2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing

Energy-aware Workflow Job Scheduling for Green Clouds

Fei Cao
Department of Computer Science
Southern Illinois University, Carbondale
Carbondale, IL, 62901, USA
Email: vicky@siu.edu

Michelle M. Zhu
Department of Computer Science
Southern Illinois University, Carbondale
Carbondale, IL, 62901, USA
Email: mzhu@cs.siu.edu

***Abstract*—With the increasing deployment of many data centers and computer servers around the globe, the energy cost on running the computing, communication and cooling together with the amount of CO_2 emissions have increased dramatically. In order to maintain sustainable Cloud computing with ever-increasing problem scale, we design and develop energy-aware scientific workflow scheduling algorithm to minimize energy consumption and CO_2 emission without sacrificing Quality of Service (QoS) such as response time specified in Service Level Agreement (SLA). The underlying available computing capacity and network bandwidth is represented as time-dependent because of the dual operation modes of on-demand and reservation instances supported by many commercial Cloud data centers. The Dynamic Voltage and Frequency Scaling (DVFS) is utilized to lower the CPU frequencies of virtual machines as long as the finishing time is still before the specified deadline. Our resource provision and allocation algorithm aims to meet the response time requirement and minimize the Virtual Machine (VM) overhead for reduced energy consumption. The consolidated VM reuse can lead to higher resource utilization rate for higher system throughput. The effectiveness of our algorithm is evaluated under various performance metrics and experimental scenarios using software adapted from open source CloudSim simulator. The simulation results show that our algorithm is able to achieve an average up to 30% of energy savings.**

I. Introduction

The flexible utility-oriented pay-as-you-go Cloud computing model has demonstrated tremendous potential for both commercial and scientific users to access and deploy their applications anytime from anywhere at reasonable prices depending on their QoS specifications. Gartner estimated that the market opportunity for Cloud computing will be worth around \$150 billion by 2014 [1]. The computing power of the Cloud environment is supplied by a collection of data centers that are typically installed with hundreds to thousands of servers which are built on virtualized compute and storage technologies. Meanwhile equally massive cooling systems are required to keep the servers within normal operating temperatures. Servers and cooling systems make up about 80% of all the electricity used within a data center [2]. However, these infrastructures consume tremendous amounts of energy. For example, a typical data center with 1000 racks consumes about 10 Megawatt of power during normal operation [3]. Over the past decade, the cost of servers running and cooling systems has increased by 400% [4], and following the current usage and efficiency trends, energy consumption by data centers could nearly double in another five years to more than 100 billion kWh [5]. Besides the energy cost, data centers also produce considerable amount of CO_2 emissions which significantly contribute to the growing environmental issue of Global Warming. Gartner estimated that the Information and Communication Technologies (ICT) industry generates about 2% of the total global CO_2 emissions in 2007 [6]. Therefore, Cloud providers should also ensure that their data centers are CO_2 emission regulation compliant to meet the future permissible restrictions [7], [8]. Reducing energy consumption for modern data centers has been recognized as an ever increasingly important issue for operation cost, environment footprint and system reliability. Furthermore, less energy consumption means less heat generated to maintain the system in a relatively cool temperature to reduce the hardware related failures for longer Mean Time Between Failures (MTBF). Hence, energy-efficient Cloud computing technologies are highly desirable for future sustainable ICT [9] for cost effectiveness, environmental friendliness as well as stable system operation.

Many scientific applications such as Nimbus [10] and Eucalyptus [11] as well as some Cloud systems such as DOE Magellan [12] are modeled as workflows, which can be as simple as a single task or as complex as a Directed Acyclic Graph (DAG). The dependency and parallelism embedded in a workflow requires that the tasks be dispatched to a group of distributed VMs in order to maximize the execution efficiency. Also the Cloud resource availability map is time-dependent as many Cloud providers support both on-demand and reservation VM allocations. Our Cloud model accommodates these issues to establish a realistic and useful testbed for energy-efficient scientific computing research.

The rest of the paper is organized as follows. Section II discusses the related works on energy/CO_2-efficient schedulers, VM allocations and DVFS technology. Section III presents our green Cloud system architecture, and defines the green Cloud energy model as well as the scientific workflow model. Section IV defines the problem formulation and the algorithm design details. Section V explains the evaluation methodology and simulation setup followed by the comprehensive experimental results. Conclusion and future work can be found in Section VI.

II. Related Work

The operation of large geographically distributed Cloud data centers requires considerable amount of energy that

978-0-7695-5046-6/13 \$26.00 © 2013 IEEE
DOI 10.1109/GreenCom-iThings-CPSCom.2013.58

accounts for a large portion of the total operational costs [13], [14]. There are many research works addressing energy-efficient computation for either cluster servers or virtualized servers. Technologies such as Dynamic Voltage and Frequency Scheduling (DVFS) and Dynamic Power Management (DPM) [15] were extensively studied and widely deployed for energy savings. For cluster servers, Kim et al. [16] proposed power-aware scheduling algorithms for bag-of-tasks applications with deadline constraints on DVFS-enabled cluster systems. Chen et al. [17] presented a formalism to the dynamic optimization problem of server provisioning and DVFS control for multiple applications including response-time Service Level Agreements (SLA) and costs of server shutdowns. Wang et al. [18] presented a threshold-based algorithm for efficient power management of a single heterogeneous soft real-time cluster, where thresholds are generated to divide the workload into several ranges. The power manager dynamically measures and predicts the cluster workload and make corresponding decisions. For virtualized servers, Laszewski et al. [19] proposed an efficient scheduling algorithm to allocate VMs in a DVFS-enabled cluster by dynamically scaling the supplied voltage to reduce power consumption. Cardosa et al. [20] proposed an approach for power-efficient VM allocation in virtualized heterogeneous computing environments. They took advantage of the min-max resource partitions and shared parameters of Virtual Machine Monitor (VMM), which represented the minimum, maximum and proportion of the CPU allocated to VMs. The approach is only suitable for enterprise environments as it does not support strict SLAs. Liu et al. [21] presented the GreenCloud architecture which aimed to reduce power consumption while guaranteeing the performance from users' perspective by supporting optimized VM migration and placement.

