Self-Consumption Optimization of Renewable Energy Production in Distributed Clouds

Benjamin Camus, Anne Blavette, Fanny Dufossé, Anne-Cécile Orgerie

▶ **To cite this version:**

HAL Id: hal-01856660
https://hal.archives-ouvertes.fr/hal-01856660

Submitted on 13 Aug 2018

Self-Consumption Optimization of Renewable Energy Production in Distributed Clouds

Benjamin Camus*, Anne Blavette†, Fanny Dufossé‡ and Anne-Cécile Orgerie*
* Univ. Rennes, Inria, CNRS, IRISA – Rennes, France – Email: {benjamin.camus, anne-cecile.orgerie}@irisa.fr
† Univ. Rennes, CNRS, SATIE, Rennes, France – Email: anne.blavette@ens-rennes.fr
‡ Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP, LIG, 38000 Grenoble, France – Email: fanny.dufosse@inria.fr

Abstract—The growing appetite of new technologies, such as Internet-of-Things, for Cloud resources leads to an unprecedented energy consumption for these infrastructures. In order to make these energy-hungry distributed systems more sustainable, Cloud providers resort more and more to on-site renewable energy production facilities like photovoltaic panels. Yet, this intermittent and variable electricity production is often uncorrelated with the Cloud consumption induced by its workload. Geographical load balancing, virtual machine (VM) migration and consolidation can be used to exploit multiple Cloud data centers' locations and their associated photovoltaic panels for increasing their renewable energy consumption. However, these techniques cost energy and network bandwidth, and this limits their utilization. In this paper, we propose to rely on the flexibility brought by Smart Grids to exchange renewable energy between distributed sites and thus, to further increase the overall Cloud's self-consumption of the locally-produced renewable energy. Our solution is named SCORPIUS: Self-Consumption Optimization of Renewable energy Production In distribUted cloudS. It takes into account telecommunication network constraints and electrical grid requirements to optimize the Cloud's self-consumption by trading-off between VM migration and renewable energy exchange. Our simulation-based results show that SCORPIUS outperforms existing solutions on various workload traces of production Clouds in terms of both renewable self-consumption and overall energy consumption.

Index Terms—Distributed cloud computing, renewable energy, self-consumption, energy-efficient consolidation.

I. INTRODUCTION

The growing appetite of new technologies, such as Internet-of-Things, for data center and Cloud resources leads to an unprecedented energy consumption for these infrastructures [1]. For instance, according to a recent report, in 2014, data centers in the U.S. consumed about 1.8% of total U.S. electricity consumption [2]. This large electricity consumption contributes to global greenhouse gas (GHG) emissions, that are a major cause for climate changes. In order to make these energy-hungry distributed systems more sustainable, Cloud providers resort more and more to on-site renewable energy production facilities like photovoltaic panels. Major Cloud actors like Amazon AWS, Apple or Microsoft are involved in projects to build solar power facilities [3]. Yet, this intermittent and variable electricity production is often uncorrelated with the Cloud consumption induced by its workload [4].

Geographical load balancing is a resource allocation technique exploiting multiple Cloud data centers' locations and their associated photovoltaic panels for increasing their re-

newable energy consumption [5]. Consolidation consists in packing the virtual machines (VMs) on the smallest number of servers in order to switch off unused ones [6]. These solutions can be combined with VM live-migration to dynamically strengthen the workload consolidation on the sites producing more renewable energy. While this technology enables real-time load migration, it costs energy and network bandwidth, and this limits its utilization in practice. Consequently, some of the electricity generated by on-site facilities cannot be consumed by the Cloud.

This paper aims at optimizing the self-power consumption of a distributed Cloud infrastructure with on-site photovoltaic electricity generation. We propose to rely on the flexibility brought by Smart Grids to exchange renewable energy between data centers and thus, to further increase the overall Cloud's self-consumption of the locally-produced renewable energy. Our solution is named SCORPIUS: Self-Consumption Optimization of Renewable energy Production In distribUted cloudS. It optimizes the Cloud's self-consumption by trading-off between VM migration and renewable energy exchange. This optimization is based on an original Smart Grid model to exchange renewable energy between distant sites.

SCORPIUS has been implemented on the SimGrid simulation toolkit [7]. This simulator embeds accurate energy consumption models [8], reliable TCP/IP network models [9] and fine-grain VM and live-migration abstractions [10]. The simulation-based evaluation shows that SCORPIUS outperforms classical greedy and round-robin allocation methods on production workload traces that comprise heterogeneous VMs.

The rest of the paper is organized as follows. Section II presents the background on Smart Grids and Section III the related work on energy savings in Cloud infrastructures. Section IV details the employed Cloud model and Section V specifies the Smart Grid model that we propose to exchange renewable energy. Section VI describes our approach SCORPIUS. Simulation setup is presented in Section VII and results are exploited in Section VIII. Section IX concludes this work and introduces future work.

II. BACKGROUND ON SMART GRIDS

A smart grid is the combination of an electrical network with a dedicated ICT control infrastructure which allows the former to increase its flexibility. Hence, the imminent large-scale deployment of these smart grids will allow the emergence

of new energy management approaches which are intended to increase the share of renewables in the energy mix.

One of these approaches is called collective self-consumption and has appeared, or is about to appear, in several countries [11]. This concept consists in allowing multiple consumers and renewable energy producers, all located in a small geographical area, to establish power purchase agreements (or similar schemes) between themselves. Legislation differs between countries, in particular regarding the definition of the geographical area which can span from a single site (e.g. a building equipped with photovoltaic (PV) panels on its roof) to a wider, but still local, geographical area where consumers and producers are connected to the same low-voltage feeder. This paper proposes an extended, multi-site version of the collective self-consumption approach, where data centers located over a wider geographical area (e.g. over a entire country) would be allowed to exchange their excess of PV electricity between themselves. This approach would allow harnessing the spatial diversity of photovoltaic energy over short timescales in the order of minutes. Hence, it would represent an alternative to VM migration which costs energy. However, exchanging energy, even though it is considered to be performed for free between data centers, implies using the electrical network, which also comes at a cost. This cost, based on network use tariffs, covers technical and non-technical sub-costs, such as:

- Power losses in the distribution equipment (mostly lines and cables), for instance due to their electrical resistance
- Electrical equipment aging
- Grid management services such as metering,
- etc.

Hence, this paper will compare the economical performances of the two mentioned approaches, namely VM migration and energy exchange, as well as the performances of an approach combining these.

III. RELATED WORK ON ENERGY SAVINGS IN CLOUDS

As their increasing electricity bill raises environmental issues, Cloud providers resort more and more to renewable energy [3]. In 2016, according to its environmental responsibility report, 100% of the electricity used by Apple-operated data centers came from renewable energy [12]. In 2011, when Apple started to report on the carbon emissions of their data centers, they were already claiming to reduce them by 56% compared to the case where they would be entirely supplied from the electrical network, whose electricity shows a less environmentally favorable energy mix. Meanwhile, from 2011 to 2016, these carbon emissions should have been multiplied by almost 5 due to the increase in number of Apple-owned data centers [12]. However, this is not the case as this renewable energy is mostly provided by Apple-owned electricity generation facilities including solar arrays, wind farms, biogas fuel cells, and micro-hydro generation systems [12].

Although Infrastructure-as-a-Service (IaaS) Cloud providers intensify their part of renewable energy consumption, they often overestimate their use in proportion to the total consumption and consequently, underestimate their dependence on coal [13]. Indeed, the intermittent nature of current most commonly-used renewable sources (i.e. sun, wind) causes major challenges. Hence, an ideal Cloud manager should match its energy consumption with the renewable energy production. Yet, these two curves are *a priori* uncorrelated.

