J Grid Computing (2013) 11:129–148
DOI 10.1007/s10723-012-9218-3

Assessing Green Strategies in Peer-to-Peer Opportunistic Grids

Lesandro Ponciano · Francisco Brasileiro

Received: 15 November 2011 / Accepted: 22 June 2012 / Published online: 11 July 2012

Abstract Opportunistic peer-to-peer (P2P) Grids are distributed computing infrastructures that harvest the idle computing cycles of computing resources geographically distributed. In these Grids, the demand for resources is typically bursty. During bursts of resource demand, many Grid resources are required, but on other occasions they may remain idle for long periods of time. If the resources are kept powered on even when they are neither processing their owners' workload nor Grid jobs, their exploitation is not efficient in terms of energy consumption. One way to reduce the energy consumed in these idleness periods is to place the computers that form the Grid in a "sleeping" state which consumes less energy. In Grid computing, this strategy introduces a tradeoff between the benefit of energy saving and the associated costs in terms of increasing the job response time, also known as makespan, and reducing the hard disks' lifetime. To mitigate these costs, it is usually introduced a timeout policy together with the sleeping state, which tries to avoid useless state transitions. In this work, we use simulations to analyze the potential of using sleeping states to save energy in each site of a P2P Grid. Our results show that sleeping states can save energy with low associated impact on jobs' makespan and hard disks' lifetime. Furthermore, the best sleeping strategy to be used depends on the characteristics of each individual site, thus, each site should be configured to use the sleeping strategy that best fits its characteristics. Finally, differently from other kinds of Grid infrastructures, P2P Grids can place a machine in sleeping mode as soon as it becomes idle, i.e. it is not necessary to use an aggressive timeout policy. This allows increases on the Grid's energy saving without impacting significantly the jobs' makespan and the disks' lifetime.

Keywords Energy saving strategies · Peer-to-peer Grids · Resource availability · Sleeping states

This work has been funded by CNPq/Brazil (grants 560262/2010-8 and 305858/2010-6) and the European Commission through the DEGISCO project (grant agreement nr RI-261561).

L. Ponciano · F. Brasileiro (✉)
Departamento de Sistemas e Computação,
Universidade Federal de Campina Grande,
Av. Aprígio Veloso, s/n - Bloco CO, 58429-900,
Campina Grande, PB, Brazil
e-mail: fubica@dsc.ufcg.edu.br

L. Ponciano
e-mail: lesandrop@lsd.ufcg.edu.br

1 Introduction

Opportunistic Grids are distributed computing infrastructures that harvest the idle computing

cycles of computing resources geographically distributed. This kind of system is suitable for the execution of the so called Bag-of-Tasks (BoT) applications—those parallel applications that can be divided into a large number of independent tasks which do not communicate with each other. Applications with these characteristics are abundant in a variety of settings, including data mining, simulations, fractal calculations, computational biology, computer imaging, among many others [11]. This, together with the current availability of an enormous amount of underutilized processing resources, has made very popular the use of opportunistic Grids to run scientific BoT applications [10].

Regarding the motivation that the resource owners have to make their resources available to the Grid, opportunistic Grids can be broadly divided in two classes: volunteer computing and Peer-to-Peer (P2P) Grids. In volunteer computing, resource owners have an altruistic behavior, and donate their resources without expecting any measurable return in exchange. This kind of Grid was popularized firstly by the pioneer SETI@home project,[1] and then by a number of subsequent projects that are powered by the BOINC middleware,[2] such as the World Community Grid [8]. On the other hand, in P2P Grids, resource owners expect to be able to have their donations reciprocated by the recipients of these donations. Examples of such Grids, are those powered by the OurGrid middleware [11].

These distinct motivations also lead to a different profile of both the system's resources and the demand for the resources that are available in the two classes of systems. In volunteer computing Grids, personal computers of home users comprise the majority of the resources made available, and the demand for resources is usually greater than the amount of resources available. If a project has no jobs, the volunteer usually migrates to a different project that can use the volunteer's resources. Moreover, volunteers may donate their resources to many projects. As a result, resources in volunteer computing systems are seldom idle. On the other hand, desktops connected by a local area network comprise the typical resources that are managed by each peer of a P2P Grid, and the demand for resources is typically bursty [20]. Bursty demand is characterized by brief periods of high demand for resources followed by, possibly long, periods of idleness [16]. The implications that these differences have in the energy efficiency of P2P Grids is the main issue that we investigate in this paper.

In P2P Grids, many Grid resources are required during bursts of resource demand, but on other occasions they may remain idle for long periods of time. If the resources are kept powered on even when they are neither processing their owners' workload nor Grid jobs, the Grid exploitation is not efficient in terms of energy consumption. Increasing the energy efficiency of this kind of infrastructure is an area of both economic [24] and environmental [9] interest. In this work we evaluate the use of "sleeping" states to reduce the energy consumed in the idleness periods.

Sleeping states are approaches founded on power management primitives [6, 9, 14], which allow a machine or part of it to be almost completely turned off, entering very low power consumption states, while all their functionalities are frozen. Sleeping states can be thought as deeper idle states, achieving higher energy savings [9]. In contrast to the benefit of energy saving, sleeping states present negative side effects in the form of increments on the jobs' response time, and a potential reduction on disk's lifetime [14, 35].

The extra cost on job's response time is due to the time taken to wake up the resource when a new task arrives [6, 15]. The reduction on disk's lifetime is due to the starting and stopping of the hard disk drives revolutions when sleeping states are used [35]. In order to reduce these costs, it is usually introduced a timeout policy together with the sleeping state [4, 6, 15]. This policy assumes that it is highly likely that a resource remains idle if it has been idle for the timeout time; thus, the resource sleeps only after a fixed inactivity time has elapsed. This minimizes the situations in which the resource is put in a sleeping state, only to be soon after brought back to an active state.

[1] Search for Extraterrestrial Intelligence, http://setiathome.ssl.berkeley.edu/.

[2] Berkeley Open Infrastructure for Network Computing, http://boinc.berkeley.edu/.

In this work, we evaluate the impact on energy savings, jobs' response time (also referred to as the jobs' *makespan*) and disks' lifetime if a P2P Grid uses sleeping states, instead of leaving the resources in an idle state when they are neither processing their owners' workload nor Grid jobs. We focus on two sleeping states: hibernate, a deep sleeping state, and standby, an intermediate sleeping state between hibernate and the idle state. These states are differentiated by the power consumption and the time to wake-up, or latency; the deeper sleeping state has the least power consumption and the longest latency. We analyze how the typical demand and resource availability in P2P Grids have impacts on the sleeping states effectiveness. In addition, we evaluate the minimum amount of machine idle time that is required before a sleeping state should be applied.

Our method is based on trace-driven simulation. First, we characterize P2P Grid resource availability in traces of real systems that represent typical sites of a P2P Grid. This characterization allow us to fully understand the potential of use of sleeping states to save energy in this kind of system. Then, we use trace-driven simulation to analyze the tradeoff between the benefit of energy saving and the associated costs in terms of increasing the job's makespan and the effects on hard disks' lifetime. Finally, we perform a sensitivity analysis in order to measure how sensitive are the results to the parameters used.

Our main results are:

- We found that in P2P Grids the lengths of machine availability interval vary significantly with the Grid site. In some sites the machines present small intervals of availability, while in other sites this interval is longer. This difference impacts on the sleeping states effectiveness. In Grid sites where the machines have small availability intervals, standby achieves higher energy savings than hibernate, but in Grid sites where the machines have longer availability intervals, hibernate achieves higher energy savings than standby. Therefore, sites of a P2P Grid should, autonomously and individually, define the sleeping strategyy that best suits their characteristics.
- We found that the use of sleeping strategies during periods of low contention for resources in a P2P Grid may result in fewer hard disk transitions than it would occur if the resource owner, instead of donating his machine to the Grid, adopted a strategy to place it in a sleeping mode, when it was idle. This is the case because, when sleeping strategies are used by the P2P Grid, there are sessions of availability where the machines remain running a Grid job and they do not sleep.
- In Grid infrastructures in which resources are dedicated to the Grid at all times, a timeout period is commonly used to avoid useless sleep transitions [13, 25, 33]. We identified that in P2P Grids this is not the case, i.e. it is not necessary to wait any timeout period in the idle state, before setting a resource to sleep. This increases the energy savings without impacting significantly on the job's makespan and the disks' lifetime.

The remainder of this paper is organized as follows. In the next section, we present background information on energy efficiency and discuss the related work. In Section 3, we present the simulation model that we have used to evaluate the effectiveness of sleeping states in P2P Grids. Next, we present the experimental setup (Section 4) that allows us to analyze the tradeoff between the benefit of energy saving and its associated costs. The simulation results are presented and discussed in Section 5. Finally, in Section 6, we present the conclusions of this work.

2 Background and Related Work

In this section, we provide some background on the problem and discuss related work. We first introduce sleeping states highlighting the main tradeoffs that are relevant to this study. Next, we discuss related work on the evaluation of sleeping states effectiveness in several kinds of systems.

2.1 Background

In a P2P Grid, the resources are used to run Grid applications during their *intervals of avail-*

ability. An availability interval is a continuous time in which the machine is available to be used to run Grid applications, i.e. the owner is not using the resource. It is important to notice that for a machine to be used by the P2P Grid, the machine owners' have to disable the power management performed by the machine's operating system when the owner is not using it; otherwise, the operating system can set to sleep a machine that is executing a Grid job.

