The Importance of Switch Dimension for Energy-Efficient Datacenter Networks

Indra Widjaja[a], Anwar Walid[a], Yanbin Luo[b], Yang Xu[b,*], H. Jonathan Chao[b]

[a]*Bell Labs, Alcatel-Lucent, 600 Mountain Ave, Murray Hill, NJ 07974, USA*
[b]*Polytechnic Institute of New York University, 2 MetroTech Center, Brooklyn, NY 11201, USA*

Abstract

Saving power in datacenter networks has become a pressing issue as the networks tend to consume steady power even when many servers may be idle during periods of low activity. While in operation, ElasticTree and CARPO can save power consumed by a fat-tree network by using Sleep Mode where some components such as ports and switches are turned off when traffic demand in the network is relatively moderate. In this paper, we propose a new approach by exploring the design stage of a datacenter network and focus on how to choose the right switch size that can potentially save the most power during the expected operation of the network. We also consider Speed Scaling where the power of a switch can be varied by adjusting its processing rate according to its traffic demand. We first perform analysis and simulation to investigate the power-saving performance of different switch sizes, power-saving modes and traffic demand patterns. Based on fat-tree networks with sleep mode and supporting a fixed number of servers, our findings reveal that deploying a large number of small switches is more power-efficient than a small number of large switches when the traffic demand is relatively moderate or when servers exchanging traffic are in close proximity. With speed scaling, the reverse is generally true. We confirm our findings using traffic traces from a production data center. Our approach for studying the fat-tree topology design is also extendible to other types of datacenter network topologies.

Keywords: Datacenter Networks; Power Saving; Switch Design; Power Modeling

1. Introduction

High energy consumption has become a serious concern in the design of large-scale enterprise data centers which aim to provide reliable and scalable computing infrastructure for massive Internet services. In addition to high electricity bills and negative environmental implications, increased power consumption may lead to system failures, as data centers increasingly deploy new high-density servers, while their power distribution and cooling systems are approaching their peak capacity. Energy proportional computing [1] has emerged as a new paradigm for the design and operations of datacenter servers where the energy consumption is made to scale with the CPU speed using dynamic voltage and frequency scaling (DVFS) [2] (also known as speed-scaling). More recently, there is new awareness and efforts in tackling energy consumption at the datacenter communication network, which consists of the switches and the links that interconnect the servers [3], [4], [5]. While the typical share of energy consumption of the datacenter communication network is only 10-20% [4], this share is expected to increase as the servers' share of energy consumption decreases with the maturing of energy efficient computing, and as the required network bisection bandwidth increases to handle increasing communication demands [5].

In this paper our goal is to enable energy-proportional datacenter communication networks. In particular, we are interested in making the amount of energy consumed proportional to the traffic intensity (offered load) in the network. Prior work [3] and [4] focused on developing algorithms for dynamically adjusting the set of active links and switches in a particular datacenter topology, namely the fat-tree topology, to satisfy changing datacenter traffic loads. Our contributions center on developing fundamental insights into key structural and scaling properties of the datacenter network that facilitate energy-proportional communications and on how the current and future technologies impact and modify these properties. Such insights are useful in guiding the design and deployment of future datacenter topology designs and analysis of different competing alternatives. Importantly, the findings transcend the particular choice of the interconnections topology, i.e. fat-tree, flattened butterfly [5], Hypercube, de Bruijn [6], etc., and shed light on the relative number of switches and their sizes that are optimal for given energy-efficient technology. Our results are derived based on formulation of a network design optimization problem whose solution provides the optimal size and topology of the communication network that supports certain number of servers and their traffic loads. Other important elements in the design optimization are the power consumption models of the ports and switches and the level of safety margin (additional capacity beyond normal levels) to handle unpredictable traffic surges.

Recent measurement studies of switch power consumption [7] show that turning on the switch consumes most of the pow-

*Corresponding author

Email addresses: iwidjaja@research.bell-labs.com (Indra Widjaja), anwar@research.bell-labs.com (Anwar Walid), yluo04@students.poly.edu (Yanbin Luo), yangxu@poly.edu (Yang Xu), chao@poly.edu (H. Jonathan Chao)

er; going from zero to full rate increases power by less than 8%. Therefore, with today's technology, our best option for saving energy in the network is to manage the non energy-proportional network components intelligently. In particular, a switch or port is opportunistically turned off, referred to here as "sleep mode" operation, during periods of low traffic demands to achieve most of the power-saving benefits. Thus, a network of non-proportional components can act as a load-proportional ensemble. There are design choices for building a communication network to support certain number of servers, where each design choice specifies the number of switches, their sizes and their interconnection topology. With the sleep mode operation, we show that a topology with many small switches is more energy-efficient than a topology with few large switches when servers are communicating in close proximity or when the traffic demand is low.

The advances made in making energy consumption of servers load-proportional using speed-scaling will likely help improve the energy dynamic range of the switches and ports, and make inter-server communication energy cost more proportional to the amount of data being transmitted. [5] gives a proposal for dynamically tuning individual plesiochronous links to match the required load while consuming as little power as possible. [5] also highlights opportunities for future energy-proportional switch chips. As Ethernet switches typically come in different sizes, it is natural to ask whether a specific switch size can optimize a given objective function during the operation of a data center. In this paper, we use energy as the objective function we want to minimize. Our study reveals that when switches operate in speed-scaling mode, the optimal topology favors few large switches, which is contrary to our findings with the sleep-mode operation.

This paper is an extension of our previous paper [8]. In particular, we conducted more simulations to test the packet delay and loss rate performance when power-saving features of the data center network are enabled. The results show that the impact of traffic consolidation to the performance is minor when a proper safety margin is set. Besides the synthetic traffic used in [8], a real-life datacenter traffic trace is used in this extended version to test the datacenter network.

The contributions of the paper can be summarized as follows.

1. We developed two power consumption models for fat-tree datacenter networks with two different power saving techniques (i.e., sleep mode and speed-scaling mode). The developed power models will potentially benefit future datacenter research by providing a methodology for calculating the datacenter power consumption. Our approach for studying the fat-tree topology design is also extendible to other types of datacenter network topologies.
2. We evaluated the power models using both synthetic and real-life datacenter traffic. The results confirm the accuracy of the power models.
3. Based on the power models, we developed the fundamental insights into the key structural and scaling properties of datacenter networks that facilitate energy-proportional communications and studied how the current and future technologies impact and modify these properties.

In the next section, we summarize the related work. Sec. 3 provides the preliminaries on generalized fat-trees. We then present power consumption analysis with sleep mode in Sec. 4 followed by speed-scaling mode in Sec. 5. Experiments with packet and flow models using both synthetic traffic and real traces are investigated in Sec. 6. We also evaluate the impact of energy-saving mechanisms on quality-of-service measures such as packet loss rate and packet delay. Finally, we conclude the paper in Sec. 7.

2. Related Work

Several recent researches have studied the issue of power consumption of network devices in the data center, and proposed schemes to achieve energy proportionality in Data Center Networks (DCNs). ElasticTree [3] proposes to dynamically turn on or off switches and links based on time-varying traffic of DCN to achieve a minimum-power network subnet. The traffic flows are consolidated onto as few routes as possible by flow rerouting in the minimum-power network. The consolidation approach of ElasticTree is developed based on a real data center trace, in which the rate of each flow was assumed to be constant during the consolidation phase. CARPO [4] observed that bandwidth demands of most flows do not peak at the same time in real DCNs, and proposed a correlation-aware traffic consolidation scheme that consolidates flows with low correlation together to save more power.

Some other researches proposed schemes to adapt the link rate according to the workload on each link [5] [9] [10]. Compared to the schemes using traffic consolidation, the schemes using link rate adaptation can only achieve small power saving due to the fact that the majority amount of power of a DCN is consumed by components such as fans and switch fabrics.

