Energy Proportional Datacenter Networks

Dennis Abts
Google Inc.
Madison, WI
dabts@google.com

Michael R. Marty
Google Inc.
Madison, WI
mikemarty@google.com

Philip M. Wells
Google Inc.
Madison, WI
pwells@google.com

Peter Klausler
Google Inc.
Madison, WI
pmk@google.com

Hong Liu
Google Inc.
Mountain View, CA
hongliu@google.com

ABSTRACT

Numerous studies have shown that datacenter computers rarely operate at full utilization, leading to a number of proposals for creating servers that are *energy proportional* with respect to the computation that they are performing. In this paper, we show that as servers themselves become more energy proportional, the datacenter network can become a significant fraction (up to 50%) of cluster power. In this paper we propose several ways to design a high-performance datacenter network whose power consumption is more proportional to the amount of traffic it is moving— that is, we propose *energy proportional datacenter networks*.

We first show that a flattened butterfly topology itself is inherently more power efficient than the other commonly proposed topology for high-performance datacenter networks. We then exploit the characteristics of modern plesiochronous links to adjust their power and performance envelopes dynamically. Using a network simulator, driven by both synthetic workloads and production datacenter traces, we characterize and understand design tradeoffs, and demonstrate an 85% reduction in power — which approaches the ideal energy-proportionality of the network.

Our results also demonstrate two challenges for the designers of future network switches: 1) We show that there is a significant power advantage to having independent control of each unidirectional channel comprising a network link, since many traffic patterns show very asymmetric use, and 2) system designers should work to optimize the high-speed channel designs to be more energy efficient by choosing optimal data rate and equalization technology. Given these assumptions, we demonstrate that energy proportional datacenter communication is indeed possible.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network communications; Topology; B.4.3 [**Interconnections**]: Fiber optics; Topology

General Terms

Performance, Design

Keywords

Low-power networking, Datacenter networks, Interconnection networks

1. INTRODUCTION

The cost of power and its associated delivery and cooling are becoming significant factors in the total expenditures of large-scale datacenters. Barroso and Hölzle recently showed a mismatch between common server workload profiles and server energy efficiency [3]. In particular, they show that a typical Google cluster spends most of its time within the 10-50% CPU utilization range, but that servers are inefficient at these levels. They therefore make the call for *energy proportional computing systems* that ideally consume almost no power when idle and gradually consume more power as the activity level increases.

Servers and their processors are the obvious targets to improve energy proportionality because they are today's primary power consumers. Today's typical multi-tiered datacenter network consumes little power, relative to servers, because of its high degree of *over-subscription*. As an example of over-subscription, machines connected to the same rack switch (i.e., the first tier) have significantly more bandwidth to each other than to machines in other racks. The level of bandwidth over-subscription is typically an order of magnitude or more for each subsequent tier.

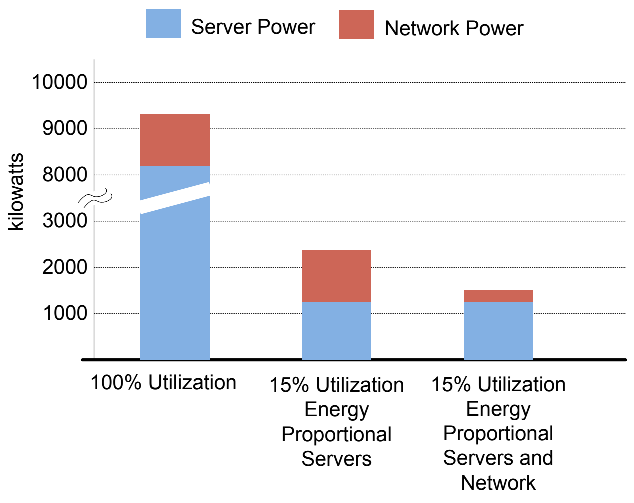


Figure 1: Comparison of server and network power

Going forward, there are many factors that will drive adoption of high-performance datacenter networks with much greater bisection bandwidth, thereby eliminating or reducing over-subscription. Greenberg et al. [7] argue that less over-subscription enables dynamic resource allocation across large pools of servers. Thus clusters can be virtualized and servers can achieve better utilization because communication-intensive jobs can be run anywhere rather than being constrained to a single tier or rack (i.e., rack affinity). Over-subscription can also couple the failure domains to the bandwidth domains. For example, if applications must schedule with rack affinity for bandwidth reasons, then the failure of a rack switch or power distribution unit can adversely affect service availability. Decoupling the failure domain from the available network bandwidth domain can enable greater opportunities for cluster-level file systems such as Lustre [24], GFS [6], Hadoop [5] and more. Barroso and Hölzle also suggest that even with abundant amounts of thread-level parallelism, the effects of Amdahl's Law can start to become a limitation without cost-effective, high port-count, high bandwidth networking [4]. We also see the emergence of faster storage technologies, like flash and phase-change memory, as another driver of increased cluster network bandwidth. A substantial increase in cluster bisection bandwidth, however, requires significantly more switching chips with faster, more power-consuming links. We therefore argue that network power will become a first-order operating expenditure in future datacenter networks.

Figure 1 illustrates the impact of network power in a target system we develop in Section 2, where we assume each of 32k servers consumes 250 watts at peak load. Even with a high-performance datacenter network based on a folded-Clos topology (i.e., fat trees) [2, 7, 17, 20] the network consumes only 12% of overall power at full utilization. But since servers typically operate at much lower levels of utilization, and are becoming more energy proportional, the network power cannot be ignored — much in the same way that Amdahl's arguments for performance demonstrate diminishing returns when focusing on only one part of the system. For instance, if the system is 15% utilized (servers and network) and the servers are fully energy-proportional, the network will then consume nearly 50% of overall power. At 15% load, making the network energy proportional results in a savings of 975,000 watts regardless of whether servers are energy proportional. Assuming an average industrial electricity rate of \$0.07 per kilowatt-hour [25], and a datacenter PUE of 1.6,[1] this results in a savings of approximately \$3.8M over a four year service life of the network!

Unfortunately, existing or proposed datacenter networks exhibit very little *dynamic range*. That is, the power consumed when the network is idle is still significant compared to the power consumed when the network is fully utilized. A primary reason is that high-speed channels are "always on" regardless of whether they are flowing data packets. The channels cannot be quickly deactivated and then reactivated without negotiation from both sides of the channel to establish data rate, symbol alignment, and lane alignment. Further exacerbating the problem is that deactivating a link appears as if the link is faulty to the routing algorithm, in which case packets must either be buffered or routed around the deactivated link. Buffering for extended periods of time is not feasible and will lead to packet discards or backpressure depending on the network's flow control mechanism. On the other hand, changing routes in many networks require coordination among all nodes to ensure that no newly injected traffic takes a path that would cross an inactive link.

