Computer Networks 57 (2013) 179–196

Contents lists available at SciVerse ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

VMPlanner: Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers

Weiwei Fang [a,*], Xiangmin Liang [a], Shengxin Li [a], Luca Chiaraviglio [b], Naixue Xiong [c]

[a] School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China
[b] Department of Electronics, Politecnico di Torino, Turin 10129, Italy
[c] School of Computer Science, Colorado Technical University, Colorado Springs, Colorado 80907, USA

ARTICLE INFO

Article history:
Received 5 March 2012
Received in revised form 22 August 2012
Accepted 12 September 2012
Available online 23 September 2012

Keywords:
Data center
VM placement
Green networking
Energy efficiency

ABSTRACT

In recent years, the power costs of cloud data centers have become a practical concern and have attracted significant attention from both industry and academia. Most of the early works on data center energy efficiency have focused on the biggest power consumers (i.e., computer servers and cooling systems), yet without taking the networking part into consideration. However, recent studies have revealed that the network elements consume 10–20% of the total power in the data center, which poses a great challenge to effectively reducing network power cost without adversely affecting overall network performance. Based on the analysis on topology characteristics and traffic patterns of data centers, this paper presents a novel approach, called VMPlanner, for network power reduction in the virtualization-based data centers. The basic idea of VMPlanner is to optimize both virtual machine placement and traffic flow routing so as to turn off as many unneeded network elements as possible for power saving. We formulate the optimization problem, analyze its hardness, and solve it by designing VMPlanner as a stepwise optimization approach with three approximation algorithms. VMPlanner is implemented and evaluated in a simulated environment with traffic traces collected from a data center test-bed, and the experiment results illustrate the efficacy and efficiency of this approach.

1. Introduction

Cloud computing is now driving the creation of data centers that hold a large number of computer servers and that support a variety of online applications (e.g., web searches like Google or social networks like Facebook) as well as infrastructural services (e.g., distributed file systems like HDFS or distributed execution engines like MapReduce) [1]. The motivations for establishing such massive data centers are both economic and technical: to leverage the economics of scale to amortize the total cost of bulk deployments and to benefit from the ability to dynamically reallocate resources among services in the case of workload changes or equipment failures. The cost is also quite large, e.g., it has been reported that data center power usage in US doubled between year 2000 and 2006 to nearly 61 billion kW h (1.5% of total US electricity consumption), and is projected to double again by 2011 to more than 100 billion kW h [2]. The computer servers and cooling systems have been found as the top two power consumers in current data centers, and most research efforts focus on making them even more energy efficient. In contrast, the underlying network infrastructure, namely routers, switches and high-speed links, has received little attention because today's network elements altogether take up a relatively small proportion (about 10–20%) of a data center's energy budget [3]. With energy management techniques for servers and cooling well in place, by Amdahl's Law, the share of data center power consumed by network elements is expected to grow rapidly over

* Corresponding author. Tel.: +86 13 641168371; fax: +86 10 51688152.
E-mail addresses: wwfang@bjtu.edu.cn (W. Fang), 11125116@bjtu.edu.cn (X. Liang), 09281118@bjtu.edu.cn (S. Li), chiaraviglio@tlc.polito.it (L. Chiaraviglio), nxiong@coloradotech.edu (N. Xiong).

1389-1286/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comnet.2012.09.008

the next few years. In additional to the power bills, the large power consumption will also put a lot of stress on power delivery to and heat removal from network elements as well as the hosting facility. Thus, it is the high time to address the challenge for reducing the network power cost while maintaining the network system performance in cloud data centers.

Power management in computers has evolved around hardware support for power performance and idle sleep states [4]. The former (e.g., P-states of CPU) intends to save power during active times, by lowering down operating frequency to different scales, while the latter (e.g., C-states of CPU) intends to save power during idle times, by powering down sub-components to different extents. Such a design philosophy is also followed by researchers for reducing network energy consumption [5,6]. Unfortunately, today's networking devices are not designed to be energy proportional (i.e., the amount of energy consumed is proportional to the offered load), since the fixed overheads such as chips, transceivers and fans waste power at low loads. It has been found that the energy consumption of network elements at the idle state still accounts for more than 85% of that at the working state [7]. In short, there are still lots of work to be done to make rate adaptation more practical for power saving on network elements [8]. On the other hand, most of the early studies on idle sleep have treated routers and switches as isolated devices, and focused on reducing power consumption at the node level by putting idle components (e.g., line cards and ports) to sleep [5,9]. However, how to handle idle-time traffic on behalf on the sleeping devices to preserve their network presence so far still remains as a challenging problem for node level solutions [10].

Considering the high path redundancy and low link utilization in today's large networks (especially data center networks), the network-level solutions on sleep scheduling of network elements for power saving have been explored by researchers in recent years [11,12]. By aggregating and switching traffic onto fewer number of paths, a network operator can free some devices and links from carrying data traffic and put them to sleep for energy conservation. This type of solution requires network-wide coordination of both traffic demands and network resources. The challenges are twofold, namely how to optimize traffic distribution (if possible) and flow routing to make as many unneeded network elements as possible to maximize the power conservation, and how to effectively achieve power conservation without adversely affecting network performance.

To address the challenges stated above, this paper proposes VMPlanner, a network-wide power manager that optimizes virtual machine (VM) placement and traffic flow routing to reduce data center power costs by sleep scheduling of network elements. Compared with existing power management schemes, the main contributions of this work are listed as follows:

- We address the power-saving issue of data center networks with network-aware VM placement and power-aware traffic routing. We formulate it as a combinatorial optimization problem, and show that it is NP-complete.
- We decompose the optimization problem, and find it in fact consists of three classic NP-complete problems, namely: (1) traffic-aware VM grouping (Balanced Minimum K-cut Problem [13], BMKP), (2) distance-aware VM-group to server-rack mapping (Quadratic Assignment Problem [14], QAP), and (3) power-aware inter-VM traffic flow routing (Multi-Commodity Flow Problem [12], MCFP). Accordingly, VMPlanner is designed to integrate efficient approximation algorithms for solving the NP problems. Furthermore, we provide considerations on how to implement VMPlanner in practice with OpenFlow technology [15].
- We develop a custom simulator for evaluating VMPlanner under different network topologies and different traffic scenarios with traffic traces from a data center test-bed [16]. Extensive simulation results illuminate the distinguished performance of VMPlanner.

The rest of this paper is organized as follows. Section 2 presents related works. Section 3 describes the design details of VMPlanner, and Section 4 discusses potential implementation issues. Section 5 evaluates VMPlanner using typical network topologies and real traffic traces. Finally, the paper is concluded in Section 6.

2. Related works

In this section, we briefly review previous works related to this work, namely data center networking, green data center, and network-aware VM placement.

2.1. Data center networking

The goal of data center network is to interconnect a massive number of servers with networking devices (e.g., switches) and high-speed links. Conventional data centers follow to a great extend a common network architecture that is known as the three-tier architecture [12,17]. At the bottom level (known as the access tier), servers are organized in racks, and each server in a rack connects to one (or two, for redundancy) Top-of-Rack (ToR) switches. Each ToR switch connects to one (or two) switches at the aggregation tire, and finally, each aggregation switch connects with multiple switches at the top level (known as the core tier). The topology scaling limitations [12] as well as other problems [17] such as limited inter-sever capacity, fragmentation of resources and poor resource utilization, have prompted recently many parallel efforts in redefining the network architecture of the data centers. Most new architecture designs [18], e.g., VL2 [17] and Fat-Tree [19], share similar richly-connected topologies (typically a form of Clos network [20]), differing more on how addressing and routing are implemented.

2.2. Green data center

Greening data centers is becoming an increasingly important topic for cloud service providers in recent years, due to the high operational expenditure for power consumption as well as the concerns for global warming and

energy crisis. A lot of research efforts have been put on power management techniques for data center servers [21], including chip multiprocessing, dynamic voltage and frequency scaling, sleep scheduling, and VM management. Meanwhile, smart cooling solutions are proposed for data centers to improve the cooling effectiveness while reducing the power costs for cooling [22,23].

Recently, some works have focused on reducing the power consumption of network elements [24,25], especially those in cloud data centers. ElasticTree [12] consolidates traffic flows in the data center network onto a small set of switches and links such that unused network elements can be turned off for power saving. Shang et al. [26] propose a routing technique for reducing the number of switches used for routing a given traffic matrix over a given data center topology. Neither of these two approaches takes VM placement and migration into consideration. More recently, Mahadevan et al. [27] present Urja, a system for monitoring and analyzing network traffic and power usage. Based on the measurement results, the authors suggest various possible approaches, such as network traffic aggregation, server load consolidation and VM placement optimization, for moving towards an energy proportional data center network. However, the design details on how to implement the techniques in practice are not explicitly given in this work.

2.3. Network-aware VM placement

In virtualization-based data centers, the problem of VM placement has become a crucial issue and attracted significant attention. Traditional works [28,29] mostly focus on consolidate VMs for improving server resource (e.g., CPU, memory and disk) utilization and reducing server power consumption, yet without taking network topology and network traffic into consideration. Some recent research works have studied VM placement for optimizing network traffic in data centers. Meng et al. [30] investigate traffic-aware virtual machine placement in data center networks. Given the traffic matrix in a data center and the communication cost between every pair of servers, the paper presents algorithms for placing VMs at the servers such that the total communication cost is minimized. However, this work does not consider network power optimization. McGeer et al. [31] propose to use techniques such as traffic aggregation and VM placement while simultaneously turning off as much switches as possible to save network power. The placement problem is modelled as the well-known VLSI Global Routing problem, and solved by a classic linear-time heuristic algorithm. However, this work is designed specially for the networks with a Fat-Tree topology. More recently, Mann et al. [32] formulate the VM placement and routing of traffic demands for reducing network power as an optimization problem, and propose a greedy heuristic solution called VMFlow. However, VMFlow assumes that the VM placement mapping is one-to-one, i.e., at most one VM or a VMset (i.e., a group of VMs that are consolidated on a server and will not be separately migrated to different severs) can be mapped to a server. This assumption does not hold well in practice.

