A Green Model of Cloud Resources Provisioning

Meriem Azaiez[1], Walid Chainbi[1] and Hanen Chihi[2]
[1]*National Engineering School of Sousse, University of Sousse, Sousse, Tunisia*
[2]*Institute of Computer Sciences of Ariana, Ariana, Tunisia*

Keywords: Cloud Computing, Optimization, Scheduling, Green Computing, CloudSim, Genetic Algorithm.

Abstract: The evolution of network technologies and their reliability on the one hand, and the spread of virtualization techniques on the other hand, have motivated the use of execution and storage resources allocated by distant providers. These resources may progress on demand. Cloud computing deals with such aspects. However, these resources are greedy in energy because they consume huge amounts of electrical energy, which affects the invoicing of Cloud services which depends on run-time and used resources. The environment is affected too due to the emission of greenhouse gas. Therefore, we need Green Cloud computing solutions that reduce the environmental impact. To overcome this Challenge, we study in this paper the relationship between Cloud infrastructure and energy consumption. Then, we present a genetic algorithm based solution that schedules Cloud resources and optimizes the energy consumption and CO_2 emissions of Cloud computing infrastructure based on geographical features of data centers. Unlike previous work, we propose to optimize the use of Cloud resources by scheduling dynamically the customers applications and therefore reduce energy consumption as well as the emission of CO_2. The optimal solution of scheduling is found using multi-objective genetic algorithm. In order to test our model, we extended the CloudSim simulator with a module implementing the dynamic scheduling of customers applications. The experiments show promising results related to the adoption of our model.

1 INTRODUCTION

Cloud computing is an emerging field which becomes increasingly popular. But, this technology is identified as one of the fastest growing consumers of energy. This consumption of energy will reach in 2020, more than three times compared to today (Relaxnews, 2010). This problem is caused by the energy consumed by data centers. Indeed, the data centers energy consumption increases with the number of centers and with data center workload. This consumption is amplified when the cooling infrastructure and auxiliary equipment are included which represents more than 50% of the power consumption (Zhang et al., 2012).

Another problem with energy is the emission of greenhouse gas which reached the 2% of the total amount of CO_2 emissions in the world (TUAL, 2013) and will quadruple in 2020 (Thrash, 2012). This problem brings a huge impact to the environment. Therefore, a new challenge appears to deal with the energy consumption and greenhouse gas emissions.

Many studies have addressed the green provisioning of resources to reduce energy consumption in Cloud environment. Most of these works use static allocation methods such as FCFS (First-Come-First-Serve) to ensure performance and quality of services (Calheiros et al., 2011). These strategies are very simple, but the problem with them is the need of many available resources. The energetic efficiency of these resource provisioning methods depends on the number of customers applications. Other methods used in the Cloud environment are deployed pre allocation strategies (Nair and Jayarekha, 2012). But the problem here is the prediction of required resources, which is difficult.

Unlike theses works, we propose to include green aspect in Cloud environment to support the Green Cloud computing. Indeed, the present study allows finding a solution to the green provisioning by scheduling customers applications to optimize the use of Cloud resources and therefore reduce energy consumption as well as the emission of CO_2.

We use the information of the Cloud infrastructure resources and their relations with energy consumption and CO_2 emissions. The main objectives are to min-

Azaiez M., Chainbi W. and Chihi H..
A Green Model of Cloud Resources Provisioning.
DOI: 10.5220/0004940701350142
In *Proceedings of the 4th International Conference on Cloud Computing and Services Science* (CLOSER-2014), pages 135-142
ISBN: 978-989-758-019-2
Copyright © 2014 SCITEPRESS (Science and Technology Publications, Lda.)

imize these two factors on the one hand, and to reduce the cost of services on the other hand. To optimize these objectives, we include our parameters in a multi-objective genetic algorithm and we execute the optimal solution in our Cloud which is tested as an extension of the CloudSim framework.