Our goal is to provide a network that supports *energy proportional communication*. That is, the amount of energy consumed is proportional to the traffic intensity (offered load) in the network. We tackle this goal from multiple angles. First we demonstrate that a datacenter network based on the flattened butterfly topology results in a more power-efficient network and therefore lowers operational expenditures. With a solid foundation for a high-performance power-efficient network, we then add dynamic range by periodically estimating the future bandwidth needs of each link and reconfiguring its data rate to meet those requirements while consuming less power. Specifically, this paper makes the following contributions:

- We analytically demonstrate how a network based on the flattened butterfly topology consumes less power than a bandwidth-comparable and size-comparable folded-Clos (recently proposed for datacenter interconnects [2, 7, 20]). Our example demonstrates \$1.6M in energy savings over a four-year lifetime of the network.
- We develop techniques to make the datacenter network more energy proportional by increasing its *dynamic range*. We do so by exploiting properties of both the flattened butterfly topology and modern high-speed links that support multiple data rates.
- Using detailed event-driven simulation, we evaluate our energy-proportional communication schemes with traces taken from a production datacenter running web search applications. We show a $6\times$ reduction in power using the same topology, leading to a potential four-year energy savings of an *additional* \$2.4M in a full-scale network.
- Finally, based on our analysis, we propose opportunities for switch designers to further enable energy proportional networks, and we introduce the concept of *dynamic topologies*, where in addition to dynamically adjusting the data rate of existing links, those links are dynamically powered up and down, in effect creating a dynamically changing topology.

2. LOW-POWER, HIGHLY-SCALABLE TOPOLOGIES

In this section, we propose the flattened butterfly (FBFLY) topology as a cornerstone for energy-proportional communication in large-scale clusters with 10,000 servers or more. We will show why the topology, by itself, is lower in power than a comparable folded-Clos with equivalent bisection bandwidth and same number of hosts. While commonplace in high-performance computing interconnects, folded-Clos networks (aka fat trees [17]) are being proposed as a means to provide a datacenter network with little or no over-subscription [7, 9].[2] Not only does the flattened butterfly result in lower power at full utilization, but it is also more amenable to techniques to increase the dynamic range of network power, the topic of Section 3.

2.1 Background: Flattened Butterfly

The *flattened butterfly* (FBFLY) k-ary n-flat topology [15] takes advantage of recent high port count switches [16, 23] to create a scalable, yet low-diameter network. This is accomplished by making the deliberate tradeoff of fewer physical links compared to a

[1]The PUE, or Power Usage Effectiveness, is the ratio of a datacenter's total power to the power actually used by computing equipment. We use a value of 1.6, which is the middle-point between industry-leading datacenters (1.2 [12]) and the EPA's 2007 survey (2.0 [26]).

[2]In this paper, we do not consider traditional multi-tiered cluster networks, common in today's datacenters, because of their high degrees of over-subscription, leading to low performance for workloads that are not localized to a single rack or tier.

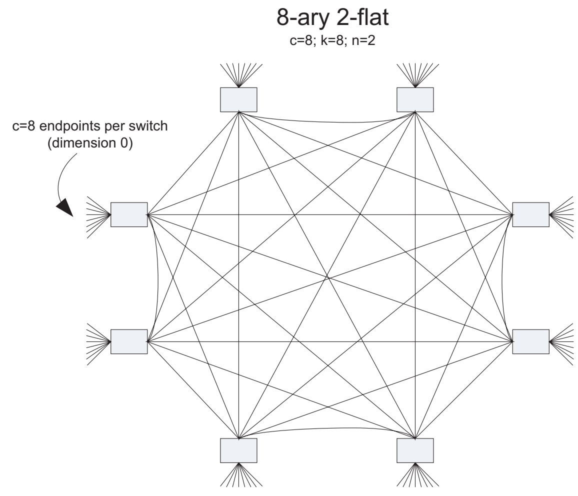


Figure 2: Logical diagram of an 8-ary 2-flat flattened butterfly topology.

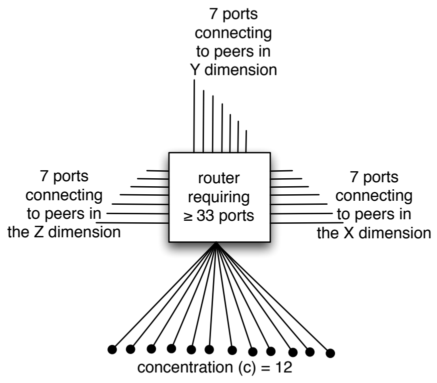


Figure 3: A 33-port router necessary to implement an 8-ary 4-flat with a concentration of 12 (over-subscription of 3:2).

folded-Clos, at the expense of increased routing complexity to load balance the available links. The FBFLY topology, for example, *requires* adaptive routing to load balance arbitrary traffic patterns, whereas a folded-Clos has multiple physical paths to each destination and very simple routing.

A flattened butterfly is a multi-dimensional direct network, in many ways like a torus (k-ary n-cube), where every switch in the network is connected to hosts as well as other switches. The biggest difference is that in a torus, each dimension is connected as a ring, and in an FBFLY, each dimension is fully connected. That is, within a FBFLY dimension all nodes connect to all others.

This interconnection is depicted in Figure 2, which shows 2-dimensional FBFLY (8-ary 2-flat) with $8 \times 8 = 64$ nodes and eight 15-port switch chips. Scaling the number of dimensions in a FBFLY consists, essentially, of taking this single 8-switch group, replicating it 8 times, and interconnecting each switch with its peer in the other 7 groups (i.e., each upper-left switch connects to the other 7 upper left switches in the other 7 groups). Doing so yields an 8-ary 3-flat with $8^3 = 512$ nodes, and 64 switch chips each with 22 ports. The flattened butterfly topology and packet traversal can also be explained with a simple metaphor. Consider a chessboard, with eight squares in each of two dimensions. Each square on the chessboard represents a switch that routes packets. Packets traverse the flattened butterfly in the same manner that a rook moves on a chessboard: on each turn (hop), we can move to an arbitrary square (switch) in a given dimension.

Though a FBFLY scales exponentially with the number of dimensions, you can also scale by increasing the radix. When possible, it is advantageous to build the highest-radix, lowest dimension FBFLY that scales high enough and does not exceed the number of available switch ports. This reduces the number of hops a packet takes as well as the number of links and switches in the system.

As you scale up a FBFLY, there are two properties of the topology that reduce both capital and operational expenditures: 1) the topology can take advantage of *packaging locality* in the sense that nodes which are in close physical proximity can be cabled with inexpensive copper cables, 2) it uses fewer optical transceivers and fewer switching chips than a comparable folded-Clos, and therefore consumes less power, as we show in Section 2.2.

