J-OPT: A Joint Host and Network Optimization Algorithm for Energy-Efficient Workflow Scheduling in Cloud Data Centers

Amanda Jayanetti and Rajkumar Buyya
Cloud Computing and Distributed Systems (CLOUDS) Laboratory
School of Computing and Information Systems
The University of Melbourne, Australia
Email: amjayanetti@student.unimelb.edu.au, rbuyya@unimelb.edu.au

***Abstract*—Workflows are a popular application model used for representing scientific as well as commercial applications, and, cloud data centers are increasingly used in the execution of workflow applications. Existing approaches to energy-efficient workflow scheduling in cloud computing environments have primarily focused on the optimization of server utilization. The majority of works have ignored the impact of scheduling decisions on the data center network (DCN). However, studies have revealed that the DCN consumes 10-20% of the total data center power, and, this percentage could rise much higher depending on the utilization level of the data center. This paper proposes an energy-efficient workflow scheduling approach (J-OPT) that jointly optimizes the power consumption of servers and networking elements in cloud data centers. J-OPT considers precedence constraints and data dependencies among workflow tasks as well as communication requirements among task instances in the formulation of topology-aware scheduling decisions. The proposed approach is evaluated using synthetic and real world workflow traces in a simulated environment. Results of the experiments demonstrate that J-OPT outperforms state-of-the-art algorithms in terms of total power savings by 8% and 30% under high and low data center utilization levels, respectively.**

***Keywords*-energy efficiency; workflow scheduling; topology aware scheduling**

I. INTRODUCTION

Cloud computing has emerged as a defacto platform for efficiently delivering computing services to consumers on a pay-as-you-go basis. Due to the ever-increasing popularity of the cloud computing paradigm, the number and scale of cloud data centers have significantly grown over the last decade. With the rapid expansion in the size of data centers, the power consumed by datacenter elements has significantly increased imposing a considerable strain on the environment as well as on profit maximization goals of cloud providers [1]. It has been estimated that the energy consumption of cloud data centers in the worst case can reach 8000 TWh by 2030 [2]. Hence, significant research efforts have been rendered by both academia as well as industry to devise energy-efficient resource management and scheduling techniques for cloud data centers.

While servers are the major source of power consumption in data centers, data center networks (DCNs) also account for 10%-20% of total power consumption. This percentage could rise as high as 50% in data centers with energy-proportional servers, under light job loading conditions [3].Therefore, energy consumed by datacenter networks and networking devices that facilitate communications among hundreds of thousands of concurrently executing instances is a non-trivial factor that contributes to increasing the overall energy consumption of data centers considerably. Furthermore, over-utilized network devices lead to the creation of congestion hotspots resulting in undesirable packet losses, and imbalanced use of network links reduces the overall utilization of the data center networks. Scheduling algorithms that are agnostic to the communication patterns of underlying workloads are unlikely to be efficient at exploiting the power savings that can be achieved by the joint optimization of compute and networking elements in cloud data centers.

Before the emergence of software-defined cloud data centers, joint optimization was a challenging task that was not commonly adopted due to inherent complications associated with the integrated consolidation process. With the emergence of software-defined cloud data centers, this is no longer a complicated task as all the data collected by monitoring tools are available in real time at a centralized controller, and, the decisions made by the analysis of integrated monitoring data can be enforced on the physical devices through a software layer [4].

The focus of this paper is on cloud workloads that can be represented as Directed Acyclic Graphs (DAGs). Not only scientific workflows but also a wide variety of batch workloads and distributed applications can be modeled using the DAG execution model [5]. Although a large body of literature on workflow scheduling techniques in cloud environments exist, only a few have considered energy efficiency as a primary objective. Amongst the studies that have considered energy efficiency as a primary objective, only a handful has attempted to optimize the utilization of servers and networking elements in an integrated manner.

Furthermore, a vast majority of the energy-efficient workflow scheduling approaches proposed in the literature focus on the problem of scheduling a single workflow or multiple workflows that belong to a single user on a fixed or variable number of virtual instances. With the growing popularity of multi-tenant public cloud platforms [6], the need for advanced scheduling techniques that are capable of satisfying diverse concurrent resource requirements of workflows submitted by multiple tenants in an energy-efficient manner has emerged.

Motivated by the aforementioned opportunities, we propose a topology-aware scheduling algorithm that jointly optimizes the utilization of computing and networking elements for energy-efficient scheduling of workflows, in multi-tenant public cloud platforms. To the best of our knowledge, this is the first

work to perform a thorough disaggregated analysis of the power consumption behavior of workflow executions in cloud computing environments.

The rest of the paper is organized as follows: In section 2, we review the literature on state-of-the-art approaches related to the scope of this paper. In section 4, we formulate the power model and energy optimization problem. In section 5, we present the proposed algorithm. Followed by this, we present the performance evaluation of the algorithm in section 6. Finally, in section 7, we conclude the paper with a summary and suggestions for future work.

II. RELATED WORK

The problem of workflow scheduling in cloud computing environments has been extensively studied in a large number of research studies [7]. Hence, we have limited the scope of this literature review to focus on a set of selected workflow scheduling algorithms that consider energy efficiency as a primary objective. We have also reviewed a number of joint host and network optimization algorithms which have not been oriented towards a specific application type.

A. Energy Efficient Workflow Scheduling

Energy efficient workflow scheduling algorithms proposed in literature can be broadly classified into two categories: Meta-heuristic based algorithms and heuristic algorithms. For instance, multi objective particle swarm optimization has been used in a study by S. Yassa et. al. [8] to schedule workflows considering the objectives of minimizing cost, execution time and maximizing energy efficiency. DVFS technique has been used in this study to minimize energy consumption. Mezmaz et. al. [9] also used DVS techniques in a bi-objective study aimed at minimizing makespan and energy consumption in task scheduling. A bi-objective hybrid genetic algorithm has been used in this study to achieve the conflicting goals. Although scheduling algorithms based on meta-heuristics are capable of attaining better solutions compared to list based and clustering based algorithms, they are less appropriate for highly dynamic cloud environments due to high computational costs and time complexities.

