INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 2 (June), pp. 241-251

Network Element Scheduling for Achieving Energy-Aware Data Center Networks

W. Fang, X. Liang, Y. Sun, A.V. Vasilakos

Weiwei Fang, Xiangmin Liang, Yantao Sun
School of Computer and Information Technology
Beijing Jiaotong University, Beijing 100044, China
{wwfang, 11125116, ytsun}@bjtu.edu.cn

Athanasios V. Vasilakos
Department of Computer and Telecommunications Engineering
University of Western Macedonia
Kozani GR 50100, Greece
vasilako@ath.forthnet.gr

Abstract: The goal of data center network is to interconnect a massive number of servers so as to provide reliable and scalable computing and storage infrastructure for cloud-based Internet services and data-intensive scientific applications. Recent studies reveal that the network elements consume 10∼20% of the overall power in a data center, which has introduced a challenge to reducing network energy cost without adversely affecting network performance. Considering unique features of traffic patterns and network topologies in data centers, this paper proposes a novel Network Element Scheduling Scheme (NESS) to reduce data center energy consumption from the networking perspective. The core idea is to turn on only a minimal subset of network elements to satisfy routing requirements, and put to sleep or shut down the rest unneeded ones for energy saving. In NESS, the logical network architecture formed by the active elements not only achieves the basic purpose for server interconnections in data centers, but also can support multi-path routing between pairs of hot servers for load balancing. Simulation experiments are performed in representative data center network topologies, and the results demonstrate the effectiveness of NESS in energy conserving on network elements in data centers.

Keywords: data center networks, green computing, energy aware, Steiner tree.

1 Introduction

In recent years, many large data centers are built around the world to provide highly reliable and scalable infrastructure for cloud services and scientific computations. As the actual tendency is to exponentially increase the number of demanded servers, a natural consequence is that the power consumption becomes a critical concern for data center operators. For example, it has been reported that data center power usage in U.S. doubled between 2000 and 2006 to nearly 61 billion kilowatt-hours, representing 1.5% of all U.S. electricity consumption [1]. Researchers are now seeking to find effective solutions to make data centers reduce power consumption while keep the desired service performance. Most of the recent research has focused on reducing the two major components of data center power usage: servers and cooling [2]. However, the underlying network infrastructure, namely routers, switches, high-speed links, still lacks effective energy management solutions. The networking part of data center has been found to consume 10∼20% of its total power consumption [2], and thus should not be neglected.

Fortunately, there is an opportunity for substantial reductions in the energy consumption of data center networks due to two factors. On one hand, the high network capacity of data center networks is specially provisioned for worst-case or busy-hour load, and far from being exceeded by traffic load most of time. Moreover, data center traffic varies considerably over time exhibiting temporal patterns (e.g., daily, weekly, monthly and yearly [3]), and over location exhibiting spatial patterns (e.g., hot/non-hot server racks [4]). In the data center with rich link connectivity, a great number of network elements may consequently work in idle state. On the other hand, today's network elements are not energy proportional, since fixed overheads such as fans, switch chips and transceivers waste power at low loads. It has been found that the energy consumption of common networking devices at the idle state still accounts for more than 85% of that at the working state [5]. The implication of these factors is that significant amount of power energy is wasted on idle elements in the data center network.

To address the challenges stated above, this paper proposes a network element scheduling scheme (NESS) that acts as a network-wide energy optimizer for data center networks. It selects a subset of network elements that must stay active to meet traffic routing requirements, and then puts as many unneeded routers, switches and links as possible into dormant mode. Considering unique features of network topologies and traffic patterns in data centers, NESS achieves this goal in two steps. Firstly, NESS models and solves the basic problem of network element scheduling for guaranteeing server interconnection and traffic routing by using the Steiner tree framework. On the basis of the initial selection result, NESS then additionally activates as few unselected network elements as possible to support different degrees of multi-path routing between server pairs. Network elements not involved in the routing service are finally powered off or put into sleep state. We have conducted extensive simulations in typical architecture models of data center network to illuminate the effectiveness and performance of the proposed scheme.

The rest of this paper is organized as follows. Section 2 presents related works. Section 3 describes the NESS scheme in detail. Section 4 explores on experiment results, and finally the paper is concluded in Section 5.