2.1.1 *Over-subscription in Flattened Butterfly*

While we argue for high performance datacenter networks with little over-subscription, the technique remains a practical and pragmatic approach to reduce power (as well as capital expenditures), especially when the level of over-subscription is modest. Though the original work on the flattened butterfly topology did not describe it, over-subscription can easily be achieved, if desired, by changing the *concentration* (c), or number of terminal nodes, at each switch. For example, we could provide a concentration of 12 nodes per switch in an 8-ary 4-flat using a 33 ported router (see Figure 3) which would allow the network to scale to: ck^{n-1}, or $12 \times 8^3 = 6144$ nodes, instead of 4096 nodes with 8 nodes per switch. We express this using the tuple (c, k, n) to describe a k-ary n-flat with c nodes per switch. Nonetheless, our evaluation and topology comparison in this paper does not over subscribe the network in order to provide the maximum performance ceiling when operating at full power.

2.2 Topology Power Comparison

A datacenter network based on the flattened butterfly topology uses less hardware compared to a folded-Clos network of equivalent size and performance. This, by itself, results in a more power-efficient network and lower operating expenditures. To illustrate this power efficiency, we develop a baseline system with 32k server nodes and compare the first-order part counts between a flattened butterfly and folded-Clos. The number of switch chips, in particular, dominates the power consumption of the network. We also report the number of expensive optical transceivers required by the two topologies, since they tend to dominate the capital expenditure of the interconnect. A more detailed capital expenditure cost comparison between the two topologies can be found in [15].

For the purposes of this comparison, we make the following assumptions:

- A 36-ported switch (router node) where ports operate at 40 Gb/s,
- Each switch consumes 100 watts regardless of whether of what combination of "always on" links it is driving, e.g., electrical backplane, electrical cable, or optical,[3]
- Each host network interface controller (NIC) consumes 10 watts at full utilization, and
- The same switch chips are used throughout the interconnect.

[3]We arrive at 100 Watts by assuming each of 144 SerDes (one per lane per port) consume ≈0.7 Watts

Table 1: Comparing energy consumption of different topologies for fixed bisection bandwidth

Parameter Description	Folded Clos	FBFLY (8-ary 5-flat)
Number of hosts (N)	32k	32k
Bisection B/W (40 Gb/s links)	655 Tb/s	655 Tb/s
Number of electrical links	49,152	47,104
Number of optical links	65,536	43,008
Number of switch chips	**8,235**	**4,096**
Total power (W)	**1,146,880**	**737,280**
Power per bisection B/W (W/Gb/s)	1.75	1.13

The second assumption simplifies the comparison even though our data in Figure 5, showing the profile of an existing switch chip, uses 25% less power to drive an electrical link compared to an optical link. This represents a second-order effect in our comparison, and is actually disadvantageous for the flattened butterfly, since other factors come into play (such as packaging locality).

When comparing topologies, something must be held constant in order for the results to be meaningful— our comparison is based on a fixed bisection bandwidth. The number of ports per switch per switch necessary to build a k-ary n-flat FBFLY with concentration c is: $p = c + (k-1)(n-1)$. Thus a 32k node 8-ary 5-flat with $c = k = 8$ requires 36 ports. We assume that all short electrical links (<5m) will use passive copper cables, and anything longer requires an optical link. The comparable 3-stage folded-Clos is built from the same 36-ported switch. For the flattened butterfly, the first dimension, which interconnects all the switches within a local domain, can use short (<1m) electrical links. Similarly, all $c = k = 8$ links from the hosts to switch can be electrical. In general, this *packaging locality* allows for e electrical links: $e = (k-1) + c$, which make a significant fraction of the overall links:

$$f_e = \frac{(k-1)+c}{c+(k-1)(n-1)}$$

In this case $\frac{15}{36} \approx 42\%$ of the FBFLY links are inexpensive, lower-power, electrical links. However, for ease of comparison we assume that *all* links are the same power efficiency (which does not favor the FBFLY topology).

The total number of switches for the FBFLY is given by:

$$S_{FBFLY} = k^{n-1} = 8^4 = 4,096$$

For the folded-Clos, we use 27 36-port switches to build a 324-port non-blocking router chassis for stage-2 and stage-3 of a multi-staged folded-Clos network.[4] We assume that the backplane connections within the 324-port stage-2 and stage-3 switches are free. Thus the folded-Clos requires

$$S_{stage3}\left\lceil\frac{32k}{324}\right\rceil = 102, S_{stage2}\left\lceil\frac{32k}{324/2}\right\rceil = 203$$

chassis, and the total number of switch chips required for the folded-Clos is: $S_{Clos} = 27 \times (S_{tier3} + S_{tier2}) = 27 \times 305 = 8,235$. In this example 32k system build with 36-ported switches, there are 8,235 switches in the folded-Clos, however only ports on 8,192 switches are used.[5] The results are summarized in Table 1.

[4]We could also build a 648-ported chassis, which might reduce cost, but will require the same number of switch chips, and same total power according to our simple model.

[5]There are some unused ports which we do not count in the power analysis.

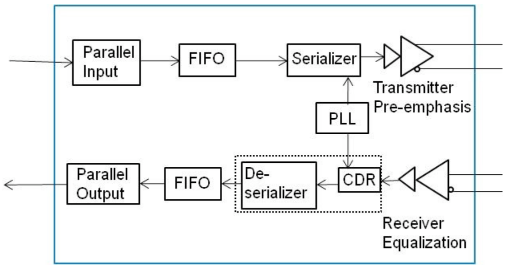


Figure 4: Example block diagram of a SerDes block.

Overall, as Table 1 shows, the cluster with the flattened butterfly interconnect uses 409,600 fewer watts than a folded-Clos with the same bisection bandwidth. Assuming an average industrial electricity rate of \$0.07 per kW hour [25] and an average PUE of 1.6 [26], using the flattened butterfly topology alone results in over \$1.6M of energy savings over a four-year lifetime of the cluster. Nonetheless, the "always on" nature of today's networks leaves a lot of cost savings on the table without additional mechanisms: Even the baseline FBFLY network consumes 737,280 watts resulting in a four year power cost of \$2.89M.

We have considered other configurations, using different switch chips with more ports and more power demand, and the trends shown in Table 1 continue to hold for this scale of cluster.

3. ADDING DYNAMIC RANGE

The previous section showed that the flattened butterfly topology inherently consumes less power than a bandwidth- and size-comparable folded-Clos. In this section, we seek to further improve network power by making communication more energy-proportional to the amount of data being transmitted. We do this by dynamically tuning individual links to match the required performance while consuming as little power as possible.

