J Supercomput (2010) 51: 352–373
DOI 10.1007/s11227-010-0414-2

Designing and evaluating an energy efficient Cloud

Laurent Lefèvre · Anne-Cécile Orgerie

Published online: 12 March 2010

Abstract Cloud infrastructures have recently become a center of attention. They can support dynamic operational infrastructures adapted to the requirements of distributed applications. As large-scale distributed systems reach enormous sizes in terms of equipment, the energy consumption issue becomes one of the main challenges for large-scale integration. Like any other large-scale distributed system, Clouds face an increasing demand in energy. In this paper, we explore the energy issue by analyzing how much energy virtualized environments cost. We provide an energy-efficient framework dedicated to Cloud architectures and we validate it through different experimentations on a modern multicore platform. We show on a realistic example that our infrastructure could save 25% of the Cloud nodes' electrical consumption.

Keywords Energy efficiency · Clouds · Virtualization · Migration · Energy awareness · Power management

1 Introduction

Cloud infrastructures have recently become a center of attention. They can support dynamic operational infrastructures adapted to the requirements of distributed applications. As large-scale distributed systems reach enormous sizes in terms of equipment, the energy consumption issue becomes one of the main challenges for large-scale integration. They provide computing power and storage on demand and so they

L. Lefèvre
INRIA RESO, LIP (UMR CNRS, INRIA, ENS, UCB), Université de Lyon, Lyon, France
e-mail: laurent.lefevre@inria.fr

A.-C. Orgerie (✉)
École Normale Supérieure de Lyon, LIP, INRIA RESO, 46 allée d'Italie, 69364 Lyon Cedex 07, France
e-mail: annececile.orgerie@ens-lyon.fr

perfectly respond to users' requirements. Like any other large-scale distributed system, Clouds face an increasing demand in energy. Energy could become the critical resource in the future.

This paper deals with the support of energy-efficient frameworks dedicated to Cloud architectures.[1] It presents the first step of our long-term work whose goal is to better understand the usage of large-scale distributed systems and to propose methods and energy-aware software frameworks able to reduce energy consumption in such systems.

Virtualization is a key feature of the Clouds, as it allows high performance, improved manageability, and fault tolerance. In the first step of our work, our infrastructure focuses its action on this essential aspect. It also uses migration, that brings the benefit of being able to move workload between the virtual machines. We have not yet studied the effects of consolidation and aggregation of virtual machines on the execution time.

In this paper, we propose:

- analyses and experiments of the electric cost of virtual machines;
- an original Cloud infrastructure that aims to save energy without impacting the computing performances;
- an experimental evaluation of our infrastructure showing that we could save 25% of energy compared to a basic Cloud resources management system.

The remainder of this paper is organized as follows. Section 2 reviews related works. This is followed by an evaluation of the electric cost of a virtual machine in Sect. 3. Section 4 outlines the architecture and the components of the energy-aware Cloud infrastructure that we propose. After describing our experimental methodology, we evaluate our infrastructure in Sect. 5. The conclusion and future works are reviewed in Sect. 6.

2 Related works

2.1 Cloud computing

Innovative technologies have broadly contributed to the expansion of Clouds. They differ from Grids as explained in [3] and can be a part of the next-generation data centers with virtualized nodes and provisioning on demand. Clouds are already used by numerous companies. For instance, Salesforce.com handles 54,000 companies and their 1.5 million employees via just 1,000 servers [2]. Different constructors, like IBM [1], also support and provide Clouds infrastructures and services for customer companies.

Cloud computing is dynamically scalable by nature, and virtualized resources are often provided as a service over the Internet [9]. This opens up wide new horizons

[1]This research is supported by the GREEN-NET INRIA Cooperative Research Action: http://www.ens-lyon.fr/LIP/RESO/Projects/GREEN-NET/.

where everything is considered as a service (infrastructure, platform, software, computing, storage). Among other advantages of the Clouds are scalability, cost, and reliability. However, cloud providers, such as Amazon,[2] are likely to face doubts from their customers on security, as well as loss of control over sensitive data. Accounting is another key challenge: they need to stay competitive while remaining economically viable.

2.2 Virtualization and migration

Virtualization is the key feature of the Clouds that allows improving the efficiency of large-scale distributed systems [17]. This technology needs powerful resource management mechanisms [7] to benefit from live migration and suspend/resume mechanisms that allow moving a virtual machine from a host node to another one, as well as stopping the virtual machine and starting it again later. The design of resource-management policies is challenging (NP-hard problem) and dynamic.

Live migration [5] greatly improves the capacities and the features of Cloud environments: it facilitates fault management, load balancing, and low-level system maintenance. Migration operations imply more flexible resource management. When a virtual machine is deployed on a node, we can still move it to another one. It offers a new stage of virtualization by removing the concept of locality in virtualized environments. However, this technique is complex and more difficult to use over MAN/WAN [18] than in a cluster. IP addressing is a problem since the system should change the address of the migrated virtual machine which does not remain in the same network domain. Moreover, it impacts the performances of the virtual machines by adding a non-negligible overhead [19].

2.3 Energy management in Clouds

With virtualization came some ideas regarding energy management [10, 17]. Indeed, large-scale distributed systems are always increasing in size and thus in power consumption. Each node can be virtualized, and can host several virtual machines. So, in the same way, virtualization addresses the limitations in cooling and power delivery. This lead to the design of new energy management techniques in virtualized systems. In [12], the authors propose a management system that uses different power management policies on the virtualized resources of each virtual machine to globally control the power consumption. The emerging concept of consolidation is directly linked with energy management in clouds. The consolidation techniques aim to manage the jobs and combine them on the physical nodes. These techniques can be used to optimize energy usage [16].

[2] http://aws.amazon.com.

3 Energy cost of virtual machines

3.1 Context

Cloud computing seems to be a promising solution to the increasing demand of computing power needed by more and more complex applications. We have seen that virtualization is promoted by several researches to decrease the energy consumption of large-scale distributed systems. However, the studies often lack real values for the electric consumption of virtualized infrastructures.

Our goal is to determine the energy cost of the life of a virtual machine (VM) by measuring the cost when a job is running on it, the cost when the VM does nothing (no job), and the cost of a migration. These measurements will be used for calibrating our energy-aware models. To answer these questions, we have performed some experiments.

Our experimental Cloud consists of HP Proliant 85 G2 Servers (2.2 GHz, 2 dual core CPUs per node) with XenServer 5.0[3] installed on them. Each Cloud node is linked to an external wattmeter that logs the power consumption of the node each second. These energy logs are sent to an energy data collector which stores them. The energy data collector is linked to the Cloud portal, so users can access these logs. The measurement precision of our experimental platform is 0.125 watts, and the maximum frequency is one measure per second, which is precise enough to have a good idea about the consumption of virtual machines.

3.2 The life of a virtual machine

Virtual machine technology is the essential building-block of Cloud infrastructures. VMs are spawned on demand to meet the needs of users, so they can create their own VMs.

Figure 1 shows the boot of a VM from $t = 10$ to $t = 30$, a *cpuburn*[4] running on the VM from $t = 40$ to $t = 100$ and then the shutting down of the VM from $t = 110$ to $t = 122$.

