Joint Virtual Machine Assignment and Traffic Engineering for Green Data Center Networks

Lin Wang[†◇], Fa Zhang[†], Athanasios V. Vasilakos[‡], Chenying Hou[†◇], Zhiyong Liu[♮]
[†]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[‡]University of Western Macedonia, Athens, Greece
[◇]University of Chinese Academy of Sciences, Beijing, China
[♮]State Key Labotary for Computer Architecture, ICT, CAS, Beijing, China
{wanglin, zhangfa, houchenying, zyliu}@ict.ac.cn, vasilako@ath.forthnet.gr

ABSTRACT

The popularization of cloud computing brings emergency concern to the energy consumption in big data centers. Besides the servers, the energy consumed by the network in a data center is also considerable. Existing works for improving the network energy efficiency are mainly focused on traffic engineering, i.e., consolidating flows and switching off unnecessary devices, which fails to comprehensively consider the unique features in data centers. In this paper, we advocate a joint optimization for achieving energy efficiency of data center networks by proposing a unified optimization framework. In this framework, we consider to take advantage of the application characteristics and topology features, and to integrate virtual machine assignment and traffic engineering. Under this framework, we then devise two efficient algorithms, TE_VMA and TER, for assigning virtual machines and routing traffic flows respectively. Knowing the communication patterns of the applications, the TE_VMA algorithm is purposeful and can generate desirable traffic conditions for the next-step routing optimization. The TER algorithm makes full use of the hierarchical feature of the topology and is conducted on the multipath routing protocol. The performance of the overall framework is confirmed by both theoretical analysis and simulation results, where up to 50% total energy savings can be achieved, 20% more compared with traffic engineering only approaches.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Miscellaneous

Keywords

Energy Efficiency, Data Center Networks, Virtual Machine Assignment, Traffic Engineering.

1. INTRODUCTION

The widespread adoption of cloud computing leads to the extensive deployment of data centers. Because of the public concern of being green recently, it came to be realized that the energy consumed by those huge data centers is too tremendous to neglect. Consequently, achieving energy efficiency in such data centers becomes an important issue. When concentrating on the energy consumption of a data center, the first attention would be paid on the servers, since all tasks have to be processed by servers. However, when sophisticated energy-efficient techniques are involved in servers, the energy consumed by the huge amount of network devices (mainly switches) used for providing sufficient connection bandwidth between each pair of servers also emerges as a first-order concern. It has been pointed out by Abts *et al.* [1] that in a typical data center from Google the power consumed by the network takes a fraction of 20% of the total consumption when the servers are fully utilized, but this fraction will increase to 50% if the server utilization decreases to 15% which is a normal case in reality. Therefore, it is also eager to have efficient strategies for greening the network in data centers.

A significant amount of work has been done for approaching energy proportionality in data center networks. Among them, the most straightforward way consists in reducing the number of active network elements. This is generally achieved by consolidating flows and shutting down unnecessary devices (e.g. [11], [16], [19], [21]). The feasibility of this approach is formed by connectivity redundancy and traffic load variation in current data center networks. Apart from that, there are also some works aiming at designing energy-efficient topologies, providing similar connectivity while reducing the amount of used network devices [1, 12]. However, both directions of work are generally conducted without deeply exploring the unique features in data centers.

We observe that compared with traditional IP networks, networks used in data centers usually possess quite some differences, including regular communication patterns of applications, freedom of endpoints of traffic flows and particular structures of topologies. Firstly, the applications running in data centers are concentrated, most of which have very regular communication patterns. This observation has been confirmed by Xie *et al.* [20]. In that work, they profiled the network patterns of several MapReduce [6] jobs, including Sort, Word Count, Hive Join and Hive Aggregation which are very representative in cloud data centers. They observed that all the jobs generate considerable traffic during only 30%-60% of the entire execution. Moreover, the communication patterns of these applications can mainly be classified into three major categories: single peak, repeated fixed-width peaks and varying height and width peaks. Following this result, it is quite reasonable to model cloud applications with their communication patterns.