3. Preliminaries

We focus on fat-tree networks as they are well-known and widely deployed in data centers. Our methodology, however, is also applicable to other networks. A fat-tree maintains constant bisection bandwidth as one traverses from the switches at the bottom of the tree to the switch at the root [11]. Typically, fat-tree networks for data centers are constructed with two or three levels of switches. In practice, the network is built by replicating constant-degree switches at each level to support the required bandwidth [12]. At the bottom, half of the ports of a switch are used to connect to servers (or racks of servers) and the other half to other switches. At other levels, all ports are used to connect to other switches.

Prior work on fat-tree datacenter networks only considers a configuration with 3 levels of switches [3][4]. In [12], it is shown that a fat-tree network using k-port switches with 3 levels requires $5k^2/4$ switches and can support $k^3/4$ servers (or racks). In this section, we describe the construction of a fat-tree

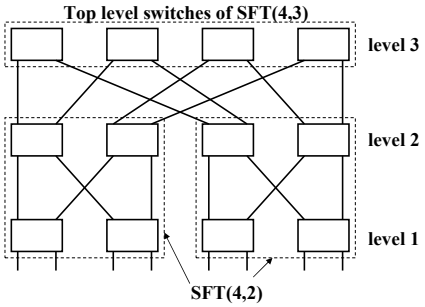

Figure 1: SFT$(4, 3)$ constructed from two SFT$(4, 2)$'s.

network with arbitrary number of levels. This allows us to pose a deeper power-consumption problem at the *design stage* where one can choose the right topology (e.g., for a fat-tree, what is the optimal number of levels?) with the right switch size (e.g., for a fat-tree, what is the value of k?). At the *operation stage*, we also address speed-scaling mode while prior work only deals with sleep mode.

To describe the construction of a fat-tree with an arbitrary number of levels, we adopt some notations in [13][14]. Let FT(k, n) denote a fat-tree constructed by n levels of k-port switches, where k is a multiple of 2. We call level-n switches for those at the top and level-1 switches for those that connect to servers at the bottom. A fat-tree can be constructed by connecting each of the top-level (level-n) switches of FT(k, n) with k sub-fat-trees SFT$(k, n-1)$'s that have $n-1$ levels of switches. FT and SFT are different in that the top-level switches of SFT must provide up-links while those of FT do not. In other words, each switch of SFT has $k/2$ up-links and $k/2$ down-links, while each switch at level n has k down-links and no up-links.[1] In general, SFT(k, l), $1 < l < n$, can be recursively constructed by connecting each of the top-level (level-l) switches of SFT(k, l) with $k/2$ SFT$(k, l-1)$'s. At level 2, each of the top-level switches of SFT$(k, 2)$ is connected to $k/2$ SFT$(k, 1)$'s, where each SFT$(k, 1)$ is just a single switch with $k/2$ down-links connecting to servers. Fig. 1 shows an example of SFT$(4, 3)$.

To see the inter-connections, note that each of the $k/2$ SFT$(k, 1)$'s has $k/2$ up-links; each of the up-links connects to one of the $k/2$ top-level switches of SFT$(k, 2)$. Thus, the total number of up-links from $k/2$ SFT$(k, 1)$'s to $k/2$ top-level switches of SFT$(k, 2)$ is $(k/2)^2$. Generally, it can be easily seen by induction that the total number of up-links from $k/2$ SFT$(k, l-1)$'s to $(k/2)^{l-1}$ top-level switches of SFT(k, l) is $(k/2)^l$, for $1 < l < n$. Since there are k SFT$(k, n-1)$, the total number of up-links from k SFT$(k, n-1)$'s to the top-level switches of FT(k, n) is $2(k/2)^n$. Thus, the number of switches at level n is $2(k/2)^n/k = (k/2)^{n-1}$. For $l \neq n$, the number of switches at each level is $2(k/2)^{n-1}$. Thus, the total number of switches in FT(k, n) is $(2n-1)(k/2)^{n-1}$ and the total number of servers (racks) supported is $N_h = 2(k/2)^n$. Note that for a given number of servers N_h, one can choose different pairs of (k, n) to support the servers. Generally, it may not be possible to find the pairs that give the same value $2(k/2)^n$. In this case, some sub-fat-trees SFT$(k, \cdot)$'s of FT(k, n) can be removed if the number of servers is less than those that can be supported by the 'full' fat-tree.

[1] In practice, a few up-links are needed to connect the level-n switches through routers to the Internet.

4. Power Consumption with Sleep Mode

4.1. Formulation for General Datacenter Networks

We assume that the power consumption of a datacenter network depends on the switch power and link power. We use the term sleep mode to indicate that a given network component can be turned off when there is no traffic through it. The optimization problem for minimizing power consumed by the network can be formulated as an integer linear program. Formally, let a datacenter network be represented as a graph $G = (N, L)$, where N is the set of switches and L is the set of unidirectional links. Let M be the set of server-to-server unidirectional traffic with each flow $m \in M$ of rate d^m entering an ingress switch s^m and exiting an egress switch t^m. If some of the traffic m is routed through the link from switch i to switch j, we let x_{ij}^m represent the bandwidth consumed on the link (i, j) by the traffic. We assume the capacity of the link (i, j) is C_{ij}. Usually, $C_{ij} = C_{ji}, \forall i, j$. Let $X_{ij}, \forall (i, j) \in L$, denote a binary variable that is equal to 1 if link (i, j) is enabled and 0 otherwise. In practice, $X_{ij} = X_{ji}$. Also, let Z_i denote a binary variable that is equal to 1 if switch i is enabled and 0 otherwise, $\forall i \in N$. To quantify the power consumption, we further let P_i^s denote the power of switch $i \in N$. We also let $P_{i,j}^l$ denote the power needed to enable unidirectional link (i, j). Here, a switch includes switching fabrics, line cards and possibly other modules, but excludes link.

For a given traffic demand matrix, the optimization problem that minimizes power consumption can be formulated as follows.

$$\text{minimize} \sum_{(i,j)\in L} P_{i,j}^l X_{ij} + \sum_{i \in N} P_i^s Zi \tag{1}$$

subject to

$$\sum_{j\in N:(i,j)\in L} x_{ij}^m - \sum_{j\in N:(j,i)\in L} x_{ji}^m = \begin{cases} d^m, & i = s^m, m \in M \\ -d^m, & i = t^m, m \in M \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$\sum_{m\in M} x_{ij}^m \leq C_{ij}, \quad \forall (i, j) \in L. \tag{3}$$

$$X_{ij} = X_{ji}, \quad \forall (i, j) \in L. \tag{4}$$

$$\sum_{m\in M} x_{ij}^m / C_{ij} \leq X_{ij}, \quad \forall (i, j) \in L. \tag{5}$$

$$X_{ij} \leq Z_i, \quad \forall (i, j) \in L. \tag{6}$$

$$Z_i \leq \sum_{m\in M, j\in N:(i,j)\in L} x_{ij}^m, \quad \forall i \in N. \tag{7}$$

Eq. (1) defines the objective function to be minimized. Eq. (2) ensures flow conservation. While Eq. (3) represents the

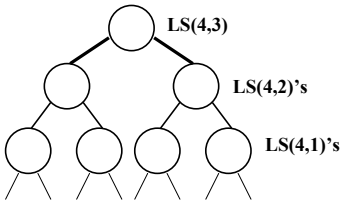

Figure 2: A representation of Fig. 1 in terms of logical switches.

bandwidth constraint on the link, Eq. (4) states the natural bi-directional property of the activity of a link. Eq. (5) ensures that a link is enabled when there is a non-zero flow through the link, while Eq. (6) ensures that a switch is enabled when one of its links is enabled. Finally, Eq. (7) ensures that a switch is deactivated when there is no flow through it.

The above formulation has been implemented in AMPL and solved by CPLEX. Unfortunately, the problem is NP-complete for integer flows, which leads to long runtime for datacenter networks with realistic sizes. We thus explore an alternative method in the following subsection.

