UNIVERSIDAD DE SEVILLA

DEPARTAMENTO DE LENGUAJES Y SISTEMAS INFORMÁTICOS

Energy and performance-aware scheduling and shut-down models for efficient Cloud-Computing data centers

International Doctoral Dissertation presented by

D. Damián Fernández Cerero

advised by

Dr. Alejandro Fernández-Montes González

and **Dr. Juan Antonio Ortega Ramírez.**

September 2018.

A mis padres.

Agradecimientos

Quisiera expresar mi más sincero agradecimiento a mis directores de tesis, Dr. D. Juan Antonio Ortega Ramírez y Dr. D. Alejandro Fernández-Montes González, por toda la paciencia, esfuerzo y tiempo dedicado a la elaboración de este trabajo. Han conseguido que, gracias a su apoyo y a la vez exigencia, evolucione, no sólo como investigador y profesional, sino como persona. En especial a Ale, que siempre tuvo fe en mí y cada día de este largo camino ha conseguido no dejar de ser a la vez jefe, compañero y amigo.

De la misma manera, no puedo olvidar al Dr. D. Juan Antonio Álvarez, al Dr. D. Francisco Velasco Morente y al Dr. D. Luis González Abril por los buenos momentos, sabios consejos y experiencia que me han brindado, así como la inestimable sabiduría y comprensión que siempre muestran.

A mis padres Natividad y Juan José, quienes han sido los responsables de poder alcanzar esta meta, al igual que todas las demás en mi vida. Siempre os estaré agradecido, y siempre seréis el pilar fundamental, aunque no siempre pueda veros. Del mismo modo, a mi hermano.

A todas las personas que me han acompañado en este camino. Aunque la vida nos separe en ciertas ocasiones, siempre estaréis conmigo.

A todos mis compañeros del departamento de Lenguajes y Sistemas Informáticos, por vuestros apoyos, ánimos y guía. A mis compañeros de despacho Andrés, Bedilia y Jesús por soportarme y compartir tantas horas, charlas y risas. También a los compañeros de despacho de Alejandro, Juan Antonio e Isabel, por aguantar con inagotable paciencia mis innumerables allanamientos de despacho.

A los colegas de Cracovia, quienes han hecho que tenga un segundo hogar y han encauzado el desarrollo de esta tesis. También a los compañeros del Laboratoire de l'Informatique du Parallelisme del ENS de Lyon por apoyarme en la fase temprana del desarrollo de este trabajo.

A todos, muchas gracias.

CONTENTS

II Selected Research Papers

III Final remarks

LIST OF FIGURES

ABSTRACT

This Doctoral Dissertation, presented as a set of research contributions, focuses on resource efficiency in data centers. This topic has been faced mainly by the development of several energy-efficiency, resource managing and scheduling policies, as well as the simulation tools required to test them in realistic cloud computing environments.

Several models have been implemented in order to minimize energy consumption in Cloud Computing environments. Among them: a) Fifteen probabilistic and deterministic energy-policies which shut-down idle machines; b) Five energy-aware scheduling algorithms, including several genetic algorithm models; c) A Stackelberg game-based strategy which models the concurrency between opposite requirements of Cloud-Computing systems in order to dynamically apply the most optimal scheduling algorithms and energy-efficiency policies depending on the environment; and d) A productive analysis on the resource efficiency of several realistic cloud–computing environments.

A novel simulation tool called SCORE, able to simulate several data-center sizes, machine heterogeneity, security levels, workload composition and patterns, scheduling strategies and energy-efficiency strategies, was developed in order to test these strategies in large-scale cloud-computing clusters. As results, more than fifty Key Performance Indicators (KPI) show that more than 20% of energy consumption can be reduced in realistic high-utilization environments when proper policies are employed.

PART I

Preface

CHAPTER 1

INTRODUCTION

Today's scientists have substituted mathematics for experiments, and they wander off through equation after equation, and eventually build a structure which has no relation to reality.

Nikola Tesla

1.1 Research Motivation

Cloud computing and large-scale web services have transformed computer cluster and big-data environments, which have led to a new scenario where these infrastructures are as energy greedy as many factories. The latest estimations consider that data centers account for approximately 1.5% of global energy consumption [1]. In Figure 1.1 the evolution of data-center energy consumption is shown. It can be noticed that the growth of such energy consumption has decreased thanks to the application of various energy-efficiency models.

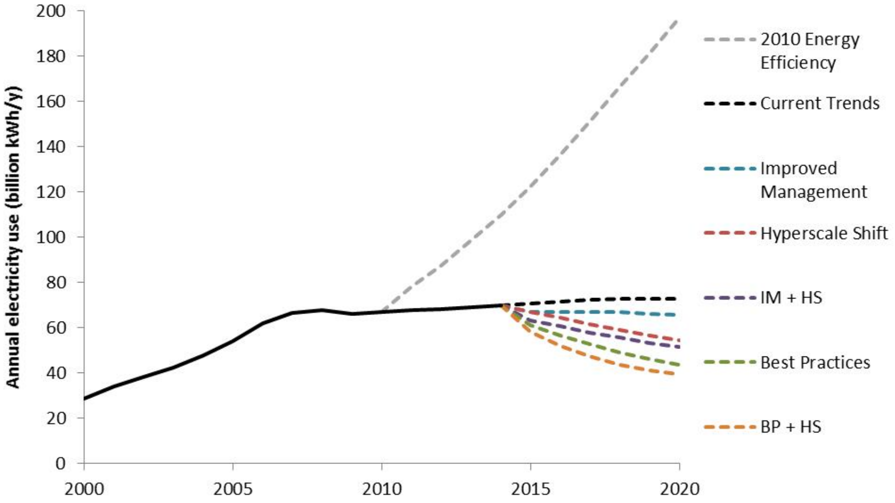


Figure 1.1: Data-Center Energy consumption evolution

The evolution of cloud computing and big data services has enabled the industry to process huge amounts of data in a reliable and distributed way; however, fast-response and low latency are also needed in this late stage of cloud computing.

Several actors have made improvements in particular subsystems or frameworks, such as: parallel and distributed algorithms; distributed file systems; resource managers; and execution engines. Such developments have often resulted in a fragmented and heterogeneous software environment whose complexity is constantly rising.

Many of these improvements offer various vertical all-in-one solutions to solve each problem, others build new generalist solutions over de-facto standard systems, such as Hadoop Distributed File System [2], YARN [3], and Spark [4]. This mix of solutions forces system administrators to use various technologies or even multiple stacks of technologies. In many cases, there is no compatibility between them. Thus, a wrong architectural decision may cause business-critical negative impact.

According to Koomey Law [5], every new server generation provided higher computing power since the 1950s. Such an increase made resource-efficiency strategies not critical. However, the progression of the computing power is not as fast as it was in the past, and it is limited by Margolus - Levitin theorem [6]. Such a limitation forces data-center industry to adopt new strategies to fulfill the ever-increasing computing requirements while maintaining data-center operation costs [7].

Data centers do not utilize the same amount of resources at any time, which leads to many servers remaining idle during low-utilization periods. Software systems should be able to make energy-aware scheduling decisions in order to achieve energy proportionality while maintaining Service Level Objectives (SLO).

In addition, energy-proportionality models should not be focused only on specific frameworks or subsystems, since data-center workload is constantly evolving. This problem led certain researchers to shift their focus towards the creation of an evolution of the resource managers: a distributed data-center operating system [8] which could manage the resource utilization of every subsystem at a higher level instead of per-framework basis. The evolution of resource schedulers to a kind of data-center operating system enables power proportionality to be achieved.

Since a wide range of frameworks are deployed on the same group of resources in these systems, energy-efficiency efforts must focus on the core component of the system - the resource manager, or data center operating system -, instead of on each framework separately.

The described drawbacks in terms of heterogeneity, and requirements in terms of resource efficiency, make necessary the development of energy and performance-aware models to achieve higher energy proportionality. Such models are to be applied to the high-level resource manager layer in large-scale realistic scenarios.

1.2 Research Methodology

This research follows the standard scientific research technique [9] which includes the following phases:

1. **Study and analysis of the current state of the art.** This first step helped to clearly define the research question and motivate the novelty of the work, since already-existing models, solutions and data were analyzed.

2. **Novel theoretical solutions.** In this phase new theoretical models which can help to answer the research question are developed.

3. **Implementation of the theoretical solutions.** Several algorithms, policies and techniques have been developed to empirically test the theoretical models. Other models have been developed to rigorously compare the proposed solutions.

4. **Simulation and empirical analysis.** The required simulation tools which implement the proposed models have been developed to correctly test and analyze the results provided.

1.3 Research Question

The research question that leads this thesis dissertation is:

Which are the best strategies to reduce energy consumption in realistic large-scale Cloud-Computing clusters with no notable negative impact on cluster performance?

1.4 Research Objectives

The objectives of this thesis dissertation aim to concretely contribute to the development of energy-aware cloud computing data centers by answering the research question in many areas. Among them:

- Proof that fear to the application of energy-efficiency policies that shut down underutilized machines should be overcome in order to achieve higher efficiency levels.

- Proof that a simulation tool able to simulate large-scale data centers with high performance can be built to trustfully test the models proposed. This simulation tool may include several energy efficiency policies, scheduling algorithms, resource managers and workloads.

- Proof that energy consumption in monolithic-scheduling data centers can be successfully reduced without notably impacting performance if the correct set of energy-efficiency policies based on the shut-down of idle machines are applied.

- Proof that genetic algorithms are an excellent solution to efficiently distribute tasks among servers in data centers taking into account performance, energy, and security restrictions.

- Proof that models based on games theory, such as the Stackelberg model, are an excellent choice to successfully model the concurrency between data-center subsystems with opposite needs, and that this model can be used for the dynamic application of resource-efficiency policies.

- Proof that the productive analysis of realistic Cloud-Computing data centers can empirically guide data-center administrators to perform efficiency-related decisions.

1.5 Success Criteria

Success will be achieved if the research question and objectives are resolved, by testing that the models and the developed tools and algorithms which support them achieve better grades of energy efficiency. Simulation tools have been built in order to properly implement the energy-efficiency models and proof their validity and improvements.

The experimentation should be performed by following industry standards to recreate realistic, complex and heterogeneous scenarios which could be easily adopted by real-life partners. Such scenarios imply thousands or even tens of thousands of servers, several workload composition and patterns, and scheduling algorithms.

1.6 Thesis outline

The document is structured as follows: Chapter 2 introduces the problem of saving energy in Cloud-Computing large-scale data center facilities.

In Part II, six published papers have been selected. These papers address this thesis objectives. Such journals are included in the Thomson Reuters Journal Citation Reports (JCR) ranking integrated with the Institute for Scientific Information (ISI) web of knowledge.

Finally, in Part III, discussion, conclusions and future work are presented.

CHAPTER 2

CLOUD-COMPUTING DATA CENTERS

$$\begin{gathered}\vec{E}\cdot = \frac{Q_{enc}}{0}\\ \vec{B}\cdot = 0\\ \oint \vec{E}\cdot = -\\ \oint \vec{B}\cdot = 0\ + i_{enc}\end{gathered}$$

James Clerk Maxwell

2.1 Introduction

Cloud Computing is based on offering any computing resource as a service to the user. This business model enables users to reduce the management and operation costs related to physical infrastructures and human resources in order to properly run them.

In such paradigm, all user data and applications are stored in external facilities, which are usually backed by providers' data centers and computing clusters. Final users may access to those data and applications through the Internet if and only if they have the security permissions required.

Several service layers compose the cloud-computing business paradigm, among them:

- **Software as a Service**, where both the application and data to be processed are stored in external data centers and final users have access to them through web browsers or clients;
- **Platform as a Service**, where users utilize the toolkits and libraries offered by the provider to develop, configure and deploy software; and
- **Infrastructure as a Service**, where external computing, storing and networking resources, among others, are provided and used by final users.

The spread of this paradigm due to the growth of large internet companies, such as Google, Amazon, or Microsoft, has led to a higher utilization of data centers and computing clusters as the main computational core of Cloud Computing.

2.2 Data-Center architecture

Cloud-Computing data centers are composed of a complex mix of software and hardware solutions, which must collaborate to achieve high performance and reliability levels. Figure 2.1 shows a simple Cloud-Computing data-center architecture. In such a simple architecture, data centers are usually divided in management, monitoring, and virtualization modules, in addition to computing and storing resources.

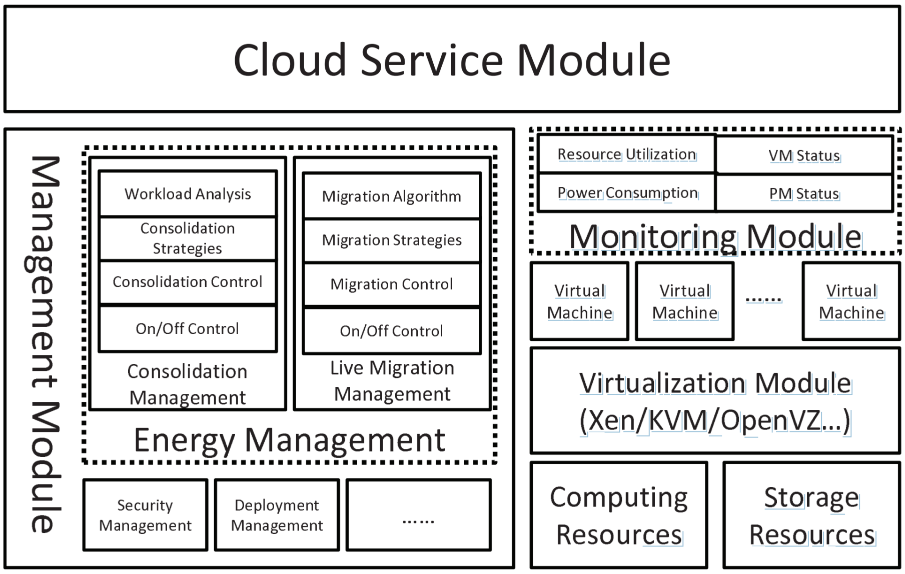


Figure 2.1: Simple Virtual Machine Based Energy-Efficient Data Center Architecture for Cloud Computing [10].

2.2.1 Hardware facilities

Data centers are not only composed of computational clusters, but many other physical resources are required in order to properly run such a facility. Among them:

- **Networking** facilities, such as routers, switches, and panels, including their respective installations and redundant components to achieve higher reliability.
- **Security access** facilities, such as cameras, security and telecommunication centers.
- **Fire control** facilities, such as humidity, smoke and fire detectors.

- **Electric** facilities, such as battery rooms, emergency generators, and power distribution equipment.

- **Cooling** facilities, including chillers, water-based cooling systems, and temperature regulators.

The aforementioned facilities typically consume more than 40% of the total energy of data centers, as shown in Figure 2.2. Many efforts have been done to reduce energy consumption in those facilities, such as: data-center cooling and temperature management [11] [12], energy-efficient hardware [13] [14] [15]; and power distribution [16].

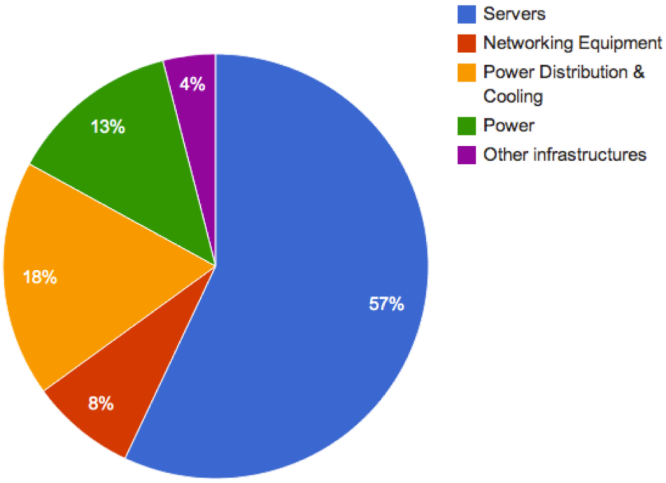


Figure 2.2: Distribution of electricity costs in Amazon Data Centers [17].

However, more than half of the energy is still consumed by the thousands, and even tens of thousands of servers these facilities are composed of.

2.2.2 Servers energy efficiency

In classic physics, Energy is the capacity to produce work (W), understood as the ability to move along a distance. Energy is often measured in kilowatt hours, (kWh).

Energy is a combination of power and time. The fewer power (watts) or the shorter time, the more energy reduction.

In the International System of Units (SI), energy is measured in joules (J) which is the energy expended to apply a force of one newton through a distance of one metre, or in passing an electric current of one ampere through a resistance of one ohm for one second, that is, $J = (kg \cdot m^2)/s^2 = N \cdot m = Pa \cdot m^3 = W \cdot s$

The reduction of energy consumed by servers in large-scale Cloud-Computing data centers may be targeted to different layers, as shown in Figure 2.3:

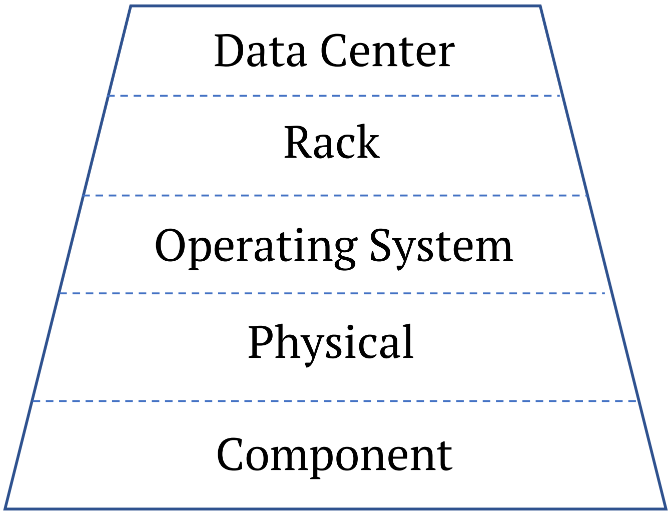


Figure 2.3: Server energy-consumption layers

- **Components layer**, which are the core pieces servers are composed of. Examples of energy-efficiency improvements in this layer include: Dynamic Frequency and Voltage Scaling (DVFS) in CPUs and memory; and new-generation SSD hard disks.

- **Physical layer**, where the server manages the power states of the hardware pieces. Blade servers, which can be power-managed at a chasis level, are a good example of improvements of energy-efficiency in this layer.

- **Operating system layer**, where software make decisions along with hardware to reduce energy consumption. The utilization of a different number of cores depending on the server workload is a good example of strategies applied at this level.
- **Rack layer**, where power consumption of the lower levels are aggregated in order to make higher-level energy-efficiency decisions.
- **Data-Center layer**, where energy-efficiency policies may be applied to anything within data center, such as racks, hardware, and software. High level tasks distribution according to cooling, energy and performance requirements are the key tools in this layer, such as the shut-down of under-utilized servers.

Cloud-computing cluster servers may switch between *On* (executing tasks), *Idle* (waiting for tasks), and *Off.* Each state has a particular power consumption. Some of the selected papers of this thesis dissertation focus on changing the aforementioned servers states.

2.2.3 Workloads

Figure 2.4 shows a simple typical workflow in Cloud-Computing data centers: the user submits a set of *Jobs* that must be executed by the computational clusters. Then, the resource manager coordinates the process of scheduling the *Tasks* the *Jobs* are composed of, while several systems apply the monitoring and security services, among others, required to a successful execution. This processing takes some time, which is called *queue time.* After this process, the *Tasks* are executed and the results stored or returned to the final user after the execution time. The whole time needed to fully execute all the *Tasks* in a *Job* is called the *Job makespan.*

Cloud Computing data centers and computational clusters do not usually execute a homogeneous set of *Jobs*, but several applications which share the same resource pool. In addition, the workload is constantly evolving and the workload pressure depends on the time, days, and geographical cluster location.

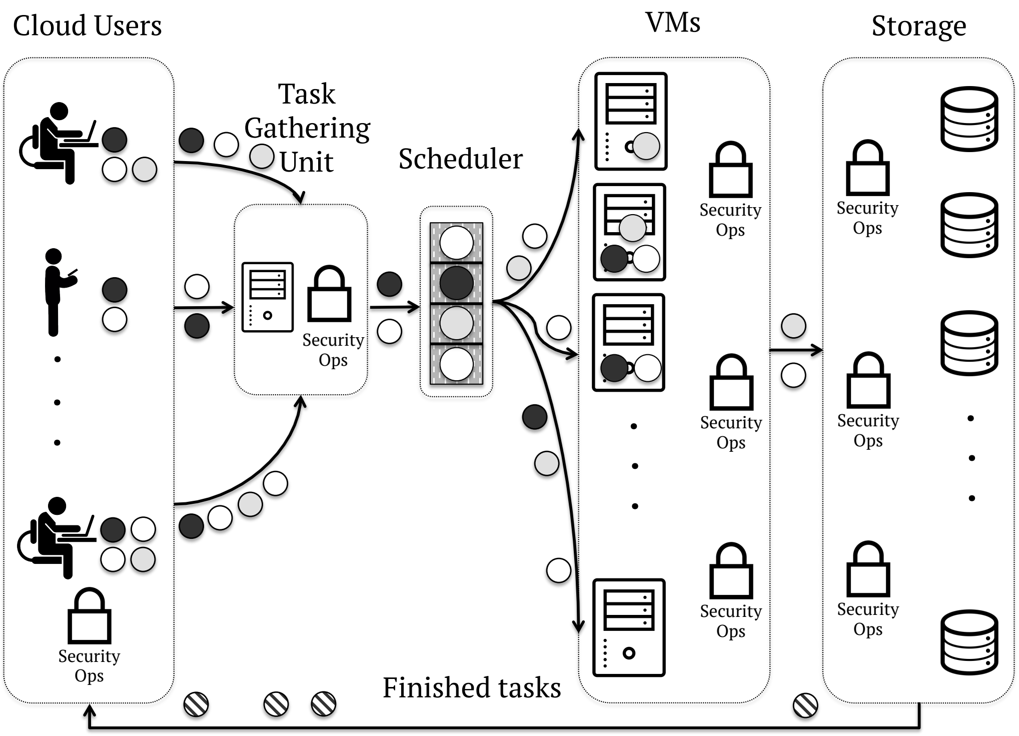


Figure 2.4: Cloud Computing workflow example

In some of the selected papers of this thesis dissertation, the trends presented by Google [18, 19] have been followed in order to create a realistic workloads, which are composed of two kinds of workloads:

- **Batch jobs**, which perform a concrete computation and then finish. These jobs have a determined start and end. MapReduce jobs are an example of *Batch* jobs.

- **Service jobs**, which are jobs that usually run longer than *Batch* jobs, and provide end-user operations and infrastructure services. As opposed to *Batch*, these jobs have no determined end. Web servers, distributed file systems, or services like BigTable [20] are good examples of *Service* jobs.

2.2.4 Resource managers

Resource schedulers and managers have evolved significantly from monolithic designs to more distributed and flexible strategies, which made them become one of the most critical parts of the data-center operating systems. Several degrees of parallelism have been added to overcome the limitations present in centralized monolithic scheduling approaches when complex and heterogeneous systems with a high number of incoming jobs are considered. The following scheduling models are studied in this work:

- **Monolithic**: A centralized and single scheduler is responsible for scheduling all tasks in the workload in this model [21]. This scheduling approach may be the perfect choice when real-time responses are not required [22, 23], since the omniscient algorithm performs high-quality task assignations by considering all restrictions and features of the data center [24, 25, 26, 27] at the cost of longer latency [23]. The scheduling process of a monolithic scheduler, such as that given by Google Borg [28], is illustrated in Figure 2.5.

- **Two-level**: This model achieves a higher level of parallelism by splitting the resource allocation and the task placement: a central manager blocks the whole cluster every time a scheduler makes a decision to offer computing resources to schedulers; and a set of parallel application-level schedulers perform the scheduling logic against the resources offered. This strategy enables the development of sub-optimal scheduling logic for each application, since the state of the data center is not shared with the central manager nor with the application schedulers. The workflow of the Two-level schedulers ([29, 3] Mesos and YARN respectively), is represented in Figure 2.6.

- **Shared-state schedulers**: On the other hand, in Shared-state schedulers, such as Omega [19], the state of the data center is available to all the schedulers. The central manager coordinates all the simultaneous parallel schedulers, which perform the scheduling logic against an out-of-date copy of the

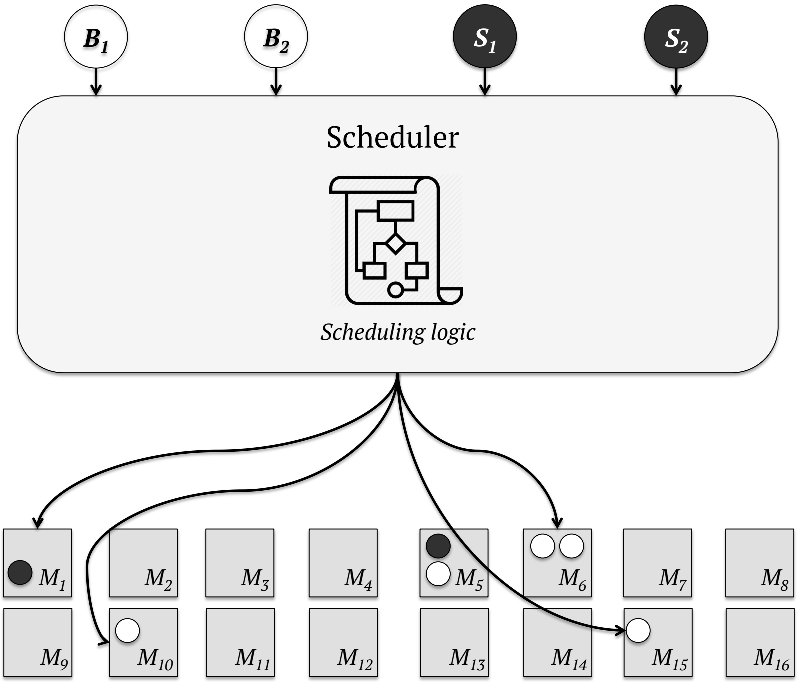


Figure 2.5: Monolithic scheduler architecture, , M - Worker Node, S - Service task, B - Batch task.

state of the data center. The scheduling decisions are then committed to the central manager, which strives to apply these decisions. Since schedulers use non-real-time views of the state of the data center, the commits performed by the central manager can result in a conflict when chosen resources are no longer available. In such a scenario, the local view of the state of the data center stored in the scheduler is refreshed before the repetition of the scheduling process. The workflow of the Shared-state scheduling model is represented in Figure 2.7.

2.3 Key performance indicators

In order to properly define the workload and to model the operational environment that the simulation tool will follow, hundreds of parameters have been covered, studied and described in the selected research papers. In this thesis dissertation,

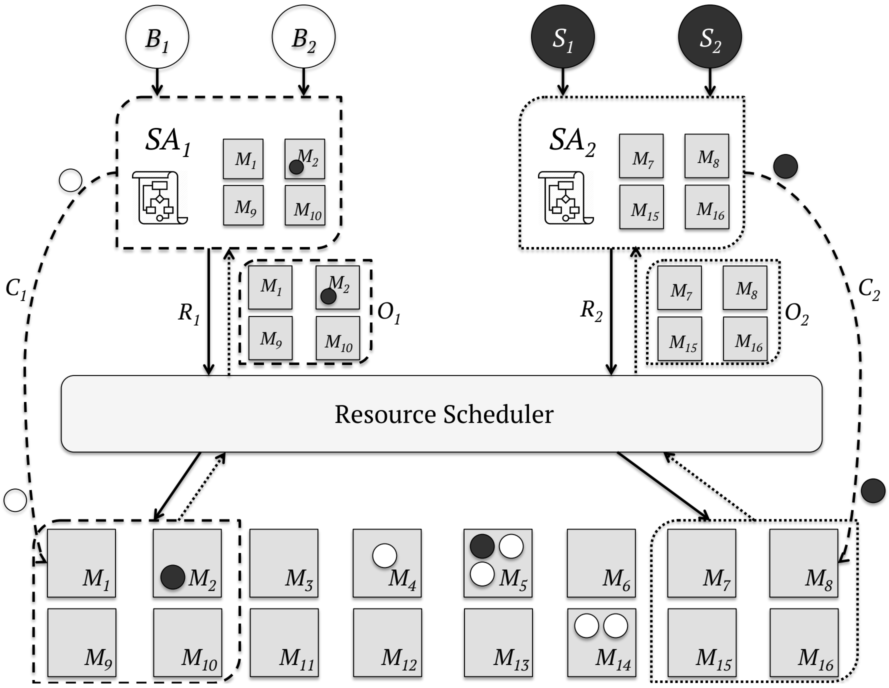


Figure 2.6: Two-level scheduler architecture, C - Commit, O - Resource offer, SA - Scheduler Agent.

the most relevant ones are presented in order to correctly understand the behaviour of our work and experimentation.

2.3.1 Workload and environmental parameters

Various parameters are considered in order to model the operational state of the data center and the characteristics of the workload. Among them:

- **Inter-arrival time**: represents the time between two consecutive *Service* or *Batch Jobs*. It determines also the amount of *Jobs* executed in a specific window time.
- **Number of tasks**: represents the number of *Tasks* a *Job* is composed of.
- **Job duration**: represents the time that a *Job* is consuming resources in the data center.

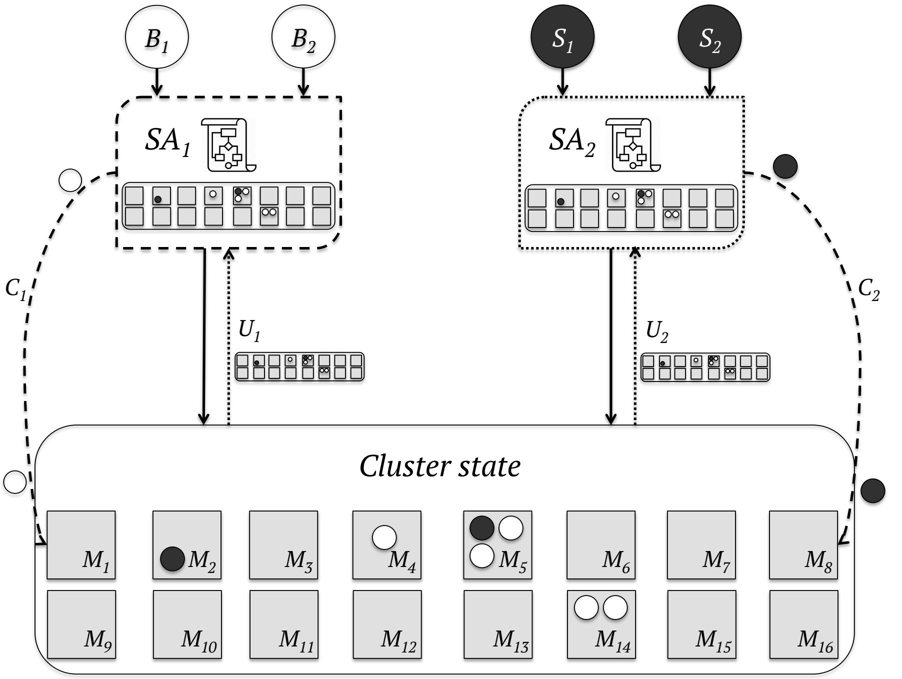


Figure 2.7: Shared-state scheduler architecture, U-Cluster State Update.

- **Job think time**: represents the time needed to make a schedule decision. This time is a job-related overhead.

- **Task think time**: In addition to the *Job think time*, the *Task think time* represents the amount of time needed to schedule each task in a particular *Job*.

- **CPU utilization**: The average CPU utilization in terms of the overall data center during the simulated operation time.

- **RAM utilization**: The average RAM memory utilization in terms of the overall data center during the simulated operation time.

2.3.2 Energy savings indicators

In order to describe the energy savings and its related behaviour, several parameters are taken into consideration. Among them:

- **Energy consumption**: The total energy consumed.
- **Energy savings**: The total energy saved compared to the base, non-efficient data center.
- **Total shut-down operations**: The total number of shutting downs performed during the overall simulated operation time. The higher the number, the more stress hardware will suffer.
- **Energy saved per shut-down operation**: Represents the energy saved against the shut-down operations performed. The higher the number, the more will be saved with the lesser hardware impact.
- **Time shut-down per shut-down cycle**: This indicator will show the amount of time a machine stays in a low-energy state without a power-on action being requested. The higher the time a shutting-down cycle is, the lesser the impact on the hardware. It can also represent a better or more conservative workload prediction.
- **Idle resources**: Represents the percentage of resources that are turned on and are not being used in each moment against the data-center resources. Lowering idle resources is a main concern, representing a good workload prediction and resource fit, therefore achieving the highest energy efficiency.

2.3.3 Performance indicators

In order to describe the impact of the different energy-efficiency models on the current data center performance, several parameters are studied and analyzed, including:

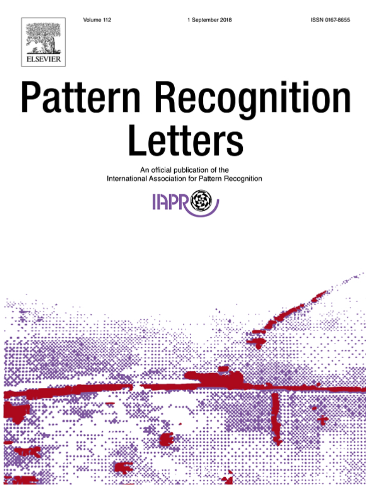
- **Makespan**: Represents the actual time needed for a *Job* to fully complete the execution of all the *Tasks* it is composed of.
- **Job queue time first scheduled**: Represents the time a *Job* is in the queue until the first of its *Tasks* is scheduled for the first time. It is usually related to the final user experience.

- **Job queue time fully scheduled**: Represents the time a *Job* is in the queue until all its *Tasks* are scheduled (not finished). It is usually related to the real computing experience and makespan.
- **Jobs timed out scheduling**: A *Job* is marked as timed out and left without scheduling when the scheduler tries to schedule the same job 100 times, or 1000 consecutive times the same task in a *Job*. The higher the number, the worst performance is achieved.
- **Busy time**: Represents the time employed by schedulers to perform scheduling decision *Tasks*. As the same workload is executed, a higher busy time will represent a higher scheduler occupation, worsening the overall scheduling performance.
- **Job think Time**: Represents the actual time needed for the scheduler to make a *Job* schedule decision.

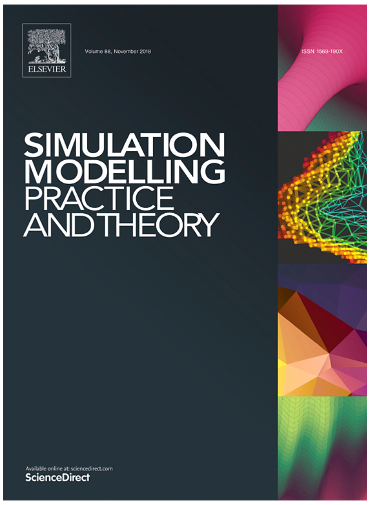
The aforementioned parameters constitute a subset of the total parameters under analysis. As more than one hundred configuration and results parameters are involved in the papers presented in this thesis dissertation, only the ones needed to a proper high-level understanding of the ongoing research have been described in this Section. Deeper explanations on such parameters can be found in the selected papers presented in Part II.

PART II

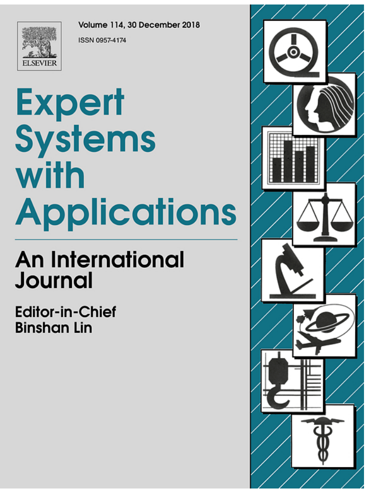
Selected Research Papers


Volume 112
1 September 2018
ISSN 0167-8655
ELSEVIER
Pattern Recognition Letters
An official publication of the International Association for Pattern Recognition
IAPR

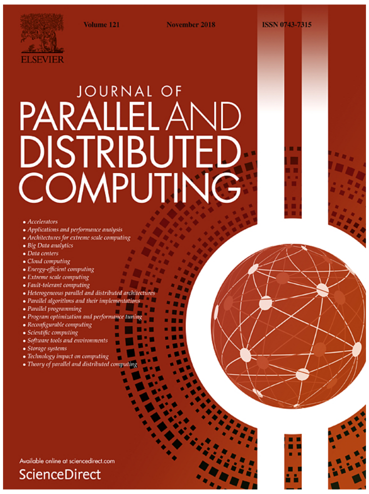
Energy wasting at internet data centers due to fear.


ELSEVIER
Volume 88, November 2018
ISSN 1569-190X
SIMULATION MODELLING PRACTICE AND THEORY
Available online at: sciencedirect.com
ScienceDirect

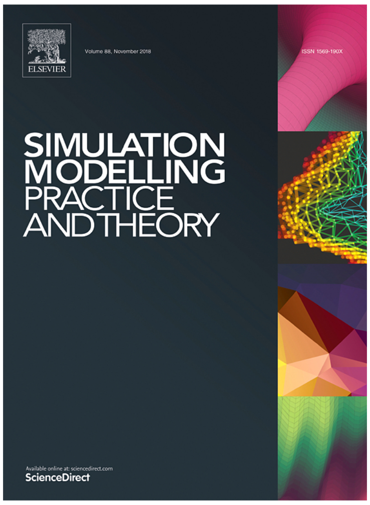
SCORE: Simulator for cloud optimization of resources and energy consumption.



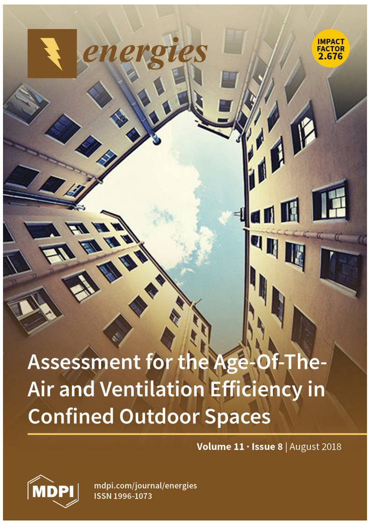
Energy policies for data-center monolithic schedulers.


ELSEVIER
Volume 121
November 2018
ISSN 0743-7315
JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING
Available online at sciencedirect.com
ScienceDirect

Security supportive energy-aware scheduling and energy policies for cloud environments.


ELSEVIER
Volume 88, November 2018
ISSN 1569-190X
SIMULATION MODELLING PRACTICE AND THEORY
Available online at: sciencedirect.com
ScienceDirect

GAME-SCORE: Game-based energy-aware cloud scheduler and simulator for computational clouds.


energies
IMPACT FACTOR 2.676
Assessment for the Age-Of-The-Air and Ventilation Efficiency in Confined Outdoor Spaces
Volume 11 · Issue 8 | August 2018
MDPI
mdpi.com/journal/energies
ISSN 1996-1073

Productive Efficiency of Energy-Aware Data Centers.

Energy wasting at internet data centers due to fear

This paper presents the initial step towards the achievement of this thesis dissertation, and fulfills the first of its research objectives: *"Proof that fear to the application of energy-efficiency policies that shut down underutilized machines should be overcome in order to achieve higher efficiency levels"*, since in current data centers, the fear experienced by data-center administrators presents an ongoing problem due to the low percentage of machines that they are willing to switch off in order to save energy.

This risk aversion can be assessed from a cognitive system. The purpose of this paper is to demonstrate the extra costs incurred by maintaining all the machines of a data center executing continuously due to fear to damaging hardware, degradation of service, and losing data.

To this end, an objective function which minimizes energy consumption depending on the number of times that the machines are switched on/off is provided. The risk aversion experienced by these data center administrators can be measured from the percentage of machines that they are willing to switch off.

The main contribution to the research community is the empirical analysis which shows that it is always the best option to turn off machines in order to reduce costs, given a formulation of the cognitive aspects of the fear experienced by data-center administrators.

This work was published in *Pattern Recognition Letters.* This Journal is indexed in JCR with an **Impact Factor of 1.586**. The Journal stands in ranking **Q2** in Computer Science, Artificial Intelligence (59/130).

Pattern Recognition Letters 67 (2015) 59–65

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Energy wasting at internet data centers due to fear[☆]

Alejandro Fernández-Montes[a,*], Damián Fernández-Cerero[a], Luis González-Abril[b], Juan Antonio Álvarez-García[a], Juan Antonio Ortega[a]

[a] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, Av. Reina Mercedes s/n, Sevilla, 41012, Spain
[b] Departamento de Econoía Aplicada I, Universidad de Sevilla, Av. Ramón y Cajal s/n, Sevilla, 41018, Spain

ARTICLE INFO

Article history:
Available online 3 July 2015

Keywords:
Costs of fear
Energy efficiency
Data center
Grid computing
Cloud computing

ABSTRACT

The fear experienced by datacenter administrators presents an ongoing problem due to the low percentage of machines that they are willing to switch off in order to save energy. This risk aversion can be assessed from a cognitive system. The purpose of this paper is to demonstrate the extra costs incurred by maintaining all the machines of a data center executing continuously for fear of damaging hardware, degradating the service, or losing data. To this end, an objective function which minimizes energy consumption depending on the number of times that the machines are switched on/off is provided. The risk aversion experienced by these data center administrators can be measured from the percentage of machines that they are willing to switch off. It is shown that it is always the best option to turn off machines in order to reduce costs, given a formulation of the cognitive aspects of the fear experienced by datacenter administrators.

1. Introduction

A data center is a facility used to house computer systems and associated components, such as telecommunications and storage systems. It generally includes redundant or backup power supplies, redundant data communications connections, environmental controls (e.g., air conditioning, fire suppression) and various security devices. Large data centers are industrial-scale operations that can consume as much electricity as a small town and sometimes constitute a major source of air pollution in the form of diesel exhaust.

The main purpose of a data center is to run applications, perform tasks or store data. The many examples of internet and computing services performed by data centers include:

The spread of cloud and grid computing paradigms has increased the size and usage of data centers; today there are thousands of data centers worldwide, which means millions of machines in total.

The majority of these facilities are located in the USA (about 25% of the total energy consumption of data centers worldwide [20]) and to a lesser extent in Europe. However, large companies such as Google locate a number of their data centers in high latitudes near the north pole to minimize cooling costs, which represent almost 40% of total energy consumption of these infrastructures [1].

Energy consumption by data centers has grown in the past ten years to 1.5% of worldwide energy consumption [25]. Major companies have therefore addressed their energy-efficiency efforts to areas such as cooling [7], hardware scaling [8] and power distribution [9], thereby slowing down the growth in power consumption in these facilities in recent years as we can see in Fig. 1, which shows the latest predictions.

In addition to these areas of work, saving energy by switching on/off machines in grid computing environments has been simulated using various energy efficiency policies, such as turning off every machine whenever possible, and turning off a number of machines depending on workload [10].

Although it has been demonstrated that about 30% of energy can be saved by applying these energy-aware policies [11], big companies still prefer not to adopt such policies due to their potential impact on the hardware, the possibility of damaging machines, and the costs associated with this hardware deterioration.

The purpose of this paper is to compute the costs imposed by the risk aversion experienced by data center administrators on switching off machines, and to show that even when taking these fears into consideration, some servers of the data center should still be turned off to minimize energy consumption and overall costs.

1.1. Cognitive systems modeling emotions

In psychology [33], emotion is a subjective, conscious experience characterized primarily by psycho-physiological expressions, biological reactions, and mental states. It is influenced by hormones and neurotransmitters, such as dopamine, noradrenaline, serotonin, oxytocin, cortisol, and gamma-aminobutyric acid. Furthermore,

[☆] This paper has been recommended for acceptance by Lledó Museros.
[*] Corresponding author. Tel.: +34 954 559 769; fax: +34 954 557 139.
E-mail address: afdez@us.es (A. Fernández-Montes).

http://dx.doi.org/10.1016/j.patrec.2015.06.018
0167-8655/© 2015 Elsevier B.V. All rights reserved.

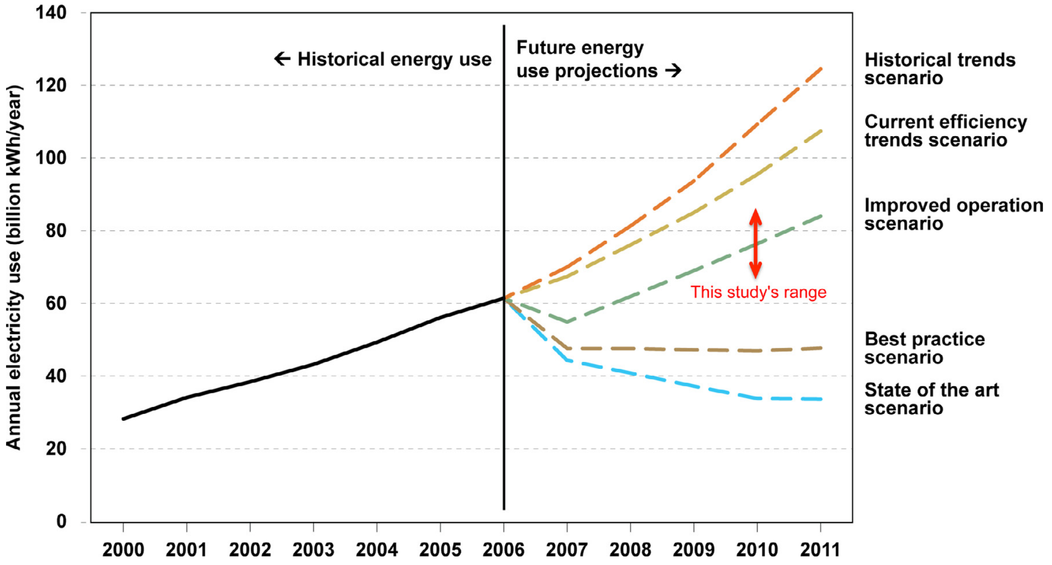


Fig. 1. Data center energy consumption worldwide [24].

neurologists [14] have made progress in demonstrating that emotion is as, or more, important than reason in the process of making decisions. Modeling emotions is a problem tackled from diverse knowledge areas: robot-based systems [6], music [30], videogames and virtual worlds [15] and domain-independent systems [16]. Moreover, emotion recognition systems [18] are on the rise in effective computing research. Data can be obtained from diverse sources: physiological signals (electromyogram, blood pressure, skin conductance, respiration rate and electroencephalogram rate), speech and facial expressions. Focusing on the emotional fear, it appears in response to a specific and immediate danger or a future specific unpleasant event. It can be measured and detected through biosignals such as irregular heart and respiration rate [5,19], visual signals (head gestures, nods and shakes) [17] and facial feature information [34]. Several studies [21] using optogenetic techniques have shown how aversive experiences trigger memories and suggest that combined hebbian and neuromodulatory processes interact to engage associative aversive learning.

Our interest in this paper is to model a function that quantifies the costs of the fear experienced by a datacenter operator on deciding whether a machine must be switched off. According to Michael Tresh, formerly a senior official at Viridity, a company that delivers energy-optimization to data centers: *“Data center operators live in fear of losing their jobs on a daily basis, because the business won’t back them up if there’s a failure.”* The startup ‘Power Assure’ which is focused on energy management, marketed a technology that enables commercial data centers to safely power down servers when they are not needed, but, as the manager of energy efficiency programs at the utility, Mary Medeiros McEnroe, explains that, even with aggressive programs to entice its major customers to save energy, Silicon Valley Power, a not-for-profit municipal electric utility, failed to persuade a single data center to use that technology. *“It’s a nervousness in the I.T. community that something isn’t going to be available when they need it”* [13]. Moreover, Power Assure, was dissolved in october 2014. Its technology was based on algorithms that enabled optimal server capacity and application needs to be calculated and to automatically shut off unnecessary capacity or spin up more capacity based on actual application demand. Jennifer Koppy, research director for data center management at International Data Corporation (IDC), said Power Assure’s energy management technology was *“extremely forward-looking … they had a superb idea, but I don’t think the market is ready yet.”*

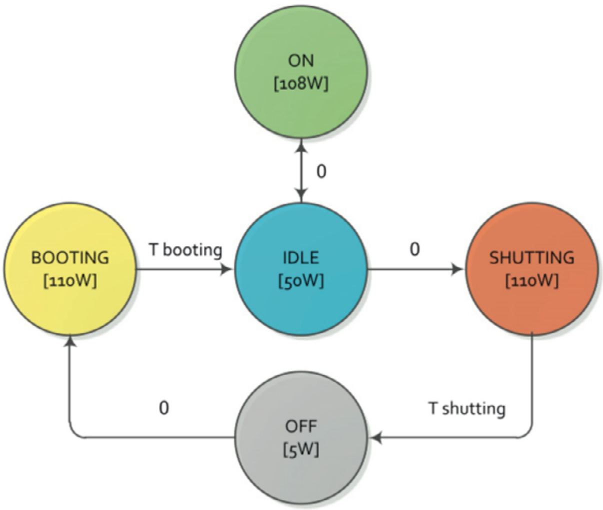


Fig. 2. Life cycle of a data center server [10].

2. Problem analysis

It makes sense that one of the most effective ways to achieve considerable energy savings is to turn off computers that are not being used. Although this idea is generally accepted by users, and hence most personal computers are turned off at night or during periods of low usage, it is seldom implemented in data centers or at enterprise level.

Although the average server utilization within data centers is very low (typically between 10% and 50% [4]), very few companies prefer to turn off the machines that are not in use rather than leaving them in an idle state. While idle servers consume half the energy of those in a state of intensive use [24], this remains a high direct and indirect energy cost due to the increased need for cooling. The several different states through which a machine can pass are shown in Fig. 2. In this state diagram the average power consumption of a common server per CPU in each state is also shown, and the time needed to

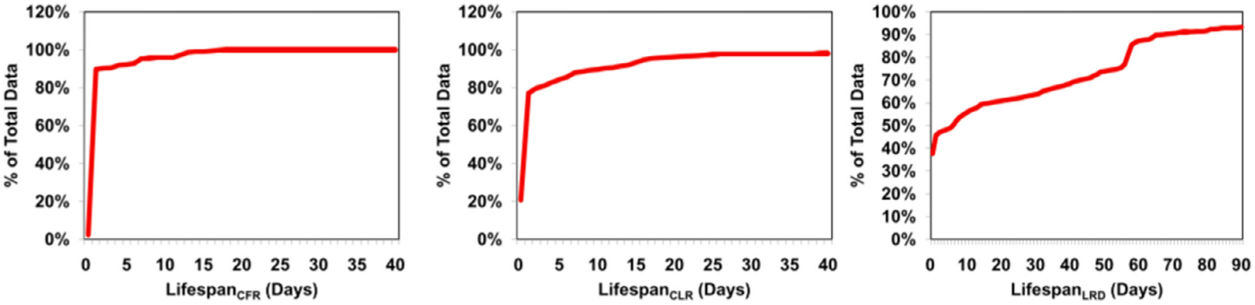


Fig. 3. File access pattern in Yahoo cluster [23].

change from one state to another. Notice that a server with 4 CPUs spends 4 times the energy shown in each state, i. e. 432 Watts∗h in ON state.

It has to be noted that very few machines cannot be switched off. Some of the machines from the data center act as master nodes, while the vast majority of machines act as slave nodes which are candidates to be switched off.

The main reasons why IT departments generally prefer to keep machines idle are for fear of:

- **Hardware damage**: It is known that due to a high number of switching on/off cycles, some computer hardware components suffer stress, which can lead to computer deterioration. We incur this as a cost: **The repair cost**. The component that is usually damaged is the hard drive [29], which has other implications besides simply their repair or replacement costs. However, due to the constant improvements of these components and the new SSD hard drives, it can be expected that the failure rate of these pieces of hardware will diminish over time, and therefore these new drives will reduce this type of fear.
- **Service degradation**: When a task needs the service of this damaged computer which can no longer perform a service, in a new cost is incurred due to the worsening in service quality, response times, etc.: **The opportunity cost**. Despite this potential opportunity cost, as we have seen, the server utilization within data centers is very low therefore, in a distributed environment, is highly unlikely that no other machine in the data center can provide the service that this machine was providing.
- **Data loss**: This is a critical issue in a data center infrastructure. If the machine (and its hard drive) that has been damaged was the only one that stored certain data and this data has been lost, certain critical operations could not be performed and it would entail very high operation costs. However, as mentioned above, distributed systems such as data centers typically replicate their data between multiple machines across the data center servers, and therefore, it is highly unlikely for information to be lost. Data loss will only happen if data has just been created and has not had time to be replicated.

Due to these fears experienced by the IT staff from big internet companies, file distribution policies within data centers are designed to minimize the possibility of losing any data, thereby maximizing the availability of data and the available computing capacity to perform tasks associated with it.