The remainder of this paper is organized as follows. Section 2 discusses related work. A detailed description of our solution is presented in section 3. Then in section 4, we provide technical details of our work. Section 5 deals with the experiments and the results of the simulations which are produced with comparisons and detailed analysis. Finally, the paper ends with brief conclusive remarks and discussion on future studies directions.

2 RELATED WORK

Cloud resources optimization is difficult to meet because of the uncertainty of future consumer demand and resource prices. It has been a topic of research interest and development for many years. To address the growing challenge, techniques from many disciplines were integrated synergistically. Next, we present the state of the art of cloud resources optimization methods.

Cloud computing's usage-based pricing model creates an incentive for subscribers to optimize the utilization of the rented resources. Borovskiy et al. (Borovskiy et al., 2011) devise a formal approach for distributing workload among a minimum number of servers. They model this problem as a linear programming problem and describe two solution approaches. The first one generates a set of candidate blocks and then composes an optimal partition by solving an integer programming problem. The second approach solves the set partitioning problem with column generation technique. The disadvantage of such method is its difficulty to consider the purpose of Cloud resources optimization because of the nonlinear characteristics of users' demands distribution.

Chaisiri et al. (Chaisiri et al., 2012) propose a method to optimize Cloud resources cost. The under provisioning problem can occur when the reserved resources are unable to fully meet users' demands due to its uncertainty of the workload distribution. To address this problem, the authors propose an optimal cloud resource provisioning algorithm based on stochastic programming model.

Regarding the problem of the description of the Cloud resources characteristic with nonlinear equation, some researchers propose the use of a stochastic optimization approach. For example, Li proposes a model based on stochastic integer programming for Cloud resources optimization (Li, 2012). He proposes to address the SLA-aware resource composition problem. He defines a stochastic integer programming model for resource composition and provides an algorithm that implements Grbner based theory to solve this problem (Buchberger, 2001).To solve the minimization problem of Cloud infrastructure cost, Zhao et al. developed a deterministic model for resource reservation planning, using a mixed integer linear algorithm, to generate optimal decisions given fixed parameters (Zhao et al., 2012). In addition, they proposed a stochastic model of resource rental planning which explicitly takes into account the uncertainty of resources and users' demand in the decision making process. One major disadvantage of such approaches, especially in dynamic environments where the optimal solution changes over time, is that the parameter estimation phase significantly delays the implementation of an optimal solution.

Other researchers use the constraint satisfaction problem (CSP) approach to solve the problem of Cloud resources optimization. Van et al. propose a two-level based architecture that defines a clear separation between application specific functions and a generic global decision level (Nguyen Van et al., 2009). They use utility functions to map the current state of each application for a scalar value that quantify the "satisfaction" of each application in terms of its performance targets. These utility functions are also means of communication with the layer of global decision which builds a global utility function including the costs of resource management. The stage of provisioning of virtual machines has been separated from the stage of placement of virtual machines within the global decision layer loop and formulates both problems as constraint satisfaction problem. Doughertya et al. propose a model driven approach to optimize the configuration, the energy consumption and the cost of infrastructure for Cloud infrastructure self-scaling to create green IT environments that reduce emissions resulting from the use of redundant resources unused (Dougherty et al., 2012), (Dougherty et al.,). They proposed to decompose the model to four sub-problems to ensure infrastructure auto-scaling: explaining how virtual machine configurations can be captured; describe how these models can be transformed in constraint satisfaction problems for the configuration and optimization of energy consumption; showing how optimal auto-scale configurations can be derived from these CSPs with a constraint solver and present a case study of energy consumption and cost reduction of production of this model-driven approach. The main drawback of these methods is

there exponential complexity.

Bio-inspired scheduling algorithms are often used in heterogeneous computing environments. Chaisiri et al. propose an optimal VM placement algorithm which optimally allocates VM to multiple cloud providers and follows optimally in advance the resources reservation (Chaisiri et al., 2009). This algorithm minimizes the cost virtual machines hosting in an environment of multiple cloud providers. Van et al. show that the ability to automate the provisioning and dynamic placement of virtual machines, taking into account both the software-level' SLA and resource cost with high-level handles for the monitor to specify compromise between the two (Van et al., 2009). The model defines a support for heterogeneous applications and workloads including both enterprise online applications with stringent QoS requirements and batch-oriented CPU intensive applications. It is not focused on optimization problems that are NP-hard in their general form.