3. VMPlanner design

3.1. Research background and motivation

Today's data center networks have been designed and operated with little considerations of energy efficiency. They are typically provisioned with redundant links and excessive bandwidth for accommodating peak traffic loads and potential link failures, and run well below capacity most of the time. Fig. 1 presents a simple illustration of network topologies for three data center network architectures, namely Fat-Tree [19], VL2 [17], and VL2N-Tree adopted by our data center [16]. In particular, VL2N-Tree is a variant of the traditional 2N-Tree topology [12] in which the core tier and the aggregation tier form a VL2-like Clos topology. It is noticeable that in these mesh-like topologies a group of switches at the same tier are fully connected by links with another group of switches at the neighbouring tier, so that multiple paths can be established in the network for routing inter-server traffic. As the switch group size is proportional to the switch port count, real data center networks constructed by enterprise-grade switches with tens of or hundreds of ports can provide a much higher degree of link redundancy and path redundancy. Take Fat-Tree for an example, there are $k^2/4$ shortest-hop paths between any two servers on different pods when the k-port switches are used [19].

By contrast with the capacity over-provisioning, the link utilization in data center networks has been found very low during most of the time (even as high as 40% of links are unused and idle [33]), and the network capacity is generally far from being exceeded by traffic load [12]. Thus, it is reasonable to infer that significant amount of energy is wasted on the idle network elements which are still not energy proportional for now. These findings provide unique opportunities for power-aware traffic engineering in data center networks. Intuitively, when there are multiple paths between the same server pair, and the traffic volume of each inter-server flow is low, one can move and aggregate network traffic onto a fewer number of paths so that the remaining links and even switches that do not carry any traffic can be put into dormant state for energy conservation. While this intuition is relatively straightforward, the issue of how to turn off as many network elements as possible to minimize the total network power is not straight forward.

Generally, we can address this problem from two interlinked perspectives: traffic demand and capacity supply. On the one hand, the aggregate traffic rates perceived by every switch could be minimized so as to potentially reduce the total traffic demands on the provided path capacity. Such an optimization objective can be achieved by the VM migration technique, since in data centers the network applications are deployed commonly on virtual servers but

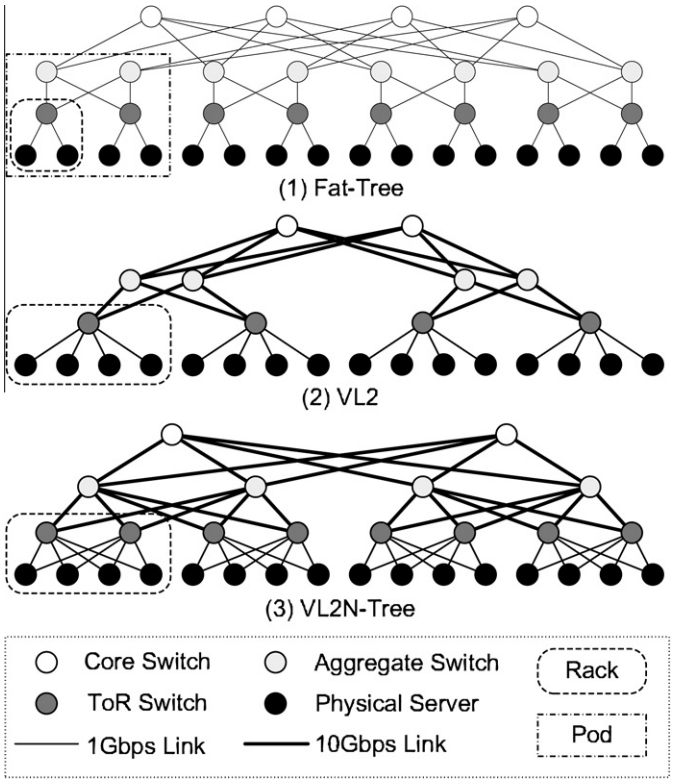

Fig. 1. Illustration of three data center network topologies.

not directly on physical servers [28]. By optimizing the placement of VMs on host servers, the traffic distribution among VMs can be better aligned with the communication distance between them, e.g., VMs with large mutual bandwidth usage will be assigned to host servers in close proximity. On the other hand, the final paths selected for routing traffic flows between different server pairs should share as many switches and links in common as possible, so long as the total flows assigned on each link do not exceed the link capacity. Such an optimization objective can be achieved by the policy-based routing technique, which has already been used in data centers for efficient use of network resources [15].

To show how the joint optimizations described above can help to minimize the network power cost, we use a simple example of communication in a Fat-Tree pod for illustration purpose, reported in Fig. 2. In this scenario, there are four physical servers in which each one is able to host one of the VMs from v_1, v_2, v_3, v_4. The inter-VM traffic volumes are as follows: $f(v_1, v_3) = f(v_2, v_4) = 0.7$ Gbps, $f(v_1, v_2) = f(v_3, v_4) = 0.1$ Gbps. Before optimization, all of the switches and links are required to stay active so as to provide enough path bandwidth for traffic routing. After optimizing the VM placement and the traffic routing, one switch and two inter-switch links become idle and can actually be put to sleep, while the remaining network elements

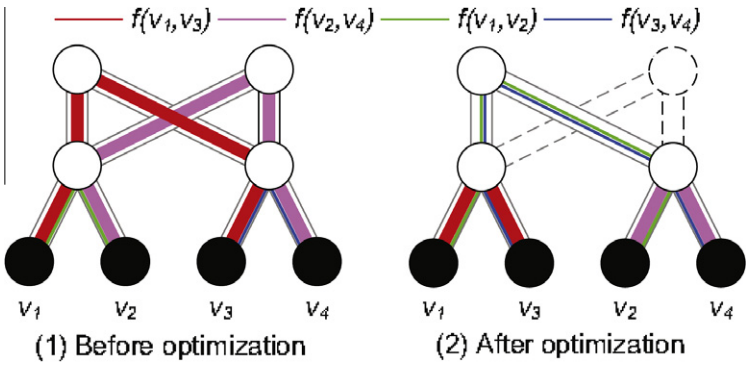

Fig. 2. A motivation example showing the optimization result.

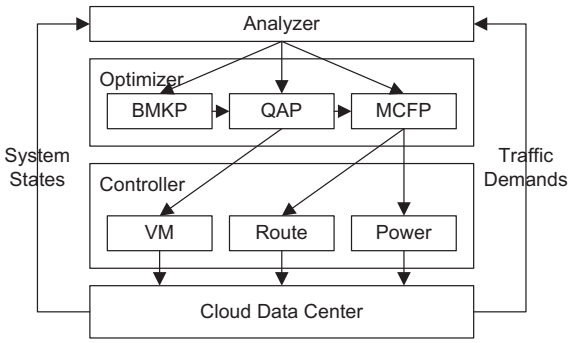

Fig. 3. System diagram of VMPlanner.

Table 1
Simulation parameters for the three network architectures.

	Fat-Tree	VL2	VL2N-Tree
$\|Y\|$	1024	1024	1024
n	2	2	2
r	8	16	16
K	128	64	64
Switch			
Core	64	8	8
Aggr	128	16	16
ToR	128	64	128
Link			
Core-Aggr	1024	128	128
Aggr-ToR	1024	128	128
ToR-Server	1024	1024	2048

are sufficient to satisfy the bandwidth demands from network traffic.

Due to the complexity of both network topology and traffic distribution in data centers, performing network-wide optimization is not as simple and easy as in this example. In the following, we will formally formulate the optimization problem, and then present our solution VMPlanner.

3.2. Problem formulation and analysis

The data center network topology can be modelled as a simple undirected graph $G(V,E)$, 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 physical servers and the network switches. The set of them can be denoted by Y and Z respectively. Therefore, $V = Y \cup Z$. The edge $e \in E$ represents a communication link between a server and a switch, or between a pair of switches. The bandwidth capacity of e is denoted by $C(e)$.

We consider a common scenario in which a set of VMs X is waiting to be placed on the physical servers in set Y. It is assumed that $|Y| \geqslant |X|/n$, where $|Y|$ denotes the cardinality of Y, and $n \geqslant 1$ is the number of VMs that a physical server can accommodate. For any placement scheme that assigns $|X|$ VMs to physical servers on a many-to-one (n:1) basis, there is a corresponding permutation function $\pi : X \to Y$. We are given a set of D traffic demands (flow rates) among the applications deployed in VMs from X, and the traffic rate from VM $i \in X$ to $j \in X (i \neq j)$ is denoted by $d_{i,j} \in D$. The traffic flows from the server I hosting i (i.e., $I = \pi(i)$) to the server J hosting j (i.e., $J = \pi(j)$) that is routed through edge e is denoted by $f^e_{I,J} (I \neq J)$. The routing mechanism has to ensure that a single path $P_{I,J} (I \neq J)$ composed of a connected set of unique edges is traversed for each traffic demand $d_{i,j}$.

