# A Pareto-based GA for Scheduling HPC Applications on Distributed Cloud Infrastructures

Yacine Kessaci, Nouredine Melab
and El-Ghazali Talbi
*INRIA Lille, CNRS/LIFL, Universite Lille 1.*
*Parc Scientifique de la Haute Borne,*
*40 avenue Halley Bt.A, Park Plaza,*
*59650 Villeneuve d'Ascq, France.*
*{yacine.kessaci, nouredine.melab, talbi}@lifl.fr*

## ABSTRACT

*Reducing energy consumption is an increasingly important issue in cloud computing, more specifically when dealing with High Performance Computing (HPC). Minimizing energy consumption can significantly reduce the amount of energy bills and then increases the provider's profit. In addition, the reduction of energy decreases greenhouse gas emissions. Therefore, many researches are carried out to develop new methods in order to consume less energy. In this paper, we present a multi-objective genetic algorithm (MO-GA) that optimizes the energy consumption, $CO_2$ emissions and the generated profit of a geographically distributed cloud computing infrastructure. We also propose a greedy heuristic that aims to maximize the number of scheduled applications in order to compare it with the MO-GA. The two approaches have been experimented using realistic workload traces from Feitelson's PWA Parallel Workload Archive. The results show that MO-GA outperforms the greedy heuristic by a significant margin in terms of energy consumption and $CO_2$ emissions. In addition, MO-GA is also proved to be slightly better in terms of profit while scheduling more applications.*

**KEYWORDS:** scheduling, cloud computing, green computing, resource allocation, multi-objective optimization, genetic algorithm

## 1. INTRODUCTION

Cloud computing appears nowadays to be increasingly adopted in many areas. The field of high performance computing (HPC) does not derogate from the rule. However, computers use a significant and growing portion of energy in the world. Therefore, energy-aware computing is crucial for large-scale systems that consume considerable amount of energy. A recent study [1] shows that in 2005, the power used by servers represents about 0.6% of total U.S. electricity consumption. That proportion grows to 1.2% when cooling and auxiliary infrastructures are included. In the same year, the aggregate electricity bill for operating those servers and associated infrastructure was about \$2.7 billions and \$7.2 billions for the U.S. and the world, respectively. The total electricity consumed by servers doubled over the period 2000 to 2005 in worldwide.

On the other hand, green house gas emission is reaching a critical limit. A recent work 2007 [2] estimates that the global Information and Communications Technology (ICT) industry accounts for approximately 2% of global carbon dioxide emissions, this is equivalent to the amount emitted by the aviation. To face this phenomenon different governments are fixing limits for (ICT) industries.

Energy consumption has another drawback, affecting the profit of the providers. Indeed, according to Amazone's estimate [3], the energy-related costs amount represents 42% of the total data center budget, and includes both direct power consumption 19% and cooling infrastructure 23%, these values are normalized with a 15 years amortization. It clearly appears that all the problems cited before are somehow related and thus have to be treated simultaneously.

In this paper, we present a new work that aims to deal with energy, green house gas emission and profit at the same time using a Pareto approach. Indeed, we propose a meta-scheduler that uses a multi-objective genetic algorithm (MO-GA) to find the best scheduling according to those three objectives. This approach uses the geographical distribution of the data centers to find the best scheduling since the energetic, $CO_2$ and electricity pricing specifications of each area can be different. Our approach aims also to give the best Quality of Service (QoS) by meeting the maximum application's deadlines. Genetic algorithms make it possible by exploring a great range of potential solutions to a problem.

The remainder of the paper is organized as follows. In Section 2 we present the related work to our approach. Section 3 presents the application, system and energy models used in our problem modeling. Our approach is presented in Section 4. The results of our experimental study are discussed in Section 5. The conclusion is drawn in Section 6.

## 2. RELATED WORK

After a race to performance, utility and cloud computing paradigm are facing an energy problem. Hence, several works have been proposed in the field of the energy aware computing. However, most of those approaches tackle this topic by referring to single data center and focusing on scheduling dedicated applications. In [4, 5] for example a hardware technique (DVFS) is proposed, it consists of varying the CPU frequency in order to minimize the energy consumption. The drawback of this type of methods is the assumption that they make about a tight coupling between the tasks and the resources. Another way of reducing cloud computing energy footprints is proposed in [6]. This work uses the possibilities offered by the virtualization in order to apply a task consolidation through two heuristics in order to maximize the resource utilization. In [7] the author presents a reinforcement learning approach to deal with the optimization of two main aspects, performances and power consumption. All the previous presented works aim to reduce the energy consumption on single data centers or on multiple servers geographically concentrated. Except the work proposed in [8] which deals with energy consumption reduction in large-scale computational grids like Grid5000, by switching off idle nodes in a clever way.