2 Related Works

The issue considered in this paper involves two network-related research directions, data center networking and green networking. The rest of this section presents the state-of-art related to these two directions.

The research on data center networking mainly focuses on how to implement a network infrastructure that achieves the following goals [2] [6]: (1) it must be scalable to an increasing number of servers; (2) it must be fault tolerant against various types of hardware failures; (3) it must be able to provide high network capacity; (4) it must achieve high utilization and be cost efficient. Due to the limitations of the conventional tree-based architecture [7], a number of novel network architectures for data center networks have been proposed recently, which can be roughly divided into two categories. One is the switch-centric architecture, which organizes switches into structures rather than trees and puts interconnection intelligence on switches, such as Fat-Tree [8], VL2 [7] and Portland [9]. The other is the server-centric architecture, which puts interconnection intelligence on servers and uses switches only as cross-bars, such as BCube [10], FiConn [11] and DCell [6]. Accordingly, each of the architecture proposals has its own solution for node addressing and traffic routing [6] [8]. Furthermore, multi-path routing [6] [8] [12] [13] has been exploited for load balancing in data center networks. Other complementary research for data center networking has focused on TCP incast problem [14], traffic-aware virtual machine migration [15], switch design [16] [17] or cost efficiency [2] [3], etc.

Reduction of unnecessary energy consumption, referred to as "green networking", has become a major concern in wired/wireless networking, because of the potential economical benefits and the expected environmental impact. [18] is a pioneer work on this topic, and the authors of [18] suggested putting network elements to sleep for saving energy in local area network in a later paper [19]. Additionally, link rate adaptation is also employed to reduce energy consumption in Ethernet [20]. Based on these techniques, the IEEE 802.3az Energy Efficient Ethernet task force proposes the Low Power Idle solution to reduce power consumption of Ethernet devices [21]. ElasticTree [3] is a pioneer work that optimizes the energy consumption of data center networks by turning off unnecessary links and switches during off-peak hours. It models the problem based on the multi-commodity flow model, in which some parameters, e.g., server traffic demand, are difficult to be accurately obtained in practice. Besides, ElasticTree focuses on only tree-based topologies such as FatTree. Another similar work [22] models the same problem as a 0-1 Knapsack model, and proposes a heuristic based solution. However, it doesn't support multi-path routing between server pairs for load balancing. VMFlow [23] is a recent work on how to migrate virtual machines among data center servers to minimize the amount of serving network elements while satisfying a large fraction of the network traffic demands.

3 NESS: Network Element Scheduling Scheme

3.1 Research Motivations

A data center network is typically provisioned for peak traffic load, and run well below capacity most of the time. From previous researches, we can discover the two important characteristics (i.e., temporal and spatial) of data center traffics, both of which can help us to define the design goals.

On the one hand, network traffic and its temporal dynamics implicitly reflect the behavioural pattern of end users for whom the data center provide services [3]. For example, traffic may vary daily (e.g., more email exchanging during the day), weekly (e.g., more enterprise data processing on weekdays), monthly (e.g., more multimedia file sharing on holidays), and yearly (e.g., more e-shopping and e-payment in December). Rare events like cable breaks or breaking news may hit the peak capacity, but most of the time data center traffic follows the temporal pattern and actually can be satisfied by a subset of active network elements [3].

On the other hand, physical servers in the data center are commonly organized into racks and richly-interconnected by a number of links, switches and routers. The typical upper-layer applications usually generate a traffic demand with only a few of server racks being hot (i.e., sending or receiving a large volume of traffic) [4]. Moreover, the hot racks generally exchange much of their data with only a few other racks. Such a spatial pattern of data center traffic is determined by the role and tasks of servers in the current application. To avoid network congestion in data centers [4], it is necessary to further provide different degrees of multipath routing for the flows from/to hot servers.

Motivated by the analysis above, we propose NESS, a network element scheduling scheme that acts as a network-wide energy optimizer for data center networks. The following subsections will present the key issues on its design and implementation in detail.

