Journal of Interconnection Networks
© World Scientific Publishing Company

Reducing Power Consumption in Data Centers by Jointly Considering VM Placement and Flow Scheduling

DAWEI LI and JIE WU

Department of Computer and Information Sciences, Temple University
Philadelphia, PA, 19122, United States of America
{dawei.li, jiewu}@temple.edu

Received 1 March 2015
Revised 1 September 2015

Two important components that consume the majority of IT power in data centers are the servers and the Data Center Network (DCN). Existing works fail to fully utilize power management techniques on the servers and in the DCN at the same time. In this paper, we jointly consider VM placement on servers with scalable frequencies and flow scheduling in the DCN, to minimize the overall system's power consumption. Due to the convex relation between a server's power consumption and its operating frequency, we prove that, given the number of servers to be used, computation workloads should be allocated to severs in a balanced way, to minimize the power consumption on servers. To reduce the power consumption of the DCN, we further consider the flow requirements among the VMs during VM allocation and assignment. Also, after VM placement, flow consolidation is conducted to reduce the number of active switches and ports. We notice that, choosing the minimum number of servers to accommodate the VMs may result in high power consumption on servers, due to servers' increased operating frequencies. Choosing the optimal number of servers purely based on servers' power consumption leads to reduced power consumption on servers, but may increase power consumption of the DCN. We propose to choose the optimal number of servers to be used, based on the overall system's power consumption. Simulations show that, our joint power optimization method helps to reduce the overall power consumption significantly, and outperforms various existing state-of-the-art methods in terms of reducing the overall system's power consumption.

Keywords: Data centers, data center networks, virtual machine (VM) placement, flow scheduling, joint power optimization.

1. Introduction

High power consumption in data centers has become a critical issue. As reported recently [1], the total power consumption of data centers worldwide has already reached the level of an entire country, such as Argentina or the Netherlands. High power consumption of data centers not only results in high electricity bills, increases carbon dioxide emissions, but also increases the possibility of system failures, because the corresponding power distribution and cooling systems are approaching their peak capacity.

Within a data center, IT equipment consumes about 40% of the total power.

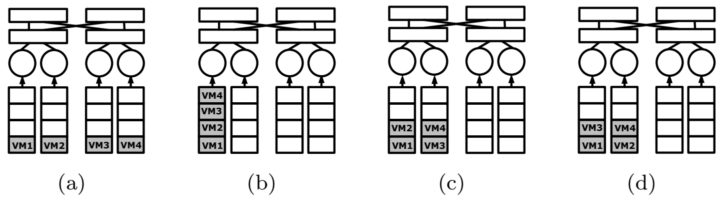


Fig. 1. Motivational Example

Two important components that consume the majority of IT power in data centers are the servers and the Data Center Network (DCN). In a typical data center, the servers take up the majority of the IT power (up to 90%) when they are fully utilized [2–4]. As servers are becoming more and more power efficient [5], the DCN, which mainly includes the switches that interconnect the servers, can consume an amount of power comparable to that of the servers, i.e., 50% of the IT power, when servers are at typical low utilization levels.

Most often, services are provided by data centers to customers through various virtualization technologies, and are presented as Virtual Machines (VMs). A set of VMs not only have computation requirements; they also require communications among themselves to complete the specified tasks. Intuitively, power management on the servers and in the DCN should be considered jointly to reduce the overall system's power consumption.

1.1. *Motivational Example*

We give an example that motivates our work in this paper. Some assumptions are not explicitly stated, and will be explained later. We consider a two-level network with four switches and four servers; they are connected as in one pod of the classic Fat-Tree [6] architecture. We consider ideal servers whose power consumption is $p(u) = p_0^v + u^2$ [7], where u is the utilization assigned to the server, and $p_0^v = 0.4$ is the static power of the entire server. The static power consumption is the power consumed by a server when it is assigned no workload. Each VM has a utilization requirement of 0.25. The only flows among the VMs are $flow_{1,3}$ from VM1 to VM3, and $flow_{2,4}$ from VM2 to VM4. Both of the flows are with bandwidth requirement 1. All the network links are bidirectional, and both directions have the same capacity, 1. Power consumption on a switch is a constant power $p_0^w = 0.2$ plus power consumption of its active ports. Each active port consumes $p^{port} = 0.02$. Notice that, we are always considering power consumption, instead of energy consumption, since the a data center is an always on infrastructure. For the same reason, we are always considering server utilization and bandwidth, instead of absolute work amount and data communication volume, respectively.

One extreme assignment is to distribute the VMs equivalently among all the servers, as shown in Fig.1(a); in this case, all the four switches should be powered on to satisfy the communication requirements of the two flows. The total power consumption of the system is $p_1 = 4 \times (0.4 + 0.25^2) + 4 \times 0.2 + 12 \times 0.02 = 2.89$.

Another extreme assignment is to power on just one server as shown in Fig. 1(b). In this case, $flow_{1,3}$ and $flow_{2,4}$ are contained in the server and do not need to travel through the DCN. Thus, no switches are required to be powered on. The total power consumption of the system is $p_2 = 0.4 + 1 = 1.4$.

Between the two extreme assignments, we can choose to power on half of the servers. If we assign VM1 and VM2 to one server, and VM3 and VM4 to another, as shown in Fig. 1(c), only one switch and its two ports needs to be powered on; the total power consumption is $p_3 = 2 \times (0.4 + 0.5^2) + 0.2 + 2 \times 0.02 = 1.54$. Since VM1 and VM3 have a communication requirement, and VM2 and VM4 have a communication requirement, intuitively, we should assign VM1 and VM3 to one server, and assign VM2 and VM4 to the other server, as shown in Fig. 1(d); in this case, the two flows do not need to travel through the DCN, and the total power consumption is $p_4 = 2 \times (0.4 + 0.5^2) = 1.3$.

Notice that, even for this simple example, we have various options regarding which servers and switches to power on. General problems with large numbers of VMs, servers, and switches will be more difficult to solve. Notice also that, Fig. 1(d) achieves the minimal power consumption; it provides us key intuitions: we should place VMs with large communication requirements close to each other, and care should be taken to decide how many servers should be powered on to accommodate the VMs.

1.2. *Contributions and Paper Organization*

In this paper, we jointly consider VM placement on servers with scalable frequencies and flow scheduling/consolidation in the DCN, to optimize the overall system's power consumption. Our main contributions are as follows.

- To the best of our knowledge, jointly considering servers with scalable frequencies and flow scheduling in the DCN lacks extensive research efforts. Since modern power/energy-optimized servers are usually designed to have scalable frequencies, our work contributes to power optimization in practical data centers.
- We prove that, given the number of servers to be used, to achieve minimal power consumption on servers, computation workloads of the VMs should be allocated to the chosen severs in a balanced way. This is due to the convex relation between a server's power consumption and its operating frequency. We further consider the flow requirements among the VMs during VM allocation and assignment, to reduce the power consumption on the DCN. Also, after VM placement, flow consolidation is conducted to reduce the number of active switches and ports.
- We propose to choose the optimal number of servers to be used to accommodate the VMs, based on the overall system power consumption, instead of just the power consumption on servers. Our proposal is based on the fol-

lowing two facts. Choosing the minimum number of servers to be used may result in high power consumption on servers, due to servers' increased operating frequencies. Choosing the optimal number of servers purely based on servers' power consumption leads to reduced power consumption on servers, but may increase power consumption on the DCN, resulting in an increased overall power consumption.