Kessaci et al. (Kessaci et al., 2011) propose to optimize the allocation of VM requests using a Pareto-based meta-heuristic approach. In fact, they use a multi-objective genetic algorithm and propose to adopt new mutation and crossover strategies to produce new generations. The three objectives are considered in the optimization process: minimize both energy consumption and CO2 emissions of the cloud infrastructures and maximize the profit of the suppliers. To formulate the problem, they use a real encoding. Each individual is a vector representing the result of processing a pool of applications received during the scheduling cycle. The used encoding identifies three main features: the index of the vector represents the applications, the value of each cell identifies the VM on which the application will be scheduled and the maximum number of application. The Initialization of the MOGA population is divided into three steps: read the application pool with the greedy method (Black,), initializes one or two elements of the population by the result of the first step and randomly initializes the rest of the population. The problem of this approach is the use of the greedy algorithm to initialize the population of the MOGA algorithm. Despite their simplicity, greedy algorithms can be subtle and they are costly and mostly provide a local minimum. All of the presented approaches take into account the optimization of the Cloud resources but they do not consider the relationship between the satisfaction of users' requests and the optimization of providers' infrastructures cost. They do not pay attention on how each one of those parts can affect the other. In fact, the optimization of cost and response time of clients' requests are closely related to the number of available and active resources. Our objective is to minimize the number of the active resources that minimize energy consumption. To resolve this problem, we propose to use a multi-objective optimization. The initialization of the MOGA population is a real time process. It amounts to read the application pool and extracts useful information for the optimization algorithm.

3 THE PROPOSED APPROACH

Cloud computing can be represented in three main categories which are based on the capacity of abstraction and the paradigm of services. Thus we have the SaaS, PaaS and IaaS. In our work, we focus on IaaS as our goal is Cloud requests scheduling.

Iaas provides processing capacities and storage as well as network components as standardized services. These services manage a workload requested by customers applications.

3.1 The Mathematical Model

The Cloud model adopted by this study is IaaS with two-tier architecture as shown in figure 1. On the one hand, we have the Cloud services provider and on the other hand, the customer applications which need services.

Figure 1: The architecture of our Cloud model.

The optimization of our objectives, which are the energy consumption and the CO_2 emission, is owed to the exploitation of features offered by geographical distribution of data centers. Indeed, the parameters of energy model as well as CO_2 model are different from one geographical site to another.

In the Cloud, there are two sources of energy consumption: energy from computing equipment, which is the energy required for calculation and energy from auxiliary equipment, which is the energy required for cooling.

To make the energy equation, we use the model of CMOS (Complementary Metal-Oxide Semiconductor) processors by adding two constraints of the problem which are run time (t_{run}) and number of processors required to run the customer application (nb_{pr}). Thus, the final formula of the energy required for calculation is

$$E_{cal} = (\alpha f^3 + \beta) \times t_{run} \times nb_{pr} \quad (1)$$

The energy performance coefficient (COP) is the ratio between the produced heat and the energy consumed in the treatment. This factor leads us to deduct the cooling energy which is

$$E_{aux} = \frac{E_{cal}}{COP} \quad (2)$$

Therefore, the energy to minimize is the total energy:

$$E_{total} = E_{cal} + E_{aux} \quad (3)$$

Regarding the second objective, which is the emission of greenhouse gas, we use in its formula CO_2 rate, which depends on geographical locations. The formula of this second objective is

$$CO_2 = E_{total} \times Rate_{CO_2} \quad (4)$$

To optimize simultaneously these two objectives, we use the multi-objective optimization techniques (Goldberg, 1996).

