# Running Industrial Workflow Applications in a Software-defined Multi-Cloud Environment using Green Energy Aware Scheduling Algorithm

Zhenyu Wen*, *Member, IEEE,* Saurabh Garg†, Gagangeet Singh Aujla‡, *Senior Member, IEEE,* Khaled Alwasel§, Deepak Puthal*, *Senior Member, IEEE,* Schahram Dustdar¶, *Fellow, IEEE,* Albert Y. Zomaya‖, *Fellow, IEEE,* Rajiv Ranjan**, *Senior Member, IEEE*.

***Abstract*—Industry 4.0 have automated the entire manufacturing sector (including technologies and processes) by adopting Internet of Things and Cloud computing. To handle the workflows from Industrial Cyber-Physical systems, more and more data centers have been built across the globe to serve the growing needs of computing and storage. This has led to an enormous increase in energy usage by cloud data centers which is not only a financial burden but also increases their carbon footprint. The private Software Defined Wide Area network (SDWAN) connects a cloud provider's data centers across the planet. This gives the opportunity to develop new scheduling strategies to manage cloud providers workload in a more energy-efficient manner. In this context, this paper addresses the problem of scheduling data-driven industrial workflow applications over a set of private SDWAN connected data centers in an energy-efficient manner while managing trade-off of a cloud provider' revenue. Our proposed algorithm aims to minimize the cloud provider's revenue and the usage of non-renewable energy by utilizing the real-world electricity prices with the availability of green energy on different cloud data centers, where the energy consumption consists of the usage of running application over multiple data centers and transferring the data among them through SDWAN. The evaluation shows that our proposed method can increase usage of green energy for the execution of industrial workflow up to $3\times$ times with a slight increase in the cost when compared to cost-based workflow scheduling methods.**

***Index Terms*—Software Defined Networking; Green Energy; Industrial Workflow Applications; Big Data; Industrial Clouds.**

## I. Introduction

The Industry 4.0 revolution helps to gather data in real-time and then analyze it at remote clouds to improve the industrial processes and detect faulty operations. This helps to realize the future operating problems in advance and make the industrial processes more stringent. This transition triggers high quality productivity that further improves the industrial productivity, economics, and workflows. However, this makes more data-driven undertakings which increases the data sharing across multiples sites and even across industrial (factory) boundaries. As a result, the dependence on Cloud computing will increase though the deployment of machine data and functionality over remote clouds. Eventually, this leads to an increase in the industrial workflow applications running at the cloud data centers. However, more and more industrial applications are harnessing cloud resources. To satisfy the needs of their customers and industrial workflow applications, cloud computing providers in general maintain very large infrastructure.

It is quite evident that the amount of energy consumed by the world's data centers – the repositories of billions of gigabytes of information – will exponentially increase over the next decade, putting an enormous strain on energy sources. In 2010, electricity usage in global data centers accounted for about 1.3 per cent of total electricity usage worldwide [1]. Data centers now consume about three per cent of the global electricity supply. Clearly, to operate such large infrastructure, a very large amount of electricity is required depending on the size of the data centers. This also contributes significantly to their high operational cost. According to a report published by the European Union, a decrease in emission volume of 15-30 per cent is required before 2020 to keep the global temperature increase below 2°C. Thus, high energy consumption and the carbon footprint of cloud data center infrastructures have become key environmental concerns and have immense potential to deal a hefty blow to efforts to contain global warming. The majority of cloud providers (such as Amazon, Apple, and Google) offer a multi-cloud environment (including industrial clouds) which is Geo-distributed across different countries and are connected via software-defined network (SDN) [2]. Some of them are designed to utilize the green energy, provided from local providers[1]. So, this provides an opportunity for the cloud providers to schedule their workloads to the data centers which utilize more renewable energies.

### A. Research Question

**How to execute an industrial workflow application across multiple data centers via private SDWAN?** To exe-

Z. Wen, and D. Puthal are with the School of Computing, Newcastle University, United Kingdom. E-mail:{Zhenyu.Wen, Deepak.Puthal}@newcastle.ac.uk

S. Garg is with the University of Tasmania, Australia. E-mail: Saurabh.Garg@utas.edu.au

G. S. Aujla is with the Department of Computer Science, Durham University, Durham, UK. E-mail: gagangeet.s.aujla@durham.ac.uk

K. Alwasel is with the School of Computing Science, Newcastle University, United Kingdom and also with the College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia. E-mail:k.alwasel@seu.edu.sa

S. Dustdar is with the TU Wien, Austria. E-mail: dustdar@dsg.tuwien.ac.at

A. Y. Zomaya is with the School of Computer Science, the University of Sydney, Australia. E-mail: albert.zomaya@sydney.edu.au

R. Ranjan are with the School of Computing Science, Newcastle University, United Kingdom and also with the Chinese University of Geosciences, Wuhan, China E-mail:Raj.Ranjan@newcastle.ac.uk

[1]www.google.com/green

cute an industrial workflow task on cloud, we need sufficient computing resources provided by cloud provider, the codebase of the task belonged to user and data provided by user or generated from upstream tasks. To this end, the workflow scheduler has to handle the three factors at the same time i.e., resource provisioning, task provisioning and data provisioning. Some works [3], [4] have developed the schedulers to run scientific workflow over multiple data centers. However, they do not consider the software-defined data centers that offers more flexible for design new green energy scheduling algorithm. WARM [5] aim of scheduling the tasks in SDN-based cloud data center while maximizing the revenue of the cloud provider. It optimizes the latency of tasks both in network and VM. However, this solution is not suitable for multiple clouds scenarios.

**How to optimize for energy efficiency while avoiding SLA violations?** While Cloud computing aims to optimize the use of hosted ICT resources, the Cloud providers do not *(yet)* have an effective solution for simultaneously optimizing energy consumption and SLAs (e.g. deadline, processing cost) especially for Big Data-driven Scientific and industrial workflow application scheduling in software-defined multi-cloud environments. One of the major reasons for this state of affairs is that cloud providers operate multiple large data centers distributed across multiple locations. Depending on the location of the data center and location of the application owner, the scheduling process for cloud applications has to automate cloud data center selection and, in doing so, ensure that SLA (e.g. application hosting cost, application run-time performance) and energy (e.g. total electricity bills, sustainability goals) requirements are met at the same time, which are often conflicting. When selecting ICT resources (e.g. virtual machines, containers, storage space) from multiple data centers, cloud providers must consider heterogeneous set of criteria and complex dependencies across multiple layers (e.g. application level, data center level), which is impossible to resolve manually.

There is a substantial amount of related work addressing the improvement of the carbon footprint of data centers by managing customer workloads at different levels such as storage, computation, and network [6], [7], [8]. However, most of these solutions are not directly applicable in the context of scientific or industrial workflow applications which is the focus of this paper. For each application workload and execution profile, a different strategy is needed to minimize their energy usage while optimizing SLAs. Scientific and industrial workflows are one of the most complex applications where several tasks have to be executed in a synchronous manner to achieve the required Quality of Service [9]. Communication between different tasks makes the matter worse as the energy usage of the network also needs to be considered with other constraints. Several authors have proposed usage of multiple data center locations to improve energy cost and also minimize environmental impact [10]. However, these solutions are designed for simple applications that consist of tasks which can run independently. Also, the existing solutions do not take the advantages of SDN network to account to further improving the data provision strategies.