On one hand, the renewable production can be adjusted through the use of energy storage devices [14]. Yet, this solution is costly and far from ideal as these devices present charge and discharge maximal rates, depth of discharge lower bounds and strong aging effects [15]. Instead, Cloud providers operating on-site renewable energy production, such as Apple, circumvent the issue by selling their production surplus and keeping the 'green credit' of it, while still relying on the local electric grid (mostly based on coal and nuclear power) when their production is insufficient [16]. Furthermore, green energy availability highly depends on the data centers' location that is fixed upon construction [17].

On the other hand, Cloud providers can try to adjust the workload to the energy production. For instance, opportunistic scheduling aims at postponing the Cloud's workload during low-production periods to wait for renewables availability [18]. Distributed Cloud systems can perform geographic load balancing and follow-the-sun resource management in order to increase the green energy use [5], [19], [20]. Consolidation algorithms can also optimize the number of used resources by migrating virtual machines (VM) and switch off unused resources [6]. The consolidation ratio can be further increased by over-commitment techniques that try to jointly minimize Service Level Agreement violation and power consumption [21], [22].

Follow-the-sun and consolidation techniques rely on VM migration capabilities that directly depend on telecommunication network bandwidth, inside and in-between the data centers [23]. Yet, these network constraints, usually underestimated in literature [5], and the related energy consumption prevent VM migration from being the optimal solution that dynamically adjust the Cloud workload and the on-site electricity generation. Furthermore, Cloud data centers are required to provide a high level of availability to their customers, and consequently, some parts of the workload cannot be reshaped or postponed [1]. Smart Grids can bring flexibility in the electricity sources [24] and management [4] for distributed Clouds. But, to the best of our knowledge, no previous work has studied their potential to share renewable energy between the Cloud sites and thus, to increase the overall self-consumption of the infrastructure.

IV. DISTRIBUTED CLOUD MODEL

In this section, we describe the models and underlying assumptions of the Cloud infrastructure.

A. Cloud infrastructure

We consider a distributed cloud with several data centers (DCs) spread across a country. To simplify, we assume all the servers to be homogeneous (i.e. identical memory, number of

cores and energy consumption profiles). Each DC has a different number of servers. Contrary to other work in literature, the complete IP network topology – both between and inside DCs – with its bandwidth and latency constraints, is taken into account. The cloud is powered by a national electrical grid that, following the worst case scenario, only supplies the DCs with so-called *brown* energy, i.e. non-renewable. To reduce its carbon footprint, each DC is equipped with a set of on-site photovoltaic (PV) panels. As it strongly depends on the weather conditions, the PV production is intermittent and variable, and its forecasts include a certain level of error. Each DC is considered as a renewable energy producer and consumer that can trade energy with the electrical grid. The cloud is managed in a centralized way. Its workload consists of Virtual Machines (VMs) submitted at regular time slots. Each VM has different memory, CPU (i.e. cores) and execution time requirements which are *a priori* known. The cloud manager can deploy a VM on any server. It can also perform live migrations of VMs to change their location on the fly. However, we do not consider over-commitment – i.e. servers must always have enough resources to run all their allocated VMs – as they can cause performance interference leading to SLA violations [25].

B. Servers and VM live migration

A server can be in four states: ON, OFF, POWERING_ON and POWERING_OFF. The power consumption of servers ON changes linearly with CPU usage [8]. It is the sum of a fixed part P_{idle} and a variable part proportional to its utilization. Each of the three other states is associated with a static power consumption [26]. A server remains in the POWERING_ON (resp. POWERING_OFF) state for a given fixed duration before being ON (resp. OFF).

We consider that the migration of a VM consists in (1) transferring the VM memory pages, (2) sending a message to notify the end of the stop-and-copy step, and (3) sending the commitment message. Hence, following the TCP/IP model of [9], the duration T_m to migrate a VM is equal to: $T_m = 3L_m + {S_m}/{\rho_m}$ with S_m the memory size of the VM, L_m the latency of the route used for the transfer (equals to 13.01 times the sum of the physical latency of the links), and ρ_m the throughput of the transfer. The transfer throughput can be limited either by the physical bandwidth or the latency of the route. Therefore, it is equal to: $\rho_m = \min\left(0.92B_m, {W}/{2L_m}\right)$ with B_m the physical bandwidth and $W = 4194304$B, the TCP maximum window size. All along the migration, the VM runs on the source server. We consider that the migration requires a full core on the destination server. As a consequence, both the source and the destination servers consume energy during the migration [10].

V. ENERGY EXCHANGE MODEL FOR SMART GRIDS

In this section, we detail our original model for exchanging renewable energy through Smart Grids between data centers generating on-site solar electricity. Note that, as all the data centers are owned by a single-tenant Cloud provider, our model does not target variable local prices as for other work in literature aiming at optimizing the electricity-related expenses [22], [27].

A. Virtual energy pool

The cloud acts as an aggregator of self-consuming data centers which are supplied either by themselves or by another energy supplier, if necessary. This proposed approach constitutes a multi-site version of, and is inspired from, recently deployed, as well as currently envisaged, collective self-consumption schemes [28], [11]. The development of these schemes may also be greatly facilitated by the expected deployment of peer-to-peer energy trading on which a number of projects has been launched already [29].

A data center is considered to consume the PV electricity generated by the panels located on its roof in priority. Then, it injects its excess of PV electricity into the electrical grid, if any. A certain amount of this electricity excess will be considered as virtually exchanged from this data center to (at least) another if one or more data centers consume the same amount of PV electricity during the same time interval Δt_g. This interval corresponds to the usually smallest sampling period of electricity meters.

Otherwise, in the case where the data center is in deficit of energy (i.e. its PV electricity generation is less than its consumption), it consumes electricity from the electrical grid. As explained earlier, a certain amount of this electricity consumption can be considered as exchanged with other data centers if the same amount of PV electricity is injected into the grid by at least one data center during the same time interval Δt_g. Finally, the rest of the electricity consumed by the data center is considered as supplied by another energy supplier.

In other words, the sum of the PV electricity injections into the grid, and exchanged between data centers, constitutes a **virtual pool** into which data centers in deficit of such electricity can tap. The rest of the PV electricity injections into the grid, if any, is considered to be consumed by other energy customers during this time interval.

Fixed prices for the electricity purchased or sold outside the cloud were considered here, as proposed by many energy suppliers [30], [31], as opposed to variable (and sometimes quite volatile), prices indexed on the energy market prices.

B. Grid use cost

The cost of the energy exchanged between data centers (through the virtual pool), is considered as equal to zero. However, tapping into the virtual pool implies using the electrical grid, which comes at a cost, due to power losses, electrical equipment aging, required grid management services, etc. . Assuming that the contract power S_i of any data center i is never exceeded (thus leading to no penalties), this cost C_N, estimated here based on the French grid use tariff (called TURPE [32]), can be expressed as:

$$C_N = C + \sum_{i}^{N_{DC}} bS_i + \sum_{j}^{4} c_j E_j$$

where C is the fixed share of the grid use tariff, N_{DC} is the number of data centers in the cloud, S_i is the contract power of data center i, b is the power coefficient, c_j is the energy coefficient corresponding to period j, and E_j is the energy consumed by the cloud (and originating from the virtual pool) over period j. The latter can be expressed as:

$$E_j = \sum_{t_g} e_j \Delta t_g \quad \min\left(\sum_i \max(0, P_{PV_i}(t_g) - P_{DC_i}(t_g)),\right.$$
$$\left. \sum_i \max(0, P_{DC_i}(t_g) - P_{PV_i}(t_g)))\right) \quad (1)$$

where e_j equals to 1 if t_g is included in period j, 0 otherwise.