A number of other heuristic algorithms have also been proposed for energy efficient workflow scheduling. Zotkiewicz et. al. [10] presented an energy and communication aware scheduling strategy for SaaS applications in a cloud data center by modeling the applications as dynamic workflows. The proposed scheduling strategy operates with the objectives of minimizing energy consumption and average makespan of all submitted workflows. Although inter task communication aspects are considered in this study, network awareness has only been incorporated as a secondary condition used in the event of a tie between multiple equally energy efficient servers. In a study by M. Sharifi et. al. [11] with similar objectives, a two phase solution (PASTA) has been proposed for scheduling workflows with the objective of minimizing energy consumption and the average makespan of all submitted workflows. In the first phase tasks are ordered to minimize schedule length and in the second phase the pre ordered tasks are reordered such that they are executed on a set of computing resources with the least power consumption. However, this study does not extensively consider inter-task communication aspects in scheduling decisions.

In a study by X. Xu et. al. [12] an energy aware resource allocation has been presented in which task requests are always allocated to the host with lowest baseline energy consumption. This study relies on the assumption that all the instances of a task should be scheduled on the same physical server to minimize the cost of inter-task communication. While, this approach would incur gains when all instances can be accommodated on a single server, it will fail to support workflows with tasks which contains hundreds of thousands of parallel instances. H. Chen et. al. [13] attempted to improve energy efficiency by dynamically scaling up and down computing resources based on the rate of arrival of workflows. Tasks are assigned to computing resources that can meet the deadline requirements with minimal resource wastage. This study relies on the assumption that all the inter-machine communications are performed without contention. This assumption does not hold well in practice.

B. Topology Aware Energy Efficient Scheduling

DENS methodology introduced by D. Kliazovich et. al [14], presents an energy efficient scheduling method with network awareness which monitors the status of network elements (switches, links) and incorporates their feedback to scheduling decisions. This study considers the tradeoffs between consolidating workloads on to a minimum number of servers and the resulting impact on hardware reliability of servers and the possibility of the creation of hotspots in data center networks. In a study with similar objectives by Cao et. al. [15], a scheduling method which consists of three subroutines each of which is solved by a different algorithm has been proposed. In the first subroutine, VMs are sorted in the order of decreasing computational resources and packed into a set of virtual hosts. The assignments of VMs to virtual hosts are readjusted in the second subroutine to minimize inter-host traffic and in the third subroutine virtual hosts are mapped to physical hosts following a greedy algorithm.

S. Vakilinia [16] proposed a joint optimization approach for power minimization in large-scale multi-tenant cloud datacenters taking into account the power consumption of servers, network communications, VM migrations, heterogeneity of servers and workloads as well as resource and bandwidth constraints. They have modeled the optimization problem with IQP (Integer Quadratic Programming). This work suggests that rather than performing VM placement and migration in two steps, higher efficiencies can be achieved by jointly considering placement and migration decisions in each scheduling iteration. A joint host-network optimization method for energy efficient VM consolidation is presented in [17]. In order to achieve the objectives of optimal server placement and network flow routing, the VM placement problem is converted to a routing problem and a single solution is developed to address both

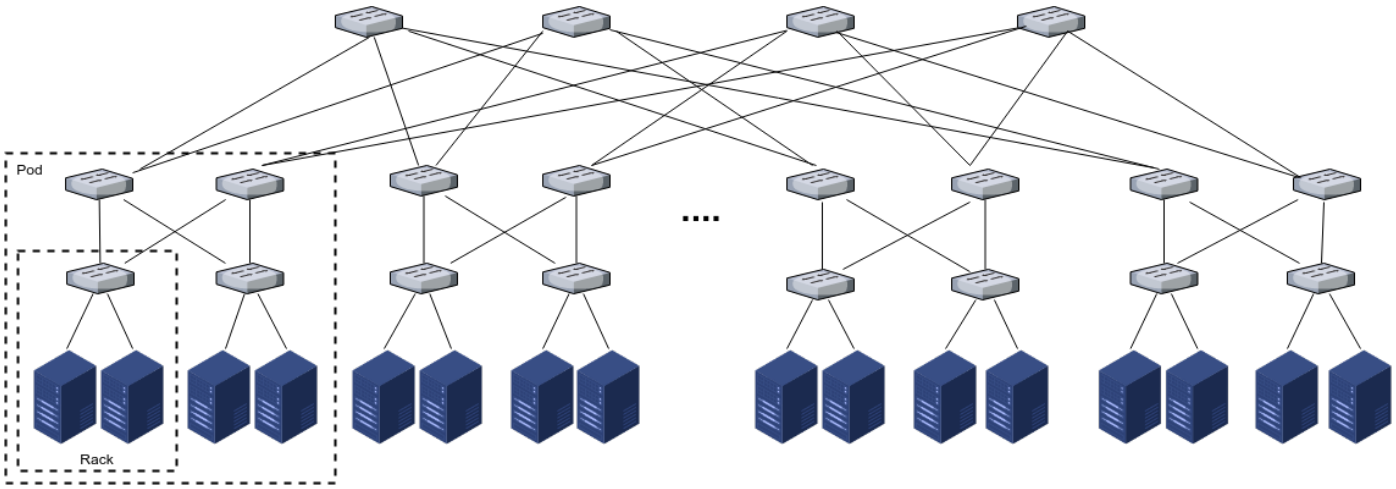


Fig. 1: Fat tree network topology

requirements. All these approaches have focused on the general problem of joint host and network optimization, and, they have not considered the potential power savings achievable through fine tuning the algorithms to suit the characteristics and communication patterns of the underlying workloads.

The workflow scheduling algorithms reviewed in section A have attempted to improve the energy efficiency of task schedules with little or no consideration about the impact of scheduling decisions on the data center network. In contrast, we attempt to enhance overall energy efficiency by jointly optimizing the utilization of servers as well as switches used in workflow executions. Our method also differs from topology aware resource allocation methods reviewed in section B, since we take into account the distinct features of workflows in the formulation of scheduling decisions.