There are two notable aspects. The first one is that the average power consumption from $t = 30$ to $t = 40$ is equal to the idle consumption (P_{idle}) which is the consumption of the node when it does nothing. So, an idle VM (with nothing running on it) does not consume energy. This is a counter-intuitive conclusion: the hypervisor consumes as much energy with 7 idle VMs than without. The second point is that the boot and the shutdown consume really less energy than a *cpuburn*. So, in future experiments and to simplify the lecture of the experimental results, we will not present these periods of boot and shutdown. We will just show the time periods with the jobs. For clarity also, the jobs will all be *cpuburn*. So we are able to compare the power consumption of the different Cloud nodes with jobs running on them.

[3]XenServer is a cloud-proven virtualization platform that delivers the critical features of live migration and centralized multi-server management (http://citrix.com/English/ps2/products/product.asp?contentID=683148).

[4]cpuburn is a software designed to apply a high load to the processor (http://pages.sbcglobal.net/redelm/).

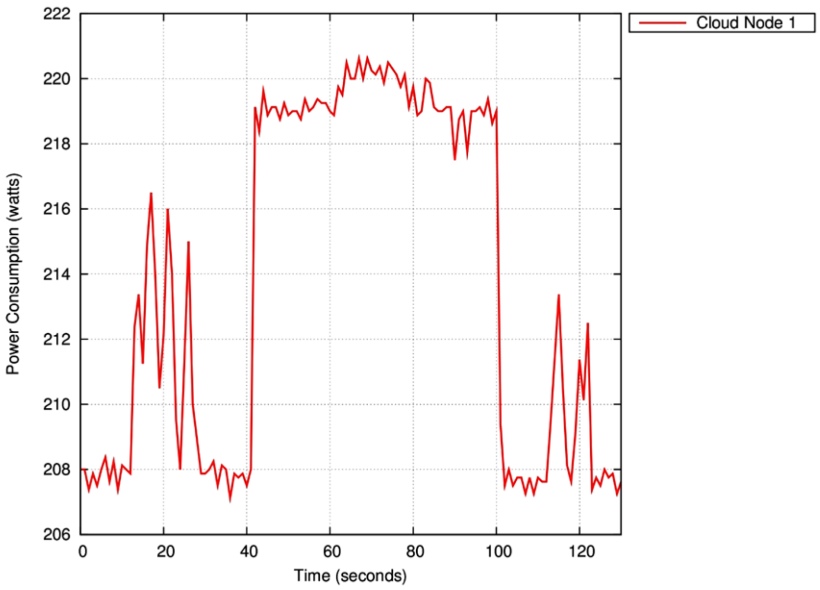


Fig. 1 Life of a virtual machine: boot, run, and halt

3.3 Migration

On Xen, the VM live migration consists of transferring its memory image from the host Cloud node to the new one. If the VM is running, it is stopped for a period of time during the copy of the last memory pages (the ones that are often modified). So when a migration occurs, the end of the job which is in the migrated VM is delayed by a certain amount of time (the period during which the VM is stopped). We denote that time by T_m. This time does not include the whole migration process's duration. Indeed, we have competition at the hypervisor level. If several migrations are required at the same time on the same node, they are queued and processed one by one by the hypervisor (this can also be influenced by the network bandwidth).

In Fig. 2, we have launched one by one six *cpuburn* on six different VMs on Cloud node 1. The first starts at $t = 10$; we see that the consumption increases to 209 watts. Then the second starts and the consumption reaches 230 watts. The third starts, and the node consumes 242 watts. The fourth leads to 253 watts. The apparition of the fifth and the sixth jobs does not increase the consumption. Indeed, as the jobs are CPU intensive (*cpuburn* uses 100% of a CPU capacity) and as there are only four cores on the node (2 dual core CPUs), they are fully used with the first four VMs. The fifth VM appears as "free" in terms of energy cost because it shares already fully used resources. Obviously, these "energy-free" VMs have a cost in terms of performances.

Each *cpuburn* job lasts 300 seconds (Fig. 2). At $t = 110$, we launch the migration of the 6 VMs from Cloud node 1 to Cloud node 2. The migration requires sustained attention from the hypervisor that should copy the memory pages and send them to the new host node. So it cannot handle 6 migrations at the same time, they are done one by one.

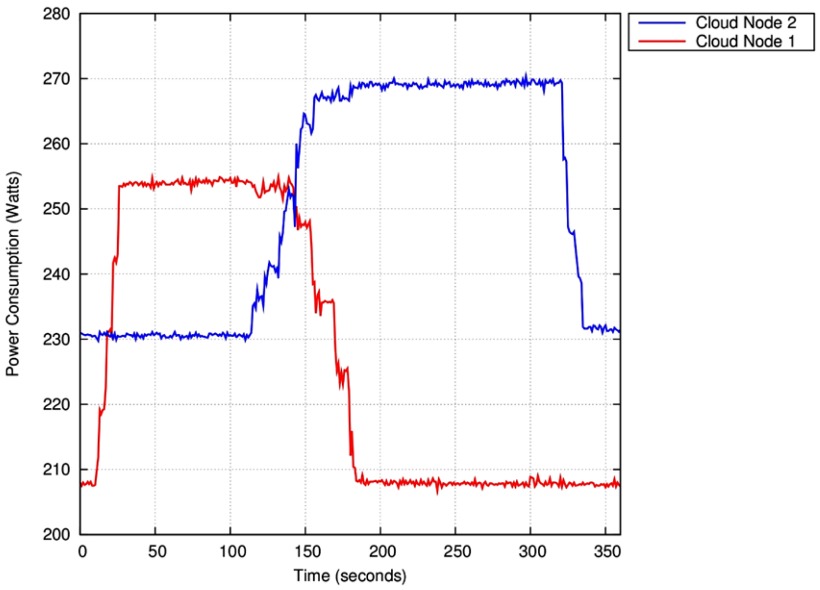


Fig. 2 Migration of virtual machines

The competition occurs, and we see with the power consumption of Cloud node 2 that the VMs arrived one by one. The consumption of Cloud node 1 begins to decrease during the migration of the third VM. At that time, only three VMs keep running on the node. Each job ends 5 seconds late, this is T_m. The competition that occurs during the migration request does not affect more the jobs running on the last migrated VMs since they are still running during their wait for a migration.

4 Green Open Clouds

Some previous works on operational large-scale systems show that they are not utilized at their full capacity [11]. Resources (computing, storage and network) are not used in a constant way by applications and users of large-scale distributed systems (e.g., clusters, grids, clouds). Some inactivity periods can be observed, monitored and predicted. During these periods, some energy-aware frameworks can reduce energy consumption at a global level. We focus on the usage and the energy analysis of an experimental platform (i.e., the Grid5000 infrastructure [4]). With a set of hardware energy sensors, we monitor the energy profile of computing and communicating nodes in order to incorporate live energy values in dynamic decision models. We also propose tools, portals and frameworks feed these results, in real-time, back to users and to cloud and grid middleware.

By generalizing the EARI framework (Energy Aware Resource Infrastructure) proposed for energy-efficient experimental Grids [14, 15], we design the Green Open Cloud architecture (GOC)—an energy-aware framework for Clouds (Fig. 3). The described solution supports the "do the same for less" approach, dealing with efficient On/Off models combined with prediction solutions.

