How Much Energy can Green HPC Cloud Users Save?

David Guyon, Anne-Cécile Orgerie, Christine Morin, Deb Agarwal

How Much Energy can Green HPC Cloud Users Save?

David Guyon*, Anne-Cécile Orgerie†, Christine Morin‡ and Deb Agarwal§
*University of Rennes 1, IRISA, France, Email: david.guyon@irisa.fr
†CNRS, IRISA, France, Email: anne-cecile.orgerie@irisa.fr
‡Inria, IRISA, France, Email: christine.morin@inria.fr
§Lawrence Berkeley National Lab. Berkeley, USA , Email: daagarwal@lbl.gov

***Abstract*—Cloud computing has become an attractive and easy-to-use solution for users who want to externalize the run of their applications. However, data centers hosting cloud systems consume enormous amounts of energy. Reducing this consumption becomes an urgent challenge with the rapid growth of cloud utilization. In this paper, we explore a way for energy-aware HPC cloud users to reduce their footprint on cloud infrastructures by reducing the size of the virtual resources they are asking for. We study the influence of green users on the system energy consumption and compare it with the consumption of more aggressive users in terms of resource utilization. We found that larger resources are more energy demanding even if they are faster in executing the applications. But, reducing too much the resources' size is also not beneficial for the energy consumption. A tradeoff lies in between these two options.**

***Keywords*-Cloud computing; green computing; HPC applications**

I. Introduction

High Performance Computing (HPC) infrastructures are usually massive buildings that contain hundreds of servers with powerful hardware [1]. These infrastructures run scientific applications requiring tremendous amounts of computing resources to execute. These applications are often organized in workflow structures and each step of the workflow may be a computation that needs important amounts of CPU, memory and storage resources.

Cloud computing has become a cost effective alternative to HPC machines [2] and some of the less resource intense HPC applications tend to migrate to clouds [3]. Cloud computing offers *elasticity* which allows applications to reduce over- and under-provisioning of resources. A user only pays for the resources its application is using and, in some cases, the application execution has an overall lower pricing compared to HPC solutions [3].

As a consequence of the cloud computing success, the global energy consumed by data centers has increased significantly [4], [5]. Studies show that cloud computing represented about 0.5% of the worldwide energy consumption in 2008 and is predicted to quadruple by 2020. We are currently facing important climate changes which call for a reduction of the ecological impact of computing. To reduce the electrical consumption of cloud infrastructures, *consolidation* mechanisms pack the virtual machines (VMs) on the least number of servers, without impacting application performance, in order to turn off the unused servers in case of moderate load.

Idle servers indeed consume extensive amounts of energy [6]. However, such consolidation techniques are only efficient if virtual resources are not kept idle by the users for no work. Indeed, if the cloud provider does not over-commit the physical resources, the user that uses only partly the virtual machines resources is wasting the rest. Thus, energy-efficient users need to properly size their VMs. For a given parallel application, several VM sizes are possible, each offering a different tradeoff between the overall energy consumption and the performance (i.e. runtime). This tradeoff is complex to determine: small-sized VMs may be easier to pack into server machines, while larger VMs may end their work faster. While it is logical that well-dimensioned machines are more energy efficient, defining their size is not an easy task for the users.

In a previous work [7], we present a cloud system involving users in the energy optimization system. A user who agrees to reduce her impact on the environment can choose a more energy-efficient execution mode, implying a lost in performance, by executing her application on less resources on the infrastructure. The unused resources are free for another application and thus, this approach favors a better consolidation of the whole system. The better the consolidation, the lower the electrical consumption. The proposed system offers three execution modes based on [8]: *Big*, *Medium* and *Little*. An algorithm selects the size of the VMs for executing each task of the workflows depending on the selected execution mode. The *Medium* mode executes using the user-specified VM resources for each workflow stage. The *Little* and *Big* modes respectively decreases or increases the VMs by one size for the whole workflow.