4.2. Efficient Method for Fat-Tree Networks

4.2.1. Power Consumption Analysis

The formulation in the previous subsection allows for different switches to have different number of ports, different links to have different capacities, and the topology to be arbitrary. On the other hand, a fat-tree topology exhibits high regularity. In particular, links typically have the same capacity, switches have the same size, and the topology is regular. We take advantage of the regularity of a fat-tree network to decide whether to turn on/off switches or ports[2]. The following approach gives the exact same results for power consumption as the approach in the previous subsection, but comes with much less computational burden.

From the description in Sec. 3, note that each of the top-level switches of SFT(k, l) (or FT(k, n) at level n) has a link to each of the SFT($k, l-1$)'s. This observation allows us to view the top-level switches of a given SFT(k, l) as *one logical switch*, LS(k, l), consisting of multiple switches. Each logical switch LS(k, l) can continue to maintain the connectivity to all its children LS($k, l-1$)'s as long as at least one switch in LS(k, l) remains on. The number of switches that can be turned off depends on the load of the logical switch. Fig. 2 shows the simplified representation of Fig. 1 in terms of logical switches.

Let the capacity of each k-port switch be denoted by $C_s(k)$= $k\,C_p$, where C_p is the port/link capacity. Since the number of switches per logical switch at level l is $(k/2)^{l-1}$, the maximum capacity of a logical switch is

$$C_{LS}(k, l) = (k/2)^{l-1} C_s(k), \tag{8}$$

when all switches and ports are turned on. Let the aggregate traffic rate from a given LS(k, l) to its parent LS($k, l+1$) be denoted by $R^u(k, l)$ and the aggregate traffic rate from the parent LS($k, l+1$) to the LS(k, l) be denoted by $R^d(k, l)$. Then, $R(k, l) = \max\{R^u(k, l), R^d(k, l)\}$ is the aggregate traffic that needs to be supported by the *up-link* ports of LS(k, l) or the corresponding SFT(k, l). The number of up-link ports that are to be turned on at LS(k, l) for minimum power consumption is

$$N_p^u(R, k, l) = \max\{\lceil R(k, l)/C_p \rceil, 1\}, \forall\, l < n,$$

with at least one port turned on to maintain connectivity to all servers. The rest of the up-link ports of LS(k, l) can be turned off and the corresponding *down-link* ports at the parent LS($k, l+1$) are also turned off. As the parent LS($k, l+1$) has multiple children, the parent will turn off its corresponding down-link ports for each child. Let $N_p^d(R, k, l+1)$ denote the sum of all down-link ports that are to be turned on at a parent LS($k, l+1$) to incur minimum power consumption. At each level l, the number of switches at logical switch LS(k, l) that are to be turned on is

$$N_s(R, k, l) = \begin{cases} \max\{\lceil N_p^d(R, k, l)/k \rceil, 1\}, & l = n, \\ \max\{\lceil N_p^d(R, k, l)/(k/2) \rceil, 1\}, & 1 < l < n, \\ 1, & l = 1, \end{cases}$$

where we assume that at least one switch per logical switch is turned on to maintain connectivity.

The calculation for power consumption with sleep mode is illustrated in Algorithm 1. The computational complexity is $O((k/2)^n)$, which is extremely low as it is linear in the number of servers. In the algorithm, we assume that the switches and their down-link ports at level-1 are always turned on to maintain connectivity to all servers.

Algorithm 1 Power consumption with sleep mode

```
1: INPUT: k, n and traffic demand matrix.
2: OUTPUT: Power consumption.
3: Construct a tree with each node representing a logical switch.
4: Route all the traffic demand.
5: for each level l from 1 to n
6:   for each LS(k, l) from 0 to 2(k/2)^{n-l} - 1
7:     Retrieve saved result from children for N_p^d(R, k, l), if l > 1.
8:     Compute N_s(R, k, l), if l > 1.
9:     Turn off unused down-links and switches, if l > 1.
10:    Compute N_p^u(R, k, l).
11:    Turn off unused up-links.
12:    Update the corresponding N_p^d(R, k, l+1) at LS(k, l)'s parent.
13:      and save the result for steps 7-9, if l < n.
14:  end for
15: end for
16: Compute power based on the number of enabled switches and links.
```

Note that the above computation of $N_p^u(R, k, l)$ is generally unachievable if the traffic from SFT(k, l) is spread over all links of SFT($k, l+1$). However, $N_p^u(R, k, l)$ is achievable if the traffic is packed to the minimum number of up-link ports that can support the aggregate traffic. This can be done, for example, if each SFT(k, l) with its up-link ports numbered from 0 to $(k/2)^l - 1$, sends its up-link traffic to SFT($k, l+1$) packed to the left part from port 0 to port j_l^{max}, where $j_l^{max} < (k/2)^l$. To have efficient packing where each up-link port from 0 to j_l^{max} will have nearly equal load, we adopt an approach where

[2] A port is equivalent to a pair of unidirectional links at each of the two endpoints.

each switch applies Valiant Load Balancing (VLB) among the active ports. VLB has been effectively applied in VL2 when each server typically sends and receives many concurrent packet flows [15]. If each of SFT(k, l)'s, $l = 1, \cdots, n-1$, packs its up-link traffic to its left-most ports and the interconnections to SFT$(k, l+1)$ form a perfect-shuffle [16], then the traffic will be packed to the left most switches of SFT$(k, l+1)$; among those that carry traffic, the traffic will also be packed to the left most down-link ports. Although the above computation assumes perfect packing, in practice there may be slight load imbalances among different active ports because flows may have varying rates.

4.2.2. Power-Consumption Evaluation

We evaluate fat-tree networks for different configurations using Algorithm 1. For computing power consumption, we categorize a switch into two components: chassis and ports. The chassis includes the switching fabric and other modules such as line cards, processing modules, etc. It has been shown that the power consumption of a multi-stage switching fabric scales according to $O(k \log k)$, while a crossbar-type switching fabric scales according to $O(k^2)$ [17], where k is the number of ports. For other modules such as line cards, it is reasonable to assume that the power consumption scales according to $O(k)$. Suppose there is a choice of several switch sizes where the smallest one is equipped with k_{min} ports. Then, the chassis power of a k-port switch, when the switch is turned on, can be expressed as

$$P^{s(k)} = \eta_1(k/k_{min}) + \eta_2(k \log(k))/(k_{min} \log(k_{min}), \quad (9)$$

and the chassis power is zero if the switch is turned off. The first term represents the scaling due to other modules and the second term represents the scaling due to a switching fabric. Note that we take a more conservative power model for the switching fabric.

In the following, we assume that the parameter values in Eq. (9) are $\eta_1 = 50$W and $\eta_2 = 50$W. In addition, each switch port has a capacity $C_p = 10$ Gbps and consumes 2W when it is turned on (active). A switch port consumes zero power if it is turned off. We observe that the general qualitative results (i.e., advantages of using small switches versus large switches) do not change when we experimented with other combinations of the above parameter values.

For the traffic patterns, we decided to adopt those described in previous work ([3][4]) to maintain consistent qualitative results. We first consider the case where each server i, for $i = 0, \cdots, N_h - 1$, sends traffic to a fixed server j at a rate given by

$$\lambda_{i,j} = \begin{cases} d, & \text{if } j = (i+z) \bmod N_h \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

We set $z = 1$ for *near* traffic and $z = N_h/2$ for *far* traffic in Eq. (10). A mixture of the two consists of far traffic of rate αd and near traffic of rate $(1-\alpha)d$, where α is the percentage of far traffic over the total traffic.

Fig. 3 compares the powers consumed by two different fat-tree configurations that support the same number servers $N_h = 8192$: FT$(16, 4)$ and FT$(128, 2)$. While FT$(16, 4)$ needs 3584 small switches (16-port), FT$(128, 2)$ can be deployed with 192 large switches (128-port). As can be seen from the figure, if $\alpha = 0$ (all near traffic), the power consumed with small switches is always less than that consumed with large switches. This is because only a low percentage of small switches need to be turned on and these small switches consume less power than the large ones. If $\alpha = 0.3$ (30% far traffic), small switches still consume less power than large switches when the traffic load (d/C_p) is below 0.35 approximately. Beyond that, many more small switches need to be turned on and the power consumption with small switches becomes larger than that with large switches.