Let T^e_v and H^e_v denote the binary decision variable indicating whether $v \in V$ is the tail or head node of link e (in the direction of $P_{I,J}$), respectively. Besides, let $p_s(z)$ and $p_l(e)$ denote the power consumption of switch z and link e, while $B_s(z)$ and $B_l(e)$ denote the binary decision variables indicating whether switch z and link e is powered on, respectively. The general optimization problem can be formulated as follows:

$$\textbf{Minimize} \sum_{z \in Z} p_s(z) B_s(z) + \sum_{e \in E} p_l(e) B_l(e) \tag{1}$$

subject to the following constraints:

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

$$V' = Y' \cup Z' \tag{3}$$

$$Y' \subseteq Y \tag{4}$$

$$Z' \subseteq Z \tag{5}$$

$$\pi(i) \in Y', \quad \forall i \in X \tag{6}$$

$$|\{i | \pi(i) = I, i \in X\}| \leqslant n, \quad \forall I \in Y' \tag{7}$$

$$B_s(z) = \begin{cases} 1, & \forall z \in Z' \\ 0, & \forall z \in Z - Z' \end{cases} \tag{8}$$

$$B_l(e) = \begin{cases} 1, & \forall e \in E' \\ 0, & \forall e \in E - E' \end{cases} \tag{9}$$

$$\sum_{e \in E} f^e_{I,J} T^e_v - \sum_{e \in E} f^e_{I,J} H^e_v = \begin{cases} \sum_{\substack{\pi(i)=I \\ \pi(j)=J}} d_{i,j}, & v = J \\ -\sum_{\substack{\pi(i)=I \\ \pi(j)=J}} d_{i,j}, & v = I, \quad I \neq J, \ \forall i, j \in X, \ \forall v \in V \\ 0, & v \neq I, J \end{cases} \tag{10}$$

$$\sum_{I,J \in Y} f^e_{I,J} \leqslant C(e) \times B_l(e), \quad I \neq J, \quad \forall e \in E \tag{11}$$

$$B_l(e) \leqslant B_s(z) \leqslant \sum_{e \in E} B_l(e), \quad T^e_z + H^e_z = 1, \quad \forall z \in Z, \quad \forall e \in E \tag{12}$$

In the above formulation, the term of the objective represents the total network power consumption. Constraints (2)–(5) imply that a subset of physical servers (i.e., Y') are selected for hosting the VMs in X while a subset of network elements (i.e., Z' and E') are selected for satisfying traffic demands in D. Constraint (6) states that any virtual machine in X must be hosted by a physical server in Y'.

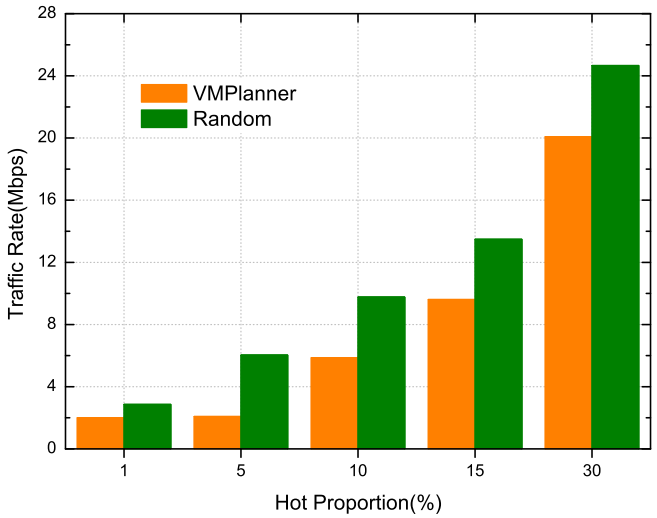

Fig. 4. Average inter-group traffic volume ($K = 128$).

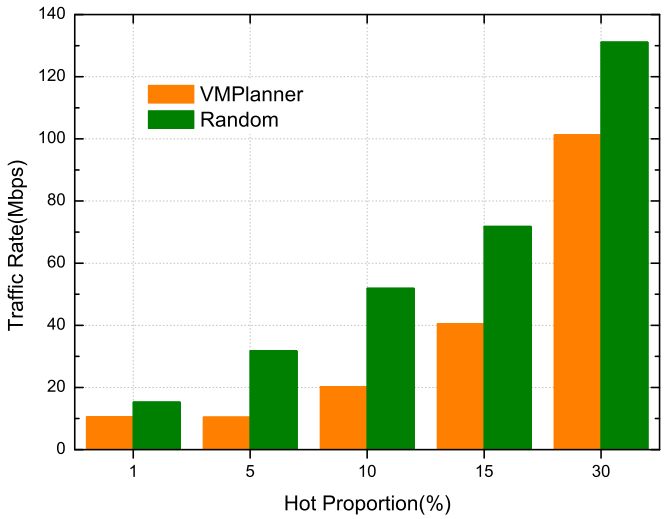

Fig. 5. Average inter-group traffic volume ($K = 64$).

Table 2
Comparisons on the aggregate rates of inter-rack traffic perceived by every switch (unit: Mbps).

	1%	5%	10%	15%	30%
Fat-Tree					
Random	159,435	166,173	465,636	764,079	1,660,290
Distance-aware	158,855	164,912	459,951	756,302	1,649,030
VL2					
Random	201,611	201,709	390,170	778,959	1,949,030
Distance-aware	201,097	201,239	387,470	774,006	193,8740
VL2N-Tree					
Random	191,027	190,829	368,967	738,056	1,843,520
Distance-aware	190,406	190,405	365,684	731,988	1,833,670

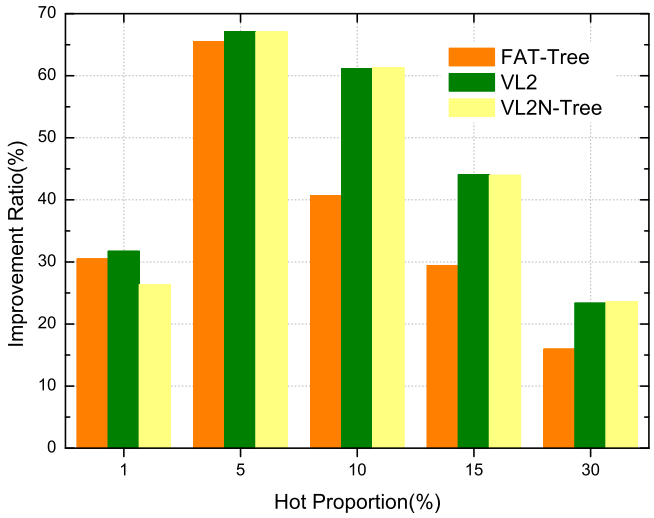

Fig. 6. Improvements on the aggregate traffic by VMPlanner optimization for different topologies.

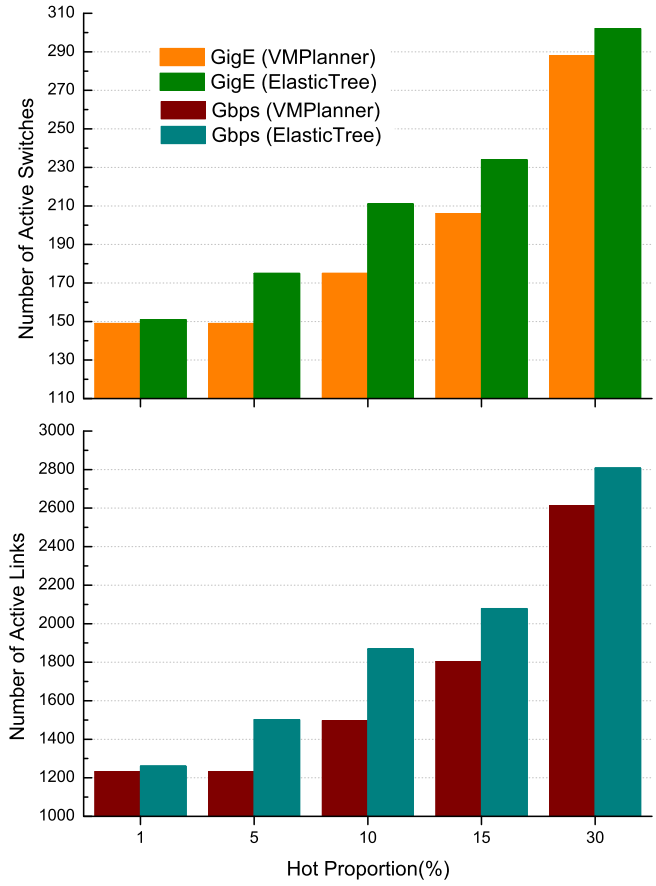

Fig. 7. Number of active network elements in Fat-Tree ($|X| = 2048$).

Constraint (7) ensures that the virtual machines hosted by a physical server is within its hosting capacity. Constraints (8) and (9) states that value assignments for the binary decision variables $B_s(z)$ and $B_l(e)$. Constraint (10) states the classical flow conservation constraints. Constraint (11) ensures that the total flow assigned on each link does not exceed the link capacity, and also implies that traffic flow is restricted to only those active network elements, i.e., $f^e_{IJ} = 0$ when $B_l(e) = 0$. Constraint (12) ensures that all links connected to switch u are also put to sleep when this switch is put to sleep, and vice versa.

We now analyze the computational complexity of this optimization problem.

Theorem 1. *The problem of optimizing network power consumption in data centers with network-aware VM placement and power-aware traffic routing is NP-complete.*

Proof. We can prove this theorem by restriction [34], a most frequently applicable technique for proving NP-completeness. As illustrated in Fig. 2, the placement of VMs onto the hosting servers has direct impacts on the subsequent routing optimization and the final energy conservation. To achieve the final objective in Function (1), we could firstly enumerate all possible mapping combinations between X and Y on a n-to-1 basis. Then for each set mapping (VM placing) trial, we will have a corresponding matrix representing traffic demands between server pairs, which can consequently be used for flow assignment and routing optimization. At last, we can compute the network power consumption for each trial, and find the optimal solution from the results. It is obvious that power-aware routing optimization is inevitable in the above process, no matter how large or small the problem size is. The routing sub-problem has been proved as an NP-complete problem by a reduction from the Multi-Commodity Flow Problem [12], which is known as NP-complete [34]. Therefore, the whole optimization problem can be proved as NP-complete by restriction. □

Although the enumeration-based method given in the proof is simple and direct, it cannot scale to the size of current data centers, and thus is impossible to be used in practice. It is necessary to develop the solution and algorithms with acceptable time complexity for our purpose.