Secondly, the virtualized environment of data centers enables the free deciding of flow endpoints. In order to achieve

Greenmetrics '13 Pittsburgh, Pennsylvania USA

better resource utilization, applications are running in virtual machines hosted by physical servers in data centers. Since the traffic flows on the network are generated by these virtual machines, the endpoints of flows will be dependent on virtual machine assignment. If we allocate these virtual machines in a careful way, the traffic on the network can be shaped into a desirable condition, giving convenience to the energy saving efforts we will make on the network.

Lastly, the topology features such as symmetry, hierarchy in data center networks can help with designing topology-specific energy-saving strategies. Usually, data center networks have well-structured topologies which are particularly designed for guaranteeing throughput and quality-of-service. Given this, the energy optimization schemes conducted for general networks may not be efficient enough for data center networks. For instance, taking advantage of the hierarchical structure, the virtual machine assignment can be accomplished in a way such that the virtual machine pairs with high communication traffic are connected with short paths, resulting in smaller total traffic on the network and also smaller energy consumption. By exploiting these unique features in data centers, better energy-saving strategies can be introduced.

Inspired by the above properties, we carry out a unified framework to achieve energy efficiency in data center networks. In this framework, the application-level communication pattern enables better knowledge of the network traffic, eliminating the unprecise traffic prediction lying in old methods. In addition, we adopt both virtual machine assignment and traffic engineering, providing a substantial improvement to the state-of-the-art traffic engineering based optimization. Moreover, the algorithms for both virtual machine assignment and traffic engineering are devised based on a comprehensive learning of the topology features, which, as a consequence will have better efficiency compared to other general approaches. Both theoretical analysis and experimental results confirm the performance of the framework. With our framework involved, the energy savings achieved in data center networks can reach 50%, almost two times of old traffic engineering based solutions.

Our contributions are summarized as follows: 1) We propose a unified framework, where three significant unique features in data centers, as discussed above, are jointly considered; 2) By exploring these unique features, we devise two specific algorithms, TE_VMA and TER, for virtual machine assignment and routing optimization respectively; 3) We show by both analytical and extensive simulation that the proposed framework can help achieve further energy savings compared with traditional approaches.

The remainder of the paper is structured as follows. Section 2 defines and models the problem. Section 3 gives an overview of the framework, where both the energy-efficient virtual machine assignment and the routing problem will be tackled. Section 4 provides some experimental results that validate the performance of the efficiency, and also discusses the online extension of the framework.

2. THE MODEL

Data center. We describe a data center as a centralized computing system where a set of servers are connected by a specifically designed network. Assume there is a set $\mathbf{S}$ of servers. For the sack of tractability, we consider homogeneous data centers, i.e., all the servers in $\mathbf{S}$ are assumed to be identical. In order to improve the flexibility and resource utilization, jobs are processed in virtual machines that are hosted by servers. All the servers are connected by a network $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is the set of commodity switches and $\mathbf{E}$ is the set of links. In general, $\mathbf{G}$ is well structured with a particular topology such as FatTree [3, 13], BCube [8] or DCell [9], and all the switches in $\mathbf{G}$ are identical. In particular, our model and results will be based on FatTree topology where there are multiple pods each of which comprises of a fixed number of racks and a fixed number of servers are placed in each rack.

We use a refined speed scaling model to describe the power consumption fashion of network devices, where each switch consumes power superadditively with its load, and a fixed amount of power is consumed in the meantime [5]. Formally, for $v \in \mathbf{V}$, we are given a power function $f(x_v) = \sigma + \mu x_v^\alpha$ where $x_v > 0$ is the traffic load and σ, μ and $\alpha > 1$ are parameters[1]. The power σ is known as the startup power consumed by the chassis for keeping a switch active even when it is idle. For $x_v = 0$, we set $f(x_v) = 0$. Usually the chassis power takes a large portion of the total, i.e., σ is large [17]. The total power consumption of the network then can be expressed by

$$P = \sum_{v \in \mathbf{V}} f(x_v) = \sum_{v \in \mathbf{V}: x_v \neq 0} (\sigma + \mu x_v^\alpha)\,. \tag{1}$$