- We conduct various simulations to compare our joint power optimization method with several state-of-the-art ones. Results show that, our joint method helps to reduce the overall system's power consumption significantly, and outperforms existing state-of-the-art methods in terms of reducing the overall system's power consumption.

The rest of the paper is organized as follows. Related works are provided in Section 2. The system model is described in Section 3. In Section 4, we discuss our joint power optimization method for the problem in special cases, where servers have special operating frequencies and VMs have special computation requirements. In Section 5, we consider the general cases. Simulations are conducted in Section 6. Conclusions are made in Section 7.

2. Related Work

There have been a lot of existing works on power management solely on servers. A heuristic-based solution outline for the power-aware consolidation problem of virtualized clusters is presented in [8]. The problem of optimally allocating a power budget among servers in order to minimize mean response time is addressed in [9]. Reference [7] considers the relation between the server's performance and power consumption; based on the relation, a mixed integer programming model is applied to achieve improvement on power-efficiency while providing performance guarantees. In [10], the power consumption and application-level performances of servers are controlled in a holistic way so that explicit guarantees on both can be explicitly achieved. [11] considers energy management on heterogeneous data centers where the workloads on servers are highly dynamic; similarly to our work in this paper, the authors consider hierarchical energy management, combining dynamic voltage & frequency scaling and job dispatching.

A large number of works also consider power management in the DCN. [12] conducts traffic consolidation based on the correlation-aware analysis for data center network traffic flows. [13] conducts energy-aware flow scheduling in a distributed way, which can avoid the single point of failure problem and has good scalability. [14] discusses how to reduce energy consumption in high-density DCNs from a routing perspective; the key idea is to use as few network devices to provide the routing service as possible, with little to no sacrifice on the network performance. [15] presents a framework for DCN power efficiency; basically, it periodically tries to find a minimum power network subset that satisfies current traffic conditions. [16] considers the power optimization of DCNs in a hierarchical perspective; it establishes a two-level power optimization model to reduce the power consumption of DCNs by switching off network switches and links while guaranteeing full connectivity and maximum link utilization.

Quite a few works consider coordinated power management on both servers and DCNs. Many of the works are covered in a survey [17]. We discuss several works that are highly related to ours. [18] presents a data center scheduling methodology that combines energy efficiency and network awareness and aims to achieve the balance between individual job performances, traffic demands, and energy consumed by the data center. The correlation analysis in [12] is extended to also consider the VM utilization levels to ensure joint power optimization of the DCN and the servers [19]. [20] also considers joint power optimization through VM placement on servers and flows scheduling on the DCN; they assume that each VM can be assigned a set of specified servers. [21] presents a general framework for joint power optimization on servers and the DCN. The problem of achieving energy efficiency is characterized as a time-aware model, and is proven to be NP-hard. [22] points out several new directions on making the future data center more energy efficient. These existing works fail to fully utilize the power management techniques on servers and in the DCN at the same time. In this paper, we jointly consider VM placement on servers with scalable frequencies and flow scheduling in DCNs, to minimize the overall system's power consumption.

3. System Model and Problem Formulation

3.1. *Platform Model*

We consider the Fat-Tree DCN architecture, where each switch has K ports. Each *Top of Rack (ToR)* switch connects to $N_r = K/2$ servers, and $K/2$ *aggregate* switches. Each pod consists of $K/2$ ToR switches, and thus, $N_p = (K/2)^2$ servers. The architecture consists of K pods. At the top of the architecture are K *core* switches. The total number of servers is $N = K^3/4$, and the total number of switches is $W = 5K^2/4$. Fig. 2 shows a Fat-Tree architecture with $K = 4$.

3.1.1. *Server Power Consumption Model*

We assume that servers can work on a set of discrete frequencies $\{f_1, f_2, \cdots, f_F\}$, where F is the total number of available frequencies. Without loss of generality, we assume that these frequency values are sorted in ascending order. All frequency values are normalized by the maximum frequency, f_F, i.e., $f_F = 1$. We adopt the power consumption model for servers in [7]. We denote the power consumption of a server, when it is running CPU-intensive services, with normalized CPU utilization, u under frequency f by $p^v(u, f)$. Then, we have:

$$p^v(u, f) = p_0^v + uf^2, \tag{3.1}$$

where p_0^v is the static power that is independent of the CPU utilization and the operating frequency. Notice that, $p^v(u, f)$ represents the power consumption of the

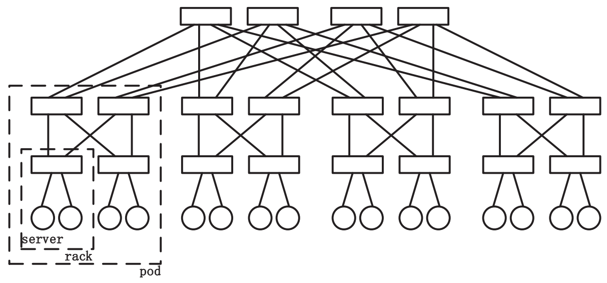


Fig. 2. The Fat-Tree architecture with $K = 4$. Rectangle represent switches, and circles represent servers.

entire server, instead of just the power consumption of the CPU. p_0^v can consist of static power of the CPU and the powers of various other devices, such as disks, RAMs, and network interface cards, etc. uf^2 can be regarded as the dynamic power consumption that is proportional to the square of the operating frequency. This power consumption model for the entire server has been verified to have high accuracy through various experiments [7].

3.1.2. *Switch Power Consumption Model*

We adopt the simplified switch power model as in [19]. The power consumption of a switch can be calculated as follows:

$$p^w = \begin{cases} p_0^w + n_a p^{port} & \text{, if the switch is on;} \\ 0 & \text{, if the switch is off.} \end{cases} \tag{3.2}$$

where p_0^w is the switch chassis power, n_a is the number of active ports on the switch, and p^{port} is the power consumption of one port. We do not consider link power consumption explicitly, because it can be easily incorporated into the switch port's power consumption.

3.2. *VM Model*

We consider a set of small application-specific VMs, $\mathcal{V} = \{v_1, v_2, \cdots, v_M\}$, where M is the total number of VMs. The VM model is abstracted from [7]. The services that the VMs provide are CPU-intensive; each VM consumes a fixed amount of CPU time while the frequency of the CPU is kept fixed. In other words, each VM requires a normalized utilization under the maximum frequency of the server, $u(v_i, f_F)$, i.e.,$u(v_i, 1)$. Then, the normalized utilization of v_i under frequency f, is $u(v_i, f) = u(v_i, f_F)f_F/f = u(v_i, 1)/f$. The normalization is conducted such that, the amounts of workloads that the VM completes during a certain amount of time at different frequencies are equal, i.e., $u(v_i, f)f = u(v_i, 1)$.