Most of previous works focused on reduced energy usage instead of the profit boost while reducing the carbon emissions. Recently, a few research work began to take the environment sustainability issue such as the CO_2 emission into consideration. Garg et al. [3] proposed near-optimal energy-efficient scheduling policies on how to determine the mapping order of application/data center pairs along with DVFS strategy to minimize the CO_2 emission or maximize Cloud provider's profit. However, complex workflow structure which were commonly used by the scientific community, and virtualization mechanism was not considered.

Our work addresses the problem of scheduling the scientific workflow applications as opposed to individual tasks under a time-dependent Cloud environment. Both objectives of QoS and energy efficiency are also considered.

III. Green Cloud System Architecture and Analytical Models

A. Green Cloud System Architecture

Inspired by the architecture model by Beloglazov [22], [23], our system architecture for energy-efficient Cloud computing is shown in Fig. 1. There are basically four main entities involved:

1. **Users**: Submit service requests (i.e., workflow applications, QoS requirements, etc.) from anywhere to the Cloud.

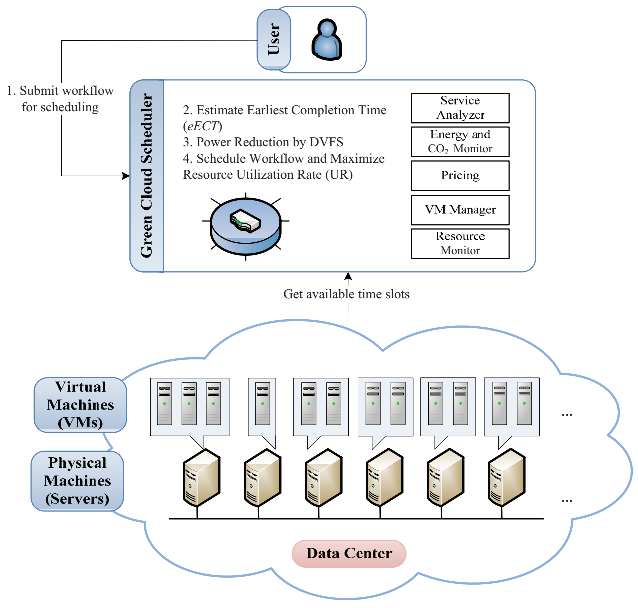
User
1. Submit workflow for scheduling
Green Cloud Scheduler
2. Estimate Earliest Completion Time (eECT)
3. Power Reduction by DVFS
4. Schedule Workflow and Maximize Resource Utilization Rate (UR)
Service Analyzer
Energy and CO2 Monitor
Pricing
VM Manager
Resource Monitor
Get available time slots
Virtual Machines (VMs)
Physical Machines (Servers)
...
...
Data Center

Fig. 1. Green Cloud system architecture.

2. **Green Cloud Scheduler**: Acts as an interface between the users and the Cloud infrastructure to enable energy/CO_2-efficient Cloud services, and performs the actual scheduling of workflows. The interaction of the following components is required:

a) Service Analyzer: Analyzes the service requirements (e.g., user specified deadline) of a submitted application to determine whether the data center can meet the requirements.

b) Energy and CO_2 Monitor: Monitors energy consumption for the servers and cooling system, as well as the CO_2 emission. Keep updating the Coefficient Of Performance factor (COP) as the efficiency of the cooling system, CO_2 emission rate, CPU power-frequency relationship, and electricity cost.

c) Pricing: Decides service charges and calculates energy cost and profit.

d) VM Manager: Updates VM status in order to provision new VMs and reuse existing VMs.

e) Resource Monitor: Monitors the actual usage of resources and accounts for the resource cost.

3. **Virtual Machines (VMs)**: Multiple independent VMs can be created and deployed on a single physical server.

4. **Physical Machines (servers)**: The data center consists of multiple clusters which comprise of multiple physical machines with limited OS.

B. Green Cloud Energy Model

According to [24], server loads and cooling system contribute about 80% of the total energy cost. Other energy usage such as lighting and office space conditioning are not considered due to their negligible contributions. We assume that a data center can run several pre-determined VM instances with different CPU and memory capacities with different prices. The related parameters of a data center are given in Tab. I.

TABLE I. PARAMETER OF A DATA CENTER

Parameters	Definitions
$[f_{i,min}, f_{i,max}]$	CPU frequency range of CPU i
COP	Efficiency of the cooling system
c	Electricity cost (\$/kWh)
r_{co_2}	CO_2 emission rate (kg/kWh)
ξ	Executing price (\$/hour)

1) Data Center Server Power Consumption: Power consumption by computing servers in data centers is mostly determined by the CPU, memory, disk storage and network interfaces. In comparison with other system resources, the CPU consumes the major portion of the energy. Consequently, we only consider the CPU power consumption for servers. Since a server can run copies of pre-configured VMs with different CPU frequencies, the frequency of a CPU also refers to the frequency of the associated VM.