Applications. Applications running in data centers usually have regular communication patterns. Characterizing and utilizing this application-level traffic information will provide a better understanding to the traffic pattern in the network. Assume we are given a set $\mathbf{J}$ of jobs that needs to be processed simultaneously. Each job $j \in \mathbf{J}$ is attached with an arrival time T_j^a and a departure (finish) time T_j^d and also a set of N_j tasks each of which is supposed to be processed by a single virtual machine. Denote the set of virtual machines required for processing all the jobs as $\mathbf{M}$. Between any pair of the virtual machines belonging to the same job, there will be a traffic flow (if necessary) for exchanging data during the execution of the job. According to the traffic profiles of applications in data centers, we assume that the communication of a job is concentrated on some discrete time periods, during each of which the traffic generated by this job stay stable. We call each continuous communication-intensive period a *transfer*. Formally, for each job j, we assume a set

$$\mathbf{T}_j = \left\{ (T_{j1}^s, T_{j1}^t, \mathbf{B}_{j1}), ..., (T_{jL_j}^s, T_{jL_j}^t, \mathbf{B}_{jL_j}) \right\} \tag{2}$$

which contains L_j transfers that are given by 3-tuples, where L_j is the number of transfers. In each 3-tuple, the first two elements T_{jk}^s and T_{jk}^t $(1 \leq k \leq L_j)$ represent the start and end time of the k-th transfer respectively, and the third element $\mathbf{B}_{jk}$ denotes the traffic matrix of the virtual machines accordingly. Outside these periods, we assume that all the elements in the traffic matrix is negligible.

Problem description. Our goal is to assign all the virtual machines (jobs) to servers and route the traffic flows between virtual machines on network $\mathbf{G}$ such that the total energy consumption of the network during the execution of all the jobs is minimized.

[1]Since we consider identical switches, we assume uniform power functions $f_v(\cdot)$.

Let $\Delta_{sm} \in \{0,1\}$ indicate whether virtual machine $m \in \mathbf{M}$ is assigned to servers $s \in \mathbf{S}$. Then, we have the following constraints.

$$\sum_{s \in \mathbf{S}} \Delta_{sm} = 1, \quad \forall m \in \mathbf{M} \tag{3}$$

$$\sum_{m \in \mathbf{M}} \Delta_{sm} \cdot \gamma_m \leq \Gamma_s, \quad \forall s \in \mathbf{S} \tag{4}$$

where γ_m denotes the resources requirement of virtual machine m and Γ_s is the total resources of server s. The first constraint means that each virtual machine has to and can only be assigned to one server, while the second one means that the total resources required by all the virtual machines assigned to a server cannot exceed the total resources that a server can provide.

For a given time t, we denote the set of jobs that are in a transfer as $\mathbf{J}(t)$ where

$$\mathbf{J}(t) = \left\{ j \in \mathbf{J} \mid \exists k_j, t \in [T^s_{jk_j}, T^t_{jk_j}] \right\}. \tag{5}$$

Then, we obtain the set of traffic flows on the network at time t. Denote this set as $\Phi(t)$, we have

$$\Phi(t) = \{(s_1, s_2, \mathbf{B}_{jk_j}[m_1 : m_2]) \mid \Delta_{s_1 m_1} = 1 \wedge ... \\ ... \wedge \Delta_{s_2 m_2} = 1 \wedge s_1, s_2 \in \mathbf{S} \wedge j \in \mathbf{J}(t)\}. \tag{6}$$

Let $\Psi_{\phi e} \in \{0,1\}$ indicate whether traffic flow $\phi \in \Phi(t)$ is routed through link $e \in \mathbf{E}$ and $x_e(t)$ be the total load carried by link e. Then, we have

$$x_e(t) = \sum_{\phi \in \Phi(t)} \Psi_{\phi e} \cdot |\phi| \tag{7}$$

$$x_v(t) = \frac{1}{2} \sum_{\{e | v \text{ is incident to } e\}} x_e(t) \tag{8}$$

$$\Psi_{\phi e} : \text{flow conservation} \tag{9}$$

Let $T^a = \min_{j \in \mathbf{J}}\{T^a_j\}$ and $T^d = \max_{j \in \mathbf{J}}\{T^d_j\}$. Then, the objective can be presented as

$$\min E = \int_{T^a}^{T^d} P(t)dt = \int_{T^a}^{T^d} \sum_{v \in \mathbf{V}: x_v \neq 0} (\sigma + \mu x_v(t)^\alpha)dt. \tag{10}$$

Not surprisingly, this problem is NP-hard. This can be proved by a reduction from the quadratic assignment problem. Moreover, even with a determined assignment of virtual machines, the energy-aware routing on the network is also NP-hard and even has no bounded ratio approximations [5]. By utilizing the unique features in data centers, we propose a unified framework in which efficient heuristics are provided.

