networking [1, 12], but note that EPN addresses a somewhat different question along the lines of which links to turn off, or change the data rate on. It does not examine the issue of arbitrary subsets of servers in an oversubscribed network achieving high throughput.

it in prior work. Even the most recent proposal, ProjecToR [6], prioritizes direct connections in its "opportunistic topology". For the TMs under discussion, their algorithms would use almost all flexible links for the opportunistic topology, leaving only 2 links per busy ToR to connect to the idle ToRs, thus severely impairing their potential as relays. Even for the "dedicated topology" (if these TMs were to be a long-term, stable feature), ProjecToR's algorithm will pick a direct-connection topology, which indeed minimizes their weighted path length metric, instead of a topology which makes better use of idle ToRs, at the cost of longer paths.

While I am as yet unable to resolve the broader question "Can topology adaptation reach, or even exceed, TP?", for the model of topology adaptation considered here, the answer is negative. In fact, the following results show that this model could perform worse than certain statically-wired networks.

3 Wired $\neq$ Inflexible

The Helios authors' grim assessment regarding the inflexibility of statically-wired networks was noted in §1. This assessment was perhaps accurate at the time, but in the intervening years, several network designs have been proposed, which as we shall see here, are much closer to throughput proportionality than the fat-tree. Further, these results will also show that topology adaptation when restricted to schemes that prioritize direct links between communicating ToRs may perform worse. First, however, I describe the methodology used to arrive at these results and its limitations.

3.1 Methodology

Both statically-wired networks and topology adaptation are evaluated here under skewed but difficult (ideally, worst-case) TMs — in line with the definition of traffic proportionality §2.2, we want an oversubscribed network to provide high throughput for *any* TM involving small subsets of servers.

This evaluation borrows heavily from recent work on comparing topologies [14], using the throughput evaluation tool made available [13] by those authors, to evaluate three recently proposed statically-wired topologies — SlimFly [3], Jellyfish [18], and Longhop [19] — under difficult, skewed TMs. Each topology is evaluated assuming perfect TE, by maximizing the minimum flow (for fairness across flows) in a fluid-flow model using a linear program solver [9]. The assumption of perfect TE is discussed in §4.

A series of skewed TMs is used, increasing the fraction of server racks participating in the TM, with no flows between non-participating racks. For static networks, only results for the longest matching TMs [14], which have been shown (empirically) to be harder than TMs such as all-to-all, are included. In these longest matching TMs, each participating rack sends all its traffic to one other rack. The rack-pairs are chosen such as to maximize the distance between communicating racks, using a heuristic: maximum-weight matching, with the weights being the distances between racks. Intuitively, flows along long paths consume resources on many edges, and the large rack-to-rack flows reduce opportunities for load balancing traffic, unlike TMs with many smaller flows headed to different destinations. Thus, while finding the worst-case TM is a computationally non-trivial problem the complexity of which is not well understood [14], I made my best efforts to evaluate statically-wired networks under difficult TMs.

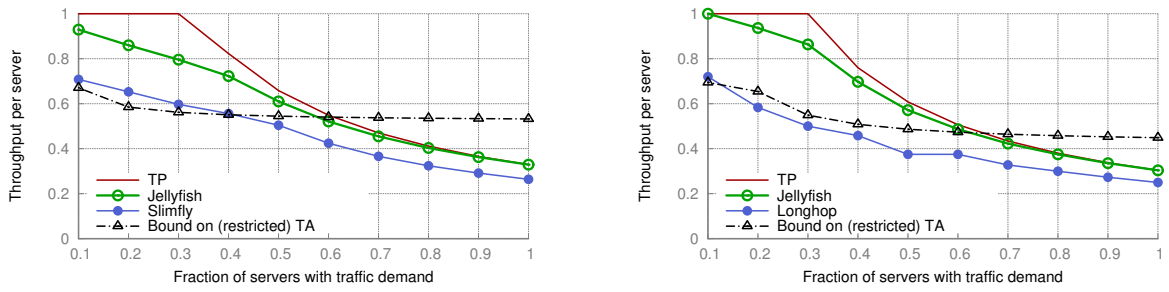
A generic adaptive topology is considered, which allows a certain number of connections per rack to be flexible, allowing each to be connected to other racks' available flexible connections. This allows arbitrary topologies under a degree constraint at every ToR. This model covers (or is more general than) past designs [5, 7, 10, 11, 16, 21, 23] [6][4] [4, 15][5]. As described earlier, multi-hop routes over flexible connections are allowed.

