Power-Aware Virtual Machine Scheduling on Clouds Using Active Cooling Control and DVFS

Daniel Guimaraes do Lago[1], Edmundo R. M. Madeira[2], Luiz Fernando Bittencourt[3]
IC - Institute of Computing – UNICAMP - University of Campinas
Av. Albert Einstein, 1251, Campinas, SP, Brazil - 13083-852
daniellago85@lrc.ic.unicamp.br[1], {edmundo[2], bit[3]}@ic.unicamp.br

ABSTRACT
This paper presents a virtual machine scheduling algorithm for Cloud Computing based on Green Computing concepts. The goal of this algorithm is the minimization of energy consumption in task executions in a cloud computing environment. This algorithm uses some features like shutdown of underutilized hosts, migration of loads of hosts that are operating below a certain threshold, and DVFS. It also applies the concept of active cooling control in order to minimize power consumption. The choice of which host will receive load is based on the concept of higher energy efficiency from the hosts, which is given by the ratio of MIPS by the energy consumed of each host. Results from simulation of this algorithm confronted with other surveyed algorithms have shown that it can improve the power consumption in clouds composed of heterogeneous datacenters while being equivalent to the best algorithms in homogeneous datacenters.

Categories and Subject Descriptors
C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *Cloud Computing*; D.4.1 [**Operating Systems**]: Process Management – *Scheduling*

General Terms
Algorithms, Management, Measurement, Performance, Design, Experimentation, Verification.

Keywords
Cloud Computing, Green Computing, Scheduling, Distributed Power Management, Dynamic Voltage and Frequency Scaling, Fan Control.

1. INTRODUCTION
One of the most notable emerging technologies that acts on the Internet nowadays is the Cloud Computing. This concept refers to the use of processing, memory and storage capacities from shared computers and servers connected through the Internet, as a utility following the principle of grid computing. It can be done at infrastructure, platform, development and service level.

This work aimed to design and evaluate an algorithm for job scheduling in clouds, using the concepts of Green Computing. This algorithm focuses on minimizing power consumption and energy costs in datacenters and, consequently, the potential pollution generated by them. To achieve this, principles of distributed power management, dynamic adjustment of voltage and frequency, load migration, and shutdown of underutilized hosts were used.

Power-aware dynamic application placement can address underutilization of servers as well as the rising energy costs in a datacenter [1]. In order to improve energy saving, this algorithm also introduces the use of active cooling control to minimize power consumption, originally used for noise reduction. Furthermore, the algorithm is based on the efficiency of consumed energy, trying to allocate virtual machines on the host with the best ratio between MIPS and power consumed, which is actually useful in heterogeneous datacenters.

This paper is organized as follows: Section 2 describes some related works, and in Section 3 key concepts of the algorithm are presented. In Section 4 we show how the algorithm works, followed by simulation results in Section 5. Conclusions and suggestions for future work are presented in Section 6.

2. RELATED WORK
Scheduling in clouds can be separated into two main views: from the cloud user and from the cloud provider. From the users' view, a scheduler may have objectives such as minimizing both the execution time and user's budget [2]. On the other hand, from the cloud provider view, a workload scheduler may need to improve resource utilization while reducing maintenance and energy consumption costs [1]. This work focus on scheduling from the cloud provider point of view.

Beloglazov et al. [3] propose a resource management system for efficient power consumption that reduces operating costs and provides quality of service. Energy saving is achieved through the continued consolidation of virtual machines according to resource utilization, virtual network topologies established through the virtual machines, and temperature states of the nodes. Simulation results show 83% of saving in comparison to a non power aware system and 63% comparing to a system that applies only DVFS.

Duy et al. [4] design, implement, and evaluate a scheduling algorithm integrating a predictor of neural networks to optimize the power consumption of servers in a cloud. The prediction of future workload is based on demand history. According to the prediction, the algorithm turns off unused servers and restarts them to minimize the number of servers running, thus also minimizing energy use. Simulations with two workloads found that this model is able to reduce up to 46% energy consumption in comparison to a non power aware system.