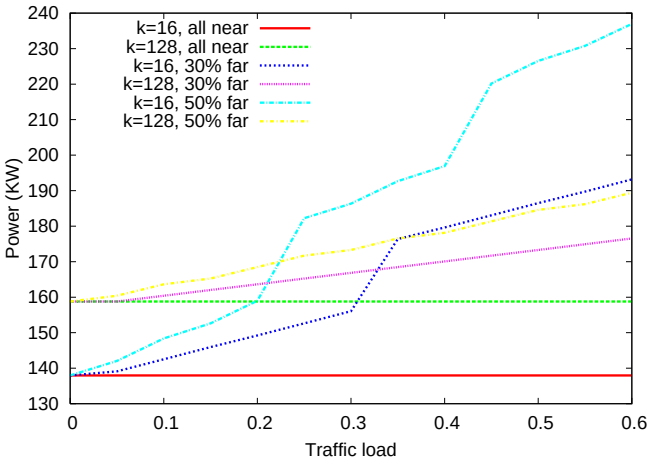

Figure 3: Power consumption with sleep mode under near-far traffic.

The main observation is that when the far traffic is dominant *and* the traffic load is relatively high, it is generally more power-efficient to use large switches than small switches. Interestingly, even if far traffic is dominant, small switches can be more power-efficient if the traffic load is low. This situation, for example, can be applicable for a data center that experiences low traffic load most of the time with possible surges of traffic load occurring less frequently during busy hours. In addition, if most communicating servers can be localized and results in mostly near traffic, then deploying small switches is also more advantageous than large switches *independent* of the traffic load.

One may note from Fig. 3 that the power saving using small switches may first seem small (up to 12%) compared to that using large switches. If one notes that the maximum power saving using large switches in this case is about 40%, the additional saving using small switches is actually appreciable. A disadvantage of using small switches is that it incurs additional hops, which translates to additional delay.

We next consider another traffic pattern where each server i, for $i = 0, \cdots, N_h - 1$, sends traffic to another server at a random location j. In particular, we assume that server i chooses server j according to the following geometric distribution:

$$Pr\{j = (i+z) \bmod N_h\} = (1-p)^{z-1}p, \; z = 1, 2, \cdots. \quad (11)$$

The average distance between two communicating servers is approximately $1/p$ when N_h is large, but can be considerably

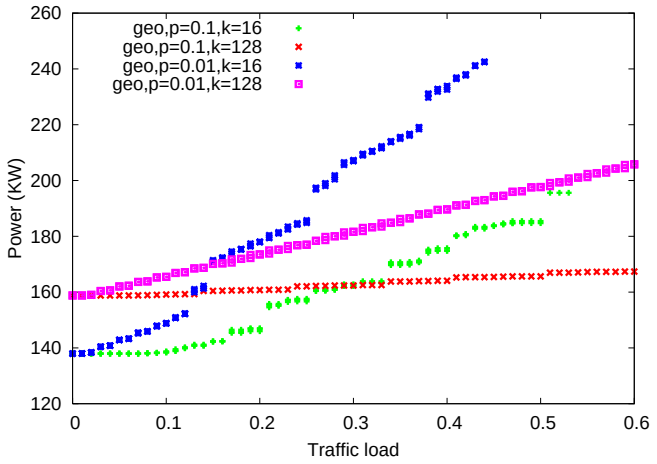

Figure 4: Power consumption with sleep mode under geometric traffic.

smaller if N_h is very small due to the modulo operation.

Fig. 4 shows the network power consumption under geometric traffic pattern, computed using Algorithm 1. Again, we observe that fat-tree with small switches consumes less power than that with large switches when the traffic load is moderate. When the traffic patterns are more localized (larger values of p), FT$(16, 4)$ can be more power-efficient than FT$(128, 2)$ over a wider range of traffic load. We have also experimented with much larger datacenter networks, and found that the tradeoffs remain consistent.

5. Power Consumption with Speed-Scaling Mode

5.1. Formulation for General Datacenter Networks

Another approach to reduce power consumption is to vary the clock frequency (speed) of some components in the network. This approach is typically called *speed scaling* or rate adaptation [9], where the frequency at which a network component (e.g., a switch) operates can be increased or decreased to match the offered load. The power consumption of a switch i is modeled as an affine function to reflect power proportionality [9] and is given by

$$P_i^s(x) = a_i x + b_i, \tag{12}$$

where a_i and b_i are some constants and x is the traffic load at the switch. Since a link/port generally cannot be subjected to speed scaling, we assume that a link can only be turned on or off. For example, Ethernet can only adjust its link speed at 10, 100, or 1,000 Mbps, but not at higher rate that is usually used in latest deployment. The optimization problem with speed scaling can be formulated as follows.

$$\text{minimize} \sum_{(i,j)\in L} P_{i,j}^l X_{ij} + \sum_{i\in N,(j,i)\in L, m\in M} P_i^s(x_{j,i}^m + d^m) \tag{13}$$

subject to

$$\sum_{j\in N:(i,j)\in L} x_{ij}^m - \sum_{j\in N:(j,i)\in L} x_{ji}^m = \begin{cases} d^m, & i = s^m, m \in M \\ -d^m, & i = t^m, m \in M \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

$$\sum_{m\in M} x_{ij}^m \le C_{ij}, \quad \forall (i,j) \in L. \tag{15}$$

$$X_{ij} = X_{ji}, \quad \forall (i,j) \in L. \tag{16}$$

$$\sum_{m\in M} x_{ij}^m / C_{ij} \le X_{ij}, \quad \forall (i,j) \in L. \tag{17}$$

The objective function in Eq. (13) consists of link power using sleep mode and switch power using speed-scaling mode. The interpretations of Eq. (14)-Eq. (17) are the same as before. As the above formulation is also impractical for reasonably sized networks, we will present a method that takes advantage of the regularity of a fat-tree network.

5.2. Efficient Method for Fat-Tree Networks

5.2.1. Power Consumption Analysis

The computation of the network power consumption with speed scaling is similar to that with sleep mode. One primary difference is that switches employing speed scaling are never turned off. Suppose the power of a switch i receiving traffic at rate λ_i is given by $P_i^s(\lambda_i) = a_i\lambda_i + b_i$. If switches are within a logical switch LS(k, l), then $a_i = a$ and $b_i = b, \forall i \in LS(k, l)$, as these switches are of the same type. The power consumed by a logical switch LS(k, l) that is carrying traffic at rate λ_i at each switch i within the logical switch is given by

$$P^{LS} = \sum_{i\in LS(k,l)} (a_i\lambda_i + b_i) = \sum_{i\in LS(k,l)} (a\lambda_i + b) = a\lambda + bN_{LS},$$

where the aggregate traffic rate is $\lambda = \sum_{i\in LS(k,l)} \lambda_i$ and the number of switches in a logical switch is N_{LS}. The above implies that the power consumed by a logical switch is invariant to how the aggregate traffic rate is distributed among the switches within a given logical switch as long as no switch is overloaded. As a result, the computation for power consumption at a logical-switch level is simplified as we only need to consider the aggregate traffic at a logical switch (but not at the individual switch). Nevertheless, traffic packing with speed scaling is still beneficial as it will minimize the number of ports that need to be turned on.

In addition to considering the traffic between LS(k, l) and LS$(k, l+1)$ (i.e., $R^u(k, l)$ and $R^d(k, l)$) as described in Sec. 4.2.1, we also need to consider the traffic from each source LS$(k, l-1)$ to each destination LS$(k, l-1)$ that passes through a given LS(k, l) in order to account *all traffic entering* LS(k, l). Let the aggregate rate of all traffic entering LS(k, l) be denoted by $\check{R}(k, l)$. We summarize the procedure for implementing speed-scaling mode in Algorithm 2.