In the present paper, we evaluate the impact of the proportion of users selecting the *Big*, *Medium* or *Little* mode on a data center's energy consumption. Our evaluations have been done using three kinds of scientific workflows, energy consumption measurements for the execution of these workflows on a real platform and traces of jobs submitted to a production HPC center. We evaluated the data center energy consumption for different proportions of users selecting the three available modes. The simulation results show promising energy savings when the amount of users selecting the *Big* mode is low. It also shows that using the *Little* mode compared to the *Medium*

mode does not always provide the best performance/energy saving tradeoff.

The paper is organized as follows. Section II presents our methodology. The experimental setup is explained in Section III and the simulations' results detailed in Section IV. Finally, we conclude the paper in Section V.

II. Methodology

For evaluating the impact of energy-aware users on a HPC cloud, we conducted an experimental study using a real public workload trace from a production data center. A job in our workload is an execution of one of three different scientific applications. The energy consumption of these applications running with all possible execution modes was measured on a real cloud infrastructure. Each job runs with an execution mode and we varied this distribution of the modes in order to have different profiles of user population. The energy consumption of the data center is calculated with each profile distribution in order to evaluate the impact of the execution mode choices on the data center's electrical consumption.

The nodes of the simulated data center are inspired by the nodes of the Taurus cluster of Grid'5000, a French platform for experimenting distributed systems. Each node of this cluster has 12 Intel Xeon E5-2630 CPU cores, 32GB of memory, 598GB of hard drive and a 10 Gigabit Ethernet connection.

For executing the applications on the hardware we used virtualization based on the KVM technology. A VM has a fixed size in terms of CPU, memory and disk resources. We considered different kinds of VMs with different amounts of CPU, memory and disk resources, called *flavors* and we selected 5 flavors similar to those offered by the Amazon EC2 cloud [9]. The list of flavors used in our system is presented in Table I. This table also contains the EC2 instance equivalent and their US East hourly pricing.

TABLE I: Details of the VM flavors used in the system with their Amazon EC2 instance equivalent and their US East hourly pricing.

Flavor	RAM	CPU	Disk	EC2 instance equiv.	
tiny	0.5 GB	1	5 GB	t2.nano	$0.0065
small	2 GB	1	20 GB	t2.small	$0.026
medium	4 GB	2	40 GB	t2.medium	$0.052
large	8 GB	4	80 GB	c4.xlarge	$0.209
xlarge	16 GB	8	160 GB	c4.2xlarge	$0.419

In our cloud system, incoming jobs are executed directly and cannot be batched for a later execution. Any job submission implies a VM creation for each task of the workflow. A consolidation mechanism creates the VMs on specific servers in order to optimize their resource utilization (*Greedy* algorithm). If a server does not host any VM, it is powered down in order to reduce the data center's power consumption.

The workload corresponds to the job submission distribution over a day. We took a 2 year long trace from a real production HPC platform located in the Czech republic [10]. From this trace we analyzed the daily submission distribution and used a k-mean algorithm to find different distribution profiles. From these profiles we retained one with a submission peak during the working hours.

A job submission is a request to start an application. We selected real scientific applications that execute as a workflow. Workflows are composed of a sequence of sequential and/or parallel tasks with data dependencies. We selected 3 applications from different scientific areas that exhibit different behaviors in terms of resource consumption: disk-intensive, CPU-intensive and memory-intensive. The chosen applications are the following ones: *Montage*, *Blast* and *Palmtree*. They are presented in more detail in Section III-C.

Each job runs according to an execution mode. This mode has an impact on the size of the VMs where the job is running and consequently on the execution time of the job. A probabilistic distribution algorithm takes as input the percentage of jobs in each execution mode. As output, the algorithm fairly distributes the 3 workflows to each job and set each job execution mode following the input.

The electrical consumption of the servers has been measured thanks to the fine-grained wattmeters available on them [11]. Three measures have been recorded: when the server is powered down, when the server is on but not used (*idle*) and when it is fully used. The energy consumption of the workflows has also been measured. The execution logs of each workflow in each execution mode contains the run time and the energy consumption of each task. To obtain accurate electrical measures, the servers were loaded at their maximum capacity by duplicating the tasks running on them. Then the dynamic consumption is distributed evenly across the tasks.