These distribution policies do not aim at energy efficiency. To achieve this energy efficiency, data center managers rely on hardware systems that work by: switching off some components - mainly the hard drive - to a state of inactivity; improving cooling systems; adopting chiller-free cooling strategies; or by raising operating temperature [7].

A performance penalty is imposed on hardware components left in a state of inactivity and the entire data center has to assume a delay of up to several seconds for inactive drives. In addition, we must take into account that there is a trend among these infrastructures that involves the utilization of multiple hard drives – ranging from 4 to 6 – rather than RAID systems, which are less energy efficient. In this type of system, hard disk consumption only accounts for 10% of energy consumption; the bulk of the energy is consumed by harder scalable hardware components such as RAM or CPU, which consume about 63% of the total energy [28].

To achieve this high availability of data stored in the data center, many parallel-computing frameworks and distributed file systems such as Hadoop [31] and GFS [12], make use of data replication as a strategy to maximize its availability and fault-tolerance, distributing it in accordance with policies that minimize the possibility of corruption in all stored replicas and thereby the irretrievable loss of any data.

The above policies meet the requirements satisfactorily, since they minimize the risk of data loss within the data center. However, these kinds of policies have some disadvantages, including:

- **Location and status of data are not taken into account**: Temporal data locality is essential to building operating optimization policies for the data center due to the usage of and access to file patterns. Therefore, the computation required to execute the related tasks follows a pattern as shown in Fig. 3.
  In the case study of the Yahoo! Hadoop cluster that serves as a base for GreenHDFS [22], 60% of this cluster total space was being used by data that is not often accessed. The current average lifetime during which a piece of data is often used is 3 days in 98% of cases, and even exceptions to this pattern of use, 80% of the files were used intensively for fewer than 8 days. Within the group of files that are not frequently used, non-access periods varied between 1 and 18 days [23].
- **File distribution policies are not efficient-friendly**: Current distribution policies scatter data blocks between the largest possible number of machines with the aim of minimizing the risk of losing any data due to hardware failure on the machine, failure of facility components at rack level, etc. Servers are therefore constantly underused as mentioned above, which in turn results in low power usage in data storage and associated computing, as well as making impossible an orderly shutdown of these servers impossible without jeopardizing the proper functioning of the data center. Moreover, these distribution policies are based on the static and constant replication model, where all file blocks have the same number of copies and are distributed following the same rules, regardless of the access or computing needs.

For the reasons discussed (the low rates of storage and computing power utilization of these facilities), it seems that if efficient distribution policies are applied in conjunction with switching on/off policies, then not only will data center performance be free from compromise in achieving greater energy efficiency, but also substantial improvements in both aspects can be achieved due to the inefficiency of the current data distribution policies. Of course, this kind of efficient distribution and switching on/off policies can never jeopardize the availability and integrity of data, but must minimize (if not improve) impact on overall data center performance. Within these distributions and machine power on/off policies we can highlight:

- **Covering subset**: These policies are based on splitting the data center into many disjoint areas so that a number of replicas of each file are stored. The goal of systems that implement these policies is to switch off the maximum number of sectors in the data center to achieve greater energy savings, without affecting the correct operation [26] [35]. The disadvantages of the systems that implement these policies are:
  - The worsening write rate due to write-offloading associated with writing on machines that are not running at the time that writing occurs [3].
  - The number of replicas of each file is constant and static.
  - Neither data time locality nor file utilization pattern are taken into account.

  Systems like Sierra [32] and Rabbit [2] obtain a very high energy proportionality with virtually no impact on the availability and only a slight impact on the overall performance of the data center.
- **Data temperature**: Systems that apply these policies are based on the temporal locality and frequency of use of the files stored in the data center to consistently assign them a temperature (the more frequently used the file is, the hotter the temperature) and redistribute them into two areas: a hot zone aimed at maximizing the performance and availability of data stored on it; and a cold zone whose aim is to minimize the energy consumption of the machines assigned to this area. In such systems, such as GreenHDFS [22], the ultimate goal is to efficiently distribute the machines between these different areas, maximizing the overall performance thanks to improvements in the hot zone, minimizing the overall energy consumption thanks to improvements in the cold zone, increasing the time response as little as possible when reading files from machines switched off (in GreenHDFS, only 2.1% of the readings were affected by this temporary penalty due to switching on the machine at the time of the reading), thereby significantly reducing the energy consumption of servers: 24% in the case of GreenHDFS [23].
- **Dynamic replication**: Other solutions, such as Superset [27], take the above strategies as a starting point, but also take into account the “temperature” of the data above a threshold, not only to power on/off machines, but also to increase or decrease the number of copies of stored data, thereby preserving the availability of data and reducing overall energy consumption thanks to the switching on/off policies and improved performance. This is achieved by transferring storage space and computing power from the cold files that are not frequently used, to those files that need these resources, i.e, the hottest files.

As we have discussed, the problems related to the server shutdown are not critical and do not endanger the proper operation of these infrastructures. Therefore, this paper studies the costs caused by risk aversion, and the energy savings and reduced environmental impact that could be achieved if this fear is overcome.

3. Theoretical analysis

A function that quantifies the costs of fear, i.e. the costs associated with the belief that turning off data center machines imposes a greater cost than the energy savings achieved, is proposed. From this function, an assessment of the risk aversion to switching off machines is provided.

Let us present the problem. Given a set of tasks to be computed in a period of time T, it is assumed that the minimum power consumption, min, is achieved by turning off the machines whenever possible, and that the maximum power consumption, Max, is obtained in the case that the machines never are turned off. Hence, the extra expense imposed due to the consumption from all those machines remaining turned on without interruption is given by

$$M = Max - min$$

Let us suppose that a datacenter has n machines, all of them equal. This act is justified since actually, data center machines are grouped by racks of identical machines. Even machines from different racks share the same components or at least components are produced by the same manufactures.

Let N_j be the maximum number of times that a machine j, $j = 1, \ldots, n$, can be turned on given an operation time T. This value is computed as a maximum that depends on operation time T, shutting down time (T_{off}, time needed to switch off a machine) and turning on time (T_{on}, time needed to switch on a machine from the off state) as follows:

$$N_j = \frac{T}{T_{off} + T_{on}}$$

Therefore, by considering that all machines are equal ($N_j = N$), the maximum number of times that the machines of the datacenter can be turned on given an operation time T is

$$N_1 + \cdots + N_n = n \cdot N$$

Let X_i^j be the random variable which takes the value 1 if a computer j breaks down on power switching i and 0 otherwise. Hence, if the probability of $X_i^j = 1$ is p_i, that is $P(X_i^j = 1) = p_i^j$, then X_i^j follows a Bernoulli model and, hence $E[X_i^j] = p_i^j$. With respect to p_i^j, some considerations must be given:

- As aforementioned, all machine of the datacenter are supposed equal, therefore $p_i^j = p_i$ for any $j = 1, \ldots, n$.
- p_i depends on the power switching i and this values can be considered constant within a horizon of the framework T. Clearly, $p_i = p_i(t)$ and $\frac{d\,p_i(t)}{dt} > 0$, that is, the probability of malfunction of a machine is going to increase during its life. Nevertheless, the technology of this machine provides that this probability decreases slowly, $\frac{d\,p_i(t)}{dt} \approx 0$, and the considered operation time, T, is short compared to its lifetime. Hence, $p_i = p \approx constant$ can be considered.
- With respect to the value of p. The advance in the technology indicates that the real value of p_i is to be very close to 0. Nevertheless, from a cognitive point of view, the data center administrator can consider it is a high value and this is the reason why it would never be a good idea to switch off machines.

It is worth noting that if a machine breaks down, there are other machines of the datacenter available to replace its operational requirements. Let n_j be the number of times that a machine j should be switched off, $0 \le n_j \le N$.

From here, if x denotes the number of power cycles, a new random variable

$$S(x) = \sum_{j=1}^{n} \sum_{i=1}^{n_j} X_i^j, \qquad x = 0, 1, \ldots, n \cdot N$$

where $x = \sum_{j=1}^{n} n_j$, that is, x is the number of power cycles performed in all machines. Thus, $S(x)$ is a random variable that represents the number of machines broken down in x power cycles and, hence, $0 \le S(x) \le n$ for any x.

The average number of damaged machines after x switching on/off cycles, that is, the expectation of the random variable $S(x)$ is calculated as follows:

$$E[S(x)] = E\left[\sum_{j=1}^{n} \sum_{i=1}^{n_j} X_i^j\right] = \sum_{j=1}^{n} \sum_{i=1}^{n_j} E[X_i^j] = p \cdot \sum_{j=1}^{n} n_j = x \cdot p$$

Furthermore, the cost of repairing computers damaged by the switching on/off cycles has also to be taken into account. Let $C_r > 0$ be the average cost of repairing the computer. Hence, the costs of fear, derived from switching on/off machines, denoted by C_{fear}, can be given as follows:

$$C_{fear}(x) = x \cdot p \cdot C_r, \quad x = 0, 1, \ldots, n \cdot N$$

In addition, if a computer is turned off and then there is a request that requires the machine to be turned on, then the client will need to wait until the computer is turned on. Considering C_o as the opportunity cost that measures the value that a customer gives to that lost time, and T_{on} as the time needed for a computer to be turned on. Then, the turn on costs, denoted by C_{on}, can be quantified as follows:

$$C_{on}(x) = x \cdot T_{on} \cdot C_o, \quad x = 0, 1, \ldots, n \cdot N$$

Therefore, the total cost of turning off x machines, denoted by $C(x)$, is given as $C(x) = C_{fear}(x) + C_{on}(x)$, that is,

$$C(x) = x \cdot (T_{on} \cdot C_o + p \cdot C_r) \qquad x = 0, \ldots, n \cdot N \tag{1}$$

From the above function, the cost of switching off the machines is as follows:

$$C(n \cdot N) = n \cdot N \cdot (T_{on} \cdot C_o + p \cdot C_r)$$

Nowadays due to the different aspect of the life among them, the cognitive aspect, most companies prefer not to turn off machines so this decision implies that $C(n \cdot N) > M$. The main aim of this paper is to show that this is not an optimal decision.

First, in order to simplify the function given in (1), the variable $y = \frac{x}{n \cdot N}$ is considered which indicates the proportion (per unit) of the number of switching on/off cycles applied against the natural maximum applicable. Hence,

$$C(y) = y \cdot n \cdot N \cdot (T_{on} \cdot C_o + p \cdot C_r) \tag{2}$$

From the definition of y, it can be seen that $(1 - y)$ represents the percentage of switching on/off cycles not applied to the machines. Assuming that the cost of having all the extra machines turned on, M, is proportional to the percentage of switching on/off cycles applied as represented by y, then $(1 - y) \cdot M$ represents the cost of having the machines switched on. From these latter two costs, the cost for having a percentage of machines turned off, is given by

$$f(y) = y \cdot n \cdot N \cdot (T_{on} \cdot C_o + p \cdot C_r) + (1 - y) \cdot M \quad 0 \leq y \leq 1 \tag{3}$$

Since M is not null, and $C(1) = n \cdot N \cdot (T_{on} \cdot C_o + p \cdot C_r) > M$ due to current fear experienced by most companies, the value

$$A = \frac{C(1)}{M} > 1$$

is considered and the cost function, denoted by $f_{current}$, is written as follows:

$$f_{current}(y) = A \cdot y + (1 - y) \qquad 0 \leq y \leq 1,\ A > 1 \tag{4}$$

Following the current hypothesis which assumes that switching off any machine implies more cost (the so-called fear cost), this objective function reaches its minimum when $y_0 = 0$, i.e., when no machines are turned off and they maintain continuous execution, and $f_{current}(y_0) = 1$ (An example of this kind of function is given in Fig. 4).

4. Fear cost

In this section, a new cost function is given by assuming that the switching off/on of machines in moderation may have a benefit.

First, let us indicated that the function $f_{current}(y)$ verifies that $\frac{d\,f_{current}(y)}{dy} = A - 1 = cte$, that is, the increment of the emotional cost caused by the modification of the percentage of power cycles is constant for the datacenter administrator. However, this is not a realistic hypothesis since by taking into account the pessimism (cognitive aspect) of the administrator, the $f_{current}(y)$ function must verify that $\frac{d^2 f_{current}(y)}{dy^2} > 0$ since, for instance, the incremental cost to change of 0.1–0.2 must be smaller than the incremental cost to change of 0.7 to 0.8.

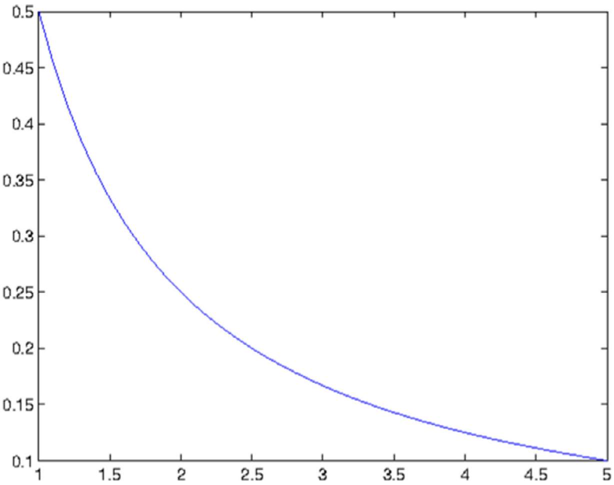


Fig. 4. Graphical representation of the point where the minimum of function (5) is attained.

Hence, the new function cost, denoted by f_{prop}, must verify that

- If y is near to zero, then $f_{prop}(y) < f_{prop}(0)$ since the switching off/on of machines in moderation may have a benefit. Furthermore, in order to provide a regular function it is imposed that $\frac{d\,f_{prop}(y)}{dy}$ exists for any y.
- By following the commentary of the $f_{current}(y)$ function with respect to the second derivative, the $\frac{d^2 f_{prop}(y)}{dy^2} > 0$ is required.
- Furthermore, a similar to $f_{current}(y)$ functional form is required for $f_{prop}(y)$.

Thus, the simplest function with these conditions is:

$$f_{prop}(y) = Ay^2 + (1 - y) \qquad 0 \leq y \leq 1,\ A > 1 \tag{5}$$

Moreover, another justification of this new function is that it lends less weight to the value of $C(y)$ given in (2) in the function (3). Hence the importance of switching off machines is relaxed.

The function (5) is convex (see Fig. 5) and reaches its minimum at the point:

$$y_0 = \frac{1}{2A} \tag{6}$$

and $f_{prop}(y_0) = 1 - \frac{1}{2A}$. This means that the ideal situation is to switch off $\frac{1}{2A}$% of machines, and the savings, as a percentage, are equal to the percentage of machines switched off.

Thus, if A has a high value, this favours the shutdown of the servers in the data center. And, if A is close to 1 it favours keeping 50% machines on/idle which is a consequence of the supposition that the ‘switching off/on of machines in moderation may have a benefit’.

Hence, a coefficient, denoted by $fear$, which measures the risk aversion to switching off the machines, is modeled as follows:

$$fear = 1 - \frac{1}{A}$$

This value verifies $0 \leq fear \leq 1$ and satisfies:

- $fear = 0\ (A = 1)$ implies low risk aversion, and under the hypothesis of ‘switching off/on machines in moderation may have a benefit’, means switching off 50% of the machines.

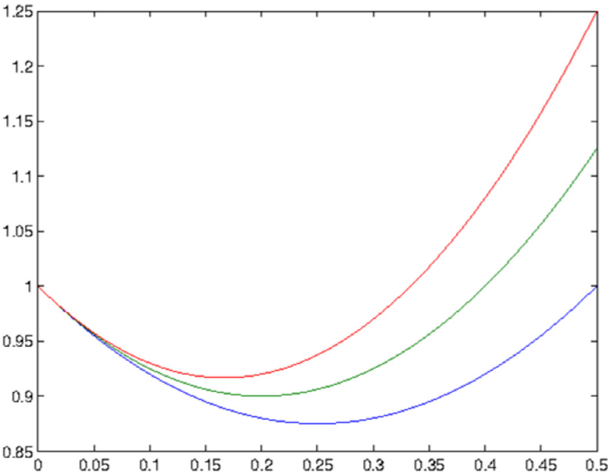


Fig. 5. Graphical representation of the function (5) for $A = 2$ (blue), 2.5 (green) and 3 (red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article).

- $fear = 1$ $(A = \infty)$ implies maximum risk aversion, and therefore machines are never switched off.

As aforementioned, most data center companies currently do not shut down servers, so the value of A is set to ∞ in the proposed function and hence the number of machines switched off is 0 $(y_0 = 0)$.

Based on these developments, it is possible to model the risk aversion experienced by data center companies by posing a simple question: *What percentage of machines are you willing to switch off?* For instance, if the answer is 10%, the equation $0.1 = \frac{1}{2A}$ is resolved, which means that $A = 5$, thus the $f_{prop}(y) = 5y^2 + (1 - y) \qquad 0 \le y \le 1$ and from this point the emotion of the fear experienced by the company is as follows:

$$A = 5 \Rightarrow fear = 1 - \frac{1}{5A} = 0.8$$

In contrast, if the answer is 40% of machines, then $A = \frac{5}{4}$, and therefore: $fear = 0.2$.

5. Conclusions

In this paper we have presented the cost of risk aversion to which most companies currently subscribe due to the false belief that turning off machines in data centers involves more costs than savings.

In order to demonstrate this, an objective function has been proposed which determines that a lower total cost can always be attained by turning off data center servers a number of times, showing that the current belief is a mistake that should be corrected by applying shutting on/off policies.

As future work, we plan to measure the extra costs associated with turning off the machines in terms of hardware damage and to measure the energy savings that could be obtained by building a software system which implements policies for energy efficiency in data centers.

Acknowledgments

This research is supported by the project Simon(TIC-8052) and Context-Learning (P11-TIC-7124) of the Andalusian Regional Ministry of Economy, Innovation and Science and by the Spanish Ministry of Economy and Competitiveness through the project HERMES - Healthy and Efficient Routes in Massive Open Data-based Smart Cities (TIN2013-46801-C4).

References

[1] N. Ahuja, C. Rego, S. Ahuja, M. Warner, A. Docca, Data center efficiency with higher ambient temperatures and optimized cooling control, in: Proceedings of the Semiconductor 27th Annual IEEE Thermal Measurement and Management Symposium (SEMI-THERM), IEEE, 2011, pp. 105–109.
[2] H. Amur, J. Cipar, V. Gupta, G.R. Ganger, M.A. Kozuch, K. Schwan, Robust and flexible power-proportional storage, in: Proceedings of the 1st ACM Symposium on Cloud Computing, ACM, 2010, pp. 217–228.
[3] H. Amur, K. Schwan, Achieving power-efficiency in clusters without distributed file system complexity, in: Computer Architecture, Springer, 2012, pp. 222–232.
[4] L.A. Barroso, U. Hölzle, The case for energy-proportional computing, IEEE computer 40 (12) (2007) 33–37.
[5] G. Chanel, K. Ansari-Asl, T. Pun, Valence-arousal evaluation using physiological signals in an emotion recall paradigm, in: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 2007 (ISIC), IEEE, 2007, pp. 2662–2667.
[6] M. Díaz, J. Saez-Pons, M. Heerink, C. Angulo, Emotional factors in robot-based assistive services for elderly at home, in: Proceedings of the IEEE RO-MAN, 2013, IEEE, 2013, pp. 711–716.
[7] N. El-Sayed, I.A. Stefanovici, G. Amvrosiadis, A.A. Hwang, B. Schroeder, Temperature management in data centers: why some (might) like it hot, ACM SIGMETRICS Performance Evaluation Review 40 (1) (2012) 163–174.
[8] X. Fan, W.-D. Weber, L.A. Barroso, Power provisioning for a warehouse-sized computer, in: ACM SIGARCH Computer Architecture News, volume 35, ACM, 2007, pp. 13–23.
[9] M.E. Femal, V.W. Freeh, Boosting data center performance through non-uniform power allocation, in: Proceedings of the Second International Conference on Autonomic Computing, 2005. ICAC 2005, IEEE, 2005, pp. 250–261.
[10] A. Fernández-Montes, L. Gonzalez-Abril, J.A. Ortega, L. Lefèvre, Smart scheduling for saving energy in grid computing, Expert Syst. Appl. 39 (10) (2012a) 9443–9450.
[11] A. Fernández-Montes, F. Velasco, J. Ortega, Evaluating decision-making performance in a grid-computing environment using DEA, Expert Syst. Appl. 39 (15) (2012b) 12061–12070.
[12] S. Ghemawat, H. Gobioff, S.-T. Leung, The google file system, ACM SIGOPS Oper. Syst. Rev. volume 37 (2003) 29–43.
[13] J. Glanz, Power, pollution and the internet, NY Times 22 (2012).
[14] D. Goleman, S. Sutherland, Emotional Intelligence: Why it can Matter more than IQ, Bloomsbury, London, 1996.
[15] J. Gratch, S. Marsella, Tears and fears: Modeling emotions and emotional behaviors in synthetic agents, in: Proceedings of the Fifth International Conference on Autonomous Agents, ACM, 2001, pp. 278–285.
[16] J. Gratch, S. Marsella, A domain-independent framework for modeling emotion, Cogn. Syst. Res. 5 (4) (2004) 269–306.
[17] H. Gunes, M. Pantic, Dimensional emotion prediction from spontaneous head gestures for interaction with sensitive artificial listeners, in: Intelligent virtual agents, Springer, 2010, pp. 371–377.
[18] H. Gunes, B. Schuller, M. Pantic, R. Cowie, Emotion representation, analysis and synthesis in continuous space: a survey, in: Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), IEEE, 2011, pp. 827–834.
[19] A. Haag, S. Goronzy, P. Schaich, J. Williams, Emotion recognition using biosensors: first steps towards an automatic system, in: Proceedigns of the Affective dialogue systems, Springer, 2004, pp. 36–48.
[20] N. Hayes, DatacenterDynamics Global Industry Census 2012, Technical Report, DatacenterDynamics, 2012.
[21] J.P. Johansen, L. Diaz-Mataix, H. Hamanaka, T. Ozawa, E. Ycu, J. Koivumaa, A. Kumar, M. Hou, K. Deisseroth, E.S. Boyden, et al., Hebbian and neuromodulatory mechanisms interact to trigger associative memory formation, Proc. Natl. Acad. Sci. 111 (51) (2014) E5584–E5592.
[22] R.T. Kaushik, M. Bhandarkar, Greenhdfs: towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster, in: Proceedings of the USENIX Annual Technical Conference, 2010, p. 109.
[23] R.T. Kaushik, M. Bhandarkar, K. Nahrstedt, Evaluation and analysis of greenhdfs: a self-adaptive, energy-conserving variant of the hadoop distributed file system, in: Proceedings of the Cloud IEEE Second International Conference on Computing Technology and Science (CloudCom), IEEE, 2010, pp. 274–287.
[24] J. Koomey, Growth in data center electricity use 2005 to 2010, A report by Analytical Press, completed at the request of The New York Times (2011).
[25] J.G. Koomey, Worldwide electricity used in data centers, Environ. Res. Lett. 3 (3) (2008) 034008.
[26] J. Leverich, C. Kozyrakis, On the energy (in) efficiency of hadoop clusters, ACM SIGOPS Oper. Syst. Rev. 44 (1) (2010) 61–65.
[27] X. Luo, Y. Wang, Z. Zhang, H. Wang, Superset: a non-uniform replica placement strategy towards high-performance and cost-effective distributed storage service, in: Proceedings of the International Conference on Advanced Cloud and Big Data (CBD), IEEE, 2013, pp. 139–146.
[28] D.A. Patterson, The data center is the computer, Commun. ACM 51 (1) (2008) 105.
[29] E. Pinheiro, W.-D. Weber, L.A. Barroso, Failure trends in a large disk drive population, in: FAST, volume 7, 2007, pp. 17–23.
[30] E. Schubert, Modeling perceived emotion with continuous musical features, Music Percept. 21 (4) (2004) 561–585.

[31] K. Shvachko, H. Kuang, S. Radia, R. Chansler, The hadoop distributed file system, in: Proceedings of the IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), 2010, IEEE, 2010, pp. 1–10.

[32] E. Thereska, A. Donnelly, D. Narayanan, Sierra: practical power-proportionality for data center storage, in: Proceedings of the Sixth Conference on Computer Systems, ACM, 2011, pp. 169–182.

[33] P.A. Thoits, The sociology of emotions, Annu. Rev. Sociol. (1989) 317–342.

[34] M. Wöllmer, A. Metallinou, F. Eyben, B. Schuller, S.S. Narayanan, Context-sensitive multimodal emotion recognition from speech and facial expression using bidirectional lstm modeling, in: Proceedings of the INTERSPEECH, 2010, pp. 2362–2365.

[35] Z. Zeng, B. Veeravalli, Do more replicas of object data improve the performance of cloud data centers? in: Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing, IEEE Computer Society, 2012, pp. 39–46.

SCORE: Simulator for cloud optimization of resources and energy consumption

After the motivation presented in the previous paper, we needed to meet the second research objective of this thesis dissertation: *"Proof that a simulation tool able to simulate large-scale data centers with high performance can be built to trustfully test the models proposed. This simulation tool may include several energy efficiency policies, scheduling algorithms, resource managers and workloads"*, and we needed to face also the complex challenge of achieving efficiency both in terms of resource utilization and energy consumption, especially in large-scale wide-purpose data centers that serve cloud-computing services. Simulation presents an appropriate solution for the development and testing of strategies that aim to improve efficiency problems before their applications in production environments. Various cloud simulators have been proposed to cover different aspects of the operation environment of cloud-computing systems, but they lack crucial features needed to achieve the goal of this thesis dissertation.

In this paper we present our next contribution, the SCORE simulation tool, which is dedicated to the simulation of energy-efficient monolithic and parallel-scheduling models and for the execution of heterogeneous, realistic and synthetic workloads. The simulator has been evaluated through empirical tests. The results of the experiments performed confirm that SCORE is a performant and reliable tool for testing energy-efficiency, security, and scheduling strategies in cloud-computing environments. This paper is the result of my first research stage in Cracow. With this work, both research groups started a tight research collaboration that continues in the present.

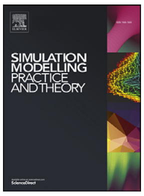
This work was published in *Simulation Modelling Practice and Theory*. This Journal is indexed in JCR with an **Impact Factor of 2.092**. The Journal stands in ranking **Q1** in Computer Science, Software Engineering (21/104).

Simulation Modelling Practice and Theory 82 (2018) 160–173

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat



SCORE: Simulator for cloud optimization of resources and energy consumption

Damián Fernández-Cerero[a,*], Alejandro Fernández-Montes[a], Agnieszka Jakóbik[b], Joanna Kołodziej[c], Miguel Toro[a]

[a] Escuela Técnica Superior de Ingeniería Informática, Universidad de Sevilla, Av. Reina Mercedes s/n, Sevilla, Sevilla 41012, Spain
[b] Institute of Computer Science, Cracow University of Technology, Warszawska 24, Cracow 31-155, Poland
[c] Research and Academic Computer Network (NASK), Kolska 12, Warsaw 01-045, Poland

ARTICLE INFO

Article history:
Received 28 November 2017
Revised 3 January 2018
Accepted 6 January 2018
Available online 12 January 2018

Keywords:
Data center
Energy saving
Cloud computing
Energy-aware scheduling

ABSTRACT

Achieving efficiency both in terms of resource utilisation and energy consumption is a complex challenge, especially in large-scale wide-purpose data centers that serve cloud-computing services. Simulation presents an appropriate solution for the development and testing of strategies that aim to improve efficiency problems before their applications in production environments. Various cloud simulators have been proposed to cover different aspects of the operation environment of cloud-computing systems. In this paper, we define the SCORE tool, which is dedicated to the simulation of energy-efficient monolithic and parallel-scheduling models and for the execution of heterogeneous, realistic and synthetic workloads. The simulator has been evaluated through empirical tests. The results of the experiments confirm that SCORE is a performant and reliable tool for testing energy-efficiency, security, and scheduling strategies in cloud-computing environments.

1. Introduction

Cloud-computing (CC) and large-scale web services have had a notable impact in the data center scenario and the big-data environment, since they enable huge amounts of data to be processed in a reliable and distributed way. However, the CC services in the big-data era should meet new requirements from the end users, such as fast-response and low latency.

Major CC service providers, such as Google, Microsoft and Amazon, are constantly developing new applications and services. As the number of these services grows, many types of applications have to be deployed on the same hardware. Virtualisation of the resources enabled the resource utilisation to be improved by designing general and wide-purpose data centers. These facilities can handle an enormous workload with various requirements. Therefore, many well-known solutions, such as fragmenting the data center into a set of clusters that are responsible for executing only one kind of application, are no longer needed.

Large-scale data centers which execute heterogeneous workloads on shared hardware resources bring new challenges in addition to those inherent to small and medium-sized clusters, not least because scheduling such an amount of work may exceed the capacity of a centralized monolithic scheduler. In order to overcome this limitation, several scheduling models

[*] Corresponding author.
E-mail addresses: damiancerero@us.es (D. Fernández-Cerero), afdez@us.es (A. Fernández-Montes), agneskrok@gmail.com (A. Jakóbik), joanna.kolodziej@nask.pl (J. Kołodziej), migueltoro@us.es (M. Toro).

https://doi.org/10.1016/j.simpat.2018.01.004
1569-190X/© 2018 Elsevier B.V. All rights reserved.

with different degrees of parallelism have been developed, such as the two-level approach of Mesos [1], the shared-state approach of Omega [2], and various approaches for scheduling in large-scale grid systems [3].

The aforementioned infrastructures usually consume as much energy as do many factories and small cities, and account for approximately 1.5% of global energy consumption [4].

Although, data centers may be used by users worldwide, they are usually deployed on a continental basis. Therefore, these facilities are usually under higher pressure during day-time hours than in night-time hours. This and other reasons, such as the fear of any change that could break operational requirements [5], and the complexity of the systems involved, lead to an over-provision of data-center infrastructures. This decision leads to servers being kept underused or in an idle state, which is highly inefficient from an energy-consumption perspective.

Many models have been implemented in order to minimise the energy consumption in data centers, such as chiller-free cooling systems, and hardware model improvements, such as dynamic voltage and frequency scaling (DVFS). However, another approach that may lower the energy consumption considerably in data centers is seldom implemented. This strategy involves switching off idle servers.

The strategies that shut down idle machines may have a negative impact in terms of performance of the whole system. The inactive machines would not be able to execute large workloads in a short time. Therefore, the optimal energy-aware strategies must guarantee an appropriate level of reduction of energy consumption. In order to test these strategies and to measure the impact in terms of performance and energy consumption, a trustworthy simulation tool is required. In addition, the chosen simulator has to be able to reproduce the conditions present in real data centers. This requirement is critical, since the simulation is usually the step prior to implementing these strategies in working data centers. Therefore, the "optimal" energy-aware cloud simulator should guarantee the following achievements: a) Low resource consumption – data centers are composed of thousands of data servers which consume a huge amount of energy; b) Simulation of the parallel-scheduling models – the monolithic-scheduling model may prove ineffective in the large-scale CC systems; c) Easy codes for replication and extensions –generic model focused on shut-down, power-on and scheduling strategies, since many low-level aspects, such as hardware and networking details, may be overlooked, and d) Validation of results against a trustworthy source. This means that the simulation tool has been compared with the real-life system that it simulates.

Several simulators were evaluated in this paper. These tools followed various simulation models, such as: a) Discrete-event systems; b) Multi-agent systems; c) Multi-paradigm systems; and d) Hierarchic systems. The simulators under evaluation presented a wide range of purposes and levels of detail. However, by using these simulators, it is very difficult to achieve the aforementioned properties. In this paper, we propose a new data-center simulation tool, namely the *SCORE Cloud simulator*, which is our proposal for an "optimal" energy-aware CC simulation package. SCORE is based on the Google Omega lightweight simulator [2], which was extended by the implementation of the hybridisation of the discrete-event and the multi-agent scheduling and resource utilisation models. The Google Omega lightweight simulator has been validated against Google data centers, which makes it suitable for being the core of a trustworthy simulation tool. This is critical, since the authors have not been able to validate SCORE against real-world data centers due to the huge size of the clusters taken into consideration.

The paper is organised as follows. In Section 2, a simple comparative analysis is presented of the most relevant cloud computing simulators available. In Section 3, the high-level architectural decisions of SCORE are highlighted. The scheduling models under evaluation are described in Section 3.1. In Section 3.3, the core modules of SCORE, which are related to energy efficiency, are described. A number of the parameters available for the configuration of the experimentation are shown in Section 4. In Section 4.2, the workload employed to perform the experimentation in SCORE is characterised. In Section 5, the data available as a result of the experiments performed in SCORE, as well as the means to retrieve this data are explained. The various experimentation scenarios and the result parameters are shown in Section 6. Finally, the conclusions and future work are discussed in Section 7.

2. Related Work

In recent years, many simulators have been developed for the modelling of the main components of the computational cloud systems. In this section, a simple survey and comparative analysis is provided. The following basic trade-offs should be considered when developing a simulation tool: a) Performance versus features; b) Performance versus accessibility; c) Accessibility versus features; and d) Performance versus accuracy.

Although visual general-purpose simulators, such as Insight Maker [6], could be used to simulate computational cloud systems, the development of the model of an energy-efficient cloud-computing infrastructure, such as that described earlier, would be failure-prone and over-sized.

On the other hand, there are other non-general-purpose simulators, such as ElasticTree [7], and CloudSched [8], which are focused on the energy consumption of networking elements and scheduling policies, respectively. Due to this specialisation, these simulators present major limitations and restrictions.

In addition, we evaluated wide-ranging cloud-computing simulation tools, which cover more elements of the Cloud-Computing systems (CC). Each of the frameworks studied simulates different aspects of the CC systems to a different degree of detail. These modelling and implementation decisions render each system different in terms of performance and features provided, which, in turn, makes some of them more suitable for the aforementioned purpose. This class of simulators includes:

Table 1
Simulator comparison.

Simulator	Scheduling models	Energy aware	Energy strategies	Scheduling policies	Performance
GridSim	N	N	N	Y	Medium
CloudSim	N	Y	Y	Y	Medium
GreenCloud	N	Y	Y	Y	Low
Google Omega paper	Y	N	N	N	High
Grid'5000 Toolbox	N	Y	Y	N	High

- **GridSim**. This toolkit [9], based on the SimJava library [10], models various aspects of grid systems, such as users, machines, applications, and networks. However, it fails to consider several cloud-computing features. It also lacks the energy-consumption perspective.
- **CloudSim**. This popular cloud simulator is based on SimJava and GridSim and is mainly focused on IaaS-related operation environment features [11]. It presents a high level of detail, and therefore allows several VM allocation and migration policies to be defined, networking to be considered, features and energy consumption to be taken into account. However, it features certain disadvantages when applied for the simulation of large data-center environments: this high level of detail means that CloudSim is considered cumbersome to execute, especially for data centers composed of thousands to tens of thousands of machines. In addition, it is not principally designed to simulate multiple scheduling models, but takes a largely a monolithic approach.
- **GreenCloud**. This simulator is an extension of the NS2 network simulator. Its purpose is to measure and compute the energy consumption at every data center level, and it pays special attention to network components [12]. However, its packet-level nature compromises performance in order to raise the level of detail, which may be not optimal for the simulation of large data centers. In addition, it is not designed to offer ease of development and extension in various scheduling models.
- **Google Omega lightweight simulator**. This simulator is designed for the comparison of various scheduling models in large clusters. To this end, it focuses on maximizing the performance of the simulations by reducing the level of detail. However, it is not designed to easily develop and extend other scheduling strategies. In addition, this tool fails to consider energy consumption.
- **Grid'5000 Toolbox**. Grid'5000 was built upon a network of dedicated clusters. The infrastructure of Grid'5000 is geographically distributed over various sites, of which the initial 9 are located in France. Grid'5000 Toolbox [13] simulates the behaviour of Grid'5000 resources for real workloads while changing the state of the resources according to several energy policies. The simulator includes:
  a) A GUI that allows the user to perform a set of simulations for each location and to execute a set of energy policies; b) A graphical visualisation of the resources during the simulation, including their states, and future and past jobs; c) A graphical view of the results through several charts and spreadsheets.
  On the other hand, the simulator fails to include various scheduling frameworks and it does not simulate the behaviour or consumption of network devices and resources.

Several of these simulators are well-known, and in-depth comparisons have been presented in the literature [14–17]. In this work, the most important aspects regarding the development and application of power-off/on strategies in order to minimise the energy consumption are presented. Among these:

- **Scheduling models**: This parameter reflects whether the simulation tool has different scheduling models implemented, such as parallel, distributed, and monolithic approaches. In addition, this parameter also considers whether the tool is designed to easily extend and develop new scheduling frameworks, and not only allocation policies.
- **Energy aware**: This parameter reflects whether the simulation tool is capable of measuring and computing energy consumption and efficiency parameters.
- **Shut-down and Power-on policies**: This parameter reflects whether the simulation tool has different shut-down and power-on algorithms implemented. In addition, it also considers whether the tool is designed to easily extend and develop new energy policies.
- **Scheduling strategies**: This parameter reflects whether the simulation tool has different allocation/scheduling algorithms implemented. In addition, it also considers whether the tool is designed to easily extend and develop new strategies.
- **Performance**: This parameter reflects the amount of time for a simulation to be run in a comparable environment. The amount of computational and memory resources is also considered.

A short comparative between the simulation tools described are presented in Table 1.

The simulator described in this work, SCORE, is a high-performance cloud-computing simulation tool focused on energy-efficiency in large data centers. This tool is designed to be easily extended, and offers several strategies already implemented and ready to use, for various scheduling models, allocation, and shut-down and power-on strategies. The high performance

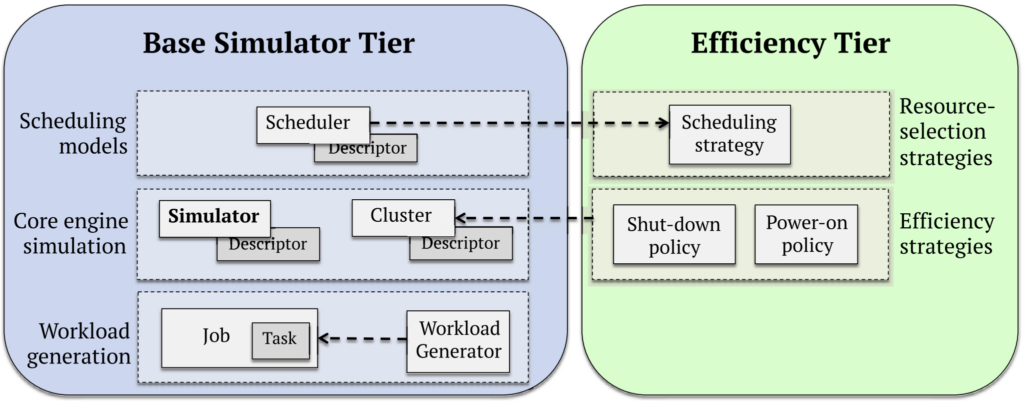


Fig. 1. SCORE architecture.

and ease of use have been achieved by minimising the low-level features, such as networking, low-level machine details, and low-level task details.

3. SCORE Architecture

The SCORE simulator developed herein is based on the model proposed by Robinson in [18]. The main workflow of that simulator can be defined as follows: 1. Definition and initialisation of the problem; 2. Determination of the modeling and general objectives; 3. Identification of the model inputs; 4. Identification of the model outputs; and 5. Determination of the model content and level of detail.

Based on this workflow model in CC, we developed the SCORE architecture as presented in Fig. 1. The SCORE source code is publicly available at:

https://github.com/DamianUS/cluster-scheduler-simulator.

The architectural model is composed of two main modules, namely *the core simulator – (CS)* and *the efficiency module – (EF)*. The core simulator is the core execution engine, and it has been inherited from the Omega lightweight simulator as explained in Section 2. Although the main layers in the *CS* model are based on the same structure present in the original simulator, we made several modifications in order to be able to perform the experimentation related to the security and energy-efficiency. Each experiment is a set of executions and each execution defines all the operational environment details, such as energy policies and scheduling models.

The architecture of *CS* is a 3–layer architecture with the *Workload generation* as the first layer, which is responsible for the generation of the CC workload that will be used in every run of a single experiment. The workload is created only once. This is critical, since it enables the parameters and other aspects used within each run to be reliably compared. The workload is generated by the various data-center utilisation patterns. For instance, the day/night pattern is the predominant pattern in the data centers that execute large web services and applications, since this is common human behaviour.

A workload is composed of a set of *Jobs*. In the same way, a *Job* is composed of a *Bag of Tasks*. A *Task* is the minimum execution unit that may be deployed on a computational server. Each *Task* is mapped to a linux container which is deployed in a similar (but more lightweight) way as a virtual machine (VM). This modern and more flexible virtualisation strategy replaces the traditional virtualisation strategy based on independent virtual machines. Each *Task* deployed on a linux container requires a given amount of computational and memory resources for a given time to be successfully completed. In the current version of this simulator, no linux-container migration nor server consolidation strategy is considered. Thus, once a *Task* is deployed on a server, it runs on this machine until its completion.

The *Core Engine Simulation* layer performs all the simulation duties, reads the workload generated, and performs the scheduling decisions to deploy the tasks on the worker nodes. The *Cluster* is defined by its *Descriptor* and represents the number of computational servers and their features. The current version of SCORE does not provide any networking capabilities due to two main reasons: a) The main goal of this work is to provide a performant and low resource-consuming tool capable of simulating large-scale data centers. These requirements impose serious restrictions regarding the level of detail of the developed features. b) There are several simulation tools that focus on networking details, as stated in Section 2.

Finally, the *Scheduling models* layer implements various scheduling frameworks, such as Omega [2], Mesos [1], and Monolithic models. These schedulers perform the resource-allocation process, and dictate the scheduling decisions to the *Core Engine Simulation* layer.

We developed the *EF* module in order to apply several energy-efficiency and resource-selection strategies. Therefore, this module is logically divided into two layers of the same name. The *Efficiency strategies* layer defines the policies for shutting-down and powering-on computational servers with the objective of optimising the energy consumption of the data

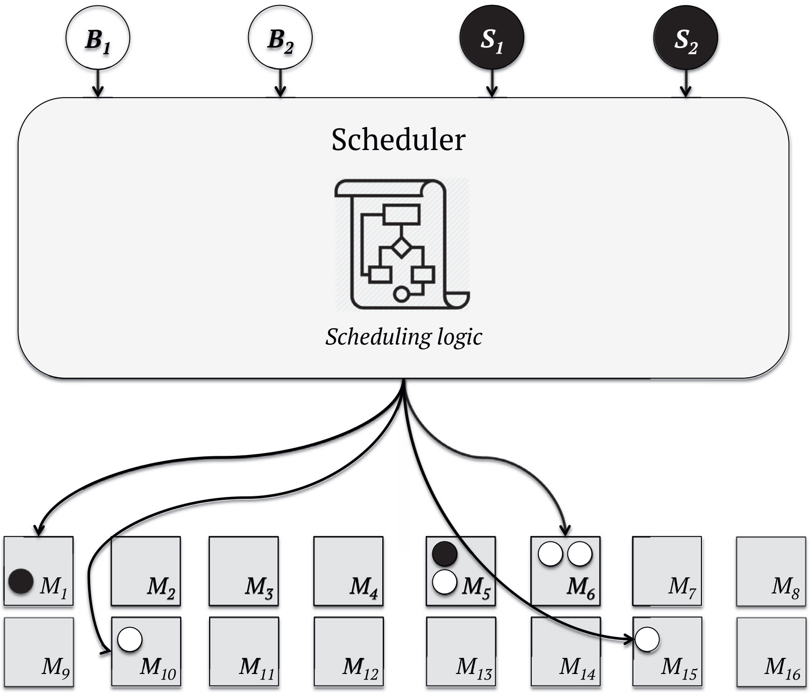


Fig. 2. Monolithic scheduler architecture, B - Batch-type task, S - Service-type task, M - Worker Node.

center. These policies can act over the *Cluster*, changing the energy state of the working nodes from powered-on to shut-down and viceversa. Finally, in the *Resource-selection strategies* layer, several approaches for the reservation of resources are implemented, such as maximizing the dispersion of tasks, deploying them randomly, and minimising the dispersion of tasks. These selection decisions therefore have an impact on the overall performance and energy-saving results.

3.1. Scheduling models

- **Monolithic schedulers**: In this model, a single, centralized scheduling algorithm is employed for all jobs. Google Borg [19] is an example of this kind of scheduling model. The workflow of the monolithic scheduler is demonstrated in Fig. 2.
- **Two-level schedulers**: In this model, the resource allocation and task placement concerns are separated. There is a unique, centralised active resource manager that offers computing resources to multiple parallel, independent, application-level scheduling nodes, as shown in Fig. 3. This approach allows the task-placement logic to be developed for every single application, but also allows the cluster state meta-data to be shared between these schedulers. Mesos [1] is an example of this kind of scheduler.
- **Shared-state schedulers**: In this model, the cluster meta-data is shared between all scheduling agents. The scheduling process is performed by using an out-of-date copy of this shared cluster meta-data. When one of these parallel schedulers performs a scheduling decision based on the probably stale cluster meta-data, the scheduling agent commits the scheduling decision as a transaction in an optimistic way, as shown in Fig. 4. Hence, if any of the scheduling operations committed cannot be applied because the chosen computing resources are no longer free, then that scheduling operation is repeated by the scheduler until no conflicts are found. Google Omega is an example of this kind of model.
- **Fully-distributed schedulers**: In this model, the scheduling frameworks have various independent scheduling nodes which work with a local and out-of-date vision of the cluster state with no central coordination. Sparrow [20] is an example of this kind of scheduler.
- **Hybrid schedulers**: In this model, several scheduling strategies are used (typically a fully distributed architecture is combined with a monolithic or shared-state design) depending on the workload. There are usually two scheduling paths: a distributed path for short or batch tasks, and a centralized path for the remaining tasks. Mercury [21] provides an example of this kind of scheduler.

3.2. SCORE energy-awareness model

SCORE has been developed to enrich the Google Omega Lightweight Simulator. A main feature included is the capability of performing energy-efficiency analysis by applying an energy-consumption model. The CPU is considered in order to compute the energy consumption. The proposed energy-awareness model considers the following states for each CPU core in a

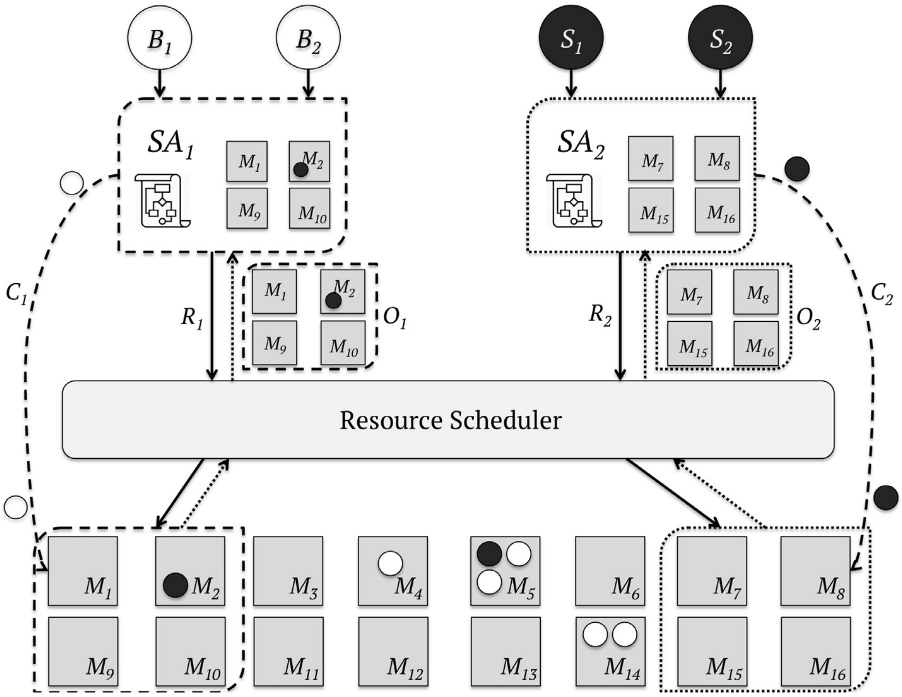


Fig. 3. Two-level scheduler architecture, SA - Scheduler Agent, O - Resource offer, C - Commit.

server: a) *On:* 150W b) *Idle:* 70W. The energy consumption is linearly computed in terms of the utilisation of each CPU core. The following machine power states have also been considered: a) *Off:* 10W b) *Shutting down:* 160W * number of cores c) *Powering on:* 160W * number of cores.

The total energy consumed by the whole data center is measured from time to time by checking the power state of every machine. This time interval is a configurable parameter.

Regarding the shut-down process time parameters, the following values have been assumed: a) $T_{On \rightarrow Hibernated}$: 10s, and b) $T_{Hibernated \rightarrow On}$: 30s. The power states and transitions are shown in Fig. 5. All the aforementioned power and time parameters can be modified for each experiment.

3.3. SCORE energy-efficiency modules

As aforementioned, the energy-efficiency tier is composed of the following three modules:

- **Shut-down module**: This module is responsible for shutting down the computational servers in order to minimise energy consumption. Several strategies may be used in order to shut down the machines. Each shut-down strategy is implemented in the form of a *Shut-down policy*.
- **Power-on module**: This module is responsible for waking up the machines required to meet present or future workload demands. Several strategies may be used in order to minimise the negative performance impact caused by machines that are not available to immediately execute tasks because they are shut down. Each of these strategies is implemented in the form of a *Power-on policy*.
- **Scheduling module**: This module is responsible for determining which tasks should be deployed on which machine.

3.3.1. Shut-down policies

Power-off policies are responsible for deciding whether or not a machine should be shut down and responsible for triggering the order for the shut-down operation.

In this work, authors have divided the process into making a decision of whether a shut-down action must be taken, and carrying on the actual action of ordering the shut-down of the machine in order to allow various combinations of strategies to be performed. The workflow of this process is illustrated in Fig. 6.

Shut-down decision policies can be: deterministic, such as always shutting-down; or probabilistic, such as shutting down machines following the exponential policy. These decision policies always return a Boolean value which determines whether a given machine must be shut down. In order to make the decision, this policy may check various *Cluster* variables after having finished a task and having freed the resources.

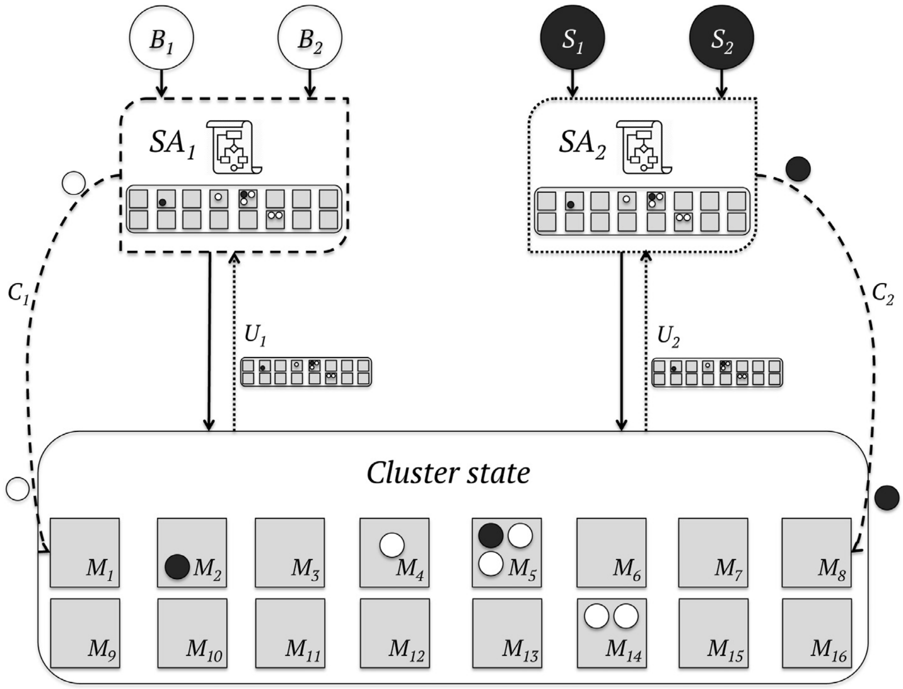


Fig. 4. Shared-state scheduler architecture, U-Cluster State Update.

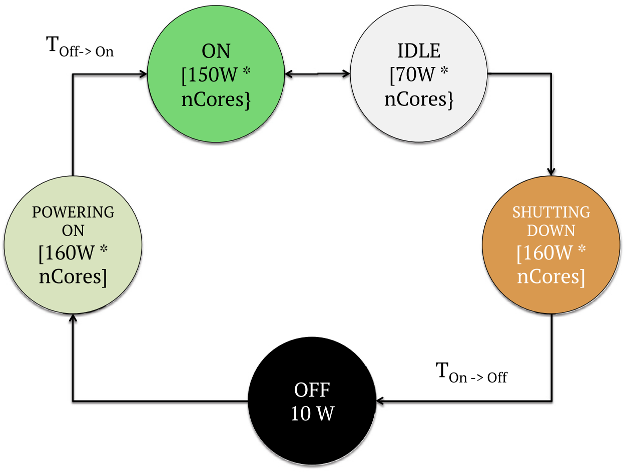


Fig. 5. Machine power states.