In the context of topology adaptation, longest matching TMs are meaningless: by changing the topology, distances between racks can be changed. As discussed earlier, uniform traffic where every participating rack equally communicates with every other participating rack is difficult for such designs. Instead of evaluating specific mechanisms for computing the flexible topologies, I compute an upper-bound (using the method in [17]) on the performance of *any* topology which could be built using fixed degree at each ToR, limited to the ToRs participating in the TM. As noted before, this bound only captures the restricted topology adaptation model. For TMs where only a subset of racks communicate with each other, such schemes would use most available ports on the involved ToRs for connectivity between these racks, perhaps connecting a minimal number of ports to the non-participating racks to maintain network connectivity. An example where this strategy is sub-optimal was provided in §2.3; here, this problem is quantified more broadly.

Limitations: (1) I am as yet unable to comment more broadly on the relative power of topology adaptation versus traffic engineering, limiting the analysis to a restricted model of topology adaptation. This is a primary goal of future work. For now, I hope that the reader recognizes that so far, the community has not even raised this question. (2) These comparisons give a sizable unfair advantage to topology adaptation: the adaptive topology is given the same number of *flexible* ports as the static networks is given *fixed* ports even though flexible ports are costlier ($\sim 2\times$ or more for recent designs [6, 11] even after including cable cost with the fixed port). Unsure of how to *fairly* adjust the compared topologies for equal cost, I left the balance tipped in favor of topology adaptation. (3) Only throughput was evaluated, and not latency. However, each of the 3 static networks evaluate provide shorter paths than fat-trees. Further, most of the latency in such networks is not propagation delay, but

[4] As those authors note, ProjecToR's ToR-fan-out of 18,432, is with regard to *different* configurations, not at the same time.

[5] By allowing parallel edges between the same pair of racks, this model also captures designs which use multiple wavelengths with WDM between the same pair of racks.



(a) (b)

Figure 4: *Throughput proportionality and (restricted) topology adaptation compared with (a) SlimFly and a same-equipment Jellyfish network; and (b) Longhop and a same-equipment Jellyfish network. An equal-cost, oversubscribed fat-tree would flat-line at throughput per server of roughly* 0.2 *in both cases.*

queuing, the reduction of which is a hot research topic. These results can be interpreted as showing only that the bandwidth is available; translating it to low latency or flow completion time is up to routing, congestion control, flow scheduling, etc.

3.2 Results

All 3 static networks evaluated achieve much higher performance than a same-cost oversubscribed fat-tree would achieve. Fig. 4(a) shows results for SlimFly (578 ToRs, 25 network- and 24 server-ports per ToR) [3], a Jellyfish [18] topology built with exactly the same equipment (including servers), the throughput proportionality (TP) curve using the throughput achieved for Jellyfish at x=1.0 as the base (*i.e.,* α in Fig. 2), as well as an upper bound on the throughput of *any* topology built with a restricted version of topology adaptation (TA), where the topology connects all the ports of the participating racks to each other. As a smaller and smaller fraction of servers is involved in the TM (leftward along the x-axis), throughput increases. For a hypothetical TP-network built at the same oversubscription as Jellyfish, in this case, when fewer than 30% of the servers are involved, each would obtain full throughput. An equal-cost fat-tree (not shown) would flat-line at ∼20% throughput. For (restricted) topology adaptation, even the upper bound is significantly lower than Jellyfish's throughput when a smaller fraction of servers is involved. This is the operating regime for many deployments — recent measurements across 4 large production clusters showed 46-99% of rack-pairs exchanging *no* traffic [6].

