Usando as Estratégias Sobreaviso e Hibernação para Economizar Energia em Grades Computacionais Oportunistas

Lesandro Ponciano, Francisco Brasileiro, Jaindson Santana, Marcus Carvalho, Matheus Gaudencio

Universidade Federal de Campina Grande
Departamento de Sistemas e Computação
Av. Aprígio Veloso, s/n, Bodocongó
58.429-900 - Campina Grande, PB - Brasil
{lesandrop, fubica}@dsc.ufcg.edu.br, {jaindson, marcus, matheusgr}@lsd.ufcg.edu.br

Abstract

Nowadays, opportunistic grids are getting more and more popular. In these systems, idle time of computing resources is used to process third-party workloads. Thus, the energy efficiency of those grids is also an increasing concern. This paper evaluates two green computing strategies to reduce energy consumption in resources of an opportunistic grid, namely: Standby and Hibernate. Both techniques are used when a resource is idle, i.e., available to the grid, but there is no work to be processed. Our evaluation uses a simulation model to assess the cost incurred in terms of increased the response time (makespan) of the jobs executed, as well as the energy savings achieved. We simulate an opportunistic grid that uses both strategies and also a scenario without a green computing strategy. As expected, both techniques increased the makespan of the jobs executed, but improved energy savings when compared to the scenario that does not use a green computing strategy. Moreover, for the scenarios evaluated, the Standby approach resulted in greater savings and in a smaller impact on the application makespan, being a better strategy to be used in such grids.

Keywords: Green computing, energy saving, sleeping strategy, standby, hibernate, opportunistic grids.

Resumo

Grades oportunistas são sistemas computacionais que têm sido amplamente utilizados para execução de aplicações científicas. Nesses sistemas, o tempo ocioso dos recursos computacionais é aproveitado para executar aplicações de terceiros. Nos últimos anos tem aumentado também a preocupação com a eficiência energética dessas grades. Este trabalho avalia o impacto do uso de estratégias para diminuir o consumo de energia dos recursos de uma grade oportunista. Duas estratégias são analisadas: Sobreaviso (Standby) e Hibernação (Hibernate). Elas são utilizadas quando as máquinas estão inativas, i.e., estão disponíveis para a grade, mas não têm nenhuma tarefa para executar. Avaliamos, também, após quando tempo de inatividade (TI) as estratégias devem ser utilizadas. Como esperado, ambas as estratégias de computação verde impactam o tempo de resposta das aplicações executadas, mas reduzem o gasto da infraestrutura com energia. Nos cenários avaliados, a estratégia Sobreaviso resultou em uma economia de energia equivalente à estratégia Hibernação, mas em um menor impacto no tempo de resposta das aplicações. Os resultados obtidos mostram, também, que a estratégia pode ser utilizada tão logo a máquina torne-se inativa, não sendo necessário aguardar um tempo de inatividade. Isso aumenta a economia de energia e não impacta significativamente o tempo de resposta das aplicações.

Palavras-chave: Computação verde, economia de energia, estratégia de economia de energia, sobreaviso, hibernação, grades oportunistas.

1. INTRODUÇÃO

A evolução dos sistemas computacionais tem sido marcada pela busca por mais poder computacional a qualquer custo [9]. No entanto, com o aumento do poder computacional, aumentou-se também o consumo de energia desses sistemas e, por consequência, a emissão de dióxido de carbono (CO_2) no meio ambiente. Os problemas ambientais gerados pelo aumento da emissão de CO_2 e o custo financeiro ocasionado pelo consumo de energia têm impulsionado estudos que visam o desenvolvimento de mecanismos e tecnologias que façam uso mais eficiente da energia. Esses mecanismos e tecnologias são denominados de *computação verde* (*green computing*) [29, 9].

O objetivo de usar a energia de modo mais eficiente tem influenciado diversas áreas da computação, como, por exemplo, o projeto de hardware e a gerência de centros de processamento de dados (*datacenters*) [3] e grades computacionais de serviço [19, 25, 21, 22]. Este trabalho analisa o uso de estratégias para economizar energia em grades computacionais oportunistas. Os recursos pertencentes a este tipo de grade são utilizados de forma oportunista: se o recurso não estiver sendo utilizado pelo dono do recurso, o poder computacional disponível, porém ocioso, pode ser utilizado para executar tarefas de terceiros que tenham sido submetidas à grade.

Quando um computador está disponível para a grade, mas a grade não o utiliza, ele fica em estado `ocioso`. Ao longo do tempo, a permanência dos computadores nesse estado caracteriza ciclos de ociosidade na grade e desperdício de energia. Estudos anteriores mostram que esses ciclos de ociosidade são comuns em diversos tipos de grandes computacionais [13, 17]. Um computador ocioso apresenta menor consumo de energia do que quando está executando alguma aplicação, porém esse consumo de energia é ainda significativo [12].

Duas estratégias para reduzir o consumo de energia nesses ciclos de ociosidade são: Sobreaviso (`Standby`) e Hibernação (`Hibernate`). A estratégia Sobreaviso, também conhecida por Suspensão para a Memória de Acesso Aleatório (RAM, do inglês *Random Access Memory*), consiste em manter a memória RAM ativa e reduzir a atividade do disco rígido e do processador. Já a estratégia Hibernação, também conhecida por Suspensão para o Disco, consiste em salvar o estado da memória RAM no disco rígido, reduzir o uso de energia da RAM, do disco rígido e do processador. Essas estratégias são definidas pelo padrão de configuração avançada e interface de energia (ACPI, do inglês *Advanced Configuration and Power Interface* [8]) - padrão aberto que unifica a configuração dos dispositivos e a interface de gerência de energia pelos sistemas operacionais.

Essas estratégias desativam diversos dispositivos para reduzir o consumo de energia do computador, mas elas permitem que o computador seja reativado eletronicamente através de um comando a algum dispositivo mantido em estado de espera, como: teclado, *modem*, interface de rede local (conhecido como *Wake-on-LAN* - WOL) ou interface de Barramento Serial Universal (USB, do inglês *Universal Serial Bus*), que, ao ser acionado, coloca todo o computador novamente em atividade. O tempo gasto para colocar e retirar um computador do estado Sobreaviso é menor do que do estado Hibernação, uma vez que não é preciso mover os dados entre a memória RAM e o disco rígido. Por outro lado, um computador em Sobreaviso apresenta maior consumo de energia do que em Hibernação, uma vez que a memória RAM permanece energizada.