3. THE FRAMEWORK

In order to better seize the new opportunities brought by utilizing the unique features in data centers, we propose a unified framework for solving the network energy optimization problem. An overall picture of the framework can be found in Figure 1. In this section, we describe in detail the conduction of each component of the framework.

3.1 The Applications

We begin with exploring the characteristics of the applications as we have discussed before that the communication patterns of the applications appear with regularities. The communication patterns of applications can be obtained by profiling runs of jobs. By keeping more or less the same condition for profiling and production running, the parameters we obtained in profiling will be consistent for production runs. One possible realization of this kind of profiling can be found in [20]. The profiling process may bring ineluctable profiling overhead, but it can be drastically reduced if the same types of jobs with the same input size are run repeatedly, which is quite normal in cloud data centers for iterative data processing [2].

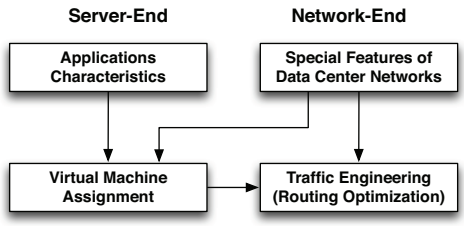

Figure 1: An overview of the unified framework for network energy optimization in data centers.

3.2 The Data Center Network

In order to provide better reliability and richer connectivity, many alternatives have been proposed to traditional 2N tree topology [3, 7, 13, 8, 9]. On one hand, data center networks are richly connected and can support any communication pattern with high bandwidth. On the other hand, the data center network can bring another benefit – the regularity of the topology. Most topologies follow a multi-tier tree architecture where the scalability is usually achieved by scaling up each individual switch. Since such topologies in different scales possess almost the same properties, the optimization efforts for small-scale networks can be easily adapted to extensive networks while requiring very slight changes, enabling us to make use of the unique features of the well-structured topologies to improve network performance by gaining insights from small-scale networks.

3.3 Virtual Machine Assignment

Virtual machine assignment provides the possibility of combining the characteristics of applications and traffic engineering. With the goal of improving the energy efficiency of the network, virtual machine assignment can be accomplished by integrating the characteristics of the running applications and the special features of the network topology. For example, if we know the communication patterns of applications, we can assign jobs in a way that the pairs of virtual machines with great traffic are assigned into servers with short paths. The traffic on the network will be accordingly reduced, as well as the energy consumption.

After observing the characteristics of the applications and the unique features of data center networks, we propose three main principles for virtual machine assignment in order to achieve energy efficiency of the network in data centers. There principles are either intuitions or derived based on the power model we have in Section 2. Due to the space limitation, we omit all the proofs here and provide them in the full version of the paper [18].

Principle 1: The optimal virtual machine assignment consists in compacting virtual machines into racks as tightly as possible in order to minimize the power consumption of

Algorithm 1 TE_VMA

Input: data center network topology $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, set of servers $\mathbf{S}$ and set of jobs $\mathbf{J}$

Output: assignment for every virtual machine $m \in \mathbf{M}$

1: **for** $j \in \mathbf{J}$ **do**
2: Transform virtual machines $\mathbf{M}$ to super virtual machines $\mathbf{M}_s$
3: **end for**
4: Cluster the jobs in $\mathbf{J}$ into $(N^{pod}+1)$ groups, i.e., $\mathbf{J} = (\cup_{n\in[1,N^{pod}]}\mathbf{J}_n) \cup \mathbf{J}_{N^{pod}+1}$
5: **for** $1 \le n \le N^{pod}$ **do**
6: Partition the super virtual machines for each job $j \in \mathbf{J}_j$ into K parts using the min-k-cut algorithm
7: Assign the super virtual machines of each job into servers according to the partition
8: **end for**
9: Assign the virtual machines of jobs in $\mathbf{J}_{N^{pod}+1}$ into vacant servers in the first N^{pod} pods flexibly.

top-of-rack switches.