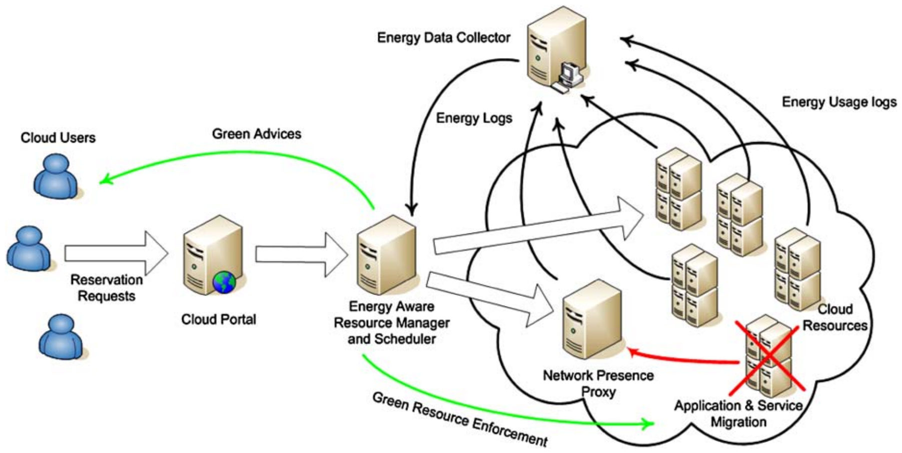


Fig. 3 The Green Open Cloud infrastructure (GOC)

4.1 Green Open Cloud architecture

Virtualization solutions appear as alternative approaches for companies to consolidate their operational services on a physical infrastructure, while preserving specific functionalities inside the Cloud perimeter (security, fault tolerance, reliability, etc.). These consolidation approaches are explored to propose some energy reduction, while switching OFF unused computing nodes. We study the impact of virtual machines aggregation in terms of energy consumption. Some load-balancing strategies associated with migration of virtual machines inside the Cloud infrastructures will be presented.

We will present and describe the GOC architecture which supports the following facilities:

– Switching OFF unused computing, networking and storage resources;
– Predicting computing resources usage in order to switch ON the nodes which are required in a near future;
– Aggregating some reservations to avoid too frequent On/Off cycles;
– Green policies which allow users to specify their requests in terms of energy targets.

The GOC infrastructure is added to the usual resource manager of the Cloud as an overlay. We do not modify the job-scheduling policies, for example, in order to be adaptable to all the resource managers (such as Eucalyptus, for example, [13]). The GOC infrastructure embeds (Fig. 3):

– A set of electrical sensors providing dynamic and precise measurements of energy consumption;
– An energy data collector;
– A trusted proxy for supporting the network presence of switched OFF cloud nodes;
– An energy-aware resource manager and scheduler.

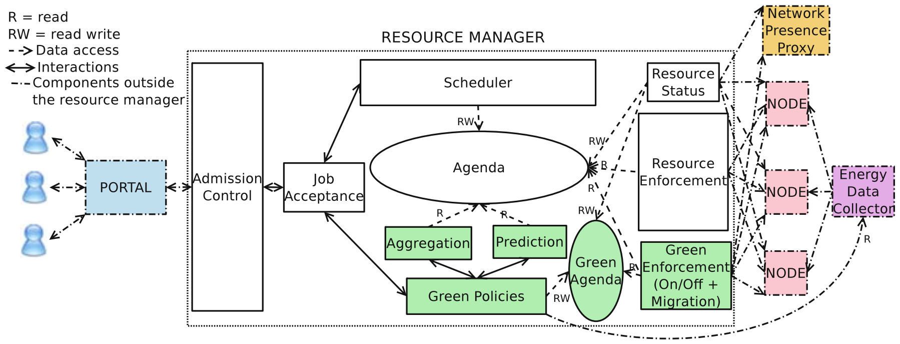


Fig. 4 Architecture of the GOC resource manager

The electrical sensors are connected to each node and they send their consumption measurements to the energy collector. The resource manager has access to this collector and requests the energy logs when needed. It sends them to the Cloud portal in a well-presented manner so that the users can see them and see their impact on the power consumption of the nodes. The Cloud portal is responsible for providing some web services to the users and it is the access point for outside.

The first idea to save energy is to switch off the unused nodes because, as we have seen in Sect. 3, an idle node consumes a lot. We will thus develop our prediction algorithms which aim to switch on the nodes when required. The energy-aware resource manager takes the decisions concerning the shutdown and the boot of the nodes (green resource enforcement). The energy aware resource manager also provides “green” advice to users in order to increase their energy awareness. This consists in proposing several solutions to the user when he submits a job: running it now if possible, running it later and aggregate it with other ones (on the same nodes), or allowing migration decisions for running jobs. These green policies would increase resource sharing and so decrease the energy consumption of the Cloud. If the job is urgent, it can still be run immediately if there are available nodes. On the resource manager architecture (Fig. 4), the white boxes are the usual components of a resource manager, the green ones are GOC features. GOC is designed as an overlay on top of the existing resource manager utilities.

4.2 Network presence

As we switch off the unused nodes, they do not answer to the Cloud resource manager. So they can be considered as dead (not usable). This is a problem that we solve by using a trusted proxy. When we switch off a Cloud node, we migrate its basic services (such as ping or heartbeat services, for example) on this proxy and it will answer for 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 [6].

4.3 Queue and predictions

The Cloud portal transmits the user's jobs to the resource manager which schedules them on the different Cloud nodes. But it treats the users requests one by one. So the requests are first queued. A key feature of our infrastructure consists in switching off the unused nodes. But we need to boot them before a burst of jobs. Otherwise, if there are no more available host nodes, we should switch on the sleeping nodes when the requests arrive and we will delay them by the booting time.

Our prediction algorithms are based on the average values of the last submissions' inter-arrival times. When we have idle nodes (because they have finished their jobs or because we have migrated their VMs to free them), we need to know if we can switch them off. So we have defined a period of time denoted T_s. This time is such that the node consumes as much power when it is idle as when we switch it off and switch it on again during that time. So T_s is defined by:

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

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

When a node is free, we predict the next job, and if this job occurs in less than T_s seconds and if this node is required for this predicted job, we leave it on. Otherwise we switch it off. Our prediction is computed as follows: it 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 few previous predictions. The error is the difference between the true value and the predicted one.

We have seen in [14] 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 a lot of disk accesses and is really fast to compute, which are crucial features for real-time infrastructures.

4.4 Comparison between GOC and EARI

The GOC infrastructure is an adaptation of our Energy-Aware Reservation Infrastructure [15] that deals with Grids environments. This infrastructure is quite different from our previous work. Indeed, in EARI we only handle the reservations. Each user that wants to submit a job should specify its length in time, its size in number of nodes, and a wanted start time. With GOC, we just need the size in terms of number of VMs and there is no reservation, no agenda. So it greatly modifies the specifications of the elementary entity: the job in our case, the reservation for EARI. It implies different management algorithms for the jobs (to act live and not in the future) and different prediction algorithms since we want to predict the next submission and not the next reservation (in EARI, the user does a submission for a reservation which is in the future).

In both infrastructures, we guarantee the performance. We do not impact on the resources wanted by the user nor on the end time if the user does not agree. We have validated EARI by using real usage traces of an experimental Grid. It is not possible with GOC since we do not have access to Cloud usage logs, so the GOC infrastructure is validated with some usage scenarios.