Denote $\mathcal{V}_j = \{v_{j,1}, v_{j,2}, \cdots, v_{j,|\mathcal{V}_j|}\}$ as the set of VMs assigned onto the the jth server. Denote $f^v(j)$ as the frequency of the jth server. For the jth server to fulfill

Table 1. Notations

Notation	Description
N, W	the numbers of servers and switches.
M	the number of virtual machines.
K	the number of ports on a switch.
$\mathcal{V}$	the set of all VMs.
$\mathcal{V}_j$	the set of VMs assigned to the jth server.
p_0^v	the constant power of a server when it is active.
$f^v(j)$	the jth server's execution frequency.
P_i^v	power consumption of the ith server.
$p^v(i)$	the total power consumption of the i chosen servers.
F	the number of available frequencies of every server.
p_0^w	the switch chassis power.
p^{port}	the per-port power consumption.
$s_{i,j}$	the status of the jth port of the ith switch.
$u(v, f)$	VM v's normalized utilization under frequency f.
$flow_{i,j}$	the flow from VM, v_i to VM, v_j.
$b(v_i, v_j)$	bandwidth requirement of $flow_{i,j}$.

the execution requirements of the virtual machines assigned onto it, the normalized utilization under $f^v(j)$ should also be less than or equal to 1. If the assigned workload utilization is greater than 1, the server has to switch to a higher frequency level. Thus, we have the following requirements for all the servers.

$$\sum_{i=1}^{|\mathcal{V}_j|} u(v_{j,i}, f^v(j)) \leq 1. \forall j = 1, 2, \cdots, N. \tag{3.3}$$

Given the assignments of VMs to servers, the power consumption on the jth server can be calculated as follows:

$$P_j^v = \begin{cases} p_0^v + \sum_{i=1}^{|\mathcal{V}_j|} u(v_{j,i}, f^v(j))(f^v(j))^2 & , |\mathcal{V}_j| \geq 1; \\ 0 & , |\mathcal{V}_j| = 0. \end{cases} \tag{3.4}$$

There may exist a flow from the ith VM to the jth VM, denoted by $flow_{i,j}$. We denote $b(v_i, v_j)$ as the bandwidth requirement of $flow_{i,j}$. Notice that $b(v_j, v_i)$ may not be equal to $b(v_i, v_j)$, because the two flows, $flow_{i,j}$ and $flow_{j,i}$ are with different directions. Also, $flow_{i,j}$ and $flow_{j,i}$ are independent from each other, and they can take different routing paths.

Given the assignments of VMs to servers, and given a routing path for all flows, the port status of each switch can be determined. Denote $s_{i,j}$ as the status of the jth port of the ith switch, $i = 1, 2, \cdots, W, j = 1, 2, \cdots, K$. Specifically, $s_{i,j} = 1$ if and only if the jth port of the ith switch is active, and $s_{i,j} = 0$ if and only if the jth port of the ith switch is off. Thus, the power consumption of the ith switch is:

$$P_i^w = \begin{cases} p_0^w + (\sum_{j=1}^{K} s_{i,j}) p^{port} & , \text{if } \sum_{j=1}^{K} s_{i,j} \geq 1; \\ 0 & , \text{if } \sum_{j=1}^{K} s_{i,j} = 0. \end{cases} \tag{3.5}$$

3.3. *Problem Formulation*

In a practical data center, for a time period, we are given a set of VMs, $\mathcal{V}$, and their communication bandwidth requirement matrix $b_{M\times M}$. We need to consider the following issues: i) allocate the VMs to servers, ii) set the frequency of each server, and iii) decide which switches and ports to use to support the communication requirements among all the VMs. The constraints are that the VMs' execution and communication requirements are met. Our final goal is to minimize the overall power consumption on both the servers and the DCN. The problem can be formulated as follows.

$$\min \quad \sum_{j=1}^{N} P_j^v + \sum_{i=1}^{W} P_i^w \tag{3.6}$$

$$s.t. \quad \sum_{i=1}^{|\mathcal{V}_j|} u(v_{j,i}, f^v(j)) \leq 1. \forall j = 1, 2, \cdots, N. \tag{3.7}$$

$$\sum_{j=1}^{N} |\mathcal{V}_j| = M; \tag{3.8}$$

$$\sum_{flow_{i,j} \in link_k} b(v_i, v_j) \leq 1, \forall \text{ all links in the DCN.} \tag{3.9}$$

Equation (3.6) is the objective. Constraints in (3.7) ensure that each server is assigned a workload with utilization less than 1, under the adopted frequency $f^v(j)$. Constraints in (3.8) actually guarantee that each VM is assigned to a certain server. Constraints in (3.9) require that the communication volume through each link does not exceed the capacity of the link. More detailed problem formulations have been presented in [19] and [20], and the problems have been shown to be NP-complete. Our problem is similar to the problem formulations in essence, and considers frequency scaling capabilities on servers; thus, this is also NP-complete. We do not provide the detailed formulations here, and just introduce several important notations and the basic ideas.

In the following section, we consider special cases, where servers have special discrete frequency levels, and VMs have special utilization requirements. We move onto the general cases in Section 5.

4. Special Cases

In this section, we assume that the difference between each server's two consecutive frequencies is uniform, and is equal to f_1. Thus, $f_i = if_1, \forall i = 1, \cdots, F$. We also assume that each VM's normalized utilization under the maximum frequency is f_1. Thus, a server at frequency f_i can host i VMs. If a server is assigned a certain number (greater than zero) of VMs, we require the server to be *on*; otherwise, we do not need to use this server, and can put this server in *off* state, consuming zero power. We consider the following question: what is the optimal VM allocation to optimize the power consumption on servers, given the number of servers to be used?

4.1. *Optimal VM Allocation Given the Number of Servers to Use*

We define the following property of a VM allocation:

Definition. A VM allocation is *balanced* if the numbers of VMs assigned to any two servers, which are on, differ no greater than 1; otherwise, the assignment is *unbalanced.*

Illustration. For example, if there are 10 servers in total, the first 7 servers are assigned 6 VMs each, the eighth server is assigned 5 VMs, and the last two servers are off, then, this assignment is balanced. However, if the eighth server is assigned 4 VMs, this assignment is unbalanced.

Theorem 4.1. An optimal VM allocation that minimizes the power consumption on servers is balanced.