Berl et al. [5] study and review the use of methods and energy-efficient techniques in the context of cloud computing, especially

in the field of hardware and network infrastructure. They propose the installation of specific plug-ins and energy control centers for large-scale networks, achieving significant impact on the cloud. It reduces energy consumption in software and hardware, improving load balancing and communication energy consumption.

Garg et al. [6] propose a nearly optimal scheduling algorithm with policies that exploits the heterogeneity between multiple datacenters in a cloud provider. Several factors are considered in energy efficiency (such as energy cost, rate of carbon emissions, load, and power efficiency of processor), while switching between different datacenters according to their location, architectural design, and system administration. The authors state that the proposed scheduling policies help to achieve savings of up to 25% compared with the existing policies.

Binder and Suri [7] propose an algorithm of allocation and dispatch of tasks that minimizes the number of active servers required, managing to achieve an energy consumption which is inversely proportional to the number of concurrent threads running in workloads.

Srikantaiah et al. [8] study how to obtain consolidation of energy efficiency based on the interrelationship among energy consumption, resource utilization, and performance of consolidated workloads. It is shown that there is an optimal point of operation among these parameters based on the Bin Packing problem applied to the problem of consolidation.

Nathuji et al. [9] and Hu et al. [10] use virtual machines with the objective of reducing energy consumption in virtualized environments. in virtualized enterprise systems. In [1], Dasgupta et al. pose workload normalization across heterogeneous systems, such as clouds, as a challenge to be addressed.

As shown above, many initiatives have been taken in many ways to make datacenters operating in a cloud more environment-friendly. The main differences from the algorithm proposed in this work to other ones are the focus on heterogeneous datacenters and energy benefits provided by active cooling control for unknown sized workload processing.

3. BASIC CONCEPTS

In this section we present some basic concepts regarding virtual machines and energy consumption.

3.1 Virtual Machine Migration

Migration of virtual machines seeks to improve manageability, performance, and fault tolerance of systems. More specifically, the reasons that justify VM migration in a production system include: the need to balance system load, which can be accomplished by migrating VMs out of overloaded/overheated servers; and the need of selectively bringing servers down for maintenance after migrating their workload to other servers [11].

In the algorithm proposed in this work, we use the concept of migration of virtual machines in order to migrate the virtual loads of underutilized hosts and then turn them off, thereby maximizing energy savings.

3.2 Dynamic Voltage and Frequency Scaling

Dynamic Voltage Scaling (DVS) [12] is a technique of power management in a computer architecture where the tension of a particular component can be changed according to some parameters. Manufacturers of processors benefit from DVS features to minimize power consumption of their devices and also cutting the frequency of such equipment to enable them to remain stable.

Intel Speed Step technology or AMD Coll'n'Quiet automatically adjusts, in hardware, the operating voltage and frequency of their processors according to their workloads. If there are high loads of work, the processor raises the levels of voltage and frequency, otherwise these levels decrease. The set of possible states of frequency and voltage is called P-States. Although the processors come with such technologies, their support is usually disabled on the motherboard, requiring attention from the system administrator to activate them. This is the Dynamic Voltage and Frequency Scaling concept.

In the proposed algorithm the concept of DVFS is always applied, with the goal of having the intelligent use of the processor power, adapted to the loads that it needs to process.

3.3 Fan Control

Fan Control is one of the given names to represent the technology that monitors the temperature of the computers and, through thermal conditions, regulates the intensity of active coolers. Several motherboard manufacturers have several modes of operation of this technology, each one with their own names, for example: AOpen: SilentTEK; ASUS: Q-Fan; MSI: Core Center; Abit: µGuru; Gigabyte: EasyTune 6; Intel S478: Active Monitor / Desktop Control Center; Intel S775: Desktop Utilities and Dell Inspiron/Latitude/Precision: I8kfanGUI.

Some of these technologies control the rotation speed of the fan through the operating system or directly from the BIOS, but they all have a similar behavior: control of cooler intensity according to the processor heating.

Although this technology was originally designed to reduce noise of active heatsink, it has a direct consequence: the reduction of energy consumption. For example, when a fan is powered by 12 volts and becomes powered by 5 volts, it will consume 83% less power. In this work we used this mechanism along with DVFS to optimize energy consumption.