3.1 Background: Plesiochronous Links

High-speed channels are typically constructed from several serialized *lanes* that operate at the same data rate, with a *physical unit* (phit) being striped across all the active lanes. Channels commonly operate *plesiochronously*, where the core logic in the router operates at a frequency different than that of the I/O channels. All I/O channels can themselves run asynchronously, where frequency may vary a small amount ($\approx \pm$100ppm) among all the ports. The physical layer (PHY) uses a serializer/deserializer (*SerDes*, see Figure 4), which accepts a phit of data and "squeezes" it onto a high-speed serial bit stream for transmission. On the other side of the channel, the SerDes receives the serial bit stream, where it reassembles the phit and passes it up to the *data-link* layer. Frequency differences

Table 2: InfiniBand support for multiple data rates.

InfiniBand data rate	Name	Data rate (Gb/s)
Single Data Rate	1× SDR	2.5 Gb/s
	4× SDR	10 Gb/s
Double Data Rate	1× DDR	5 Gb/s
	4× DDR	20 Gb/s
Quad Data Rate	1× QDR	10 Gb/s
	4× QDR	40 Gb/s

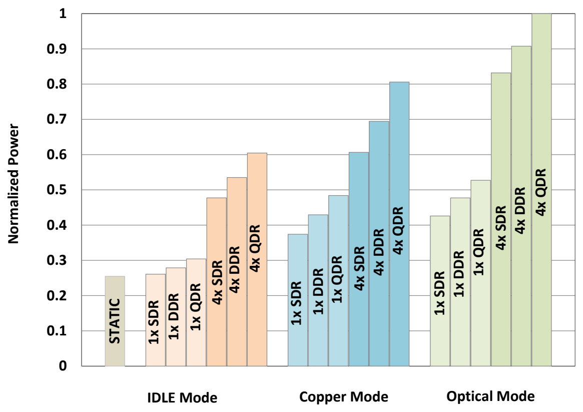


Figure 5: Dynamic range of an InfiniBand switch chip for both copper and optical links.

among ports and between ports and core router logic are absorbed by the input and output FIFOs of the SerDes.

The problem, in terms of power, is that these channels are *always on* regardless of whether they are flowing data packets, because they must still send *idle* packets to maintain byte and lane alignment across the multiple lanes. Even though they are always on, plesiochronous channels often do have *dynamic range*, in terms of their ability to vary their data rate and power consumption.

One good example of dynamic channel range is the InfiniBand architectural specification, which defines multiple operational data rates (see Table 2). Figure 5 illustrates the normalized dynamic range of an off-the-shelf InfiniBand switch available today, where it is possible to manually adjust the link rates corresponding to Table 2. The maximum link rate of 40 Gb/s is obtained with four lanes running at quad data rate (QDR) of 10 Gb/s each. However it is possible to operate the link with less lanes and at a lower data rate to reduce the power consumption of the always-on link. The dynamic range of this particular chip is 64% in terms of power, and 16X in terms of performance. Though we lack detailed dynamic power range numbers, the Cray YARC [23] switch has 64 ports that similarly can operate as 1×, 2×, or 3× lanes with each lane operating at a range of frequencies from 1.25–6.25 Gb/s. Thus each YARC link can transmit 1.25–18.75 Gb/s in each direction.

Both InfiniBand and YARC allow links to be configured for a specified speed and width, though the *reactivation time* of the link can vary from several nanoseconds to several microseconds. For example, when the link rate changes by 1×, 2×, and 4× (e.g., InfiniBand's SDR, DDR and QDR modes), the chip simply changes the receiving *Clock Data Recovery* (CDR) bandwidth and re-locks the CDR. Since most SerDes today use digital CDR at the receive path (Figure 4), the locking process for receiving data at a different data rates is fast, ≈50ns–100ns for the typical to worst case.

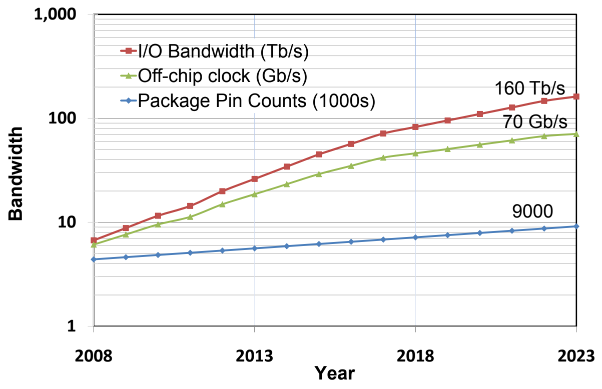


Figure 6: Bandwidth trends from International Roadmap for Semiconductors (ITRS).

Adding and removing lanes is a relatively slower process compared to link rate changes, but the process could be optimized within a few microseconds.

While there exists opportunity to improve the energy-efficiency of each channel [10], going forward we expect more I/Os per switch package, operating at higher data rates, further increasing chip power consumption (Figure 6).

3.2 Exploiting a Link's Dynamic Range

The previous section outlines how modern plesiochronous links already show dynamic range in their performance and power, but in current networks and switches, maximum link speed is typically something that must be manually configured. We propose adding additional switch mechanisms to exploit those capabilities by reconfiguring link speeds on-the-fly to match bandwidth (and power) with the traffic demand. In doing so, communication can be made more energy proportional without fundamentally changing the topology, traffic routes, or even performance of the network.

The essence of our proposal is to periodically estimate the future bandwidth needs of each link, and reconfigure its data rate to meet those requirements while consuming less power. However a couple pragmatic requirements arise when taking this approach: 1) this approach requires the ability to accurately predict the future bandwidth requirements of the link, and 2) it requires the ability to tolerate a link reactivation that is non-instantaneous.

Predicting future bandwidth demands can be done, in classic computer architecture fashion, by using recent history as a guide. This could be accomplished with the mechanisms already used for congestion sensing in adaptive routing [14, 23]. For example, credit-based link-level flow control can deliver *precise* information on the congestion of upstream receive buffers, or *channel utilization* can be used over some timescale as a proxy for congestion. However, because datacenter workloads are bursty over a wide range of timescales, care must be taken to avoid both averaging over too long of a timescale, and meta-instability arising from too-frequent reconfiguration.

A number of strategies can be employed to tolerate reactivation latency. One option is to remove the reactivating output port from the list of legal adaptive routes and drain its output buffer before reconfiguration. This scheme relies on sufficient path diversity within the topology so that traffic that might normally route to that port is able to be redirected to other paths.

A second option is to continue to allow traffic to be routed to the output port, and then either drop packets (in a lossy network), or supply back-pressure (in a loss-less network) when the output buffers fill up. Any packet routed to that port will incur an additional latency as the link reactivates, but if the reactivation is small, this latency is likely to be tolerated by the application. This scheme relies on the congestion-sensing and adaptivity mechanisms to automatically route around the link that is undergoing reconfiguration in order to avoid excessive back-pressure or packet loss.