3.3. Solution and algorithm design

According to the previous analysis, we designed an approach VMPlanner to solve the problem stepwise, by

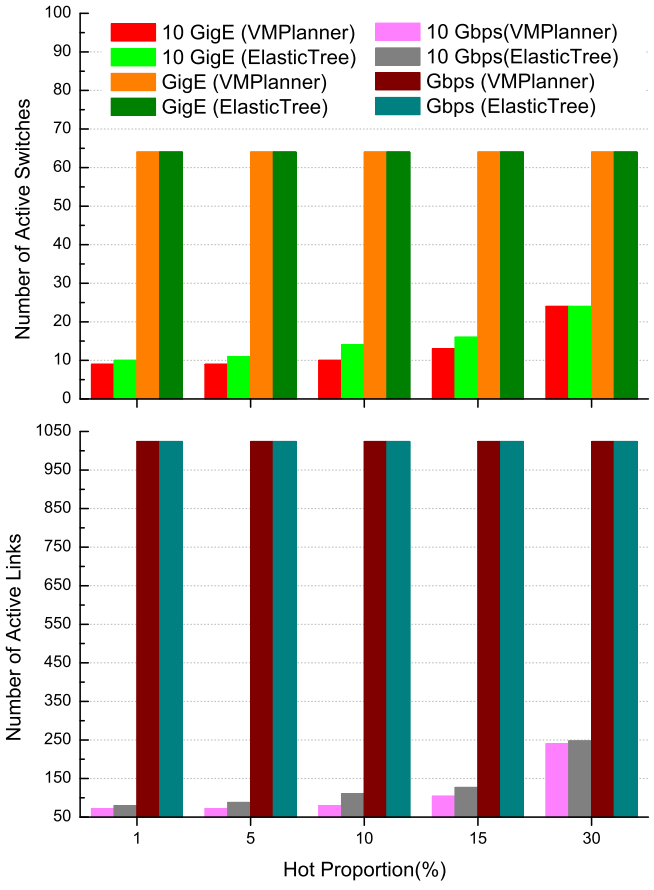

Fig. 8. Number of active network elements in VL2 ($|X| = 2048$).

leveraging unique features of topology characteristics and traffic patterns in data centers.

Recall the typical network topologies illustrated in Fig. 1, we have two important observations on server racks. Firstly, the ToR switches are rarely powered off because they are the indispensable devices for server interconnection and traffic routing. A ToR switch can be safely put to sleep for power saving only when no server in its rack has computing job and communicating traffic. Secondly, the communication distance between a VM pair placed in different racks is irrelative to the specific servers for hosting these two VMs, and is judged by the hop distance between rack ToR switches. Based on these observations, we propose to solve the placement sub-problem by mapping VM-groups to server-racks on a one-to-one basis instead of mapping VMs to individual servers on an *n*-to-one basis. The VM-groups are obtained through the BMKP algorithm [13] which ensures that VM pairs with high mutual traffic rate are within the same VM-group. Such a feature is consistent with the fact that traffic generated from a small number of VMs comprise a large fraction of the total traffic [17,30]. Then, the mapping of VM-groups to server-racks are realized by adopting the QAP algorithm [14] which ensures that the traffic distribution among VM-groups could be better aligned with the communication distance between them, e.g., the VM-groups with large mutual bandwidth usage are assigned to host servers in close proximity. By placing VMs in such a manner, the aggregate rates of inter-rack traffic perceived by every switch can be minimized, so that the total traffic demands on the provided path capacity can be minimized. Finally, the MCFP algorithm [12] is used to optimize routing and find the minimum-power network subset that satisfies the traffic conditions.

3.3.1. Traffic-aware VM grouping

The intuition of traffic-aware VM grouping is to partition VMs into a set of VM-groups so that the overall inter-group traffic volume is minimized while the overall intra-group traffic volume is maximized. Let r denotes the server amount in a rack, VMPlanner needs to partition X into $K = \frac{|X|}{s}$ disjoint subsets $\{X_1, X_2, \ldots, X_K\}$ with equal size $s = n \times r$, while minimizing

$$\sum_{I'=1}^{K-1} \sum_{J'=I'+1}^{K} \sum_{\substack{i \in X_{I'} \\ j \in X_{J'}}} (d_{i,j} + d_{j,i}) \tag{13}$$

Noted that the formulation described above assumes $|X|$ is exactly divisible by s. If it is not the case, we can always make it hold by introducing dummy VMs which do not receive or send any traffic. Obviously adding such dummy VMs does not affect VM grouping.

The presented formulation falls into the class of Balanced Minimum K-cut Problem (BMKP), which is a known NP-complete problem. The grouping method used in this work is adapted from the algorithm in [13]. The approximation ratio for the algorithm is $\frac{K-1}{K}|X|$.

The pseudo-code of our adapted algorithm is described in Algorithm 1. A brief explanation is given as follows: suppose we need to find a VM-group with size s, we first find all the min-cut for every VM pair by applying the classical Gomory–Hu's algorithm [35]. It is shown in [35] that there are only $|X|-1$ distinct min-cut among the $\frac{|X|(|X|-1)}{2}$ total pairs of VMs. Then we find a subset of these $|X|-1$ min-cut such that their removal from G'' leads to a partition with the requested size. This process continues until all VM-groups with requested size are formed. With similar proof as in [13] we can shown that this process terminates after find K groups with size s. Algorithm 1 is reported to have complexity $\mathcal{O}(|X|^4)$ [13].

Algorithm 1. VM grouping

1: $K \leftarrow \frac{|X|}{s}$
2: Construct a weighted complete graph $G''(X, E'')$, in which weight $w_{i,j} = d_{i,j} + d_{j,i}, \forall e = (i, j) \in E''$
3: $S \leftarrow |X|$
4: Compute Gomory-Hu tree for G'' and obtain $|X|-1$ cuts $\{m_c\}$ which contain the minimum weight cuts for all VM pairs
5: Sort $\{m_c\}$ by increasing weight
6: **for** $k = 1$ **to** K **do**
7: Clear X_k
8: Find the minimum c' such that removing $\{m_1, \cdots, m_{c'}\}$ will partition G'' into two parts: G''_1 with size s and G''_2 with size $S - s$
9: $X_k \leftarrow G''_1$
10: $G'' \leftarrow G''_2$
11: $S \leftarrow S - s$
12: **end for**
13: **return** $\{X_k\}$

3.3.2. Distance-aware VM-group to server-rack mapping

After VM grouping, VMPlanner needs to optimally assign VM-groups to server-racks such that the mapping minimizes the total inter-rack traffic load in the network. For any placement approach that assigns K VM-groups to K server-racks on a one-to-one basis, there is a corresponding permutation function π'. Let $c_{\pi'(I'),\pi'(J')}$ denotes the number of switches on the routing path from the rack hosting VM-group $X_{I'}$ to the one hosting $X_{J'}$, we can formally define the distance-aware VM-group to server-rack mapping problem as finding a permutation π' to minimize the following objective function

$$\sum_{I',J' \in [1,K]} c_{\pi'(I'),\pi'(J')} \sum_{\substack{i \in X_{I'} \\ j \in X_{J'}}} (d_{i,j} + d_{j,i}) \tag{14}$$

The formulation described above assumes equal number of VM-groups and server-racks, i.e., $|Y| = |Y'| = K \times r$. When there are more available server-racks than VM-groups (i.e., $|Y| > |Y'|$), we can just choose K racks

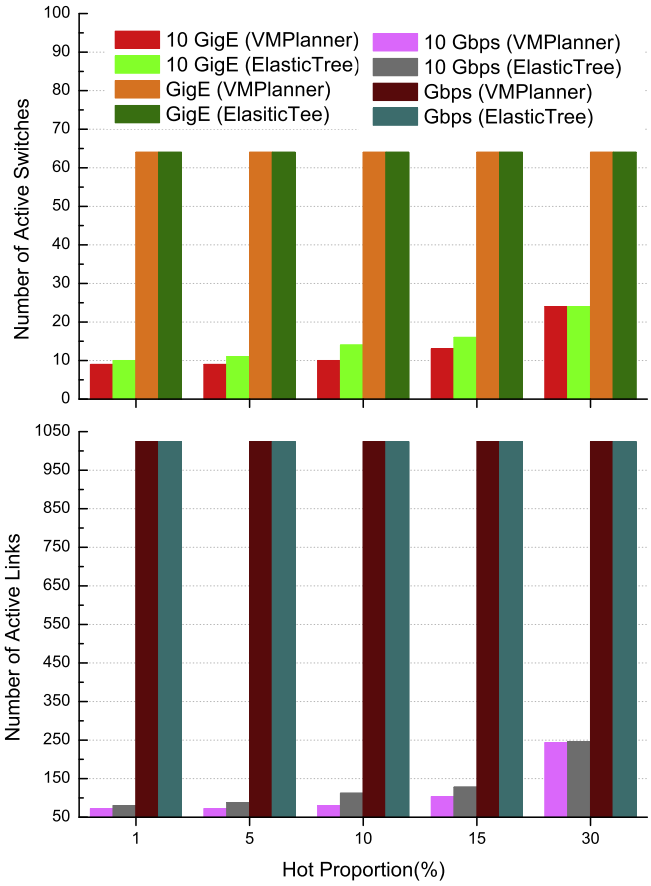

Fig. 9. Number of active network elements in VL2N-Tree ($|X| = 2048$).

sequentially from the first one in the network topology, while the remaining racks as well as their ToR switches are left unused and can be put to sleep for power saving.

The presented formulation falls into the class of Quadratic Assignment Problem (QAP), which is a known NP-complete problem. In fact, it is one of the most difficult problems in NP-hard class, and it even cannot be approximated efficiently within some constant approximation ratio. Exact algorithms are still not practical to solve QAP problems with size > 20 due to very high time complexity, which makes the development of heuristic algorithms to provide good quality solutions in a reasonable time. The assignment method used in this work is adapted from the algorithm in [14] that is based on Tabu Search [36].