3.2 Design Details

In data center networks, the physical servers are interconnected by a number of high-speed links, switches and even routers. The data center network topology can be modelled as a simple undirected weighted graph *G(V, E)* with equal edge weights [15], where V is the set of vertices and $E \subseteq V \times V$ is the set of edges. There are two types of vertices in V: the servers and the networking devices (i.e., switches and routers). The sets of them can be denoted by V_s and V_d respectively. Therefore, $V = V_s \cup V_d$. The edge $e \in E$ represents a communication link between a server and a networking device, or between a pair of networking devices. As an illustration, we show four typical data center topologies in Figure 1, namely 2N-Tree [3], VL2 [15], Fat-Tree and BCube. It must be noted that in the first three architectures $V_s \cap V_d = \emptyset$, while in BCube $V_s \subset V_d$.

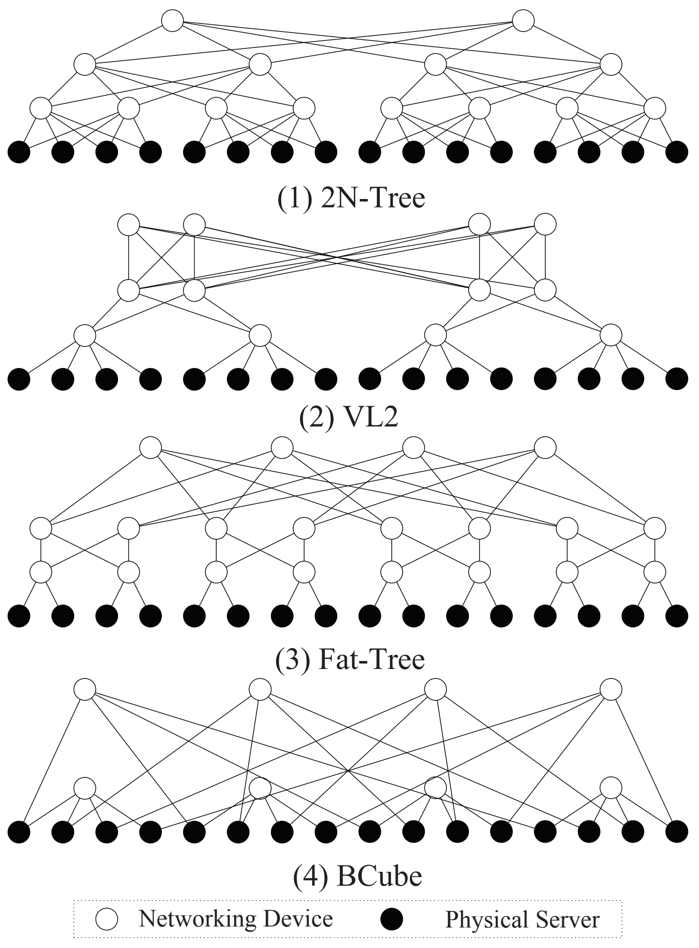

Figure 1: Illustration of state-of-the-art data center topologies.

Before problem formulation, we first define the following notation: $|X|$ denotes the cardinality of set X, $f_{i,j}$ denotes the flow from server i to server j, $p_{i,j}$ denotes the path having i and j as its ends, and $L_{i,j}$ denotes the physically upper bound for total routing paths. Now, the network element scheduling problem can be formally formulated as follows:
Minimize:

$$|V'_d| \tag{1}$$

Subject to:

$$G'(V', E') \subseteq G(V, E) \tag{2}$$

$$V' = V'_d \cup V'_s \tag{3}$$

$$V'_d \subseteq V_d \tag{4}$$

$$V'_s = \{i \in V_s \mid \sum_{\forall j \in V_s} (f_{i,j} + f_{j,i}) > 0\} \tag{5}$$

$$|\{p_{i,j} \mid i,j \in V'_s, \forall k \in p_{i,j}, k \in V'_d\}| \propto \sum(f_{i,j} + f_{j,i}) \tag{6}$$

$$1 \leq |\{p_{i,j} \mid i,j \in V'_s, \forall k \in p_{i,j}, k \in V'_d\}| \leq L_{i,j} \tag{7}$$

In the above formulation, the term of the objective represents the set of active networking devices for interconnections of servers having incoming and/or outgoing traffic. Formula (5)∼(7) guarantee different degrees of multipath routing between pairs of servers can be provided according to traffic requirements. Moreover, Formula (2) implies that unneeded idle links can actually be turned off to further reduce energy consumption [3] [5] since $E' \subseteq E$.