Algorithm 2 Power consumption with speed-scaling mode

1: INPUT: k, n and traffic demand matrix.
2: OUTPUT: Power consumption.
3: Construct a tree with each node representing a logical switch.
4: Route all the traffic demand.
5: for each level l from 1 to n
6: for each LS(k, l) from 0 to $2(k/2)^{n-l} - 1$
7: Retrieve saved result from children for $N_p^d(R, k, l)$, if $l > 1$.
8: Turn off unused down-links, if $l > 1$.
9: Compute P^{LS} (power at LS(k, l)).
10: Compute $N_p^u(R, k, l)$.
11: Turn off unused up-links.
12: Update the corresponding $N_p^d(R, k, l+1)$ at LS(k, l)'s parent.
13: and save the result for steps 7-8, if $l < n$.
14: end for
15: end for
16: Compute total power from the switches and the number of enabled links.

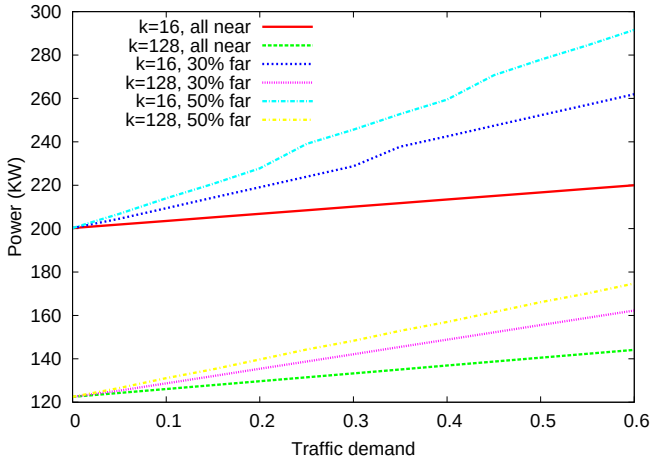

Figure 5: Power consumption with speed-scaling mode under near-far traffic.

5.2.2. *Power-Consumption Evaluation*

We assume that a logical switch LS(k, l) uses speed-scaling and its power consumption [9] is given by

$$P^{LS} = P^{idle} + (P^{max} - P^{idle})\check{R}(k, l)/C_{LS}(k, l),$$

where $C_{LS}(k, l)$ is the capacity of a logical switch given in Eq. (8). P^{idle} is the power consumption when the logical switch is idle (i.e., all switch ports are enabled but don't have traffic) and cannot be eliminated through speed-scaling. P^{max} is the maximum power consumption when the incoming traffic rate reaches the switch capacity. Since a logical switch LS(k, l) consists of $(k/2)^{l-1}$ k-port switches, $P^{max} = (k/2)^{l-1}P^{s(k)}$, where $P^{s(k)}$ is the power consumption of a k-port switch given in Eq. (9).

Fig. 5 shows the power consumption using the same near-far-traffic model as in Sec. 4. It is assumed that $P^{idle} = 0.5P^{max}$. The qualitative results remain unchanged over a wide range of P^{idle} values. Because small switches require more ports and there is no opportunity to turn off switches with speed-scaling, a large number of small switches are likely to consume more power than a small number of large switches irrespective of the traffic patterns.

Fig. 6 shows the corresponding curves under geometric traffic. We also see similar observations as with the near-far traffic

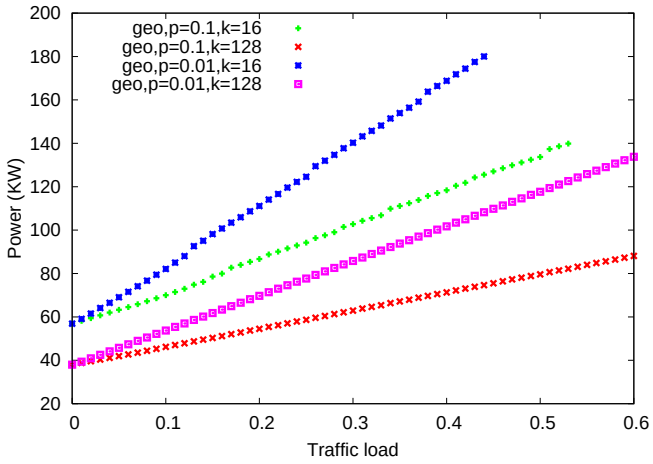

Figure 6: Power consumption with speed-scaling mode under geometric traffic.

in that small switches never show an advantage at any traffic load. It can be concluded that networks using large switches are generally more power-efficient than those using small switches with speed scaling.

6. Simulation Evaluation

6.1. *Simulation Setting*

In this section, we turn to simulation to evaluate the power-saving performance of fat-tree networks with different switch sizes. The simulation experiments were conducted on a data-center testbed built on NS-3, supporting packet mode and flow mode. We use the bin-packing algorithm proposed in ElasticTree [3] to consolidate flows to a minimum set of paths that are required to sustain different traffic loads. Typically datacenter networks incorporate some level of capacity safety margin to prepare for traffic surges [3]. In such cases, the network may allocate more capacity than essential for normal workload. To implement safety margin, we monitor the utilization of each outgoing port of a switch. If the safety margin is set to θ, then a new port on the same side of the switch will be enabled (opened) when the utilization exceeds $1 - \theta$. The corresponding port in another switch will also be enabled to establish the new link. The performance metrics evaluated in the simulation include power-saving efficiency, packet delay, and packet-loss rate.

In the packet-mode simulation, each server injects a certain number of UDP flows into the fat-tree network with packets of size 1.5KB. To emulate flow arrivals and terminations, we give each flow two states: ON and OFF. The duration of each flow (i.e., the ON state) is an exponentially distributed random variable, which is determined when the flow is generated. The idle time (i.e., the OFF state) of each flow is also an exponentially distributed random variable, decided when the previous ON state finishes. In our simulation, the average ratio of ON period to OFF period is 3.

With packet-mode simulation, we are able to obtain performance metrics including power-saving efficiency, packet delay, and packet-loss rate. However, because of its long sim-

ulation time, only relatively small fat-tree networks can be evaluated in a reasonable period of time. Here we use packet-mode simulation to evaluate two 512-server fat-tree networks, one with 4 levels (i.e., FT$(8, 4)$) and the other with 2 levels (i.e., FT$(32, 2)$). The link capacity of the fat-tree networks is assumed to be 1 Gbps, and the link delay is 2 microseconds. Each output port of the switch has a buffer space of 50 1.5KB packets (i.e., 75KB).

Flow-mode simulation captures flow arrival and termination events without injecting packets as in the packet mode. The durations of a flow and between flows are exponentially distributed. Due to its coarser granularity, flow-mode simulation can be used to evaluate power-saving efficiency for large-scale fat-tree networks with different flow arrival patterns. Two 8192-server fat-tree networks with different switch sizes (i.e., FT$(16, 4)$ and FT$(128, 2)$) are used in the evaluation.

We consider two different traffic patterns in the evaluation, *near* and *far*, which are defined in Eq. 10. Each switch distributes packets evenly across its output ports to the upper-level switches using hashing on a per-flow basis. Here we assume that there are many flows between each source-destination pairs so that flow sizes are relatively small compared with the link capacity. In such a case, the traffic loads on different parallel links are nearly equal.

Two power models (sleep mode and speed-scaling mode) are used in simulation to evaluate the power consumption of the fat-tree networks. Parameters used in simulations are the same as those used in analysis, unless stated otherwise.

6.2. *Power-saving Performance*

Figure 7 shows the power consumption of two 8192-server fat-tree networks (FT$(16, 4)$ and FT$(128, 2)$) using the sleep mode. The evaluation is based on three different traffic patterns and the safety margin is set to $\theta = 0$. The three traffic patterns are: (1) all near: meaning that 100% traffic from each server is near traffic (i.e., $\alpha = 0$); (2) 30% far: meaning that 30% traffic from each server is far traffic and 70% is near traffic (i.e., $\alpha = 0.3$); (3) 50% far: meaning that 50% traffic from each server is far traffic and 50% is near traffic (i.e., $\alpha = 0.5$). We can see that when using the sleep mode, small switches are more power-efficient than large switches when servers are communicating in close proximity or when the traffic demand is low. When servers are communicating with other far servers or the traffic demand is high, large switches are more power-efficient. The simulation result matches our analytical result very well.