Fig. 4(b) shows broadly similar results for the Longhop topology [19] (512 ToRs, 10 network- and 8 server-ports per ToR). In this configuration, Jellyfish is even closer to TP. For SlimFly and Longhop, results are from a single run – both the TM and the topology are deterministic. For Jellyfish, results are averages over 10 runs. The largest variance across these experiments was less than 3% of the mean.

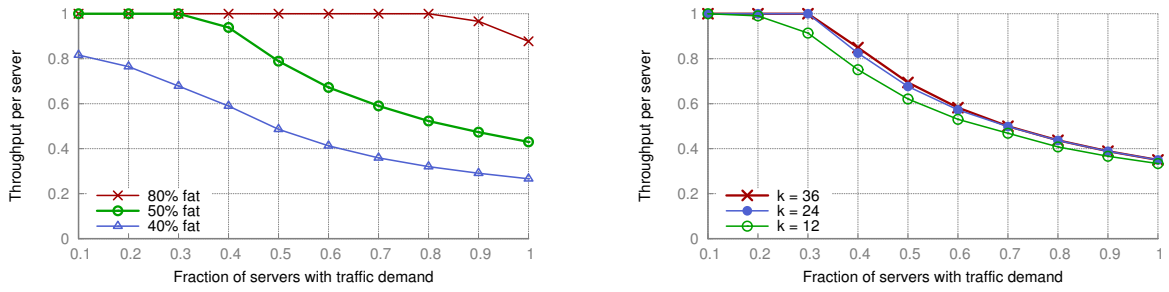
I reiterate: TA is more general than traffic engineering — at the very least, TA could just build the same topology as the best-known static network (although at greater cost). I only evaluated direct-connection heuristics, and as shown, even these could have a significant (more than 2×) advantage over an oversubscribed fat-tree, so the results do not contradict the claims made by topology adaptation work with regards to the performance advantage over oversubscribed fat-trees. Unlike claims in TA literature however, TA does not achieve performance similar to full fat-trees here. This is likely due to the workloads tested being adversarial, although a distributed join or a map-reduce shuffle over a set of racks could conceivably produce such TMs. Even if TA beats the bound[6] for "real" workloads (for which, unfortunately, no standard, replicable benchmark exists), this does not diminish static networks, which achieve high throughput even under adversarial workloads, as plausible alternatives to TA. This raises a related issue worth future exploration: can the design of traffic engineering also benefit from the workload characteristics exploited by TA, instead of the traditional high bar of "near-optimal under near-arbitrary traffic"?

I also compared fat-trees and Jellyfish directly. Fig. 5(a) shows the results for a Jellyfish network built using the same number of servers, and 80%, 50%, and 40% of the switches available to a full fat-tree with k=20 (*i.e.,* 500 switches, with 20 ports each, and 2000 servers). These results are averages over 10 runs, and variances are less than 1% of the mean, except for the smallest TMs in the most oversubscribed network (40%), where they are smaller than 5%. With 50% of the fat-tree's switches (and 37.5% its network cables; the server-switch cables are the same number, of course), Jellyfish can provide nearly full bandwidth as long as <40% of servers participate in the TM.

Fig. 5(b) shows that Jellyfish's advantage is consistent, or improves with scale, as it is built using the same set of switches as full fat-trees built with k=12, 24, and 36, but with twice the servers in each case. Note that the Jellyfish topologies are being impaired more severely than may be evident at first: adding more servers at a switch with the same port-count also reduces the number of network ports available to connect to other switches. Results for k=12 and 24 are averages over 10 runs, with variances $\leq$ 1% for all but the smallest TMs at the smaller size. For k=36, only one run was used, due to the limited running time available, and the knowledge of small variance at larger sizes.

Thus, the static topologies show significant potential for

[6]The bound applies to uniform TMs, including the restricted all-to-all TMs considered here. For certain TMs, like a rack-level permutation, TA can trivially provide full throughput.