The power consumption p_i of a CPU i has both the constant and variable parts. According to previous models [3], [18], [17], we approximate p_i by $p_i = \delta_i + \alpha_i f_i^3$, where δ_i denotes the constant power consumption, $\alpha_i f_i^3$ denotes the variable power consumption that is varied with CPU operating frequency f_i, and α_i stands for the proportionality constant. The CPU supports Dynamic Voltage and Frequency Scaling (DVFS) to operate under multiple voltage levels with appropriate frequencies in the range of $[f_{i,min}, f_{i,max}]$. Since current commercial CPUs support discrete frequency levels, only those supported frequencies in the range can be used.

2) Task Execution Time and CPU Frequency: DVFS is an efficient technology to reduce variable power dissipation by identifying computing regions where CPU frequency can be lowered with negligible performance loss. Since a lower CPU frequency results in more execution time of a workflow, the execution time of a task m_j on CPU i is calculated by Eq. 1:

$$t_i^j(f) = t_i^j(f_{i,max}) \times \frac{f_{i,max}}{f} \tag{1}$$

where $t_i^j(f)$ is the execution time of task m_j running at CPU frequency f on CPU i, and $t_i^j(f_{i,max})$ is the execution time of task m_j running at maximum CPU frequency $f_{i,max}$.

3) Green Cloud Energy Cost: Inspired by previous work [3], executing a workflow $\mathcal{W}$ at a data center results the following:

(1) Energy consumption of CPU i executing task m_j

$$EC_i^j = (\delta_i + \alpha_i (f_i^j)^3) \times t_i^j(f_{i,max}) \times \frac{f_{i,max}}{f_i^j} \tag{2}$$

where f_i^j is the frequency of CPU i executing task m_j, and $t_i^j(f_{i,max})$ is the execution time of task m_j running on CPU i at maximum frequency.

(2) Total energy consumption

$$E = (1 + \frac{1}{COP}) \sum_{j=1}^{n} EC_i^j \tag{3}$$

(3) Energy cost

$$C = E \times c \tag{4}$$

(4) CO_2 emission

$$CO2E = E \times r_{co_2} \tag{5}$$

(5) Profit of executing workflow $\mathcal{W}$

$$Prof = \xi \times \sum_{j=1}^{n} \left(t_i^j(f_i^j) \times \frac{f_i^j}{f_{unit}} \right) - C \tag{6}$$

where $t_i^j(f_i^j)$ is the execution time of task m_j running on CPU i at frequency f_i^j, and ξ is the unit executing price of the data center (i.e., the execution price of running at frequency f_{unit} = 1.0 GHz).

C. Workflow Scheduling Model

We construct the workflow scheduling model as the workflow task graph and the underlying Cloud environment (i.e., a data center) to facilitate the mathematical formulation of the scheduling problem.

1) Graph Notations: A workflow of a distributed computing application is constructed as a Directed Acyclic Graph (DAG) $G_m = (V_m, E_m)$ with $|V_m| = n$. Vertices are used to represent the set of computing tasks $V_m = \{m_1, m_2, ...m_n\}$: m_1 is the starting task and m_n denotes the ending task. The weight w_{ij} on edge e_{ij} represents the size of data transferred from task m_i to task m_j. The dependency between a pair of tasks is shown as a directed edge. Task m_j receives a data input w_{ij} from each of its preceding tasks m_i and performs a predefined computing routine whose complexity is modeled as a function $\zeta_j(\cdot)$ of the total aggregated input data size z_j. However, in real scenario, the complexity of a task is an abstract quantity which not only depends on the computational complexity of its own function but also on the implementation details realized in its algorithm. Upon completion of execution of task m_j, data output w_{jk} will be sent to each of its succeeding tasks m_k. A task cannot start its execution until all input data required by this task arrives. To generalize our model, if a workflow has multiple starting or ending tasks, a virtual starting or ending task of zero complexity can be created and connected to all starting or ending tasks without any data transfer along the edges.

The Cloud environment (i.e., a data center) where the VMs are reserved, deployed and run on servers is assumed to support both advance VM reservation and on-demand requests. Thus, the resource map of a data center is time-dependent, which means that the available resources of each server and the bandwidth of each link are changing from time to time. For general purposes, we model a data center as a complete network graph $G_s = (V_s, E_s)$ with $|V_s| = m$, consists of a set of servers $V_s = \{v_1, v_2, ...v_m\}$. The allocable computing power of v_j at time t is represented as $p_{j,t}$. The network link L_{ij} between server v_i to v_j is featured by bandwidth $b_{ij,t}$ at time t, and link delay d_{ij}.

The allocable VM power during time interval $[t_1, t_n]$ is defined as $\mathcal{P}_{j,t_1,t_n}$. Let $z_i \times \zeta_j(\cdot)$ denote the aggregated and complexity normalized input data size of task m_i, the execution time of task m_i on server v_j during time interval t_1 and t_n is $t^i_{j,t_1,t_n} = \frac{z_i \times \zeta_j(\cdot)}{\mathcal{P}_{j,t_1,t_n}}$. Similarly, the minimum time to transfer data of size s along link L_{ij} during time interval $[t_1, t_n]$ is $t_{ij,t_1,t_n} = \frac{s}{\mathcal{B}_{ij,t_1,t_n}} + d_{ij}$.