Figure 8 shows the power consumption of two 8192-server fat-tree networks (FT$(16, 4)$ and FT$(128, 2)$) in the speed-scaling mode. We can see that with the speed-scaling mode, large switches are more power-efficient than small switches, which is also consistent with our analytical result.

From Figure 7 and Figure 8, we can see that the power consumption of the fat-tree networks is roughly linear to the traffic load when sleep mode or speed-scaling mode is enabled. A fat-tree network without power saving will consume more power that is also independent of the traffic load.

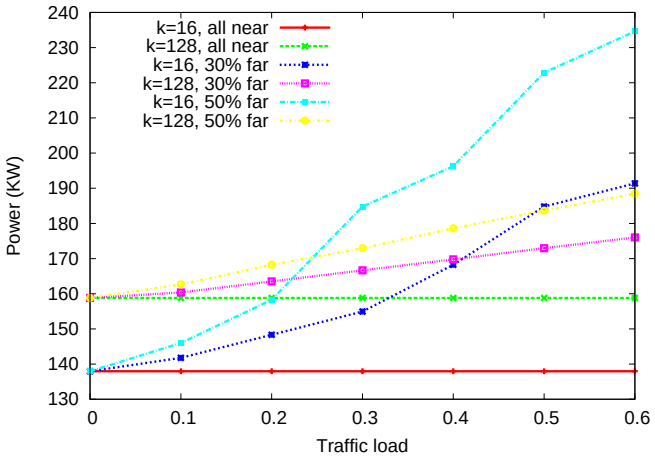

Figure 7: Power consumption of two 8192-server fat-tree networks FT$(16, 4)$ and FT$(128, 2)$ in sleep mode ($\theta = 0$).

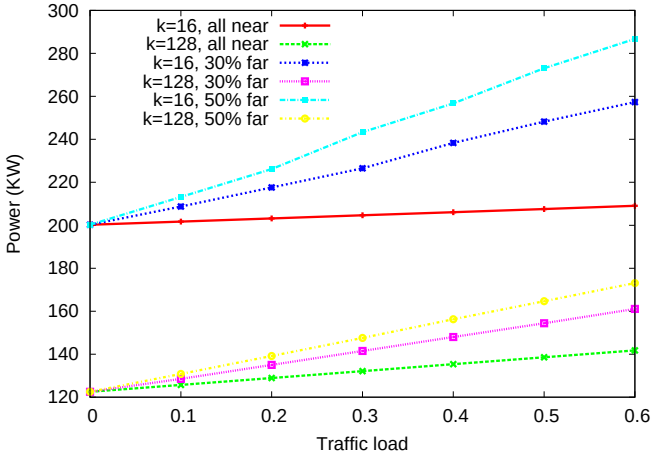

Figure 8: Power consumption of two 8192-server fat-tree networks FT$(16, 4)$ and FT$(128, 2)$ in speed-scaling mode ($\theta = 0$).

6.3. Packet Delay and Loss Rate

Since the packet-mode simulation is very time-consuming, we evaluate packet delay and loss rates only on small fat-tree networks with 512 servers using sleep mode. With sleep mode, the fat-tree initially starts as a *thin-tree* when the load is low as the bisection bandwidth decreases at higher levels. It becomes fatter as the load increases. We want to study if the power-saving feature (i.e., sleep mode) adopted by the fat-tree networks will have any adverse impact on performance. Figure 9 shows the packet delay and packet-loss rates in a 4-level, 512-server fat-tree network FT$(8, 4)$. The traffic pattern from each server consists of 50% near traffic and 50% far traffic, and the safety margin is $\theta = 0.1$. We can see that the packet-delay curve shows a sawtooth pattern, with an increasing trend as the load increases. The sawtooth pattern of the delay curve is the consequence of flow consolidation. Packet delay in a fat-tree network is dominated by the most congested link(s). When the traffic load is very low, the system only needs a minimal spanning tree and no congestion occurs. Also, the packet delay is very low. As the traffic load increases, multiple links of the minimal spanning tree starts to be more congested as can be observed by the increasing trend of the packet delay. When the load on the most congested link exceeds a pre-determined threshold (i.e., $1 - \theta = 0.9$ in this case), a new path will be activated to relieve the current congestion. That is why there are recurrent drops in the packet delay, forming a sawtooth pattern, as the traffic load increases.

The above phenomenon can be easily understood when we look at Figure 9 and Figure 10 together. Figure 10 shows the average link utilization values at different levels of the fat-tree network. These curves also show a sawtooth pattern. When the traffic load increases, the link utilization values at different levels increase linearly but with different slopes. The differences in slopes are due to the fact that the number of opened links between levels 3∼4 is only $2/k$-th (in this case, $k = 8$) of that between levels 2∼3, while the total traffic loads on both levels are identical. So, when the traffic load increases, the load on links between levels 3∼4 increases much faster than those on links between levels 2∼3 and level 1∼2. When the links between levels 3∼4 become congested, new links between levels 3∼4 will be opened to relieve the congestion. This is the reason that we see the sawtooth pattern occurring first on links between levels 3∼4. It is also intuitively clear that the peaks and valleys that appear in the delay curve of Figure 9 and the level 3∼4 utilization curve of Figure 10 always occur at the same value of the traffic load. This is because high link utilization directly causes high packet delay. Since we use a relatively large safety margin value of 0.1 in this scenario, packet loss is generally rare, as shown in Figure 9.

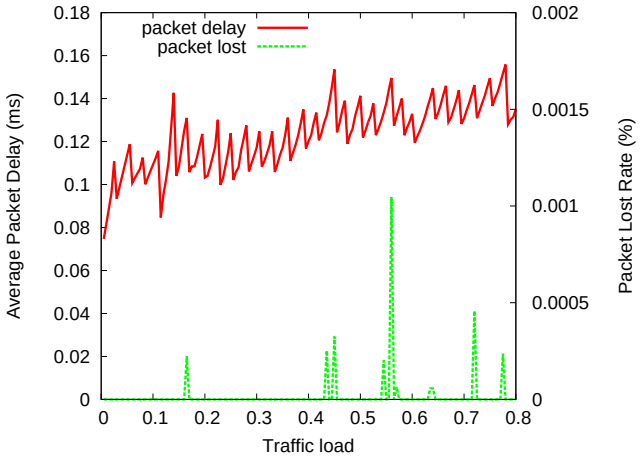

Figure 9: Packet delay and loss rate of fat-tree network FT$(8, 4)$ (safety margin $\theta = 0.1$, traffic pattern: 50% far).

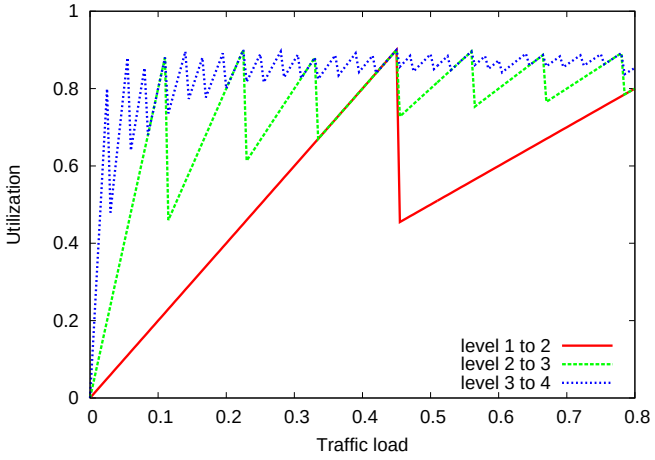

Figure 10: Average utilizations of links between different levels of fat-tree network FT$(8, 4)$ (safety margin: $\theta = 0.1$, traffic pattern: 50% far).