4. THE PROPOSED ALGORITHM

The algorithm proposed in this work is named Lago Allocator. It is designed to handle computing clouds consisting of non-federated datacenters, so it is assumed that all centers are in the same cloud. It also considers that the DVFS and Fan Control are enabled for all hosts.

This algorithm has in its scope datacenters with homogeneous or heterogeneous machines. Also, it does not need knowledge about the workload weight that will be allocated to virtual machines in the datacenter: it dynamically checks the processing needs and optimizes energy consumption on-the-fly.

In a datacenter operating in a cloud, one of the servers must perform the function of a broker, which deals with events like resource requests, virtual machine creation, workload return, and finalization of workloads and virtual machines.

The main acting point of the Lago Allocator is on the host allocation for virtual machines. This allocation is done based on the policy of minimizing energy consumption. Before presenting the algorithm we describe some functions used by it.

The *getCurrentMIPSUtilization* function returns the current MIPS utilization of a host. The *getMIPS* function returns the maximum

MIPS utilization of a virtual machine or a host. The available MIPS of a host can be obtained by subtracting its processing entities MIPS from the sum of the MIPS of the VMs that are allocated to it.

The *getMaximumPower* function returns the maximum power of a host, composed of its base power (motherboard, disks, RAM, etc) plus the processor power at maximum usage (DVFS off) plus the maximum fan power usage (Fan Control off). The *getPower* function returns the current power usage of a host, which is result from its base power plus the processor current power (with DVFS on) plus the current fan power usage (with Fan Control on). Since it depends on the platform, it should be implemented accordingly.

The *getPowerAfterAllocation* function returns the estimated power of a host after an allocation of a virtual machine. *getUtilizationOfCpu* returns the current CPU use percentage and *getTotalMips* returns the total MIPS a CPU supports. In a linear power model the *getPowerAfterAllocation* function can be defined in equation (1), where *paa* is the power consumption after virtual machine allocation, h_{sp} is the host static power, $h_{max\ pow}$ is the maximum host power consumption, $MIPS_c$ is the host current MIPS in use, $MIPS_{vm}$ is the virtual machine MIPS consumption, and $MIPS_{max}$ is the total host MIPS.

$$paa = h_{sp} + (h_{max\,pow} - h_{sp}) \times \frac{MIPS_c + MIPS_{vm}}{MIPS_{max}} \quad (1)$$

The Lago Allocator algorithm is defined as follows:

Lago Allocator Algorithm: findHostForVm(vm)

```
1. bestEfficiency = -∞
2. bestHost = null
3. foreach host in datacenter do
4.   utilization = getCurrentMIPSUtilization(host) +
     getMIPS(vm)
5.   if (utilization < getMIPS(host)) then
6.     efficiency = getMIPS(host) / getMaximumPower(host)
7.     if (efficiency > bestEfficiency) then
8.       bestEfficiency = efficiency
9.       bestHost = host
10.    else if ( efficiency == bestEfficiency) then
11.      pw_vm_at_host = getPower(bestHost) +
         getPowerAfterAllocation(host, vm)
12.      pw_vm_at_bestHost = getPower(host) +
         getPowerAfterAllocation(bestHost, vm)
13.        if (pw_vm_at_host < pw_vm_at_bestHost) then
14.          bestHost = host;
15.        else if (pw_vm_at_host == pw_vm_at_bestHost) then
16.          if (getUtilizationOfCpu(host) >
             getUtilizationOfCpu(bestHost)) then
17.            bestHost = host
18.          else if (getUtilizationOfCpu(host) ==
             getUtilizationOfCpu(bestHost)) then
19.            if (getTotalMips(host) >
               getTotalMips(allocatedHost)) then
20.              bestHost = host
21.            end if
22.          end if
23.        end if
24.      end if
25.   end if
26. end foreach
27. return bestHost
```

For each virtual machine to be allocated in the datacenter it is checked on all hosts which are those that can receive the incoming virtual machine. The primary condition for a host to receive a virtual machine is having sufficient MIPS available for the VM to be allocated (line 5). For each candidate host, the efficiency of energy that it consumes is verified (line 6). The algorithm ranks the host that is more energy efficient to allocate the virtual machine (line 7).

