Energy Optimizations for Data Center Network: Formulation and its Solution

Shuo Fang, Hui Li, Chuan Heng Foh, Yonggang Wen
School of Computer Engineering
Nanyang Technological University
Singapore

Khin Mi Mi Aung
Data Storage Institute
A*STAR
Singapore

***Abstract*—Data center consumes increasing amount of power nowadays, together with expanding number of data centers and upgrading data center scale, its power consumption becomes a knotty issue. While main efforts of this research focus on server and storage power reduction, network devices as part of the key components of data centers, also contribute to the overall power consumption as data centers expand. In this paper, we address this problem with two perspectives. First, in a macro level, we attempt to reduce redundant energy usage incurred by network redundancies for load balancing. Second, in the micro level, we design algorithm to limit port rate in order to reduce unnecessary power consumption. Given the guidelines we obtained from problem formulation, we propose a solution based on greedy approach with integration of network traffic and minimization of switch link rate. We also present results from a simulation-based performance evaluation which shows that expected power saving is achieved with tolerable delay.**

***Index Terms*—Power management, Optimization, Data center**

I. Introduction

With increasing scale of data center, it has become an energy hog which draws more and more attentions as currently it is estimated to consume probably 120 billion kilowatt hours per year in 2011 [1]. Several proposals have attempted to achieve more energy-efficient data centers [2], [3], [4]. Yet the main effort of these works are concentrated on reduction of server energy cost and related cooling. Along with the improvement in power savings in servers, network power consumption remains almost at the same stage as of now.

However, studies on data center traffic characteristics show that network is seldom utilized at its peak capacity. Research works given in [5], [6] and [7] reveal the bursty nature of traffic. They reported that the “elephant” flows only last about one second. In contrast, data center network devices are running at almost the same level regardless of their traffic loads [8], thus a large part of network energy is wasted in idle state. Observing these phenomena, proposals with energy proportional network in data centers have been developed including ElasticTree [8] with a central control optimizer, and energy proportional network [6] with an independent control of unidirectional channels on network links. They are interesting proposals with consideration of one or several aspects in reducing network energy consumption.

However, with our observation, we find that redundancies in network architecture take considerable amount of energy consumption. In data centers, for the purpose of increasing network bandwidth and fault tolerance, redundancies of network devices exist which consume large proportion of powers. Many recently proposed data center architectures such as Fat Tree [9], DCell [10] and VL2 [11] are all equipped with redundancies. Indeed, redundancy and load balancing function are helpful in appearance of heavy traffic load, but considering the comparatively low traffic in data center in most of times, we strongly argue optional load balancing is a preferable way in saving power in data center.

On the other hand, data center switches can be configured at several link rates, both studies of adaptive link rate (ALR) [12] and our experiments indicate that different link rates of switch ports consume energy at various levels. As a result, instead of all switch ports running at the highest rate, namely 10Gbps at the moment, adaptive transition of port operating rate will save more power especially in an under utilized network environment.

Our goal in this paper is to adaptively activate or deactivate part of network topology to minimize switch usage as well as adjust link rates of switch ports according to traffic loads, in order to save energy in data centers. Our proposed solution is based on Fat Tree topology which is an attractive solution for modern data centers, however, with simple modifications, our solution can adapt to most of data center architectures which we may leave as future works. The rest of the paper is organized as follows. Section II introduces background information by presenting our experiment results for switch power consumption level and describing Fat Tree topology with terms that we employ in our proposal. Section III analyzes the network model and formulates an optimization problem, after discussion based on the formulation, we develop a greedy approach. We further test our solution with OMNET++ simulator in Section IV and present results regarding simulation tests with different settings. Important conclusions are drawn in Section V.

II. Background

In this section, we offer background information for our proposals. Firstly, we brief Fat Tree topology with some terms we introduce to better describe our solution. Secondly, we introduce the concept of ALR and present our experiment results for validation of these statements.