C. Energy cost (outside the virtual pool)

The cost of the energy (in €/kWh) supplied by another energy supplier to the cloud is considered to include the grid use cost, as done usually. This cost C_E can be expressed as: $C_E = C_{E,f} + C_{E,v}$ where $C_{E,f}$ and $C_{E,v}$ are the fixed and the variable shares of the energy cost C_E respectively. The variable share $C_{E,v}$, corresponding to the amount of energy ΔE, can be expressed as:

$$C_{E,v} = \begin{cases} -0.06\Delta E & \text{if } \Delta E \text{ is injected into the grid} \\ 0.15\Delta E & \text{if } \Delta E \text{ is purchased} \end{cases}$$

C_E is calculated over each time slot of 10 minutes. This duration corresponds to the smallest temporal resolution of load curves which is provided to customers connected at the distribution level by smart meters in France [33].

VI. SCORPIUS DESCRIPTION

The SCORPIUS method determines VMs' allocation and migration based on comparisons of expected energy costs. Several algorithms determine allocations and migrations decisions at each time slot.

Our approach is based on a stochastic modeling of photovoltaic production. Any decision targets the minimization of the expected energy cost. The expected cost computation, detailed in Section VI-A, is thus a major contribution to this paper. This computation is first used to determine arriving VMs' placement, in Algorithm 2 as detailed in Section VI-B. These allocation decisions are revised by Algorithm 3. The running VMs are then considered for migrations. This step is determined by Algorithm 5 in Section VI-C. The last Algorithm 6, detailed in Section VI-D, migrates VMs independently inside each DC in order to switch off servers. These different steps are summarized in Algorithm 1.

Algorithm 1: General algorithm	
Allocate new VMs;	(Algorithm 2)
Allocation revision;	(Algorithm 3)
Migrate running VMs;	(Algorithm 5)
for $1 \leq i \leq M$ **do**	
Consolidate DC_i	(Algorithm 6)
end for	

A. Expected brown consumption and energy exchange

The photovoltaic energy production on data center DC_i during future time slot t is modeled by a normal law $\mathcal{N}(Eg_i(t), p_i(t))$ truncated in 0. This model is grounded on numerous scientific works that assume that the forecast error of PV production follows a normal distribution [34], [35], [36], [37], [38]. The values of Eg_i and p_i for any DC_i are then determines according to the trace of local green production of the day. Following our approach in [19], our implementation of SCORPIUS computes each expected PV power production $Eg_i(t)$ by averaging a reference PV power production trajectory scaled according to the last PV power production received from i, as depicted on Figure 1. Concerning parameter p_i, we use the standard deviation of the scaled trajectory.
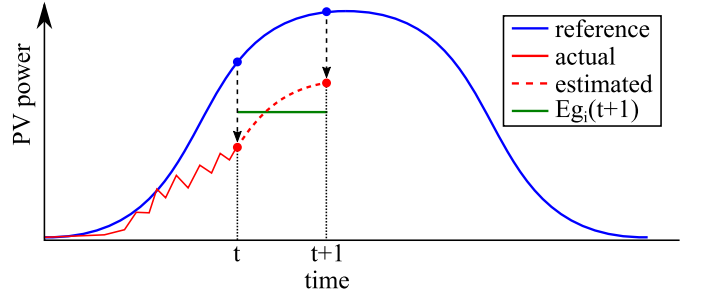

Fig. 1. Expected PV power production computation for a time-slot from t to $t+1$ (from [19]).

Based on this probability distribution, we can compute the expected value of $B_i(t)$, the power consumption of DC_i not provided by local PV generation during time slot t for a total consumption of power P_i:

$$Ec_i(P_i, t) = (P_i - Eg_i)\frac{\Phi_i(P_i) - \Phi_i(0)}{1 - \Phi_i(0)} - p_i^2\frac{\phi_i(0) - \phi_i(P_i)}{1 - \Phi_i(0)},$$

with $\phi_i(x) = \frac{1}{p_i\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-Eg_i}{p_i}\right)^2}$ and $\Phi_i(x) = \frac{1}{2}\left(1 + \text{erf}\left(\frac{x-Eg_i}{p_i\sqrt{2}}\right)\right)$. Computation details on this formula are given in [19]. The expected value of $Gr_i(t)$, the remaining green power (i.e. photovoltaic power), is computed in the same way:

$$EgR_i(P_i, t) = (Eg_i(t) - P_i)\frac{1 - \Phi(P_i)}{1 - \Phi(0)} + p_i(t)^2\frac{\phi(P_i)}{1 - \Phi(0)}.$$

This expected remaining green energy (ERGE) is used in the following to order data centers from the more favorable to green computation to the more expensive. Denote that Formula 1 can be written:

$$E_j = \sum_{t \in j} \Delta t \times E\left(\min\left(\sum_i Gr_i(t), \sum_i B_i(t)\right)\right).$$

The exact value of this expected amount cannot be computed. We thus estimate it by evaluating:

$$\begin{aligned} E'_j &= \sum_{t \in j} \Delta t \min\left(\sum_i E(Gr_i(t)), \sum_i E(B_i(t))\right) \\ &= \sum_{t \in j} \Delta t \min\left(\sum_i EgR_i(P_i, t), \sum_i Ec_i(P_i, t)\right). \end{aligned}$$

This formula strongly simplifies the computation, and gives experimentally good results –i.e. based on our experimental tests, it gives similar results to Monte Carlo simulations with only 1.5% of average error. Concerning the electricity sale at a given time slot, the average value is given by: $E_{sale} = E(\max(\sum_i B_i(t) - P_i(t), 0))$. This formula cannot be evaluated exactly by fast algorithm. Thus, an approximate formula is used.

We consider a random variable B following a truncated normal law defined by parameters $Eg = \sum_i Eg_i$ and $p = \sqrt{\sum_i p_i^2}$. This corresponds to applying the additivity property of normal laws to truncated normal laws. Finally, we obtain formula:

$$\begin{aligned} E'_{sale} = & \quad E(B - P | B > P) P(B > P) \\ = & \quad (Eg - P)\frac{1 - \Phi(P)}{1 - \Phi(0)} + p^2 \frac{\phi(P)}{1 - \Phi(0)} \end{aligned}$$

with $P = \sum_i P_i$. As previously, the average distance has been experimentally validated. The same way, the average purchase amount is evaluated by:

$$\begin{aligned} E'_{sale} = & \quad E(P - B | P > B) P(P > B) \\ = & \quad (P - Eg)\frac{\Phi(P) - \Phi(0)}{1 - \Phi(0)} - p^2 \frac{\phi(0) - \phi(P)}{1 - \Phi(0)}. \end{aligned}$$

To determine the best choice for a VM allocation or migration, the expected cost computation requires to be evaluated over many time slots. However, this computation cannot be done over all time slots for long tasks. We thus use a parameter n_{eval} determining the maximum number of evaluated time slots. Another constraint for evaluations is the computation of parameters Eg_i and p_i. If the evaluation for the current day can be based on the first hours of sun, we do not make hypothesis on the next day PV generation. We thus evaluate the expected costs of VMs until the end of the day, or the end of the VM. More precisely, for a VM of duration t_e allocated at time t, we evaluate the expected cost between t and $t_{max} = min(t_{sunset}, t + t_e)$ and the evaluated time slots will be defined by $t_k = t + k \times \frac{t_{max} - t}{n_{eval}}$, for k between 0 and $n_{eval} - 1$.