2) Workflow Resource Cost Computation: The resource cost of a workflow includes the task running time, and overhead for VM startup, idle and shutdown time. The total resource cost for server v_j considering the allocated computing capacity and the overhead cost is calculated by Eq. 7, where t_{start} and t_{shut} are the VM startup and shutdown time, respectively, $t^{M_{j,l}}_{l,t_s,t_e}$ is the execution time of all the tasks assigned to the server v_j's lth VM, and $Idle(j)$ is the idel time betwenn the execution of two adjacent tasks assigned to that VM.

$$\begin{aligned} RC_j &= \sum_{\forall VM \in v_j} \mathcal{P}_{l,t_s,t_e} \times (t_{start} + t^{M_{j,l}}_{l,t_s,t_e} + Idle(j) + t_{shut}) \\ overhead_j &= \sum_{\forall VM \in v_j} \mathcal{P}_{l,t_s,t_e} \times (t_{start} + Idle(j) + t_{shut}) \end{aligned} \tag{7}$$

3) Resource Utilization Rate: Utilization Rate (UR) indicates the effective resource utilization of a workflow and is defined as the ratio of the useful running cost to the total cost including overhead of the VM's startup, idle and shutdown time (i.e., $UR = 1 - \frac{\sum_{j=1}^{m} overhead_j}{\sum_{j=1}^{m} RC_j}$). It is the Cloud provider's desire to maximize this ratio in order to improve the system throughput and reduce the energy cost.

IV. Problem Formulation and Algorithm Design

A. Problem Formulation

The scheduling problem is defined as follows:

Definition 1: For a particular workflow $\mathcal{W}$ submitted by a user with specified Quality of Service (QoS) requirement (i.e., deadline), our objective is to schedule the workflow to the data center such that the energy cost and CO_2 emission can be minimized, and the resource utilization rate can be maximized within the deadline constraint for higher profit and less environmental footprint.

B. Algorithm Design

Algorithm 1 ERAS-$\mathscr{D}(\mathcal{W}, deadline)$

Input:
$\mathcal{W}$: a workflow
$deadline$: user specified deadline
Output:
An energy-efficient workflow schedule with maximized UR within user's deadline constraint

```
1: Calculate $eECT$;
2: if $eECT > deadline$ then
3:     RETURN NoPossibleMapping;
4: end if
5: PowerReductionDVFS($eECT, deadline$);
6: if BackwardScheduling($\mathcal{W}, deadline$) then
7:     RETURN MappingSuccessful;
8: else
9:     RETURN MappingFailed;
10: end if
```

A three-step workflow scheduling algorithm, namely Energy-efficient Resource Allocation for workflow Scheduling under Deadline constraint (ERAS-$\mathscr{D}$) is proposed. The first two steps estimate the workflow completion time and optimal CPU frequency for each task in the workflow. The third step performs the actual task mapping and VM allocation. The pseudocode of ERAS-$\mathscr{D}$ is provided in Alg. 1.

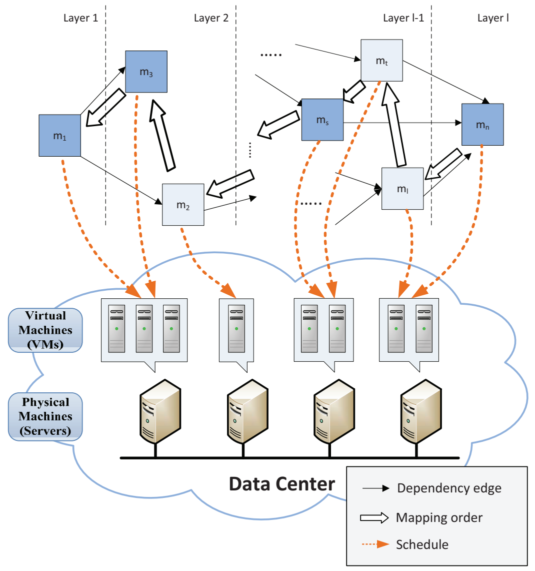

Fig. 2. Backward layer-based time-dependent VM allocation procedure.

Step 1 Earliest Completion Time Estimation Phase: To estimate an earliest completion time ($eECT$) of a workflow, we simply assume that all CPUs in a data center run at their maximum frequencies, and each task starts executing on a new VM as soon as all the input arrive. We determine the minimum execution time of the critical path (CP) as the $eECT$. If user specified deadline is smaller than $eECT$, it means that the data center is unable to finish executing before the deadline, and the user needs to specify a larger deadline.

Step 2 DVFS Phase: The objective of this step is to obtain an optimal frequency for each task to minimize the energy consumption subjects to the execution deadline. The energy consumption and CO_2 emission can be reduced by using DVFS. As a lower CPU frequency results in more execution time, the specified deadline should not be compromised. $AET^i_j = t^i_j \times \frac{deadline}{eECT}$ computes the acceptable execution time of task m_i scheduled on server v_j. Combined with Eq. 1, we can get the minimum acceptable frequency f^i_j. Moreover, the estimated start time eST_i for each task m_i can be determined.

From the graph of energy consumption at different frequencies of previous work [3], we can observe the existence of minimum energy consumption. The optimal frequency $f_{j,opt} = \sqrt[3]{\frac{\delta_j}{2\alpha_j}}$ can be derived. As the optimal CPU frequency depends on static variables, the local minima can be pre-computed. The resulting $f_{j,opt}$ is not bounded to $[max(f_{j,min}, f^i_j),\ f_{j,max}]$ which is the operating frequency range for a CPU j to fulfill the deadline requirement, and current commercial CPUs only support discrete frequency levels, we choose a frequency f in the range of $[max(f_{j,min}, f^i_j),\ f_{j,max}]$ which is supported by the CPU, and is nearest to $f_{j,opt}$ for executing task m_i on CPU j.