A. Fat Tree Architecture

Clos network topology [13] has been proposed half a centenary ago for telephone switches, it aims to deliver high network throughput with commodity switches. Recently, in

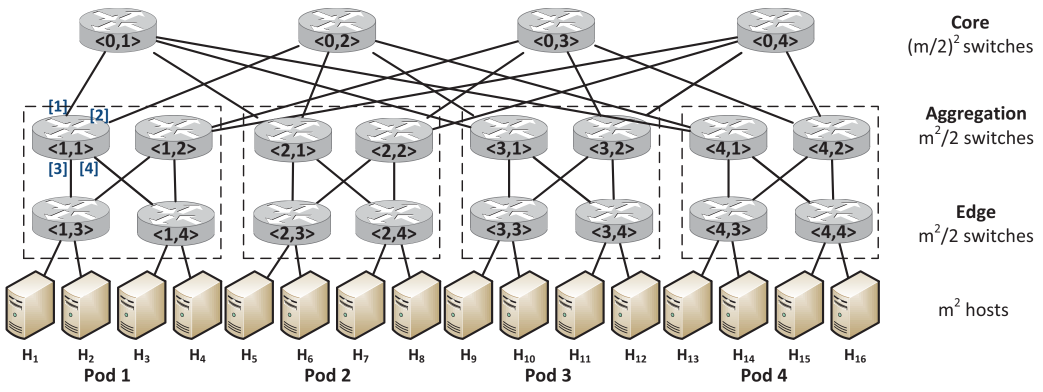

Fig. 1. Illustration of 4-ary Fat Tree topology.

[9], Al-Fares *et al.* adopt a standard instance of a Clos topology called Fat Tree, which provides non-blocking network connection with 1:1 oversubscription ratio [9]. This proposal is widely acknowledged in the field, which is our considered topology for this analysis. However, with some adjustments our proposal can also be applied to other topologies.

Details about Fat Tree as a data center architecture can be found in [9]. However, for description completeness, we summarize those related to our work here. In a data center of a m-ary tree as shown in Fig. 1, it is made up by m-port switches, where m should be a positive even number. A Fat Tree contains three layers, namely edge layer, aggregation layer and core layer. The lower two layers are separated into m pods, each containing two layers of $m/2$ switches, with lower edge switches and upper layer aggregation switches. In the upmost layer, there are $(m/2)^2$ core switches, each core switch has a connection to each of the m pods.

For clear explanation, we use an ordered pair $\langle p, r\rangle \in \mathcal{R}$ to denote a switch, where set $\mathcal{R}$ is

$$\begin{aligned}\mathcal{R} &= \{\langle 0, r\rangle \,|r \in \{1, 2, ..., (m/2)^2\}\} \\ &\bigcup \ \{\langle p, r\rangle \,|p \in \{1, 2, ..., m\}, r \in \{1, 2, ..., m\}\}\end{aligned}$$

and p denotes the pod identity of switches from left to right, except that $\langle 0, r\rangle$ means core switches, and r denotes the sequence of switches in each pod from left to right and top to bottom, as shown in Fig. 1.

The topology follows strict connection rules. For the edge switch, half of the edge switch ports are connected to $m/2$ end hosts and the rest half of switch ports are connected to the $m/2$ aggregation switches in the same pod, in other words, each edge switch has access to each aggregation switch in the same pod and no access to switches in the other pod.

Each pod connects to each core switch through its aggregation switches. Each aggregation switch connects to $m/2$ core switches, each core switch has access to all the pod, port k of each core switch is connected to Pod k. Here, $k \in \{1, 2, ...m\}$ denotes the switch port number, from left to right and top to bottom.