O uso dessas estratégias em uma grade computacional oportunista, requer uma avaliação tanto da redução do consumo de energia obtida, quanto do custo associado a essa economia em termos do aumento no tempo de resposta das aplicações (também conhecido por *makespan*) submetidas à grade, ocasionado pelo tempo gasto para colocar e retirar os recursos da grade de tais estados. Isso porque, quando uma nova tarefa é submetida à grade, ela precisará esperar o tempo necessário para que o recurso seja colocado de volta em um estado completamente operacional. Além disso, mostra-se necessário avaliar o número de transições realizadas entre o estado ativo e os estados de baixo consumo de energia, uma vez que alguns componentes do computador, como o disco rígido, são fabricados para tolerar um número máximo de transições durante o seu tempo de vida [15]. Nesse sentido, o uso das estratégias Sobreaviso e Hibernação pode reduzir a vida útil dos recursos da grade computacional, caso elas provoquem um aumento excessivo no número de transições [15, 26] levando os componentes a atingirem o limite de transições antes do tempo estimado.

Ao se utilizar as estratégias de economia de energia, como Sobreaviso e Hibernação, é necessário decidir após quando tempo de inatividade do recurso a estratégia deverá ser utilizada. Esse tempo é denominado *tempo de inatividade* (TI). O TI é o tempo máximo em que um recurso deve permanecer ocioso aguardando a chegada de uma nova requisição, antes que seja colocado em um estado de baixo consumo de energia [4, 2]. Se o intervalo entre requisições de recursos é longo, mostra-se vantajoso transitar o recurso para um estado de baixo consumo de energia tão logo ele se torne ocioso, mas se esse intervalo for pequeno é mais vantajoso que o recurso permaneça ocioso aguardando a chegada de uma nova requisição.

Deste modo, usar um TI pequeno possibilita que o recurso seja adormecido tão logo se torne ocioso, o que pode aumentar a economia de energia. No entanto, esse valor de TI pode aumentar o tempo de reposta das aplicações, pois o recurso precisará ser reativado assim que chegar uma nova requisição. Usar um TI grande faz com que o recurso permaneça no estado de ociosidade durante

mais tempo. Esse tempo no estado ocioso pode reduzir a economia de energia, mas permite que o computador responda instantaneamente a uma nova requisição o que pode reduzir o tempo de resposta das aplicações. Geralmente, busca-se definir um TI que equilibre essas duas possibilidades [2, 4, 1]. Neste artigo avaliamos o uso de diferentes valores de TI associados às estratégias Sobreaviso e Hibernação.

O objetivo deste artigo é avaliar como as estratégias Sobreaviso, Hibernação e os valores de TI utilizados impactam na economia de energia, no tempo de resposta e no número de transições realizadas pelos recursos em uma grade computacional oportunista, identificando cenários em que essas estratégias minimizam o consumo de energia e avaliando o impacto no tempo de resposta. As duas principais contribuições deste artigo são:

- Avaliamos o uso das estratégias Sobreaviso e Hibernação em um domínio administrativo de uma grade computacional oportunista. Os resultados mostram que Sobreaviso e Hibernação reduzem o consumo de energia da infraestrutura em mais de 80% em cenários de baixa contenção de recursos. Essas estratégias têm pequeno impacto no tempo de resposta das aplicações, aumentando o mesmo em no máximo 5% com o uso da estratégia Hibernação e em não mais que 1% com o uso da estratégia de Sobreaviso. Na maior parte dos cenários avaliados, a estratégia de Sobreaviso apresentou economia de energia similar a Hibernação e menor impacto no tempo de resposta das aplicações.

- Avaliamos também o uso de diferentes valores de TI associados às estratégias Sobreaviso e Hibernação. Foram avaliados cinco valores de TI utilizados em outros trabalhos disponíveis na literatura: 0 (sem espera), 300 [12, 18, 26], 600 [20, 5], 900 [8] e 1.200 segundos (temporizador mais agressivo). Os resultados obtidos mostram que quanto maior o valor de TI menor é a economia de energia. Além disso, nos cenários onde há grande contenção pelos recursos da grade, observamos que variar o valor de TI não impacta significativamente no tempo de resposta das aplicações. No entanto, valores de TI maiores resultam em menor número de transições entre o estado ativo e os estados de economia de energia.

As demais seções deste artigo estão organizadas da seguinte forma. Na Seção 2 é apresentado o estado da arte do uso de estratégias de economia de energia em sistemas computacionais. Na Seção 3 é apresentado o modelo de simulação de eventos discretos utilizado; em seguida, na Seção 4 , é descrito o projeto dos experimentos. Os resultados da avaliação do uso das estratégias Sobreaviso e Hibernação, e de diferentes valores de TI em uma grade computacional oportunista são apresentados e analisados na Seção 5. Por fim, na Seção 6 são descritas as conclusões e os trabalhos futuros.

2. ESTADO DA ARTE

Nos últimos anos, diversas abordagens têm sido propostas para analisar e reduzir o consumo de energia em sistemas computacionais, através de melhores projetos do hardware e do software desses sistemas, além do ambiente onde os mesmos estão inseridos. Em hardware, tem-se buscado desenvolver computadores mais econômicos em termos de consumo de energia. Já em software, um caminho é o desenvolvimento de softwares mais otimizados a fim de evitar processamento excessivo [3]. Em relação ao ambiente de implantação, toda a infraestrutura de suporte é considerada, incluindo principalmente o sistema de refrigeração das máquinas.

Esta preocupação com o consumo de energia não é nova. As indústrias de dispositivos móveis e *laptops* têm desenvolvido muitas pesquisas nesse sentido. No entanto, a sua motivação é o fato da energia ser algo bastante escasso (uso de baterias recarregáveis com suporte limitado, por exemplo). Além disso, as técnicas utilizam, entre outras coisas, padrões de comportamento do usuário que nem sempre se aplicam a todos os sistemas, o que invalidaria o uso dessas técnicas em outros cenários. Mesmo com a impossibilidade de reuso destas técnicas, tem-se observado uma melhora na relação entre o consumo de energia e utilização (carga de trabalho) dos computadores ao longo dos tempos. No entanto, esta relação ainda pode ser melhorada [3].

Outro aspecto que pode ser explorado visando a diminuição do consumo de energia é a mudança do estado dos dispositivos dependendo da sua carga de uso. Alguns trabalhos já foram realizados com o objetivo de mensurar os gastos e abordar diferentes estratégias. Talebi et al. [28] reporta práticas que podem ser usadas em salas de aula e laboratórios de pesquisa. São destacadas as práticas: *(i)* não utilizar protetor de tela que possui animações gráficas; *(ii)* configurar o monitor para usar o modo *sleep*, no qual ele passa a consumir menos energia (reativação, por exemplo, com a movimentação do *mouse*); *(iii)* configurar o computador para que o disco rígido entre em modo *sleep*, que implica na redução dos movimentos do disco rígido quando o computador está ocioso; *(iv)* configurar o computador para usar a estratégia Sobreaviso, que consiste em manter os dados na memória RAM e reduzir a atividade dos outros componentes; *(v)* configurar o computador para usar a estratégia Hibernação, em que os dados são armazenados no disco rígido e os demais componentes são desligados, de modo que ao ser ligado novamente, os dados sejam recuperados do disco rígido e

o computador retorne ao estado em que estava.