Proof. We choose to prove it by contradiction. Assume that the optimal allocation, $\mathcal{A}^{opt}$, is unbalanced, then, there exist at least two servers, S_1 and S_2, whose assigned numbers of VMs, n_1 and n_2, respectively, differ greater than 1, i.e., $|n_1 - n_2| > 1$. Without loss of generality, we assume that $n_1 > n_2 + 1$. For the server with n_1 VMs, its optimal power consumption is $p_1 = p_0^v + (n_1 f_1)^2$, and for the server with n_2 VMs, its optimal power consumption is $p_2 = p_0^v + (n_2 f_1)^2$. Based on $\mathcal{A}^{opt}$, we develop another assignment, $\mathcal{A}^*$, by moving one VM on server S_1 to server S_2, while keeping other VM assignments on other servers unchanged. In $\mathcal{A}^*$, the power consumption on server S_1 is $p_1^{'} = p_0^v + ((n_1 - 1)f_1)^2$, and the power consumption on server S_2 is $p_2^{'} = p_0^v + ((n_2 + 1)f_1)^2$. The difference of the power consumption on two servers S_1 and S_2 in $\mathcal{A}^{opt}$ and $\mathcal{A}^*$ can be calculated as:

$$\begin{aligned} p_1^{'}+p_2^{'}-(p_1+p_2) =& ((n_1-1)f_1)^2 + ((n_2+1)f_1)^2 \\ & -((n_1 f_1)^2 + (n_2 f_1)^2) \\ =& 2f_1^2(n_2 + 1 - n_1) < 0. \end{aligned} \tag{4.1}$$

Thus, the overall power consumption of $\mathcal{A}^*$ is less than that of $\mathcal{A}^{opt}$, which contradicts the assumption that $\mathcal{A}^{opt}$ is an optimal allocation. Hence, this theorem is proven. □

4.2. *Server-Level VM Grouping*

Assume that we have chosen to use n servers. In the above subsection, we have shown that, in order to achieve the minimal power consumption on the n servers, the VM allocation should be balanced. In other words, we should allocate $\lceil M/n \rceil$ VMs to each of the first $(M \mod n)$ servers, and allocate $\lfloor M/n \rfloor$ VMs to each of the next $n - (M \mod n)$ servers. Since our goal is to minimize the system's overall power consumption, we also need to consider which VMs should be allocated to which servers, such that less numbers of switches and switch ports should be active to support the communication requirements among VMs, and that the power consumption of the DCN is minimized. The intuition is to assign a group of VMs that have high communication requirements among themselves to the same server, and assign VMs with low communication requirements to different servers. We define

Algorithm 1 Special Case VM Grouping – SCVG($\mathcal{V}, n$)

```
Input: The VM set, $\mathcal{V}$ and number of servers to use, $n$;
Output: A partition of the VMs to $n$ groups, $V_1^g, \cdots, V_n^g$;
1: $\mathcal{V}^* = \emptyset$; $V_i^g = \emptyset$, $\forall 1 \leq i \leq n$;
2: for $j = 1, \cdots, n$ do
3:     Find $v \in \mathcal{V}$, such that $v$ has the least weight with VMs in $\mathcal{V}^*$;
4:     $\mathcal{V} = \mathcal{V} - v$; $\mathcal{V}^* = \mathcal{V}^* + v$; $V_j^g = V_j^g + v$;
5: while $\mathcal{V} \neq \emptyset$ do
6:     Find the groups that have the least number of VMs, $V_{j_1}^g, \cdots, V_{j_{n^*}}^g$;
7:     for $i = 1, \cdots, n^*$ do
8:         Find $v_i^* \in \mathcal{V}$, such that $v_i^*$ has the greatest weight, $r_i$ with VMs in $V_{j_i}^g$,
   among all VMs in $\mathcal{V}$;
9:     $v^* = \arg_{v_i^*} \max\{r_i\}$;
10:    $\mathcal{V} = \mathcal{V} - v^*$; $V_{j_i}^g = V_{j_i}^g + v^*$;
```

the (communication) weight between two VMs, v_i and v_j, by $w(v_i, v_j)$, which can be calculated as the sum of the bandwidth requirements of $flow_{i,j}$ and $flow_{j,i}$:

$$w(v_i, v_j) = b(v_i, v_j) + b(v_j, v_i). \tag{4.2}$$

Thus, we need to partition the VMs to n *groups*, $V_1^g, V_2^g, \cdots, V_n^g$, such that the overall weight between VMs, which are not assigned to the same group, is minimized, and that the numbers of VMs in the n groups differ by at most 1. Notice that, each group V_i^g will be assigned to a server.

The above VM grouping problem falls into the class of the balanced minimum k-cut problem, which is a known NP-complete problem [23]. We develop a heuristic algorithm to solve it. First, all groups are initialized as empty, i.e., $V_i^g = \emptyset$, $\forall 1 \leq i \leq n$. Second, we add one VM to each of the n groups; during this process, the VMs are added sequentially. The VM to be added to the first group can be chosen randomly; after that, the VM is removed from the remaining VM set. To add a VM to the jth group, we choose the VM that has the minimum weight with VMs that have already been allocated. The process continues until each of the n groups has one and only one VM. The intuition is that VMs that have a low weight between them should be allocated to different groups. Finally, we allocated VMs to the groups one by one; in each step, we choose to add a VM to the group that has the least number of VMs. The VM to be added is chosen, such that the VM has the greatest weight to all the VMs in the group, among all other VMs. Since in each step, there might be several groups having the same minimum number of VMs, to break the ties, we choose the group that achieves the maximum communication requirement with the corresponding VM among the tying groups. Algorithm 1 gives the details of our algorithm.

4.3. *Rack-Level VM Grouping*

Notice that, given the number of servers to be used, n, the minimum number of racks that need to be used is $n_r = \lceil n/(K/2) \rceil$, where $K/2$ is the number of servers in a rack. In order to power on the minimum number of ToR switches, we choose to use exactly n_r racks. Another question still needs to be answered: how many servers should be powered on in each specific rack? The constraint of this problem is that, the number of servers in each rack is no greater than N_r. To avoid unnecessarily powering on the aggregate switches due to unbalanced numbers of active servers in racks, we adopt another balanced assignment: the first $(n \mod n_r)$ racks power on $\lceil n/n_r \rceil$ servers each, and the next $n_r - (n \mod n_r)$ racks power on $\lfloor n/n_r \rfloor$ servers each.

Next, we need to decide which groups from $V_1^g, \cdots, V_n^g$, should be assigned to the same rack. The intuition is similar: we intend to assign the groups with high communication requirements among themselves into the same rack, while assigning the groups with low communication requirements into different racks, to reduce the inter-rack communication. We define the weight between two groups, V_i^g and V_j^g as follows:

$$w(V_i^g, V_j^g) = \sum_{v_k \in V_i^g, v_l \in V_j^g} w(v_k, v_l). \tag{4.3}$$

Then, we need to partition the n groups into n_r balanced *parts*, $V_1^r, \cdots, V_{n_r}^r$. The same algorithm as that of Algorithm 1 can be applied to partition the n groups to n_r parts. Notice that, each part V_i^r will be assigned to a rack.

4.4. *Pod-Level VM Grouping*

We choose the minimum number of pods that should be used, $n_p = \lceil n_r/(K/2) \rceil$, where $K/2$ is the number of ToRs in a pod. We need to decide which parts should be allocated to the same pod. To avoid unnecessarily powering on core switches due to unbalanced numbers of active racks in pods, we adopt another balanced allocation: the first $(n_r \mod n_p)$ pods power on $\lceil n_r/n_p \rceil$ racks each, and the next $n_p - (n_r \mod n_p)$ pods power on $\lfloor n_r/n_p \rfloor$ racks each. We further need to partition the n_r parts into n_p *categories*, $V_1^p, \cdots, V_{n_p}^p$, such that the inter-category communication is minimized. We define the weight between two parts, V_i^r and V_j^r as follows:

$$w(V_i^r, V_j^r) = \sum_{V_k^g \in V_i^r, V_l^g \in V_j^r} w(V_k^g, V_l^g). \tag{4.4}$$

The same algorithm as that of Algorithm 1 can be applied to partition the n_r parts to n_p categories. Notice that each category, V_i^p will be assigned to a pod.