In the remaining, we denote P_i the power consumption of data center DC_i with current allocated and pre-allocated VMs, and $EC(P_1, ..., P_{N_{DC}})$ the expected cost for current allocation. For allocation algorithms, $P_i(VM_j)$ is the extra cost corresponding to the VM VM_j on DC_i. In migration algorithms, $P_i(VM_j)$ is the expected extra cost due to migrations. For example, if we decide to migrate VM from DC_i to DC_j, we obtain respectively power consumption $P_i - P_i(VM)$ and $P_j + P_j(VM)$. For the sake of simplicity, we denote the modified expected cost $EC(P_i - P_i(VM), P_j + P_j(VM))$ as it is clear that no other DC is modified.

B. VM allocation

Allocation decisions are taken by two successive algorithms. The first algorithm, Algorithm 2, pre-allocates VMs by decreasing size using a First-Fit approach (this pre-allocation does not imply any VM boot). The second algorithm, Algorithm 3, revisits these decisions with a second pass and effectively launches the new VMs. A server can run a given VM if and only if it satisfies its CPU and memory requirement. We define the *volume* of a VM as the product of these two values. In the same way, the free volume of a server is the product of idle CPUs and unused memory.

Algorithm 2 orders VMs by decreasing volume, and servers by increasing free volume. For each VM, it considers the servers with minimum free volume that can run it (servers that meet CPU and memory constraints of the VM). Then, it pre-allocates the VM on the one with minimum expected cost, that is with minimum difference between expected cost with current allocated and pre-allocated VMs, and expected cost with this additional VMs. Thus, if no server ON can run it on any DC, this algorithm compares expected costs including the switch ON costs.

Algorithm 2 has complexity $O(|L|log(|L|) + |L|(|S| + n_{eval}))$, if we consider the number of DC as a constant and $|S| < |L|$. We do the same assumptions in all complexity formulas.

Algorithm 2: Centralized VM allocation algorithm

> L = sorted list of new VMs (arriving at time t) by decreasing volume
> S = sorted list of not full servers in all DCs by increasing size of available number of vCPU
> **for all** $l \in L$ **do**
> > find list S_{opt} of servers with minimum volume v
> > find server $s \in S_{opt}$ with minimum cost:
> > $$\min_{1 \le i \le M} Ec_i(P'_i + P_{idle} + c_{L[0]} * P_c + E_{ON}) - EP'_i$$
> > allocate l on s
> **end for**

The second algorithm, detailed by Algorithm 3, evaluates the pre-allocations (done by the previous algorithm) by data centers. It considers data centers by increasing ERGE, that is from the worst to the better, and revisits all pre-allocations. For all VMs by decreasing volume, if its current server is incomplete (server with non null free volume), the algorithm finds a server ON that can run it on the best possible DC for ERGE criteria, and changes the allocation if it reduces the global expected energy cost. Our method targets as many full servers as possible, then VM pre-allocated on full servers are not moved.

Algorithm 3 has complexity $O(|Lt_{max}|log(|Lt_{max}|) + |Lt_{max}|(s_{max} + n_{eval}))$, with $|Lt_{max}|$ the maximum value of $|Lt_i|$ and s_{max} the maximum number of servers of a data center.

C. VM migration

For a VM running on a data center with high power consumption regarding the PV generation, we consider two possible options. In the first case, the VM continues on this DC, occurring costs for the virtual pool usage or for purchasing energy, possibly for the duration of the VM. In the second case, the VM is moved to a better location and

---

Algorithm 3: Allocation revisions.

> Order DC by increasing ERGE
> Let Lt_i the list of pre-allocated VMs on incomplete servers of DC_i
> **for all** DC_s in $\{1..M-1\}$ **do**
> > $DC_r = M$
> > Order Lt_{DC_s} by decreasing volume
> > **while** $DC_s < DC_r$ AND Lt_{DC_s} not empty **do**
> > > **for all** VM $t \in Lt_{DC_s}$ **do**
> > > > **if** a server of DC_r can run t AND
> > > > $EC(P_{DC_s} - P_{DC_s}(t), P_{DC_r} + P_{DC_r}(t)) <$
> > > > $EC(P_{DC_s}, P_{DC_r})$ **then**
> > > > > Pre-allocate t on DC_r
> > > > > Remove t from Lt_{DC_s}
> > > >
> > > > **end if**
> > >
> > > **end for**
> > > $DC_r - -$
> >
> > **end while**
>
> **end for**

---

migration costs are incurred, proportionally to the migration duration. For any considered migration, we compare these two costs to determine if the migration is desirable.

VM migration implies to consider migration duration, bandwidth constraints and synchronization issues. Algorithm 5 determines the VMs to migrate, the receiving data centers, and the precise migration time interval in the time slot. To simplify synchronization issues, we limit each DC to send VMs to only two DCs during the considered time slot, or to receive VMs from only two DCs. The DCs are ordered by increasing ERGE. Then, for the considered time slot, the best DC (with higher ERGE) will receive VMs from the two worst DCs (with lower ERGE); the second best will receive VMs from the second and the third best, and so on. This two-DCs rule could seem to be a strong limitation, but simulations show that this limit is far to be reached in practice. In addition, as this operation is executed at each time slot, the potentially delayed migrations are only delayed by 5 minutes.

Algorithm 5 first lists the VMs to migrate for each DC, called Lm_i. Running VMs are ordered by decreasing remaining time, and the first VMs are added to the list until migration times reach the duration Δt of the time slot (5 minutes in our experiments). Then, the list is ordered by decreasing volume for a First-Fit algorithm. For given receiving and sending DCs, t_{min} and t_{max} correspond to the bounds of an interval of time available for migration. It is the larger time interval available on both receiving and sending DCs, knowing that exactly one of these two DCs was already examined for migration during the previous step. These values are respectively initialized to 0 and Δt. Then, t_{send} is the end of the last decided migration between these two DCs. Detailed computations are given in Algorithm 4.

Finally, Algorithm 5 is designed as follows. It first orders DCs by increasing ERGE, and creates a list of VMs to migrate.

Then, it considers DCs one by one in increasing order of ERGE (from the minimum green energy available). For each DC, it considers a first DC where to send VMs. Then, in decreasing order of VM volume, it determines (line 13) if the migration is possible (if a server can run it and if migration time is available) and preferable (in terms of energy cost). After all the VMs of the list are examined, the same is done with a second receiving DC. This is repeated until all DCs have been considered for either sending or receiving VMs.

Algorithm 5 has complexity $O(|Lm_{max}|log(|Lm_{max}|) + |Lm_{max}|(s_{max} + n_{eval}))$.

Algorithm 4: Compute t_{min} and t_{max}

> **if** $t_{min} = t_{send}$ **then**
> > $t_{min} = 0$; $t_{max} = \Delta t$
>
> **else**
> > **if** $t_{min} \leq \Delta t - t_{send}$ **then**
> > > $t_{min} = t_{send}$; $t_{max} = \Delta t$
> >
> > **else**
> > > $t_{max} = t_{min}$; $t_{min} = 0$
> >
> > **end if**
>
> **end if**
> $t_{send} = t_{min}$

Algorithm 5: Migration algorithm

1: Order DC by increasing ERGE
2: Create lists Lm
3: $DC_{receiving} = M$; $DC_{sending} = 1$;
4: $t_{send} = t_{min} = 0$; $t_{max} = \Delta t$;
5: **while** $DC_{sending} < DC_{receiving}$ **do**
6:    Order $Lm_{DC_{sending}}$ by decreasing volume
7:    Compute t_{min} and t_{max}
8:    **if** $Lm_{DC_{sending}}$ is not empty **then**
9:       $DC_{receiving} - -$
10:       Determine migration decisions
11:    **end if**
12:    $DC_{sending} + +$
13: **end while**

D. Consolidation

Eventually, the consolidation algorithm, Algorithm 6, migrates VMs between servers within each data center. The objective is to switch off the highest possible number of servers. This is done by a binary search on the incomplete servers. For each data center, the incomplete servers are ordered by decreasing free volume. Then, the objective of the binary search is to determine the number of the first servers that can be switched off after migration of the VMs on the remaining servers of the DC. In worst case, no server is switched off and no migration is executed.