In order to minimize energy usage while avoiding SLA violations of workflow applications, we need new system and algorithmic solutions that can consider several factors including dependency between different tasks with data transfer cost in private SDWAN, in addition to energy cost and carbon footprint associated with application execution. To this end, we propose an adaptive genetic algorithm-based mechanism to schedule workflow applications considering application users' requirements such as deadline and budget. To minimize the carbon footprint [11], the proposed algorithm selects the schedule that favors the data centers where green energy being utilized. However, as green energy availability varies with time [12], thus our algorithm also considers resources from different data centers. Moreover, from the user perspective execution cost and minimum execution time is also important; our algorithm also considers this trade-off between execution cost, usage of green energy and execution time. In particular, our proposed algorithm minimizes execution cost while selecting solutions with minimum carbon footprint for overall schedule by using multiple data centers with more green energy usage (Table III highlights the novelty of our work). The contributions of this papers are:

- A new SDN-based Workflow Broker (SDNWB) to deploy industrial workflow tasks across multiple software-defined data centers while automating the task provisioning, data provisioning and resource provisioning.
- An adaptive Genetic Algorithm (GA) and associated SDNWB for green scheduling of industrial workflow applications.
- Trade-off analysis of different factors such as energy cost, green energy availability and workflow requirements based on real data.
- Extensive experimental evaluation to study the feasibility of the proposed scheduling algorithm and architecture.

## II. System Model

Fig1 present a high level system model with components of SDNWB utilized by public cloud providers for executing the industrial workflow application.

The system $S$ in this paper consists of a set of software-defined data center owned by a provider such as Amazon EC2. $S = \{R_1, R_2, \ldots, R_3\} \cup D$, where $R_i, 1 \leq i \leq k, = \{vm_1, vm_2, \ldots, vm_i, vm_k\} \cup d_i$.

In a data center $R_i$, $vm_i$ is a virtual machine for hosting application services (e.g. a workflow task) and $d_i$ is the cloud specific data repository (such as S3 in the case of Amazon S3 cloud).

A public cloud provider utilizes *Workflow Orchestrator* that deploys the industrial workflow application across multiple data center and the *SDN Controller* optimizes the data transferring among the data centers while executing a workflow application, such that the application can be executed with minimal execution cost and carbon footprint. In particular, the users submit their industrial workflows with all the executables and information such as execution requirements, task description, and the desired security requirements to our broker. *Workflow Orchestrator* is responsible for matching

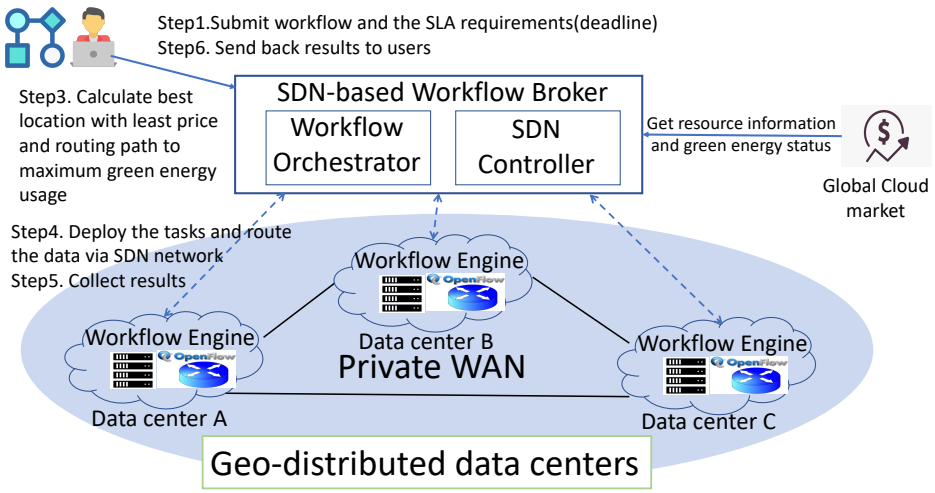


Fig. 1: High level system model for running industrial workflows

different workflow tasks to different data centers based on their electricity prices and usage of green energy. Based on the planning, the *Workflow Orchestrator* interacts with each data center to prepare virtual machines to execute the workflow tasks in a defined order. *SDN Controller* manages the data transfer between different tasks during execution by configuring the flowable of the related SDN-switches.

### A. Cost model

We assume that each $DC_i$ has three types of VM: *small*$(VM)$, *medium*$(VM)$ and *large*$(VM)$. The price of these three VMs are: $4 * Price(Small(VM)) = 2 * Price(medium(VM)) = Price(large(VM)) = 4M$. In this paper, we assume that the VMs are charged according to how long they are used for, therefore the cost of running *small*$(VM)$ for four minutes is $4M$.

Based on the assumptions above, we can have the cost of executing $v_i$ over *small*$(VM)$ as follows:

$$Cost(v_i, vm_i) = (T_l(small(VM))) + T_e(v_i, vm_i) + T_t(v_i, vm_i)) * Price(vm_i) \quad (1)$$

where $T_l(vm_i)$ is time required to launch $VM_i \in \{small(VM), medium(VM), large(VM)\}$, and $T_e(v_i, vm_i)$ represents the time required execute $v_i$ over $vm_i$. $T_t(v_i, vm_i)$ is the the time required transfer $v_i$'s input data to $vm_i$. However, if all the input data are in the same VM, the transferring is equal to 0, i.e., $T_t(v_i, vm_i) = 0$. The models for calculating these times will be detailed in the next subsection. Based on equation 1, we can identify the total cost of executing a workflow over a set of VMs that are allocated over different data centers as follows:

$$TCost(\lambda, VM) = \sum_{v_i \in \lambda,\, VM_i \in VM} Cost(v_i, VM_i) \quad (2)$$

### B. Performance model

As mentioned above, the makespan (or performance) of executing an industrial workflow application over different VMs that are deployed over different data centers includes three parts: the time the VM ($T_l$) is launched, transferring input data to destination VM ($T_t$) and executing the service ($T_e$). In order to model this, we assume that the launching time of each type of VM is the same which is equal to $T_l$. Next, the network throughput (or bandwidth) is $tp$, $2tp$ and $3tp$ corresponding to $small(VM)$, $medium(VM)$ and $large(VM)$, noting $tp_{vm_i}$. However, data transmission rate is not only affected by throughput of the deployed VMs, but also the geographical location of the VMs.

*The same data center:* a workflow includes two tasks $v1$, $v2$, where the data is generated from $v1$ and transferred to $v2$. They are allocated to two different VMs $VM_1$ and $VM_2$, and the VMs are deployed over the same data center. Thus, the throughput between $v1$ and $v2$ is $tp_{v1 \to v2} = min(tp_{vm_1}, tp_{vm_2})$. Thus, the time required to transfer $P$ size of data from $v1$ to $v2$ is: $T_t(v1, v2) = \frac{P}{min(tp_{vm_1}, tp_{vm_2})}$.

Moreover, the host VMs of $v1$ and $v2$ are allocated in *different data centers*. For instance, if the $vm_1$ which is used to host $v1$ is deployed on $DC_1$, and $vm_2$ which is used to host $v2$ is deployed on $DC_2$. So the time of transferring $P$ data from $v1$ to $v2$ is: $T_t(v1, v2) = \frac{P}{min(tp_{vm_1}, tp_{vm_2})} + P * \varphi * H(DC_1, DC_2)$, where $\varphi$ is the average latency incurred at each hop and $H(DC_1, DC_2)$ is the number of core network hops between two data centers.