Such a scheduling problem can be easily proved to be NP-complete by restricting that only one path is established between any server pair (i,j). In the equal-edge-weight graph $G(V,E)$, the objective of minimizing $|V'_d|$ is equal to minimizing $|E'|$, and thereby is equal to determining the minimal-weight connected sub-graph $G'(V',E')$ spanning V'_s. This is identical to the Sterner tree problem in the equal-edge-weight graph, which is a NP-complete problem [24]. Therefore, the problem above is NP-complete. Because by now no polynomial-time algorithm is available to obtain the optimal solution of a NP-complete problem, we propose NESS, a scheme that solves the scheduling problem modelled above heuristically in two steps.

In the first step, NESS solves the Steiner tree problem on graph $G(V,E)$ with the terminal node set V'_s. A lot of heuristic algorithms can be used to obtain the results, among which MPH (Minimum Path Heuristic) is one of the best-known solution [24] [25]. The output vertex set V'_{d1} is a subset of V_d, representing the minimal number of networking devices that can guarantee the basic purpose for server interconnection and traffic routing.

In the second step, NESS selects from $V_d \backslash V'_{d1}$ as few vertices as possible to construct multiple path between some vertices in V'_s that represents the hot servers. To achieve this goal, NESS integrates the well-known ECMP (Equal-Cost Multiple-Path) routing mechanism [12] for discovering multiple paths connecting hot servers, and then chooses the paths containing more vertices already in the set V'_{d1}. Denoting the set of newly selected vertices in this step as V'_{d2}, we finally have $V'_d = V'_{d1} \cup V'_{d2}$.

To illustrate the approach stated above, we take an example in the Fat-Tree topology shown in Figure 2. Assume $V'_s = \{v_{21}, v_{25}, v_{27}, v_{28}\}$, and v_{21}, v_{28} are hot servers requiring two routing paths to be established between them. In the first step, NESS executes the MPH algorithm to solve the Steiner tree problem, and obtains $V'_{d1} = \{v_1, v_5, v_7, v_{13}, v_{15}, v_{16}\}$. In the second step, NESS can discover totally four equal-cost shortest paths through a basic version of ECMP [8] as follows:

$$\begin{aligned} path1 &: v_{21} \to v_{13} \to v_5 \to v_1 \to v_7 \to v_{16} \to v_{28} \\ path2 &: v_{21} \to v_{13} \to v_5 \to v_2 \to v_7 \to v_{16} \to v_{28} \\ path3 &: v_{21} \to v_{13} \to v_6 \to v_3 \to v_8 \to v_{16} \to v_{28} \\ path4 &: v_{21} \to v_{13} \to v_6 \to v_4 \to v_8 \to v_{16} \to v_{28} \end{aligned}$$

Then we can choose path1 and path2 for v_{21}, v_{28} since path2 requires to additionally activate only v_2, while path3 and path4 requires to activate v_6, v_3, v_8 and v_6, v_4, v_8 respectively. Therefore, $V'_{d2} = \{v_2\}$.

3.3 Implementation Issues

As a scheduling software, NESS consists of four logical modules, i.e., network analyzer, network scheduler, power controller and route controller. The network analyzer obtains network topology and traffic requirement by performing statistics and analysis to the collected running data of the data center network. The role of network scheduler is to find the minimum network subset that can satisfy current traffic according to the approach stated in Section 3.2. With

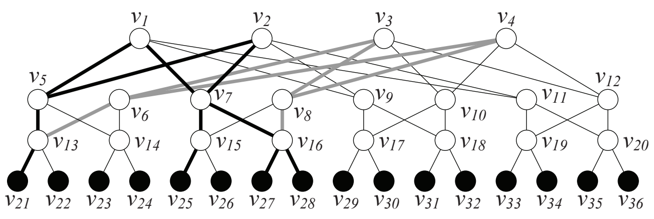

Figure 2: An Fat-Tree example for illustrating network element scheduling in NESS.

the input of topology and traffic conditions from the analyzer, the scheduler outputs the set of active elements to power controller and the set of flow routes to route controller. The power controller toggles the power states of different types of network elements (i.e., links, switches and routers), while the route controller checks routing paths for traffic flows and pushes routes into the network.