4.5. *VM Placement and Flow Scheduling*

Since the Fat-Tree architecture is a symmetric architecture, after the above hierarchical VM grouping process, we can just place the first category on the first pod, place the first part of the first category on the first rack of the first pod, and place the first group of each part on the first server in the corresponding rack. After this VM placement, we then consider assigning flows among VMs to switches, ports and links, to meet the communication requirements among all VMs. We assume that a flow cannot be split to take different paths.

Given the VM placement, each flow may have multiple paths to take. If a flow's source and destination VMs belong to different servers in the same rack, the path for the flow is unique. If a flow's source and destination VMs belong to different racks in the same pod, there are $K/2$ different shortest length paths. If a flow's source and destination VMs belong to different pods, there are $(K/2)^2$ shortest length paths. In order to reduce the power consumption for communications, a common intuition is to consolidate the flows to the minimal number of switches and ports, such that the remaining network elements can be put into sleep mode for power saving. We develop a heuristic based on the greedy bin packing strategy: for each flow between a pair of VMs, assign the flow to its leftmost path that has enough capacity for the flow. We conduct this assignment until we have assigned all the flows. Upon finishing, we can get the numbers of active aggregate switches, active core switches, and the corresponding ports that should be active. Thus, the total power consumption of the DCN can be easily achieved.

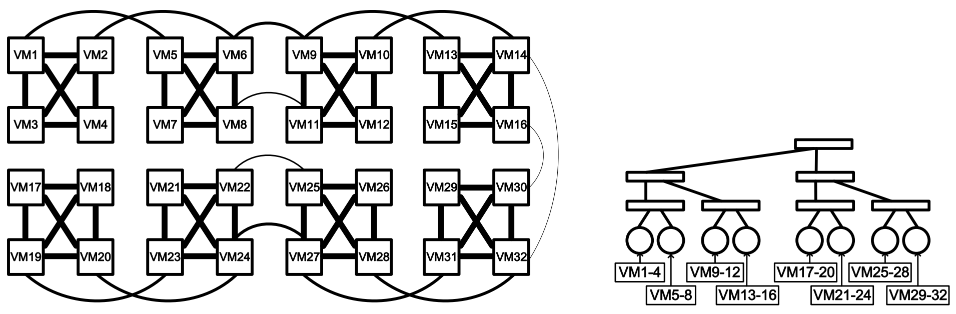
4.6. *Our Overall Approach*

In the previous subsections, we have assumed to use a given number of servers, i.e., n. In this subsection, we address how to choose the n to minimize the overall system's power consumption. Given the number of VMs to assign, $M, (M \leq NF)$, the minimum number of servers that should be used is $n^{min} = \lceil M/F \rceil$, and the maximum number of servers that can be used is $n^{max} = \min\{M, N\}$. If the number of used servers is $n, (n^{min} \leq n \leq n^{max})$, according to Theorem 4.1, in a balanced VM allocation, the optimal power consumption of the n servers can be calculated as follows.

$$\begin{aligned} p^v(n) =& (M \mod n)(p_0^v + ((\lceil M/n \rceil f_1)^2) \\ &+ (n - (M \mod n))(p_0^v + (\lfloor M/n \rfloor f_1)^2). \end{aligned} \tag{4.5}$$

A simple method to determine the number of servers to use is to purely base it on the power consumption of the servers, i.e., choosing the number that minimizes $p^v(n)$. However, our goal is to minimize the power consumption of the overall system, instead of the power consumption only of the servers. We denote the power consumption of the switches (if using n servers), after the hierarchical VM grouping and flow consolidation processes, by $p^w(n)$. We propose to determine the optimal number of servers to be used as follows.

$$n^{opt} = \arg_i \min_{i=n^{min}}^{n^{max}} \{p^v(i) + p^w(i)\}. \tag{4.6}$$



(a) 32 VMs and their communications.

(b) VM assignment.

Fig. 3. An illustration example. (a) The communication requirements among VMs based on the all the flows among VMs. Thick lines represent high bandwidth requirements, and thin lines represent low bandwidth requirements. (b) A final VM assignments to the servers, racks, and pods.

Given a problem instance, we first determine n^{opt} through the hierarchical VM grouping, and flow consolidation processes. Then, we take the corresponding VM grouping and flow consolidation strategies, that minimize the overall system's power consumption. This completes our joint power optimization method for the overall problem in the special cases.

4.7. *An Illustration Example*

Assume the we have 32 VMs to be assigned to a Fat-Tree with $K=4$, which can accommodate 16 servers. Assume that the optimal number of servers to be used is determined as $n^{opt}=8$. The communication requirements among VMs are shown in Fig. 3(a), where thick lines represent high bandwidth requirements, while thin lines represent low bandwidth requirements. First, the 32 VMs should be partitioned into $n^{opt}=8$ groups, such that the inter-group communications are minimized. After that, we need to partition the 8 groups into 4 parts, where each part will be assigned to a rack, such that the inter-part communications are minimized. Finally, we need to further partition the 4 parts into 2 categories, such that the inter-category communications are minimized.

5. General Cases

In general cases, servers may have an arbitrary set of discrete frequencies, and VMs can have different utilization requirements. Instead of considering the general cases directly, we first consider the ideal case, where servers can have continuous operating frequencies, and VMs can be split arbitrarily.

5.1. *Optimal VM Allocation Given the Number of Servers to Use*

Assume that the total utilization of all the VMs is U. If we have decided to use n servers to host the VMs, the optimal workload allocation is to evenly allocate the workload of VMs to the n servers. We have the following theorem.

Algorithm 2 General Case VM Grouping – SCVG($\mathcal{V}, n$)

```
Input: The VM set, $\mathcal{V}$ and number of servers to use, $n$;
Output: A partition of the VMs to $n$ groups, $V_1^g, \cdots, V_n^g$;
 1: $\mathcal{V}^* = \emptyset$; $V_i^g = \emptyset$, $\forall 1 \le i \le n$;
 2: for $j = 1, \cdots, n$ do
 3:     Find $v \in \mathcal{V}$, such that $v$ has the least weight with VMs in $\mathcal{V}^*$;
 4:     $\mathcal{V} = \mathcal{V} - v$; $\mathcal{V}^* = \mathcal{V}^* + v$; $V_j^g = V_j^g + v$;
 5: while $\mathcal{V} \neq \emptyset$ do
 6:     Find the groups that have the least VM utilization, $V_{j_1}^g, \cdots, V_{j_{n^*}}^g$;
 7:     for $i = 1, \cdots, n^*$ do
 8:         Find $v_i^* \in \mathcal{V}$, such that $v_i^*$ has the greatest weight, $r_i$ with VMs in $V_{j_i}^g$,
    among all VMs in $\mathcal{V}$;
 9:     $v^* = \arg_{v_i^*} \max\{r_i\}$;
10:     $\mathcal{V} = \mathcal{V} - v^*$; $V_{j_i}^g = V_{j_i}^g + v^*$;
```

Theorem 5.1. Given a set of VMs whose total utilization is U. The optimal way to allocate the workload to n servers is to allocate U/n to each of the servers, and each server operates on frequency U/nf_F, i.e., U/n.