Other approaches treat about the economic side in cloud computing, like in [9], where two algorithms are proposed based on a pricing model using processor sharing in order to balance between conflicting objectives (profit and resource utilization). In [10] Burge et al describes a method for heterogeneous machines that maximizes the profit by affecting the requests to the machines according to their energy cost. Another approach based on genetic algorithms and dealing with profit is presented in [11]. In this work a linear programming driven genetic algorithm is proposed. In fact, this latter aims to give the best meta-scheduling in a utility grid based on the idea of minimizing the combined costs of all users in a coordinate way.

All of the last presented approaches take into account the profit in their study but they don't consider the relationship between energy, green house gas emissions and profit. They also do not pay attention on how each one of those criterions can affect the others. The work presented by Garg et al in [12] traits those points, by proposing a new energy model that includes gas emissions and pricing. Several heuristics are proposed to find a good tradeoff between the objectives. However, this approach is an aggregation of objectives (i.e. it can only optimize one objective at time).

To deal with all the misses mentioned before, we propose a meta-scheduler using a multi-objective genetic algorithm to optimize the whole three objectives at the same time. In other words, our new approach provides a set of Pareto solutions (i.e. non-dominated solutions).

## 3. DISTRIBUTED CLOUD SCHEDULING MODEL

### 3.1. SYSTEM MODEL

Our model is based on a Infrastructure As A Service (IAAS) cloud model. Indeed, we are dealing with a two third architecture with in both sides a distributed cloud provider and clients. These latter have access to the cloud by requesting resources to the provider. The service proposed by the cloud provider in our approach is offering infrastructures to the clients in order to compute their HPC applications. The role of this work is to help the provider to optimize a certain number of criterions while proposing its service. The model of our cloud is geographically distributed over the world. The originality of this approach is to propose a meta-scheduling algorithm that uses a multi-objective genetic algorithm in order to find the best scheduling to the applications over the time. The treated objectives of our algorithm are energy, carbon emissions and profit. The meta-scheduler algorithm aims to give the best Quality of Service (QoS) by meeting applications deadlines and respecting the model's constraints, while helping the provider to maximize his profit, minimize his energy consumption and his gas emissions. The optimization of the objectives is due to the characteristics offered by the geographical distribution of the data centers.

## 3.2. ENERGY MODEL

The energy consumption of a data center results from IT equipments and auxiliary equipments. In our work we consider only cooling energy consumption among the auxiliary equipment and computing energy consumption in IT equipments. Indeed, since our approach treats about HPC applications, the most energy consumption is resulted from the intensive computation. Our approach also does not pay attention on how the energy is optimized within the data center. This is due to the fact that our work deals with a meta-scheduler which aims to prove the advantage of a geographical dispersion in a cloud.

Our processors energy model is derived from the power consumption model in Complementary Metal-Oxide Semiconductor (CMOS) logic circuits given by $P = \alpha f^3 + \beta$.

In addition, another source of energy consumption needs to be taken into account. In fact, the energy used for cooling the data centers is consequent and has to be integrated in our energy model. Energy dedicated for cooling is tightly related to the geographical area where the data center is situated since the temperature changes from an area to another. To compute cooling energy amount, each data center has a coefficient called $COP$ which represents the ratio between the energy dedicated for the execution of the request and the energy used for cooling the system. By using $COP$ the meta-scheduler is able to deduce the energy consumed by each data center for cooling their devices.