Principle 2: Distributing the virtual machines from the same job into k racks will incur less power consumption compared to compacting them into one single rack, where α is the exponent in the power function, and $4^{\frac{\alpha}{\alpha-1}} < k \le K$, and $K > 4^{\frac{\alpha}{\alpha-1}}$ is the total number of racks in one pod.

Principle 3: As long as there are enough resources in one pod, the optimal assignment will never split virtual machines from the same job into different pods.

Based on the three principles, the virtual machine assignment is accomplished by a Traffic-aware Energy-efficient Virtual Machine Assignment algorithm TE_VMA which is presented in Algorithm 1. The algorithm takes a set of jobs (virtual machines) with communication patterns and a set of servers as input, and returns the job (virtual machine) assignment.

We first carry out a transformation from virtual machines to super virtual machines, since the multiple allocation of virtual machines in one server will increase the assignment complexity. This transformation is based on one important observation that compacting the virtual machines with high communication traffic will reduce the total traffic on the network, as well as the power consumption. We define a referential traffic matrix $\mathbf{B}_j^{ref}$ for each job $j \in \mathbf{J}$, where

$$\mathbf{B}_j^{ref}[x:y] = \sum_{i=1}^{L_j}(T_{ji}^t - T_{ji}^s)\cdot \mathbf{B}_{ji}[x:y]. \quad (11)$$

For each job j, we run the following process iteratively: in each iteration, we choose the greatest value in matrix $\mathbf{B}_j^{ref}$. Suppose the element we choose is in the x_1-th row and y_1-th column. Then, we combine the x_1-th and y_1-th virtual machines by updating the matrix using the following equations, assuming $x_1 < y_1$ without loss of generality.

$$\mathbf{B}_j^{ref}[x_1:y] \leftarrow \mathbf{B}_j^{ref}[x_1:y] + \mathbf{B}_j^{ref}[y_1:y] \quad (12)$$

$$\mathbf{B}_j^{ref}[x:x_1] \leftarrow \mathbf{B}_j^{ref}[x:x_1] + \mathbf{B}_j^{ref}[x:y_1] \quad (13)$$

$$\mathbf{B}_j^{ref}[x_1:x_1] \leftarrow 0 \quad (14)$$

After this update, we remove the y_1 row and the y_1 column from matrix $\mathbf{T}_j^{ref}$. Then, we choose again the greatest value in the x_1-th row or column, and combine the corresponding virtual machines as we have done above. We call the combination of multiple virtual machines as a super virtual machine. Repeat this procedure until no virtual machines can be combined into one super virtual machine anymore due to the resource limitations. Note that some special restrictions can also be introduced in order to satisfy application-specific requirements, such as parallelism and I/O concerns.

Secondly, we cluster jobs into different pods. We assume that the servers in one pod are sufficient for accommodating the virtual machines from each job. If not, we can always assign "big" jobs (which cannot be accommodated in one single pod) and then consider the assignment problem with "small" jobs. From principle 1 and 3 we know that the total number of pods has to be minimized. We estimate the number of pods to be used by summing up the resources requested by all the jobs (based on the number of super virtual machines). Assume N^{pod} pods are needed in total to handle all the jobs. Then, we cluster the set of jobs into N^{pod} subsets using a revised k-means clustering algorithm. The underlying intuition is that consolidating jobs with strongly different communication patterns into the same pod will help improve the utilization of the network, as well as reducing the peak traffic in the network.