III. PROBLEM MODELING

A workflow can be modeled as a Directed Acyclic Graph (DAG) $G = (V, E)$, where $V = T_1, T_2, ..T_n$ is the set of tasks and E is the set of edges which represent the data dependencies among tasks in a workflow and the weight of an edge $e_{i,j} = (T_i, T_j)$ represents the size of data to be transmitted from T_i to T_j. The edge $e_{i,j}$ also represents a precedence relation between tasks T_i and T_j such that T_i is the parent task of T_j and T_j is the child task of T_i. Accordingly, the execution of T_j can only start after the execution of T_i is completed. In this study, we have considered a divisible task model in which a task is composed of one or more instances which can be scheduled to execute in parallel on one or more physical resources. The execution of a task is considered to be complete when all the instances of it has finished execution.

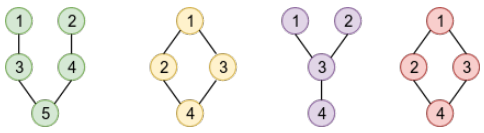


Fig. 2: Example workflows

The objective of this study is to schedule workflows in an energy efficient manner by jointly minimizing the power consumption of hosts, switches and links in the data center. In this section we formulate the joint server and network power optimization as a mono-objective optimization problem.

A. Power Model

Notations used in this paper for the formulation of optimization problem is presented in table VIII. For the calculation of power consumption of servers, we used the CPU utilization based power model presented in [18]. Accordingly, power consumption of server i is defined as shown in the following equation.

$$P_i^{server} = \begin{cases} P_i^{idle} + (P_i^{dynamic} - P_i^{idle}).u_i, & \text{if } u_i > 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

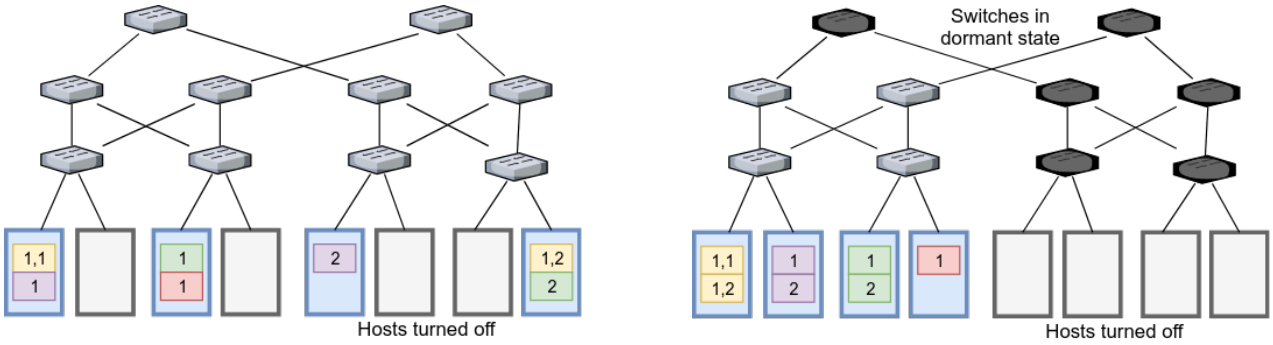
Idle power consumption is a constant factor which incurs irrespective of the utilization level of a server and it can only be eliminated by turning off the servers. Dynamic power consumption of a server can be accurately computed by considering that the CPU utilization and the power consumption of a server follows a linear relationship [18].

The power model presented in [19] is used to compute the power consumption of switches. This model computes the power consumption of a switch as the sum of static power and the port power based on the number of active ports as shown in the following equation.

$$P_k^{switch} = \begin{cases} P_k^{static} + P_k^{port}.n_k, & \text{if switch } k \text{ is on} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

B. Problem Formulation

The focus of this work is on system wide minimization of energy consumed by servers and networking elements. Hence, the optimization objective can be formulated as:



(a) Topology unaware scheduling

(b) Topology aware scheduling

Fig. 3: A comparison of topology aware and unaware energy efficient workflow scheduling approaches

Table. I: Problem Notation

Notation	Description
M	Total number of servers
N	Total number of switches
u_i	CPU utilization percentage of server i
P_i^{server}	Power consumption of server i
P_i^{idle}	Idle power consumption of server i
$P_i^{dynamic}$	Peak power consumption of server i
P_k^{switch}	Power consumption of switch k
P_k^{static}	Power consumption of switch k without traffic
P_k^{port}	Power consumption of each port of switch k
n_k	Number of active ports on switch k

$$\text{Minimize:} \sum_{i=1}^{M} P_i^{server} + \sum_{k=1}^{N} P_k^{switch} \tag{3}$$

Note that we have ignored the energy consumption of external and internal communications among tasks from the problem formulation since they are negligible in comparison to the total power consumption of servers and switches [12].

IV. PROPOSED J-OPT ALGORITHM

Scheduling techniques that are agnostic to the impact of resource allocation strategies on the DCN are less effective in terms of total power savings that can be achieved by workload consolidation. Aforementioned scenario is illustrated through the simple example in Figure 3. Figure 3a and Figure 3b illustrate possible outcomes of a scheduling iteration in which workflows shown in Figure 2 are scheduled by a topology-unaware and a topology-aware energy-efficient resource allocation technique, respectively. This example demonstrates the manner in which the topology-aware resource allocation technique achieves comparatively more power savings by putting unused networking elements into dormant state.

The proposed topology-aware heuristic-based algorithm, J-OPT aims to schedule precedence constrained tasks in an energy efficient manner by jointly optimizing the utilization of servers and networking elements.

A. Task Prioritization

Upon the submission of a new job, the computational as well as communication requirements of the tasks are analyzed and a rank is assigned to each task accordingly. The rank is an indication of a tasks priority, and is applicable only within the context of a job. If multiple ready tasks from the same job falls into a scheduling iteration, they are ordered based on rank and inserted into the global task queue. The scheduler respects the insertion order of task queue when resource allocations are made and operates in a FCFS (First Come First Serve) basis to ensure fairness in multi-tenant environments. However, the arrival order is disregarded if a task with a higher priority arrives at the priority queue.