## 3.3. PROBLEM DESCRIPTION

Our problem is composed of two third parties. The first party is a cloud provider which has $N$ data centers geographically distributed over different areas in the world. The second party is clients with $J$ HPC applications that have to be executed on the data centers. The problem consists of scheduling $J$ applications on $N$ data centers. We know that the task scheduling problem in general is NP-hard [14]. Therefore our multi-objective scheduling problem is NP-hard as well. Thus, a metaheuristic algorithm (Genetic Algorithm) appears to be the most appropriate approach to adopt. In our description the provider has to pay the execution price of the used data center $i$, this price is the result of the electricity consumption during the computation. According to this latter the provider fixes a price for the clients. We note the fixed client price per hour by $p^c$ (\$/CPU/hour). The $CO_2$ amount of each data center $i$ is calculated from a ratio noted $r_i^{CO_2}$. This latter is an average value that varies according to the way the data center's electricity is produced.
During the scheduling process, the user submits a request for a HPC application $j$. A request is defined by a triplet $(e_j, n_j, d_j)$, all the triplet information are given by the user during the reservation, except the starting time of the application $(t_j)$ which is deduced from the submission time. The elements of the triplet represent the runtime of the application $(e_j)$, the number of processors needed by the user for his application $(n_j)$ and finally the deadline after what the application will be considered as failed $(d_j)$. We inspired our triplet from Amazon EC2 [13] which asks the user to know about the duration time of his application. Thus, the user has sometimes to pay for a longer reservation to ensure the completion of his application even if this latter finishes before the end of the reservation time.

The objective functions of our approach aim to minimize both energy consumption and carbon emissions and maximize the profit of the entire distributed cloud simultaneously. This is done always by respecting the following constraints:

- The application $j$ has to finish before $d_j$ else the scheduling is rejected
- Each application $j$ can be affected to one and only one data center $j$

## 4. MULTI-OBJECTIVE GENETIC ALGORITHM FOR META-SCHEDULER

### 4.1. PROBLEM ENCODING

In order to formulate our problem without overriding the previous constraints (i.e. The application has to finish before its deadline and each application can be scheduled only on one data center), we propose an encoding for the MO-GA individuals see Figure 1.

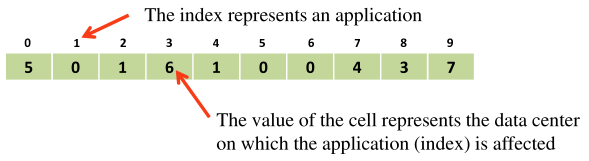
The index represents an application
0 1 2 3 4 5 6 7 8 9
5 0 1 6 1 0 0 4 3 7
The value of the cell represents the data center on which the application (index) is affected

**Figure 1. Problem Encoding**

Figure 1 represents one possible scheduling among plenty that proposes the genetic algorithm. In the proposed example we identify three major specifications. The indexes of the table depict the applications that are scheduled, the number which is contained by each cell of the table identifies the data center to which the application is allocated. In other words if we look at Figure 1, the first cell represents the first application of the pool that is currently treated by the MO-GA, in this case this application is allocated to the data center 5. The second application is allocated to the data center 0 and so on. This encoding tells us about the number of applications contained by the pool, our example contains 10 applications. This encoding helps us to deal with the characteristics of our problem. Indeed, it allows scheduling all the applications of the pool, each application will be affected to one and only one data center. A data center can handle more than one application and not all the data centers are necessarily used in each solution. The last constraint that cannot be handled by the proposed encoding is the deadline constraint. We deal with this latter in the algorithm by rejecting the solutions that do not respect the deadlines.

### 4.2. META-SCHEDULING ALGORITHM STEPS

Before each scheduling, the meta-scheduler waits for a fixed period called scheduling cycle. This period helps to gather a pool of applications in order to have a larger choice and thus optimize the future scheduling. Once this phase done the pool is treated by the MO-GA to find the best schedulings possible on the data centers. The result of the execution is stored on a Pareto front. Once the set of Pareto solutions (schedulings) is proposed, the algorithm chooses one scheduling according to the user's choice. The chosen solution from the Pareto set is used as a state for the data centers. This state will be a basis from which the next iteration of the algorithm will make another execution on a new pool of applications. The algorithm keeps iterating and proposes schedulings for each pool of applications until no more application arrives.

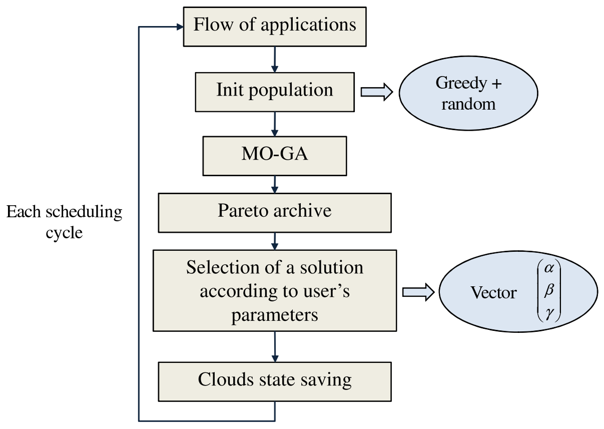
Flow of applications
Init population
Greedy + random
MO-GA
Pareto archive
Each scheduling cycle
Selection of a solution according to user's parameters
Vector $\begin{pmatrix}\alpha\\ \beta\\ \gamma\end{pmatrix}$
Clouds state saving

**Figure 2. The Flowchart of the Meta-scheduler Algorithm**

### 4.3. GENETIC ALGORITHM OPERATORS

The role of the genetic algorithm is to make a number of combination from the initial population using genetic operators in order to find the best scheduling according to the specified objectives. The genetic algorithm starts by initializing the population. Our initialization works in two phases. The first phase aims to maximize the number of treated applications thanks to a greedy method, while the second phase adds diversity to the population by using a random method. This population is used to generate offsprings based on two operators named mutation and crossover. Each time a modification is performed by those operators to the individual, an evaluation operator (fitness) is called to evaluate the offsprings. In other words this operator calculates the energy consumption, $CO_2$ emissions and the generated profit of each scheduling (solution). The next step of the MO-GA is the selection of the best solutions among the offsprings and the replacement of other solutions according to a replacement strategy in order to keep a constant number of individuals in the population. Our GA's selection operator is based on a tournament strategy. The algorithm stops when no new best solution is found after a fixed number of generations. The principle of our mutation operator is conventional. Indeed, the operator chooses randomly two integers $i$ and $j$ such that $1 \leq i < j \leq n$. Then, the operator swaps the two applications $i$ and $j$. The crossover operator uses two solutions $s1$ and $s2$ to generate two new solutions $s'1$ and $s'2$. The operator uses two points on each solution to perform the crossing over.

## 4.4. DISTRIBUTED CLOUD STATE SELECTION

In our meta-scheduling algorithm there is a selection step which comes right after the end of the multi-objective genetic algorithm. This step aims to pick up a solution among the Pareto set in order to fix the distributed cloud state. This state will be the starting point from which the next execution of the MO-GA will schedule a new pool of applications. The idea behind choosing a Pareto approach in our work is to propose to the provider the biggest number of solutions. Each one of these solutions is better than the other regarding a certain objective. The mechanism of selection of the solution can be seen in different ways. The first mechanism is a manual choice done at each step by the provider, the second is our solution that uses a vector as an input parameter in order to automate the progression of the experimentations. Our vector parameter is a three dimensional vector. Indeed, since we deal with three objectives each dimension represents a weighting for a particular objective. In the state selection step, the vector has a direction on which it points to. This direction is set by the provider. The solution that is the nearest to the vector's direction is the one which is chosen among the other in the Pareto set.

## 5. EXPERIMENTS AND RESULTS

This section presents the results obtained from our comparative experimental study. The experiments aim to demonstrate and evaluate the contribution of the multi-objective evolutionary approach compared to a maximum scheduling application heuristic.

## 5.1. EXPERIMENTAL SETTINGS

The experimental settings concern both sides of our model, client side with its applications and provider side with the hardware configuration of the cloud.

- Application settings: Since our approach deals with HPC applications, we used realistic workloads traces from Feitelson's Parallel Workload Archive (PWA) [15]. We used the whole archive time workload. The workload traces stretch over a period of five months of applications (January 2007 to June 2007). The traces that we use are the Lawrence Livermore National Laboratory (LLNL) from the Thunder cluster because of their high rate of resources utilization 87.6%, this helps to simulate a heavy workload scenario. The information that we extract from LLNL's workload traces are the submit time, the runtime and the number of required processors. The traces do not have information about the applications deadlines. We inspired our self from the method presented in [16] to generate synthetically the deadlines for the needs of our experimentations. The applications were classified into two classes named High Urgency (HU) and Low Urgency (LU). 80% of the applications belong to (LU) and 20% to (HU). A (HU) application has 3 times less time on average to finish its execution then (LU) application.

- Distributed cloud settings: In our approach we use 8 data centers geographically distributed with the same specifications as in [12]. The COP (power usage efficiency) of each data center is given by a uniform distribution between [0.6,3.5]. The electricity prices and carbon emission rates are taken from respectively US Energy Information Administration (EIA) report [17] and US Department of Energy (DOE) [18]. Since we are dealing with a meta-scheduler we do not use energy reducing techniques within the data centers.

## 5.2. ALGORITHM PARAMETERS

In our experimentations we use some parameters (state selection vector, varying arrival rate of applications, client execution price and scheduling cycle). The state selection vector presented in Section 4.4 is used in order to make the experiments continue from a pool of applications to another. We performed experiments with 4 different vectors. The first vector without advantaging any of the three objectives, the second with a maximum advantage to energy criterion, the third with a maximum advantage to $CO_2$ criterion and the last one with a maximum advantage to the profit criterion. Concerning varying arrival rate we use variations of the original workload by changing in each arrival rate the submit time of the applications. We used 4 arrival rates in our experiments (Low, Medium, High, Very high). Each move from an arrival rate to another represents 10 times more applications arrival during the same period of time. Thus, by shortening the submit time of the applications we increase the workload. The client price is fixed as to be twice the energy cost of a data center. Scheduling cycle represents the period where in our algorithm the meta-scheduler waits for applications in order to optimize the scheduling. Table 1 summarizes the parameters used in our experiments.

**Table 1. Experimental Parameters**

| Parameter | Value |
|---|---|
| Total number of applications | 128662 |
| State selection vector | $\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right)$ |
| | (1,0,0) (0,1,0) (0,0,1) |
| Arrival rate | Low Medium High Very high |
| Client execution price | \$0.40/CPU/h |
| Scheduling cycle | 50s |

## 5.3. PERFORMANCE EVALUATION

To the best of our knowledge no previous approach deals with a Pareto multi-objective genetic algorithm for a distributed cloud meta-scheduler. Thus, we perform a bench of experimentations with different parameters. In addition of optimizing the three objectives, the approach has first to satisfy the maximum number of clients (applications). A comparison between our approach and a maximum scheduling application heuristic appears to us to be the best choice to evaluate our work. In order to switch from a data center state to another we used 4 different vectors Table 1. These vectors help through their parameters to choose the scheduling that will be used for the switching state. The vectors can help also to extract the best solution among the Pareto set for a given objective to be able to compare our Pareto approach to the heuristic. The results are presented in Table 2. Experiments show that our approach by using a vector setting to Average see Table 2, improves the results obtained by the greedy heuristic while scheduling more applications. Setting the vector to Average consists of not advantaging any of the three objectives. In addition, our experimentations indicate that we can separate the arrival rate into two parts according to the behavior of the meta-scheduler. Indeed, we noticed that the vector state selection settings while advantaging one objective helps to find better solutions for this objective only for the Low and Medium arrival rates. This phenomenon can be explained by the fact that for High and Very high application arrival rates, the meta-scheduler handles a huge pool of applications. In this case, when this latter chooses a state that advantage a specified objective, this will lead it to a local optima and will make all the good data centers for the specified objective busy. Hence, the next pool of applications will be allocated into bad data centers and leads to worst results.

To summarize, in our approach, the energy consumption is reduced up to **4,66%**, the $CO_2$ emissions up to **10,85%** and the profit maximized up to **1,62%** compared to the maximum scheduling application heuristic, while scheduling averagely **2,67%** more application then this latter.

## 6. CONCLUSION

In this paper, we presented a new meta-scheduler using a multi-objective genetic algorithm to minimize energy consumption, gas emission and maximize the profit while respecting application deadlines. The energy saving of our approach exploits the geographical distribution of the data centers that compose the distributed cloud. According to our classification, the new work can be considered as an optimizing multi-objective method with a Pareto approach.

Our new approach has been evaluated with realistic workloads traces from Feitelson's Parallel Workload Archive (PWA) [15].

Experiments show that our multi-objective GA improves on average the results obtained by the greedy method particularly in reducing $CO_2$ emissions. Indeed, the $CO_2$ emission is reduced by up to **10,85 %**, the energy consumption by up to **4,66 %** and the profit is maximized by up to **1,62%**. In addition, our approach schedules on average **2,67%** more applications then the heuristic that maximizes the number of scheduled application.

Therefore, one of the main perspectives of the work presented in this paper is to determine on one hand a way to minimize more the energy consumption by using Dynamic Voltage Scaling (DVS) within the data centers, and on the other hand to modify the model by allowing delays for the applications by introducing a new pricing model with penalties. In addition, we can also imagine a dynamic meta-scheduler which will reassign applications during a scheduling phase on different data centers to optimize energy and/or profit. However, this will depend on the flexibility, the data transfer cost and the CPU time complexity of the applications since we deal with HPC applications.

**Table 2. Non Specific Objective Oriented Selection Vector Results**

| Vector settings | Average | | | | Greedy | | | |
|---|---|---|---|---|---|---|---|---|
| Value for each criterion / Arrival rate | Energy (kW h) | $CO_2$ (Kg) | Profit ($) | Failed applications | Energy (kW h) | $CO_2$ (Kg) | Profit ($) | Failed applications |
| Low | 1.807e+06 | 709050 | 4.726e+06 | **2270** | 1.807e+06 | 709039 | 4.726e+06 | 2270 |
| Medium | 1.813e+06 | 717417 | 4.700e+06 | **2588** | 1.833e+06 | 727748 | 4.696e+06 | 2635 |
| High | 1.931e+06 | 837531 | 4.683e+06 | **3045** | 2.026e+06 | 939550 | 4.677e+06 | 3357 |
| Very high | 2.036e+06 | 972912 | 4.660e+06 | **5168** | 2.036e+06 | 972912 | 4.660e+06 | 5168 |

## REFERENCES

[1] J. G. Koomey, "Estimating total power consumption by servers in the u.s. and the world."

[2] Gartner. Gartner estimates ict industry accounts for 2 percent of global co2 emissions. http://www.gartner.com/it/page.jsp?id=503867.

[3] J. Hamilton, "Cooperative expendable micro-slice servers (cems): Low cost, low power servers for internet-scale services," in Proceedings of 4th Biennial Conference on Innovative Date Systems Research (CIDR), Asilomar, California, USA, January, 2009.

[4] Y. C. Lee and A. Y. Zomaya, "Minimizing energy consumption for precedence-constrained applications using dynamic voltage scaling," in CCGRID'09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009, pp. 92–99.

[5] N. B. Rizvandi, J. Taheri, A. Y. Zomaya, and Y. C. Lee, "Linear combinations of dvfs-enabled processor frequencies to modify the energy-aware scheduling algorithms," Cluster Computing and the Grid, IEEE International Symposium on, vol. 0, pp. 388–397, 2010.

[6] Y. Lee and A. Zomaya, "Energy efficient utilization of resources in cloud computing systems," The Journal of Supercomputing, pp. 1–13, 2010, 10.1007/s11227-010-0421-3.

[7] G. Tesauro, R. Das, H. Chan, J. O. Kephart, D. Levine, F. L. R. III, and C. Lefurgy, "Managing power consumption and performance of computing systems using reinforcement learning," in NIPS, 2007.

[8] A.-C. Orgerie, L. Lefevre, and J.-P. Gelas, "Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems," in Parallel and Distributed Systems, 2008. ICPADS '08. 14th IEEE International Conference on, 2008, pp. 171 –178.

[9] Y. C. Lee, C. Wang, A. Y. Zomaya, and B. B. Zhou, "Profit-driven service request scheduling in clouds," in Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on, May 2010, pp. 15 –24.

[10] J. Burge, P. Ranganathan, and J. Wiener, "Cost-aware scheduling for heterogeneous enterprise machines (cash em)," in Cluster Computing, 2007 IEEE International Conference on, 2007, pp. 481 –487.

[11] S. Garg, P. Konugurthi, and R. Buyya, "A linear programming driven genetic algorithm for meta-scheduling on utility grids," in Advanced Computing and Communications, 2008. ADCOM 2008. 16th International Conference on, 2008, pp. 19 –26.

[12] S. K. Garg, C. S. Yeo, A. Anandasivam, and R. Buyya, "Environment-conscious scheduling of hpc applications on distributed cloud-oriented data centers," Journal of Parallel and Distributed Computing, vol. In Press, Corrected Proof, pp. –, 2010.

[13] Amazon elastic compute cloud (amazon ec2. http://aws.amazon.com/fr/ec2/.

[14] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. New York, NY, USA: W. H. Freeman & Co., 1979.

[15] D. Feitelson. (2009, Aug) Parallel workloads archive. http://www.cs.huji.ac.il/labs/parallel/workload.

[16] S. Venugopal, X. Chu, and R. Buyya, "A negotiation mechanism for advance resource reservations using the alternate offers protocol," in Quality of Service, 2008. IWQoS 2008. 16th International Workshop on, june 2008, pp. 40 –49.

[17] (2007) (EIA) http://www.eia.doe.gov/cneaf/electricity /epm/table5_6_a.html.

[18] (2007)(DOE) http://www.eia.doe.gov/oiaf/1605 /pdf/Appendix20F_r071023.pdf.