Energy Aware Clouds

Anne-Cécile Orgerie, Marcos Dias de Assunção and Laurent Lefèvre

Abstract Cloud infrastructures are increasingly becoming essential components for providing Internet services. By benefitting from economies of scale, Clouds can efficiently manage and offer a virtually unlimited number of resources, and can minimise the costs incurred by organisations when providing Internet services. However, as Cloud providers often rely on large data centers to sustain their business and offer the resources that users need, the energy consumed by Cloud infrastructures has become a key environmental and economical concern. This chapter presents an overview of techniques that can improve the energy efficiency of Cloud infrastructures. We propose a framework termed as Green Open Cloud, which uses energy efficient solutions for virtualised environments; the framework is validated on a reference scenario.

1 Introduction

Cloud solutions have become essential to current and future Internet architectures as they provide on-demand virtually unlimited numbers of compute, storage and network resources. This elastic characteristic of Clouds allows for the creation of computing environments that scale up and down according to the requirements of distributed applications.

Anne-Cécile Orgerie
ENS Lyon - LIP Laboratory (UMR CNRS, INRIA, ENS, UCB), University of Lyon - 46 allée d'Italie 69364 Lyon Cedex 07 - France, e-mail: annececile.orgerie@ens-lyon.fr

Marcos Dias de Assunção
INRIA - LIP Laboratory (UMR CNRS, INRIA, ENS, UCB) - University of Lyon - 46 allée d'Italie 69364 Lyon Cedex 07 - France, e-mail: marcos.dias.de.assuncao@ens-lyon.fr

Laurent Lefèvre
INRIA - LIP laboratory (UMR CNRS, INRIA, ENS, UCB) - University of Lyon - 46 allée d'Italie 69364 Lyon Cedex 07 - France, e-mail: laurent.lefevre@inria.fr

Through economies of scale, Clouds can efficiently manage large sets of resources; a factor that can minimise the cost incurred by organisations when providing Internet services. However, as Cloud providers often rely on large data centers to sustain their business and supply users with the resources they need, the energy consumed by Cloud infrastructures has become a key environmental and economical concern. Data centers built to support the Cloud computing model can usually rely on environmentally unfriendly sources of energy such as fossil fuels [2].

Clouds can be made more energy efficient through techniques such as resource virtualisation and workload consolidation. After providing an overview of energy-aware solutions for Clouds, this chapter presents an analysis of the cost of virtualisation solutions in terms of energy consumption. It also explores the benefits and drawbacks that Clouds could face by deploying advanced functionalities such as Virtual Machine (VM) live migration, CPU throttling and virtual CPU pinning. This analysis is important for users and administrators who want to devise resource allocation schemes that endeavour to reduce the CO_2 footprint of Cloud infrastructures.

As we attempt to use recent technologies (*i.e.* VM live migration, CPU capping and pinning) to improve the energy efficiency of Clouds, our work can be positioned in the context of future Clouds. This work proposes an energy-aware framework termed as Green Open Cloud (GOC) [21] to manage Cloud resources. Benefiting from the workload consolidation [34] enabled by resource virtualisation, the goal of this framework is to curb the energy consumption of Clouds without sacrificing the quality of service (in terms of performance, responsiveness and availability) of user applications. All components of the GOC architecture are presented and discussed: Green policies, prediction solutions and network presence support. We demonstrate that under a typical virtualised scenario GOC can reduce the energy used by Clouds by up to 25% compared to basic Cloud resource management.

The remaining part of this chapter is organised as follows. Section 2 discusses energy-aware solutions that can be applied to Clouds. Then, the energy cost of virtual machines is investigated in Section 3. The GOC architecture is described in Section 4 and evaluated on a typical virtualised scenario Section 5.

2 Overview of Energy Aware Techniques for Clouds

Current Web applications demand highly flexible hosting and resource provisioning solutions [35]. The rising popularity of social network Web sites, and the desire of current Internet users to store and share increasing amounts of information (*e.g.* pictures, movies, life-stories, virtual farms) have required scalable infrastructure. Benefiting from economies of scale and recent developments in Web technologies, data centers have emerged as a key model to provision resources to Web applications and deal with their availability and performance requirements. However, data centers are often provisioned to handle sporadic peak loads, which can result in low resource utilisation [15] and wastage of energy [12].

The ever-increasing demand for cloud-based services does raise the alarming concern of data center energy consumption. Recent reports [29] indicate that energy costs are becoming dominant in the Total Cost of Ownership (TCO). In 2006, data centers represented 1.5 percent of the total US electricity consumption. By 2011, the current data center energy consumption could double [31] leading to more carbon emissions. Electricity becomes the new limiting factor for deploying data center equipments.

A range of technologies can be utilised to make cloud computing infrastructures more energy efficient, including better cooling technologies, temperature-aware scheduling [24, 9, 30], Dynamic Voltage and Frequency Scaling (DVFS) [14, 33], and resource virtualisation [36]. The use of VMs [3] brings several benefits including environment and performance isolations; improved resource utilisation by enabling workload consolidation; and resource provisioning on demand. Nevertheless, such technologies should be analysed and used carefully for really improving the energy-efficiency of computing infrastructures [23]. Consolidation algorithms have to deal with the relationship between performance, resource utilisation and energy, and can take advantage from resource heterogeneity and application affinities [34]. Additionally, techniques such as VM live-migration [6, 13], can greatly improve the capacities of Cloud environments by facilitating fault management, load balancing and lowering system maintenance costs. VM migration provides a more flexible and adaptable resource management and offers a new stage of virtualisation by removing the concept of locality in virtualised environments [38].

The overhead posed by VM technologies has decreased over the years, which has expanded their appeal for running high performance computing applications [37] and turned virtualisation into a mainstream technology for managing and providing resources for a wide user community with heterogeneous software-stack requirements. VM-based resource management systems such as Eucalyptus [26] and OpenNebula [10], allow users to instantiate and customise clusters of virtual machines atop the underlying hardware infrastructure. When applied in a data center environment, virtualisation can allow for impressive workload consolidation. For instance, as Web applications usually present variable user population and time-variant workloads, virtualisation can be employed to reduce the energy consumed by the data center environment through server consolidation whereby VMs running different workloads can share the same physical host. By consolidating the workload of user applications into fewer machines, unused servers can potentially be switched off or put in low energy consumption modes. Yet attracting virtualisation is, its sole use does not guarantee reductions in energy consumption. Improving the energy efficiency of Cloud environments with the aid of virtualisation generally calls for devising mechanisms that adaptively provision applications with resources that match their workload demands and utilises other power management technologies such as CPU throttling and dynamic reconfiguration; allowing unused resources to be freed or switched off.

Existing work has proposed architectures that benefit from virtualisation for making data centers and Clouds more energy efficient. The problem of energy-efficient resource provisioning is commonly divided into two subproblems [22]: at micro-

or host level, power management techniques are applied to minimise the number of resources used by applications and hence reduce the energy consumed by an individual host; and at a macro-level, generally a Resource Management System (RMS) strives to enforce scheduling and workload consolidation policies that attempt to reduce the number of nodes required to handle the workloads of user applications or place applications in areas of a data center that would improve the effectiveness of the cooling system. Some of the techniques and information commonly investigated and applied at a macro- or RMS-level to achieve workload consolidation and energy-efficient scheduling include:

- Applications workload estimation;
- The cost of adaptation actions;
- Relocation and live-migration of virtual machines;
- Information about server-racks, their configurations, energy consumption and thermal states;
- Heat management or temperature-aware workload placement aiming for heat distribution and cooling efficiency;
- Study of application dependencies and creation of performance models; and
- Load balancing amongst computing sites;

Server consolidation has been investigated in previous work [8, 39, 40, 5, 18, 20, 16, 34]. A key component of these systems is the ability to monitor and estimate the workload of applications or the arrival of user requests. Several techniques have been applied to estimate the load of a system, such as exponential moving averages [4], Kalman filters [17], autoregressive models, and combinations of methods [19, 5]. Provisioning VMs in an IaaS environment poses additional challenges as information about the user applications is not always readily available. Section 4.3 describes an algorithm for predicting the characteristics of advance reservation requests that resemble requests for allocating virtual machines.