A third option is to combine these two methods depending on the length of time the reactivation is expected to take.

Exploiting links' dynamic range is possible with other topologies, such as a folded-Clos. However, there are a couple of important factors that make a high-performance flattened butterfly a great fit. First, a FBFLY already relies on adaptive routing in order to balance the load and achieve high performance. These same mechanisms can be used to predict a link's future bandwidth demands. Second, in a FBFLY, the choice of a packet's route is inherently a *local* decision (though it must abide by global rules). This nicely matches our proposed strategy, where the decision of link speed is also entirely local to the switch chip.

3.3 Proposed Heuristics

With the inherent capabilities of the flattened butterfly topology, heuristics for power-optimizing links can range from simple to complex. We propose a simple mechanism where the switch tracks the utilization of each of its links over an *epoch*, and then makes an adjustment at the end of the epoch. We set a target utilization for each link, and if the actual utilization is less than the target, we detune the speed of the link to half the current rate, down to the minimum. If the utilization exceeds the target, then the link rate is doubled up to the maximum. The target utilization should not be set too high, because that will saturate the network and lead to excessive performance degradation. The epoch duration needs to be sized to amortize the time needed to reactivate the link, but still react quickly enough to bursty and dynamic traffic patterns.

We use link utilization over the previous epoch as the only input to our decision function. The reason we do not need to consider other common inputs for adaptive routing, such as output buffer occupancy or available credits, is that utilization effectively captures both: if we have data to send, and credits to send it, then the utilization will go up, and we should upgrade the speed of the link. If we either don't have data or don't have enough credits, utilization will fall, and there is no reason to keep the link at high speed.

When links are undergoing reactivation, we do not explicitly remove them from the set of legal output ports, but rather rely on the adaptive routing mechanism to sense congestion and automatically route traffic around the link.

3.3.1 Independent Channel Control

The routing algorithm views each unidirectional channel in the network as a routing resource. However, the physical and data-link layer may treat a *pair* of unidirectional channels, in opposing directions, as a combined entity (typically called the *link*). Thus, the load on the link may be *asymmetric*, i.e., one channel may have much higher load than the other, but the link pair must be reconfigured together to match the requirements of the channel with the highest load.

Although not a capability of current chips, we also propose and evaluate the ability to *independently* tune the data rates of the unidirectional channels of a link pair.

Based on the data in Figure 5 that reflects an existing switch chip, there is not much power saving opportunity for powering off links entirely. Nonetheless, Section 5 discusses the opportunity to power off links if future switch chips offered a true "power off" state.

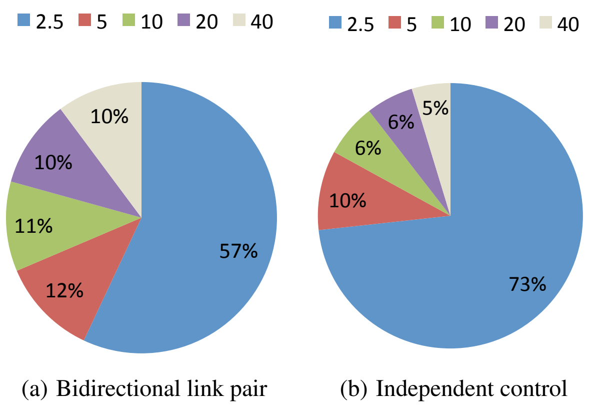


Figure 7: Fraction of time spent at each link speed (Gb/s).

4. EVALUATION

Having identified a flattened butterfly as inherently lower-power, we focus the detailed evaluation of our dynamic range techniques on that topology alone, though the proposed mechanisms and policies described can also apply to other topologies.

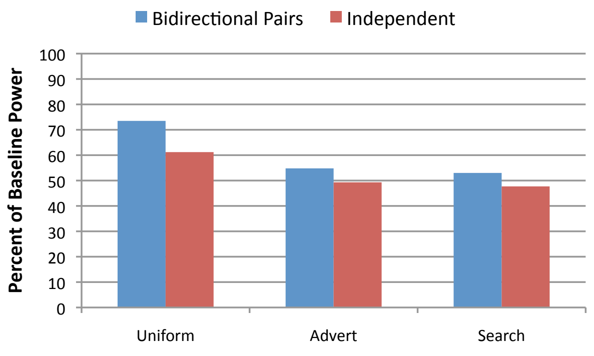
4.1 Methodology

We evaluate an energy-proportional FBFLY using an in-house event-driven network simulator, which has been heavily modified to support future high-performance networks. We model a 15-ary 3-flat FBFLY (3375 nodes) with no over-subscription, so that every host, on a uniform random traffic pattern, can inject and receive at full line rate. Links have a maximum bandwidth of 40 Gb/s, and can be detuned to 20, 10, 5 and 2.5 Gb/s, similar to the InfiniBand switch in Figure 5. We model a network with credit-based, cut-through flow control, and adaptively route on each hop based solely on the output queue depth. Switches are both input and output buffered. We assume the same reactivation time, defaulting to a conservative value of 1μs, no matter what mode the link is entering. We study the sensitivity to link activation time in Section 4.2.2.

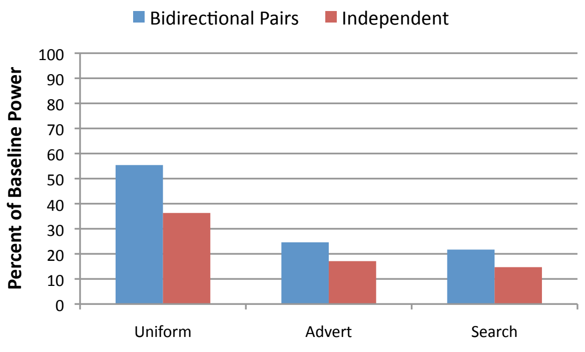
We use three workloads for this study: one synthetic and two taken from traces of a production Google datacenter. *Uniform* is a uniform random workload, where each host repeatedly sends a 512k message to a new random destination. *Advert* is a trace from an advertising service, and *Search* is from a web search service. In both applications, distributed file system traffic (e.g., GFS [6]) accounts for a significant fraction of traffic. In order to model future applications which take advantage of such a high-performance network, the later two workloads have been significantly scaled up from the original traces, and application placement has been randomized across the cluster in order to capture emerging trends such as cluster virtualization. Though scaled up, these workloads share a common trait: they are very bursty at a variety of timescales, yet exhibit low *average* network utilization of 5–25%.

4.2 Results

In this section we describe results from our simulations. We compare results to a baseline full power system, as well as one with *ideal* energy proportionality.