Step 3 Backward Workflow Scheduling Phase:

For the last step of $ERAS - \mathscr{D}$, Alg. 2 is developed to perform VM allocation to minimize the VM overhead and maximize UR within user's deadline constraint. In this step, a layer-based backward scheduling strategy is used to perform

$$LFT_i = \begin{cases} \min\limits_{\forall m_k \in suc(m_i)} \left(LFT_k - t^{k}_{k',t_s,t_e} - t^{ik}_{i'k',t_{ls},t_{le}} \right) & i \neq n \\ deadline & i = n \end{cases} \tag{8}$$

the task mapping from the last layer to the first layer. By adopting the backward scheduling strategy, each task has a larger scheduling interval, which increases the chance of VM reuse. From our intensive experiments, it is demonstrated that a backward scheduling strategy always achieves better UR than a forward scheduling strategy.

Fig. 2 shows the scheduling procedure of VM allocation which aims to schedule tasks on specific VMs running at their desired frequencies. A layer-based sorting of a DAG into different layers based on task dependencies is applied. Tasks in the same layer can be executed simultaneously and there will be no more than one task from the CP in the same layer. Each task will be given a priority value depending on their loads. Tasks on the CP (shown in dark shade in Fig. 2) will be given the highest priority compared with other tasks from the same layer. Each server maintains a time window matrix by using each start and end time of time slots to look up for available time slots to reuse existing VMs or allocate new VMs.

A brief description of Alg. 2 is presented as follows.

1. Perform layer-based sorting and assign the priority value to each task.

2. Calculate the latest finish time LFT_i from the ending task to the starting task using Eq. 8 where $suc(m_i)$ is the set of succeeding tasks of m_i, and dependency edge e_{ik} is mapped to network link $L_{i'k'}$. LFT_i of task m_i is to ensure that m_i's subsequent task's EST will not be delayed. LFT_n of the ending task m_n is simply set as the specified deadline. The possible start time ST_i of task m_i is in between eST_i and $LFT_i - AET_i$. The latter time $LFT_i - AET_i$ should always be greater than the former time eST_i, otherwise the mapping fails.

3. Map tasks in a backward direction starting from the ending task to possible servers. There would be two cases:

Case 1: If there are existing VMs allocated on that server, **ReuseVM()** is called to check whether we can reuse them.

Case 2: If no VM exists on that server or no VM can be reused, **AllocateNewVM()** is called to allocate a new VM by looking up the time window matrix.

4. Select the server with the maximum resource utilization rate for this task.

1) Virtual Machine Reuse Method: Two conditions must be satisfied if we reuse a VM if possible: 1) The available VM resource should be sufficient to run the task. 2) Any possible idle time should be less than the time to start up a shut down a VM and start up a new one.

2) Allocate New Virtual Machine: With given CPU frequency (calculated by Step 2), possible start time ST_i, and the latest finish time LFT_i of task m_i, a server will be found to allocate a new VM with provided frequency and enough execution time during the time interval $[eST_i, LFT_i]$.

Algorithm 2 BackwardScheduling($\mathcal{W}, deadline$)

Input:
$\mathcal{W}$: a workflow
$deadline$: user specified deadline
Output:
VM Allocation strategy with maximized UR within user's deadline constraint

```
1: set $maxUR$ = MAXVALUE;
2: for all $m_i$ in $\mathcal{W}$ do
3:     Calculate latest finish time $LFT_i$;
4:     Calculate possible start time $ST_i$;
5: end for
6: Conduct layer-based sorting and assign priority values;
7: Sort tasks in each layer in a decreasing order based on their priority value;
8: $MaxLayer$ = total number of layers in $G_t$;
9: for $l$ = layer $MaxLayer$ to 1 do
10:     SortedArray = sorted tasks in current layer;
11:     for all $m_i \in$ SortedArray do
12:         Update $LFT_i$;
13:         if $LFT_i$ - $eST_i$ < $AET_i$ then
14:             RETURN MappingFailed;
15:         end if
16:         for all $v_j \in G_s$ do
17:             if $v_j$ has allocated VMs in between time window [$eST_i$,$LFT_i$] then
18:                 if ReuseVM() then
19:                     Calculate $UR$;
20:                 else
21:                     AllocateNewVM();
22:                     Calculate $UR$;
23:                 end if
24:             else
25:                 AllocateNewVM();
26:                 Calculate $UR$;
27:             end if
28:             if $UR > maxUR$ then
29:                 $maxUR$ = $UR$;
30:             end if
31:         end for
32:         Assign $m_i$ to server with $maxUR$;
33:     end for
34: end for
35: RETURN MappingSuccessful;
```

V. Experimental Results

A. Experimental Setup

We use open source Java-based CloudSim toolkit [25] to model our green Cloud infrastructure and evaluate our scheduling algorithm. Many Java classes have been adapted to accommodate our workflow application structure as well as the time-dependent Cloud resources.

1) Data Center Configuration: 7 different data centers with different configurations are modeled as listed in Tab. II. CPU power factors are derived from Wang and Lu's work [18]. We consider discrete CPU frequencies in the range of $[max(0.8, f_{i,min}), f_{i,max}]$ with a step of 0.2 GHz (e.g., for data center 1, we consider CPU frequencies of 0.8, 1.0, 1.2,

TABLE II. DATA CENTER CONFIGURATIONS

Data Center ID	Location	Electricity Cost ($/kWh)	CO_2 Emission Rate (kg/kWh)	CPU Power Factors		CPU Frequency Level (GHz)		
				δ	α	$f_{i,min}$	$f_{i,max}$	$f_{i,opt}$
1	New York, USA	0.18	0.466	65	7.5	0.675	1.8	1.63
2	California, USA	0.16	0.350	75	5	0.75	2.0	1.957
3	Colorado, USA	0.12	0.909	75	5.2	0.9	2.4	1.932
4	Texas, USA	0.11	0.730	90	4.5	1.125	3.0	2.154
5	Beijing, China	0.08	0.839	105	6.5	1.125	3.0	2.006
6	Australia	0.22	0.924	90	4	1.2	3.2	2.241
7	Germany	0.28	0.539	105	4.4	1.2	3.2	2.285

[a] Energy cost reflects average commercial rates till June 2012 based on a US Energy Information Administration (EIA) report [26] and global electricity price from [27].
[b] CO2 emission rates are derived from a US Department of Energy (DOE) document [28] (Appendix F-Electricity Emission Factors 2007).