5 Experimental validation

5.1 Experimental methodology

Our Cloud platform consists of HP Proliant 85 G2 Servers (2.2 GHz, 2 dual core CPUs per node). XenServer 5.0 is installed on each node. In the following experiments, we focus on the energy consumption of only two Cloud nodes for clarity reasons. However, as we will see, our infrastructure is scalable. As all the decisions regarding the starting and shutting down of Cloud nodes are taken by the energy data collector (and sent to the resource manager to apply them), the resource manager is not overloaded by these decisions. It just requires a few simple commands to send to the cloud nodes. It does not require any computation. There is no limitation to the scalability of our infrastructure.

We have taken a simple example to illustrate and analyze the working of our infrastructure. 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 for a user to launch as soon as possible. When a user submits a reservation, he predicts the length in time and the number of resources required. This reservation mechanism is an unusual scenario. We place this research in the context of next generation clouds where frameworks should be able to help cloud providers to avoid over-provisioning of resources. Having scheduling reservations with bounds will help clouds’ providers better manage their platforms. This type of workload can already be used for reservations with budget (money) constraints which bounds the user possibility of resource reservation (and thus the reservation length), service clouds with never ending applications where some adaptations and resource switching off are scheduled (maintenance, cost changes in electricity with night/day changes, etc.).

Our job arrival scenario is as follows:

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

Our reservations are short, in order to keep the experiment and the graph representation readable. 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 energy consuming than a *cpuburn* (see Sect. 3.2). So we will just represent the *cpuburn* in order to reduce the length of the graphs and for a better readability.

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, we then attribute its jobs to the awake nodes if they can afford it; otherwise we switch it on.

Thus, in order to validate our infrastructure, we have studied two different schedulings:

- *Round-robin*: first job to the first Cloud node, second job to the second one, and so on. When all the nodes are idle, we change the order of the nodes (we do not always attribute the first job in the queue to the first node).
- *Unbalanced*: we put all the jobs we can on the first Cloud node and if we still have jobs in the queue, we use the second node and so on (as before, when all the nodes are idle, we change the order to balance the roles).

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

- *Basic*: we do not change anything;
- *Balancing*: we use migration to balance the load between the Cloud nodes;
- *On/off*: we switch off the unused nodes;
- *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.

For each scheduling, we have run the four scenarios on our experimental platform and we have logged the energy consumption. We will analyze these logs in the two following subsections.

5.2 Round-robin scheduling

5.2.1 Basic scenario

The basic scenario depicts the way VMs are managed when there is no green or load-balancing policy. It is the easiest possible management. Figure 5 shows the Gantt chart of such a scenario with the round-robin scheduling of the jobs.

Figure 6 presents the power consumption (in watts) for the two Cloud nodes. We see that a VM with a *cpuburn* inside costs about 10 watts; this represents about 5% of the idle consumption. We observe that the fifth VM costs nothing because Cloud node 2 consumes as much power at $t = 360$ with 4 VMs as at $t = 380$ with 5 VMs.

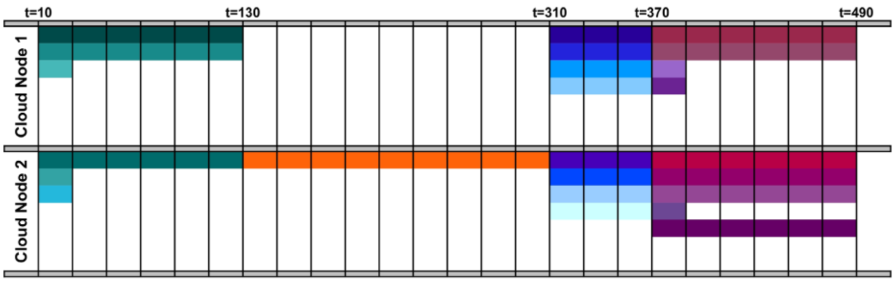


Fig. 5 Gantt chart for the basic scenario with round-robin scheduling

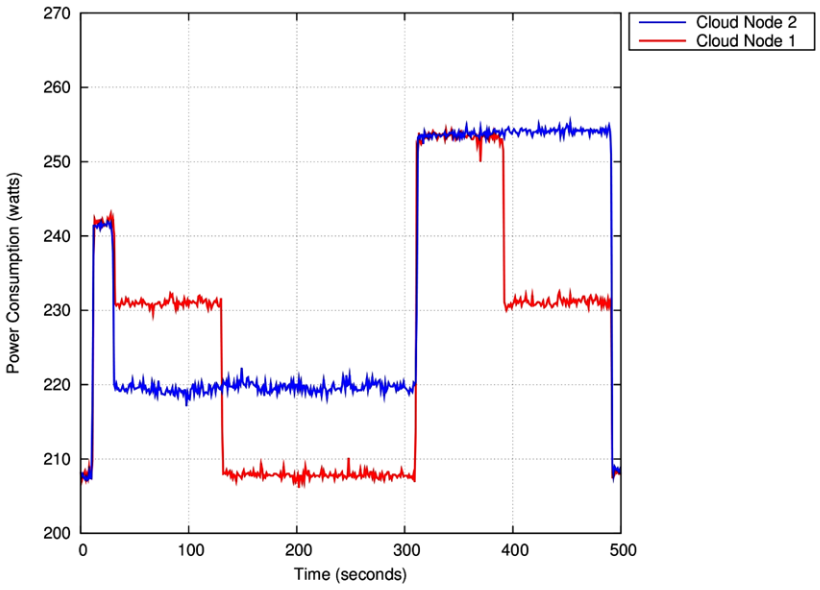


Fig. 6 Power consumption for the basic scenario with round-robin scheduling

5.2.2 Balancing scenario

This scenario is also widely used in resource management. We try to balance the load between the Cloud nodes by using migration if necessary. This helps to preserve the nodes from overheating and to prevent overused nodes to have technical problems prematurely since the other ones are not used. It is linked to the principle that it is better to have a working node than a running node which is doing nothing.

We see in Fig. 7 that this scenario implies a migration of the last scheduled job to have three jobs at each Cloud node. As migration takes time, we do not migrate small jobs (with a duration lower than 20 seconds in that example). This time bound is denoted T_a. We also wait T_a seconds after the beginning of a job before migrating it. It indeed allows initializing the job and the VM, so the migration is simplified, and we avoid the migration of small jobs or the migration of crashed jobs or VMs