Several shut-down policies have been implemented and tested in-depth, as shown in Section 6. These policies include: *Never power off, Always power off, Shut down depending on data-center load, Leave a security margin, Exponential*, and *Gamma*. In addition, various power-off decision policies can be combined by using the logical operators *and* and *or* to achieve policies of a more flexible and complex nature.

3.3.2. Power-on policies

Power-on policies are responsible for maintaining sufficient resources available in order to properly execute the arriving jobs. As the complement to shut-down policies, the strategies developed are critical to guarantee that heterogeneous workloads and peak loads can be executed without causing a negative impact in the overall data-center performance, without breaking SLAs/SLOs, and without affecting the user experience.

As opposed to power-off policies, which make a decision and perform an action independently for each machine, power-on policies work with the overall cell state in order to turn on multiple machines if required by the workload. This power-on

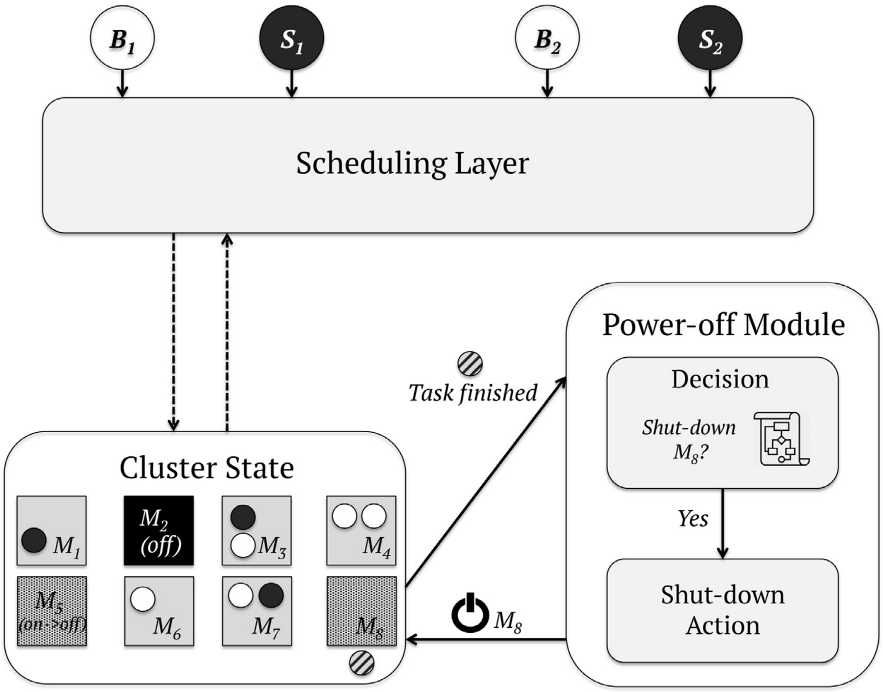


Fig. 6. Power-off module architecture.

business logic is conceptually executed when a new job is scheduled, as opposed to the shut-down business logic, which is executed when resources are released. Of course, each scheduler has different scheduling processes, so powering-on cycles can therefore vary depending on the scheduler. In order to make power-on decisions, the system may require several items of operational information, such as: a) The *Job* that triggered the power-on action; and b) The scheduler model that triggered the power-on action.

In order to be realistic, this simulation tool usually works with a heterogeneous workload with no evident usage pattern. This kind of workload is really complex to predict, and therefore shut-down policies may negatively impact the data-center performance if they are unable to predict near-future workload requirements properly.

Various deterministic and probabilistic power-on decision and action policies have been developed to face this challenge. These policies include: *Never power on, Power-on only the required machines, Power-on a fixed number of machines to maintain a security margin, Power-on a percentage of machines to maintain a security margin*, and power-on policies which make decisions based on statistical distributions, such as *Exponential*, and *Gamma*.

In addition, it is especially interesting that various power-on decision policies may be combined using the logical operators *and* and *or* to achieve policies of a more flexible and complex nature. With this strategy, the benefits of predictive policies may be enjoyed without giving up the possibility of turning on the required machines if the prediction fails or a peak load arrives.

3.3.3. Scheduling strategies

SCORE designs the scheduling strategy as a plug-in piece that is established at the experiment creation time. This scheduling strategy is used by all the schedulers of all scheduling models in order to determine on which machines the tasks should be deployed. Thus, the scheduling strategy works as a black box which uses the information of the whole cluster and of the job for these schedulers, and returns them the mapping between tasks and the machine to be applied. Once this mapping is available, each scheduling model deploys these tasks on the chosen machines depending on their own scheduling logic, as illustrated in Fig. 7.

Several scheduling strategies have been developed. These strategies include: those based on the ETC-matrix genetic process [22], such as *ETC minimising makespan* [23], and *ETC minimising energy* [24], *Random, Spread tasks the maximum, Greedy minimising energy, Greedy minimising makespan, Spread tasks the minimum, Spread tasks the minimum with randomness.*

4. Experiment analysis

4.1. General parameters

Table 2 presents the key parameters of the simulator used in experiments.

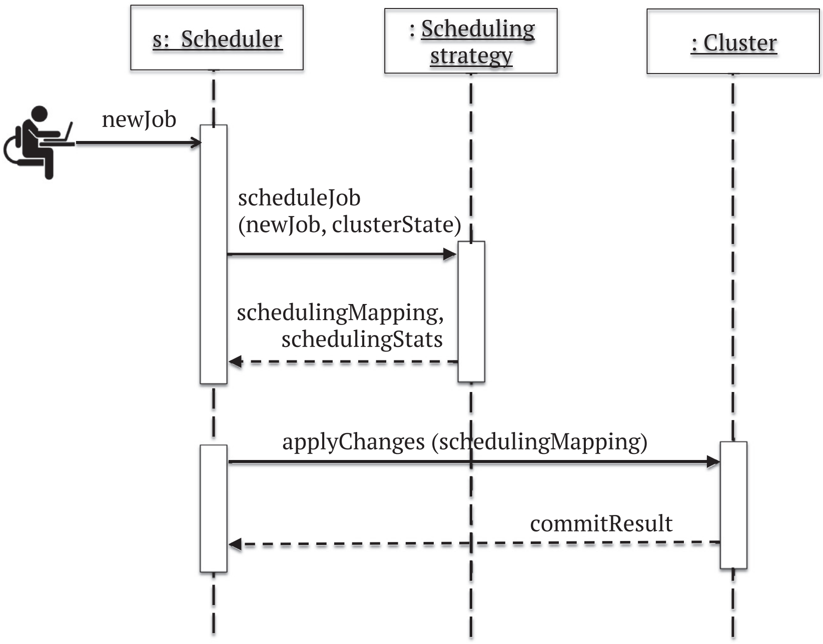

s: Scheduler
: Scheduling strategy
: Cluster
newJob
scheduleJob (newJob, clusterState)
schedulingMapping, schedulingStats
applyChanges (schedulingMapping)
commitResult

Fig. 7. Scheduling-strategy workflow.

4.2. Workload parameters

In order to perform realistic experimentation, the workload present in Google traces [25] is considered. Based on the studies made by the research community in [26,27], it is known that realistic jobs are composed of one or more tasks, sometimes thousands of tasks. In addition, two types of jobs are to be considered:

- **Batch jobs**: This workload is composed of jobs which perform a computation and then finish. These jobs have a determined start and end. MapReduce jobs are an example of a *Batch* job.
- **Service jobs**: This workload is composed of long-running jobs which provide end-user operations and infrastructure services. As opposed to *Batch jobs*, these jobs have no determined end. Web servers or services such as BigTable [28] are good examples of a *Service* job.

In order to properly define the workload and create the model for the generated jobs, the following job attributes are considered: a) Inter-arrival time, which represents the time elapsed between two consecutive *Service* jobs or between two *Batch* jobs; b) Number of tasks, which is usually higher for *Batch* jobs than for *Service* jobs; c) Job duration, which may be modified by the machine performance profile; and d) Resource usage, which represents the amount of CPU and RAM that every task in the job consumes [2].

4.2.1. Workload generation

Regarding the generation of the workload, several approaches are implemented and may be used, ranging from uniform generators, followed by several degrees of exponential generators and generators that rely on traces to model the tasks. This workload is generated at the beginning and used for all the experiments of that execution. The following workload generators are available in SCORE:

- **Uniform**: This strategy generates jobs at a uniform rate, of a uniform size and which consume the same amount of resources.
- **Exponential**: This approach generates workloads with jobs that have the inter-arrival time, the number of tasks, and the task duration sampled from exponential distributions. Two versions of the *Exponential* generator have been implemented in order to create both a flat and a day/night patterned workloads.
- **Exponential built from a trace file**: This generator creates workloads where the duration and the number of tasks of all jobs are sampled from exponential distributions built from a trace file. In addition, the *Exponential* and the *Exponential built from a trace file* can be chosen on a per-parameter basis.
- **Trace file**: This approach generates workloads that reproduce a trace file.

Trace-related workload generators require a trace file that contains one job per line. Each line must present the following columns separated by a whitespace: 1. Submission time; 2. Number of tasks; 3. Job duration; 4. Number of CPU cores required by the tasks; and 5. Amount of RAM required by the tasks.

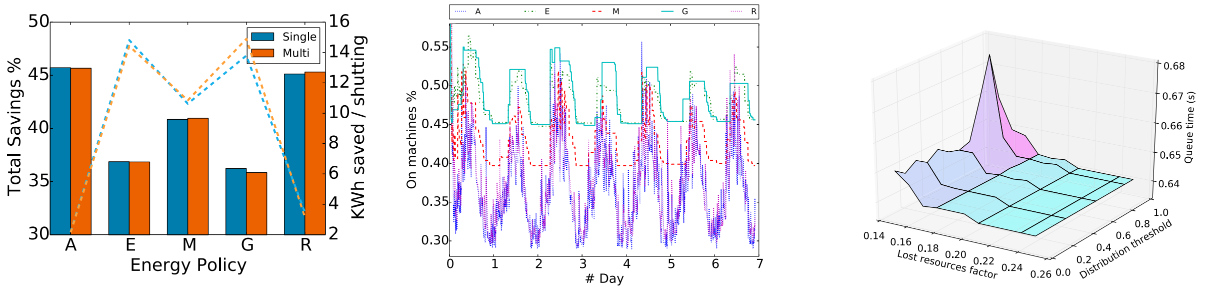
Table 2
Configurable experiment parameters.

Parameter	Description	Values
Cluster	Parameters related to the data center that must be fixed for all experiments	
#Machines	Data-center size	[1 - ∞]
#Cores	Number of CPU cores for every machine	[1 - ∞]
RAM	Amount of RAM in GB for every machine	[0.1 - ∞]
Heterogeneity	Flag to decide whether data-center machines are heterogeneous	Boolean
Machine performance profile	Describe the performance of every machine in the data center. The lower this value, the more performant the server is	Array, size: number of machines [0.01 - ∞]
Machine security profile	Describe the security of every machine in the data center. The higher this value, the more secure the server is [23]	Array, size: number of machines [1 - 5]
Machines energy profile	Describe the energy consumption of every machine in the data center. The lower this value, the more energy-efficient the server is	Array, size: number of machines [0.01 - ∞]
Power-on time	The time required to boot a server, in seconds	[0.1 - ∞]
Shut-down time	The time required to hibernate a server, in seconds	[0.1 - ∞]
Performance	Parameters related to performance iterated in order to create all experiment variations	
Per-job algorithm time	Time spent (in seconds) by the scheduler in order to make a job-level scheduling decision. This simulates the performance of the scheduling algorithm	Array [0.001 - ∞]
Per-task algorithm time	Time spent (in seconds) by the scheduler in order to make a task-level scheduling decision. This simulates the performance of the scheduling algorithm	Array [0.001 - ∞]
Blacklist	The percentage of machines not to be used	Array [0.0 - ∞]
Inter-arrival	This parameter rewrites the inter-arrival time generated for all jobs, replacing it with a fixed time instead	Array [0.001 - ∞]
Energy	Parameters related to performance iterated in order to create all experiment variations	
Shut-down policies	The shut-down policies to be run, such as: *Always power off, Exponential*, and *Gamma*	Array
Power-on	The power-on policies to be run	Array
Scheduling	The scheduling strategies to be run	Array
Sorting	These strategies are used by greedy scheduling strategies to sort the candidate servers for their later selection	Array
Specific	Parameters used by specific schedulers	
Schedulers assigned to workloads	Mapping that describes how many and which schedulers are assigned to which workload type (Batch/Service) used by non-monolithic schedulers. Each row adds a new scheduler to serve a workload	Map [Scheduler name -> Workload name]
Conflict mode	Approach used by non-monolithic schedulers to decide whether a *commit* results in conflict	resource-fit sequence-numbers
Transaction mode	Approach used by non-monolithic schedulers to decide what to do when a *commit* results in conflict	all-or-nothing incremental

5. Information retrieval

The results of the experiments are stored in one to several Google protocol buffer files. The main blocks of these protocol buffer files include the following:

- **Experiment Environment** section stores the information related to the global configuration of the experiments.
- **Experiment Results** section stores the information related to global results for each of the experiments performed.
- **Workload Stats** section stores the workload-specific information for both *Batch* and *Service* jobs for each *Experiment Result*, including the details of the genetic process when used.
- **Scheduler Stats** section stores the scheduler-specific information for each *Experiment Result*, including daily and workload-related details.



(a) Energy savings vs. kWh saved per shut-down

(b) # Powered-on machine evolution for multiple policies

(c) Queue-time detail for Exponential policy

Fig. 8. Simulation results graphic scripts.

Table 3
Experiment outputs.

Parameter	Description	Values
Performance	Output parameters related to the data center overall and per-workload performance	
Queue time until first deploy	Represents the time a job waits in the queue until its first task is scheduled (in seconds).	[0.0 - ∞]
Queue time until full deploy	Represents the time a job waits in the queue until it is totally scheduled (not completion).	[0 - ∞]
Timed-out jobs	Number of jobs left unscheduled after 100 unsuccessful scheduling tries for a given job or 1000 tries for any given task in a job.	[0.0 - ∞]
Scheduler occupation	Percentage of scheduler utilisation on average	[0.0 - 100.0]
Job scheduling attempts	Number of scheduling operations needed to fully deploy a job.	[0 - ∞]
Task scheduling attempts	Number of tasks scheduling operations needed to fully deploy a job.	[0 - ∞]
Energy-efficiency	Output parameters related to the data-center resource and energy efficiency.	
Energy consumed	Total data-center energy consumption (in kWh)	[0.0 - ∞]
Energy saved vs. current system	Total energy saved by applying energy-efficiency policies compared to the same scenario with no energy-efficiency policies applied (in kWh).	[0.0 - ∞]
Shut-downs	Number of shut-down operations.	[0 - ∞]
Idle resources	Percentage of resources operating in an *Idle* state on average.	[0.0 - 100.0]
KWh saved per shut-down operation	This represents the energy saved against the number of shut-downs performed. It shows the *goodness* of the shut-down operations performed.	[0.0 - ∞]

- **Efficiency Stats** section stores the energy-efficiency parameters for each *Experiment Result*.
- **Measurements** section stores the performance and energy-efficiency metrics gathered in each measurement performed every few seconds for each *Experiment Result* in order to show the cluster evolution.

The information stored in the protocol buffer files can't be visualized directly. Hence, a set of python scripts, aimed to create valuable human-readable and graphic information from these files, have been implemented. The results of some of these graphic scripts are illustrated in Fig. 8.

5.1. Output indicators

Table 3 presents the most relevant results indicators of the experimentation performed in terms of performance and energy-efficiency.

6. Examples of usage

In this section, a set of experiments have been run in order to illustrate certain experiment parameterisation and results related to both performance and energy-efficiency. Although each experiment may show a different subset of parameters and results, all the parameters are used and returned in each experiment. In addition, several parameters have been fixed in order to keep the tests simple, such as always using the default power-on policy, which aims to power on machines when a workload cannot be served.

Each experiment runs for a 7-day operation period and processes a day-night patterned workload that uses, on average, approximately 30% of the resources, with load peaks achieving approximately 60% of data-center utilisation. The parameters of the jobs in this workload are generated by using an exponential distribution for the inter-arrival time, number of tasks, and duration, while the security constraints are generated randomly. Regarding these constraints, the following values are used: a) *Batch* jobs are composed of 50 tasks and *Service* of 9 tasks on average; b) *Batch*-job tasks take 90 seconds and *Service*-job tasks 2,000 seconds to finish on average; and c) *Batch*-job tasks consume 0.3 CPU cores and 0.5 GB of memory, while *Service*-job tasks consume 0.5 CPU cores and 1.2 GB of memory.

The data center is composed of 1,000 machines of 4 CPU cores and 8GB RAM. The energy, performance, and security profiles of these machines are randomly generated.

6.1. Genetic process experimentation

In this test, a set of experiments focused on illustrating certain results of the genetic process of the ETC-matrix-based scheduling strategies are run by using: a) a monolithic scheduling model, and b) fixed algorithm times.

The results corresponding to *Batch* jobs and the *Single-path monolithic scheduler* are presented in Table 4, where it can be observed that the genetic scheduling strategy focused on minimising the makespan achieves better performance results, and it is shown how this makespan average evolves between the genetic-process epochs.

6.2. Performance experimentation for the Omega scheduler

Several parameters are available for the evaluation of the impact on the operation environment of the energy-efficiency policies and algorithm performance. A number of these parameters are presented after having executed a new set of experiments by using the following simulation configuration: a) the Omega scheduling model, with 4 schedulers responsible for serving *Batch* jobs and 1 scheduler for *Service* jobs; b) one scheduling strategy which strives to maximise machine usage while minimising resource contention; and c) the *Resource-fit* conflict mode.

Table 4
Parameters of the minimising-makespan genetic process.

Shut-down	Scheduling	Savings (%)	Makespan Avg. (s)	Epoch 0 (s)	Epoch 100 (s)
Never off	Makespan	N/A	236.01	324.02	202.07
Never off	Energy	N/A	290.55	N/A	N/A
Random	Makespan	55.88	258.41	344.51	273.99
Random	Energy	55.31	311.91	N/A	N/A

Table 5
Omega performance experimentation.

Shut-down policy	Transaction mode	Per-job alg. time (s)	Per-task alg. time (ms)	Savings (%)	Queue time until first deploy (ms)	Queue time until full deploy (ms)
Never off	all-or-nothing	0.1	10	N/A	0.0	0.0
Never off	all-or-nothing	0.1	100	N/A	560.0	1,021.4
Never off	all-or-nothing	1.0	10	N/A	0.2	0.2
Never off	all-or-nothing	1.0	100	N/A	791.8	1,604.0
Never off	incremental	0.1	10	N/A	0.0	0.0
Never off	incremental	0.1	100	N/A	12.9	27.5
Never off	incremental	1.0	10	N/A	0.0	0.0
Never off	incremental	1.0	100	N/A	18.5	38.9
Always off	all-or-nothing	0.1	10	46.08	0.1	0.8
Always off	all-or-nothing	0.1	100	40.86	2,294.7	5,368.4
Always off	all-or-nothing	1.0	10	44.97	1.2	2.2
Always off	all-or-nothing	1.0	100	38.78	3,503.1	9,061.6
Always off	incremental	0.1	10	45.06	0.1	0.5
Always off	incremental	0.1	100	45.10	36.9	84.6
Always off	incremental	1.0	10	45.02	1.1	2.5
Always off	incremental	1.0	100	45.05	46.8	107.9

Table 6
Mesos energy-efficiency experimentation.

Shut-down policy	Per-job alg. time (s)	Per-task alg. time (ms)	Energy consumed (kWh)	Energy Saved (kWh)	# shut downs	Idle resources (%)
Never off	0.1	10	57,259	0	0	69.30
Never off	0.1	100	57,359	0	0	69.29
Never off	1.0	10	57,372	0	0	69.27
Never off	1.0	100	57,440	0	0	69.28
Always off	0.1	10	31,138	25,774	18,520	5.97
Always off	0.1	100	31,748	25,208	18,473	7.28
Always off	1.0	10	31,642	25,321	19,719	7.00
Always off	1.0	100	31,808	25,137	17,550	7.48
Random	0.1	10	32,003	24,949	9,136	7.97
Random	0.1	100	32,694	24,295	8,968	9.55
Random	1.0	10	32,702	24,225	8,816	9.82
Random	1.0	100	32,739	24,241	9,606	9.65
Exponential	0.1	10	34,952	21,842	1,156	16.43
Exponential	0.1	100	36,119	20,823	1,286	18.33
Exponential	1.0	10	35,953	20,951	1,214	17.87
Exponential	1.0	100	36,513	20,408	1,816	19.49

The results corresponding to *Batch* jobs are presented in Table 5, where it can be observed that the algorithm performance and the conflict-handling strategy have a notable impact both on queue times and on energy consumption.

6.3. Energy-efficiency experimentation for the Mesos scheduler

In addition to performance parameters, the energy-efficiency parameters constitute the core of the simulation tool. In order to illustrate these parameters, several experiments have been run by using the same configuration as laid out in Section 6.2 for the *Omega* scheduler.

The results corresponding to *Batch* jobs are presented in Table 6, where it can be observed that the shut-down policies exert a major impact on energy consumption and hardware stress, and that a high level of energy-efficiency can be achieved without performing many shut-down operations, thereby minimising the performance impact.

7. Summary and future work

Our simulation tool: SCORE is presented as an extension to the Google Omega lightweight simulator, a simulator focused on the comparison of the performance between scheduling models, especially parallel frameworks in large-scale data centers. The model of the Google Omega lightweight simulator has been enhanced with extensions and improvements for: a) The development of an energy consumption model; b) The extension and creation of allocation policies; c) The extension and creation of energy-efficiency policies based on shutting down and powering on machines; d) The addition of heterogeneity to data center machines; and e) The consideration of security profiles.

It has been shown that the application of all the features of this tool to heterogeneous workloads in large-data centers results in major potential improvements from both the energy-efficiency and performance point of view.

As future work, several aspects of the tool are being improved. These improvements include:

- Greater ease of use and extendability of the implemented code.
- Addition of a visual interface to set the experiment configurations and to execute the experiments.
- Addition of a real-time visualizer of the power and performance state of the machines.
- Incorporation of support for other workload constraints, such as time and precedence constraints.
- Tasks dependency, which implies, among others, networking considerations.
- Development of *Tasks* migration and server consolidation techniques.

Acknowledgement

The research is supported by the VPPI - University of Sevilla. This article is based upon work from COST Action IC1406 “High-Performance Modelling and Simulation for Big Data Applications” (cHiPSet), supported by COST (European Cooperation in Science and Technology).

References

[1] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A.D. Joseph, R.H. Katz, S. Shenker, I. Stoica, Mesos: A platform for fine-grained resource sharing in the data center., NSDI 11 (2011) 22.

[2] M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, J. Wilkes, Omega: flexible, scalable schedulers for large compute clusters, in: Proceedings of the 8th ACM European Conference on Computer Systems, ACM, 2013, pp. 351–364.

[3] J. Kołodziej, Evolutionary hierarchical multi-criteria metaheuristics for scheduling in large-scale grid systems, 419, Springer, 2012.
[4] J. Koomey, Growth in data center electricity use 2005 to 2010, A report by Analytical Press, completed at the request of The New York Times 9 (2011).
[5] A. Fernández-Montes, D. Fernández-Cerero, L. González-Abril, J.A. Álvarez-García, J.A. Ortega, Energy wasting at internet data centers due to fear, Pattern Recognit. Lett. 67 (2015) 59–65.
[6] S. Fortmann-Roe, Insight maker: A general-purpose tool for web-based modeling & simulation, Simulation Model. Practice Theory 47 (2014) 28–45.
[7] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, N. McKeown, Elastictree: Saving energy in data center networks., in: Nsdi, 10, 2010, pp. 249–264.
[8] W. Tian, Y. Zhao, M. Xu, Y. Zhong, X. Sun, A toolkit for modeling and simulation of real-time virtual machine allocation in a cloud data center, IEEE Trans. Autom. Sci. Eng. 12 (1) (2015) 153–161.
[9] R. Buyya, M. Murshed, Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing, Concurrency Comput. 14 (13-15) (2002) 1175–1220.
[10] F. Howell, R. McNab, Simjava: A discrete event simulation library for java, Simul. Series 30 (1998) 51–56.
[11] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A. De Rose, R. Buyya, Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software 41 (1) (2011) 23–50.
[12] D. Kliazovich, P. Bouvry, S.U. Khan, Greencloud: a packet-level simulator of energy-aware cloud computing data centers, J. Supercomput. 62 (3) (2012) 1263–1283.
[13] A. Fernández-Montes, L. Gonzalez-Abril, J.A. Ortega, L. Lefèvre, Smart scheduling for saving energy in grid computing, Expert Syst. Appl. 39 (10) (2012) 9443–9450.
[14] K. Bahwaireth, E. Benkhelifa, Y. Jararweh, M.A. Tawalbeh, et al., Experimental comparison of simulation tools for efficient cloud and mobile cloud computing applications, EURASIP J. Inf. Secur. 2016 (1) (2016) 1–14, doi:10.1186/s13635-016-0039-y.
[15] W. Tian, M. Xu, A. Chen, G. Li, X. Wang, Y. Chen, Open-source simulators for cloud computing: Comparative study and challenging issues, Simul. Model. Pract. Theory 58 (2015) 239–254.
[16] C. Badii, P. Bellini, I. Bruno, D. Cenni, R. Mariucci, P. Nesi, Icaro cloud simulator exploiting knowledge base, Simul. Model. Pract. Theory 62 (2016) 1–13.
[17] G. Kecskemeti, Dissect-cf: a simulator to foster energy-aware scheduling in infrastructure clouds, Simul. Model. Pract. Theory 58 (2015) 188–218.
[18] S. Robinson, Conceptual modelling for simulation part ii: a framework for conceptual modelling, J. Oper. Res. Soc. 59 (3) (2008) 291–304.
[19] A. Verma, L. Pedrosa, M. Korupolu, D. Oppenheimer, E. Tune, J. Wilkes, Large-scale cluster management at google with borg, in: Proceedings of the Tenth European Conference on Computer Systems, ACM, 2015, p. 18.
[20] K. Ousterhout, P. Wendell, M. Zaharia, I. Stoica, Sparrow: distributed, low latency scheduling, in: Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, ACM, 2013, pp. 69–84.
[21] K. Karanasos, S. Rao, C. Curino, C. Douglas, K. Chaliparambil, G.M. Fumarola, S. Heddaya, R. Ramakrishnan, S. Sakalanaga, Mercury: Hybrid centralized and distributed scheduling in large shared clusters., in: USENIX Annual Technical Conference, 2015, pp. 485–497.
[22] A. Jakobik, D. Grzonka, J. Kolodziej, H. González-Vélez, Towards secure non-deterministic meta-scheduling for clouds, in: 30th European Conference on Modelling and Simulation, ECMS 2016, Regensburg, Germany, May 31 - June 3, 2016, Proceedings., 2016, pp. 596–602, doi:10.7148/2016-0596.
[23] A. Jakobik, D. Grzonka, F. Palmieri, Non-deterministic security driven meta scheduler for distributed cloud organizations, Simul. Model. Practice Theory (available online 4 November 2016). https://doi.org/10.1016/j.simpat.2016.10.011.
[24] A. Jakóbik, D. Grzonka, J. Kołodziej, Security supportive energy aware scheduling and scaling for cloud environments, in: European Conference on Modelling and Simulation, ECMS 2017, Budapest, Hungary, May 23-26, 2017, Proceedings., 2017, pp. 583–590, doi:10.7148/2017-0583.
[25] C. Reiss, J. Wilkes, J.L. Hellerstein, Obfuscatory obscanturism: making workload traces of commercially-sensitive systems safe to release, in: 3rd International Workshop on Cloud Management (CLOUDMAN), IEEE, Maui, HI, USA, 2012, pp. 1279–1286.
[26] O.A. Abdul-Rahman, K. Aida, Towards understanding the usage behavior of Google cloud users: the mice and elephants phenomenon, in: IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Singapore, 2014, pp. 272–277.
[27] S. Di, D. Kondo, C. Franck, Characterizing cloud applications on a Google data center, in: 42nd International Conference on Parallel Processing (ICPP), Lyon, France, 2013.
[28] F. Chang, J. Dean, S. Ghemawat, W.C. Hsieh, D.A. Wallach, M. Burrows, T. Chandra, A. Fikes, R.E. Gruber, Bigtable: A distributed storage system for structured data, ACM Trans. Comput. Syst. (TOCS) 26 (2) (2008) 4.

Energy Policies for Data-Center Monolithic Schedulers

Once the simulation tool was ready and its results empirically tested, it was the moment to work on the third research objective of this thesis dissertation: *"Proof that energy consumption in monolithic-scheduling data centers can be successfully reduced without notably impacting performance if the correct set of energy-efficiency policies based on the shut-down of idle machines are applied"*. To this aim, we put our focus on the development, study, testing, and analysis of a set of energy policies which consitute the core of this thesis dissertation. Cloud computing and data centers that support this paradigm are rapidly evolving in order to satisfy new demands. These ever-growing needs represent an energy-related challenge to achieve sustainability and cost reduction.

We defined an expert and intelligent system that applies various energy policies. These policies are employed to maximize the energy-efficiency of data-center resources by simulating a realistic environment and heterogeneous workload in a trustworthy tool.

The contributions include a deep description of the impact of 6 different power-off policies - applied at the resource manager level - in terms of performance and energy consumption on a well-defined, rich and realistic heterogeneous workload that follows the trends present in Google Traces by running a huge amount of experiments for centralized monolithic scheduling frameworks.

In addition, an environmental and economic impact of around 20% of energy consumption can be saved in high-utilization scenarios without exerting any noticeable impact on data-center performance if an adequate policy is applied.

This work was published in *Expert Systems with Applications.* This Journal is indexed in JCR with an **Impact Factor of 3.768**. The Journal stands in ranking **Q1** in three categories: Computer Science, Artificial Intelligence (20/132), Engineering, Electrical Electronic (42/260), and Operations Research & Management Science (8/83).

Expert Systems With Applications 110 (2018) 170–181

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Energy policies for data-center monolithic schedulers

Damián Fernández-Cerero*, Alejandro Fernández-Montes, Juan A. Ortega

Department of Computer Languages and Systems, University of Seville, Av. Reina Mercedes S/N, Seville, Spain

ARTICLE INFO

Article history:
Received 5 December 2017
Revised 15 May 2018
Accepted 3 June 2018
Available online 4 June 2018

Keywords:
Energy policies
Efficiency
Data center
Simulation software
Decision making

ABSTRACT

Cloud computing and data centers that support this paradigm are rapidly evolving in order to satisfy new demands. These ever-growing needs represent an energy-related challenge to achieve sustainability and cost reduction. In this paper, we define an expert and intelligent system that applies various energy policies. These policies are employed to maximize the energy-efficiency of data-center resources by simulating a realistic environment and heterogeneous workload in a trustworthy tool. An environmental and economic impact of around 20% of energy consumption can be saved in high-utilization scenarios without exerting any noticeable impact on data-center performance if an adequate policy is applied.

1. Introduction

Cloud computing and large-scale web services have transformed the data-center scenario and the big-data environment, and have led to a new scenario where these infrastructures are as energy greedy as many factories. The latest estimations claim that data centers account for approximately 1.5% of global energy consumption (Koomey, 2011).

In this new scenario, data centers are in constant evolution towards servicing multiple heterogeneous workloads on the same hardware resources. This strategy enables higher energy-efficiency levels to be achieved by turning off idle resources in low-utilization periods. Decision-support systems are one of the main applications for expert systems. This work presents an automated decision-support system aimed to make the best decisions to improve the energy efficiency of the system through a better management of data-center resources and jobs placement. We develop, apply, and analyze various energy policies based on shutting machines off in order to reduce data-center energy consumption while preserving the cluster performance.

This approach has yet to be widely applied due to various reasons, such as: (a) Natural human behaviour and the fear of any change that could break operational requirements (Fernández-Montes, Fernández-Cerero, González-Abril, Álvarez-García, & Ortega, 2015); (b) the complexity and heterogeneity of all the subsystems involved; and (c) power-off policies, and (d) the fast development of new systems and paradigms that could break the established standards and systems. However, keeping servers underutilized or in idle state is highly inefficient from an energy-efficiency perspective.

On the other hand, the research community has made many efforts in other areas in order to achieve energy proportionality (Jakóbik, Grzonka, Kolodziej, Chis, & González-Vélez, 2017), such as: data-center operating temperature and cooling systems (El-Sayed, Stefanovici, Amvrosiadis, Hwang, & Schroeder, 2012; Sharma, Bash, Patel, Friedrich, & Chase, 2005), hardware energy proportionality (Fan, Weber, & Barroso, 2007; Miyoshi, Lefurgy, Van Hensbergen, Rajamony, & Rajkumar, 2002), upgrading hardware pieces such as HDDs to operate with non-mechanical devices such as SSDs (Andersen & Swanson, 2010), and improving power distribution infrastructures (Femal & Freeh, 2005) that have been put into production in various data centers from top-tier companies such as Google, Microsoft, and Amazon.

The paper is organized as follows. The related work is described in Section 2 and various powering-off resources strategies are shown in Section 3. Section 4 presents the simulation tool adapted and used for the experimentation environment shown in Section 5.

Finally, results are shown and analyzed in Section 6, where we compare energy-saving outcomes and the performance impact for each energy-efficiency policy. Conclusions are drawn in Section 7.

2. Related work

Many efforts have been made in order to increase resource and energy efficiency in data centers. The proposed strategies range from energy-aware scheduling algorithms to power-off heuristics

[*] Corresponding author.
E-mail addresses: damiancerero@us.es (D. Fernández-Cerero), afdez@us.es (A. Fernández-Montes), jortega@us.es (J.A. Ortega).

https://doi.org/10.1016/j.eswa.2018.06.007
0957-4174/© 2018 Elsevier Ltd. All rights reserved.

Table 1
Related work summary.

Ref.	Title: Performance evaluation of a green scheduling algorithm for energy savings in cloud computing	Savings
Duy et al. (2010)		~45%
	Strategy: Power off policy based on a neural network predictor	
	Evaluation: [8–512] nodes cluster simulation	
	Workload: End user homogeneous requests that follow a day/night pattern	
Ref.	Title: Energy efficient utilization of resources in cloud computing systems	Savings
Lee and Zomaya (2012)		[5–30]%
	Strategy: Energy-aware task consolidation heuristic based on different cost functions	
	Evaluation: Simulation of a not stated size cluster	
	Workload: Synthetic workload in terms of number of tasks, inter arrival time and resource usage	
Ref.	Title: Dynamic energy-aware scheduling for parallel task-based application in cloud computing	Savings
Juarez et al. (2018)		[20–30]%
	Strategy: Polynomial-time and multi-objective scheduling algorithm for DAG jobs	
	Evaluation: Experimentation on a 64 nodes cluster	
	Workload: Synthetic directed acyclic graph-based workload	
Ref.	Title: Energy efficient resource management in virtualized cloud data centers	Savings
Beloglazov and Buyya (2010)		~80%
	Strategy: VM allocation and migration policies + Always off policy	
	Evaluation: 100 nodes cluster simulation using CloudSim	
	Workload: Synthetic workload that simulates services that fulfill the capacity of the cluster	
Ref.	Title: Saving energy in data center infrastructures	Savings
Ricciardi et al. (2011)		[20–70]%
	Strategy: Safety margin power-off policy	
	Evaluation: 100 and 5000 nodes cluster simulation	
	Workload: Synthetic workload that follows a day/night pattern	

Table 2
Summary of the pros and cons of the energy-aware scheduling algorithms in the related work.

Duy et al. (2010) Performance evaluation of a green scheduling algorithm for energy savings in cloud computing	
Pros	Deeply described neural-network-based algorithm; Empirically measured power consumption
Cons	No focus on overall performance, only in drop rate; Small data-center size ([8–512] nodes) Short simulation period (2 days); No evaluation of huge & heterogeneous workload (cloud computing)
Fernández-Cerero et al. (2018) Security supportive energy aware scheduling and scaling for cloud environments	
Pros	Load balancing and VM scaling techniques; Computes security constraints Proposal of an energy-aware Genetic Algorithm
Cons	Focused on DVFS, not on shutting-down machines; Only for Independent Batch Scheduling environment No evaluation of huge & heterogeneous workload (real-life cloud computing system); Tiny cluster (5 VMs)
Juarez et al. (2018) Dynamic energy-aware scheduling for parallel task-based application in cloud computing	
Pros	DAG and data-aware workload; Multi-heuristic scheduling algorithm
Cons	Small data-center size (64 nodes max.); Only evaluates the makespan and total energy consumed No evaluation of huge & heterogeneous workload (real-life cloud computing system) Not focused on shutting-down machines, but in various DAG workloads Not clear about the cluster utilization (and the theoretical maximal energy efficiency)
Lee and Zomaya (2012) Energy efficient utilization of resources in cloud computing systems	
Pros	Large and detailed experimentation; Allows task migration
Cons	Focused on task scheduling, not on the shut-down of machines. No evaluation of huge & heterogeneous workload (real-life cloud computing system) No evaluation of the performance impact of the proposed strategies

that aim to minimize the number of idle nodes. A summary of these efforts is presented in Table 1, and a summary of the pros and cons of the related work regarding energy-aware scheduling algorithms, VM scaling and migration, and proposals based on shutting-down idle nodes is presented in Tables 2–4, respectively.

A substantial part of these approaches has been directed towards energy-aware scheduling strategies that could lead to powering off idle nodes, such as Duy, Sato, and Inoguchi (2010), Fernández-Cerero, Jakóbik, Grzonka, Kołodziej, Fernández-Montes (2018), Juarez, Ejarque, and Badia (2018), and Lee and Zomaya (2012). In Duy et al. (2010), a Green Scheduling Algorithm based on neural networks is proposed. This algorithm predicts workload demand in order to apply only one power-off policy to idle servers. These experiments simulate a small data center (512 nodes as a maximum) which serves an homogeneous workload composed of end-user facing tasks which follow a day/night pattern. Lee and Zomaya (2012) present two energy-aware task consolidation heuristics. These strategies aim to maximize resource utilization in order to minimize the wasted energy used by idle resources. To this end, these algorithms compute the total cpu time consumed by the tasks and prevent a task being executed alone. Juarez et al. (2018) propose an algorithm that minimizes a multi-objective function which takes into account the energy-consumption and execution time by combining a set of heuristic rules and a resource allocation technique. This algorithm is evaluated by simulating DAG-based workloads, and energy-savings in the range of [20–30%] are shown. Fernández-Cerero et al. (2018) propose energy-aware scheduling policies and methods based on Dynamic Voltage and Frequency Scaling (DVFS) for scaling the virtual resources while performing security-aware scheduling decisions.

In addition, different techniques of energy conservation such as VM consolidation and migration (Beloglazov, Abawajy, & Buyya, 2012; Beloglazov & Buyya, 2010, 2012; Sohrabi, Tang, Moser, & Aleti, 2016) are also proposed. Beloglazov and Buyya (2010) describe a resource management system for virtualized cloud data centers that aims to lower the energy consumption by applying a set of VM allocation and migration policies in terms of current CPU usage. This work is extended by focusing on SLAs restrictions in Beloglazov et al. (2012) and by developing and compar-

Table 3
Summary of the pros and cons of the VM scaling and migration algorithms in the related work.

Beloglazov and Buyya (2010) Energy efficient resource management in virtualized cloud data centers & Beloglazov et al. (2012) Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing	
Pros	VM resizing and migration; Thermal and network considerations.
Cons	Not focused on shutting down machines, but on VM placement; Small data-center size (100 nodes) No evaluation of huge & heterogeneous workload (real-life cloud computing system) No evaluation of the performance impact of the proposed strategies (only SLA violations)
Sohrabi et al. (2016) Adaptive virtual machine migration mechanism for energy efficiency	
Pros	Machine learning for re-scheduling tasks when hosts become overloaded; Real-life workload
Cons	Not focused on shutting down machines, but in VM placement;Not large data-center size (800 machines) No detailed evaluation of the performance impact (only SLA violations & makespan)
Beloglazov and Buyya (2012) Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers	
Pros	Dynamic VM resizing and migration; Dynamic host overloading algorithms; Real-life workload; Extensive experimentation
Cons	Not focused on shutting down machines, but in VM placement; Not large data-center size (800 machines) No detailed evaluation of the performance impact (only SLA violations)

Table 4
Summary of the pros and cons of the proposals based on shutting-down idle nodes in the related work.

Ricciardi et al. (2011) Saving energy in data center infrastructures	
Pros	Day-night workload pattern; Two different-sized data centers (5000 and 100 nodes)
Cons	Only one energy-efficiency policy based on a security margin No performance impact evaluation; No description of workload and simulation tool
Amur et al. (2010) Robust and flexible power-proportional storage	
Pros	Near optimal power proportionality; Various data-layout policies Almost no negative impact in data loss; Good experimental analysis based on standard benchmarks.
Cons	Focused only on cluster storage; Small data center (25 nodes); Read-only workload
Kaushik and Bhandarkar (2010) Greenhdfs: towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster	
Pros	Cold and hot data areas; Real-life HDFS traces workload; Large Yahoo! data center (2600 nodes)
Cons	Focused only on cluster storage; Few details on the simulation tool and performance impact
Luo et al. (2013) Superset: a non-uniform replica placement strategy towards high-performance and cost-effective distributed storage service	
Pros	The dynamic replication may improve both energy efficiency and performance Extensive experimentation with a comparative with Thereska et al. (2011)
Cons	Focused only on cluster storage; Few details of the simulation tool; Small data center (240 nodes)
Thereska et al. (2011) Sierra: practical power-proportionality for data center storage	
Pros	Real-life workload presenting a day/night pattern; No extra capacity nor migration required Read & write workload; Network-aware; Extensive experimentation
Cons	Focused only on cluster storage; Small data center (31 nodes)

ing various adaptive heuristics for dynamic consolidation of VMs in terms of resource usage in Beloglazov and Buyya (2012). These migration policies are evaluated by simulating a 100-node cluster. Energy reductions up to approximately 80% are shown with low impact on quality of service and SLAs. In Sohrabi et al. (2016), a Bayesian Belief Network-based algorithm that aims to allocate and migrate VMs is presented. This algorithm uses the data gathered during the execution of the tasks in addition to the information provided at submission time in order to decide which of the virtual machines are to be migrated when a node is overloaded. In Ricciardi et al. (2011), a different approach is proposed. In this work, Ricciardi et al. present a data center energy manager that relies on day/night workload patterns in order to aggregate traffic during night periods and therefore turn off idle nodes. The authors apply a power-off policy based on a safety margin in order to minimize the negative impact on performance. To evaluate this strategy, two different data centers of 5000 and 100 nodes are simulated. In this kind of scenario, potential energy reductions between approximately 20 and 70% are shown.

The application of these techniques together presents a major opportunity in various large-scale scenarios, such as Grid 5000 (De Assuncao, Gelas, Lefevre, & Orgerie, 2012).

In order to achieve energy proportionality, many efforts (Amur et al., 2010; Kaushik & Bhandarkar, 2010; Luo, Wang, Zhang, & Wang, 2013; Thereska, Donnelly, & Narayanan, 2011) have been made in only one subset of all the systems, since these represented the main bottleneck when they were written. In Amur et al. (2010), a power-proportional distributed file system that stores replicas of data on non-overlapping subsets of nodes is proposed. These subsets of different sizes contain one replica for each file. This partitioning strategy lets the administrator decide the number of datasets to be kept turned on to serve incoming requests, and therefore it gives the administrator the opportunity to control the trade-off between energy consumption and performance. Kaushik and Bhandarkar (2010) present a variant of Hadoop Distributed File System that divides the cluster in two zones in terms of data usage pattern. The first zone, called the *Hot Zone*, contains the subset of fresh data that is more likely to be accessed short term. The second zone, called the *Cold Zone*, contains the set of files with low spatial or temporal popularity with few to rare accesses. Once the cluster is divided in these two zones, an aggressive power-off policy is applied to the *Cold Zone*. This energy-efficiency strategy achieves approximately 26% energy reduction without notably worsening the overall performance and reliability in a three-month simulation based on a Yahoo! cluster configuration. In Thereska et al. (2011), the cluster is partitioned in order to create different non-overlapping data zones. Each of these zones contains one replica of the cluster data. Once the cluster is partitioned, the system lets the administrator power off the desired number of zones, depending on the aggressiveness of the energy-efficiency strategy. Luo et al. (2013) propose a non-uniform replica placement strategy in terms of data popularity. This strategy aims to increase the number of available parallel replicas for data that is very likely to be accessed, and to lower the number of replicas of the low-used data that is rarely accessed in order to power off the maximum number of nodes without affecting the overall performance. In order to evaluate this strategy, a Zipf distribution-based workload and a real trace of Youku is executed in a 240-nodes simulated cluster.

This paper follows a different approach: to deeply describe the impact of 6 different power-off policies in terms of performance and energy consumption on a well-defined, rich and realistic heterogeneous workload that follows the trends present in Google Traces by running a huge amount of experiments for centralized monolithic scheduling frameworks. In order to better characterize the impact of these power-off policies and unlike the presented related work, this paper does not focus on developing energy-aware VM allocation or migration policies, but the authors use a Best-fit-like VM allocation heuristic and does not apply VM migration strategies as stated in Section 5. In addition, these power-off policies are applied at the data center operating system / resource manager level, not to a framework or subsystem like some of the related work presented. This difference makes it possible to apply the proposed power-off policies to any framework that can run as a VM / Linux container on the data center.

3. Power-off policies

In this work, we have developed several deterministic and probabilistic power-off decision policies. These power-off decision policies form the core of the work since they have much more impact on data-center efficiency and performance than anything else.

From among the *deterministic* policies, the following policies have been developed:

- *Never power off*: This power-off decision policy disables the power-off process, and therefore represents the current scenario.
- *Always power off*: This power-off decision policy will shut down every machine after freeing all the resources under use, whenever possible.
- *Maximum load*: This power-off decision policy takes into account the maximum resource pressure of the data-center load and compares it to a given threshold μ. If the current load is less than this given threshold μ, then the machine will be powered off.
- *Minimum free-capacity margin*: This power-off decision policy assures that at least a given percentage of resources μ is turned on, free, and available in order to respond to peak loads.

Regarding among the *probabilistic* policies, the following policies have been implemented:

- *Random*: This policy switches off and randomly leaves the resources idle by following a Bernoulli distribution whose parameter is equal to 0.5. This policy is useful to ascertain the accuracy of the predictions made by the following probabilistic policies.
- *Exponential*: The Exponential distribution, denoted by $Exp(\lambda)$, describes the time between events in a Poisson process, that is, a process in which events occur continuously and independently at a constant average rate $(1/\lambda)$. Under the hypothesis that the arrival of new jobs follows an Exponential distribution, this energy policy attempts to predict the arrival of new jobs that can harm the data-center performance due to the lack of sufficient resources for their execution.
  To compute the λ parameter, the most recent jobs are taken into account. The size of these last jobs is denoted as *Window size*. Thus, every time a shut-down process is executed, the mean time between these last jobs that could not be served at the time of making the decision is computed, and denoted by δ. Hence, $\lambda = 1/\delta$ by using the method of maximum likelihood. The probability of the arrival of a new job can then be computed by means of the exponential cumulative density function (cdf), as $cdf(T_s)$[1]$= 1 - e^{-T_s/\delta}$. Therefore, given a decision threshold μ value, the following conditions are imposed:

$$\begin{cases} \text{if } cdf(T_s) >= \mu & \text{then leave resources } Idle \\ \text{if } cdf(T_s) < \mu & \text{then switch resources } Off \end{cases}$$

- *Gamma*: The Gamma distribution, denoted by $\Gamma(\alpha, \beta)$, is frequently used as a probability model for waiting times and presents a more general model than the Exponential distribution. Under the hypothesis that the arrival of new jobs follows a Gamma distribution, this energy policy attempts to predict the arrival of the amount of new jobs required to oversubscribe the available resources.
  and takes into account the *Lost factor* described in the *Exponential* policy. are:
  - $mem_{available}$: memory in *Idle* state.
  - $cpu_{available}$: computational resources in *Idle* state.
  - mem_{mean}: mean RAM used by last jobs.
  - cpu_{mean}: mean computational resources used by last jobs.
  - δ: mean inter-arrival time of last jobs.
  - α_{cpu}: as $cpu_{available}/cpu_{mean}$.
  - α_{mem}: as $mem_{available}/mem_{mean}$.

  The parameters of the Gamma distribution are then estimated as: $\alpha = Min(\alpha_{cpu}, \alpha_{mem})$ and $\beta = \delta$. Finally the probability of the arrival of new jobs is computed by means of the cumulative density function (cdf) with:

$$cdf(T_s) = \frac{\gamma(\alpha, \beta x)}{\Gamma(\alpha)}$$

  Hence, given a decision threshold μ value, the following conditions are imposed:

$$\begin{cases} \text{if } cdf(T_s) >= \mu, & \text{then leave resources } Idle \\ \text{if } cdf(T_s) < \mu, & \text{then switch resources } Off \end{cases}$$

4. Simulation tool

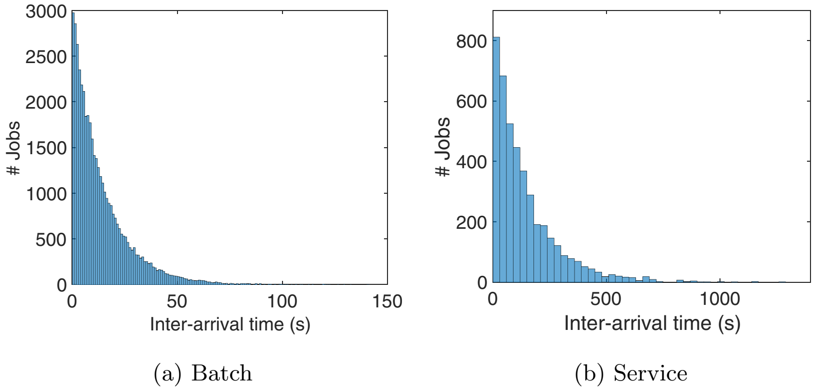
In this paper, we extended the Google lightweight simulator presented in Schwarzkopf, Konwinski, Abd-El-Malek, and Wilkes (2013) in order to perform energy-efficiency analysis. This simulator lets the authors focus on the development of energy-efficiency policies and perform simulations of the different scheduling frameworks and various data-center environments, while abstracting the details of each of them. The following energy states are considered : (a) *On*: 150 W (b) *Off*: 10 W (c) *Idle*: 70 W (d) *Shutting Down*: 160 W (e) *Powering On*: 160 W. The energy consumption is linearly computed in terms of the usage of each core.

Moreover, this tool provides us with a trustworthy implementation of the monolithic scheduling processes, and results have been contrasted to Google's realistic simulator (Schwarzkopf et al., 2013). The simulator employed can be found at https://github.com/DamianUS/cluster-scheduler-simulator.

5. Experimentation

In order to test and measure the achieved power savings and the consequent impact on data-center performance, a set of experiments have been run. Each of these experiments simulates a period of seven days of operation, and applies various combinations of the energy policies developed and described in Section 5.2. These experiments are designed to simulate realistic and heterogeneous environments.

[1] T_S is defined *as the minimum time that ensures energy saving if a resource is switched off between two jobs*(Orgerie, Lefèvre, & Gelas, 2008).



(a) Batch

(b) Service

Fig. 1. Workload inter-arrival histogram.

In order to create a realistic and trustworthy testbed, realistic Google traces (Reiss, Wilkes, & Hellerstein, 2011, 2012b) were chosen and the interpretations carried out over these traces by the research community (Abdul-Rahman & Aida, 2014; Di, Kondo, & Franck, 2013; Liu & Cho, 2012; Reiss, Tumanov, Ganger, Katz, & Kozuch, 2012a) were studied.

In the following subsections, the test suite and environment designed and used are presented.

5.1. Workload

Jobs are composed of one or more tasks: sometimes thousands of tasks. In this work, two types of jobs are considered:

- *Batch* jobs: This workload is composed of jobs which perform a computation and then finish. These jobs have a determined start and end. MapReduce jobs are an example of a *Batch* job.
- *Service* jobs: This workload is composed of long-running jobs which provide end-user operations and infrastructure services. As opposed to *Batch* jobs, these jobs have no determined end. Web servers or services, such as BigTable (Chang et al., 2008), are good examples of a *Service* job.

Synthetic workloads are generated in each experiment run by replicating the behaviour of those workloads present in typical Google data centers. Therefore, although the workload generated in each simulation run is unique, they follow the same model design.