(a) (b)

Figure 5: *A direct comparison between a full bandwidth fat-tree and an oversubscribed Jellyfish network: (a) Jellyfish with fewer switches — 80%, 50%, and 40% — as a $k = 20$ fat-tree, while supporting the same number of servers. With 50% fewer switches, it still provides nearly full bandwidth connectivity between any 40%-subset of servers. (b) This advantage is consistent or improves with larger k (12, 24, 36). Jellyfish supports 2× the fat-tree's servers in each case.*

accommodating skewed TMs, and these results call for: (a) investigating advanced traffic engineering mechanisms over such networks; and (b) exploring topology adaptation methods beyond heuristics which prioritize direct connections between communicating ToRs.

4 But ... *optimal* TE?

Efficient TE, rather than optimal, would suffice for static networks to deliver comparable or higher performance at the same cost as TA. Even in comparisons ignoring TA's additional costs, for TMs involving 40% or fewer servers, just 70%-of-optimal TE would make Jellyfish's performance comparable to state-of-the-art TA (Fig. 4). In those same scenarios, with optimal TE, Jellyfish beats the fat-tree by 3× or more (Fig. 4), and achieving smaller gains in practice would still be valuable. Further, developing efficient TE for appropriate static networks is likely the path of lesser resistance, as TA faces significant challenges of its own: lack of operator experience with the technologies involved; unfamiliar problems in device packaging, spatial planning, and organization; monitoring and debugging the highly ephemeral networks; impact of environmental factors like dust, vibration, and temperature on device alignment and functioning; the reliability and lifetime of the used devices; and perhaps other, as yet unanticipated problems. Static networks on the other hand exploit standard, commodity equipment that has been in deployment for years. While Jellyfish's randomness may deter some operators, deterministic options like SlimFly would also yield meaningful gains, and newer deterministic topologies like XPander [20] (evaluation left to future work) may even achieve performance identical to Jellyfish.

I acknowledge the possibility that topology adaptation could provide a latency advantage over static networks: connecting ToR pairs directly shortens paths and reduces the number of queues a flow crosses. However, the extent of this advantage under practical topology adaptation, TE, and congestion control remains entirely unknown. Future work will explore whether similar or better results can be achieved over static networks at comparable or lower cost.

5 Discussion & Conclusion

This work suggests an intuitive metric for a network's flexibility in accommodating skewed TMs — throughput proportionality — and demonstrates that while fat-trees indeed score low on TP, other recent designs for statically-wired networks would be significantly better *if* traffic engineering were optimal. Further, presently common methods of topology adaptation may not exploit the evidently greater flexibility of the broader idea of topology adaptation, and can perform worse than statically-wired networks.

My goal is not to undermine the clear ingenuity of topology-adaptation work, but rather to assess ways of pushing both static and dynamic network designs farther. The findings presented here raise several questions worth exploring:

- What is the difference in the relative power of traffic engineering on static networks, versus adapting the topology to traffic? Which techniques can reach or exceed traffic proportionality?
- Say a "flexible" port costs δ times a static, electrical port. Then what is the largest δ for which optimal (practical) topology adaptation beats optimal (practical) TE over the best-known *equal-cost* static network?
- What traffic engineering mechanisms work best for large oversubscribed networks when only a fraction of servers have high traffic demands? Can something simple like Valiant load balancing [22] achieve good results in this setting? With a large fraction of servers and ToRs not participating in the TM, randomizing traffic through less congested switches could work well.

In summary, I hope that this work encourages a rethink of the design space for data center networks and their accompanying traffic engineering or topology adaptation mechanisms.

Acknowledgments

I thank Ratul Mahajan for his insights on the limitations of the topology adaptation model considered here; and Torsten Hoefler, Simon Kassing, Michael Schapira, Laurent Vanbever, and Roger Wattenhofer for their helpful feedback.

6 References

[1] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu. Energy proportional datacenter networks. *ACM/IEEE ISCA*, 2010.