Depending on above discussed connection relationship, we use a matrix function $C(\langle p_l, r_l\rangle, k_l, \langle p_h, r_h\rangle, k_h)$ to denote connections between ${k_l}^{th} \in \{1, 2, ..., m/2\}$ port of lower layer switch $\langle p_l, r_l\rangle$ and ${k_h}^{th} \in \{1, 2, ..., m\}$ port of higher layer switch $\langle p_h, r_h\rangle$, thus

$$C\left(\langle p_l, r_l\rangle, k_l, \langle p_h, r_h\rangle, k_h\right) = \begin{cases} 1, \mathscr{C} \\ 0, otherwise \end{cases}, \qquad (1)$$

where the condition $\mathscr{C}$ is given by

$$\left(p_h = 0, k_h = p_l, r_h = k_l + (r_l - 1) \times \frac{m}{2}\right) \qquad (2)$$

$$\text{or} \quad \left(p_h = p_l \neq 0, r_h = k_l, k_h = r_l - \frac{m}{2}\right). \qquad (3)$$

Eqn. (2) specifies connections between core and aggregation switches, in which port k of core switch $\langle 0, r\rangle$ is connected to Pod k, and aggregation switches are connected to core switches with a stride of $m/2$. Eqn. (3) presents connections between aggregation and edge switches, in which the switches are in the same pod and each port is connected to a switch in the other layer.

B. Power Consumption of Switch Links

It has been revealed in [12], [14] that energy consumption varies among different link rates in Cisco Catalyst 2970 switches. However, experiment results are lack of performances of 10G switch links. To report a more timely power consumption rate of switches, we set up a test scenario for Dell PowerConnect 8024F switch as shown in Fig. 2 using Hameg HM8021 Multimeter. After power measurement test, we yield power consumption rate with different number of active links as shown in Fig. 3. In our experiment, the switch power consumption does not vary when traffic load changes. That means, either with no traffic load or full load, a switch consumes the same amount of power given that a certain set of link rates. From this figure, it can be seen that switches consume more power given increasing either link rates or number of links. We also observe power consumption follows a linear pattern in terms of number of links. We can roughly calculate out that a 100Mbps link consumes no more power than system idle state, while each 1Gbps link consumes around 1.2W more and a 10Gbps link consumes around 4.3W more power, with a base of 110W. When all ports are idle, a switch can switch to SLEEP mode which consumes 10W. In order to verify the linear increment of power, we further test on different settings with a mix of link rates as shown in Table I, which shows the similar results as we expect.

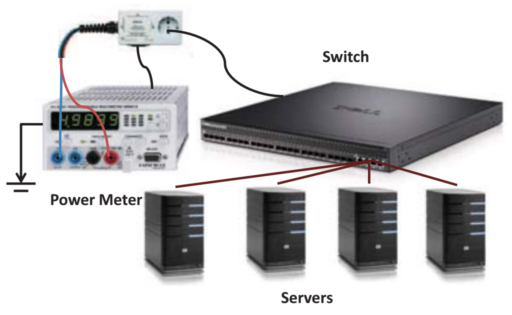

Fig. 2. Our power measuring testbed.

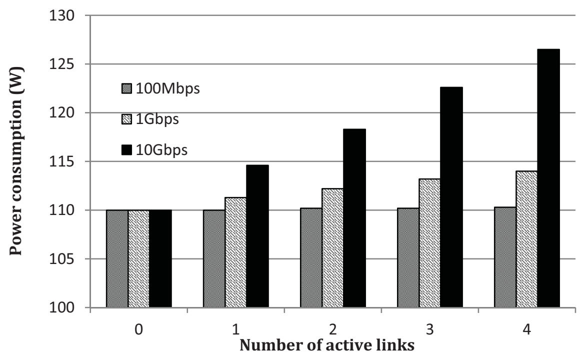

Fig. 3. Power consumption of Dell PowerConnect 8024F.

TABLE I
SWITCH POWER CONSUMPTION OF DELL POWERCONNECT 8024F.

Number of links			Power consumption
100Mbps	1Gbps	10Gbps	(W)
0	1	1	115.6
1	0	2	118.7
0	1	2	119.7
0	2	2	120.9

III. MODEL FORMULATION AND ITS SOLUTION

In this section we formulate an optimization problem given the constrains of a port's output is equal to another port's input if a connection between the two ports exists. Further, we obtain a greedy approach and present later in this section.