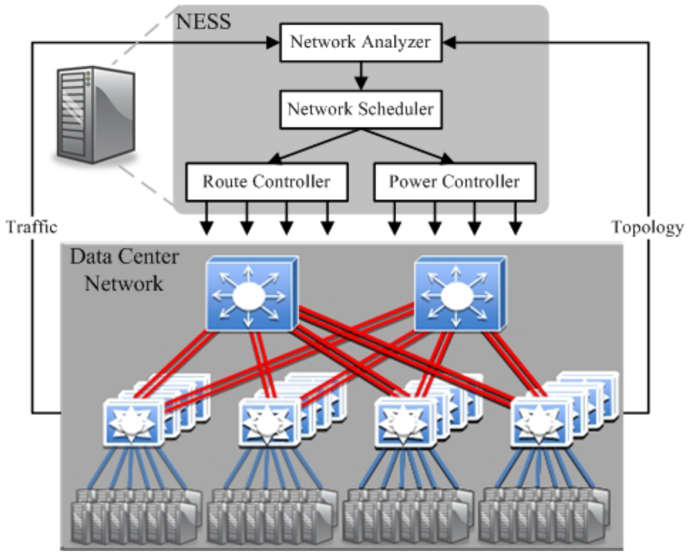

Figure 3: System diagram of NESS.

Similar to ElasticTree [3], NESS can be implemented as a NOX application [26] to run atop a network of OpenFlow switches [27]. OpenFlow is an open standard for commercial switches and routers to enable controlling the forwarding plane by a software running on a separate server, and NOX is an open-source OpenFlow controller that is designed to provide a simplified platform for writing network control software in C++ or Python. The route controller in NESS can be implemented with NOX. Besides, we can leverage the existing mechanisms such as SNMP Set operations and command line interface to support the power control features of NESS. Moreover, the network analyzer can collect the state records of traffic and topology through SNMP Get operations and passive packet tracing [28].

4 Experimental Evaluation

In this section, we evaluate the performance of NESS experimentally, using a custom simulator developed in C++ with the Boost Graph Library [29]. This simulator supports constructing the four types of data center network architectures described in Section 3.2 with a number of 48-port Gigabit Ethernet switches [8] [27]. By using this simulator, we have created data center communication scenarios with different network sizes (e.g., a network of S=2304/4608/13824 end servers), different traffic conditions (e.g., a network of servers with h=1%/10% hot ones [4] in which each hot server randomly selects about 10% other ones under different racks as its communication counterparts) and different routing requirements (e.g., a pair of hot servers may require 2∼6 equal-cost paths for traffic multiplexing). For a specific scenario, the simulation has been carried out independently for multiple times by network reconstruction, and the results are averaged over these runs. Figure 4 shows the percentage of the dormant network elements in 2N-Tree, VL2, Fat-Tree and BCube respectively.