1.4, 1.6, and 1.8 GHz). We assume that the COP value is 1.5. The unit executing price is $0.4/hour as derived from [3]. We set the VM startup time as 100 seconds, and the shutdown time as 8 seconds based on the real-world study in [29].

Ten different sizes of workflow jobs (Workflow ID = 1 to 10) represented by a two-tuple $(n, |E_m|)$ are created for experimental purposes. The n defines the number of tasks and $|E_m|$ represents the number of dependency edges. The number of tasks ranges from 10 to 500 and number of dependency edges ranges from 20 to 1000. We develop a workflow generator class to randomly generate our test workflows with varying parameters within a suitably predefined range of values: (i) the complexity of each task; (ii) the number of inter-task communications and the data transfer size between two tasks.

2) Performance Metrics: We consider the following performance metrics:

- Workflow completion time/makespan
- Energy consumption
- Energy cost
- CO_2 emission
- Provider's profit
- Resource utilization rate

3) Experimental Scenarios: We evaluate our algorithm from the following experimental scenarios:

- Effect of DVFS and scheduling policy
- Effect of resource utilization rate maximization

B. Analysis of Results

To compare with our algorithm, a forward workflow scheduling (FWS) algorithm is developed. FWS is similar to Step 3 except that it performs layer-based sorting in a forward direction and uses tasks' earliest starting time (EST) to find a schedule that minimizes the workflow completion time.

1) Effect of DVFS and Scheduling Policy: We use the metrics of workflow completion time, energy consumption, energy cost, CO_2 emission, and provider's profit to examine the effect of the two phases. For each metric, we compare the following phases of our algorithm with FWS: earliest completion time estimation ($eECT$), DVFS ($DVFS$), backward workflow scheduling (BWS). Since we have 10 workflows and 7 data centers, we compare each metric versus different data centers or different workflows. As it is not realistic to draw all the results of metrics v.s. all the workflows and data centers here (e.g., completion time v.s. different data centers under all 10 workflows, and completion time v.s. different workflows under all 7 data centers), we therefore give part of the results but with complete coverage of situations (e.g., completion time v.s. different data centers under workflow 8, and completion time v.s. different workflows under data center 4). Since the user specified deadline directly effects the result, it is used as a baseline for comparison in workflow completion time. The result is plotted in Fig. 3.

We can observe that the workflow completion time increases from $eECT$ to $DVFS$ since the frequency is degraded. From $DVFS$ to BWS, the completion time increases due to VM allocation where each task may not start executing right after all the input data are available, whereas $DVFS$ does not consider actual resource availability and start executing as soon as all the input data are available. As we use a backward strategy starting from the LFT of the last task (i.e., the deadline) to find VMs to reuse in order to maximize the resource utilization rate, the completion time is close to the deadline. For energy consumption v.s. different data centers, it decreases by about 0% to 33% from $eECT$ to $DVFS$ due to DVFS technology, 2% to 23% from $DVFS$ to BWS due to VM allocation policy with backward VM reuse strategy. Note that the 0% decrease for data center 1 and 2 is because the optimal CPU frequencies to execute the workflow at data center 1 and 2 are the same as their maximum frequencies. For energy consumption v.s. different workflows, it decreases by about 1% to 11% from $eECT$ to $DVFS$, and 3% to 20% from $DVFS$ to BWS. Similarly, the energy cost and CO_2 emission also drop by each step. As the user charge depends on the actual execution time (the accumulated running time of each task in the workflow) which is a fixed value after the executing frequency for each task is determined, provider's profit increases due to the savings in energy cost. In comparison with FWS, BWS completes later than FWS since FWS aims to minimize the workflow completion time while BWS aims to maximize the resource utilization with a backward strategy that makes the completion time close to the deadline, but BWS outperforms FWS in all other aspects.

2) Effect of Resource Utilization Rate Maximization: To examine the effect of resource utilization rate maximization by BWS, we compare it with FWS. In Fig. 4, we compare the resource utilization rate versus different data centers or workflows. For different data centers (the left side of Fig. 4) The resource utilization rate increases about 13% to 20% for workflow 8. For different workflows (the right side of Fig. 4), the resource utilization rate increases about 15% to 27% for data center 4.