To this end, we first define a traffic pattern vector $\mathbf{B}_j^{pat}$. Each dimension of $\mathbf{B}_j^{pat}$ indicates the average traffic between any two virtual machines of job j in each unit of time, which is calculated as follows.

$$B_j^{avg}(t) = \begin{cases} \dfrac{\sum_{x,y}\mathbf{B}_{jk}(t)[x:y]}{|\mathbf{B}_{jk}|/2}, & \text{for } t \in [T_{jk}^s, T_{jk}^t]; \\ \varepsilon, & \text{otherwise;} \end{cases} \quad (15)$$

where ε is infinitesimal to indicate the background traffic. Then, the traffic pattern vector can be expressed as

$$\mathbf{B}_j^{pat} = (B_j^{avg}(T^a), B_j^{avg}(T^a+1), ..., B_j^{avg}(T^d)), \quad (16)$$

Given two jobs $j_1, j_2 \in \mathbf{J}$ with traffic pattern vectors $\mathbf{B}_{j_1}^{pat}$ and $\mathbf{B}_{j_2}^{pat}$ respectively, we define the distance between the two jobs as

$$dis(j_1, j_2) = dis(\mathbf{B}_{j_1}^{pat}, \mathbf{B}_{j_2}^{pat}) = \frac{1}{||\mathbf{B}_{j_1}^{pat} - \mathbf{B}_{j_2}^{pat}||_2}. \quad (17)$$

This definition of distance supposes that any two jobs with similar traffic patterns will have a large distance between them. The clustering algorithm then partitions all the jobs into some sets where jobs in each set will be assigned to one pod. The algorithm works as follows: 1) Choose N^{pod} jobs and put them into sets $\mathbf{J}_n$ for $n \in [1, N^{pod}]$ with one job per set. Use the traffic pattern vectors of these jobs as center vectors $\mathbf{C}_n$ of these sets. We adopt this center initialization step directly from the refined k-means++ algorithm which can guarantee an approximation ratio $O(\log N^{pod})$ in expectation. 2) For each of the remaining jobs j, find the nearest set n with respect to the distance $dis(\mathbf{B}_j^{pat}, \mathbf{C}_n)$. If this job can be accommodated into this set without violating the resources limitation in one pod, put this job into set $\mathbf{J}_n$. 3) Update the center vector of each set n by averaging all the traffic pattern vectors of jobs in $\mathbf{J}_n$.

$$\mathbf{C}_n = \frac{\sum_{j\in\mathbf{J}_n}\mathbf{B}_j^{pat}}{|\mathbf{J}_n|} \quad (18)$$

Procedure 2) and 3) will be repeated until all the jobs have been assigned. If there are some jobs that cannot find any set to accommodate, put them into an extra set $\mathbf{J}_{N^{pod}+1}$.

Algorithm 2 TER

Input: data center topology $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, set of servers $\mathbf{S}$ and the assignment of virtual machines
Output: routes of flows

```
for t ∈ [T^a, T^d] do
    Obtain the traffic flows on the network at time t according to the virtual machine assignment
    for n ∈ [1, N^pod] do
        Estimate the number N_n^agg of aggregation switches that will be used in the n-th pod
        Choose the first N_n^agg aggregation switches
    end for
    Estimate the number N^core of core switches that will be used and choose them
    Use multipath routing to distribute all the flows evenly on the network formed by the selected switches
    Turn the unused switches into energy-saving mode
end for
```

Thirdly, we choose $N^{pod}+1$ empty pods and assign the jobs in each set to one pod respectively. According to principle 2, we distribute the super virtual machines from the same job into multiple racks. Since allocating the super virtual machines with higher traffic into the same rack will reduce the total network traffic, the problem becomes to partition the set of super virtual machines into K subsets such that the summation of the traffic between any two subsets in the partition is minimized. This is equivalent to the well-know minimum k-cut problem which requires finding a set of edges whose removal would partition a weighted graph to k connected components and the weight sum of these removed edges is minimized. The partition algorithm we use here is adapted from the one used in [15].

3.4 Energy-Efficient Routing

We now focus on the traffic engineering in the network to achieve energy conservation. Before providing the energy-efficient routing algorithm, we study the relation between energy consumption and routing strategy.