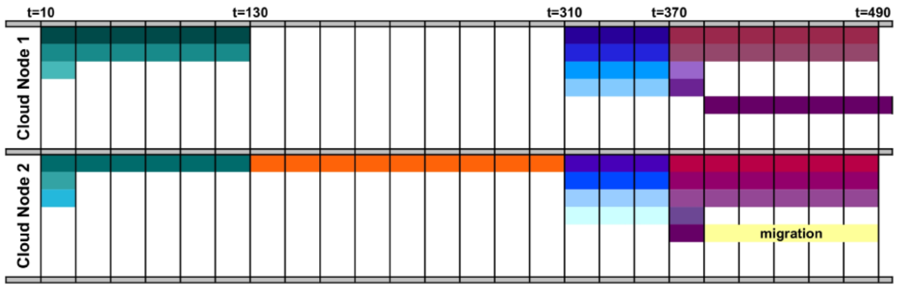


Fig. 7 Gantt chart for the balancing scenario with round-robin scheduling

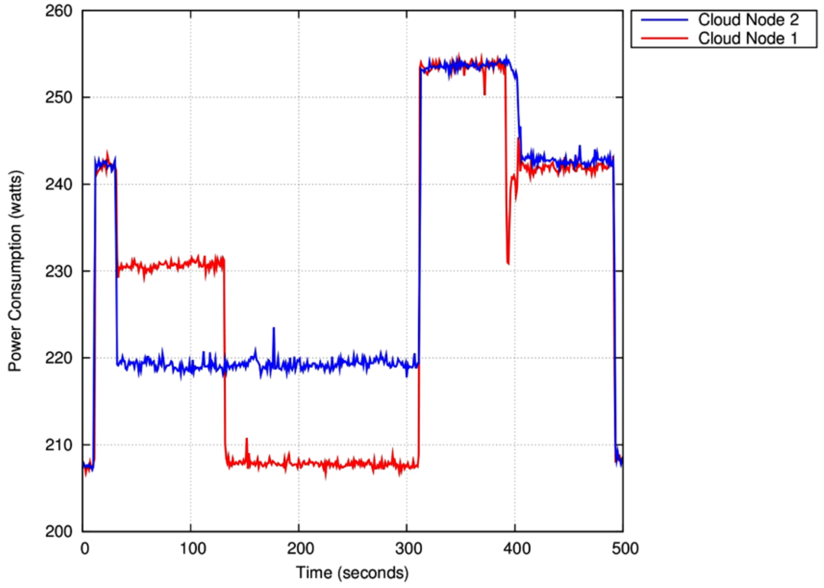


Fig. 8 Power consumption for the balancing scenario with round-robin scheduling

(in the case of a technical failure or bad user configuration of its VM). The energy profile of this scenario (see Fig. 8) points that it takes 5 seconds to migrate one VM from one Cloud node to the other. Actually, it takes a little bit more (we should add the processing time of the migration request), but the job on the VM is interrupted during only 5 seconds, this is T_m. So the last job ends 5 seconds later.

Even if it is not usual now in cloud infrastructures to support job migration, this operation seems not too expensive for computing applications with large durations. For an application with lots of disk accesses and high network usage, this migration time would be greater and thus cost more. Indeed, during the migration process, the two nodes consume energy for this. We plan to do experiments with such applications in our future works.

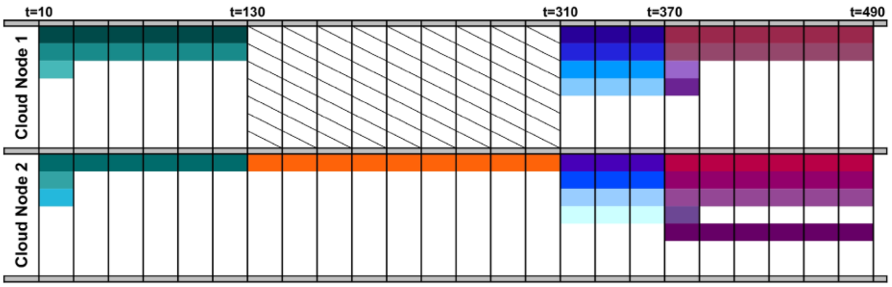


Fig. 9 Gantt chart for the on/off scenario with round-robin scheduling

5.2.3 On/Off scenario

This scenario (Fig. 9) comes from a simple on/off algorithm: the unused nodes are switched off and we predict the next use to switch them on again. In Fig. 9, we can observe that Cloud Node 1 is unused between $t = 130$ and $t = 310$ seconds. As we do not have access to the Cloud provider's logs, it is hard to validate our prediction algorithms. Thus in this experiment, we have assumed that we have achieved a good job prediction and that Cloud node 1 has been booted just in time for its next job (Fig. 10, the boot takes 110 seconds). We can see an evaluation of our prediction algorithms in a Grid context with real Grid traces in [14]. Our algorithms achieve 70% of good predictions on average. During the boot, we see an impressive consumption peak that corresponds to the physical start of all the fans and the ignition of all the node components. However, the on/off algorithm is more energy efficient since an idle node has a really high power consumption (around 210 watts) and the off consumption is low (around 25 watts). To save energy, the idle time should be greater than a certain time that we have called T_s. Here the idle time is greater than T_s, so we can switch off the node. It takes 10 second to shut down Cloud node 1, so T_s is greater than 120 seconds (the time to shut down the node and to switch it on again).

As we can see from the graph (Fig. 10), this scenario consumes less energy than the two previous. We could also increase dynamicity and reduce energy usage by using suspend (to disk or to ram) and resume operations instead of a full switch OFF and boot operations. This would also lower T_s.

5.2.4 Green scenario

Our infrastructure, described in Fig. 4, includes on/off mechanisms, prediction algorithms, and migration to enforce virtual machine aggregation on some nodes to increase the number of idle nodes that will be switched off, the scenario is available on Fig. 11. We notice that the job starting at $t = 130$ on Cloud node 2 is reallocated on Cloud node 1 since Cloud node 2 is switched off and Cloud node 1 is free. Moreover, the third job starting at $t = 10$ is migrated from Cloud node 2 to Cloud node 1 since it allows Cloud node 2 to shut down. The same context appears with two jobs starting at $t = 370$ on Cloud node 1; they are thus migrated to Cloud node 2, and Cloud node 1 is switched off.

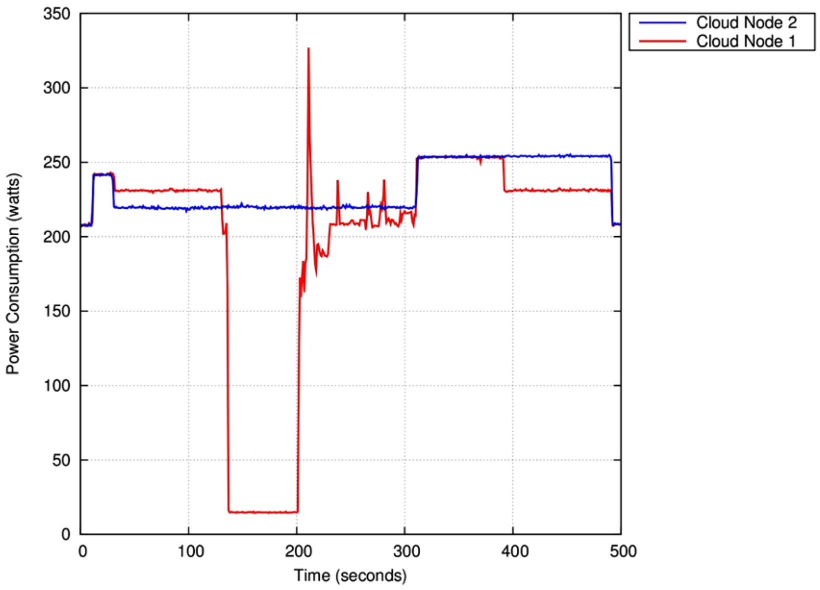


Fig. 10 Power consumption for the on/off scenario with round-robin scheduling

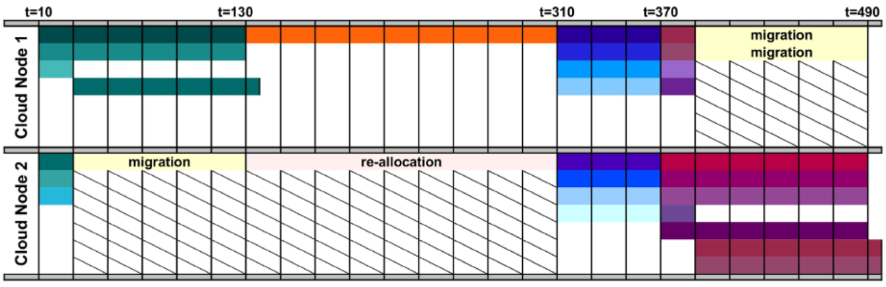


Fig. 11 Gantt chart for the green scenario with round-robin scheduling

The power consumption of the two Cloud nodes during this experiment is presented in Fig. 12. Compared to the previous scenario's consumption, this one entails larger off-periods, so the consumption is lower even with the cost of three migrations. We notice the small consumption peaks due to migrations cost around $t = 35$ and $t = 395$.