Fitted with workload-estimation techniques, these systems provide schemes to minimise the energy consumed by the underlying infrastructure while minimising costs and violations of Service Level Agreements (SLAs). Chase *et al.* [5] introduced MUSE, an economy-based system that allocates resources of hosting centers to services aiming to minimise energy consumption. Services bid for resources as a function of delivered performance whilst MUSE switches unused servers off. Kalyvianaki *et al.* [18] introduced autonomic resource provisioning using Kalman filters. Kusic *et al.* proposed a lookahead control scheme for constantly optimising the power efficiency of a virtualised environment [20]. With the goal of maximising the profit yielded by the system while minimising the power consumption and SLA violations, the provisioning problem is modelled as a sequential optimisation under uncertainty and is solved using the lookahead control scheme. Placement of

applications and scheduling can also take into account the thermal states or the heat dissipation in a data center [24]. The goal is scheduling workloads in a data center, and the heat they generate, in a manner that minimises the energy required by the cooling infrastructure, hence aiming to minimise costs and increase the overall reliability of the platform.

Although consolidation fitted with load forecasting schemes can reduce the overall number of resources used to serve user applications, the actions performed by RMSs to adapt the environment to match the application demands can require the relocation and reconfiguration of VMs. That can impact the response time of applications, consequently degrading the QoS perceived by end users. Hence, it is important to consider the costs and benefits of the adaptation actions [40]. For example, Gueyoung *et al.* [16] have explored a cost-sensitive adaptation engine that weights the potential benefits of reconfiguration and their costs. A cost model for each application is built offline and to decide when and how to reconfigure the VMs, the adaptation engine estimates the cost of adaptation actions in terms of changes in the utility, which is a function of the application response time. The benefit of an action is given by the improvement in application response time and the period over which the system remains in the new configuration.

Moreover, consolidation raises the issue of dealing with necessary redundancy and placement geo-diversity at the same time. Cloud providers, as Salesforce.com for example, that offer to host entire websites of private companies [11], do not want to lose entire company websites because of power outages or network access failures. Hence, outage and blackout situations should be anticipated and taken into account in the resource management policies [32].

While the macro-level resource management performs actions that generally take into account the power consumption of a group of resources or the whole data center, at the host-level the power management is performed by configuring parameters of the hypervisor's scheduler, such as throttling of Virtual CPUs (VCPU) and using other OS specific policies. In the proposed architectures, hosts generally run a local resource manager that is responsible for monitoring the power consumption of the host and optimising it according to local policies. The power management capabilities available in virtualised hosts has been categorised as [25]: "soft" actions such as CPU idling and throttling; "hard" actions like DVFS; and consolidating in the hypervisor. CPU idling or soft states consist in changing resource allotments of VMs and attributes of the hypervisor's scheduler (*e.g.* number of credits in Xen's credit scheduler) to reduce the CPU time allocated to a VM so that it consumes less power. Hard actions comprise techniques such as scaling the voltage and frequency of CPUs. Consolidation can also be performed at the host-level where the VCPUs allocated to VMs can be configured to share CPU cores, putting unused cores in idle state, hence saving the energy that would otherwise be used by the additional core to run a VM.

Nathuji and Schwan [25] presented VirtualPower, a power management system for virtualised environments that explores both hardware power scaling and software-based methods to control the power consumption of underlying platforms. VirtualPower exports a set of power states to VM guests that allow guests to use

and act upon these states thus performing their own power management policies. The soft states are intercepted by Xen hypervisor and are mapped to changes in the underlying hardware such as CPU frequency scaling according to the virtual power management rules. The power management policies implemented in the guest VMs are used as "hints" by the hypervisor rather than executable commands. They also evaluate the power drawn by cores at different frequency/voltage levels and suggest that such technique be used along with soft schemes.

3 Investigating the Energy Consumption of Virtual Machines

With the aim of integrating soft schemes such as CPU throttling, the next sections describe simple experiments to evaluate the distance between the idle consumption and the consumption at high utilisation of virtualised servers. The experiments evaluate the additional power drawn by VMs and the impact of operations such as CPU idling, consolidation at the host level and VM live-migration.

3.1 Experimental Scenario

The evaluation has been performed on a testbed composed of HP Proliant 85 G2 servers (2.2GHz, 2 duo core CPUs per node) with Xen open source 3.4.1. Each node is connected to an external wattmeter that logs its instant power consumption; one measurement is taken each second. The storage of these energy logs is performed by a data collector machine. The precision of measurements of this setup is 0.125 Watts whereas the frequency of measurements is one second.

3.2 Virtual Machine Cost

In order to be energy efficient, virtual machines should ideally start and halt quickly and without incurring too much energy usage. It is hence crucial to understand the cost of basic virtual machine operations and statuses (such as boot, run, idle, halt) in terms of energy consumption.

The experiments reported here describe the stages of booting, running and shutting down virtual machines. Each virtual machine is configured to use one VCPU without pinning (*i.e.* we do not restrict which CPUs a particular VCPU may run on using the generic *vcpu-pin* interface), and no capping of CPU credits. We measure the energy consumption by running different configurations, one at a time, each with a different number of virtual machines; from one to six virtual machines on one physical resource.

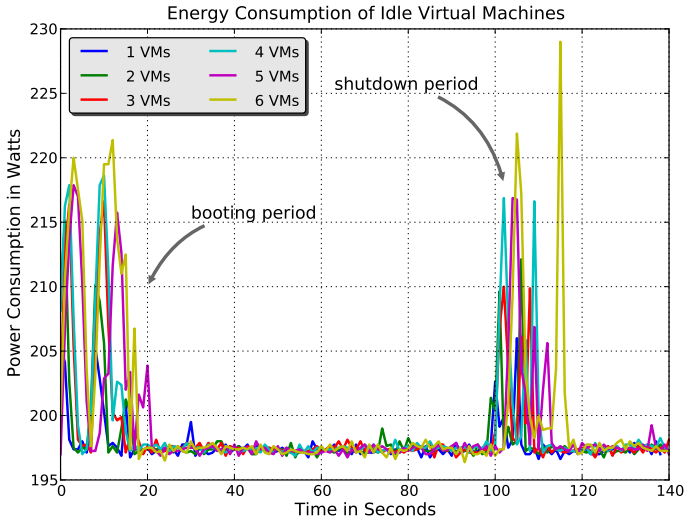

Fig. 1 Energy consumption of different numbers of idle virtual machines on one physical machine.

The graph in Figure 1 shows the energy consumption of virtual machines that are initialised, but do not have a workload (*i.e.*, they do not execute any application after booting). As shown by this figure, the start and shutdown phases of VMs consume an amount of energy that should not be ignored when designing resource provisioning or consolidation mechanisms. The graph in Figure 2, on the other hand, shows the consumption of virtual machines that execute a CPU intensive sample application (*i.e.* cpuburn[1]) once they finish booting. The CPU intensive workload runs for 60 seconds once the virtual machine is initialised and we wait for some time before shutting it down to recognise more clearly the shutdown stage in the graphs.

As shown by Figure 2, the increase in energy consumption resulting from increasing the number of virtual machines in the system depends largely on the number of cores in use. Although the 5th and 6th VMs seem to come at a zero cost in terms of energy consumption since all the cores were already in use, in reality the application performance can be degraded. We have not evaluated the performance in this work since we run a sample application, but in future we intend to explore the performance degradation and the trade-off between application performance and energy savings.

[1] cpuburn is a test application that attempts to use 100% of CPU.

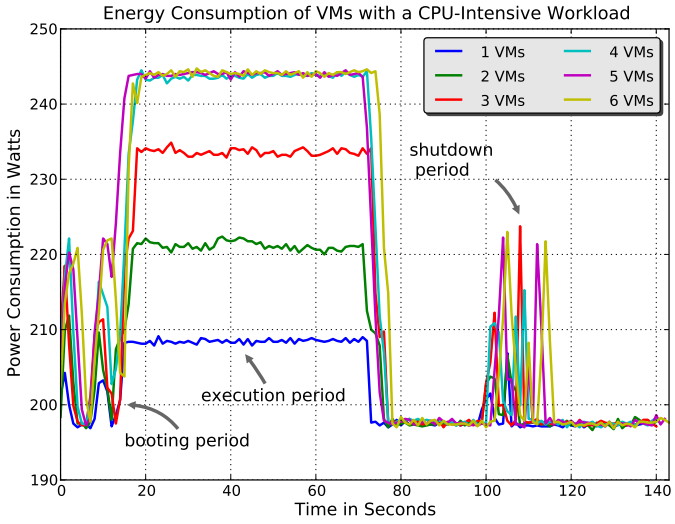

Fig. 2 Energy consumption of virtual machines running a CPU intensive workload on a shared physical resource.