The subsequent job attributes have been covered and studied:

- *Inter-arrival time*: The inter-arrival time represents the time between two consecutive *Service* jobs or two consecutive *Batch* jobs. It also determines the amount of jobs executed in a specific time window. The inter-arrival time between two *Batch* jobs is usually shorter than that between two *Service* jobs, as illustrated in Fig. 1, leading to a higher number of *Batch* jobs, as illustrated in Fig. 4.
- *Number of tasks*: This parameter represents the number of tasks that comprise a job. As illustrated in Fig. 2, *Batch* jobs are composed of a higher number of tasks than *Service* jobs.
- *Job duration*: This parameter represents the time that a job consumes resources in the data center. As illustrated in Fig. 3, *Batch* jobs require less time to complete than *Service* jobs.
- *Resource usage*: Taking into account the parameters described above, although *Batch* jobs and tasks constitute the vast majority, the higher resource utilization and duration of *Service* jobs results in our synthetic workload as illustrated in Fig. 4. In this figure, it can be noticed that less than 10% of jobs in the workload are *Service* jobs, while less than 3% of tasks are *Service* tasks. It should be borne in mind, however, that almost 40% of CPU and 50% of RAM resources are used by *Service* jobs.

Taking into account the aforementioned environment and workload scenario, the generated workload is composed of 43,050 *Batch* jobs, 4238 *Service* jobs. This represents one week of operation time, and reaches 57, 81% computational power and 48.33% memory in use on average.

5.2. Experiments performed

After simulating a wide range of values for every parameter described in Section 3, for comprehension purposes, the most interesting and representative have been chosen:

In order to prevent resource contention, a power-on policy which turns on the necessary machines whenever the workload resource demands are higher than available machines, and a scheduling strategy which tries to fill every machine to the maximum (90%) while maintaining some randomness (Khaneja, 2015) is used. It is worth mentioning that in the experiments that simulate the *Never power off* policy, a scheduling strategy where resources are chosen randomly is used to represent the base scenario.

6. Results

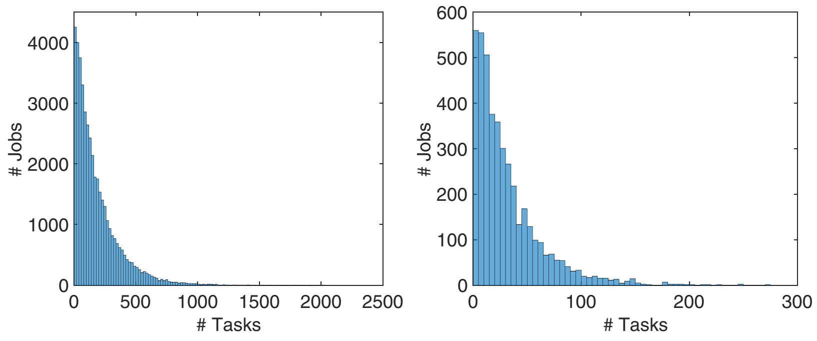
In this section, the obtained results are illustrated through key performance indicators concerning a) energy savings and b) impact over performance. In this way, energy savings and performance are analyzed and compared for each energy policy.

6.1. Energy savings indicators

The following indicators were selected in order to describe the energy savings and the behaviour of the powering on/off operations:

- *Energy consumed vs. current system*: The overall energy used in each experiment against the current[2] operation energy utilization.
- *Power-off operations*: The total number of shut-downs performed over all the resources during the overall simulated operation time.
- *KWh saved per shut-down*: This represents the energy saved against the shut-downs performed. It shows the *goodness* of the power-off actions performed.

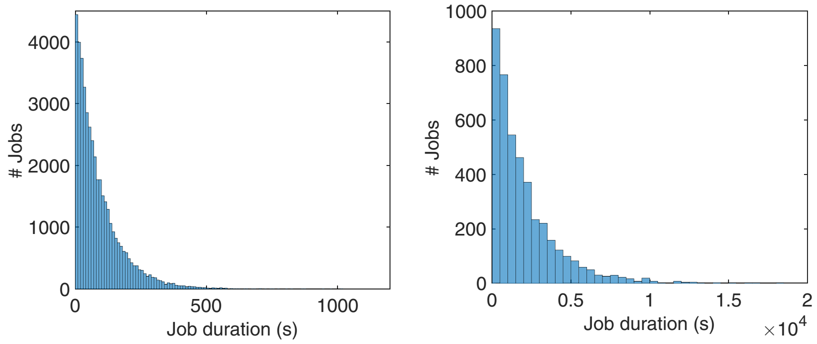
[2] Current operation for the same data center and workload, but without applying energy-saving polices.



(a) Batch

(b) Service

Fig. 2. Histogram of the number of tasks for workload.



(a) Batch

(b) Service

Fig. 3. Workload job-duration histogram.

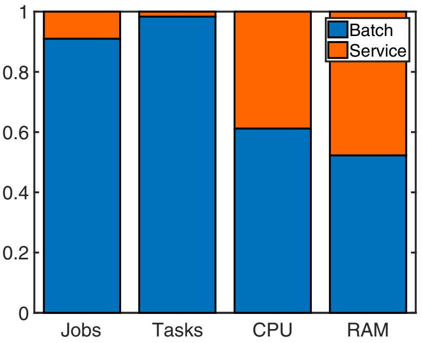


Fig. 4. Workload resource usage.

- *Idle resources*: Represents the percentage of resources in an idle state (turned on but not in use).

6.2. Performance indicators

The following indicators were selected as the most significant in the description of the impact of the various energy-efficiency policies on data-center performance.

- *Job queue time (first scheduled)*: Represents the time a job waits in the queue until its first task is scheduled.
- *Job queue time (fully scheduled)*: Represents the time a job waits in the queue until it is totally scheduled (not finished).

Table 5
Summary of energy savings for the best energy policies. N – *Never power off.* A – *Always power off.* R – *Random.* L – *Maximum load.* M – *Minimum free-capacity margin.* E – *Exponential.* G – *Gamma.*

Energy policy	Energy % vs. Current	Power offs (10^3)	KWh saved Shutt.	Idle resources %	KWh saved (10^3)	Cost savings ($)
N	100	0.00	n/a	42.21	0.00	0
A	80.25	64.52	1.72	8.35	110.83	15,517
R	80.73	39.16	2.76	9.18	108.15	15,141
L	**80.21**	67.20	1.65	**8.27**	**111.09**	**15,553**
M	82.35	9.04	10.96	11.97	99.04	13,866
E	82.34	9.01	**11.00**	11.95	99.12	13,877
G	82.7	**8.98**	10.82	12.56	97.12	13,597

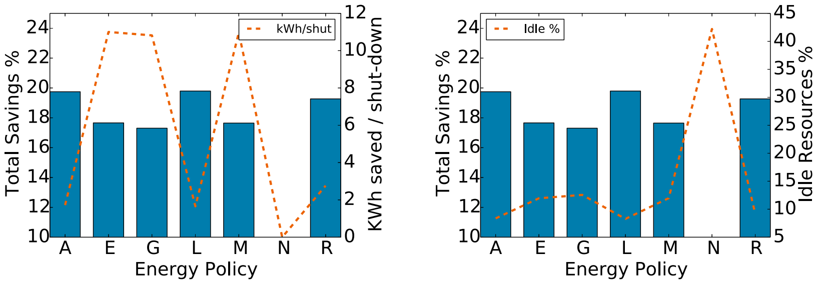
- *Job think time*: Represents the time needed for a schedule decision to be made.
- *Timed-out jobs*: A job is marked as timed out and left without scheduling when the scheduler completes 100 tries to schedule the same job, or 1000 tries of any task of the job. In all our experiments, the number of timed-out jobs is always 0.
- *Scheduler occupation fraction*: This represents the scheduler usage.

6.3. General results

In order to analyze and compare the performance of each family of policies, the best and exemplary energy policy from each family has been selected, in terms of the combination of energy-

Table 6
Summary of performance impact of best energy policies. B – *Batch* workload, S – *Service* workload.

Energy	Time first scheduled (s)				Time fully scheduled (s)				Sched.
policy	Mean		90p.		Mean		90p.		occu-
	B	S	B	S	B	S	B	S	pation(%)
N	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
A	0.22	0.22	0.78	0.84	0.30	0.32	0.92	0.95	15.18
R	0.20	0.21	0.72	0.80	0.25	0.27	0.80	0.84	14.79
L	0.22	0.22	0.78	0.83	0.31	0.33	0.94	0.95	15.20
M	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
E	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
G	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30



(a) Savings vs. KWh/shut-down (b) Savings vs. Idle resources

Fig. 5. Energy-saving comparison. A – *Always power off.* E – *Exponential.* G – *Gamma.* L – *Maximum load.* M – *Minimum free-capacity margin.* N – *Never power off.* R – *Random.*

saving and performance results. Table 5 shows performance key indicators, while Table 6 shows energy related results. Fig. 5a and 5 b summarize and illustrate these numeric results.

From these results, several conclusions can be stated. In general, the more shut-downs there are, the more energy is saved, or from another point of view, the less idle the resources, the less energy wasted. Fig. 5b shows this behaviour, since the *Always power off* energy policy and other policies that tend to switch off resources are always the greatest energy savers, achieving savings of approximately 20%. This first conclusion provides evidence previously shown by Fernández-Montes, Gonzalez-Abril, Ortega, and Lefèvre (2012) in similar environments.

However, it should be borne in mind that the accuracy of the employed policies depends on the distribution of the data-center workload. The application of these policies without any previous knowledge of the workload and its distribution may be hard and might achieve sub-optimal results.

Fig. 5a shows that *Exponential, Gamma* and *Minimum free-capacity margin* policies perform fewer shut-down operations, but in a highly planned manner, and therefore the quantity of energy saved per shut-down operation is approximately 6 times better (from 2 to 12 kWh), and total savings are approximately 18%, which is only 2% less than the policies of the highest energy savings, while performing 85% less shut-down operations compared to those performed by *Always power off* and *Maximum load* policies.

In terms of costs, the saved energy adds up to a total of $15 K for 7 days, and hence, under similar conditions, this would indicate $60 K a month or $720 k a year.[3]

In terms of performance, Fig. 6a and 6 b show that the more shut-downs are performed, the more probability of causing a negative impact on the performance. This is noticeable for the *Always power off* and *Maximum load* policies. The negative impact in terms of queue time is shown on the queue-time parameters, such as *Job queue time (first scheduled)* and *Job queue time (fully scheduled)* parameters, which suffer a mean impact of 15% and 60%, respectively compared to those of the base/current scenario. The *Random* policy acts as an intermediate stage between the two previously stated sides. The queue-time parameters, such as *Job queue time (first scheduled)* and *Job queue time (fully scheduled)*, suffer a mean impact of 5% and 15%, respectively.

On the other hand, once again, *Exponential, Gamma* and *Margin* energy policies do not affect negatively to the performance, but achieve major energy savings (∼ 18%).

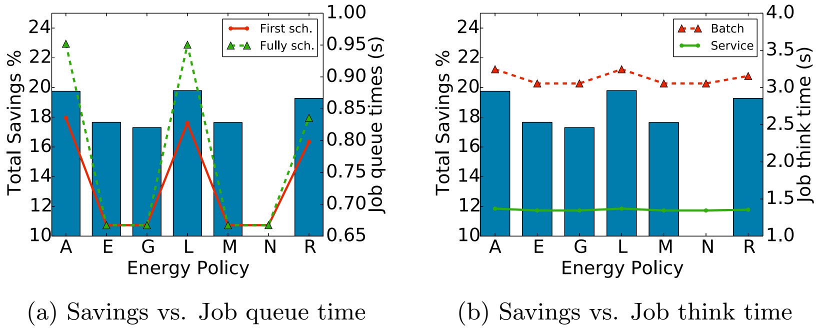
In order to better understand the behaviour of these energy policies, Fig. 7 shows the evolution of the resource state for each policy.

It should be borne in mind that there is a short-time period at the beginning until each policy reaches its normal pattern. This adjusting period occurs due to the *On* state of all the resources of the data center at the beginning of the simulation. Two groups of policies can be determined according to their behaviour. On one hand, the *Always power off, Maximum load* and *Random* policies suffer from the same problem: they try to adjust available resources to fit, as much as possible, the current workload demand, which leads to a high number of power on/off operations. Moreover, it can be observed that the time needed by the *Random* policy to adjust to workload changes is double that of the *Always power off* policy, since the *Random* policy performs half the number of shut-down operations compared to the *Always power off* policy.

On the other hand, prediction-based policies perform much smoother adjustments to the workload, therefore leading to a lower number of power on/off operations.

Finally, at the end of day #1, there is a peak of machines that are switched on for *Always power off*-like policies. Hence, it should be pointed out that maintaining a set of machines as a security margin can lead to the ability to satisfy the workload needs in a much more gradual way. Moreover, these workload peaks do not affect these prediction-based energy policies. Aggressive policies

[3] $0.14 per KwH was considered to compute economic costs.



(a) Savings vs. Job queue time (b) Savings vs. Job think time

Fig. 6. Comparison of energy savings vs. performance.

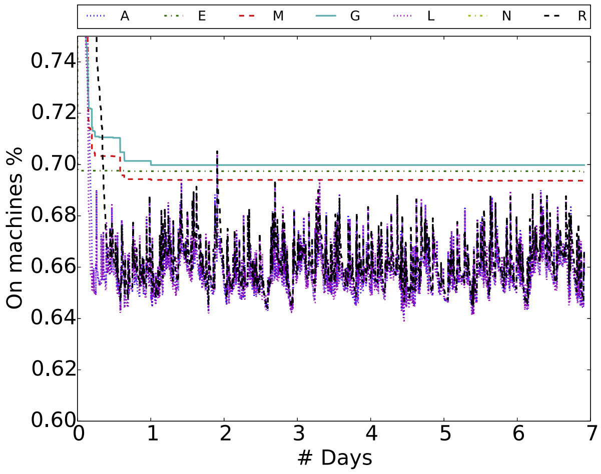


Fig. 7. Behaviour of energy policies.

solve these load bursts by switching on a large set of machines, even larger than actually needed for that moment.

The presented evidences lead us to recognize that controlled and prediction-based polices are preferable to deterministic policies.

6.4. Exponential policy: detailed results

Exponential power-off policies described in Section 3 show a high dependency on the *Lost factor* parameter. This parameter represents the percentage of resources that can not be used even if they are available because these resources are insufficient to hold the task. For example, lets consider a workload where all tasks will consume 1 GB of memory and 2 CPU cores. In this scenario, even if a machine has 900 MB of memory and 1.8 CPU cores available, these resources will be completely useless and should not be computed as available resources. The *Lost factor* allows the authors to take into account the useful available resources instead the total not-used resources.

In order to fully understand the results presented in this section, it should be borne in mind the nature of the workload employed: a vast majority are low-resource consuming jobs comprising very few tasks which are easily to serve. Due to this, the risk of not satisfying the requirements of these tasks is very low, tending to 0. In the other hand, very few jobs are composed of an enormous number of tasks, where it is almost impossible to serve their requirements. This means that the risk of not satisfying the requirements of these tasks tends towards 1. Due to this, the decision threshold μ has a lower impact in terms of performance and energy savings, unless a value extremely close to 0 or 1 is taken, whereby it behaves as the *Never power off* or *Always power off* policies, respectively.

In addition, the number of these high-demanding jobs is very low. This leads to a poor prediction when only a low number of the last jobs are taken into account. Thus, the *Window size* values evaluated are of less impact in terms of performance and energy savings than the *Lost factor* parameter.

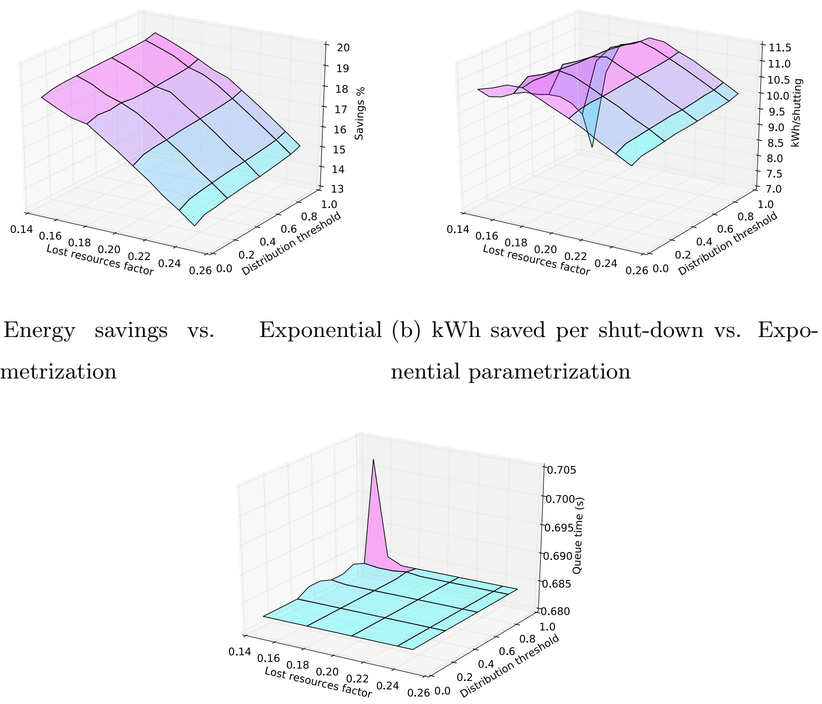
Fig. 8 presents the dependency on the *Lost factor* clearly. In terms of kWh saved per shut-down, as shown in Fig. 8b, the best results are reached when a *Lost factor* of 20% is considered. This value makes sense, because as stated in Section 5, our environment is designed to simulate the one presented in Lo, Cheng, Govindaraju, Ranganathan, and Kozyrakis (2015), which attains a level of utilization of 90% of resources without causing any noticeable negative impact.

- *Energy savings*: In terms of energy savings, as presented in Table 7 and in Fig. 8a, for low *Lost factor* values, the *Exponential* policy behaves similar to the *Always power off* policy, and achieves the highest rates of energy savings at the expense of a negative performance impact, as presented in Table 8. The higher this parameter increases, the lower the number of power-off cycles, and approaches the *Never power off* policy.
- *Performance*: In terms of performance, as presented in Table 8, the *Exponential* policy follows the same trend present in the energy savings. However, it can be observed that if 20% of resources are taken as unusable (*lost factor*), as suggested by the *kWh saved per shut-down* parameter, then a virtually non-negative impact in terms of performance is imposed. Moreover only ∼2% more of energy is consumed compared to *Always power off* policy, but only ∼15% of the number of shut-downs is performed. In addition this is consistent with the *Minimum free-capacity margin* policy. Finally, if the *Lost factor* value continues to rise above ∼20%, it does not impact positively in terms of performance, but negatively in terms of energy savings.

6.5. Gamma policy: detailed results

As described for the *Exponential* policy, the exponential nature of the generated workload links the *Gamma* policy performance impact and energy savings to the *Lost factor* parameter, whereby the rest of the parameters, *Window size* and decision threshold μ, hold a minor influence.

In terms of behaviour, the *Gamma* policy follows the same trends present in the *Exponential* policy. However, due to the difference in the predictive model construction, the *Gamma* policy be-



(a) Energy savings vs. Exponential parametrization

(b) kWh saved per shut-down vs. Exponential parametrization

(c) Queue time vs. Exponential parametrization

Fig. 8. Energy savings and performance indicators in Exponential parametrization.

Table 7
Energy savings for *Exponential* policies. Exponential parameterization: [*Decision threshold* μ, *Window size, Lost factor*].

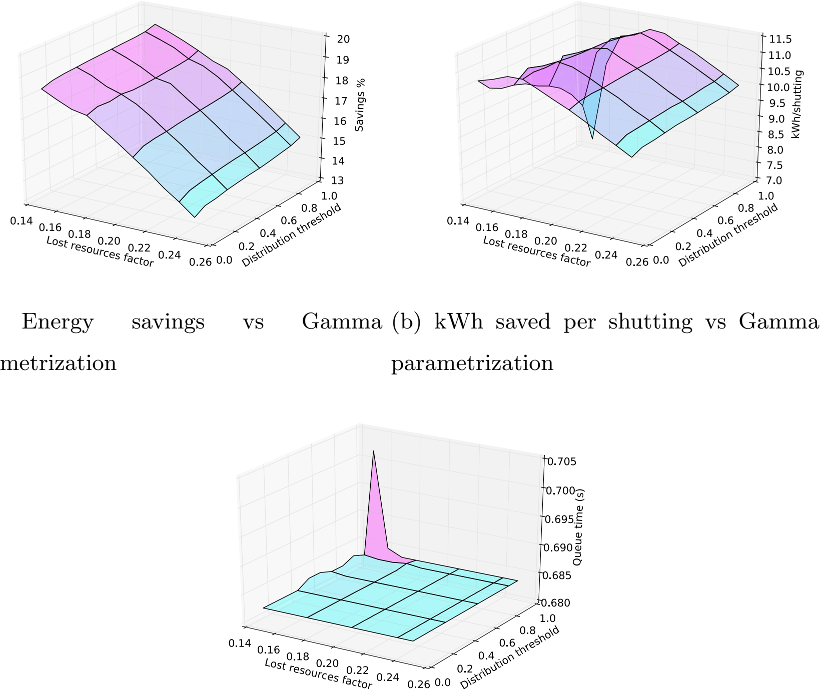
Energy policy		Energy	Power	KWh	Idle	KWh	Cost
Acr.	Params	% vs. Current	offs (10^3)	saved Shutt.	resources %	saved (10^3)	savings ($)
N	n/a	100	n/a	n/a	42.21	n/a	0
A	n/a	80.25	64.52	1.72	8.35	110.83	15,517
R	[0.50]	80.73	39.16	2.76	9.18	108.15	15,141
E	[0.30, 25, 0.10]	80.01	64.94	1.73	7.93	112.21	15,710
E	[0.30, 25, 0.15]	80.68	19.00	5.71	9.11	108.42	15,179
E	**[0.30, 25, 0.20]**	82.34	9.01	11.00	11.95	99.12	13,877
E	[0.30, 25, 0.25]	85.06	8.54	9.82	16.62	83.83	11,736
E	[0.30, 25, 0.30]	88.34	8.03	8.16	22.22	65.47	9166

Table 8
Performance results for the *Exponential* energy-efficiency policy.

Energy policy	Time first scheduled (s)				Time fully scheduled (s)				Sched.
	Mean		90p.		Mean		90p.		occu-
	B	S	B	S	B	S	B	S	pation(%)
N	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
A	0.22	0.22	0.78	0.84	0.30	0.32	0.92	0.95	15.18
R [0.50]	0.20	0.21	0.72	0.80	0.25	0.27	0.80	0.84	14.79
E [0.30, 25, 0.10]	0.22	0.22	0.77	0.84	0.30	0.32	0.92	0.95	15.15
E [0.30, 25, 0.15]	0.19	0.20	0.65	0.69	0.20	0.21	0.68	0.69	14.45
E [0.30, 25, 0.20]	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
E [0.30, 25, 0.25]	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
E [0.30, 25, 0.30]	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30

Table 9
Energy-saving results for Gamma energy policy. Gamma parameterization: [*Decision threshold* μ*, Window size, Lost factor*].

Energy policy		Energy	Power	KWh	Idle	KWh	Cost
		%vs.	offs	saved	resources	saved	savings
Acr.	Params	Current	(10^3)	Shutt.	%	(10^3)	($)
N	n/a	100	n/a	n/a	42.21	0.00	0
A	n/a	80.25	64.52	1.72	8.35	110.83	15,517
R	[0.50]	80.73	39.16	2.76	9.18	108.15	15,141
G	[0.90, 25, 0.10]	80.28	60.21	1.84	8.40	110.67	15,494
G	[0.90, 25, 0.15]	81.03	15.08	7.06	9.71	106.45	14,902
G	**[0.90, 25, 0.20]**	82.7	8.98	10.82	12.56	97.12	13,597
G	[0.90, 25, 0.25]	85.36	8.49	9.67	17.13	82.14	11,500
G	[0.90, 25, 0.30]	88.44	7.99	8.12	22.40	64.89	9084



(a) Energy savings vs Gamma parametrization

(b) kWh saved per shutting vs Gamma parametrization

(c) Queue time vs Gamma parametrization

Fig. 9. Energy savings and performance indicators in Gamma parametrization.

haves slightly less aggressively in terms of number of shut-downs applied.

- *Energy savings*: In terms of energy savings, as presented in Table 9 and in Fig. 9a and stated in the *Exponential* policy, if the *Lost factor* is too low, then the *Gamma* policy behaves like the *Always power off* policy, in that it achieves the highest rates of energy savings at the expense of a negative performance impact, as presented in Table 10. The higher this parameter increases, the lower the number of power-off cycles, and approaches the *Never power off* policy.
- *Performance*: In terms of performance, as presented in Table 10, the *Gamma* policy follows the same trend present in the energy savings. However, it can be observed that if 20% of resources are taken as unusable (*Lost factor*), as suggested by the *kWh saved per shut-down* parameter, then a virtually non-negative impact in terms of performance is imposed. Moreover only ~2.5% more of energy is consumed compared to *Always power off* policy, but only ~13% of the number of shut-downs is performed. In addition this is consistent with the *Minimum free-capacity margin* and *Exponential* policies. Finally, if the *Lost factor* value continues to rise above ~20%, then it does not impact positively in terms of performance, but negatively in terms of energy savings.

Table 10
Performance results for the Gamma energy policy.

Energy policy	Time first scheduled (s)				Time fully scheduled (s)				Sched.
	Mean		90p.		Mean		90p.		occu-
	B	S	B	S	B	S	B	S	pation(%)
N	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
A	0.22	0.22	0.78	0.84	0.30	0.32	0.92	0.95	15.18
R [0.50]	0.20	0.21	0.72	0.80	0.25	0.27	0.80	0.84	14.79
G [0.90, 25, 0.10]	0.21	0.22	0.77	0.83	0.29	0.31	0.89	0.94	15.09
G [0.90, 25, 0.15]	0.19	0.20	0.65	0.68	0.20	0.20	0.66	0.69	14.40
G [0.90, 25, 0.20]	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
G [0.90, 25, 0.25]	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30
G [0.90, 25, 0.30]	0.19	0.19	0.63	0.67	0.19	0.19	0.63	0.67	14.30

7. Conclusions

We have empirically proven that a suitable policy in data centers can save a considerable amount of energy and reduce the pollution of CO_2 in the atmosphere. Industrial partners willing to deploy this kind of energy-saving policies would have a direct positive impact on their competitiveness: in addition to become greener by minimizing the environmental impact, these policies may notably reduce their operation costs.

Several energy-saving policies have been explained, and their advantages and disadvantages have been presented, which outline which policy is more suitable for each data-center operational environment and administrator criteria. The behaviours of these energy policies are also consistent for various scheduler strategies.

This work characterizes the impact of these power-off policies. Unlike the presented related work, it is focused on the use of a Best-fit-like VM allocation heuristic. In addition, these power-off policies are applied at the data center operating system/resource manager level, not to a framework or to a subsystem. This approach makes it possible to apply the proposed power-off policies to any framework that can run as a VM/Linux container on the data center.

In this work, we go beyond the presented state of the art by focusing on the development of realistic, empirically-driven and production-ready energy policies that have a minor impact on data-center performance. These policies are simulated on a realistic environment that has been contrasted with real-life production systems, such as those of Google data centers. We can point out the following strengths in our research method: (a) A clear description of data-center utilization and workload distribution, which follow the industry trends; (b) A detailed explanation on the workload parameters, classification, generation and heterogeneity; (c) A complete description of the scheduling model and algorithms employed; and (d) A detailed explanation on the impact on both the main goals of our system: energy-efficiency and performance. On the other hand, the greatest weaknesses of this work include: (a) The lack of means to contrast the provided results with a real-life system; and (b) The lack of some real-life system aspects in simulation, such as task inter-dependency, networking and data-related considerations. However, we plan to overcome these limitations in future steps of this research.

The authors consider that prediction-based policies present much better behaviour for the data center, since they perform a much lower number of power-off cycles and save considerable amounts of energy. Moreover, it is also shown that it is possible to save energy by switching off machines and maintaining QoS and SLA levels, even for data centers in great demand.

For future work, the authors aim to focus on the following research directions:

1. Development of energy-efficiency policies based on machine learning, especially deep learning techniques.
2. Utilization of no-monolithic scheduling frameworks, such as two-level, shared state, distributed and hybrid schedulers.
3. Development of an intelligent system that may dynamically change the scheduling framework depending on environmental and workload-related parameters, as well as the study of the impact of such a system in terms of energy efficiency and performance.
4. Development of new simulation features, such as new workload patterns, task inter-dependency, networking and data-related considerations.

Acknowledgments

This research is supported by the VPPI - University of Sevilla.

References

Abdul-Rahman, O. A., & Aida, K. (2014). Towards understanding the usage behavior of Google cloud users: the mice and elephants phenomenon. In *Proceedings of the IEEE international conference on cloud computing technology and science (Cloudcom), Singapore* (pp. 272–277).

Amur, H., Cipar, J., Gupta, V., Ganger, G. R., Kozuch, M. A., & Schwan, K. (2010). Robust and flexible power-proportional storage. In *Proceedings of the first ACM symposium on cloud computing* (pp. 217–228). ACM.

Andersen, D. G., & Swanson, S. (2010). Rethinking flash in the data center. *IEEE Micro, 30*(4), 52–54.

Beloglazov, A., Abawajy, J., & Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Generation Computer Systems, 28*(5), 755–768.

Beloglazov, A., & Buyya, R. (2010). Energy efficient resource management in virtualized cloud data centers. In *Proceedings of the tenth IEEE/ACM international conference on cluster, cloud and grid computing* (pp. 826–831). IEEE Computer Society.

Beloglazov, A., & Buyya, R. (2012). Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience, 24*(13), 1397–1420.

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., et al. (2008). Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS), 26*(2), 4.

De Assuncao, M. D., Gelas, J.-P., Lefevre, L., & Orgerie, A.-C. (2012). The green gridâ;;5000: Instrumenting and using a grid with energy sensors. In *Remote instrumentation for Escience and related aspects* (pp. 25–42). Springer.

Di, S., Kondo, D., & Franck, C. (2013). Characterizing cloud applications on a Google data center. In *Proceedings of the forty-second international conference on parallel processing (ICPP). Lyon, France*.

Duy, T. V. T., Sato, Y., & Inoguchi, Y. (2010). Performance evaluation of a green scheduling algorithm for energy savings in cloud computing. In *Proceedings of the IEEE international symposium on parallel and distributed processing, workshops and Ph.D. forum (IPDPSW)* (pp. 1–8). IEEE.

El-Sayed, N., Stefanovici, I. A., Amvrosiadis, G., Hwang, A. A., & Schroeder, B. (2012). Temperature management in data centers: Why some (might) like it hot. *ACM SIGMETRICS Performance Evaluation Review, 40*(1), 163–174.

Fan, X., Weber, W.-D., & Barroso, L. A. (2007). Power provisioning for a warehouse–sized computer. In *ACM sigarch computer architecture news: 35* (pp. 13–23). ACM.

Femal, M. E., & Freeh, V. W. (2005). Boosting data center performance through non-uniform power allocation. In *Proceedings of the second international conference on autonomic computing (ICAC'05)* (pp. 250–261). IEEE.

Fernández-Montes, A., Fernández-Cerero, D., González-Abril, L., Álvarez-García, J. A., & Ortega, J. A. (2015). Energy wasting at internet data centers due to fear. *Pattern Recognition Letters, 67*, 59–65.

Fernández-Montes, A., Gonzalez-Abril, L., Ortega, J. A., & Lefèvre, L. (2012). Smart scheduling for saving energy in grid computing. *Expert Systems with Applications, 39*(10), 9443–9450.

Fernández-Cerero, D., Jakóbik, A., Grzonka, D., Kołodziej, J., & Fernández-Montes, A. (2018). Security supportive energy-aware scheduling and energy policies for cloud environments. *Journal of Parallel and Distributed Computing, 119*, 191–202. doi:10.1016/j.jpdc.2018.04.015.

Jakóbik, A., Grzonka, D., Kolodziej, J., Chis, A. E., & González-Vélez, H. (2017). Energy efficient scheduling methods for computational grids and clouds. *Journal of Telecommunications and Information Technology, 1*, 56.

Juarez, F., Ejarque, J., & Badia, R. M. (2018). Dynamic energy-aware scheduling for parallel task-based application in cloud computing. *Future Generation Computer Systems, 78*, 257–271. https://doi.org/10.1016/j.future.2016.06.029.

Kaushik, R. T., & Bhandarkar, M. (2010). Greenhdfs: Towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster. In *Proceedings of the usenix annual technical conference* (p. 109).

Khaneja, G. (2015). *An experimental study of monolithic scheduler architecture in cloud computing systems*. Ph.D. thesis. University of Illinois at Urbana-Champaign.

Koomey, J. (2011). *Growth in data center electricity use 2005 to 2010: A report by Analytical Press, completed at the request of The New York Times*: 9. Analytic Press.

Lee, Y. C., & Zomaya, A. Y. (2012). Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing, 60*(2), 268–280.

Liu, Z., & Cho, S. (2012). Characterizing machines and workloads on a Google cluster. In *Proceedings of the eight international workshop on scheduling and resource management for parallel and distributed systems (SRMPDS). Pittsburgh, PA, USA*.

Lo, D., Cheng, L., Govindaraju, R., Ranganathan, P., & Kozyrakis, C. (2015). Heracles: improving resource efficiency at scale. In *ACM sigarch computer architecture news: 43* (pp. 450–462). ACM.

Luo, X., Wang, Y., Zhang, Z., & Wang, H. (2013). Superset: A non-uniform replica placement strategy towards high-performance and cost-effective distributed storage service. In *Proceedings of the international conference on advanced cloud and big data (CBD)* (pp. 139–146). IEEE.

Miyoshi, A., Lefurgy, C., Van Hensbergen, E., Rajamony, R., & Rajkumar, R. (2002). Critical power slope: Understanding the runtime effects of frequency scaling. In *Proceedings of the sixteenth international conference on supercomputing* (pp. 35–44). ACM.

Orgerie, A.-C., Lefèvre, L., & Gelas, J.-P. (2008). Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems. In *Proceedings of the fourteenth IEEE international conference on parallel and distributed systems* (pp. 171–178). IEEE.

Reiss, C., Tumanov, A., Ganger, G. R., Katz, R. H., & Kozuch, M. A. (2012a). Heterogeneity and dynamicity of clouds at scale: Google trace analysis. *ACM symposium on cloud computing (SOCC). San Jose, CA, USA*.

Reiss, C., Wilkes, J., & Hellerstein, J. L. (2011). Google cluster-usage traces: format + schema. *Technical Report*. Mountain View, CA, USA: Google Inc.

Reiss, C., Wilkes, J., & Hellerstein, J. L. (2012b). Obfuscatory obscanturism: Making workload traces of commercially-sensitive systems safe to release. In *Proceedings of the third international workshop on cloud management (Cloudman)* (pp. 1279–1286). Maui, HI, USA: IEEE.

Ricciardi, S., Careglio, D., Sole-Pareta, J., Fiore, U., Palmieri, F., et al. (2011). Saving energy in data center infrastructures. In *Proceedings of the first international conference on data compression, communications and processing (CCP)* (pp. 265–270). IEEE.

Schwarzkopf, M., Konwinski, A., Abd-El-Malek, M., & Wilkes, J. (2013). Omega: Flexible, scalable schedulers for large compute clusters. In *Proceedings of the eight ACM European conference on computer systems* (pp. 351–364). ACM.

Sharma, R. K., Bash, C. E., Patel, C. D., Friedrich, R. J., & Chase, J. S. (2005). Balance of power: Dynamic thermal management for internet data centers. *IEEE Internet Computing, 9*(1), 42–49.

Sohrabi, S., Tang, A., Moser, I., & Aleti, A. (2016). Adaptive virtual machine migration mechanism for energy efficiency. In *Proceedings of the fifth international workshop on green and sustainable software* (pp. 8–14). ACM.

Thereska, E., Donnelly, A., & Narayanan, D. (2011). Sierra: Practical power-proportionality for data center storage. In *Proceedings of the sixth conference on computer systems* (pp. 169–182). ACM.

Security Supportive Energy-Aware Scheduling and Energy Policies for Cloud Environments

As a second step in the collaboration we started in Cracow we explored the fourth research objective of this thesis dissertation: *"Proof that Genetic algorithms are an excellent solution to efficiently distribute tasks among servers in data centers taking into account performance, energy, and security restrictions"*. We defined and developed a set of performance and energy-aware strategies for resource allocation, task scheduling, and for the hibernation of virtual machines. The idea behind this model is to combine energy and performance-aware scheduling policies in order to hibernate those virtual machines that operate in idle state. The efficiency achieved by applying the proposed models has been tested using the realistic large-scale cloud-computing system simulator, that is, the SCORE simulator. Obtained results show that a balance between low energy consumption and short makespan can be achieved.

Several security constraints may be considered in this model. Each security constraint is characterized by: a) Security Demands (SD) of tasks; and b) Trust Levels (TL) provided by virtual machines. SD and TL are computed during the scheduling process in order to provide proper security services.

The main contributions include the combination of the following two different approaches for the improvement of energy efficiency into one model: a) an energy-aware scheduler that assigns tasks to VMs according to security demands; and b) a set of energy-efficiency policies that hibernate idle resources.

Experimental results show that the proposed solution reduces up to 45% of the energy consumption of the cloud-computing system. Such a significant improvement was achieved by the combination of an energy-aware scheduler with energy-efficiency policies focused on the hibernation of VMs.

This work was published in *Journal of Parallel and Distributed Computing.* This Journal is indexed in JCR with an **Impact Factor of 1.815**. The Journal stands in ranking **Q2** in Computer Science, Theory & Methods (33/103).

J. Parallel Distrib. Comput. 119 (2018) 191–202

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Security supportive energy-aware scheduling and energy policies for cloud environments

Damián Fernández-Cerero [a,*], Agnieszka Jakóbik [b], Daniel Grzonka [b], Joanna Kołodziej [b], Alejandro Fernández-Montes [a]

[a] Department of Computer Languages and Systems, University of Seville, Spain
[b] Department of Computer Science, Cracow University of Technology, Poland

HIGHLIGHTS

- We propose energy-efficiency strategies for task scheduling and hibernating VMs.
- We combine energy and time-based criteria in order to sleep idle resources.
- We take into account several security constraints in our model.
- The effectiveness of the proposed model has been confirmed by simulation experiments.

ARTICLE INFO

Article history:
Received 3 October 2017
Received in revised form 29 March 2018
Accepted 24 April 2018
Available online 4 May 2018

Keywords:
Cloud computing
Energy efficiency
Independent task scheduling
Genetic algorithms
VM hibernating
Cloud security

ABSTRACT

Cloud computing (CC) systems are the most popular computational environments for providing elastic and scalable services on a massive scale. The nature of such systems often results in energy-related problems that have to be solved for sustainability, cost reduction, and environment protection.

In this paper we defined and developed a set of performance and energy-aware strategies for resource allocation, task scheduling, and for the hibernation of virtual machines. The idea behind this model is to combine energy and performance-aware scheduling policies in order to hibernate those virtual machines that operate in idle state. The efficiency achieved by applying the proposed models has been tested using a realistic large-scale CC system simulator. Obtained results show that a balance between low energy consumption and short makespan can be achieved.

Several security constraints may be considered in this model. Each security constraint is characterized by: (a) Security Demands (SD) of tasks; and (b) Trust Levels (TL) provided by virtual machines. SD and TL are computed during the scheduling process in order to provide proper security services.

Experimental results show that the proposed solution reduces up to 45% of the energy consumption of the CC system. Such significant improvement was achieved by the combination of an energy-aware scheduler with energy-efficiency policies focused on the hibernation of VMs.

1. Introduction

Virtualization of resources and Containerization Platforms, such as Docker, have improved the resource efficiency in Cloud Computing (CC) environments. These strategies allow the execution of several heterogeneous services, such as MapReduce frameworks, web servers, databases, and multi-purpose virtual machines on the same physical resources. Although both performance and energy efficiency in CC environments depend mainly on hardware features, proper scheduling policies may significantly shorten task completion time, which can lead to the reduction of the energy consumption in CC environments [29,33].

CC systems should ensure the appropriate security level for every task deployed on the system [37], and must provide tools for CC operators to develop security frameworks that suit their use cases [38].

In this paper, we defined various energy-efficient optimization strategies for multi-purpose central, monolithic schedulers in CC systems. The energy efficiency is achieved through dedicated policies that hibernate virtual machines that run in an idle state. Moreover, we present a new model for the calculation of the energy consumption of security operations. Based on this model, CC operators are able to select one of the possible security levels.

[*] Corresponding author.
E-mail addresses: damiancerero@us.es (D. Fernández-Cerero), akrok@pk.edu.pl (A. Jakóbik), dgrzonka@pk.edu.pl (D. Grzonka), jokolodziej@pk.edu.pl (J. Kołodziej), afdez@us.es (A. Fernández-Montes).

https://doi.org/10.1016/j.jpdc.2018.04.015
0743-7315/© 2018 Elsevier Inc. All rights reserved.

This information enables users to set longer or shorter keys for cryptographic procedures. Such key-scaling related services are available in Amazon Cloud, RackSpace, OpenStack, and Google Clouds [1,3–5].

The presented model may be used in any High-Performance Computing system that requires the assignation of tasks to computing units. The computing units used in this work, thus, virtual machines, are characterized by their computing capacity. This model could be adapted to work with any type of computing unit characterized by its computational power, such as those used in edge computing networks, grid computing systems, systems based on micro-containers, and small data centers.

The paper is organized as follows. In Section 2, we present the state of the art in measurement of energy for virtualized environments and optimization of energy consumption in CC. In Section 3, we present various approaches and methods: (a) a methodology for estimation of power consumption of virtual machines in CC; (b) a Batch Scheduling problem in CC with security criteria; (c) computation of the total energy consumed by a given task in a schedule; (d) a multi-objective scheduling problem with energy consumption and security; and (e) energy-efficiency policies based on hibernating virtual machines are presented. In Section 4, the experimental environment and scenarios, where the two more representative energy-efficiency strategies have been implemented, are described. We evaluate our models through extensive realistic simulation. Achieved results are presented and analyzed in Section 5. Finally, the paper is summarized in Section 6.

2. Related work and progress beyond the state-of-the-art

Several strategies have been developed over last years for the estimation of energy consumption of virtual machines in Cloud Computing systems. The power requirements of physical servers in a cluster can be measured by the means of established procedures [35,46,47] focused on measuring the utilization of microprocessors. The measurement process is more complex when virtual machines are considered [16,53].

The VM energy consumption may be computed in terms of the CPU, memory, and IO utilization, as proposed by Li, et al. in [54].

In [13], the virtual machine energy consumption was computed according to hardware performance results collected from various components, mainly the CPU-related ones. The memory utilization is considered in the approach proposed by Krishnan in [53]. The energy consumption of both network interface cards and hard drives was also taken into account in the model presented by Wassmann et al. in [68].

In addition, a linear model based on nine independent parameters was proposed by Bertran et al. in [12] in order to measure virtual machines energy consumption. These parameters were, among others, the first level cache activity and the number of accesses to the first level cache per cycle.

On the other hand, a Gaussian Stochastic Mixture model was proposed in [18] by Dhiman in opposition to the aforementioned linear mathematical models with independent parameters. However, none of the proposed strategies are sufficient to deal with realistic cloud virtual resource allocation and scheduling problems [59].

Various tools aiming to compute VM power consumption have been proposed in an isolated way from cloud platforms. These algorithms, such as FitGreen [20], Julemeter [44], and the algorithm proposed by Murwantara [57], need special configurations to access the hardware layer. Hence, they can only be deployed as an external framework at the cloud provider or Infrastructure as a Service level.

As the importance of CC rises, the energy efficiency of these infrastructures becomes more and more important. These facilities, which consume as much energy as many factories, are responsible for approximately 1.5% of global energy consumption [52].

The strategies developed for optimization of energy consumption in CC may be classified into three major categories:

- **Cooling strategies**. The goal of these strategies is the reduction of the energy consumption of chillers, which represents an important part of the total energy used by a data center. A dynamic thermal management system at the data center level was proposed by Sharma et al. [65]. Rising the data center operating temperature was proposed by El-Sayed et al. [21], whereas Gao et al. extensively evaluated the risks related to this approach [28]. On the other hand, Zimmerman et al. proposed the reutilization of the wasted heat in order to propose a hot water-cooled data center prototype [70]. A multi-stage outdoor air-enabled cooling system composed of a water-side economizer, an air-side economizer, and mechanical cooling was proposed by Kim et al. [50].
- **Hardware-related strategies**. Many hardware-based models have been proposed in order to achieve high energy-conservation levels. Dynamic Voltage Frequency Scaling (DVFS) model is one of the most popular approaches. Miyoshi et al. evaluated benefits of using CPU DVFS [56], while David et al. applied this technique to memory components [17]. The replacement of mechanical components, such as HDDs, with non-mechanical devices, such as SSDs, was proposed by Andersen et al. [8]. Regarding the power supply, a dynamic and non-uniform global power-allocation model among nodes was proposed by Femal et al. [23].
- **VM consolidation and migration**. Several strategies have been developed in order to schedule VMs and redistribute them to reduce the energy consumption. Beloglazov et al. [10] propose a resource management system focused on the minimization of the energy consumption by the utilization of VM allocation and migration policies. This work is extended by the proposal of several heuristics for dynamic consolidation of VMs in [11].

While many of these strategies have been adopted by companies, such as Google, Microsoft, and Amazon, there is another area of research that has been barely implemented on CC systems: the achievement of power-proportional systems by turning off idle resources. The idea is that the energy consumption of CC systems should be proportional to workload requirements, which are hardly ever stable.

There are some reasons that prevent the shut-down of machines that run in an idle state, including: the fear of any change that could break operational requirements [25], the complexity and heterogeneity of all the subsystems involved, and the fast development of new systems and paradigms that could break the established standards and systems. However, keeping servers underutilized or in an idle state is highly inefficient from an energy-efficiency perspective.

Much effort has been made by the research community in order to hibernate underutilized resources. A power-proportional distributed file based on the partition of data centers according to redundant data replicas was proposed by Amur et al. [7]. In such systems, servers that store redundant replicas of data may be switched off. On the other hand, Kaushik and al. proposed in [48] a variant of Hadoop Distributed File System that partitions data centers into zones according to the data usage, which enables servers that store low-used data to be shut down.

Other approaches have been proposed for small mobile systems, such as Virtual Backbone Scheduling [69], and multi-flow multicast transmission [66]. These strategies are well-known in wireless networks and sensor networks [22] environments. However, the described approaches are not easily applicable to CC systems, since the shut-down of CC servers is more complex and expensive than in the aforementioned mobile systems.

The novelty of the research presented in this paper is the combination of the following two different approaches for the improvement of energy efficiency into one model: (a) an energy-aware scheduler that assigns tasks to VMs according to security demands; and (b) a set of energy-efficiency policies that hibernate underutilized resources, based on the energy policies presented in [26,27] for Grid Computing systems. These energy-efficiency policies have been evolved in order to be applied to CC systems.

Major contributions of this paper include:

1. The proposal of a task service model combining a security-aware task scheduler with a set of energy-efficiency policies.
2. The implementation and testing of the proposed model by using a realistic CC simulator.
3. The analysis of the impact of the proposed algorithms on the task processing flow and the energy consumption of the CC system.

Moreover, we developed a theoretical model for the schedule of tasks according to the energy consumption of the security operations related to tasks.

3. Approaches for energy saving and security issues in CC environments

3.1. VM power consumption

The construction of a model for the energy consumption of virtual machines in CC systems is not straightforward. It depends on several elements and processes, including the virtualization process. However, the power consumption of various components, such as microprocessors, memory, devices, hard drives, and networks may be measured by the means of frameworks like Watts UP PRO Power, and APIs like Amazon Cloud CloudWatch metrics [2].

Moreover, models of energy consumption for virtual machines may be defined as an extension of those applied to physical servers, as long as the virtual machines features are taken into consideration.

Let P_{Static} denote the power a server required to run all the tasks that a VM needs to be ready for work. $P_{Virtual}$ denotes the dynamic power used by VMs hosted by that machine. The overall server power consumption may be described as follows [40]:

$$P_{Phys} = P_{Static} + \sum_{i=1,\ldots,m} P(VM_i) = P_{Static} + P_{Virtual}, \qquad (1)$$

where $P(VM_i)$ denotes the energy consumed by the ith virtual machine and m is the number of available VMs. This value is estimated by the means of several approaches. The non-observable parameter $P(VM_i)$ is derived from the observable parameter P_{Phys}.

These methods are mathematical models that consider the power-related resources as independent parameters. Several samples of P_{Phys} are typically collected to estimate the $P(VM_i)$ parameter. This data can be collected by following a black-box approach, i.e. by using a virtual machine hypervisor. On the other hand, a proxy may be deployed in each VM if a white-box strategy is to be used to collect this data [34].

In this work we follow the approach presented in [13]. This means that the energy consumption of virtual machines is based on VM states (working, idle, or hibernated).

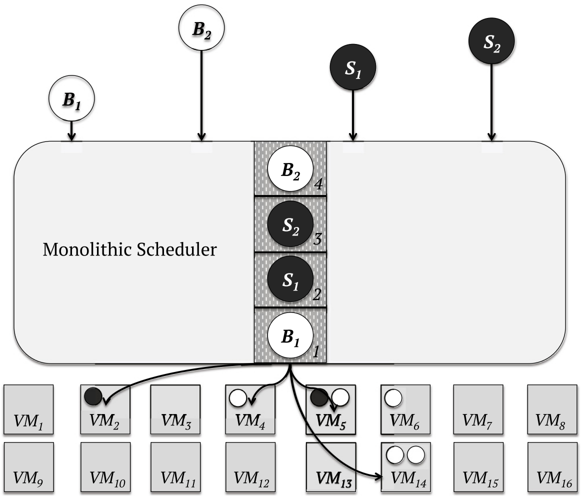


Fig. 1. Single-path scheduling workflow, *B* - Batch type task, *S* - Service type task, *M* - Virtual Machine.

3.2. Monolithic scheduling with a central scheduler

Conceptually, a monolithic scheduler is an omniscient unit responsible for all scheduling decisions, for the allocation all resources, and for maintaining the task deployment process. In this model all workloads are governed by the same scheduler and all tasks are processed by the same scheduling logic [64]. The scheduling algorithm applies a set of heuristics according to tasks requirements, then deploys the tasks on the chosen resources and updates the system state, as illustrated in Fig. 1. Monolithic schedulers usually implement complex scheduling algorithms in order to fulfill various workload types. In this work we consider two types of tasks:

- **Batch tasks**: This type of workload is composed of several independent tasks that can be processed in parallel. Tasks arrive at the system at the same time. Execution of a batch is completed when all of the tasks are finished. After that the whole batch may be processed by another service, stored, or send back to the end user. MapReduce jobs are an example of batch tasks.
- **Service tasks**: This type of workload is composed of long-running tasks. As opposed to the batch tasks, these tasks have no determined end, but are submitted by an operator (or an automated equivalent) and are killed when they are no longer required. Web server instances or service instances, such as BigTable [15], are good examples of service tasks.

In addition to the classical scheduling-related challenges, like minimizing the time a task waits in a queue, satisfying task constraints, respecting priorities, fulfilling end-user SLAs, etc., the ever-growing use of the Cloud Computing paradigm and large-scale web services add several new challenges, such as: (a) scalability; (b) flexibility; (c) scheduling algorithms complexity; and (d) environment fragmentation. These challenges have been addressed by developing new distributed approaches and the scheduling process, such as: (a) shared state scheduling frameworks (e.g. Google Omega [64]); (b) two-level scheduling frameworks (e.g. Mesos [36]); (c) distributed scheduling frameworks (e.g. Sparrow [58]); and (d) hybrid scheduling frameworks (e.g. Mercury [45]).

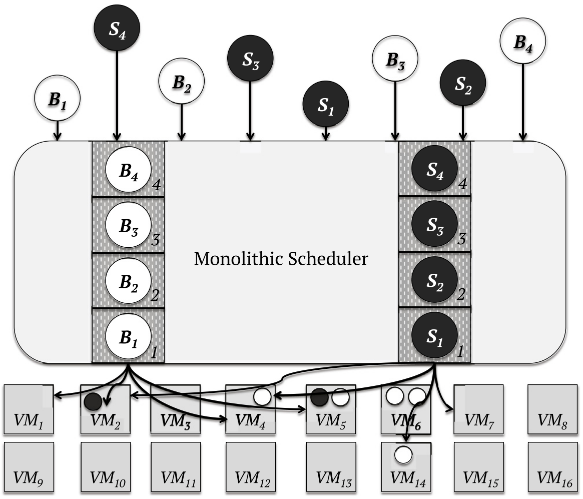


Fig. 2. Multi-path scheduling workflow.