A. Assumptions

In this system, we make the following assumptions:

- a user application is a workflow composed of one or more sequential steps, each step having one or more parallel tasks ;
- each task of a workflow executes in a separate VM ;
- each task can exploit all the cores available in its VM, whatever the number of cores. It is the users' responsibility to implement tasks that automatically adapt their execution to use all the cores available ;
- a VM always has enough disk space and memory for the task to execute, even when the *Little* execution mode is selected.

B. System

The system architecture is presented in Fig. 1. The user, at the top, sends a request to execute her workflow application. The request contains the workflow structure (number of steps and parallel tasks) and the amount of CPU, memory and disk space required by default by each step. She also indicates the execution mode for the run of her application.

Inspired by the ARM big.LITTLE (which is a heterogeneous processor) V. Villebonnet et al. introduce in [8] the *Big*, *Medium* and *Little* (BML) infrastructure. Their idea consists in reaching energy proportionality by using heterogeneous

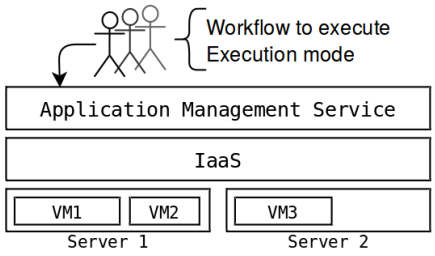

Fig. 1: The users send their applications description, with for each of them the selected execution mode, to the cloud infrastructure which contains the servers hosting the users' VMs.

processors for variable workloads: if the workload is low, it is executed on the Little processor, while when it is high, it smoothly migrates to the Big processor. Similarly, in our system the VMs' sizes for executing a workflow is chosen according to an energy/performance tradeoff depending on the execution mode selected by the user. This is why in the rest of the paper, we opt for the same terminology which is easier to handle and it highlights the main variable of our system: the VM size[1].

The size of the VM for a given task is selected according to the specified amount of CPU, memory and disk space required (Section III-C details how the resource amount is defined). The VM flavor with just enough resources is the one selected for the *Medium* execution mode. The *Big* execution mode selects the VM flavor one size larger and for the *Little* mode, it selects the VM flavor one size smaller. For example, an application asking for an amount of resources matching the *medium* flavor will be assigned the *large* VM flavor in the *Big* execution mode and the *small* VM flavor in the *Little* execution mode.

A VM placement algorithm creates the VMs on specific servers in order to favor the consolidation of the whole system and reduce the global energy consumed. A simple *Greedy* algorithm [12] implementation is used to solve this complex bin packing problem. The servers are sorted in ascending order of available resources and the first one suitable for the VM creation is selected. This algorithm avoids the fragmentation of VMs across servers.

III. Experimental Setup

An evaluation of the energy consumed in our cloud system with different user profile distributions has been conducted using simulation. In order to have a simulator as realistic as possible, we took a job arrival trace from an existing HPC center, we selected real scientific applications that we ran on a cloud infrastructure to get execution logs and finally we designed the simulated infrastructure based on the hardware configuration of the real cluster we used. This is described in more detail in the remainder of this section.

[1]To avoid confusion between the *Medium* mode and the *medium* flavor, the modes always start with a capital letter

A. Simulator

A cloud simulator has been developed in Python. It reproduces the behavior of a cloud system that takes the users into consideration in order to optimize the energy efficiency of the whole system. This simulator takes the following inputs:

- an arrival trace of request submissions (workload) based on real data in order to have a realistic use case ;
- a panel of execution logs of scientific applications measured on a real cloud infrastructure that ran with the three execution modes ;
- profile distribution probability represented by a percentage parameter to configure the amount of applications to execute in the different modes ;
- information about the servers to use in the simulated data center such as the hardware resources (CPU, disk and memory) and the power consumption of a single node in idle and off states based on real measurements on the machines we used to run the scientific applications.