(a) Assuming optical channel power from Figure 5.



(b) Assuming ideally energy-proportional channels.

Figure 8: Network power when dynamically detuning FBFLY links

4.2.1 Power and Performance

Our first set of experiments explore the ability of our proposed heuristics to downgrade links to match the bandwidth required by the workload. Figure 7 shows the aggregate fraction of time links spend at different speeds for the *Search* workload. This figure assumes a 1μs reconfiguration time for the links, a 10μs epoch, and a 50% target channel utilization. We model both mechanisms described in Section 3.3 for reducing the speed and power of a link, where a bidirectional link-pair must be tuned to the same speed (Figure 7(a)), and the more flexible option, where a unidirectional channel can be tuned independently to different speeds (Figure 7(b)).

We first see that energy proportionality works: in a workload with low average utilization, most links spend a majority of their time in the lowest power/performance state.

We also see that the ability to independently control each unidirectional channel nearly halves the fraction of time spent at the faster speeds (10, 20, and 40 Gb/s), and commensurately increases the fraction of time spent at the slowest speed (2.5 Gb/s). This channel asymmetry arises in part due to workload characteristics. For example, depending on replication factor and the ratio of reads to writes, a file server in a distributed file system may respond to more reads (i.e., inject data into the network) than writes (i.e., receive data from the network), or vice versa. Perhaps surprisingly, the charts look very similar for the uniform random workload (not shown) even though the average channel utilization is very uniform. The reason is simply that the workload is bursty across the relatively short 10μs epoch.

Figure 8(a) shows the percent of power consumed by an energy proportional flattened butterfly network compared to a baseline FBFLY with all links operating at full bandwidth (40 Gb/s). We report power based on the amount of time each links spends at a given speed given the data from Figure 5. Because links spend a majority of their time in low-power mode, the power for the entire network approaches the relative power of that slowest mode ($\approx$42% of full power).

Figure 8(b) shows the same experiments as shown in Figure 8(a), except that we assume channels are ideally energy-proportional with offered load themselves. Thus a channel operating at 2.5 Gb/s uses only 6.125% the power of a channel operating at 40 Gb/s. We can see that together, energy proportional channels and the ability to independently tune each unidirectional channel provide a $6\times$ advantage in terms of network power for both advertising and search workloads.

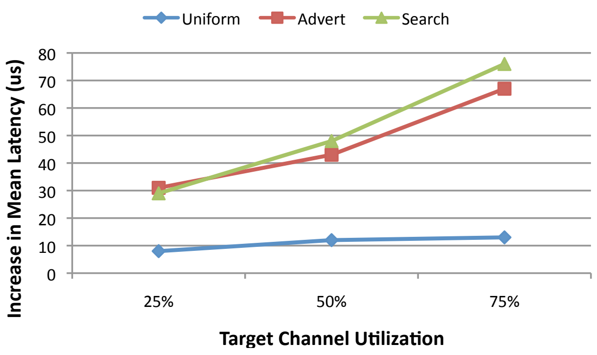
The middle column of points in Figure 9(a) demonstrates the difference in performance when using an energy proportional FBFLY. This figure shows that the increase in mean latency, at a 50% target utilization and 1μs reactivation time, is only 10–50μs. Unlike many distributed scientific computations, typical datacenter applications can tolerate such small changes in latency. Though not shown, the additional mean latency with independent channel tuning is 75μs for the two datacenter application traces and 200μs for the uniform random workload.

A flattened butterfly network that always operated in the slowest and lowest power mode would consume 42% of the baseline power (or 6.1% assuming ideal channels). Our proposed energy proportional FBFLY comes within a few percent of that low power configuration in several cases. However, unlike all of the configurations above, ***a network that always operates in the slowest mode fails to keep up with the offered host load.***

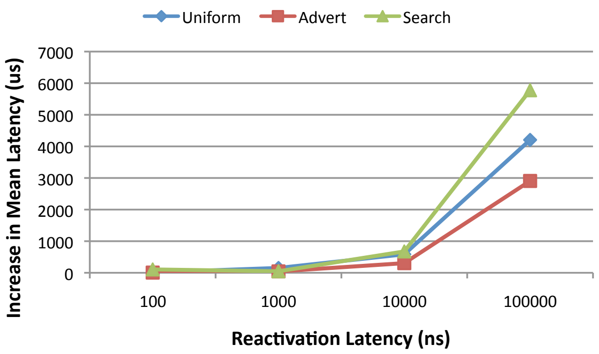
An ideally energy proportional FBFLY network would include ideal channels and zero reactivation time. When a link had something to transmit, it does so at full speed (and power), and when it does not have something to transmit, it consumes no power. In other words, the energy consumed by the network would exactly equal the average utilization of all links in the network. On the baseline system, our three workloads have an average utilization, and hence ideal power relative to the baseline, of 23% for *Uniform*, 6% for *Search* and 5% for *Advert*. Using our proposed heuristics, ideal independent channels, and a conservative reconfiguration time of 1μs, we can achieve relative power of 36%, 17%, and 15% of baseline, respectively— a remarkable feat considering the simplicity of our proposed heuristics.

4.2.2 Sensitivity to Target Link Utilization and Reconfiguration Latency

Target link utilization is an important factor when tuning link speeds. If set too low, then the network is missing out on opportunities to save power. If set too high, the network will be beyond saturation and performance will suffer. In Figure 9(a), we explore three values for target link utilization: 25, 50, and 75%. For the workload traces, the latency increases substantially more at 75% utilization than at 25%. Running with utilization much higher than 75% is not feasible for these workloads, because the network saturates. We don't show the power data when varying target utilization, though interestingly, there is very little difference for any of the workloads when using reactivation times of 1μs or less. The reason is, again, that these workloads are bursty: either they are sending data, and in (or transitioning to) high-power mode, or not



(a) Additional mean latency for a range of target channel utilization. Based on $1\mu s$ reactivation time and paired links.



(b) Additional mean latency for a range of reactivation times. Based on 50% target utilization and paired links.

Figure 9: Sensitivity of target channel utilization and reactivation times

sending data and in (or transitioning to) the low-power mode. They don't spend much time in the intermediate modes when the reactivation times (and hence utilization measurement epoch) is small.

Finally, we report our results assuming a range of link reactivation penalties, from 100ns–100μs in Figure 9(b). We set the epoch at 10$\times$ the reactivation latency, which bounds the overhead of reactivation to 10%. Yet as the reactivation time rises to 10μs, the increase in mean network latency approaches 1ms. With a 100μs reactivation, latency rises to several additional milliseconds, an overhead that can impact many different types of applications. Clearly our proposed techniques only become feasible when the reactivation time is less than 10μs. Though we do not show a power graph, increasing the reactivation time (and hence utilization measurement epoch) does decrease the opportunity to save power. Especially for the *Uniform* workload, which is less bursty than the traces at larger timescales, the power savings completely disappear for 100μs. For the two traces at 50% utilization, the reduction in power savings for 100μs is only 2–5%. Fortunately actual link reactivation time of modern links can be much lower, less than our default value of 1μs, which shows significant power reductions.