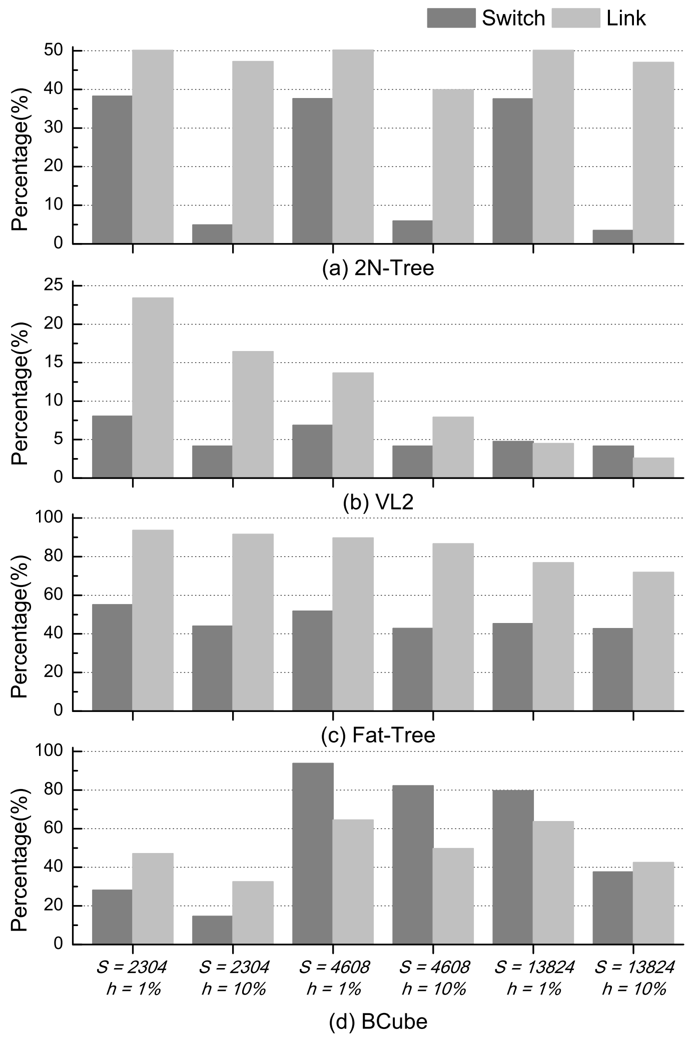

Figure 4: The percentage of dormant network elements in six different networking scenarios.

We have several observations concerning the simulation results:

(1) NESS effectively reduces the total amount of active switches as well as active links to different extents in typical communication scenarios under the four data center network architectures.

(2) All else being equal, the communication scenario with more hot servers generally will have greater demand for network resources, i.e., more network elements have to be kept active by NESS for fulfil routing requirements.

(3) For 2N-Tree, we can find that its dormant percentages of network elements remain essentially flat under the same condition of hot rate h. That's because in 2N-Tree both the total

amount of network elements (i.e., switches or links) and the amount of the ones selected by NESS are approximately in direct proportion to the network size. In general, our simulation results on 2N-Tree (Figure 4 (a)) indicate that we can roughly predict the energy saving for applying NESS to the large-scale data center network built in this architecture through small-scale experiments.

(4) For VL2, we can find that its dormant percentages of both switch and link are the lowest among the four architectures. An examination of simulation traces reveals that the total amount of switches remains the same (i.e., 336) for all the six simulation scenarios, while most of these switches (i.e., 288) and the links from them to the servers have to be kept active. Therefore, the optimization space for NESS is quite limited in VL2 architecture, especially when the hot rate h is relatively high. From the results in Figure 4 (b), the scenario with fewer hot servers has higher dormant percentage of switches, especially when the number of hot servers is less than 230 (i.e., 10% of 2304). Moreover, the dormant percentage of links decreases accordingly along with the increment of network size.

(5) For Fat-Tree, we can find that that it has the relatively highest dormant percentage of links among the four architectures. Besides, the dormant percentage of switches is also relatively high (i.e., about 50%) for each scenario. An examination of simulation traces reveals that Fat-Tree has the most switches (i.e., 2880) and links (i.e., 50000∼70000) among the four architectures. While such redundancy of available network elements makes the fat-tree topology attractive for fault-tolerance, it also brings about considerable energy wastage due to element idle in most cases [3] [22]. Moreover, as that of VL2, all scenarios have the same amount of switches and the one with fewer hot servers will have higher dormant percentage of switches. In all, the simulation results in Figure 4 (c) indicate that there is a tradeoff between fault tolerance and energy consumption for Fat-Tree.

(6) For BCube, the simulation results are significantly different from those of switch-centric architectures, i.e., the dormant percentages increase sharply when S grows lager than 2304. That's because: when S=2304, a 2-level BCube$_1$ built with totally 96 switches (48 for level 0 and 48 for level 1) is enough. However, a 3-level BCube$_2$ has to be constructed with 2304 switches for level 2 when S>2304. All these level 2 switches only interconnects two BCube$_1$ when S=4608 and six BCube$_1$ when S=13824. Actually, the utilization of these switches are quite low in result of high redundancy, and thus most of them can be put into dormant state. Moreover, it can also be found that the increment of network size under the same condition of BCube level, e.g., from 4608 to 13824, will lead to higher element utilization and lower dormant percentages. In all, the simulation results in Figure 4 (d) indicate that for each BCube level the dormant percentages of network elements will decrease from high to low along with the increment of network size.