Proof. Assume that the ith server is assigned workload $u_i \le 1$, $\forall 1 \le i \le n$. If server i operates on frequency $f^v(i)$, the assigned normalized utilization under frequency $f^v(i)$ should be less than or equal to 1, i.e., $u_i/f^v(i) \le 1$. The power consumption on server i is $P_i^v = p_0^v + u_i f^v(i)$. Since $u_i \le f^v(i) \le f_F = 1$, Server i's power consumption is minimized when $f^v(i) = u_i$. Thus, the minimal power consumption of server i is $P_i^v = p_0^v + u_i^2$. The power consumption of all severs is $p_g^v(n) = \sum_{i=1}^n P_i^v = np_0^v + \sum_{i=1}^n u_i^2$. Notice that $\sum_{i=1}^n u_i = U$. According to Jensen's inequality, $(\sum_{i=1}^n u_i^2)/n \ge (\sum_{i=1}^n u_i)^2/n^2$. Thus, $p_g^v(n)$ is minimized when $u_1 = u_2 = \cdots = u_n = U/n$. The minimal power consumption on the n servers is $p_g^v(n) = np_0^v + U^2/n$. □

5.2. *Our Overall Approach*

Theorem 5.1 indicates that, in the ideal situation, workload of VMs should be perfectly balanced and partitioned to servers. Since modern power-optimized servers usually have a considerable number of discrete frequencies and we are considering small application-specific VMs, it does provide a reference for how to allocate VMs to servers in practical situations.

For a practical situation, where servers have a set of discrete frequencies to operate on, and the VMs cannot be split, given the number of servers to be used, to achieve the workload balanced partition is already NP-hard [24] without considering the communications among VMs. We develop another heuristic based on the worst-fit partitioning scheme [24] to achieve workload balanced partition and reduce the communication requirement among the partitions. The algorithm is no different in essence from Algorithm 1. The only different is that, in each step, instead of choosing the groups with the minimum number of assigned VMs, the algorithm in the general case, will choose the group with the minimum assigned utilization, to add a corresponding VM. Algorithm 2 shows the algorithm for the general cases.

After that, the hierarchical VM grouping procedures, which allocate groups to parts and parts to categories, and the flow consolidation process, are essentially the same as those described in Section 4 for the special cases.

We propose to determine the number of servers to minimize the overall system power consumption as follows.

$$n^{opt} = \arg_i \min_{i=n^{min}}^{n^{max}} \{p_g^v(i) + p_g^w(i)\}, \tag{5.1}$$

where $p_g^v(i)$ and $p_g^w(i)$ is the power consumption of servers and switches, respectively, after the hierarchical VM placement and flow consolidation processes for the general cases.

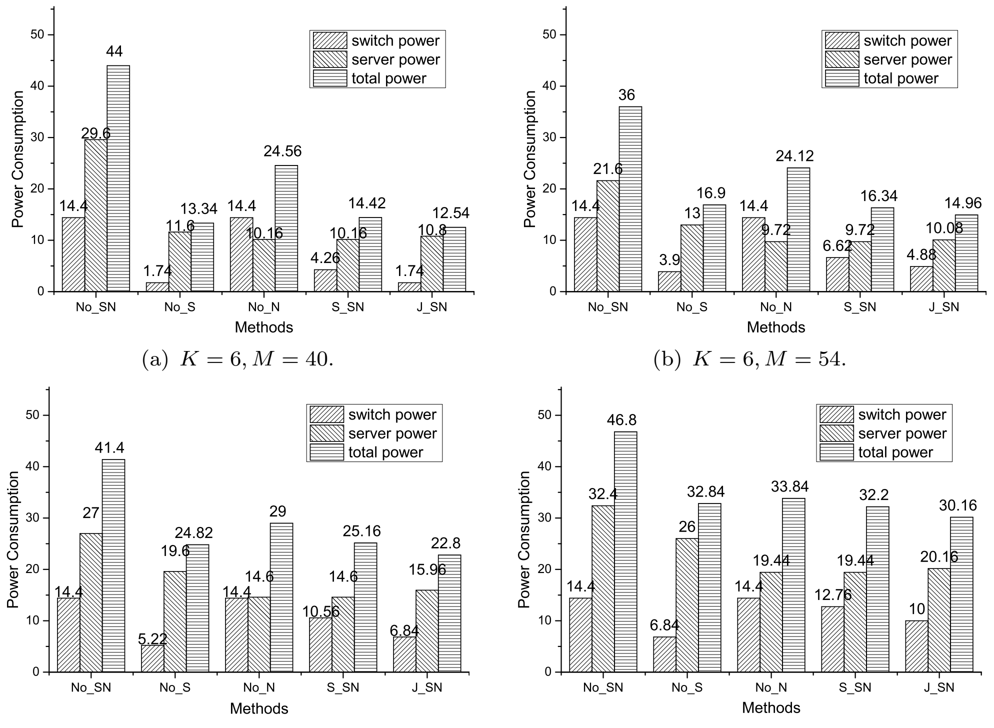
6. Simulations

We conduct extensive simulations to justify the power reductions of our proposed joint power optimization method. We denote our overall method by **J_SN**, because our method **J**ointly applies the power management techniques on the servers with **S**calable frequencies, and on the data center **N**etwork to minimize the overall system's power consumption. We compare our method with the following important methods.

No_SN: This method does not allow frequency scaling on servers, nor does it apply flow consolidation in the DCN. All servers operate on the maximum frequency. Power consumption on a sever is the static power p_0^v plus the utilization assigned to the server times 1. Notice that, we normalize all frequency values by the maximum frequency value, f_F. Also, all power consumption values are normalized by the maximum dynamic power consumption, i.e., $f_F^2 = 1$. In this method, all switches in the DCN are on, and all ports on each switch are active. Power consumption of a switch is equal to p_0^w plus power consumption of all active ports.

No_S: This method does not allow frequency scaling on servers; it does apply flow consolidation in the DCN. It chooses the minimum number of servers to accommodate all the VMs. All the chosen servers operate on the maximum frequency; other servers are powered off, and consume zero power. Algorithms similar to ours are applied to conduct balanced VM allocation and assignment, and flow scheduling in the DCN. Notice that, this method is the state-of-the-art workload consolidation method.

No_N: This method does apply frequency scaling on servers; it does not apply flow consolidation in the DCN. Based on the power consumption characteristics of servers, it chooses the optimal number of servers to power on, to minimize the power consumption on servers. However, all switches are powered on, and all ports



(a) $K = 6, M = 40$.

(b) $K = 6, M = 54$.

(c) $K = 6, M = 81$.

(d) $K = 6, M = 108$.

Fig. 4. Simulations for the special cases.