[2] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. *ACM SIGCOMM*, 2008.

[3] M. Besta and T. Hoefler. Slim Fly: A Cost Effective Low-Diameter Network Topology. *IEEE/ACM SC*, 2014.

[4] K. Chen, A. Singla, A. Singh, K. Ramachandran, L. Xu, Y. Zhang, X. Wen, and Y. Chen. OSA: An Optical Switching Architecture for Data Center Networks with Unprecedented Flexibility. *USENIX NSDI*, 2012.

[5] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat. Helios: A hybrid electrical/optical switch architecture for modular data centers. *ACM SIGCOMM*, 2010.

[6] M. Ghobadi, R. Mahajan, A. Phanishayee, H. Rastegarfar, P.-A. Blanche, M. Glick, D. Kilper, J. Kulkarni, G. Ranade, and N. Devanur. ProjecToR: Agile Reconfigurable Datacenter Interconnect. *ACM SIGCOMM*, 2016.

[7] M. Glick, D. G. Andersen, M. Kaminsky, and L. Mummert. Dynamically reconfigurable optical links for high-bandwidth data center networks. *Optical Fiber Communication Conference*, 2009.

[8] Google. Pulling Back the Curtain on Google's Network Infrastructure, 2015. https://goo.gl/hx0vz3.

[9] Gurobi Optimization Inc. Gurobi optimizer reference manual. http://www.gurobi.com, 2013.

[10] D. Halperin, S. Kandula, J. Padhye, P. Bahl, and D. Wetherall. Augmenting data center networks with multi-gigabit wireless links. *ACM SIGCOMM*, 2011.

[11] N. Hamedazimi, Z. Qazi, H. Gupta, V. Sekar, S. R. Das, J. P. Longtin, H. Shah, and A. Tanwer. Firefly: A reconfigurable wireless data center fabric using free-space optics. *ACM SIGCOMM*, 2014.

[12] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown. Elastictree: Saving energy in data center networks. *USENIX NSDI*, 2010.

[13] S. A. Jyothi, A. Singla, B. Godfrey, and A. Kolla. Topobench. https://github.com/ankitsingla/topobench.

[14] S. A. Jyothi, A. Singla, B. Godfrey, and A. Kolla. Measuring and understanding throughput of network topologies. *CoRR*, abs/1402.2531, 2014.

[15] Y. J. Liu, P. X. Gao, B. Wong, and S. Keshav. Quartz: A New Design Element for Low-Latency DCNs. *ACM SIGCOMM*, 2014.

[16] G. Porter, R. Strong, N. Farrington, A. Forencich, P. Chen-Sun, T. Rosing, Y. Fainman, G. Papen, and A. Vahdat. Integrating microsecond circuit switching into the data center. *ACM SIGCOMM*, 2013.

[17] A. Singla, P. B. Godfrey, and A. Kolla. High throughput data center topology design. *USENIX NSDI*, 2014.

[18] A. Singla, C.-Y. Hong, L. Popa, and P. B. Godfrey. Jellyfish: Networking Data Centers Randomly. *NSDI*, 2012.

[19] R. V. Tomic. Optimal networks from error correcting codes. *ACM/IEEE ANCS*, 2013.

[20] A. Valadarsky, M. Dinitz, and M. Schapira. Xpander: Unveiling the Secrets of High-Performance Datacenters. *ACM HotNets*, 2015.

[21] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. S. E. Ng, M. Kozuch, and M. Ryan. c-through: Part-time optics in data centers. *ACM SIGCOMM*, 2010.

[22] R. Zhang-Shen and N. McKeown. Designing a predictable internet backbone with valiant load-balancing. *IEEE IWQoS*, 2005.

[23] X. Zhou, Z. Zhang, Y. Zhu, Y. Li, S. Kumar, A. Vahdat, B. Y. Zhao, and H. Zheng. Mirror mirror on the ceiling: Flexible wireless links for data centers. *ACM SIGCOMM*, 2012.