Task prioritization uses the concept of upward rank presented in the well-known low complexity algorithm HEFT [20]. Although, HEFT is a good algorithm with high performance, it does not consider energy efficiency as an objective. In each scheduling iteration, ready tasks of a job are ordered in decreasing order of upward rank before being submitted to the global task queue. The upward rank of a task is computed recursively using the following equation:

$$rank_u(T_i) = \overline{w_i} + \max_{T_j \in succ(T_i)} (\overline{c_{i,j}} + rank_u(T_j) \tag{4}$$

where $\overline{w_i}$ is the average computation cost of task T_i and $\overline{c_{i,j}}$ is the average communication cost of edge $e_{i,j} = (T_i, T_j)$. For the exit task (a task with no children), rank can be computed as follows:

$$rank_u(exit) = \overline{w_{exit}} \tag{5}$$

The average communication cost of an edge $e_{i,j}$ can be computed as follows:

$$\overline{c_{i,j}} = \overline{L} + \frac{data_{i,j}}{\overline{R}} \tag{6}$$

where $\overline{L}$ is the average communication start up time and $\overline{R}$ is the average communication rate among servers and $data_{i,j}$ is the amount of data to be transferred from task T_i to task T_j.

Algorithm 1 J-OPT

Input: PM: List of servers
Input: $G(V, E)$: Data center network
Input: R_i: Total resource requirements of task T_i
Input: PM_{conn}: List of hosts in which predecessors of task T_i with data dependencies executed
Output: $allocMap$: Resource allocation map

1: $subgraphList \leftarrow \emptyset$
2: $subgraphList \leftarrow NEIGHBOR\text{-}GEN(PM, PM_{conn}, G(V, E))$
3: **for** each $PM_{cand} \in subgraphList$ **do**
4: Order servers in PM_{cand} in descending order of desirability score
5: **while** $PM_{cand} \neq \emptyset$ **do**
6: $P \leftarrow PM$ with highest desirability score in PM_{cand}
7: **for** each $r \in R_i$ **do**
8: **if** allocation(r, P) is successful **then**
9: **if** $P \notin PM_{conn}$ **then**
10: $PM_{conn} \leftarrow PM_{conn} \cup P$
11: Recompute desirability scores for remaining servers in PM_{cand} with equation 10
12: $R_i \leftarrow R_i - r$
13: $PM_{cand} \leftarrow PM_{cand} - P$
14: **if** $R_i = \emptyset$ **then**
15: **return** true
16: **return** false

Algorithm 2 NEIGHBOR-GEN

Input: PM: List of all servers
Input: PM_{root}: A list of servers, the sub-graphs of which are to be generated
Input: $G(V, E)$: Data center network
Output: A list of neighbor subgraphs of servers in PM_{root}

1: $currLevel \leftarrow 1$
2: $subgraphList \leftarrow \emptyset$
3: Mark all $H \in PM$ as unassigned
4: **while** $currLevel < 3$ **do**
5: **for** each $P \in PM_{root}$ **do**
6: $neighSubgraph \leftarrow \emptyset$
7: **if** P is unassigned **then**
8: $neighSubgraph \leftarrow \{P\}$
9: **for** each unassigned $H \in PM$ **do**
10: $paths \leftarrow$ getPaths$(P, H, currLevel)$
11: **if** $paths \neq \emptyset$ **then**
12: $neighSubgraph \leftarrow neighSubgraph \cup \{H\}$
13: Mark H as assigned
14: $subgraphList \leftarrow subgraphList \cup neighSubgraph$
15: $currLevel \leftarrow currLevel + 1$
16: **for** each unassigned $H \in PM$ **do**
17: $subgraph \leftarrow subgraph \cup \{H\}$
18: $subgraphList \leftarrow subgraphList \cup subgraph$ ▷ append the $subgraph$ of all unassigned servers to $subgraphList$
19: **return** $subgraphList$

B. Network Aware Resource Allocation

In this section we present the topology-aware resource allocation technique used for mapping computing resources to task instances. The data center network can be represented as an undirected graph $G(V, E)$ where V is the set of vertices and E is the set of edges. The set of vertices represent both servers as well as switches and the set of edges represent communication links between pairs of switches and between servers and switches.

Tasks in a workflow may have data dependencies giving rise to inter-task communications and associated communication costs. Furthermore, in the multi-instance task model considered in this work, a single task may have multiple instances that can be mapped on to one or more physical nodes and the instances may be communicating with each other during the period of execution.

To minimize the cost of communication, resource allocation algorithm should attempt to place instances of the same task on physical nodes that are as close as possible to each other in terms of the number of network hops. The placement is further complicated since it should be such that the aggregate distance to physical nodes in which the predecessor tasks with data dependencies executed is minimized. Accordingly, this resource allocation problem reduces to a variant of the proven NP hard problem in [21].

Algorithm 1 operates with the local optimization perspective of placing communicating instances of a particular workflow on servers which are in close proximity such that the total number of networking elements used during the execution of a workflow is minimized. To achieve this, we introduce the concept of a dynamically expanding set of servers (PM_{conn}) which is provided as an input to the algorithm. PM_{conn} is initialized with the set of servers in which predecessors of the current task with data dependencies executed. PM_{conn} expands dynamically as more and more servers are selected for satisfying the resource requests of a newly arrived task.

Next we introduce the concept of neighbor subgraph of a server at a pre-defined hierarchical level in the DCN. Hierarchical level is defined in a topology specific manner and in this study we have considered a fat tree topology [22] with 3 hierarchical levels (Rack, Pod and DCN) as indicated in Figure 1. For a particular server, the neighbor subgraph at level 1 constitute the set of servers on the rack in which the server resides. Neighbor subgraph at level 2 includes the set of servers located in the pod (group of racks) to which the server belongs.

Algorithm 2 is used to obtain the set of neighbor subgraphs of the servers in PM_{conn} at hierarchical levels 1 and 2 of the DCN. Depending on the topology of the data center network, Algorithm 2 can be replaced with a more sophisticated subgraph generation method. It should be noted that the subgraph generation overhead can be completely eliminated by pre-computing the subgraphs of all servers at each hierarchical level of the DCN.