3.3 Migration Cost

This experiment evaluates the energy consumption when performing live migration of four virtual machines running a CPU intensive workload. After all virtual machines are initialised, at time 40 seconds, the live migration starts; one virtual machine is migrated at every 30 seconds. The results are summarised in Figure 3.

The figure shows that migrating VMs leads to an increase in energy consumption during the migration time – factor evidenced by the asymmetry between the lines. Even though the VMs migrated in this scenario had only 512MB of RAM, the experiment demonstrates that when migrating with the goal of shutting down unused resources, one must consider that during the migration, two machines will be consuming energy.

3.4 Capping and Pinning of VCPUs

This experiment measures the energy consumption of virtual machines when we vary parameters of the credit scheduler used by Xen hypervisor. The first parameter evaluated is the cap of CPU utilisation. Valid values for the cap parameter when using one virtual CPU are between 1 and 100, and the parameter is set using the *xm sched-credit* command. Initially, we run a virtual machine with a CPU intensive workload and measure the energy consumption as we change the CPU cap. The experiment considers the following caps: 100, 80, 60, 40 and 20. Once the virtual

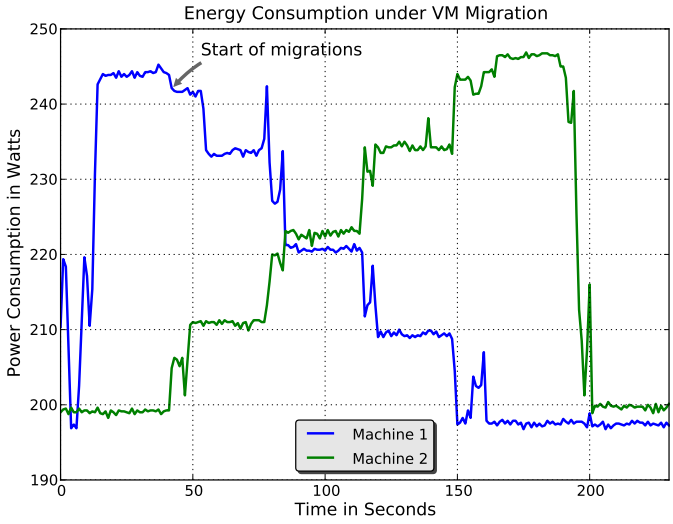

Fig. 3 Migration of 4 virtual machines starting after 40 seconds, one migration every 30 seconds.

machine is initialised and the CPU intensive workload is started, the virtual machine remains during one minute under each cap and is later shut down. The results are summarised in Figure 4.

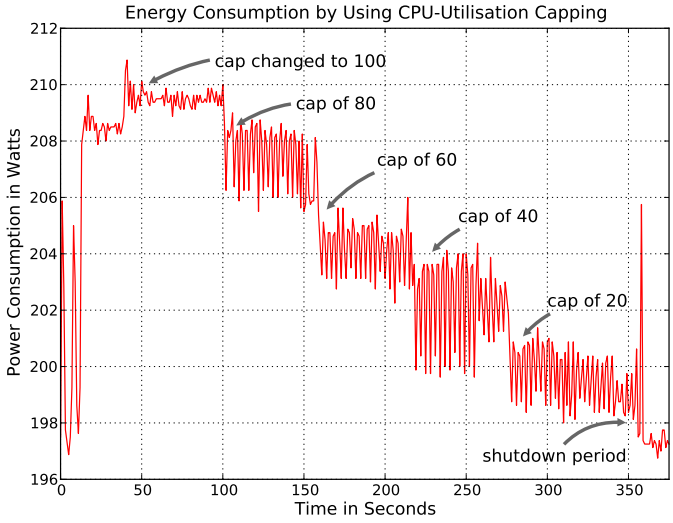

Fig. 4 Energy consumption of a virtual machine under different CPU utilisation caps.

Although capping has an impact on power consumption, this impact is small when considering only one core, and under some cap values the energy consumption does not remain stable. However, the difference in consumption is more noticeable when throttling the CPU usage of multiple VMs.

When virtual machines are initialised, the credit scheduler provided by Xen hypervisor is responsible for deciding which physical cores the virtual CPUs will utilise. However, the assignment of virtual CPUs to physical cores can be modified by the administrator via Xen's API by restricting the physical cores used by a virtual machine; operation known as "pinning". The second set of experiments described here evaluate the energy consumption when changing the default pinning carried out by Xen's hypervisor.

This experiment first initialises two VMs with the CPU intensive workload; then, leaves them run under default pinning for one minute; after that, throttles the VCPUs by forcing them to share the same core; next, changes the core to which the VCPUs are pinned at each minute; after that, removes pin restrictions; and finally shuts down the VMs. Figure 5 summarises the results.

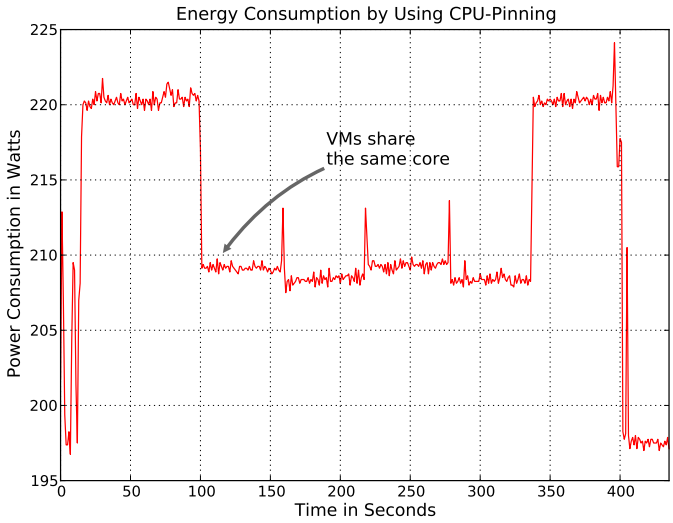

Fig. 5 Energy consumption by making the virtual CPUs of two VMs share the same core.

The energy consumption reduces as the number of utilised cores in the system decreases. Although this may look obvious, the results demonstrate that it is possible to consolidate a workload at the host level and minimise the energy consumption by using capping. For example, a hosting center may opt for consolidating workloads if a power budget has been reached or to minimise the heat produced by a group of servers. Our future work will investigate the trade-off's between performance degradation and energy savings achieved by VCPU throttling when provisioning resources to multi-tier Web applications.

4 The Green Open Cloud

4.1 The Green Open Cloud Architecture

The energy footprint of current data centers has become a critical issue, and it has been shown that servers consume a great amount of electrical power even when they are idle [28]. Existing Cloud architectures do not take full advantage of recent techniques for power management, such as Virtual Machine (VM) live migration, advance reservations, CPU idling, and CPU throttling [25].

We attempt to make use of recent technologies to improve the energy efficiency of Clouds, thus placing our work in the context of future Clouds. This work proposes an energy-aware framework termed as Green Open Cloud (GOC) [21] to manage Cloud resources. Benefitting from the workload consolidation [34] enabled by resource virtualisation, the goal of this framework is to curb the energy consumption of Clouds without sacrificing the quality of service (in terms of performance, responsiveness and availability) of user applications.

In the proposed architecture, users submit their reservation requests via a Cloud portal (*e.g.* a Web server) (Figure 6). A reservation request submitted by a user to the portal contains the required number of VMs, its duration and the wished start time. GOC manages an agenda with resource reservations; all reservations are strict, and once approved by GOC, their start times cannot change. This is like a green Service Level Agreement (SLA) where the Cloud provider commits to give access to required resources (VMs) during the entire period that is booked in the agenda. This planning provides a great flexibility to the provider, that can have a better control on how the resources are provisioned.