The execution time of each service depends on the performance of the host VM, in general, the larger VM has better performance. Therefore the execution of time $v_i$ which is hosted on $vm_i$ can be represented as: $T_e(v_i, vm_i)$. Thus, the makespan of executing $\lambda$ is :

$$Makespan(\mathcal{V}, VM) = \sum_{\substack{v_i, v_j \in \mathcal{V} \subset \lambda \\ vm_i, vm_j \in VM \\ i \neq j}} T_l(v_i, vm_i) + T_t(v_i, v_j) + T_t(v_i, vm_i) \quad (3)$$

where $\mathcal{V}$ is a set of the services which belong to the critical path.

### C. Energy model

There are two key operations in the execution of a industrial workflow application where energy is spent: a) in data processing or computation and b) communication. For computation, in general the power consumption of a server varies as a function of its utilization level. If a server is idle, the power saving mechanism lowers the frequency of CPU and thus only a small proportion ($\alpha$) of peak power is utilized. If $\rho$ is peak power consumption and $u$ is utilization of resources, then the power consumption by a host in the data center will be:

$$P_{host}^{comp} = \alpha * \rho + (1 - \alpha) * \rho * u \tag{4}$$

A host may run several VMs at a time thus this power will be spent by each VM according to its usage of resources. Even though a host may have several resources such as CPU cores, disk, memory and other elements, we assume that $u_{vm^i}$ indicates aggregate resources utilized by each VM $i$ hosted on the server. The power consumption of server (host) will be:

$$P_{host}^{comp}(VM) = \sum_{vm_i \in VM} (\alpha * \rho + (1 - \alpha) * \rho * u_{vm_i}) \tag{5}$$

Let one VM $i$ be transmitting data to another VM $j$. Let $H(i, j)$ be the number of hops or routers/switches between these VMs. For communication, the power consumption depends on the bandwidth used by a VM in communication and number of routers those data need to be transmitted from to reach to the destination VM. If $B$ is the total bandwidth available and $\xi_{router}$ is the power consumed by a router, the power consumption for the communication will be

$$P^{comm}(vm_i, vm_j) = \xi_{router} * H(i, j) * \frac{tp_{v1 \to v3}}{B} \tag{6}$$

Therefore, the total energy consumption can be formalized as:

$$TEnergy(\lambda, VM) = P_{host}^{comp}(VM) + \sum_{\substack{vm_i \in VM \\ vm_j \in VM}} P^{comm}(vm_i, vm_j) \tag{7}$$

### D. Electricity cost

The electricity cost is caused by data processing and communication. We compute the electricity cost of data processing by multiplying the local electricity price with energy consumed by the corresponding VM as shown in Eq 8, where $Eprice(vm_i)$ represent electricity price of the data center $vm_i$ deployed.

$$E_{host}^{comp}(VM) = \sum_{vm_i \in VM} [(\alpha * \rho + (1 - \alpha) * \rho * u_{vm_i}) * Eprice(vm_i)] \tag{8}$$

Regarding the electricity cost cause by data exchanging, we assume that the electricity price is a constant value $\Omega$ for each hop. As the result, the total electricity cost for running a workflow with deployment solution $\lambda$ is formalized in Eq 9.

$$TEle(\lambda, VM) = E_{host}^{comp}(VM) + \sum_{\substack{vm_i \in VM \\ vm_j \in VM}} P^{comm}(vm_i, vm_j) * \Omega \tag{9}$$

## III. Proposed Energy-Aware Industrial Workflow Scheduling Algorithm: GreenGA

In this paper, we aim to find an optimized solution that maximizes the proportion of renewable energy and minimizes the real electricity cost under deadline constraints from users. Therefore, this can be considered a dual objective optimization problem.

Given the complexity of the problem with multiple objective functions and constraints, it is not possible to find the solution to the scheduling problem in polynomial time. Thus, we adapted a well known evolutionary algorithm (i.e. Genetic Algorithm) which is known to find the near-optimal solution for scheduling workflow applications. Previously the genetic algorithm has been applied for optimizing makespan of workflow applications, however its applicability and performance has not been tested for optimizing different factors such as revenue, energy consumption and carbon footprint. In our approach, we first converted this multi-objective problem into a single objective optimization problem by multiplying each objective; the resultant problem is formulated as:

$$\begin{aligned} &min\ (f(\lambda) * g(\lambda)) \\ &s.t.\ f(\lambda) = TEle(\lambda, VM) \\ &\quad g(\lambda) = (1 - \sigma(\lambda)) TEnergy(\lambda, VM) \\ &\quad Makespan(\mathcal{V}, VM) \leq deadline \\ &\quad TCost(\lambda, VM) \leq budget \end{aligned} \tag{10}$$

$f$ represents the total monetary cost of running a given workflow with deployment solution $\lambda$. Next, $g$ indicates the non-green energy consumption and where $\sigma(\lambda)$ is a function that calculates the proportion of renewable energy consumption on deployment solution $\lambda$. Finally, $deadline$ and $budget$ are given by users as the hard constraints of the execution time and the maximum deployment cost of the given workflow.

### A. Algorithm details

The aim of genetic algorithm (GA) is to search the solution space and find the best value for objective function or fitness function i.e. combination of renewable energy and energy cost.

To this end, we need to encode the objective function in equation 10. Our deployment solution $\lambda$ is encoded as a vector $[s_1^i; s_2^j :::: s_n^k]$, where $s_j^i$ means that service or task $s_i$ is deployed on cloud $c_j$. Therefore, the vector can be used to compute the value of the objective function as well as the constraints based on the cost model, energy model and performance model. After encoding, we can perform the adaptive GA to compute a sub-optimal solution through the following four phases:

1) *Candidate List Generation and Initializing population:* Initially, we randomly generate the population which is

coded as above. We select the the clouds from the "Candidate List" that lists the clouds meet the constraints of workflow task such as deadline to reduce the possibility of generating an infeasible solution.

2) *Selection:* To generate efficient solutions, two factors (selection pressure and population diversity) have to be carefully considered. We utilized the elitism method [13] given in Algorithm 1, to prevent superior individuals from getting destroyed in crossover and mutation. Thus, if a solution is tagged as elitist, it should be part of the new population generation process. This method can ensure that our algorithm does not waste time to re-discovery the good results that have been already obtained in previous generations. For the selection process two methods are used: fitness function and diversity analysis. The fitness function can transfer the fitness of a coding into a numeric representation to select superior solutions. The fitness function is the same as the objective function defined in Equation 10 which consists of green energy usage and execution cost. The diversity analysis is important step as it influences further steps of crossover and mutation. Low diversity of population usually indicates a local extreme which impacts the search for optimal solutions.
3) *Crossover and Mutation:* Crossover aims to exchanges the parts of two chromosomes to generate two new chromosome. In this paper, we use one-point crossover [14]. Mutation can enhance the search range, thus we developed a solution that randomly select a small proportion (as described in [15]) of chromosomes in current generation and changing them to new feasible chromosomes for next generation. We set a very small initial mutation rate as 0.015. However, it can be dynamically adjusted by our proposed algorithm, detailed in the following.
4) *Diversity Maintenance:* The diversity of a chromosome affects by the mutation rate adjusted by Algorithm 2. Firstly, we compute the density $d$ of the population by comparing unique chromosomes ($sr$) with total number of population ($size$). Next, we increase the mutation rate if $d$ is less than predefined threshold. However, if the mutation rate is higher than its maximum rate, it will be decreased. The increasing and decreasing step is computed as $\frac{1.75}{|\lambda|*|pop|}$ following the suggestion of [16].