The pseudo-code of our adapted algorithm is described in Algorithm 2. A brief explanation is given as follows: The tabu search procedure starts with an initial, randomly generated solution π'' (Line 1) and selects a best-quality solution π' among a part of the neighbours of π'' obtained by non-tabu moves (Lines 10–24). A move is an operation which, when applied to a certain solution, generates a neighbour solution of it. In QAP, the neighbourhood is the pair-exchange neighbourhood and the moves are usually transpositions (Line 28). In general, one of the moves which most decreases the total cost C_{best} will be chosen, otherwise the one that least degrades the objective function will be chosen (Line 15). Then the current solution is updated, i.e., it is substituted by the selected solution (Lines 29–32). In order to avoid returning to the local optimum just visited, the reverse move now must be forbidden. This is done by storing the moves which are expected to lead to search cycles in the tabu list L (Lines 34–35). In [14], the size of tabu list is randomly decided between 0.9 K and 1.1 K during the search (Lines 34–35). However, forbidding certain moves could prohibit visiting relevant or interesting moves, as exemplified by those that lead to a better solution than the best one found so far. Consequently, an aspiration criterion $A_c = K^2/2$ is introduced to distinguishes the potentially interesting moves among the forbidden ones (Line 14). The search stops when a stop criterion, i.e., the limited number of iterations $t_{iteration}$, is fulfilled (Line 7). Noted that the parameter $\Delta_{I',J'}$ denotes the cost difference if elements I' and J' are transposed in current permutation π''. The complexity of Algorithm 2 is $\mathcal{O}(t_{iteration}K^2)$. Future Details on the algorithm is omitted in this paper due to space limitations, and interested reader can refer to [14,36] for more on QAP and Tabu Search.

Algorithm 2. VM-group to server-rack mapping

Generate a random mapping permutation π''
$\pi' \leftarrow \pi''$
Calculate $C_{current}$ according to Function (14)
$C_{best} = C_{current}$
Calculate $\Delta_{I',J'}, \forall I', J' \in [1, K], I' < J'$
$L_{I',J'} \leftarrow -(K \times I' + J')$
for $k = 1$ **to** $t_{iteration}$ **do**
  $I' \leftarrow \infty$
  $\Delta_{min} \leftarrow \infty$
  for $I' = 1$ **to** $K - 1$ **do**
    for $J' = I' + 1$ **to** K **do**
      $A_l =$ **false**
      $A_u \leftarrow (L_{I',\pi''(J')} < k$ **or** $L_{J',\pi''(I')} < k)$
      $A_s \leftarrow (L_{I',\pi''(J')} < k - A_c$ **or** $L_{J',\pi''(I')} < k - A_c$ **or** $C_{current} + \Delta_{I',J'} < C_{best})$
      if $((A_s$ **and not** $A_l)$ **or** $(A_s$ **and** A_l **and** $\Delta_{I',J'} < \Delta_{min})$ **or** (**not** A_s **and not** A_l **and** $\Delta_{I',J'} < \Delta_{min}$ **and** $A_u))$ **then**
        $I'' \leftarrow I'$
        $J'' \leftarrow J'$
        $\Delta_{min} \leftarrow \Delta_{I',J'}$
        if A_s **then**
          $A_l \leftarrow$ **true**
        end if
      end if
    end for
  end for
  if $I'' == \infty$ **then**
    return NULL
  else
    $Transpose(\pi''(I''), \pi''(J''))$
    $C_{current} \leftarrow C_{current} + \Delta_{I'',J''}$
    if $C_{current} < C_{best}$ **then**
      $C_{best} \leftarrow C_{current}$
      $\pi' \leftarrow \pi''$
    end if
    $L_{I'',\pi''(J'')} \leftarrow k + Random(0.9K, 1.1K)$
    $L_{J'',\pi''(I'')} \leftarrow k + Random(0.9K, 1.1K)$
    Calculate $\Delta_{I',J'}, \forall I', J' \in [1, K], I' < J'$
  end if
end for
return π'

3.3.3. Power-aware inter-VM traffic flow routing

After placing VMs onto rack servers, VMPlanner can move and aggregate network traffic onto a fewer number of paths so as to put the remaining network elements into dormant state for energy conservation now. Since the permutation π and the set Y' is known to VMPlanner, we are able to obtain Z' and E' in this step.

The original problem presented in Section 3.2 can now be transformed into a variant of Multi-Commodity Flow Problem (MCFP) that is augmented with binary variables for the power states of links and switches. It minimizes the total network power cost by solving a Mixed-Integer Linear Program (MILP), which is shown to be NP-complete [34]. The formal method to solve the MILP problem is flexible enough to support arbitrary topologies, but can only scale up to networks with small size [12]. The solution used in this work is adapted from the algorithm in [12] that is based on Greedy Bin-Packing, which has also been used in [31,32].

The Greedy Bin-Packing algorithm is described as follows: for each flow, VMPlanner evaluates possible routing paths and then chooses the leftmost one with sufficient bandwidth capacity. By leftmost, we mean in reference to a single tier in a structured topology, such as the ones in Fig. 1. Within a tier, paths are chosen in a deterministic left-to-right order, as opposed to a random order, which would evenly spread traffic flows [17]. When all flows have been assigned (which is not guaranteed), the algorithm can return the minimum-power network subset as well as routing paths for each flow. In some cases, this approach will not find a satisfying assignment for all flows. This is found as an inherent problem with any greedy flow assignment strategy [12]. In such a case, the greedy search has to enumerated all possible routing paths, and assign the flow to the path with the lowest load. This algorithm is reported to have complexity $\mathcal{O}(|Y|^{2.5})$ [12].

4. Implementation issues

As shown in Fig. 3, the VMPlanner system for data centers consists of three logical modules, i.e., analyzer, optimizer and controller. The analyzer obtains system states (e.g., VM, server and network) and traffic demands by performing statistics and analysis (e.g., traffic prediction [12]) to the collected running data of the data center. The role of optimizer is to execute the three algorithms in Section 3. With the input of system and traffic conditions from the analyzer, the optimizer outputs the result of VM server mapping, the set of active network elements, and the set of traffic flow routes to the controller. The controller is responsible for controlling VM (i.e., placing or migrating every VM onto the assigned server), route (i.e., checking routing paths for traffic flows and pushing routes into the network), and power (i.e., toggling the power states of different types of network elements).

The analyzer can collect the sate information of system and traffic through SNMP Get operations and passive packet tracing [33]. Similar to ElasticTree [12], VMPlanner can be implemented as a NOX application [37] to run atop a network of OpenFlow switches. OpenFlow is an open standard for commercial switches and routers to enable controlling the forwarding plane and realizing the policy-based routing 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 controlling in VMPlanner can be implemented with NOX. Popular IaaS (Infrastructure as a Service) platforms have provided interfaces for supporting automated VM management

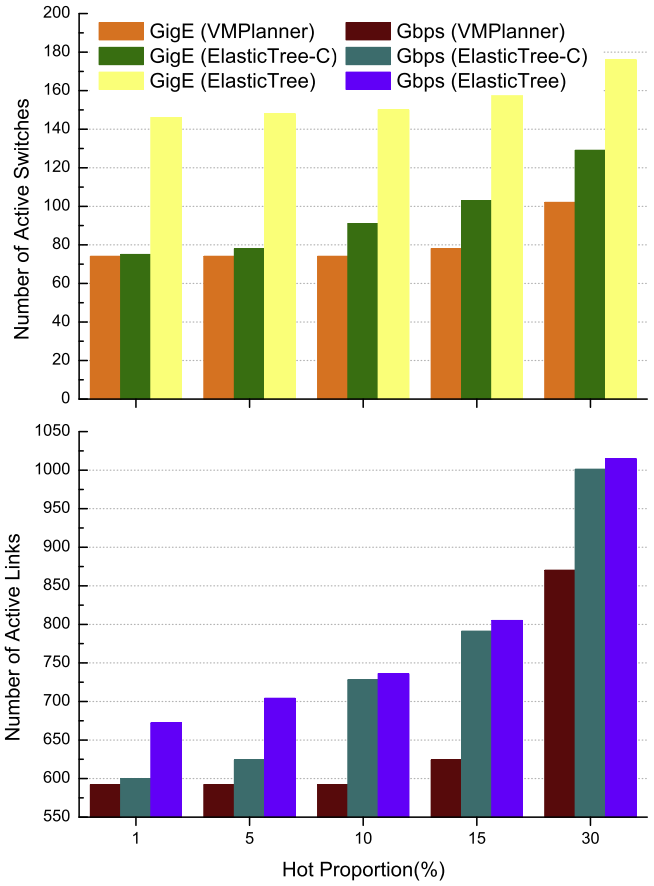

Fig. 10. Number of active network elements in Fat-Tree ($|X| = 1024$).

[38]. Moreover, existing mechanisms such as SNMP Set Operations and command line interface can be leveraged to support power controlling [12].

5. Performance evaluation

We have evaluated the performance of VMPlanner experimentally, using a custom simulator developed in C++. Our purpose is not to construct a perfectly realistic simulation, but to demonstrate the advantages of optimizing VM placement and traffic routing, which we believe, are of practical importance for reducing network power costs in cloud data centers.

To obtain more convincing results, we took the simulation parameters and the traffic conditions by making reference to real cases from a private data center test-bed [16]. This data center hosts about 2000 servers, which are organized into a VL2N-Tree architecture as that in Fig. 1 with 16-port GigE switches and 24-port 10 GigE switches. We conducted a measurement study in this data center, and obtained traffic traces containing the incoming and outgoing network traffic for nearly 2000 VMs. Based on this traces, we observed the same trends as described in [30,32]. Firstly, traffic distribution for individual VMs is highly uneven. At any time, only 15–20% VMs are sending or receiving packets. Secondly, though the average traffic rates are divergent among VMs, the rate for most of the VMs are relatively stable when the rate is computed at large time intervals. A relatively large proportion of VMs have a relatively small average rate of 10–100 kbps, while a relatively small proportion of VMs have a relatively higher average rate of 1500–2500 kbps. In this work, the proportion of traffic with high-rate is called hot proportion. This proportion varies between 1% and 30% over time [12]. Thirdly, there is a low correlation between the average pairwise traffic rate and the end-to-end latency. In addition to these observations, the specific simulation parameters for the three network architectures in Fig. 1 are listed in Table 1. Based on the traces and the settings, we conducted extensive simulations to verify the performance of VMPlanner. Unless otherwise mentioned, the results are obtained by averaging from multiple runs.