A. Optimization Formulation of the Problem

For each switch $\langle p, r\rangle$, $\lambda_{k,d}^{\langle p,r\rangle}$ denotes the arrival load at incoming port k that is destined to destination d. Likewise, for each outgoing port k of switch $\langle p, r\rangle$, $\mu_{k,d}^{\langle p,r\rangle}$ denotes the distribution of traffic load on switch $\langle p, r\rangle$. The traffic load in each outgoing port also presents as input traffic load for its adjacent port. That means,

$$\lambda_{k_h,d}^{\langle p_h,r_h\rangle} = \sum_{p_l,r_l,k_l} \mu_{k_l,d}^{\langle p_l,r_l\rangle} \cdot C\left(\langle p_l,r_l\rangle, k_l, \langle p_h,r_h\rangle, k_h\right), \quad (4)$$

and

$$\lambda_{k_l,d}^{\langle p_l,r_l\rangle} = \sum_{p_h,r_h,k_h} \mu_{k_h,d}^{\langle p_h,r_h\rangle} \cdot C\left(\langle p_l,r_l\rangle, k_l, \langle p_h,r_h\rangle, k_h\right). \quad (5)$$

Our objective is to choose link rates $l_i^{\langle p,r\rangle}$ for each switch, where $i \in \{1, 2, ..., m\}$, so as to minimize the total power consumption $\sum_{p,r} P(\sum_i l_i^{\langle p,r\rangle})$ as follows,

$$\min \sum_{\langle p,r\rangle} P(\sum_i l_i^{\langle p,r\rangle}) \quad (6)$$

subject to

$$\sum_k \lambda_{k,d}^{\langle p,r\rangle} = \sum_k \mu_{k,d}^{\langle p,r\rangle}, \quad (7)$$

$$\sum_d \lambda_{k,d}^{\langle p,r\rangle} \leq l_k^{\langle p,r\rangle}, l_i \in \mathcal{L}, \quad (8)$$

$$\sum_d \mu_{k,d}^{\langle p,r\rangle} \leq l_k^{\langle p,r\rangle}, l_i \in \mathcal{L}, \quad (9)$$

for each switch $\langle p, r\rangle$ where $\mathcal{L} = \{100\text{Mbps}, 1\text{Gbps}, 10\text{Gbps}\}$. The constraint in Eqn. (7) presents the equality of incoming traffic and outgoing traffic destined to the same end host. Eqn. (8) constrains that link rate of a port should be capable of handling all the incoming traffic on this port. The same condition applies to outgoing traffic as shown in Eqn. (9) for full duplex link of switches.

It is easily observed that inter-pod packet routing in a Fat Tree topology is divided into two procedures. Packets often first travel to a core switch from the source, and then back to the edge switch connecting the destination. We shall call the packet forwarding from sources to cores as uplink transmission flow and packet forwarding from cores to destinations as downlink transmission flow.

In finding the minimum power consumption, from Eqn. (6), we know that a lesser number of activated links leads to lower power consumption. For the uplink transmission flow, to lower the number of activated links yet supporting the load, the Equations (7)-(9) suggest the aggregation of traffic flows. That is, a smallest number of output ports is used for uplink transmission.

For the downlink transmission flow, we notice that there is always a single path packets can travel from a particular core to its destination. The optimal power consumption is achieved when packets are aggregated during the uplink transmission in a way that packets to the same destination are aggregated to the same core to achieve maximum sharing the downlink transmission. This can be achieved by aggregating packets of the same destination first, as specified in Eqn. (7).

B. A Greedy Approach

Based on our observation from the system formulation, we thus propose a greedy approach which satisfies both requirements indicated by Equations (6)-(9) by dynamic load balancing and adaptive rate migrating, in designing the routing algorithm. The key idea in this solution is to utilize as few switches, switch links and switch link rates as possible.