A tie can occur, especially in homogeneous environments. A tiebreaker comparison is made as follows: since the system has DVFS enabled, along with Fan Control, there will probably exist some hosts consuming less energy than others. So, hosts are confronted against each other and the lesser energy consumer is chosen to allocate the virtual machine (line 13).

Yet there may be a tie, for example if there are two or more machines of equal power consumption that still have not received virtual machines. Thus another tiebreaker is performed: the host with the highest CPU utilization (under 100%, as checked on line 5) is chosen. This choice is based on the fact that if a CPU is processing more load, it will take longer to reach its virtual load migration threshold (line 16). If there is still a tie, the host with more processing power is chosen, since this is potentially able to receive a greater number of workloads (line 19).

Once all the virtual machines have been created, the loads are submitted for processing. If there are not enough virtual machines available for processing such loads, the loads that could not be processed will be delayed. With all loads submitted, the algorithm checks the hosts and turns off virtual machines which have completed the processing of their loads.

At this point the algorithm checks which hosts are being utilized below a defined threshold and tries to migrate the virtual machines from these hosts to other ones, and then shutdown the underutilized hosts. The choice of target host in the migrations is done with the same algorithm.

The optimal value of threshold may be difficult to calculate. There are academic researches on which would be the best way to calculate this threshold. Mukherjee and Sahoo [14], for example, propose the use of a honeybee colony system to calculate good thresholds for virtual loads migrations. As future work we consider adding techniques like this to implement adaptive thresholds for our algorithm.

5. SIMULATION

To achieve an efficient simulation that addresses various scenarios, the choice of a robust simulator is essential. The research conducted in this work indicated that the best simulator for this purpose would be *CloudSim* [13] (version 2.2.1).

5.1 Simulation Rules

For evaluation purposes, the relevant metrics of the simulation are the cloudlets completion time and energy consumption. In CloudSim, a cloudlet represents a task that is submitted to a datacenter virtual machine.

To make appropriate comparisons between the algorithms in this paper, some definitions are presented: *A* represents a cloud scheduling algorithm; *C* represents a set of configurations which an algorithm uses, namely: shutdown of unused hosts, migration of virtual loads, DVFS, and Fan Control; and *P* represents a set of simulation parameters: number of hosts, virtual machines number, and millions of instructions of each cloudlet.

EC(A, C, P) is the average energy consumption of algorithm *A* with the set of configurations *C* and parameters *P; ECI(A, C, P)* is the 95% confidence interval of *EC(A, C, P). TC(A, C, P)* is the average makespan time of algorithm *A* with the set of configurations *C* and parameters *P*; and *TCI(A, C, P)* is the 95% confidence interval of *TC(A, C, P)*.

Each configuration was simulated 30 times. An algorithm A_1 with C_1 settings and simulation parameters *P* is considered better in energy consumption than algorithm A_2 with C_2 settings and parameters *P*, $A_1 \neq A_2$, if the inequality shown in (2) is true.

$$EC(A_1, C_1, P) + ECI(A_1, C_1, P) < EC(A_2, C_2, P) - ECI(A_2, C_2, P) \quad (2)$$

An schedule given by an algorithm A_1 is considered intolerably slower than A_2 for simulation parameters *P* if for the same set of configurations *C* and simulation parameters *P* the inequality (3) is satisfied.

$$TC(A_1, C, P) + TCI(A_1, C, P) > 2 \times (TC(A_2, C, P) - TCI(A_2, C, P)) \quad (3)$$

5.2 Datacenter Configuration

A datacenter is a physical set of machines connected by a network available to receive the virtual machines and workloads accordingly (cloudlets).

The simulations to evaluate the Lago Allocator algorithm were conducted with small, medium, and large datacenters having homogeneous and heterogeneous hosts. For small datacenters, we considered that they have 10 hosts, 20 virtual machines, and receive 20 cloudlets. In medium sized datacenters, these numbers are 100 hosts, 200 virtual machines, and 200 cloudlets. In large datacenters these values are 1000 hosts, 2000 virtual machines, and 2000 cloudlets.

5.3 Common Hosts Configuration

It is defined that the usual settings of the regular host to homogeneous and heterogeneous are as follows: Each host has 1 processing entity (the processor), 1 TB of disk space, 24 GB of RAM and gigabit ethernet. It is assumed that the hosts are running on a datacenter with x86 architecture, Xen as virtual machine monitor, and Linux as OS.