Ao se utilizar estratégias como Sobreaviso e Hibernação que mudam o estado do recurso, mostra-se necessário definir um temporizador para determinar após quando tempo de inatividade (TI) a estratégia deverá ser usada [4]. Temporizadores têm sido usados na gerência de energia em sistemas operacionais para computadores pessoais [8, 12, 20, 5]. Nesses sistemas, o valor de TI varia entre 5 e 15 minutos. O Condor, *middleware* para grade computacional, utiliza um TI de 2 horas [7].

Um projeto da Escola de Educação da Universidade de Indiana [11] visa implantar um mecanismo para colocar computadores *desktops* em modo *sleep* quando não estiverem em uso, utilizando um temporizador de 2 horas e 15 minutos. Em um projeto piloto, obteve-se redução no consumo de energia em $48,3\%$ para um *cluster* de 11 computadores *desktops* e em $30,9\%$ na ala de escritórios. Essa redução equivale a uma economia de até US$ $500.000,00$ por ano para a universidade. Entretanto, o foco do trabalho é a redução do consumo utilizando o estado *sleep*, sem se preocupar com o tempo e o consumo de energia necessários para colocar e retirar os computadores desse estado. Em uma máquina que está sendo explorada de forma oportunista, onde esse tipo de operação de entrada e saída da máquina da grade pode ser frequente, não é claro que essa estratégia possa trazer economias similares.

Há também estudos que visam investigar estratégias de escalonamento ciente do consumo de energia. Essas estratégias visam minimizar o consumo de energia com o mínimo de impacto no desempenho das aplicações. Sharma e Aggarwal [27] analisam um escalonamento ciente do consumo de energia em grades de *desktops*. No entanto, eles analisam aplicações que fazem uso intensivo de memória, além disso, os *desktops* estão sempre disponíveis para a grade. De outro modo, Lammie, Brenner e Thain [18] analisam estratégias de escalonamento de cargas de trabalho de grades computacionais em *clusters multicores*. O objetivo do escalonamento também é minimizar o consumo de energia e maximizar o desempenho, no entanto, apenas o uso do processador foi considerado. São propostas três técnicas para atingir este objetivo: desativar as máquinas que se encontram subutilizadas, utilizando um temporizador de 300 segundos; prover um escalonamento inteligente das tarefas de modo a minimizar o número de máquinas ativas; e fazer um dimensionamento dinâmico da frequência, de modo a ativar e a desativar as máquinas de acordo com a demanda. Ambos os estudos indicam redução significativa no consumo de energia e pouco impacto no desempenho.

Zong et al. [30] propõem um *framework* para simulação e avaliação da eficiência energética de algoritmos de escalonamento de tarefas que fazem uso intensivo de dados. Para avaliar o *framework*, utilizou-se uma política de escalonamento que consiste em escalonar tarefas para nodos que economizam mais energia (energeticamente mais eficientes). Assim, o escalonador dá prioridade a escalonar tarefas que fazem uso intensivo de dados a recursos mais eficientes para este tipo de tarefa, podendo retirar demais tipos de tarefas desses recursos, reservarem recursos para alocar a tipos específicos de tarefas, e dá preferência por alocar em um mesmo recurso tarefas com dependências entre si, a fim de evitar consumo de energia com carga extra de comunicação. A grade utilizada considera total disponibilidade dos recursos e o escalonamento apenas de aplicações que fazem uso intensivo de dados, o que difere do ambiente e do tipo de tarefas submetidas a grades computacionais oportunistas.

O presente trabalho, diferente dos demais apresentados nesta seção, visa analisar estratégias de economia de energia no contexto de grades computacionais oportunistas. Essas grades apresentam um fator não investigado pelos demais trabalhos em grades computacionais que é aplicar essas estratégias em um cenário onde a disponibilidade das máquinas varia ao longo do tempo. A semelhança com os demais trabalhos está na utilização de estratégias de redução do consumo de energia durante os ciclos de ociosidade.

Este artigo é uma versão estendida do trabalho de Ponciano et al. [24]. Ponciano et al. avalia as estratégias Sobreaviso e Hibernação por meio de um estudo de caso realizado com um rastro da grade computacional OurGrid [6]. Os resultados mostram que ambas as estratégias possibilitam economizar energia na infraestrutura e impactam o tempo de resposta das tarefas. As estratégias são avaliadas em apenas dois cenários de contenção: alta e baixa. Ponciano et al. não avalia o uso de TI para decidir quando as estratégias devem ser utilizadas. Este trabalho avalia o uso das estratégias Sobreaviso e Hibernação em diversos cenários de contenção de recursos [24] e em conjunto com diferentes valores para TI. Além disso, este trabalho utiliza duas novas métricas: tempo de resposta das aplicações e número de transições realizadas pelas máquinas.

3. Estratégias para economia de energia em Grades Oportunistas

Nossa avaliação do uso de estratégias de economia de energia em grades computacionais oportunistas utiliza um modelo simulado para avaliar o impacto das estratégias Sobreaviso, Hibernação e diferentes valores de TI na economia de energia da infraestrutura, tempo de resposta das aplicações e número de transições realizadas pelas máquinas.

Nosso modelo de grade computacional baseia-se no *middleware* OurGrid [6]. A grade é composta por três componentes: *broker*, *workers* e *peer*. O *broker* é o com-

```
J1-1  1050208880   218
J2-1  1050208892   1107
J2-2  1050208892   723
```

(a) Demanda

```
V4-linux  1104904800   1636
V2-linux  1104904800   3611
V4-linux  1104914800   3666
```

(b) Disponibilidade

```
V2-linux  110700
V3-linux  177120
V4-linux  114810
```

(c) Processamento

Figura 1. Exemplos dos rastros utilizados nas simulações

ponente da grade que provê uma interface para que os usuários submetam aplicações para serem executadas nos recursos da grade. Outra função do *broker* é escalonar as tarefas das aplicações para os *workers* disponíveis. Os *workers* são agentes da grade que executam nos recursos. Os workers recebem e executam tarefas escalonadas pelos *brokers*. Os *workers* também monitoram e reportam o estado do recurso ao *peer*, a fim de informar quando ele está ou não disponível para executar uma tarefa. O *peer* é o componente responsável por gerenciar os recursos de um domínio administrativo e por se comunicar com outros *peers* da grade a fim de obter ou doar *workers*.

Por simplicidade, nosso modelo de simulação não considera o mecanismo de priorização utilizado pelo OurGrid para doação de recursos entre diversos *peers* quando a grade se encontra em alta contenção. Deste modo, nosso modelo de simulação foca em apenas um domínio administrativo (ou *site*) da grade, i.e., há apenas um *peer* e todas as máquinas da grade são gerenciadas por ele. Do ponto de vista de nossa avaliação, essa simplificação não tem um impacto grande nos resultados, haja vista que as decisões para aumentar a eficiência da grade são tomadas de forma autônoma por cada domínio administrativo.