Algorithm 6 has complexity $O(l \times |S|log(|S|))$ with l the number of VMs in servers S.

SCORPIUS has thus, at each time slot, a complexity of

$$O(l \times log(l) + l \times (s_{max} + n_{eval}) + s_{max} \times l \times log(s_{max})),$$

with s_{max} the maximum number of servers in a DC and l the VMs involved (waiting or running).

Algorithm 6: Consolidation on each DC

Order list S of incomplete servers by decreasing empty volume
$k = \lfloor |S|/2 \rfloor; i = |S|; sol = true$
while $i >= 2$ **do**
  $i = \lceil i/2 \rceil$
  if sol **then**
    $k+ = i$
  else
    $k- = i$
  end if
  if VMs of the k first servers of S can be allocated on the $|S| - k$ last ones **then**
    $sol = true$
  else
    $sol = false$
  end if
end while
if sol=false **then**
  $k - -$
end if
Allocate VMs of the k first servers of S on the $|S| - k$ last ones
Turn off the k first servers

VII. SIMULATION SETUP

We perform modeling and simulations to evaluate the performance of our approach. We ensure rigorous experiments by using the validated SimGrid simulation framework [7] and by considering a realistic context. This context consists of a real cloud infrastructure topology and characteristics, traces of Cloud workload and PV production, and electricity costs based on French tariffs. In the following, we detail this simulation process.

A. Simulation Platform

SimGrid is a modeling and simulation platform dedicated to distributed systems analysis. It embeds sound simulation models of CPUs, TCP/IP networks, VMs, and energy consumption, which have been both theoretically and experimentally assessed by numerous scientific works [9], [10], [8]. The platform accurately simulates the resources usage (i.e. CPU and bandwidth sharing), the execution time and the energy consumption of distributed applications.

B. Simulation inputs

Platform: In our experiments, we consider the infrastructure of the French experimental testbed Grid'5000 [39]. Thus, our cloud comprises 9 geographically distributed DCs for a total of 1,035 servers. The DCs are linked together thanks to 10 Gbps links, and the servers use 1 Gbps links. Within a DC, servers are connected following a tree topology.

Servers: Servers are based on the Taurus cluster of Grid'5000. They are equipped with 2 Intel Xeon E5-2630 CPU with 12 cores each, 32GB memory, 598GB storage. We implement the power model of [18], that is based on real measurements made on Taurus, to simulate the power consumption of each node. Each server consumes 8 W when powered OFF, 97 W when idle, and 220 W at 100% CPU load. Moreover, on average, 103 W are consumed during 6 seconds when powering off, and 127 W during 150 seconds when powering on.

Workloads: We test SCORPIUS with two different real workloads shown in Figure 2. The Eucalyptus IaaS Cloud traces of [40] combine the traces of six different real production systems. The Google trace [41] is an execution trace of Google clusters.

They consist in the list of VM arrival times, required numbers of cores, actual average CPU utilization, and execution time. Following the typical characteristics of T2 VM instances of Amazon EC2 [42], we assume all the VMs to require 2Go of RAM per required core. As the Eucalyptus trace does not contain information of the actual CPU utilization, we assume it to be 100% of the reserved cores (that is the worst case from an energy consumption point of view). During the simulation, SCORPIUS determines which servers are used to run the VMs of the workload.

We scale these workloads for the maximal workload peak to use 80% of the Cloud total resources. It is important to note that these workloads do not constitute a favorable scenario for on-site PV production, because they do not follow the day/night pattern of PV productions. Such unfavorable workloads are typical of IaaS Clouds [40] and thus, represent realistic scenarios.
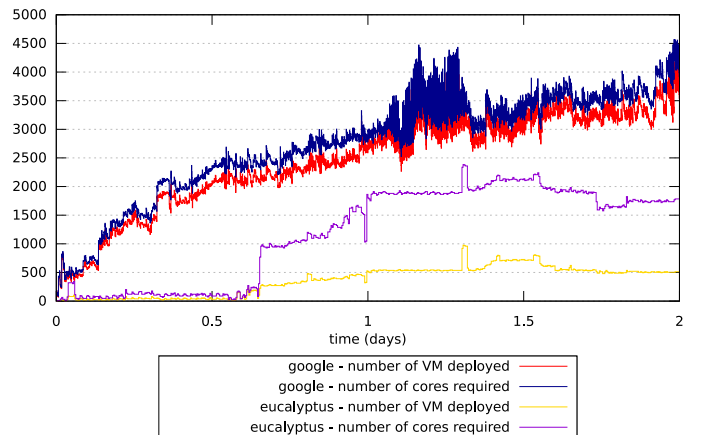

Fig. 2. Workload traces used in our simulation.

Photovoltaic panel traces: We use real recordings of photovoltaic power production collected by the Photovolta project [43] carried out at the University of Nantes. This data is collected every five minutes from four Sanyo HIP-

240-HDE4 PV panels. We use recordings corresponding to different dates to have heterogeneous trajectories between DCs (and thus to represent solar irradiance differences between sites spread across the country). We dimension the PV production of each DC to have one panel for 3 servers. We scale the trace accordingly in our simulations.

Network use cost: The numerical values of the cost were computed based on the French network tariff (TURPE). The corresponding equations were described in Section V-B. The interested reader can refer to [32] for more information. The fixed share of the network use cost corresponds to costs related to management services and was estimated at 397.92€/year. Regarding the power and energy coefficients b and c_j respectively, four time periods are to be considered: peak hours and season peak (period 1, as indicated in Table I), off-peak hours and season peak (period 2), peak hours and season valley (period 3), off-peak hours and season valley (period 4). In addition, two tariff options are proposed: long use time and short use time. As the considered data centers are in service 24/7 and exhibit no day/night patterns, the long use time option is selected. Hence, the following numerical values for coefficients b and c_j have been defined as: $b = 18.34$€/kVA/year

TABLE I
VALUES FOR THE ENERGY COEFFICIENT c_j

	Period 1	Period 2	Period 3	Period 4
c_j (c€/kWh)	4.18	2.81	1.89	1.74

Energy cost: The amount of PV electricity injected into the grid and to be absorbed by other consumers is considered to be sold at a cost of 0.06€/kWh. In the opposite case, the energy supplied to the cloud is considered to be purchased at the cost of 0.15€/kWh. These numerical values are based on French tariffs [44]. Finally, the accumulated amount of energy exchanged between data centers is averaged over time slots of 10 minutes. This duration corresponds to the smallest temporal resolution at which customers can monitor their load curve using smart meters in France [45].

VIII. VALIDATION

In order to validate our approach, we implement SCORPIUS in the SimGrid framework and compare its performance with two classical approaches. In the following, we describe the experimental setup and discuss the results of our simulation.

A. Algorithm implementation

In our simulation, the SCORPIUS algorithm runs each five minutes. At each time slot, we record the PV power generations of each DC, and compute standard deviations $p_i(t)$ using this history.