The bottom line is that for an insignificant change in performance, an energy proportional flattened butterfly can deliver up to a 6.6$\times$ reduction in power. If we extrapolate this reduction to our full-scale network presented in Section 2.2, the potential additional four-year energy savings is \$2.5M.

5. DISCUSSION AND FUTURE WORK

Although the results in the previous section are impressive, there is still room for significant reductions in network power.

5.1 Better Heuristics

With the flexibility of the flattened butterfly topology, we were able to apply a simple but effective heuristic to change the performance and power of individual links. However there remains opportunity to do even better: there is still a 3$\times$ difference between our results and an ideally energy proportional network. With bursty workloads, it may be advantageous to immediately tune links to either their lowest or highest performance mode without going through the intermediate steps. A better algorithm might also take into account the difference in link resynchronization latency to account for whether the lane speed is changing, the number of lanes are changing, or both. Opportunities may also exist for more complex predictive models.

5.2 Dynamic Topologies

In this paper, we demonstrated energy proportional networking by dynamically changing the performance and power envelope of individual links. However we did not evaluate powering off a link for two reasons. First, we based our evaluation on the data from the switch chip characterized in Figure 5, which shows very little additional power savings in shutting off a link entirely. Second, powering off links fundamentally changes the topology and routing, which is not currently supported in our simulation environment. Yet a flattened butterfly is actually quite amenable to powering off links because routing decisions can be made locally. In contrast, powering off a link in the folded-Clos topology requires propagating routing changes throughout the entire network.

Though we do not evaluate it in detail, we wish to highlight a fertile area of future innovation: *dynamic topologies*, which could take advantage of future switch chips that offer a true power-off state. From a flattened butterfly, we can selectively disable links, thereby changing the topology to a more conventional mesh or torus. For example, we can disable links in the flattened butterfly topology to make it appear as a multidimensional *mesh*. As the offered demand increases, we can enable additional wrap-around links to create a *torus* with greater bisection bandwidth than the mesh at the expense of more power consumed by wrap-around links.[6] Additional links (which are cabled as part of the topology) are dynamically powered on as traffic intensity (offered load) increases. The notion of energy-proportional dynamic topology requires an energy-aware routing algorithm capable of placing new routes with live traffic. There are other subtleties to consider. For example, in a network with credit-based flow control, one direction of a link cannot operate without the other direction active in order to receive credits back.

5.3 Opportunities for Future Switch Chips

Independent of the wire protocols (e.g., ethernet, InfiniBand, proprietary, etc.) there are certain requirements that *enable* fine-grained energy proportional networks. Foremost is the ability to sense congestion at individual output links, and *dynamically* adapt around congestion that results from either hotspot traffic or power-optimized links. Adaptive routing techniques have been common

[6]Though constructing a torus with radix $k > 4$ requires additional virtual channels and awareness in the routing algorithm to avoid toroidal deadlocks.

to high-performance computing [13, 14, 22, 23] community. Our results indicate that using the same heuristics for adaptive routing can be applied to energy-proportional switching. The decision to change the data rate of a link can be made by hardware, firmware, or with an embedded processor as part of a managed switch.

Second, our results showed much more significant power-saving opportunities for links that are truly energy proportional. For example, a link configured for 2.5 Gb/s should ideally use only 6.25% the power of the link configured for 40 Gb/s. In contrast, Figure 5 shows that a switch chip today still consumes 42% the power when in the lower performance mode.

Third, we highlight the opportunity to allow individual channels, in a bidirectional link, to operate at different performance and power modes.

6. RELATED WORK

The area of large-scale interconnection networks has been a fertile area of research, however, the notion of *energy proportional* communication in large datacenter networks has remained an important and unexplored topic. Most of the research to date has focused on providing a more energy-efficient communication channel by optimizing the physical characteristics of the link [21].

Heller, et al., recently proposed ElasticTree [11] to save energy in datacenter networks. They do so by continuously monitoring datacenter traffic conditions at the system scale, and then determining which parts of the fabric can be powered off. Our work focuses on lower-power topologies and fine-grained techniques to reduce a link's power usage at the time-scale of nanoseconds or microseconds.

The ethernet ALR work [8] proposes similar ideas using ethernet, and evaluates them with an analytic model and simulation using synthetic traffic patterns. Our evaluation uses actual traces from a production datacenter. Since ethernet data rates vary by factors of 10 (e.g. 100 Mb/s, 1 Gb/s, 10 Gb/s) dynamic rate adapting typically requires link re-initialization, with reactivation latency in excess of 1ms, which adds an unacceptable delay for all the workloads evaluated.

Other approaches [18] have focused on the MPI run-time system to activate or deactivate links involved in the traffic pattern. This approach requires detailed knowledge of the traffic pattern *a priori*. Unfortunately, unlike high-performance computer (HPC) systems, datacenter networks run multiple workloads simultaneously, making the traffic pattern difficult or impossible to predict at the time of job scheduling. Our approach takes advantage of the congestion sensing heuristics used for adaptive routing and piggybacks on these techniques to sense traffic intensity, dynamically activating links as they are needed. Likewise, as the offered load is decreased, we sense the lower channel utilization and reduce the link speed to save power, similar to the approach of Mahadevan et al. [19] used for ethernet. However, rate adapting their ethernet channels takes 1–3 *seconds*. To tolerate several seconds of traffic disruption requires significant buffering and path diversity to mitigate performance impact during link rate adjustments.

The Hyper-X network, proposed by Ahn, et al., [1], also recognizes the benefits of a flattened butterfly topology for building highly scalable networks. They do not address power or energy efficiency in their work. Al-Fares, et al., [2] present a scheme to build a high-bandwidth datacenter network in a folded-Clos topology with commodity ethernet chips. They find that their network, composed of 1 Gb/s ethernet chips, uses considerably less power than a hierarchical network composed of high-end, power inefficient 10 Gb/s switches. Our topology comparison is independent of any particular switch chip. A switch with sufficient radix, routing, and congestion-sensing capabilities allows building a flattened butterfly topology, which uses half the number of switch chips of the folded-Clos. We then go even further by dynamically adjusting the power envelope of the network to increase its energy proportionality, an issue not tackled by Al-Fares et al. The VL2 datacenter network proposed by Greenberg, et al., [7] also uses the folded-Clos topology, as does Portland [20]. They do not address the issue of power.