Each job submission in the workload simulates the execution of an application starting at a specific time (date during the day) using the arrival trace and is attributed an execution mode with respect to the profile distribution probability given as parameter.

The output of the simulator is the energy consumed during a whole day by the workload run on the simulated cloud infrastructure. It also generates the complete simulation log details for debugging purpose.

B. Arrival Trace

We used a realistic job submission trace as an input of our simulator. The original trace is 2 years long and comes from the utilization records of the MetaCentrum Czech National Grid [13]. We executed a k-mean algorithm on this archive and retrieved a 24h long trace containing a total of 1506 job submissions. The jobs distribution represents a typical daily use with submission peak during the working hours.

C. Execution Logs

The simulator utilizes execution logs of workflow applications that we ran on a real cloud infrastructure. Three scientific applications from completely different research domains have been carefully selected in order to represent the computations we can find in data centers, such as memory-intensive, data-intensive and CPU-intensive tasks.

1) Montage Workflow: Montage [14] is an engine to build astronomical image mosaics for astronomers. Its workflow structure is composed of 3 parallel tasks that download data and run calculation of it, and then a single ending task that creates the final mosaic thanks to the 3 intermediate data given by the first step. The workflow is mainly IO-intensive and CPU-intensive during the calculation.

The number of hardware resources given to each task has been selected by experimentation in order to have a *Medium* execution that runs for less than an hour. The tasks of the first step need 2 cores, 2 GB of RAM and 10 GB of disk space. The second step requires 1 core, 4 GB of RAM and 20 GB

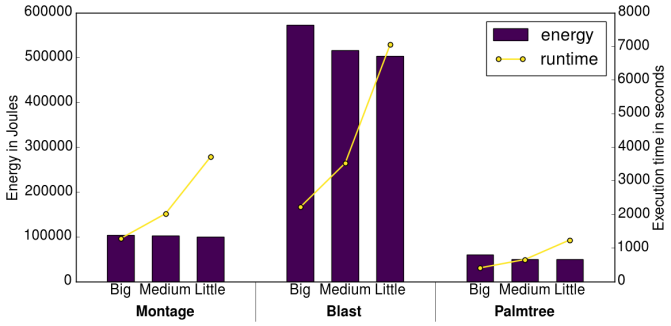

Fig. 2: Energy consumption and execution time of each workflow in each execution mode.

of disk space. Thus, the VM flavor used for these tasks with the *Medium* mode is the *medium* size.

2) Blast Workflow: Blast [15] is a program that compares nucleotide or protein sequences to sequence databases and calculates the statistical significance of matches. The workflow structure is composed of 4 parallel tasks, each searching for a match from a file containing 10 000 nucleotide sequences into the complete nucleotide sequences database of mouse. The execution of the workflow has a cyclic use of the memory and constantly uses the CPUs, making it a memory-intensive and a CPU-intensive application.

Again, the required hardware resources have been selected by experimentation. Each task asks for 4 cores, 2 GB of RAM and 10 GB of disk space and executes for about an hour. The selected VM flavor for the *Medium* mode is the *large* size.

3) Palmtree Workflow: Palmtree [16] is a library for the parallelization of Monte Carlo methods where the challenge is the proper management of the random numbers. The workflow structure is composed of 2 parallel tasks and its execution is CPU-intensive only.

Experimentation on this workflow showed us that assigning 4 cores, 2 GB of RAM and 10 GB of disk space to each task gives a good execution tradeoff. The *large* VM flavor is used to run the tasks when the *Medium* mode is selected.

D. Power Consumption Measurement

The three applications in our benchmark have been executed on servers equipped with fine-grained wattmeters of the Lyon site of Grid'5000 in order to have the energy consumed by their run in each execution mode. In order to have a fair energy sharing, we loaded as much as possible the servers by duplicating the jobs.

E. Performance versus Cost Tradeoff

A summary of the execution time versus the energy consumption of each workflow in each execution mode is given in Fig. 2. The number of servers required to run the workflows increases when the *Big* mode is selected which explains the energy consumption increasing. On the other hand, the execution time increases by a factor of 3 and more when the *Little* mode is selected.