Regarding cloudlets, each virtual load uses one processing entity. Also, each cloudlet submitted to the datacenter, before processing, has 300 bytes (standard of CloudSim models) and 300 bytes of data after the processing (standard of CloudSim models). Each cloudlet in a datacenter has 10,000, 15,000, 20,000 and 25,000 Millions of Instructions in a round-robin distribution.

The datacenter virtual machines have processing capabilities of 500, 750, and 1000 MIPS in a round-robin distribution. For example, in a simulation where 20 virtual machines are generated, 7 VMs are created with 500 MIPS, 7 VMs with 750 MIPS and 6 VMs with 1000 MIPS. Moreover it was adopted that each virtual machine has 128 MB of RAM and 2,500 Kbps of bandwidth. Each virtual machine has 2,500 MB of image size and Xen as virtual machine monitor.

In the simulations with DVFS, it was assumed that when the host is working at minimum level of processing, it consumes 70% of its maximum power, and the growth of its load adopts a linear model. For example, when a 250 watts host is working at 0% capacity it consumes 175 watts; when working at 50% capacity it consumes 212 watts, and when working at full capacity it consumes 250 watts. For the simulations with activated threshold for virtual machines migration, we try to migrate VMs when the hosts are operating below 80%, followed by their shutdown.

For the simulations, we based on the Intel i5 750 stock cooler which works with 0.6 amps and maximum voltage of 12 volts. In other words, it consumes 7.2 watts of power. Although active cooling mechanisms allow the complete shutdown of the cooler under good circumstances, we decided to keep a safety margin to guarantee that the simulation is valid, so the fan never turns off. Thus we adopt the minimum consumption of the fan when it is at low speed, at a voltage of 5 volts.

The warming of the processors depends on many factors, including temperature, sink type, architecture, and lithography. Given that there is no pattern of warming among the various existing processors, a model that is valid for all of them is very difficult to estimate. Since the processor temperature is directly proportional to the power dissipated and that grows proportionally to the frequency of the processor and the square of its operating voltage, it was adopted the cooler power consumption is proportional to the power consumed by the processor with DVFS enabled, adopting a linear model for this.

5.3.1 Settings for Homogeneous Datacenters

For the simulations in homogenous datacenters, the following settings were adopted: each host in a homogeneous datacenter has one processing entity (processor containing one core) with 2,000 MIPS; and each host power consumption is set to 250 watts.

5.3.2 Settings for Heterogeneous Datacenters

For the simulations in heterogeneous datacenters, the following settings were adopted: each host in a heterogeneous datacenter has a single processing entity, containing the following amount of MIPS in a round-robin distribution: 1000, 1500, 2000, 2500, 3000; and each host in a heterogeneous datacenter has the following power consumption in a round-robin distribution: 200, 250, 300 and 350, as exemplified in Table 1.

Table 1: Example of Initial Configuration of hosts in a heterogeneous datacenter

Host	***h1***	***h2***	***h3***	***h4***	***h5***	***h6***	***h7***
MIPS	1000	1500	2000	2500	3000	1000	1500
Power	200	250	300	350	200	250	300

5.4 Algorithms and Settings

To validate the proposed algorithm, the results of its simulation were compared to the results of two literature classic scheduling algorithms – Round Robin and Best Resource Selection – and the best algorithm focusing on power management on CloudSim – Minimum Power Diff.

In the Round Robin algorithm, each virtual machine is allocated in a different host, making a cycle. Hosts that can not allocate virtual machines are skipped. When there are no more hosts capable of receiving virtual machines, the allocation will be delayed. In the Best Resource Selection algorithm, the host with the highest ratio (MIPS in use) / MIPS, and currently supporting virtual machines coming, is allocated. This ensures minimization of migrations and tends to produce results faster. Minimum Power Diff, implemented by Beloglazov et al., is used as the primary model of energy savings in CloudSim. Each incoming virtual

machine is allocated to the host which will consume less energy to execute the VM. The chosen algorithms have been implemented using different combinations of the following features: Power Aware (ability to turn off unused hosts); DVFS (Dynamic Frequency and Voltage Scaling treating the P-States as a linear model where the unused host consumes 70% of its power and fully used host consumes 100%); Threshold (migration of virtual loads trying to keep the processing entities of the hosts with at least 80% activity); and Fan Control (proposed in this work, to use the fan control with linear power consumption proportional to the DVFS).