However, for most usual scenarios, such as those present in low and mid-size CC infrastructures up to approximately 10,000 machines, monolithic scheduling frameworks, such as Google Borg [14], are still the best and simplest option.

Two monolithic scheduling approaches are taken into consideration in this paper:

- **Single-path**: This scheduling strategy uses a single scheduling path for every task in the workload, as shown in Fig. 1.
- **Multi-path**: This scheduling strategy uses several scheduling paths by taking advantage of internal parallelism and multi-threading to solve head-of-line blocking and scalability issues, among others. In this work, the multi-path scheduling process represents a system composed of two scheduler paths. The first scheduler path performs the scheduling logic related to batch tasks, whereas the second one is responsible for the scheduling logic related to service tasks. In this approach, any given service task would only need to wait in queue until all previous service tasks are scheduled, since they are independently scheduled, as shown in Fig. 2.

3.2.1. *Batch task scheduling considering security demands*

In this work, we consider the problem of Independent Batch Scheduling in large Cloud Computing systems. Fig. 3 shows the workflow of the simulated environment, which is composed of the following processes: (a) generation and collection of tasks; (b) task scheduling; (c) task execution; (d) results storage; (e) communication with end-users; and (f) management of the security issues related to all the aforementioned processes.

However, a single batch may contain tasks that require different security levels: e.g. the process of an open-access free stock and the process of clinical images of a hospital. The security demands of tasks were introduced in order to meet these security-related requirements [31,32,42]. The scheduler computes these security demands by implementing a security demand vector that represents the security requirements of the tasks:

$$SD = [sd_1, \dots, sd_n], \tag{2}$$

where sd_j is specified by the jth task in the batch. On the other hand, different computing units may offer different security services and levels. Amazon Cloud offers high security standards, whereas a private Cloud with an older version of software may offer a lower security level. To reflect this situation, the following trust level vector is introduced:

$$TL = [tl_1, \dots, tl_m]. \tag{3}$$

It represents the security capacities of all VMs in the system. All the parameters assume values in the range [0,1], where 0 means the lowest security level for a task and the least trusted VM. A particular task will be scheduled to a VM which offers a security level greater or equal than that demanded by the task.

In order to achieve an effective and efficient scheduling process, the previously developed Non-Deterministic Central Scheduler based on a Genetic Algorithm [30,41,42] has been chosen as the main scheduling policy for the monolithic scheduler. In addition to the aforementioned makespan-focused Genetic Algorithm, a new criterion that takes the energy consumption of every task into account is considered. The developed scheduling policy relies on an Expected Time to Compute (ETC) matrix [51], adapted to virtual machines (ETC_V). The ETC_V matrix can be defined as follows [39]:

$$ETC_V = [ETC_V[j][i]]_{j=1,\dots,n;i=1,\dots,m} \tag{4}$$

where

$$ETC_V[j][i] = wl_j/cc_i, \tag{5}$$

where cc_i denotes the computational capacity of the ith virtual machine and wl_j is the workload of the jth task; n and m represent the number of tasks and number of virtual machines, respectively.

Security demands involve additional security operations that must be performed before or after task execution. The possible security operations in the CC system are denoted by a padlock icon in Fig. 3. Security issues may require additional computing time. For this reason, we used an extended version of the ETC_V matrix — $SBETC$ (Security Biased Expected Time to Compute) matrix. This matrix takes the additional security bias (SB) parameter b into consideration in order to represent the time spent for security operations [42]:

$$b(sd_j, wl_j, tl_i, cc_i). \tag{6}$$

All the biases give the matrix representation:

$$SB[j][i](SD, TL) = [b(sd_j, wl_j, tl_i, cc_i)], \tag{7}$$

where SD and TL denote the security demand vector (see Eq. (2)), and the trust level vector (see Eq. (3)) for the VMs in the system, respectively.

The ETC_V matrix can be evolved to the Security Biased Expected Time to Compute ($SBETC$) matrix when the security biases are considered:

$$SBETC[j][i](SD, TL) = wl_j/cc_i + b(sd_j, wl_j, tl_i, cc_i) \tag{8}$$

$$SBETC(SD, TL) = SB(SD, TL) + ETC_V. \tag{9}$$

The main goal of the scheduling and allocation processes is to find an optimal solution for the specified criteria. Among all batch workload scheduling process factors, the makespan is considered the main objective. It can be described as follows:

$$C_{\max} = \min_{S \in Schedules} \left\{ \max_{j \in Tasks} C_j \right\}, \tag{10}$$

where C_j is the jth task completion time. $Tasks$ is the set of tasks in the batch of task, while $Schedules$ represents the set of all possible schedules that may be generated for the tasks of that batch of task, as illustrated in Fig. 4.

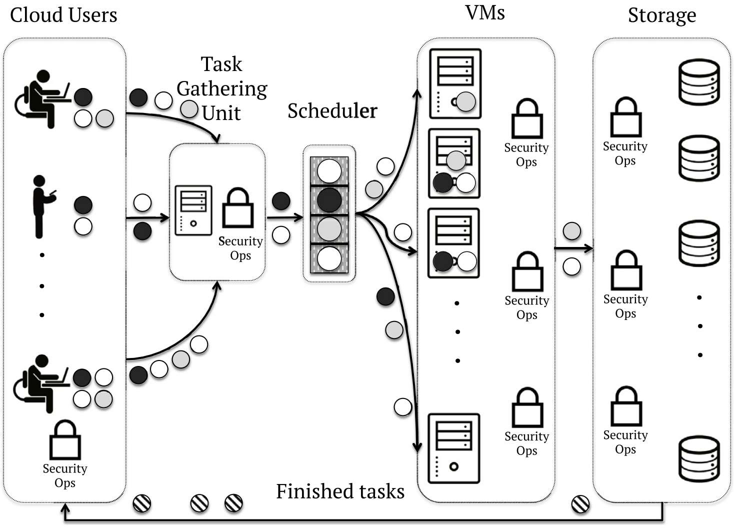


Fig. 3. Cloud computing system workflow, Security Ops — additional security operations and procedures.

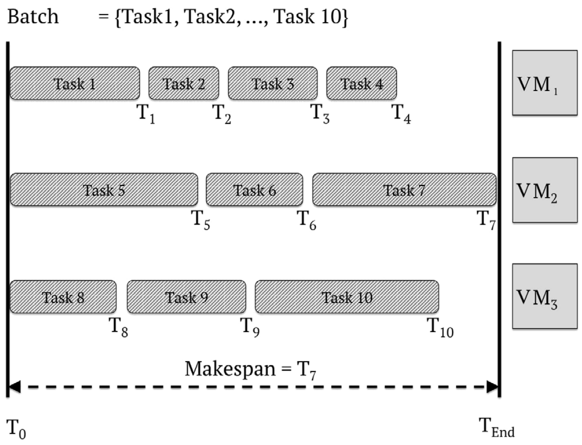


Fig. 4. Makespan measuring workflow.

3.3. Genetic algorithm

The scheduling of tasks in cloud-computing data centers constitutes an NP-complete problem [67], whose complexity depends on the features considered [51], such as: (a) the number scheduling of criteria to be optimized (one vs. multi-criteria); (b) nature of the environment (static vs. dynamic); (c) nature of tasks (*Batch* or *Service*); and (d) dependency between tasks (independent vs. dependent).

In this work, we use a heuristic algorithm that takes into account the aforementioned requirements in order to solve the NP-complete problem. This scheduling algorithm is based on a genetic algorithm with dedicated population representation [30,39], which can be characterized as follows: (a) a single gene represents one task, which is unique within the population; (b) each chromosome is composed by a set of tasks (genes); (c) each individual is composed of one chromosome and represents a scheduling assignation for a single computing node; (d) the population is composed of *m* individuals and represents a schedule for all *n* tasks; (e) the fitness function depends on the optimization objectives presented in Section 3.5. All individuals take part in the reproduction process. Individuals presenting the lowest value for the fitness function (best adapted) are crossed with worst-adapted individuals (those that show the highest values for the fitness function). Crossing involves exchanging genes between chromosomes. The population obtained in the evolution process defines the suboptimal schedule.

3.4. Energy calculation

Two different power states are considered for each virtual machine in the CC: *busy* (100% core computational power is used for task computing) and *idle* state. Let: t^i_{idle} denote the time the *i*th machine spends in an idle state; t^i_{busy} — the time the machine spends in computing tasks-related operations; P^i_{idle} — the required power for a machine to run in idle state; and P^i_{busy} — the power required by a machine to perform actual computing operations. The power required to perform security-related activities is assumed to be the same as in busy mode.

The aforementioned parameters may vary in each schedule and can be defined as follows [42]:

$$t^i_{busy} = \max_{j\in Tasks^i} C_j \tag{11}$$

$$t^i_{idle} = C_{\max} - t^i_{busy} \tag{12}$$

$$t^i_{sec} = \sum_{j\in Tasks^i} b^i_j \tag{13}$$

where $Tasks^i$ represents the tasks assigned to VM_i and t^i_{sec} denotes the time devoted to processing only the security-related operations.

The total energy consumption can be denoted as follows:

$$E_{total} = \sum_{i=1}^{m} \int_0^{C_{max}} Pow_{VM_i}(t)dt = \sum_{i=1}^{m} (P^i_{idle} * t^i_{idle} + P^i_{busy} * (t^i_{busy} + t^i_{sec})). \tag{14}$$

The presented energetic model is designed for the assignation of tasks to virtual machines. However, this model could be adapted to work with other scheduling technologies, such as that based on the assignation of tasks in the form of Linux micro-containers to computing nodes. These micro-containers run only for the task execution time. The energetic model could also be extended in order to consider specific hardware configurations.

3.5. Energy-aware scheduling objectives

The determination of the solution that minimizes the makespan of a given schedule that assumes a constant computing power may be defined as follows:

$$\underset{s \in Schedules}{\text{argmin}} \sum_{\substack{i=1,\dots,m \\ j=1,\dots,n}} \left(\frac{wl_j}{cc_i} + b^i\right)\delta_{i,j}(s) \tag{15}$$

where $\delta_{i,j}(s)$ equals one when the schedule s assigns the jth task to the ith VM. Otherwise, $\delta_{i,j}(s)$ equals zero.

Moreover, the determination of the solution that minimizes the total energy consumption of a given schedule can be written as:

$$\underset{s \in Schedules}{\text{argmin}} \sum_{i=1,\dots,m} \left(\sum_{\substack{j=1 \\ \delta_{i,j}(s)=1}}^{n} P^i_{busy}\left(\frac{wl_j}{cc_i} + b^i\right) + \sum_{\substack{j=1, \\ \delta_{i,j}(s)=0}}^{n} P^i_{idle} t^i_{idle} \right) \tag{16}$$

with the following constraints (see Eqs. (2) and (3)):

$$sd_j \leq tl_i. \tag{17}$$

Various energy-saving approaches may be tested by modifying the trust level of any given machine thanks to the SBETC matrix. Moreover, numerous complex and realistic scenarios may be simulated in order to check whether these strategies can be used in real-life Cloud Computing systems.

In this work, we proposed the following four energy-aware and time-aware scheduling policies based on Eqs. (15) and (16):

1. **Makespan-centric scheduling**. Whenever two given schedules achieve the same (or close) makespan, the less energy-consuming schedule is selected. This approach is desirable when the makespan is the main scheduling objective and the importance of the reduction of energy consumption is low.
2. **Energy-centric scheduling**. Whenever two given schedules present approximately the same energy consumption, the schedule with the shorter makespan is selected. This approach is suitable when the energy efficiency is the main objective and the execution time is not critical.
3. **Makespan-centric scheduling until a given makespan threshold**. In this policy, the minimization of the makespan is the main goal. Once a makespan threshold is achieved, then the minimization of the energy consumption becomes the main objective.
4. **Energy-centric scheduling until a given energy-consumption threshold**. In this policy, the minimization of the energy consumption is the main goal. Once a energy-consumption threshold is achieved, then the minimization of the makespan becomes the main objective.

3.6. Energy policies based on the hibernation of virtual machines

The volume of work that must be executed at any given time by a CC system may significantly change, especially with peak loads largely exceeding mean loads. The proper execution of this ever-changing workload while achieving energy-proportionality represents a major challenge. CC operators may choose between the following strategies in order to face the challenge: (a) the over-provision of the data-center to satisfy worst-case scenarios; and (b) the adjustment of the available resources according to the present and future workload demands.

The first of these two approaches represents the main trend implemented in the vast majority of large Cloud Computing systems. However, this strategy requires a high amount of energy to keep servers in an idle state during long periods of time, while they wait to serve worst-case peak loads.

Many software solutions implement the second strategy by switching off either server components or whole servers to reduce the energy consumption in low-utilization periods. However, this approach could damage end-user experience and SLAs if these workload peaks are not properly determined and served.

Various energy-efficiency policies based on the shut-down of machines have been tested in grid computing scenarios, including: (a) the shut-down of every machine whenever possible; and (b) the shut-down of machines according to the workload demands. These policies have shown good energy-savings in [26,27]. In this work, we adapted these energy-efficiency policies, which are designed for grid computing environments, in order to be applied in CC systems.

Our aim is the development of energy-efficiency policies that rely on resource schedulers that could in CC systems of different sizes, and that may serve various and heterogeneous workloads rather than focusing on a specific scenario or infrastructure.

The power-off energy-efficiency policies are responsible for deciding whether any given machine should be turned off or kept in an idle state, and for performing the actual hibernation process while keeping the environment state information up to date.

These power-off policies may be deterministic, such as the *Always power off* policy, or probabilistic, which forecast future workload demands based on historical data and then to perform required actions according to this prediction. Power-off policies may check various system, workload and machine parameters in order to make decision about shutting any given machine down.

The following deterministic policies have been considered in this paper:

1. **Never power off**: This power-off policy prevents any given virtual machine to be hibernated. This is the current operating approach in many real Cloud Computing systems nowadays. Due to this, the power-off policy should be considered and studied so the energy savings achieved by any other power-off policy can be compared to the current power consumption scenario.
2. **Always power off**: Opposite to the *Never power off* policy, this policy always tries to hibernate any virtual machine that becomes idle.

The shut-down process is performed whenever any resource in use (RAM, CPU) is released due to the execution of a task finished. At this moment, the system makes a decision whether the machine those resources belongs to should be turned off or not. The system prevents any virtual machine that is executing tasks from being hibernated.

4. Evaluation of energy-aware scheduling vs. makespan scheduling in cloud computing systems

We propose an environment that simulates a monolithic scheduling framework to serve realistic and heterogeneous workloads in order to test the proposed strategies. The CC environment has been simulated for seven days of operation time and various combinations of the energy policies developed and described in Section 3.6 have been evaluated.

In the following subsections a simulation tool, a test suite and a designed environment are presented in detail.

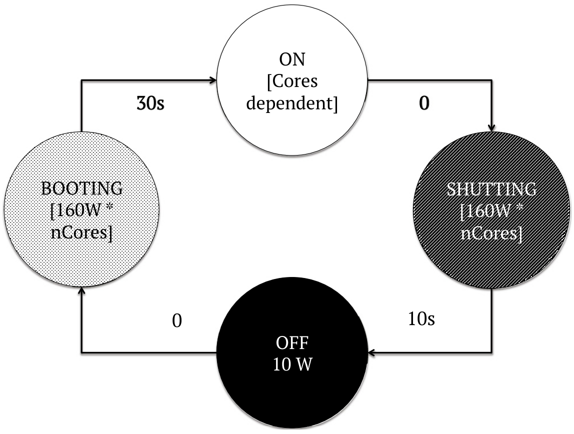


Fig. 5. Machine power states.

4.1. Simulation tool

In this work we used the SCORE simulator presented in [24]. This simulator enables us to focus on the development of energy-efficiency policies and on the performance of simulations of various scheduling frameworks and data-center environments. This simulation tool has been modified in order to perform energy-efficiency analysis by applying an energy-consumption model which considers the following states for each CPU core in a machine: (a) *On*: 150 W (b) *Idle*: 70 W. The energy consumption is linearly computed in terms of the utilization of each CPU core. In addition to these CPU core power consumption states, the following machine power states have been assumed: (a) *Hibernated*: 10W (b) *Hibernating*: 160 W * number of cores (c) *Powering On*: 160 W * number of cores.

Regarding the shut-down process time parameters, the following values have been considered: (a) $T_{On\rightarrow Hibernated}$: 10 s, and (b) $T_{Hibernated\rightarrow On}$: 30 s. The power states and transitions are shown in Fig. 5.

In order to develop and apply our energy-efficiency policies, a new set of modules has been built on top of the current simulator. Among these additions, the following can be found: (a) sorting, (b) scheduling, and (c) power-off policies.

However, in order to preserve trust in the schedulers' implementations, the behavior of the overall simulation process has not been modified. Instead of modifying the current implementation, hooks were placed in key parts of the simulation process to execute our developed policies and to register new key performance indicators, which have been added in order to measure the impact on data center energy consumption.

As a result of this approach, the developed energy-efficiency policies have achieved a high level of isolation from the base simulator implementation, thereby affecting the original simulator design to a minimum extent.

4.2. Cloud computing center

A CC data center composed of 1000 heterogeneous virtual machines has been modeled. Each machine has the following features:

- **Computing profile**: Processor's millions of instructions per second (MIPS) have been simulated by generating randomly a [1× - 4×] computing speed factor. Thus, a given VM may be, as a maximum, four times faster than the slowest one: $cc_i \in [75000, 300000]$ MIPS.
- **Energy profile**: Processor's power consumption heterogeneity has been simulated by generating randomly a [1× - 4×] energy consumption factor. Thus, a given machine *M* may be (as a maximum) four times more energy-wasting than the more efficient one. Hence, for a 4-core server, the maximum power consumption may be described as: $P_{total} \in [300, 1200]$ W.
- **Security profile**: Cryptographic services have been chosen according to the FIPS standard [32], and ISO/IEC 19790 standard [33] for security requirements for cryptographic modules, as described in [42]. These standards specify four operating levels of general security requirements for cryptography modules, which have been simulated by randomly generating a security factor in the range [1–4]. Therefore, $TL \in [0.25, 1]$.
- **Computational resources**: Every machine has 4 CPU cores and 16 GB of RAM.

4.3. Workload

The patterns present in the realistic Google traces [61,62] were followed to generate the synthetic workload used in the experimentation. The interpretations by [6,19,55,60] have been studied to model the synthetic workloads.

These workload tasks are composed of one or more (sometimes more than thousand) tasks. Every task is modeled to use a given number of millions of instructions (MI).

Moreover, the two types of tasks described in Section 3.2 are considered.

Each experiment executes the workload generated by replicating the behavior of the workload present in typical Google data centers. Therefore, although the workload generated in each simulation run is unique, it follows the same model design. In this workload, the vast majority of tasks are batch tasks, however, over half of the available resources are reserved to service tasks.

Moreover, batch tasks are usually composed of a greater number of tasks than service tasks. However, these tasks require fewer resources and run for a shorter time than service tasks. Hence, the simulator generates a day/night patterned synthetic dataset composed of tasks whose attributes follow an exponential distribution.

Taking into account the aforementioned environment and workload scenario, the generated workload presents 22,208 batch tasks and 2252 service tasks for each experiment that simulates 7 days of operation time, reaching 30.08% of average computational power and 25.72% of memory in use. This data center utilization rates follow industry trends presented in [9,63].

4.4. Key performance indicators

In order to measure the results of the application of energy-efficiency policies that switch machines on/off, the authors need to measure key performance indicators of the data-center operation. These indicators have been divided into two categories: (a) energy savings; and (b) performance.

The following indicators were selected in order to describe the energy savings and the behavior of the powering on/off operations:

- **Energy consumption**: The total energy consumed in each experiment, E_{total} (14).
- **Energy savings**: The total energy saved in each experiment.

The following indicators were selected as the most relevant performance indicators:

- **Queue time**: Represents the time a task waits in the queue until it is scheduled. This indicator is usually related to the real computing experience, and therefore it is critical to maintain this time as short as possible.

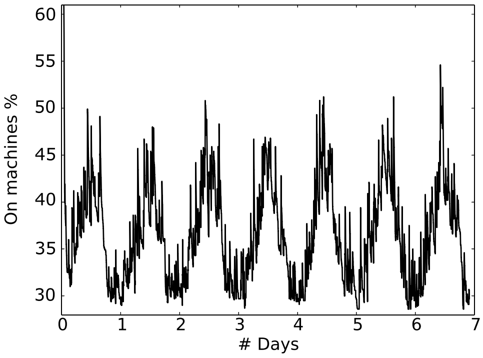


Fig. 6. Percentage of powered-on machines when the *Always power off* policy is used for the single-path scheduler.

- **Makespan**: C_{max} (see (10)).

In order to analyze and compare the energy savings and the performance impact of deploying hibernating energy-efficiency policies, the simplest and most aggressive energy policies have been applied, i.e., the *Never power off* and the *Always power off*. They will be applied to the most representative scheduling policies proposed in Section 3.5. Among them:

- The Makespan-centric scheduling (policy 1) is applied to batch tasks. The scheduling policy tries to load every machine up to 90%. The rest of the computational power is used for service tasks (cf. [49]). The evolution of the fitness function value in average of the genetic process applied to batch tasks can be observed in Fig. 7b;
- The Energy-centric scheduling (policy 2) is applied to batch tasks. The same scheduling policy described in the Makespan-centric scheduling is used for service tasks. The evolution of the fitness function value in average of the genetic process applied to batch tasks may be observed in Fig. 7a;
- The Random strategy for both batch and service tasks. This strategy selects a random machine from the subset of machines that meet tasks requirements. This scheduling policy is especially important because many of top-industry companies implement a similar strategy, such as round robin-like methods.

The scheduling algorithm workflows may be described as follows: The random scheduler assigns tasks to VMs randomly, according to the Round Robin-like schema (i.e., the *Random* strategy). The GMakespan (Genetic-based with makespan as the main objective) scheduler assigns tasks according to the solution of the optimization problem shown in (15), by the means of the genetic algorithm described in the policy 1 in Section 3.5. The GEnergy (Genetic-based with energy as a main objective) scheduler assigns tasks according to the solution of optimization problem presented in (16), by the means of the genetic algorithm described in the policy 2 in Section 3.5. The *Never power off* policy lets VMs be in an idle state when the execution of tasks is finished, while the *Always power off* policy hibernates them.

5. Results and discussion

In this section, the simulation results obtained for the *Makespan-centric*, *Energy-centric* and *Random* scheduling policies are discussed through key performance indicators concerning: (a) energy savings; and (b) performance impact. We choose to only show the results obtained for the most representative scheduling policies since the *Makespan-centric scheduling until a given makespan threshold* and the *Energy-centric scheduling until a given energy-consumption threshold* are mixed strategies that could blur the main differences between the opposite *Energy-centric* and *Makespan-centric* scheduling policies. The energy savings and performance are analyzed and compared between the current/base system energy policy (*Never power off* policy) and the *Always power off* energy-efficiency policy. Table 1 shows numeric results for the single-path scheduler and Table 2 presents those for the multi-path scheduler. In general, the more hibernations there are, the more energy is saved, or from another point of view, the less idle the resources, the less energy is wasted.

Moreover, it can be observed that the utilization of only the genetic algorithm that focuses on the minimization of the energy consumption results in a higher energy consumption than the genetic algorithm that focuses on the minimization of the makespan (56.17 MWh vs. 55.96 MWh, as shown in Table 1, scenario Never off policy, *GEnergy* Scheduler vs *GMakespan* Scheduler). On the other hand, it can be noticed that high energy savings up to approximately 45% may be achieved by applying the *Always off* policy for the genetic algorithm that focuses on the minimization of the makespan (30.45 MWh consumed with the *Always off* policy vs. 55.96 MWh consumed with the *Never off* policy for the *GMakespan* Scheduler, as presented in Table 1). This behavior, that is similar for the Monolithic multi-path scheduler presented in Table 2, means that only a 20% of energy is wasted from the theoretical optimum instead of 70% of the current approach.

Regarding the performance, the application of the *Always power off* energy-efficiency policy has a negative impact of approximately 35% in terms of scheduling queue time. This behavior can be observed in Table 1, where *Batch* tasks wait on average approximately 20 more seconds (+40%) in queue for the *GMakespan* scheduler and 15 more seconds (+30%) for the *GEnergy* scheduler in queue. Similarly, *Batch* tasks have a longer makespan (+60s. and +120s. on average for the *GMakespan* and *GEnergy* schedulers respectively, as presented in Table 2) when the combination of the *Always power off* energy-efficiency policy and the genetic algorithm is used. This makespan impact is especially negative when the genetic algorithm that focuses on the minimization of energy is used (+140s. for the *Always off* policy and *GEnergy* vs. *Random* schedulers respectively, as shown in Table 2). On the other hand, it is noticeable that *Service* tasks never suffer from this negative makespan impact as shown in Tables 1 and 2.

It can be noticed that the scheduling policy is crucial not only for the performance, but also for the whole hibernation process. As shown in Table 1, the *Random* scheduling policy almost prevents any hibernation. In this case the *Always power off* policy results in a similar energy consumption (54.66 MWh) compared to that achieved by the *Never power off* policy (56.19 MWh). Table 2 shows that the Multi-path monolithic scheduler presents the same behavior as the Single-path monolithic scheduler, except for the queue times, which are notably lower (−85% between queue time results shown in Table 1 and those presented in Table 2) due to the Multi-path approach preventing the head-of-line blocking issue. Table 3 summarizes the impact of the *Never power off* policy in terms of both queue times and energy consumption compared to leaving machines in an idle state. Fig. 6 presents the percentage of VMs in a hibernated mode for a seven-day time span. It can be observed that the *Always power off* policy fits perfectly the clear day/night pattern workload. Taking into consideration the aforementioned results, we can state that the *Makespan-centric scheduling* policy provides the best results for the goals under consideration, thus: the minimization of the energy consumption through the application of hibernation policies with a minor negative impact on the CC system performance.

Table 1
Results for Monolithic Single-path scheduler.

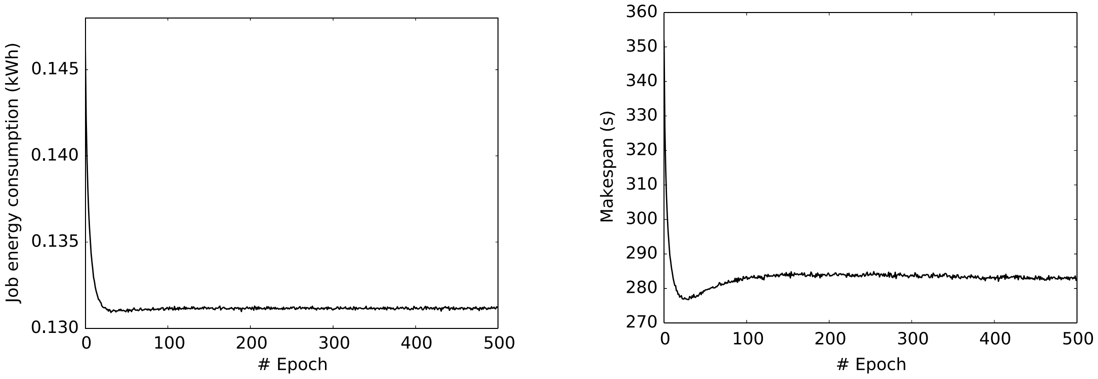
Policy	Scheduler	Energy (MWh)	E savings (MWh)	Queue time (ms)		Makespan (s)	
				Batch	Service	Batch	Service
Never off	Random	56.19	0.00	49.70	57.70	177.44	1,988.21
Always off	Random	54.66	1.57	49.70	57.70	177.21	1,988.21
Never off	GMakespan	55.96	0.00	49.70	57.70	235.71	1,988.21
Always off	GMakespan	30.43	25.92	71.40	69.90	258.95	1,988.30
Never off	GEnergy	56.17	0.00	49.70	57.70	287.48	1,988.21
Always off	GEnergy	30.68	25.83	66.90	69.10	310.19	1,988.38

Table 2
Results for Monolithic Multi-path scheduler.

Policy	Scheduler	Energy (MWh)	E savings (MWh)	Queue time (ms)		Makespan (s)	
				Batch	Service	Batch	Service
Never off	Random	56.21	0.00	06.90	06.50	178.56	1,920.60
Always off	Random	55.00	1.13	06.90	06.50	178.56	1,920.60
Never off	GMakespan	55.90	0.00	06.90	06.50	236.01	1,920.21
Always off	GMakespan	30.08	26.12	10.30	07.20	258.46	1,920.66
Never off	GEnergy	56.09	0.00	06.90	06.50	290.55	1,920.60
Always off	GEnergy	30.65	25.76	11.00	06.90	313.41	1,920.67

Table 3
Always off policy results vs. current situation, represented by the *Never power off* policy.

Scheduler	Strategy	Savings	Queue time diff		Makespan diff	
			Batch	Service	Batch	Service
Random	Single-path	02.79%	0	0	−00.13%	N/A
GMakespan	Single-path	45.99%	+43.67%	+21.14%	+45.94%	N/A
GEnergy	Single-path	45.71%	+34.60%	+19.76%	+74.81%	N/A
Random	Multi-path	02.01%	0	0	0	N/A
GMakespan	Multi-path	46.48%	+49.27%	+10.77%	+44.75%	N/A
GEnergy	Multi-path	45.67%	+59.42%	+06.15%	+75.52%	N/A



(a) Task energy consumption.

(b) Task makespan.

Fig. 7. Genetic process fitness evolution.

An important observation about the genetic process used for finding the solution of the minimization problem (Eqs. (15) and (16)) is that an early stopping strategy should be incorporated. From Fig. 7b and 7a it can be seen that the genetic process should be stopped after approximately 50 epochs in order to achieve the best results.

6. Summary

In this paper a model for reducing the energy consumption in CC environments has been described. The presented approach enables us to reduce the energy consumption of the CC system up to 45%. The proposed model is composed of two parts: (a) an energy-aware independent batch scheduler; and (b) a set of energy-efficiency policies for the hibernation of idle VMs. We proposed four scheduling policies for the control of the energy consumption and the makespan during the assignation of tasks to VMs.

The contributions of this work include:

1. The scheduler task assignation to VMs based on a makespan optimization process. As a result, each batch of tasks is computed in the shortest time, taking into account the current state and the characteristics of the CC system.
2. The hibernation of virtual machines that remain in an idle state, while the rest of VMs continue to execute the batch of tasks. This guarantees the maximum positive impact on the system performance since it does not negatively impact the virtual machines under use.

The proposed scheduler takes the security demands of each task and trust levels of VMs that are computing those tasks into account. Additionally, the proposed model enables us to compute the energy consumption of the whole system, including the energy spent on performing security operations.

The developed methods were tested using the realistic workload of Google traces for seven consecutive days on the simulated environment equipped with 1000 virtual machines. The experimental results show that the application of the proposed model, especially that parameterized with a scheduling policy focused on the minimization of the makespan, in addition to an energy-efficiency policy based on the hibernation of every virtual machine whenever possible, could successfully reduce the energy consumption of large-scale data centers which securely serve heterogeneous workloads without notably impacting on the cloud computing system overall performance.

The next stage of our research is the optimization of security operations. We intend to apply game theory solutions which have been developed previously (see: [43]) for the optimization of the Trust Levels of VMs and for the decision of the applied security biases.

Acknowledgment

This article is based upon work from COST Action IC1406 "High-Performance Modelling and Simulation for Big Data Applications" (cHiPSet), supported by COST (European Cooperation in Science and Technology).

The research is supported by the VPPI — University of Sevilla.

References

[1] Amazon Cloud Scaling Service, URL http://docs.aws.amazon.com/autoscaling/latest/userguide/scaling_plan.html.

[2] Amazon CloudWatch, URL https://aws.amazon.com/cloudwatch/.

[3] Google Cloud Scaling Service, URL https://cloud.google.com/compute/docs/autoscaler/scaling-cpu-load-balancing.

[4] OpenStack Cloud Scaling Service, URL https://wiki.openstack.org/wiki/Heat/AutoScaling.

[5] Rackspace Cloud Scaling Service, URL https://support.rackspace.com/how-to/rackspace-auto-scale-overview/.

[6] O.A. Abdul-Rahman, K. Aida, Towards understanding the usage behavior of Google cloud users: the mice and elephants phenomenon, in: IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Singapore, 2014, pp. 272–277.

[7] H. Amur, J. Cipar, V. Gupta, G.R. Ganger, M.A. Kozuch, K. Schwan, Robust and flexible power-proportional storage, in: Proceedings of the 1st ACM symposium on Cloud computing, ACM, 2010, pp. 217–228.

[8] D.G. Andersen, S. Swanson, Rethinking flash in the data center, IEEE Micro 30 (4) (2010) 52–54.

[9] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al., A view of cloud computing, Commun. ACM 53 (4) (2010) 50–58.

[10] A. Beloglazov, R. Buyya, Energy efficient resource management in virtualized cloud data centers, in: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, IEEE Computer Society, 2010, pp. 826–831.

[11] A. Beloglazov, R. Buyya, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, Concurr. Comput.: Pract. Exper. 24 (13) (2012) 1397–1420.

[12] R. Bertran, Y. Becerra, D. Carrera, V. Beltran, M. Gonzalez, X. Martorell, J. Torres, E. Ayguade, Accurate energy accounting for shared virtualized environments using PMC-based power modeling techniques, in: 2010 11th IEEE/ACM International Conference on Grid Computing, 2010, pp. 1–8. http://dx.doi.org/10.1109/GRID.2010.5697889.

[13] A.E.H. Bohra, V. Chaudhary, VMeter: Power modelling for virtualized clouds, in: 2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010, pp. 1–8. http://dx.doi.org/10.1109/IPDPSW.2010.5470907.

[14] B. Burns, B. Grant, D. Oppenheimer, E. Brewer, J. Wilkes, Borg, omega, and kubernetes, Commun. ACM 59 (5) (2016) 50–57.

[15] F. Chang, J. Dean, S. Ghemawat, W.C. Hsieh, D.A. Wallach, M. Burrows, T. Chandra, A. Fikes, R.E. Gruber, Bigtable: A distributed storage system for structured data, ACM Trans. Comput. Syst. (TOCS) 26 (2) (2008) 4.

[16] M. Colmant, M. Kurpicz, P. Felber, L. Huertas, R. Rouvoy, A. Sobe, Process-level power estimation in VM-based systems, in: Proceedings of the Tenth European Conference on Computer Systems, EuroSys '15, ACM, New York, NY, USA, 2015, pp. 14:1–14:14. http://dx.doi.org/10.1145/2741948.2741971.

[17] H. David, C. Fallin, E. Gorbatov, U.R. Hanebutte, O. Mutlu, Memory power management via dynamic voltage/frequency scaling, in: Proceedings of the 8th ACM International Conference on Autonomic Computing, ACM, 2011, pp. 31–40.

[18] G. Dhiman, K. Mihic, T. Rosing, A system for online power prediction in virtualized environments using Gaussian mixture models, in: Design Automation Conference, 2010, pp. 807–812. http://dx.doi.org/10.1145/1837274.1837478.

[19] S. Di, D. Kondo, C. Franck, Characterizing cloud applications on a Google data center, in: 42nd International Conference on Parallel Processing (ICPP), Lyon, France, 2013.

[20] C. Dupont, T. Schulze, G. Giuliani, A. Somov, F. Hermenier, An energy aware framework for virtual machine placement in cloud federated data centres, in: 2012 Third International Conference on Future Systems: Where Energy, Computing and Communication Meet (e-Energy), 2012, pp. 1–10. http://dx.doi.org/10.1145/2208828.2208832.

[21] N. El-Sayed, I.A. Stefanovici, G. Amvrosiadis, A.A. Hwang, B. Schroeder, Temperature management in data centers: why some (might) like it hot, ACM SIGMETRICS Perform. Eval. Rev. 40 (1) (2012) 163–174.

[22] B. Fateh, M. Govindarasu, Joint scheduling of tasks and messages for energy minimization in interference-aware real-time sensor networks, IEEE Trans. Mob. Comput. 14 (1) (2015) 86–98. http://dx.doi.org/10.1109/TMC.2013.81.

[23] M.E. Femal, V.W. Freeh, Boosting data center performance through non-uniform power allocation, in: Second International Conference on Autonomic Computing, ICAC'05, IEEE, 2005, pp. 250–261.

[24] D. Fernández-Cerero, A. Fernández-Montes, A. Jakóbik, J. Kołodziej, M. Toro, SCORE: Simulator for cloud optimization of resources and energy consumption, Simul. Model. Pract. Theory 82 (2018) 160–173. http://dx.doi.org/10.1016/j.simpat.2018.01.004.

[25] A. Fernández-Montes, D. Fernández-Cerero, L. González-Abril, J.A. Álvarez-García, J.A. Ortega, Energy wasting at internet data centers due to fear, Pattern Recognit. Lett. 67 (2015) 59–65.

[26] A. Fernández-Montes, L. Gonzalez-Abril, J.A. Ortega, L. Lefèvre, Smart scheduling for saving energy in grid computing, Expert Syst. Appl. 39 (10) (2012) 9443–9450.

[27] A. Fernández-Montes, F. Velasco, J. Ortega, Evaluating decision-making performance in a grid-computing environment using DEA, Expert Syst. Appl. 39 (15) (2012) 12061–12070.

[28] X. Gao, Z. Xu, H. Wang, L. Li, X. Wang, Why some like it hot too: Thermal attack on data centers, in: Proceedings of the 2017 ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems, ACM, 2017, pp. 23–24.

[29] D. Grzonka, The analysis of OpenStack cloud computing platform: Features and performance, J. Telecommun. Inf. Technol. 3 (2015) 52–57.

[30] D. Grzonka, A. Jakóbik, J. Kołodziej, S. Pllana, Using a multi-agent system and artificial intelligence for monitoring and improving the cloud performance and security, Future Gener. Comput. Syst. (2017). http://dx.doi.org/10.1016/j.future.2017.05.046.

[31] D. Grzonka, J. Kołodziej, J. Tao, Using Artificial Neural Network For Monitoring And Supporting The Grid Scheduler Performance, in: 28th European Conference on Modelling and Simulation, ECMS 2014, Brescia, Italy, May 27–30, 2014, Proceedings, 2014, pp. 515–522. http://dx.doi.org/10.7148/2014-0515.

[32] D. Grzonka, J. Kołodziej, J. Tao, S.U. Khan, Artificial neural network support to monitoring of the evolutionary driven security aware scheduling in computational distributed environments, Future Gener. Comput. Syst. 51 (2015) 72–86. http://dx.doi.org/10.1016/j.future.2014.10.031.

[33] D. Grzonka, M. Szczygiel, A. Bernasiewicz, A. Wilczyński, M. Liszka, Short analysis of implementation and resource utilization for the openstack cloud computing platform, in: 29th European Conference on Modelling and Simulation, ECMS 2015, Albena (Varna), Bulgaria, May 26–29, 2015. Proceedings, 2015, pp. 608–614. http://dx.doi.org/10.7148/2015-0608.

[34] C. Gu, H. Huang, X. Jia, Power metering for virtual machine in cloud computing-challenges and opportunities, IEEE Access 2 (2014) 1106–1116. http://dx.doi.org/10.1109/ACCESS.2014.2358992.

[35] T.W. Harton, C. Walker, M. O'Sullivan, Towards power consumption modeling for servers at scale, in: 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC), 2015, pp. 315–321. http://dx.doi.org/10.1109/UCC.2015.50.

[36] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A.D. Joseph, R.H. Katz, S. Shenker, I. Stoica, Mesos: A platform for fine-grained resource sharing in the data center, in: NSDI, vol. 11, 2011, pp. 22–22.

[37] A. Jakóbik, Big Data Security, in: F. Pop, J. Kołodziej, B. Di Martino (Eds.), Resource Management for Big Data Platforms: Algorithms, Modelling, and High-Performance Computing Techniques, Springer International Publishing, Cham, 2016, pp. 241–261. http://dx.doi.org/10.1007/978-3-319-44881-7_12.

[38] A. Jakóbik, A cloud-aided group RSA scheme in Java 8 environment and OpenStack software, J. Telecommun. Inf. Technol. JTIT (2) (2016) 53–59.

[39] A. Jakóbik, D. Grzonka, J. Kołodziej, Security supportive energy aware scheduling and scaling for cloud environments, in: European Conference on Modelling and Simulation, ECMS 2017, Budapest, Hungary, May 23–26, 2017, Proceedings, 2017, pp. 583–590. http://dx.doi.org/10.7148/2017-0583.

[40] A. Jakóbik, D. Grzonka, J. Kołodziej, A.E. Chis, H. Gonzalez-Velez, Energy efficient scheduling methods for computational grids and clouds, J. Telecommun. Inf. Technol. 1 (2017) 56–64.

[41] A. Jakóbik, D. Grzonka, J. Kołodziej, H. González-Vélez, Towards secure non-deterministic meta-scheduling for clouds, in: 30th European Conference on Modelling and Simulation, ECMS 2016, Regensburg, Germany, May 31–June 3, 2016, Proceedings, 2016, pp. 596–602. http://dx.doi.org/10.7148/2016-0596.

[42] A. Jakóbik, D. Grzonka, F. Palmieri, Non-deterministic security driven meta scheduler for distributed cloud organizations, in: High-Performance Modelling and Simulation for Big Data Applications, Simul. Model. Pract. Theory 76 (2017) 67–81. http://dx.doi.org/10.1016/j.simpat.2016.10.011.

[43] A. Jakóbik, A. Wilczyński, Using polymatrix extensive stackelberg games in security — Aware resource allocation and task scheduling in computational clouds, J. Telecommun. Inf. Technol. 1/2017 (1) (2017) 71–80.

[44] A. Kansal, F. Zhao, J. Liu, N. Kothari, A.A. Bhattacharya, Virtual machine power metering and provisioning, in: Proceedings of the 1st ACM Symposium on Cloud Computing, SoCC '10, ACM, New York, NY, USA, 2010, pp. 39–50. http://dx.doi.org/10.1145/1807128.1807136.

[45] K. Karanasos, S. Rao, C. Curino, C. Douglas, K. Chaliparambil, G.M. Fumarola, S. Heddaya, R. Ramakrishnan, S. Sakalanaga, Mercury: hybrid centralized and distributed scheduling in large shared clusters, in: USENIX Annual Technical Conference, 2015, pp. 485–497.

[46] H. Kataoka, D. Duolikun, T. Enokido, M. Takizawa, Power consumption and computation models of a server with a multi-core cpu and experiments, in: 2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops, 2015, pp. 217–222. http://dx.doi.org/10.1109/WAINA.2015.127.

[47] H. Kataoka, A. Sawada, D. Duolikun, T. Enokido, M. Takizawa, Energy-aware server selection algorithms in a scalable cluster, in: 2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA), 2016, pp. 565–572. http://dx.doi.org/10.1109/AINA.2016.154.

[48] R.T. Kaushik, M. Bhandarkar, Greenhdfs: towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster, in: Proceedings of the USENIX Annual Technical Conference, 2010, p. 109.

[49] G. Khaneja, An experimental study of monolithic scheduler architecture in cloud computing systems (Ph.D. thesis), University of Illinois at Urbana-Champaign, 2015.

[50] J.-Y. Kim, H.-J. Chang, Y.-H. Jung, K.-M. Cho, G. Augenbroe, Energy conservation effects of a multi-stage outdoor air enabled cooling system in a data center, Energy Build. 138 (2017) 257–270.

[51] J. Kołodziej, Evolutionary Hierarchical Multi-Criteria Metaheuristics for Scheduling in Large-Scale Grid Systems, Springer Publishing Company, Incorporated, 2012.

[52] J. Koomey, Growth in data center electricity use 2005 to 2010, A report by Analytical Press, completed at the request of The New York Times 9, 2011.

[53] B. Krishnan, H. Amur, A. Gavrilovska, K. Schwan, VM power metering: Feasibility and challenges, SIGMETRICS Perform. Eval. Rev. 38 (3) (2010) 56–60. http://dx.doi.org/10.1145/1925019.1925031.

[54] Y. Li, Y. Wang, B. Yin, L. Guan, An online power metering model for cloud environment, in: 2012 IEEE 11th International Symposium on Network Computing and Applications, 2012, pp. 175–180. http://dx.doi.org/10.1109/NCA.2012.10.

[55] Z. Liu, S. Cho, Characterizing machines and workloads on a Google cluster, in: 8th International Workshop on Scheduling and Resource Management for Parallel and Distributed Systems (SRMPDS), 2012, Pittsburgh, PA, USA.

[56] A. Miyoshi, C. Lefurgy, E. Van Hensbergen, R. Rajamony, R. Rajkumar, Critical power slope: understanding the runtime effects of frequency scaling, in: Proceedings of the 16th International Conference on Supercomputing, ACM, 2002, pp. 35–44.

[57] I.M. Murwantara, B. Bordbar, A Simplified Method of Measurement of Energy Consumption in Cloud and Virtualized Environment, in: Proceedings of the 2014 IEEE Fourth International Conference on Big Data and Cloud Computing, BDCLOUD '14, IEEE Computer Society, Washington, DC, USA, 2014, pp. 654–661. http://dx.doi.org/10.1109/BDCloud.2014.47.

[58] K. Ousterhout, P. Wendell, M. Zaharia, I. Stoica, Sparrow: distributed, low latency scheduling, in: Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, ACM, 2013, pp. 69–84.

[59] J. Read, What is an ECU? CPU Benchmarking in the Cloud. URL http://blog.cloudharmony.com/2010/05/what-is-ecu-cpu-benchmarking-in-cloud.html.

[60] C. Reiss, A. Tumanov, G.R. Ganger, R.H. Katz, M.A. Kozuch, Heterogeneity and dynamicity of clouds at scale: Google trace analysis, ACM Symposium on Cloud Computing (SoCC), San Jose, CA, USA, 2012.

[61] C. Reiss, J. Wilkes, J.L. Hellerstein, Google cluster-usage traces: format + schema, Technical report, Google Inc., Mountain View, CA, USA, 2011.

[62] C. Reiss, J. Wilkes, J.L. Hellerstein, Obfuscatory obscanturism: making workload traces of commercially-sensitive systems safe to release, in: 3rd International Workshop on Cloud Management, CLOUDMAN, IEEE, Maui, HI, USA, 2012, pp. 1279–1286.

[63] S. Ruth, Reducing ICT-related carbon emissions: an exemplar for global energy policy? IETE Tech. Rev. 28 (3) (2011) 207–211.

[64] M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, J. Wilkes, Omega: flexible, scalable schedulers for large compute clusters, in: Proceedings of the 8th ACM European Conference on Computer Systems, ACM, 2013, pp. 351–364.

[65] R.K. Sharma, C.E. Bash, C.D. Patel, R.J. Friedrich, J.S. Chase, Balance of power: Dynamic thermal management for internet data centers, IEEE Internet Comput. 9 (1) (2005) 42–49.

[66] W. Tu, Efficient resource utilization for multi-flow wireless multicasting transmissions, IEEE J. Sel. Areas Commun. 30 (7) (2012) 1246–1258. http://dx.doi.org/10.1109/JSAC.2012.120810.

[67] J. Ullman, NP-complete scheduling problems, J. Comput. System Sci. 10 (3) (1975) 384–393. http://dx.doi.org/10.1016/S0022-0000(75)80008-0.

[68] I. Wassmann, D. Versick, D. Tavangarian, Energy consumption estimation of virtual machines, in: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13, ACM, New York, NY, USA, 2013, pp. 1151–1156. http://dx.doi.org/10.1145/2480362.2480579.

[69] Y. Zhao, J. Wu, F. Li, S. Lu, On Maximizing the lifetime of wireless sensor networks using virtual backbone scheduling, IEEE Trans. Parallel Distrib. Syst. 23 (8) (2012) 1528–1535. http://dx.doi.org/10.1109/TPDS.2011.305.

[70] S. Zimmermann, I. Meijer, M.K. Tiwari, S. Paredes, B. Michel, D. Poulikakos, Aquasar: A hot water cooled data center with direct energy reuse, Energy 43 (1) (2012) 237–245.

Damián Fernández-Cerero received the B.E. degree and the M.Tech. degree in Software Engineering from the University of Sevilla. In 2014, he joined the Department of Computer Languages and Systems, University of Seville, as a Ph.D. student. In 2016 he was invited by at ENS-Lyon and in 2017 at Cracow University of Technology to work in saving energy solutions for cloud infrastructures. Currently he both teaches and conducts research at University of Sevilla. He has worked on several research projects supported by the Spanish government and the European Union. His research interests include energy efficiency and resource scheduling.

Agnieszka Jakóbik received her M.Sc. in the field of stochastic processes at the Jagiellonian University, Cracow, Poland and Ph.D. degree in the field of neural networks at Tadeusz Kosciuszko Cracow University of Technology, Poland, in 2003 and 2007, respectively. From 2009 she is an Assistant Professor at Faculty of Physics, Mathematics and Computer Science, Tadeusz Kosciuszko Cracow University of Technology. Her main scientific and didactic interests are focused mainly on Artificial Intelligence: Artificial Neural Networks, Genetic Algorithms, and additionally on Clouds Security, Parallel Processing and Cryptography.

Daniel Grzonka received his B.Sc. and M.Sc. degrees with distinctions in Computer Science at Cracow University of Technology, Poland, in 2012 and 2013, respectively. Currently, he is Research and Teaching Assistant at Cracow University of Technology and Ph.D. student at Polish Academy of Sciences in cooperation with Jagiellonian University. He is also a member of Polish Information Processing Society, cHiPSet IC1406 COST Action, co-chair of the HiP-MoS track of the ECMS 2016 and 2017, and IPC member of several international conferences. The main topics of his research are grid and cloud computing, multi-agent systems and high-performance computing. For more information, please visit: www.grzonka.eu.

Joanna Kołodziej is a Professor in Research and Academic Computer Network (NASK) in Warsaw. She is a vice Head of the Department for Sciences and Development in Institute of Computer Science of Cracow University of Technology. She serves also as the President of the Polish Chapter of IEEE Computational Intelligence Society. She published over 150 papers in the international journals and conference proceedings. She is also a Honorary Chair of the HiP-MoS track of ECMS. The main topics of her research are artificial Intelligence, grid and cloud computing, multiagent systems. The detailed information is available at www.joannakolodziej.org.

Alejandro Fernández-Montes received the B.E. degree, M. Tech. and International Ph.D. degrees in Software Engineering from the University of Sevilla, Spain. In 2006, he joined the Department of Computer Languages and Systems, University of Sevilla, and in 2013 became Assistant Professor. In 2008 and 2009 he was invited to the ENS-Lyon, in 2012 to the Universitat Politécnica de Barcelona and in 2016 to Shanghai Jiao Tong University for sharing experiences and knowledge in saving energy solutions for Data Centers. His research interests include energy efficiency in distributed computing, applying prediction models to balance load and applying on–off policies to Data Centers.

GAME-SCORE: Game-based energy-aware cloud scheduler and simulator for computational clouds

Once energy-aware scheduling models and energy policies were analyzed in monolithic environments, we worked on the fifth research objective of this thesis dissertation as part of my second research stage in Cracow: *"Proof that models based on games theory, such as the Stackelberg model, are an excellent choice to successfully model the concurrency between data-center subsystems with opposite needs, and that this model can be used for the dynamic application of resource-efficiency policies"*. Optimization of the energy consumed in cloud computational clusters and computing servers is usually related to scheduling problems. It is very difficult to define an optimal scheduling policy without negative influence into the system performance and task completion time. In this work, we present an extension of the previously published simulation tool for Cloud Computing, GAME-SCORE, which implements the scheduling model based on a Stackelberg game with the workload scheduler and energy-efficiency agent as the main players in that game. We used the GAME-SCORE simulator for the analysis of the efficiency of the proposed game-based scheduling model. The obtained results show that Stackelberg cloud scheduler is better than static energy-optimization strategies and may achieve a fair balance between low energy consumption and makespan in a very short time.

The main contributions of this work include: a) An energy-efficient model based on Stackelberg Game which balances the trade-offs of any energy-aware cluster: energy efficiency and performance, through the dynamic application of shut-down policies; and b) A simulation tool called GAME-SCORE which implements this model.

This work was published in *Simulation Modelling Practice and Theory*. This Journal is indexed in JCR with an **Impact Factor of 2.092**. The Journal stands in ranking **Q1** in Computer Science, Software Engineering (21/104).

Simulation Modelling Practice and Theory xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

GAME-SCORE: Game-based energy-aware cloud scheduler and simulator for computational clouds

Damian Fernández-Cerero*,[a], Agnieszka Jakóbik[b], Alejandro Fernández-Montes[a], Joanna Kołodziej[b]

[a] Department of Computer Languages and Systems, University of Seville, Av. Reina Mercedes S/N, Seville, Spain
[b] Department of Computer Science, Cracow University of Technology, Poland

ARTICLE INFO

Keywords:
Computational cluster
Energy saving
Cloud computing
Energy-aware scheduling
Stackelberg game

ABSTRACT

Energy-awareness remains one of the main concerns for today's cloud computing (CC) operators. The optimisation of energy consumption in both cloud computational clusters and computing servers is usually related to scheduling problems. The definition of an optimal scheduling policy which does not negatively impact to system performance and task completion time is still challenging. In this work, we present a new simulation tool for cloud computing, GAME-SCORE, which implements a scheduling model based on the Stackelberg game. This game presents two main players: a) the scheduler and b) the energy-efficiency agent. We used the GAME-SCORE simulator to analyse the efficiency of the proposed game-based scheduling model. The obtained results show that the Stackelberg cloud scheduler performs better than static energy-optimisation strategies and can achieve a fair balance between low energy consumption and short makespan in a very short time.

