Bullfighting extreme scenarios in efficient hyper-scale cluster computing

Damián Fernández-Cerero[1] · Francisco J. Ortega-Irizo[2] · Alejandro Fernández-Montes[1] · Francisco Velasco-Morente[2]

Abstract

Data centres are quickly evolving to support new demands for Cloud-Computing services. Extreme workload peaks represent a challenge for the maintenance of the performance and service level agreements, even more when operation costs need to be minimised. In this paper, we first present an extensive analysis of the impact of extreme workloads in large-scale realistic Cloud-Computing data centres, including a comparison between the most relevant centralised resource-managing models. Moreover, we extend our previous works by proposing a new energy-efficiency policy called *Bullfighter* which is able to keep performance key performance indicators while reducing energy consumption in extreme scenarios. This policy employs queue-theory distributions to foresee workload demands and adapt automatically to workload fluctuations even in extreme environments, while avoiding the fine-tuning required for other energy policies. Finally, it is shown through extensive simulation that *Bullfighter* can save more than 40% of energy in the aforementioned scenarios without exerting any noticeable impact on data-centre performance.

Keywords Energy policies · Efficiency · Data centre · Simulation software · Decision making

1 Introduction

Sustainability is one of the major economic issues of the last decade, and still researchers are encouraged to keep energy consumption as low as possible in all domains, specially in production industries. Information Technology industries should be one of the leaders in this context, as the evolution and development of its infrastructures have a very short life cycle, enabling researchers to propose new methodologies and strategies to lead the sustainability requirements. Minimisation of energy consumption is one of the main areas of work to reduce costs and preserve environmental sustainability.

Cloud computing paradigm has led to a new scenario where each of this infrastructures are as energy greedy as traditional industrial factories and small towns. Data centres constitute the computational and storage core for cloud computing. The latest tendencies show that data centres account for approximately 2% of global energy consumption [24], and it increases by 5% annually [26].

The huge amount of energy consumed by Cloud-Computing data centres provides justification for a study into energy-saving methodologies, either from an economic or ecological point of view. To this end, cloud computing operational policies must be analysed in order to be optimised. The workload that has to be processed by Cloud-Computing data centres usually follows a pattern related to queue theory. These workloads are process by schedulers, which may follow various centralized strategies:

✉ Alejandro Fernández-Montes
afdez@us.es

Damián Fernández-Cerero
damiancerero@us.es

Francisco J. Ortega-Irizo
fjortega@us.es

Francisco Velasco-Morente
velasco@us.es

1 Department of Computer Languages and Systems, University of Seville, 41012 Seville, Spain

2 Department of Applied Economy I, University of Seville, 41018 Seville, Spain

monolithic, two-level and shared state as detailed in Sect. 2. As Cloud-Computing infrastructures are usually composed by thousands of machines and experimentation also needs to be performed through long periods, simulation tools have been widely used by the research community.

Energy efficiency may be improved in several areas, such as hardware power consumption, energy distribution, network equipment, and cooling. In this work we focus on workload consolidation to reduce the number of servers in an idle state due to workload fluctuations, such as day/night patterns. Such model is interesting since current data centres usually operate at less than half of their maximum capacity.

The shut-down of servers in low-utilisation periods is challenging, above all, because the incoming workload must be successfully foreseen. Furthermore, when the workload arrival pattern does not follow a stable pattern, such as one following the Exponential distribution and/or day/night patterns, performing bad shut-down decisions may impose a critical performance penalty due to insufficient available servers.

Current trends tend to consolidate various workloads in the same hardware to improve resource efficiency. In these large-scale multi-purpose data centres, the different workload patterns of these heterogeneous workloads may lead to extreme workload arrival patterns. In green data centres that employ consolidation and shut-down policies, such extreme peaks may break service level agreements (SLAs) if the chosen policies are not prepared to adapt to rapidly-changing environments.

In this paper, we focus on such extreme scenarios, and present the following novelties:

– Characterisation of the impact in terms of performance and energy consumption of extreme workloads in large-scale realistic Cloud-Computing data centres, including several centralised resource- managing models employed in industry.
– A new energy-efficiency policy called *Bullfighter* which is able to automatically adapt to workload fluctuations even in extreme scenarios without fine tuning.

The rest of the paper is structured as follows: Sect. 2 includes a brief summary of some approaches found in the literature and separated by their perspectives. Section 3 describes the theoretical framework and formalisation proposed to support the models developed. Section 4 explains in detail the proposed Energy-efficiency policy named *Bullfighter* and then experimentation and results are presented and discussed in Sect. 5. Finally, conclusions are drawn in Sect. 6.

2 Related work

Data-centre energy efficiency can be addressed from various perspectives, from an industrial point of view improving facilities organisation, cooling, arrangement to a software related solutions. From the latter proposed strategies range from energy-aware scheduling algorithms to power-off heuristics that aim to minimize the idleness time of computational resources. In [7] a neural network predictor is proposed to support a green scheduling algorithm, and in [15] the energy-aware scheduling proposal is backed by a multi-objective algorithm backed by a directed acyclic graph of jobs. The most related approaches to this work are the ones that aim to switch off resources such as safety margin power-off policy proposed by [22], migration policies to free resources proposed by [1, 2, 25] or energy-aware task consolidation proposed by [19]. In this work, 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 centres of 5000 and 100 nodes are simulated. In this kind of scenario, potential energy reductions between approximately 20% and 70% are shown.

This paper follows the approach of deeply describe the impact of a power-off policy when the workload pattern present unexpected high or low peaks of jobs. The impact is analyzed 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 various centralized scheduling frameworks. The proposed power-off policy is applied at the resource manager level, not to a framework or subsystem, making it possible to be applied to any framework.

2.1 Resource and energy efficiency in Cloud-Computing data centres

Data centres are usually governed by resource managers in order to orchestrate the whole process flow from the arrival of jobs to their deployment to the computational resources and execution tracking. There are various resource-managing approaches from centralised to fully distributed and even hybrid solutions. Centralised approaches are currently the most popular solutions, where we can highlight three alternatives known as monolithic (e.g. Google Borg) [27], shared-state (e.g. Omega) [23] and two-level (e.g. Mesos) [13]. In monolithic models, a centralised scheduling algorithm is employed for all jobs. Two-level alternatives separate the responsibilities of resource allocation and task placement, there is a unique and centralised resource manager that exposes computing resources to multiple parallel scheduling nodes, enabling task-placement logic to be developed for each application. It also allows the cluster state to be shared between clusters. Finally, in shared-state alternatives the cluster meta-data is shared between all schedulers but the scheduling process is performed by using a copy of the cluster state data, that can be out of date. Thus, when one of these parallel schedulers makes a decision, it tries to commit it as a transaction, and in case that it cannot be applied because the selected resources are no longer free, the scheduling operation is retried until no conflicts are found.

2.2 Extreme values

The extreme-value theory was developed by studying the limiting distribution of the random variables sequences $\max\{X_1, \ldots, X_n\}$ and $\min\{X_1, \ldots, X_n\}$, where $\{X_1, \ldots, X_n\}$ are independent and identically distributed (iid) random variables. Historically, work on extreme value problems may be dated back to as early as 1709, when N. Bernoulli discussed the mean largest distance from the origin when n points lie at random on a straight line of length t [18]. Gumbel's book of 1958 [12] contains a very extensive bibliography of the developed literature up to that point of time. Of course, since then, many more refinements of the original ideas and further theoretical developments and fields of applications have emerged.

There are three different families of models of Extreme Value distributions: Gumbel, Fréchet and Weibull. These three families can be unified by means of the Generalized Extreme Value Family. An extensive revision of Extreme Values distributions and their applications can be seen in Reiss and Thomas [14]. Extreme value analysis (EVA) is useful for the study of extreme deviations of a random variable and has been applied to topics as varied as Finance [8], Structural Design [6] and Environmental Modelling [17]. In field of Engineering, EVA has been widely used, for example, in Mechanical Engineering [3], and Teletraffic Engineering [21].

3 Theoretical framework

3.1 Workload model

A workload is a set of jobs that has to be deployed and executed by the data-centre resources. Workloads are composed of jobs $\mathcal{W} = \{J_j\}_{j=1}^{n}$, and jobs are composed of tasks $\mathcal{T}_j = \{t_{ji}\}_{i=1}^{n_j}$. Workloads are usually divided depending on the duration and goal of their jobs into batch or service workloads:

- *Batch* jobs are the ones that perform a computation and then finish. These jobs have a determined start and end (e.g. Map-reduce jobs)
- *Service* jobs are long-running which provide end-user operations and infrastructure services and they have no determined end. Web servers or services, such as BigTable [16], are good examples of *Service* jobs.

The following job attributes have been considered and studied:

- *Inter-arrival time* X_j represents the time between two consecutive jobs J_j and J_{j-1}. Therefore it also influences the amount of jobs executed in a specific time window. The inter-arrival time between two *Batch* jobs is usually shorter than the time between two *Service* jobs. According to the queue theory, the inter-arrival time usually follows an exponential distribution. Notwithstanding, other distributions, such as the extreme-value Weibull distribution may be employed.
- *Number of tasks* $n_j \sim Exp(\lambda_t)$ represents the number of tasks that compose a job. The number of tasks of a particular job J_j is generated following an Exponential distribution with a given the mean value $1/\lambda_t$.
- *Job duration* $d_j \sim Exp(\lambda_d)$ represents the time during which a job J_j consumes resources in the data centre. The duration of all tasks of a particular job J_j is generated by the means of the Exponential distribution with the given expected value $1/\lambda_d$.
- *Resource usage* is the amount of CPU K_{CPU} and RAM K_{RAM} that all the tasks of all the jobs of a particular workload consumes.