3.2 The Multi-objective Optimization

A multi-objective optimization problem is to optimize several objective functions simultaneously. Our optimization problem is defined as follows:

Minimize $F(x) = (f_1(x), f_2(x))$ where both functions to optimize f_1 and f_2 are respectively the function of energy and the function of CO_2, $x = (x_1, , x_n)$ is the vector of decision variables which are the run time and the number of required processors, $F(x)$ is the vector of objectives which will be optimized.

The single optimal solution in the mono-objective optimization problem is replaced by the concept of Pareto optimal solutions in multi-objective optimization problem. Therefore, the optimal solution is not a single solution but a set of solutions. To find the right balance of solutions, it is necessary to identify the relation between these objectives. The most used relation is the relation of Pareto dominance. For this relation, all efficient solutions are called the Pareto Front. This set of solutions is a balance where no improvement can be made on an objective without degradation of at least another objective. So the purpose is to obtain the Pareto front or converge as much as possible on this front. Figure 2 shows an example of dominance relation in case there are two objectives to be maximized.

To solve our problem of multi-objective optimization based on Pareto-solving methods, we use the heuristic algorithms. These algorithms are used to explore the possible solutions space seeking the best solution. Among these algorithms, we use multi-objective genetic algorithm (MOGA).

Different models have been proposed for multi-objective genetic algorithms including VEGA (Vector

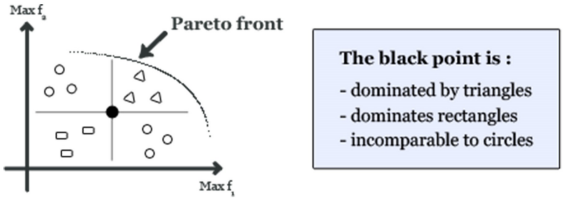

Figure 2: The relation of dominance.

Evaluated Genetic Algorithm), NPGA (Niched Pareto Genetic Algorithm), NSGA (Non Dominated Sorting Genetic Algorithm) etc. We adopt in our project, the NSGA-II because it uses an elitist approach that saves and injects the best solutions found in previous generations in the new generations (Melcher, 2007),(Deb et al., 2002). It uses a sorting procedure based on the non-dominance which is faster. It requires no parameter setting. It uses also a comparison operator based on a calculation of the Crowding distance. This distance is calculated from nearby solutions.

In our study, this algorithm makes the scheduling of the execution of customers applications with resource optimization. And to assess the effectiveness of the algorithm as well as the solution, we calculate the energy consumption and the emission of CO_2 after the execution of the applications.

4 SIMULATION ENVIRONMENT

4.1 CloudSim Extension

Since CloudSim provides an extensible framework for modeling and simulation of Cloud computing infrastructures and services, the present work is included in CloudSim as an extension. In fact, to create our application, we leverage some basic functions of CloudSim and we extend some characteristics and features.

In CloudSim, the management of Cloud resources and all allocation policies are standard and don't consider some characteristics of data centers. So, in the present work, we use this specification to investigate new model for resource allocation. This new technique is in accordance with ecological standards.

4.2 Class Diagram

Our application divided into two classes diagram:

The first class diagram is presented in figure 3. In this diagram we have three types of classes.

To meet the needs of our application, we add two classes. The first one is *Localisation*. This class contains all locations specifications of Datacenter. The