5.1. Network traffic distribution

In the first set of experiments, we evaluated the performance of VMPlanner on traffic distribution optimizations. We compared VMPlanner with a random placement in which any VM has equal probability to be assigned to any group and any rack. Such a random placement reflects assignment of VMs that does not take advantage of the traffic distribution optimization.

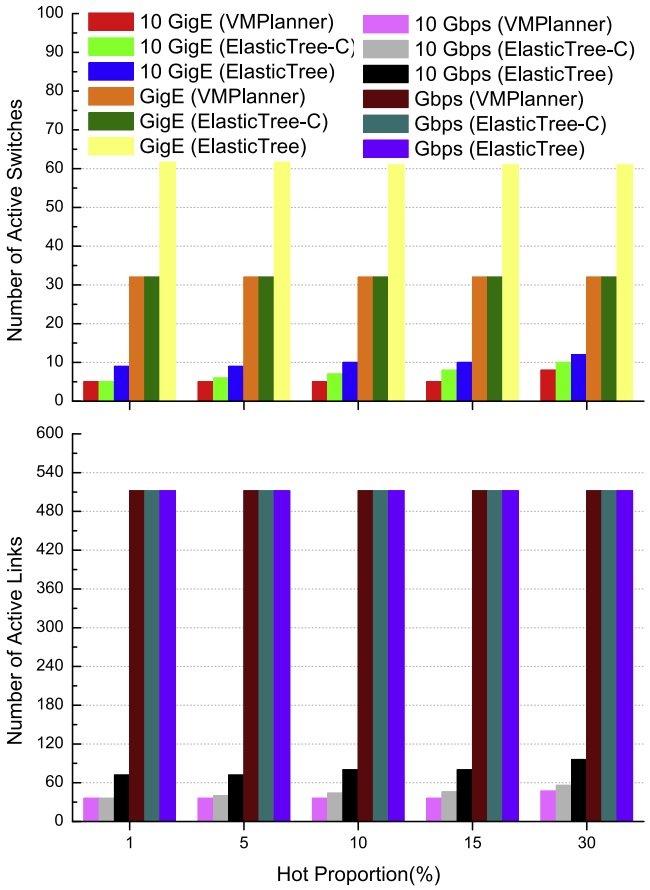

Fig. 11. Number of active network elements in VL2 ($|X| = 1024$).

Figs. 4 and 5 show comparison results on the average inter-group traffic volume for $K = 128$ (Fat-Tree) and $K = 64$ (VL2 and VL2N-Tree), respectively. It can be noticed that by traffic-aware VM grouping, the inter-group traffic is reduced by 15–67% than that of random grouping. The comparison is glaringly obvious (nearly 70% of reduction) when the hot proportion is equal to 5%, but since then the gap becomes smaller as the hot proportion increases. This indicates that the improvement space of the random approach increases with the increase of traffic variance only within a certain range, but not as always as depicted in [30]. The reason is that, when there are a large proportion of high-rate inter-VM flows, not all of them can be constrained within the group range. With the increase of hot proportion, more and more high-rate flows are becoming inter-group flows, leading to relatively less opportunities for BMKP optimizations. From the results, we can also observe that the improvement is slightly higher when $K = 64$ than when $K = 128$ under the same traffic conditions.

Table 2 shows the comparisons on the aggregate rates of inter-rack traffic after VM-group to server-rack mapping based on the BMKP-based VM grouping results. Generally, the QAP-based mapping strategy outperforms that of random mapping, although not so dramatically. The underlying reasons are twofold. On the one hand, the rate differences of inter-rack traffic is not very significant after the BMKP optimization. On the other hand, the inter-rack distance (i.e., the number of intermediate switches) is either 3 or 5 in the three topologies. Due to these factors, the optimization space is not so large according to Function (14). However, we argue that distance-aware mapping has to be retained in VMPlanner, since VMPlanner is designed as a general approach for data center networks with various traffic patterns [17,33] and complex interconnect topologies [20,39].

At last, we show the comparisons on the aggregate traffic under the conditions of VMPlanner optimization (i.e., traffic-aware VM grouping + distance-aware VM-group to server-rack mapping) and no optimization (i.e., Random-based VM group + Random-based VM-group to server-rack mapping) in Fig. 6. Generally, VMPlanner can help to achieve different degrees of improvement on traffic distribution under conditions of different network topologies and different traffic patterns. VL2 and VL2N-Tree have almost the equal improvement ratios, due to their structure similarities. Fat-Tree has nearly equal ratios with them when the hot proportion is 1% and 5%, but significant smaller ratios than them when the hot proportion is relatively larger.

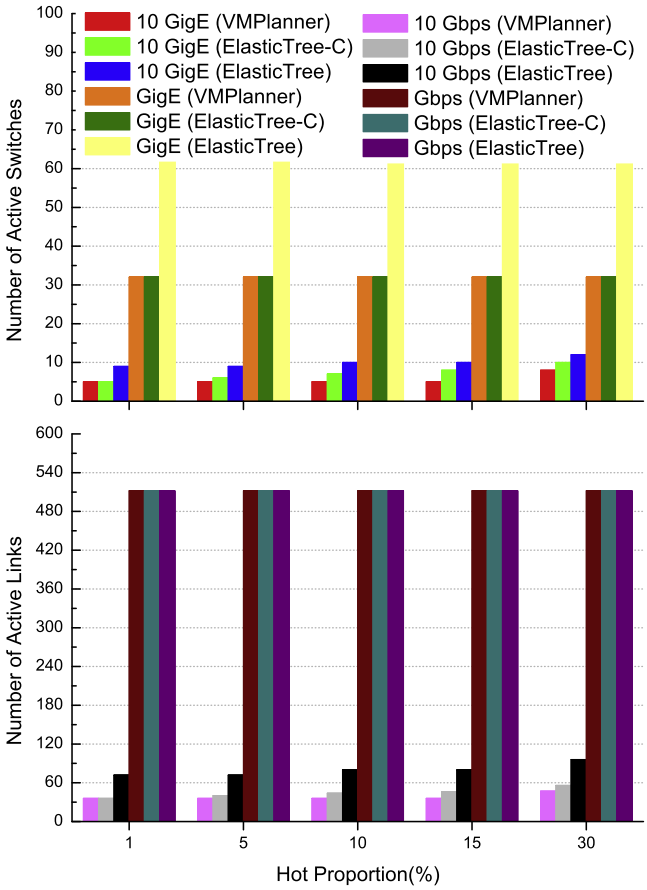

Fig. 12. Number of active network elements in VL2N-Tree ($|X| = 1024$).

5.2. Network element scheduling

In the second set of experiments, we evaluated the performance of VMPlanner on sleep scheduling capabilities. VMPlanner is compared to the state-of-the-art baseline ElasticTree on how many switches and links at the minimum are kept active for satisfying traffic demands. Unlike VMPlanner, ElasticTree does not perform traffic optimization before power management.

In the first scenario, we let $|X| = 2048$, $|Y'| = |Y|$, i.e., all physical servers are needed for hosting the VMs. We follow the random strategy for VM placement in this scenario. The results for the three architectures are illustrated in Figs. 7–9. In these figures, the number of switches and links are separately plotted to make the results more clear.

For Fat-Tree, we can notice that the usage of network elements grows with the increase of traffic volume (i.e., hot proportion). Nearly half of the switches and more than $\frac{1}{3}$ of the links are put to active when the hot proportion is 1%, while nearly all of the switches and links are put to active when the hot proportion is 30%. Noted that Fat-Tree is designed to use only GigE switches and Gbps link. VMPlanner outperforms ElasticTree by 1.32–17.06% on GigE switches and 2.30–19.90% on Gbps links under different traffic conditions. However, the improvement is very slight (less than 10%) when the hot proportion is 1% and 30%. The reason is as what we explained in Section 5.1, i.e., the optimization space for traffic distribution is relatively small when the traffic variance is very small or very large.

For VL2, we can notice that the usage of high-bandwidth network elements grows with the increase of traffic volume (i.e., hot proportion). Nearly half of the 10 GigE switches and about $\frac{1}{3}$ of the links are put to active when the hot proportion is 1%, while nearly all of the switches are links are put to active when the hot proportion is 30%. Noted that the GigE ToR switches and the Gbps ToR-Server links are all kept active for working and cannot be put into sleep. VMPlanner outperforms ElasticTree by 0–28.57% on 10 GigE switches and 2.83–27.93% on 10 Gbps links under different traffic conditions. When the hot proportion is 30%, we can notice that the improvement is very slight.