The above steps are repeated until the termination constraints are reached.

**Time Complexity.** The proposed method is split into four phases: selection, crossover, mutation and diversity maintenance. The time complexity of the selection phase is $O(|P| \times |G| \times |\mathcal{O}|)$, where P is the size of population; G is the number of generations $\mathcal{O}$ represent the total number of tasks of a given workflow. For crossover phases, we need to operate each individual in every generation, so the complexity of both is $O(|P| \times |G|)$. Although mutation does not operate each individual in each generation, the mutation rate is decided on diversity maintenance phase which requires to sort solutions. Thus, the time complexity mutation and diversity maintenance together is $O(|P| \times |G|)$. As a result, the over all time complexity of the proposed method is $O(|P| \times |G| \times |\mathcal{O}|)$.

**ALGORITHM 1:** Elitist Prevention

```
s–elitist size; elist– elitist list; pop– all individuals O–task list;
  C–cloud list
if elist is empty then
    ▷ ASCsort sorts the pop as ascending order
    pop ← ASCsort (pop)
      ▷ copy the first s number of solutions to elist
    elist ← from pop[0] to pop[s − 1]
end
for o in O do
    for c in C do
        pop ← combine(elist, pop) pop ← ASCsort (pop)
          ▷delete s numbers of pop in tail
        elist ← from pop[0] to pop[s − 1]
    end
end
```

**ALGORITHM 2:** Diversity Protection

```
pop– all individuals; size–size of pop; threshold–threshold of
  diversity; rate–the current mutation rate; Max–maximum
  mutation rate.
▷ function removeDup removes the duplication
rpop ← removeDup(pop)
sr ← |rpop|
d ← 1 − sr/size
if d < threshold then
    increase rate
end
else if rate > Max then
    decrease rate
end
```

### B. Termination Method

If the number of iterations $iter$ is $\infty$, GA can provide an optimal solution. However, the computation resource is limited. In this paper, we terminate our algorithm if there is not further improvement of the solution in a fixed number of interactions $R$.

## IV. Performance Evaluation

### A. Experimental setup

In this work, we use CloudSim [17] to simulate the multiple data center environments to investigate our algorithm. Cloudsim is one of the most widely used simulators in the world and it was evaluated and made comparison with the real-world test-beds in many scenarios, including deploying scientific and industrial workflow on multiple clouds [4], [18].

*1) Cloud provider configurations:*

- **Data center Location and Proportion of Green Energy** We assume there are six **cloud data centers** which are allocated in different areas as shown in Fig. 2. The

| Workflow | Medium | Large | Very large |
|---|---|---|---|
| CyberShake | 30 | 100 | 1000 |
| Montage | 25 | 100 | 1000 |
| LIGO | 30 | 100 | 1000 |
| Epigenomics | 24 | 100 | 995 |

TABLE I: Number of tasks of each workflow at each scale.

| VM type | Small | Medium | Large |
|---|---|---|---|
| Price | 10.5($/h) | 12.8($/h) | 30($/h) |
| Bandwidth | 1000(Kb/s) | 1500(Kb/s) | 4000(Kb/s) |
| CPU | 70(mips) | 80(mips) | 100(mips) |
| Energy consumption | 4.5(kw/h) | 6.5(kw/h) | 10.5(kw/h) |
| VM size | 1000(GB) | 2000(GB) | 4000(GB) |
| RAM | 512(MB) | 2000(MB) | 6000(MB) |

TABLE II: The configuration of different type of VMs

proportion of the usage of green energy of the given area is $\{0.895, 0.895, 0.934, 0.932, 0.622, 0.071\}$.

- **VM configuration** We assume each data center has three types of VM: Small, Medium and Large. Table II shows the configuration of each type of VM. The Small VM for example, it will cost 10.5 US Dollar per hour and its network bandwidth, CPU, VM size and RAM are 1000(Kb/s), 70(mips), 1000(GB) and 512(MB) respectively, where the mips describes the CPU powers, i.e., millions of instructions per second. Also, we assume that the Small VM consume 4.5 kw electric energy per hour.
- **Data center networking**. Fig. 2 indicates the allocation of the data center, and where the weight of each edge represents the number of hops that have been passed for transferring data from one data center to another. Also we assume that the network latency between data centers will add $0.3$ extra time for transferring data from one data center to another on average.
- **Electricity Prices** Market electricity price varies by country and by hour of the day. We use United Kingdom day-ahead market[2] observed over one week to simulate the market electricity cost of each data centre. The prices are modified based on the cost of energy in the countries in which the data centers are allocated [3]. More details can be found in [12].

*2) User Configuration:*

- **Workflow generation** To evaluate our algorithm, four common workflow applications are consider: CyberShake (earthquake risk characterisation), Montage (generation of sky mosaics), LIGO (detection of gravitational waves) and Epigenomics (bioinformatics).[4]. Table I shows the number of tasks of each workflow application. Notably, our simulator only consider input and output data size and execution time of each task.
- **Deadline generation** In this paper, deadline is a hard constraint which is defined as the mean of *fastest solution* and *slowest solution*. *fastest solution* is the deployment solution, deploying the workflow over most the powerful VMs in the same data center, and the *slowest solution* is to deploy the workflow over the least powerful VMs across different data centers.

[2]http://www.nordpoolspot.com accessed 01-06-2015

[3]https://en.wikipedia.org/wiki/$Electricity_pricing$

[4]The XML description files of the workflows are available via the Pegasus project: https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator

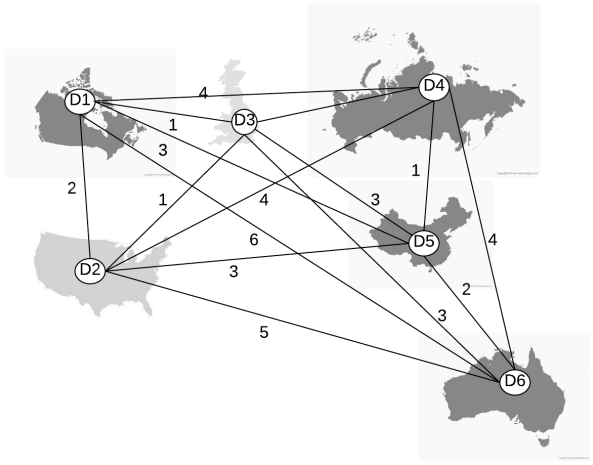


Fig. 2: data centers

### B. Experimental results

We evaluate the performance of our proposed algorithm based on five objectives: electricity cost, energy consumption, deadline and proportion of the usage of renewable energy. To this end, we compare the solutions generated by our proposed algorithm with the best and worst case of each objective. Moreover, we evaluate the performance of our adaptive algorithm in two ways: 1) to optimize only one objective and keep others within predefined constraints; 2) to optimize more than one objectives by mapping these objectives into a weighted linear function while ensuring other objectives within the predefined constraints.

*1) Electricity cost:* We develop two versions of GA-based algorithms namely EleCostGA and GreenGA to optimize the workflow deployment across multiple data centers. EleCostGA only considers minimization of the electricity cost, while meeting other constraints in terms of makespan, energy consumption, and user budgets. GreenGA shares the same constraints as EleCostGA, but also aims to minimize both electricity cost and consumption of non-renewable energy (i.e. green energy).