O modelo de simulação é de eventos discretos guiados por rastros. O simulador recebe dois rastros como entrada: um rastro de submissão de aplicações e um rastro que descreve a variação na disponibilidade das máquinas para a grade. A Figura 1(a) apresenta um modelo do rastro de submissão de aplicações. O rastro contém uma tarefa em cada linha. As tarefas são constituídas de um identificador que indica o código da aplicação e o código da tarefa, um instante de submissão, e um valor estimado do tempo total de execução, respectivamente.

O rastro de variação na disponibilidade das máquinas para a grade descreve o oportunismo da grade, em que um recurso só é utilizado pela grade quando o usuário local não está utilizando-o. Um exemplo de rastro de disponibilidade das máquinas é apresentado na Figura 1(b). Em cada linha, tem-se uma máquina. Cada máquina é constituída, respectivamente, pelo instante de tempo em que ela se torna disponível para a grade e por quanto tempo ficou disponível. Um arquivo de configuração define o número de ciclos por segundo que cada máquina é capaz de executar, como mostrado na Figura 1(c). O processamento de uma tarefa da grade é interrompido se o recurso que está executando a tarefa for requisitado pelo usuário local. Nesse caso, o modelo simulação considera a implementação de *checkpoint*, de modo que, se uma máquina tornar-se indisponível durante a execução de determinada tarefa, todo o processamento já realizado é salvo e a tarefa é submetida novamente quando houver uma máquina disponível e ela reinicia a execução a partir do ponto em que foi interrompida. Neste trabalho o uso de *checkpoints* visa principalmente agilizar o tempo de simulação.

No momento em que uma máquina que se encontra disponível para a grade fica ociosa, i.e., não há tarefa da grade para ser executada, é inicializado um temporizador (TI) que define o tempo máximo em que a máquina permanecerá ociosa aguardando a chegada de uma nova tarefa. Durante esse tempo, se uma tarefa for submetida, a máquina pode iniciar a execução imediatamente. De outro modo, se o temporizador expirar sem que nenhuma tarefa seja submetida, a máquina transita para um estado de economia de energia, que pode ser Sobreaviso ou Hibernação. Esses estados são utilizados apenas nas máquinas que executam um agente *worker*.

Uma máquina em Sobreaviso opera em uma potência P_s e em Hibernação ela opera em uma potência P_h, esses estados podem reduzir o consumo de energia da máquina uma vez que essas potências são menores do que a potência P_i em que a máquina opera quando está ociosa. A potência P_i é menor que P_o, potência em que a máquina opera quando está executando uma aplicação. A redução da potência provida pelos estados Sobreaviso e Hibernação ocorre em razão da desativação de alguns componentes. De modo geral, P_s corresponde à potência da memória RAM, e outros dispositivos utilizados para acordar a máquina via WoL, teclado ou *mouse*. Por sua vez, a potência P_h corresponde apenas à potência dos dispositivos utilizados no WoL. Dado que a máquina permanece em um estado de economia de energia (v) durante Δt unidades de tempo a economia gerada em relação ao estado ocioso é dado por $\xi = (P_i - P_v) * \Delta t$, onde v pode ser Sobreaviso (s) ou Hibernação (h).

É necessário um tempo para que uma máquina transite completamente entre um estado de economia de energia e um estado ativo. Esse tempo é definido como a latência do estado. Durante essa latência a máquina é considerada ativa e, portanto, ela consome uma potência correspon-

dente ao estado ativo. No entanto, durante a latência, a máquina não pode ser utilizada para executar uma tarefa para grade, mas tão logo a transição tenha sido completada a máquina pode ser acordada ou uma tarefa pode ser escalonada para ser executada por ela. A latência do estado ocioso é desprezível ($L_i = 0$). Em Sobreaviso a latência (L_s) corresponde ao tempo gasto para ativar ou desativar alguns componentes, como o disco rígido e o processador. A latência do estado Hibernação (L_h) é maior que a do estado Sobreaviso, uma vez que envolve movimentar dados entre a memória RAM e o disco rígido, além de desativar os componentes.

Deste modo, o uso de Sobreaviso e Hibernação implica em um compromisso entre o benefício em termos de redução da potência provida pelo estado ($P_h < P_s < P_i$) e um custo associado em termos do aumento da latência para acordar a máquina ($L_h > L_s > L_i$). Um problema gerado pela latência é que, caso uma nova tarefa seja submetida à grade, será necessário aguardar esse tempo até que a máquina seja acordada e a tarefa possa iniciar a execução.

A cada instante, um recurso da grade pode estar em um de 4 estados possíveis (Figura 2): *(i)* `Ocioso`, caso não exista uma tarefa da grade alocada e nenhuma estratégia de economia de energia está sendo utilizada; *(ii)* `Grade`, caso esteja executando alguma tarefa da grade; *(iii)* `Usuário`, caso o usuário a esteja utilizando, neste caso o recurso não está disponível para a grade; *(iv)* ou em um estado de `economia de energia`, que pode ser uma das duas estratégias consideradas: `Sobreaviso` ou `Hibernação`. A Figura 2 apresenta um diagrama com os estados em que as máquinas da grade podem estar. Para facilitar a compreensão, esse diagrama apresenta exemplos de valores típicos do consumo de energia e do tempo gasto na transição de estados referentes aos estados Sobreaviso, Hibernação e Ocioso.

Para avaliar o uso das estratégias Sobreaviso e Hibernação em grades computacionais oportunistas são consideradas três métricas: *economia de energia*, *tempo de resposta* e *número de transições*. Para medir a economia de energia provida por Sobreaviso e Hibernação em relação ao estado Ocioso, calculamos o consumo de energia de todos os recursos durante o tempo do experimento. O consumo de energia de um recurso é dado pela Equação 1, onde se têm o tempo que o recurso permaneceu (t) e a potência em que operou (p) em cada uma das seguintes situações: (i) fazendo transição de estado (m); (ii) em estado de economia de energia (e) ou (iii) executando uma tarefa na grade (g). O estado Usuário e as transições para ele não são considerados no cálculo do custo energético da grade. A economia de energia é a diferença entre a energia consumida pelos recursos em uma configuração em que as máquinas são mantidas ociosas e a mesma configuração, mas utilizando uma estratégia de economia de energia.

$$C = t_m \times p_m + t_e \times p_e + t_g \times p_g \quad (1)$$

O número de transições realizadas por cada máquina da grade consiste na contagem do número de transições realizadas entre qualquer estado da grade e os estados de economia de energia. Esse cálculo considera, inclusive, as transições realizadas entre os estados de economia de energia e o estado Usuário.

Por fim, consideramos o impacto das estratégias no tempo de resposta das aplicações. O tempo de resposta de uma aplicação do tipo saco-de-tarefas é o tempo decorrido entre a submissão da aplicação e o tempo de término da execução da última tarefa da aplicação. Para avaliar o atraso no tempo de resposta gerado pelo uso das estratégias Sobreaviso e Hibernação em relação à configuração em que as máquinas são mantidas no estado Ocioso, utilizou-se a métrica *slowdown*.