1. Introduction

New paradigms, such as cloud computing, and the ever-growing web applications and services, have imposed new challenges to traditional high-performance computing (HPC) systems. In the same time, computational clouds that provide the core foundation for the parallel computing solutions have grown drastically in recent years to satisfy the ever-evolving user requirements. Modern large-scale cloud computing systems are composed of thousands of computational distributed servers. The energy consumed by such cloud computing systems may be compared to the energy utilized by small towns and large factories. computational clusters account for more than 1.5% of global energy consumption [36].

Several hardware and infrastructure models have been recently developed for the successful reduction of the energy consumption in real-life large-scale computational clusters. The most popular models and technologies include:

(a) cooling and temperature management [17,45];
(b) memory and CPU power proportionality [18,39];
(c) construction of energy-efficient flash hard disks [2]; and
(d) new models in energy transportation [19].

* Corresponding author. Tel.: +655793390.
E-mail addresses: damiancerero@us.es (D. Fernández-Cerero), akrok@pk.edu.pl (A. Jakóbik), afdez@us.es (A. Fernández-Montes), jokolodziej@pk.edu.pl (J. Kołodziej).

https://doi.org/10.1016/j.simpat.2018.09.001
Received 24 June 2018; Received in revised form 28 August 2018; Accepted 1 September 2018

Also the resource management and scheduling models in clouds are defined with the energy optimization modules. Energy utilization policies may be based on various power related physical models, however the most popular scenario is to switch off idle servers. Although such power-off strategy is commonly used in small-area grids and clusters [42], in realistic CC systems, the existing power-off models need to be improved especially in the case of dynamical changes in the task workloads and cloud resource infrastructure [24]. It is also important to point out that decisions taken in the organizations might affect both positively and negatively to different parts of the systems [43]

In this work, the balance between two opposed needs of the data-center environment is modelled by means of a Stackelberg Game (SG) as an extension of a previous work presented in [22], where we presented the theoretical model for a game-based energy-aware scheduling algorithm and an algorithm for the dynamic selection of energy-efficiency policies. On one hand, the performance side, represented by the *Scheduling Manager*, which wants tasks to be processed as fast as possible, while the efficiency side (CC provider), represented by the *Energy-Efficiency Manager*, wants the minimization of the energy consumption of the computational cluster.

In our SG model, the *Scheduling Manager*, that is the leader of the game, processes firstly every task to make its decision (move). Once the particular *Task* is processed by the leader, then the follower, that is the *Energy-efficiency Manager*, handles it to make its move. This competition process is implemented in a trustworthy simulation tool focused on simulating realistic large-scale computational-cluster scenarios. Our contributions in this paper include:

(a) An energy-efficient model based on Stackelberg Game which balances the trade-offs of any energy-aware cluster: energy efficiency and performance, through the dynamic application of shut-down policies; and
(b) A simulation tool called GAME-SCORE which implements this model. The presented tool is able to simulate large-cluster environments that supports popular cloud computing services.

The paper is organized as follows. In Section 2 we present a simple analysis on the most relevant energy-aware cluster simulators, as well as a brief description of the main models for energy efficiency in CC systems. In Section 3, we present the developed simulation tool, GAME-SCORE, as well as the formal definition of the implemented Stackelberg Game model for the balance between energy consumption and performance in CC systems. This model theoretical core of this work. Due to this, we also present simple theoretical example of its utilization in this section. The experimental environment, analysis and results obtained are presented in Section 4. Finally, the conclusions and future work are discussed in Section 5

2. Related work

Many efforts have been made in order to increase resource and energy efficiency in computational clouds. Most of the energy is consumed in computational clusters, where the data necessary for the computation is stored. However, the optimization of the scheduling procedures may significantly reduce the time of keeping the requested data ready for use. The data records may be archived when the computation is complete or simply removed from the computational cluster based on the specific end users requests. In this section, we first survey in Section 2.1 the most popular cloud simulators for large-scale clusters, with a special focus on the energy awareness. We present then a simple comparison of the evaluated simulators and critical analysis for better motivation of our work. In in Section 2.2, we define a simple taxonomy of the energy-aware cloud schedulers and survey the classes of models considered in the experimental evaluation presented in this paper.

2.1. Simulation tools for energy-aware cloud scheduling

Cloud simulators are still the main virtual environments for evaluation of the new models of cloud services and schedulers. In this section, we evaluate the following most relevant cloud-computing simulators for the implementation and evaluation of energy-efficiency techniques.

GreenCloud [34] is the extension of the NS2 network simulator. Its purpose is to measure and compute the energy consumption at every computational cluster level, and it pays special attention to network components. However, its packet-level nature compromises performance in order to raise the level of detail, which may be not optimal for the simulation of large computational clusters. In addition, it is not designed to offer ease of development and extension in various scheduling models.

CloudSim is based on SimJava and GridSim and is mainly focused on IaaS-related operation environment features [8]. It presents a high level of detail, and therefore allows several VM allocation and migration policies to be defined, networking to be considered, features and energy consumption to be taken into account. However, it features certain disadvantages when applied for the simulation of large data-center environments: CloudSim is considered cumbersome to execute for large scale scenarios, as well as being closely bound to only monolithic scheduling models.

CloudReports is a highly extensible simulation tool for energy-aware Cloud Computing environments. The tool uses the CloudSim toolkit as its simulation engine and provides features such as a graphic user interface, reports generation, simulation data exportation, power utilization models and an API for easily extend the tool [49]. However, as CloudReports is based on CloudSim, it fails to achieve good performance levels when it comes to large-scale data-center infrastructures, as well as not providing easy and out-of-the-box energy-efficiency strategies based on the shut-down of idle nodes for several scheduling frameworks.

CloudAnalyst adds visual modelling and simulation of large-scale applications that are deployed on Cloud Infrastructures to CloudSim. Several configuration parameters may be set at a high level by using this GUI. However, CloudAnalyst keeps the performance limitations of CloudSim. In addition, this simulator does not provide power-consumption measures out of the box, as it is focused on the visual modelling of data-center infrastructures, not in energy-awareness [53].

Omnet + + [51] is focused on modelling communication networks (mainly), multiprocessors and other distributed or parallel systems. OMNeT + + is public-source discrete-event simulation tool which has been used as the core simulation engine to test several energy-efficiency techniques in computational clusters . However, this simulator does not provide ready-to-use tools for the measurement and implementation of energy-aware algorithms, and the main focus on modelling networking may make some scheduling-related implementation complicated.

GDCSimGreen [27] focuses on the simulation of the physical behaviour of a computational cluster. This implies the evaluation of energy efficiency of computational clusters under various workload characteristics, platform power management schemes, and scheduling algorithms. However, the low-level hardware and cooling characteristics simulated by this tool make it sub-optimal for large-scale computational clusters due to performance issues. In addition, this simulator does not provide easy and out-of-the-box tools for the simulation of energy-efficiency policies based on the shut-down of the idle machines.

Grid'5000 Toolbox simulates the behaviour of Grid'5000 (France) resources for real workloads while changing the state of the resources according to several energy policies. The simulator includes:

(a) A GUI that allows the user to simulate a set energy policies for each location of Grid'5000;
(b) A graphical visualisation of the state of the resources during the simulation;
(c) A graphical view of the results.

On the other hand, the simulator fails to include various scheduling frameworks and it does not simulate the behaviour or consumption of network devices and resources.

CoolEmAll [11] is focused on evaluation the thermal side of the data-center operation in order to achieve energy-efficiency by combining the optimization of IT, cooling and workload management. On the other hand, the support for testing energy-aware scheduling algorithms as well as workload consolidation and other strategies based on the shut-down of idle nodes is not extensively covered. In addition, this simulator is not designed to achieve high performance when large-scale cluster are considered.

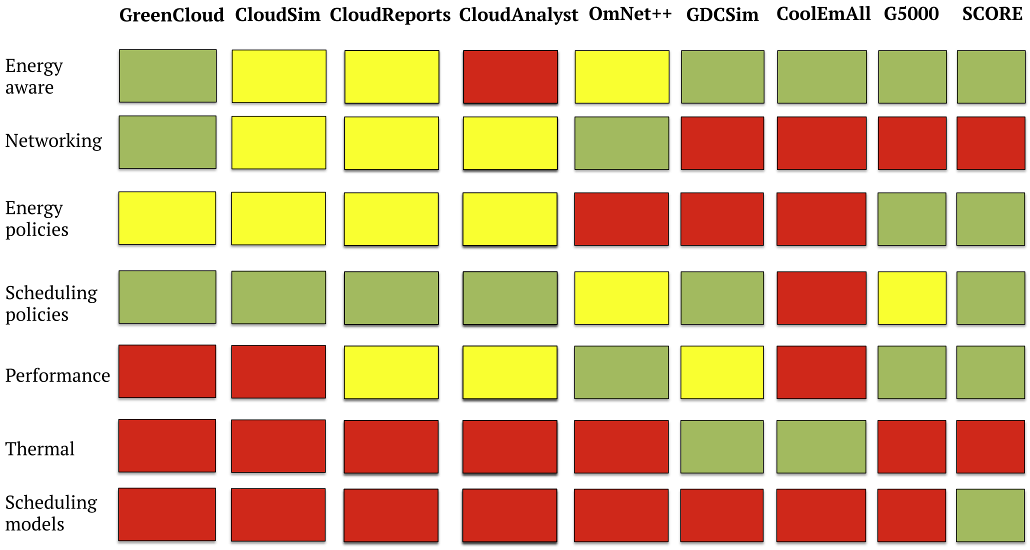
SCORE [20] is designed for the comparison of various scheduling framework in large clusters. To this end, it focuses on maximizing the performance of the simulations by reducing the level of detail . The simulation tool enables the modelling of heterogeneous, realistic and synthetic workloads, as well as it provides the tools to easily develop both new energy-aware scheduling policies for various scheduling frameworks and energy-efficiency policies based on the shut-down of idle nodes.

To summarize, the most widely adopted simulation tools for cloud computing clusters lack some critical features, as shown in Figure 1, that motivated us to develop ENERGY-SCORE based on the SCORE simulator, mainly:

(a) the capacity to dynamically switch between energy-efficiency and scheduling strategies; and
(b) the performance required to run large-scale, heterogeneous and realistic experimentation in a reasonable time.

2.2. Taxonomy of energy-efficiency techniques

In Fig. 2, we present a simple taxonomy of the main techniques to improve energy-efficiency in cluster scheduling.



	GreenCloud	CloudSim	CloudReports	CloudAnalyst	OmNet++	GDCSim	CoolEmAll	G5000	SCORE
Energy aware	🟩	🟨	🟨	🟥	🟨	🟩	🟩	🟩	🟩
Networking	🟩	🟨	🟨	🟨	🟩	🟥	🟥	🟥	🟥
Energy policies	🟨	🟨	🟨	🟨	🟥	🟥	🟥	🟩	🟩
Scheduling policies	🟩	🟩	🟩	🟩	🟨	🟩	🟥	🟨	🟩
Performance	🟥	🟥	🟨	🟨	🟩	🟨	🟥	🟩	🟩
Thermal	🟥	🟥	🟥	🟥	🟥	🟩	🟩	🟥	🟥
Scheduling models	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟩

Fig. 1. Brief comparison between available simulation tools. Green parameters mean that either the characteristic is high or the easiness to implement that characteristic is high. The same applies for medium and low (yellow and red respectively).

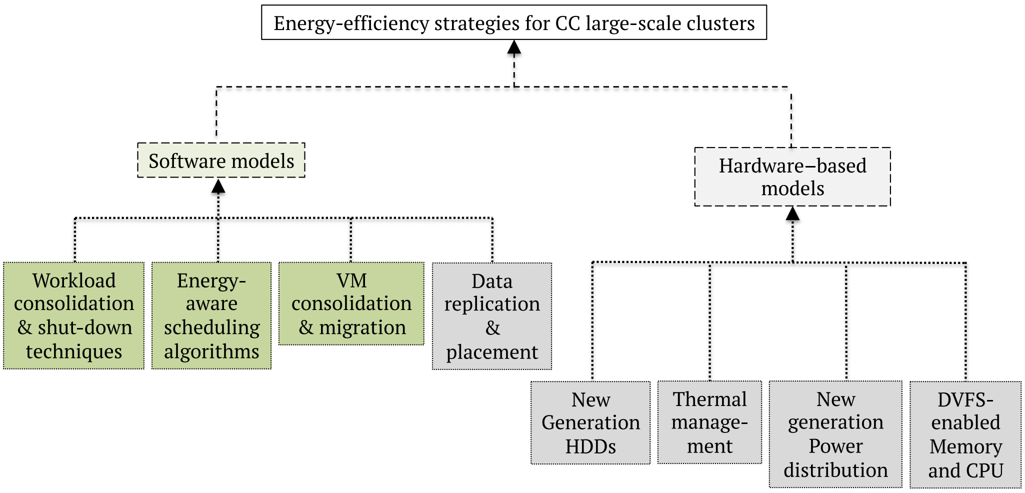

Energy-efficiency strategies for CC large-scale clusters
Software models
Hardware–based models
Workload consolidation & shut-down techniques
Energy-aware scheduling algorithms
VM consolidation & migration
Data replication & placement
New Generation HDDs
Thermal manage-ment
New generation Power distribution
DVFS-enabled Memory and CPU

Fig. 2. Generic taxonomy of energy-efficiency techniques for clusters. The strategies in studied in this paper are marked in green.

There are two main categories of schedulers defined in that taxonomy, namely software and hardware-based models. The former focus on the improvement of several pieces of the physical infrastructure of the cloud, such as cooling equipment and thermal management, power distribution and hardware. The latter, on the other hand, focuses on the development of software strategies which make an smarter use of the physical computational nodes.

In this paper, we study the following three classes of software-based models:

(a) Algorithms focused on the consolidation of workload and shut-down of idle nodes.
(b) Energy-aware scheduling algorithms; and
(c) VM scaling and migration algorithms;

These three classes have been chosen since they are the core of resource-managing systems. Resource managers are the responsible for the allocation and execution of tasks, the main goal of computational clusters. For the very same reasons, data replication and placement models have not been covered, since they only affect to distributed file systems.

The first considered category contains the schedulers defined for the reduction of energy consumption in computational clusters, which manly focus on the consolidation of the workload. That consolidation is necessary for the proper estimation of the number of idle nodes in the physical cloud clusters and switch them into sleeping mode [7,21,23]. In [7], Berral et al. propose a consolidation model that combines

(a) the shut-down of idle servers;
(b) power-aware workload-consolidation algorithms; and
(c) machine-learning techniques

to improve energy-efficiency in computational clusters. The computational cluster with 400 nodes is simulated using the OmNet+ + simulator. The results obtained show that about 10% of energy consumption can be reduced without negatively impacting on SLAs. In [21] the authors developed the static power-efficiency policies based on the idea of deactivation of the idle nodes in the realistic environments. The SCORE simulator [20] was defined and used for simulation the the large-scale computational clusters with 5000 machines and for heterogeneous workloads. 20% energy reduction results are shown without notably impacting on data-center performance. The SCORE simulator was also used for evaluation of the energy-aware scheduling strategies based on the genetic algorithms with additional scheduling criteria such as security requirements defined by the end users [23]. We defined a realistic cloud environment with 1000-nodes computational nodes and executed the realistic heterogeneous workloads. The implemented scheduling model allowed to save up to 45% of the consumed energy.

A substantial part of the efforts on improving energy-efficiency has been directed towards energy-aware scheduling strategies that could lead to powering off idle nodes, such as [29,33,37]. In [37], Lee et al. present two energy-aware task consolidation heuristics. These strategies aim to maximize resource utilization in order to minimize the wasted energy used by idle resources. To this end, these algorithms compute the total cpu time consumed by the tasks and prevent a task being executed alone. In [33], Juarez et al. propose an algorithm that minimizes a multi-objective function which takes into account the energy-consumption and execution time by combining a set of heuristic rules and a resource allocation technique. This algorithm is evaluated by simulating DAG-based workloads, and energy-savings in the range of [20-30%] are shown. In [29], Jakóbik et al. propose energy-aware scheduling policies and methods based on Dynamic Voltage and Frequency Scaling (DVFS) for scaling the virtual resources while performing security-aware scheduling decisions.

In addition, different techniques of energy conservation such as VM consolidation and migration [4–6,48] are also proposed. In [5], Beloglazov et al. describe a resource management system for virtualized cloud computational clusters that aims to lower the energy consumption by applying a set of VM allocation and migration policies in terms of current CPU usage. This work is extended by focusing on SLAs restrictions in [4] and by developing and comparing various adaptive heuristics for dynamic consolidation of VMs in terms of resource usage in [6]. These migration policies are evaluated by simulating a 100-node cluster. Energy reductions up to approximately 80% are shown with low impact on quality of service and SLAs. In [48] a Bayesian Belief Network-based algorithm that aims to allocate and migrate VMs is presented. This algorithm uses the data gathered during the execution of the tasks in addition to the information provided at submission time in order to decide which of the virtual machines are to be migrated when a node is overloaded.

In this work we present a different approach to the energy-aware scheduling problem: we model the concurrency between energy efficiency and performance as rival players in a Stackelberg-Game model. This game-based model enables us to balance optimally the trade-off between these two opposite needs in energy-aware computational clusters.

The main simulator characteristics necessary to implement the Stackelberg-Game model are the following:

(a) The simulation tool must be energy-aware and provide the tools to measure the energy consumption and reduction;
(b) The simulation tool must provide several already-implemented scheduling models;
(c) The simulation tool must provide several already-implemented scheduling algorithms;
(d) The simulation tool must provide several already-implemented energy-efficiency policies based on the shut-down of idle nodes; and
(e) The simulation tool must be performant when large-scale computational clusters (thousands or even tens of thousands of nodes) are evaluated

. As presented in this Section, the SCORE simulator fulfills the majority of requirements. In this work we extend the SCORE simulator in order to implement the Stackelberg-Game model.

Differently to most of the studied strategies and simulation tools which implement them, which rely on static scheduling strategies for the consecution of energy efficiency, we model the trade-offs of energy-efficient computational clusters, that are performance and energy efficiency, as the sides of this game. The application of the proposed model results on the balance between fast and reliable task execution and low energy consumption. Hence, the major contributions of this work include a model for the dynamic application of energy-efficiency policies based on the Stackelberg-Game model, as well as a trustworthy simulation tool, GAME-SCORE, that implements this model.

3. GAME-SCORE simulator

In this section we define the GAME-SCORE simulator, which is the extension of SCORE [20] simulator, following the same design pattern: a hybrid approach between discrete-event and multi-agent simulation. The main aim of the GAME-SCORE is the simulation of energy-efficient IaaS of the clouds. However, this simulation tool has one limitation that may have a negative impact for the reduction of the energy consumption in computational clusters: the application of energy policies is made statically. Hence, only one static energy policy is applied at the beginning of the experiment and cannot be changed in runtime. This makes the simulation tool sub-optimal for realistic heterogeneous workloads and changing operation environments such as those present in Cloud-Computing scenarios.

In the following sections, we describe in detail the developed simulation tool and its modules. This simulation tool enables us to dynamically choose between a catalog of energy-efficiency policies that shut-down idle machines in runtime. In addition we present, as a realistic use case, an algorithm based on the Stackelberg Game which makes use of this feature. However, any other strategies aiming to dynamically switch between a set of energy-efficiency policies and/or scheduling algorithms could be easily implemented with this new simulation tool.

3.1. GAME-SCORE components

The GAME-SCORE base architecture is composed of two main modules:

- **Core Simulator Module**, responsible for executing the experiments, and is composed of three submodules:
  - **Workload generation**, responsible for the generation of the synthetic workloads, either based on statistical distributions or on real-life workload traces.
  - **Core engine**, responsible for the creation of the simulation environment, cluster, and the multiple scheduling agents. This engine is the module which actually runs the simulation.
  - **Scheduling module**, responsible for the assignation of tasks to worker nodes, as well for the implementation of several scheduling framework models.
- **Energy-Efficiency Module**, responsible for the implementation of the energy-efficiency policies based on the shut-down of idle machines.

In addition, we extended this base architecture to implement the Stackelberg Game process as means to dynamically switch

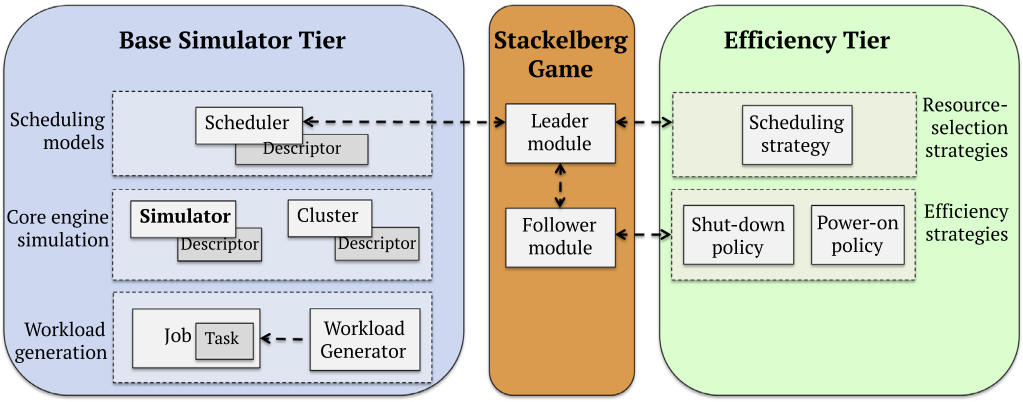


Fig. 3. GAME-SCORE architecture

between energy-efficiency policies. To this aim, we developed the following modules which negotiate between the *Scheduling* module and the *Energy-Efficiency* module:

(a) a Central Energy-efficiency Manager module which governs the catalog of energy-efficiency policies;
(b) a Stackelberg-Game manager which implements the concurrency-based model;
(c) a module for the Leader player to apply its decisions; and
(d) a module for the Follower player to apply its decisions;

. The resulting architecture of the proposed simulator is shown in Figure 3. The GAME-SCORE source code is publicly available at: https://github.com/DamianUS/game-score.

3.2. Shut-down decision policies

We assume in our model that the energy conservation policies do not have a notable negative impact on the performance of the whole computational cluster Therefore we define in our model a *Central Energy-efficiency Manager* that decides the power-off strategy to be applied, which deactivates the servers in an idle mode. It should be noted that *Always* strategy cannot be kept active when a machine computes tasks and send/receive data. In the case of huge workloads, where tasks and data may leave and arrive dynamically from and to the cloud servers, the active servers may be overloaded and the whole task execution process can be significantly delayed. Therefore, there is a need of the development of the decision model which allows us to activate the *Always* power-off strategy in the optimal periods. The following shut-down decision policies have been implemented in our model:

- **Margin** – this decision strategy activates the *Always* power-off strategy only if, at least, a specified amount of resources (servers) is ready to accept the incoming tasks.
- **Random**– in this case, the *Always* power-off strategy is activated randomly. This strategy is usually defined together with the *Never* shut-down policy, where all servers are kept in the active mode (it happens usually in realistic cloud computational clusters) and the *Always* shut-down scenario, where all idle machines are switched-off.
- **Exponential** – in this strategy, the *Always* shut-down strategy is activated depending on the probability of one (large) incoming task of oversubscribing the available resources. This probability is computed by the means of the Exponential distribution.
- **Gamma** – in this case, the *Always* shut-down strategy is activated depending on the probability of incoming tasks (in a given window time) of oversubscribing the available resources. This probability is computed by the means of the Gamma distribution.

The utilization of the *Energy-efficiency Manager* in our model does not guarantee the fair reduction of the energy consumed by the cloud system. Therefore, we define another component of the model, that is the *Scheduling Manager*. This component allows the optimal schedule of tasks onto the cloud servers based on the energy-conservation criterion. In this work, we focus on the problem of the independent tasks scheduling. We use the genetic cloud scheduler developed in [31] and ETC Matrix scheduling model described in [29]. The makespan constitutes the most representative parameter of the performance, and hence it becomes the scheduling goal.

3.3. Stackelberg Game used for scheduling decisions

In the model presented in this work we used the Stackelberg Game played by two opponents *Scheduling Manager* – Leader and Energy-efficiency Manager –Follower. Similar strategies have been used for energy-aware resource allocation [3]. The interaction between the previously described modules in this game is shown as a sequence diagram in Figure 4

As a model for balancing scheduling efficiency and energy minimization we used a non-zero symmetric game, defined by:

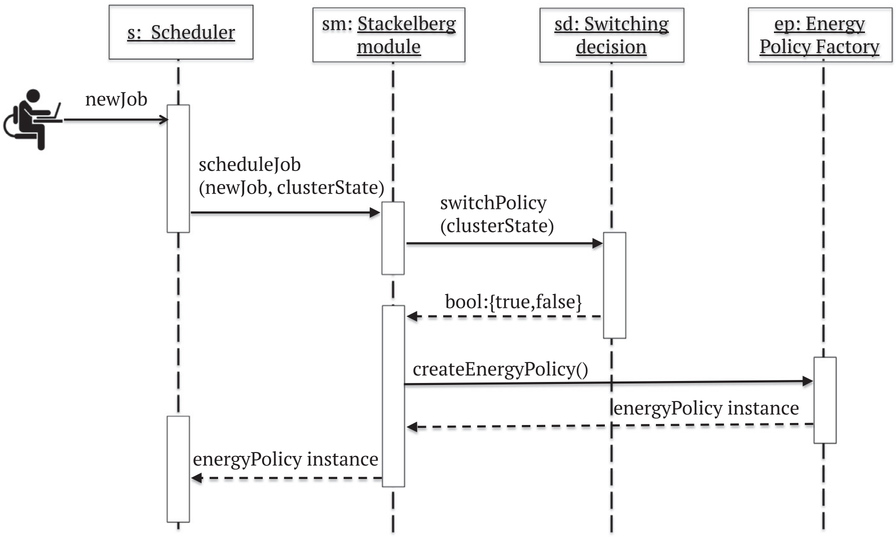


Fig. 4. Sequence diagram of the interactions between modules of the Stackelberg Game workflow

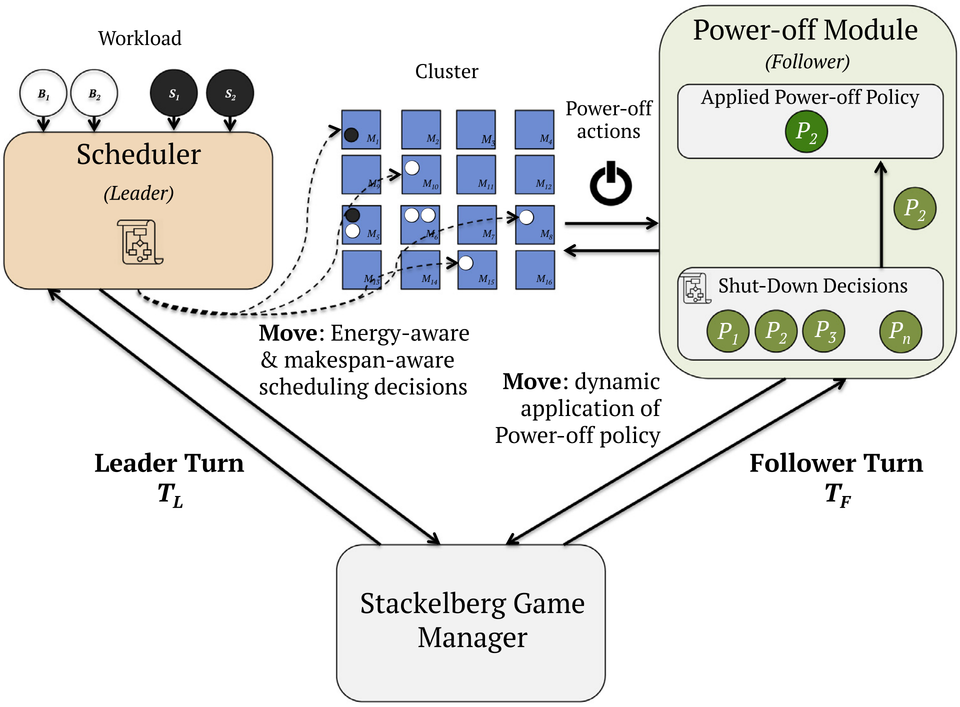


Fig. 5. Stackelberg Game workflow, scheduling workflow, *B* - Batch type task, *S* - Service type task, *M* - Virtual Machine

$$\Gamma_n = ((N, \{S_i\}_{i \in N}, \{Q_i\}_{i \in N}) \tag{1}$$

where $N = \{1, 2\}$ denotes the set of players, $\{S_1, S_2\}$ $(cardS_i \geq 2; i = 1, 2)$ denotes the set of strategies for them $\{H_1, H_2\}$; H_i: $S_1 \times \times S_2 \rightarrow \mathbb{R}$; $\forall_{i=1,2}$ denotes payoff functions for each player.

Both players are making decisions according their payoffs. A decision is a selection of one single action from the set of possible actions. Possible actions are defines as elements of strategy sets f$\{S_1, S_2\}$. The sets of actions for each player are chosen to be beneficial for this player. The payoff function is measuring the quality of actions by assigning the real value to each set of decisions. In the model pure strategies and mixed strategies are considered, see [54]. Let us denote by s_i the **Pure strategy** of the player i and the set of all pure strategies specified for player i is denoted by S_i. The **mixed strategy of the player** i is denoted by $\sigma_i \in S_i \subset \Delta S_i$ and allows to randomize over pure strategies:

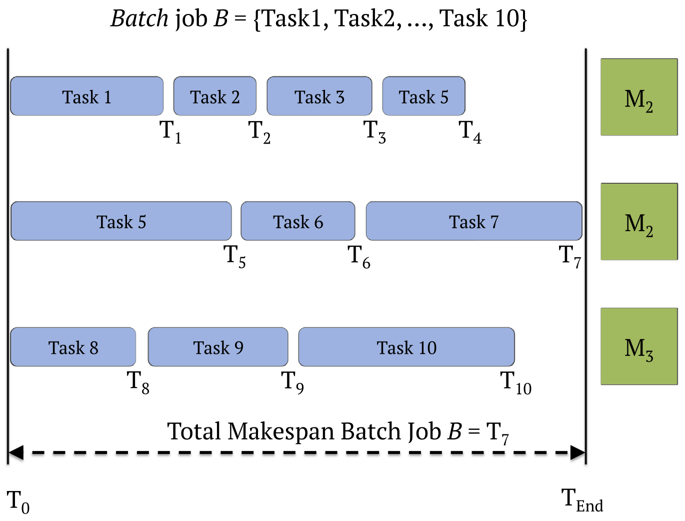


Fig. 6. Example of Leader player (Scheduler) makespan computation

$$\sigma_i = \{\sigma_i(s_{i_1}), \sigma_i(s_{i_2}), \ldots, \sigma_i(s_{i_m})\}, \tag{2}$$

where $\sigma_i(s_i)$ denotes the probability that the player i choses the pure strategy s_i.

In Stakelberg Games (SG), the leader of the game is privileged to play first, and the second players (the follower) are obliged to make their decisions after him [54]. In our model we proposed o non zero sum game, to allow the leader and the follower define their strategies separately.

The leader of the game, *Scheduler* component is making decisions how to dispatch tasks among the *Computing Nodes*. These *Computing Nodes* are grouped into *Computational Units*, denoted as CU_1, CU_2, $\ldots CU_P$. Incoming *Jobs* are composed of a set of independent *Tasks* which can be executed in parallel.

Single decision of the leader is a schedule calculated for the given batch of tasks and available set of *Computational Units*. The cardinality of the strategy set for the leader equals all possible schedules. Let us denote this number by P possible decisions. The strategy vector $\sigma_i(s_i)$ represents the probability for a *Job* to be assigned to the CU_p, for $p = 1, 2, \ldots, P$. The s_i may be taken from the set $1, 2, \ldots, P$.

The expected payoff of the game leader is depends on the completion time of all the *Tasks* in the scheduled *Job*, thus, the makespan of that *Job*, as shown in Figure 6. The leader plays to in order to minimize the makespan. In our model we used a Monolithic Scheduler [35] which makes scheduling decisions based on the Expected Time to Compute (ETC) matrix, defined as follows:

$$ETC = [ETC[j][i]]_{j=1,\ldots,n}^{i=1,\ldots,m^p} \tag{3}$$

where

$$ETC[j][i] = wl_j/cc_i^p \tag{4}$$

In this equation, cc_i^p is the computational capacity of the i-th Computing Node (CN) in the pth Computing Unit (CU) in Giga Flops per Second (GFLOPS) and wl_j represents the workload of j-th task in Flops (FLO); n and m^p denote the number of tasks and number of Computing Nodes in the pth Computing Unit respectively, see [32]. The makespan is defined as follows

$$C_{\max}(wl_1, \ldots, wl_n, cc_1^p, \ldots, cc_{m^p}^p, m^p, n, p) = \tag{5}$$

$$= \min_{S \in Schedules} \left\{ \max_{j \in Tasks} C_j \right\}, \tag{6}$$

where C_j is the completion time of the j-th task. *Tasks* represents the set of tasks in the *Job*, and *Schedules* is the set of all possible schedules that can be generated for the *Tasks* of that *Job*. The shortest makespan is calculated by using the Expected Time to Compute (ETC) matrix. In this matrix, the cell in the ith row and the jth column represents the completion time of the jth task if is it executed on the ith CN.

The optimal schedule is is the best decision for the game leader, therefore utility function value for the game leader is defined as:

$$H_1(\sigma_1, \sigma_2) = \sum_{p=1,\ldots,P} \sum_{l=1,\ldots,L} \sigma_1^p \sigma_2^l C_{\max}(wl_1, \ldots, wl_n, cc_1^p, \ldots, cc_{m^p}^p, m^p, n, p) \tag{7}$$

where L indicates the number of decisions that may be taken by the game follower. The value of the makespan depends on the computational power of the CNs. These parameters may be modified by the follower player:

$$\begin{aligned}&H_1(\sigma_1, \sigma_2)=\\&\textstyle\sum_{p=1,\dots,P}\sum_{l=1,..,L}\\&\sigma_1^p\sigma_2^l C_{\max}(wl_1, \dots,wl_n, cc_1^p(l), \dots,cc_{m^p}^{p}(l), m^p(l), n, p(l))\end{aligned}\tag{8}$$

The follower in the game is the *Central Energy-efficiency Manager*. It applies energy policies to all the CNs in the computational cluster. The follower may decide about the $cc_i^p(l)$ and the $m^p(l)$. The payoff for the follower is defined as the energy consumed by the CC system for the execution of the schedule computed by the *Scheduler*:

$$\begin{aligned}&H_2(\sigma_1, \sigma_2)=\\&\textstyle\sum_{i=1,\dots,m}\sum_{j=1,..,n}\\&\sigma_1^j\sigma_2^i E(wl_1, \dots,wl_n, cc_1^p, \dots,cc_{m^p}^{p}, m^p, n, p, schedule)\end{aligned}\tag{9}$$

After the follower has made his decision, the leader is considering new batch of tasks. It computes next schedule and the game is repeated. Both the payoffs depends on both players moves.

In order to calculate the the follower payoff, the following equation was introduced:

E_{total} is the total energy consumed by particular *Job*, t_{idle}^i is the idle time of CU after it calculated assigned tasks ; t_{busy}^i is the time that the *i*-th CN is devoting on computing tasks; P_{idle}^i is the power a CN requires to remain in a idle state; P_{busy}^i is the power a CN consumes during computing tasks.

The time that the *i*-th CN spends on computing tasks depends on the schedule that was decided be the game leader:

$$t_{busy}^i = max_{j\in Tasks\ scheduled\ for\ CN_i} C_j\tag{10}$$

and the idle time of the *i*-th CN may be calculated as follows:

$$t_{idle}^i = C_{max} - t_{busy}^i\tag{11}$$

This model assumes that the next batch of tasks may be scheduled is the previous batch was calculated. Assuming that, the total energy consumed may be calculated in the following way:

$$\begin{aligned}&E_{total} = \textstyle\sum_{i=1}^m \displaystyle\int_0^{C_{max}} Pow_{CN_i}(t)dt=\\&\textstyle\sum_{i=1}^m (P_{idle}^i{*}t_{idle}^i + P_{busy}^i{*}t_{busy}^i + P_{sleep}^i{*}t_{sleep}^i + P_{off}^i{*}t_{off}^i + P_t^i{*}t_t^i)\end{aligned}\tag{12}$$

where P_{sleep}^i, t_{sleep}^i is power consumed during sleeping mode, and time spend in this state, P_{off}^i, t_{off}^i is during power consumed during being powered of (assumed as zero) and time spend in this state. Values P_t^i and t_t^i are accumulated power and time spend during all transitions from one state to another.

Our model allows competition between two aims: to compute tasks as fast as possible and to apply the more optimal power states to the CNs in order to maximize the energy efficiency. q The core off the game is to solve two optimization problems. First is to find the the best decision for the game leader, that is finding the solution of the problem

$$\begin{aligned}&argmax_{\sigma_1^1,\sigma_1^2,\dots,\sigma_1^P}\textstyle\sum_{p=1,\dots,P}\sum_{l=1,..,L}\\&\sigma_1^p\sigma_2^l C_{\max}(wl_1, \dots,wl_n, cc_1^p, \dots,cc_{m^p}^{p}, m^p, n, p)\end{aligned}\tag{13}$$

with the following constraints

$$\textstyle\sum_{p=1,\dots,P}\sigma_1^p = 1\tag{14}$$

$$\forall\ \sigma_1^p\colon \sigma_1^p \in [0, 1]\tag{15}$$

and second, to find the best decision for the game follower:

$$\begin{aligned}&argmax_{\sigma_2^1,\dots,\sigma_2^m}\textstyle\sum_{i=1,\dots,m}\sum_{j=1,..,n}\\&\sigma_1^j\sigma_2^i E(wl_1, \dots,wl_n, cc_1^p, \dots,cc_{m^p}^{p}, m^p, n, p, schedule)\end{aligned}\tag{16}$$

with the following constraints

$$\textstyle\sum_{i=1,\dots,m}\sigma_2^i = 1\tag{17}$$

$$\forall\ \sigma_2^i\colon \sigma_2^i \in [0, 1]\tag{18}$$

Considering pure strategies problem (16)-(18) may be solved by the direct search.The best strategy is one of the *m* possible problem solutions. The number of possible solutions for the problem (13)-(15) is P! and the problem of finding the best schedule is consider to be NP-hard. Therefore we applied Genetic Algorithm search method for finding suboptimal solution. During such procedure the suboptimal schedule is found and the strategy vector equals one for that schedule. Probabilities of all other schedules are assumed to be equal zero. If several sub-optimums were found each is granted the same probability that equals one divided by the number of them.

Considering mixed strategies optimization problem (16)-(18) is may be solved by one of the classical techniques, for example the simplex method. The space of the *argmax* search is $[0, 1]^m$. Mixed strategies for the problem (16)-(18) was not implemented as the part of this research.

3.4. A simple theoretical example of the proposed algorithm

For the illustration of the game implemented in the GAME-SCORE simulator, lets consider the simplest possible environment, consisting in only two computing units. Each unit is equipped with only one node. The computing capacities of the CUs are: CU_1 $cc_1^1 = 1$ GFLOPS/sec. and CU_2 $cc_1^2 = 1$ GFLOPS/sec.

The Jobs that will be considered by the scheduler are composed of three independent tasks, having the following workload: $wl_1 = 1$ FLO, $wl_1 = 2$ FLO and $wl_1 = 4$ FLO.

The Energy-Efficiency manager may choose only from two energy policies: keeping all unused CUs into idle state (strategy 1), and always switch all idle CUs to sleep mode (strategy 2). For the clearance of the presentation the transition time and energy are omitted and the rest of characteristics are assumed in the following form:

$$\begin{array}{ll} P_{idle}^1 = 2\text{ MWh} & P_{idle}^2 = 4\text{ MWh} \\ P_{busy}^1 = 10\text{ MWh} & P_{busy}^2 = 20\text{ MWh} \\ P_{sleep}^1 = 0.1\text{ MWh} & P_{sleep}^2 = 0.2\text{ MWh} \end{array}$$

In the previous round, the result of the strategy computed by the *Follower* player was: $\sigma_2^1 = 1/3$, $\sigma_2^2 = 2/3$. Now it is the *Leader*'s turn.

The algorithm for the application of the next Stackelberg game round is composed of the following steps:

1. Computation of the *Leader*'s move. This step is, in turn, composed of the following substeps:
   (a) Computation *ETC* matrix, 4, based on the characteristics of the incoming *Job*:

$$ETC = \begin{bmatrix} 1/1 & 2/1 & 4/1 \\ 1/2 & 2/2 & 4/2 \end{bmatrix}$$

   (b) Find possible schedules. In this simple example the utilization of a genetic algorithm to solve the NP-hard problem is not necessary. The optimal schedule may by computed by means of brute force. We will represent the schedules as follows: (tasks assigned to CU_1|tasks assigned to CU_2). All possible schedules are the following.:
      (1) $s_1 = (1, 2, 4| -)$ and $s_2 = (-|1, 2, 4)$. This schedule represents the mapping of all tasks to the selected *CU*
      (2) $s_3 = (1, 2|4)$, $s_4 = (1, 4|2)$, $s_5 = (4, 4|1)$. This schedule is the result of the assignation of one task to CU_2, while the rest of the tasks are assigned to CU_1
      (3) $s_6 = (1|2, 4)$, $s_7 = (2|1, 4)$, $s_8 = (4|1, 2)$. This schedule is the result of the assignation of one task is assigned to CU_1, while the rest of the tasks are assigned to CU_2.
   (c) Computation of the payoff function 8. To this aim, we need to calculate the makespan for all the possible schedules, based on *ETC* matrix:
      (1) $C_{max}(s_1) = 1/1 + 2/1 + 4/1 + 0 = 7$ and $C_{max}(s_2)$=0 + 1/2 + 2/2 + 4/2 = 3.5) sec.
      (2) $C_{max}(s_3) = 1/1 + 2/1 + 4/2 = 5$, $C_{max}(s_4) = 1/1 + 4/1 + 2/2 = 6$, $C_{max}(s_5) = 4/1 + 1/2 + 2/1 = 3.5$ sec.
      (3) $C_{max}(s_6)$=(1/1 + 2/2 + 4/2 = 4, $C_{max}(s_7) = 2/1 + 1/2 + 4/2 = 4.5$, $C_{max}(s_8)$=4/1 + 1/2 + 2/2 = 6.5) sec.
   (d) Find the schedule that maximize the payoff, which means the schedule that results in the shortest makespan,6 for the particular *Job*. If the run of the Genetic Algorithm results in a set of equal suboptimal schedules, then we can assign probabilities to them to avoid the "local minima" trap. According to the resulting makespans for the schedules s_2 and s_5, we may randomize the selection, and set the mixed strategy vector in the following form:

$$(sigma_1^1, sigma_1^2, \ldots, sigma_1^8) = (0, 1/2, 0, 0, 1/2, 0, 0, 0)$$

   Therefore the *Leader*'s payoff function is:

$$\begin{aligned} & H_1(\sigma_1, \sigma_2)= \\ & \textstyle\sum_{p=1,\ldots,8} \sum_{l=1,2} \\ & \sigma_1^p \sigma_2^l C_{\max}(s_p)= \\ & 1/2*1/3*3.5 + 1/2*1/3*3.5 + 1/2*2/3*3.5 + 1/2*3/3*3.5 \\ & = 4.08(3) sec. \end{aligned} \qquad (19)$$

   (e) Selection of a single action: We chose it randomly, with equal probability among schedules s_2 and s_5. In this case, the schedule

number 2 was selected.
   (f) Submission of this schedule to both the *CU* and the *Follower* player.
2. Computation of the *Follower*'s move. This step is, in turn, composed of the following substeps:
   (a) Computation of the *Idle* and *Busy* times, see eq. 13 and 14 respectively for all the CUs. Given the resulting schedule: $s_2 = (-|1, 2, 4)$ results in $t^1_{busy} = 0$ sec. and$t^2_{busy} = 3.5$ sec.
   (b) Computation of the total energy consumption 21 of the particular schedule for all available energy policies:
   - If the policy number 1 is applied:

$$\begin{aligned}&E_{total}=\\&\qquad\sum_{i=1}^{2}(P^i_{idle}{*}t^i_{idle} + P^i_{busy}{*}t^i_{busy})=\\&P^1_{idle}{*}t^1_{idle} + P^1_{busy}{*}t^1_{busy} + P^2_{idle}{*}t^2_{idle} + P^2_{busy}{*}t^2_{busy}=\\&\qquad 2{*}3.5 + 0 + 0 + 4{*}3.5 = 21KWh\end{aligned} \tag{20}$$

   - If the policy number 2 is applied:

$$\begin{aligned}&E_{total}=\\&\qquad\sum_{i=1}^{2}(P^i_{sleep}{*}t^i_{sleep} + P^i_{busy}{*}t^i_{busy})=\\&P^1_{sleep}{*}t^1_{sleep} + P^1_{busy}{*}t^1_{busy} + P^2_{sleep}{*}t^2_{sleep} + P^2_{busy}{*}t^2_{busy}=\\&\qquad 0.1{*}3.5 + 0 + 0 + 4{*}3.5 = 14.35KWh\end{aligned} \tag{21}$$

   (c) Find the energy policy that minimizes the energy consumption, that is, the maximization of the payoff function. In this particular example, the optimal energy policy is the policy number 2: $\sigma^1_2 = 0$, $\sigma^2_2 = 1$.
   (d) If necessary, we may randomize over several suboptimal solutions. In this simple example, this step is not necessary.
   (e) Computation of the *Follower's* payoff 22, as follows:

$$\begin{aligned}&H_1(\sigma_1, \sigma_2)=\\&\textstyle\sum_{p=1,\ldots,8}\sum_{l=1,2}\\&\sigma_1^P\sigma_2^l E_{total} = 1/2{*}1 + 14.35KWh\end{aligned} \tag{22}$$

   (f) Selection of the decision made by the system: The application of policy number 2 (only).
   (g) Application of this energy policy to the *CU*s, and return the decision to the game *Leader*.

These steps are repeated for every incoming *Job*. In real-life scenarios we have to employ a Genetic Algorithm (GA) to sole the NP-hard problem of the *Leader*'s move (the resulting scheduling) and a Simplex method to compute the follower strategy.

4. Experimental analysis

In this work, we aim to empirically demonstrate that the proposed simulation tool, GAME-SCORE, as well as the implemented energy-aware scheduling algorithm based on the Stackelberg-Game model may have a notable positive impact in terms of energy efficiency and performance, as well as an optimal balance between them.

In the following subsections we present the scheduling framework and algorithm considered, the simulation environment, the parameters and KPIs under evaluation and the considered realistic scenarios where we compare our proposal to static energy-efficiency policies.

4.1. Scheduling framework

In this experimental analysis, we employ a *Monolithic centralized scheduler* model. This scheduling model [28] works very well under low job-arrival rate conditions, such as long-running MapReduce jobs [12], since latencies of seconds or minutes [15] are acceptable in this context. This kind of scheduler can perform high-quality scheduling decisions [13,56] by examining the whole cluster state to determine the performance impact of hardware heterogeneity and interference in shared resources [25,38,41,46,55], and thereby it can choose the best resources for each task. This model leads to higher machine utilization [52], shorter execution times, better load balancing, more predictable performance [14,57], and increased reliability [44]. The scheduling process for the monolithic centralized scheduler is illustrated in Figure 7

Various parameters may characterize the jobs that make up the workloads of the Cloud-Computing system [40]. In this work, we focus on the following main attributes of the job j_k:

- **Job inter-arrival time** TI_{j_k}- represents the time elapsed between two consecutive jobs (j_k) submissions of the same workload type

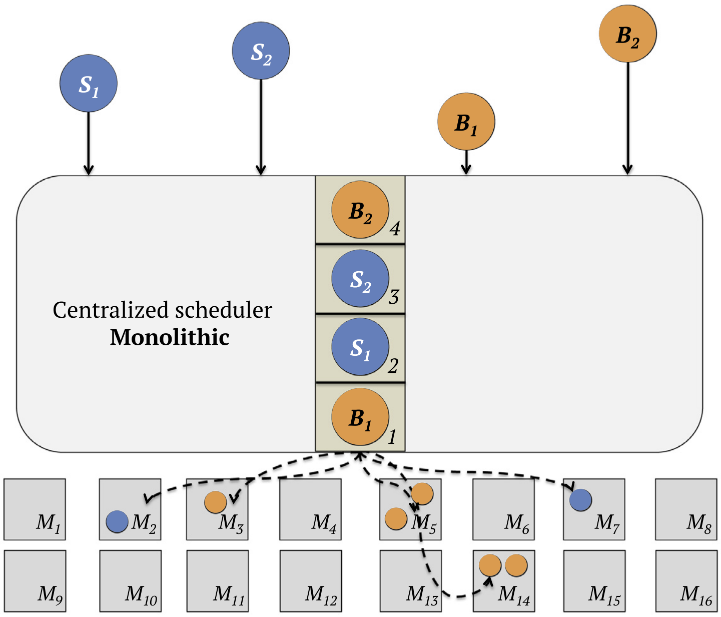


Fig. 7. Monolithic centralized scheduling workflow, B - Short-running Batch task, S - Long-running Service task, M - Machine

W. Thus, the number of jobs to be scheduled and executed by the Cloud-Computing system in a given time is defined by this parameter.

- **Job duration time** TD_{j_k}- is the completion time of the j_k job in the Cloud-Computing system
- **Number of tasks** TT_{j_k}- is the number of tasks that makes up the job j_k.

The performance efficiency of any Cloud-Computing *Scheduler* is related to the number of jobs that can be scheduled in a given time as well as the quality of the scheduling decisions. We consider the processing time (makespan) of the total set of jobs, that is, the workload W_s, as the main key performance indicator of the scheduling quality.

Usually, the term *workload* is conformed by the whole set of inputs related to Cloud-Computing systems, such as: applications, service packages and related data required by tasks. In Cloud Computing, such inputs are often submitted by the Cloud-Computing users by means of cloud services hosted in Cloud clusters. It should be also borne in mind that Cloud-Computing workloads are not usually composed of real-time applications.

4.1.1. Genetic Algorithm for searching optimal schedule

The scheduling of tasks in cloud-computing computational clusters constitutes an NP-complete problem [50], whose complexity depends on the features considered [35], such as:

(a) the number of scheduling criteria to be optimized (one vs. multi-criteria);
(b) nature of the environment (static vs. dynamic);
(c) nature of tasks (*Batch* or *Service*); and
(d) dependency between tasks (independent vs. dependent).

In this work, we use a heuristic algorithm that takes into account the aforementioned requirements in order to solve the NP-complete problem. This scheduling algorithm is based on a genetic algorithm with dedicated population representation [26,30], which can be characterized as follows:

(a) a single gene represents one task, which is unique within the population;
(b) each chromosome is composed by a set of tasks (genes);
(c) each individual is composed of one chromosome and represents a scheduling assignation for a single computing node;
(e) the population is composed of *m* individuals and represents a schedule for all *n* tasks;
(f) the fitness function depends on the optimization objectives

. All individuals take part in the reproduction process. Individuals presenting the lowest value for the fitness function (best adapted) are crossed with worst-adapted individuals (those that show the highest values for the fitness function). Crossing involves exchanging genes between chromosomes. The population obtained in the evolution process defines the suboptimal schedule, as

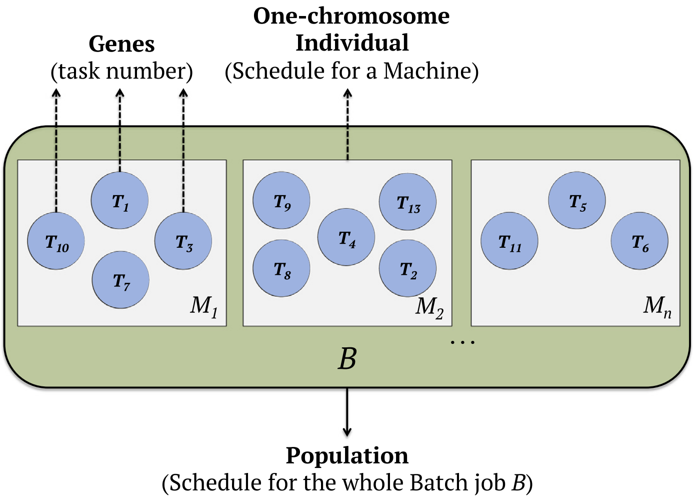


Fig. 8. Genetic algorithm model for energy-aware scheduling

shown in Figure 8.