In order to achieve energy saving, the following two questions need to be answered: how many active switches are sufficient and how to distribute the traffic flows among these active switches? As long as we know the answer to the first question, the second question is then easy to be answered. With a fixed number of switches active, the best way to achieve energy saving is to distribute the network traffic evenly on all the active switches according to the super-additively way power being consumed. This even distribution of traffic now can be done by many multi-path routing protocols such as Equal Cost Multi-Path (ECMP) and Valiant Load Balancing (VLB), because of the rich connectivity existing in data center networks. Some more sophisticated techniques such as Hedera [4] and MPTCP [10, 14] can also be applied to ensure uniform traffic spread despite flow length variations.

The first question can be answered by the following theorem. Due to the space limitation, we omit the proof here.

THEOREM 1. *In the optimal energy saving solution, the number of active switches in the network is minimized.*

Based on the answers to the above two questions, we propose a Two-phase Energy-efficient routing algorithm TER,

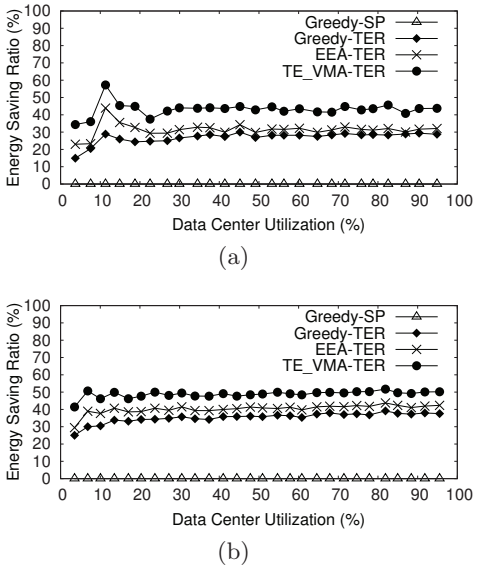

Figure 2: Energy saving ratios using different VM assignment and routing algorithms in two data center networks in different scales with (a) 1024 **and (b)** 3456 **servers connected respectively.**

as presented in Algorithm 2. For each unit of time in $[T^a, T^d]$, we repeat the following two phase. In the first phase, the algorithm aims to find a subset of switches in a bottom-up manner. The estimation of the number of active devices is accomplished by a simple calculation where we divide the total traffic by the capacity of each device. However, as it can happen that the multipath routing algorithm may not evenly distribute the traffic flows perfectly, we use the first fit decreasing algorithm, a good approximation algorithm for bin packing problem where we treat the flows as objects and the capacity of each switch as the bin size, to ensure that all the traffic flows can be carried by the selected switches. In the second phase, we borrow some multi-path routing protocol to obtain the routes of the flows in the network with the selected switches active, and the others will be powered down or turned into some low-power state. As the virtual machine assignment is accomplished by taking advantage of the application-level traffic pattern, the status of most switches will maintain the same during most of the time while the network traffic variates. According to the routes of flows, the routing tables are generated and sent to corresponding switches in runtime by a centralized OpenFlow controller.

4. SIMULATION AND DISCUSSION

We validate TE_VMA and TER algorithms with synthetic input and compare the energy consumption with 3 solutions generated by other algorithms, the greedy assignment and shortest path routing solution (Greedy-SP), the greedy assignment and TER solution (Greedy-TER), the energy-efficient assignment (without virtual machine to super virtual machine transformation) and TER solution (EEA-TER). All the values are normalized by the results of Greedy-SP, which can be seen in Figure 2.

We observe that a) up to 30% energy reduction can be achieved by applying the TER algorithm; b) the well de-

signed virtual machine to super virtual machine transformation is very helpful; c) compared to the traffic engineering based solutions, the purposeful virtual machine assignment can bring us 20% more savings; d) the total energy saving ratio can be as large as 50% when the whole framework with TE_VMA and TER algorithms are used. In the meantime, we notice that the running times of our algorithms are also reasonable compared with other approaches.

The simulation results confirm that the room for saving energy in current data center networks remains quite large, and besides traffic engineering, a huge amount of energy can be further saved by a joint optimization of applications, virtual machine assignment and network routing, which is the goal of our framework.