5.5 Simulation Results

Thirty simulations were performed for each algorithm for each of the settings mentioned in the previous section. In this section acronyms are used to describe algorithms and their settings used for each set of simulations: RR = Round Robin; BRS = Best Resource Selection; MPD = Minimum Power Diff; LA = Lago Allocator. Acronyms for algorithm settings: N = Non Power Aware; P = Power Aware; PD = Power Aware and DVFS; PDT = Power Aware, DVFS and Threshold; PDTF = Power Aware, DVFS, Threshold and Fan Control; PT = Power Aware and Threshold. For all values of energy measures presented in this section the metric is kilowatts * hour, and for measurements of time, seconds.

As shown in Figures 1, 3 and 5, the best algorithms in power consumption are the MPD and LA, nearly tied. However, the Figures 2, 4 and 6, identify a slight better performance of LA.

Table 2 presents a comparison between the best competing algorithm, the MPD with the PDT settings, compared to the algorithm proposed in this work, the LA with the PDTF settings. Even running the MPD with Fan Control mechanisms used by the LA, results suggest that LA performs better in large-scale heterogeneous environments, as shown in Table 3.

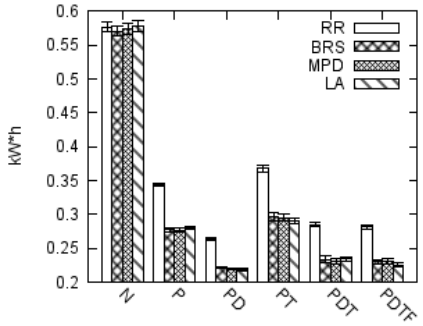

Figure 1: Small Homogeneous Datacenter Power Consumption

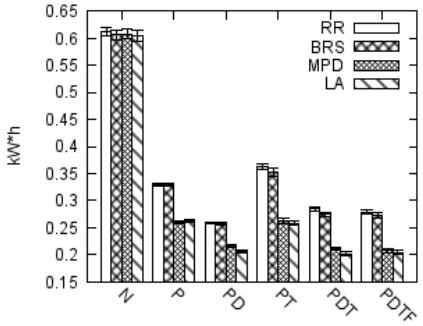

Figure 2: Small Heterogeneous Datacenter Power Consumption

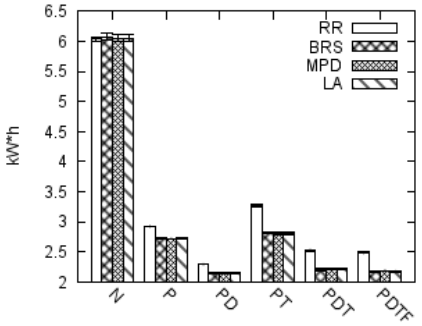

Figure 3: Medium Homogeneous Datacenter Power Consumption

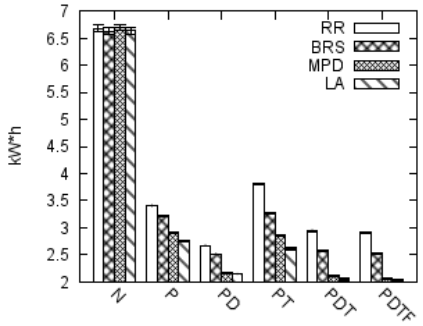

Figure 4: Medium Heterogeneous Datacenter Power Consumption

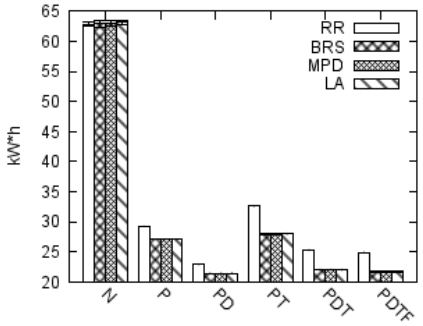

Figure 5: Large Homogeneous Datacenter Power Consumption

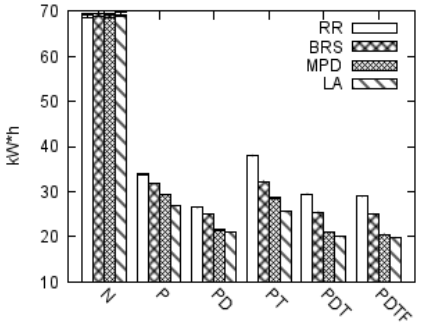

Figure 6: Large Heterogeneous Datacenter Power Consumption

Table 2: Competitive Comparison Between Minimum Power Diff PDT and Lago Allocator PDTF

Scenario	*MPD PDT-ECI* *MPD PDT+ECI*	*LA PDTF-ECI* *LA PDTF+ECI*	*Conclusion*
Small Homo	0.227	0.221	***Tie***
	0.234	0.229	
Small Hetero	0.209	0.201	***LA is better (0.5% to 7%)***
	0.215	0.208	
Medium Homo	2.204	2.163	***LA is better (1% to 3.1%)***
	2.229	2.183	
Medium Hetero	2.091	2.007	***LA is better (2.8% to 5.3%)***
	2.114	2.034	
Large Homo	21.982	21.674	***LA is better from (1.1% to 1.8%)***
	22.061	21.751	
Large Hetero	20.890	19.770	***LA is better (5.3% to 6.1%)***
	20.974	19.834	