VI. CONCLUSIONS

In this paper, we propose a three-step heuristic workflow scheduling algorithm, namely ERAS-$\mathscr{D}$ to address the various objectives including guaranteed QoS, reduced energy and CO_2 emission for energy-efficient and environmental-friendly data centers. The first step of ERAS-$\mathscr{D}$ estimates

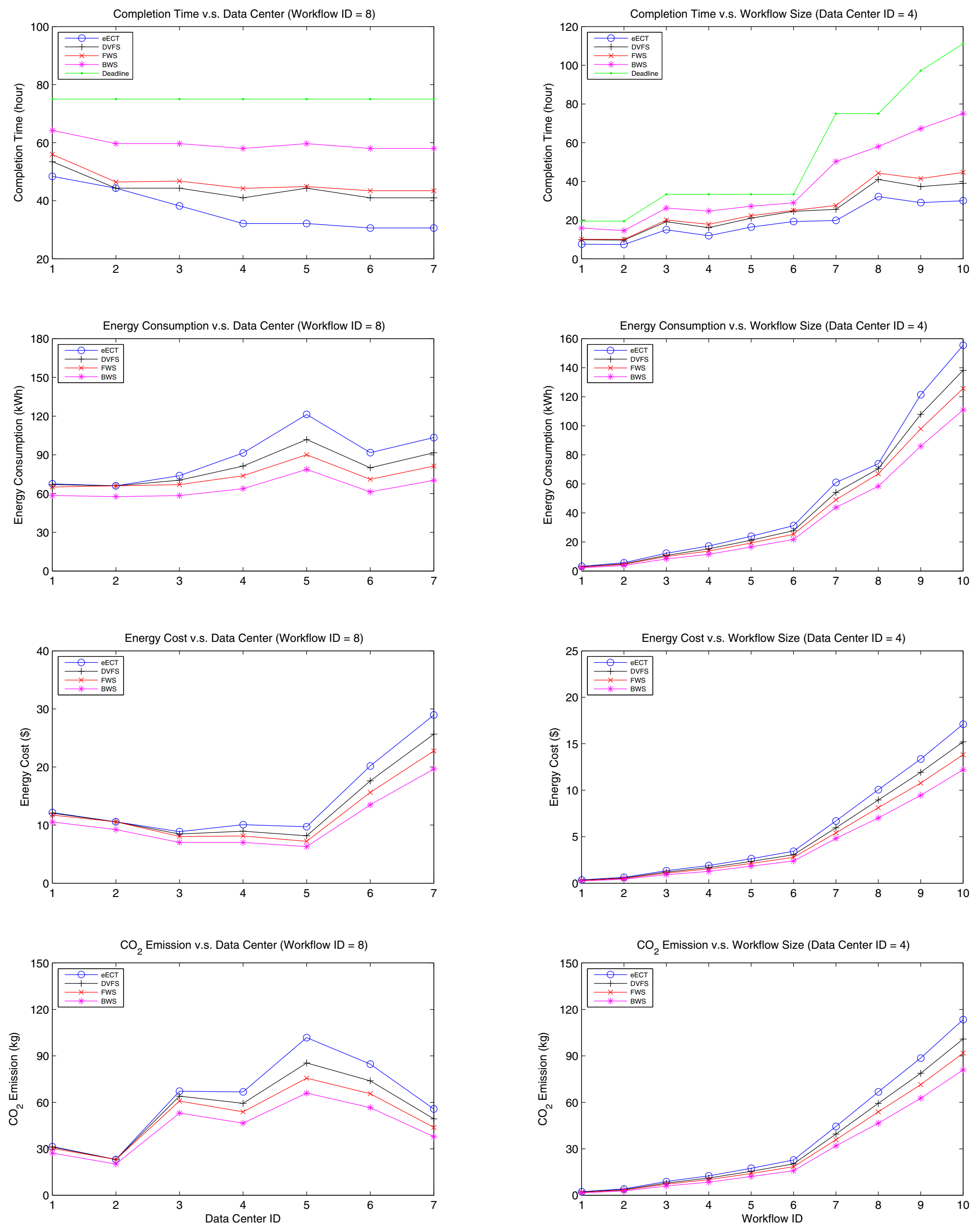

Fig. 3. Effect of DVFS and scheduling policy.

an earliest completion time for executing the workflow. The second step uses DVFS technology to reduce the unnecessary energy consumption without affecting the deadline. In Step 3, backward task scheduling is conducted to schedule each task to strategically selected VM in order to improve the resource utilization by maximizing the VM reuse and minimizing the VM idle time. Thorough experiments are performed to demonstrate the efficiency of ERAS-$\mathscr{D}$. Energy consumptions, energy cost and CO_2 emissions are decreased whereas provider's profit is increased in each step, and resource utilization rate is improved in comparison with a forward workflow scheduling strategy. Our future plan is to run our experiments on a local private Cloud, called Saluki Cloud established and managed by Eucalyptus with a few Beowulf clusters. We also would like to extend our model to consider possible VM migration, hibernation of some selected idle servers without affecting the

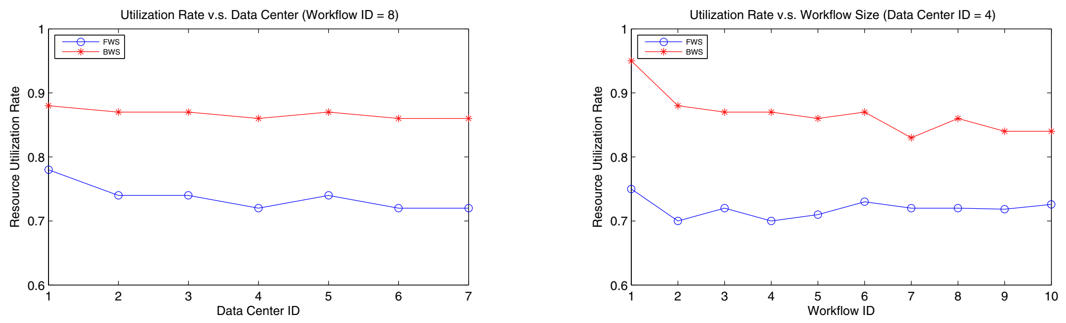

Fig. 4. Effect of resource utilization rate maximization.

response delay to future jobs.