B. Concurrent approaches

We also simulate two classical greedy approaches for the sake of comparison with the SCORPIUS approach. **Round-Robin** distributes the VMs fairly among the DCs regardless

their PV power production. **First-Fit** deploys each VM on the first (according to an arbitrary predefined order) DC which can host it. These two approaches have been selected based on the approaches implemented in practice in current IaaS software stacks. Indeed, by default, Eucalyptus and CloudStack use a first fit VM placement algorithm [46], [47], while OpenStack's default scheduler employs a combination of filters and weights to spread VMs across all hosts evenly [48], thus obtaining an allocation similar to round-robin with homogeneous servers.

We consider three different algorithms based on each of these two approaches:

1) without using the virtual pool and without switching OFF idle servers. Thus, the data centers cannot exchange their excess of local PV production that are thus lost. These algorithms represent allocation techniques deployed in current production systems. This behavior represents current implementations of Cloud stacks.
2) without using the virtual pool, but with switching OFF the idle servers. These algorithms represent classical approaches of the literature.
3) using the virtual pool and switching OFF idle servers. These algorithm are used to compare SCORPIUS with state-of-the-art algorithms using our proposed energy exchange model.

We also implement the highly-cited reference approach of [49] for reducing power consumption of distributed Clouds. The approach proposes a Modified Best Fit Decreasing (MBFD) algorithm to allocate incoming VMs to servers. A consolidation strategy also uses MBFD to migrate VMs that run on underused servers (i.e. servers with more than 50% of unused CPU). This consolidation may occurs between different DCs. After each consolidation, the unused servers are immediately shut down to reduce the power consumption of the Cloud. Like in SCORPIUS, the consolidation strategy use VM live migration. However, on the contrary to SCORPIUS, the MBFD algorithm does not take into account the local green power productions when allocating VMs. It also neglects the network constraints and the remaining execution time of VMs when migrating them. Finally, the energy cost of these live migrations is not considered. In our simulation, we compare the performance of the MBFD algorithm both with and without using the virtual pool.

The performance of the First-Fit and MBFD algorithms strongly depends on the order of the DCs considered by the algorithm. Thus, we test two opposite configurations corresponding to the best and the worst possible orders. The best (respectively worst) order means the case where the DCs are sorted in decreasing (respectively increasing) total amount of generated PV electricity.

C. Results

In order to validate our proposition, we simulate two days of the Cloud execution. During the simulation, we compute cumulative total and brown energy consumption of the Cloud. We also compute its local self-consumption ratio which corresponds to the ratio of the PV energy consumed locally by

DCs by the total amount of energy it consumes. Finally, we compute the collective self-consumption ratio which is the ratio of PV energy consumed by the cloud, including the virtual pool (in the case where our energy model is considered), by the total amount of consumed energy. At the beginning of the simulation, the Cloud starts without any deployed VM. For the Google (respectively Eucalyptus) trace, during the first simulated day, the VM workload reaches 50% (respectively 33%) of the cloud total capacities. As a consequence, we only measure the performance of the different approaches on the second day. Table II shows the results for the second day.

We can see that the performances are similar with the Google and Eucalyptus workloads: the ranking of the best solutions is almost identical. When we consider the best scenario for First-Fit, we observe better performance than for Round-Robin. Indeed, in all the cases, the best First-Fit consumes less brown energy than Round-Robin and has a higher self-consumption ratio. However, it is worth noting that this best scenario is very unlikely to occur in real production systems as it represents an ideal DC ordering. Moreover, as we can see in the worst scenario when there is no renewable energy exchange, First-Fit can consume significantly more brown energy than Round-Robin.

In all the considered cases, MBFD consumes more total and brown energy than First-Fit and Round Robin. This result might seems counter-intuitive as MBFD performs consolidations that should reduce the power consumption of the Cloud. However, MBFD does not take into account the network constraints and the energy costs of live migrations. Thus, the MBFD consolidation strategy performs too many live migrations of VMs –i.e. the energy saved by switching OFF servers is less than the energy cost of the live migrations. Moreover, some of these migrations are simultaneous and therefore compete for the network bandwidth, which increase their duration and energy costs. Finally, like First-Fit, MBFD performance exhibits large variations between the worst and best scenarios without the energy exchange model (0.4 MWh of difference in the brown energy consumption).

We can observe that switching OFF idle servers – as done by the state-of-the-art solutions – highly reduces the total and brown energy consumption of the cloud (both of about 20% for the Google workload and 40% for the Eucalyptus one). Except for First-Fit in the worst scenario, this also slightly increases the self-consumption in the Cloud.

The proposed energy exchange model significantly reduces the brown consumption thanks to energy exchanges through the virtual pool. We can observe that, thanks to the pool, the green energy losses are reduced by 13% for Round Robin and 34% (respectively 5%) for First-Fit and MBFD in the worst (respectively best) scenario. SCORPIUS exhibits the best performances among all the evaluated approaches. Indeed, its total and brown energy consumption values are lower than all the tested Round-Robin variants, with 3.27 MWh for the total consumption of Round-Robin in the best case, against 3.25 MWh for SCORPIUS, thus saving at minimum 0.02 MWh per day. Besides, when comparing SCORPIUS

with the best First-Fit approach that switches off servers but does not use energy exchange (i.e. representing state-of-the-art approaches), we are able to save about 0.09 MWh per day of brown consumption for Google trace and 0.10 MWh for Eucalyptus trace. This 0.10 MWh of saved brown energy for SCORPIUS represents an extrapolated saving of 36.5 MWh per year for the Cloud (i.e. its brown energy consumption decreases of around 7.6%).

Except when considering our energy model with the Google traces, SCORPIUS consumes even less brown and total energy than the best possible (and unlikely) scenario of First-Fit execution. Indeed, only with Google traces, the best First-Fit approach achieves a brown consumption of about 0.01 MWh less than SCORPIUS. But, in this case as in the others, SCORPIUS has the highest local self-consumption, which means that it uses the electric grid (and the virtual pool) less than the other approaches, and has therefore less impact on it.

Actually, we can observe from the SCORPIUS simulation results of Figure 3, that SCORPIUS uses in priority the DC with the best PV production (e.g. Grenoble, Rennes and Sophia). At the opposite, few VMs are deployed on the DCs with lower power productions (e.g. Reims, Luxembourg and Lille), thus increasing the local self-consumption of the Cloud.

D. Comparison with a theoretical lower bound

Considering the size and complexity of the scheduling problem (NP-hard), an optimal off-line solution cannot be computed in reasonable time to give more guarantees. However, to provide indirect guarantees, we compute a theoretical (i.e. unreachable) lower bound for the total and brown energy consumption. To compute this lower bound, we consider a simpler problem where all the Cloud servers and energy productions are grouped in a single DC. We also neglect the delays and energy cost of VM migrations. Each time a VM starts or stops, we reallocate all the running VMs to the servers using a best-fit decreasing algorithm. The VMs are deployed in priority on the servers that are switched ON, and all the idle servers are turned off. For the Google (resp. Eucalyptus) workload, we found a theoretical lower bound of 3.10 MWh (resp. 1.98 MWh) for the total energy consumption, and of 1.97 MWh (resp. 1.14 MWh) for the brown energy consumption. Thus, SCORPIUS is close to the optimal solution as its brown power consumption is only 4% above this unreachable lower bound for the Google workload and 2.3% above for the Eucalyptus workload.