In the initial stage, network system begins with no active switches, switches are only enabled when packet arrives. Packets are automatically routed to a path on a spanning tree with the least link rate given the traffic load. Port buffers are examined every β packets, links are upgraded whenever potential congestions happen which is indicated by buffer level

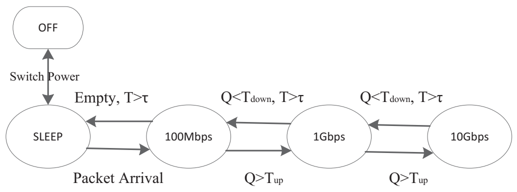

Fig. 4. Port state transition.

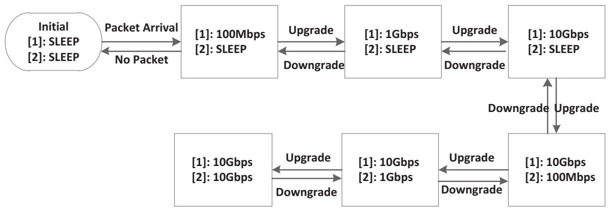

Fig. 5. Switch state transition.

exceeding a preset upgrade threshold T_{up}, and downgraded when a relief on the link is detected. To avoid a frequently change of state transition on a port, we only consider that a buffer level has been lower than a preset downgrade threshold T_{down} for a time interval of τ. If any link almost reaches its maximum capacity, 10Gbps at the current state, its switch will expand the tree structure by activating an adjacent port to support heavier traffic load. Disabling of ports happens in the condition that their buffers have been empty for a certain period of time. Moreover, traffic load will be distributed to with a certain quota according to the link rate on each port.

To be specific, from a port's point of view, each port may experience several states, namely SLEEP, 100Mbps, 1Gbps and 10Gbps. The transitions within these states are performed according to two thresholds, T_{up} and T_{down}, and an interval time τ. Figure 4 illustrates the transitions. That means, a port upgrades to a higher rate if its queue length exceeds the upgrade threshold T_{up}, and downgrades to a lower rate, if its queue length has been lower than the downgrade threshold T_{down} for time of τ. However, it is a bit different for upgrading from or downgrading to SLEEP state, which upgrades whenever a packet arrives and downgrades only when its queue has been empty for τ seconds. In simulation, we create a time trigger whenever queue length goes under T_{down}, which might be canceled if the queue length rises over T_{down}. Thus, the port will downgrade once the trigger enables.

From a switch's point of view, when all of its links almost reach their capacities which is indicated by that all the activated ports are running at their maximum speed and a queue length exceeds the upgrade threshold. In such a case, another port will open to combat congestion through its neighbor links. In contrary, when the congestion is released, extra link will downgrade or even close to save energy. In other words, Port$_x$ can only open if Port$_{x-1}$ is not upgradeable and Port$_x$ can downgrade only if Port$_{x+1}$ is closed. As constrained by this algorithm, there is at most one active link not running at maximum link rate because a port can only be activated or downgraded when all the other active ports are transmitting at the highest link rate. Let us take the simple Fat Tree in Fig. 1 as an example, it shows that ports in switch $\langle 1, 1\rangle$ are Port$_1$ to Port$_4$. By default, when a packet of an uplink transmission flow arrives, the switch will automatically open Port$_1$. When Port$_1$ becomes congested with link rate of 10Gbps, Port$_2$ will be enabled and running at 100Mbps as shown in Fig. 5.

As for a switch with $x+1$ active links, there are fully utilized links from Port$_1$ to Port$_x$, while Port$_{x+1}$ might run at a medium rate. So the concern here is to distribute loads among flows with a proportion roughly equal to rate ratio among links. This distribution ratio guarantees that the potential port for degradation will not be overflowed by excessive traffic while other ports are free of congestion. To decide which port to route to, the mechanism first calculate the total bandwidth of all $x+1$ active links to yield proportion of each port, then it generates a random number which assigns the flow to a certain port. In simulation, this random number is generated by a hash function, which hashes according flow information. Noted here that with each activating or deactivating of a port, a rescheduling of flow assignment will follow. Here we only consider the flow reschedule without packet reordering. We assume that packet reordering is handled by upper layer protocol such as TCP.