References

[1] "Gartner Newsroom, Gartner says worldwide cloud services market to surpass $68 billion in 2010. http://www.gartner.com/it/page.jsp?id=1389313."

[2] "Data center energy characterization study site report. http://hightech.lbl.gov/documents/data_centers/dc_benchmarking/data_center_facility1.pdf."

[3] S. Garg, C. Yeo, A. Anandasivam, and R. Buyya, "Energy-efficient scheduling of HPC applications in cloud computing environments," CoRR abs/0909.1146, 2009.

[4] D. Filani, S. G. J. He, M. Rajappa, A. Kumar, P. Shah, and R. Nagappan, "Dynamic data center power management: Trends, issues, and solutions," *Intel Technology Journal*, vol. 12, no. 1, pp. 59–68, 2008.

[5] U.S. Environmental Protection Agency ENERGY STAR Program, "Report to congress on server and data center energy efficiency public law 109-431. http://hightech.lbl.gov/documents/data_centers/epa-datacenters.pdf," 2007.

[6] "Gartner Newsroom, Gartner estimates ICT industry accounts for 2 percent of global CO2 emissions. http://www.gartner.com/it/page.jsp?id=503867," April 2007.

[7] K. G. Brill, "What CIOs should know about carbon taxes. http://www. forbes.com/2009/07/14/carbon-tax-datacenter-technology-cio-network-carbon.html?feed=rss technology cionetwork," 2009.

[8] H. S. Dunn, "The carbon footprint of ICTs," ICTs and Environmental Sustainability, Tech. Rep., 2010.

[9] A. Berl, E. Gelenbe, M. Girolamo, G. Giuliani, H. Meer, M. Dang, and K. Pentikousis, "Energy-efficient cloud computing," *The Computer Journal*, vol. 53, no. 7, pp. 1045–1051, 2010.

[10] "Nimbus. http://nimbusproject.org."

[11] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. So-man, L. Youseff, and D. Zagorodnov, "The Eucalyptus open-source cloud-computing system," in *9th IEEE International Symposium on Cluster Computing and the Grid (CCGrid '09)*, 2009, pp. 124–131.

[12] "Magellan: Cloud computing for science. http://www.alcf.anl.gov/magel lan."

[13] X. Fan, W.-D. Weber, and L. Barroso, "Power provisioning for a warehouse-sized computer," in *the 34th Annual International Symposium on Computer Architecture (ISCA '07)*, San Diego, CA, 2007, pp. 13–23.

[14] R. Raghavendra, R. Parthasarathy, T. Vanish, Z. Wang, and X. Zhu, "No "power" struggles: Coordinated multi-level power management for the data center," in *the 13th International Conference on Architectural Support for Programming Languages and Operating Systems*, New York, NY, 2008, pp. 48–59.

[15] T. Horvath, T. Abdelzaher, K. Skadron, and X. Liu, "Dynamic voltage scaling in multitier web servers with end-to-end delay control," *IEEE Transactions on Computers*, vol. 56, no. 4, pp. 444–458, 2007.

[16] K. Kim, R. Buyya, and J. Kim, "Power aware scheduling of bag-of-tasks applications with deadline constraints on DVS-enabled clusters," in *the Seventh IEEE International Symposium on Cluster Computing and the Grid*, Rio de Janeiro, Brazil, 2007, pp. 541–548.

[17] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing server energy and operational costs in hosting centers," *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, pp. 303–314, 2005.

[18] L. Wang and Y. Lu, "Efficient power management of heterogeneous soft real-time clusters," in *the 2008 Real-Time Systems Symposium*, Barcelona, Spain, 2008, pp. 323–332.

[19] G. Laszewski, L. Wang, A. Younge, and X. He, "Power-aware scheduling of virtual machines in DVFS-enabled clusters," in *IEEE International Conference on Cluster Computing and Workshops (CLUSTER '09)*, 2009, pp. 1–10.

[20] M. Cardosa, M. R. Korupolu, and A. Singh, "Shares and utilities based power consolidation in virtualized server environments," in *the 11th IFIP/IEEE International Symposium on Integrated Network Management (IM '09)*. IEEE Communications Society, 2009, pp. 327–334.

[21] L. Liu, H. Wang, X. Liu, X. Jin, W. He, Q. Wang, and Y. Chen, "GreenCloud: A new architecture for green data center," in *6th International Conference Industry Session on Autonomic Computing and Communications Industry Session*, Barcelona, Spain, 2009, pp. 29–38.

[22] R. Buyya, A. Beloglazov, and J. Abawajy, "Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges," in *the 2010 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2010)*, Las Vegas, USA, July 2010, arXiv preprint arXiv:1006.0308.

[23] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Journal of Future Generation Computer Systems*, vol. 28, pp. 755–768, 2012.

[24] Pacific Northwest National Laboratory, "Data center energy efficiency. http://www.pnl.gov/computing/resources/esdc/sc07_bof/sc07bof_tschudi.pdf," 2007.

[25] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. 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] "US energy information administration (EIA) report. http://www.eia.gov/ electricity/monthly/," U.S. Department of Energy, 2012.

[27] "Global electricity price comparison. http://en.wikipedia.org/wiki/electri city_pricing."

[28] U.S. Department of Energy, "Voluntary reporting of greenhouse gases: Appendix F. electricity emission factors. http://www.eia.gov/oiaf/1605/pdf/Appendix%20fr071023.pdf." U.S. Department of Energy, 2012.

[29] M. Mao and M. Humphrey, "A performance study on the VM startup time in the cloud," in *5th International Conference on Cloud Computing (Cloud 2012)*, Honolulu, Hawaii, USA, June 2012, pp. 423–430.