Following the pay-as-you-go philosophy of Clouds, the GOC framework delivers resources in a pay-as-you-use manner to the provider: only utilised resources are powered on and consume electricity. The first step to reduce electricity wastage is to shut down physical nodes during idle periods. However, this approach is not trivial as a simple on/off policy can be more energy consuming than doing nothing because powering nodes off and on again consumes electricity and takes time. Hence, GOC uses prediction algorithms to avoid frequent on/off cycles. VM migration is used to consolidate the load into fewer resources, thus allowing the remaining resources to be switch off. VCPU pinning and capping are used as well for workload consolidation. GOC can also consolidate workloads considering time by aggregating resource reservations. When a user submits a reservation request, the Green Open Cloud framework suggests alternative start times for the reservation request along with the predicted amount of energy it will consume under the different scenarios. In this way, the user can make an informed choice and favour workload aggregation in time, hence avoiding excessive on/off cycles. On/off algorithms also pose problems for resource management systems, which can interpret a switched-off node as a failure. To address this issue, GOC uses a trusted proxy that ensures the nodes' network presence; the Cloud RMS communicates with this proxy instead of the switched-off nodes (Figure 6).

The key functionalities of the GOC framework are to:

- monitor Cloud resources with energy sensors to take efficient management decisions;
- provide energy usage information to users;
- switch off unused resources to save energy;
- use a proxy to ensure network presence of switched-off resources and thus provide inter-operability with different RMSs;
- use VM migration to consolidate the load of applications in fewer resources;
- predict the resource usage to ensure responsiveness; and
- give "green" advice to users in order to aggregate resource requests.

The GOC framework works as an overlay atop existing Cloud resource managers. GOC can be used with all types of resource managers without impacting on their workings, such as their scheduling policies. This modular architecture allows a great flexibility and adaptivity to any type of future Cloud's RMS architecture. The GOC framework relies on energy sensors (wattmeters) to monitor the electricity consumed by the Cloud resources. These sensors provide direct and accurate assessment of GOC policies, which helps it in using the appropriate power management solutions.

The GOC architecture, as described in Figure 6, comprises:

- a set of energy sensors providing dynamic and precise measurements of power consumption;
- an energy data collector which stores and provides the energy logs through the Cloud Web portal (to increase the energy-awareness of users);
- a trusted proxy for supporting the network presence of switched-off Cloud resources; and
- an energy-aware resource manager and scheduler which applies the green policies and gives green advice to users.

The Cloud RMS manages the requests in coordination with the energy-aware resource manager which is permanently linked to the energy sensors. The energy-aware resource manager of GOC can be implemented either as an overlay on the existing Cloud RMS, or as a separate service. Figure 7 depicts a scenario where GOC is implemented as an overlay on the existing Cloud RMS. In the resource manager, the beige boxes represent the usual components of a future Cloud RMS (with agenda) whereas the green boxes depict the GOC functionalities. These add-ons are connected to the RMS modules and have access to the data provided by users (*i.e.* submitted requests) and the data in the reservation agenda.

When a user submits a request, the *admission control* module checks whether it can be admitted into the system by attempting to answer questions such as: Is the user allowed to use this Cloud? Is the request valid? Is the request compliant with

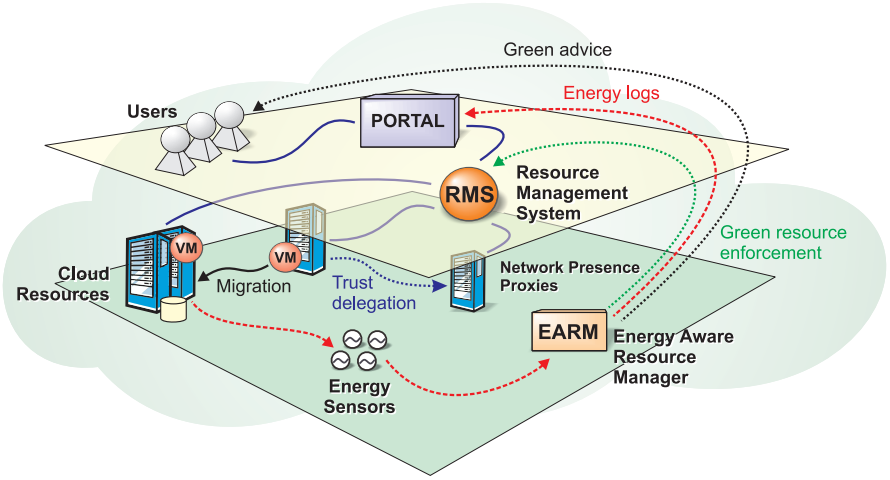

Fig. 6 The GOC architecture.

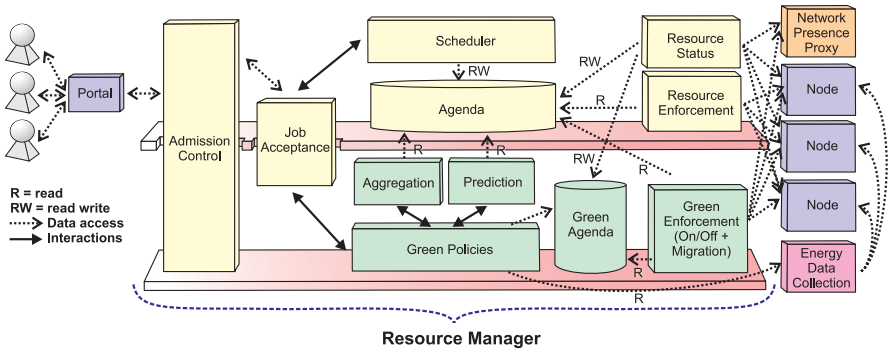

Fig. 7 The GOC resource manager.

the Cloud's usage chart? Then, the *job acceptance* module transfers the request to the *scheduler* and to the *green resource manager* (also called energy-aware manager and labeled "*Green Policies*" in Figure 7). The *scheduler* queries the *agenda* to see whether the requested reservation can be placed into it whilst the *green resource manager* uses its *aggregation* and *prediction* modules to find other less energy-consuming possibilities to schedule this reservation in accordance with its green policies. All the possible schedules (from the Cloud's *scheduler* and from the *green resource manager*) are then transferred to the *job acceptance* module which presents them to the user and prompts her for a reply.

The *agenda* is a database containing all the future reservations and the recent history (used by the prediction module). The *green agenda* contains all the decisions of the green resource manager: on/off and VM migrations (when to switch on and off the nodes and when to migrate VMs). These decisions are applied by the *green*

enforcement module. The *resource enforcement* module manages the reservations (gives the VMs to the users) in accordance with the *agenda*. The *resource status* component periodically polls the nodes to know whether they have hardware failures. If the nodes are off, the component queries the *network presence proxy*, so the nodes are not woken up unnecessarily. The *energy data collector* module monitors the energy usage of the nodes and gives access to this information to the *green resource manager*. These data are also put on the Cloud Web portal, so users can see their energy impact on the nodes and increase their energy-awareness.

4.2 Network Presence

As we switch off unused nodes, they do not reply to queries made by the Cloud resource manager, which can be considered by the RMS as a resource failure. This problem is solved by using a trusted proxy to ensure the network presence of the Cloud resources. When a Cloud node is switched off, all of its basic services (such as ping or heartbeat services) are migrated to the proxy that will then answer on behalf of the node when asked by the resource manager. The key issue is to ensure the security of the infrastructure and to avoid the intrusion of malicious nodes. Our trust-delegation model is described in more details in previous work [7]. This mechanism allows a great adaptivity of the GOC framework to any Cloud RMS.

4.3 Prediction Algorithms

As reservations finish, the prediction algorithm predicts the next reservation for the freed resources. If the predicted reservation of a resource is too close (in less than T_s seconds), the resource remains powered on (it would consume more energy to switch it off and on again), otherwise it is switched off. This idea is illustrated by Figure 8.

For a given resource, T_s is given by the following formula:

$$T_s = \frac{E_{ON \to OFF} + E_{OFF \to ON} - P_{OFF}(\delta_{ON \to OFF} + \delta_{OFF \to ON})}{P_{idle} - P_{OFF}}$$

where P_{idle} is the idle consumption of the resource (in Watts), P_{OFF} the power consumption when the resource is off (in Watts), $\delta_{ON \to OFF}$ the duration of the resource shutdown (in seconds), $\delta_{OFF \to ON}$ the duration of the resource boot, $E_{ON \to OFF}$ the energy consumed to switch off the resource (in Joules) and $E_{OFF \to ON}$ the energy consumed to switch on the resource (in Joules).