IV. Experimental Validation

Table II presents our simulation results. We simulate a full day and a cluster with 330 servers (minimum number of servers required to be able to respond to the demand in the highest demand case). Each row presents the results for a profile distribution following the percentages given in the 3 first columns. All results are the average of 10 simulations and contain the energy consumption in KWh of the whole cluster, the maximum number of hosts required to execute the workload and the standard deviations.

The gray row corresponds to a simulation on a usual cloud infrastructure without any energy optimization. The unused servers are not powered down and all users select the *Big* execution mode because it reflects a common behavior when users want results as soon as possible. The last column is the percent of energy saved compared with the scenario of the first row. A scenario with a 50% energy saving means its execution consumes half of the execution with the scenario of the first row.

As we can see in the simulation results, a cloud system that turns off unused servers consumes less than 50% of usual cloud systems. The simulations in which the most energy has been saved are when 100% of the users selected the *Medium* and the *Little* execution modes. In these two cases, the energy saving is around 63% in comparison with the consumption of the first row scenario. The simulation costing the most in terms of energy is the scenario where 100% of the users select the *Big* execution mode, corresponding to an energy saving of 53.68%. It shows we can save important amounts of energy by avoiding the *Big* mode. Informing users about how much more their application consumes compared to another mode may encourage them to select a more energy-efficient execution mode and thus, motivates the implementation of an incentive mechanism. It also shows that the gap between the *Little* and *Medium* modes is very small. Selecting the *Little* mode is not always the best performance and energy tradeoff. Indeed, the energy may be very similar between the two modes and the execution time much longer in the *Little* mode compared with the *Medium* mode.

When 100% of the workload is using the *Little* mode, we can see that the workload uses a maximum of 143 servers out of the 330 servers available in the simulated cluster. Fewer servers turned on means a lower energy cost on the cooling system of the data center. It also means the cloud provider could buy less servers and thus, fewer ones to recycle after their lifespan. From another point of view, the low server utilization means this system can handle a higher number of users if most of them continue to use the *Little* mode.

In a realistic situation, they won't be 100% users using the same execution mode but rather a few percent in each of them. For table dimension reasons, Table II does not contain all possible distribution configurations but still reveals a link between the user profiles and the energy consumed. If we sort the table by descending order of *Big* users, we can see the energy consumption and the number of used hosts

TABLE II: The simulation results give the consumption of a whole cluster used during 24h from 2AM to 2AM the next day and the maximum number of hosts used with various profiles of job execution modes.

Big	Medium	Little	Energy (KWh)	Std dev energy	Hosts used	Std dev hosts	Energy saved
100	0	0	632.489	16.277	282	7.909	0.00 %
100	0	0	292.941	3.690	292	16.806	53.68 %
0	100	0	234.122	4.882	168	6.363	62.98 %
0	0	100	231.921	3.840	143	3.187	63.33 %
80	0	20	273.205	6.021	236	16.117	56.80 %
60	0	40	269.969	3.497	208	11.071	57.32 %
40	0	60	258.138	3.980	190	14.935	59.19 %
20	0	80	246.996	3.701	170	6.610	60.95 %
20	20	60	246.590	5.482	167	9.843	61.01 %
20	60	20	242.464	4.013	171	9.243	61.67 %

decreasing. So, the lower the number of *Big* users, the lower the consumption. However, for a fixed amount of *Big* users, the percentage variation of *Medium* and *Little* users has a small impact on the energy consumption. Thus, the system does not save much more energy with more *Little* users than *Medium* but globally allows the system to run the workload on fewer servers.

V. Conclusion

In this paper we present a simulation-based evaluation on how much an energy-aware cloud system could save in energy consumed by involving users in the energy conservation. In this system users can select an execution mode for running their applications. An execution mode controls the size of the application's VMs. The higher the mode, the larger the VMs and vice versa. A consolidation algorithm packs the VMs into a minimum number of servers in order to have a maximum of servers powered down. The smaller the VMs, the better the consolidation and the lower the global energy consumption of the infrastructure.