Accordingly, Table 1 specifically illustrates to how much extent power energy can be saved by applying NESS to data center networks. The power consumption parameters of network elements are obtained from a previous work [5]. For the three switch-centric architectures above, $P_{VL2}<P_{2N-Tree}<P_{Fat-Tree}$. For the sever-centric architecture BCube, the scenario with a smaller network size at a BCube level (e.g., S=4608 at BCube$_2$) will have higher reduction percentage of power consumption due to high redundancy of network elements.

5 Conclusion

In this paper, we introduce the energy cost problem in today's data centers. Based on detailed analysis on traffic patterns and network topologies, we propose a novel scheme (NESS) to reduce data center energy consumption from the networking perspective. Extensive simulation results demonstrate that it is possible to switch off idle networking elements, so that the total network power consumption can be reduced without adversely affecting network performance. The work presented in this paper is somewhat preliminary, but shows that energy-aware networking in the

Table 1: Power Consumption Reduction Percentage P.

S	2304		4608		13824	
h	1%	10%	1%	10%	1%	10%
2N-Tree	43.99%	25.44%	43.74%	22.50%	43.68%	24.78%
VL2	10.66%	6.26%	8.59%	5.12%	4.63%	3.43%
Fat-Tree	66.08%	57.59%	62.87%	55.69%	55.56%	52.18%
BCube	37.40%	23.34%	80.87%	78.97%	76.09%	38.69%

data center is promising [30] [31]. In the future, we plan to implement NESS on our prototype system of data center network, which is still under development now.

Acknowledgements

This work was partially supported by the China Postdoctoral Science Foundation (20110490282), the Fundamental Research Funds for the Central Universities of China (2011JBM225) and the Intel China Research Institute - BJTU Collaborative Research Program (K09L00030).

Bibliography

[1] U.S. Enviromental Protection Agency (EPA), Report to Congress on Server and Data Center Energy Efficiency: Public Law 109-431, *EPA ENERGY STAR Program*, August, 2007

[2] A.G. Greenberg, P. Lahiri, D.A. Maltz, P. Patel and S. Sengupta, The Cost of a Cloud: Research Problems in Data Center Networks, *ACM SIGCOMM Computer Communication Review*, 39(1):68-73, 2009

[3] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee and N. Mckeown, ElasticTree: Saving Energy in Data Center Networks, *Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 249-264, 2010

[4] D. Halperin, S. Kandula, J. Padhye, P. Bahl and D. Wetherall, Augmenting Data Center Networks with Multi-Gigabit Wireless Links, *Proceedings of the 2011 ACM SIGCOMM conference*, 38-49, 2011

[5] A.P. Bianzino, C. Chaudet, F. Larroca, D. Rossi and J.L. Rougier, Energy-Aware Routing: a Reality Check. *Proceedings of the 2010 IEEE GLOBECOM workshops*, 1422-1427, 2010

[6] C.X. Guo, H.T Wu, K. Tan, L. Shi, Y.G. Zhang and S.W. Lu, DCell: A Scalable and Fault-Tolerant Network Structure for Data Centers. *Proceedings of the 2008 ACM SIGCOMM conference*, 75-86, 2008

[7] A. Greenberg, J.R. Hamilton and N. Jain, VL2: A Scalable and Flexible Data Center Network. *Proceedings of the 2009 ACM SIGCOMM conference*, 51-62, 2009

[8] M. Al-Fares, A. Loukissas and A. Vahdat, A Scalable, Commodity Data Center Network Architecture. *Proceedings of the 2008 ACM SIGCOMM conference*, 63-74, 2008

[9] R.N. Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya and A. Vahdat, PortLand: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric. *Proceedings of the 2009 ACM SIGCOMM conference*, 39-50, 2009