6.4. The Impact of Safety Margin on Performance

Figure 11 and Figure 12 show the packet delay and packet-loss rates of the 512-server fat-tree network FT$(8, 4)$ under safety margins 0.05 and 0.01, respectively. Other settings in these two figures are identical to those in Figure 9. We can see that a slight change to the safety margin from 0.01 to 0.05 can lead

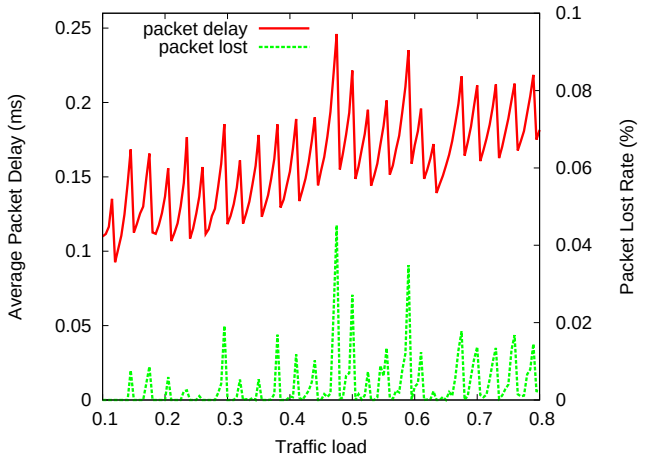

Figure 11: Packet delay and loss rate of fat-tree network FT$(8, 4)$ (safety margin: $\theta = 0.05$, traffic pattern: 50% far).

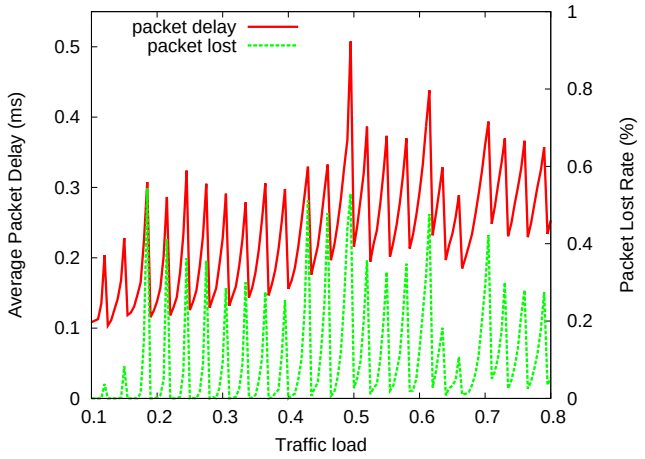

Figure 12: Packet delay and loss rate of fat-tree network FT$(8, 4)$ (safety margin: $\theta = 0.01$, traffic pattern: 50% far).

to significant changes on the performance, especially on the packet-loss rate. The impact becomes smaller when we continue to increase the safety margin.

6.5. The Impact of Safety Margin on Power Consumption

In Figure 13, we show the power consumption of the fat-tree network FT$(8, 4)$ when different safety margins are used. We observe that the impact of the safety margin on power consumption is generally minor.

Based on the simulation results above, it is wise to select a moderate safety margin (e.g., $\theta = 0.1$) in the fat-tree network to achieve a good balance between packet delay, packet-loss rate and power consumption.

6.6. The Impact of Switch Size on Performance

Figure 14 shows the average link utilization between levels 2 and 1 of a 2-level 512-server fat-tree network FT$(32, 2)$ under 50% far traffic pattern. We set $\theta = 0.1$. As shown in the figure,

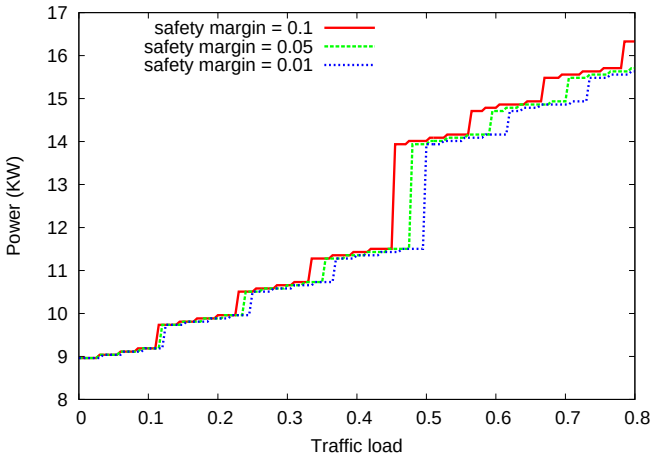

Figure 13: Power consumptions of fat-tree network FT$(8, 4)$ under different safety margins.

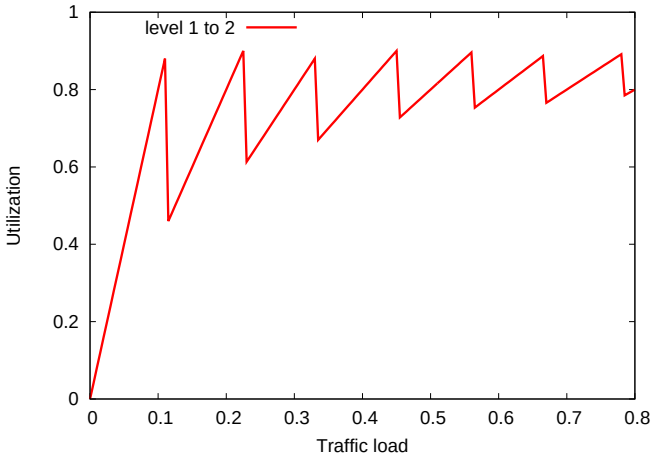

Figure 14: Average utilization of links between level 2 and level 1 of fat-tree network FT$(32, 2)$ (safety margin: $\theta = 0.1$, traffic pattern: 50% far).

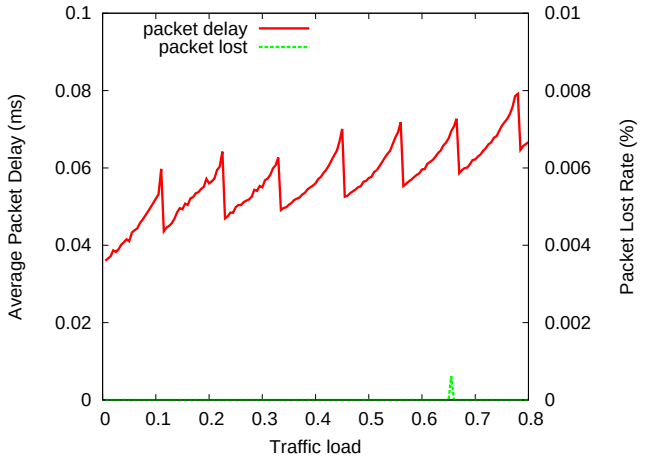

Figure 15: Packet delay and loss rate of fat-tree network FT$(32, 2)$ (safety margin=0.1, traffic pattern: 50% far).

the curve has a sawtooth pattern similar to those in the 4-level fat-tree network FT$(8, 4)$.

Figure 15 shows the packet delay and packet loss-rate performance of FT$(32, 2)$. Packet loss is generally low in this case. When compared with Figure 9, we can see that large switches with fewer hops have better packet delay and loss-rate performance.

6.7. Evaluation using Real-life Datacenter Traffic

The evaluation presented above is based on synthetic traffic. We now evaluate the power saving efficiency of fat-tree networks with different switch sizes using traffic traces from a real datacenter. We were able to collect the traffic traces at several aggregation links from a production data center for a period of six hours on 8 Nov. 2012. This data center consists of over 2000 servers, serving over one million intranet users to provide a variety of services including email service, SAN based file sharing, Microsoft IIS based web cluster, traditional mainframe applications such as HR, payroll, and purchasing, new collaboration tools such as SharePoint and helpdesk ticketing system, and separate servers farms for application development, staging and quality analysis (QA). The average link utilization over the measurement period (i.e., six hours) is shown in Figure 16.