We simulated a typical daily use of a data center running 3 real scientific applications and varied the amount of applications in each execution mode. The simulation results show a saving of energy of more than 50% whatever the selected mode compared with cloud infrastructures where the servers are not turned off when not used. Scenarios where the most energy is saved are when 100% of users select the *Medium* and *Little* execution modes. However, cloud users tend to over-commit their job reservations [17] and they end up selecting the *Big* mode while the *Medium* mode is sufficient. The simulator results show the importance of reducing the amount of users using the *Big* mode and also that selecting the *Little* mode is not always the best practice because energy savings may be low and the application execution time much higher than with the *Medium* mode.

Acknowledgment

Experiments presented in this paper were carried out using the Grid'5000 experimental test-bed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

References

[1] insideHPC, "What is high performance computing?" 2012. [Online]. Available: http://insidehpc.com/hpc-basic-training/what-is-hpc/

[2] A. Gupta, L. V. Kale, F. Gioachin, V. March, C. H. Suen, B.-S. Lee, P. Faraboschi, R. Kaufmann, and D. Milojicic, "The Who, What, Why and How of High Performance Computing Applications in the Cloud," in *IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2013, p. 12.

[3] A. Gupta and D. Milojicic, "Evaluation of HPC Applications on Cloud," in *Open Cirrus Summit (OCS)*, Oct. 2011, pp. 22–26.

[4] W. Forrest, J. M. Kaplan, and N. Kindler, "Data centers: How to cut carbon emissions and costs," Dec. 2008.

[5] J. Koomey, *Growth in data center electricity use 2005 to 2010*. Oakland, CA: Analytics Press, 2011.

[6] A.-C. Orgerie, M. D. d. Assuncao, and L. Lefevre, "A Survey on Techniques for Improving the Energy Efficiency of Large-scale Distributed Systems," *ACM Comput. Surv.*, vol. 46, pp. 47:1–47:31, Apr. 2014.

[7] D. Guyon, A.-C. Orgerie, and C. Morin, "Energy-efficient User-oriented Cloud Elasticity for Data-driven Applications," in *IEEE International Conference on Green Computing and Communications (GreenCom)*, Dec. 2015, pp. 376–383.

[8] V. Villebonnet, G. Da Costa, L. Lefevre, J.-M. Pierson, and P. Stolf, "Big, Medium, Little: Reaching Energy Proportionality with Heterogeneous Computing Scheduler," *Parallel Processing Letters*, vol. 25, 2015.

[9] "Amazon EC2." [Online]. Available: https://aws.amazon.com/ec2/

[10] "Parallel Workloads Archive." [Online]. Available: http://www.cs.huji.ac.il/labs/parallel/workload/

[11] M. D. De Assuncao, J.-P. Gelas, L. Lefevre, and A.-C. Orgerie, "The Green Grid5000: Instrumenting and using a Grid with energy sensors," in *Remote Instrumentation for eScience and Related Aspects*. Springer, 2012, pp. 25–42.

[12] M. Yue, "A simple proof of the inequality FFD (L) 11/9 OPT (L) + 1, L for the FFD bin-packing algorithm," *Acta Mathematicae Applicatae Sinica*, vol. 7, no. 4, pp. 321–331, 1991.

[13] "The MetaCentrum 2 log." [Online]. Available: http://www.cs.huji.ac.il/labs/parallel/workload/l_metacentrum2/index.html

[14] "Montage." [Online]. Available: http://montage.ipac.caltech.edu/

[15] "Blast." [Online]. Available: http://blast.ncbi.nlm.nih.gov/Blast.cgi

[16] L. Lenôtre, "A Strategy for Parallel Implementations of Stochastic Lagrangian Simulation," Inria, Tech. Rep., Nov. 2015.

[17] R. Ghosh and V. K. Naik, "Biting Off Safely More Than You Can Chew: Predictive Analytics for Resource Over-Commit in IaaS Cloud," in *IEEE 5th International Conference on Cloud Computing (CLOUD)*, June 2012, pp. 25–32.