[10] C.X. Guo, G.H. Lu, D. Li, H.T. Wu, X. Zhang, Y.F. Shi, C. Tian, Y.G. Zhang and S.W. Lu, BCube: A High Performance, Server-Centric Network Architecture for Modular Data Centers. *Proceedings of the 2009 ACM SIGCOMM conference*, 63-74, 2009

[11] D. Li, C.X. Guo, H.T. Wu, Y.G. Zhang and S.W. Lu, Ficoon: Using Backup Port for Server Interconnection in Data Centers. *Proceedings of the 28th IEEE International Conference on Computer Communications (INFOCOM)*, 2276-2285, 2009

[12] C. Hopps, Analysis of an Equal-Cost Multi-Path Algorithm. *IETF RFC 2992*, November 2000

[13] K. Xi, Y.L. Liu and H.J. Chao, Enabling Flow-based Routing Control in Data Center Networks using Probe and ECMP. *Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM) Workshops*, 614-619, 2011

[14] J. Zhang, F.Y. Ren and C. Lin, Modelling and Understanding TCP Incast in Data Center Networks. *Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM)*, 1377-1385, 2011

[15] X.Q. Meng, V. Pappas and L. Zhang, Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement. *Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM)*, 1154-1162, 2010

[16] D.A. Joseph, A. Tavakoli and I. Stoica, A Policy-aware Switching Layer for Data Centers. *ACM SIGCOMM Computer Communication Review*, 38(4): 51-62, 2008

[17] N. Farrington, G. Porter, S. Radhakrishnan, H.H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen and A. Vahdat. Helios: A Hybrid Electrical/Optical Switch Architecture for Modular Data Centers. *Proceedings of the 2010 ACM SIGCOMM conference*, 339-350, 2010

[18] M. Gupta and S. Singh, Greening of the Internet. *Proceedings of the 2003 ACM SIGCOMM conference*, 19-26, 2003

[19] M. Gupta and S. Singh, Using Low-Power Modes for Energy Conservation in Ethernet LANs. *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM)*, 2451-2455, 2007

[20] C. Gunaratne, K. Christensen, B. Nordman and S. Suen, Reducing the Energy Consumption of Ethernet with Adaptive Link Rate (ALR). *IEEE Transactions on Computers*, 57(4): 448-461, 2008

[21] R. Hays, Active/Idle Toggling with Low-Power Idle. *Presentation for IEEE 802.3az Task Force*, 2008

[22] Y.F. Shang, D. Li and M.W. Xu, Energy-Aware Routing in Data Center Network. *Proceedings of the 1st ACM SIGCOMM Workshop on Green Networking 2010*, 1-8, 2010

[23] V. Mann, A. Kumar, P. Dutta and S. Kalyanaraman, VMFlow: Leveraging VM Mobility to Reduce Network Power Costs in Data Centers. *Lecture Notes in Computer Science Volume 6640 (IFIP Networking 2011)*, 198-211, 2011

[24] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. *W. H. Freeman Publishers Company*, 1979

[25] H. Takahashi, A. Matsuyama, An Approximate Solution for the Steiner Problem in Graphs. *Mathematica Japonicae*, 24: 571-577, 1980

[26] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. Mckeown, S. Sehnker. NOX: Towards an Operating System for Networks. *ACM SIGCOMM Computer Communication Review*, 38(3): 105-110, 2008

[27] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Perterson, J. Rexford, S. Shenker, J. Turner, OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communication Review*, 38(2): 69-74, 2008

[28] T. Benson, A. Anand, A. Akella, M. Zhang, Understanding Data Center Traffic Characteristics. *ACM SIGCOMM Computer Communication Review*, 40(1), 92-99, 2010.

[29] The Boost Graph Library, *http://www.boost.org/doc/libs/release/libs/graph.*

[30] D. Abts, M.R. Marty, P.M. Welles, P. Klausler, H Liu, Energy Proportional Datacenter Networks. *Proceedings of the 37th ACM/IEEE International Symposium on Computer Architecture (ISCA)*, 338-347, 2010

[31] K. Chen, C.C. Hu, X. Zhang, K. Zheng, Y. Chen, A.V. Vasilakos, Survey on Routing in Data Centers: Insights and Future Directions. *IEEE Network*, 25(4), 6-10, 2011