The proposed algorithm shows improvement in almost all scenarios compared to the MPD PDT. Only in small and homogeneous datacenter there is no clear significant improvement. This indicates that the proposed algorithm with the proposed settings performs better than the other ones evaluated. The best result occurs for the case of a large heterogeneous datacenter, where this improvement can reach 6.1%. This progress suggests that the greater the datacenter, the greater the difference between LA and MPD. Even in the scenario where the algorithm proposed in this work is faced with the same settings (PDTF) in MPD, the LA has best performance, especially in the case of heterogeneous datacenters.

Table 3: Competitive Comparison Between Minimum Power Diff PDTF and Lago Allocator PDTF

Scenario	*MPD PDTF-ECI MPD PDTF+ECI*	*LA PDTF-ECI LA PDTF+ECI*	*Conclusion*
Small Homo	0.225	0.221	***Tie***
	0.234	0.229	
Small Hetero	0.204	0.201	***Tie***
	0.211	0.208	
Medium Homo	2.172	2.163	***Tie***
	2.192	2.183	
Medium Hetero	2.035	2.007	***LA is better (0.05% to 3%)***
	2.066	2.034	
Large Homo	21.649	21.674	***Tie***
	21.720	21.751	
Large Hetero	20.449	19.770	***LA is better (3.1% to 4%)***
	20.558	19.834	

Considering both tables, it is clear that LA is a competitive algorithm in homogeneous datacenters and better in heterogeneous datacenters. In addition, simulations with homogeneous cloudlets (15,000 MI only) on homogeneous datacenters showed similar behavior when compared to the same datacenter type running heterogeneous cloudlets.

In the simulations carried out, LA was never considered unacceptably slower than any other algorithm. As expected, the executions were faster in a not power aware environment. LA PDTF remained between 15% (in the simulations with heterogeneous small datacenters) and 43% (in the simulations with homogeneous large datacenters) slower than the best algorithm in each of these simulations, respecting the imposed rules of acceptable execution times. When compared to the Minimum Power Diff, LA provided average makespan around 3% smaller, thus improving the cloudlets execution times besides minimizing the energy consumption.

Regarding the number of virtual machines migrations, the LA PT algorithm presented 29.6±0.52, 25.67±0.98, 308.7±1.23, 307.76±1.07, 3114.43±4.05, 3115.87±4.32 migrations in Small Homo, Small Hetero, Medium Homo, Medium Hetero, Large Homo, and Large Hetero scenarios, respectively (considering a 95% confidence interval). Moreover, when compared to LA P, the LA PT presented an average time increment of 29%, 13%, 43%, 43%, 77%, and 76%, for the same scenarios mentioned above – caused exclusively by virtual machines migrations.