IX. CONCLUSION AND FUTURE WORK

Improving the sustainability of Cloud infrastructures is currently a major challenge. In this paper, we propose to exploit Smart Grids' flexible energy management to come to the rescue to energy-hungry Clouds. We introduce a novel concept of virtual pool where renewable electricity produced locally on each Cloud data center's site can be virtually transferred to other sites. The proposed model formalizes the cost of exchanging electricity through this virtual pool. An energy-efficient Cloud management approach is designed on

TABLE II
SIMULATED OVERALL CUMULATIVE CLOUD PERFORMANCES.

GOOGLE WORKLOAD

			total consumption	brown consumption	local self-consumption	collective self-consumption	green lost
current implementation	Round Robin		4.11 MWh	2.96 MWh	27.92 %		22.18 %
	First-Fit	best	4.10 MWh	2.88 MWh	29.92 %		16.78 %
		worst		3.10 MWh	24.42 %		32.07 %
State-of-the-art solutions	Round Robin		3.27 MWh	2.27 Wh	30.47 %		32.43 %
	First-Fit	best	3.25 MWh	2.14 MWh	34.03 %		25.03 %
		worst		2.54 MWh	21.85 %		51.88 %
	MBFD	best	3.42 MWh	2.29 MWh	33.2 %		22.91 %
		worst	3.48 MWh	2.72 MWh	21.5 %		49.32 %
With energy pool	Round Robin		3.27 MWh	2.06 MWh	30.47 %	37.14 %	17.66 %
	First-Fit	best	3.25 MWh	2.04 MWh	34.03 %	37.08 %	18.33 %
		worst			21.85 %		
	MBFD	best	3.42 MWh	2.19 MWh	33.2 %	36.13 %	16.12 %
		worst	3.48 MWh	2.23 MWh	21.51 %	35.86 %	15.49 %
	SCORPIUS		3.25 MWh	2.05 MWh	33.47 %	36.82 %	18.94 %

EUCALYPTUS WORKLOAD

|			total consumption	brown consumption	local self-consumption	collective self-consumption	green lost
current implementation	Round Robin		3.36 MWh	2.37 MWh	29.39 %		32.99 %
	First-Fit	best	3.36 MWh	2.26 MWh	32.74 %		25.32 %
		worst		2.52 MWh	25.07 %		42.82 %
State-of-the-art solutions	Round Robin		2.04 MWh	1.39 Wh	32.05 %		55.64 %
	First-Fit	best	2.04 MWh	1.27 MWh	37.81 %		47.64 %
		worst		1.69 MWh	17.50 %		75.76 %
	MBFD	best	2.06 MWh	1.28 MWh	38.02 %		46.95 %
		worst		1.69 MWh	17.89 %		75.03 %
With energy pool	Round Robin		2.04 MWh	1.19 MWh	32.05 %	41.92 %	41.98 %
	First-Fit	best	2.04 MWh	1.19 MWh	37.81 %	41.97 %	41.88 %
		worst			17.50 %		
	MBFD	best	2.06 MWh	1.19 MWh	38.02 %	41.94 %	41.49 %
		worst			17.89 %		
	SCORPIUS		2.02 MWh	1.17 MWh	39.91 %	42.01 %	42.36 %
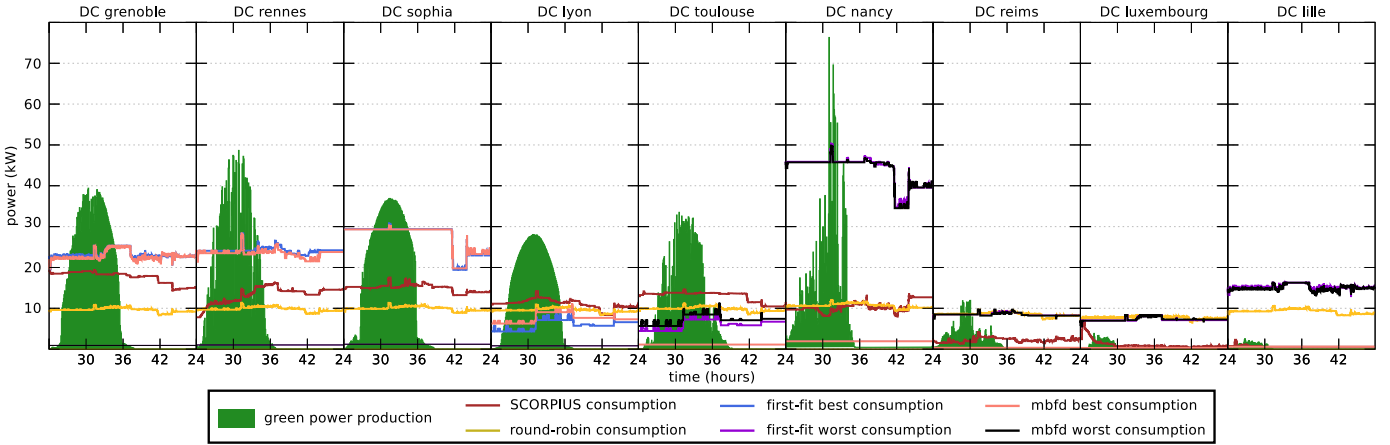
Fig. 3. Comparison of the Cloud simulated consumptions during the second day for the EUCALYPTUS workload with SCORPIUS and the other state-of-the-art solutions.

top of this model. This approach named SCORPIUS (Self-Consumption Optimization of Renewable energy Production In distribUted cloudS) combines energy exchanges and virtual machine migrations among data centers in order to increase the renewable energy consumption. Experimental results on various Cloud production traces show the effectiveness of SCORPIUS in comparison with currently implemented methods and state-of-the-art solutions.

Our future work will consider other types of renewable energy sources and energy storage devices. This requires new prediction models for the expected electricity production and novel algorithms for deciding when to store or release energy.

REFERENCES

[1] C.-H. Hsu, Q. Deng, J. Mars, and L. Tang, "SmoothOperator: Reducing Power Fragmentation and Improving Power Utilization in Large-scale Datacenters," in *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2018, pp. 535–548.

[2] A. Shehabi, S. Smith, N. Horner, I. Azevedo, R. Brown, J. Koomey, E. Masanet, D. Sartor, M. Herrlin, and W. Lintner, "United States Data Center Energy Usage Report," Lawrence Berkeley National Laboratory, Tech. Rep., 2016.

[3] "Clicking Green: who is winning the race to build a green Internet," Greenpeace report, 2017.

[4] T. Dandres, R. F. Moghaddam, K. K. Nguyen, Y. Lemieux, R. Samson, and M. Cheriet, "Consideration of marginal electricity in real-time minimization of distributed data centre emissions," *Journal of Cleaner Production*, vol. 143, pp. 116 – 124, 2017.

[5] A. Rahman, X. Liu, and F. Kong, "A Survey on Geographic Load Balancing Based Data Center Power Management in the Smart Grid Environment," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 214–233, 2014.

[6] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment," *IEEE Network*, vol. 29, no. 2, pp. 56–61, 2015.

[7] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, scalable, and accurate simulation of distributed applications and platforms," *Journal of Parallel and Distributed Computing*, vol. 74, no. 10, pp. 2899–2917, Jun. 2014.

[8] F. C. Heinrich, T. Cornebize, A. Degomme, A. Legrand, A. Carpen-Amarie, S. Hunold, A.-C. Orgerie, and M. Quinson, "Predicting the Energy Consumption of MPI Applications at Scale Using a Single Node," in *IEEE Cluster*, Sep. 2017.

[9] P. Velho, L. Schnorr, H. Casanova, and A. Legrand, "On the Validity of Flow-level TCP Network Models for Grid and Cloud Simulations," *ACM Trans. on Modeling and Computer Simulation*, vol. 23, no. 4, 2013.