In the same way, we define temporal parameters based on the energy consumption of the resources to know in which case it would be more energy efficient to use VM migration. T_m is the bound on the remaining reservation's duration beyond

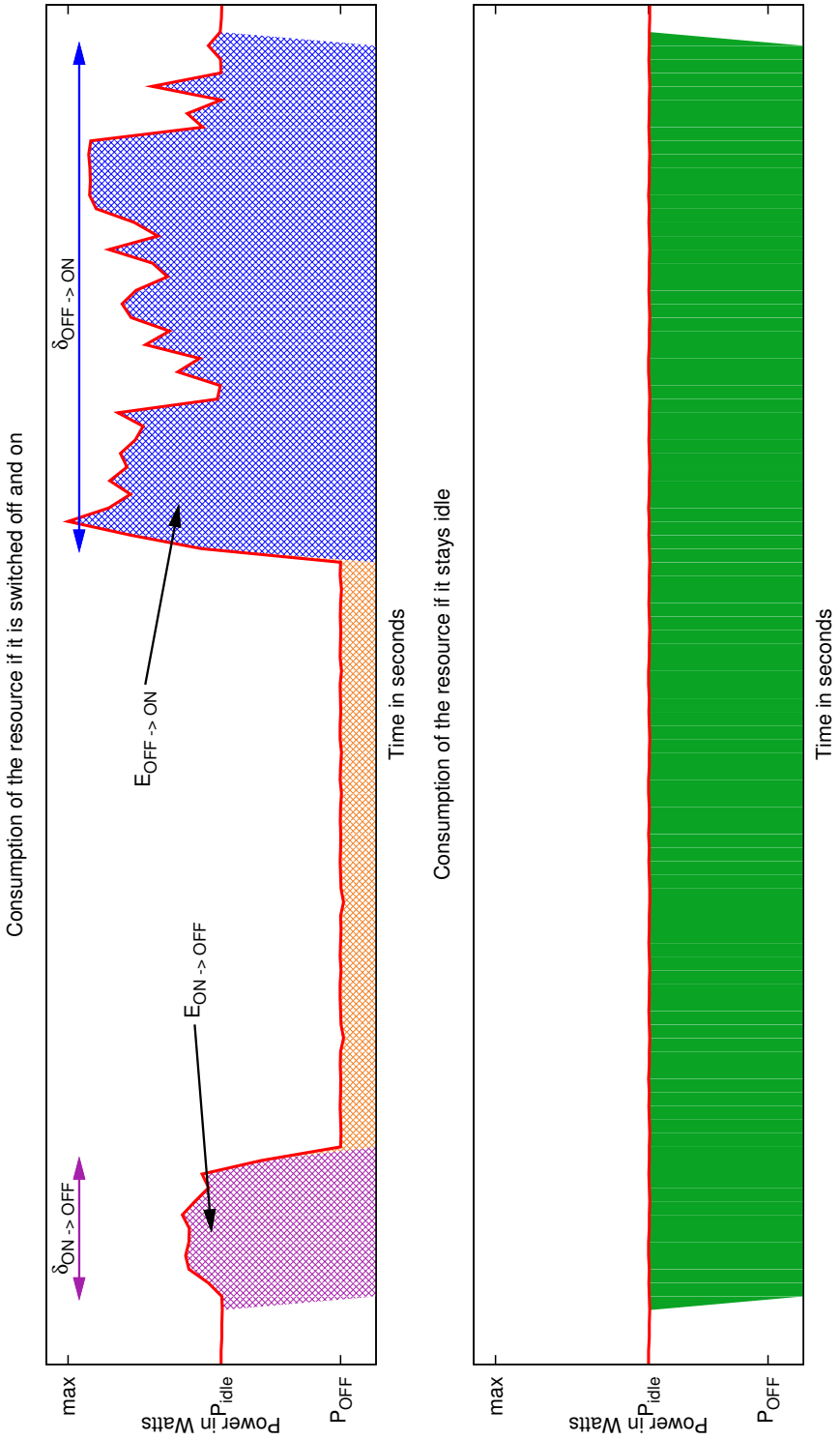

Fig. 8 Definition of T_s

which migration is more energy efficient: if the reservation is still running for more than $T - m$ seconds, the VMs should be migrated, otherwise they stay on their respective physical hosts. As the migration takes time, migration of small jobs is not useful (with a duration lower than 20 seconds in that example). This lower time bound is denoted T_a. If a migration is allowed, the *green enforcement* module also waits T_a seconds after the beginning of a job before migrating it. It indeed allows initialising the job and the VM, so the migration is simplified and we avoid the migration of small jobs, crashed jobs or VMs (in case of technical failure or bad configuration of the VM).

For the next reservation, the predicted arrival time is the average of the inter-submission time of the previous jobs plus a feedback. The feedback is computed with the previous predictions; it represents an average of the errors made by computing the n previous predictions (n is an integer). The error is the difference between the true observation and the predicted value. For estimating other features of the reservation (size in number of resources and length), the same kind of algorithm is used (average of the previous values at a given time).

We have seen in previous work [27] that even with a small n (5 for example) we can obtain good results (70% of good predictions on experimental Grid traces). This prediction model is simple, but it does not need many accesses and is really fast to compute, which are crucial features for real-time infrastructures. This prediction algorithm has several advantages: it works well, it requires a small part of history (no need to store big amounts of data) and it is fast to compute. Therefore, it will not delay the whole reservation process, and as a small part of the history is used, it is really responsive and well adapted to request bursts as it often occurs in such environments.

4.4 Green Policies

Among the components of a Cloud architecture, up to now, we have focused on virtualisation, which appears as the main technology used in these architectures. It also uses migration to dynamically unbalance the load between the Cloud nodes in order to shut down some nodes, and thus save energy.

However, GOC algorithms can employ other Cloud components, like accounting, pricing, admission control, and scheduling. The role of the green policies is to implement such strong administrator decisions at the Cloud level. For example, the administrator can establish a power budget per day or per user. The *green policies* module will reject any reservation request that exceeds the power budget limit. This mechanism is similar to a green SLA between the user and the provider.

Green accounting will give to "green" users (the users that accept to delay their reservation in order to aggregate it with others to save energy) credits which are used to give more priority to the requests of these users when a burst of reservation requests arrives. A business model can also lean on this accounting to encourage users to be energy aware.

5 Scenario and Experimental Results

5.1 Experimental Scenario

This section describes the first results obtained with a prototype of GOC. Our experimental platform consists of HP Proliant 85 G2 Servers (2.2 GHz, 2 dual core CPUs per node) with XenServer 5.0[2] [3, 1] on each node.

The following experiments aim to illustrate the working of GOC infrastructure in order to highlight and to compare the energy consumption induced by a Cloud infrastructure with different management schemes. Our experimental platform consists of two identical Cloud nodes, one resource manager that is also the scheduler, and one energy data collector. All these machines are connected to the same Ethernet router. In the following we will call 'job' a reservation made by a user to have the resources at the earliest possible time. When a user submits a reservation, she specifies the length in time and the number of resources required. Although this reservation mechanism may look unusual, it is likely to be used by next generation Clouds where frameworks would support these features to help cloud providers avoid overprovisioning resources. Having scheduling reservations with limits, such as a time duration, will help Cloud managers to manage their resources in a more energy-efficient way. This is, for instance, the case of a user with budget constraints and whose reservations will have a defined time-frame that reflects how much the user is willing to pay for using the resources. In addition, this reflects a scenario where service clouds with long-live applications have their resource allotments adapted according to changes in cost conditions (*e.g.* cost changes in electricity) and scheduled maintenances.

Our job arrival scenario is as follows:

- $t = 10$: 3 jobs of length equals to 120 seconds each and 3 jobs of length 20 seconds each;
- $t = 130$: 1 job of length 180 seconds;
- $t = 310$: 8 jobs of length 60 seconds each;
- $t = 370$: 5 jobs of length 120 seconds each, 3 jobs of length 20 seconds each and 1 job of length 120 seconds, in that order.

The reservations' length is short in order to keep the experiment and the graph representation readable and understandable. Each reservation is a computing job with a *cpuburn* running on the VM. We have seen that the boot and the shutdown of a VM are less consuming than a *cpuburn* and that an idle VM does not consume any noticeable amount of energy (see Figure 2). Hence, we will just include the *cpuburn* burn phase in the graphs in order to reduce their length and improve their readability (VMs are booted before the experiment start and halted after the end of the experiment).

[2] XenServer is a cloud-proven virtualisation platform that delivers the critical features of live migration and centralised multi-server management.