As a conclusion for the round-robin scheduling, it appears that the green scenario is, as expected, the less consuming in terms of Cloud nodes power consumption. These results do not include the network costs of the migration, for example, but they seem negligible compared to a Cloud node's consumption [8].

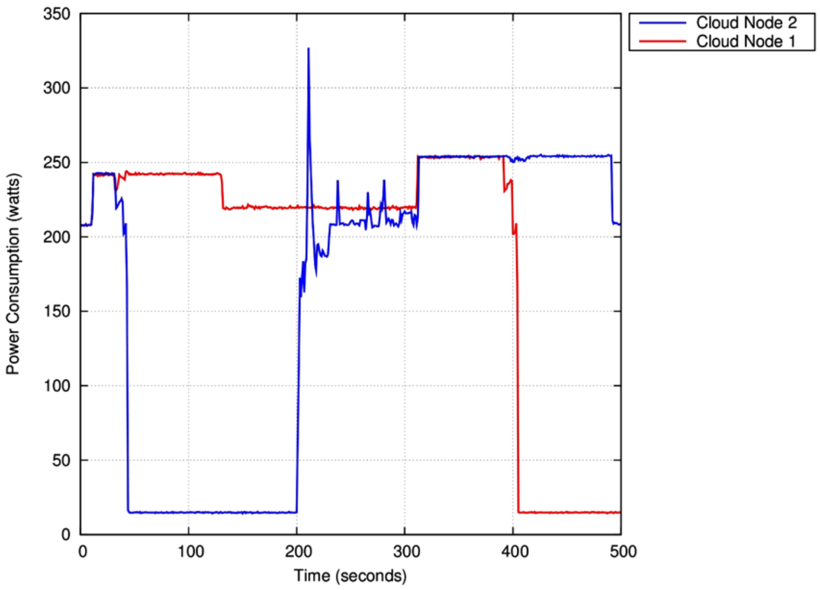


Fig. 12 Power consumption for the green scenario with round-robin scheduling

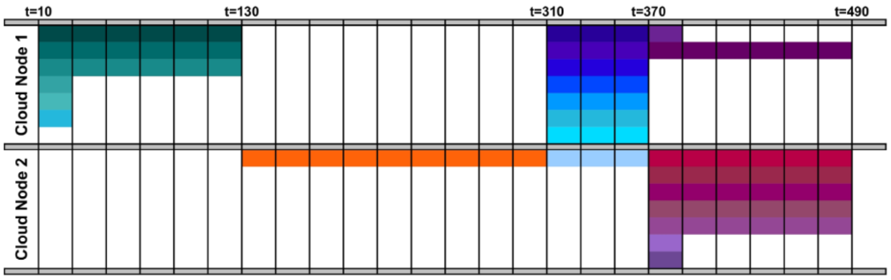


Fig. 13 Gantt chart for the basic scenario with unbalanced scheduling

5.3 Unbalanced scheduling

5.3.1 Basic scenario

At first glance, the unbalanced scheduling seems more appropriate to save energy, as we group the jobs on the smallest number of Cloud nodes. However, this scheduling does not take into account the size of the jobs. Thus, after the end of the small jobs, the nodes are used by only a few jobs which could have been aggregated on the same node. The basic scenario is depicted in Fig. 13. We see that the first burst of jobs is allocated to Cloud node 1 at $t = 10$, and the second burst which occurs at $t = 130$ is allocated to Cloud node 2; we alternate the first node to use.

We see on the consumption logs for this scenario (Fig. 14) that as previously, from 4 to 7 VMs on a Cloud node, the node consumption is the same. The consumption

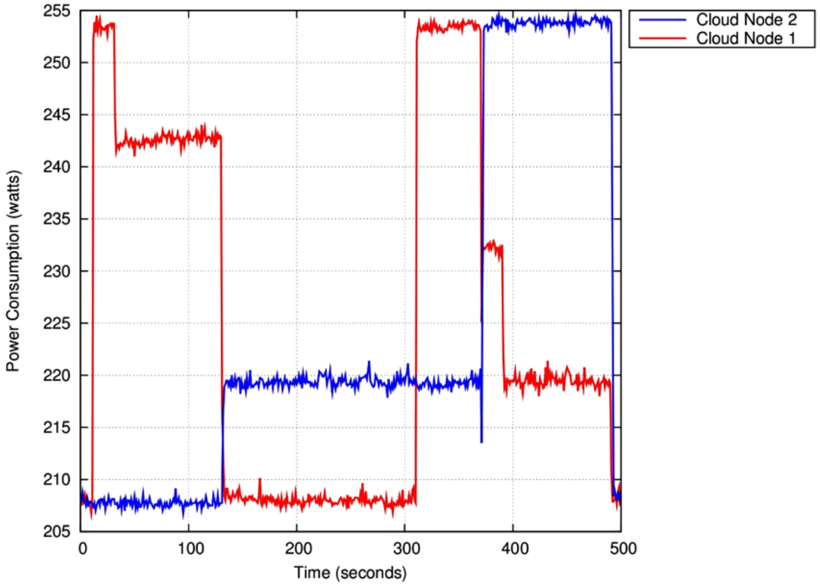


Fig. 14 Power consumption for the basic scenario with unbalanced scheduling

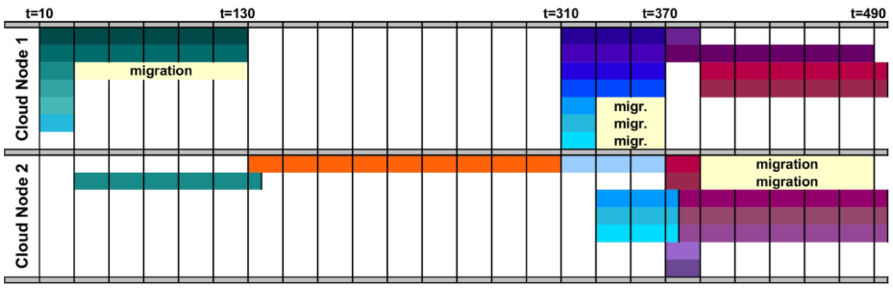


Fig. 15 Gantt chart for the balancing scenario with unbalanced scheduling

levels are also really noticeable.These levels correspond to VMs that use their virtual CPU intensively.

5.3.2 Balancing scenario

The balancing scenario (Fig. 15) does not perform well with unbalanced scheduling since those two approaches are opposite. It leads to a great number of VM migrations.

We see the result of these numerous migrations in Fig. 16—more energy is consumed. Moreover, it seems that the Xen hypervisor does not handle simultaneous migrations very well. The migrations are queued (the VMs are still running) and done one by one as suggested by the sawtooth curve of Cloud node 2 after $t = 330$. During that time, Cloud node 1 still consumes a lot of energy.