We also believe that the proposed framework and algorithms can be adapted to online cases very conveniently. One simple idea is that for each job arrival, we first apply the super virtual machine transformation, and then compute the distances between this job and other jobs that are running in the data center using their aligned traffic patterns. According to the distances, we assign this job, and then the rest of our framework can be applied directly. We leave a deliberated design for online cases as future work.

5. CONCLUSIONS

In this paper, we study the energy saving problem in data center networks with a comprehensive consideration of the unique features in data centers. By exploring the communication patterns of the applications and the regularities of the network topology, we propose a unified framework where virtual machine assignment and traffic engineering are integrated in order to take advantage of those features. We also devise two specific algorithms for virtual machine assignment and network routing respectively. The performance of the framework, as well as the proposed algorithms, has been confirmed by both theoretical analysis and simulation results. Up to 20% further energy savings can be achieved compared with traffic engineering only approaches.

6. ACKNOWLEDGMENT

This research was partially supported by NSFC Major International Collaboration Project 61020106002, NSFC&RGC Joint Project 61161160566, NSFC project grant 60921002 and 61202059, and the Comunidad de Madrid grant S2009TIC-1692, Spanish MICINN/MINECO grant TEC2011-29688-C02-01.

7. REFERENCES

[1] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu. Energy proportional datacenter networks. In *ISCA*, 2010.

[2] Agarwal, Sameer, Kandula, Srikanth, Bruno, Nicolas, Wu, Ming-Chuan, S. Ion, and J. Zhou. Re-optimizing data-parallel computing. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, NSDI'12.

[3] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. In *SIGCOMM*, 2008.

[4] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat. Hedera: Dynamic flow scheduling for data center networks. In *NSDI*, pages 281–296, 2010.

[5] M. Andrews, A. Fernández Anta, L. Zhang, and W. Zhao. Routing for energy minimization in the speed scaling model. In *INFOCOM*, pages 2435–2443, 2010.

[6] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. In *OSDI*, pages 137–150, 2004.

[7] A. G. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta. Vl2: a scalable and flexible data center network. In *SIGCOMM*, pages 51–62, 2009.

[8] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu. Bcube: a high performance, server-centric network architecture for modular data centers. In *SIGCOMM*, 2009.

[9] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu. Dcell: a scalable and fault-tolerant network structure for data centers. In *SIGCOMM*, pages 75–86, 2008.

[10] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. F. Towsley. Multi-path tcp: a joint congestion control and routing scheme to exploit path diversity in the internet. *IEEE/ACM Trans. Netw.*, 16(6):1260–1271, 2006.

[11] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown. Elastictree: Saving energy in data center networks. In *NSDI*, pages 249–264, 2010.

[12] L. Huang, Q. Jia, X. Wang, S. Yang, and B. Li. Pcube: Improving power efficiency in data center networks. In *IEEE CLOUD*, pages 65–72, 2011.

[13] R. N. Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat. Portland: a scalable fault-tolerant layer 2 data center network fabric. In *SIGCOMM*, pages 39–50, 2009.

[14] C. Raiciu, S. Barré, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley. Improving datacenter performance and robustness with multipath tcp. In *SIGCOMM*, pages 266–277, 2011.

[15] H. Saran and V. V. Vazirani. Finding k cuts within twice the optimal. *SIAM J. Comput.*, 24(1), 1995.

[16] Y. Shang, D. Li, and M. Xu. Energy-aware routing in data center network. In *Green Networking*, 2010.

[17] N. Vasic, P. Bhurat, D. M. Novakovic, M. Canini, S. Shekhar, and D. Kostic. Identifying and using energy-critical paths. In *CoNEXT*, page 18, 2011.

[18] L. Wang, F. Zhang, J. A. Aroca, A. V. Vasilakos, K. Zheng, C. Hou, D. Li, and Z. Liu. GreenDCN: a general framework for achieving energy efficiency in data center networks, In *JSAC*, accepted.

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

[20] D. Xie, N. Ding, Y. C. Hu, and R. R. Kompella. The only constant is change: incorporating time-varying network reservations in data centers. In *SIGCOMM*, pages 199–210, 2012.

[21] Y. Zhang and N. Ansari. Hero: Hierarchical energy optimization for data center networks. In *ICC*, pages 2924–2928, 2012.