These experiments, although in a small scale for clarity sake, represent the most unfavourable case in terms of energy consumption as *cpuburn* fully uses the CPU which is the most energy consuming component of physical nodes. Moreover, as CPU is a great energy consuming component, *cpuburn* jobs are accurately visible on the power consumption curves: clear steps are noticeable for each added VM (as previously noticed in Figure 2).

Each node can host up to seven VMs. All the hosted VMs are identical in terms of memory and CPU configuration and they all host a *debian etch* distribution. As our infrastructure does not depend on any particular resource manager, we do not change the scheduling of the reservations and the assignment of the virtual machines to physical machines. The only exception is when a physical node is off, where we then attribute its jobs to the awoken nodes if they can afford it; otherwise we switch it on.

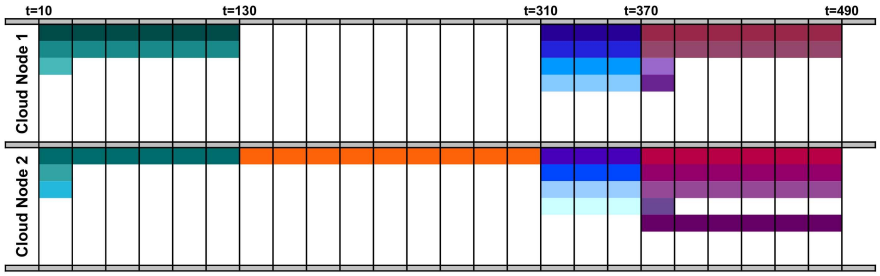

Fig. 9 Gantt chart for the round-robin scheduling.

In order to validate our framework and to prove that it achieves energy saving regardless of the underlying scheduler, we have studied two different schedulings:

- *round-robin*: first job is assigned to the first Cloud node, the second job to the second node, and so on. When all the nodes are idle, the scheduler changes their order (we do not always attribute the first job in the queue to the first node). The behaviour of this scheduling mechanism with the previously defined job arrival scenario is shown in Figure 9. This is a typical distributed-system scheduling algorithm.
- *unbalanced*: the scheduler puts as many jobs as possible on the first Cloud node and, if there are still jobs left in the queue, it uses the second node, and so on (as before, when all the nodes are idle, we change the order to balance the roles). We can see this scheduling with the previously defined job arrival scenario in Figure 10. This scheduling is broadly used for load consolidation.

These two scheduling algorithms are well-known and widely used in large-scale distributed system schedulers. For each of these algorithms, we use four scenarios to see the difference between our VM management scheme and a basic one. The four scenarios are as follows:

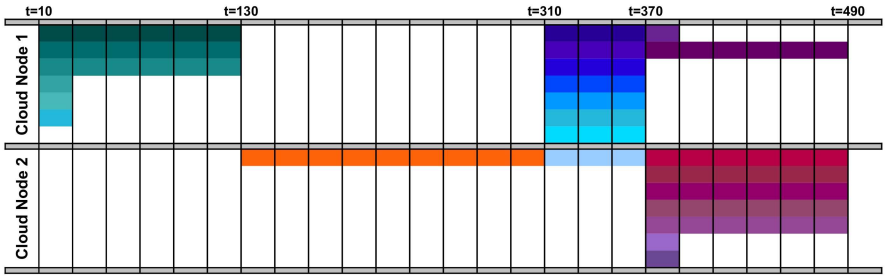

Fig. 10 Gantt chart for the unbalanced scheduling.

- *basic*: no changes are made, this scenario represents the Clouds with no power management.
- *balancing*: migration is used to balance the load between the Cloud nodes. This scenario presents the case of a typical load balancer which aims to limit node failures and heat production.
- *on/off*: unused nodes are switched off. This is the scenario with a basic power management;
- *green*: we switch off the unused nodes and we use migration to unbalance the load between Cloud nodes. This allow us to aggregate the load on some nodes and switch off the other ones. This is the scenario that corresponds to GOC.

5.2 Results

For each scheduling, we have run the four scenarios, one at a time, on our experimental platform and we have logged the energy consumption (one measure per node and per second at the precision of 0.125 Watts). A more extensive discussion on the results is available in previous work [21].

Figure 11 shows the course of experiment for the two nodes with the round-robin scheduling applied to the green scenario. The upper part of the figure shows the energy consumption of the two nodes during the experiment while the lower part presents the Gantt chart (time is in seconds).

At time $t = 30$, the second job (first VM on Cloud node 2) is migrated to free Cloud node 1 and then to switch it off. This migration does not occur before $T_a = 20$ seconds. Some small power peaks are noticeable during the migration. This confirms the results of Section 3.3 stating that migration is not really costly if it is not too long.

This migration leads to the re-allocation of the job starting at $t = 130$ on Cloud node 1 since Cloud node 2 has been switched off and Cloud node 1 is available. At $t = 200$, Cloud node 2 is booted to be ready to receive the jobs starting at $t = 310$. During the boot, an impressive consumption peak occurs. It corresponds to the physical start of all the fans and the initialisation of all the node components. Yet, this peak is more than compensated by the halting of the node ($P_{OFF} = 20$ Watts)

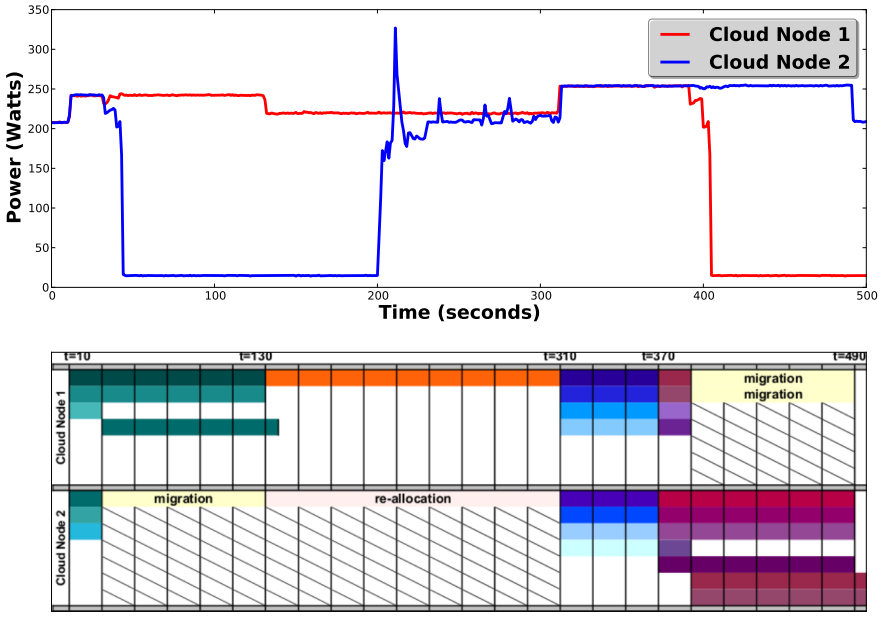

Fig. 11 Green scenario with round-robin scheduling.

during the time period just before the boot since the node inactivity time was greater than T_s (defined in Section 4.3).

During the seventh job, one can notice that a running VM with a *cpuburn* inside consumes about 10 Watts which represents about 5% of the idle consumption of the Cloud node. At time $t = 390$, two new VM migrations are performed, and they draw small peaks on the power consumption curves. From that time, Cloud node 1 is switched off and Cloud node 2 has 6 running VMs. We observe that the fifth and the sixth VMs cost nothing (no additional energy consumption compared to the period with only four VMs) as explained in Section 3.2 as this Cloud node has four cores. This experiment shows that GOC greatly takes advantage of the idle periods by using its green enforcement solutions: VM migration and shut down. Job re-allocation also allows to avoid on/off cycles. Significant amounts of energy are saved.

The green scenario with unbalanced scheduling is presented in Figure 12. For the green scenario, the unbalanced scheduling is more in favour of energy savings. Indeed, two migrations less are needed than with the round-robin scheduling, and all the first burst jobs are allocated to Cloud node 1 allowing Cloud node 2 to stay off. In fact, this is the general case for all the scenarios (Figure 13): the unbalanced scheduling is more energy-efficient since it naturally achieves a better consolidation on fewer nodes.