Table 2. Overall Power Consumption for Different Platform Parameters (Special Cases)

p_0^v	$p_0^w = 0.2$					$p_0^w = 0.4$					$p_0^w = 0.6$				
	No_SN	No_S	No_N	S_SN	J_SN	No_SN	No_S	No_N	S_SN	J_SN	No_SN	No_S	No_N	S_SN	J_SN
0.1	41.40	31.28	28.44	27.92	26.56	50.40	36.28	37.44	36.72	32.96	59.40	41.28	46.44	45.52	38.16
0.2	46.80	33.48	33.84	33.32	30.16	55.80	38.48	42.84	42.12	35.66	64.80	43.48	51.84	50.92	40.86
0.3	52.20	35.68	38.16	33.76	33.16	61.20	40.68	47.16	40.16	38.36	70.20	45.68	56.16	46.56	43.44
0.4	57.60	37.88	41.76	37.36	35.86	66.60	42.88	50.76	43.76	41.06	75.60	47.88	59.76	50.16	45.74
0.5	63.00	40.08	45.18	38.56	38.56	72.00	45.08	54.18	43.76	43.44	81.00	50.08	63.18	48.96	48.04

are active.

S_SN: This method applies both frequency scaling on servers and flow consolidation in the DCN. However, it first chooses the optimal number of servers to power on, to minimize the power consumption on servers, as in **No_N**. After that, algorithms similar to ours are applied to conduct balanced VM allocation and assignment, and flow consolidation in the DCN.

6.1. *Simulation Settings*

We conduct simulations for both special cases and general cases, for Fat-Tree architectures with $K = 6$. For an architecture with switch port number $K = 6$, the total number of servers is $N = 54$.

Table 3. Overall Power Consumption for Different Platform Parameters (General Cases)

p_0^v	$p_0^w = 0.2$					$p_0^w = 0.4$					$p_0^w = 0.6$				
	No_SN	No_S	No_N	S_SN	J_SN	No_SN	No_S	No_N	S_SN	J_SN	No_SN	No_S	No_N	S_SN	J_SN
0.1	42.09	32.46	32.32	31.36	29.91	51.09	37.46	41.32	39.95	36.11	60.09	42.46	50.32	48.55	41.63
0.2	47.49	34.76	37.34	35.98	33.21	56.49	39.76	46.34	44.38	39.14	65.49	44.76	55.34	52.77	44.03
0.3	52.89	37.06	41.80	39.10	36.43	61.89	42.06	50.80	46.49	41.63	70.89	47.06	59.80	53.89	46.43
0.4	58.29	39.36	45.19	40.77	39.23	67.29	44.36	54.19	47.17	44.03	76.29	49.36	63.19	53.57	48.83
0.5	63.69	41.66	47.63	42.83	41.63	72.69	46.66	56.63	49.03	46.43	81.69	51.66	65.63	55.23	51.23

6.1.1. *The Special Cases*

Each server has a set of five available frequencies: $f_1 = 0.2$, $f_2 = 0.4$, $f_3 = 0.6$, $f_4 = 0.8$, and $f_5 = 1$. Each VM's normalized utilization is 0.2. For each VM, we assume that it has a random number of flows going out of it to other VMs. The random numbers follow a distribution close to the normal distribution with the center being K.

To evaluate the performances of our proposed joint power optimization method under various computation workloads, we choose fixed values for p_0^v, p_0^w, and p^{port}. We set $p_0^v = 0.2$, i.e., the static power is 20% of the maximum dynamic power. We set $p_0^w = 0.2$, and $p^{port} = 0.02$. The settings are based on practical values, and similar settings have been applied in previous works [12]. As reported in [25], servers usually have a low utilization in data centers, though they are designed to accommodate the worst cases. We vary the number of VMs to be equal to the number of servers, 1.5 times the number of servers, and 2 times the number of servers. These correspond to different average data center utilizations, namely, 0.2, 0.3, and 0.4, respectively.

To evaluate the performances of our method under different data center configurations, we fix the number of VMs as $M = 108$, vary p_0^v as 0.1, 0.2, 0.3, 0.4, and 0.5, and vary p_0^w as 0.2, 0.4, and 0.6.

6.1.2. *The General Cases*

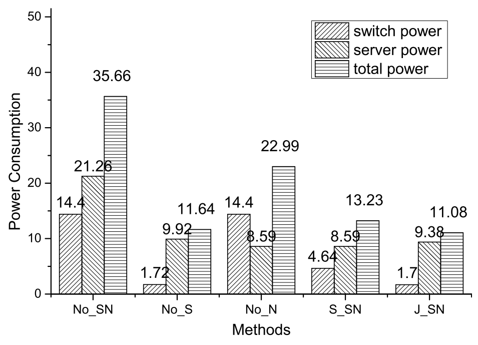
We also design simulations for the general cases. To verify that our joint method applies to general cases, we let servers have the following set of operating frequencies: $f_1 = 0.3$, $f_2 = 0.5$, $f_3 = 0.75$, $f_4 = 0,8$, and $f_5 = 1$, instead of the previous case where $f_i = if_1$. We randomly generate the VM's utilizations within [0.1, 0.3], with a mean value of 0.2.

To evaluate the performances of our method under various computation workloads, we choose fixed values for p_0^v, p_0^w, and p^{port}. We set $p_0^v = 0.2$, $p_0^w = 0.2$, and $p^{port} = 0.02$. The number of VMs is varied to be N, $3N/2$, and $2N$, respectively.

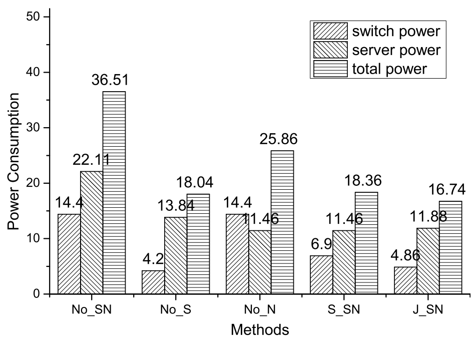
To evaluate the performances of our method under different data center configurations, we assume that $M = 108$ and $p^{port} = 0.02$, and vary p_0^v and p_0^w.

6.2. *Simulation Results and Analysis*

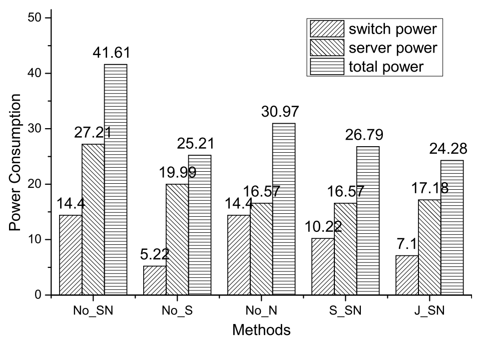
The simulation results for the special cases are shown in Fig. 4 and Table 2. As we can see from Fig. 4, servers consume the majority of the power, especially when frequency scaling on servers are not allowed. Thus, applying power management techniques on servers has great potential for power reduction. When frequency s-



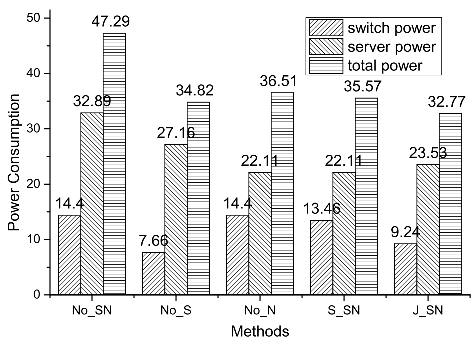
(a) $K = 6, M = 40$.