Figs. 3, 4, 5 show the results of applying both algorithms to different types of workflow. The Lower_bound represents the lowest electricity cost that be obtained without considering any constraints, where Y axis is the ratio of the results generated by proposed algorithms with the Lower_bound (i.e., $\frac{EleCostGA}{Lower\_bound}$ or $\frac{GreenGA}{Lower\_bound}$).

The results illustrate that the cloud providers have to spend more when they optimise the proportion of the usage of renewable energy. However, with the increasing size of the workflows, the differences of electricity cost for GreenGA and ElecCostGA become smaller.

Lower_bound is generated by a greedy based method which is briefly illustrated as follows:

**Lower_bound**. The electricity cost is calculated by $excutionTime * electricityPrice * Consumption$. In this paper, we do not have the electricity price of each hop when data is transferred from one data center to another. Also, in practice this cost is considered by cloud providers. Therefore,

the electricity cost of execution workflow over multiple data centers, which can be simply computed by adding the electricity cost of each data center where the target workflow is allocated. So the Lower_bound can be obtained by minimizing the electricity cost of executing each task of the workflow. To this end, we first select the VMs which can minimize the $excutionTime * Consumption$ of each task, and then choose the data centers which have the cheapest electricity price for deploying the selected VMs (mainly considering when the selected VMs are started and how long the selected VMs will be launched).

Fig. 6 shows the relation between the number of generations of GA and electricity cost. We first set the result of 10 generations as the baseline and then compute how many percentage can be saved with the increasing generations.

*2) Energy efficiency:* In this paper, we do not aim to minimize the energy consumption for executing workflow applications across a set of data centers. However, our proposed method allows cloud providers to set a constraint for energy consumption. To this end, we first provide the Lower_bound and Upper_bound of energy consumption for executing a workflow application over available data centers.

**Lower_bound and Upper_bound**. The Lower_bound of energy consumption is computed by selecting the most energy efficient VMs inside the same data center to execute the given workflow. Furthermore, the most energy efficient Vs for a task $V_i$ is: $\arg\min_{VM_i \in VM} Makespan(v_i, VM_i) * P_{host}^{comp}(VM_i)$. The Upper_bound includes both energy consumption for VM and transferring data over hops. Therefore, we choose the type of VM that consumes the most energy and the data center which contains the most hops for data transmission.

Figs. 7, 8, 9 and 10 indicate the energy consumption of executing the given workflows over different data centers. The Y-axis indicates the ratio of the energy consumption of different solution with the Lower_bound of energy consumption. Although both GreenGA and ElectCostGA are not designed to minimize energy consumption, they can guarantee energy consumption that will meet the predefined constraints, while minimizing the electricity cost.

*3) Green energy efficiency:* This subsection shows the proportion of the usage of renewable energy of each solution which is generated by different algorithms. Figs. 11, 12 and 13 show that the solutions which are generated by GreenGA use more green energy than those generated by EleCostGA. However, the difference reduces with the increasing size of workflows. The larger size of workflow corresponding to more deployment solutions, the GA based methods are easier to reach local optimal. The local optimal has a very high probability of causing the termination of the program by meeting the pre-defined termination condition as described in Subsection III-B.

*4) Performance/makespan:* Deadline is a hard constraint, which means that the execution time of each submitted workflow must be equal to or less than the specific deadline. Fig 14 shows the time saving of the generated solutions, comparing with user given deadlines, where the Y-axis represents the ratio of saving time and the given deadline($\frac{savedTime}{deadLine}$). The X-axis is the type of workflow, where "M_EP", "L_EP" and "VL_EP" are the medium, large and very large size of "Epigenomics" workflows. The results show that all generated solutions can guarantee the given deadline.

## V. Energy efficiency in SDWAN

To evaluate the energy efficiency and flexibility of the SDN for data transmission in WAN, we conduct the following experiments by using IoTSim-SDWAN [19], which simulates multiple cloud data center connected via traditional WAN and SDWAN environments. It provides the facilities to evaluate energy consumption of networks in both traditional WAN and SDWAN environments.

**Experiment configuration**. We consider *two* types of network topology: *small scale WAN* and *large scale WAN*. The number of hops in the large scale WAN are twice as in small scale WAN. Regarding the data centers and workflows, we keep configuration similar to the experiments performed in the previous sections. In the SDN-enabled WAN environment, we use the shortest path to transfer the data between two data centers. However, in the traditional WAN environment, we randomly select a path for the data transmission.

We report the experimental results as the ratio of the energy consumption of SDN based solution and Non-SDN based solution i.e., $SDN/Non-SDN$. Fig. 15 16 and 17 show that SDN-enabled environment consume less energy than traditional WAN environment on transferring data across multiple clouds. The SDWAN solution can save energy up to 32.5%, compared to non-SDWAN solution. However, the advantage reduces with the increase size of workflow. For example, the energy saving of SDWAN solution is around 21.5%. This is because the small scale network topology has less option of routing paths in which are very similar number of hops. The increase size of the workflow, the more shortest paths are selected by the random solution.

When the network topology becomes more complicated, the advantage of the SDWAN become more significant. Compared to non-SDWAN solution, SDWAN solution consumes 73.8% less energy as shown in Fig. 18 19 and 20. Similar to the small scale network typology case, this advantage degrades with the increase size of the workflow. However, this degradation is very slow. From medium size workflow to very large size workflow, the energy saving is reduced from 77% to 70.75%.

## VI. Discussion

The evaluations in the simulated environment show that our proposed algorithm outperforms the comparison methods in terms of green-energy efficiency and network efficiency. In order to conduct the evaluations in the real world environments, the following challenges need to be considered: 1) multi-data-centre SDWAN network which are currently owned and by managed by cloud providers (e.g., Google, Facebook) which is restricted or no access to the network control plane of the data centres; 2) monitoring the energy consumption, while running workflow applications in data centers, is not currently supported by the proprietary cloud monitoring tools (e.g. AWS Cloudwatch); 3) the use of real data centre servers and networks for benchmarking energy efficiency and