In this work we propose that a P2P Grid implements sleeping states using the Advanced Configuration and Power Interface (ACPI) [14]. ACPI is an open industry specification that defines power states of a whole computer system as the power states of each device. In this work, we focus on sleeping states specified on ACPI, given that this specification is used by several popular operating systems that normally execute on P2P Grid resources (e.g. Microsoft Windows [23], and Linux distributions [26]). An ACPI-compliant system has five sleeping states that are differentiated by the power consumption and the latency to wake-up.

In order to save energy, sleeping states normally interrupt active network connections when the machine is asleep, but some sleeping states allow machines to be waken-up by receiving a Wake-on-LAN packet ("magic packet" [2]) through the network interface. In a P2P Grid, wake up by network interface is required to allow the Grid middleware to manage sleeping states "on-the-fly" and without human intervention.

In this work, we focus on two ACPI sleeping states[3] capable of Wake-on-LAN: *standby* (S3), and *hibernate* (S4). Standby, or suspend-to-RAM, is a low-latency sleeping state, since the RAM is kept powered on. On the other hand, hibernate, or suspend-to-disk, consumes less power, but requires a longer latency in order to transit the machine to the operational state, since information needs to be retrieved from the disk.

To be advantageous to use sleeping states in a system, the power saved by the sleeping state needs to amortize the additional power consumption occurred during the transit time to enter/exit the state. Furthermore, the transition time may impact on the makespan of the Grid applications. Consequently, it generates a tradeoff between the benefit of energy saving and the associated cost in terms of increasing applications' makespan.

Another sleeping-state issue is disk reliability [35]. The hard disk devices have a limited number of spin-up cycles due to the wearing of the device's mechanical elements, such as bearings. Manufacturers design their disks to support a certain number of cycles, e.g. Seagate SATA hard disks are designed to support 50,000 spin-up cycles in a lifetime of 5 years [32, 35]. This is equivalent to a daily average of 27 spin-up cycles. Both standby and hibernate have impacts on disk lifetime, since each transition from a sleeping state leads to an extra spin-up cycle.

The sleeping states costs', delay and disk reliability, have been studied in several systems, such as: personal computers [25, 33], clusters [20, 27] and mobile computing [6]. In these systems, in order to reduce these costs, it is usually introduced a timeout policy together with a sleeping state [4, 6, 15]. The timeout policy defines the length of an inactivity period before which a sleeping state should not be applied. This policy assumes that it is likely that the machine remains sleeping for a long time if it has been idle for the inactivity time period.

In this work, we evaluate the effectiveness of such a timeout policy used together with sleeping states in P2P Grids. Furthermore, we analyze the sleeping states tradeoffs in this kind of Grids.

2.2 Related Work

Several studies deal with reduction of energy consumption using sleeping states in office environments [7], clusters [20], and other kinds of Grid infrastructures, such as infrastructures with resources dedicated to the Grid at all times [13, 25, 33], infrastructures with homogeneous resources [33], and/or infrastructures where resources are subject to previous reservation [25, 33].

Schott and Emmen [31] take a broader approach to investigate whether volunteer computing Grids, due to their distributed nature, have

[3]We highlight only the ACPI states relevant to this study; for information about other states, see the ACPI specification [14].

an overall green advantage over service Grids. They also discuss several approaches that can be used by volunteer systems to optimize the energy efficiency of these systems, including the use of sleeping states.

In this work, we analyze these states in P2P Grids. Differently from those infrastructures where the resources are dedicated to either their owners or to the Grid infrastructure, in P2P Grids, overtime, each resource can be used both by its owner, as well as by Grid users when it is not being used by its owner. Thus, a P2P Grid resource in a sleeping state can wake-up both when the owner returns to use it, as well as when it is allocated to run a job submitted by a Grid user. How this difference impacts on sleeping states effectiveness when they are used in P2P Grids resources is the issue addressed in this work.

Previous studies on Grids have focused on analyzing how machine availability and Grid workload impact on opportunistic Grid performance such as: applications' makespan, and cycles of CPU and/or bandwidth available [29] or processing waste caused by task failures [12, 18, 19]. However, to the best of our knowledge, no previous study on P2P Grid has investigated the impact of Grid workload and machine availability on P2P Grids' energy efficiency, and evaluated the use of sleeping states to increase this efficiency. Our initial study [28] has analyzed the impact of sleeping states on one administrative domain of an opportunistic Grid. The present work complements this effort by: (i) analyzing the break-even time and the maximum potential of sleeping states to save energy in P2P Grids, (ii) addressing the sleeping states tradeoff in terms of the benefit of energy saving and its associated impacts on job makespan and disks' lifetime, and (iii) investigating how timeout policies can be used to mitigate this tradeoff.

3 Estimating Energy Consumption, Job Makespan and Disk Reliability

This section details our simulation model for evaluating the effects of sleeping states on energy consumption, jobs's makespan, and disk's reliability in P2P Grids.

3.1 P2P Grid Overview

A P2P Grid represents a federation of resources and users that belong to different administrative domains (or sites). Each site manages a number of machines that it offers to the P2P Grid when there is no local demand for them. This enables a site to give to its local users access not only to its local machines, but also to the machines offered by the other sites. The design of our P2P Grid simulator is based on the OurGrid middleware [11]. We model the three main components of an OurGrid system, namely: peer, worker, and broker.

In the OurGrid middleware every resource in a Grid site is managed by a *Peer*. In each Grid site, only the machine where the peer runs is a dedicated resource to the Grid (although it can be shared with other services of the site), and all other machines in the site are made available to the Grid only when its owner is not using it; thus, the usage patterns of the site's users defines the availability patterns of the Grid machines of that site. We model the Network of Favors [3], which is the prioritization mechanism used by the OurGrid peers to decide how to allocate their idle resources when they are simultaneously requested by more than one peer. This mechanism is crucial to create an incentive for the donation of resources among peers.

The *Worker* is the component that executes on Grid resources that are used to run tasks of Grid jobs. The worker monitors the resource utilization and detects when the owner is not using it. Upon detecting that a resource is not in use by the owner, the worker passes this information on to the peer to which it is connected. Resources can be claimed back by their owners at any time. If this occurs while the worker is running a task of a Grid job, we assume that the computation is checkpointed and restarted later at another worker that is available to the Grid. We decided to assume checkpoints in order to speed up our simulations. In our evaluation, checkpoints do not alter the relative performance of the mechanisms studied given that chekpointing is used both in the baseline scenarios, as well as in the evaluation scenarios.

The *Broker* component provides an interface that Grid users use to submit their jobs to the P2P

Grid. Brokers are also responsible for scheduling jobs on the Grid workers. When a Grid user submits a job, the broker that runs at its site requests workers from the local peer (the site's peer). The local peer returns to the broker the local workers that are available, and, if these are not enough to fulfill the broker's request, it contacts the other peers to discover workers that are available at remote sites and that can be used to fully or partially fulfill the request. When the workers are received, the broker schedules the job's tasks to execute on these workers.

3.2 Energy Saving Strategies

In this work, we focus on the use of sleeping states in Grid resources that execute a work agent. Our model aims to evaluate the energy consumption of these resources due to their exploitation by the Grid. For this reason, we model only the machine's energy consumption during the time that it is available to be used by the Grid, i.e. during availability intervals. During these intervals, a machine can be in one of following four states: active (a), idle (i), standby (s) or hibernate (h). Each of these states is associated to a time spent in the state (Δ), a power consumption (P) in Watts, and a latency (L) to transit to/from any other state. Let P_a, P_i, P_s, and P_h be, respectively, the power consumption associated to the active, idle, standby and hibernate states. Also, let L_{s_1,s_2} be the latency associated with the transition from state s_1 to s_2.

The *active state* indicates that the machine is executing a task submitted by a Grid user. A machine is in the *idle state* when it is available to the Grid, but there are no Grid tasks currently allocated to it. The latency incurred to change the machine to/from idle to the active state is negligible and is considered to be zero ($L_{i,a} = L_{a,i} = 0$). Thus, in this state, a machine can start the execution of a new task as soon as it arrives.

The sleeping states *standby* and *hibernate* are states equivalent to the idle state. The difference is that these sleeping states save energy when compared to the energy consumption of a machine in the idle state. On the other hand, the latencies associated with the transitions from any of these states to any other state are not negligible, and respect the following relations: $P_h < P_s < P_i$, and $L_{h,a} = L_{a,h} = L_{i,h} > L_{s,a} = L_{a,s} = L_{i,s} > L_{i,a} = L_{a,i}$.

For each use of sleeping state, the energy consumed during the transitions is computed as $E_{\text{transitions}} = (L_{a,x} + L_{x,a}) \times P_a, x \in \{h, s\}$, which is the latency for a machine to sleep and wake up, consuming the power associated with the active state. It is important to notice that the higher the latency the greater energy cost of using the state. Therefore, the machine will need to remain asleep longer time for the transition cost to be paid and the sleeping state achieve energy savings. Another point is that state transitions should not be performed too frequently, otherwise they may increase the delay and reduce the disk's lifetime.

A timeout policy is usually used to mitigate the problem of excessive state transitions. The timeout policy defines an inactivity time (t) that is the minimum amount of machine idle time after which a sleeping state can be applied. A timeout is started when the machine is available to be used by the Grid and there is no task currently allocated to it. During the timeout period the machine remains in the idle state. The machine transits to a sleeping state if no Grid task arrives and the timeout expires. The energy consumed during the timeout period is computed as $E_{\text{timeout}} = \Delta_t \times P_i$, where Δ_t is the time spent in the idle state ($\Delta_t \leq t$).