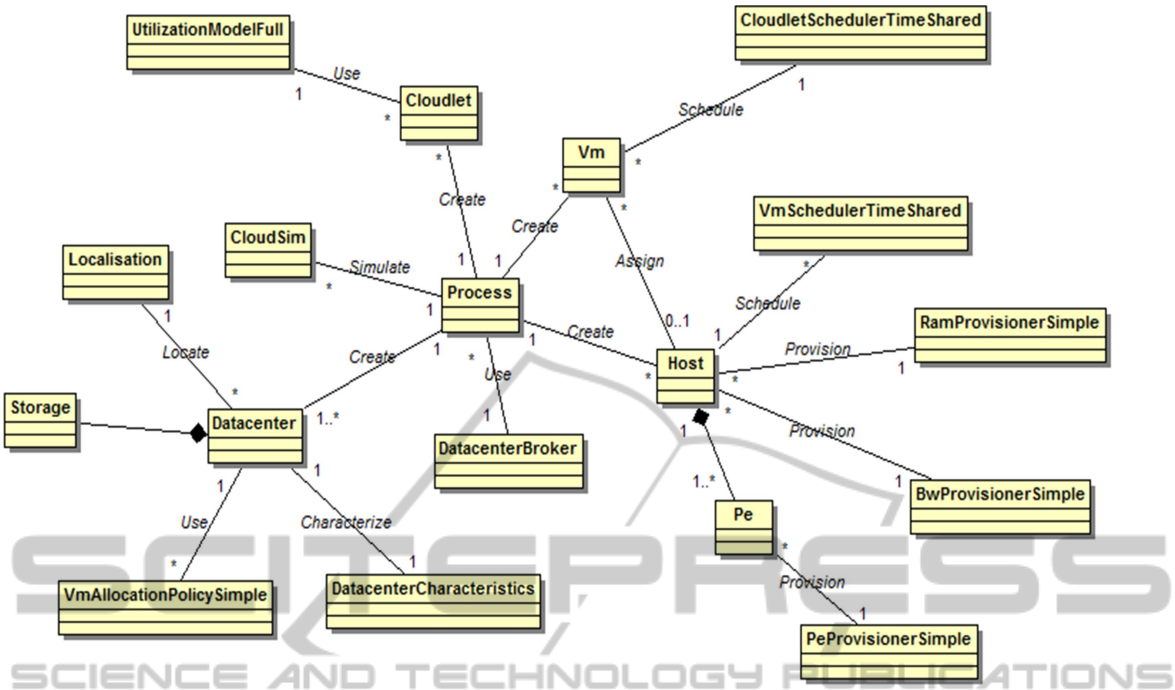

Figure 3: Project infrastructure class diagram.

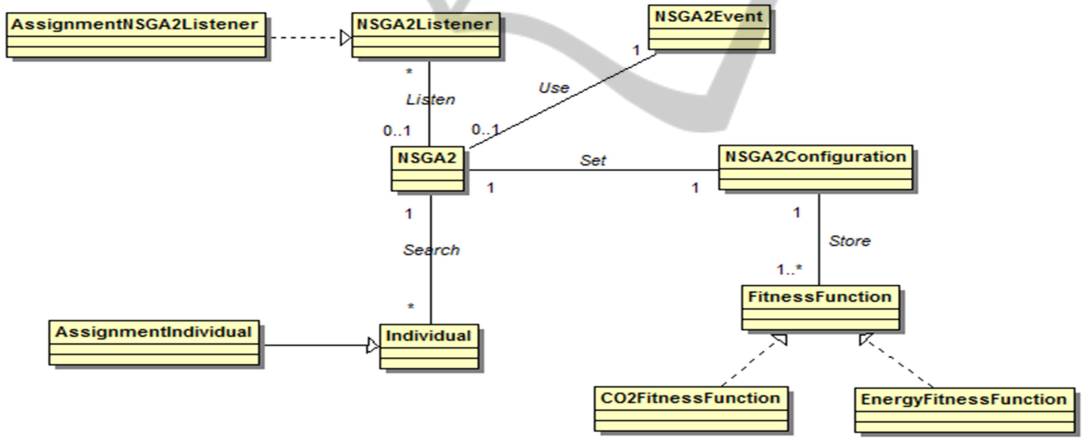

Figure 4: MOGA implementation class diagram.

second class is *Process*. This class is in charge of all communications between graphical interfaces, data access, Cloud configuration and simulation. Also, it allows to manage all Cloud components.

We modify the class *DatacenterBroker* to adapt it to our application. This class denotes a Cloud broker. It is a mediator between users requests and Cloud infrastructure. Since *DatacenterBroker* contains the process of creation of Cloud infrastructure and management of resource allocation policies, we change some details and we implement our new model in this class.