IV. Simulation and Results

In this section we present and compare computer simulation results of a common routing algorithm and our solution. The two algorithms are simulated using OMNET++ [15], which is a component-based, modular and open-architecture discrete event-driven network simulator. OMNET++ provides detailed simulations of various network architecture, components and events so that a fair comparison can be guaranteed by utilizing OMNET++ simulation platform.

To better illustrate, here we use a simple 4-ary Fat Tree as show in Fig. 1. We integrate two parts of testing into simulations, the first of which shows mechanism testing of our simulation model and the second of which presents a more realistic scenario with performance evaluations and comparisons.

We firstly run a simulation test with a flow generated from H_4 to H_8. Simulation settings are listed in Table IV. The flow starts at 1s and ends at 6s. In between, the flow frequency changes on every second, which starts with an packet generating interval of 10ms at 1s, grows to 0.2ms at 2s and 0.01ms at 3s, then returns to 0.2ms at 4s and 10ms at 5s, and finally stops at 6s. The simulation parameters are listed in Table III. Figure 6 shows the throughput changes of the system which presents a clearer idea of the frequency changes. We tick Y-axis exponentially to ensure the small

TABLE II
Simulation settings for single flow test.

Parameter	Value
Number of flows	1
Sender	H_4
Receiver	H_8
Flow starts time	1s
Flow ends time	6s

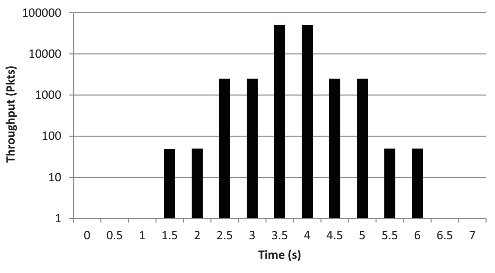

Fig. 6. Throughput at H_8.

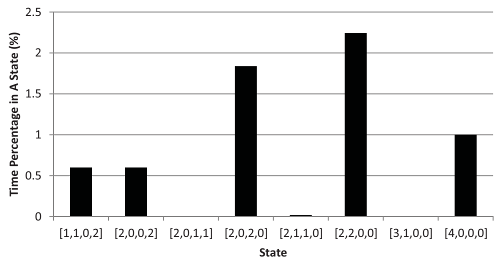

Fig. 7. Switch state time for aggregation switch $\langle 1,1\rangle$.

amount of traffic beginning at 1s and ending at 6s to be visible as compared to the throughput at 3.5s and 4s.

During the simulation, we can see the flow takes the route $H_4 \rightarrow \langle 1,4\rangle \rightarrow \langle 1,1\rangle \rightarrow \langle 0,1\rangle \rightarrow \langle 2,1\rangle \rightarrow \langle 2,4\rangle$, whose links increase rate as the flow frequency climbs and reduce link rate when the flow frequency drops. Here we present statistics of switch state which records down time spends on each state of aggregation switch $\langle 1,1\rangle$ in Fig. 7. In this figure, we denote switch state in an $[a,b,c,d]$-tuple, in which a denotes number of ports in SLEEP mode, b,c,d denote number of ports running at rate 10Mbps, 1Gbps and 10Gbps respectively.

From this figure, we see that rather than running at all ports on 10Gbps rate in a common data center network, switch $\langle 1,1\rangle$ in our solution changes its state to some lower data consumption modes which yield a less power consumption along the way. Here we do not present those states maintaining an almost zero state time in Fig. 7.