4.1.2. *Workload types*

The quality of the scheduling process has a notable impact in Cloud-Computing systems, both on the overall quality of the cloud services as well as on the fulfillment of Service Level Agreements (SLAs).

We can classify the workloads to be processed according to two main characteristics:

- The internal architecture of the workload, that is, the relationship between jobs in the same workload. In this model, the kind and number of jobs that form the cloud applications, as well as the dependencies between them describe the whole workload. Hence, such jobs may be processed as a Directed Acyclic Graph [9], in parallel, and sequentially.
- The processing model of the jobs. In this model, we consider the following type of jobs:
  - **Batch workload *BW*–** this workload is composed of jobs that have a strictly-specified job arrival, start and completion times since these jobs are designed to perform a given computation and then finish.
  - **Service workload *SW*** – this workload is defined as a set of long-running jobs which usually need a higher amount of resources. These jobs have a determined job arrival and start time, but the completion time is not a priori determined.

    As real-life examples of the aforementioned workloads, MapReduce jobs [12] are classified as belonging to the *Batch* workload *BW*. On the other hand, long-running services such as BigTable [10] and HDFS [47], and web servers make up the *Service* workload *SW*.

In this experimental analysis, we focus on the evaluation of the proposed simulation tool and the dynamic application of energy-efficiency policies based on the Stackelberg-Game model. To this aim, We created heterogeneous and realistic workloads composed of *BW* and *SW* jobs based on the trends of the industry and Google Data-Center traces present in [1,16].

4.2. *Simulation environment*

We used the GAME-SCORE simulator to perform a simple experiment that simulates seven days of operation time of a computational cluster composed of 1,000 heterogeneous machines of 4 CPU cores and 8GB RAM and one central monolithic scheduler. Each machine has the following features:

- **Computing profile**: Differences in the processor's computing power has been mocked by generating randomly a [1x - 4x] computing speed factor. Thus, a given computing node may be, as a maximum, four times faster than the slowest one.
- **Energy profile**: Processor's power consumption heterogeneity has been simulated by generating randomly a [1x - 4x] energy consumption factor. Thus, a given machine *M* may be (as a maximum) four times more energy-consuming than the more efficient one. Hence, for a 4-core server, the maximum power consumption may be described as: $P_{total} \in$ [300, 1200] W.

In this experiment, we chose an heterogeneous day-night patterned mixed workload. This workload, which is composed of 22,000 *Batch* jobs and 2,200 *Service* jobs, uses 30% of the computational cluster computational resources on average, with peak loads that achieve 60% of utilization.

In order to create this realistic cloud scheduling scenario, the following workload parameters were considered:

- **Job inter-arrival**: The inter-arrival time TI_{j_k} of *BW* jobs is sampled from an exponential distribution whose mean value is 90 (seconds). For *SW* jobs, this inter-arrival time is sampled from an exponential distribution with a mean value of 900 (seconds).
- **Job structure**: The number of tasks TT_{j_k} for each job in *BW* is sampled from an exponential distribution with a mean value is 50, while the number of tasks for each job in *SW* is sampled from an exponential distribution whose mean value is 9.
- **Task duration**: The duration TD_{j_k} of *BW*-jobs tasks is sampled from an exponential distribution whose mean value is 90 (seconds). For *SW*-jobs tasks, this duration is sampled from an exponential distribution with a mean value of 2000 (seconds).
- **Resource usage**: in *SW*-jobs tasks consume 0.3 CPU cores and 0.5 GB of memory, in *SW*-job tasks consume 0.5 CPU cores and 1.2 GB of memory.

4.3. Energy-efficiency indicators

For the analysis of the impact in terms of the reduction of the energy consumption of the Cloud-Computing system, the following energy-efficiency parameters are considered:

- E_c **– Energy consumed**: This parameter represents the total energy used by the Cloud-Computing system.
- E_s **– Energy saved**: This parameter represents the total energy saved by the Cloud-Computing system compared to the current[1] operation energy consumption.
- ***SD*** **– Number of shut-downs**: The total number of shut-down operations performed over all the resources during the simulated operation time. This parameter can be related to the hardware stress due to booting actions.
- E_sSD **– Energy saved per shut-down**: This parameter computes the energy saved against the shut-downs performed. Hence, it shows the *efficiency* of the shut-down actions performed.
- ***IR*** **– Idle resources**: This parameter represents the amount of resources turned on but not in use.

4.4. Scheduling efficiency indicators

For the comparison and evaluation of the performance of the Cloud-Computing system, we define the following key performance indicators (KPIs):

- JQT_{fi} **– Job queue times until first scheduled**: This parameter represents the time a job needs to wait in queue until it scheduled for the first time.
- JQT_{full} **– Job queue times until fully scheduled**: This parameter represents the time a job needs to wait in queue until it is fully scheduled.
- ***SBT*** **– Scheduler busy time**: This parameter represents the total time spent by the scheduler performing scheduling operations.
- MS_t **– Final makespan**: This parameter represents the total time spent by jobs in the Cloud-Computing system on average. It is worth to mention that only the *Batch* workload *BW* has makespan, since *Service* workload *SW* has no determined end, but usually these jobs are killed by operators or automated systems when they are no longer necessary.
- MS_0 **– Epoch 0 makespan**: This parameter makes reference to the makespan of jobs in the first iteration of the genetic algorithm on average.

4.5. Simple example for Always and Never power-off policies in SCORE simulator

In this experiment, we aim to empirically show a simple strategy where a dynamic change of *Power-off* policy could represent a significant improvement of energy-efficiency.

The Stackelberg process described previously is applied for every scheduling decision in the system. In this experiment, the *Shut-down decision policy* used to switch the *Power-off* policy is made based on cluster available resources. Every time that the idle resources exceed a given threshold, the *Always* power-off policy is applied. On the other hand, when the amount of available resources is lower than that threshold, the *Never* power-off policy is applied. The results of the application of the Stackelberg game against the static energy policies are presented in Tables 1 and 2.

This experimentation shows that the Stackelberg model applies a minor negative impact (+ 15%) in terms of queue times compared to the *Never* shut-down decision (17 vs 20 ms.), while the negative impact of the *Always* and *Random* strategies are +160% and 80% (17 vs. 44 and vs. 30.5 ms.) respectively. In terms of energy consumption, the *Stackelberg* model only consumes 10% more energy than the *Always* and *Random* shut-down policies (29 vs 32 MWh). On the other hand, the *Always* and *Random* strategies achieve approximately 7% lower final average makespan time (146 vs. 155 s.) due to the dynamic changes of the *Stackelberg* model.

4.6. Extended example in SCORE simulator

In this section, we extended the simple experimentation presented in Section 4.5. In order to keep results comparable, we reused all the configuration parameters taken for the large-scale CC system shown in Section the 4.5. However, in this experiment the *Central*

[1] Current operation for the same computational cluster and workload, but without applying energy-saving polices

Table 1
Energy-efficiency results for the simple Stackelberg experiment

Strategy	E_c (MWh)	E_s (MWh)	Savings (%)	*SD*	E_sSD (kWh)	*IR* (%)
Static-Never	55.65	0	0	0	N/A	70.71
Static-Always	29.04	26.83	48.03	19,722	1.36	3.49
Static-Random	29.33	26.47	47.44	10,943	2.42	4.49
Stackelberg	**32.24**	**23.65**	**42.32**	**1,665**	**14.21**	**11.11**

Table 2
Performance results for the simple Stackelberg experiment

Strategy	Workload	JQT_{full} (ms)	JQT_{fi} (ms)	*SBT* (h)	MS_t (s)	MS_0 (s)
Static-Never	Batch	17.05	17.02	3.71	142.55	177.65
Static-Never	Service	20.08	20.07	0.12	N/A	N/A
Static-Always	Batch	43.93	19.58	4.25	146.12	185.74
Static-Always	Service	35.11	21.19	0.13	N/A	N/A
Static-Random	Batch	30.50	18.44	4.02	143.98	180.48
Static-Random	Service	33.46	22.03	0.12	N/A	N/A
Stackelberg	**Batch**	**20.40**	**17.62**	**3.75**	**155.04**	**179.63**
Stackelberg	Service	29.91	21.53	0.12	N/A	N/A

Energy-efficiency Manager switches dynamically between the *Never* and the *Always* power-off policies by applying every *Shut-down decision policy* described in Section 3.2. The results obtained are shown in Table 3 and 4.

In general, the Stackelberg process may apply a negative impact in terms of makespan due to that the *Power-off* policy may suddenly change. This change can impact on two consecutive scheduling processes of a single job, which could apply a performance penalty if there are no sufficient resources to immediately execute the job tasks. This negative impact can be mitigated by the scheduler when only one static *Power-off* policy is applied. It should be borne in mind that only *Batch* jobs would suffer from this negative impact since *Service* jobs have no determined finish.

This experimentation shows that the results of the Stackelberg model depends directly on the *Shut-down decision* policy. More conservative probabilistic models, such as Exponential and Gamma, achieve and 45% faster queue times than a *Random* strategy (33 vs 18 and 19 ms.) respectively while consuming approximately 8 and 12% more energy (29.4 vs. 31.5 and 33.8 MWh) respectively. On the other hand, strategies that rely on leaving a security margin of free resources, such as *Margin*, could achieve approximately 40% faster queue times than a *Random* strategy (33 vs 20 ms.), and it would only consume 10% more energy (29.4 vs 32.2 MWh). It can be noticed that conservative strategies such as Gamma apply almost no stress to the hardware, performing approximately 1,000 shut-downs in a week of operation time, which represents 10% of those performed by the *Random* decision policy.

4.7. Results summary

The results obtained show that in general, the Stackelberg-Game-based can balance the trade-offs present in energy-efficient computational clusters: energy-consumption reduction vs. performance.

As showed, our model can notably improve the scheduling performance compared to static energy-efficiency policies while achieving almost the same levels of energy efficiency when the proper energy policies and switching decisions are used. However, the Stackelberg process applies a minor negative impact in terms of makespan due to the sudden changes of the *Power-off* policy applied.

Table 3
Energy-efficiency results for the extended Stackelberg experiment, where the *Always* and *Never* shut-down policies are switched following several decision policies

Strategy	Switch Decision	E_c (MWh)	E_s (MWh)	Savings (%)	*SD*	E_sSD (kWh)	*IR* (%)
Static-Never	N/A	55.65	0	0	0	N/A	70.71
Static-Always	N/A	29.04	26.83	48.03	19,722	1.36	3.49
Stackelberg	Random	29.37	26.43	47.36	11,434	2.31	4.63
Stackelberg	Margin	32.24	23.65	42.32	1,665	14.21	11.11
Stackelberg	Exponential	31.48	24.32	43.49	2,998	8.08	10.05
Stackelberg	Gamma	33.78	22.25	39.71	1,074	20.72	14.63

Table 4
Performance results for the extended Stackelberg experiment, where the *Always* and *Never* shut-down policies are switched following several decision policies

Strategy	Workload	Switch Decision	JQT_{full} (ms)	JQT_{fi} (ms)	*SBT* (h)	MS_t (s)	MS_0 (s)
Static-Never	Batch	N/A	17.05	17.02	3.71	142.55	177.65
Static-Never	Service	N/A	20.08	20.07	0.12	N/A	N/A
Static-Always	Batch	N/A	43.93	19.58	4.25	146.12	185.74
Static-Always	Service	N/A	35.11	21.19	0.13	N/A	N/A
Stackelberg	Batch	Random	33.29	18.78	9.88	143.07	181.42
Stackelberg	Service	Random	33.17	22.51	0.69	N/A	N/A
Stackelberg	Batch	Margin	20.40	17.62	3.75	155.04	179.63
Stackelberg	Service	Margin	29.91	21.53	0.12	N/A	N/A
Stackelberg	Batch	Exponential	18.92	17.16	9.20	144.08	178.28
Stackelberg	Service	Exponential	24.67	20.39	0.66	N/A	N/A
Stackelberg	Batch	Gamma	18.23	17.12	9.20	163.80	180.00
Stackelberg	Service	Gamma	22.05	20.14	0.66	N/A	N/A

5. Conclusions

In this paper, we presented a new simulation tool called GAME-SCORE which implements method that focus on the balance between two opposite needs of every energy-efficient CC system: high performance throughput and low energy consumption.

The proposed simulation tool and model are based on a non-zero sum Stackelberg Game with the leader player, the *Scheduling Manager*, which tries to minimize the makespan with its scheduling decisions while the follower player, the *Energy-efficiency Manager*, responds to the leader player move with the application of energy-efficiency policies that may shut-down the idle machines. These strategies are represented by the independent utility functions for each player. Our model enables the dynamic application of energy-efficiency strategies depending on the current and predictable workload.

The results of our simple experimental evaluation show that the proposed model perform better than the application of only one energy-efficiency policy, both in terms of energy-efficiency and performance. This means that the Stackelberg Game model can balance better between opposed needs (performance and energy efficiency) and can adapt better to heterogeneous workloads.

It could be also observed in the experimental analysis, that probabilistic decision strategies that try to predict the short-term future workload can balance better between energy consumption and performance impact.

For the presented reasons, we consider that the proposed simulator GAME-SCORE overperforms other simulators which only permit the application of static energy-aware scheduling algorithms and static energy-efficiency policies based on the shut-down of idle machines.

The presented model is just our first step towards the development of the new scheduling and resource allocation policies in order to optimize the energy utilization in the whole cloud distributing system . The model improvement plans include:

(a) exploration of more advanced energy policies;
(b) introduction of multiple players in order to play several games simultaneously without any central energy manager;
(c) examination of more scheduling models, such as two-level or shared-state models;
(d) test more complex and dynamic scheduling strategies;
(e) inclusion of VM/container migration and consolidation; and
(f) empirical comparison of the simulation results with real-life data.

.

Acknowledgements

This article is based upon work from COST Action IC1406 “High-Performance Modelling and Simulation for Big Data Applications” (cHiPSet), supported by COST (European Cooperation in Science and Technology) and by the VPPI - University of Sevilla.

References

[1] O.A. Abdul-Rahman, K. Aida, Towards understanding the usage behavior of Google cloud users: the mice and elephants phenomenon, IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Singapore, (2014), pp. 272–277. doi:10.1109/CloudCom.2014.75 .
[2] D.G. Andersen, S. Swanson, Rethinking flash in the data center, IEEE micro 30 (4) (2010) 52–54.
[3] G.S. Aujla, M. Singh, N. Kumar, A. Zomaya, Stackelberg game for energy-aware resource allocation to sustain data centers using res, IEEE Transactions on Cloud Computing (2017).
[4] A. Beloglazov, J. Abawajy, R. Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing, Future generation computer systems 28 (5) (2012) 755–768.
[5] A. Beloglazov, R. Buyya, Energy efficient resource management in virtualized cloud data centers, Proceedings of the 2010 10th IEEE/ACM international

conference on cluster, cloud and grid computing, IEEE Computer Society, 2010, pp. 826–831.

[6] A. Beloglazov, R. Buyya, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, Concurrency and Computation: Practice and Experience 24 (13) (2012) 1397–1420.

[7] J.L. Berral, I.n. Goiri, R. Nou, F. Julià, J. Guitart, R. Gavaldà, J. Torres, Towards energy-aware scheduling in data centers using machine learning, Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking, e-Energy '10, ACM, New York, NY, USA, 2010, pp. 215–224, https://doi.org/10.1145/1791314.1791349.

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

[9] M.C. Calzarossa, M.L. Della Vedova, L. Massari, D. Petcu, M.I. Tabash, D. Tessera, Workloads in the clouds, Principles of Performance and Reliability Modeling and Evaluation, Springer, 2016, pp. 525–550.

[10] F. Chang, J. Dean, S. Ghemawat, W.C. Hsieh, D.A. Wallach, M. Burrows, T. Chandra, A. Fikes, R.E. Gruber, Bigtable: A distributed storage system for structured data, ACM Transactions on Computer Systems (TOCS) 26 (2) (2008) 4.

[11] L. Cupertino, G. Da Costa, A. Oleksiak, W. Pia, J.-M. Pierson, J. Salom, L. Siso, P. Stolf, H. Sun, T. Zilio, et al., Energy-efficient, thermal-aware modeling and simulation of data centers: the coolemall approach and evaluation results, Ad Hoc Networks 25 (2015) 535–553.

[12] J. Dean, S. Ghemawat, Mapreduce: simplified data processing on large clusters, Communications of the ACM 51 (1) (2008) 107–113.

[13] C. Delimitrou, C. Kozyrakis, Paragon: Qos-aware scheduling for heterogeneous datacenters, ACM SIGPLAN Notices, 48 ACM, 2013, pp. 77–88.

[14] C. Delimitrou, C. Kozyrakis, Quasar: resource-efficient and qos-aware cluster management, ACM SIGPLAN Notices, 49 ACM, 2014, pp. 127–144.

[15] C. Delimitrou, D. Sanchez, C. Kozyrakis, Tarcil: reconciling scheduling speed and quality in large shared clusters, Proceedings of the Sixth ACM Symposium on Cloud Computing, ACM, 2015, pp. 97–110.

[16] S. Di, D. Kondo, C. Franck, Characterizing cloud applications on a Google data center, 42nd International Conference on Parallel Processing (ICPP), Lyon, France, (2013).

[17] N. El-Sayed, I.A. Stefanovici, G. Amvrosiadis, A.A. Hwang, B. Schroeder, Temperature management in data centers: why some (might) like it hot, ACM SIGMETRICS Performance Evaluation Review 40 (1) (2012) 163–174.

[18] X. Fan, W.-D. Weber, L.A. Barroso, Power provisioning for a warehouse-sized computer, ACM SIGARCH Computer Architecture News, 35 ACM, 2007, pp. 13–23.

[19] M.E. Femal, V.W. Freeh, Boosting data center performance through non-uniform power allocation, Second International Conference on Autonomic Computing (ICAC'05), IEEE, 2005, pp. 250–261.

[20] D. Fernández-Cerero, A. Fernández-Montes, A. Jakóbik, J. Kołodziej, M. Toro, Score: Simulator for cloud optimization of resources and energy consumption, Simulation Modelling Practice and Theory 82 (2018) 160–173, https://doi.org/10.1016/j.simpat.2018.01.004.

[21] D. Fernández-Cerero, A. Fernández-Montes, J.A. Ortega, Energy policies for data-center monolithic schedulers, Expert Systems with Applications (2018), https://doi.org/10.1016/j.eswa.2018.06.007.

[22] D. Fernández-Cerero, A. Jakóbik, A. Fernández-Montes, J. Kołodziej, Stackelberg game-based models in energy-aware cloud scheduling, in: L. Nolle (Ed.), Proc. 32nd European Conference on Modelling and Simulation ECMS 2018 (ECMS, Wilhelmshaven, Germany, May 2018), ECMS '18, European Council for Modelling and Simulation, Dudweiler, Germany, 2018, pp. 460–467.

[23] D. Fernández-Cerero, A. Jakóbik, D. Grzonka, J. Koodziej, A. Fernández-Montes, Security supportive energy-aware scheduling and energy policies for cloud environments, Journal of Parallel and Distributed Computing 119 (2018) 191–202, https://doi.org/10.1016/j.jpdc.2018.04.015.

[24] A. Fernández-Montes, D. Fernández-Cerero, L. González-Abril, J.A. Álvarez-García, J.A. Ortega, Energy wasting at internet data centers due to fear, Pattern Recognition Letters 67 (2015) 59–65.

[25] S. Govindan, J. Liu, A. Kansal, A. Sivasubramaniam, Cuanta: quantifying effects of shared on-chip resource interference for consolidated virtual machines, Proceedings of the 2nd ACM Symposium on Cloud Computing, ACM, 2011, p. 22.

[26] D. Grzonka, A. Jakóbik, J. Kołodziej, S. Pllana, Using a multi-agent system and artificial intelligence for monitoring and improving the cloud performance and security, Future Generation Computer Systems (2017), https://doi.org/10.1016/j.future.2017.05.046. (in press)

[27] S.K.S. Gupta, R.R. Gilbert, A. Banerjee, Z. Abbasi, T. Mukherjee, G. Varsamopoulos, Gdcsim: A tool for analyzing green data center design and resource management techniques, 2011 International Green Computing Conference and Workshops, (2011), pp. 1–8, https://doi.org/10.1109/IGCC.2011.6008612.

[28] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, A. Goldberg, Quincy: fair scheduling for distributed computing clusters, Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles, ACM, 2009, pp. 261–276.

[29] A. Jakóbik, D. Grzonka, J. Kołodziej, Security supportive energy aware scheduling and scaling for cloud environments (2017).

[30] A. Jakóbik, D. Grzonka, J. Kołodziej, Security supportive energy aware scheduling and scaling for cloud environments, European Conference on Modelling and Simulation, ECMS 2017, Budapest, Hungary, May 23-26, 2017, Proceedings. (2017), pp. 583–590, https://doi.org/10.7148/2017-0583.

[31] A. Jakóbik, D. Grzonka, J. Kolodziej, A.E. Chis, H. González-Vélez, Energy efficient scheduling methods for computational grids and clouds, Journal of Telecommunications and Information Technology (1) (2017) 56.

[32] A. Jakobik, D. Grzonka, F. Palmieri, Non-deterministic security driven meta scheduler for distributed cloud organizations, Simulation Modelling Practice and Theory. in press(available online 4 November 2016). doi:10.1016/j.simpat.2016.10.011.

[33] F. Juarez, J. Ejarque, R.M. Badia, Dynamic energy-aware scheduling for parallel task-based application in cloud computing, Future Generation Computer Systems (2016).

[34] D. Kliazovich, P. Bouvry, S.U. Khan, Greencloud: a packet-level simulator of energy-aware cloud computing data centers, The Journal of Supercomputing 62 (3) (2012) 1263–1283.

[35] J. Kołodziej, Evolutionary Hierarchical Multi-Criteria Metaheuristics for Scheduling in Large-Scale Grid Systems, Springer Publishing Company, Incorporated, 2012.

[36] J. Koomey, Growth in data center electricity use 2005 to 2010, A report by Analytical Press, completed at the request of The New York Times 9 (2011).

[37] Y.C. Lee, A.Y. Zomaya, Energy efficient utilization of resources in cloud computing systems, The Journal of Supercomputing 60 (2) (2012) 268–280.

[38] J. Mars, L. Tang, Whare-map: heterogeneity in homogeneous warehouse-scale computers, ACM SIGARCH Computer Architecture News, 41 ACM, 2013, pp. 619–630.

[39] A. Miyoshi, C. Lefurgy, E. Van Hensbergen, R. Rajamony, R. Rajkumar, Critical power slope: understanding the runtime effects of frequency scaling, Proceedings of the 16th international conference on Supercomputing, ACM, 2002, pp. 35–44.

[40] R. Nallakumar, N. Sengottaiyan, K.S. Priya, A survey on scheduling and the attributes of task scheduling in the cloud, Int. J. Adv. Res. Comput. Commun. Eng 3 (10) (2014) 8167–8171.

[41] R. Nathuji, A. Kansal, A. Ghaffarkhah, Q-clouds: managing performance interference effects for qos-aware clouds, Proceedings of the 5th European conference on Computer systems, ACM, 2010, pp. 237–250.

[42] E. Niewiadomska-Szynkiewicz, A. Sikora, P. Arabas, J. Kołodziej, Control system for reducing energy consumption in backbone computer network, Concurrency and Computation: Practice and Experience 25 (12) (2013) 1738–1754.

[43] J.M. Pérez-Álvarez, M.T. Gómez-López, A.J. Varela-Vaca, F.F. de la Rosa Troyano, R.M. Gasca, Governance knowledge management and decision support using fuzzy governance maps, Business Process Management Workshops - BPM 2016 International Workshops, Rio de Janeiro, Brazil, September 19, 2016, Revised Papers, (2016), pp. 208–219, https://doi.org/10.1007/978-3-319-58457-7_16.

[44] M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, J. Wilkes, Omega: flexible, scalable schedulers for large compute clusters, Proceedings of the 8th ACM European Conference on Computer Systems, ACM, 2013, pp. 351–364.

[45] R.K. Sharma, C.E. Bash, C.D. Patel, R.J. Friedrich, J.S. Chase, Balance of power: Dynamic thermal management for internet data centers, IEEE Internet Computing 9 (1) (2005) 42–49.

[46] D. Shue, M.J. Freedman, A. Shaikh, Performance isolation and fairness for multi-tenant cloud storage. OSDI, 12 (2012), pp. 349–362.

[47] K. Shvachko, H. Kuang, S. Radia, R. Chansler, The hadoop distributed file system, 2010 IEEE 26th symposium on mass storage systems and technologies (MSST), IEEE, 2010, pp. 1–10.

[48] S. Sohrabi, A. Tang, I. Moser, A. Aleti, Adaptive virtual machine migration mechanism for energy efficiency, Proceedings of the 5th International Workshop on Green and Sustainable Software, ACM, 2016, pp. 8–14.

[49] T.T. Sá, R.N. Calheiros, D.G. Gomes, CloudReports: An Extensible Simulation Tool for Energy-Aware Cloud Computing Environments, Springer International

Publishing, Cham, 2014, pp. 127–142.

[50] J. Ullman, Np-complete scheduling problems, Journal of Computer and System Sciences 10 (3) (1975) 384–393, https://doi.org/10.1016/S0022-0000(75)80008-0.

[51] A. Varga, Discrete event simulation system, Proc. of the European Simulation Multiconference (ESM'2001), (2001).

[52] A. Verma, L. Pedrosa, M. Korupolu, D. Oppenheimer, E. Tune, J. Wilkes, Large-scale cluster management at google with borg, Proceedings of the Tenth European Conference on Computer Systems, ACM, 2015, p. 18.

[53] B. Wickremasinghe, R.N. Calheiros, R. Buyya, Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications, 2010 24th IEEE International Conference on Advanced Information Networking and Applications, (2010), pp. 446–452, https://doi.org/10.1109/AINA.2010.32.

[54] A. Wilczyński, A. Jakóbik, Using Polymatrix Extensive Stackelberg Games in Security–Aware Resource Allocation and Task Scheduling in Computational Clouds, Journal of Telecommunications and Information Technology 1 (2017).

[55] H. Yang, A. Breslow, J. Mars, L. Tang, Bubble-flux: Precise online qos management for increased utilization in warehouse scale computers, ACM SIGARCH Computer Architecture News, 41 ACM, 2013, pp. 607–618.

[56] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, I. Stoica, Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling, Proceedings of the 5th European conference on Computer systems, ACM, 2010, pp. 265–278.

[57] X. Zhang, E. Tune, R. Hagmann, R. Jnagal, V. Gokhale, J. Wilkes, Cpi 2: Cpu performance isolation for shared compute clusters, Proceedings of the 8th ACM European Conference on Computer Systems, ACM, 2013, pp. 379–391.

DAMIÁN FERNÁNDEZ CERERO received the B.E. degree and the M.Tech. degrees in Software Engineering from the University of Sevilla. In 2014, he joined the Department of Computer Languages and Systems, University of Seville, as a PhD. Student. Currently he both teaches and conducts research at University of Sevilla. He has worked on several research projects supported by the Spanish government and the European Union. His research interests include energy efficiency and resource scheduling in computational clusters. His e-mail address is: damiancerero@us.es

AGNIESZKA JAKÓBIK (KROK) received her M.Sc. in the field of stochastic processes at the Jagiellonian University, Cracow, Poland and Ph.D. degree in the field of neural networks at Tadeusz Kosciuszko Cracow University of Technology, Poland, in 2003 and 2007. She is an Assistant Professor. Her e-mail address is: agneskrok@gmail.com

ALEJANDRO FERNÁNDEZ-MONTES GONZÁLEZ received the B.E. degree, M. Tech. and International Ph.D. degrees in Software Engineering from the University of Sevilla, Spain. In 2006, he joined the Department of Computer Languages and Systems, University of Sevilla, and in 2013 became Assistant Professor. His research interests include energy efficiency in distributed computing, applying prediction models to balance load and applying on-off policies to computational clusters. His e-mail address is: afdez@us.es

JOANNA KOŁODZIEJ is an associate professor in Research and Academic Computer Network (NASK) Institute and Department of Computer Science of Cracow University of Technology. She serves also as the President of the Polish Chapter of IEEE Computational Intelligence Society. Her e-mail address is: joanna.kolodziej68@gmail.com

Productive Efficiency of Energy-aware Data Centers

As the final step of this work we met the last research objective of this thesis dissertation: *"Proof that the productive analysis of realistic Cloud-Computing data centers can empirically guide data-center administrators to perform efficiency-related decisions"*. Data centers may reach energy consumption levels comparable to many industrial facilities and small-sized towns. Therefore, innovative and transparent energy policies should be applied to improve energy consumption and deliver the best performance. This paper compares, analyzes and evaluates various energy-efficiency policies which shut down underutilized machines, on an extensive set of data-center environments.

Data Envelopment Analysis (DEA) is then conducted for the detection of the best energy-efficiency policy and data-center characterization for each case. This analysis evaluates energy-consumption and performance indicators for Natural DEA and Constant Returns to Scale (CRS). We identify the best energy policies and scheduling strategies for high and low data-center demands and for medium-sized and large data centers; moreover, this work enables data-center managers to detect inefficiency and to implement further corrective actions.

The major contributions of this paper can be summarized as follows: a) Extensive empirical experimentation and analysis of various cloud-computing scenarios with a trustworthy and detailed simulation tool. b) Impact analysis in terms of energy consumption and performance of several energy-efficiency policies which shut down idle machines by means of DEA. c) DEA-conducted analysis of performance impact and energy consumption of a set of scheduling models for large-scale data centers. d) Empirical determination and proposal of corrective actions to achieve optimal efficiency.

This work was published in *Energies.* This Journal is indexed in JCR with an **Impact Factor of 2.676**. The Journal stands in ranking **Q2** in Energy Fuels (48/97).

energies

MDPI

Article

Productive Efficiency of Energy-Aware Data Centers

Damián Fernández-Cerero [1,*], Alejandro Fernández-Montes [1] and Francisco Velasco [2]

[1] Department of Computer Languages and Systems, University of Seville, 41012 Sevilla, Spain; afdez@us.es
[2] Department of Applied Economy I, University of Seville, 41018 Sevilla, Spain; velasco@us.es
* Correspondence: damiancerero@us.es; Tel.: +34-954-559-531

Received: 17 July 2018; Accepted: 1 August 2018; Published: 8 August 2018

Abstract: Information technologies must be made aware of the sustainability of cost reduction. Data centers may reach energy consumption levels comparable to many industrial facilities and small-sized towns. Therefore, innovative and transparent energy policies should be applied to improve energy consumption and deliver the best performance. This paper compares, analyzes and evaluates various energy efficiency policies, which shut down underutilized machines, on an extensive set of data-center environments. Data envelopment analysis (DEA) is then conducted for the detection of the best energy efficiency policy and data-center characterization for each case. This analysis evaluates energy consumption and performance indicators for natural DEA and constant returns to scale (CRS). We identify the best energy policies and scheduling strategies for high and low data-center demands and for medium-sized and large data-centers; moreover, this work enables data-center managers to detect inefficiencies and to implement further corrective actions.

Keywords: data envelopment analysis; return-to-scale; cloud computing; efficiency; energy policies

1. Introduction

Data centers, which constitute the computational muscle for cloud computing, can be compared in energy consumption to many industrial facilities and towns. The latest trends show that these infrastructures represent approximately 2% of global energy consumption [1], with a 5% annual growth rate [2].

The data envelopment analysis mathematical model enables the management organizational divisions to measure the performance of an organization by providing the relative efficiency of each organizational unit. This relative efficiency measurement can be applied to a set of decision-making units, also known as DMUs, or for productive efficiency. The productive efficiency, also called technical efficiency, involves a collection of inputs (the resources needed for the production) and outputs (the production achieved). To this end, DEA constructs an "efficiency frontier" which places the relative performance of all units so these can be contrasted. This method is notably well-suited for the examination of the behavior of complex relations, even unknown, between numerous inputs and outputs, where the decisions made are affected by a level of uncertainty [3]. Moreover, DEA has been used both in private [4,5] and in public contexts [6–9].

Many initiatives have emerged looking for the decrease of the consumption of energy and the CO_2 trace of data-centers, especially those of a medium and large size. These facilities are composed of thousands and even tens of thousands of machines.

A substantial part of these initiatives focuses on the improvement of the Power Usage Effectiveness (PUE), that is the amount of energy consumed in non-computational tasks, such as power supply, cooling and networking components. This accounts for more than half of the energy consumption of an Internet data-center (IDC).

Several strategies are proposed to significantly improve energy efficiency in large-scale clusters [10]: cooling and temperature management [11,12]; power proportionality for CPU and memory hardware components [13,14]; fewer energy-hungry and non-mechanical hard disks [15]; and new proposals for energy distribution [16].

On the other hand, almost 50% of energy is consumed by computational servers to satisfy the incoming workload. The job arrival is not stable over time, but usually presents correlative low and high periods, such as those present in day/night and weekday/weekend workload patterns.

Such scenarios present a huge opportunity for the improvement of energy efficiency through proper scheduling and through the application of low-energy consumption modes to servers, since keeping servers in an idle state is extremely energy-inefficient. Many energy-aware schedulers, which aim to raise server usage, have been proposed in order to free up the maximum amount of machines so that they may put into hibernation [17–19]. In addition to these schedulers, several energy-conservation strategies may be applied in virtualized environments, such as the consolidation and migration of virtual machines [20,21].

Other strategies focus on the reduction of energy consumption in specific scenarios, such as those of distributed file systems [22,23].

The most aggressive approach involves the shut-down of underutilized servers in order to minimize energy consumption. Several shut-down policies have been proposed for grid computing environments in [24]. This strategy is yet to be widely implemented in working data-centers since a natural reticence to worsening QoS is usually present in data-center operators [25].

The innovation of the research presented in this paper involves the utilization of data envelopment analysis (DEA) as a mathematical technique to compare the efficiency regarding the consumption of energy and the performance of various workload scenarios, scheduling models and energy efficiency policies. This efficiency analysis enables data-center operators to make appropriate decisions about the number of machines, the scheduling solution and the shut-down strategy that must be applied so that data-centers run optimally. The final goal is the maximization of the productive efficiency, which is computed as the amount of energy consumed to serve a workload with a determined performance.

The major contributions of this paper can be summarized as follows:

1. Extensive empirical experimentation and analysis of various cloud-computing scenarios with a trustworthy and detailed simulation tool.
2. Impact analysis in terms of the energy consumption and performance of several energy efficiency policies, which shut-down idle machines by means of data envelopment analysis.
3. DEA-conducted analysis of the performance impact and energy consumption of a set of scheduling models for large-scale data-centers.
4. Empirical determination and proposal of corrective actions to achieve optimal efficiency.

The work is organized as follows. In Section 2, the authors introduce the current literature for the utilization of DEA presented for various areas, as well as the DEA model employed in this work. In Section 3, we briefly explain the set of energy efficiency policies that shut down idle servers. The scheduling models considered are explained in Section 4. In Section 5, the tool used for the simulation, the experimental environment, the energy model and DEA inputs/outputs are presented. Natural constant returns to scale (CRS) DEA results are described and analyzed in Section 6. Finally, we summarize this paper and present conclusions in Section 7.

2. Data Envelopment Analysis Model

Data envelopment Analysis (DEA) is a method that analyzes the connections between the outputs and inputs required in a production process in order to establish the efficiency frontiers [26]. This non-parametric technique was first described for the determination of the efficiency of DMUs by [27] and was formally defined by [28]. DEA has been proposed to measure the efficiency in various areas of operations research and management science [29–32]. Moreover, it has been applied to measure the environmental performance by other authors [33–40], who describe the gains of this method in the field of environmental management, which is a matter of undoubted relevance for the valuation of the sustainable development ability and pathway [41]. A critical feature of DEA for environmental analysis is the inclusion of desirable and undesirable outputs along with its own production variables, which cannot be isolated in an environmental analysis model of these features [42]. In this way, ref. [36] have refined a non-radial and radial model of DEA for environmental measurements. This approach separates the outputs into desirable and undesirable and presents two concepts: natural and managerial disposability. In this work, we employ the DEA radial approach for environmental assessments proposed by [37]. It should be borne in mind that a main feature of this approach is the utilization of DEA-RAM (range-adjusted measure), first proposed by [43] to treat in a unified manner the analysis of managerial and natural disposability.

2.1. Natural Disposability

Natural disposability refers to a DMU that improves its efficiency by decreasing its inputs in order to decrease its undesirable outputs, as well as to increase the desirable outputs.

In Model (1), each j-th DMU $j = 1, \ldots, n$, considers inputs $X_j = (x_{1j}, \ldots, x_{mj})^T$ for the production of desirable outputs $G_j = (g_{1j}, \ldots, g_{sj})^T$ and undesirable outputs $B_j = (b_{1j}, \ldots, b_{hj})^T$. Furthermore, d_i^x, $i = 1, \ldots, m$, d_r^g, $r = 1, \ldots, s$ and d_f^b, $f = 1, \ldots, h$ are all slack variables which are related to inputs, desirable and undesirable outputs, respectively. $\lambda = (\lambda_1, \ldots, \lambda_n)^T$ are structural or intensity variables, which are unknown and are used for the connection of the input and output vectors by means of a convex combination. R is the range resolute through the lower and upper limits of inputs, desirable outputs and undesirable outputs, denoted by:

$$R_i^x = (m+s+h)^{-1}\,(\max\{x_{ij}/j = 1, \ldots, n\} - \min\{x_{ij}/j = 1, \ldots, n\})$$

$$R_r^g = (m+s+h)^{-1}\,(\max\{g_{rj}/j = 1, \ldots, n\} - \min\{g_{rj}/j = 1, \ldots, n\})\ \textit{and}$$

$$R_f^b = (m+s+h)^{-1}\,(\max\{b_{fj}/j = 1, \ldots, n\} - \min\{b_{fj}/j = 1, \ldots, n\})$$

The natural efficiency of the k-th policy is computed by the following CRS and radial VRS model (see [37] for a better understanding):

$$\max\ \xi + \epsilon\,(\sum_{i=1}^{m} R_i^x\, d_i^x + \sum_{r=1}^{s} R_r^g\, d_i^g + \sum_{f=1}^{h} R_f^b\, d_i^b)$$

$$\begin{array}{rcl}
s.t.\quad \sum_{j=1}^{n} x_{ij}\lambda_j + d_i^x & = & x_{ik}\,, i = 1, \ldots, m\,, \\
\sum_{j=1}^{n} g_{rj}\lambda_j - d_r^g - \xi g_{rk} & = & g_{rk}\,, r = 1, \ldots, s\,, \\
\sum_{j=1}^{n} b_{fj}\lambda_j + d_f^b + \xi b_{fk} & = & b_{fk}\,, f = 1, \ldots, h\,, \\
d_i^x & \geq & 0\,, i = 1, \ldots, m\,, \\
d_r^g & \geq & 0\,, r = 1, \ldots, s\,, \\
d_f^b & \geq & 0\,, f = 1, \ldots, h\,, \\
\xi & & \text{Unrestricted}
\end{array} \tag{1}$$

where the unrestricted parameter ξ denotes an unknown inefficiency rate expressing the gap between the efficiency frontier and an empirical group of undesirable and desirable outputs. The parameter ϵ takes the value of 0.0001 in this work to minimize the influence of slack variables. If the restriction $\sum_{j=1}^{n} \lambda_j = 1$ is added to Model (1), then the obtained model is a VRS (Model (1*)).

The first restriction in equation systems ((1), (1*)) explores the values of λ_j to create a composite unit, considering inputs such as: $\sum_{j=1}^{n} x_{ij}\lambda_j = -d_i^x + x_{ik}\,, i = 1, \ldots, m.$ The values of the inputs can be decreased when the positive slack variables d_i^x are present. This may unquestionably vary the given rates, which implies that the system presents some inefficiencies.

In the same way, the second restriction, $\sum_{j=1}^{n} g_{rj}\lambda_j = d_r^g + \xi g_{rk} + g_{rk}\,, r = 1, \ldots, s$, indicates that the desirable outputs can be maintained or increased by making an increase of the slack variable d_r^g and a radial expansion ξg_{rk}.

The third restriction, $\sum_{j=1}^{n} b_{fj}\lambda_j = -d_f^b - \xi b_{fk} + b_{fk}\,, f = 1, \ldots, h$, shows the decrease of the inputs, and then, we could reduce the undesirable outputs both in their slack variables and radially.

The objective function considers that two origins of inefficiency may be established. A k-policy can be considered efficient when the following two conditions are met: (a) $\xi = 0$; (b) $d_i^x = 0$, $d_r^g = 0$, $d_f^b = 0$. In this case, the k-policy belongs to the efficiency frontier, since it fulfills the constraints present in equation systems ((1), (1*)), and consequently, the objective function takes a value of zero. Otherwise, the value of the objective function for non-efficient policies is greater than zero, due to possible displacements in the slack variables and radial movements.

The natural efficiency is then computed by:

$$\theta^* = 1 - \left[\xi^* + \epsilon \, (\sum_{i=1}^{m} R_i^x \, d_i^{x^*} + \sum_{r=1}^{s} R_r^g \, d_i^{g^*} + \sum_{f=1}^{h} R_f^b \, d_i^{b^*}) \right]$$

The value of this unified efficiency measure ranges between zero and one. If the k-policy is efficient, then the objective function of equation systems ((1), (1*)) is zero, and hence, the efficiency score equals $\theta^* = 1$. Slack variables resulting in the optimality of the models represented in equation systems ((1), (1*)) show the level of inefficiency.

2.2. *Managerial Disposability*

The managerial efficiency of the k-th policy is evaluated by the following CRS and VRS radial model [37]:

$$\begin{array}{rcl}
s.t. \quad \sum_{j=1}^{n} x_{ij}\lambda_j - d_i^x & = & x_{ik}\,, i = 1, \ldots, m\,, \\
\sum_{j=1}^{n} g_{rj}\lambda_j - d_r^g - \xi g_{rk} & = & g_{rk}\,, r = 1, \ldots, s\,, \\
\sum_{j=1}^{n} b_{fj}\lambda_j + d_f^b + \xi b_{fk} & = & b_{fk}\,, f = 1, \ldots, h\,, \\
d_i^x & \geq & 0\,, i = 1, \ldots, m\,, \\
d_r^g & \geq & 0\,, r = 1, \ldots, s\,, \\
d_f^b & \geq & 0\,, f = 1, \ldots, h\,, \\
\xi & & \text{Unrestricted}
\end{array} \tag{2}$$

Similarly, if the restriction $\sum_{j=1}^{n} \lambda_j = 1$ is added to Model (2), then the obtained model is a VRS (Model (2*)). In this model (2), increasing the inputs is allowed since new technologies that emit less CO_2 emissions to the atmosphere can be used.

By using the VRS models, we can obtain the returns to scale (RTS) and damage to scale (DTS) (see [37] for a better understanding). It is clear that for the natural efficiency, the returns to scale have to be increasing, and for managerial efficiency, the damages to scale have to be decreasing. Otherwise, the technical units are not working well and should correct the imbalances, using the information of the efficient units to which they have to be similar (peers).

3. Energy Policies for Data Centers at a Glance

The following set of energy efficiency policies for shutting down underutilized machines have been developed in this work as an evolution of those presented in [24], which have been adapted to the more complex reality of the cloud-computing paradigm:

- Never: prevents any shut-down process.
- Always: shuts down every server running in an idle state.
- Load: shuts down machines when data-center load pressure fails to reach a given threshold.
- Margin: assures that a determined number of machines are turned on and available before shutting down any machine.

- Random: shuts down machines randomly by means of a Bernoulli distribution with parameter 0.5.
- Exponential: shuts down machines when the probability of one incoming task negatively impacting on the data-center performance is lower than a given threshold. This probability is computed by means of the exponential distribution.
- Gamma: shuts down machines when the probability of incoming tasks oversubscribing to the available resources in a particular time period is lower than a given threshold; this probability is computed by means of the Gamma distribution.

4. Scheduling Models for Data Centers at a Glance

Cluster schedulers constitute a core part of cloud computing systems, since they are responsible for optimal task assignation to computing nodes. Several degrees of parallelism have been added to overcome the limitations present in central monolithic scheduling approaches when complex and heterogeneous systems with a high number of incoming jobs are considered. The following scheduling models are studied in this work:

- Monolithic: A centralized and single scheduler is responsible for scheduling all tasks in the workload in this model [44]. This scheduling approach may be the perfect choice when real-time responses are not required [45,46], since the omniscient algorithm performs high-quality task assignations by considering all restrictions and features of the data-center [47–50] at the cost of longer latency [46]. The scheduling process of a monolithic scheduler, such as that given by Google Borg [51], is illustrated in Figure 1.

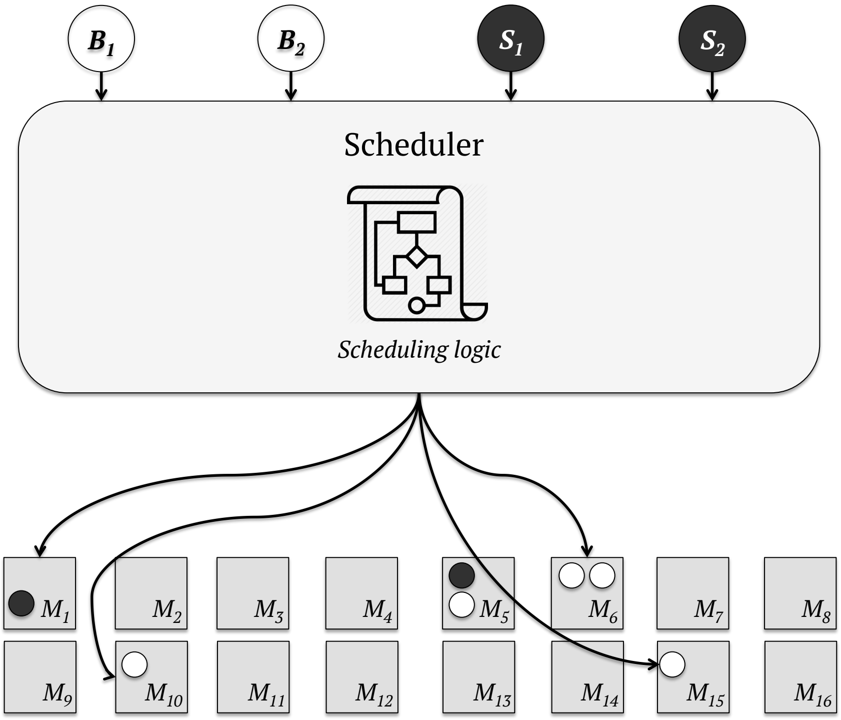


Figure 1. Monolithic scheduler architecture. M, worker node; S, service task; B, batch task [52].

- Two-level: This model achieves a higher level of parallelism by splitting the resource allocation and the task placement: a central manager blocks the whole cluster every time a scheduler makes a decision to offer computing resources to schedulers; and a set of parallel application-level schedulers performs the scheduling logic against the resources offered. This strategy enables the development of sub-optimal scheduling logic for each application, since the state of the data-center is not shared with the central manager, nor with the application schedulers. The workflow of the Two-level schedulers [53,54] is represented in Figure 2.

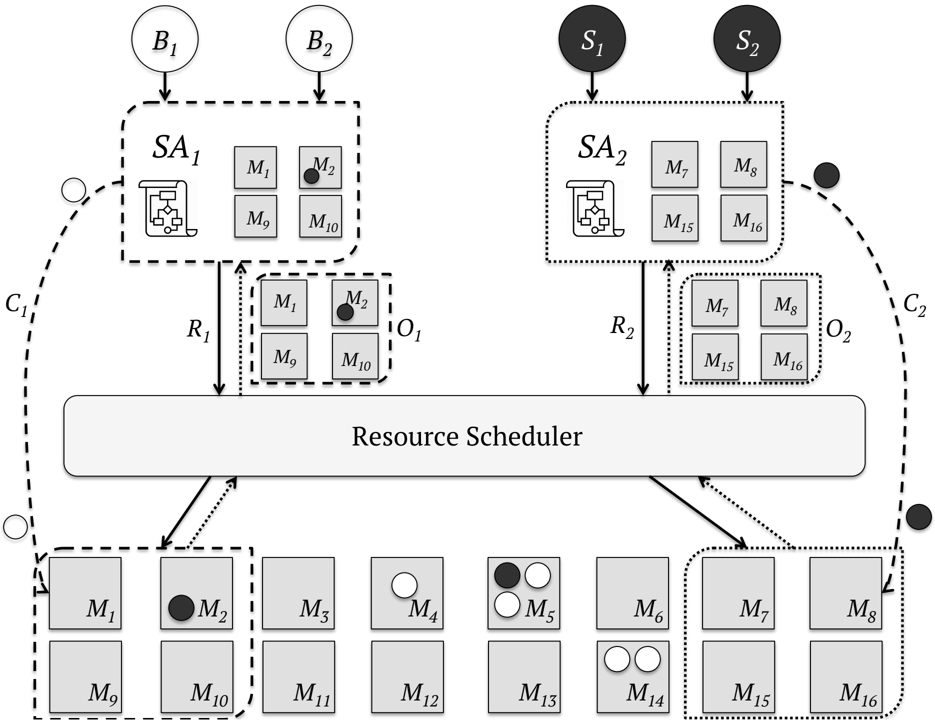


Figure 2. Two-level scheduler architecture. C, commit; O, resource offer; SA-, scheduler agent [52].

- Shared-state schedulers: On the other hand, in shared-state schedulers, such as Omega [55], the state of the data-center is available to all the schedulers. The central manager coordinates all the simultaneous parallel schedulers, which perform the scheduling logic against an out-of-date copy of the state of the data-center. The scheduling decisions are then committed to the central manager, which strives to apply these decisions. The utilization of stale views of the cluster by the schedulers can result in conflicts, since the chosen resources may not longer be available. In such a scenario, the local view of the state of the data-center stored in the scheduler is refreshed before the repetition of the scheduling process. The workflow of the shared-state scheduling model is represented in Figure 3.

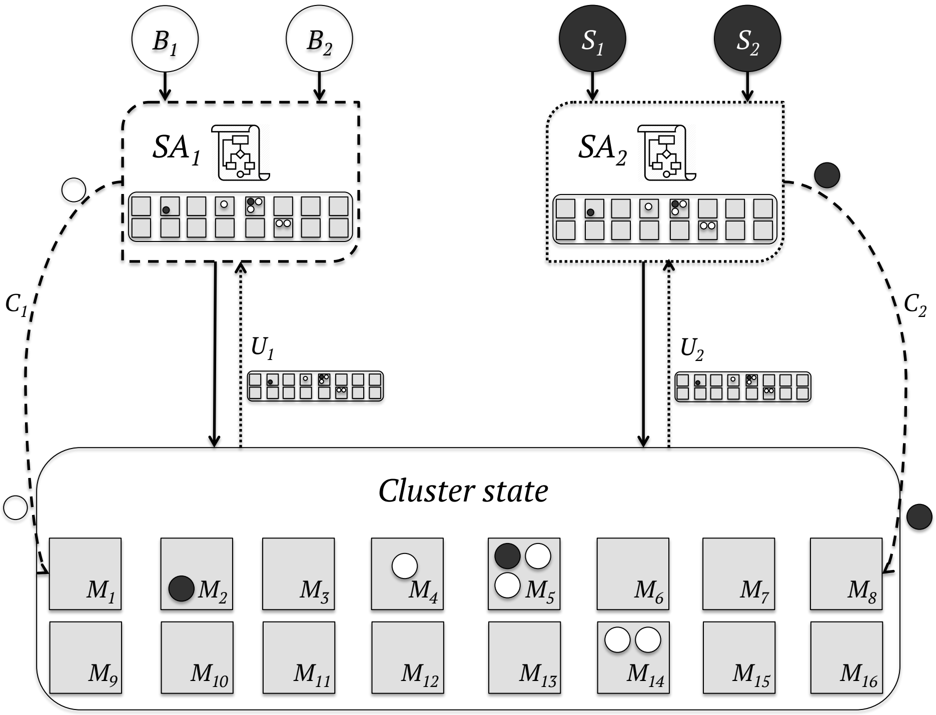


Figure 3. Shared-state scheduler architecture. U, cluster state update [52].

5. Methodology

In these next sections, the experimental environment designed for the implementation of the natural CRS DEA analysis is presented. The workflow followed in this work is shown in Figure 4.

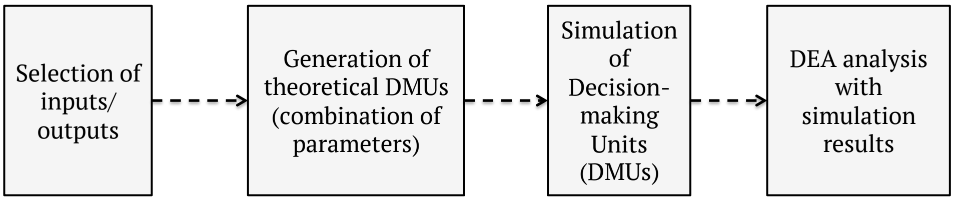

Selection of inputs/ outputs
Generation of theoretical DMUs (combination of parameters)
Simulation of Decision-making Units (DMUs)
DEA analysis with simulation results

Figure 4. Methodology workflow employed in this work. DEA, data envelopment analysis.