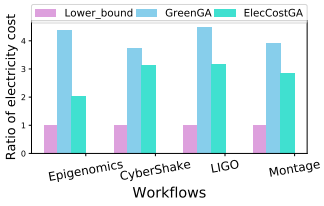


Fig. 3: Electricity cost for medium size workflow

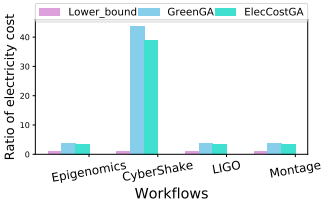


Fig. 4: Electricity cost for large size workflow

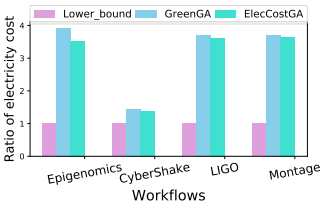


Fig. 5: Electricity cost for very large size workflow

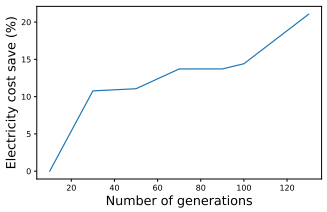


Fig. 6: Electricity cost vs number of generation

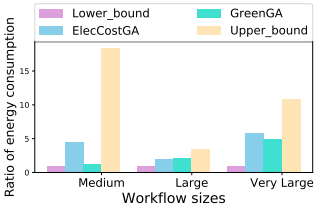


Fig. 7: Energy consumption for Epigenomics

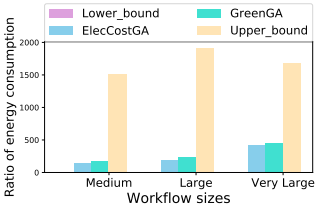


Fig. 8: Energy consumption for CyberShake

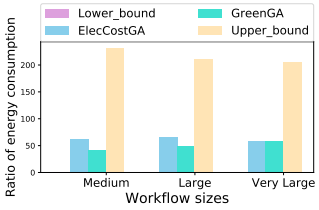


Fig. 9: Energy consumption for Montage

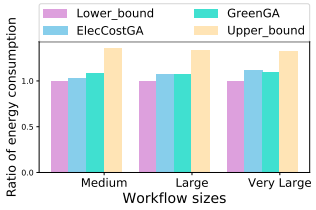


Fig. 10: Energy consumption for LIGO

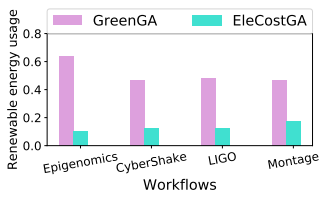


Fig. 11: The proportion of usage of renewable energy for medium size workflow

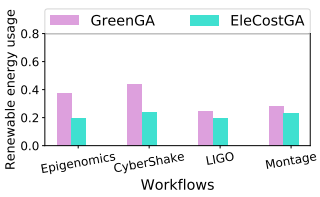


Fig. 12: The proportion of usage of renewable energy for large size workflow

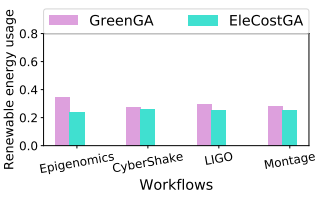


Fig. 13: The proportion of usage of renewable energy for very large size workflow

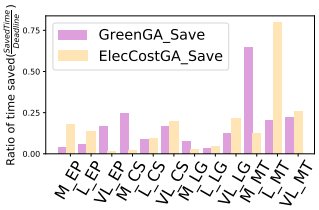


Fig. 14: The presentation of time save comparing with deadline

TABLE III: Compare with other related work

| **Feature/ Research work** | [7] | [8] | [20] | [5] | [4] | [3] | [21] | [22] | Our work |
|---|---|---|---|---|---|---|---|---|---|
| *Workflow application* | × | × | × | × | ✓ | ✓ | × | × | ✓ |
| *SDN enbaled* | × | × | × | ✓ | × | × | ✓ | ✓ | ✓ |
| *WAN aware* | × | × | × | × | × | × | × | × | ✓ |
| *Multiple clouds* | × | × | ✓ | × | ✓ | ✓ | ✓ | ✓ | ✓ |
| *Green energy* | ✓ | × | × | × | × | × | ✓ | ✓ | ✓ |
| *Energy minimization* | × | ✓ | ✓ | × | × | × | ✓ | ✓ | ✓ |
| *Cost minimization* | ✓ | ✓ | ✓ | × | ✓ | ✓ | ✓ | ✓ | ✓ |

workflow application performance is often constrained by their heterogeneity (e.g., hypervisor type, network type). To overcome the above-mentioned challenges, we propose the potential solutions as follows. One can utilize the network trace data (e.g., from B4 project [2] ) to parameterize a micro-benchmark which is emulated by a lab level test-bed. Regarding to the large scale experiments, the environments can be simulated using the real world network trace data to parameterize our simulators. Similarly the energy consumption of different VMs, can be modelled based on the data center cluster traces such as Microsoft Azure Dataset [5] and Alibaba Cluster Data [6].

## VII. Related work

### A. Cost and performance-based tasks scheduling

To improve the performance of run a workflow application, [23] introduced an auto-scaling method to allocate workflow tasks to a set of VMs to meet the deadline constraints.

[24] considered the monetary cost for running workflow over cloud. They developed the algorithm to overcome the trade-off between makespan and financial cost. An new algorithm was proposed in [17] that utilizes the idle time of

[5]https://github.com/Azure/AzurePublicDataset

[6]https://github.com/alibaba/clusterdata

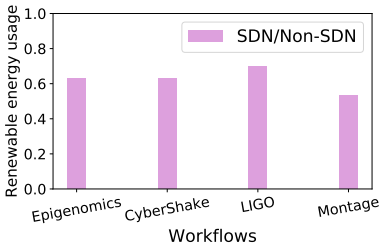


Fig. 15: Energy consumption for transferring medium size workflow data in small scale WAN

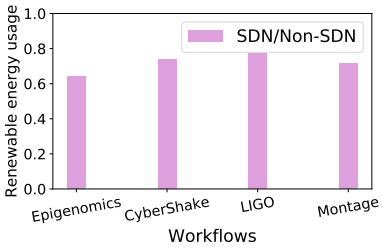


Fig. 16: Energy consumption for transferring large size workflow data in small scale WAN

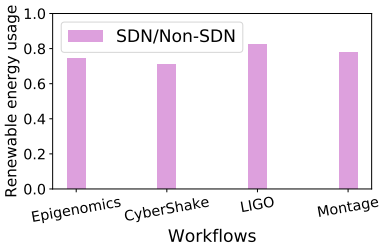


Fig. 17: Energy consumption for transferring very large size workflow data in small scale WAN

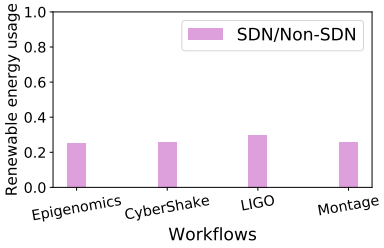


Fig. 18: Energy consumption for transferring medium size workflow data in large scale WAN

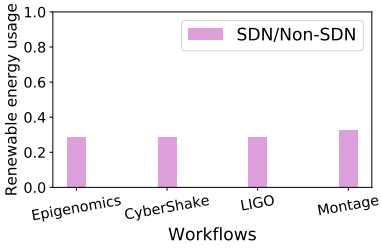


Fig. 19: Energy consumption for transferring large size workflow data in large scale WAN

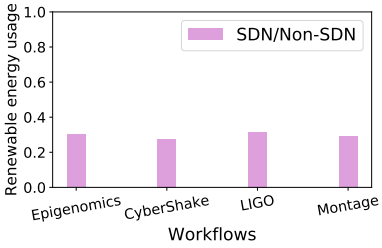


Fig. 20: Energy consumption for transferring very large size workflow data in large scale WAN

provisioned resources and surplus budget to scale up the execution of workflow application to catch of chance of meeting deadlines. There are also some algorithms [25] considering security for running the workflow applications on cloud.

In multiple clouds case, [20] proposed an effective algorithm to scheduling the tasks across public cloud and private, while minimizing the cost and ensuring the delay within a boundary. However, this method is not suitable for scientific or industrial workflow applications. PANDA [26] was developed to schedule workflow across private cloud and public while finding the "best"trade-off between performance and cost. Fard et al. in [27] solve the trade-off between monetary cost and completion time via a Pareto-optimal based algorithm. However, none of them considered security and cloud availability change. Security was consider in [28] that introduced a static method to optimize the deployment of a workflow application on multiple cloud by considering security, makespan and monetary cost.