Besides, we also study latency in deploying our proposed solution, since the switching of ports and port rates may result in a longer delay. In Fig. 8, we show the delay statistics destined to host H_5. The flow end-to-end delay is generally maintained at a very low value. Even the highest delay time in this simulation is considered as acceptable as [12].

Secondly, we test our solution in a series simulations with a more general and realistic settings, and we increase number of

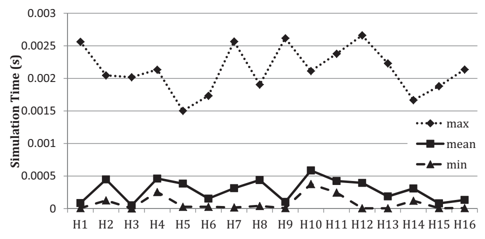

Fig. 8. Max, mean and min End-to-End delay of H_5.

flows to present a general system performance and capacity. In these tests, traffic streams are randomly generated from all of the hosts, H_1 to H_{16}, to all other hosts, with flow frequency (also denoted in terms of packet generation interval) from 0.1ms to 1ms. These flows start randomly from 1s to 6s, and their end time randomly falls into the range in between of their start time to 15s. Our simulation results including the tests on 10, 20, 50, 100, 200 and 300 flows. In these simulations, we focus firstly on calculating the total energy saving for described system given different flow numbers and secondly on obtaining delay statistics to provide a guideline as well as justify the feasibility of our solution.

According to switch consumption we measure in Section II, we can theoretically estimate total energy consumption in our system, based on the switch state statistics yielded by simulations. We list the energy usage in Table V as compared to a common Fat Tree (FT) implementation. The results here are obtained through a procedure to achieve accuracy of calculation. For each number of flows, we first collect the switch state information, sum the time spent in each switch state for each switch in the system, and further integrate states that have the same power consumption. Then we calculate the total power consumption according to power measurement on Dell PowerConnect 8024F switch from the discussions in Section II.

The results we obtained here show that the overall power consumption increases as the number of flows increases, the mere inconsistency may due to unbalance flow distribution from random generation. We see there is no difference on power consumption for sleep mode and 100Mbps on a port in the abovementioned Dell model, and for other data rate, upgrading also does not intrigue significant power change. However, preliminary measurement on other models varies more among link rates, that means, here we just show the worst case results but our results still achieves the goal. We would like to mention here, the difference of energy usage in Table V for 20, 50 flows comes from the randomly ending of

TABLE III
PARAMETER SETTINGS FOR SINGLE FLOW TEST.

Parameter	Value
State changing interval τ	0.3s
Buffer checking interval β	20 packets
Upgrade Threshold T_{up}	50 packets
Downgrade Threshold T_{down}	1 packet

TABLE IV
SIMULATION SETTINGS FOR MULTIPLE NUMBER OF FLOWS TEST.

Parameter	Value
Number of flows	10, 20, 50, 100, 200, 300
Sender	H_1-H_{16}
Receiver	H_1-H_{16}
Flow starts time	1s-6s
Flow ends time	flow's start time to 15s

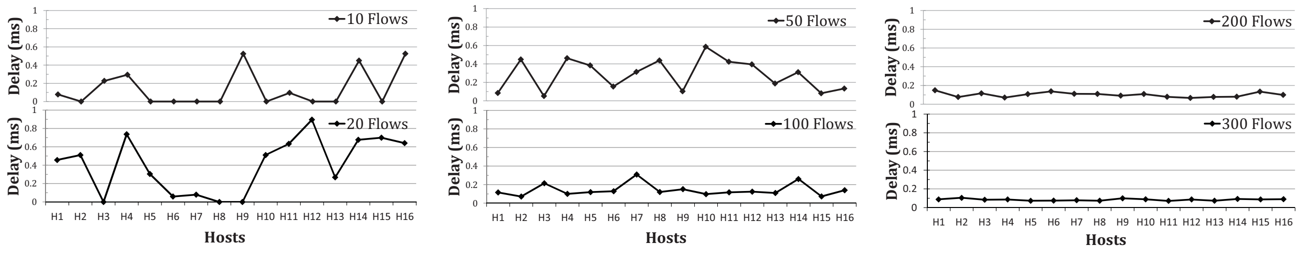

Fig. 9. Hosts delay statistics.

flows in which all flows end before 15s, thus the total usage in Fat Tree are less than the other tests. Figure 9 shows latency statistics which gives mean latency for each host with each number of flows. From the figure, we can see that the end-to-end delay is within a low and constrained range even with heavier traffic load.