5.1. Simulation Tool

The SCORE simulator [52] is employed in this work, since simulation is the best alternative in scenarios where the implementation of the proposed strategies on real large-scale data-centers remains unfeasible. This simulator provides us with the tools for the development and application of the energy policies described in Section 3 and the scheduling models presented in Section 4 on realistic large-scale cloud computing systems.

5.2. Environment and DMU Definition

Following the trends presented in [56,57], two utilization environments have been simulated in this paper for seven days of operation:

- the low-utilization scenario, which represents highly over-provisioned infrastructures and achieves an average utilization of approximately 30%.
- the high-utilization scenario, which represents facilities of a more efficient nature that use approximately 65% of available resources on average.

These scenarios are applied to three data-center sizes: (a) Small: composed of 1000 computing servers; (b) Medium: composed of 5000 computing servers; and (c) Large: composed of 10,000 computing servers. Each server is equipped with four CPU cores and 8 GB of RAM.

Decision-making units (DMUs) are defined by the following elements: (a) an energy efficiency policy; (b) a scheduling model; and (c) a workload scenario.

5.3. Energy Model

The following states are presented for each resource in the energy model applied in this work: (a) Idle: when the machine is not executing tasks; and (b) Busy: otherwise.

Let t^i_{idle} represent the time the i-th resource is idle, and let t^i_{busy} denote the time during which the machine is computing tasks. In the same way, P^i_{idle} and P^i_{busy} represent the power required for the machines to run in these states, respectively.

The time a machine spends on executing a job may be defined as follows:

$$t^{ij}_{busy} = \max_{t \in Tasks^i} C_t \tag{3}$$

where $Tasks^{ij}$ represents the tasks of the j-th job assigned to M_i and C_t denotes the completion time of the t-th task of the j-th job.

In the same way, the total time a machine is executing tasks and the total time it is in an idle state may be defined as follows:

$$t^i_{busy} = \sum_{j=1}^{j} t^{ij}_{busy} \tag{4}$$

$$t^i_{idle} = t^i_{total} - t^i_{busy} \tag{5}$$

where t^i_{total} represents the total operation time. Therefore, we can express the energy consumption as follows:

$$\sum_{i=1}^{m} (P^i_{busy} * t^i_{busy} + P^i_{idle} * t^i_{idle}) \tag{6}$$

The considered power states, transitions and values for the energetic model are shown in Figure 5.

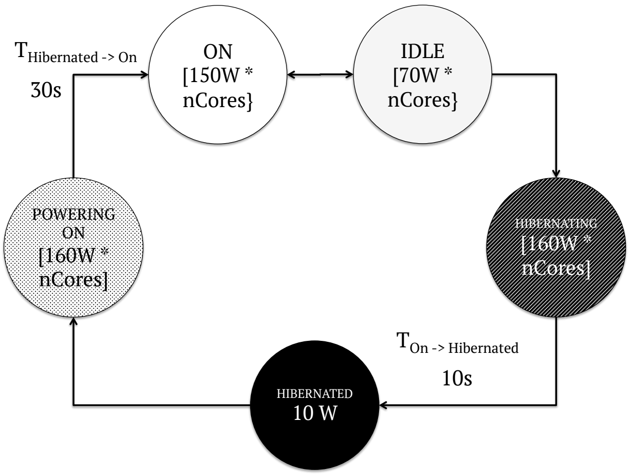


Figure 5. Machine power states [52].

5.4. DEA Inputs and Outputs

The inputs and outputs considered in DEA analysis and representative experimentation values are shown in Tables 1 and 2, respectively. One hundred and eight DMUs were analyzed, which were the result of the combination of all energy policies, scheduling models, data-center sizes and workload types described in Sections 3, 4 and 5.2, respectively. However, for clarity, a subset of the most interesting eighteen DMUs i shown in this paper. Each environment presents the following inputs and outputs:

- Inputs: Two inputs are considered in this work: (a) the number of machines in the data-center (D.C.), as shown in Section 5.2; and (b) the number of shut-down operations performed. These inputs may be reduced or kept equal.
- Outputs: One desirable output and two undesirable outputs are considered in this paper: (a) the time used to perform tasks' operations. The longer the time, the less idle the data-center. This good input can be maximized or kept equal; (b) the energy consumption of the data-center. The lower the energy consumption, the more efficient the data-center. This bad input may be reduced or kept equal; and (c) the average time jobs spend in a queue until they are scheduled. The shorter the time, the more performant the system is. This bad input may be reduced or kept equal.

Table 1. DEA inputs and outputs. Action column arrows mean whether the input/output value may be decreased (down arrow), increased (up arrow) or kept equal.

Parameter	Description	Action
Inputs		
Data-center size	Number of machines in the data-center	↓ ↔
#shut-downs	Number of shut-down operations	↓ ↔
Outputs		
Computation time	Total amount of useful task computation	↑ ↔
Energy consumption	Total data-center energy consumption	↓ ↔
Queue time	Average time until jobs are fully scheduled	↓ ↔

Table 2. Sample from the dataset for DEA analysis. The full dataset showing the results for the 108 DMUs analyzed can be found as the Supplementary Material. Energy policies, scheduling models, data-center sizes and workload types can be found in Sections 3, 4 and 5.2, respectively. D.C., data-center.

DMU			Inputs			Outputs	
Energy Policy	**Scheduling Model**	**Work-Load**	**D.C. Size**	**#Shut-Downs**	**Computing Time (h)**	**MWh Consumed**	**Queue Time (ms)**
Always	Monolithic	High	1000	37,166	104.42	49.01	90.10
Margin	Mesos	High	1000	13,361	104.26	49.65	1093.00
Gamma	Omega	High	1000	14,252	104.17	49.60	0.10
Always	Mono.	Low	1000	36,404	49.25	23.92	78.30
Exponential	Mesos	Low	1000	19,671	49.63	24.65	1188.70
Load	Omega	Low	1000	32,407	49.34	24.19	1.10
Margin	Mono.	High	5000	6981	99.96	237.09	126.20
Gamma	Mono.	High	5000	9877	99.96	235.92	129.80
Random	Mesos	High	5000	33,589	100.03	234.90	1122.60
Margin	Omega	High	5000	8578	100.26	239.13	0.70
Exponential	Omega	High	5000	11,863	100.26	236.95	1.00
Margin	Omega	Low	5000	15,452	46.70	115.82	0.50
Margin	Mono.	High	10,000	9680	101.56	481.36	325.20
Gamma	Mono.	High	10,000	11,388	101.56	479.36	327.90
Margin	Omega	High	10,000	18,150	101.63	486.11	2.60
Gamma	Omega	High	10,000	18,409	101.63	484.69	2.50
Gamma	Mesos	Low	10,000	29,707	45.83	228.31	1107.60
Random	Omega	Low	10,000	40,772	46.09	233.50	3.80

6. Natural CRS DEA Results

The whole dataset included as an Appendix is analyzed by means of natural CRS and VRS DEA. However, only the most relevant natural CRS DEA results for the most representative DMUs, which are presented in Table 2, are described in this section.

An efficiency analysis depending on the data-center size and on the energy policy is shown in Tables 3 and 4. The following conclusions can be drawn:

- The best efficiency levels are achieved for small data-centers. The data-center size input is predominant in this group of DMUs, since no major differences between energy policies, scheduling frameworks and workload scenarios are present (σ = 0.01, $\overline{x}$ = 0.99).
- Mid-size data-centers should use the margin energy policy and monolithic or Omega schedulers and should avoid all other energy policies and the Mesos scheduler. Moreover, high workload scenarios are also more efficient than low workload scenarios. In addition, the following DMUs achieve a good level of efficiency, but they do not belong to the efficiency frontier: (a) the DMU combining the Gamma energy policy and the monolithic or Omega schedulers; (b) the DMU combining the exponential energy policy and the Omega scheduler.
- No DMU is efficient in large-scale data-centers. However, the following DMUs present good levels of efficiency: (a) the DMUs combining the Gamma, exponential or margin energy policy with the high workload scenario and the monolithic scheduler; and (b) the DMUs combining the Gamma or margin energy policy with the high workload scenario and the Omega scheduler.
- In high-loaded scenarios, the monolithic scheduler presents the lowest deviation regardless of the data-center size (σ = 0.32).

We can determine that it is always inefficient to operate in a low utilization scenario in medium-sized and large data-centers. Moreover, both the margin and the probabilistic energy policies (Gamma and exponential) perform more efficiently than the rest of the energy policies, as shown in Figure 6. The monolithic scheduler seems to achieve good results even for large-scale data-centers, while the two-level scheduling approach has a negative impact on data-center performance. However, the trends show that the performance of the monolithic scheduling approach suffers from degradation on larger data-centers and higher workload pressure, and hence, lower efficiency levels are to be expected if larger sizes and higher utilization scenarios are to be considered.

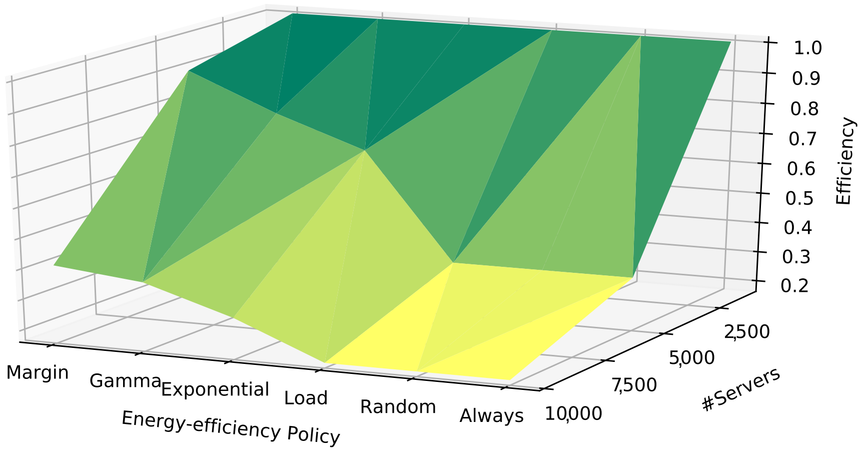


Figure 6. Summary of DEA natural constant returns to scale (CRS) efficiency results for energy efficiency policies.

The actions proposed for the improvement of efficiency of the most relevant DMUs are shown in Table 5.

Table 3. Efficiency analysis for data-center sizes.

Scheduling	**Workload**	**Data-Center Size**			**Efficiency**	
Model	**Scenario**	**1000**	**5000**	**10,000**	σ	$\overline{x}$
Monolithic	High	1.00	0.60	0.37	0.32	0.66
Monolithic	Low	0.98	0.33	0.18	0.43	0.49
Mesos	High	1.00	0.47	0.18	0.41	0.55
Mesos	Low	0.97	0.32	0.17	0.43	0.49
Omega	High	1.00	0.62	0.27	0.36	0.63
Omega	Low	0.97	0.32	0.17	0.43	0.49
	σ	0.01	0.14	0.08		
	$\overline{x}$	0.99	0.44	0.23		
					0.40	0.55

Table 4. Efficiency analysis of energy policies.

	Scheduling Model										
Energy	**Monolithic**			**Mesos**			**Omega**			**Efficiency**	
Policy	**1000**	**5000**	**10,000**	**1000**	**5000**	**10,000**	**1000**	**5000**	**10,000**	σ	$\overline{x}$
Always	0.99	0.33	0.18	0.99	0.33	0.18	0.99	0.33	0.18	0.37	0.50
Random	0.99	0.33	0.18	0.98	0.32	0.18	0.98	0.33	0.18	0.37	0.50
Load	0.99	0.33	0.18	0.99	0.33	0.18	0.99	0.33	0.18	0.37	0.50
Margin	0.99	0.66	0.42	0.99	0.53	0.18	0.99	0.66	0.30	0.31	0.63
Exp.	0.99	0.54	0.31	0.99	0.40	0.18	0.99	0.58	0.22	0.33	0.58
Gamma	0.99	0.58	0.38	0.98	0.47	0.18	0.99	0.61	0.29	0.31	0.61

6.1. Proposed Corrections for a Sample DMU

DMU #104 is selected to illustrate how corrective actions are proposed by DEA in order to achieve efficiency. This DMU is defined by the combination of the random energy efficiency policy, the Omega scheduling model and a low utilization workload scenario.

DMU #104 presents a natural efficiency of 0.1697. This means it is far from being efficient. The following corrective actions are suggested for it to belong to the efficiency frontier, as shown in Table 6:

- The time the data-center spends on task computation must be increased by 38.28 h (+83%).
- Energy consumption must be reduced by 193.88 MWh (−83%).
- The average time jobs wait in a queue must be reduced by 3.23 s (−83%).
- The number of servers must be reduced by 9190 (−92%).
- Shut-down operations must be reduced by 9680 (−24%).

In addition to these corrective actions, the peers this DMU should emulate are #13, #34 and #18. This means that the workload must be increased, and better energy efficiency policies, such as margin and always, must be used. The full dataset containing all the DMUs and DEA analysis and corrections can be found as Supplementary Material in the Appendix.

Some of the proposed changes involve the switching of the scheduling framework, which is hardly achievable with the current resource manager systems. To implement these corrections, a resource managing system able to dynamically change the scheduling framework during runtime would be necessary. Such a system is an interesting improvement to the current state of the art that the DEA analysis leads us to develop.

Table 5. Resulting proposed corrections following DEA analysis. Peer projections for a DMU indicate which DMU it should emulate. The following actions may be taken for each input and output: ↑ when the parameter must be increased; ↓ if the parameter must be reduced; and ↔ if no further actions are needed to achieve efficiency.

DMU				Peer	Corrections				
#	Energy Policy	Sched. Model	Work-load	Projec-tions	D.C. Size	#Shut-downs	Comp. Time	Energy Cons.	Queue Time
1	Always	Mono.	High	↔	↔	↔	↔	↔	↔
10	Margin	Mesos	High	4 (88%)	↓	↔	↑	↓	↓
18	Gamma	Omega	High	↔	↔	↔	↔	↔	↔
19	Always	Mono.	Low	↔	↓	↓	↑	↓	↓
29	Exp.	Mesos	Low	23 (56%) 22 (48%)	↓	↓	↑	↓	↓
33	Load	Omega	Low	31 (100%)	↓	↓	↑	↓	↔
40	Margin	Mono.	High	↔	↔	↔	↔	↔	↔
42	Gamma	Mono.	High	6 (59%) 41 (41%)	↓	↔	↑	↓	↓
44	Random	Mesos	High	7 (100%)	↓	↔	↑	↓	↓
52	Margin	Omega	High	↔	↔	↔	↔	↔	↔
53	Exp.	Omega	High	16 (63%) 52 (36%)	↓	↔	↑	↓	↓
70	Margin	Omega	Low	18 (72%)	↓	↓	↑	↓	↓
76	Margin	Mono.	High	6 (55%) 40 (45%)	↓	↔	↑	↓	↓
78	Gamma	Mono.	High	6 (90%)	↓	↔	↑	↓	↓
88	Margin	Omega	High	18 (95%)	↓	↔	↑	↓	↓
90	Gamma	Omega	High	18 (96%)	↓	↔	↑	↓	↓
102	Gamma	Mesos	Low	1 (49%) 22 (38%)	↓	↔	↑	↓	↓
104	Random	Omega	Low	13 (53%) 34 (36%)	↓	↓	↑	↓	↓

7. Conclusions and Policy Implications

In this work, we have confirmed the hypothesis that DEA constitutes a powerful tool for the analysis of technical efficiency in cloud-computing scenarios where large-scale data-centers provide the computational core.

Data envelopment analysis provides cloud-computing operators with the means for the identification of which data-center configuration better suits their requirements, both in terms of performance and energy efficiency.

This methodology allows us to analyze several energy efficiency policies that shut down idle servers, so that their behavior and differences can be compared in various data-center environments. It has been proven that policies based on a security margin and those that use statistical tools to predict the future workload, such as exponential and Gamma, deliver better results than policies based on data-center workload pressure and random strategies.

In addition, it has been empirically shown that even under medium and high workload pressure, in data-centers composed of up to 10,000 machines, monolithic schedulers perform better than other scheduling models, such as the two-level and shared-state approaches.

Table 6. Corrections proposed for DMU #104.

<table>
<tr><th colspan="6">Results for DMU #104</th></tr>
<tr><td colspan="6">Natural Efficiency = 0.1697</td></tr>
<tr><td colspan="6">Projection Summary:</td></tr>
<tr><th colspan="2">Variable</th><th>Original Value</th><th>Radial Movement</th><th>Slack Movement</th><th>Projected Value</th></tr>
<tr><td>Output</td><td>Computation (h)</td><td>46.09</td><td>+83%</td><td>0</td><td>84.37</td></tr>
<tr><td>Output</td><td>MWh consumed</td><td>233.50</td><td>−83%</td><td>0</td><td>39.62</td></tr>
<tr><td>Output</td><td>Queue time (ms)</td><td>3.80</td><td>−83%</td><td>0</td><td>0.6</td></tr>
<tr><td>Input</td><td>#Servers</td><td>10,000</td><td>0</td><td>−9190</td><td>810</td></tr>
<tr><td>Input</td><td>#Shut-downs</td><td>40,772</td><td>0</td><td>−9680</td><td>31,092</td></tr>
<tr><td colspan="6">Listing of Peers:</td></tr>
<tr><th colspan="3">Peer</th><th colspan="3">Lambda Weight</th></tr>
<tr><td colspan="3">#13</td><td colspan="3">53%</td></tr>
<tr><td colspan="3">#34</td><td colspan="3">36%</td></tr>
<tr><td colspan="3">#18</td><td colspan="3">11%</td></tr>
</table>

Finally, cloud-computing infrastructure managers are provided with empirical knowledge of which data-centers are not being used optimally, and hence, they can make decisions regarding the shut-down of machines in order to achieve higher utilization levels of the cloud-computing system as a whole.

As future work related to the limitations of the presented work, we may include:

- The addition of different kind of workload patterns, as well as real workload traces.
- The analysis of other scheduling models, such as distributed and hybrid models.
- The development of a new-generation resource-managing system that could dynamically apply the optimal scheduling framework depending on the environment and workload.
- The analysis of simulation data with other DEA approaches, such as Bayesian and probabilistic models, which could minimize the impact of the noise in current DEA models.

Supplementary Materials: Supplementary Materials are available online at http://www.mdpi.com/1996-1073/11//2053/s1.

Author Contributions: D.F.-C. and A.F.-M. conceived of and designed the experiments. D.F.-C. performed the experiments. D.F.-C., A.F.-M. and V.F. analyzed the data. D.F.-C. and F.V. contributed reagents/materials/analysis tools. D.F.-C., A.F.-M. and F.V. wrote the paper.

Funding: This research was funded by VI Plan Propio de Investigación y Transferencia—University of Seville 2018 grant number 2018/00000520.

Acknowledgments: The research is supported by the VI Plan Propio de Investigación y Transferencia (VIPPI), University of Seville.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Koomey, J. *Growth in Data Center Electricity Use 2005 to 2010*; Analytical Press: Piedmont, CA, USA, 1 August 2011.
2. Van Heddeghem, W.; Lambert, S.; Lannoo, B.; Colle, D.; Pickavet, M.; Demeester, P. Trends in worldwide ICT electricity consumption from 2007 to 2012. *Comput. Commun.* **2014**, *50*, 64–76. [CrossRef]
3. Gómez-López, M.T.; Gasca, R.M.; Pérez-Álvarez, J.M. Decision-Making Support for the Correctness of Input Data at Runtime in Business Processes.*Int. J. Cooper. Inf. Syst.* **2014**, *23*. [CrossRef]
4. Amirteimoori, A.; Emrouznejad, A. Optimal input/output reduction in production processes. *Decis. Support Syst.* **2012**, *52*, 742–747. [CrossRef]

5. Chiang, K.; Hwang, S.N. Efficiency measurement for network systems IT impact on firm performance. *Decis. Support Syst.* **2010**, *48*, 437–446.
6. Chang, Y.T.; Zhang, N.; Danao, D.; Zhang, N. Environmental efficiency analysis of transportation system in China: A non-radial DEA approach. *Energy Policy* **2013**, *58*, 277–283. [CrossRef]
7. Arcos-Vargas, A.; Núñez-Hernández, F.; Villa-Caro, G. A DEA analysis of electricity distribution in Spain: An industrial policy recommendation. *Energy Policy* **2017**, *102*, 583–592. [CrossRef]
8. Gonzalez-Rodriguez, M.; Velasco-Morente, F.; González-Abril, L. La eficiencia del sistema de protección social español en la reducción de la pobreza. *Papeles de Población* **2010**, *16*, 123–154.
9. Afonso, A.; Schuknecht, L.; Tanzi, V. Public sector efficiency: Evidence for new EU member states and emerging markets. *Appl. Econ.* **2010**, *42*, 2147–2164. [CrossRef]
10. Jakóbik, A.; Grzonka, D.; Kolodziej, J.; Chis, A.E.; González-Vélez, H. Energy Efficient Scheduling Methods for Computational Grids and Clouds. *J. Telecommun. Inf. Technol.* **2017**, *1*, 56–64.
11. Sharma, R.K.; Bash, C.E.; Patel, C.D.; Friedrich, R.J.; Chase, J.S. Balance of power: Dynamic thermal management for internet data centers. *IEEE Internet Comput.* **2005**, *9*, 42–49. [CrossRef]
12. El-Sayed, N.; Stefanovici, I.A.; Amvrosiadis, G.; Hwang, A.A.; Schroeder, B. Temperature management in data-centers: Why some (might) like it hot. In Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems, London, UK, 11–15 June 2012; pp. 163–174.
13. Miyoshi, A.; Lefurgy, C.; Van Hensbergen, E.; Rajamony, R.; Rajkumar, R. Critical power slope: Understanding the runtime effects of frequency scaling. In Proceedings of the 16th International Conference on Supercomputing, New York, NY, USA, 22–26 June 2016; pp. 35–44.
14. Fan, X.; Weber, W.D.; Barroso, L.A. Power provisioning for a warehouse-sized computer. In Proceedings of the 34th Annual International Symposium on Computer Architecture, San Diego, CA, USA, 9–13 June 2007; pp. 13–23.
15. Andersen, D.G.; Swanson, S. Rethinking flash in the data-center. *IEEE Micro* **2010**, *30*, 52–54. [CrossRef]
16. Femal, M.E.; Freeh, V.W. Boosting data-center performance through non-uniform power allocation. In Proceedings of the Second International Conference on Autonomic Computing (ICAC'05), Seattle, WA, USA, 13–16 June 2005, doi:10.1109/ICAC.2005.17.
17. Jakóbik, A.; Grzonka, D.; Kołodziej, J. Security supportive energy aware scheduling and scaling for cloud environments. In Proceedings of the 31st European Conference on Modelling and Simulation (ECMS 2017), Budapest, Hungary, 23–26 May 2017; pp. 583–590.
18. Juarez, F.; Ejarque, J.; Badia, R.M. Dynamic energy-aware scheduling for parallel task-based application in cloud computing. *Future Gener. Comput. Syst.* **2018**, *78*, 257–271. [CrossRef]
19. Lee, Y.C.; Zomaya, A.Y. Energy efficient utilization of resources in cloud computing systems. *J. Supercomput.* **2012**, *60*, 268–280. [CrossRef]
20. Sohrabi, S.; Tang, A.; Moser, I.; Aleti, A. Adaptive virtual machine migration mechanism for energy efficiency. In Proceedings of the 5th International Workshop on Green and Sustainable Software, Austin, TX, USA, 14–22 May 2016; pp. 8–14.
21. Beloglazov, A.; Buyya, R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data-centers. *Concurr. Comp.-Pract. E* **2012**, *24*, 1397–1420. [CrossRef]
22. Kaushik, R.T.; Bhandarkar, M. Greenhdfs: Towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster. In Proceedings of the 2010 International Conference on Power Aware Computing and Systems, Vancouver, BC, Canada, 3–6 October 2010; pp. 1–9.
23. Luo, X.; Wang, Y.; Zhang, Z.; Wang, H. Superset: A non-uniform replica placement strategy towards high-performance and cost-effective distributed storage service. In Proceedings of the 2013 International Conference on Advanced Cloud and Big Data, Nanjing, China, 13–15 December 2013.
24. Fernández-Montes, A.; Gonzalez-Abril, L.; Ortega, J.A.; Lefèvre, L. Smart scheduling for saving energy in grid computing. *Expert Syst. Appl.* **2012**, *39*, 9443–9450. [CrossRef]
25. Fernández-Montes, A.; Fernández-Cerero, D.; González-Abril, L.; Álvarez-García, J.A.; Ortega, J.A. Energy wasting at internet data-centers due to fear. *Pattern Recogn. Lett.* **2015**, *67*, 59–65. [CrossRef]
26. Farrell, M.J. The measurement of productive efficiency. *J. R. Stat. Soc. Ser. A (Gen.)* **1957**, *120*, 253–290. [CrossRef]

27. Charnes, A.; Cooper, W.W.; Rhodes, E. Measuring the efficiency of decision making units. *Eur. J. Oper. Res.* **1978**, *2*, 429–444. [CrossRef]
28. Banker, R.D.; Charnes, A.; Cooper, W.W. Some Models for Estimating Technical and Scale Inefficiencies in Data Envelopment Analysis. *Manag. Sci.* **1984**, *30*, 1078–1092. [CrossRef]
29. Fernández-Montes, A.; Velasco, F.; Ortega, J. Evaluating decision-making performance in a grid-computing environment using DEA. *Expert Syst. Appl.* **2012**, *39*, 12061–12070. [CrossRef]
30. Campos, M.; Fernández-Montes, A.; Gavilan, J.; Velasco, F. Public resource usage in health systems: A data envelopment analysis of the efficiency of health systems of autonomous communities in Spain. *Public Health* **2016**, *138*, 33–40. [CrossRef] [PubMed]
31. Fernández-Serrano, J.; Berbegal, V.; Velasco, F.; Expósito, A. Efficient entrepreneurial culture: A cross-country analysis of developed countries. *Int. Entrep. Manag. J.* **2017**, *14*, 105–127. [CrossRef]
32. Exposito, A.; Velasco, F. Municipal solid-waste recycling market and the European 2020 Horizon Strategy: A regional efficiency analysis in Spain. *J. Clean. Prod.* **2018**, *172*, 938–948. [CrossRef]
33. Scheel, H. Undesirable outputs in efficiency valuations. *Eur. J Oper. Res.* **2001**, *132*, 400–410. [CrossRef]
34. Färe, R.; Grosskopf, S.; Hernandez-Sancho, F. Environmental performance: An index number approach. *Resour. Energy Econ.* **2004**, *26*, 343–352. [CrossRef]
35. Zhou, P.; Ang, B.W.; Poh, K.L. Measuring environmental performance under different environmental DEA technologies. *Energy Econ.* **2008**, *30*, 1–14. [CrossRef]
36. Sueyoshi, T.; Goto, M. Returns to scale and damages to scale on US fossil fuel power plants: Radial and non-radial approaches for DEA environmental assessment. *Energy Econ.* **2012**, *34*, 2240–2259. [CrossRef]
37. Sueyoshi, T.; Goto, M. DEA radial measurement for environmental assessment: A comparative study between Japanese chemical and pharmaceutical firms. *Appl. Energy* **2014**, *115*, 502–513. [CrossRef]
38. Halkos, G.E.; Tzeremes, N.G. Measuring the effect of Kyoto protocol agreement on countries' environmental efficiency in CO_2 emissions: An application of conditional full frontiers. *J. Prod. Anal.* **2014**, *41*, 367–382. [CrossRef]
39. Sanz-Díaz, M.T.; Velasco-Morente, F.; Yñiguez, R.; Díaz-Calleja, E. An analysis of Spain's global and environmental efficiency from a European Union perspective. *Energy Policy* **2017**, *104*, 183–193. [CrossRef]
40. Vlontzos, G.; Pardalos, P. Assess and prognosticate green house gas emissions from agricultural production of EU countries, by implementing, DEA Window analysis and artificial neural networks. *Renew. Sustain. Energy Rev.* **2017**, *76*, 155–162. [CrossRef]
41. Yu, S.H.; Gao, Y.; Shiue, Y.C. A Comprehensive Evaluation of Sustainable Development Ability and Pathway for Major Cities in China. *Sustainability* **2017**, *9*, 1483. [CrossRef]
42. Dios-Palomares, R.; Alcaide, D.; Pérrez, J.D.; Bello, M.J.; Prieto, A.; Zúniga, C.A. The Environmental Efficiency using Data Envelopment Analysis: Empirical methods and evidences. In *The stated of the Art for Bieconomic and Climate Change*; Editorial Universitaria UNAN Leon, Ed.; Red de Bioeconomia y Cambio Climático: Cordoba, Spain, 2017; p. 48.
43. Cooper, W.W.; Park, K.S.; Pastor, J.T. RAM: A range adjusted measure of inefficiency for use with additive models, and relations to other models and measures in DEA. *J. Prod. Anal.* **1999**, *11*, 5–42. [CrossRef]
44. Delimitrou, C.; Kozyrakis, C. Paragon: QoS-aware scheduling for heterogeneous datacenters. In Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems, Houston, TX, USA, 16–20 March 2013; pp. 77–88.
45. Isard, M.; Prabhakaran, V.; Currey, J.; Wieder, U.; Talwar, K.; Goldberg, A. Quincy: Fair scheduling for distributed computing clusters. In Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles, Big Sky, MT, USA, 11–14 October 2009; pp. 261–276.
46. Delimitrou, C.; Sanchez, D.; Kozyrakis, C. Tarcil: Reconciling scheduling speed and quality in large shared clusters. In Proceedings of the Sixth ACM Symposium on Cloud Computing, Kohala Coast, HI, USA, 27–29 August 2015; pp. 97–110.
47. Grandl, R.; Ananthanarayanan, G.; Kandula, S.; Rao, S.; Akella, A. Multi-resource packing for cluster schedulers. *ACM SIGCOMM Comput. Commun. Rev.* **2015**, *44*, 455–466. [CrossRef]
48. Zaharia, M.; Borthakur, D.; Sen Sarma, J.; Elmeleegy, K.; Shenker, S.; Stoica, I. Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling. In Proceedings of the 5th European Conference on Computer systems, Paris, France, 13–16 April 2010; pp. 265–278.

49. Delimitrou, C.; Kozyrakis, C. Quasar: Resource-efficient and QoS-aware cluster management. In Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems, Salt Lake City, UT, USA, 1–5 March 2014; pp. 127–144.
50. Zhang, X.; Tune, E.; Hagmann, R.; Jnagal, R.; Gokhale, V.; Wilkes, J. CPI 2: CPU performance isolation for shared compute clusters. In Proceedings of the 8th ACM European Conference on Computer Systems, Prague, The Czech Republic, 15–17 April 2013; pp. 379–391.
51. Verma, A.; Pedrosa, L.; Korupolu, M.; Oppenheimer, D.; Tune, E.; Wilkes, J. Large-scale cluster management at Google with Borg. In Proceedings of the Tenth European Conference on Computer Systems, Bordeaux, France, 21–24 April 2015; p. 18. [CrossRef]
52. Fernández-Cerero, D.; Fernández-Montes, A.; Jakóbik, A.; Kołodziej, J.; Toro, M. SCORE: Simulator for cloud optimization of resources and energy consumption. *Simul. Model. Pract. Th.* **2018**, *82*, 160–173. [CrossRef]
53. Hindman, B.; Konwinski, A.; Zaharia, M.; Ghodsi, A.; Joseph, A.D.; Katz, R.H.; Shenker, S.; Stoica, I. Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center. In Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation, Boston, MA, USA, 30 March–1 April 2011; pp. 295–308.
54. Vavilapalli, V.K.; Murthy, A.C.; Douglas, C.; Agarwal, S.; Konar, M.; Evans, R.; Graves, T.; Lowe, J.; Shah, H.; Seth, S.; *et al*. Apache hadoop yarn: Yet another resource negotiator. In Proceedings of the 4th Annual Symposium on Cloud Computing, Santa Clara, CA, USA, 1–3 October 2013; p. 5.
55. Schwarzkopf, M.; Konwinski, A.; Abd-El-Malek, M.; Wilkes, J. Omega: Flexible, scalable schedulers for large compute clusters. In Proceedings of the 8th ACM European Conference on Computer Systems, Prague, The Czech Republic, 15–17 April 2013; pp. 351–364.
56. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; et al. A view of cloud computing. *Commun. ACM* **2010**, *53*, 50–58. [CrossRef]
57. Ruth, S. Reducing ICT-related carbon emissions: An exemplar for global energy policy? *IETE Tech. Rev.* **2011**, *28*, 207–211. [CrossRef]

PART III

Final remarks

CHAPTER 9

CONCLUSIONS AND FUTURE WORK

The Earth is the cradle of humanity, but mankind cannot stay in the cradle forever

Konstantin Tsiolkovsky

9.1 Conclusions

This thesis dissertation focuses on the problem resource efficiency in data centers, from both the energy-efficiency and the performance points of view. Nowadays, such a topic is critical, since huge-scale data-center energy efficiency impacts, not only to the economic balance of large companies worldwide, but on our environment in a moment where global warming is worsening.

Moreover, this thesis dissertation explores and utilizes several models to accomplish the aforementioned ambitious and complex goals, such as: a) energy-policies which shut-down idle machines; b) complex energy-aware scheduling algorithms; c) models based on games theory; and d) DEA productive efficiency analysis.

It has been proved that highly-utilized realistic large-scale cloud-computing clusters can cut down their electricity consumption by more than 20% when the proposed models are employed. The presented results encourage data-center administrators to employ not only efficiency policies related to hardware and cooling, but software solutions to achieve energy proportionality.

Furthermore, the negative impact of the application of such models is not significant in comparison to the energy consumption reduction, and the related economic and environmental costs.

Simulation tools have been developed in order to analyze energy consumption and performance at large-scale cloud-computing data centers, whereby several energy-saving, scheduling algorithms and resource managers have been studied. This tool has been widely tested in order to obtain reliable results, and has been published and shared within the European network **COST Action IC1406: High-Performance Modelling and Simulation for Big Data Applications (cHiPSet)**.

9.2 Future work

This thesis dissertation has led to new research interests and collaborations which will be explored in the future, including:

– Dynamic management of resource managers depending on operational and workload behaviour. This research line is being currently explored, and as a first result of this work we published an international conference paper with the colleagues of Lyon and Cracow entitled ”Quality of cloud services determined by the dynamic management of scheduling models for complex heterogeneous workloads”.

- Extensive analysis of the proposed models in centralized Two-level and Shared-state resource managers.
- Adaptation of the proposed models to non-centralized resource managers, such as distributed and hybrid schedulers.
- Development of new energy-efficiency policies focused on the shut-down of idle machines based on artificial intelligence, such as Support Vector Machines (SVM) and Artificial Neural Networks (ANN).
- Adaptation of the proposed models to federated clouds which are usually employed in fog computing and Internet of the Things (IoT) scenarios.
- Development of more complex energy-aware operation models based on games theory to efficiently manage the concurrency between scheduling agents, scheduling algorithms and resource managers.

APPENDIX A

CURRICULUM

A.1 Research papers

A.1.1 JCR Indexed Journals

1. Title: **Energy wasting at internet data centers due to fear.**

   Authors: **Alejandro Fernández-Montes, Damián Fernández-Cerero, Luis González-Abril, Juan Antonio Álvarez-García, and Juan Antonio Ortega.**

   Published in: **Pattern Recognition Letters**, December 2015, Volume 67, Part 1, 2015, Pages 59-65, ISSN: 0167-8655,

   DOI: 10.1016/j.patrec.2015.06.018,

   Q2. JCR-2015 IF:1.586.

2. Title: **SCORE: Simulator for cloud optimization of resources and energy consumption.**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, Agnieszka Jakóbik, Joanna Kołodziej and Miguel Toro.**

Published in: **Simulation Modelling Practice and Theory**, March 2018, Volume 82, 2018, Pages 160-173, ISSN 1569-190X,

DOI: 10.1016/j.simpat.2018.01.004,

Q1. JCR-2017 IF:2.092.

3. Title: **Energy policies for data-center monolithic schedulers.**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, and Juan A. Ortega.**

   Published in: **Expert Systems with Applications**, November 2018, Volume 110, 2018, Pages 170-181, ISSN 0957-4174,

   DOI: 10.1016/j.eswa.2018.06.007,

   Q1. JCR-2017 IF:3.768.

4. Title: **Security supportive energy-aware scheduling and energy policies for cloud environments.**

   Authors: **Damián Fernández-Cerero, Agnieszka Jakóbik, Daniel Grzonka, Joanna Kołodziej and Alejandro Fernández-Montes.**

   Published in: **Journal of Parallel and Distributed Computing**, September 2018, Volume 119, 2018, Pages 191-202, ISSN 0743-7315.

   DOI: 10.1016/j.jpdc.2018.04.015,

   Q2. JCR-2017 IF: 1.815.

5. Title: **GAME-SCORE: Game-based energy-aware cloud scheduler and simulator for computational clouds.**

   Authors: **Damián Fernández-Cerero, Agnieszka Jakóbik, Alejandro Fernández-Montes, and Joanna Kołodziej**

   Published in: **Simulation Modelling Practice and Theory**, In Press, ISSN 1569-190X,

   DOI: 10.1016/j.simpat.2018.09.001,

   Q1. JCR-2017 IF: 2.092.

6. Title: **Productive Efficiency of Energy-Aware Data Centers.**

Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, and Francisco Velasco**

Published in: **Energies**, August 2018, Volume 11, 2018, ISSN 1996-1073, DOI: 10.3390/en11082053,

Q2. JCR-2017 IF: 2.676.

A.1.2 Other Journals

1. Title: **BlockChain Secure Cloud: A new generation Integrated Cloud and BlockChain Platforms —General Concepts and Challenges.**

   Authors: **Joanna Kołodziej, Andrzej Wilzyński, Damián Fernández-Cerero, and Alejandro Fernández-Montes.**

   Published in: **European CyberSecurity Journal**, Volume 4, Issue 2, September 2018. ISSN: 2450-21113. Pages: 28-35.

A.1.3 International Conferences

1. Title: **Fear Assessment: Why data center servers should be turned off.**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, Luis González Abril, Juan Antonio Ortega, Juan A. Álvarez.**

   Published in: **17th International Conference of the Catalan Association for Artificial Intelligence (CCIA 2014)**, Barcelona, Spain, October 22-24, 2014. Frontiers in Artificial Intelligence and Applications 269, IOS Press 2014, ISBN 978-1-61499-451-0, Pages: 253-256.

2. Title: **Creating Virtual Humans with Game Engines for Evaluate Ambient Assisted Living Scenarios.**

   Authors: **Manuel Sánchez Palacios, Juan Antonio Álvarez-García, Luis Miguel Soria-Morillo, Damián Fernández-Cerero.**

Published in: **Ambient Intelligence - Software and Applications - 7th International Symposium on Ambient Intelligence, ISAmI 2016**, Seville, Spain, June 1-3, 2016. Advances in Intelligent Systems and Computing 476, Springer 2016, ISBN 978-3-319-40113-3. Pages: 105-112.

3. Title: **Stackelberg Game-Based Models In Energy-Aware Cloud Scheduling.**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, Agnieszka Jakóbik, Joanna Kołodziej.**

   Published in: **32nd European Conference on Modelling and Simulation, ECMS 2018.**, Wilhelmshaven, Germany, May 22-25, 2018, Proceedings. European Council for Modeling and Simulation 2018. Pages: 460-467.

4. Title: **Quality of cloud services determined by the dynamic management of scheduling models for complex heterogeneous workloads.**

   Authors: Damián Fernández-Cerero, Alejandro Fernández-Montes, Joanna Kołodziej and Laurent Lefèvre.

   Published in: **11th International Conference on the Quality of Information and Communications Technology, QUATIC 2018**, Coimbra, Portugal, September 4-7, 2018. ISBN: 978-1-5386-5841-3. Pages: 210-219, DOI: 10.1109/QUATIC.2018.00039.

A.1.4 National Conferences

1. Title: **Energy Efficient Resource Usage in Data Centers: GreenDoop.**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, and Juan A. Ortega.**

   Published in: **XV Jornadas de Arca. Sistemas Cualitativos y sus Aplicaciones en Diagnosis, Robótica e Inteligencia Ambiental,** Murcia, Spain, June 24-27, 2013. ISBN: 978-84-616-7622-4. Page: 95.

2. Title: **¿Es eficiente apagar máquinas en un centro de datos?**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, Luis González Abril, and Juan A. Ortega.**

   Published in: **XVI Jornadas de Arca. Sistemas Cualitativos y sus Aplicaciones en Diagnosis, Robótica e Inteligencia Ambiental**, Cádiz, Spain, June 22-28, 2014. ISBN: 978-84-606-6085-9. Page: 67.

3. Title: **Un big picture sobre las tecnologías de computación y almacenamiento distribuidos.**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, and Juan A. Ortega.**

   Published in: **XVII Jornadas de Arca. Sistemas Cualitativos y sus Aplicaciones en Diagnosis, Robótica, Inteligencia Ambiental y Ciudades Inteligentes**, Vinaros, Spain, June 23-27, 2015. ISBN: 978-84-608-5599-6. Pages: 11-13.

4. Title: **Adecuación de las plataformas de enseñanza virtual en la enseñanza y despliegue de nuevas titulaciones. Aplicación práctica a Ingeniería de la salud.**

   Authors: **Alejandro Fernández-Montes, Damián Fernández-Cerero, and Juan A. Ortega.**

   Published in: **XII Foro Internacional sobre la Evaluación de la Calidad de la Invesitgación y de la Educación Superior (FECIES)**, Sevilla, Spain, July 9-11. ISBN: 978-84-608-9267-0.

5. Title: **Evaluación de las plataformas de enseñanza virtual universitaria para la mejora del aprendizaje guiado en las diferentes etapas académicas: grado, máster, doctorado y títulos propios.**

   Authors: **Damián Fernández-Cerero, Alejandro Fernández-Montes, and Luis Miguel Soria Morillo.**

   Published in: **XII Foro Internacional sobre la Evaluación de la**

Calidad de la Invesitgación y de la Educación Superior (FECIES), Sevilla, Spain, July 9-11. ISBN: 978-84-608-9267-0.

A.2 Grants

1. In 2014, I was granted with a competitive Ph.D. student (pre-doctoral) contract in University of Seville.

2. In 2016, I was financially supported by the University of Seville to perform an international research stage of four months in Lyon, France, entitled: Design and application of power provisioning policies in heterogeneous data centers (PP2016-5817).

3. In 2017, I was financially supported by the University of Seville to perform an international research stage of four months in Lyon, France, entitled: Design and application of predictive power-efficient policies in heterogeneous data center (PP2017-8672).

4. In 2018, I was financially supported by the University of Seville to perform an international research stage of two months in Cracow, Poland, entitled: Dynamic application of power-efficient policies in data centers based on Stackelberg Games.

5. In 2018, I was financially supported by the European Union through the STSM 41058 framed within the COST Action IC1406: High-Performance Modelling and Simulation for Big Data Applications (cHiPSet) to perform a research stage of 3 months in Cracow, Poland, entitled: Dynamic management of scheduling models for complex heterogeneous workloads.

6. In 2018, I was granted with a highly-competitive Marie Curie European Post-doctoral Fellowship in Dublin City University.

A.3 Research stages

During my Ph.D. student period, I spent 13 months on international research stages:

1. **Lyon, September - December 2016 (4 months)**. I was hosted by Laurent Lefèvre in École Normale Superiore de Lyon to develop the research project financially supported by the University of Seville: Design and application of power provisioning policies in heterogeneous data centers (PP2016-5817). As a result of this collaboration, a new research line has been started, including one published paper in an international conference.

2. **Cracow, June - October 2017 (4 months)**. I was hosted in Cracow University of Technology to develop the research project financially supported by the University of Seville: Design and application of predictive power-efficient policies in heterogeneous data center (PP2017-8672). We published several conference and high-impact journal papers as result of this collaboration.

3. **Cracow, May - September 2017 (5 months)**. I was hosted in Cracow University of Technology to develop two research projects, one financially supported by the University of Seville: Dynamic application of power-efficient policies in data centers based on Stackelberg Games, and the other one financially supported by the European Union through the STSM 41058 framed within the COST Action IC1406: High-Performance Modelling and Simulation for Big Data Applications (cHiPSet) to perform the project: Dynamic management of scheduling models for complex heterogeneous workloads. We published several conference and high-impact journal papers as result of this collaboration.

A.4 R&D projects

1. Title: **Simon. Saving Energy by Intelligent Monitoring (TIC-8052)**.

   Main researcher: **Juan Antonio Ortega Ramírez.**

   Granting Entity: **Consejería de Economía, Innovación y Ciencia.**

   Period: **2012-2014**.

   Reference: **TIC-8052.**

2. Title: **Arquitectura para la eficiencia energética y sostenibilidad en entornos residenciales** .

   Main researcher: **Juan Antonio Ortega Ramírez.**

   Granting Entity: **Ministerio de Ciencia e Innovación.**

   Period: **2009 - 2012**.

   Reference: **TIN2009-14378-C02-01.**

A.5 Industry projects

I have directed and performed transference of the knowledge acquired during my Ph.D. studies to industry partners through the following projects:

1. Title: **Optimización energética del centro de datos de DELEM**.

   Main researchers: **Damián Fernández-Cerero, Alejandro Fernández-Montes**

   Granting Entity: **Delem Ocio, S.L.**

   Period: **2017-2018.**

   Reference: **P043-17/E17.**

2. Title: **COSMIC: Desarrollo de la Plataforma Eficiente Cloud de Sokar Mechanics**.

Main researchers: **Damián Fernández-Cerero, Alejandro Fernández-Montes**

Granting Entity: **Sokar Mechanics, S.L.**

Period: **2017-2018.**

Reference: **P057-17/E17.**

3. Title: **COSMIC2: Mejora de la Plataforma Eficiente Cloud de Sokar Mechanics**.

   Main researchers: **Damián Fernández-Cerero, Alejandro Fernández-Montes**

   Granting Entity: **Sokar Mechanics, S.L.**

   Period: **2018-2019.**

   Reference: **P043-18/E17.**

A.6 Others

1. I am an active member of the **COST Action IC1406: High-Performance Modelling and Simulation for Big Data Applications (cHiPSet)**, attending to its meetings and being part of the European network collaboration.
2. I am a reviewer for the JCR-indexed journal *Simulation Modelling Practice and Theory.*
3. I am a reviewer for the JCR-indexed journal *Transactions on Services Computing.*
4. I am a reviewer for the JCR-indexed journal *Journal of Systems and Software.*
5. I attended and was a member of the organization committee of the national conference **Jornadas Ibéricas de Infraestructuras de Datos Espaciales 2015 (JIIDE 2015)**, Seville, Spain, 4-6 November 2015.

BIBLIOGRAPHY

[1] Arman Shehabi, Sarah Smith, Dale Sartor, Richard Brown, Magnus Herrlin, Jonathan Koomey, Eric Masanet, Nathaniel Horner, Inês Azevedo, and William Lintner. United states data center energy usage report. 2016.

[2] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *2010 IEEE 26th symposium on mass storage systems and technologies (MSST)*, pages 1–10. IEEE, 2010.

[3] Vinod Kumar Vavilapalli, Arun C Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, et al. Apache hadoop yarn: Yet another resource negotiator. In *Proceedings of the 4th annual Symposium on Cloud Computing*, page 5. ACM, 2013.

[4] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 2–2. USENIX Association, 2012.

[5] Jonathan G Koomey. Outperforming moore's law. *IEEE Spectrum*, 47(3):68–68, 2010.

[6] Norman Margolus and Lev B Levitin. The maximum speed of dynamical evolution. *Physica D: Nonlinear Phenomena*, 120(1-2):188–195, 1998.

[7] Hadi Esmaeilzadeh, Emily Blem, Renee St Amant, Karthikeyan Sankaralingam, and Doug Burger. Dark silicon and the end of multicore scaling. In *Computer Architecture (ISCA), 2011 38th Annual International Symposium on*, pages 365–376. IEEE, 2011.

[8] Matei Zaharia, Benjamin Hindman, Andy Konwinski, Ali Ghodsi, Anthony D Joesph, Randy Katz, Scott Shenker, and Ion Stoica. The datacenter needs an operating system. In *Proceedings of the 3rd USENIX conference on Hot topics in cloud computing*, pages 17–17. USENIX Association, 2011.

[9] Chakravanti Rajagopalachari Kothari. *Research methodology: Methods and techniques.* New Age International, 2004.

[10] Kejiang Ye, Dawei Huang, Xiaohong Jiang, Huajun Chen, and Shuang Wu. Virtual machine based energy-efficient data center architecture for cloud computing: a performance perspective. In *Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*, pages 171–178. IEEE Computer Society, 2010.

[11] Ratnesh K Sharma, Cullen E Bash, Chandrakant D Patel, Richard J Friedrich, and Jeffrey S Chase. Balance of power: Dynamic thermal management for internet data centers. *IEEE Internet Computing*, 9(1):42–49, 2005.

[12] Nosayba El-Sayed, Ioan A Stefanovici, George Amvrosiadis, Andy A Hwang, and Bianca Schroeder. Temperature management in data centers: why some (might) like it hot. *ACM SIGMETRICS Performance Evaluation Review*, 40(1):163–174, 2012.

[13] Akihiko Miyoshi, Charles Lefurgy, Eric Van Hensbergen, Ram Rajamony, and Raj Rajkumar. Critical power slope: understanding the runtime effects of frequency scaling. In *Proceedings of the 16th international conference on Supercomputing*, pages 35–44. ACM, 2002.

[14] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. Power provisioning for a warehouse-sized computer. In *ACM SIGARCH Computer Architecture News*, volume 35, pages 13–23. ACM, 2007.

[15] David G Andersen and Steven Swanson. Rethinking flash in the data center. *IEEE micro*, 30(4):52–54, 2010.

[16] Mark E Femal and Vincent W Freeh. Boosting data center performance through non-uniform power allocation. In *Second International Conference on Autonomic Computing (ICAC'05)*, pages 250–261. IEEE, 2005.

[17] James Hamilton. Cost of power in large-scale data centers. *Blog entry dated*, 11:28, 2008.

[18] Charles Reiss, John Wilkes, and Joseph L. Hellerstein. Obfuscatory obscanturism: making workload traces of commercially-sensitive systems safe to release. In *3rd International Workshop on Cloud Management (CLOUDMAN)*, pages 1279–1286, Maui, HI, USA, April 2012. IEEE.

[19] Malte Schwarzkopf, Andy Konwinski, Michael Abd-El-Malek, and John Wilkes. Omega: flexible, scalable schedulers for large compute clusters. In *Proceedings of the 8th ACM European Conference on Computer Systems*, pages 351–364. ACM, 2013.

[20] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)*, 26(2):4, 2008.

[21] Christina Delimitrou and Christos Kozyrakis. Paragon: Qos-aware scheduling for heterogeneous datacenters. In *ACM SIGPLAN Notices*, volume 48, pages 77–88. ACM, 2013.

[22] Michael Isard, Vijayan Prabhakaran, Jon Currey, Udi Wieder, Kunal Talwar, and Andrew Goldberg. Quincy: fair scheduling for distributed computing clusters. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 261–276. ACM, 2009.

[23] Christina Delimitrou, Daniel Sanchez, and Christos Kozyrakis. Tarcil: reconciling scheduling speed and quality in large shared clusters. In *Proceedings of the Sixth ACM Symposium on Cloud Computing*, pages 97–110. ACM, 2015.

[24] Robert Grandl, Ganesh Ananthanarayanan, Srikanth Kandula, Sriram Rao, and Aditya Akella. Multi-resource packing for cluster schedulers. *ACM SIGCOMM Computer Communication Review*, 44(4):455–466, 2015.

[25] Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma, Khaled Elmeleegy, Scott Shenker, and Ion Stoica. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In *Proceedings of the 5th European conference on Computer systems*, pages 265–278. ACM, 2010.

[26] Christina Delimitrou and Christos Kozyrakis. Quasar: resource-efficient and qos-aware cluster management. In *ACM SIGPLAN Notices*, volume 49, pages 127–144. ACM, 2014.

[27] Xiao Zhang, Eric Tune, Robert Hagmann, Rohit Jnagal, Vrigo Gokhale, and John Wilkes. Cpi 2: Cpu performance isolation for shared compute clusters. In *Proceedings of the 8th ACM European Conference on Computer Systems*, pages 379–391. ACM, 2013.

[28] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. Large-scale cluster management at google with borg. In *Proceedings of the Tenth European Conference on Computer Systems*, page 18. ACM, 2015.

[29] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D Joseph, Randy H Katz, Scott Shenker, and Ion Stoica. Mesos: A platform for fine-grained resource sharing in the data center. In *NSDI*, volume 11, pages 22–22, 2011.