7. CONCLUSIONS

As servers themselves become more energy proportional with respect to the computation that they are performing, the network becomes a significant fraction of cluster power. In this paper we propose several ways to build a cluster-scale network whose power consumption is more proportional to the amount of traffic it is transmitting.

As a first step, we compare the flattened butterfly and folded-Clos topologies and show that flattened butterfly alone is more energy-efficient resulting in a lower operating expense. Second, we show that the dynamic range for a commercial switch chip, which provides nearly 60% power savings compared to full utilization, can be used to dynamically reduce power. Detailed event-driven simulations of both synthetic workloads and production datacenter traces are used to characterize and understand design tradeoffs. By operating the network links at a data rate in proportion to the offered traffic intensity, an *energy proportional* datacenter network can further improve on the energy savings offered by the flattened butterfly topology. Modern switch chips capable of congestion sensing and adaptive routing already have the essential ingredients to make energy-proportional communication viable for large-scale clusters. Together, we show that our proposals can reduce operating expenses for a large-scale cluster by up to $3M over a four-year lifetime.

Our results also demonstrate two challenges for the designers of future switches: 1) We show that there is significant advantage to having independent control of each unidirectional channel comprising a network link, since these channels typically see asymmetric use, and 2) ideally, high-speed channel designs will evolve to be more energy proportional themselves, i.e., a link operating at 2.5 Gb/s should consume proportionately less power than a link operating at 40 Gb/s.

Acknowledgements

We are grateful to Urs Hölzle, Luiz Barroso, Chris Johnson, Bob Felderman, and Jim Laudon for supporting this work. We thank Florentina Popovici, Dan Gibson, and the anonymous reviewers for helpful feedback.

8. REFERENCES

[1] Jung Ho Ahn, Nathan Binkert, Al Davis, Moray McLaren, and Robert S. Schreiber. HyperX: topology, routing, and packaging of efficient large-scale networks. In *SC '09: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, pages 1–11. ACM, 2009.

[2] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A scalable, commodity data center network architecture. In *SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, 2008.

[3] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[4] Luiz André Barroso and Urs Hölzle. *The Datacenter as a Computer: An Introduction to Design of Warehouse-scale Machines*. Morgan Claypool, 2009.

[5] Borthakur, Dhruba. The Hadoop distributed file system: Architecture and design. `http://hadoop.apache.org/common/docs/r0.18.0/hdfs_design.pdf`.

[6] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The Google file system. In *19th ACM Symposium on Operating System Principles*, 2003.

[7] Albert Greenberg, James R. Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A. Maltz, Parveen Patel, and Sudipta Sengupta. Vl2: a scalable and flexible data center network. In *SIGCOMM '09: Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, pages 51–62, 2009.

[8] Chamara Gunaratne, Kenneth Christensen, Bruce Nordman, and Stephen Suen. Reducing the energy consumption of ethernet with Adaptive Link Rate (ALR). *IEEE Trans. Comput.*, 57:448–461, April 2008.

[9] Chuanxiong Guo, Guohan Lu, Dan Li, Haitao Wu, Xuan Zhang, Yunfeng Shi, Chen Tian, Yongguang Zhang, and Songwu Lu. BCube: A high performance, server-centric network architecture for modular data centers. In *SIGCOMM '09: Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, pages 63–74, 2009.

[10] H. Hatamkhani and Yang C-K.K. A study of the optimal data rate for minimum power of I/Os. In *IEEE Transactions on Circuits and Systems*, pages 1230–1234, 2006.

[11] Brandon Heller, Srini Seetharaman, Priya Mahadevan, Yiannis Yiakoumis, Puneet Sharma, Sujata Banerjee, and Nick McKeown. Elastictree: Saving energy in data cneter networks. In *Proceedings of the 7th USENIX Symposium on Networked System Design and Implementation (NSDI)*, pages 249–264. ACM, 2010.

[12] Google Inc. Efficient computingâ step 2: efficient datacenters. `http://www.google.com/corporate/green/datacenters/step2.html`.

[13] Nan Jiang, John Kim, and William J. Dally. Indirect adaptive routing on large scale interconnection networks. In *ISCA '09: Proceedings of the 36th annual International Symposium on Computer Architecture*, pages 220–231, 2009.

[14] John Kim, William J. Dally, and Dennis Abts. Adaptive routing in high-radix clos network. In *SC '06: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, page 92, 2006.

[15] John Kim, William J. Dally, and Dennis Abts. Flattened butterfly: a cost-efficient topology for high-radix networks. In *ISCA '07: Proceedings of the 34th annual International Symposium on Computer Architecture*, pages 126–137, 2007.

[16] John Kim, William J. Dally, Brian Towles, and Amit K. Gupta. Microarchitecture of a high-radix router. In *ISCA '05: Proceedings of the 32nd Annual International Symposium on Computer Architecture*, pages 420–431, 2005.

[17] Charles E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Trans. Comput.*, 34(10):892–901, 1985.

[18] Jian Li, Lixin Zhang, Charles Lefurgy, Richard Treumann, and Wolfgang E. Denzel. Thrifty interconnection network for hpc systems. In *ICS '09: Proceedings of the 23rd International Conference on Supercomputing*, pages 505–506, 2009.

[19] Priya Mahadevan, Puneet Sharma, Sujata Banerjee, and Parthasarathy Ranganathan. Energy aware network operations. In *INFOCOM'09: Proceedings of the 28th IEEE International Conference on Computer Communications Workshops*, pages 25–30, 2009.

[20] Radhika Niranjan Mysore, Andreas Pamboris, Nathan Farrington, Nelson Huang, Pardis Miri, Sivasankar Radhakrishnan, Vikram Subramanya, and Amin Vahdat. Portland: a scalable fault-tolerant layer 2 data center network fabric. *SIGCOMM Comput. Commun. Rev.*, 39(4):39–50, 2009.

[21] Rambus Corporation. `http://www.rambus.com/us/patents/innovations/detail/low_power_multi_gbps.html`.

[22] S. Scott and G. Thorson. The Cray T3E Network: Adaptive Routing in a High Performance 3D Torus. In *Hot Interconnects 4*, Aug. 1996.

[23] Steve Scott, Dennis Abts, John Kim, and William J. Dally. The blackwidow high-radix clos network. In *ISCA '06: Proceedings of the 33rd annual International Symposium on Computer Architecture*, pages 16–28, 2006.

[24] Sun Microsystems. Lustre file system. `http://www.sun.com/software/products/lustre/`.

[25] U.S. Department of Energy. Average retail price of electricity. `http://www.eia.doe.gov/cneaf/electricity/epm/table5_6_a.html`.

[26] U.S. Environmental Protection Agency. Report to congress on server and datacenter energy efficiency. Public Law 109-431, August 2, 2007.