Generally, most of the results of VL2N-Tree are similar to those of VL2, due to their structure similarities. We can notice that the usage of high-bandwidth network elements grows with the increase of traffic volume (i.e., hot proportion). Nearly half of the 10 GigE switches and about $\frac{1}{3}$ of the links are put to active when the hot proportion is 1%, while nearly all of the switches are links are put to active when the hot proportion is 30%. Half of the GigE ToR switches and Gbps ToR-Server links are kept active for

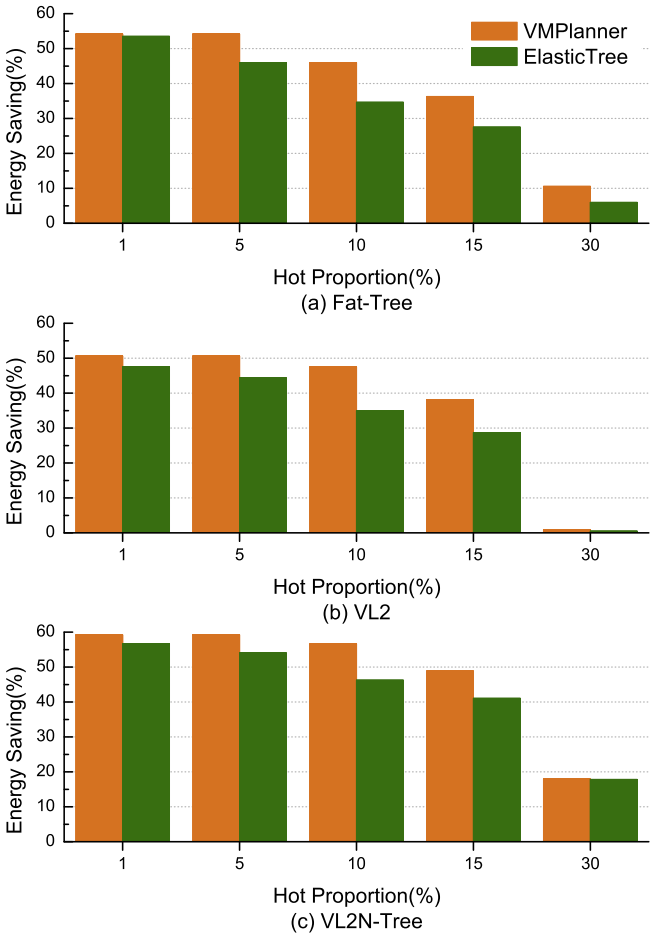

Fig. 13. Energy savings for different network architectures ($|X| = 2048$).

working, while the other halves originally designed as redundant ones are safely put to sleep. VMPlanner outperforms ElasticTree by 0–28.57% on 10 GigE switches and 1.22–28.57% on 10 Gbps links under different traffic conditions. When the hot proportion is 30%, we can notice that the improvement is very slight.

In the second scenario, we let $|X| = 1024$, and thus at the minimum $\frac{|Y|}{2}$ physical servers are enough for hosting the VMs. In terms of the VM placement strategies, we extend the basic ElasticTree into a variant called ElasticTree-C. In ElasticTree, any VM has equal probability to be assigned to any one that is available for hosting VMs from all the $|Y|$ physical servers. In ElasticTree-C, we use the same strategy for choosing racks and servers as VMPlanner, i.e., choose K racks sequentially from the first one in the network topology. In short, the difference between them is whether the VMs are placed and distributed compactly or not in the data center. The results for the three architectures are illustrated in Figs. 10–12.

For Fat-Tree, VMPlanner outperforms ElasticTree and ElasticTree-C, by 38.07–51.55% and 1.33–24.27% on GigE Switches, and by 11.90–22.48% and 1.33–21.11% on Gbps links, respectively. ElasticTree has the worst performance among the three approaches, because of not only the unoptimized traffic distribution but also the random placement of VMs. Thus, much more ToR and aggregate switches have to be kept active even though some racks may only host one VM. Although ElasticTree-C adopts the compact placement strategy as VMPlanner, it is inferior to the latter because it does not take optimizing traffic distribution into account for VM placement.

For VL2, VMPlanner outperforms ElasticTree and ElasticTree-C, by 33.33–50.0% and 0–37.5% on 10 GigE Switches, and by 50–55% and 0–21.74% on 10 Gbps links, respectively. ElasticTree has to use all of the GigE switches, while VMPlanner and ElasticTree-C use only half of the GigE switches with the compact placement strategy. VMPlanner and ElasticTree-C has the same performance when the hot proportion is 1%, because the overall traffic variance is very small. However, BMKP optimization is more capable when the hot proportion is 30% than in the first scenario.

Generally, most of the results of VL2N-Tree are similar to those of VL2, because of their structure similarities. However, VL2N-Tree has more redundant ToR switches and Tor-Server links than VL2, and thus the improvements on them through VMPlanner and ElasticTree are in fact relatively higher than the latter.

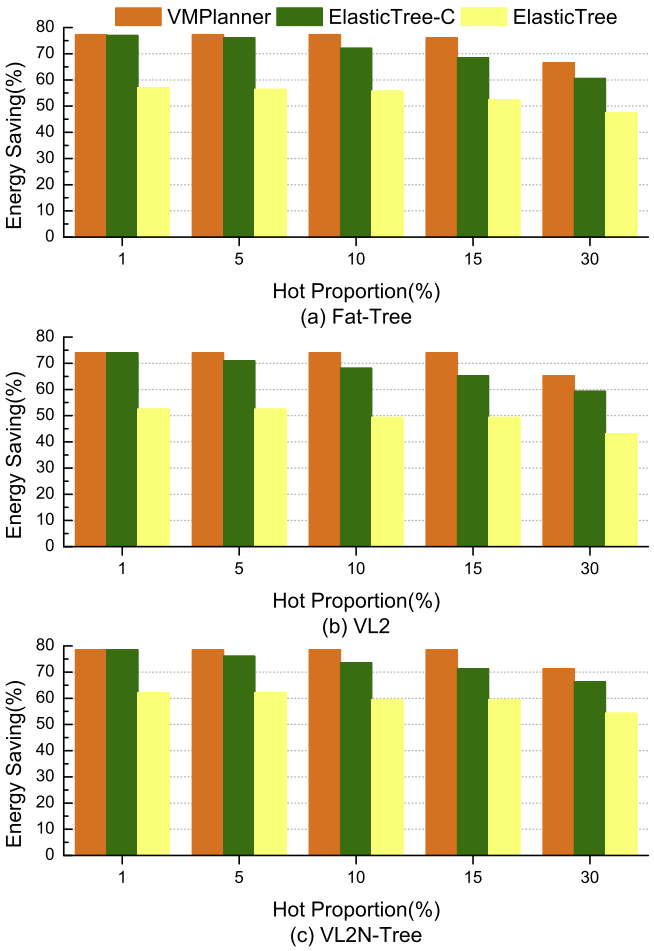

Fig. 14. Energy savings for different network architectures ($|X| = 1024$).

5.3. Network power conservation

Based on the scheduling results above, we present the comparisons on the total energy savings in Figs. 13 and 14. The power consumption parameters of network elements are obtained from [40]. Specifically, the power parameters for 10 GigE Switch, GigE switch, 10 Gbps link, Gbps link are 300 W, 30 W, 6 W, and 0.4 W, respectively.

Generally, VMPlanner outperforms ElasticTree on energy saving performance under conditions of different network topologies and different traffic patterns. When $|X| = 2048$, VMPlanner outperforms ElasticTree by 0.66–11.35% in Fat-Tree, by 0.38–12.53% in VL2, and by 0.13–10.39% in VL2N-Tree, respectively. When $|X| = 1024$, VMPlanner outperforms ElasticTree by 19.10–23.61% in Fat-Tree, by 21.47–24.62% in VL2, and by 16.36–18.96%, respectively. For ElasticTree-C, VMPlanner still outperforms it by 0.31–7.59% in Fat-Tree, 0–8.68% in VL2, and by 0–7.17% in VL2N-Tree, respectively. Referring to the experimental results in previous sub-sections, it can also be found that the performance gaps are relatively significant when the traffic distribution can be better optimized by VM placement, e.g., when the hot proportion is 5%, 10% or 15%. These findings indicate that VMPlanner is best fit to the data center devoted to workloads with very heterogeneous traffic among VMs, while ElasticTree-C may be used instead in the data center that is devoted to just one application with homogeneous traffic pattern among VMs. Moreover, VL2N-Tree has the highest energy saving ratio among the three under the same traffic condition, because it is designed to have many redundant ToR switches and ToR-Server links for reliability guaranteeing.

6. Conclusions

This paper presents VMPlanner, a framework for reducing power costs of network elements in data centers. VMPlanner takes advantage of the flexibility provided by dynamic VM migration and programmable flow-based routing, that are available in modern data centers, to optimize network power consumption while satisfying network traffic demands. We formulate the optimization problem, analyze its NP-hardness, and decompose it into three sub ones to solve it step-by-step with VMPlanner. We have implemented and evaluated VMPlanner in a simulated environment running real data center traffic workloads. Evaluations based on typical network topologies and real traffic demands show that, through joint opti-

mization on VM placement and traffic routing, VMPlanner is more capable to save network power than the state-of-the-art baseline ElasticTree.

The work in this paper is very preliminary, but demonstrates that it is both promising and feasible to reduce network power costs in cloud data centers by jointly optimizing VM placement and traffic routing. Regarding the applicability of VMPlanner in data centers, there are still some limitations and challenges remain to be solved. Firstly, in current solution the conditions of VMs and inter-VM flows are assumed to be accurately predicted and proactively obtained according to historical analysis. When the number of total VMs and the rates of inter-VM flows dynamically varies over time, the costs for frequently re-executing the entire solution and frequently re-allocating the total would be considerably large. It is necessary to minimize such costs by improving VMPlanner on the capabilities of accommodating changing conditions. Secondly, typically VMs with higher inter traffic may correspond to the same task or customer in a multi-tenant data center. Assigning them under one ToR switch results in everything under one failure domain. It is necessary for VMPlanner to spread the VMs corresponding to the same task or customer across the topology. Thirdly, we would like to examine the trade-offs between energy efficiency, end-to-end latency and system robustness. VMPlanner may incorporate some level of switch/link redundancy so as to reduce the packet latency in the switch buffer and to accommodate traffic surges and network failures. The issues mentioned above, as well as VMPlanner implementation in an experimental test-bed, are left as future works.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable feedback. This work was supported in part by the National Science Foundation of China (NSFC) under Grant No. 61202430, the Fundamental Research Funds for the Central Universities of China under Grant No. 2011JBM225, the ZTE – BJTU Collaborative Research Program under Grant No. K11L00190, and the European Union Seventh Framework Programme (FP7/2007-2013) under Grant No. 257740.