O *slowdown* é a razão entre o tempo de resposta em uma configuração que utiliza uma estratégia de economia de energia e o tempo de resposta em uma configuração em que as máquinas são mantidas ociosas. Quando o *slowdown* é menor que 1, tem-se um ganho no tempo de resposta, de outro modo, quando o *slowdown* é maior que 1 ocorre um atraso.

De um modo geral, a dinâmica da simulação de eventos discretos funciona como segue. Quando uma nova aplicação é submetida à grade, o *peer* escolhe aleatoriamente, no conjunto de máquinas disponíveis, um subconjunto de máquinas para escalonar todas as tarefas da aplicação. Caso esse subconjunto de máquinas não seja suficiente para suprir toda a demanda da aplicação, o *peer* escolhe aleatoriamente entre as máquinas que se encontram em Sobreaviso ou Hibernação quais devem ser reativadas para atender a essa demanda. Uma vez escolhidas as máquinas, o *peer* fornece-as ao *broker* que submeteu a aplicação. Quando o *broker* recebe as máquinas, ele escalona as tarefas por ordem de criação para serem executadas nas máquinas ordenadas por ordem de chegada (FCFS, do inglês *first-come first-served*). Quando uma máquina termina de executar a tarefa ela inicia um temporizador (TI). Durante o tempo do temporizador ela aguarda a chegada de uma nova tarefa para ser executada. Caso o temporizador expire e nenhuma tarefa tenha sido submetida, a máquina transita para um estado de economia de energia. A qualquer instante da simulação uma máquina pode ser preemptada pelo usuário. A simulação termina quando o rastro de submissão de aplicações termina. Caso o rastro de variação na disponibilidade das máquinas termine antes que o rastro de submissão de aplicações, o rastro de variação na disponibilidade das máquinas retorna ao início, com os instantes de mudanças de estados atualizados. O simulador calcula o tempo de

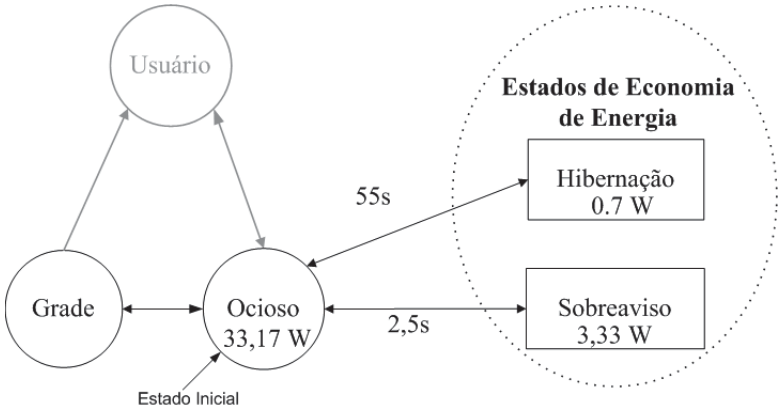


Figura 2. Estados das Máquinas

resposta de cada tarefa e de cada aplicação, o número de transições realizadas, o tempo gasto realizando transições e a energia consumida por cada máquina da grade.

4. PROJETO DOS EXPERIMENTOS

Nesta seção apresentamos o projeto dos experimentos, descrevemos os rastros, arquivos de configuração e os cenários avaliados. Para representar a variação da disponibilidade das máquinas ao longo do tempo, utilizou-se um rastro de *desktop* que apresenta dados sobre o uso das máquinas, identificando os períodos em que as máquinas estão ociosas (em todos os rastros utilizados neste trabalho, as informações referentes a tempo são medidas em segundos). O rastro utilizado foi o DEUG [17], que possui um período não contínuo de 1 mês, com informações sobre máquinas *desktop* da Universidade de París-Sud. Como as informações sobre disponibilidade nesse rastro não formam um período contínuo, foi feito um tratamento dos dados para gerar um novo rastro sem interrupções. O tratamento consistiu na replicação de períodos similares para os períodos em que não havia informações.

Do rastro DEUG também foram obtidas informações sobre o poder de processamento das máquinas. Nesse rastro, o poder de processamento é descrito pelo número de operações por segundo que a máquina pode executar. Há informações desse valor para as máquinas ao longo do tempo, e o valor varia em função da utilização da máquina no momento em que a medição foi realizada. Deste modo, utilizamos o maior desses valores para representar a capacidade de processamento de cada máquina, como apresentado na Figura 1(c).

Para estimar a frequência dos processadores, utilizamos a relação: $1,5$ GHz equivalente a 110.700 operações por segundo (como apresentado em Kondo et al. [17]). O tempo de execução das tarefas no rastro

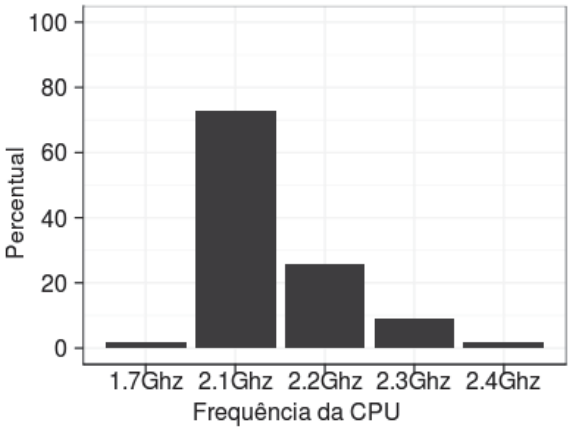


Figura 3. Histograma das frequências de CPU das máquinas utilizadas nas simulações

equivale à execução na máquina mais rápida da amostra (2.4 GHz). A variação da frequência da CPU da amostra de máquinas utilizada nas simulações é descrita pelo histograma apresentado na Figura 3.

Avaliamos valores de TI de 0, 300, 600, 900 e 1.200 segundos. Quando um tempo igual ao valor de TI passa sem que nenhuma tarefa seja submetida, a máquina inicia a transição para um estado de economia de energia. Como mencionado anteriormente, avaliamos dois estados de economia de energia: Sobreaviso e Hibernação. Estes estados são comparados com o estado ocioso, em que não é utilizada uma estratégia de economia de energia. O tempo de transição e o consumo de energia no estado de economia de energia são dados pela estratégia utilizada. A Tabela 1 apresenta os valores considerados nas simulações.

O valor de referência para o consumo de energia das máquinas quando estão executando uma tarefa foram obtidos com base na lista de máquinas avaliadas pela En-

Tabela 1. Parâmetros utilizados nas Simulações

Estratégia	Tempo de transição	Custo de energia
Ocioso	0 s	33, 17 W
Sobreaviso	2, 5 s	3, 33 W
Hibernação	55 s	0.7 W

ergyStar [10]. Procuramos os valores das máquinas que possuíam uma mesma configuração geral do sistema, mas com diferentes frequências de CPU. Os tempos de transição de estados são os mesmos utilizados por Orgerie et al. [22]. Durante o período de transição, a máquina opera em sua potência máxima [21].