For each use of the states standby and hibernate, the energy consumed during Δ_l units of time that the machine remains asleep is computed as $E_{\text{asleep}} = \Delta_l \times P_x, x \in \{h, s\}$, depending on the lower power consumption considered for the machine in the period of time Δ_l.

3.3 Simulation Dynamics

We use trace-driven simulation to compute the energy consumption, job makespan and number of state transitions when sleeping strategies and a timeout policy are used by a P2P Grid. The simulator receives as input a Grid configuration file, a trace that describes job submission, and a trace that describes the Grid machines' availability over time. The Grid configuration file describes the machines characteristics and how they are distributed among the Grid peers. The dynamic of the trace-driven simulation is as follows.

When a Grid user submits a job through its Grid broker, the broker requests workers from the local peer. The number of works requested is the number of tasks in the user's BoT job. If required, the peer wakes up machines that are in a sleeping state, in this case the peer first wakes up workers that were least recently sent to sleep. The peer requests workers to other Grid peers if the number of resources available is not enough to execute the current demand.

When the broker receives the workers, it uses a First Come First Served (FCFS) scheduling, which allocates the first task available to the first worker available. When a worker finishes a task execution, another task can be allocated to it or it is returned to the peer, if the broker has no more tasks to run. Upon receiving a worker, the peer could donate it to another broker or take the decision to switch it to a sleeping state, if there are no brokers with pending requests for workers. Depending on the simulation scenario, the peer can use a timeout policy before it triggers a transition to a sleeping state.

The simulation ends when all the jobs in the trace complete their executions. As the workload is processed, the simulator generates the following outputs: makespan of each job, transitions of each machine, and the energy consumption of each machine of each P2P Grid site. In the next section, we detail the simulation output.

3.4 Simulation Output

The impact of sleeping states on the disk lifetime is measured by the number of machine transitions to/from a sleeping state, which we generalize as *sleeping state transitions*. The amount of sleeping state transitions is enough to measure the impact of sleeping state on disk's lifetime given that each sleeping transition corresponds to the use of one start or stop in the hard disk rotation of those estimated for its entire lifetime.

We analyze the impact of sleeping states on jobs' makespan and energy saving comparing the Grid configuration in analysis with a configuration of reference. In this way, the energy savings strategies in a Grid configuration A is measured by comparing the energy consumption of this configuration with the energy consumption of a similar configuration of reference where the only difference is that machines are never placed in a sleeping state. This is expressed as $\xi_A = (E^{A,\text{idle}} - E^A)/E^{A,\text{idle}}$.

The makespan of a BoT job J_i is the difference between the earliest time of submission (e_i) of any of its tasks, and the latest time of completion of any of its tasks (l_i), more formally $m_{J_i} = e_i - l_i$ [17]. To measure the impact on the makespan when comparing the different strategies expressed in two Grid configurations, we use the slowdown metric, defined for a job J_i as: $s_{J_i} = (m^A_{J_i} - m^{A,\text{idle}}_{J_i})/m^{A,\text{idle}}_{J_i}$, where $m^A_{J_i}$ is the makespan for a Grid configuration A, while $m^{A,\text{idle}}_{J_i}$ is the makespan for the Grid configuration of reference.

4 Experimental Setup

Now, we turn to describe our method for analyzing the impact of sleeping states in a P2P Grid. In this section, we detail the data used in our trace-driven simulations, and the evaluated scenarios.

4.1 Datasets

As mentioned earlier, the simulator receives as input three kinds of files, as follows: (i) one Grid configuration file that represents the Grid infrastructure, (ii) one trace of machine availability that represents the variations in machines' availability over time, and (iii) one trace of BoT jobs, which we use to simulate the demand for resources generated by P2P Grid users.

In order to simulate a P2P Grid infrastructure, we used information of machines from three representative sites of the OurGrid Community P2P Grid,[4] namely: AESA, LCC, and LSD. The power consumption of each machine was retrieved from the specification provided by their manufacturers. We used this data to generate an empirical distribution function (EDF) that fits the power consumption and CPU speed of the machines considered. Then, we used this distribution to gener-

[4] http://status.ourgrid.org/

ate Grid infrastructures with different number of resources.

To estimate the energy consumption of processors, we considered the thermal design power (TDP) offered by the manufacturers AMD [1] and Intel [15]. TDP is the maximum power that a processor can draw for a thermally significant period while running commercially useful software. TDP is enough to represent the CPU power consumption given that tasks executed in P2P Grid are usually CPU-bound [11, 12, 19], i.e. they use the maximum CPU speed during their executions. The estimated power consumption of memory, disk, mother board, and the components that enable Wake-on-LAN were obtained from previous work [5, 15, 20, 21].

For simplicity, we considered that the power consumption and latencies associated with the standby and hibernate states are the same for all Grid machines. A machine in the state hibernate consumes the power required to enable the Wake-on-LAN devices (P_h), which is typically 0.70 W [15]. In standby the power consumption (P_s) is 3.33 W [15], given that both Wake-on-LAN devices and the memory remain on. The latency required to transit from any state to standby is 2.5 s ($L_{i,s} = 2.5$), and 55 s for the transition from any state to hibernate ($L_{i,h} = 55$) [25]. The latency associated to hibernate can be influenced by some settings or operating system dependencies. However, in this study we focus on the time required to reactivate the system state stored on the hard disk. We evaluate the impact on our results of variations on both power consumption and latency of sleeping states by performing a sensitivity analysis on these parameters.

To represent variations in machine availability over time, we used data from the OurGrid community. The data displays the use of each machine in each OurGrid site identifying the time when it starts and stops available and unavailable intervals. The raw data were collected from 120 machines in the AESA, LCC, and LSD sites, between April 1, 2011 and April 30, 2011. We selected data of these sites because they differ in several ways.

The AESA site aggregates machines of a dedicated cluster; therefore, these machines are available all the time (except for a few downtime periods). The LCC site aggregates desktops of a computing classroom. These desktops are used by students, and several students may use the same machine over the period of a day. Finally, the LSD site is composed of desktops of a research lab. In this lab, typically, each desktop is used only by one person over months. These differences on resource's use impact on the machine availability over time. As we discussed in Section 3, in P2P Grids the length of machine availability periods impacts the effectiveness of the sleeping state, since the transition costs need to be amortized by the energy savings achieved during these periods; the longer the period, the larger the energy savings.

In order to understand how generic are the lengths of machine availability in the datasets of the sites AESA, LCC and LSD from the OurGrid Community Grid, we compared them with those of other Grid infrastructures. We obtained traces of the following Grid infrastruces in the Failure Trace Archive:[5] (i) Condor_cae [36], a dataset that contains traces recorded by the Condor Grid middleware from the desktop systems at CAE platform; (ii) Condor_nd [30], a dataset that contains traces recorded by the Condor Grid middleware from the desktop systems at the University of Notre Dame; and, (iii) Entropia_sdsc [19], a dataset that contains traces recorded by the Entropia Grid middleware from the desktop systems at the San Diego Supercomputer Center (SDSC). These datasets were analyzed and characterized by previous studies [19, 30, 36].

The Empirical Cumulative Distribution Function (ECDF) presented in Fig. 1 shows the length of machine availability intervals of the sites AESA, LCC, LSD, Condor_cae, Condor_nd, and Entropia_sdsc. Using the two-sample Kolmogorov-Smirnov test, we measured the largest distance D between the empirical distribution of these datasets. We found that the distance D between AESA and Condor_cae is 0.1143, LSD and Condor_nd is 0.1178, and Entropia_sdsc and LCC is 0.1204. These values are less than 1.36, the upper bound that allows two datasets to be considered close for the significance level of 0.05 [34]. This indicates that our datasets are close to those

[5] http://fta.inria.fr

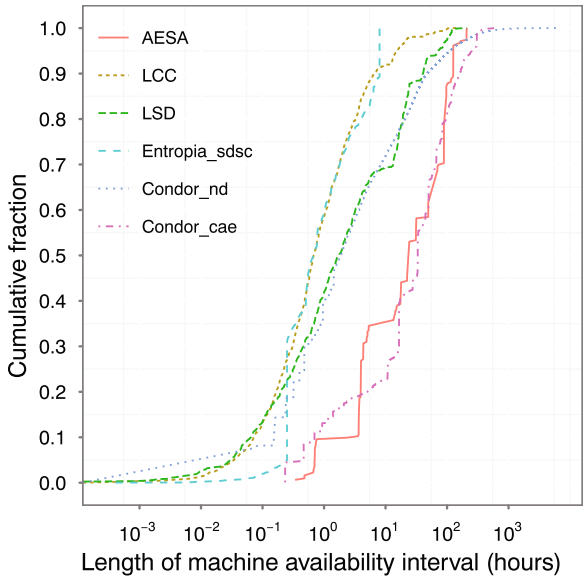


Fig. 1 Empirical Cumulative Distribution Function (ECDF) of lengths of machine availability intervals in the sites AESA, LCC and LSD, and in the Grid infrastructures: Condor_cae, Condor_nd and Entropia_sdsc

studied in previous work in terms of the lengths of machine availability intervals. The advantage of our datasets for the present work is that, in addition to machine availability, they also present data for speed and power consumption of each machine.