### *B. Green scheduling*

Various existing proposals [21], [22] suggested different solutions with respect to green scheduling in cloud computing. The author from [29] proposed a predictive energy saving online scheduling algorithm to reduce the energy consumption of distributed web search engines. Y.Li et al.[30] proposed an energy model for edge and core cloud, to estimate the energy consumption based on the number of IoT devices and the desired application QoS. There have been many works [31][32][6][33] that have proposed techniques to improve cloud data center efficiency in terms of electricity usage and decreasing their carbon footprint. For instance, Aksanli et al. [31] proposed a database scheduling strategy which predicts the green energy availability reducing rescheduling of jobs. Goiri et al. [32] also predict green energy availability to schedule map reduce jobs. Deng et al. [33] proposed an online algorithm to minimize the operational cost of data centers by using mutiple energy resources. Kaushik et al. [34] proposed an energy saving cloud storage solution by dividing the storage structures in different zones based on power characteristics. Most of the above works focus on single data centers. Garg et al. [11] proposed a green cloud framework which utilizes multiple clouds to improve energy consumption. However, the work does not give a mechanism to maximize the usage of green energy. Kiani et al.[35] share similar aims to ours i.e. to increase green energy usage and cutting the cost of electricity across multiple data centers. They utilize the concept of decomposing the workload into green and brown, however they focus on a simple workload consisting of individual tasks. Giacobbe et al. [10] and [36] introduce approaches for migrating Virtual Machines among more distributed Federated clouds where costs ([36]) and environmental impact (using renewable energy along with the selection of data centers with the lower PUE [10]) are taken into account. [7] proposed an algorithm that focus on ensuring the deadline of executing tasks on green data centers. These works consider simple computation resources at the level of generic VMs, no further investigation from this perspective is performed.

In summary, to best of our knowledge, our proposed GA-based scheduling is the first work that focuses on increasing usage of green energy and minimizing electricity cost for workflow application execution in multiple cloud data centers.

We also consider variable electricity cost across different cloud data centers. Table III provides a comparison between our work and the state-of-the-arts.

## VIII. Conclusion and Future Work

In recent years, several works have attempted to develop mechanisms to be able to efficiently execute industrial workflow applications in software-defined cloud environments with minimal cost. However, over the years, as usage of cloud increases, concern about its carbon footprint has also became a critical topic of research. In this context, this paper proposed an adaptive GA-based industrial workflow scheduling algorithm that utilizes multiple software-defined cloud data center resources not only to improve green energy usage but also keep the cost of execution to a minimum. The performance of our algorithm has been evaluated using real industrial workflow workload with different sizes under various configurations of virtual machines. We compared our algorithm with another GA-base algorithm that just optimizes electricity cost. The experimental results clearly show that our proposed algorithm favors more green energy usage with expenditure similar to the base algorithm for large and very large size workflows. For smaller size workflows, with 10-20 per cent increase in electricity cost, our algorithm can generate a schedule that uses almost 200 per cent times more green energy.

In future, we will evaluate our proposed algorithm in real cloud environments and integrate with workflow engines. We will also work on improving the algorithm by considering dynamic changes in green energy availability.

## Acknowledgment

This research is supported by the PACE project (EP/T021985/1), SUPER project (EP/R033293/1), the grant of National Natural Science Foundation of China (62072408) and Zhejiang Provincial Natural Science Foundation of China (LY20F020030).

## References

[1] J. Koomey *et al.*, "Growth in data center electricity use 2005 to 2010," *A report by Analytical Press, completed at the request of The New York Times*, vol. 9, p. 161, 2011.

[2] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu *et al.*, "B4: Experience with a globally-deployed software defined wan," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 3–14, 2013.

[3] Z. Wen, J. Cała, P. Watson, and A. Romanovsky, "Cost effective, reliable and secure workflow deployment over federated clouds," *IEEE Transactions on Services Computing*, vol. 10, no. 6, pp. 929–941, 2016.

[4] Z. Wen, R. Qasha, Z. Li, R. Ranjan, P. Watson, and A. Romanovsky, "Dynamically partitioning workflow over federated clouds for optimising the monetary cost and handling run-time failures," *IEEE Transactions on Cloud Computing*, 2016.

[5] H. Yuan, J. Bi, M. Zhou, and K. Sedraoui, "Warm: Workload-aware multi-application task scheduling for revenue maximization in sdn-based cloud data center," *IEEE Access*, vol. 6, pp. 645–657, 2017.

[6] J. Wu, S. Guo, J. Li, and D. Zeng, "Big data meet green challenges: Greening big data," *IEEE Systems Journal*, vol. 10, no. 3, pp. 873–887, 2016.

[7] H. Yuan, J. Bi, M. Zhou, and A. C. Ammari, "Time-aware multi-application task scheduling with guaranteed delay constraints in green data center," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1138–1151, 2017.

[8] J. Bi, H. Yuan, W. Tan, M. Zhou, Y. Fan, J. Zhang, and J. Li, "Application-aware dynamic fine-grained resource provisioning in a virtualized cloud data center," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 1172–1184, 2015.

[9] K. Vahi, I. Harvey, T. Samak, D. Gunter, K. Evans, D. Rogers, I. Taylor, M. Goode, F. Silva, E. Al-Shakarchi *et al.*, "A case study into using common real-time workflow monitoring infrastructure for scientific workflows," *Journal of grid computing*, vol. 11, no. 3, pp. 381–406, 2013.

[10] M. Giacobbe, A. Celesti, M. Fazio, M. Villari, and A. Puliafito, "Evaluating a cloud federation ecosystem to reduce carbon footprint by moving computational resources," in *2015 IEEE Symposium on Computers and Communication (ISCC)*, July 2015, pp. 99–104.

[11] S. K. Garg, C. S. Yeo, and R. Buyya, "Green cloud framework for improving carbon efficiency of clouds," in *European Conference on Parallel Processing*. Springer, 2011, pp. 491–502.

[12] R. Blanco, M. Catena, and N. Tonellotto, "Exploiting green energy to reduce the operational costs of multi-center web search engines," in *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 1237–1247.

[13] D. Bhandari, C. Murthy, and S. K. Pal, "Genetic algorithm with elitist model and its convergence," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 10, no. 06, pp. 731–747, 1996.

[14] R. Poli and W. B. Langdon, "Genetic programming with one-point crossover and point mutation," in *Soft Computing in Engineering Design and Manufacturing*. Springer-Verlag, 1997, pp. 180–189.

[15] D. E. Golberg, "Genetic algorithms in search, optimization, and machine learning," *Addion wesley*, vol. 1989, p. 102, 1989.

[16] J. Smith and T. C. Fogarty, "Self adaptation of mutation rates in a steady state genetic algorithm," in *Proceedings of IEEE international conference on evolutionary computation*. IEEE, 1996, pp. 318–323.

[17] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw. Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[18] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Algorithms for cost-and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds," *Future Generation Computer Systems*, vol. 48, pp. 1–18, 2015.

[19] K. Alwasel, D. N. Jha, E. Hernandez, D. Puthal, M. Barika, B. Varghese, S. K. Garg, P. James, A. Zomaya, G. Morgan *et al.*, "Iotsim-sdwan: A simulation framework for interconnecting distributed datacenters over software-defined wide area network (sd-wan)," *Journal of Parallel and Distributed Computing*, 2020.

[20] H. Yuan, J. Bi, W. Tan, M. Zhou, B. H. Li, and J. Li, "Ttsa: An effective scheduling approach for delay bounded tasks in hybrid clouds," *IEEE transactions on cybernetics*, vol. 47, no. 11, pp. 3658–3668, 2016.