From the above simulation results, we conclude that using our proposed solution, the mean delay among all hosts tends to be less as the number of flows increases. This is mainly due to the increase of link rate as traffic increases, thus gives a lesser latency. On the other hand, a higher traffic load gives higher stability of operation with less state transitions and traffic is more evenly distributed. These all account for the smaller and more evenly distributed delay with more number of flows.

V. Conclusions

We have presented the formulation and solution of our work as a novel power management proposal for data center network. By measuring the real power consumption of a switch model and implementing simulation test, we estimated power saved in our proposed solution. In the simulations we observed a saving of 60% in light traffic load, while we did not see much delay incurred by our algorithm, and delays are shortened with increasing number of flows. The design principle of our solution was to explore using of load balancing feature in networks' view and link rate levels in switches' view. In our future work, we will study more models of switches in their power rate and minimize overall power consumption with an algorithmic optimization solution.

References

[1] U. D. of Energy, "Data Center Energy Consumption Trends," http://www1.eere.energy.gov/femp/program/dc_energy_consumption.html, 2010.

[2] R. Bianchini and R. Rajamony, "Power and energy management for server systems," *Computer*, vol. 37, no. 11, pp. 68 – 76, nov. 2004.

[3] E. Pakbaznia and M. Pedram, "Minimizing data center cooling and server power costs," in *Proceedings of the 14th ACM/IEEE international symposium on Low power electronics and design*. ACM, 2009, pp. 145–150.

[4] D. Meisner, B. Gold, and T. Wenisch, "Powernap: eliminating server idle power," *ACM Sigplan Notices*, vol. 44, no. 3, pp. 205–216, 2009.

[5] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. ACM, 2009, pp. 202–208.

[6] D. Abts, M. Marty, P. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," *ACM SIGARCH Computer Architecture News*, vol. 38, no. 3, pp. 338–347, 2010.

[7] T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 92–99, 2010.

[8] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "Elastictree: Saving energy in data center networks," in *Proceedings of the 7th USENIX conference on Networked systems design and implementation*. USENIX Association, 2010, pp. 17–17.

[9] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4. ACM, 2008, pp. 63–74.

[10] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: a scalable and fault-tolerant network structure for data centers," in *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4. ACM, 2008, pp. 75–86.

[11] A. Greenberg, J. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. Maltz, P. Patel, and S. Sengupta, "Vl2: a scalable and flexible data center network," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 51–62, 2009.

[12] C. Gunaratne, K. Christensen, and B. Nordman, "Managing energy consumption costs in desktop pcs and lan switches with proxying, split tcp connections, and scaling of link speed," *International Journal of Network Management*, vol. 15, no. 5, pp. 297–310, 2005.

[13] C. Clos, "A study of non-blocking switching networks," *Bell System Technical Journal*, vol. 32, no. 2, pp. 406–424, 1953.

[14] K. Christensen, P. Reviriego, B. Nordman, M. Bennett, M. Mostowfi, and J. Maestro, "Ieee 802.3 az: the road to energy efficient ethernet," *Communications Magazine, IEEE*, vol. 48, no. 11, pp. 50–56, 2010.

[15] "OMNeT++," http://www.omnetpp.org.

TABLE V
Energy usage comparison.

Number of flows	Energy usage (J)	
	Our solution	FT
10	14009	37800
20	21844	36540
50	21786	36540
100	26250	37800
200	29119	37800
300	29585	37800