3.2 Utilisation of the Weibull model

In order to model extreme inter-arrival times, the Weibull model has been considered for several reasons. In this case, extreme inter-arrival times present positive small values and therefore the model must be obtained as the limiting distribution of the random variables sequence $\min\{X_1, \ldots, X_n\}$ and must have a positive range. The only extreme value family that verifies these two conditions is the Weibull family. Specifically, we use the following definition: a random variable X follows the two-parameter Weibull distribution $W(\alpha, \lambda)$ (and we write $x \sim W(\alpha, \lambda)$) if its probability density function (pdf) is:

$$f(x) = \alpha\lambda(x\lambda)^{\alpha-1}e^{-(x\lambda)^{\alpha}}, \quad x > 0,$$

where α and λ are the shape and scale parameters respectively.

The Exponential distribution is commonly used to model the arrival of workload to data centres. Therefore, one of the objectives of this work is to compare the results obtained from the Weibull and Exponential models. A random variable X follows the Exponential distribution $Exp(\lambda)$ (and we write $x \sim Exp(\lambda)$) if its pdf is:

$$f(x) = \lambda \exp\{-\lambda x\}, \quad x > 0,$$

where $\lambda > 0$ is the scale parameter.

Given the definition of the Weibull model used above, the Exponential distribution consists in a particular case of the Weibull family (for the Weibull model the most usual definition in the literature is $W(\alpha, \beta)$, where $\beta = \lambda^{-1}$). Specifically, if $\alpha = 1$,, a random variable X follows a Weibull distribution $W(\alpha = 1, \lambda)$ if and only if X follows the Exponential distribution $Exp(\lambda)$. This fact facilitates the comparison of results between both models.

Since our objective consists on the modelling of extremely small inter-arrival times, we consider only values of the parameter less than or equal to one, since the closer to zero, the greater the probability of obtaining extremely small values (Table 1). If a random variable $x \sim Exp(\lambda)$ is taken to model inter-arrival times, it is well known that the expected value can be noted as $E[X] = \lambda^{-1}$. For each $0 < \alpha_0 \le 1$, a random variable $X_0 \sim W(\alpha_0, \lambda_0)$ is considered, with the condition that the mean of inter-arrival times equals that of the Exponential, that is, $E[X_0] = \lambda^{-1}$. Taking into account that $E[X_0] = \Gamma\left(1 + \alpha_0^{-1}\right)/\lambda_0$, the Weibull model used to generate inter-arrival times may be described as $X_0 \sim W\left(\alpha_0, \lambda_0 = \Gamma\left(1 + \alpha_0^{-1}\right)\lambda\right)$, where $\Gamma(\cdot)$ denotes the Euler's Gamma Function [18].

The values of $P[X_0 \le 0.1]$ obtained for $\lambda = 1$ and several values of α_0 are shown in Table 1. These values are the probabilities to obtain an inter-arrival time lesser than 10% of the expected time. For $\alpha = 1$ (Exponential model), the probability is approximately equals to 10%. This probability increases as the alpha value decreases, as already indicated above, reaching a value of 97.3% for $\alpha_0 = 0.1$.

Table 1 Probabilities of obtaining an inter-arrival time lesser than 10% of the expected time

λ_0	$P[X_0 \le 0.1]$
0.1	0.973
0.2	0.807
0.3	0.624
0.4	0.475
0.5	0.361
0.6	0.275
0.7	0.21
0.8	0.16
0.9	0.123
1	0.095

3.3 Performance model

In this work we evaluate the performance of the executed jobs, especially Batch jobs. This performance is represented by the following key performance indicators (KPIs):

- *Queue times* The time jobs spend in queue waiting for their first task $q_{(1)j}$ and last task $q_{(n_j)j}$ to be deployed.
- *Makespan* Representing the time since the submission of the job until its completion. The makespan of the job J_j may be described as follows: $\mathcal{C}_j = q_{(n_j)j} + d_j$

3.4 Energy model

In this work, the energy consumption is measured as follows:

$$EC(\mathcal{T}) = \sum_{t \in \Delta} \sum_{m \in \mathcal{M}} \mathcal{P}(e_{m,t})\ \delta$$

Let $t \in \Delta = \{\delta, 2\delta, \ldots, \mathcal{T}\}$, where $\mathcal{T}$ is the total simulation time. For each period of time δ, the state of each machine $e_{m,t}$ is measured, and the energy consumption computed depending of the power consumption $\mathcal{P}(e)$ of that particular state. The power states considered in this work $e \in \{$ On, Idle, Switching $\}$, where On denotes the power consumption of a machine when executing a task, Idle when the machine is waiting for an incoming job to be executed, and Switching if the machine is shutting-down or switching-on.

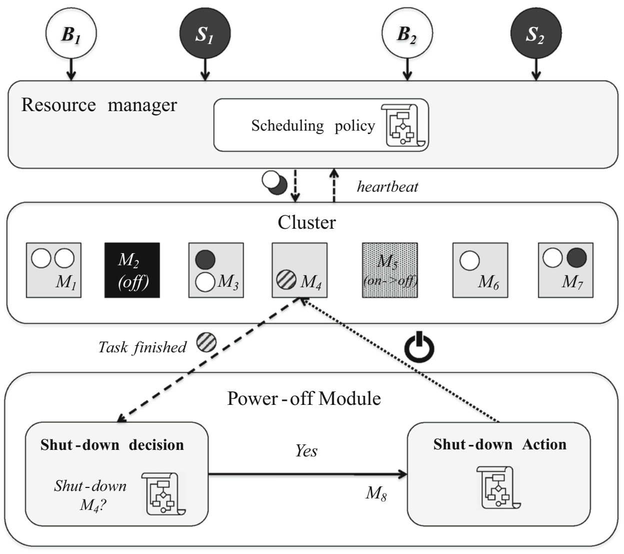


Fig. 1 Shut-down process architecture

4 Bullfighter energy-efficiency policy

Energy-efficiency policies are responsible for the shut-down of machines in idle state. As illustrated in Fig. 1, resource managers employ scheduling policies which are the responsible to decide where a particular task t is to be deployed. Every time a task finishes, the power-off module makes a decision whether the machine that was executing that particular task should be powered off or not. In case affirmative, a shut-down action is performed on the machine. The shut-down action must ensure that that machine is not executing any tasks at the time of shut-down.

A set of energy-efficiency policies were developed in [10], including: (1) always power off; (2) random power off; (3) keeping a security margin; (4) power-off depending on the resource occupation of the data centre; and (5) probabilistic policies which employ queue-theory distributions, such as Gamma and Exponential, to forecast the incoming workload. Nonetheless, the energy-efficiency policies that achieve the best results, such as those that use probabilistic and security-margin approaches, precise a fine tuning that requires a deep knowledge on the data-centre workload patterns.

In this work we propose *Bullfighter*, which is able to overcome the aforementioned limitations related to fine-tuning depending on the environment by providing a new simple approach, which may be formally described as follows:

Let m be a machine with p_m tasks $\{t_m^i\}_{i=1}^{p_m}$, where $m \in \mathcal{M} = \{m_1, m_2, \ldots, m_n\}$. A specific task t_m^h is considered.

A decision is defined as

$$d(t_m^h) = \begin{cases} 0 & \text{The machine } m \text{ is shut down .} \\ 1 & \text{The machine } m \text{ is not shut down .} \end{cases}$$

If $d(t_m^h) = 1$, an action $a(m) = 1$ is taken.

Let τ be the future time, in which it is intended that the data centre can execute the incoming workload.

let $j(\tau)$ be the number of the last jobs executed, as a sample of the incoming workload in a time τ. Let $IW(\tau)$ be, therefore, the incoming workload to be done in a time τ.

Let s be the number of predictions executed to analyze if at time τ, $IW(\tau)$ is executed, that is to say $s = \#(IW(\tau))$

Let $p = \frac{\sum_{m \in \mathcal{M}} a_s(m)}{s}$ be the percentage of positive decisions (meaning that the machine m can be safely shut down) to a given decision threshold D in order to make the final shut down decision.

The algorithm that implements the above model is explained in Algorithm 1: it takes the last $j(\tau)$ jobs as representative for the incoming workload. Then, it performs a number of predictions s to analyse whether the data centre can execute the incoming workload in a given future time $IW(\tau)$ with the available resources alone, that is, without powering-on any machine. Finally, it computes the percentage of positive decisions (meaning that the machine m can be safely shut-down) p to a given decision threshold D in order to make the final shut-down decision.

The creation of future workload is based in queue theory. Specifically, we employ an Exponential distribution for the generation of the values for each parameter of the jobs $\mathcal{J}$ composing the workloads $\mathcal{W}$, using the means provided by the last $j(\tau)$ jobs. More details on workload generation may be found in Sect. 3.