(b) $K = 6, M = 54$.



(c) $K = 6, M = 81$.



(d) $K = 6, M = 108$.

Fig. 5. Simulations for the general cases.

caling is enabled on servers, the power consumption on servers is reduced from 21.6 (**N_SN**) to 9.72 (**No_N**), which is about a 55.00% reduction. When flow consolidation and power management on switches are involved, the power consumption on switches is reduced from 14.4 (**No_SN**) to 3.9 (**No_S**), which is about a 72.92% reduction. However, since **No_N** and **No_S** do not apply power management both on servers and in the DCN at the same time, they still have a large overall power consumption. **S_SN** and **J_SN** apply power management both on servers and in the DCN at the same time. However, **S_SN** chooses the number of servers to use, to minimize the power consumption only on servers; its power consumption on switches is much greater than that of **J_SN**, though its power consumption on servers is lower than that of **J_SN**. Our **J_SN** method chooses the number of servers to minimize the overall power consumption; thus, it achieves the minimum overall power consumption. From Table 2, we can see that the power reduction achieved by **J_SN** is more significant when the static power consumption p_0^v is smaller. Since modern power/energy-optimized servers, especially in data centers, tend to have low static power consumption, **J_SN** can achieve a considerable power reduction in modern and future data centers. All simulation results show that our joint power optimization method achieves the minimum power consumption among all methods. This is because our method always target at reducing the overall system's power consumption.

The simulation results for the general cases are shown in Fig. 5 and Table 3. The power consumption of various methods demonstrate similar patterns to those in the special cases. These results verify that our joint power optimization methods works well in general cases.

7. Conclusion

In this paper, we consider joint power optimization through VM placement on servers with scalable frequencies and flow scheduling in DCNs. Due to the convex relation between a server's power consumption and its operating frequency, we prove that, given the number of servers to be used, computation workload should be allocated to the chosen severs in a balanced way, to achieve minimal power consumption on servers. To reduce the power consumption in the DCN, we further consider the flow requirements among the VMs during VM partitioning and assignment. Also, after VM placement, a flow consolidation is applied to reduce the number of active switches and ports. Extensive simulations show that, our joint power optimization method outperforms various existing state-of-the-art methods in terms of saving the system's overall power consumption.

References

1. W. Forrest, J. Kaplan, N. Kindler, Data centers: how to cut carbon emissions and costs, McKinsey Bus. Technol. 14 (10) (2008) 4–13.
2. D. Abts, M. Marty, P. Wells, P. Klausler, H. Liu, Energy proportional datacenter networks, in: ISCA, 2010, pp. 338–347.
3. B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, N. McKeown, Elastictree: Saving energy in data center networks, in: 7th USENIX NSDI, 2010, pp. 17–17.
4. R. Lent, Analysis of an energy proportional data center, Ad Hoc Networks 25, Part B (2015) 554–564.
5. L. A. Barroso, U. Holzle, The case for energy-proportional computing, Computer 40 (12) (2007) 33–37.
6. M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, in: ACM SIGCOMM Conf. on Data Comm., 2008, pp. 63–74.
7. V. Petrucci, E. Carrera, O. Loques, J. C. B. Leite, D. Mosse, Optimized management of power and performance for virtualized heterogeneous server clusters, in: 11th IEEE/ACM CCGrid, 2011, pp. 23–32.
8. S. Srikantaiah, A. Kansal, F. Zhao, Energy aware consolidation for cloud computing, in: 2008 Conference on Power Aware Computing and Systems, 2008, pp. 10–10.
9. A. Gandhi, M. Harchol-Balter, R. Das, C. Lefurgy, Optimal power allocation in server farms, in: Eleventh International Joint Conference on Measurement and Modeling of Computer Systems, 2009, pp. 157–168.
10. X. Wang, Y. Wang, Co-con: Coordinated control of power and application performance for virtualized server clusters, in: Quality of Service, International Workshop on, 2009, pp. 1–9.

11. E. Rotem, U. Weisser, A. Mendelson, A. Yassin, R. Ginosar, Energy management of highly dynamic server workloads in an heterogeneous data center, in: 2014 24th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS), 2014, pp. 1–5.
12. X. Wang, Y. Yao, X. Wang, K. Lu, Q. Cao, Carpo: Correlation-aware power optimization in data center networks, in: IEEE INFOCOM, 2012, pp. 1125–1133.
13. R. Liu, H. Gu, X. Yu, X. Nian, Distributed flow scheduling in energy-aware data center networks, IEEE Communications Letters 17 (4) (2013) 801–804.
14. Y. Shang, D. Li, M. Xu, Energy-aware routing in data center network, in: the First ACM SIGCOMM Workshop on Green Networking, 2010, pp. 1–8.
15. N. H. Thanh, B. D. Cuong, T. D. Thien, P. N. Nam, N. Q. Thu, T. T. Huong, T. M. Nam, Ecodane: A customizable hybrid testbed for green data center networks, in: Advanced Technologies for Communications, International Conference on, 2013, pp. 312–317.
16. Y. Zhang, N. Ansari, Hero: Hierarchical energy optimization for data center networks, in: IEEE ICC, 2012, pp. 2924–2928.
17. K. Bilal, S. Malik, O. Khalid, A. Hameed, V. W. E. Alvarez, R. Irfan, S. Shrestha, D. Dwivedy, M. Ali, U. Khan, A. Abbas, N. Jalil, S. Khan, A taxonomy and survey on green data center networks, Future Generation Computer Systems 36 (2014) 189–208.
18. D. Kliazovich, P. Bouvry, S. Khan, Dens: Data center energy-efficient network-aware scheduling, in: Green Computing and Communications, IEEE/ACM Int'l Conference on Cyber, Physical and Social Computing, 2010, pp. 69–75.
19. K. Zheng, X. Wang, L. Li, X. Wang, Joint power optimization of data center network and servers with correlation analysis, in: IEEE INFOCOM, 2012, pp. 1125–1133.
20. H. Jin, T. Cheocherngngarn, D. Levy, A. Smith, D. Pan, J. Liu, N. Pissinou, Joint host-network optimization for energy-efficient data center networking, in: IEEE IPDPS, 2013, pp. 623–634.
21. L.Wang, F. Zhang, J. A. Aroca, A. Vasilakos, K. Zheng, C. Hou, D. Li, Z. Liu, Greendcn: A general framework for achieving energy efficiency in data center networks, IEEE JSAC 32 (1) (2014) 4–15.
22. F. Chong, M. Heck, P. Ranganathan, A. Saleh, H. Wassel, Data center energy efficiency: Improving energy efficiency in data centers beyond technology scaling, Design Test, IEEE 31 (1) (2014) 93–104.
23. H. Saran, V. V. Vazirani, Finding k-cuts within twice the optimal, in: Foundations of Computer Science, 32nd Annual Symposium on, 1991, pp. 743–751.
24. H. Aydin, Q. Yang, Energy-aware partitioning for multiprocessor real-time systems, in: IPDPS, 2003, pp. 113–121.
25. P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, K. Salem, Adaptive control of virtualized resources in utility computing environments, in: 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems, 2007, pp. 289–302.