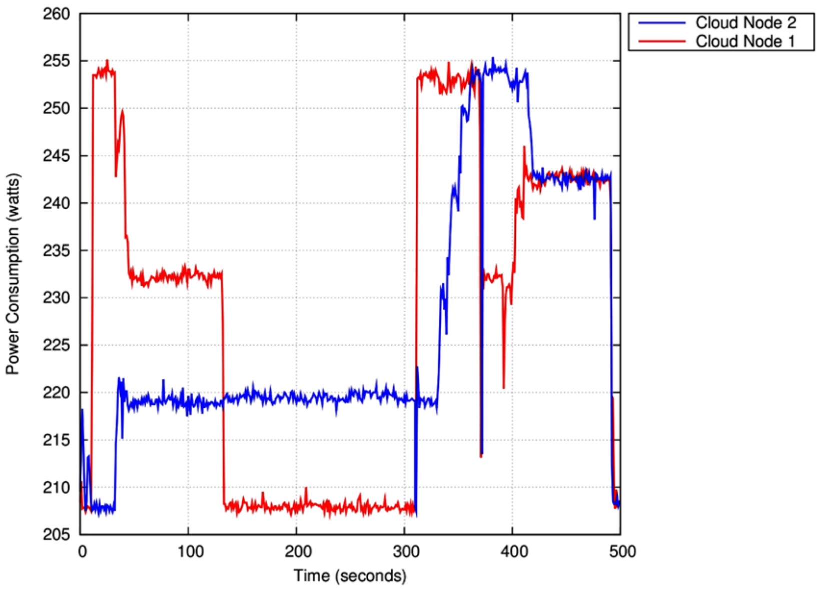


Fig. 16 Power consumption for the balancing scenario with unbalanced scheduling

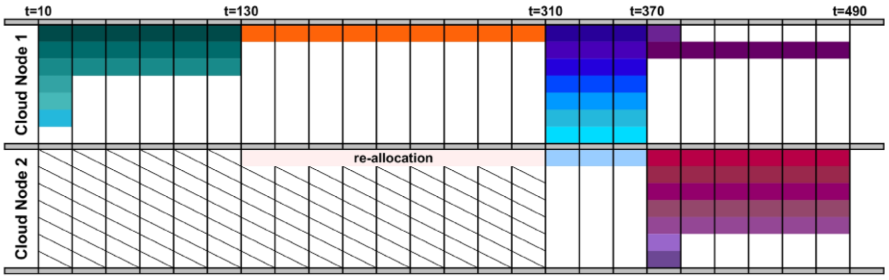


Fig. 17 Gantt chart for the on/off scenario with unbalanced scheduling

5.3.3 On/Off scenario

The on/off scenario (Fig. 17) is more energy efficient with unbalanced scheduling than with round-robin scheduling since this scheduling promotes the idleness on the greatest possible number of Cloud nodes. The job starting at $t = 130$ is reallocated to Cloud node 1 because Cloud node 2 is off.

As before, we switch on Cloud node 2 at $t = 200$ since it takes 110 seconds to wake up. The power consumption graph is shown in Fig. 18.

5.3.4 Green scenario

The green scenario (Fig. 19) is also more energy efficient with unbalanced scheduling for the same reasons. One migration is necessary for the last job to shut down Cloud node 1.

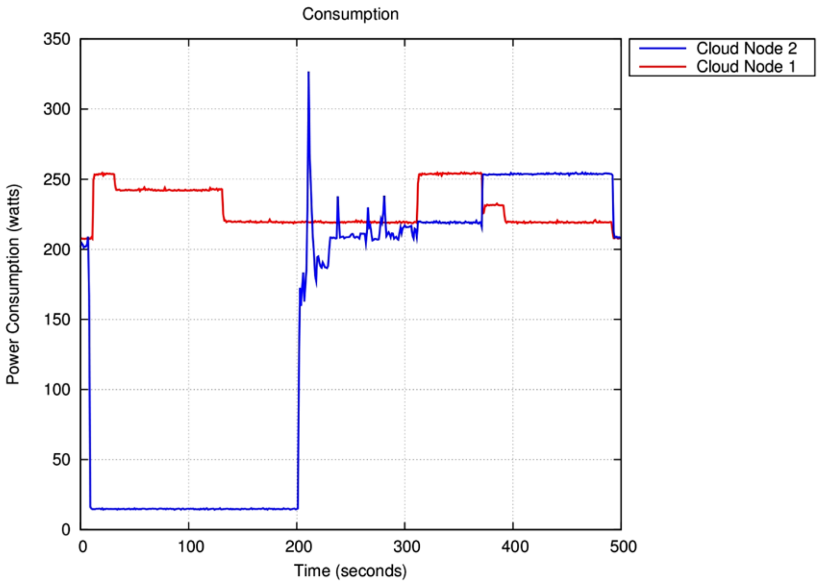


Fig. 18 Power consumption for the on/off scenario with unbalanced scheduling

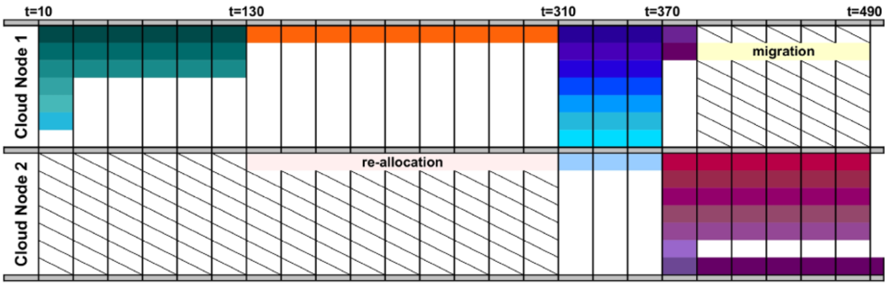


Fig. 19 Gantt chart for the green scenario with unbalanced scheduling

Figure 20 depicts the power consumption for this scenario. We notice the small consumption peak on Cloud node 1 for the migration of the last job. Like with the previous scheduling and as expected, the green scenario is the least consuming.

5.4 Comparison between the scenarios

To make a comparison between the four scenarios, we have computed the average power consumption per node (average value between Cloud node 1 and Cloud node 2) for each case. The results are shown in Fig. 21.

We clearly observe that the green scenario is the least consuming of the two schedulings. The green scenario with unbalanced scheduling consumes 25% less energy than the basic scenario with round-robin scheduling. We notice that the balancing scenario is more consuming than the basic one for unbalanced scheduling