The other classes are those that we use in this work to create the environment of our new extension. The infrastructures of Cloud are represented by the class Datacenter which manages physical machines. Technical static properties of datacenters are in *DatacenterCharacteristics* class and desired functionalities of a storage system are in *Storage* Class.

The class *Host* represents a physical machine which has one or more processing element. Also, it has memory and bandwidth, managed by allocation policies, storage capacity and provisioning policy for assignment to one or more virtual machines. Each processing element of hosts, represented by *Pe* class, has a processing capacity. The allocation in virtual machines depends on physical characteristics of hosts.

User applications, represented by the class *Cloudlet*,

are deployed in virtual machines by provisioning policy *CloudletSchedulerTimeShared*. Virtual machines are distributed in Datacenters using the *VmAllocationPolicySimple* class.

The class diagram presented in figure 4, shows the classes used to implement MOGA namely NSGAII. The *DatacenterBroker* class from the first diagram is used for linking this group of classes to the other and is used also for running the algorithm.

In this diagram, we have 2 types of classes.

Classes of library for the NSGA-II algorithm that we use: the core of the algorithm is implemented in *NSGA2* class. In this class, we create an instance of *NSGA2Configuration* to specify and store all necessary parameters for genetic algorithm.
The instance of the class *NSGA2Event* is created in each generation during the run of the algorithm. In this class, we store information about the current status of the algorithm. We use this class in *AssignmentNSGA2Listener* to extract information and print it.

Classes that we use to personalize our algorithm: *AssignmentIndividual* class implements the class *Individual*. Its used to describe the populations of genetic algorithm. Each individual present a possible solution of the resource assignment problem.
For each fitness function, we create a specific class which implements *FitnessFunction* interface. Since we have 2 fitness functions, we implement 2 classes. The first one is *EnergyFitnessFunction*, which contains the energy function and the second is *CO2FitnessFunction* which contain CO_2 function.
To observe the performance of our genetic algorithm, we implement the *NSGA2Listener* interface using *AssignmentNSGA2Listener* class. This class shows fitness function values and other detailed data of the best individuals found during the run of the algorithm.

When the genetic algorithm finish, it returns the best found populations which are non-dominated solutions.

5 RESULTS ANALYSIS

In this section, we present the experiments and the evaluation that we undertook in order to study the efficiency of our extension of CloudSim in terms of Cloud computing environments optimization.
We deploy two sets of tests. In the first one, we decide on the value of MOGA parameters. Then, in the next test, we simulate and we analyze the Cloud environment by taking into account the extension of CloudSim and we compare the results with the initial existing approach in CloudSim.

5.1 Parameters of the Algorithm

Genetic algorithms have four parameters. To maximize the efficiency of our algorithm, we have to make a good choice of its parameters values.

Regarding the first parameter which is the size of the population, it is equal to the number of customers applications to optimize. By varying the values of stop criterion of the algorithm, which is the maximum number of generation, we noticed that the best individuals are always found before the 1000th generation. Accordingly, we consider this value as the maximal generation number and as sufficient to find the solution.

In theory, it was found that the values of the parameters of evolutionary algorithms vary in a specific interval. Indeed, former studies (Mais et al.,) have shown that the best results are achieved by a value of crossover probability between 0.45 and 0.95, and a value of mutation probability between 0.01 and 0.005. To fix these two parameters, we kept the same structure of the Clouds environment, the same resources and the same customers applications. Then, we vary the two remaining parameters of MOGA to choose the values that give the best results.

To test the effect of the different values of mutation probability on our Cloud environment, we assume that the value of crossover probability is 0.9 which is a predetermined value. Also for the test of crossover probability, we use the predetermined value of mutation probability which is 0.05.

The evaluation of tests show that the variation of probability values of the two parameters in the theoretical range gives almost stable results. Hence, we keep the predefined value of crossover probability and we choose the upper border of the interval of mutation probabilities. Consequently, values of 0.9 as crossovers probability and 0.01 as mutations probability may give a satisfactory result. We keep these values to simulate the different test cases.