Para simular a demanda de aplicações na grade, utilizamos um rastro de submissão de aplicações gerado pelo modelo de aplicações do tipo saco-de-tarefas proposto por Iosup et al. [14]. Saco-de-tarefas são aplicações formadas por tarefas intendentes, i.e., que não se comunicam entre si. Esse é o tipo de aplicação mais comum em grades computacionais oportunistas. Utilizamos aplicações submetidas ao longo de dois dias e as tarefas cujo tempo de execução não ultrapassava 35 minutos, tempo máximo de tarefas comuns em grades computacionais oportunistas [16]. Variamos o número de máquinas de 10 a 100 de modo a gerar cenários de baixa e alta contenção de recursos, ou seja, alguns cenários em que as máquinas permanecem ociosas durante muito tempo e outros em que as máquinas permanecem ociosas durante pouco tempo. Utilizamos o limite de 100 máquinas, que garante que nenhuma máquina permanece ociosa durante toda simulação. Para cada configuração foram executadas 30 simulações com diferentes rastros de demanda de modo a obter um resultado com confiança estatística de 95%.

Todos os rastros de submissão de aplicações, de variação na disponibilidade das máquinas e os arquivos de configuração utilizados nas simulações realizadas neste trabalho estão disponíveis na página: `http://redmine.lsd.ufcg.edu.br/projects/list_files/green-grid`.

5. Apresentação e Análise dos Resultados

Nesta seção apresentamos os resultados da avaliação do uso das estratégias Sobreaviso e Hibernação, com diferentes valores de TI, em grades computacionais oportunistas. Avaliamos o impacto dessas estratégias nas métricas economia de energia da infraestrutura, número de transições das máquinas e slowdown das aplicações.

A Figura 4 apresenta a economia de energia e o *slowdown* provida pelas estratégias Sobreaviso e Hibernação quando são utilizados diferentes valores de TI. As Figuras 4(a) e 4(b) mostram que os diferentes valores de TI impactam de forma semelhante na economia de energia provida por Sobreaviso e por Hibernação. Quando maior o valor de TI menor é a economia de energia. Isso é ocasionado pelo aumento do tempo em que cada máquina permanece no estado ocioso aguardando a chegada de uma nova tarefa, como mostra a Figura 5, o resultado é semelhante para a estratégia Sobreaviso. Quando a grade se encontra em alta contenção as máquinas permanecem pouco tempo executando o temporizador. Esse tempo aumenta à medida que a contenção da grade é reduzida.

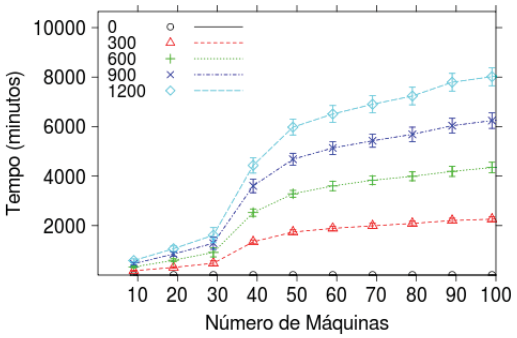
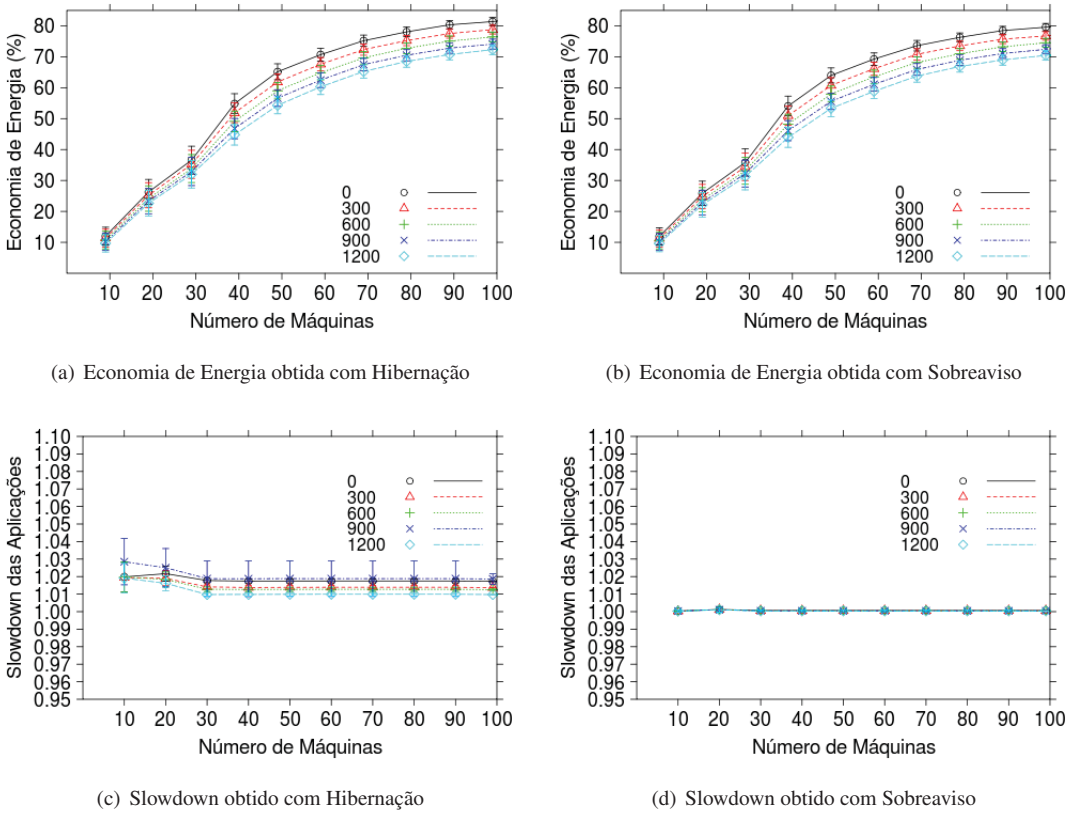


Figura 5. Total de tempo em que as máquinas permaneceram executando o temporizador com a estratégia Hibernação

As Figuras 4(c) e 4(d), mostram que as estratégias Sobreaviso, Hibernação e o uso de diferentes valores para TI não têm impacto significativo no tempo de resposta das aplicações. Isso porque o impacto que as estratégias geram no tempo de resposta é referente ao tempo gasto para acordar a máquina do estado de economia de energia, que é de 2, 5 segundos em Sobreaviso e 55 segundos em Hibernação. Esse tempo é muito pequeno comparado ao tempo de execução de aplicações do tipo saco-de-tarefas em grades computacionais oportunistas, que pode chegar a aproximadamente 35 minutos [16]. O maior *slowdown* observado foi de aproximadamente 1, 04, gerado pela estratégia Hibernação em um cenário de alta contenção (10 máquinas na grade). Esse *slowdown* equivale a um aumento de 4% no tempo de resposta das aplicações. Sobreaviso gera baixo *slowdown* independente da contenção da grade e do temporizador utilizado, o maior *slowdown* obtido foi de 1.01, que equivale a um aumento de 1% no tempo de resposta das aplicações.