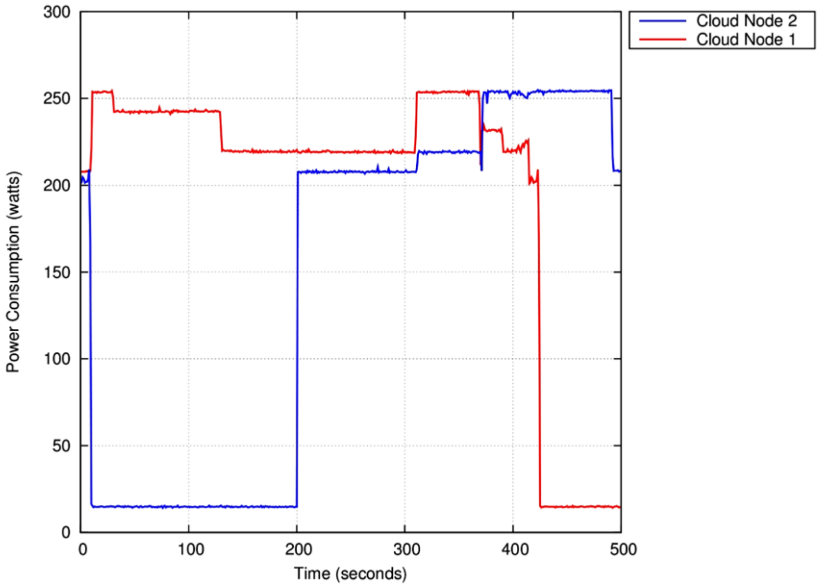
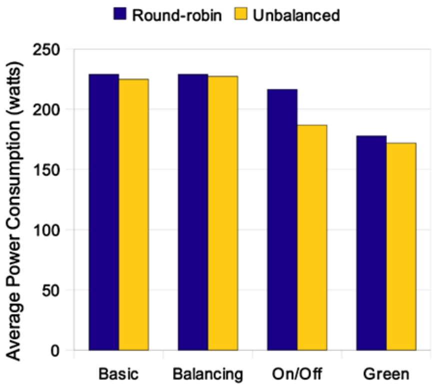


Fig. 20 Power consumption for the green scenario with unbalanced scheduling

Fig. 21 Comparison between the average power consumptions per node for the different schedulings and scenarios



due to numerous migrations as explained before. For all the scenarios, unbalanced scheduling is the least consuming even for the scenarios that do not take into account energy-aware considerations, such as the basic scenario and the balancing one. This clearly indicates that a load-balancing strategy is more energy consuming than an unbalancing one. This idea seems counter-intuitive at the first glance.

6 Conclusion and future works

This paper presents our first step in designing a Green Open Cloud architecture by taking into account the energy usage of virtualization frameworks. Based on the EARI model, we have experimentally measured and analyzed the electric cost of basic operations associated with virtual machines (boot, usage, migration, etc.). We have proposed original software frameworks able to reduce the energy usage of cloud infrastructure. These frameworks embed load-balancing solutions with migration facilities and an on/off infrastructure associated with prediction mechanisms.

In our experimental validation, the jobs are not delayed for more than 5 seconds due to migration operation (which is the value of T_m as defined in Sect. 3.3). When two migrations are required at the same time on the same node, the hypervisor lets the VM running and does the migrations one by one. Therefore, if we make migrations in order to shut down a node, we should wait longer depending on the number of VMs to migrate and so consume more. It follows that if we have numerous VMs to migrate with small jobs, it could be more energy efficient to let them end on their original node. This outcome raises questions, and especially this one: What is the optimal power consumption (the lowest bound)?

Next generation Cloud environments could support in-advance reservations. In such Cloud environments, we can propose solutions to support greater energy savings. Indeed, when a user submits a reservation, our GOC infrastructure suggests several possible start times and tries to influence the user to aggregate its reservation with other ones already inscribed in the agenda. We also could have a deadline mechanism, each user giving a deadline when it submits its job. This allows us to start its job with other ones, then freeze it and later finish it with other jobs, respecting the deadlines. This scenario could be more energy efficient than the green scenario presented previously. This leaves several tracks to explore in order to save energy. Pricing and accounting in Clouds are among other tracks that can be explored to reduce overall consumption. A model based on electricity costs with slack periods and low prices for flexible clients could be used. It would also have the advantage of increasing the energy-awareness among the clients.

As future works, we plan to study the impact of consolidation and aggregation of virtual machines on the execution time of real applications. These impacts are induced by the competition on the resources. We are currently experimenting and simulating our green strategies with open-source clouds frameworks (based on Eucalyptus framework [13]) deployed in large-scale distributed systems (i.e., the Grid'5000 platform [4]) in order to validate the proposed approach on large-scale infrastructures.

References

1. Boss G, Malladi P, Quan D, Legregni L, Hall H (2007) Cloud computing. Technical report, IBM, 8 October
2. Business Week (2009) With Sun, IBM Aims for Cloud Computing Heights, 26 March
3. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener Comput Syst 25(6):599–616
4. Cappello F et al. (2005) Grid'5000: A large scale, reconfigurable, controllable and monitorable grid platform. In: 6th IEEE/ACM international workshop on grid computing, Grid'2005, Seattle, Washington, USA, November 2005
5. Clark C, Fraser K, Hand S, Hansen JG, Jul E, Limpach C, Pratt I, Warfield A (2005) Live migration of virtual machines. In: NSDI'05: proceedings of the 2nd conference on symposium on networked systems design & implementation, Berkeley, CA, USA, pp 273–286

6. Da-Costa G, Gelas J-P, Georgiou Y, Lefèvre L, Orgerie A-C, Pierson J-M, Richard O, Sharma K (2009) 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, May 2009
7. Grit L, Irwin D, Yumerefendi A, Chase J (2006) Virtual machine hosting for networked clusters: Building the foundations for "autonomic" orchestration. In: VTDC '06: proceedings of the 2nd international workshop on virtualization technology in distributed computing, Washington, DC, USA
8. Gupta M, Grover S, Singh S (2004) A feasibility study for power management in LAN switches. In: Proceedings of the 12th IEEE international conference onnetwork protocols, ICNP 2004, 5–8 Oct 2004, pp 361–371
9. Hayes B (2008) Cloud computing. Commun ACM 51(7):9–11
10. Hermenier F, Loriant N, Menaud J-M (2006) Power management in grid computing with xen. In: XEN in HPC cluster and grid computing environments (XHPC06), Sorrento, Italy, 2006. LNCS, vol 4331. Springer, Berlin, pp 407–416
11. Iosup A, Dumitrescu C, Epema D, Li Hui, Wolters L (2006) How are real grids used? The analysis of four grid traces and its implications. In: 7th IEEE/ACM international conference on grid computing, September 2006
12. Nathuji R, Schwan K (2007) Virtualpower: coordinated power management in virtualized enterprise systems. In: SOSP '07: proceedings of twenty-first ACM SIGOPS symposium on operating systems principles, New York, NY, USA, 2007, pp 265–278
13. Nurmi D, Wolski R, Grzegorczyk C, Obertelli G, Soman S, Youseff L, Zagorodnov D (2008) The eucalyptus open-source cloud-computing system. In: Proceedings of cloud computing and its applications, Chicago, Illinois, USA, October 2008
14. Orgerie A-C, Lefèvre L, Gelas J-P (2008) 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, Dunedin, New Zealand, December 2008, pp 381–389
15. Orgerie A-C, Lefèvre L, Gelas J-P (2008) 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), Melbourne, Australia, December 2008, pp 171–178
16. Srikantaiah S, Kansal A, Zhao F (2008) Energy aware consolidation for cloud computing. In: Proceedings of HotPower '08 workshop on power aware computing and systems, December 2008
17. Talaber R, Brey T, Lamers L (2009) Using virtualization to improve data center efficiency. Technical report, The Green Grid
18. Travostino F, Daspit P, Gommans L, Jog C, de Laat C, Mambretti J, Monga I, van Oudenaarde B, Raghunath S, Wang PY (2006) Seamless live migration of virtual machines over the man/wan. Future Gener Comput Syst 22(8):901–907
19. Voorsluys W, Broberg J, Venugopal S, Buyya R (2009) Cost of virtual machine live migration in clouds: a performance evaluation. Technical report, Clouds Laboratory, University of Melbourne, Australia, 5 April