For the Grid workload, we decided to use traces generated using models of BoT applications. We generated workloads using a model of BoT applications proposed by Iosup et al. [17]. This model fits workloads of BoT applications executed in seven large Grid infrastructures. The model defines probability distributions for: BoT inter-arrival time (Weibull distribution with $\alpha = 4.25$ and $\beta = 7.86$), BoT size, i.e. the number of tasks in each job (Weibull distribution with $\alpha = 1.76$ and $\beta = 2.11$), and average task runtime and runtime variability for tasks belonging to the same job (normal distribution with $\mu = 2.73$ and $\sigma = 6.01$, and Weibull distribution with $\alpha = 2.05$ and $\beta = 12.25$, respectively). The BoT inter-arrival time distribution models both the inter-arrival time between consecutive BoT arrivals, and the variation caused by the daily submission cycle [17, 22]. We limited runtime of each task to 20 minutes to be suitable to execute it in machines exploited opportunistically [19]. Following these distributions, we generated 30 traces of BoT submission, each simulating BoT submissions in each Grid peer over a period of 30 days.

We used these traces to represent the demand for resources in the Grid. In average, the sum of all tasks runtime per trace in our 30 traces is 53,800,000 s with standard deviation of 1,792,870 s. Each second of task execution in the trace corresponds to one operation which is executed in one second in an Intel processor *Pentium* 4 of 1.8 GHz, our machine of reference. In order to compute the task's execution time in machines with other processor speeds, we considered that the execution time in a machine is proportional to its processor speed, as in Lammie et al. [20]. Thus, a machine with processor speed of x can execute $x/1.8$ operations per second.

In our experiments, we fixed the BoT trace and varied the amount of resources in the Grid in order to simulate different scenarios of Grid contention for resources. We compute the contention for resources in the Grid as the ratio d/r, where d is the amount of operations demanded by the jobs in the trace, and r is the amount of operations provisioned by all Grid machines' during their availability intervals in a period of 30 days. The closer to 1, the greater the contention for resources in the Grid. In our simulations, we varied the amount of machines in the Grid from 30 (yielding an average provision of 57,504,681 operations) up to 300 (yielding an average provision of 618,573,460 operations), leading to a variation on the Grid contention from 0.936 down to 0.087.

We simulated a P2P Grid infrastructure with 3 sites. These sites are comprised by machines with power consumptions, processor speeds and availability characteristics close to machines in the AESA, LCC and LSD sites. We varied the number of machines in the Grid from 30 to 300 varying the number of machines in each Grid site from 10 to 100. Every Grid site has the same number of machines in order to facilitate the simulation of specific number of machines in the Grid. The number of sites, and machines in each site, allow us to simulate the donation of resources between the Grid peers and several settings of Grid contention for resources.

4.2 Choosing Which Sleeping State to Use at Each Site

Each site of a P2P Grid may use a different sleeping state. The combinations of sleeping states and Grid sites in addition to the combinations required for analyzing timeout policies and Grid contention generate a prohibitive number of scenarios. We addressed this problem focusing only on three combinations of sleeping states and Grid sites; these combinations allow us to investigate the impact of sleeping states in the P2P Grid when: (i) every Grid site uses standby, (ii) every Grid site uses hibernate, and (iii) each Grid site uses the sleeping state that achieves the largest energy saving for that site during low-contention periods.

Normally, P2P Grids experiment a bursty demand, oscillating between periods of high and low contention for resources. During a burst of resource demand, many Grid resources are required, but on other times the Grid operates in low contention, and the resources may remain idle for long periods. These idle times are the main opportunity to save energy using sleeping states. The approaches where every Grid site uses the same sleeping state, either standby or hibernate, allows us to evaluate the impact of these sleeping states, while the approach where each site uses the sleeping state that achieves the largest energy saving during low-contention periods allows us to analyze the maximal potential of energy savings.

In order to choose for each Grid site which sleeping state achieves the largest energy saving during low-contention periods, we investigated the sleeping states that best fits the length of machine availability intervals in each site. We first computed the break-even time of sleeping states, i.e. the sleeping time that must elapse, so that hibernate achieves higher energy savings than standby. Then, in each Grid site, we computed the percentage of machine availability intervals that are longer than the break-even time. When this value surpassed the 50 percentile, we considered that hibernate was more suitable to be used in the site; otherwise, we considered that standby was more appropriate. The break-even time is computed by each Grid site in order to choose the sleeping state to be used in the site.

The break-even value of a site is a function of the average power consumption of the machines that comprise the site. To assess the impact that the average power consumption of a site has on the break-even value of different sites, we have considered the average energy consumption of the EDF used to select machines for a given site (referred as *Average*), as well as two extreme cases: (i) *High*, in which the power consumption of all machines in the site is equal to the power consumption of the machine with the highest power consumption among all machines found in the three representative sites of the OurGrid Community; and (ii) *Low*, in which the power consumption of all machines in the site is equal to the power consumption of the machine with the lowest power consumption in those peers. In all cases we have considered sleeping sections ranging from 0–6 h. In this analysis, the machine sleeps when the availability interval starts and it wakes up exactly when this interval ends. We measured the energy consumed during the sleeping time and the transition time.

Figure 2 shows the energy consumed by these states. The break-even time between standby and hibernate ($\Delta_{s,h}$) is the point where the two lines in each graph intersect. It increases when the average power consumption of the machines increases, ranging from 1.331 to 2.218 h. This indicates that, on average, hibernate achieves higher energy savings than standby for sleeping intervals longer than 1.886 h, and always achieves higher energy savings whenever the sleeping intervals are larger than 2.218 h.

Comparing the length of availability interval at each site (Fig. 3) and the break-even times (Fig. 2), we can see that more than 75 % of the intervals of the machines in the LCC site are smaller than the largest break-even time between hibernate and standby (2.218 h). This indicates that standby can clearly achieve greater energy savings than hibernate when used in machines of sites that have this kind of availability distribution. We can also see that approximately 50 % of the lengths of machine availability intervals in the LSD site are near the largest break-even time. Thus, in sites that have this kind of machine availability distribution, standby and hibernate will present similar energy savings in low-contention periods.

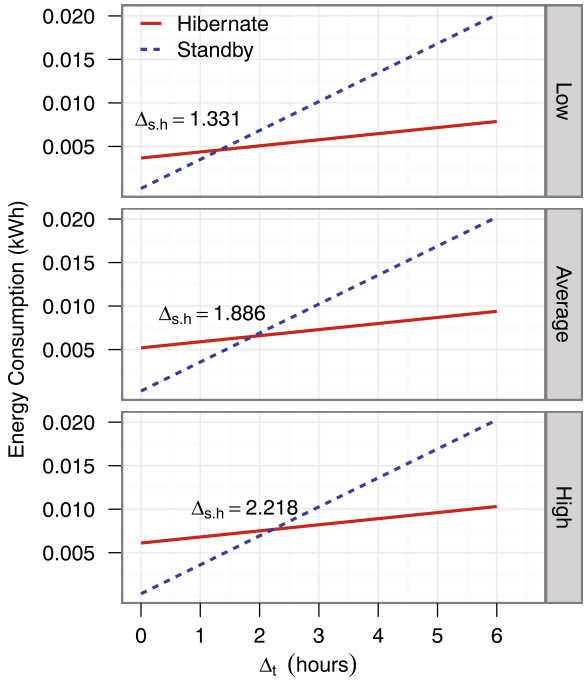


Fig. 2 Average of energy consumption as a function of the length of sleeping intervals (Δ_t) using the states standby and hibernate. The break-even time between standby and hibernate is indicated by $\Delta_{s,h}$

However, standby can reduce the impact on the jobs' makespan, given that it has a smaller latency. Thus, it is also preferable to use standby in this case. On the other hand, in the AESA site, 90 % of the intervals are longer than the largest break-even point between hibernate and standby, indicating that hibernate can achieve greater energy saving than standby in sites that have this kind of machine availability distribution.

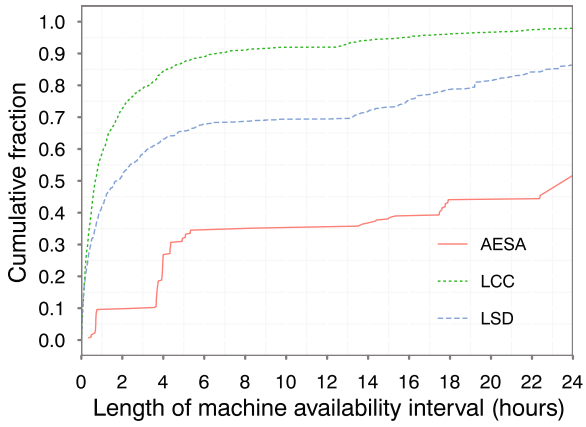


Fig. 3 Cumulative Distribution Function (CDF) of lengths of machine availability intervals in the AESA, LCC and LSD sites during April 2011. The *x*-axis is limited to intervals of up to 24 h

Based on the analysis presented in this section, we configured a P2P that uses hibernate in the AESA site, and standby in the LCC and LSD sites. Throughout this paper, we describe the latter Grid configuration as a hybrid approach. As we will detail in the next section, we compared this approach with the ones where every Grid site uses the same sleeping state (standby or hibernate).

4.3 Evaluation Scenarios

Our experiments have been designed to allow us to measure the impact on energy savings, jobs' *makespan* and disks' lifetime in the P2P Grid using standby, hibernate, or a hybrid configuration, together with a timeout policy, when compared to a Grid that leaves the resources in an idle state when it is neither processing their owners' workload nor Grid jobs. We considered three scenarios. For each scenario we varied a number of parameters. Table 1 summarizes the parameters that vary in each scenario.

In the first scenario, we evaluated the effectiveness of sleeping states in P2P Grids. In this scenario, when analyzing the impact on disks' lifetime, we focused on comparing the number of sleeping transitions the machines make when sleeping states are used by the P2P Grid with the number of sleeping transitions that would occur if the local user, instead of donating the machines to the Grid, adopted a strategy to place them in a sleeping state when they were idle. Additionally, in this scenario we also evaluate the impact of machine power consumption and CPU speed on sleeping state effectiveness. In this analysis, we vary the machines' power consumption and CPU speed in the situations presented in Section 4.2: Low, Average, and High.

In the second scenario, we evaluate the effectiveness of timeout values used together with sleeping states. The simulations performed in this scenario used basically the same setting of those used in the first scenario. The difference is that, in this scenario, together with standby and hibernate,

Table 1 Parameters

Parameter	Evaluation scenario		
	Sleeping states	Timeout	Sensitivity
#Machines in the Grid	30–300, 10 levels	30–300, 10 levels	150 (medium contention, *mc*), 300 (low contention, *lc*)
Grid approach	Standby, hibernate, hybrid	Hybrid	Standby, hibernate
Timeout value	0	0–1,200 s, 5 levels	0
Hibernate latency	55 s	55 s	Variation of 55 s in the range −100–200 %, 6 levels
Standby latency	2.5 s	2.5 s	Variation of 2.5 s in the range −100–200 %, 6 levels
Hibernate power	0.70 Watts	0.70 Watts	Variation of 0.70 Watts in the range −100–200 %, 6 levels
Standby power	3.33 Watts	3.33 Watts	Variation of 3.33 Watts in the range −100–200 %, 6 levels
Machines power and CPU speed	EDF, Low (1.8 GHz, 120 W), High (3.0 GHz, 200 W)	EDF	EDF

we evaluated timeouts values between 300 s and 1,200 s. The reason for this range is to simulate timeout values used in other kinds of systems [15, 20, 23]. We focused on analyzing how these timeouts impact the results obtained in the first scenario.

Finally, in the last scenario, we performed a sensitivity analysis. This analysis allow us to measure the impact on our results that may be due to variations on the values of sleeping states *power* and *latency*. We varied both parameters from −100 % up to 200 %, compared to the values used as input in the previous scenarios. We also analyze the following three cases where these parameters are related: (i) both latency and power varying together and in the same level; (ii) latency and power varying in an inverse level, e.g., latency increase 50 % and power decrease 50 %; and (iii) one parameter remain constant while the other vary. We analyzed the impact of these variations in a Grid configuration with low contention (*lc*, 150 machines) and in a Grid configuration with medium contention (*mc*, 300 machines).

We present results for a confidence level of 95 %. We performed simulations varying the trace of BoT jobs. We executed 30 simulations in the first and in the second scenario. In the sensitivity analysis a larger number of simulations (40) were required in order to obtain confidence intervals that allowed conclusions to be drawn. We have used the same distributions discussed in Section 4.1 in order to generate the extra 10 BoT submissions traces needed for this scenario.

The data that we have extracted from the traces, as well as our P2P Grid simulator are available for download and reuse at http://redmine.lsd.ufcg.edu.br/projects/green-grid. The data can also be viewed in the OurGrid Charts service at http://charts.ourgrid.org/.

5 Simulation Results

This section presents analysis of our simulations results on the effectiveness of sleeping states. Firstly, the analysis focuses on evaluating the use of sleeping states. Then, it is analyzed how the timeout policy impacts sleeping states costs and benefits. Finally, we analyze the sensitivity of the Grid's energy saving and jobs' makespan to variations on sleeping state power consumptions and latencies.

5.1 Impact of Sleeping Strategies in P2P Grid Sites

Now we analyze the energy saving in the Grid as a function of the number of resources provisioned, which indicates the contention for resources. Recall that the larger the number of machines in the Grid, the lower is the contention for resources. Figure 4 shows that the energy saving achieved

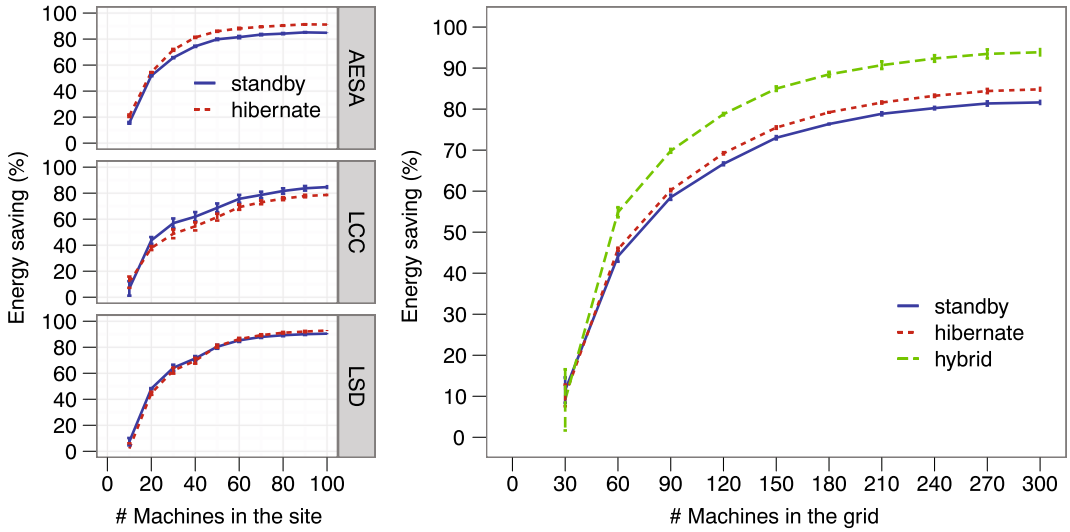


Fig. 4 Impact of sleeping-states approaches on energy saving in **a** each Grid site and **b** in all the Grid as function of the number of machines in the infrastructure

by the sleeping states increases when the contention for resources (the number of machines) in the Grid reduces. The saving surpasses 90 % for the lower contention level we analyzed (0.087). All approaches had similar results when the contention is very high. This is expected, since in this case, there is little room for energy savings using sleeping states. Standby and hibernate were quite similar in terms of energy saving, with hibernate becoming slightly better than standby when the Grid contention reduces; this is because during low contention for resources, the machines can sleep for periods of time longer than the break-even time.

We can also observe that a P2P Grid that uses a hybrid approach can achieve considerably higher energy savings than a P2P Grid where every Grid site uses the same sleeping state, either standby or hibernate. In the best setting, when contention is the lowest, the hybrid approach is able to achieve an additional 10 % in the total Grid's energy saving. Measurable differences are achieved even when the contention for resources in the Grid is relatively high. This result indicates that our strat-

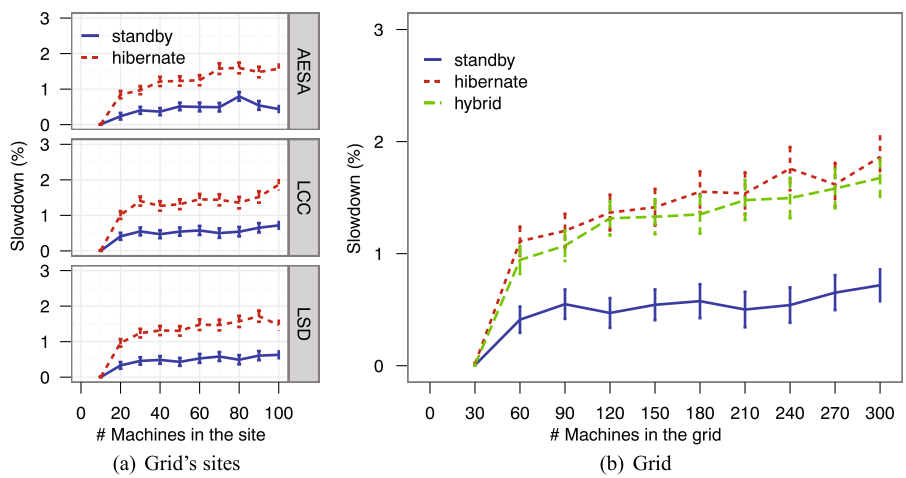


Fig. 5 Impact of sleeping-states approaches on jobs' slowdown in **a** each Grid site and **b** in all the Grid as function of the number of machines in the infrastructure

egy of using the break-even time and the length of availability interval is a simple way to increase the Grid's energy saving.

Figure 5 shows the jobs' slowdown graph for the P2P Grid when using standby, hibernate and hybrid. The main result in this figure is that all approaches present low slowdown. The maximum slowdown is less than 2.1 %, considering the upper limit of the confidence interval. This occurs because the delay that the sleeping state adds in the job execution time is relatively small compared to the overall job execution time.

Other important result is that the jobs' slowdown varies with the sleeping strategy used. The slowdowns in the approaches hibernate and hybrid is greater than in the approach standby. This result reveals the impact of the latency of the sleeping state used in the Grid approaches. The higher the latency, the higher the time the task waits the machine to wake up to start the execution. Hibernate has longer latency than standby. Hibernate generates the maximum slowdown (approximately 2 %) when the contention is the lowest. Standby presents low slowdown in all scenarios evaluated; its slowdown was always less than 1 %.

The hybrid approach combines the states standby and hibernate. As the tasks are distributed to machines in several sites of the Grid, the site that uses hibernate impacts the makespan of most of the jobs' submitted to the Grid. This explains why the slowdown of the hybrid approach is close to the slowdown of the hibernate one, even though only 1 out of the 3 sites in the hybrid Grid uses hibernate.

Regarding the impact of sleeping states on disks' lifetime, Fig. 6 exhibits the average number of sleeping transitions that occur in each P2P Grid site when the P2P Grid uses the hibernate, standby, hybrid, and when the owners of the machines use a green policy, i.e. instead of donating the machine to the Grid they simply put their machines in a sleeping state.

We first discuss the results for the LCC and LSD sites. For these sites we found that the use of sleeping strategies during periods of low contention for resources can result in fewer hard disk transitions than it would occur if the owner adopted a green policy (see Fig. 6). It happens because, when sleeping strategies are used by the P2P Grid, there are several availability intervals where the machines remain running Grid jobs until the owners reclaim their machines; thus, the machines never sleep. This does not occur when the Grid operates in a high or intermediate contention level. In this case, the number of sleeping

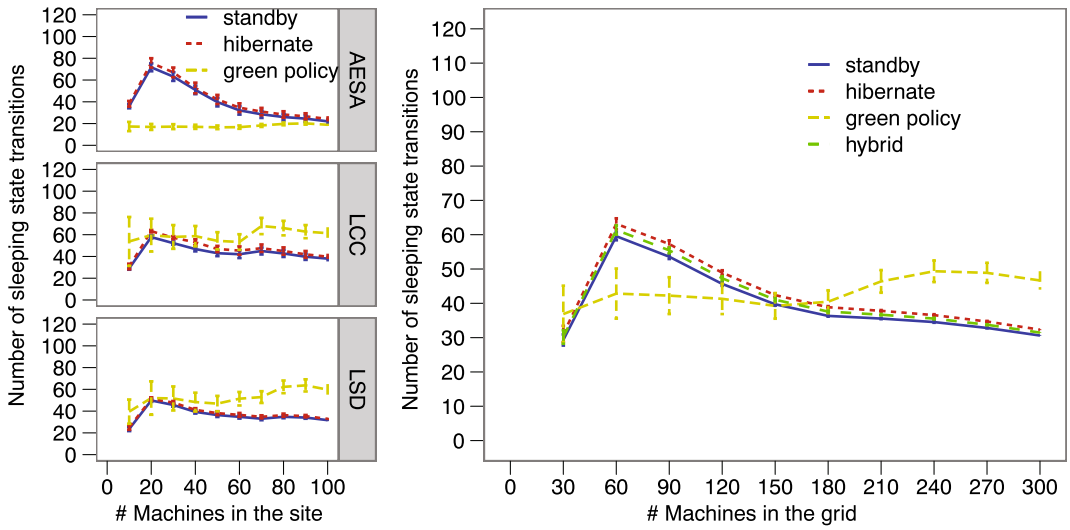


Fig. 6 Impact of sleeping-states approaches on the number of sleeping state transitions in **a** each Grid site and **b** in all the Grid as function of the number of machines in the infrastructure

transitions is similar to the one that occurs if the owners adopted a green policy. On the other hand, in the AESA site, the P2P Grid increases the number of sleeping transitions. This happens because the machines in the AESA site are dedicated to the Grid, i.e. they have no owner attached to them, therefore, they never sleep.

Regarding the impact of Grid contention on the number of sleeping transitions, we observed that the number of transitions increases when the amount of resources provisioned in each Grid site increases from 10 to 20, and then decreases for higher values of the amount of resources per site. This occurs because in the setting with 10 machines in each site, transitions to the sleeping state are rare, since there are always jobs in the queue.

In the intermediate contention levels (20 machines in each site) there are periods where there are no queues, and sleeping strategies are used; moreover, when a new job arrives, several machines need to be waked up. On the other hand, when the provision of resources increases, the average number of sleeping transitions reduces, because there are more machines available, distributing the wake ups among themselves.

In the AESA site, sleeping states generates more transitions; a maximum average value of approximately 75 in the intermediate contention level, or 2.5 transitions per day. In this site, we observed the maximum absolute value in all simulations: 16 sleeping transitions in one day. This occurs because AESA is a cluster and the resources are, normally, always available to the Grid. Thus, these resources are more susceptible to perform several sleeping transitions with variations in the Grid demand.

To understand the impact that these numbers of transitions have on hard disk's lifetime, we can compare the number of transitions observed during the simulation of one month with the number of transitions estimated by the manufacturers for an equal period. Normally, hard disks are designed to support 27 transitions daily, or 50,000 for the lifetime of 5 years [32, 35]. Therefore, approximately 810 transitions in one month. This value is more than 10 times the average number of transitions measured, and almost the double of the maximum number of sleeping transitions we observed in our simulations (480). This indicates that when sleeping states are used by the P2P Grid, the number of spin-up cycles induced by sleeping state transitions is one order of magnitude smaller than the number supported by current disks.

Finally, we also analyzed the impact of machines' power consumption and CPU speed on energy saving, jobs' makespan and number of sleep transitions. In this analysis, we kept the sites LCC and LSD configured as before, and we varied the configuration of the machines in the site AESA, so that they matched the *High*, and *Low* configurations earlier described. We analyzed scenarios of both high and low contention for resources, by keeping the same workload and increasing the number of machines in the Grid from 30 to 300. The results of these simulations showed no statistical evidences that the power consumption and CPU speed characteristics of the machines impact on the sleeping states' energy saving, jobs' makespan, and number of machine transitions.

5.2 Impact of a Timeout Policy on the Cost-Benefit Tradeoff of Sleeping States in P2P Grids

We turn now to analyze the impact of using a timeout policy together with sleeping states in a P2P Grid. During the timeout period the machine remains in the idle state; thus, it saves less energy, compared to the case where it is immediately transited to a sleeping state (see Fig. 7). Nevertheless, we observed that even with the use of time-

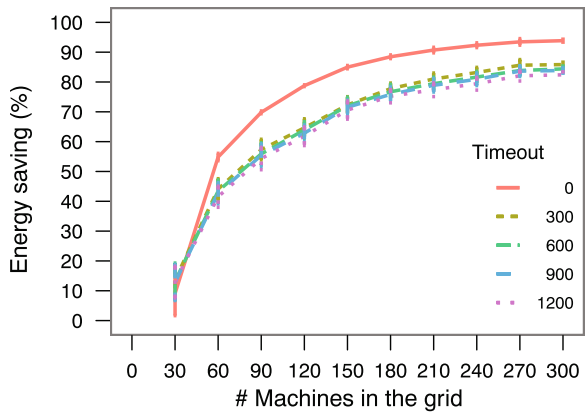


Fig. 7 Impact of timeout values on the energy saving achieved by sleeping states in a P2P Grid

outs, when a machine is put in a sleeping state, it experiments long sleeping intervals, generating substantial energy savings. We also observed that there is no difference in the energy savings when the timeout is increased from 300 to 1,200. This happens because when the timeout is longer than 300 s, in many cases, the machine is requested by the Grid or by the owner before the timeout expires.

Regarding the impact of the timeout policy in the number of state transitions, Fig. 8 shows that the timeout policy is effective in reducing the number of transitions. The reduction surpasses 50 % when the timeout is increased from 0 s to 1,200 s. In addition, as we discussed above, when the timeout is longer than 300 s, the reduction observed in the number of transitions reduces significantly given that, normally, the machine is requested by the Grid or by the owner before the timeout expires.

In general, this result indicates that this strategy can be used in situations where the number of state transitions needs to be reduced. Fortunately, as we discussed in the last section, our results show that in P2P Grids the number of sleeping transitions is less than the number that the machines are designed to support during their lifetime. Therefore, from this point of view, there is no need to use a timeout policy in a P2P Grid.

Finally, Fig. 9 shows the impact of timeout values on the jobs' makespan. We can see that

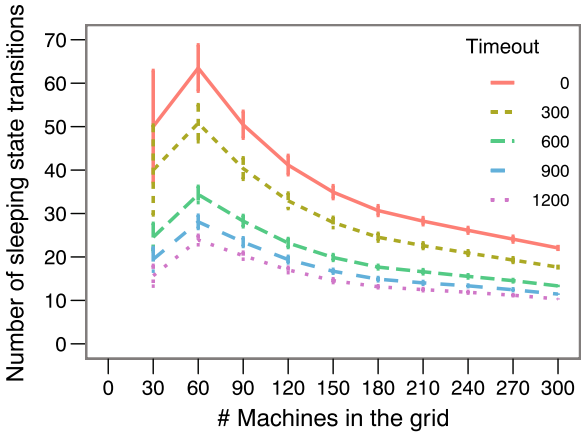


Fig. 8 Impact of timeout values on the number of sleeping state transitions

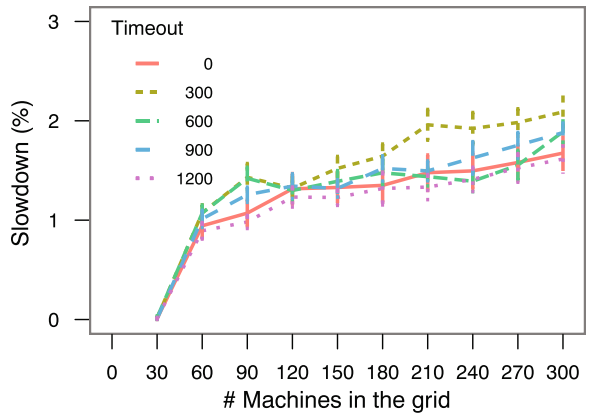


Fig. 9 Impact of timeout values on jobs' slowdown

the timeout does not impact significantly the job's makespan. Normally, when the Grid experiments a period of low contention for resources, the timeout expires and the machine enters a sleeping state. This occurs because the P2P Grid workload is typically bursty and the interval between bursts tends to be longer than the timeout value.

5.3 Parameter Sensitivity Analysis

We now turn to our sensitivity analysis. This analysis allows us to measure the impact on Grid's energy saving and jobs' makepan that is due to variations on the values of *power consumption* and *latency* of the sleeping states standby and hibernate. Variations in the parameters change the break-even time of standby and hibernate, for this reason, our analysis focuses only on the Grid configurations in which all Grid sites use the same sleeping approach, i.e. standby or hibernate.

We varied the power consumption and latency of standby and hibernate from −100 % up to 200 %, compared to the values used as input in our initial simulations. We analyzed three situations, as follows: (i) both latency and power varying together and in the same level; (ii) latency and power varying in an inverse way and in the same level; and (iii) one parameter remaining constant while the other varies. We analyzed the impact in two scenarios: low Grid contention for resources (*lc*), and medium Grid contention for resources (*mc*).

Figure 10 presents the sensitivity of energy saving. From this figure, we can note that variations only in latency (Fig. 10a) do not impact significantly the energy saving achieved by the sleeping states. On the other hand, variations in power consumption can lead to a difference of up to 6.8 % in the savings achieved by hibernate (Fig. 10b), when compared to the energy saving achieved in our reference setting ([0, 0]). When power consumption and latency vary together in the same levels (Fig. 10c), and in inverse levels (Fig. 10d), the impacts on the energy saving are similar to the case where only the power consumption varies, indicating that variances on the latencies have minimal impact on energy savings.

Variations in the power consumption of standby generates higher impact on the energy saving than variations in the power consumption of hibernate. This occurs because the power consumption of hibernate is very small, even with an increase of 200 % it does not affect significantly the energy saving. On the other side, standby is more sensible to variations in these parameters, given that its power consumption is larger.

Figure 11 shows the impact of variations in power consumption and latency of the sleeping states standby and hibernate on jobs' makespan. This figure shows that variations in the power consumption of both standby and hibernate have no impact on jobs' makespan (Fig. 11b), while variations in latency have a measurable impact (Fig. 11a). As power consumption has no impact on jobs' makespan, we omit figures that combine these parameters and focus on analyzing the impact due to variations on the latency.

Variations in latency of standby present a negligible impact on jobs' makespan, regardless of the Grid contention. In standby, the delay generated in jobs' makespan when the latency is increased in 200 % (from 2.5 s to 7.5 s) is less than 0.03 %. On

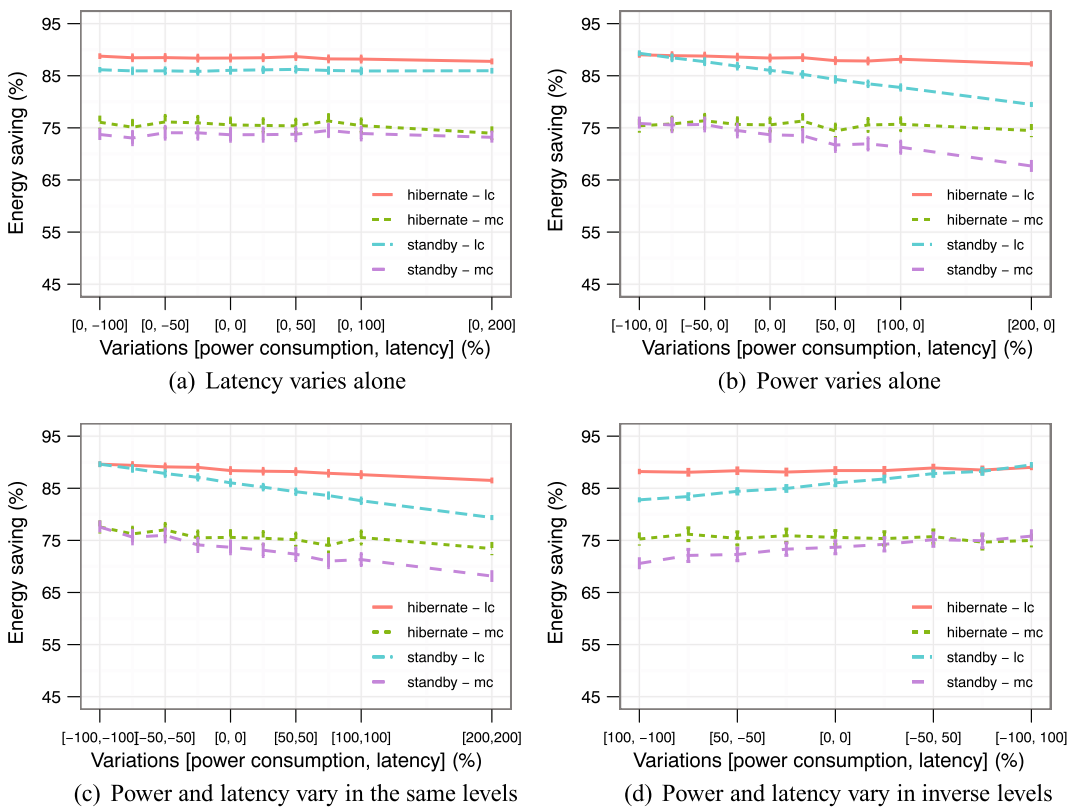
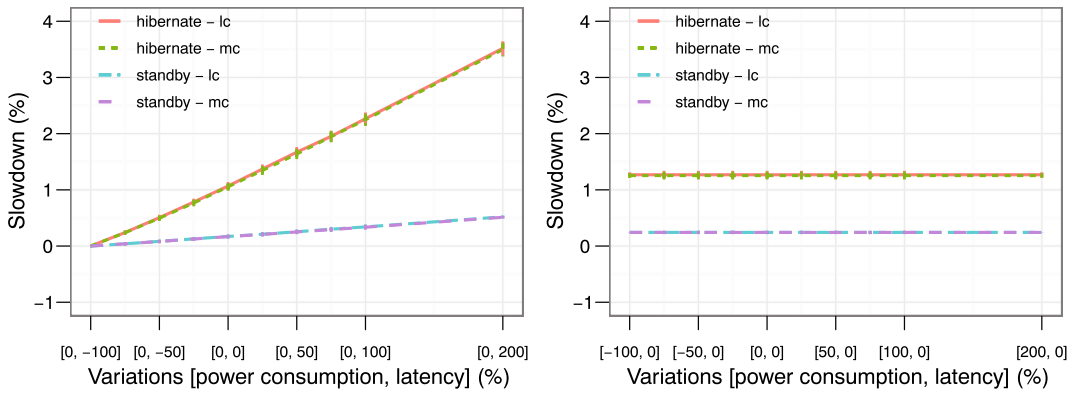


Fig. 10 Sensitivity analysis of Grid's energy saving to variations on the latency and the power consumption of sleeping states. The variation indicated by [0, 0] % uses the default values of power consumption and latency



(a) Variations in latency to wake-up (b) Variations in the power consumption

Fig. 11 Sensitivity analysis of jobs' slowdown to variations on the latency and power consumption of sleeping states. The variation indicated by [0, 0] % uses the default values of power consumption and latency

the other hand, variations in latency of hibernate generate a larger impact on the jobs' makespan. This impact indicates an increase of up to 2.5 % when the latency increases by 200 % (i.e. from 55 s to 165 s), which is still pretty small. In all scenarios, the delay generated by standby is less than the delays generated by hibernate, regardless of the Grid contention and the level of variation in power consumption and latency. As expected, when the latency reduces 100 % there is a small positive impact on the jobs' makespan for both sleeping states.

The results presented in this section show that the energy saving is more sensible to variations in the power consumption of the sleeping states, specially for the sleeping state standby. On the other side, the slowdown is more sensible to variation in the latency to transit to/from the sleeping state, specially for the sleeping state hibernate. Nevertheless, the largest impact we observed was a reduction of 6.8 % in the energy saving when the power consumption was increased 200 %, and an increase of 2.5 % in the jobs' makespan when the latency is increased 200 %, compared to our reference setting.

6 Concluding Remarks

In this paper we have evaluated the effectiveness of sleeping states and a timeout policy to save energy in P2P opportunistic Grids. When used in this kind of Grid, these strategies present a tradeoff between the energy saving potential and the associated negative impact on jobs' makepan and hard disks' lifetime. Our evaluation analyzes this tradeoff by using trace-driven simulations of a P2P Grid, exploring scenarios with realistic workloads.

We have focused on two sleeping states, denoted: standby and hibernate. We have found that both sleeping states can provide substantial savings in the energy consumption of P2P Grids. The savings can reach more than 85 % compared to the energy consumed when the resources are not set to sleep. These savings are attained with only a very small penalty in terms of the impact on jobs' makespan, and with no harm on disks' lifetime. The maximum associated increase of the jobs' makespan was of 2 %.

We have also found that the P2P Grid can achieve greater energy saving when a hybrid approach for setting the sleeping state is used; in this hybrid approach each Grid site uses the sleeping state that best fits the site, considering the expected length of machine availability interval. Our results show that this approach can further increase in up to 10 % the energy saving achieved when compared with the setting in which every Grid site uses the same sleeping approach, either hibernate or standby. Moreover, our results show that the average length of the availability interval of the machines and the break-even time of the sleeping states can be considered in order to

choose which sleeping approach to use in each Grid site.

Regarding the timeout policy, we have found that the value of the timeout does not impact significantly the energy saving and jobs' makespan generated by the sleeping states. However, this policy can reduce the impact on disks' lifetime. Fortunately, this policy is not necessary in P2P Grid sites given that our results show that in this kind of Grid the impact on disks' lifetime is less than that experienced by the machines in their normal operation if a green policy is used by their owners.

Our findings may be extended in various directions. First, given that our results show that the length of machine availability chiefly influences the effectiveness of the sleeping state used, future research could extend our evaluation exploring the effectiveness of using different sleeping states for machines in the same site. Furthermore, future research could explore how to predict the length of machine availability and the Grid workload in order to define the timeout to be used at each moment.

Acknowledgements Authors are indebted to Thiago Pereira and Lauro Costa for their comments and suggestions on earlier versions of this manuscript. Authors would also like to thank the anonymous referees for their insightful comments.

References

1. Advanced Micro Devices (AMD): Processor tech. support.amd.com/us/Processor_TechDocs/43375.pdf, online July 2011
2. Advanced Micro Devices, Inc.: Magic packet technology (1995). http://support.amd.com/us/Embedded_TechDocs/20213.pdf, online July 2011
3. Andrade, N., Brasileiro, F., Cirne, W., Mowbray, M.: Automatic Grid assembly by promoting collaboration in peer-to-peer Grids. J. Parallel Distrib. Comput. **67**(8), 957–966 (2007)
4. Augustine, J., Irani, S., Swamy, C.: Optimal power-down strategies. In: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science, pp. 530–539. IEEE Computer Society, Washington, DC (2004)
5. Baliga, B.J., Ayre, R.W.A., Hinton, K., Tucker, R.S.: Green cloud computing: balancing energy in processing, storage, and transport. Proc. IEEE **99**(1), 149–167 (2011)
6. Benini, L., Bogliolo, A., De Micheli, G.: A survey of design techniques for system-level dynamic power management. IEEE T VLSI Syst. **8**(3), 299–316 (2000)
7. Berl, A., Race, N., Ishmael, J., de Meer, H.: Network virtualization in energy-efficient office environments. Comput. Netw. **54**, 2856–2868 (2010)
8. Bertis, V., Bolze, R., Desprez, F., Reed, K.: From dedicated Grid to volunteer Grid: large scale execution of a bioinformatics application. J. Grid Computing **7**, 463–478 (2009). doi:10.1007/s10723-009-9130-7
9. Bolla, R., Bruschi, R., Davoli, F., Cucchietti, F.: Energy efficiency in the future internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures. IEEE Commun. Surv. Tutor. **13**(2), 223–244 (2011)
10. Brasileiro, F., Andrade, N., Lopes, R., Sampaio, L.: Democratizing resource-intensive e-science through peer-to-peer Grid computing. In: Yang, X., Wang, L., Jie, W. (eds.) Guide to e-Science, Computer Communications and Networks, pp. 53–80. Springer, London (2011)
11. Cirne, W., Brasileiro, F., Andrade, N., Costa, L., Andrade, A., Novaes, R., Mowbray, M.: Labs of the world, unite!!! J. Grid Computing **4**(3), 225–246 (2006)
12. Cirne, W., Brasileiro, F., Paranhos, D., Góes, L.F.W., Voorsluys, W.: On the efficacy, efficiency and emergent behavior of task replication in large distributed systems. Parallel Comput. **33**(3), 213–234 (2007)
13. Garg, S.K., Buyya, R.: Exploiting heterogeneity in Grid computing for energy-efficient resource allocation. In: Proceedings of the 17th International Conference on Advanced Computing and Communications (2009)
14. Hewlett-Packard Corporation, Intel Corporation, Microsoft Corporation, Phoenix Technologies Ltd. and Toshiba Corporation: Advanced configuration and power interface specification. http://www.acpi.info/spec.htm, online July 2011
15. Intel and U.S. Environmental Protection Agency: Energy star* system implementation whitepaper. www.intel.com/cd/channel/reseller/asmo-na/eng/339085.htm, online July 2011
16. Iosup, A., Dumitrescu, C., Epema, D., Li, H., Wolters, L.: How are real Grids used? The analysis of four Grid traces and its implications. In: Proceedings of the 7th IEEE/ACM International Conference on Grid Computing, pp. 262–269. IEEE Computer Society, Washington, DC (2006)
17. Iosup, A., Sonmez, O., Anoep, S., Epema, D.: The performance of bags-of-tasks in large-scale distributed systems. In: Proceedings of the 17th international symposium on high performance distributed computing, pp. 97–108. ACM, New York (2008)
18. Kondo, D., Chien, A.A., Casanova, H.: Resource management for rapid application turnaround on enterprise desktop Grids. In: Proceedings of the 2004 ACM/IEEE conference on Supercomputing, SC '04, pp. 1–14. IEEE Computer Society, Washington, DC (2004)
19. Kondo, D., Taufer, M., Brooks III, C.L., Casanova, H., Chien, A.A.: Characterizing and evaluating desktop Grids: an empirical study. In: Proceedings of the

18th International Parallel and Distributed Processing Symposium (2004)
20. Lammie, M., Thain, D., Brenner, P.: Scheduling Grid workloads on multicore clusters to minimize energy and maximize performance. In: 10th IEEE/ACM International Conference on IEEE Grid Computing, pp. 145–152 (2009)
21. Lu, Y.H., de Micheli, G.: Adaptive hard disk power management on personal computers. In: Proceedings of the Ninth Great Lakes Symposium on VLSI, GLS'99, pp. 1–4. IEEE Computer Society, Washington, DC (1999)
22. Lublin, U., Feitelson, D.G.: The workload on parallel supercomputers: modeling the characteristics of rigid jobs. J. Parallel Distrib. Comput. **63**, 1105–1122 (2003)
23. Microsoft Corporation: Windows power management. http://www.microsoft.com/whdc/archive/winpowmgmt.mspx, online July 2011
24. Opitz, A., König, H., Szamlewska, S.: What does Grid computing cost? J. Grid Computing **6**, 385–397 (2008). doi:10.1007/s10723-008-9098-8
25. Orgerie, A.C., Lefevre, L., Gelas, J.P.: Chasing gaps between bursts: towards energy efficient large scale experimental Grids. In: International Conference on Parallel and Distributed Computing Applications and Technologies, pp. 381–389. IEEE Computer Society, Los Alamitos (2008)
26. Pallipadi, V., Starikovskiy, A.: The ondemand governor: past, present and future. In: Proceedings of Linux Symposium, vol. 2, pp. 223–238 (2006)
27. Pinheiro, E., Bianchini, R., Carrera, E.V., Heath, T.: Dynamic cluster reconfiguration for power and performance, pp. 75–93. Kluwer Academic, Norwell (2003)
28. Ponciano, L., Brasileiro, F.: On the impact of energy-saving strategies in opportunistic Grids. In: Energy Efficient Grids, Clouds and Clusters Workshop, Proceedings. 11th ACM-IEEE International Conference on Grid Computing (Grid 2010), pp. 282–289. IEEE Computer Society, Brussels (2010)
29. Ren, X., Lee, S., Eigenmann, R., Bagchi, S.: Prediction of resource availability in fine-grained cycle sharing systems empirical evaluation. J. Grid Computing **5**, 173–195 (2007). doi:10.1007/s10723-007-9077-5
30. Rood, B., Lewis, M.J.: Multi-state Grid resource availability characterization. In: 8th Grid Computing Conference (2007)
31. Schott, B., Emmen, A.: Green desktop-Grids: scientific impact, carbon footprint, power usage efficiency. Int. J. Scalable Comput. Practice Exp. **12**(2), 257–264 (2011)
32. Seagate Technology LLC: Product manual: St31000524as, st3750525as, st3500413as, st3320413as, st3250312as, st3160316as. http://www.seagate.com/www/en-us/support/document_library, online August 2011
33. Sharma, K., Aggarwal, S.: Energy aware scheduling on desktop Grid environment with static performance prediction. In: Proceedings of the 2009 Spring Simulation Multiconference, pp. 1–8. Society for Computer Simulation International, San Diego (2009)
34. Trivedi, K.S.: Probability and Statistics with Reliability, Queuing and Computer Science Applications, 2nd edn. Wiley, Chichester (2002)
35. Xie, T., Sun, Y.: Understanding the relationship between energy conservation and reliability in parallel disk arrays. J. Parallel Distrib. Comput. **71**, 198–210 (2011)
36. Yigitbasi, N., Gallet, M., Kondo, D., Iosup, A., Epema, D.: Analysis and modeling of time-correlated failures in large-scale distributed systems. In: 8th IEEE/ACM International Conference on Grid Computing (2010)