A Figura 6 apresenta o número médio de transições realizadas pelas máquinas da grade para os estados de economia de energia. Esse resultado corresponde à simulação de dois dias de operação da grade computacional. O maior número de transições foi obtido na configuração com 50 máquinas utilizando a estratégia Hibernação e TI igual a 0. Nessa configuração cada máquina realizou em média 15 transições, o que equivale a aproximadamente 8



(a) Economia de Energia obtida com Hibernação

(b) Economia de Energia obtida com Sobreaviso

(c) Slowdown obtido com Hibernação

(d) Slowdown obtido com Sobreaviso

Figura 4. Slowdown e Economia de Energia obtidos com as estratégias Sobreaviso e Hibernação com valores de TI definidos como: 0, 300, 600 e 900 segundos.

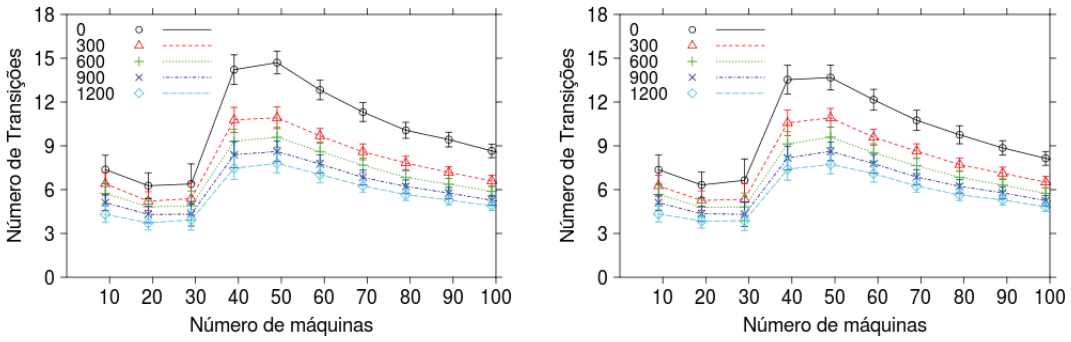
transições por dia. Esse resultado é semelhante ao obtido por Reich et al. [26], em que as máquinas realizam em média 7 transições por dia. Esse número de transições não reduz a vida útil, por exemplo, de discos rígidos SATA que podem tolerar até 500.000 transições de estado em 5 anos [15], i.e, aproximadamente 273 transições por dia.

Pela Figura 6 também se pode notar que valores de TI maiores geram menor número de transições, portanto, eles podem ser aplicados em cenários em que reduzir o número de transições for um objetivo. Pode-se observar também que quando o número de máquinas na grade é menor que 40 as máquinas realizam poucas transições, uma vez que permanecem mais tempo ocupadas executando o temporizador ou tarefas da grade. Quando o número de máquinas é entre 50 e 60 aumentam-se os períodos de ociosidade de modo que, com valor de TI igual a 0, há um aumento superior a 100% no número de transições realizadas. Por fim, quando o número de máquinas torna-se maior que 60, o número de transições tende a reduzir, uma vez que o número de máquinas na grade torna-se grande o suficiente para que nem todas as máquinas precisem ser acordadas sempre que surge uma nova demanda.

De um modo geral, as estratégias Sobreaviso e Hibernação apresentaram significante economia de energia em relação ao estado Ocioso. Não há diferença significativa na economia de energia gerada por essas estratégias. No entanto a estratégia Sobreaviso resultou em um menor impacto no tempo de resposta das aplicações quando comparado à estratégia Hibernação. Pode-se concluir, também, que o valor de TI igual a 0 é mais adequado aos contextos de contenção de recursos avaliados. Esse valor de TI aumenta pouco o número de transições, resulta em maior economia de energia e não impacta de modo significativo o tempo de resposta das aplicações.

6. Conclusões e Trabalhos Futuros

Neste trabalho avaliamos o impacto que as estratégias Sobreaviso, Hibernação e diferentes valores de TI têm na redução do custo de energia, no tempo de resposta das aplicações e no número de transições realizadas pelas máquinas em uma grade computacional oportunista sujeita a essas estratégias de economia de energia. Os resultados mostram que Sobreaviso e Hibernação reduzem



(a) Hibernação

(b) Sobreaviso

Figura 6. Número médio de transições para os estados Sobreaviso e Hibernação

o consumo de energia da infraestrutura em mais de 80% em cenários de baixa contenção de recursos. Essas estratégias têm pequeno impacto no tempo de resposta das aplicações, com um aumento de até 5% com o uso da estratégia Hibernação e um aumento menor que 1% com o uso da estratégia Sobreaviso. Na maior parte dos cenários avaliados, Sobreaviso apresentou economia de energia similar a Hibernação e menor impacto no tempo de resposta das aplicações.

Analisamos o uso de diferentes valores de TI associados às estratégias Sobreaviso e Hibernação. Foram avaliados cinco valores utilizados em outros trabalhos: 0 (sem espera), 300 [12, 18, 26], 600 [20, 5], 900 [8] e 1.200 segundos (temporizador mais agressivo). Os resultados obtidos mostram que quanto maior é o valor de TI, menor é a economia de energia. Além disso, nos cenários de contenção de recursos que foram avaliados, observamos que variar o valor de TI não impacta significativamente no tempo de resposta das aplicações. No entanto, valores de TI maiores resultam em menor número de transições entre o estado ativo e os estados de economia de energia.

Como trabalhos futuros, pode-se analisar a sensibilidade dos tempos de transição e das potências de Sobreaviso e Hibernação a fim de verificar o impacto da variação de seus valores no tempo de resposta, economia de energia e número de transições realizadas pelas máquinas em uma grade computacional oportunista. Além disso, mostra-se necessário investigar novas estratégias para escolher que subconjunto de máquinas deve ser reativado para executar as tarefas de uma aplicação quando o número de máquinas demandas é menor que o número de máquinas que se encontram em um estado de economia de energia. Alguns esforços foram dedicados nesse sentido [23], mas ainda há lacunas a serem exploradas. As estratégias de economia de energia analisadas neste artigo serão implantadas na grade computacional oportunista da Universidade Federal de Campina Grande (GridUFCG), que agregará mais de 1.000 *desktops* executando o *middleware* OurGrid [6] (http://www.ourgrid.org/).