References

[1] Kant Krishna, Data center evolution: a tutorial on state of the art, issues, and challenges, Computer Networks 53 (2009) 2939–2965.
[2] EPA, Report to Congress on Server and Data Center Energy Efficiency, Technical report, US Environmental Protection Agency, 2007.
[3] A. Greenberg, J. Hamilton, D.A. Maltz, P. Patel, The cost of a cloud: research problems in data center networks, SIGCOMM Computer Communication Review 39 (2008) 68–73.
[4] L.A. Barroso, U. Hölzle, The case for energy-proportional computing, Computer 40 (2007) 33–37.
[5] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, D. Wetherall, Reducing network energy consumption via sleeping and rate-adaptation, in: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, NSDI'08, USENIX Association, Berkeley, CA, USA, 2008, pp. 323–336.
[6] S. Herrería-Alonso, M. Rodríguez-Pérez, M. Fernández-Veiga, C. López-García, Optimal configuration of energy-efficient ethernet, Computer Networks 56 (2012) 2456–2467.
[7] A. Bianzino, C. Chaudet, F. Larroca, D. Rossi, J. Rougier, Energy-aware routing: a reality check, in: Proceedings of the 2010 IEEE GLOBECOM Workshops, GC Wkshps'10, pp. 1422–1427.
[8] R. Bolla, R. Bruschi, F. Davoli, F. Cucchietti, Energy efficiency in the future internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures, IEEE Communications Surveys Tutorials 13 (2011) 223–244.
[9] M. Gupta, S. Singh, Greening of the internet, in: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM'03, ACM, New York, NY, USA, 2003, pp. 19–26.
[10] S. Nedevschi, J. Chandrashekar, J. Liu, B. Nordman, S. Ratnasamy, N. Taft, Skilled in the art of being idle: reducing energy waste in networked systems, in: Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation, NSDI'09, USENIX Association, Berkeley, CA, USA, 2009, pp. 381–394.
[11] L. Chiaraviglio, M. Mellia, F. Neri, Reducing power consumption in backbone networks, in: Proceedings of the 2009 IEEE International Conference on Communications, ICC'09, IEEE Press, Piscataway, NJ, USA, 2009, pp. 2298–2303.
[12] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, N. McKeown, Elastictree: saving energy in data center networks, in: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI'10, USENIX Association, Berkeley, CA, USA, 2010, pp. 1–16.
[13] H. Saran, V.V. Vazirani, Finding *k*-cuts within twice the optimal, SIAM Journal of Computing 24 (1995) 101–108.
[14] E. Taillard, Robust taboo search for the quadratic assignment problem, Parallel Computing 17 (1991) 443–455.
[15] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, A. Vahdat, Hedera: dynamic flow scheduling for data center networks, in: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI'10, USENIX Association, Berkeley, CA, USA, 2010, pp. 19–19.
[16] F. Liu, B. Zhang, K. Miao, J. He, X. Wu, W. Mao, A Cloud Test Bed for China Railway Enterprise Data Center, Technical report, Intel Corporation, 2009.
[17] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel, S. Sengupta, Vl2: a scalable and flexible data center network, in: Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication, SIGCOMM'09, ACM, New York, NY, USA, 2009, pp. 51–62.
[18] Y. Zhang, A.-J. Su, G. Jiang, Understanding data center network architectures in virtualized environments: a view from multi-tier applications, Computer Networks 55 (2011) 2196–2208.
[19] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, in: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, SIGCOMM'08, ACM, New York, NY, USA, 2008, pp. 63–74.
[20] C. Clos, A study of non-blocking switching networks, Bell System Technical Journal 32 (1953) 406–424.
[21] J. Liu, F. Zhao, X. Liu, W. He, Challenges towards elastic power management in internet data centers, in: Proceedings of the 2009 29th IEEE International Conference on Distributed Computing Systems Workshops, ICDCSW'09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 65–72.
[22] G.I. Meijer, Cooling energy-hungry data centers, Science 328 (2010) 318–319.
[23] B. Fakhim, M. Behnia, S. Armfield, N. Srinarayana, Cooling solutions in an operational data centre: a case study, Applied Thermal Engineering 31 (2011) 2279–2291.
[24] W. Hou, L. Guo, X. Wei, X. Gong, Multi-granularity and robust grooming in power- and port-cost-efficient ip over WDM networks, Computer Networks 56 (2012) 2383–2399.
[25] F. Cuomo, A. Cianfrani, M. Polverini, D. Mangione, Network pruning for energy saving in the internet, Computer Networks 56 (2012) 2355–2367.
[26] Y. Shang, D. Li, M. Xu, Energy-aware routing in data center network, in: Proceedings of the First ACM SIGCOMM Workshop on Green Networking, Green Networking'10, ACM, New York, NY, USA, 2010, pp. 1–8.
[27] P. Mahadevan, S. Banerjee, P. Sharma, A. Shah, P. Ranganathan, On energy efficiency for enterprise and data center networks, IEEE Communications Magazine 49 (2011) 94–100.
[28] F. Machida, D.S. Kim, J.S. Park, K. Trivedi, Toward optimal virtual machine placement and rejuvenation scheduling in a virtualized data center, in: Proceedings of the 2008 IEEE International Conference on Software Reliability Engineering Workshops, ISSRE Wksp'08, pp. 1–3.

[29] A. Kochut, On impact of dynamic virtual machine reallocation on data center efficiency, in: Proceedings of the 2008 IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems, MASCOTS'08, pp. 1–8.
[30] X. Meng, V. Pappas, L. Zhang, Improving the scalability of data center networks with traffic-aware virtual machine placement, in: Proceedings of the 29th IEEE Conference on Information Communications, INFOCOM'10, IEEE Press, Piscataway, NJ, USA, 2010, pp. 1154–1162.
[31] R. McGeer, P. Mahadevan, S. Banerjee, On the complexity of power minimization schemes in data center networks, in: Proceedings of the 2010 IEEE Global Telecommunications Conference, GLOBECOM'10, pp. 1–5.
[32] V. Mann, A. Kumar, P. Dutta, S. Kalyanaraman, Vmflow: leveraging VM mobility to reduce network power costs in data centers, in: Proceedings of the 10th International IFIP TC Six Conference on Networking, vol. Part I: NETWORKING'11, Springer-Verlag, BerlinHeidelberg, 2011, pp. 198–211.
[33] T. Benson, A. Anand, A. Akella, M. Zhang, Understanding data center traffic characteristics, SIGCOMM Computer Communication Review 40 (2010) 92–99.
[34] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, 1979.
[35] R.E. Gomory, T.C. Hu, Multi-terminal network flows, Journal of the Society for Industrial and Applied Mathematics 9 (1961) 551–570.
[36] F. Glover, Tabu search: a tutorial, Interfaces 20 (1990) 74–94.
[37] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, S. Shenker, Nox: towards an operating system for networks, SIGCOMM Computer Communication Review 38 (2008) 105–110.
[38] D. Milojičić, I.M. Llorente, R.S. Montero, Opennebula: a cloud management tool, IEEE Internet Computing 15 (2011) 11–14.
[39] L. Gyarmati, T.A. Trinh, Scafida: a scale-free network inspired data center architecture, SIGCOMM Computer Communication Review 40 (2010) 4–12.
[40] D. Kliazovich, P. Bouvry, Y. Audzevich, S. Khan, Greencloud: a packet-level simulator of energy-aware cloud computing data centers, in: Proceedings of the 2010 IEEE Global Telecommunications Conference, GLOBECOM'10, pp. 1–5.

Weiwei Fang received his B.S. degree in Computer Science from Hefei University of Technology, Hefei, China, in 2003, and Ph.D. degree in Computer Science from Beihang University, Beijing, China, in 2010. From 2010 to 2012, he has been a Post-Doc researcher at School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China. Now he is an assistant professor at Beijing Jiaotong University. His current research interests include Data Center Networks, Distributed Computing and Wireless Communication. He has coauthored 1 book chapter and more than 30 papers, and served in the Technique Program Committee of more than 30 conferences and workshops.

Xiangmin Liang is currently a M.S. candidate at School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China. His current research interests include Data Center Networks and Wireless Broadband Networks.

Shengxin Li is currently a B.S. candidate at School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China. His research interests include Data Center Networks and Communication Protocols.

Luca Chiaraviglio is a Post-Doc researcher at the Telecommunication Networks Group of Politecnico di Torino, Italy. Between August 2009 and April 2010 he was a visiting scholar at the WING group of Boston University under the supervision of Prof. Ibrahim Matta. He has coauthored more than 30 papers published in international journals and conferences, and he participated in the program committees of several conferences including IEEE GLOBECOM, IEEE GREENCOM and IEEE PECON. His main research interests are in the field of energy-efficient networking, focusing on wired, wireless and computer networks.

Naixue Xiong received his both PhD degrees in Wuhan University, and Japan Advanced Institute of Science and Technology, respectively. Both are on computer science. From 2008 to 2012, he was a research scientist at Department of Computer Science, Georgia State University, Atlanta, USA. Now he is a professor at School of Computer Science, Colorado Technical University, Colorado Springs, USA. His research interests include Communication Protocols, Network Architecture and Design, and Optimization Theory. Until now, he published about 200 research articles, including over 80 journal papers. Some of his works were published in IEEE JSAC, IEEE or ACM transactions, IEEE INFOCOM, and IPDPS. He has been a General Chair, Program Chair, Publicity Chair, PC member and OC member of about 73 international conferences, and as a reviewer of about 56 international journals, including IEEE JSAC, IEEE Transactions on Communications, IEEE Transactions on Mobile Computing, IEEE Trans. on Parallel and Distributed Systems. He is serving as an editor for nine international journals, and a guest editor for nine international journals, including WINET and MONET.