[10] L. Pouilloux, T. Hirofuchi, and A. Lebre, "SimGrid VM: Virtual Machine Support for a Simulation Framework of Distributed Systems," *IEEE Transactions on Cloud Computing*, Sep. 2015.

[11] S. Dunlop and A. Roesch, "EU-wide solar PV business models," PV-Financing H2020 project, Tech. Rep., 2016.

[12] Apple, "Environmental Responsibility Report – 2017 Progress Report, Covering Fiscal Year 2016," Apple Inc., Tech. Rep., April 2017.

[13] L. Research, "Coal Computing: How Companies Misunderstand Their Dirty Data Centers," White paper, 2016.

[14] I. Narayanan, D. Wang, A. a. Mamun, A. Sivasubramaniam, H. K. Fathy, and S. James, "Evaluating energy storage for a multitude of uses in the datacenter," in *IEEE International Symposium on Workload Characterization (IISWC)*, 2017, pp. 12–21.

[15] Y. Ghiassi-Farrokhfal, S. Keshav, and C. Rosenberg, "Toward a realistic performance analysis of storage systems in smart grids," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 402–410, 2015.

[16] A. Epstein, "The Truth About Apple's '100% Renewable' Energy Usage," Forbes, January 2016.

[17] V. Depoorter, E. Oró, and J. Salom, "The location as an energy efficiency and renewable energy supply measure for data centres in Europe," *Applied Energy*, vol. 140, no. C, pp. 338–349, 2015.

[18] Y. Li, A.-C. Orgerie, and J.-M. Menaud, "Opportunistic Scheduling in Clouds Partially Powered by Green Energy," in *IEEE GreenCom*, 2015.

[19] B. Camus, F. Dufossé, and A.-C. Orgerie, "A stochastic approach for optimizing green energy consumption in distributed clouds," in *SMARTGREENS*, 2017, pp. 47–59.

[20] S. Figuerola, M. Lemay, V. Reijs, M. Savoie, and B. St. Arnaud, "Converged Optical Network Infrastructures in Support of Future Internet and Grid Services Using IaaS to Reduce GHG Emissions," *Journal of Lightwave Technology*, vol. 27, no. 12, pp. 1941–1946, 2009.

[21] Z. Zhou, J. Abawajy, M. Chowdhury, Z. Hu, K. Li, H. Cheng, A. A. Alelaiwi, and F. Li, "Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms," *Future Generation Computer Systems*, 2017.

[22] S. Malla and K. Christensen, "Reducing Power Use and Enabling Oversubscription in Multi-tenant Data Centers Using Local Price," in *IEEE International Conference on Autonomic Computing (ICAC)*, 2017, pp. 161–166.

[23] Y. Lin and H. Shen, "EAFR: An Energy-Efficient Adaptive File Replication System in Data-Intensive Clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1017–1030, Apr. 2017.

[24] M. Ghamkhari, A. Wierman, and H. Mohsenian-Rad, "Energy Portfolio Optimization of Data Centers," *IEEE Transactions on Smart Grid*, vol. 8, no. 4, pp. 1898–1910, 2017.

[25] X. Chen, L. Rupprecht, R. Osman, P. Pietzuch, F. Franciosi, and W. Knottenbelt, "CloudScope: Diagnosing and Managing Performance Interference in Multi-tenant Clouds," in *IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2015, pp. 164–173.

[26] I. Rais, A.-C. Orgerie, and M. Quinson, "Impact of Shutdown Techniques for Energy-Efficient Cloud Data Centers," in *ICA3PP*, Dec. 2016.

[27] S. Ren, "Managing Power Capacity as a First-Class Resource in Multi-tenant Data Centers," *IEEE Internet Computing*, vol. 21, no. 4, pp. 8–14, 2017.

[28] European Commission, "Promotion of the use of energy from renewable source, amendments adopted by the european parliament on 17 january 2018 on the proposal for a directive of the european parliament and of the council on the promotion of the use of energy from renewable sources (recast)," 2018.

[29] C. Zhang, J. Wu, C. Long, and M. Cheng, "Review of existing peer-to-peer energy trading projects," *Energy Procedia*, vol. 105, pp. 2563–2568, 2017.

[30] EDF Energy, "Fixed price electricity contracts," https://www.edfenergy.com/large-business/buy-energy/fixed-contracts, Accessed Jan., 31 2018.

[31] OVO Energy, "Official website," https://www.ovoenergy.com/pay-monthly/2-year-fixed-energy, Accessed January, 31 2018.

[32] Enedis, "TURPE 5 HBA/BT - tarifs d'utilisation des réseaux publics de distribution d'électricité," Enedis, Tech. Rep., 2017. [Online]. Available: http://www.enedis.fr/sites/default/files/plaquette_tarifaire_turpe-5.pdf

[33] "Cost-benefit analyses & state of play of smart metering deployment in the EU-27," European Commission, Tech. Rep., 2014.

[34] R. Marquez and C. F. Coimbra, "Forecasting of global and direct solar irradiance using stochastic learning methods, ground experiments and the NWS database," *Solar Energy*, vol. 85, no. 5, pp. 746 – 756, 2011.

[35] Y. Gao, J. Zhu, H. Cheng, F. Xue, Q. Xie, and P. Li, "Study of Short-Term Photovoltaic Power Forecast Based on Error Calibration under Typical Climate Categories," *Energies*, vol. 9, no. 7, 2016.

[36] Z. Ziadi, M. Oshiro, T. Senjyu, A. Yona, N. Urasaki, T. Funabashi, and C. H. Kim, "Optimal Voltage Control Using Inverters Interfaced With PV Systems Considering Forecast Error in a Distribution System," *IEEE Transactions on Sustainable Energy*, vol. 5, no. 2, pp. 682–690, 2014.

[37] S. Lin, M. Han, R. Fan, and X. Hu, "Configuration of energy storage system for distribution network with high penetration of PV," in *IET Conference on Renewable Power Generation (RPG)*, 2011, pp. 1–6.

[38] E. Lorenz, J. Hurka, D. Heinemann, and H. G. Beyer, "Irradiance Forecasting for the Power Prediction of Grid-Connected Photovoltaic Systems," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 2, no. 1, pp. 2–10, March 2009.

[39] "Grid'5000," https://www.grid5000.fr.

[40] R. Wolski and J. Brevik, "Using parametric models to represent private cloud workloads," *IEEE Transactions on Services Computing*, vol. 7, no. 4, pp. 714–725, Oct 2014.

[41] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format + schema," Google Inc., Mountain View, CA, USA, Technical Report, Nov. 2011.

[42] "Amazon ec2 instances," https://aws.amazon.com/ec2/instance-types/.

[43] "Photovolta project," http://photovolta2.univ-nantes.fr.

[44] Légifrance, "French Republic Official Journal No 0109 of 10th May 2017, text No 22," https://www.legifrance.gouv.fr/eli/arrete/2017/5/9/DEVR1712972A/jo/texte, 2017.

[45] M. Boillot, *Advanced Smartgrids for Distribution System Operators*. ISTE and Wiley, 2014.

[46] D. Gaurav, A. Raid, and R. T. S., *Energy and Thermal Aware Scheduling in Data Centers*. Wiley, 2012, ch. 11, pp. 301–337.

[47] CloudStack, "Administration Guide," http://docs.cloudstack.apache.org/projects/cloudstack-administration/en/latest/virtual_machines.html, accessed May 2018.

[48] OpenStack Configuration Reference, "Nova scheduler," https://docs.openstack.org/mitaka/config-reference/compute/scheduler.html, accessed May 2018.

[49] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755 – 768, 2012.