Lines 3-15 of Algorithm 1 attempts to find an allocation that can satisfy the total resource requirements of the current task by iterating through the neighbor subgraph list ($subgraphList$). In each iteration, the set of servers in a neighbor subgraph are considered as the candidate server list (PM_{cand}) for resource allocation. The desirability score described in section C is computed for each server in the PM_{cand} list, and the server with the highest score is selected first. The algorithm attempts to assign as many resource requests as possible to the selected server. With each successful allocation if the considered server is not currently in PM_{conn}, then it is added to PM_{conn}. Desirability scores are recomputed for remaining elements of PM_{cand} set per each new addition to PM_{conn}.

C. Desirability Score

The desirability score is used to rank the servers by considering the power efficiency of a server based on its current utilization and its physical location with respect to the set of servers in PM_{conn} . Performance per watt is used to determine the energy efficiency of a server. Servers become most energy efficient when they are operating close to maximum capacity without being over-utilized [23]. Based on this observation, the desirability score is designed to favor servers that are more loaded compared to others without being over utilized.

It is also important that the server selection is not agnostic to the impact of the new assignment on network utilization. Hence, the new server additions to PM_{conn} should minimize the use of links and switches, such that energy savings can be realized by putting the unused links and switches that do not carry traffic into power saving mode. Accordingly, the desirability score also aims to better align the traffic distribution by favoring the selection of servers that minimize the aggregate physical distance between the final PM_{conn} list in terms of the number of hops.

In order to achieve both aforementioned goals, we formulate the desirability score as a bi-objective function which combines server utilization efficiency and network utilization efficiency with the use of a normalized weight factor (α) that indicates which factor should be given more prominence for minimizing overall energy consumption. The first term of the bi-objective function is based on HEROS [23] which in turn is based on DENS [14]. It is composed of the product of the server selection function $H_s(l)$ and communication potential function $Q(u)$ shown below:

$$H_s(l) = P_s(l) * (1 - \gamma . \frac{\langle 1 \rangle}{\langle (1 + \exp^{-\frac{\mu}{maxl_s}} (l - \beta . maxl_s)} \quad (7)$$

where $P_s(l)$ is the performance per watt of the server at load l and the second term is a sigmoid function which is aimed at preventing the over utilization servers. Following the reference work [22], values 110, 0.9 and 1.2 were used for the coefficients μ, β and γ in our experiments.

The communication potential is based on the actual link load u and maximum link capacity U_{max}, and is defined as:

$$Q(u) = \exp^{-}(\frac{2u}{U_{max}})^2 \quad (8)$$

The second term of the desirability score is the distance function which is the aggregate total distance in terms of the number of network hops from current server to the set of servers in $PMconn$. It is defined as:

$$D(i) = \sum_{j=1}^{W} d_{i,j} \quad (9)$$

where W is the total number of servers currently present in PM_{conn} of task T_i, and, $d_{i,j}$ is the aggregate distance in terms of the number of network hops between the current server and the set of servers in which all instances of task T_j executed.

Finally, the desirability score which represents the desirability of a server to be selected for scheduling one or more instances of a task T_i is defined as:

$$\alpha . \frac{H_s(l).Q(u)}{H_f} + (1 - \alpha) . \frac{(D(i) + \epsilon)^{-1}}{D_f} \quad (10)$$

where H_f and D_f are two factors introduced to normalize the server selection function and distance function. ϵ is a very small positive real number.

V. PERFORMANCE EVALUATION

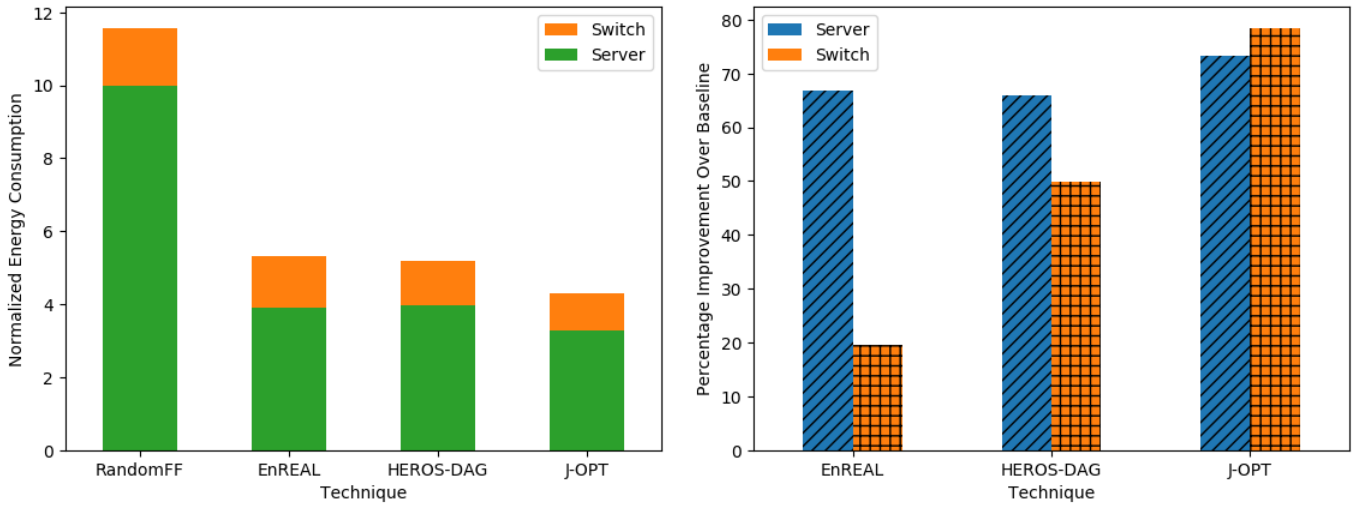
We evaluated the proposed algorithm in a simulated environment. For comparison purposes, we used three algorithms. One is a baseline algorithm based on greedy first fit, which does not attempt to improve either server utilization efficiency or network utilization efficiency. The next algorithm is based on the independent task scheduling algorithm HEROS [23]. HEROS is designed to perform a random server selection if multiple equally desirable servers are among eligible candidates for resource allocation with respect to a decision function. We have adapted HEROS to cater to the requirements of workflows and incorporated network awareness by extending the server selection mechanism based on the physical location of the server as suggested in [10] for tie-breaking. Accordingly, in case of a tie, the server which is within a minimum hop distance to servers in which predecessors of the current task executed are selected from amongst the eligible candidates. The extended algorithm is referred to as HEROS-DAG in this paper. The third algorithm referred to as EnREAL is a state of the art algorithm specifically formulated for energy-efficient scheduling of workflows in cloud computing environments.