Summarised results of all the experiments are shown in Figure 13. As expected, the green scenario which illustrates GOC behaviour is the less energy consuming

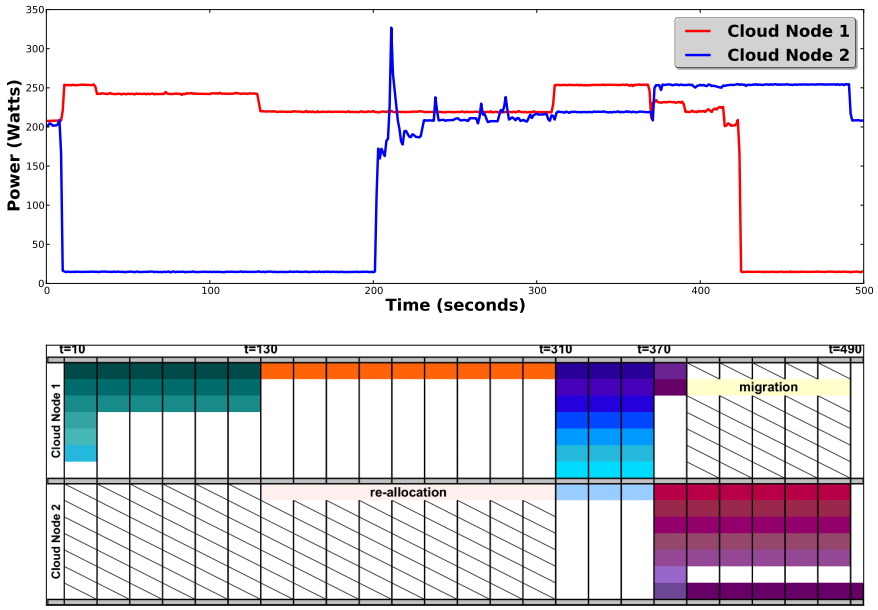

Fig. 12 Green scenario with unbalanced scheduling.

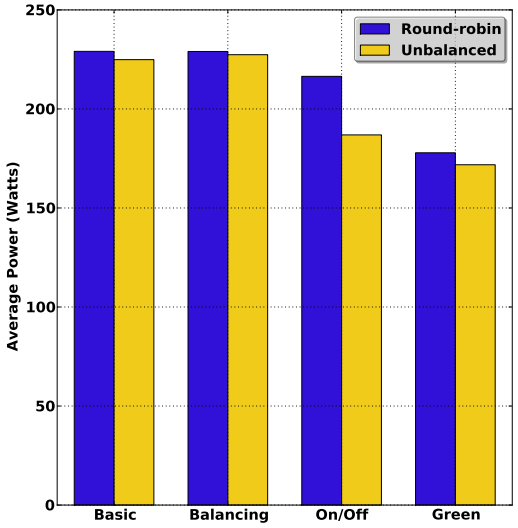

Fig. 13 Comparison results.

for the both scheduling scenarios. The balancing scenario is more energy consuming than the basic scheme for unbalanced scheduling. This phenomenon is due to numerous migrations for the balancing scenario that cancel out the benefits of these migrations.

The noticeable figure is that the green scenario with unbalanced scheduling consumes 25% less electricity than the basic scenario which shows the current Clouds administration. This figure obviously depends on the Cloud usage.

However, it shows that great energy savings are achievable with minor impacts on the utilisation: small delays occur due to migrations (less than 5 seconds for these experiments) and to physical node boots (2 minutes for our experimental platform nodes). These delays can be improved by using better VM live-migration techniques [13], and by using faster booting techniques (like suspend to disk and suspend to RAM techniques).

6 Conclusion

As Clouds are more and more broadly used, the Cloud computing ecosystem becomes a key challenge with strong repercussions. It is therefore urgent to analyse and to encompass all the stakeholders related to the Cloud's energy usage in order to design adapted framework solutions. It is also essential to increase the energy-awareness of users in order to reduce their CO_2 footprint induced by their utilisations of Cloud infrastructures. This matter lead us to measure and to study the electrical cost of basic operations concerning the VM management in Clouds, such as the boot, the halt, the inactivity and the launching of a sample cpuburn application. These observations show that VMs do not consume energy when they are idle and that booting and halting VMs produce small power peaks, but are not really costly in terms of time and energy.

As Clouds are on the way to becoming an essential element of future Internet, new technologies have emerged for increasing their capacity to provide on-demand XaaS (everything as a service) resources in a pas-as-you-use fashion. Among these new technologies, live migration seems to be a really promising approach allowing on-demand resource provisioning and consolidation. We have studied the contribution that the use of this technique could constitute in terms of energy efficiency. VM live migration, although its performance can be improved, is reliable, fast, and requires little energy compared to the amount it can save.

These first measurements have allowed us to propose an original software framework: the Green Open Cloud (GOC) which aims at controlling and optimising the energy consumed by the overall Cloud infrastructure. This energy management is based on different techniques:

- energy monitoring of the Cloud resources with energy sensor for taking efficient management decisions;
- displaying the energy logs on the Cloud portal to increase the energy-awareness;
- switching off unused resources to save energy;

- utilisation of a proxy to carry out network presence of switched-off resources and thus provide inter-operability with any kind of RMS;
- use of VM migration to consolidate the load in fewer resources;
- prediction of the next resource usage to ensure responsiveness;
- giving green advice to users in order to aggregate resources' reservations.

The GOC framework embeds features that, in our opinion, will be part of the next-generation Clouds, such as advance reservations, which enable a more flexible and planned resource management; and VM live-migration. which allows for great workload consolidation.

Further, the GOC framework was the subject of an experimental validation. The validation is made with two different scheduling algorithms to prove GOC's adaptivity to any kind of Resource Management System (RMS). Four scenarios are studied to compare the GOC behaviour with basic Cloud's RMS workings. Energy measurements on these scenarios show that GOC can save up to 25% of the energy consumed by a basic Cloud resource management. These energy savings are also reflected in the operational costs of the Cloud infrastructures.

This promising work shows that important energy savings and thus CO_2 footprint reductions are achievable in future Clouds at the cost of minor performance degradations. There is still room for improvement to increase the user's energy-awareness which is the main leverage to trigger a real involvement from the Cloud providers and designers in order to reduce the electricity demand of Clouds and contribute to a more sustainable ICT industry.

References

1. Citrix xenserver. URL `http://citrix.com/English/ps2/products/product.asp?contentID=683148`
2. Make it green - cloud computing and its contribution to climate change. Greenpeace international (2010)
3. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the art of virtualization. In: 19th ACM Symposium on Operating Systems Principles (SOSP '03), pp. 164–177. ACM Press, New York, USA (2003). DOI http://doi.acm.org/10.1145/945445.945462
4. Box, G.E.P., Jenkins, G.M., Reinsel, G.C.: Time Series Analysis: Forecasting and Control, 3rd edn. Prentice-Hall International, Inc. (1994)
5. Chase, J.S., Anderson, D.C., Thakar, P.N., Vahdat, A.M., Doyle, R.P.: Managing energy and server resources in hosting centers. In: 18th ACM Symposium on Operating Systems Principles (SOSP '01), pp. 103–116. ACM Press, Banff, Alberta, Canada (2001)
6. Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A.: Live migration of virtual machines. In: NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation, pp. 273–286. Berkeley, CA, USA (2005)
7. Da-Costa, G., Gelas, J.P., Georgiou, Y., Lefèvre, L., Orgerie, A.C., Pierson, J.M., Richard, O., Sharma, K.: The green-net framework: Energy efficiency in large scale distributed systems. In: HPPAC 2009: High Performance Power Aware Computing Workshop in conjunction with IPDPS 2009. Roma, Italy (2009)