6. CONCLUSION

The inefficient use of servers in a datacenter leads to high energy costs, expensive cooling hardware, floor space, and an adverse impact on the environment [1]. In this work we presented a scheduling algorithm for cloud computing based on green computing concepts capable of performing the task scheduling in non-federated datacenters, homogeneous and heterogeneous, with sizes of workloads known or not.

Results obtained in the simulations showed that the proposed algorithm competes with the best surveyed algorithms in homogeneous datacenters, being able to overcome them in cases of large-scale heterogeneous datacenters. Also, the results indicate that the presented algorithm performs better with heterogeneous and large datacenters. Moreover, the results showed that the use of active cooling control in the datacenter generates a small but relevant improvement.

It is proposed as future work to adapt the presented algorithm in order to be able to deal with federated datacenters. The algorithm can also improve if it is possible to measure the size of the virtual loads to be processed by applying heuristics, modeled as studies in the bin-packing problem. In addition, techniques to automatically optimize the threshold value to perform virtual machines migration, aiming to make the cloud more energy efficient, can be developed.

7. ACKNOWLEDGMENTS

We would like to thank FAPESP (2010/14592-9 and 2009/15008-1) and CNPq. Thanks to Anton Beloglazov (Department of Computer Science and Software Engineering, University of Melbourne, Australia) for the quick answers in discussions about CloudSim.

8. REFERENCES

[1] DASGUPTA, G.; SHARMA, A.; VERMA, A.; NEOGI, A.; KOTHARI, R. “Workload Management for Power Efficiency in Virtualized Datacenters”. Communications of the ACM 131-141, Volume 54, No. 7, July 2011.

[2] BITTENCOURT, L. F; MADEIRA, E. R. M. “HCOC: A Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds”. Journal of Internet Services and Applications (in press). 2011.

[3] BELOGLAZOV, A.; BUYYA R. “Energy Efficient Resource Management in Virtualized Cloud Data Centers”. 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing. 2010.

[4] DUY, T. V. T.; INOGUCHI, Y. S. Y. “Performance Evaluation of a Green Scheduling Algorithm for Energy Savings in Cloud Computing”. 2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPSW). 2010.

[5] BERL, A.; GELENBE, E.; GIROLAMO, M.; ET AL. “Energy-Efficient Cloud-Computing”. University of Passau. 2009.

[6] GARG, S. K; YEO, C. S.; ANANDASIVAM, A.; BUYYA, R. “Environment-conscious scheduling of HPC applications on distributed Cloud-oriented data center”. Journal of Parallel and Distributed Computing, 732-749, Volume 71, No. 6, Elsevier, 2011.

[7] INDER, W.; SURI, N. “Green Computing: Energy Consumption optimized service hosting”. SOFSEM '09: Proceedings of the 35 Conference on Current Trends in Theory and Practice of Computer Science, 117-128. Berlin, Heidelberg. 2009.

[8] SRIKANTAIAH, S.; KANSAL A.; ZHAO F. “Energy Aware Consolidation for Cloud Computing”. Microsoft Research, 2008.

[9] NATHUJI, R.; SCHWAN, K. "VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems". SOSP'07. Stevenson, Washington, USA. October, 2007.

[10] HU, L.; JIN, H.; LIAO, X.; XIONG, X.; LIU, H. “Magnet: A Novel Scheduling Policy for Power Reduction in Cluster with Virtual Machines”. 2008 IEEE International Conference on Cluster Computing, 13-22. 2008.

[11] VOORSLUYS, W.; BROBERG, J.; VENUGOPAL, S.; BUYYA, R.: “Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation”. Cloud Computing Lecture Notes in Computer Science, 2009, Volume 5931/2009, 254-265.

[12] RABAEY, J. M. “Digital Integrated Circuits”. Prentice Hall, 1996.

[13] BUYYA, R.; RANJAN, R.; CALHEIROS, R.N. “Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities”.International Conference on High Performance Computing & Simulation, 2009. HPCS '09. August 2009.

[14] MUKHERJEE, K.; SAHOO, G. “Green Cloud: An Algorithmic Approach”. International Journal of Computer Applications (0975-8887). Volume 9, No 9, November 2010.