A. Simulation Environment

For conducting the experiments we used CloudSimSDN [24] which is a simulation tool based on the widely used CloudSim toolkit [25]. CloudSimSDN is specifically designed as a network simulation tool which facilitates SDN features such as dynamic network configuration, programmable controller and so on. We have extended CloudSimSDN such that it can be used as a software defined cloud environment in which SDN controller can interact with resource scheduler for performing scheduling decisions and related actions. Fat Tree topology [22] was used for connecting servers in the simulated cloud data center.

B. Datasets

We conducted two sets of experiments using two different DAG workloads. In the first set of experiments we used a number of well known scientific workflow benchmarks with different resource requirements: Montage, CyberShake, Inspiral, Epigenomics and SIPHT. We used the synthetic workflow traces provided by Pegasus group which are generated using traces from actual executions of scientific workflows [25]. For evaluating the performance of algorithms we created a composite workload using the aforementioned scientific workflows. To



(a) Total energy consumption

(b) Percentage improvement over the baseline algorithm (RandomFF)

Fig. 4: Energy consumption of scientific workflow executions

model the arrival pattern of the workflows we used a Poisson distribution.

In addition to modeling scientific workflows, the DAG execution model can be used to model a wide variety of distributed applications and batch workloads. The second experiment was performed to evaluate the performance and suitability of J-OPT for scheduling non-scientific DAG workloads. For this experiment, we used a sample of 1000 workflows from workload traces extracted from a production cluster of Alibaba cloud. Alibaba cluster traces do not include details related to data dependencies among tasks in workflows. Hence, we have conducted the experiments under the assumption that data dependencies are prevalent among a parent task and a child task in a workflow with a probability of 0.5.

C. Experiment Results and Analysis

Figure 4a illustrates the normalized energy consumption incurred from the execution of the composite scientific workload. Compared to the greedy baseline RandomFF, all other algorithms have achieved significant savings in terms of total energy consumption. This is the expected behavior since EnREAL, HEROS-DAG, as well as J-OPT, incorporate multiple strategies specifically designed for enhancing the energy efficiency of scheduled workflows.

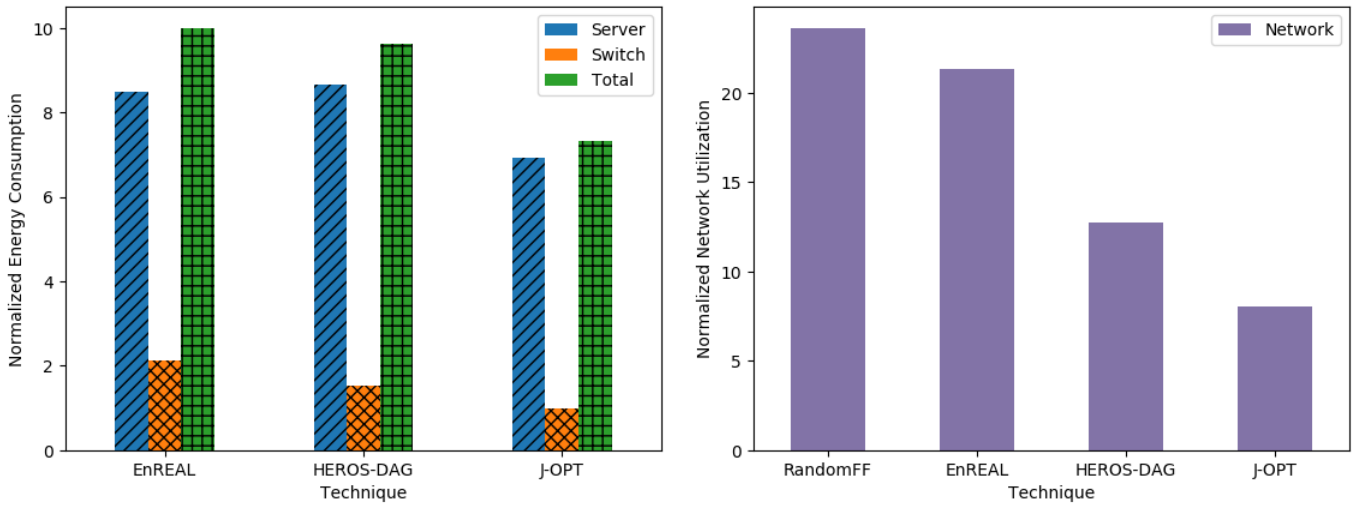
When analyzing the percentage improvements of the algorithms by disaggregating total energy consumption into its constituent components as depicted in Figure 4b, it can be observed that J-OPT far exceeds EnREAL and HEROS-DAG with respect to energy savings incurred from switch utilization. J-OPT has been able to achieve nearly 80% improvement over the switch energy consumption of greedy baseline, and 60% and 30% improvement over that of EnREAL and HEROS-DAG respectively while also reducing server energy consumption by 8%.

Although the importance of switch energy savings may appear less significant compared to total energy savings, it should be noted that as servers become fully energy proportional, the network energy consumption could rise as high as 50% of total data center energy consumption under light job loading conditions [3]. This is common in data centers since they are typically over-provisioned to meet peak load. Figure 5a and Figure 5b depict a comparison of the performance of algorithms under a heavy and light job loading scenario, respectively. Note that we have excluded the baseline algorithm's energy consumption from these comparisons since this situation occurs when servers are fully energy proportional, which is not the case with the baseline algorithm (RandomFF).

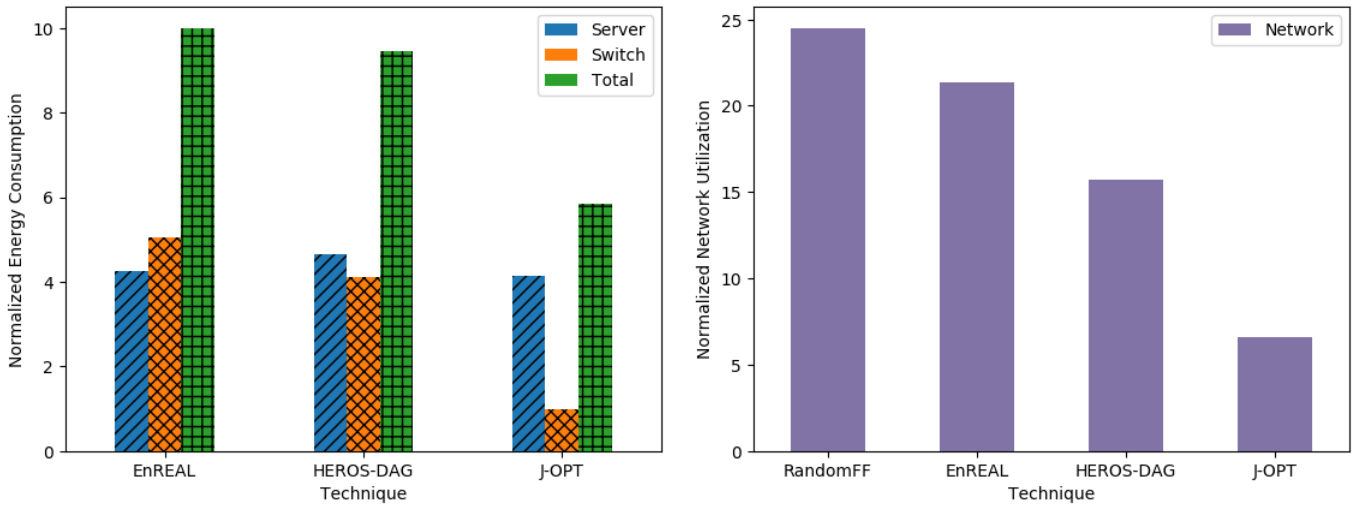
As illustrated in Figure 5b, in the light job loading scenario network switches have consumed as much power as the servers with EnREAL and HEROS-DAG. In contrast, the topology-aware resource allocation strategy of J-OPT has been able to reduce the power consumption of network switches by a significant proportion, leading to an overall reduction of 30%-40% in the total energy consumption compared to state-of-the-art algorithms. Switch level network utilization is also reduced by a factor of 2 and 3 compared to HEROS-DAG and EnREAL, respectively.

Figure 6 depicts the results obtained with the application of J-OPT and other comparison algorithms to a sample workload containing 1000 workflows from Alibaba cloud traces. Note that we have excluded EnREAL in this experiment since it is specifically designed for scientific workflows, and, the migration based resource allocation policy of EnREAL is less appropriate for scheduling short spanned tasks in Alibaba workflows. As illustrated in Figure 6a, J-OPT has outperformed the comparison algorithms in terms of total energy consumption. As previously discussed, the significance of power savings achievable with J-OPT would be even more prominent under light job loading conditions.

The superiority of J-OPT in terms of minimizing switch level



(a) Heavy job loading state



(b) Light job loading state

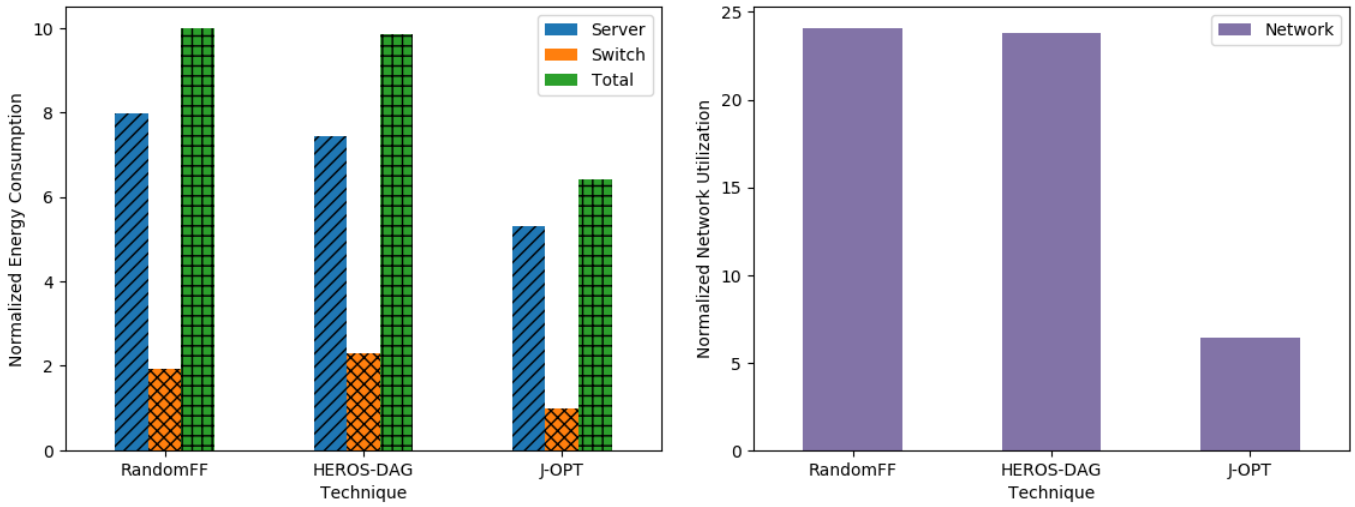
Fig. 5: Energy consumption and switch level network utilization during different data center occupancy states for scientific workflow executions

network utilization can be observed in Figure 6b. J-OPT has been able to reduce the switch level network utilization by over 70% compared to the other algorithms. This is because the topology-aware resource allocation mechanism of J-OPT attempts to place the tasks of a workflow in a group of closely located servers. This, in turn, minimizes the number of aggregate and core switches used during the execution of workflows.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented J-OPT, a novel topology-aware resource allocation technique for energy-efficient workflow scheduling in cloud data centers. J-OPT operates with the objective of minimizing total data center power consumption by jointly optimizing the utilization of servers and networking elements used in the execution of workflows. We have evaluated J-OPT in a simulated environment using synthetic and real-world workflow traces of scientific as well as commercial applications, and, the results clearly demonstrate the effectiveness of the proposed algorithm compared to state-of-the-art approaches.

The proposed technique is designed to operate with hierarchical data center topologies, and, as a future work we intend to adapt J-OPT to operate with other network topologies [26]. In this study we have considered a homogeneous configuration so that we can solely evaluate the gains that can be achieved by the joint optimization of servers and networking elements in workflow scheduling. As a future work, J-OPT could be extended and evaluated in a cloud data center with heterogeneous devices [23]. Adaptation of J-OPT to cater to the requirements of inter-cloud application brokering (provisioning and scheduling) scenarios [27] is another interesting future direction. Brokering



(a) Total energy consumption

(b) Switch level network utilization

Fig. 6: Performance of algorithms on a sample of 1000 workflows from Alibaba cluster traces

policies could be designed such that additional factors such as the availability of renewable energy [28] are also considered.

REFERENCES

[1] Y. C. Lee and A. Y. Zomaya, "Energy efficient utilization of resources in cloud computing systems," *The Journal of Supercomputing*, vol. 60, no. 2, pp. 268–280, 2012.

[2] A. Andrae and T. Edler, "On global electricity usage of communication technology: trends to 2030," *Challenges*, vol. 6, no. 1, pp. 117–157, 2015.

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

[4] Y. Jararweh, M. Al-Ayyoub, E. Benkhelifa, M. Vouk, A. Rindos *et al.*, "Software defined cloud: Survey, system and evaluation," *Future Generation Computer Systems*, vol. 58, pp. 56–74, 2016.

[5] D. Kliazovich, J. E. Pecero, A. Tchernykh, P. Bouvry, S. U. Khan, and A. Y. Zomaya, "Ca-dag: Modeling communication-aware applications for scheduling in cloud computing," *Journal of Grid Computing*, vol. 14, no. 1, pp. 23–39, 2016.

[6] B. P. Rimal and M. Maier, "Workflow scheduling in multi-tenant cloud computing environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 1, pp. 290–304, 2016.

[7] M. A. Rodriguez and R. Buyya, "A taxonomy and survey on scheduling algorithms for scientific workflows in iaas cloud computing environments," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 8, p. e4041, 2017.

[8] S. Yassa, R. Chelouah, H. Kadima, and B. Granado, "Multi-objective approach for energy-aware workflow scheduling in cloud computing environments," *The Scientific World Journal*, vol. 2013, 2013.

[9] M. Mezmaz, N. Melab, Y. Kessaci, Y. C. Lee, E.-G. Talbi, A. Y. Zomaya, and D. Tuyttens, "A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems," *Journal of Parallel and Distributed Computing*, vol. 71, no. 11, pp. 1497–1508, 2011.

[10] M. Żotkiewicz, M. Guzek, D. Kliazovich, and P. Bouvry, "Minimum dependencies energy-efficient scheduling in data centers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 12, pp. 3561–3574, 2016.

[11] M. Sharifi, S. Shahrivari, and H. Salimi, "Pasta: a power-aware solution to scheduling of precedence-constrained tasks on heterogeneous computing resources," *Computing*, vol. 95, no. 1, pp. 67–88, 2013.

[12] X. Xu, W. Dou, X. Zhang, and J. Chen, "Enreal: An energy-aware resource allocation method for scientific workflow executions in cloud environment," *IEEE Transactions on Cloud Computing*, vol. 4, no. 2, pp. 166–179, 2015.

[13] H. Chen, X. Zhu, D. Qiu, H. Guo, L. T. Yang, and P. Lu, "Eons: minimizing energy consumption for executing real-time workflows in virtualized cloud data centers," in *Proceedings of the 45th International Conference on Parallel Processing Workshops (ICPPW)*. IEEE, 2016, pp. 385–392.

[14] D. Kliazovich, P. Bouvry, and S. U. Khan, "Dens: data center energy-efficient network-aware scheduling," *Cluster Computing*, vol. 16, no. 1, pp. 65–75, 2013.

[15] B. Cao, X. Gao, G. Chen, and Y. Jin, "Nice: network-aware vm consolidation scheme for energy conservation in data centers," in *Proceedings of the 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2014, pp. 166–173.

[16] S. Vakilinia, "Energy efficient temporal load aware resource allocation in cloud computing datacenters," *Journal of Cloud Computing*, vol. 7, no. 1, p. 2, 2018.

[17] H. Jin, T. Cheocherngngarn, D. Levy, A. Smith, D. Pan, J. Liu, and N. Pissinou, "Joint host-network optimization for energy-efficient data center networking," in *Proceedings of the 27th IEEE International Symposium on Parallel and Distributed Processing*. IEEE, 2013, pp. 623–634.

[18] S. Pelley, D. Meisner, T. F. Wenisch, and J. W. VanGilder, "Understanding and abstracting total data center power," in *Workshop on Energy-Efficient Design*, vol. 11, 2009.

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

[20] H. Topcuoglu, S. Hariri, and M.-y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 260–274, 2002.

[21] O. Sinnen, *Task scheduling for parallel systems*. John Wiley & Sons, 2007, vol. 60.

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

[23] M. Guzek, D. Kliazovich, and P. Bouvry, "Heros: Energy-efficient load balancing for heterogeneous data centers," in *Proceedings of the 8th IEEE International Conference on Cloud Computing*. IEEE, 2015, pp. 742–749.

[24] J. Son, A. V. Dastjerdi, R. N. Calheiros, X. Ji, Y. Yoon, and R. Buyya, "Cloudsimsdn: Modeling and simulation of software-defined cloud data centers," in *Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 2015, pp. 475–484.

[25] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and experience*, vol. 41, no. 1, pp. 23–50, 2011.

[26] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.

[27] N. Grozev and R. Buyya, "Inter-cloud architectures and application brokering: taxonomy and survey," *Software: Practice and Experience*, vol. 44, no. 3, pp. 369–390, 2014.

[28] W. Deng, F. Liu, H. Jin, B. Li, and D. Li, "Harnessing renewable energy in cloud datacenters: opportunities and challenges," *IEEE Network*, vol. 28, no. 1, pp. 48–55, 2014.