Referências

[1] Susanne Albers. Energy-efficient algorithms. *Communications of the ACM*, 53:86–96, May 2010.

[2] John Augustine, Sandy Irani, and Chaitanya Swamy. Optimal power-down strategies. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 530–539, Washington, DC, USA, 2004. IEEE Computer Society.

[3] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40:33–37, December 2007.

[4] Luca Benini, Alessandro Bogliolo, and Giovanni De Micheli. A survey of design techniques for system-level dynamic power management. In *Readings in hardware/software co-design*, pages 231–248, Norwell, MA, USA, 2002. Kluwer Academic Publishers.

[5] Canonical Ltd. Power management in ubuntu. Disponível em `https://wiki.ubuntu.com/power-management-in-Ubuntu` Último acesso em dezembro 2010.

[6] Walfredo Cirne, Francisco Brasileiro, Nazareno Andrade, Lauro Costa, Alisson Andrade, Reynaldo Novaes, and Miranda Mowbray. Labs of the world, unite!!! *Journal of Grid Computing*, 4(3):225–246, 2006.

[7] Condor Project. Condor version 7.4.4, 2010. Disponível em: `http://www.cs.wisc.edu/condor/`. Último acesso em dezembro de 2010.

[8] Hewlett-Packard Corporation, Intel Corporation, Microsoft Corporation, Phoenix Technologies Ltd., and Toshiba Corporation. Advanced configuration and power interface specificatio, 2010. Disponível em: `http://www.acpi.info/spec.htm`. Último acesso em janeiro de 2010.

[9] The Economist. Going green. *The Economist*, Mar 2007.

[10] Energy Star. Computer-desktops & integrated computers qp list, 2009. Disponível em: `www.energystar.gov/ia/products/prod_lists/computers_prod_list.xls`. Último acesso em: setembro de 2009.

[11] Indiana University. Green computing project points to potential for energy savings, 2009. Disponível em: `http://newsinfo.iu.edu/news/page/normal/11142.html`. Último acesso em: acesso em agosto de 2009.

[12] Intel and U.S. Environmental Protection Agency. Energy star* system implementation whitepaper. Disponível em `www.intel.com/cd/channel/reseller/asmo-na/eng/339085.htm` Último acesso em dezembro 2010.

[13] Alexandru Iosup, Catalin Dumitrescu, Dick Epema, Hui Li, and Lex Wolters. 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, 2006*, pages 262–269, Washington, DC, USA, 2006. IEEE Computer Society.

[14] Alexandru Iosup, Ozan Sonmez, Shanny Anoep, and Dick Epema. The performance of bags-of-tasks in large-scale distributed systems. In *Proceedings of the 17th international symposium on High performance distributed computing*, pages 97–108, New York, NY, USA, 2008. ACM.

[15] Rini T Kaushik, Milind Bhandarkar, and Klara Nahrstedt. Evaluation and analysis of greenhdfs: A self-adaptive, energy-conserving variant of the hadoop distributed file system. In *CloudCom 2010 : Proceedings of the 2th IEEE International Conference on Cloud Computing*, pages 1–12. IEEE Computer Society, 2010.

[16] Derrick Kondo, Andrew Chien, and Henri Casanova. Scheduling task parallel applications for rapid turnaround on enterprise desktop grids. *Journal of Grid Computing*, 5:379–405, 2007.

[17] Derrick Kondo, Gilles Fedak, Franck Cappello, Andrew A. Chien, and Henri Casanova. Characterizing resource availability in enterprise desktop grids. *Future Generation Computer Systems*, 23(7):888–903, 2007.

[18] M. Lammie, P. Brenner, and D. Thain. Scheduling grid workloads on multicore clusters to minimize energy and maximize performance. In *Proceedings of the 10th IEEE/ACM International Conference on Grid Computing, 2009*, pages 145 –152, october 2009.

[19] David Meisner, Brian T. Gold, and Thomas F. Wenisch. Powernap: eliminating server idle power. In *ASPLOS '09: Proceeding of the 14th international conference on Architectural support for programming languages and operating systems*, pages 205–216, New York, NY, USA, 2009. ACM.

[20] Microsoft Corporation. Windows power management. isponível em `http://www.microsoft.com/whdc/archive/winpowmgmt.mspx` Último acesso em dezembro 2010.

[21] A. C. Orgerie, L. Lefevre, and J. P. Gelas. Chasing gaps between bursts: Towards energy efficient large scale experimental grids. In *Parallel and Distributed Computing, Applications and Technologies, 2008. PDCAT 2008. Ninth International Conference on*, 2008.

[22] Anne C. Orgerie, Laurent Lefèvre, and Jean P. Gelas. Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems. *Parallel and Distributed Systems, International Conference on*, 0:171–178, 2008.

[23] Lesandro Ponciano and Francisco Brasileiro. On the impact of energy-saving strategies in opportunistic grids. In *Energy Efficient Grids, Clouds and Clusters Workshop, proceedings of the 11th ACM-IEEE International Conference on Grid Computing (Grid 2010),*, pages 282 – 289, Bruxelas, Bélgica, 2010. ACM-IEEE.

[24] Lesandro Ponciano, Jaindson Santana, Marcus Carvalho, Matheus Gaudencio, and Francisco Brasileiro. Analise de estrategias de computacao verde em grades computacionais oportunistas. In *Anais do XXVIII Simposio Brasileiro de Redes de Computadores e Sistemas Distribuidos*, pages 307–320, Porto Alegre, Brasil, may 2010. SBC.

[25] David Przybyla and Mahmoud Pegah. Dealing with the veiled devil: eco-responsible computing strategy. In *SIGUCCS '07: Proceedings of the 35th an-*

nual ACM SIGUCCS conference on User services, pages 296–301, New York, NY, USA, 2007. ACM.

[26] Joshua Reich, Aman Kansal, Michel Gorackzo, and Jitendra Padhye. Sleepless in seattle no longer. In *USENIX ATC'10: USENIX Annual Technical Conference*, 2010.

[27] Kamal Sharma and Sanjeev Aggarwal. Energy aware scheduling on desktop grid environment with static performance prediction. In *SpringSim '09: Proceedings of the 2009 Spring Simulation Multiconference*, pages 1–8, San Diego, CA, USA, 2009. Society for Computer Simulation International.

[28] Mujtaba Talebi and Thomas Way. Methods, metrics and motivation for a green computer science program. *SIGCSE Bull.*, 41(1):362–366, 2009.

[29] Joseph Williams and Lewis Curtis. Green: The new computing coat of arms? *IT Professional*, 10(1):12–16, 2008.

[30] Ziliang Zong, Xiao Qin, Xiaojun Ruan, Kiranmai Bellam, Yiming Yang, and Adam Manzanares. A simulation framework for energy efficient data grids. In *WSC '07: Proceedings of the 39th conference on Winter simulation*, pages 1417–1423, Piscataway, NJ, USA, 2007. IEEE Press.