Algorithm 1: *Bullfighter* shut-down decision

```
Inputs: $num\_jobs \in \mathbb{N}^+$, $threshold \in (0,1) \subset \mathbb{R}$,
        $future\_time \in \mathbb{R}$, $num\_predictions \in \mathbb{N}^+$
Data: $cluster$
Output: $decision \in \mathbb{B}$
1  $Decisions \leftarrow new\_array(\textbf{len} = num\_predictions)$
2  $Past\_jobs \leftarrow cluster.get\_past\_jobs(num\_jobs)$
3  for $i \leftarrow 0$ to $num\_predictions - 1$ do
4      $arrival \leftarrow$
         $get\_average\_inter\_arrival\_time(Past\_jobs)$
5      $tasks \leftarrow get\_average\_tasks(Past\_jobs)$
6      $duration \leftarrow get\_average\_duration(Past\_jobs)$
7      $cpu \leftarrow get\_average\_cpu\_consumption(Past\_jobs)$
8      $mem \leftarrow$
         $get\_average\_memory\_consumption(Past\_jobs)$
9      $wl\_gen \leftarrow$
         $new\_workload\_gen(arrival, tasks, duration, cpu, mem)$
10     $wl \leftarrow$
         $wl\_gen.generate\_workload\_for\_time(future\_time)$
11     $Decisions[i] \leftarrow wl.cpu\_consumption <$
         $cluster.cpu\_free \;\&\; wl.mem\_consumption <$
         $cluster.mem\_free$
12 $decision \leftarrow$
     $\textbf{count}(Decisions)true/num\_predictions >$
     $threshold$
13 return $decision$
```

5 Experimentation

In this section, we perform two deep analysis: (1) the performance impact of extreme arrival rates in the most relevant centralised resource managers for various large-scale cloud-computing scenarios; and (2) the impact of the application of a novel simple power-off policy in terms of performance and energy efficiency in one representative scenario.

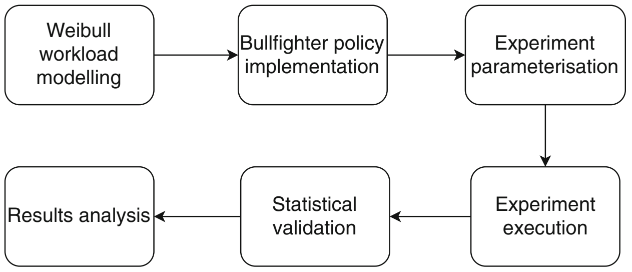

Weibull workload modelling
Bullfighter policy implementation
Experiment parameterisation
Experiment execution
Statistical validation
Results analysis

Fig. 2 Experimentation work-flow

The work-flow of the experimentation resulting from the aforementioned theoretical analysis is illustrated in Fig. 2.

We employed our previously-developed simulation tool SCORE [9, 11], which focuses on the simulation of large-scale cloud-computing data centres. This simulator has run on a server equipped with Intel Xeon E3-1220v5 4C/4T 3.00 GHz, 16 Gb of RAM, CentOS 7.1 and JRE 1.8.0.

All the experiments simulate 7 days of operation time, and similar synthetic workloads are employed in all experiments. Following the trends present in large companies, such as Google[20], and Alibaba[5], two kind of workloads sharing the same data-centre resources are generated based on statistical distributions, as explained in Sect. 3:

- Batch workloads, representing jobs which compute a task and then finish,(e.g. MapReduce jobs). In this work, Batch workloads account for approximately 90% of deployed jobs ($\sim$ 26,000), consuming approximately 60% of the data-centre resources. Each Batch job is composed of 180 tasks, which consume 0.3 CPU cores and 200 Mb of RAM during 180 s in average.
- Service workloads, representing applications, services and frameworks with no fixed end (e.g. virtual machines). Service workloads account for $\sim$ 10% of deployed jobs (approximately 2500), which consume $\sim$ 40% of the resources of the data centre. Each Service job is composed of 30 tasks, which consume 0.5 CPU cores and 700 Mb of RAM during 1000 s in average.

All the workloads follow a day/night pattern. It must be borne in mind that the value of each job parameter is generated based on statistical distributions, as described in Section 3.

The workloads are executed in data centres of 1000 and 2000 machines. The machines are considered homogeneous in terms of performance and energy consumption and equip 8 CPU cores and 16GB of RAM.

5.1 Performance impact of extreme workload-arrival rates

We first evaluate the impact of extreme distributions in the arrival of tasks and the consequences in terms of data-centre performance as a first step towards the proposal of a model for energy efficiency in extreme environments.

The Weibull distribution is employed for the generation of extreme values of task-arrival times in the workload-generation phase. The range [0.2–0.8] with a step of 0.1 is considered for α values. The parameter λ_0 is not presented since it can be obtained following the explanation given in Section 3. The results provided by these workloads are then compared with those provided by a traditional Exponential-based workload generation.

Extreme-value distributions may generate heterogeneous distribution patterns when a low value of α is taken, as shown in Fig. 3. In this work we performed 10 simulations of seven days of data-centre operation for each α value ([0.2–0.8] with a step of 0.1) of Weibull-based, and 10 simulations for the Exponential-based workload-generation scenario. The average of such results are presented. We evaluated the homogeneity of the results within populations by means of the *t*-student test and the statistical relevance of the results of different parameterisation through ANOVA tests with $\alpha = 0.1$.

Moreover, in this section, we analysed the following parameterisation for each workload-generation strategy:

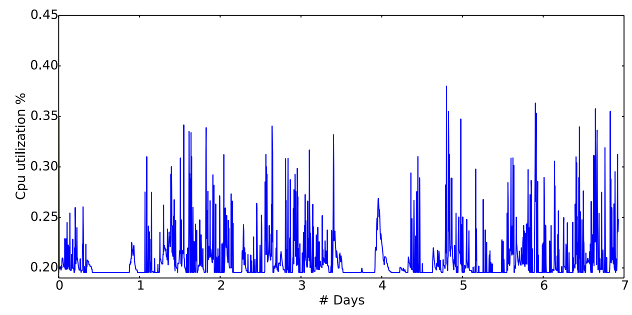
- Three of the most relevant centralised resource-managing models:
  (a) Monolithic, such as Google Borg [4];
  (b) Two-level, such as Mesos [13]; and
  (c) Shared-state, such as Google Omega [23].

  Two-level and Shared-state resource managers employ four schedulers for Batch jobs, and one scheduler for Service jobs.
- Three levels of initial utilisation: 20, 30, and 50% of resources.
- Two data-centre sizes: 1000 and 2000 machines.

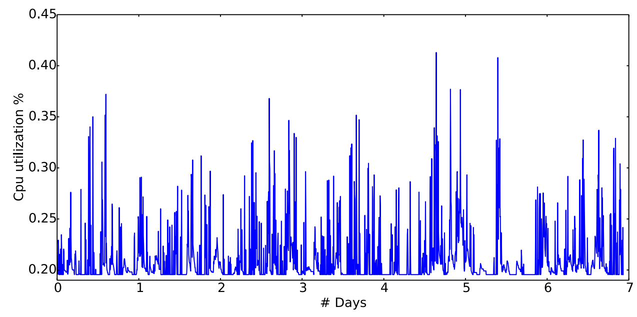
The rest of parameters are equal for all the simulation scenarios. Only performance results of Batch jobs are presented for the sake of clarity, since these are the most affected by poor scheduling performance. The complete data set, containing the results for both Batch and Service workloads, as well as many more key performance indicators can be found as Supplementary material.

The summary of the results for Monolithic, Two-level and Shared-state models are shown in Tables 2, 3, and 4, respectively.

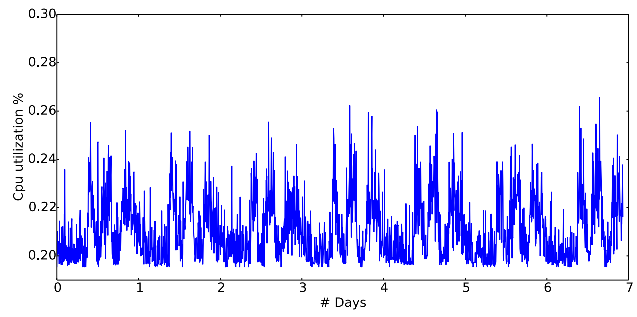
In Monolithic scheduling scenarios, extreme values have a significant impact only in terms of queue times. It can be noticed that even a value of $\alpha = 0.8$, which means it is very similar to the Exponential distribution, has a severe impact in terms of the time jobs spend in queue waiting to be scheduled. Exponential-generated workloads wait in queue 240 ms. until their first task is scheduled $q_{(1)j}$, whilst Weibull[0.8]-generated workloads wait 440 ms in average. These times represent graphically an Exponential trend. While the aforementioned time increases by approximately 50% between $\alpha = 0.8$ and $\alpha = 0.7$, it shows an increment of $\sim$ 2x for $\alpha = 0.4$ and $\alpha = 0.3$, and $\sim$ 4x between $\alpha = 0.3$ and $\alpha = 0.2$. This trend is illustrated in Fig. 4 (blue line). The very same pattern is present for the time jobs wait in queue until all their tasks are scheduled $q_{(n_j)j}$, which



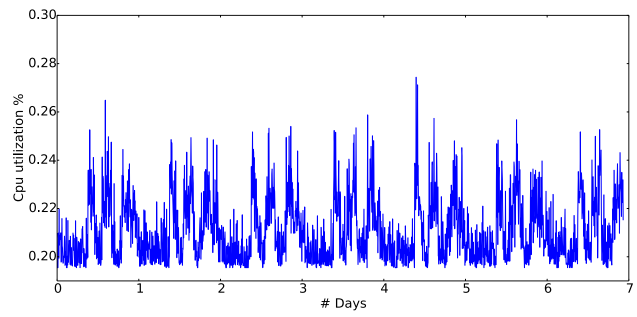
(a) Weibull 0.2 sample 1



(b) Weibull 0.2 sample 2



(c) Exponential sample 1



(d) Exponential sample 2

Fig. 3 CPU consumption comparison between workloads whose inter-arrival values are generated by means of a Weibull 0.2 distribution (**a**, **c**), and Exponential distribution (**b**, **d**). All of them follow the same day/night pattern. It can be noticed that Weibull distributions generate more heterogeneous patterns than Exponential distributions

means that the scheduler is usually able to schedule jobs in one scheduling operation. The results provided by 90 percentile of the values are consistent with the average results. The impact in terms of the makespan is minor. Finally, with this first analysis, we can state the number of available resources, which is represented by the data-centre size and the occupation, has a minimum impact in terms of performance compared to that of extreme inter-arrival values.

Two-level schedulers present a different behaviour. In this scenario, we also included the number of jobs that stayed in queue without any scheduling attempt (notation), and the number of timed-out jobs in the scheduling process, which means that the scheduler tried to schedule 100 times a particular job or 1000 times its tasks. We do not show these results for Monolithic and Shared-State resource managers because they can successfully schedule the total number of jobs in all scenarios.

Unlike in monolithic environments, the number of available resources is extremely relevant in two-level scenarios. This is shown in Weibull[0.3] results. On one hand, when the number of available resources is low (50% of initial utilisation and 1000 machines in the data centre), almost 2000 jobs could not be scheduled, and all the queue times skyrocket: jobs spend an average of 1682 s in queue for their first task to be scheduled ($q_{(1)j}$), compared to 3.73 s in less-congested scenarios. It must be noticed that this behaviour is not gradual, but shows a clear frontier where performance results become unacceptable, specially for makespan results: in this case, the average makespan is the total simulation time (1 week). On the other hand, when 2000 machines or lower utilisation environments are considered (20 or 35% utilisation), performance results fall back to reasonable values.

Moreover, the results in terms of performance provided by more extreme scenarios, such as those shown for Weibull[0.2], follow the same trend: only the environments with more available resources (data centre composed of 2000 machines and an initial utilisation of [20,35]%) present a reasonable behaviour. In the rest of Weibull[0.2] scenarios, the number of jobs timed out or not scheduled, as well as queue and makespan times, skyrocket.

In those scenarios which do not present unacceptable performance results, both makespan C_j and queue times increase with each step of the α value, as shown in the previous monolithic environment. The increase pattern of the queue time jobs wait until their first task is scheduled is illustrated in Fig. 4 (orange line). In these cases, such as Weibull[0.2] with 2000 machines and 20% of resource utilisation, two-level resource managers achieve better results in term of queue times ($\sim$ 9 vs. $\sim$ 15 s) than monolithic schedulers.

Table 2 Performance impact of extreme values on Monolithic centralised schedulers

Prefill (%)	DC size	Sched. oper.	$q_{(0)j}$		$q_{(n_j)j}$		$\mathcal{C}_j$	
			Avg	90p.	Avg.	90p.	Avg.	90p.
Exponential								
[50,35,20]	1000	28,264	0.24	0.29	0.24	0.29	39.42	90.95
[50,35,20]	2000	28,176	0.24	0.32	0.24	0.32	39.84	90.86
Weibull[0.8]								
[50,35,20]	1000	28,354	0.44	1.50	0.44	1.50	39.91	91.63
[50,35,20]	2000	28,320	0.45	1.52	0.45	1.52	39.89	92.10
Weibull[0.7]								
[50,35,20]	1000	28,149	0.62	2.27	0.62	2.27	40.21	92.72
[50,35,20]	2000	28,389	0.58	2.11	0.58	2.11	40.15	93.32
Weibull[0.6]								
[50,35,20]	1000	28,214	0.85	3.13	0.85	3.13	39.85	91.62
[50,35,20]	2000	27,948	0.85	3.09	0.85	3.09	40.12	92.49
Weibull[0.5]								
[50,35,20]	1000	27,529	1.36	4.67	1.36	4.67	40.80	93.85
[50,35,20]	2000	28,782	1.32	4.55	1.32	4.55	40.00	92.67
Weibull[0.4]								
[50,35,20]	1000	28,096	2.22	7.08	2.22	7.08	40.00	92.11
[50,35,20]	2000	28,020	2.31	7.29	2.31	7.29	40.51	93.85
Weibull[0.3]								
[50,35,20]	1000	27,973	4.68	12.96	4.68	12.96	40.00	91.80
[50,35,20]	2000	28,600	4.72	12.86	4.72	12.86	40.03	92.56
Weibull[0.2]								
50	1000	29,670	17.61	45.05	17.65	45.10	39.87	92,00
[35,20]	1000	29,642	17.58	44.89	17.58	44.89	39.84	91.86
[50,35,20]	2000	28,167	15.60	38.79	15.60	38.79	40.04	91.87

All time-related parameters are given in seconds

$q_{(0)j}$ and $q_{(n_j)j}$ represents the time jobs spend in queue waiting for their first and all their tasks to be scheduled, respectively, whilst $\mathcal{C}_j$ denotes the makespan

Shared-state resource managers seem to performance better than two-level and monolithic models when extreme values are considered. We include the number of scheduling conflicts, due to parallel schedulers trying to deploy tasks on the same resources.

Despite the increment of queue times keeps stable at $\sim$ 2–3x ratio in every step of α, an increment of $\sim$ 5x is shown in the most extreme case, the step between $\alpha = 0.3$ and $\alpha = 0.2$. Moreover, even in the worst-case scenarios (Weibull[0.2], 1000 machines, and 50% of initial occupation), shared-state schedulers keep the time jobs wait in queue until their first task is deployed as low as $\sim$ 1.6 s, in comparison with the $\sim$ 17 s and more than 17 h of monolithic and shared-state resource managers, respectively.

Regarding the number of scheduling conflicts, the increment shows a stable ratio of [1.3–1.4]x approximately between Weibull α steps, and it is sensitive to resource availability.

After this analysis, we can highlight the following: (a) extreme values always impact negatively in terms of performance; (b) the impact in terms of makespan is minor, increasing 20% in worst-case scenarios, except for very congested two-level resource managers, where the capacity of the system may be exceeded; (c) queue times become unacceptable only in very extreme cases, represented by Weibull[0.3, 0.2]; (d) only the two-level resource managers under very high pressure are unable to handle all the incoming workload; and (e) In general, the resource managers with parallelisation strategies (two-level and shared-state models) perform better than monolithic in the vast majority of scenarios. The optimistic-locking model of shared-state resource managers outperform the rest of strategies, and monolithic can only achieve better results than the pessimistic-locking strategy of two-level resource managers when very congested environments are under consideration.

Table 3 Performance impact of extreme values on Two-Level centralised schedulers

Prefill (%)	DC size	JLIQ	JTOS	$q_{(0)j}$		$q_{(n_j)j}$		$\mathcal{C}_j$	
				Avg	90p.	Avg	90p.	Avg.	90p.
Exponential									
[50,35,20]	1000	0	0	0.26	0.39	0.26	0.40	41.71	95.52
[50,35,20]	2000	0	0	0.25	0.38	0.26	0.41	41.87	95.76
Weibull[0.8]									
[50,35,20]	1000	0	0	0.40	1.22	0.40	1.24	42.09	94.92
[50,35,20]	2000	0	0	0.42	1.31	0.43	1.34	41.38	93.03
Weibull[0.7]									
[50,35,20]	1000	0	0	0.57	1.97	0.58	2.00	41.97	96.37
[50,35,20]	2000	0	0	0.61	2.02	0.62	2.05	42.27	96.50
Weibull[0.6]									
[50,35,20]	1000	0	0	0.80	2.82	0.81	2.87	41.74	96.07
[50,35,20]	2000	0	0	0.76	2.62	0.77	2.67	41.89	96.64
Weibull[0.5]									
[50,35,20]	1000	0	0	1.21	4.07	1.23	4.13	41.87	95.48
[50,35,20]	2000	0	0	1.18	4.06	1.21	4.14	41.90	95.51
Weibull[0.4]									
[50,35,20]	1000	0	0	1.90	5.97	1.95	6.10	41.78	96.29
[50,35,20]	2000	0	0	1.99	6.25	2.06	6.42	42.03	95.97
Weibull[0.3]									
50	1000	1960	1	1682	1791	13,525	45,695	172,992	604,800
[35,20]	1000	0	0	3.73	10.74	3.95	11.32	43.08	97.03
20	1000	0	0	3.72	10.67	3.92	11.23	43.05	96.97
[50,35,20]	2000	0	0	3.76	10.69	4.04	11.46	43.17	97.11
Weibull[0.2]									
50	1000	510	6006	62,318	288,351	161,592	438,771	560,465	604,800
35	1000	464	6063	8739	32305	100,859	281,669	551,542	604,800
20	1000	74	6040	3217	9354	42,429	141,007	514,255	604,800
50	2000	1137	3314	1463	4197	20,547	63,772	376,739	604,800
35	2000	0	13	9.30	25.39	22.29	34.69	1632	110.28
20	2000	0	0	9.11	24.73	12.40	33.53	50.27	108.27

JLIQ means the number of jobs left in queue without scheduling, and JTOS represents the number of jobs that ended up in timed out due to scheduling poor performance

5.2 Performance and energy-efficiency analysis of the *Bullfighter* policy

The second step in this work is the evaluation of the developed energy-efficiency policy presented in Section 4. The following values are considered for each *Bullfighter* parameter: (a) 1, 3, and 5 predictions are performed; (b) A [0.1–0.9] range with a 0.2 step is taken for threshold values; and (c) 15, 30, 45, and 60 s are taken for the future time window to be considered in predictions; and (d) 25, 50, and 100 past jobs are taken for prediction purposes. For the sake of clarity, in Table 5 we only present the most relevant results in terms of performance and energy consumption of the *Bullfighter* policy, employing the last 25 jobs (not a time window) to make the predictions, since the rest of values present a similar behaviour. The results of the Always Power off Policy, which tries to shut down every machine whenever possible are presented for comparison purposes. The extended data set can be found in Table 7 in Appendix.

For space reasons, in this table we only show one representative and simple scenario, which is configured as follows: (a) A monolithic centralised scheduler is used; (b) The data centre is equipped with 1000 machines; and (c) The data centre starts with 20% of its resources occupied. (d) No extreme values are considered, that is, an Exponential distribution is used to generate workload arrival times.

The results for the rest of scenarios presented in the previous section may be found as supplementary material.

Table 4 Performance impact of extreme values on Shared-state centralised schedulers

Prefill (%)	DC size	Failed trans.	$q_{(0)j}$		$q_{(n_j)j}$		$\mathcal{C}_j$	
			Avg.	90p.	Avg.	90p.	Avg.	90p.
Exponential								
[50,35,20]	1000	559	0.0004	0.00	0.0012	0.00	40.46	92.80
[50,35,20]	2000	393	0.0005	0.00	0.0008	0.00	39.90	91.07
Weibull[0.8]								
[50,35,20]	1000	772	0.0018	0.00	0.0028	0.00	40.40	92.36
[50,35,20]	2000	492	0.0020	0.00	0.0041	0.00	39.90	91.59
Weibull[0.7]								
[50,35,20]	1000	993	0.0050	0.00	0.0108	0.00	39.94	91.73
[50,35,20]	2000	682	0.0041	0.00	0.0081	0.00	39.91	91.84
Weibull[0.6]								
[50,35,20]	1000	1334	0.0114	0.00	0.0221	0.00	40.84	94.56
[50,35,20]	2000	857	0.0120	0.00	0.0194	0.00	39.83	92.80
Weibull[0.5]								
[50,35,20]	1000	1749	0.0336	0.00	0.0645	0.00	40.51	93.24
[50,35,20]	2000	1170	0.0284	0.00	0.0498	0.00	40.16	92.71
Weibull[0.4]								
[50,35,20]	1000	2225	0.0860	0.00	0.1629	0.00	40.80	93.18
[50,35,20]	2000	1534	0.0914	0.00	0.1437	0.00	40.25	92.91
Weibull[0.3]								
[50,35,20]	1000	2909	0.3188	0.78	0.5896	1.81	40.92	94.35
[50,35,20]	2000	2017	0.2781	0.59	0.4423	1.37	40.03	92.33
Weibull[0.2]								
[50,35]	1000	4092	1.6287	5.33	3.3886	10.19	43.35	99.84
20	1000	4071	1.6126	5.30	3.2521	9.95	43.20	99.65
50	2000	2869	1.4997	5.03	2.5863	8.02	41.71	95.69
35	2000	2848	1.4983	5.04	2.5794	7.98	41.71	95.70
20	2000	2808	1.4954	5.02	2.5607	7.95	41.67	95.71

Failed trans. the number of scheduling operations that ended up in a conflict due to more than one scheduler trying to deploy tasks on the same machine

The behaviour of the *Bullfighter* energy policy compared to that of the Always off policy depending on the number of predictions s, threshold D, and the future time window τ, is depicted in Figs. 5, 6, and 7, respectively.

In general, all the *Bullfighter* configurations perform a lower number of better-quality shut-down operations, therefore causing less hardware stress, and save a bit less energy compared to the Always power off policy. In terms of performance, the *Bullfighter* energy-efficiency policy always improve the results provided by the most aggressive policy. However, there are differences in terms of behaviour between various *Bullfighter* parameterisation.

Regarding the number of predictions s, the naive intuition would suggest that the higher the number of predictions, the lower the number of shut-downs, and the more homogeneous the behaviour. However, the results presented in Table 7 in rows #2, #22, and #42 (1, 3, and 5 predictions, respectively), clearly show that the variations in terms of performance, represented by the times in queue and makespan, and energy efficiency, represented by the number of shut-down operations and energy saving, do not present a clear pattern, as shown in Fig. 5. Hence, we can state that the number of predictions has little impact on the behaviour of the *Bullfighter* policy.

As for the impact of the threshold, the results presented in Table 7 in rows #42, #46, #50, #54, #58 (threshold 0.1, 0.3, 0.5, 0.7 and 0.9, respectively), show that the threshold has a low impact on the behaviour of the *Bullfighter* policy. On one hand, between 0.1 and 0.9 (rows #42 and #58, respectively), we can notice the following differences: (a) the number of shut-downs performed by threshold 0.9 is $\sim$ 1650 lower than that provided by threshold 0.1 ($\sim$ 10%); (b) the time jobs spend in queue until their first task is scheduled is $\sim$ 10% longer for threshold 0.1 compared to threshold 0.9 (0.44 vs 0.40 s); and (c) the time jobs spend in queue until their last task is scheduled is

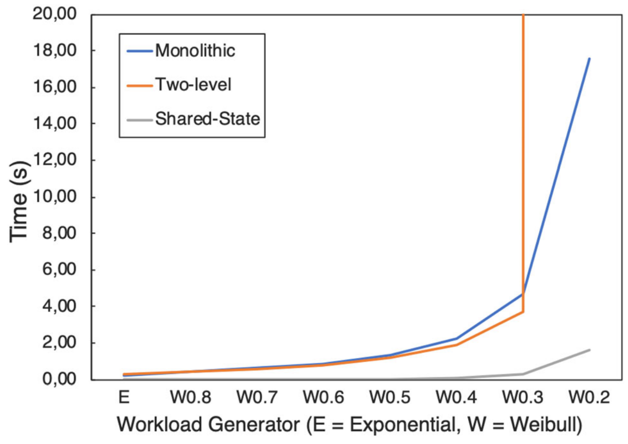


Fig. 4 Time in average jobs spend in queue until their first task is scheduled $q_{(1)j}$ per workload-generation distribution. This performance indicator is representative to the performance impact of extreme values of inter-arrival workload times. A data-centre size of 1000 machines and 35% of initial utilisation are employed to plot this figure (Color figure online)

$\sim$ 17% longer for threshold 0.1 compared to threshold 0.9 (1.46 vs 1.25 s). On the other hand, both the makespan and energy saving suffer almost not penalty.

As illustrated in Fig. 6, in general, the higher the threshold value, the higher the number of machines kept on in idle state to avoid fluctuations. Nevertheless, this pattern is not always homogeneous, and, even if present, the differences between them are low.

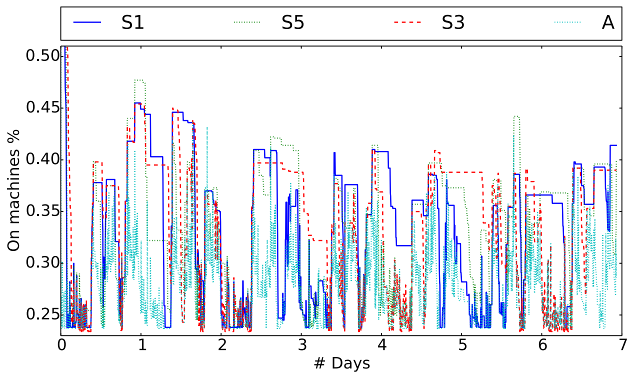


Fig. 5 Behaviour of Weibull energy policy depending on the number of predictions s. S1, S3 and S5 meaning 1, 3, and 5 predictions to make each shut-down decision, respectively. It is clearly shown that all of them can forecast the incoming workload reasonably well, preventing the system from the fluctuations present when the Always off policy is employed. Notwithstanding, the number of predictions has a minor impact and does not present a clear trend

The results presented in rows #[30–33] of Table 7, show the impact of the future time window τ considered for predictions in terms of performance and energy consumption. On one hand, when a short future time window of 15 s is used (row #30), *Bullfighter* is almost able to keep up with the energy savings results achieved by the Always off policy (57% vs 55%), but only performs approximately 55% of the shut-down operations. Hence, *Bullfighter* keeps servers down for 6.89 h, and saves 2.72 kWh per shut-

Table 5 Sample of the most relevant results in terms of performance and energy-efficiency for various *Bullfighter* configurations

#	s	D	τ	$q_{(0)j}$		$q_{(n_j)j}$		C_j		Sav. (%)	#shut(10^3)	KWh shut	h per shut
				Avg.	90p.	Avg.	90p.	Avg.	90p.				
1	Always off			0.56	1.86	2.38	6.58	46.50	109.50	56.81	30.64	1.57	3.97
2	1	0.1	15	0.42	1.16	1.39	2.57	43.57	101.14	54.58	16.91	2.73	6.92
22	3	0.1	15	0.41	1.13	1.25	2.49	43.65	101.46	54.68	17.56	2.64	6.67
30	3	0.5	15	0.41	1.17	1.25	2.56	43.27	100.62	54.65	17.00	2.72	6.89
31	3	0.5	30	0.38	0.95	1.02	1.77	42.31	98.15	53.29	11.48	3.93	9.95
32	3	0.5	45	0.32	0.71	0.65	1.14	41.42	95.63	51.35	8.52	5.10	12.92
33	3	0.5	60	0.32	0.65	0.62	0.97	41.06	94.99	49.98	5.88	7.20	18.24
42	5	0.1	15	0.44	1.22	1.46	2.77	43.71	101.07	54.76	17.43	2.66	6.73
46	5	0.3	15	0.40	1.19	1.20	2.69	43.73	101.67	54.97	18.54	2.51	6.36
50	5	0.5	15	0.41	1.16	1.23	2.50	43.19	100.40	54.76	15.86	2.92	7.40
54	5	0.7	15	0.41	1.15	1.22	2.64	43.19	100.61	54.80	16.45	2.82	7.14
58	5	0.9	15	0.40	1.11	1.25	2.36	43.22	100.40	54.77	15.78	2.94	7.44

The results of the Always power off policy are presented for comparison purposes. All the results can be found in Appendix, Table 7. Sav (%). represents the percentage of energy saved compared to that consumed by a system with no energy policies. #shut denotes the number of shut-down operations, related to hardware stress. KWh shut means the KWh saved on each shut-down operation, that is, the goodness of the shut-down operations. h per shut shows the hours machines keep off per shut-down operation. Long shut-down periods are related to higher energy efficiency, since switching on-off consumes even more energy than keeping the machine in idle state. s, D, and τ denote the number of predictions, the decision threshold and the future time window, respectively

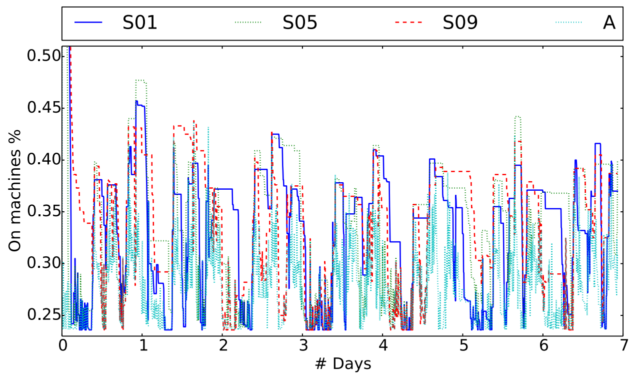


Fig. 6 Behaviour of Weibull energy policy depending on the decision threshold *D*. S01, S05, and S09 meaning 0.1, 0.5, and 0.9 threshold. In general, the higher the threshold, the more conservative *Bullfighter* is, as can be seen in the simulation start, between days 1 and 3, and in day 5. Nonetheless, this behaviour may not be always homogeneous, as seen in day 6

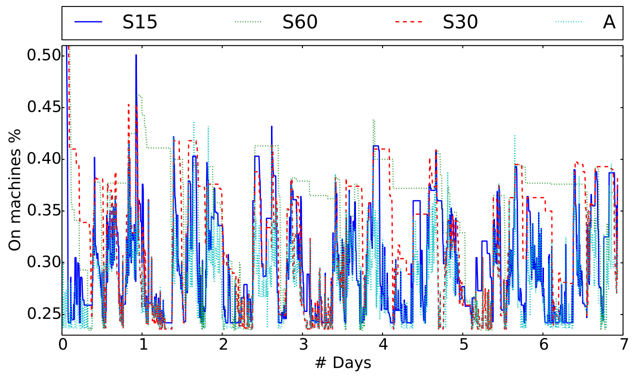


Fig. 7 Behaviour of Weibull energy policy depending on the future time window τ used for predictions. S15, S30, and S60 meaning a value of τ of 15, 30, and 60 s, respectively. In this case, it is clearly shown that, the longer the future time window under consideration, the more conservative *Bullfighter* is

down operation in average, compared to the 3.97 h and 1.57 kWh achieved by the Always power off policy. Moreover, this most-aggressive *Bullfighter* configuration cuts almost by half the time jobs spend in queue until their last task is scheduled $q_{(n_j)j}$, and $\sim$ 30% the time jobs wait in queue until their first task is scheduled $q_{(1)j}$. In terms of makespan $\mathcal{C}_j$, *Bullfighter* only shorten this time approximately 7%.

On the other hand, when the longest future time window (1 min) is considered (row #33), we find a more conservative behaviour. In this case, *Bullfighter* consumes 7% more energy than the Always power off policy, and 5% more than the most-aggressive *Bullfighter* policy (row #30), by performing only $\sim$ 20% and $\sim$ 30% of the shut-downs operations performed by Always power off and the most-aggressive *Bullfighter*, respectively. Therefore, this *Bullfighter* parameterisation is able to keep servers down for approximately three times longer (6.89 h. vs 18.24 h.), and saves three times more energy per shut-down operation than aggressive alternatives (2.72 kWh vs. 7.20 kWh). Moreover, this configuration reduces, in average, the queue times jobs wait for their first and last task to be scheduled by approximately 20% (0.41 s. vs 0.32 s.), and 50% (1.25 s. vs. 0.62 s.), respectively. Again, the impact in terms of makespan is lower, shortening this time only $\sim$ 6% (43.27 s. vs 41.06 s.). The results presented in rows #31, and #32, show a clear linear progression of the results as the value of this future time window τ increases. This very same trend is illustrated in Fig. 7.

It can be noticed in Fig. 7 between days 2 and 3, that *Bullfighter* policies which perform the shut-down decisions based on a fixed number of past jobs, present a limitation that is maximised when extreme values are present: when there is a sudden change of arrival rate, between high-arrival periods (peaks consisting in dozens of jobs per second) and low-arrival periods (no incoming jobs in minutes, for instance), the prediction module takes some of those last *n* jobs, which belong to the high-arrival period, as part of the relevant past decision period. As a result, this *Bullfighter* policy tends to keep more machines in idle state than that based on a past time window instead of a fixed number of past jobs.

As shown in Table 6, the *Bullfighter* policy which takes into account a past time window jobs (Bullfighter T) performs a higher number of shut-down operations, consequently saving slightly more energy, and impacting more severely in terms of performance than the *Bullfighter* that takes into account a fixed number of jobs (Bullfighter J). Regarding energy consumption, *Bullfighter* T performs $\sim$ 3400 shut-down operations compared to the $\sim$ 3300 performed by *Bullfighter* J in Exponential scenarios, and $\sim$ 8800 vs. $\sim$ 5000 in Weibull[0.2] environments. Furthermore, *Bullfighter* T saves 57.67% of energy compared to the 57.62% of *Bullfighter* J in Exponential environments, and 55.46% vs. 53.06% in Weibull 0.2 scenarios. Regarding performance, *Bullfighter* T makes jobs wait in queue in average 560 ms. and 1.32 s until their first and last tasks are scheduled, respectively, compared to the 590 ms and 1.26 seconds of *Bullfighter* J in Exponential scenarios. For Weibull 0.2 environments, *Bullfighter* T make jobs wait 37.65 and 101.49 s whilst *Bullfighter* T 28.03 and 73.96 s (− 30%). As depicted in Fig. 8, the more extreme the scenario, the more accentuated this trend becomes.

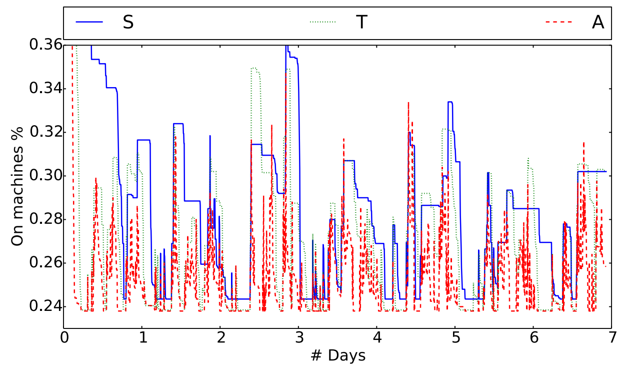
As a summary, in this section we have demonstrated the following behaviour of the *Bullfighter* policy according to its parameterisation:

(a) The impact of the number of predictions is almost negligible;

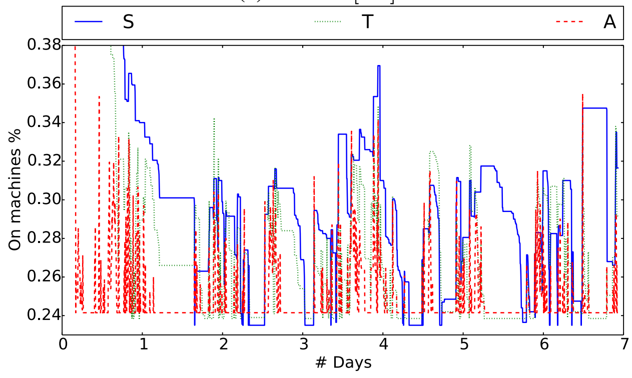
Table 6 Results in terms of performance and energy efficiency for the *Bullfighter* policy

Shut-down policy	$q_{(1)j}$		$q_{(n_j)j}$		C_j		Sav (%)	#shut (10^3)	kWh shut	h shut
	Avg.	90p.	Avg.	90p.	Avg.	90p.				
Exponential										
Always off	1.20	3.00	8.83	14.20	52.81	119.15	59.15	7.20	14.16	35.86
Bullfighter J	0.59	1.26	3.15	2.23	44.45	100.75	57.62	3.29	29.65	75.12
Bullfighter T	0.56	1.32	3.00	2.44	44.41	101.05	57.67	3.38	28.97	73.40
Weibull[0.5]										
Always off	4.55	13.62	24.17	68.62	79.06	157.50	58.94	16.62	6.03	15.27
Bullfighter J	2.01	5.69	5.86	8.33	47.63	103.72	56.24	5.24	18.35	46.50
Bullfighter T	2.20	5.96	6.87	10.22	46.29	106.32	56.63	6.26	15.45	39.13
Weibull[0.2]										
Always off	55.60	147.72	196.88	516.24	191.94	398.52	58.82	17.01	5.91	14.96
Bullfighter J	28.03	73.96	61.28	173.88	65.52	156.48	53.06	5.02	18.10	45.85
BullFighter T	37.65	101.49	106.13	301.96	136.10	237.33	55.46	8.77	10.88	27.57

Bullfighter J denotes the *Bullfighter* policy which employs a fixed number of past jobs (in this scenario 25), whilst *Bullfighter T* represents the one that considers the past jobs belonging to a past time window as relevant (5 min in this scenario). The number of predictions (5), the threshold (0.5) and the future time window (1 min) are set equal for both policies



(a) Weibull[0.8]



(b) Weibull[0.2]

Fig. 8 Comparison in terms of shut-down operations between the *Bullfighter* policy which employs a fixed number of past jobs, *Bullfighter* J (S), and the one employing a past time window, *Bullfighter* T (T). The Always power off policy (A) is presented as a reference. In general, *Bullfighter* T tries to adapt more to the current workload, which leads to a lower energy consumption and more performance impact due to workload fluctuations. It must be noticed that the more extreme the environment (Weibull[0.2]), the more marked the differences between both models

(b) The impact of the threshold is low, and is only present when a high number of predictions is considered; and
(c) The future time window used for predictions has a notable impact, both in terms of energy efficiency and performance, and presents a linear progression.
(d) *Bullfighter*s considering a fixed number of past jobs usually are usually more wary than those that take a past time window to make their predictions.

6 Conclusions and future work

In this paper we first characterise how extreme workloads impact on data-centres performance and energy consumption. As a result of this characterisation we can state that even though extreme workloads negatively impact in terms of performance, the impact is minor in shared-state resource managers and monolithic resource managers only outperformed the two-level model in very congested environments, which may be unable to handle this kind of extreme workload.

In addition, we extended our previous works by proposing a new energy-efficiency policy called *Bullfighter*. This policy employs queue-theory distributions to foresee workload demands even in extreme scenarios, while avoiding the fine-tuning required in previously developed policies.

In general, *Bullfighter* policy achieves better results in terms of balance between energy consumption and data-centre performance, since it performs a lower number of power-off cycles saving a similar amount of energy while maintaining QoS and SLA levels, even for data centres in great demand.

In summary, we presented the following novelties in this work:

- Characterisation of the impact in terms of performance and energy consumption of extreme workloads in large-scale realistic Cloud-Computing data centres, including several industry resource-managing models.
- A new energy-efficiency policy called *Bullfighter* which is able to automatically adapt to workload fluctuations even in extreme scenarios without fine tuning.

As future work, it would be interesting to explore:

- How a general computational intelligence-aided design framework could be utilised in the smart design process.
- How a dynamic change of resource managers could improve data-centre performance in extreme scenarios.

Acknowledgements Project RTI2018-098062-A-I00, funded by FEDER/Spanish Ministry of Science and Innovation—National Research Agency.

Appendix: Bullfighter parameterisation raw data

Table 7 Performance and energy-efficiency results of various *Bullfighter* configurations

#	s	D	τ	$q_{(0)j}$		$q_{(n_j)j}$		$\mathcal{C}_j$		Sav. (%)	#shut (10^3)	KWh shut	h per shut
				Avg.	90p.	Avg.	90p.	Avg.	90p.				
1	Always off			0.56	1.86	2.38	6.58	46.50	109.50	0.57	30.64	1.57	3.97
2	1	0.1	15	0.42	1.16	1.39	2.57	43.57	101.14	54.58	16.91	2.73	6.92
3	1	0.1	30	0.34	0.82	0.84	1.45	41.98	97.14	53.02	10.05	4.47	11.31
4	1	0.1	45	0.31	0.68	0.57	1.07	41.16	95.31	51.42	6.36	6.85	17.34
5	1	0.1	60	0.31	0.65	0.55	0.97	41.11	94.70	51.44	6.87	6.34	16.05
6	1	0.3	15	0.40	1.08	1.19	2.51	43.31	100.96	54.50	17.54	2.63	6.66
7	1	0.3	30	0.35	0.87	0.91	1.63	42.25	98.45	52.43	10.85	4.09	10.36
8	1	0.3	45	0.33	0.71	0.70	1.17	41.62	96.29	51.31	7.63	5.70	14.43
9	1	0.3	60	0.32	0.63	0.58	0.89	40.76	94.13	48.82	4.39	9.41	23.83
10	1	0.5	15	0.43	1.21	1.49	2.73	43.77	101.62	54.68	16.86	2.75	6.95
11	1	0.5	30	0.35	0.83	0.82	1.45	42.00	97.76	52.73	10.19	4.38	11.10
12	1	0.5	45	0.34	0.78	0.85	1.25	41.74	96.74	51.72	7.75	5.65	14.31
13	1	0.5	60	0.31	0.65	0.55	0.92	40.90	94.66	49.46	5.05	8.29	21.01
14	1	0.7	15	0.43	1.10	1.54	2.54	43.73	102.03	54.74	18.23	2.54	6.44
15	1	0.7	30	0.35	0.84	0.85	1.57	42.27	98.05	53.05	11.73	3.83	9.70
16	1	0.7	45	0.32	0.74	0.70	1.25	41.57	96.29	52.03	7.76	5.68	14.39
17	1	0.7	60	0.32	0.63	0.59	0.90	40.72	94.23	49.12	4.18	9.96	25.24
18	1	0.9	15	0.39	1.01	1.18	2.18	43.17	100.17	54.47	14.86	3.10	7.86
19	1	0.9	30	0.36	0.88	0.89	1.61	42.22	97.76	53.06	10.88	4.13	10.45
20	1	0.9	45	0.34	0.77	0.75	1.28	41.67	96.29	51.83	8.37	5.24	13.27
21	1	0.9	60	0.30	0.62	0.53	0.88	40.80	94.49	49.77	4.28	9.84	24.93
22	3	0.1	15	0.41	1.13	1.25	2.49	43.65	101.46	54.68	17.56	2.64	6.67
23	3	0.1	30	0.35	0.88	0.88	1.70	42.37	98.23	53.40	12.13	3.73	9.44
24	3	0.1	45	0.33	0.72	0.68	1.16	41.46	95.69	51.69	7.40	5.91	14.98
25	3	0.1	60	0.31	0.64	0.54	0.94	40.96	94.69	50.63	5.53	7.75	19.63
26	3	0.3	15	0.46	1.24	1.71	2.97	44.21	103.21	55.10	19.62	2.38	6.02
27	3	0.3	30	0.35	0.84	0.83	1.52	42.13	97.64	53.18	11.15	4.04	10.22
28	3	0.3	45	0.33	0.74	0.70	1.21	41.52	95.75	51.94	8.03	5.48	13.87
29	3	0.3	60	0.30	0.63	0.53	0.95	40.98	94.69	49.97	5.97	7.09	17.95
30	3	0.5	15	0.41	1.17	1.25	2.56	43.27	100.62	54.65	17.00	2.72	6.89
31	3	0.5	30	0.38	0.95	1.02	1.77	42.31	98.15	53.29	11.48	3.93	9.95
32	3	0.5	45	0.32	0.71	0.65	1.14	41.42	95.63	51.35	8.52	5.10	12.92

Table 7 (continued)

#	s	D	τ	$q_{(0)j}$		$q_{(n_j)j}$		$\mathcal{C}_j$		Sav. (%)	#shut (10^3)	KWh shut	h per shut
				Avg.	90p.	Avg.	90p.	Avg.	90p.				
33	3	0.5	60	0.32	0.65	0.62	0.97	41.06	94.99	49.98	5.88	7.20	18.24
34	3	0.7	15	0.39	1.06	1.17	2.17	42.98	99.97	54.32	14.76	3.12	7.89
35	3	0.7	30	0.36	0.89	0.90	1.73	42.36	98.45	53.50	10.81	4.19	10.61
36	3	0.7	45	0.34	0.78	0.75	1.39	41.84	97.46	51.80	8.74	5.02	12.71
37	3	0.7	60	0.31	0.64	0.59	0.90	40.98	94.64	48.81	5.23	7.90	20.02
38	3	0.9	15	0.41	1.14	1.30	2.51	43.35	100.89	54.71	15.79	2.93	7.43
39	3	0.9	30	0.35	0.84	0.85	1.54	42.27	97.97	53.03	11.17	4.02	10.19
40	3	0.9	45	0.33	0.74	0.74	1.28	41.71	96.57	51.77	9.22	4.75	12.04
41	3	0.9	60	0.31	0.64	0.58	0.93	41.04	94.80	50.10	4.80	8.83	22.38
42	5	0.1	15	0.44	1.22	1.46	2.77	43.71	101.07	54.76	17.43	2.66	6.73
43	5	0.1	30	0.36	0.89	0.93	1.71	42.54	98.45	53.87	12.38	3.69	9.33
44	5	0.1	45	0.34	0.75	0.74	1.22	41.48	95.73	51.84	7.26	6.05	15.32
45	5	0.1	60	0.32	0.67	0.66	1.02	41.12	94.93	49.42	5.30	7.90	20.01
46	5	0.3	15	0.40	1.19	1.20	2.69	43.73	101.67	54.97	18.54	2.51	6.36
47	5	0.3	30	0.37	0.92	1.02	1.82	42.74	99.06	53.63	12.51	3.63	9.19
48	5	0.3	45	0.34	0.79	0.74	1.42	41.73	96.80	52.38	9.42	4.71	11.93
49	5	0.3	60	0.32	0.68	0.63	1.01	41.17	95.07	49.43	6.71	6.24	15.80
50	5	0.5	15	0.41	1.16	1.23	2.50	43.19	100.40	54.76	15.86	2.92	7.40
51	5	0.5	30	0.34	0.82	0.80	1.45	42.00	97.39	52.66	9.81	4.55	11.51
52	5	0.5	45	0.33	0.74	0.73	1.16	41.32	95.54	51.60	6.64	6.58	16.65
53	5	0.5	60	0.32	0.68	0.59	1.00	41.01	94.90	50.26	5.44	7.83	19.83
54	5	0.7	15	0.41	1.15	1.22	2.64	43.19	100.61	54.80	16.45	2.82	7.14
55	5	0.7	30	0.37	0.94	1.01	1.74	42.41	98.56	53.17	10.98	4.10	10.38
56	5	0.7	45	0.35	0.82	0.82	1.31	41.61	96.24	51.96	7.53	5.84	14.79
57	5	0.7	60	0.32	0.66	0.59	0.98	41.08	95.02	50.81	5.78	7.44	18.85
58	5	0.9	15	0.40	1.11	1.25	2.36	43.22	100.40	54.77	15.78	2.94	7.44
59	5	0.9	30	0.35	0.82	0.91	1.52	42.10	97.61	53.09	10.89	4.13	10.45
60	5	0.9	45	0.33	0.74	0.68	1.16	41.37	95.72	51.34	7.16	6.07	15.37
61	5	0.9	60	0.32	0.68	0.61	1.04	41.16	95.47	50.23	6.07	7.01	17.76

The results of the Always poweroff policy are presented for comparison purposes. s, D, and τ denote the number of predictions, the decision threshold and the future time window, respectively

References

1. Beloglazov, A., Buyya, R.: Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Trans. Parallel Distrib. Syst. **24**(7), 1366–1379 (2013). https://doi.org/10.1109/TPDS.2012.240
2. Beloglazov, A., Buyya, R.: Openstack neat: a framework for dynamic and energy-efficient consolidation of virtual machines in openstack clouds. Concur. Comput. Pract. Exp. **27**(5), 1310–1333 (2015). https://doi.org/10.1002/cpe.3314
3. Benstock, D., Cegla, F.: Extreme value analysis (EVA) of inspection data and its uncertainties. NDT & E Int. **87**, 68–77 (2017)
4. Burns, B., Grant, B., Oppenheimer, D., Brewer, E., Wilkes, J.: Borg, omega, and kubernetes. Commun. ACM **59**(5), 50–57 (2016)
5. Cheng, Y., Anwar, A., Duan, X.: Analyzing alibaba's co-located datacenter workloads. In: 2018 IEEE International Conference on Big Data (Big Data), pp. 292–297 (2018). https://doi.org/10.1109/BigData.2018.8622518
6. Coles, S.G., Tawn, J.A.: Statistical methods for multivariate extremes: an application to structural design. J. R. Stat. Soc. Ser. C (Appl. Stat.) **43**(1), 1–31 (1994)
7. Duy, T.V.T., Sato, Y., Inoguchi, Y.: Performance evaluation of a green scheduling algorithm for energy savings in cloud computing. In: 2010 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), pp. 1–8. IEEE, Piscataway (2010)

8. Embrechts, P., Klüppelberg, C., Mikosch, T.: Modelling Extremal Events: For Insurance and Finance, vol. 33. Springer, Berlin (2013)
9. 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. Theory **82**, 160–173 (2018)
10. Fernández-Cerero, D., Fernández-Montes, A., Ortega, J.A.: Energy policies for data-center monolithic schedulers. Expert Syst. Appl. **110**, 170–181 (2018). https://doi.org/10.1016/j.eswa.2018.06.007
11. Fernández-Cerero, D., Jakóbik, A., Fernández-Montes, A., Kołodziej, J.: Game-score: game-based energy-aware cloud scheduler and simulator for computational clouds. Simul. Model. Pract. Theory **93**, 3–20 (2019)
12. Gumbel, E.J.: Statistics of Extremes. Courier Corporation, North Chelmsford (2012)
13. 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. NSDI **11**, 22–22 (2011)
14. Hüsler, J., Li, D.: Statistical analysis of extreme values with applications to insurance, finance, hydrology and other fields. In: Reiss, R.D., Thomas, M. (eds.) Statistical Analysis of Extreme Values with Applications to Insurance, Finance, Hydrology and Other Fields, pp. 144–151. Birkhäuser, Boston (2007)
15. Juarez, F., Ejarque, J., Badia, R.M.: Dynamic energy-aware scheduling for parallel task-based application in cloud computing. Fut. Gen. Comput. Syst. **78**, 257–271 (2016)
16. Kalid, S., Syed, A., Mohammad, A., Halgamuge, M.N.: Big-data NOSQL databases: a comparison and analysis of "big-table", "dynamodb", and "cassandra". In: 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), pp. 89–93 (2017). https://doi.org/10.1109/ICBDA.2017.8078782
17. Katz, R.W.: Statistics of extremes in climate change. Clim. Change **100**(1), 71–76 (2010)
18. Kotz, S., Balakrishnan, N., Johnson, N.L.: Continuous Multivariate Distributions: Models and Applications, vol. 1. Wiley, New York (2004)
19. Lee, Y.C., Zomaya, A.Y.: Energy efficient utilization of resources in cloud computing systems. J. Supercomput. **60**(2), 268–280 (2012)
20. Reiss, C., Tumanov, A., Ganger, G.R., Katz, R.H., Kozuch, M.A.: Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In: Proceedings of the Third ACM Symposium on Cloud Computing, p. 7. ACM, New York (2012)
21. Resnick, S.I., et al.: Heavy tail modeling and teletraffic data: special invited paper. Ann. Stat. **25**(5), 1805–1869 (1997)
22. Ricciardi, S., Careglio, D., Sole-Pareta, J., Fiore, U., Palmieri, F., et al.: Saving energy in data center infrastructures. In: 2011 First International Conference on Data Compression, Communications and Processing (CCP), pp. 265–270. IEEE, Piscataway (2011)
23. 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, pp. 351–364. ACM, New York (2013)
24. Shehabi, A., Smith, S.J., Sartor, D.A., Brown, R.E., Herrlin, M., Koomey, J.G., Masanet, E.R., Horner, N., Azevedo, I.L., Lintner, W.: United states data center energy usage report. Tech. Rep. (2016)
25. 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, pp. 8–14. ACM, New York (2016)
26. 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. **50**, 64–76 (2014)
27. 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, p. 18. ACM, New York (2015)

Damián Fernández-Cerero received the B.E. degree and the M.Tech. degrees in Software Engineering from the University of Sevilla. In 2018 he obtained his Ph.D. in Computer Science and was granted with a Marie Curie fellowship in Dublin City University. Currently he both teachs 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

Francisco J. Ortega-Irizo received the degree and M. degrees in Mathematics from the University of Sevilla. Currently he both teachs and conducts research at University of Sevilla. He has worked on several research projects supported by the Spanish government. His research interests include statistical models for prediction and performance evaluation.

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, I joined the Department of Computer Languages and Systems, University of Sevilla, and in 2013 became Assistant Professor. In 2008 and 2009 I was invited to the ENS-Lyon, in 2012 to the Universitat Politécnica de Barcelona and in 2016 to Shanghai Jiao Tong University for share experiences and knowledge in saving energy solutions for Data Centers. My research interests include energy efficiency in distributed computing, applying prediction models to balance load and applying on-off policies to Data Centers.

Francisco Velasco-Morente is a full professor from the University of Sevilla. Currently he both teachs and conducts research at University of Sevilla. He has worked on several research projects supported by the Spanish government. His research interests include statistical models for prediction and performance evaluation.