[21] G. S. Aujla and N. Kumar, "Sdn-based energy management scheme for sustainability of data centers: An analysis on renewable energy sources and electric vehicles participation," *Journal of Parallel and Distributed Computing*, vol. 117, pp. 228–245, 2018.

[22] ——, "Mensus: An efficient scheme for energy management with sustainability of cloud data centers in edge–cloud environment," *Future Generation Computer Systems*, vol. 86, pp. 1279–1300, 2018.

[23] M. Mao and M. Humphrey, "Auto-scaling to minimize cost and meet application deadlines in cloud workflows," in *High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for*, Nov 2011, pp. 1–12.

[24] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Cost- and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '12. Los Alamitos, CA, USA: IEEE Computer Society Press, 2012, pp. 22:1–22:11.

[25] J. Mace, A. van Moorsel, and P. Watson, "The case for dynamic security solutions in public cloud workflow deployments," in *Dependable Systems and Networks Workshops (DSN-W), 2011 IEEE/IFIP 41st International Conference on*, June 2011, pp. 111–116.

[26] M. R. H. Farahabady, Y. C. Lee, and A. Y. Zomaya, "Pareto-optimal cloud bursting," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2670–2682, 2013.

[27] H. M. Fard, R. Prodan, and T. Fahringer, "A truthful dynamic workflow scheduling mechanism for commercial multicloud environments," *IEEE Transactions on Parallel and Distributed systems*, vol. 24, no. 6, pp. 1203–1212, 2012.

[28] F. Jrad, J. Tao, I. Brandic, and A. Streit, "{SLA} enactment for large-scale healthcare workflows on multi-cloud," *Future Generation Computer Systems*, vol. 43–44, no. 0, pp. 135 – 148, 2015.
[29] M. Catena and N. Tonellotto, "Energy-efficient query processing in web search engines," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 7, pp. 1412–1425, 2017.
[30] Y. Li, A.-C. Orgerie, I. Rodero, B. L. Amersho, M. Parashar, and J.-M. Menaud, "End-to-end energy models for edge cloud-based iot platforms: Application to data stream analysis in iot," *Future Generation Computer Systems*, vol. 87, pp. 667–678, 2018.
[31] B. Aksanli, J. Venkatesh, L. Zhang, and T. Rosing, "Utilizing green energy prediction to schedule mixed batch and service jobs in data centers," *ACM SIGOPS Operating Systems Review*, vol. 45, no. 3, pp. 53–57, 2012.
[32] Í. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Greenhadoop: leveraging green energy in data-processing frameworks," in *Proceedings of the 7th ACM european conference on Computer Systems*. ACM, 2012, pp. 57–70.
[33] W. Deng, F. Liu, H. Jin, C. Wu, and X. Liu, "Multigreen: Cost-minimizing multi-source datacenter power supply with online control," in *Proceedings of the fourth international conference on Future energy systems*. ACM, 2013, pp. 149–160.
[34] R. T. Kaushik, L. Cherkasova, R. Campbell, and K. Nahrstedt, "Lightning: self-adaptive, energy-conserving, multi-zoned, commodity green cloud storage system," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. ACM, 2010, pp. 332–335.
[35] A. Kiani and N. Ansari, "Toward low-cost workload distribution for integrated green data centers," *IEEE Communications Letters*, vol. 19, no. 1, pp. 26–29, 2015.
[36] M. Giacobbe, A. Celesti, M. Fazio, M. Villari, and A. Puliafito, "An approach to reduce energy costs through virtual machine migrations in cloud federation," in *2015 IEEE Symposium on Computers and Communication (ISCC)*, July 2015, pp. 782–787.

**Zhenyu Wen** is currently a postdoc researcher with the School of Computing, Newcastle University, UK. He received MSc and Ph.D. degrees in computer science from Newcastle University, Newcastle Upon Tyne, UK in 2011 and 2015 respectively. His current research interests include Multi-objects optimisation, Crowdsources, AI and Cloud computing.

**Saurabh Garg** is s a lecturer at the University of Tasmania, Hobart, Tasmania. He is one of the few Ph.D. students who completed in less than three years from the University of Melbourne. His doctoral thesis focused on devising novel and market-oriented meta-scheduling mechanisms for distributed systems under conditions of concurrent and conflicting resource demand. He has gained about three years of experience in the Industrial Research while working at IBM Research Australia and India.

**Gagangeet Singh Aujla** is an Assistant Professor of Computer Science at Durham University. Before this, he worked as a postdoc research associate at Newcastle University, a research associate at Thapar University (India), a visiting researcher at University of Klagenfurt (Austria) and on various other academic positions. He received his PhD degree from the Thapar University (India), his Master and Bachelor degrees from the Punjab Technical University (India). For his contributions to the area of scalable and sustainable computing, I was awarded the 2018 IEEE TCSC Outstanding PhD Dissertation Award of Excellence.

**Khaled Alwasel** is currently working toward a Ph.D. in the School of Computing Science at Newcastle University, UK. He has a B.S. and M.S. in information technology from Indiana University-Purdue University Indianapolis and from Florida International University, USA. Khaled's interests lie in the areas of software-defined networking (SDN), Big Data, IoT, and Cloud Computing.

**Deepak Puthal** is an Assistant Professor (in UK, Lecturer) at School of Computing, Newcastle University, Newcastle upon Tyne, United Kingdom, and an Honorary Fellow at Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. Before this position, he was working as a lecturer at the University of Technology Sydney, and as a graduate research fellow at Data 61, CSIRO, Australia. He has received his Ph.D. in Engineering and IT from the University of Technology Sydney, Australia. He is the recipient of the 2019 Best IEEE ComSoc Young Researcher Award (For EMEA Region), 2018 IEEE TCSC Award for Excellence in Scalable Computing (Early Career Researcher) and 2017 IEEE Distinguished Doctoral Dissertation Award.

**Schahram Dustdar** is Full Professor of Computer Science and head of The Distributed Systems Group at the TU Wien, Austria. From 2004 to 2010 he was also Honorary Professor of Information Systems at the Department of Computing Science at the University of Groningen (RuG), Netherlands. From Dec 2016 until Jan 2017 he was a Visiting Professor at the University of Sevilla, Spain and from January until June 2017 he was a Visiting Professor at UC Berkeley, USA. Schahram is the Editor-in-Chief of Computing (Springer) and an Associate Editor of IEEE Transactions on Cloud Computing, IEEE Transactions on Services Computing, ACM Transactions on the Web, and ACM Transactions on Internet Technology and on the editorial board of IEEE Internet Computing and IEEE Computer.

**Albert Y. Zomaya** is currently the Chair Professor of High Performance Computing & Networking in the School of Computer Science, University of Sydney. He is also the Director of the Centre for Distributed and High Performance Computing which was established in late 2009. Professor Zomaya was an Australian Research Council Professorial Fellow during 2010-2014 and held the CISCO Systems Chair Professor of Internetworking during the period 2002–2007 and also was Head of school for 2006–2007 in the same school.

**Rajiv Ranjan** is a Full professor in Computing Science at Newcastle University, United Kingdom. Before moving to Newcastle University, he was Julius Fellow (2013-2015), Senior Research Scientist and Project Leader in the Digital Productivity and Services Flagship of Commonwealth Scientific and Industrial Research Organization (CSIRO C Australian Government's Premier Research Agency). Prior to that he was a Senior Research Associate (Lecturer level B) in the School of Computer Science and Engineering, University of New South Wales (UNSW). Dr. Ranjan has a PhD (2009) from the department of Computer Science and Software Engineering, the University of Melbourne.