5.2 Simulation Results

The purpose of this experiment is to discuss the performance analysis of our approach compared with static allocation method in terms of the efficiency of resource utilization for the same workload. The static allocation method used is the method deployed in the framework CloudSim before extension.

In these experiments, we calculate 2 metrics. The first one is the total energy consumption by the physical resources of data centers caused by customer applications workloads which is presented in figure 5.

The second one is the total emission of greenhouse gas caused by energy consumption of data centers which is shown in figure 6.

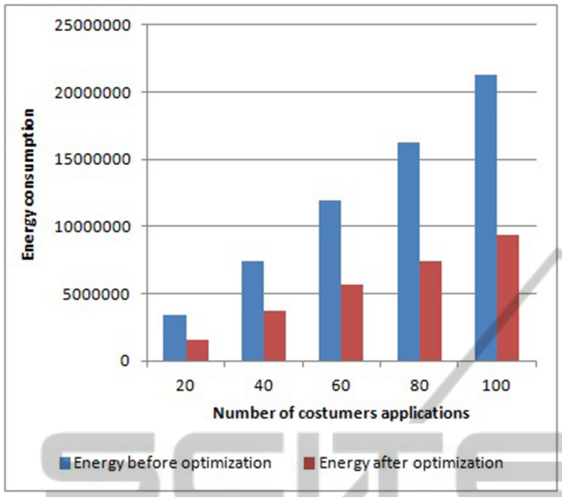

Figure 5: Energy consumption.

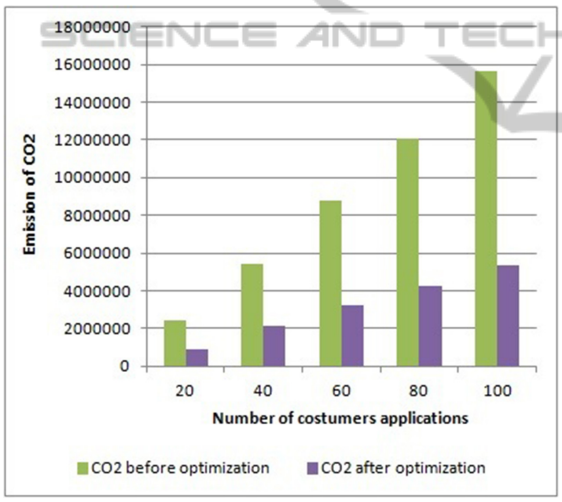

Figure 6: The emission of CO_2.

To demonstrate the amount of this optimization, we calculate the percentage of decrease of energy consumption as well as the emission of CO_2. The result of this demonstration is presented in figure 7.

The experimental results show that our proposed approach provides better results compared to results obtained by static method of resource allocation. Indeed, from these results we can conclude that for the use of our solution on Cloud environment, the energy consumption is reduced to 50% and CO_2 emissions up to 60%. This reduction depends strongly on the characteristics of available resources and the amount of applications which will be run on the Cloud. The obtained results are due to the efficient scheduling of customers applications, and to the reduction of the use of energy-intensive resources.

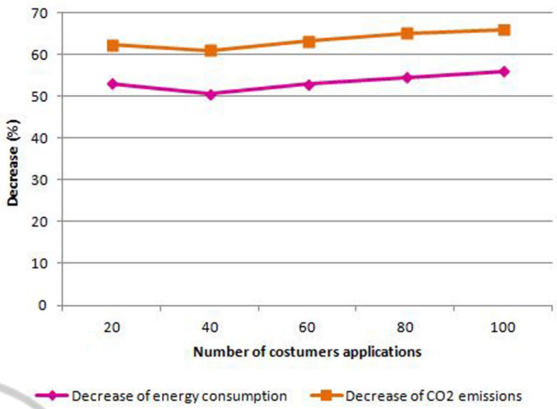

Figure 7: Percentage of decrease.