We map flows in the traffic traces to two 512-server fat-tree networks, FT$(8, 4)$ and FT$(32, 2)$, based on their source and destination IP addresses, such that on average 50% of the traffic from each server is far traffic. Each server may have multiple IP addresses for different virtual machines.

Figure 17 shows the power consumed by the fat-tree network FT$(8, 4)$ in sleep mode. We can see that the power consumption tracks the traffic load well, thanks to the power saving feature.

Figure 18 shows the power consumed by the fat-tree network FT$(32, 2)$ in sleep mode. The power consumption of FT$(32, 2)$ shows smaller fluctuation than that of FT$(8, 4)$. This is because the number of switches in FT$(32, 2)$ is lower than that in FT$(8, 4)$. Thus, fewer switches can be turned off or on in FT$(32, 2)$ when traffic load varies. The power consumption

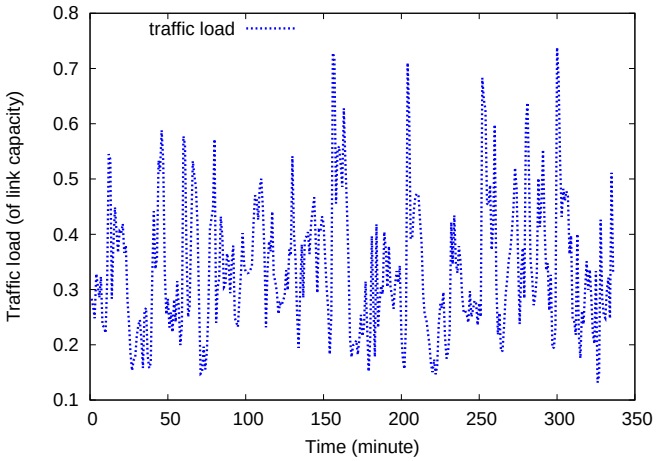

Figure 16: Average link utilization in a production datacenter network.

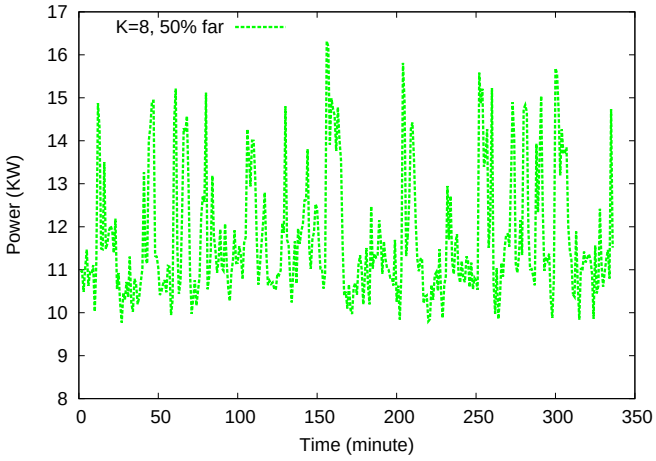

Figure 17: Power consumption of FT$(8, 4)$ under real-life datacenter traffic.

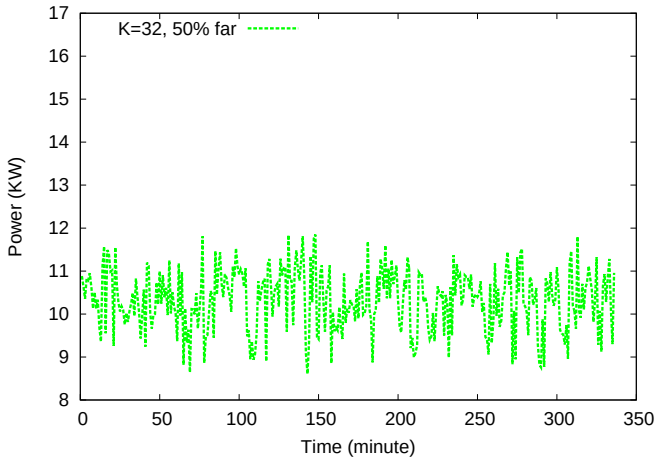

Figure 18: Power consumption of FT$(32, 2)$ under real-life datacenter traffic.

of FT$(32, 2)$ is also lower than that of FT$(8, 4)$ since far traffic is quite substantial. In this case, using large switches is generally more power-efficient than using small switches.

7. Conclusion

In this paper, we presented a methodology at a design stage of deploying a datacenter network. Given the number of servers that need to be supported in a data center and the network topology, our approach determines the right switch size that minimizes the energy consumption of the network during the expected operation of the data center. Our analytical and simulation results reveal that a fat-tree network using sleep mode with a large number of small switches would generally consume less power than the counterpart with a small number of large switches when the traffic demand is low or localized. This observation is generally not true if the network uses the speed-scaling mode for power saving. Our methodology provides useful guidance during the design stage of a datacenter network to further reduce the network power consumption proposed in prior work. In this work, we only consider advantages in terms power saving. In practice, other factors such as equipment cost, deployment or labour effort, etc., would also have to be taken into account. These factors will be investigated in the future.

References

[1] L. Barroso, U. Holzle, The case for energy-proportional computing, IEEE Computer 40 (12) (2007) 33–37.

[2] A. Gandh, M. Harchol-Balter, R. Das, C. Lefurgy, Optimal power allocation in server farms, in: SIGMETRICS, 2007, pp. 157–168.

[3] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banarjee, N. McKeown, Elastictree: Saving energy in data center networks, in: NSDI, 2010.

[4] X. Wang, Y. Yao, X. Wang, K. Lu, Q. Cao, Carpo: Correlation-aware power optimization in data center networks, in: INFOCOM, 2012, pp. 1125–1133.

[5] D. Abts, M. Marty, P. Wells, P. Klausler, H. Liu, Energy proportional datacenter networks, in: ISCA, 2010, pp. 338–347.

[6] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, S. Lu, Dcell: A scalable and fault-tolerant network structure for data centers, in: ACM SIGCOMM, 2008.

[7] P. Mahadevan, P. Sharma, S. Banerjee, , P. Ranganathan, A power benchmarking framework for network devices, in: In Proceedings of IFIP Networking, 2009.

[8] I. Widjaja, A. Walid, Y. Luo, Y. Xu, H. Chao, Small versus large: Switch sizing in topology design of energy-efficient data centers, in: 2013 IEEE/ACM 21st International Symposium on Quality of Service (IWQoS), 2013, pp. 1–6.

[9] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, D. Wetherall, Reducing network energy consumption via sleeping and rate-adaptation, in: NSDI, 2008, pp. 323–336.

[10] C. Gunaratne, K. Christensen, B. Nordman, S. Suen, Reducing the energy consumption of ethernet with adaptive link rate (alr), IEEE Transactions on Computers 57 (4) (2008) 448–461.

[11] C. E. Leiserson, Fat-trees: Universal networks for hardware-efficient computing, IEEE Trans. on Computers 34 (10) (1985) 892–901.

[12] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, ACM SIGCOMM (2008) 63–74.

[13] X.-Y. Lin, Y.-C. Chung, T.-Y. Huang, A multiple LID routing scheme for Fat-Tree-Based ifiniband networks, in: IPDPS, 2004.

[14] X. Yuan, wickus Nienaber, Z. Duan, R. Melhem, Oblivious routing in fat-tree based system area networks with uncertain traffic demands, IEEE Trans. on Networking 17 (5) (2009) 1439–1452.

[15] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, parantap Lahiri, D. A. Maltz, P. patel, S. Sengupta, Vl2: A scalable and flexible data center network, ACM SIGCOMM (2009) 51–62.

[16] J. H. Patel, Performance of processor-memory interconnections for multiprocessors, IEEE Trans. on Computers 30 (10) (1981) 771–780.

[17] V. Eramo1, A. Germoni, A. Cianfrani, E. Miucci, M. Listanti, Comparison in power consumption of mvmc and benes optical packet switches, in: IEEE NOC, 2011, pp. 125–128.