8. Doyle, R.P., Chase, J.S., Asad, O.M., Jin, W., Vahdat, A.M.: Model-based resource provisioning in a Web service utility. In: 4th Conference on USENIX Symposium on Internet Technologies and Systems (USITS'03), pp. 5–5. USENIX Association, Berkeley, CA, USA (2003)
9. Fan, X., Weber, W.D., Barroso, L.A.: Power provisioning for a warehouse-sized computer. In: ISCA '07: Proceedings of the 34th annual international symposium on Computer architecture, pp. 13–23. New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1250662.1250665
10. Fontán, J., Vázquez, T., Gonzalez, L., Montero, R.S., Llorente, I.M.: OpenNEbula: The open source virtual machine manager for cluster computing. In: Open Source Grid and Cluster Software Conference – Book of Abstracts. San Francisco, USA (2008)
11. Hamm, S.: With Sun, IBM Aims for Cloud Computing Heights (26 March 2009). URL http://www.businessweek.com/magazine/content/09_14/b4125034196164.htm?chan=magazine+channel_news
12. Harizopoulos, S., Shah, M.A., Meza, J., Ranganathan, P.: Energy efficiency: The new holy grail of data management systems research. In: Fourth Biennial Conference on Innovative Data Systems Research (CIDR) (2009). URL http://www-db.cs.wisc.edu/cidr/cidr2009/Paper_112.pdf
13. Hirofuchi, T., Nakada, H., Ogawa, H., Itoh, S., Sekiguchi, S.: A live storage migration mechanism over wan and its performance evaluation. In: VTDC '09: Proceedings of the 3rd international workshop on Virtualization technologies in distributed computing, pp. 67–74. ACM, New York, NY, USA (2009). DOI http://doi.acm.org/10.1145/1555336.1555348
14. Hotta, Y., Sato, M., Kimura, H., Matsuoka, S., Boku, T., Takahashi, D.: Profile-based optimization of power performance by using dynamic voltage scaling on a pc cluster. IPDPS 2006 (2006). DOI 10.1109/IPDPS.2006.1639597
15. Iosup, A., Dumitrescu, C., Epema, D., Li, H., Wolters, L.: How are real grids used? the analysis of four grid traces and its implications. In: 7th IEEE/ACM International Conference on Grid Computing (2006)
16. Jung, G., Joshi, K.R., Hiltunen, M.A., Schlichting, R.D., Pu, C.: A cost-sensitive adaptation engine for server consolidation of multitier applications. In: 10th ACM/IFIP/USENIX International Conference on Middleware (Middleware 2009), pp. 1–20. Springer-Verlag New York, Inc., New York, NY, USA (2009)
17. Kalman, R.E.: A new approach to linear filtering and prediction problems. Transactions of the ASME – Journal of Basic Engineering **82**(Series D), 35–45 (1960)
18. Kalyvianaki, E., Charalambous, T., Hand, S.: Self-adaptive and self-configured CPU resource provisioning for virtualized servers using Kalman filters. In: 6th International Conference on Autonomic Computing (ICAC 2009), pp. 117–126. ACM, New York, NY, USA (2009). DOI http://doi.acm.org/10.1145/1555228.1555261
19. Kim, M., Noble, B.: Mobile network estimation. In: 7th Annual International Conference on Mobile Computing and Networking (MobiCom 2001), pp. 298–309. ACM, New York, NY, USA (2001). DOI http://doi.acm.org/10.1145/381677.381705
20. Kusic, D., Kephart, J.O., Hanson, J.E., Kandasamy, N., Jiang, G.: Power and performance management of virtualized computing environments via lookahead control. In: 5th International Conference on Autonomic Computing (ICAC 2008), pp. 3–12. IEEE Computer Society, Washington, DC, USA (2008). DOI http://dx.doi.org/10.1109/ICAC.2008.31
21. Lefèvre, L., Orgerie, A.C.: Designing and evaluating an energy efficient cloud. The Journal of SuperComputing **51**(3), 352–373 (2010)
22. Liu, J., Zhao, F., Liu, X., He, W.: Challenges towards elastic power management in Internet data centers. In: 29th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW 2009), pp. 65–72. IEEE Computer Society, Washington, DC, USA (2009). DOI http://dx.doi.org/10.1109/ICDCSW.2009.44
23. Miyoshi, A., Lefurgy, C., Van Hensbergen, E., Rajamony, R., Rajkumar, R.: Critical power slope: understanding the runtime effects of frequency scaling. In: ICS '02: Proceedings of the 16th international conference on Supercomputing, pp. 35–44. ACM, New York, NY, USA (2002). DOI http://doi.acm.org/10.1145/514191.514200

24. Moore, J., Chase, J., Ranganathan, P., Sharma, R.: Making scheduling "cool": Temperature-aware workload placement in data centers. In: USENIX Annual Technical Conference (ATEC 2005), pp. 5–5. USENIX Association, Berkeley, CA, USA (2005)
25. Nathuji, R., Schwan, K.: VirtualPower: Coordinated power management in virtualized enterprise systems. In: 21st ACM SIGOPS Symposium on Operating Systems Principles (SOSP 2007), pp. 265–278. ACM, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1294261.1294287
26. Nurmi, D., Wolski, R., Crzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D.: Eucalyptus: A technical report on an elastic utility computing architecture linking your programs to useful systems. Technical report 2008-10, Department of Computer Science, University of California, Santa Barbara, California, USA (2008)
27. Orgerie, A.C., Lefèvre, L., Gelas, J.P.: Chasing gaps between bursts: Towards energy efficient large scale experimental grids. In: PDCAT 2008: The Ninth International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 381–389. Dunedin, New Zealand (2008)
28. Orgerie, A.C., Lefèvre, L., Gelas, J.P.: Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems. In: 14th IEEE International Conference on Parallel and Distributed Systems (ICPADS), pp. 171–178. Melbourne, Australia (2008)
29. Patterson, M., Costello, D., Grimm, P., Loeffler, M.: Data center TCO: a comparison of high-density and low-density spaces. In: Thermal Challenges in Next Generation Electronic Systems (THERMES 2007) (2007). URL `http://isdlibrary.intel-dispatch.com/isd/114/datacenterTCO_WP.pdf`
30. Sharma, R., Bash, C., Patel, C., Friedrich, R., Chase, J.: Balance of power: Dynamic thermal management for internet data centers. IEEE Internet Computing **9**(1), 42–49 (2005). DOI http://doi.ieeecomputersociety.org/10.1109/MIC.2005.10
31. Silicon Valley Leadership Group: Data Center Energy Forecast. White Paper (2008). URL `svlg.org/campaigns/datacenter/docs/DCEFR_report.pdf`
32. Singh, T., Vara, P.K.: Smart metering the clouds. IEEE International Workshops on Enabling Technologies **0**, 66–71 (2009). DOI http://doi.ieeecomputersociety.org/10.1109/WETICE.2009.49
33. Snowdon, D.C., Ruocco, S., , Heiser, G.: Power Management and Dynamic Voltage Scaling: Myths and Facts. In: Proceedings of the 2005 Workshop on Power Aware Real-time Computing (2005)
34. Srikantaiah, S., Kansal, A., Zhao, F.: Energy aware consolidation for cloud computing. In: Proceedings of HotPower '08 Workshop on Power Aware Computing and Systems (2008). URL `http://www.usenix.org/events/hotpower08/tech/full_papers/srikantaiah/srikantaiah/_html/`
35. Subramanyam, S., Smith, R., van den Bogaard, P., Zhang, A.: Deploying Web 2.0 applications on Sun servers and the opensolaris operating system. Sun BluePrints 820-7729-10, Sun Microsystems (2009)
36. Talaber, R., Brey, T., Lamers, L.: Using Virtualization to Improve Data Center Efficiency. Tech. rep., The Green Grid (2009)
37. Tatezono, M., Maruyama, N., Matsuoka, S.: Making wide-area, multi-site MPI feasible using Xen VM. In: Workshop on Frontiers of High Performance Computing and Networking (held with ISPA 2006), *LNCS*, vol. 4331, pp. 387–396. Springer, Berlin/Heidelberg (2006)
38. Travostino, F., Daspit, P., Gommans, L., Jog, C., de Laat, C., Mambretti, J., Monga, I., van Oudenaarde, B., Raghunath, S., Wang, P.Y.: Seamless live migration of virtual machines over the man/wan. Future Gener. Comput. Syst. **22**(8), 901–907 (2006). DOI http://dx.doi.org/10.1016/j.future.2006.03.007
39. Urgaonkar, B., Shenoy, P., Chandra, A., Goyal, P., Wood, T.: Agile dynamic provisioning of multi-tier internet applications. ACM Transactions on Autonomous and Adaptive Systems **3**(1), 1–39 (2008). DOI http://doi.acm.org/10.1145/1342171.1342172
40. Verma, A., Ahuja, P., Neogi, A.: pMapper: Power and migration cost aware application placement in virtualized systems. In: ACM/IFIP/USENIX 9th International Middleware Confer-

ence (Middleware 2008), pp. 243–264. Springer-Verlag, Berlin, Heidelberg (2008). DOI http://dx.doi.org/10.1007/978-3-540-89856-6_13