6 CONCLUSION

To address the problem of energy efficiency, we have proposed in this paper a new approach for a Cloud computing environment that schedule resources allocation based on energy optimization functions. More precisely, the presented work has optimized the resources in the Cloud, and has minimized the rate of energy consumption and CO_2 emissions by the management of Cloud computing resources and the effective choice of our genetic algorithm parameters. The experimental results have shown that the proposed approach leads to a significant reduction in energy consumption and CO_2 emissions in comparison with static techniques of resource allocation.

This study opens more challenges to reduce the impact of new technologies on the environment, encourage the ecosystems, and support energy efficiency.

In future work, we plan to integrate in our project, an agent system in order to make the Cloud environment auto-adaptive. A study is underway in order to fulfill this objective.

REFERENCES

Black, P. Greedy algorithm. dictionary of algorithms and data structures (online), us national institute of standards and technology, february 2005.

Borovskiy, V., Wust, J., Schwarz, C., Koch, W., and Zeier, A. (2011). A linear programming approach for optimizing workload distribution in a cloud. In *CLOUD COMPUTING 2011, The Second International Conference on Cloud Computing, GRIDs, and Virtualization*, pages 127–132.

Buchberger, B. (2001). Gröbner bases: A short introduction for systems theorists. In *Computer Aided Systems TheoryEUROCAST 2001*, pages 1–19. Springer.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1):23–50.

Chaisiri, S., Lee, B.-S., and Niyato, D. (2009). Optimal virtual machine placement across multiple cloud providers. In *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific*, pages 103–110. IEEE.

Chaisiri, S., Lee, B.-S., and Niyato, D. (2012). Optimization of resource provisioning cost in cloud computing. *Services Computing, IEEE Transactions on*, 5(2):164–177.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

Dougherty, B., White, J., and Schmidt, D. C. Model-driven configuration of cloud computing auto-scaling infrastructure.

Dougherty, B., White, J., and Schmidt, D. C. (2012). Model-driven auto-scaling of green cloud computing infrastructure. *Future Generation Computer Systems*, 28(2):371–378.

Goldberg, D. (19 janvier 1996). *Algorithmes génétiques: exploration, optimisation et apprentissage automatique*. Addison-Wesley France, SA, Kluwer Academic Publisher.

Kessaci, Y., Melab, N., Talbi, E.-G., et al. (2011). Optimisation multi-critère pour l'allocation de ressources sur clouds distribués avec prise en compte de l'énergie. In *Rencontres Scientifiques France Grilles 2011*.

Li, Q. (2012). Applying stochastic integer programming to optimization of resource scheduling in cloud computing. *Journal of Networks*, 7(7):1078–1084.

Mais, H.-R., Bloch, C., RAMDANE-CHERIF, W., and Chatonnay, P. Différentes opérateurs évolutionnaires de permutation: sélections, croisements et mutations.

Melcher, J. (11 September 2007). Jnsga2 1.1 - a free java library for the nsga-ii algorithm.

Nair, T. and Jayarekha, P. (2012). Pre-allocation strategies of computational resources in cloud computing using adaptive resonance theory-2. *arXiv preprint arXiv:1206.0419*.

Nguyen Van, H., Dang Tran, F., and Menaud, J.-M. (2009). Autonomic virtual resource management for service hosting platforms. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, pages 1–8. IEEE Computer Society.

Relaxnews (02-04-2010). Data centers et cloud computing mettent trop de co2 selon greenpeace.

Thrash, B. (27-02-2012). The green data center opportunity.

TUAL, M. (14-04-2013). Data centers: la donne colo.

Van, H. N., Tran, F. D., and Menaud, J.-M. (2009). Sla-aware virtual resource management for cloud infrastructures. In *Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference on*, volume 1, pages 357–362. IEEE.

Zhang, Y., Wang, Y., and Wang, X. (2012). Electricity bill capping for cloud-scale data centers that impact the power markets. In *Parallel Processing (ICPP), 2012 41st International Conference on*, pages 440–449. IEEE.

Zhao, H., Pan, M., Liu, X., Li, X., and Fang, Y. (2012). Optimal resource rental planning for elastic applications in cloud market. In *Parallel & Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International*, pages 808–819. IEEE.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS