Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Programa de Pós-Graduação em Ciência da Computação

Avaliação do Impacto de Estratégias de Economia de Energia em Grades Computacionais Entre-Pares

Lesandro Ponciano dos Santos

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Sistemas de Computação

Francisco Vilar Brasileiro
(Orientador)

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

S237a Santos, Lesandro Ponciano dos.

Avaliação do impacto de estratégias de economia de energia em grades computacionais entre-pares / Lesandro Ponciano dos Santos. – Campina Grande, 2011.

76 f. : il.

Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

Referências.

Orientador: Prof. Ph.D. Francisco Vilar Brasileiro.

1. Sistemas Distribuídos. 2. Grades Computacionais. 3. Economia de Energia. I. Título.

CDU – 004.75 (043)

"AVALIAÇÃO DO IMPACTO DE ESTRATÉGIAS DE ECONOMIA DE ENERGIA EM GRADES COMPUTACIONAIS ENTRE-PARES"

LESANDRO PONCIANO DOS SANTOS

DISSERTAÇÃO APROVADA COM DISTINÇÃO EM 28.02.2011

FRANCISCO VILAR BRASILEIRO, Ph.D
Orientador(a)

NAZARENO FERREIRA DE ANDRADE, D.Sc
Examinador(a)

DENIO MARIZ TIMÓTEO DE SOUSA, Dr.
Examinador(a)

CAMPINA GRANDE - PB

Resumo

Grades computacionais entre-pares são infraestruturas de computação que utilizam ciclos ociosos de recursos computacionais de diferentes domínios administrativos. Geralmente, a demanda por recursos nessas grades ocorre em rajadas. Durante uma rajada de demanda, muitos recursos da grade são necessários. Porém, em outros momentos, os recursos permanecem ociosos por longos períodos. Manter os recursos ociosos quando não estão em uso nem pela grade nem pelo usuário local não é eficiente em termos de consumo de energia. Uma maneira de reduzir a energia consumida pelos recursos nesses períodos é colocá-los em um modo de dormência, em que eles consomem menos energia. Nesta dissertação, avaliamos duas estratégias de dormência: Sobreaviso e Hibernação. No contexto de grades computacionais, essas estratégias apresentam um compromisso entre o benefício da economia de energia dos recursos, de um lado, e de outro lado os custos associados em termos do aumento no tempo de resposta das aplicações e do impacto no tempo de vida dos recursos. O aumento no tempo de resposta advém do tempo necessário para acordar o recurso quando surge uma nova demanda da grade. O impacto na vida útil do recurso ocorre em razão das partidas e paradas das rotações do disco rígido quando as estratégias de dormência são utilizadas. Nossa avaliação utiliza um modelo simulado para tratar esse compromisso. Avaliamos após quando tempo de inatividade (TI) as estratégias de dormência devem ser utilizadas e como escolher quais recursos acordar quando surgir uma demanda menor que a quantidade de recursos adormecidos. Nos cenários avaliados, Sobreaviso resultou em uma economia de energia equivalente a Hibernação, mas em um menor impacto no tempo de resposta. Além disso, Sobreaviso pode ser utilizado tão logo a máquina se torne inativa, não sendo necessário aguardar um TI. Isso permite aumentar a economia de energia sem gerar atrasos consideráveis no tempo de resposta. Os resultados mostram que utilizar uma estratégia de escolha que considera a eficiência energética dos recursos permite um aumento na economia de energia. Por fim, encontramos que o uso das estratégias de dormência pela grade resulta em menos partidas e paradas dos discos rígidos do que as que ocorreriam se o usuário local, ao invés de disponibilizar sua máquina para a grade, adotasse uma estratégia que a colocasse em um estado de dormência, sempre que ela ficasse ociosa.

Abstract

Peer-to-peer opportunistic 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 times they remain idle for long periods. 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 comprise the grid in a "sleeping" mode which consumes less energy. In this work, we evaluated two sleeping strategies, namely: standby and hibernate. In grid computing, these strategies show a tradeoff between the benefit of energy saving and the associated costs in terms of increasing the job makespan and the effects in hard disks' lifetime. The overhead in the makespan due to the time taken to wake up the resource when a new task arrives. The effect in hard disks' lifetime is due to starting and stopping of the hard drives revolutions when sleeping strategies are used. In this work, we use a simulated model in order to evaluate this tradeoff. We also evaluated the minimum amount of machine idle time after which a sleeping strategy should be applied and how to choose which machine should be woken up, if several options are available. Our results show that both Standby and Hibernate strategies can present great energy savings. However, Standby presents lower impact on the job's makespan. Furthermore, we have identified that sleeping strategies can be used as soon as the machine becomes idle, i.e., it is not necessary to wait any time in idle state. Regarding the strategies to chose machines, the ones that consider the machine energy efficiency increase the energy saving. Finally, we found that the use of sleeping strategies by peer-to-peer grids result in fewer hard disk transitions than would occur if the local user, instead of donate your machine to the grid, to adopt a strategy to place it in a sleeping mode, when it was idle.

Aos meus familiares. Em especial, à memória do meu pai Antonio Carlos, que me viu partir em busca desta formação, mas que descansou antes de vê-la concluída.

Agradecimentos

O trabalho descrito nesta dissertação foi orientado pelo professor Francisco Vilar Brasileiro (vulgo “Fubica”), a quem agradeço pela oportunidade, confiança e por todo auxílio que me concedeu. Contei, também, com grande ajuda do professor Nazareno Andrade, a quem agradeço pelas boas discussões sobre planejamento, execução e apresentação de resultados à comunidade científica.

Tenho muitos amigos e gostaria de agradecer nominalmente a cada um deles. Entretanto, como isso não pode ser feito aqui, extensivo a todos, agradeço a alguns que estiveram mais próximos nos últimos dois anos: Jaindson Santana, Rafael Silva e Sabrina Souto, amigos com os quais dividi a sala Volúpia e bons momentos de descontração; Jonhnny Weslley, com quem dividi apartamento e aprendi um pouco sobre a cultura piauiense; Fabrício Góes (Escócia), Dirlene Almeida (Ipatinga-MG) e Lívia Sales (Guanhães-MG), alguns dos meus amigos que estão distantes, mas que o Gmail®, msn® e Skype® ajudaram a manter próximos. Agradeço, também, a todos os colegas do Laboratório de Sistemas Distribuídos, em especial àqueles aos quais tive mais proximidade: Adabriand Furtado (vulgo “Brian”), Alan Cruz (vulgo “Tokinho”), Ana Clara (vulgo “Aninha”), Carla Araujo (vulgo “Carlinha”), David Candeia, Edigley Fraga, Giovanni Farias (vulgo “Surubim”), Paulo Ditarso, Pryscilla Dora, Ricardo Araújo, Thiago Emmanuel (vulgo “Manel”), Tomás de Barros e Vinicius Aguiar (vulgo “Maguinho”).

O trabalho apresentado nesta dissertação recebeu contribuições de diversas pessoas, entre colegas de trabalho, avaliadores dos artigos, da proposta ou da dissertação. Extensivo a todas essas pessoas, agradeço ao professor Marco Aurélio Spohn, que apresentou valiosas sugestões quando participou da banca de defesa da proposta de dissertação e das orientações do WDCOPIN. Por fim, agradeço ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro concedido por meio do processo 133448/2009-6, edital MCT/CNPq nº 70/2008.

Conteúdo

Lista de Símbolos

ACPI: Configuração Avançada e Interface de Energia (do inglês *Advanced configuration and power interface*).

APM: Gerência Avançada da Energia (do inglês *Advanced Power Management*).

BIOS: Sistema Básico de Entrada e Saída (do inglês *Basic Input/Output System*).

BOINC: *Middleware* que dá suporte ao desenvolvimento de grades de computação voluntária (do inglês *Berkeley Open Infrastructure for Network Computing*).

BoT: Saco-de-tarefas (do inglês *Bag-of-Tasks*).

CPU: Unidade Central de Processamento (do inglês *Central Processing Unit*).

CO_2: Dióxido de Carbono.

DFS: Ajuste dinâmico da frequência da CPU (do inglês *Dynamic Frequency Scaling*).

DVFS: Ajuste dinâmico da frequência e da voltagem da CPU (do inglês *Dynamic Voltage and Frequency Scaling*).

DVS: Ajuste dinâmico da voltagem da CPU (do inglês *Dynamic Voltage Scaling*).

EA: Ciente do consumo de energia (do inglês *Energy Aware*).

FCFS: O primeiro a chegar é o primeiro a ser servido (do inglês *First Come First Served*).

FDA: Função de distribuição acumulada (em inglês *Cumulative Distribution Function*(CDF)).

GHz: Gigahertz.

GPU: Unidade de Processamento Gráfico (do inglês *Graphics Processing Unit*).

HD: Disco rígido (do inglês *Hard Disk*).

kWh: Quilowatt-hora.

LRS: Menos recentemente adormecido (do inglês *Least Recently Sleeping*).

MRS: Mais recentemente adormecido (do inglês *Most Recently Sleeping*).

MW: Mega-watts

NoF: Rede de Favores (do inglês *Network of Favors*) é um mecanismo de incentivo ao compartilhamento de recursos entre domínios administrativos utilizado na grade computacional entre-pares OurGrid.

P2P: Entre-pares (do inglês *Peer-to-Peer*).

RAM: Memória de Acesso Aleatório (do inglês *Random Access Memory*).

SATA: Acrônimo de *Serial Advanced Technology Attachment*.

TI: Tempo de Inatividade, tempo que um computador permanece ocioso antes de ser adormecido.

TDP: Maior potência em que um processador pode operar (do inglês *Thermal Design Power*)

USB: Barramento Serial Universal (*do inglês Universal Serial Bus*).

W: *Watts*, unidade de potência.

WoL: Aguarda um comando via interface Ethernet (*do inglês Wake-on-LAN*)

P Potência de operação de um sistema, dada em Watts.

t Instante de tempo que marca o início de uma sessão de disponibilidade de uma máquina para a grade.

t' Instante de tempo que marca o final de uma sessão de disponibilidade de uma máquina para a grade.

E Energia elétrica consumida por um sistema.

L Latência, tempo de transição de/para um estado de dormência – tempo gasto para a máquina dormir ou acordar.

f Frequência da CPU.

d Estado de dormência, no escopo desta dissertação pode designar o Sobreaviso (s) ou o estado Hibernação (h).

h Estado de dormência Hibernação (*Hibernate*).

s Estado de dormência Sobreaviso (*Sobreaviso*).

o Estado ocioso, neste estado a máquina se encontra inativa.

a Estado ativo, a máquina se encontra executando alguma tarefa.

L_a Latência do estado ativo a.

L_o	Latência do estado ocioso o.
L_d	Latência do estado de dormência d.
L_s	Latência do estado de dormência Sobreaviso (s).
L_h	Latência do estado de dormência Hibernação (h).
P_a	Potência do estado ativo a.
P_o	Potência do estado ocioso o.
P_d	Latência do estado de dormência d.
P_s	Latência do estado de dormência Sobreaviso (s).
P_h	Latência do estado de dormência Hibernação (h).
c_d	Energia elétrica consumida por uma máquina que faz uso de um estado de dormência d e realiza uma transição de entrada e uma transição de saída do estado.
c_o	Energia elétrica consumida por uma máquina mantida ociosa nos períodos de ociosidade.
Δt	Quantidade de tempo em que uma máquina permanece em um estado, o estado é definido pelo contexto.
Δt_d	Tamanho de uma sessão de dormência, i.e., o intervalo de tempo em que uma máquina permaneceu adormecida.
$\Delta t_{d,o}$	Tamanho de uma sessão de dormência para a qual utilizar uma estratégia de dormência ou manter a máquina ociosa apresenta o mesmo consumo de energia elétrica.
$\Delta t_{s,o}$	Tamanho de uma sessão de dormência para a qual utilizar a estratégia de dormência Sobreaviso ou manter a máquina ociosa apresenta o mesmo consumo de energia elétrica.
$\Delta t_{h,o}$	Tamanho de uma sessão de dormência na qual utilizar a estratégia de dormência Hibernação ou manter a máquina ociosa apresenta o mesmo consumo de energia elétrica.
$\Delta t_{s,h}$	Tamanho de uma sessão de dormência para a qual as estratégias Sobreaviso e Hibernação apresentam o mesmo consumo de energia elétrica.
A	Uma configuração da grade composta por uma estratégia de dormência, uma estratégia de escolha de recursos e uma valor de TI.

$\overline{A}$	Um configuração da grade que usa os mesmos valores dos parâmetros de uma configuração A, mas não faz uso de um estado de dormência, i.e., as máquinas são mantidas ociosas.
ξ^A	Percentual de energia elétrica economizada por uma configuração A em relação a uma configuração de referência $\overline{A}$.
ξ_d	Energia elétrica economizada por uma máquina durante uma sessão de dormência em relação a ser mantida ociosa durante igual período.
M_d	Conjunto de máquinas que estão em um estado de dormência em um dado instante.
n	Número de tarefas de uma aplicação do tipo saco-de-tarefas.
m^X	Tempo de resposta (ou *makespan*) de uma aplicação quando utilizada uma configuração X.
E^X	Energia elétrica consumida pela grade quando utilizada a configuração X.
β^A	Percentual de atraso no tempo de resposta gerado por uma configuração A em relação a uma configuração $\overline{A}$.

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Neste capítulo, apresentamos o contexto e a motivação da pesquisa descrita nesta dissertação. Primeiro, contextualizamos os principais aspectos de eficiência energética em sistemas computacionais. Em seguida, apresentamos as grades computacionais entre-pares e a motivação para economizar energia nessas grades. Dado esse contexto, apresentamos os problemas de eficiência energética que identificamos nessas grades e os requisitos para solução desses problemas. Apresentamos uma visão geral das soluções existentes e em que elas satisfazem ou não esses requisitos. A solução desenvolvida nesta dissertação é apresentada e analisada em linhas gerais. Por fim, o capítulo é encerrado com uma descrição da organização dos demais capítulos que compõem esta dissertação.

1.1 Contexto

Nos últimos anos tem crescido a preocupação com o consumo de energia elétrica dos sistemas computacionais de um modo geral. Essa preocupação é motivada pelo impacto negativo que o aumento no consumo de energia tem gerado no meio ambiente e no custo financeiro para operação desses sistemas. O impacto no meio ambiente está relacionado à degradação de fontes não renováveis de geração de energia e à emissão de dióxido de carbono (CO_2) no meio ambiente [57]. Do mesmo modo, o alto consumo de energia tem aumentado significativamente o custo financeiro da computação realizada nesses sistemas [58; 47; 63].

Uma das razões desse aumento no consumo de energia é que, historicamente, esses siste-

mas têm sido desenvolvidos com o objetivo de obter "mais poder computacional a qualquer custo" [63; 16]. Norteado por esse objetivo, foram desenvolvidos sistemas com maior poder de processamento e capacidade de armazenamento, mas também com maior consumo de energia. Por exemplo, em 2007, cada um dos 10 computadores da lista *Top* 500 (lista dos 500 supercomputadores de maior poder de processamento no mundo) [43] consumia, em momento de pico, por volta de 10 MWh (megawatt-hora), o suficiente para abastecer uma cidade de 40.000 habitantes [16].

Deste modo, diversas abordagens têm sido propostas com o objetivo de analisar e reduzir o consumo de energia em sistemas computacionais. Algumas dessas abordagens focam em melhores projetos do hardware e do software desses sistemas, a fim de torná-los mais econômicos em termos do uso da energia elétrica [10]. No caso de centros de dados, consideram-se também as características do ambiente onde os mesmos estão inseridos. Em computadores de mesa um objetivo tem sido economizar energia quando eles se encontram ociosos. Essas diferentes abordagens convergem em um objetivo comum que é aumentar a *eficiência energética* dos sistemas computacionais.

Eficiência energética de um sistema é a medida da relação entre o desempenho na produção de bens e serviços e o consumo de energia necessário para obter esse desempenho [28; 49; 45; 24]. Por exemplo, um computador que consome 200 kWh (quilowatt-hora) para executar um conjunto de tarefas é mais eficiente no aspecto energético que outro computador que, para executar esse mesmo conjunto de tarefas, consome 300 kWh. Deste modo, em sistemas computacionais, manter recursos consumindo energia quando ociosos ou utilizá-los sem considerar sua característica energética são procedimentos que podem reduzir a eficiência energética do sistema.

1.2 Escopo

Esta dissertação trata da eficiência energética no contexto das grades computacionais. Uma grade computacional é uma federação de recursos pertencentes a domínios administrativos diferentes. Essa federação pode ser usada para prover serviços como execução de aplicações paralelas ou armazenamento de grandes quantidades de dados. Geralmente, os recursos compartilhados em grades computacionais são heterogêneos. Eles podem diferir em diversos

aspectos, como: arquitetura, softwares instalados e eficiência energética. Uma grade computacional oportunista é um tipo de grade computacional que utiliza o poder computacional de recursos durante períodos de ociosidade.

As grades computacionais oportunistas diferem em diversos modos. Existem, por exemplo, as grades de computação voluntária que são baseadas no *middleware* BOINC [6], grades que agregam recursos pelo *middleware* Condor [23] e as grades entre-pares baseadas em mecanismos de incentivo ao compartilhamento como o *middleware* OurGrid [17]. Nesta dissertação focamos em grades oportunistas em que a gerência da energia dos recursos possa ser realizada de forma autônoma em cada domínio administrativo da grade. Isso é diferente das grades de computação voluntária que utilizam o *middleware* BOINC (*Berkeley Open Infrastructure for Network Computing*). No *middleware* BOINC, cada domínio administrativo pode ser visto como um computador doador. Esses computadores doadores se conectam a um servidor central, que representa outro domínio administrativo, que é responsável por fornecer uma carga de trabalho a ser executada e receber os resultados.

Focamos em grades oportunistas que utilizam uma organização local em cada domínio administrativo como realizado pelos sistemas de *middleware* Condor e OurGrid. Nesses sistemas de *middleware*, geralmente, cada domínio administrativo agrega diversos recursos e há um componente do *middleware* responsável por gerenciar os recursos do domínio administrativo identificando, por exemplo, quando os mesmos se encontram ociosos ou executando alguma tarefa. A gerência da energia dos recursos que compõem o domínio administrativo também pode ser realizada de modo autônomo por cada domínio administrativo da grade.

No que se refere ao mecanismo de compartilhamento de recursos entre domínios administrativos, focamos no modelo entre-pares utilizado pelo *middleware* OurGrid. Esse *middleware* utiliza um mecanismos que visa incentivar a doação de recursos entre domínios administrativos diferentes. Cada domínio administrativo tem motivação para utilizar estados de economia de energia quando os recursos estão ociosos, por exemplo. Além disso, eles também têm motivação para retirar os recursos desses estados para atender a requisições de outros domínios administrativos em troca de, no futuro, serem recompensados por esse compartilhamento [7].

1.3 Definição do Problema

Esta dissertação tem por objetivo analisar o consumo de energia na infraestrutura das grades computacionais entre-pares e propor estratégias para reduzir este consumo. Nessas grades, o usuário local desabilita o sistema de gerência de energia do recurso para que o mesmo não entre em um estado de baixo consumo de energia e, assim, torne-se acessível à grade computacional. No contexto dessa dissertação, o período de compartilhamento de um recurso na grade é definido como *sessão de disponibilidade*. Uma sessão de disponibilidade do recurso para a grade é iniciada quando o usuário local deixa de utilizar o recurso e o mesmo torna-se ocioso. A sessão é finalizada tão logo o usuário local volte a utilizar o recurso, por exemplo, quando ele aciona o *mouse*, teclado, ou outro dispositivo de entrada de dados.

Deste modo, a cada instante, a *oferta de recursos* na grade é dada pelos recursos que estão em uma sessão de disponibilidade. Os *usuários da grade* podem utilizar os recursos ofertados para executar suas aplicações. Geralmente, as grades computacionais entre-pares são utilizadas para executar aplicações científicas do tipo *saco-de-tarefas* (BoT, do inglês *Bag-of-Taks*). Saco-de-tarefas são aplicações paralelas que podem ser facilmente decompostas em várias tarefas independentes, que não se comunicam entre si. A submissão de aplicações para serem executadas na grade caracteriza a *demanda de recursos* na grade. Tipicamente, essa demanda por recursos oscila em períodos de alta e baixa de acordo com as características das aplicações e o número de usuários na grade [32]. As variações na oferta e na demanda por recursos fazem com que a grade experimente diferentes características de *contenção de recursos*.

Há alta contenção de recursos quando a demanda de recursos é maior que a oferta. De outro modo, há baixa contenção quando a oferta é maior que a demanda. Nos períodos de baixa contenção há excedente de recursos e, portanto, existem recursos ociosos na grade. Os computadores de mesa, geralmente utilizados em grades computacionais entre-pares, permanecem no estado ocioso quando não estão em pleno uso. Nesse estado, esses computadores possuem baixa latência, i.e., são capazes de responder instantaneamente a uma nova instrução, mas apresentam significativo consumo de energia, geralmente, entre 49% e 78% em relação ao estado ativo [12; 20; 62].

Existem diversos estudos sobre estratégias que visam reduzir o consumo de energia de computadores quando eles se encontram ociosos [46; 13; 19; 59; 3]. Em sistemas opera-

cionais para computadores pessoais [46; 13] e em alguns sistemas de gerência de energia em redes de computadores [19; 59; 3], é comum o uso de estratégias de dormência para reduzir esse consumo de energia do computador quando ele se encontra ocioso. Essas estratégias transitam o computador do estado ocioso para um estado de dormência; Sobreaviso (Standby) e Hibernação (Hibernate) são exemplos de estratégias de dormência.

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* [19]) – 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 por meio 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 entre-pares requer uma avaliação tanto da economia de energia que podem possibilitar, 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 reali-

zadas entre o estado ativo e os estados de dormência, dado que, em cada transição, as estratégias Sobreaviso e Hibernação realizam ativação ou desativação de alguns componentes do computador, entre eles o disco rígido. Os discos rígidos que usam tecnologia SATA possuem partes mecânicas, de modo que o excesso de transições tende a reduzir a sua vida útil [66; 11; 44].

Ao se utilizar estratégias de dormência é necessário decidir um tempo máximo em que um recurso deve permanecer ocioso aguardando a chegada de uma nova requisição antes que ele seja colocado em um estado de dormência [11; 8]. Esse tempo é denominado *tempo de inatividade* (TI). Usar um valor de TI pequeno possibilita que o recurso seja adormecido tão logo se torne ocioso, o que pode aumentar a economia de energia. Contudo, 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. A permanência nesse estado impede que a máquina transite para um estado de dormência e economize energia, mas evita um impacto no tempo de resposta das aplicações, pois permite que o computador responda instantaneamente a uma nova requisição. Geralmente, busca-se definir um valor de TI que equilibre essas duas possibilidades [8; 11; 4].

Outra decisão associada ao uso de estratégia de dormência é qual *critério utilizar para escolher quais recursos devem ser acordados* quando chegar uma nova requisição e nem todos os recursos precisarem ser acordados. Esse critério pode considerar, ou não, informações sobre os recursos disponíveis na grade e/ou sobre a carga de trabalho. Não considerar essas informações pode levar à escolha de máquinas que têm um menor pode de processamento e maior consumo de energia, enquanto máquinas com maior poder de processamento e menor consumo de energia permanecem adormecidas.

Em resumo, as estratégias de dormência, o valor de TI e as estratégias de escolha de recursos podem apresentar diferentes impactos na economia de energia, no tempo de resposta e no tempo de vida dos recursos. Não identificamos na literatura estudos sobre o impacto dessas estratégias em grades computacionais entre-pares. Investigaremos esse impacto nesta dissertação a fim de avaliar o uso dessas estratégias como solução para economizar energia em grades computacionais entre-pares. Nesse sentido, destacamos dois critérios para que

uma solução seja considerada bem sucedida. O primeiro critério é que a solução deve apresentar ao domínio administrativo, i.e., a economia de energia gerada na infraestrutura deve compensar o impacto causado no tempo de resposta das aplicações. O segundo critério é que a solução não deve levar os recursos a realizarem mais transições do que o estimado para o seu tempo de vida

1.4 Resultados e Contribuições

Nesta dissertação avaliamos o uso de estratégias de economia em grades computacionais entre-pares. Avaliamos estratégias que podem ser implementadas de forma autônoma em cada domínio administrativo (ou *site*) da grade. As estratégias avaliadas podem ser divididas em três categorias: *estado de dormência*, que visa reduzir o consumo de energia de máquinas ociosas; *escolha de recursos*, que define como escolher recursos para executar aplicações de modo a reduzir o consumo de energia da grade; *tempo de inatividade* (TI), que define 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 dormência. Utilizamos um modelo simulado de uma grade computacional entre-pares para avaliar o impacto dessas estratégias na economia de energia na infraestrutura, no atraso gerado no tempo de resposta das aplicações e no número de transições realizadas pelas máquinas. A seguir destacamos os principais resultados encontrados.

Estado de dormência. Foi avaliado o uso de dois possíveis estados: Sobreaviso e Hibernação. Os resultados mostram que ambos os estados permitem economizar energia na infraestrutura, comparado com manter as máquinas no estado ocioso. Esses estados apresentam uma economia de até $65,53\%$ em cenários de baixa contenção por recursos. Esse benefício está associado a um atraso no tempo de resposta da aplicação de até $3,8\%$ com o uso do estado Hibernação e de até $2,0\%$ com o uso do estado Sobreaviso. Quanto ao número de transições realizadas pelas máquinas, identificamos que Sobreaviso e Hibernação geram em média 4 transições diárias quando a grade se encontra em média contenção. Esse número de transições é menor do que as transições que seriam realizadas caso o usuário local utilizasse uma estratégia de dormência nos períodos em que a máquina é disponibilizada para a grade. Portanto, para esse usuário, disponibilizar a máquina para uma grade computacional

entre-pares que faz uso de estratégias de dormência não aumenta a depreciação do recurso sob o ponto de vista das transições realizadas pelo disco rígido

Tempo de Inatividade. Variamos TI com valores entre 0 e 1.800 segundos. Esse intervalo permite avaliar valores de TI que são utilizados em outros trabalhos [29; 42; 59; 46; 13; 19] e ainda explorar novos valores. Observamos que quanto maior o valor de TI menor é a economia de energia na infraestrutura e o número de transições realizadas pelas máquinas. Combinar TI definido como 0 com o estado de dormência Hibernação gera um atraso no tempo de resposta de até $3,8\%$, quando a grade se encontra em média ou baixa contenção. De outro modo, todos os valores de TI avaliados apresentaram atraso similar no tempo de resposta com o uso do estado Sobreaviso. Esse atraso é de no máximo 2%. Observamos que diferentemente de outros sistemas computacionais, nas grades computacionais entre-pares não é necessário o uso de TI, principalmente, quando se utiliza Sobreaviso como estado de dormência.

Escolha de recursos. Avaliamos quatro estratégias para definir os recursos a serem acordados quando a demanda de tarefas é menor que a oferta de recursos: primeiro os recursos mais eficiente no aspecto energético (EA, do inglês *Energy-Aware*), primeiro os recursos mais recentemente adormecido (MRS, do inglês *Most Recently Sleeping*), primeiro os recursos menos recentemente adormecido (LRS, do inglês *Least Recently Sleeping*) e escolha dos recursos de forma aleatória. Encontramos que apenas a estratégia EA apresenta melhor desempenho em termos de economia de energia. Nos cenários de baixa contenção, a economia provida por EA é de até 3% maior que a economia provida pelas demais estratégias. Todas as estratégias geraram baixo impacto no tempo de resposta das aplicações e número de transições.

A principal contribuição desta dissertação é a proposta e a avaliação do uso dessas estratégias de economia de energia no âmbito das grades computacionais entre-pares.

1.5 Estrutura da Dissertação

O conteúdo seguinte desta dissertação consta de outros seis capítulos, sendo eles:

Capítulo 2 – Fundamentação Teórica. Nesse capítulo são apresentados os principais conceitos relativos à eficiência, consumo e economia de energia em sistemas computacio-

nais.

Capítulo 3 – Trabalhos Relacionados. Neste capítulo apresentamos os trabalhos relacionados à eficiência energética, que foram desenvolvidos no contexto de sistemas distribuídos ou em sistemas centralizados, mas que apresentam relação com o uso de estados de dormência, TI ou estratégias de escolha de recursos.

Capítulo 4 – Economia de Energia em Grades Computacionais Entre-Pares. Neste capítulo definimos as estratégias de economia de energia que avaliaremos no âmbito das grades computacionais entre-pares. Analisamos diversos compromissos existentes no uso de tais estratégias nessas grades.

Capítulo 5 – Materiais e Métodos de Avaliação. Nesse capítulo apresentamos todos os materiais utilizados na avaliação das estratégias e o método utilizado para realizar essa avaliação. Nesse sentido, neste capítulo são apresentados: as métricas de avaliação; o modelo de simulação, os rastros e os arquivos de configuração que foram utilizados nas simulações, e os cenários avaliados.

Capítulo 6 – Apresentação e Análise dos Resultados. Os resultados obtidos na avaliação são apresentados e analisados nesse capítulo. Mostramos o impacto das estratégias nas métricas propostas e uma análise de sensibilidade que visa estabelecer o impacto que a variação dos valores da potência e da latência dos estados de dormência Sobreaviso e Hibernação geram na economia de energia e no atraso no tempo de resposta das aplicações.

Capítulo 7 – Conclusão e Trabalhos Futuros. Nesse capítulo são apresentadas as conclusões, as contribuições desta dissertação para a área de eficiência energética e grades computacionais, as limitações dos resultados e os trabalhos que ainda se encontram em aberto.

Capítulo 2

Fundamentação Teórica

Neste capítulo, apresentamos os conceitos básicos relativos à eficiência, economia e consumo de energia em sistemas computacionais. Inicialmente, destacamos os conceitos gerais de energia, trabalho e potência. Em seguida, apresentamos os estados de uma máquina em termos da latência e da potência: ativo, ocioso e dormindo. O capítulo é encerrado com a análise dos estados de dormência Sobreaviso e Hibernação.

2.1 Principais Conceitos

De um modo geral, *energia* (E) é a quantidade total de *trabalho* executado por um sistema em um determinado período de tempo. *Potência* (P) é a taxa na qual o sistema realiza trabalho. De acordo com o Sistema Internacional de Unidades, $Watt$ (W) é a unidade básica de potência. A fonte de alimentação delimita a maior potência em que um computador pode operar. Dado que um sistema opera à potência P durante um intervalo de Δt unidades de tempo, a *energia consumida* pelo sistema nesse intervalo é dada pela equação 2.1.

$$E = P \times \Delta t \tag{2.1}$$

Essa equação mostra que a energia que um sistema consome varia com o tempo em que ele permanece no estado e a potência de operação desse estado. Consideramos o *quilowatts-hora* (kWh) como unidade de energia consumida. Por definição, um quilowatt-hora corresponde à quantidade de energia utilizada para alimentar uma carga com potência de 1 quilowatt (kW) pelo período de uma hora.

Em eficiência energética existem algumas estratégias que visam *reduzir o consumo de energia* e outras que visam *reduzir a potência*. Note que, reduzir a potência não implica necessariamente em reduzir a energia consumida, uma vez que reduzir a potência pode aumentar o tempo que o sistema precisará permanecer no estado [65; 11]. Por exemplo, reduzir a potência do processador implica em reduzir também a frequência em que ele opera. Nesse caso, reduzir a potência quando ele se encontra executando uma tarefa não necessariamente trará economia de energia, pois o processador pode demorar mais tempo para executar a tarefa. Nesta dissertação tratamos de estratégias que permitem economizar energia reduzindo a potência quando o recurso se encontra ocioso. Essas estratégias são conhecidas como estratégias de dormência.

2.2 Estratégias de Dormência

A primeira iniciativa no sentido de estimular o uso de mecanismos de redução do consumo de energia elétrica em computadores ociosos surgiu em 1992 com o lançamento do programa voluntário Energy Star pela Agência de Proteção Ambiental americana (EPA) [64]. Este programa especifica estados de dormência para computadores, monitores, impressoras e diversos outros equipamentos, estabelecendo um limite máximo de consumo para cada estado. Os equipamentos que atendem às especificações do programa podem exibir um selo, que permite ao consumidor identificar os produtos que fazem uso mais eficiente da energia.

Assim, os fabricantes dos mais diversos componentes do computador passaram a prover diferentes funcionalidades e interfaces para implementar estratégias de dormência que visam reduzir o consumo de energia quando o computador se encontra ocioso. Como uma iniciativa à padronização dessas funcionalidades e interfaces, em 1992 foi desenvolvido pela Intel e Microsoft o primeiro padrão de gerência avançada da energia (APM do inglês *Advanced Power Management*) [31]. Essa especificação define uma interface entre a parte do *firmware* encarregada da gerência de energia no nível físico e um *driver* de políticas de consumo de energia do sistema operacional. Essa interface é independente do hardware, permitindo que os detalhes do nível físico sejam abstraídos e o sistema operacional possa fazer a gerência de energia do dispositivo sem conhecer detalhes do hardware.

Em seguida o padrão APM evoluiu para o padrão de Configuração Avançada e Interface

de Energia (ACPI, do inglês *Advanced Configuration and Power Interface* [19]). Do mesmo modo que APM, ACPI descreve as interfaces entre elementos de hardware e software que permitem a execução da gerência de energia. De um modo geral, a principal diferença entre os dois padrões está no nível de detalhe das especificações. A especificação do padrão ACPI inclui uma série de recomendações para a construção de hardware compatível, o que com o padrão APM ficava completamente a cargo de cada fabricante. Uma das especificações definidas no padrão ACPI é a definição dos estados em que o computador pode operar, como: ativo, ocioso, e adormecido.

Quando um computador se encontra *ativo*, sua potência é dada pela soma das potências dos componentes utilizados para execução da atividade em processo. Geralmente, os principais componentes são: processador, dispositivos de memória, disco rígido, placa de rede, placa de vídeo, placa mãe e barramentos. Com esses componentes ativos, um computador é capaz de responder a requisições, com uma latência desprezível. Esse estado inclui o processamento, busca de dados e armazenados em memória ou cache e o tempo ocioso enquanto aguarda mais entradas do usuário antes de entrar em um estado de ociosidade.

Um computador *ocioso* é semelhante a um computador ativo. Em ambos os estados o sistema operacional e outros softwares estão carregados na memória e existe um perfil de usuário criado. As principais diferenças são que determinados barramentos têm o fornecimento de energia interrompido, a frequência da CPU é reduzida para a menor frequência possível (em processadores que dão suporte ao Ajuste Dinâmico da Frequência e Voltagem (DVFS, do inglês *Dynamic Voltage and Frequency Scaling*), como a tecnologia PowerNow da AMD [1] e a tecnologia SpeedStep [30] da Intel, e a atividade em processo esta limitada às aplicações básicas que o sistema inicia automaticamente. Com isso, uma máquina no estado ocioso apresenta uma redução da potência em relação à potência que apresenta quando se encontra executando uma tarefa, mas ele ainda é capaz de responder com latência desprezível uma instrução recebida via rede ou gerada pelo usuário.

Uma máquina no estado *adormecido* apresenta menor potência em relação ao estado ocioso. Entretanto, esse estado exige maior latência para responder a uma requisição, uma vez que a máquina precisa ser "acordada". Neste trabalho, destacaremos dois estados de dormência definidos no padrão ACPI: Sobreaviso e Hibernação. Daremos ênfase a tais estados, visto que eles permitem que a máquina seja reativada eletronicamente por meio de um pa-

cote específico enviado à interface Ethernet [19], além de impulsos via teclado ou *mouse* realizados por um usuário local.

O mecanismo que permite que a máquina seja acordada por meio eletrônico é conhecido como *Wake-on-LAN* (WoL). Esse mecanismo foi desenvolvido pela AMD e o pacote que permite que o computador seja acordado foi apelidado de "pacote mágico" (*Magic Packet*) [2]. O mesmo consiste em um pacote *broadcast* contendo em algum lugar dentro de sua carga útil seis repetições do número 255 (em hexadecimal: FF FF FF FF FF FF), seguido pelo endereço MAC da máquina de destino repetido 16 vezes [2].

O estado Sobreaviso, estado S3 do padrão ACPI, também conhecido por Suspensão para a RAM, consiste em manter energizados a memória RAM e os componentes que possibilitam o WoL e interromper o fornecimento de energia dos demais componentes do computador. A potência em que a máquina opera quando está no estado Sobreaviso (P_s) é equivalente ao somatório das potências dos componentes que são mantidos energizados. Para que uma máquina seja colocada ou retirada completamente do estado Sobreaviso é necessário esperar uma latência (L_s), geralmente menor ou igual a 5 segundos [29; 21]). Essa latência representa o tempo de transição requerido pela estratégia.

O estado Hibernação, estado S4 do padrão ACPI, também conhecido por Suspensão para o Disco, consiste em salvar o estado da memória RAM no disco rígido e interromper o fornecimento de energia do disco rígido e dos demais componentes, exceto os componentes que implementam o WoL. A potência em que o computador opera no estado de Hibernação (P_h) equivale apenas à soma das potências dos componentes que implementam o WoL. Uma máquina em Hibernação apresenta menor potência que em Sobreaviso ($P_h < P_s$), uma vez que a memória RAM não é mantida energizada. Para que um computador seja acordado do estado de Hibernação é necessário que ele seja religado e o sistema operacional recarregado. O computador pode ser religado devido ao recebimento do pacote mágico ou manualmente. Uma vez acordado, o estado da memória RAM precisa ser restaurado do disco rígido. Em razão disso, Hibernação também apresenta uma latência (L_h) e ela é maior que a latência associada ao estado Sobreaviso ($L_h > L_s$).

Um fator comum entre Sobreaviso e Hibernação refere-se ao ambiente do usuário, constituído pelo estado da memória RAM e pelas conexões de rede que se encontram ativas. Ambas as estratégias permitem salvar o estado da memória RAM quando o computador é

adormecido, mas interrompem as conexões de rede. Deste modo, o uso dessas estratégias não é adequado em sistemas em que o computador necessita estar sempre conectado. Outro fator comum entre Sobreaviso e Hibernação refere-se à influência no funcionamento do disco rígido. Dado que as duas estratégias interrompem as rotações do disco rígido a fim de economizar energia.

Normalmente, os discos rígidos operam em rotações com velocidade constante. Os discos rígidos que utilizam a tecnologia SATA (acrônimo de *Serial Advanced Technology Attachment*) geralmente operam a uma velocidade entre 3.000 e 12.000 rotações por minuto [60]. Uma máquina que faz uso das estratégias Sobreaviso e Hibernação reduz as rotações (*spinning down*) com o objetivo de reduzir a potência de operação do disco e, por meio disso, economizar energia [44]. No entanto, o uso em excesso do processo de aceleração e desaceleração tende a diminuir o tempo de vida útil do disco, uma vez que envolve partes mecânicas [11; 44; 67; 66]. Geralmente os fabricantes informam o número de partidas/paradas estimado para o tempo de vida útil do disco rígido. Por exemplo, os discos rígidos fabricados pela empresa Seagate Technology LLC são fabricadas para suportar até 50.000 partidas e paradas durante um tempo de vida útil de 5 anos [60], o equivalente a aproximadamente 27 por dia.

Capítulo 3

Trabalhos Relacionados

Identificamos diversos trabalhos que tratam de problemas de eficiência energética em sistemas paralelos e/ou distribuídos. Porém, não identificamos trabalhos aplicados a grades computacionais entre-pares. Os trabalhos apresentados neste capítulo foram encontrados tanto por meio de sites de busca como o Google Acadêmico e bibliotecas digitais da ACM e IEEE, quanto percorrendo as referências dos principais trabalhos que eram encontrados e também pela análise de artigos produzidos nas principais conferências da área de eficiência energética em sistemas computacionais, de um modo geral, e conferências mais direcionadas à eficiência energética em sistemas distribuídos.

Agrupamos esses trabalhos em dois grupos. O primeiro grupo trata da *gerência da energia dos recursos* [62; 47; 51; 11; 20; 3; 59] que se refere ao uso de estratégias de dormência e TI. O segundo grupo trata de *estratégias de escolha de recursos e escalonamento de tarefas* que têm por objetivo economizar energia [37; 52; 27; 61; 42]. Neste capítulo, analisaremos esses trabalhos com o objetivo de destacar as semelhanças e diferenças em relação ao trabalho apresentado nesta dissertação. Destacamos, também, de que forma as propostas e os resultados apresentados por esses trabalhos podem ser utilizados na economia de energia em grades computacionais entre-pares.

3.1 Gerência de Recursos

As estratégias de dormência Sobreaviso e Hibernação têm sido amplamente utilizadas por sistemas operacionais [46; 13] para economizar energia em computadores pessoais. Nesses

sistemas, o valor de TI varia entre 5 e 15 minutos. Talebi et al. [62] mostra que o uso de Sobreaviso e Hibernação em computadores de salas de aula e laboratórios de pesquisa permite uma redução no consumo de energia em até 97%. Trabalho semelhante é apresentado pela Escola de Educação da Universidade de Indiana [47]. Esse projeto tem por objetivo implantar um mecanismo para colocar os computadores de mesa em um estado de dormência quando os mesmos não estiverem em uso. Junto com esses estados, utiliza-se um TI definido como 2 horas e 15 minutos. Em uma avaliação preliminar, obteve-se uma redução no consumo de energia de $48,3\%$ em um *cluster* de 11 computadores de mesa e uma redução de $30,9\%$ na ala de escritórios. Essa redução equivale a uma economia de até US$ $500.000,00$ por ano para a universidade.

Esses não são estudos isolados. Outras organizações também têm reportado significante economia de energia com o uso de estados de dormência [22]. A diferença desses trabalhos para o que apresentamos nessa dissertação é que o foco deles é a redução do consumo de energia utilizando estados de dormência, sem se preocupar com o tempo e o custo energético necessários para se colocar e retirar os computadores desses estados. Em uma grade computacional entre-pares, onde a máquina é explorada de forma oportunista, operações de entrada e saída da máquina da grade podem ser frequente, portanto, não é claro que o uso desses estados possa trazer economias similares.

Alguns estudos identificaram um desuso de estados de dormência em sistemas corporativos [20; 3; 59]. Eles mostram que nesses sistemas muitas vezes o usuário necessita manter todo seu ambiente de trabalho sempre ativo. O ambiente de trabalho inclui as conexões de rede que são interrompidas quando se utiliza estados de dormência. Essas conexões precisam ser mantidas ativas para possibilitar acessos remotos e o funcionamento de aplicativos conectados como: e-mail, aplicações do tipo *googledocs* e sistemas de compartilhamento de arquivos, como BitTorrent. Por essa razão, tem havido uma redução no uso de estados de dormência nesses sistemas [20; 3; 59].

Tendo em vista economizar energia nesses sistemas, têm sido propostas estratégias denominadas *proxy-sleeping*. Estratégias *proxy-sleeping* salvam todo o ambiente do usuário (estado da memória e as conexões) em máquinas virtuais e migram essas máquinas para um servidor remoto, assim o recurso torna-se totalmente ocioso e um estado de dormência pode ser usado [20; 3; 59]. Reich et al. [59] reporta que o uso do estado Sobreaviso nas máquinas,

após o ambiente do usuário ter sido migrado para um servidor remoto, permite uma economia de energia de até 80% em uma infraestrutura com 52 máquinas. Nesse ambiente, 95% das máquinas realizam até 10 transições de estado por dia. O estudo mostra também que um TI de 30 minutos permite significante economia de energia, mas ela pode ser ainda maior com o uso de um TI de 5 minutos.

Diferente do ambiente corporativo tratado por Reich et al., em grades computacionais entre-pares os usuários recebem incentivos para doar todo o poder computacional dos recursos exatamente quando esses recursos se encontram totalmente ociosos. Por essa razão, não se faz necessário o uso de *proxy-sleeping* nessas grades computacionais. Além disso, no ambiente corporativo analisado por Reich et al., o recurso pode ser adormecido sempre que ele não estiver em uso pelo usuário. Isso é diferente em uma grade computacional entre-pares em que quando o usuário local não está presente o recurso é disponibilizado para a grade e, dependendo da contenção da grade, o recurso pode ser adormecido ou pode ser utilizado para executar uma aplicação submetida por um usuário da grade. Deste modo, não é claro que nessas grades pode-se obter economia similar à reportada por Reich et al. Além disso, nas grades computacionais entre-pares, a métrica de avaliação da estratégia não se baseia apenas na economia de energia e no número de transições de estado. Nessas grades é necessário avaliar também o impacto da estratégia de dormência no tempo de resposta das aplicações submetidas à grade.

Em grades computacionais, identificamos que o *middleware* Condor [18], a partir da versão 7.2, passou a implementar um módulo de gerência de energia que faz uso de estados de dormência nos recursos dos domínios administrativos. A implementação define um *daemon* para verificar periodicamente a lista de máquinas que podem ser acordadas ou adormecidas. Essa verificação é configurada para ocorrer a cada 300 segundos. Porém, é necessário um TI de 2 horas para que a máquina seja adormecida. Não encontramos estudos científicos que tratam da avaliação das estratégias implementadas nesse *middleware*.

3.2 Estratégias de escolha de recursos e escalonamento de tarefas

Identificamos poucos trabalhos que analisam estratégias que definem o critério de escolha dos recursos que deverão ser acordados para executar uma aplicação [42; 18]. Lammie et al. [42] avaliam duas estratégias de escolha de recursos em um aglomerado de computadores. Essas estratégias são combinadas com o uso de DVFS. De um modo geral, os processadores de todas as máquinas são configurados para operarem na menor frequência possível e as estratégias de escolha consistem em acordar primeiro a máquina que executou a tarefa finalizada mais recentemente ou menos recentemente. A combinação dessas estratégias visa gerenciar o tempo que as máquinas permanecem executando de acordo com a carga de trabalho a que o sistema está submetido. Isso é justificável em um ambiente de recursos homogêneos e totalmente disponíveis. Em grades computacionais entre-pares, como os recursos são heterogêneos e a sua disponibilidade para a grade pode variar de modo significativo ao longo do tempo, a escolha dos recursos que serão acordados pode ter impacto nas métricas da aplicação e na economia de energia.

Nesse sentido, o *middleware* de grade computacional entre-pares OurGrid [17], na versão 4.2.1, define que quando é necessário escolher recursos para executar as tarefas de uma aplicação, deve-se escolher primeiro os recursos que se tornaram disponíveis mais recentemente. Essa estratégia pode diminuir o tempo de resposta das aplicações, uma vez que se espera que os recursos que se tornaram disponíveis mais recentemente tendam a ter uma sessão de disponibilidade para a grade maior e, assim, reduz-se a possibilidade do recurso ser requisitado pelo usuário local e a tarefa da grade precisar ser interrompida. Não existe avaliação do desempenho dessa estratégia nem em termos do desempenho da grade, nem em termos energéticos. Realizaremos essa avaliação nesta dissertação.

No módulo de gerência de energia do *middleware* Condor [18], o critério a ser usado na escolha das máquinas que serão acordadas deve ser fornecido pelo usuário. O usuário define uma expressão booleana e são acordadas as máquinas que atendem a essa expressão. Avaliamos, nesta dissertação, estratégias de escolha de recursos baseadas no tempo em que o recurso foi adormecido e nas suas características energéticas. Além disso, apresentamos um estudo do consumo de energia e do impacto que os estados de dormência têm nas métricas

da aplicação e no uso da infraestrutura.

Identificamos diversos trabalhos que visam economizar energia em grades por meio do ajuste da voltagem e/ou da frequência das CPUs [51; 11; 4]. Em muitos casos, essas estratégias são usadas quando uma tarefa não faz uso de todo poder de processamento da CPU. Os principais sistemas operacionais utilizados pelos recursos que compõem a grade computacional entre-pares, como *Windows* [46] e *Linux* [13], implementam algoritmos que identificam as frequências que a CPU dá suporte e fazem DVFS de acordo com as variações no uso da mesma [51]. Deste modo, neste trabalho, optamos por deixar que essa gerência da frequência em que a CPU opera ao longo da execução de uma tarefa para a grade seja realizada pelo sistema operacional instalado no recurso. Assim, nos dedicamos à análise do uso de estados de dormência quando a CPU e os demais componentes do computador encontram-se totalmente ociosos.

Por fim, identificamos trabalhos que combinam o uso de DVFS, estados de dormência e o escalonamento de tarefas com restrição de tempo para serem executadas em infraestruturas de grades computacionais que admitem que os usuários façam reservas de recursos [37; 52; 27; 61]. Nessas infraestruturas, podem-se tomar decisões entre adormecer ou acordar recursos para atender uma reservar e/ou cumprir a restrição de tempo da aplicação. Em grades computacionais entre-pares não há reserva de recursos e essas grades tipicamente não são utilizadas para executar aplicações que possuam restrição de tempo. O tempo de resposta é a principal métrica da aplicação nessas grades.

3.3 Considerações Finais do Capítulo

Apresentamos neste capítulo uma revisão do estado da arte da área de gerência de energia em sistemas computacionais. Nessa revisão, nos atemos a soluções que usam estados de dormência, TI e estratégia de escolha de recursos. Em particular, constatamos que não existem trabalhos que tratam do uso dessas estratégias em grades computacionais entre-pares. Essas grades apresentam um fator não investigado pelos demais trabalhos, que é aplicar estratégias de economia de energia em um ambiente onde os recursos são usados tanto pelos usuários locais como para executar aplicações. Deste modo, esta dissertação tem por objetivo preencher essa lacuna ao avaliar o uso de estratégias de economia de energia nesse ambiente.

Capítulo 4

Economia de Energia em Grades Computacionais Entre-Pares

Neste capítulo, apresentamos estratégias que visam economizar energia em grades computacionais entre-pares. Essas estratégias podem ser divididas em três grupos: *estados de dormência*, *estratégias de escolha de recursos* e *tempo de inatividade* (TI). Elas abrangem principalmente dois componentes dessas grades: os trabalhadores e os gerentes. O trabalhador é um componente da grade que executa em cada máquina da grade. Esse componente é responsável por executar as tarefas submetidas à grade e por reportar o estado da máquina para o gerente do seu domínio administrativo. O gerente é o serviço da grade que executa em cada domínio administrativo. Ele é responsável, por exemplo, por gerenciar os trabalhadores de um domínio administrativo e por se comunicar com outros gerentes com o objetivo de doar ou receber trabalhadores. Ao longo deste capítulo descreveremos como esses componentes são afetados pelo uso das estratégias de economia de energia.

4.1 Estados de Dormência de Recursos

Em sistemas computacionais é comum o uso das estratégias de dormência Sobreaviso e Hibernação para economizar energia quando uma máquina se encontra ociosa. Em termos técnicos, esses estados podem ser utilizados nas máquinas que compõem uma grade computacional entre-pares. Isso é possível porque esses estados permitem que as máquinas sejam acordadas eletronicamente por meio de um comando enviado a algum dispositivo da

máquina mantido em estado de espera, como a interface de rede Ethernet (*Wake-on-LAN*). Deste modo, o serviço gerente que executa em cada domínio administrativo da grade pode acordar ou adormecer as máquinas de acordo com a carga de trabalho a que a grade está submetida. Em razão disso, essas máquinas da grade que executam os gerentes dos domínios administrativos são as únicas que não estão sujeitas ao uso desses estados de dormência.

Uma máquina usada de modo oportunista pode operar em diversos estados ao longo do tempo. Na Figura 4.1 apresentamos um diagrama com os principais estados. O *estado usuário local* indica que a máquina está em uso pelo seu proprietário, portanto, a máquina não se encontra disponível para grade. A máquina se torna disponível quando o usuário local deixa de utilizá-la e se torna indisponível tão logo o usuário volte a utilizá-la. Uma máquina está no *estado ativo* quando ela se encontra executando uma tarefa para a grade. Quando a máquina se encontra disponível para a grade, mas não está executando uma tarefa, ela pode está no *estado ocioso*, em um *estado de dormência* ou realizando uma *transição dormir/acordar* entre um estado de dormência e algum outro estado da grade.

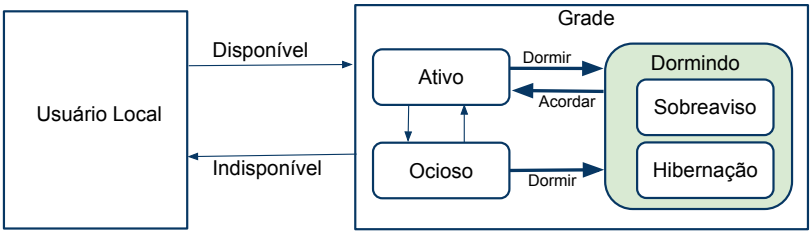


Figura 4.1. Estados de uma máquina na grade

Uma máquina permanece no *estado ocioso* (o) quando ela se encontra disponível para a grade, mas não há nenhuma tarefa da grade para ser executada. Ilustramos essa situação na Figura 4.2. Essa figura apresenta um segmento da linha do tempo de uma máquina usada de modo oportunista. A máquina inicia uma sessão de disponibilidade para a grade no instante de tempo t e a finaliza no instante t'. Durante essa sessão de disponibilidade, a máquina permanece dois momentos executando uma tarefa para a grade e outro momento ociosa. Nessa figura, indicamos o instante em que a máquina transita do estado ocioso para o estado ativo em razão da chegada de uma tarefa. No tempo em que uma máquina permanece ociosa, um estado de dormência pode ser usado com o objetivo de economizar energia.

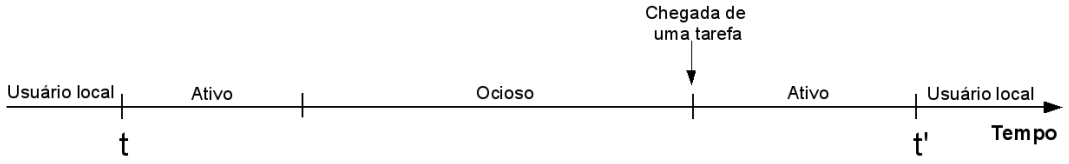


Figura 4.2. Segmento da linha do tempo de uma máquina usada de modo oportunista que é mantida ociosa quando não há uma tarefa da grade para ser executada. Cada marca na linha indica uma mudança de estado. O instante t indica o início de uma sessão de disponibilidade e o instante t' indica o fim dessa sessão.

Definimos, inicialmente, um *estado de dormência* d, que se aplica tanto a Sobreaviso como a Hibernação. Esse estado está associado a um tempo de transição. Esse tempo de transição é definido como latência do estado (L_d), i.e., o tempo necessário para que uma máquina mude de/para um estado de dormência. Durante a transição de estados, a máquina não responde a requisições realizadas pela grade ou pelo usuário local. Além disso, ela opera na potência de uma máquina ativa (P_a), que é a potência em que a máquina opera quando se encontra executando uma tarefa. Diferentemente do estado de dormência, a latência do estado ocioso é desprezível ($L_o = 0$). Portanto, para esse estado, o consumo de energia e o tempo gasto durante as transições não são significativos.

Definimos Δt_d como o tamanho de uma sessão de dormência, i.e., o tempo decorrido entre o instante em que a máquina terminou de transitar para um estado de dormência e o instante em que ela recebeu um comando para ser acordada. Durante uma sessão de dormência, a máquina opera em uma potência P_d que é menor que a potência P_o em que a máquina opera quando está ociosa. Na Figura 4.3, ilustramos o uso de um estado de dormência por uma máquina usada de modo oportunista.

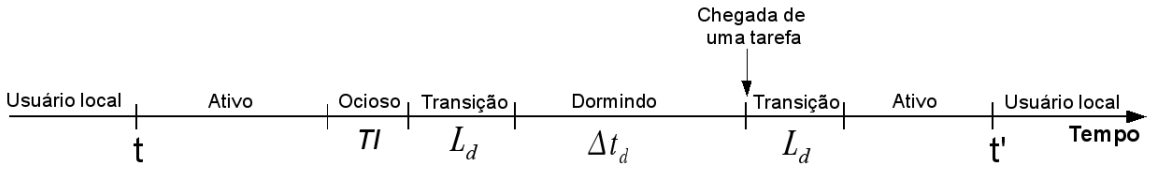


Figura 4.3. Segmento da linha do tempo de uma máquina usada de modo oportunista que faz uso de um estado de dormência quando não há uma tarefa da grade para ser executada.

Nessa figura, a máquina inicia uma sessão de disponibilidade e permanece um período executando uma tarefa da grade. Após terminar de executar a tarefa, ela permanece ociosa executando um temporizador (TI). Terminado o temporizador, a máquina inicia a transição para o estado de dormência e permanece L_d unidades de tempo nessa transição. Ela permanece em uma sessão de dormência de Δt_d unidades de tempo até ser escolhida pela grade para executar uma tarefa quando, então, inicia uma transição para o estado ativo. Essa transição dura L_d unidades de tempo. Por fim, a máquina permanece executando uma tarefa para a grade até ser requisitada pelo usuário local, no instante t'. Deste modo, nessa figura, faz-se uso de um estado de dormência quando a máquina está disponível para grade, mas não há tarefa para ser executada, diferentemente do apresentado na Figura 4.2 em que a máquina permanece ociosa.

Com o objetivo de comparar o consumo de energia de uma máquina que é mantida ociosa com o consumo de energia de uma máquina que faz uso de um estado de dormência, montamos as equações dessas duas situações. A Equação 4.1 permite calcular a quantidade de energia c_d consumida por uma máquina que permanece TI unidades de tempo aguardando a chegada de uma tarefa, realiza uma transição para um estado de dormência, permanece nesse estado durante Δt unidades de tempo e transita para o estado ativo.

$$c_d = 2 \times L_d \times P_a + \Delta t \times P_d + TI \times P_o \tag{4.1}$$

Essa equação consiste na soma da energia consumida durante TI ($TI \times P_o$), na transição de estados ($2 \times L_d \times P_a$) e durante o tempo de permanência no estado ($\Delta t \times P_d$). Do mesmo modo, a Equação 4.2 permite calcular a quantidade de energia c_o que seria consumida por essa máquina caso ela permanecesse no estado ocioso durante todo esse tempo. O que muda nessa equação em relação à Equação 4.1 é que durante o tempo que seria gasto com transição de estado ($2 \times L_d$) e o tempo de permanência no estado (Δt) a máquina opera na potência do estado ocioso (P_o).

$$c_o = 2 \times L_d \times P_o + \Delta t \times P_o + TI \times P_o \tag{4.2}$$

Dados os consumos c_d e c_o, pode-se calcular a quantidade de energia economizada pelo estado de dormência em relação ao estado ocioso, representado por ξ_d, como indicado na Equação 4.3. Essa economia é dada pela diferença entre e a energia consumida pela máquina

no estado ocioso (c_o) e a energia consumida no estado de dormência (c_d).

$$\xi_d = c_o - c_d \tag{4.3}$$

Existem situações em que c_o é menor que c_d, nessas situações o uso dos estados de dormência gera um aumento no consumo de energia de uma máquina em relação a mantê-la ociosa. Isso ocorre quando o custo energético de transitar do estado de dormência é maior que a economia gerada durante o tempo em que a máquina permaneceu adormecida. Há um ponto de equilíbrio em que o uso do estado de dormência não gera nem aumento nem economia de energia em relação ao estado ocioso. Chamamos esse ponto de $\Delta t_{d,o}$. A dedução de $\Delta t_{d,o}$ é apresentada na Equação 4.4.

$$\begin{aligned}
\overbrace{P_o \times 2 \times L_d + P_o \times \Delta t}^{\text{Energia Consumida usando o Estado Ocioso } (c_o)} &= \overbrace{P_a \times 2 \times L_d + P_d \times \Delta t}^{\text{Energia Consumida usando um Estado de Dormência } (c_d)} \\
P_o \times \Delta t - P_d \times \Delta t &= P_a \times 2 \times L_d - P_o \times 2 \times L_d \\
\Delta t(P_o - P_d) &= 2L_d(P_a - P_o) \\
\Delta t &= \frac{2 \times L_d \times (P_a - P_o)}{P_o - P_d} = \Delta t_{d,o}, \text{onde } P_o > P_d.
\end{aligned} \tag{4.4}$$

Essa equação mostra o tamanho da sessão de dormência $\Delta t_{d,o}$ para a qual o custo energético de fazer uso de um estado de dormência (c_d) é igual ao custo energético de manter a máquina ociosa (c_o). Para sessões de dormência de tamanho menores que $\Delta t_{d,o}$ a estratégia de dormência apresenta consumo de energia superior a manter a máquina ociosa. De outro modo, para sessões de dormência de tamanho superiores a $\Delta t_{d,o}$ a estratégia de dormência apresenta consumo de energia inferior a manter a máquina ociosa, gerando uma economia de energia. Nota-se, portanto, que a economia de energia provida por um estado de dormência está diretamente relacionada ao tamanho da sessão de dormência.

Em grades computacionais entre-pares, o tamanho da sessão de dormência pode ser determinado pelo gerente do domínio administrativo e/ou pelo padrão de uso da máquina pelo usuário local. O gerente do domínio administrativo pode determinar o tamanho das sessões de dormência ao decidir quando adormecer o recurso e entre acordá-lo ou não quando surge uma nova demanda. O usuário local determina o tamanho das sessões de dormência quando ele inicia e finaliza uma sessão de disponibilidade da máquina para a grade. Assim, em uma

grade computacional entre-pares tem-se a propriedade de que o tamanho da sessão de dormência de uma máquina (Δt_d) é limitado ao tamanho da sessão de disponibilidade ($t' - t$). Isso significa que a máquina permanecerá adormecida no máximo o tempo que o usuário local a deixará disponível para a grade.

Deste modo, para que uma estratégia de dormência apresente economia de energia na grade, as sessões de disponibilidade precisam ser superiores ao tempo que a máquina precisa permanecer adormecida para que gere economia de energia em relação a ser mantida ociosa, i.e., $t' - t > \Delta t_{d,o}$. Em consequência disso, mostra-se necessário avaliar o potencial de cada estado de dormência de economizar energia, considerando também essa característica das grades computacionais entre-pares.

Como justificamos anteriormente, nesta dissertação enfatizamos os estados de dormência Sobreaviso (s) e Hibernação (h). Seja $\Delta t_{s,o}$ (respectivo $\Delta t_{h,o}$) o tamanho mínimo da sessão de dormência para que não haja aumento do consumo de energia quando a máquina é colocada no estado de Sobreaviso (respectivo Hibernação). O valor de $\Delta t_{s,o}$ (respectivo $\Delta t_{h,o}$) é dado pela equação 4.4, substituindo L_d e P_d por L_s e P_s, onde L_s e P_s (respectivo L_h e P_h) são a latência e a potência da máquina no estado Sobreaviso (respectivo Hibernação).

A fim de comparar o consumo de energia dos estados Sobreaviso (c_s) e Hibernação (c_h), deduzimos na Equação 4.5 o tamanho da sessão de dormência $\Delta t_{s,h}$ para o qual esses estados apresentam um custo energético equivalente. A estratégia Sobreaviso apresenta menor consumo de energia que a estratégia Hibernação para sessões de dormência de tamanho menores que $\Delta t_{s,h}$. De outro modo, para tamanhos de sessões de dormência maiores que $\Delta t_{s,h}$, a estratégia Hibernação apresenta consumo de energia inferior a Sobreaviso.

$$
\begin{aligned}
\overbrace{P_a \times 2 \times L_h + P_h \times \Delta t}^{\text{Energia Consumida usando um Estado de Dormência } (c_h)} &= \overbrace{P_a \times 2 \times L_s + P_s \times \Delta t}^{\text{Energia Consumida usando um Estado de Dormência } (c_s)} \\
P_h \times \Delta t - P_s \times \Delta t &= P_a \times 2 \times L_s - P_a \times 2 \times L_h \\
\Delta t \times (P_h - P_s) &= 2 \times P_a \times (L_s - L_h) \\
\Delta t &= \frac{2 \times P_a \times (L_s - L_h)}{P_h - P_s} = \Delta t_{s,h}, \text{onde } P_s > P_h.
\end{aligned}
\tag{4.5}
$$

Uma vez definidas as relações entre os consumos de energia dos estados Sobreaviso, Hibernação e ocioso em função do tamanho da sessão de dormência, pode-se analisar o uso

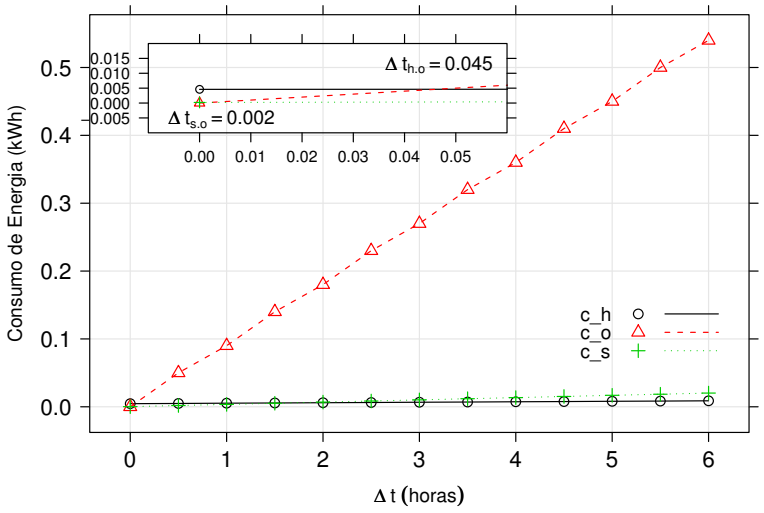
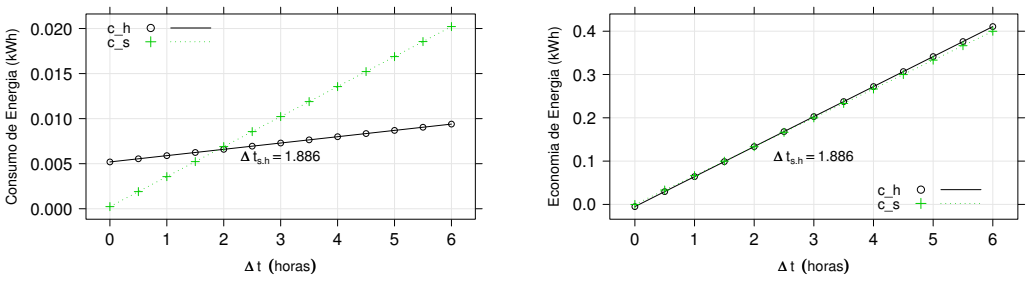


Figura 4.4. Comparação entre os estados ocioso, Sobreaviso e Hibernação em termos de consumo e economia de energia considerando uma máquina Intel E5200 @ $2,5$ GHz, 2 GB de RAM e 160 GB de HD, com as potências $P_a = 110$ Watts, $P_o = 70$ Watts, $P_s = 3,33$ Watts e $P_h = 0,7$ Watts e as latências $L_s = 2,5$ e $L_h = 55$ segundos.

dos estados de dormência em uma máquina usada de modo oportunista. Considere uma máquina típica com processador Intel E5200 @ $2,5$ GHz, 2 GB de RAM e 160 GB de HD e que opera nas seguintes potências: $P_a = 110$ Watts [9], $P_o = 70$ Watts [9], $P_s = 3,33$ Watts [29] e $P_h = 0,7$ Watts [29]. Considere também que os estados Sobreaviso e Hibernação apresentam, respectivamente, as seguintes latências de transição de estados $L_s = 2,5$ segundos [29] e $L_h = 55$ segundos [49]. A Figura 4.4 compara a energia consumida por essa máquina quando mantida ociosa, em Sobreaviso ou em Hibernação durante diferentes valores de Δt.

Ampliamos nessa figura o segmento do gráfico em que as retas de cruzam. Nesta ampliação destacamos $\Delta t_{s,o}$, tamanho de sessão de dormência a partir do qual o estado sobreaviso apresenta menor consumo de energia do que o estado ocioso e $\Delta t_{h,o}$, tamanho de sessão de dormência a partir do qual o estado Hibernação apresenta menor consumo de energia que o estado ocioso.

A Figura 4.5 mostra uma comparação apenas entre os estados Sobreaviso e Hibernação, em termos de consumo de energia (Figura 4.5(a)) e da economia de energia que eles possibilitam em relação ao estado ocioso 4.5(b). A Figura 4.5(a) mostra que existe diferença



(a) Energia consumida por uma máquina usando as estratégias Sobreaviso e Hiberanção

(b) Energia economizada por uma máquina usando as estratégias Sobreaviso em relação à máquina ser mantida no estado ocioso

Figura 4.5. Comparação entre os estados ocioso, Sobreaviso e Hibernação em termos de consumo e economia de energia considerando uma máquina Intel E5200 @ $2,5$ GHz, 2 GB de RAM e 160 GB de HD, com as potências $P_a = 110$ Watts, $P_o = 70$ Watts, $P_s = 3,33$ Watts e $P_h = 0,7$ Watts e as latências $L_s = 2,5$ e $L_h = 55$ segundos.

significativa no consumo de energia entre esses estados e indica que a partir do tamanho de sessão de dormência $\Delta t_{s,h} = 1,886$, a estratégia Sobreaviso apresenta maior consumo de energia que a estratégia Hibernação. Na Figura 4.5(b), em que se avalia a economia de energia provida em relação ao estado ocioso, as estratégias mostram-se próximas, mesmo com a variação de Δt entre 0 e 6 horas. En uma grade computacional entre-pares formada por máquinas com as características da máquina utilizada nessa análise, o estado Hibernação só apresentará economia de energia significativamente superior a Sobreaviso, se as máquinas apresentarem sessões de dormência superiores a $1,886$ horas, i.e., 1 hora e 53 minutos. No entanto, só é possível as máquinas permanecerem adormecidas por esse tempo se elas apresentarem sessões de disponibilidades maiores que $1,886$ horas ($t' - t > 1,886$).

Para avaliar se esse tamanho de sessão de disponibilidade é típico de grades computacionais entre-pares, foram analisados dados de uma grade computacional entre-pares em produção. A Figura 4.6 apresenta uma função de distribuição acumulada (FDA) do tamanho das sessões de disponibilidade do conjunto de 140 máquinas da grade computacional OurGrid [50] ao longo de uma semana. O 60 percentil dessa distribuição indica que 60% das sessões de disponibilidade têm duração igual ou inferior a 2 horas. Isso indica que mesmo se as máquinas permanecessem adormecidas durante todo o período de disponibilidade (i.e., $t' - t = \Delta t$) em 40% das sessões a estratégia Hibernação apresentaria economia de ener-

gia significativamente maior do que a estratégia Sobreaviso. Verificamos se esse tamanho de sessão de disponibilidade é característico do uso que o usuário local faz do recurso ou se é específico do rastro utilizado e encontramos diversos outros trabalhos que reportam resultados semelhantes [35; 40; 38].

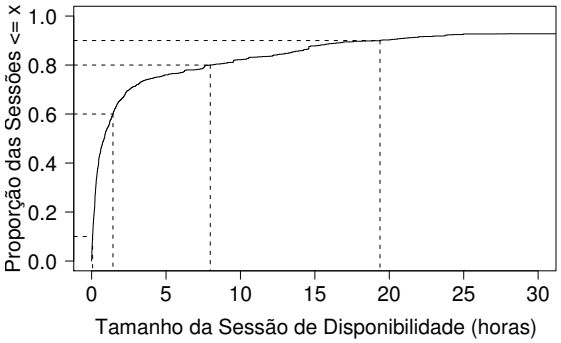


Figura 4.6. Função distribuição acumulada (FDA) do tamanho das sessões de disponibilidade de 140 máquinas ao longo de 6 dias. Dados obtidos na grade computacional OurGrid.

No entanto, ainda mostra-se necessário avaliar se ao se considerar todas as sessões de disponibilidades as estratégias apresentariam diferença significativa. Além disso, mostra-se necessário avaliar o desempenho do uso desses estados de dormência em diferentes cenários de contenção por recursos, uma vez que nesses cenários o tamanho da sessão de dormência variará não apenas com o tamanho da sessão de disponibilidade, mas também com a demanda de recursos na grade. Realizaremos essa análise nesta dissertação por meio um modelo simulado de uma grade computacional entre-pares.

4.2 Tempo de Inatividade (TI)

Nesta sessão, discutimos o impacto do uso do tempo de inatividade (TI) em conjunto com estados de dormência em uma grade computacional entre-pares. TI define após quanto tempo de inatividade um recurso deverá ser adormecido, portanto, é um tempo máximo que uma máquina pode permanecer no estado ocioso aguardando a chegada de uma nova requisição antes que seja colocada em um estado de dormência. Em uma grade computacional entre-pares, cada máquina que possui um componente trabalhador instalado pode iniciar TI e entrar

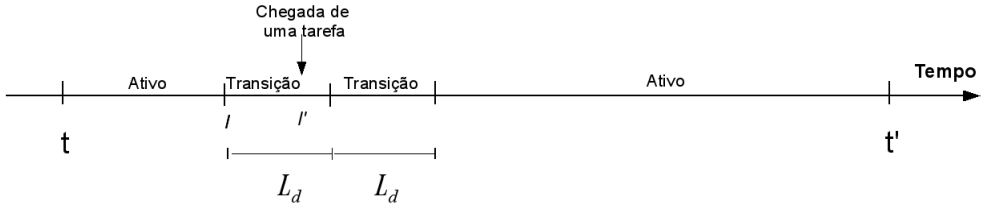
em um estado de dormência, caso necessário. Geralmente, a implementação de TI utiliza um temporizador (*timeout*) que é iniciado quando a máquina torna-se inativa. A máquina é adormecida caso o temporizador expire e nenhuma requisição tenha sido recebida. De outro modo, se a qualquer instante ao longo do intervalo de temporização a máquina receber alguma requisição (do usuário local ou da grade) o temporizador é interrompido e a máquina inicia a execução da requisição instantaneamente.

De uma forma geral, a definição de TI está relacionada ao intervalo entre requisições recebidas pelas máquinas. Se o intervalo entre requisições é longo, pode ser vantajoso em termos de economia de energia transitar o recurso para um estado de dormência tão logo ele se torne ocioso. De outro modo, se esse intervalo for pequeno, pode ser mais vantajoso, em termos do impacto no tempo de resposta da aplicação, manter o recurso ocioso aguardando a chegada de uma nova requisição. Nesse sentido, a Figura 4.7 apresenta três casos relacionados ao uso de TI em uma máquina usada de modo oportunista e que faz uso de um estado de dormência quando se encontra inativa. Cada figura apresenta um segmento da linha do tempo de uma máquina.

A Figura 4.7(a) mostra uma situação em que TI não é utilizado, o que gera implicações negativas no tempo de resposta da requisição e no consumo de energia da máquina. Nessa figura, a máquina torna-se disponível para a grade, permanece algum tempo executando uma tarefa da grade e, em seguida, no instante l, ela inicia a transição para um estado de dormência. No entanto, no instante l' a máquina é alocada para executar uma tarefa. Neste caso, a máquina precisa terminar a transição para o estado de dormência, para então, tomar conhecimento da requisição e realizar uma transição novamente para o estado ativo e, apenas, após isso ela pode iniciar a execução da tarefa. Neste caso ocorreram dois problemas.

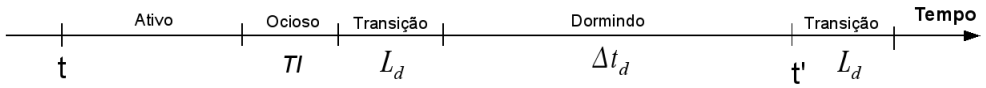
O primeiro problema é que a requisição permaneceu aguardando até que a máquina estivesse totalmente disponível para executar a tarefa, gerando um aumento no tempo de resposta referente ao tempo em que a máquina permaneceu realizando transição após o instante que a tarefa foi submetida. Esse aumento gerado no tempo de resposta pode ser calculado como: $2 \times L_d - (l' - l)$.

O segundo problema é que a transição para o estado de dormência foi inútil, uma vez que a máquina não permaneceu adormecida por um tempo significativo nem para pagar o custo energético da transição. Deste modo, ocorreu um aumento no consumo de energia em



(a) Problema do não uso de TI. Uma tarefa é submetida quando decorridos $l' - l$ unidades de tempo do início da transição para um estado de dormência. A máquina inicia a execução da tarefa, apenas, depois de completar a transição para o estado dormência e transitar novamente para o estado ativo. Neste caso, não há benefício no uso do estado de dormência

(b) Benefício do uso de TI. Uma tarefa é submetida durante a execução do temporizador. A máquina interrompe o temporizador e inicia a execução da tarefa. Não ocorreu atraso no tempo de resposta nem aumento do consumo de energia.



(c) Problema do uso de TI. Uma máquina executa todo o temporizador (TI) e, somente após isso, ela faz a transição para um estado de dormência. A máquina deixou de economizar energia durante o tempo em que permaneceu executando o temporizador.

Figura 4.7. Casos do uso de TI em uma máquina usada de modo oportunista.

relação à máquina ter sido mantida ociosa. A máquina permaneceu durante todo o tempo da transição operando à potência do estado ativo enquanto se estivesse ociosa, permaneceria nesse estado por menos tempo do que o tempo de transição e durante esse tempo operaria à potência do estado ocioso, que é menor que a potência do estado ativo. Esse aumento gerado no consumo de energia pode ser calculado como: $2 \times L_d \times P_a - P_o \times (l' - l)$.

A Figura 4.7(b) apresenta uma situação análoga à apresentada na Figura 4.7(a), com a diferença de que a máquina faz uso de TI. Nesse exemplo, o uso de TI resolve os problemas destacados na Figura 4.7(a). Neste caso, a máquina torna-se disponível para a grade, permanece algum tempo executando uma tarefa da grade e, em seguida, inicia TI. A máquina é então requisitada faltando $TI - (l' - l)$ unidades de tempo para TI ser completado. Neste

caso, não ocorreu impacto no tempo de resposta da requisição, uma vez que a máquina respondeu à requisição instantaneamente. Além disso, não ocorreu aumento ou economia na energia consumida pela máquina.

Existem, também, situações em que o uso de TI não é vantajoso. A Figura 4.7(c) apresenta uma dessas situações. Nessa figura, a máquina torna-se disponível para a grade, permanece algum tempo executando uma tarefa da grade e, em seguida, fica inativa e inicia a execução do temporizador. A máquina completa a execução do temporizador sem que nenhuma requisição seja recebida, então ela inicia a transição para um estado de dormência. Neste caso, a máquina permaneceu TI unidades de tempo no estado ocioso quando já poderia ter transitado para um estado de dormência. A energia que a máquina deixou de economizar pode ser calculada como: $(TI - (l' - l)) \times (P_o - P_d)$.

A questão que se impõe no uso do mecanismo de temporização é a definição do valor de TI, pois esse valor é que determina quando a máquina deve ser adormecida. O desejável é que TI seja grande o suficiente para que a máquina permaneça ociosa apenas nas situações em o tempo decorrido entre o término da última requisição e a chegada de outra não é grande o suficiente para que adormecê-la gere uma economia de energia. Este caso também minimiza o impacto gerado no tempo de resposta da requisição. Esse impacto não existirá na situação em que a máquina estiver executando o temporizador quando receber a requisição. De outro modo, na situação em que a máquina estiver adormecida, o impacto será de L_d, que é o menor impacto que se pode obter quando se faz uso de um estado de dormência.

Pode-se utilizar uma estratégia de predição para determinar o tamanho de TI. No entanto, temos verificado que em situações práticas predomina-se o uso de um valor fixo de TI [29; 46; 13] que pode ser configurado pelo usuário (ou administrador) do sistema. A EnergyStar recomenda o uso de TI de 300 segundos e outros trabalhos na literatura também fazem uso desse valor de TI [29; 42; 59]. Geralmente, o sistema operacional Windows e Linux utilizam um TI de 600 segundos [46; 13], que pode ser alterado pelo usuário. Por fim, a ACPI recomenda o uso de um temporizador de 900 segundos [19]. Nos próximos capítulos avaliaremos o uso desses valores de TI em uma grade computacional entre-pares. Também avaliaremos como o uso de diferentes valores de TI impacta na economia de energia e no tempo de resposta das aplicações submetidas a uma grade computacional entre-pares.

4.3 Escolha de Recursos

O problema da escolha de recursos pode ser descrito da seguinte forma. Dado M_d o conjunto de máquinas que se encontram adormecidas em um domínio administrativo da grade e n a quantidade de máquinas necessárias para atender a uma nova demanda, todas as máquinas do conjunto M_d precisarão ser acordadas se $n \geq |M_d|$, i.e., se a demanda de máquinas maior ou igual ao número de máquinas adormecidas. Caso contrário, se $n < |M_d|$, faz-se necessário utilizar algum critério para definir quais máquinas do conjunto M_d devem ser acordadas para atender essa demanda.

De certa maneira, esse problema de escolha de recursos já existe em grades computacionais, mesmo sem o uso de estados de dormência. Por exemplo, a grade computacional OurGrid até a versão 4.2.1 define que quando surgir uma demanda devem ser escolhidas primeiro as máquinas mais recentemente adormecidas (MRS, do inglês *Most Recently Sleeping*), i.e., que estão a menos tempo nesse estado. Deste modo, MRS mantém M_d como uma pilha dos identificadores das máquinas. As máquinas são empilhadas na medida em que realizam uma transição para um estado de dormência. Quando alguma máquina precisa ser acordada, MRS apenas desempilha uma máquina.

Ao se utilizar essa estratégia considerando o início da sessão de disponibilidade, espera-se que as máquinas escolhidas tenham uma sessão de disponibilidade para a grade maior. Caso isso ocorra, o uso da estratégia contribui, por exemplo, na redução da possibilidade de que uma tarefa alocada à máquina seja preemptada por uma requisição do usuário local. Quanto à complexidade da estratégia, as operações de inserção e de remoção de máquinas em M_d utilizando a estratégia MRS podem ser realizadas em tempo polinomial com complexidade $O(1)$, por se tratar da implementação de uma pilha.

Além de MRS, nesta dissertação avaliamos o aspecto energético de outras três estratégias de escolha: primeiro as máquinas menos recentemente adormecidas (LRS, do inglês *Least Recently Sleeping*); primeiro as máquinas com maior eficiência energética (EA, do inglês *Energy Aware*); e escolha aleatória. Usamos essa última estratégia como uma estratégia de referência para avaliar as demais. A seguir apresentamos as principais características dessas estratégias.

A estratégia de escolha das máquinas menos recentemente adormecidas (LRS) consiste em acordar primeiro as máquinas adormecidas há mais tempo. Deste modo, LRS mantém

M_d como uma fila dos identificadores das máquinas. As máquinas são colocadas nessa fila à medida que elas realizam uma transição para o estado de dormência. Quando alguma máquina precisa ser acordada, LRS escolhe a primeira máquina da fila. Essa estratégia possibilita que uma máquina mais recentemente adormecida (a última da fila) só seja acordada quando todas as máquinas adormecidas antes dela já tenham sido acordadas. Isso visa evitar, por exemplo, que uma mesma máquina seja adormecida e reativada várias vezes enquanto outra máquina permanece adormecida durante muito tempo. As operações de inserção e de remoção de máquinas em M_d, utilizando a estratégia LRS, podem ser realizadas em tempo polinomial com complexidade $O(1)$, por se tratar da implementação de uma fila.

Por fim, a estratégia de escolha ciente da eficiência energética das máquinas (EA) consiste em acordar primeiro as máquinas mais eficientes no aspecto energético. Uma das formas de implementar EA é manter M_d como uma lista das máquinas ordenada de forma crescente pela eficiência energética. Quando uma nova máquina for adormecida, EA verifica a eficiência energética da mesma, calculada como a frequência da CPU dividido pela potência de operação quando mantida ativa (f/P_a), e a insere na lista. Quando alguma máquina precisar ser acordada, EA escolhe a última máquina da lista, i.e., a máquina que possui a maior eficiência energética. Essa estratégia visa selecionar os recursos que tendem a economizar mais energia e a ter maior poder de processamento, que tendem a ser processadores mais modernos [41]. Com a estratégia EA implementada como uma lista ordenada, a operação de inserção de máquinas em M_d pode ser feita em $O(|M_d|)$, no pior caso, e a remoção pode ser feita em $O(1)$.

4.4 Considerações Finais

Neste capítulo propomos estratégias a fim de economizar energia em grades computacionais entre-pares. Essas estratégias podem ser divididas em três categorias: estratégias de dormência, tempo de inatividade (TI) e estratégias de escolha de recursos. As estratégias de dormência visam reduzir o consumo de energia da grade nos períodos em que a mesma experimenta uma baixa contenção de recursos e, portanto, existem recursos ociosos. Mostramos que a economia de energia provida pelas estratégias é limitada à quantidade de energia que pode ser economizada em uma sessão de dormência de tamanho igual à sessão de disponibilidade.

Propomos a avaliação de duas estratégias de dormência que podem ser implementadas em grades computacionais entre-pares: Sobreaviso e Hibernação.

Mostramos que, associado ao uso de estratégias de dormência, tem-se dois grupos de estratégias: TI e escolha de recursos. TI são limiares utilizados para decidir por quanto tempo um recurso deve permanecer ocioso aguardando a chegada de uma requisição antes que o mesmo seja adormecido. Mostramos que diferentes valores de TI podem impactar de modo diferente a economia de energia e o tempo de resposta das aplicações. Assim, propomos a avaliação, na grade computacional entre-pares, de valores de TI entre 0 e 1.800 segundos. Isso permite avaliar TIs utilizados em trabalhos realizados em outros ambientes e/ou que se encontram implementados em sistemas reais [29; 42; 59; 46; 13; 19] e ainda explorar novos cenários.

Por fim, apresentamos as estratégias de escolha de recursos. Essas estratégias são usadas para escolher dentre um conjunto de recursos que se encontram adormecidos em um domínio administrativo, qual subconjunto de recursos deve ser acordado para atender uma nova demanda. Apresentamos as estratégias EA, MRS e LRS que exploram a eficiência energética dos recursos e o tempo em que os mesmos foram adormecidos, respectivamente. Destacamos a necessidade da avaliação do desempenho dessas estratégias em um ambiente de grade computacional entre-pares.

O próximo capítulo desta dissertação apresenta os materiais e os métodos utilizados para avaliação dessas estratégias.

Capítulo 5

Materiais e Métodos de Avaliação

Neste capítulo apresentamos os materiais e métodos utilizados na avaliação das estratégias de dormência, TI e estratégia de escolha de recursos. Os materiais e métodos foram definidos a fim de atingir os seguintes objetivos:

1. Estimar o benefício da economia de energia e o custo associado ao atraso gerado pelas estratégias no tempo de resposta das aplicações em um ambiente de grade computacional entre-pares sujeito a diferentes níveis de contenção por recursos;

2. Verificar o quanto o uso das estratégias de dormência na grade computacional entre-pares pode aumentar o número de transições realizadas pelas máquinas e, portanto, reduzir a vida útil do disco rígido.

Nas próximas seções descrevemos as métricas utilizadas na avaliação de desempenho, o modelo de simulação utilizado, as configurações da grade e os rastros utilizados nas simulações. Por fim, apresentamos os cenários avaliados.

5.1 Métricas de Avaliação

Definimos três métricas de avaliação: *economia de energia*, *atraso* e *número de transições*. As métricas economia de energia e atraso são relativas a duas configurações da grade, que definimos de modo genérico como configuração A e configuração $\overline{A}$. Uma configuração A é formada por uma estratégia de dormência, uma estratégia de escolha de recursos e um valor de TI. Uma configuração $\overline{A}$ utiliza a mesma estratégia de escolha de recursos e valor

de TI que a configuração A, mas mantém as máquinas ociosas, i.e., $\overline{A}$ não faz uso de uma estratégia de dormência.

A *economia de energia* ξ^A é o percentual de energia economizada em uma configuração A em relação à configuração de referência $\overline{A}$, apresentado de modo formal na Equação 5.1, onde E^X é a energia consumida na configuração X.

$$\xi^A = \frac{E^{\overline{A}} - E^A}{E^{\overline{A}}} \times 100 \tag{5.1}$$

O *atraso* é o percentual de aumento gerado no tempo de respostas 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 término da última de suas tarefas. Deste modo, o atraso da aplicação é dado por β^A, que consiste no percentual de atraso que a configuração A gerou no tempo de resposta da aplicação em relação ao tempo de resposta obtido com a configuração de referência, apresentado de modo mais formal na Equação 5.2, onde m^X é o tempo de resposta da aplicação na configuração X.

$$\beta^A = \frac{m^A - m^{\overline{A}}}{m^{\overline{A}}} \times 100 \tag{5.2}$$

Por fim, 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 dormência. Deste modo, conta-se uma transição a cada vez que a máquina é adormecida ou acordada.

5.2 Modelo de Simulação

Existem diversos simuladores para grades computacionais, tais como o SimGrid [15] que fornece funcionalidades para simulação de aplicações distribuídas em ambientes distribuídos com recursos heterogêneos e o DGSim [5; 34] que possibilita modelar cargas de trabalho e avaliar a interoperabilidade de grades computacionais. No entanto, não encontramos um simulador de grades computacionais entre-pares que considera as características energéticas dos recursos e que permita avaliar todas as métricas relativas ao consumo de energia, como definidas na Seção 5.1. Em decorrência disso, propomos um modelo de simulação para esse fim.

O modelo de simulação proposto é baseado no *middleware* de grade computacional entre-pares OurGrid [17]. A Figura 5.1 apresenta uma visão geral da arquitetura desse *middleware*, onde destacamos os componentes *peers*, *workers* e *brokers*. Cada domínio administrativo é um *site* da grade. No OurGrid, o gerente do domínio administrativo é chamado *peer*. Existe um agente da grade instalado em cada recurso da grade que é utilizado para executar aplicações, esse agente é chamado *worker*. Por fim, o *broker* é a interface usada pelos usuários da grade para submeter suas aplicações. O *broker* também é responsável por escalonar as tarefas de uma aplicação para os recursos disponíveis. Dado que nesta dissertação não avaliamos políticas de escalonamento, consideramos que a política de escalonamento utilizada pelo *broker* é fixa e consiste em escalonar a primeira tarefa criada para o primeiro *worker* recebido (FCFS, do inglês *First-Come First-Served*).

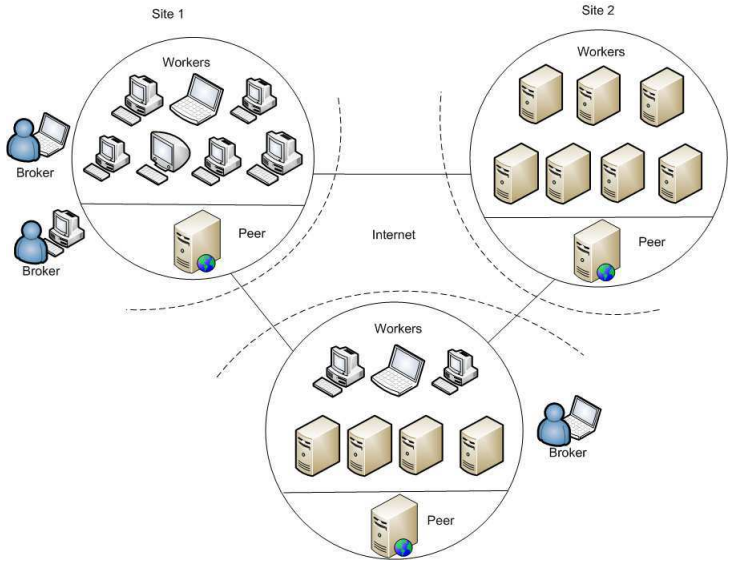


Figura 5.1. Visão Geral da Arquitetura do OurGrid.

O OurGrid utiliza a rede de favores (NoF) [7] como um mecanismo de incentivo à colaboração de recursos computacionais entre os *peers*, desencorajando o ato de *peers* que não doam *workers* (*free-riding*). Com o uso da NoF, a alocação de *workers* entre *peers* é baseada na reputação dos *peers*. Deste modo, a reputação de um *peer* D em um *peer* B é obtida a partir da contabilidade dos favores, em termos de tempo de execução, ofertados pelo *peer* B ao D e dos favores recebidos por B de D. Um *peer* pode preemptar *workers* doados a outros *peers* quando surge uma requisição de um *peer* ao qual ele deve mais favores.

Para definição das máquinas (*workers*) que compõem a grade é utilizado um arquivo de

configuração. O arquivo de configuração define os domínios administrativos e as máquinas que compõem cada domínio, um exemplo é apresentado na Figura 5.2. Por exemplo, a primeira linha dessa figura define, respectivamente, um domínio administrativo de identificador 1, a máquina `ostra` que possui 1 core, a CPU dessa máquina opera na frequência de 2.333.000 kHz e na potência de 065 Watts quando ela está ativa, a máquina pode operar nos seguintes estados: *ocioso*, com latência de 0 segundos e potência 67 Watts; *Sobreaviso* (`standby`), com latência de $2,5$ segundos e potência de $3,34$; e *Hibernação* (`hibernate`), com latência de 55 segundos e potência de $0,7$ Watts.

O modelo de simulação desenvolvido é de eventos discretos dirigido por rastros. O simulador recebe como entrada um rastro que descreve a variação na disponibilidade das máquinas e outro rastro que descreve a submissão de aplicações para serem executadas na grade.

```
1 1 ostra 1 2333000 065 {ocioso:0;67} {standby:2.5;3.34} {hibernate:55;0.7}
2 1 cangu 1 3000000 120 {ocioso:0;67} {standby:2.5;3.34} {hibernate:55;0.7}
3 1 palmi 1 2800000 095 {ocioso:0;67} {standby:2.5;3.34} {hibernate:55;0.7}
4 2 urutu 1 2800000 095 {ocioso:0;67} {standby:2.5;3.34} {hibernate:55;0.7}
5 2 palha 1 3000000 120 {ocioso:0;67} {standby:2.5;3.34} {hibernate:55;0.7}
```

Figura 5.2. Exemplo de um arquivo de configuração dos recursos que compõem a grade.

A variação na disponibilidade das máquinas para a grade ao longo do tempo é uma característica fundamental nas grades computacionais entre-pares. Nessas grades, uma máquina está disponível para grade apenas quando ela não está em uso pelo seu usuário local. Nesse sentido, o rastro de variação da disponibilidade das máquinas para a grade determina os momentos em que uma máquina se torna disponível para grade e os momentos em que é requisitada pelo seu proprietário. Cada máquina do arquivo de configuração da grade tem sua disponibilidade descrita nesse rastro.

Como exemplificado na Figura 5.3(a), o rastro de variação na disponibilidade das máquinas é composto de quatro campos: identificador do *peer* responsável pela máquina, identificador da máquina, tempo em que o evento ocorre, tipo do evento que pode ser: `grade`, caso a máquina esteja se tornando disponível para a grade ou `usuário`, caso a máquina

esteja sendo requisitada pelo usuário local. Se uma máquina for requisita pelo usuário local quando a mesma estiver executando uma tarefa da grade, todo o processamento da tarefa realizado até então é perdido. Quando isso ocorre, a tarefa é escalonada para outra máquina, onde reinicia a execução.

```
1 1 ostra 0 grade
2 1 cangu 0 grade
3 1 cangu 500 usuário
4 1 ostra 800 usuário
5 1 cangu 1500 grade
```

(a) Disponibilidade das Máquinas

```
1 1 J1 1 1050208880 218
2 2 J5 1 1050208880 218
3 1 J2 1 1050208892 1107
4 1 J2 2 1050208892 7231
5 2 J6 1 1050208892 1107
```

(b) Submissão de Aplicações

Figura 5.3. Exemplos de rastros que descrevem a variação na disponibilidade de cada máquina para a grade (a) e a submissão de aplicações do tipo saco-de-tarefas (b).

Outra característica importante nas grades computacionais entre-pares é que elas são usadas para executar aplicações do tipo saco-de-tarefas, aplicações compostas por tarefas que não se comunicam entre si. Nesse sentido, o rastro de submissão de aplicações consiste na descrição das tarefas das aplicações do tipo saco-de-tarefas submetidas à grade. Como exemplificado na Figura 5.3(b), este rastro possui cinco campos: identificador do *peer* em que a aplicação está sendo submetida, identificador da aplicação, identificador da tarefa, tempo de submissão da aplicação e um tempo de execução estimado da tarefa em uma máquina de referência. Pode haver uma ou mais tarefas associadas a cada aplicação. Quando a aplicação é composta por mais de uma tarefa, cada tarefa é definida em uma linha do rastro no qual há a repetição do identificador da aplicação. Cada tarefa da aplicação é composta de um identificador e um tempo de execução estimado, mas necessariamente devem possuir tempos de submissão iguais.

Considere o seguinte exemplo da dinâmica da simulação. Quando um *broker* interno ao domínio administrativo do *peer* D submete uma aplicação do tipo saco-de-tarefas constituída de n tarefas, a simulação de eventos discretos ocorre como segue:

1. O *peer* D verifica se existem pelo menos n *workers* disponíveis no próprio domínio

administrativo. Caso existam, ele escolhe n *workers* usando uma estratégia de escolha de recursos (apresentada na Seção 4.3) e aloca esses *workers* para o *broker* que fez a solicitação. De outro modo, caso o número de *workers* disponíveis no domínio administrativo (w) seja menor que n, o *peer* D solicita n - w *workers* a todos os outros *peers* que ele conhece;

2. Cada *peer* que recebe a requisição de D calcula quantos *workers* ele pode doar. Esse cálculo equivale à soma dos *workers* que se encontram ociosos, que podem ser acordados e que podem ser peemptados de outros *peers* usando a NoF. Após isso, cada *peer* informa a D quando *workers* pode doar. Esse valor é limitado à quantidade de *workers* solicitados por D;

3. Assim que D recebe o número de *workers* que cada *peer* pode doar, ele decide quantos *workers* confirmar com cada um deles. Se a soma do que foi recebido for menor que n, D confirma com todos os *peers* e periodicamente faz novos pedidos da quantidade que falta até que não existam tarefas para serem escalonadas. De outro modo, se a soma dos *workers* recebidos é maior que $n - w$, D confirma com os *peers* cujas respostas chegaram primeiro, até um total de $n - w$ *workers*.

4. Recebida a confirmação de D, cada peer aloca os *workers* ao *broker* de D, caso necessário eles acordam e/ou preemptam *workers* de outros *peers* usando a NoF, nessa ordem;

5. O *broker* de D recebe os *workers* em uma ordem aleatória. Após isso, ele escalona as n primeiras tarefas criadas para os primeiros *workers* recebidos (FCFS). Quando um *worker* termina de executar a tarefa, o *broker* escalona para ele a próxima tarefa da fila. Quando não há mais tarefa na fila, o *worker* é devolvido ao *peer* doador;

6. Quando um *worker* se encontra ocioso, ele inicia um temporizador (como discutido na Seção 4.2) a fim de definir por quando tempo permanecerá inativo aguardando a chegada de uma nova requisição. Quando o temporizador expira e nenhuma requisição é submetida, a máquina é colocada em um estado de dormência (como discutido na Seção 4.1).

Dada essa dinâmica de simulação, cada máquina da grade pode operar em diversos estados entre o instante em que ela inicia e o instante em que ela finaliza uma sessão de disponibilidade para a grade. Nesse sentido, a Figura 5.4 apresenta o diagrama de estados que indica os estados em que uma máquina pode operar ao longo de uma simulação e, também, os eventos que geram as transições de estado.

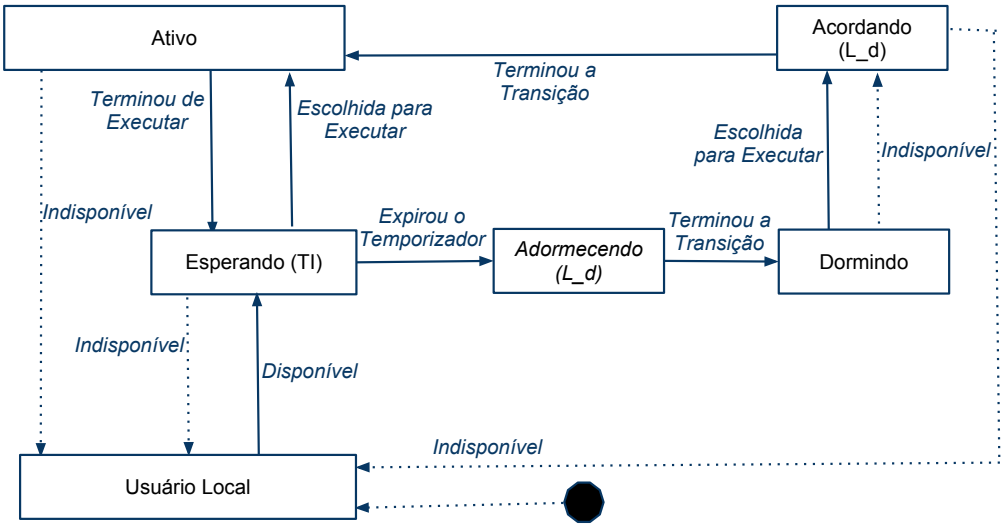


Figura 5.4. Diagrama de transição de estados de uma máquina usada de modo oportunista ao longo da simulação.

Ao longo da simulação são calculados os tempos de resposta das aplicações, o tamanho das sessões de dormência, o número de transições realizadas, o tempo permanecido e a energia consumida por cada máquina em cada estado. Nosso modelo de simulação não considera a energia consumida pelas máquinas quando as mesmas estão em uso pelo usuário local, uma vez que nesses períodos a máquina não está disponível para a grade.

5.3 Descrição dos Recursos da Grade computacional

Para simular as características energéticas e de processamento das máquinas que compõem a grade, modelamos uma amostra aleatória de 50 máquinas usadas no Laboratório de Sistemas Distribuídos da Universidade Federal de Campina Grande. Criamos um perfil de máquinas constituído pelas frequências máximas e mínimas das CPUs, potência associada a cada frequência, número de núcleos, memória, placa mãe, disco rígido e os componentes que pos-

sibilitam o WoL (apresentados no Capítulo 2). Esses são os componentes do computador que geram maior no consumo de energia [21]. Simplificação semelhante é realizada em outros estudos [25; 53].

Ajustamos uma distribuição empírica do perfil dessas máquinas. A Figura 5.5 apresenta uma distribuição empírica das frequências das CPUs, que mostra que a amostra é heterogênea. Essa distribuição permite gerar novas máquinas com o perfil dessa amostra. Deste modo, é possível construir grades com diferentes quantidades de máquinas, mas preservando as características desta amostra.

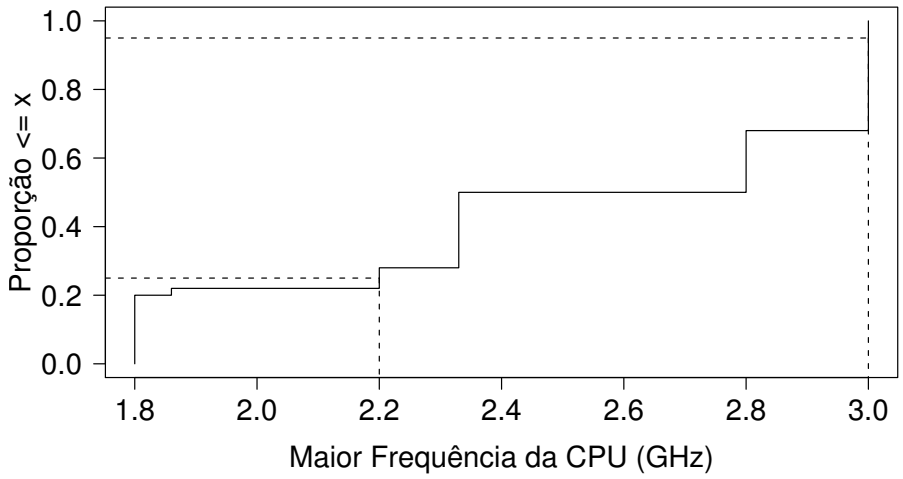


Figura 5.5. Função de distribuição acumulada (FDA) das frequências Máximas das CPUs de uma amostra de 50 máquinas

Todos os componentes listados anteriormente encontram-se energizados quando a máquina está executando uma tarefa. Alguns componentes possuem valores típicos que não tendem a variar significativamente de uma máquina para outra, são eles: memória (9 Watts [25]), disco rígido ativo ($9,5$ Watts [14]), placa mãe (25 Watts [25]) e os componentes que possibilitam o WoL ($0,7$ Watts [21]). De outro modo, a potência consumida pelo processador varia consideravelmente com a tecnologia de fabricação e as frequências que o processador admite. Considerando que as tarefas submetidas à grade fazem uso intensivo de processamento, o processador sempre opera na frequência máxima quando está executando uma tarefa. Assim, a potência em que o processador opera quando está na maior frequência é a maior potência em que ele pode operar (TDP, do inglês *Thermal Design Power*). Os

processadores das máquinas utilizadas neste trabalho foram fabricados pela empresa Intel. Obtivemos o TDP dos processadores na página Web[1] desse fabricante. As potências das máquinas quando estão executando uma tarefa, obtidas desse modo, são apresentadas na FDA da Figura 5.6, que mostra que a amostra de máquinas é heterogênea em termos da potência no estado ativo.

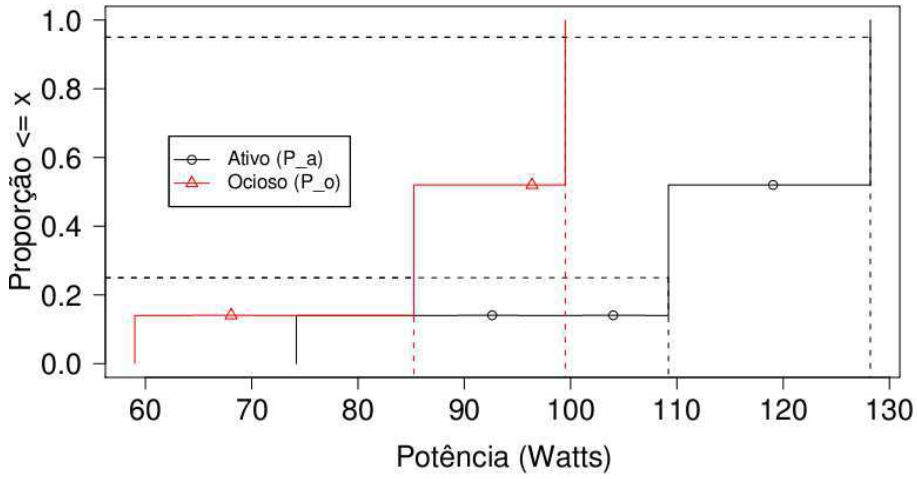


Figura 5.6. Função de distribuição acumulada (FDA) das potências nos estados executando e ocioso de uma amostra de 50 máquinas.

Quando a máquina está ociosa, o disco rígido reduz as rotações e passa a consumir menor potência, tipicamente, $6,0$ Watts [14]. O processador passa a operar na menor frequência possível. A potência em que o processador opera quando está nessa frequência é obtida como uma função linear, como feito por Gandh et al. [26]. Os demais componentes operam na mesma potência do estado ativo. Deste modo, as potências das máquinas no estado ocioso são obtidas a partir da soma das potências desses componentes. Assim, as potências da amostra de máquinas no estado ocioso são apresentadas na FDA da Figura 5.6. Nessa amostra, a redução média da potência do estado ocioso em relação ao estado ativo é de 67%. Essa redução está dentro do intervalo de 49–78% de redução, caracterizado por trabalhos anteriores [12; 20; 62].

A potência das máquinas no estado Sobreaviso é $3,33$ Watts [29], que corresponde à soma das potências da memória em estado de auto *refresh* e dos componentes que possibi-

[1] http://ark.intel.com/Default.aspx

litam o WoL. De outro modo, a potência no estado Hibernação é $0,7$ Watts, que equivale à soma das potências dos componentes que possibilitam o WoL. Essas são as potências utilizadas nas simulações realizadas nesta dissertação. A latência típica do estado Sobreaviso é $2,5$ [29], que equivale ao tempo gasto para desenergizar/reenergizar os componentes utilizados quando o computador está ativo. De outro modo, a latência do estado Hibernação é 55 segundos [49; 48], que corresponde ao tempo gasto para ativar/desativar o sistema operacional e mover os dados entre a memória RAM e o disco. Durante a latência, a máquina opera na potência do estado ativo.

As potências e latências definidas para os estados Sobreaviso e Hibernação não costumam variar significativamente entre máquinas diferentes [49; 48; 29], uma vez que os componentes e procedimentos envolvidos nesses estados são padronizados por ACPI. No entanto, realizamos uma análise de sensibilidade (apresentada no próximo capítulo) a fim de avaliar o impacto da variação desses valores no desempenho e na ordem de eficiência das estratégias propostas. Essas variações podem ocorrer por mudanças em tecnologia do hardware, por exemplo.

5.4 Rastros de Submissão de Aplicações e de Mudança na Disponibilidade das Máquinas

Como apresentado anteriormente, nosso modelo de simulação utiliza um rastro de submissão de aplicações e outro de variação na disponibilidade das máquinas para a grade. Um exemplo desse rastro é apresentado na Figura 5.3(a), na página 39. Nas simulações, utilizamos um rastro da grade computacional entre-pares OurGrid [50] que descreve a variação na disponibilidade de 200 máquinas em um período contínuo de 7 dias, que inicia na quarta-feira dia 1 de abril de 2009. Esse rastro descreve as sessões de disponibilidade das máquinas para a grade. A Figura 5.7 apresenta o número de máquinas disponíveis para a grade ao longo do tempo, seguindo esse rastro de disponibilidade.

Já os rastros de submissão de aplicações foram obtidos de um modelo analítico de aplicações do tipo saco-de-tarefas proposto por Iosup et al. [33]. Por meio desse modelo, geramos rastros que simulam a submissão de aplicações à grade. Um exemplo desse rastro é apresentado na Figura 5.3(b), na página 39. Utilizamos o modelo proposto por Iosup et al. porque

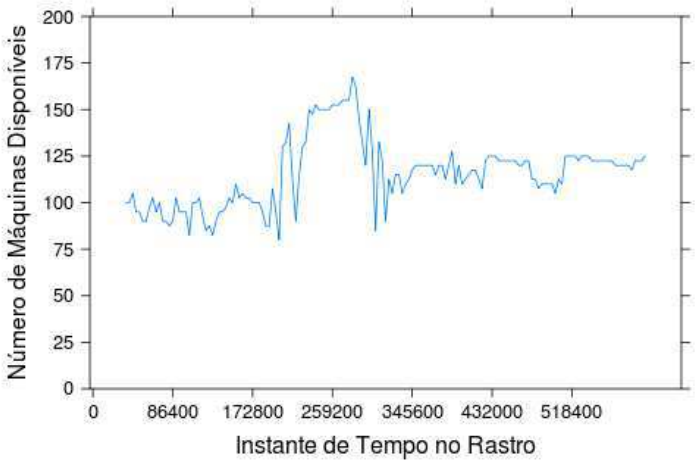


Figura 5.7. Número de máquinas disponíveis para a grade segundo o rastro de disponibilidade da grade computacional OurGrid. Dados obtidos em observações realizadas a cada 30 **minutos.**

esse modelo considera as características internas das aplicações, como: o tempo de submissão da aplicação, o número de tarefas que a compõe e uma estimativa do tempo de execução de cada tarefa. Nesse modelo, o intervalo entre submissão de aplicações é ponderado pelo ciclo diário de submissões. O ciclo diário retrata como as submissões de aplicações são distribuídas ao longo do dia, por exemplo: o intervalo entre submissão de aplicações tende a ser menor por volta das 12 horas do que às 20 horas [33]. Utilizamos aplicações submetidas ao longo de dois dias e as tarefas cujo tempo de execução não ultrapassavam 35 minutos, tempo máximo de tarefas comuns em grades computacionais oportunistas [39].

Consideramos que as estimativas do tempo de execução das tarefas contidas no rastro equivalem ao tempo que a tarefa gastaria para executar na máquina mais rápida (3 GHz) do conjunto de máquinas utilizado nos experimentos. Isso ocasiona que o tempo que as tarefas gastam para executar na simulação é maior ou igual ao tempo estimado no rastro. Essa escolha tende a reduzir os ciclos de ociosidade. Caso escolhêssemos a máquina mais lenta como a máquina de referência, as tarefas tenderiam a executar mais rápido do que no rastro o que em alguns casos pode ocasionar aumento do tamanho dos períodos em que não há tarefas da grade para serem executadas.

Também avaliamos as estratégias usando outros rastros de submissão de aplicações e de variação na disponibilidade dos recursos. Os resultados obtidos foram semelhantes aos

que apresentaremos no próximo capítulo desta dissertação, portanto, optamos por apresentar apenas os resultados obtidos com os rastros apresentados nesta seção. Os resultados obtidos com os outros rastros podem ser vistos nas referências [56; 54; 55].

5.5 Cenários de Avaliação

As variáveis envolvidas nos experimentos desta dissertação podem ser classificadas em independentes e dependentes. As variáveis independentes são as variáveis de controle, os fatores cujos valores (níveis) podem ser alterados e os impactos nas variáveis dependentes podem ser medidos. As variáveis independentes são as estratégias de dormência, estratégia de escolha de recursos, TI, tamanho da grade e os rastros de demanda, que são utilizados como fator aleatório para medir o erro estatístico dos resultados. As variáveis dependentes são a economia de energia, o atraso e o número de transições. Além das variáveis dependentes e independentes são utilizadas duas constantes: um rastro que descreve a variação na disponibilidade das máquinas para a grade e uma distribuição empírica que descreve as características das máquinas. Essas constantes são usadas para caracterizar o ambiente de uma grade computacional entre-pares, foge ao escopo desta dissertação avaliar suas variações.

A Tabela 5.1 apresenta os níveis das variáveis independentes. Esses níveis consistem nos possíveis valores para as estratégias propostas e/ou valores obtidos da literatura.

Tabela 5.1. Resumo dos Fatores e Níveis utilizados nos experimentos

Variáveis Independentes (Fatores)	**Níveis**
Estado ocioso ou de dormência	ocioso, Hibernação, Sobreaviso
Escolha de Recursos	Aleatório, EA, LRS, MRS
TI	de 300 em 300, de 0 a 1.800
Número de máquinas em cada *peer*	de 20 em 20, de 20 a 200

Fixamos o número de *peers* na grade em 3. O número de máquinas em cada *peer* varia como apresentado na Tabela 5.1. Existem *brokers* em cada *peer* que submetem aplicações para serem executadas na grade. Deste modo, ao longo da simulação os *brokers* concorrem pelos recursos disponíveis na grade. Utilizamos 30 rastros com dados de 2 dias de sub-

missões de aplicações, esses rastros foram usados como entrada nas simulações. Os rastros indicam a variável aleatória nas simulações; utilizamos 30 rastros com o objetivo de obter um erro estatístico com um nível de confiança de 95%.

Usamos o mesmo conjunto de rastro de submissão de aplicações nas simulações realizadas com cada número de máquinas da grade. Isso permite simular diferentes cenários de contenção por recursos, uma vez que aumentamos a oferta de máquinas e mantemos constante a demanda. Uma simulação só termina quando o rastro de submissão de aplicações de todos os *brokers* termina. Se as simulações durarem mais de uma semana, o rastro de variação na disponibilidade de recursos é utilizado de forma circular.

A estratégia de dormência, de escolha de recurso e os valores de TI utilizados são dados pela configuração do cenário que está sendo executado, mas de um modo geral todos os *peers* utilizam a mesma configuração, i.e., quando se está avaliando a estratégia de dormência Sobreaviso, todo os domínios administrativos da grade utilizam essa estratégia. A seguir apresentamos as configurações avaliadas:

1. **Avaliação das Estratégias de Dormência.** Nessa configuração utilizamos as estratégias de dormência Sobreaviso e Hibernação. Utilizamos a estratégia de escolha MRS e TI definido com o valor 0;

2. **Avaliação das Estratégias de Escolha de Recursos.** Nessa configuração utilizamos TI definido com o valor 0 e as estratégia de dormência como Sobreaviso e Hibernação. A estratégia de escolha de recursos varia nos níveis indicados na Tabela 5.1;

3. **Avaliação dos valores de TI.** Nessa configuração utilizamos a estratégia de dormência definida como Sobreaviso e Hibernação e a estratégia de escolha como MRS. O valor de TI varia nos níveis indicados na Tabela 5.1.

Projetamos uma análise de sensibilidade para verificar o impacto que as variações nos valores da potência e da latência dos estados de dormência têm nos resultados. Uma análise de sensibilidade é um experimento que procura estabelecer o impacto da alteração dos valores de alguns parâmetros nos resultados do experimento. Nessa análise, consideramos que as potências e as latências dos estados de dormência podem variar em proporções de -100% a 200% em relação aos valores de referência. Utilizamos os seguintes valores de referência:

para Sobreaviso $P_s = 3,33$ Watts e $L_s = 2,5$ Watts; para Hibernação $P_h = 0,7$ segundos e $L_h = 55$ segundos. Nessa avaliação, consideramos os seguintes cenários de variação da potência e da latência dos estados Sobreaviso e Hibernação.

1. Variações da latência com a potência fixa;
2. Variações da potência com a latência fixa;
3. Latência e potência variam na mesma proporção;
4. Latência e potência variam em proporção inversa;

Na análise de sensibilidade, consideramos a grade computacional em média e baixa contenção, utilizando, respectivamente, 80 e 220 máquinas em cada domínio administrativo. Avaliamos a economia de energia e o atraso gerado no tempo de resposta das aplicações.

5.6 Considerações Finais

Neste capítulo, apresentamos os materiais e métodos utilizados na avaliação das estratégias de economia de energia propostas nesta dissertação. Dado o objetivo do estudo, decidimos avaliar o desempenho das estratégias pelas seguintes métricas: economia de energia provida na infraestrutura, atraso gerado no tempo de resposta da aplicação e número de transições realizadas pelas máquinas.

Para realizar a avaliação, propomos um modelo simulado. O modelo baseia-se na grade computacional entre-pares OurGrid e captura os comportamentos relevantes às métricas em análise. O simulador foi implementado em linguagem de programação Python versão 2.5. Verificamos a implementação do modelo por meio de testes de unidades e de aceitação. Para os casos de teste avaliados a implementação apresentou o comportamento esperado. O modelo de simulação é guiado por rastro, o que permite que cargas de trabalho reais sejam utilizadas como entrada em experimentos.

Nesse sentido, utilizamos rastros que descrevem a submissão de aplicações do tipo saco-de-tarefas e rastros que descrevem a disponibilidade de máquinas para a grade. Para configurar os recursos da grade computacional utilizamos informações do consumo de energia

e poder de processamento de máquinas existentes em um ambiente de grade computacional entre-pares em produção.

O simulador, os rastros, os arquivos de configuração, os *scripts* utilizados para gerar os gráficos e os demais materiais utilizados na avaliação das estratégias encontram-se disponíveis em `http://redmine.lsd.ufcg.edu.br/projects/list_files/green-grid`. Os resultados obtidos na avaliação são apresentados e analisados no próximo capítulo desta dissertação.

Capítulo 6

Apresentação e Análise dos Resultados

Neste capítulo apresentamos os resultados da avaliação de desempenho das heurísticas discutidas no Capítulo 4. Apresentamos os resultados da economia de energia na infraestrutura, atraso no tempo de resposta das aplicações e número de transições realizadas pelos recursos. Os resultados são apresentados do ponto de vista de um domínio administrativo da grade.

6.1 Economia de Energia na Infraestrutura

Nesta seção apresentamos os resultados da economia de energia provida pelas estratégias. A Figura 6.1 mostra um gráfico com a economia de energia provida pelas estratégias de dormência Hibernação e Sobreaviso. Nesse gráfico o eixo das ordenadas indica o percentual de economia em relação ao estado ocioso e o eixo das abscissas indica o número de máquinas agregadas por cada domínio administrativo. Cada ponto no plano cartesiano é uma média da energia economizada pelo conjunto de máquinas em 30 simulações. Em cada simulação usamos um rastro de submissão de aplicações diferente, resultando em 30 rastros. Usamos o mesmo conjunto de rastro de submissão de aplicações nas simulações realizadas com cada número de máquinas indicado no eixo das abscissas. Isso permite simular diferentes cenários de contenção por recursos, uma vez que variamos a oferta de máquinas e mantemos constante a demanda indicada nos rastros.

Deste modo, o resultado apresentado na Figura 6.1 indica que tanto o uso da estratégia Sobreaviso como da estratégia Hibernação permite grande economia de energia na infraestrutura. Obteve-se uma economia de energia de $65,53\%$ no cenário em que a grade é

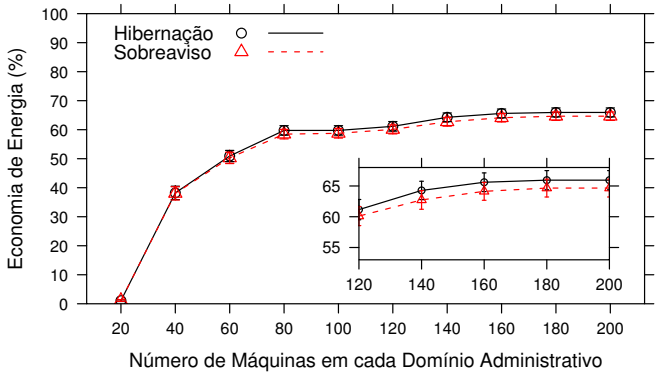


Figura 6.1. Percentual de energia economizada na grade computacional quando são usadas as estratégias Sobreaviso e Hibernação em relação a manter os recursos ociosos. A estratégia de escolha é definida como MRS e TI como 0.

submetida a uma baixa contenção (200 máquinas). Entretanto, essa economia varia significativamente com a contenção de recursos. Observamos que quando a contenção de recursos é alta (número de máquinas menor que 60) as máquinas permanecem executando tarefas da grade durante a maior parte do tempo da sessão de disponibilidade, existindo poucos momentos para que as máquinas sejam adormecidas. Além disso, quando as máquinas tornam-se ociosas e são adormecidas, elas permanecem menos tempo no estado de dormência do que permaneceriam em baixa contenção, como mostra a Figura 6.2. Como discutimos na Seção 4.1 a economia de energia obtida por uma máquina em um estado de dormência aumenta de forma linear com o tempo em que a máquina permanece adormecida.

A Figura 6.2 mostra também que a partir de aproximadamente 140 máquinas, aumentar o número de máquinas (eixo das abscissas) não gera aumento significativo no tamanho da sessão de dormência. Isso implica também em não se obter aumento significativo na economia de energia, como vimos na Figura 6.1. A razão desse comportamento é que, para essa quantidade de máquinas, o tamanho das sessões de dormência é próximo ao tamanho das sessões de disponibilidade de modo que, geralmente, as máquinas não permanecem mais que 8.000 segundos (pouco mais que 2 horas) adormecidas. A economia de energia obtida é limitada a esse tamanho de sessão de dormência. Como identificamos de forma analítica, o tamanho das sessões de dormência de uma máquina usada de modo oportunista é limitado ao tamanho das sessões de disponibilidade.

No que se refere ao impacto que o uso de TI causa na economia de energia da infraestru-

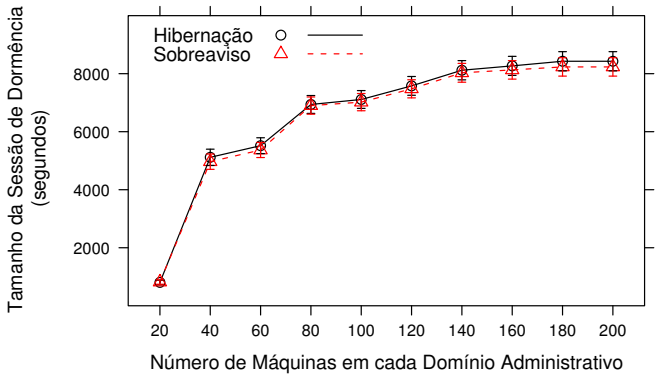
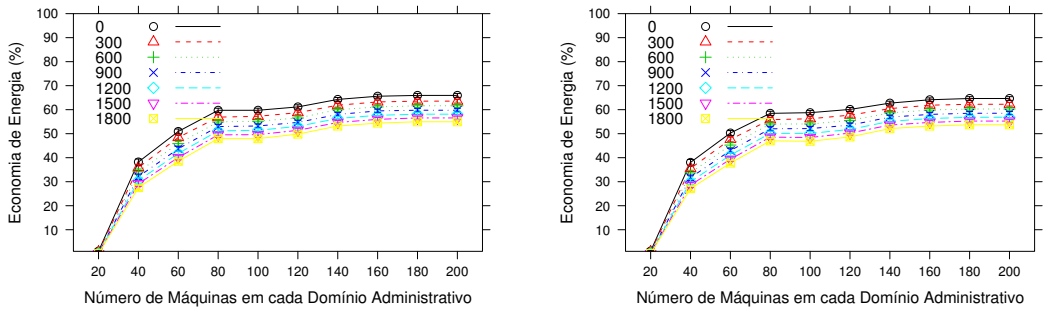


Figura 6.2. Tamanho médio das sessões de dormência dos recursos em uma grade computacional entre-pares que faz uso das estratégia Sobreaviso e Hibernação. A estratégia de escolha é definida como MRS e TI como 0.

tura, a Figura 6.3 mostra que aumentar o valor de TI causa redução na economia de energia. Quando não se faz uso de TI ($TI = 0$) a máquina é adormecida tão logo se torna inativa. O uso dessa configuração permite $65,53\%$ de economia de energia na infraestrutura quando a grade se encontra em baixa contenção. De outro modo, quando se utiliza TI definido como 1.800 segundos, a economia de energia reduz para $55,81\%$. A razão desse resultado é que durante o período de temporização as máquinas são mantidas no estado ocioso e esse estado apresenta maior consumo de energia do que o estado de dormência. Em outras palavras, o uso de TI reduz o tamanho da sessão de dormência, causando redução da economia de energia. Naturalmente, os valores de TI têm impacto semelhante na economia de energia provida pelas estratégias de dormência Sobreaviso e Hibernação.

Os resultados da avaliação do impacto das estratégias de escolha de recursos MRS, LRS, EA e Aleatória na economia de energia provida pelas estratégias Sobreaviso e Hibernação são apresentados na Figura 6.4. Esses resultados mostram que não há diferença significativa na economia de energia gerada pelo uso das estratégias de escolha quando a grade se encontra em alta contenção. Geralmente, quando surge uma demanda e a grade se encontra em alta contenção, ou as máquinas se encontram acordadas executando outras tarefas da grade ou a maior parte das máquinas que se encontram adormecidas precisam ser acordadas. Não há impacto o critério usado para escolher os recursos a serem acordadas quando todos os recursos precisarem ser acordados.

Na Figura 6.4 ampliamos a parte dos gráficos em que o número de máquinas em cada



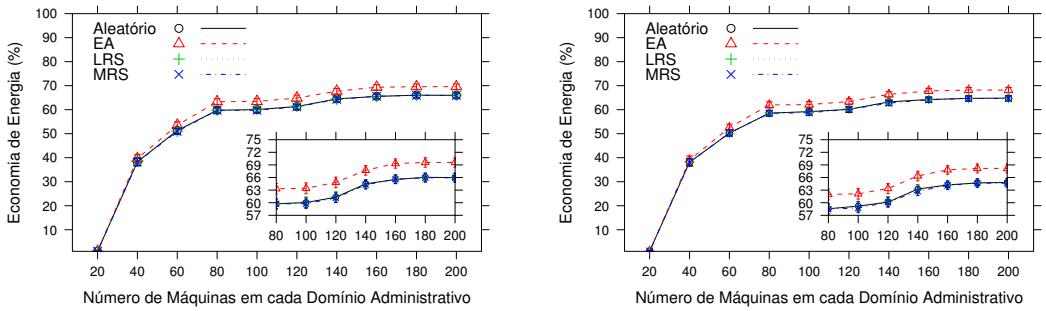
(a) Hibernação (b) Sobreaviso

Figura 6.3. Economia de energia provida pelos valores de TI usados em conjunto com as estratégias Sobreaviso e Hibernação em uma grade computacional entre-pares sujeita a diferentes níveis de contenção por recursos. A estratégia de escolha é definida como MRS.

domínio administrativo é entre 80 e 200. Isso permite destacar o desempenho das estratégias de escolha no cenário de baixa contenção. Podemos observar que nesse cenário a estratégia EA apresenta uma economia de energia 3% maior que as demais estratégias avaliadas. Essa maior economia da estratégia EA se dá pela maior quantidade e variedade de máquinas adormecidas quando a grade se encontra em baixa contenção, o que permite que as máquinas mais eficientes no aspecto energético sejam escolhidas. Observamos que as estratégias MRS e LRS, que se baseiam no tempo em que a máquina foi adormecida, apresentam economia de energia equivalente à estratégia de escolha Aleatória. Esse resultado foi observado em todos os cenários, portanto, independe da contenção da grade e da estratégia de dormência utilizada.

6.2 Tempo de Resposta das Aplicações

O propósito desta seção é apresentar e analisar o impacto no tempo de resposta das aplicações gerado pelo uso das estratégias de economia de energia. Nesse sentido, a Figura 6.5 apresenta um gráfico com o atraso no tempo de resposta das aplicações gerado pelas estratégias de dormência Sobreaviso e Hibernação. Nesse gráfico, como nos demais apresentados nesta seção, o eixo das ordenadas indica o percentual de atraso gerado com o uso das estratégias de economia de energia. O eixo das abscissas, por sua vez, indica o número de máquinas



(a) Hibernação (b) Sobreaviso

Figura 6.4. Economia de energia provida na infraestrutura pelas estratégias de escolha de recursos usadas em conjunto com as estratégias de dormência Sobreaviso e Hibernação e TI definido como 0.

agregadas por cada domínio administrativo. Cada ponto no plano cartesiano é uma média do atraso gerado no tempo de resposta das aplicações em 30 simulações, como descrito no Capítulo 5.

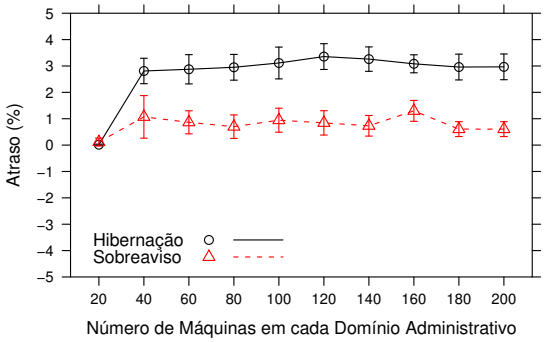
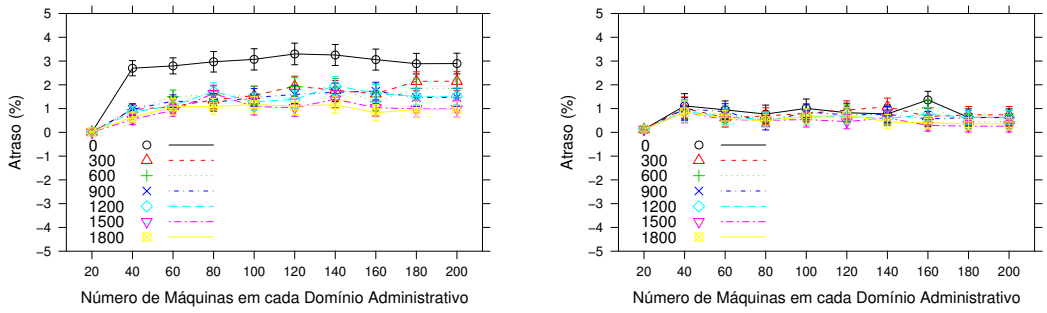


Figura 6.5. Atraso no tempo de resposta das aplicações gerado pelo uso das estratégias de dormência. Os domínios administrativos utilizam a estratégia de escolha MRS e TI definido como 0.

A Figura 6.5 mostra que o uso das estratégias de dormência Hibernação e Sobreaviso gera um atraso no tempo de resposta das aplicações. Contudo, a estratégia Hibernação gera maior atraso do que Sobreaviso na maior parte dos cenários de contenção por recursos. Podemos observar que a estratégia Hibernação ocasiona um aumento de até $3,8\%$ no tempo de resposta das aplicações e Sobreaviso apresenta um aumento de até 2%, considerando os limites superiores dos intervalos de confiança. O impacto que as estratégias geram no tempo de res-

posta 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 pequeno comparado ao tempo de execução de tarefas de aplicações do tipo saco-de-tarefas em grades computacionais oportunistas, que pode chegar a aproximadamente 35 minutos [39].

A Figura 6.6 mostra o impacto dos valores de TI no atraso gerado pelas estratégias Sobreaviso e Hibernação no tempo de resposta das aplicações. Podemos observar que o não uso de TI gera um atraso de até $3,8\%$ no tempo de resposta das aplicações quando utilizado em conjunto com a estratégia de dormência Hibernação. O que justifica esse baixo percentual de atraso é que, no pior caso, a tarefa aguarda no máximo 110 segundos até que a máquina seja acordada, i.e., o tempo necessário para que a máquina seja adormecida e acordada em seguida. Como discutimos anteriormente, esse tempo é significativamente menor que o tempo de execução de aplicações do tipo saco-de-tarefas comuns em grades computacionais oportunistas.



(a) Hibernação (b) Sobreaviso

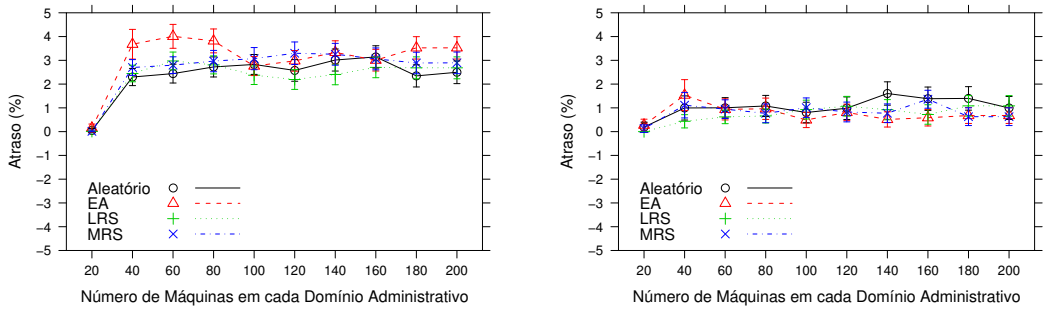
Figura 6.6. Atraso no tempo de resposta das aplicações gerado por diferentes valores de TI em conjunto com as estratégias de dormência Sobreaviso e Hibernação. Os domínios administrativos utilizam a estratégia de escolha MRS.

Observamos também que, quando TI é definido como 300 segundos, o atraso gerado no tempo de resposta das aplicações é inferior a 3%, mesmo com o uso da estratégia Hibernação. Esse atraso é equivalente ao observado quando TI é definido como 1.800 segundos. Isso significa que manter a máquina ociosa aguardando a chegada de uma nova tarefa por 300 segundos é mais vantajoso em termos de aumento no tempo de resposta do que adormecê-la tão logo se torne ociosa (TI$=0$), mas é equivalente a mantê-la aguardando por mais tempo. Em outras palavras, a partir de 300 segundos o aumento do valor de TI não gera

maior redução do atraso no tempo de resposta. Entretanto, como mostramos na Seção 6.1, aumentar o valor de TI reduz a economia de energia.

Quando se utiliza TI em conjunto com a estratégia de dormência Sobreaviso, o impacto gerado pelos valores de TI no tempo de resposta das aplicações é desprezível. Esse atraso é inferior a 2%. Portanto, com o uso da estratégia Sobreaviso, diferentemente do que ocorre com o uso da estratégia Hibernação, o não uso de TI não aumenta o atraso gerado no tempo de resposta das aplicações quando comparado aos atrasos gerados pelos outros valores de TI avaliados. Isso é explicado pelo pequeno tempo de transição da estratégia sobreaviso, que é muito próximo a responder instantaneamente à requisição.

Por fim, avaliamos o atraso que as estratégias de escolha de recursos MRS, LRS, EA e a escolha Aleatória geram no tempo de resposta das aplicações quando utilizadas em conjunto com as estratégias de dormência Sobreaviso e Hibernação. Os resultados são apresentados na Figura 6.7. Esses resultados mostram que os atrasos gerados pelas estratégias de escolha variam consideravelmente com a contenção de recursos, principalmente o atraso gerado pela estratégia EA. Não se pode dizer que uma estratégia apresente menor atraso que as demais sem levar em conta um nível específico de contenção da grade.



(a) Hibernação (b) Sobreaviso

Figura 6.7. Atraso no tempo de resposta das aplicações gerado por diferentes estratégias de escolha de recursos utilizadas em conjunto com as estratégias Sobreaviso e Hibernação. Os domínios administrativos utilizam TI definido como 0.

6.3 Número de Transições

Nesta seção apresentamos e analisamos o número de transições realizadas pelas máquinas da grade computacional quando são utilizadas as estratégias economia de energia. Essa análise permite identificar o impacto que as estratégias podem gerar na depreciação dos discos rígidos que compõem a grade, além de mostrar os cenários em que as estratégias são usadas. Outros trabalhos na literatura têm realizado esse tipo de análise em cenários diferentes de grades computacionais, como no de Kaushik et al. [36], que mostra que os recursos realizam 7 transições por dia e no de Reich et al. [59] em que há casos em que os recursos realizam até 35 transições por dia.

A Figura 6.8 apresenta um gráfico com o número de transições realizadas pelas máquinas quando são utilizadas as estratégias de dormência Sobreaviso e Hibernação. Nesse gráfico, como nos demais apresentados nesta seção, o eixo das ordenadas indica o número de transições realizadas pelas máquinas. O eixo das abscissas, por sua vez, indica o número de máquinas agregadas por cada domínio administrativo. Cada ponto no plano cartesiano é uma média do número de transições realizadas por cada máquina em 30 simulações, como descrito no Capítulo 5.

Podemos observar, pela Figura 6.8, que não há evidências estatísticas de que as estratégias de dormência Sobreaviso e Hibernação apresentem impactos diferentes no número de transições realizadas pelas máquinas. Observamos também que impacto no número de transições é gerado principalmente pela contenção da grade. São realizadas poucas transições quando a grade experimenta uma baixa contenção uma vez que existem poucos períodos de ociosidade para que as máquinas sejam adormecidas. Surgem períodos de ociosidade à medida que a contenção reduz, permitindo que as máquinas realizem mais transições para os estados de dormência. Quando a grade experimenta uma média contenção, a maior parte das máquinas precisa ser acordada quando surge uma demanda, mas quando a contenção é baixa apenas algumas máquinas precisam ser acordadas sempre que surge uma demanda. Isso faz com que a média do número de transições realizadas pelas máquinas diminua.

Avaliamos impacto da variação dos valores de TI no número médio de transições entre os estados de dormência Sobreaviso e Hibernação e os outros estados em que as máquinas da grade podem operar. Os resultados dessa avaliação são apresentados na Figura 6.9. Esse resultado mostra que aumentar o valor de TI reduz o número de transições realizadas pelas

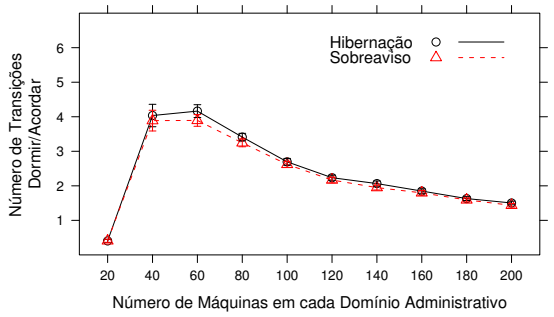
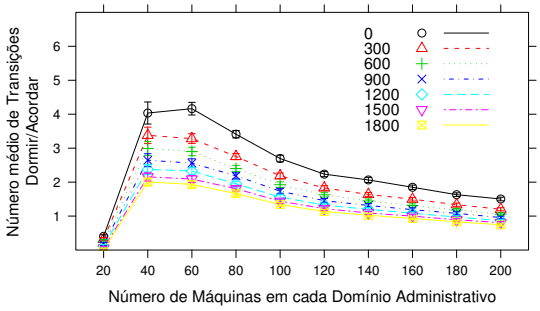


Figura 6.8. Número de transições dormir/acordar realizadas por cada máquina quando são utilizadas as estratégias de dormência Sobreaviso e Hibernação. Os domínios administrativos utilizam a estratégia de escolha MRS e TI definido como 0**.**

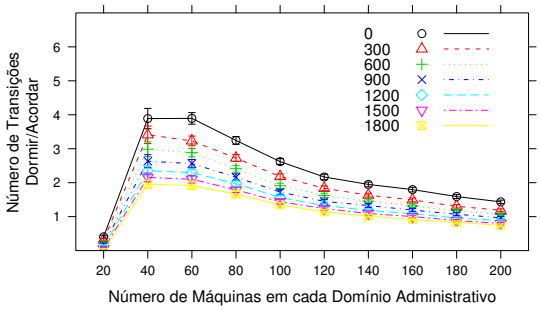
máquinas. Isso ocorre porque, sem o uso de TI sempre que a máquina se torna inativa ela faz uma transição para um estado de dormência. De outro modo, quando se utiliza TI, a máquina é mantida ociosa aguardando antes de ser adormecida, isso possibilita que uma tarefa seja submetida e a máquina seja alocada e, portanto, não transite para um estado de dormência. Deste modo, quanto maior esse tempo de espera menor é o número de transições realizadas pelas máquinas.

Avaliamos, também, o impacto das estratégias de escolha de recursos no número de transições realizadas pelas máquinas. Os resultados são apresentados na Figura 6.10. Esses resultados indicam que as estratégias impactam de modo equivalente no número de transições realizadas pelas máquinas.

É importante notar que simulamos a contenção por recursos mantendo fixos os rastros de 2 dias de submissão de aplicações e variando a oferta de recursos na grade. Como as simulações terminam quando os rastros de submissão de aplicações terminam, as simulações se referem a diferentes intervalos de tempo com cada número de máquinas na grade. Obtivemos o maior número de transições no cenário de média contenção (60 máquinas em cada domínio administrativo). Nesse cenário, as simulações se referem a um tempo médio de 1 dia, 1 horas e 32 minutos, por simplicidade seremos pessimistas e consideraremos esse período como referente a um dia. Com o uso da estratégia de dormência Hibernação, as máquinas realizam em média $4,09$ transições, com um erro estatístico de $\pm 0,36\%$ (como pode ser visto na Figura 6.8).



(a) Hibernação

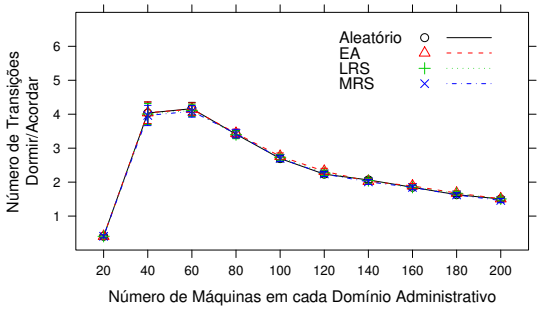


(b) Sobreaviso

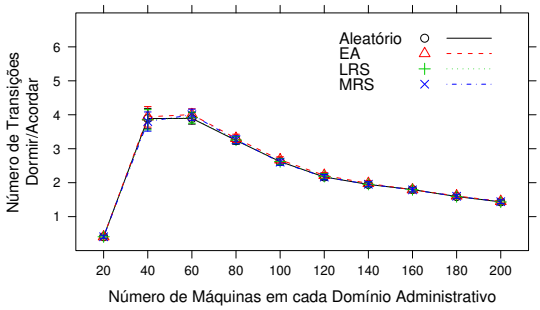
Figura 6.9. Número de transições dormir/acordar realizadas por cada máquina da grade computacional entre-pares quando se utiliza diferentes valores de TI. Os domínios administrativos utilizam a estratégia de escolha MRS.

Podemos avaliar esse número de transições realizadas pelo uso das estratégias de dormência de duas formas. Primeiro comparamos com um usuário que manteria a máquina sempre ociosa quando não a estivesse usando, i.e., um usuário que não faria uso de uma estratégia de dormência. Em situações normais a máquina desse usuário não realizaria transições. Neste caso, se esse usuário disponibilizar sua máquina para a grade a mesma passaria a realizar 4 transições diárias. Entretanto, esse número de transições equivale a, apenas, $16,67\%$ da média de 27 transições diárias estimada pelos fabricantes.

Uma segunda análise que pode ser feita é comparar o número de transições realizadas pela máquina quando disponibilizada para a grade com o número de transições que seriam realizadas se o usuário fizesse uso de uma estratégia de dormência nos períodos em que não a estivesse usando. Nesse caso, podemos considerar que a cada início e final de uma sessão de disponibilidade haveria uma transição de entrada e uma de saída de um estado de



(a) Hibernação



(b) Sobreaviso

Figura 6.10. Número de transições dormir/acordar realizadas por cada máquina quando se utiliza diferentes estratégias de escolha de recursos. Os domínios administrativos utilizam TI definido como 0.

dormência, respectivamente. A Figura 6.11 exemplifica essa situação mostrando uma função de distribuição acumulada do número de transições que seriam realizadas por 60 máquinas da grade ao longo de um dia. Nota-se que pouco mais de 60% das máquinas realizam no mínimo 4 transições diárias.

Deste modo, queremos identificar se o número de transições realizadas pela máquina quando a estratégia de dormência é usada pela grade é maior do que quando usadas pelo usuário local, nos períodos em que a máquina é compartilhada na grade. Para atingir esse objetivo utilizamos 60 máquinas escolhidas de modo aleatório no rastro de uma semana de variação na disponibilidade das máquinas utilizado nas simulações. Calculamos a média de transições que seriam realizadas pelas máquinas considerando uma transição a cada início de sessão de disponibilidade e uma transição a cada final de sessão de disponibilidade. Obtivemos que as máquinas realizariam em média $4,98$ transições diárias, com um erro estatístico

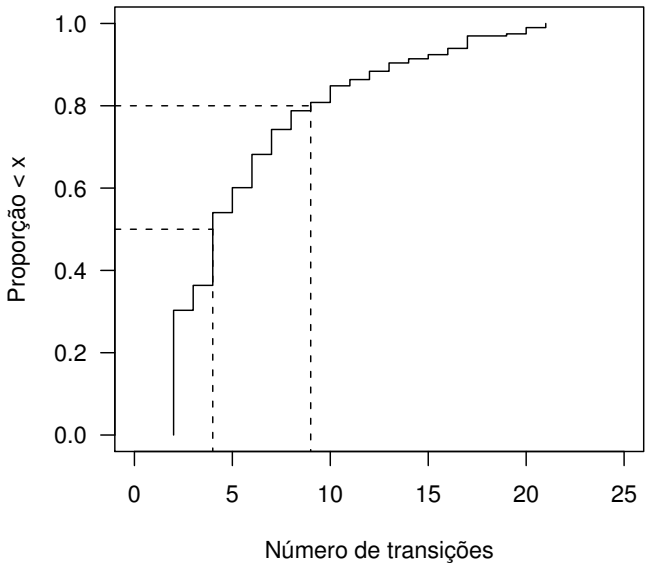


Figura 6.11. Número de transições diárias realizadas por 60 máquinas quando uma estratégia de dormência é utilizada nas sessões de disponibilidade.

de $\pm 0,23$ para um nível de confiança de 95%. Isso indica que, para os cenários que avaliamos, o uso das estratégias de dormência pela grade reduziria o número de transições da máquina. Essa redução ocorre porque, quando as estratégias são utilizadas pela grade, acontecem sessões de disponibilidade em que as máquinas permanecem toda a sessão executando tarefas da grade e, portanto, não são adormecidas.

6.4 Análise de Sensibilidade

Na Seção 6.1 identificamos que as estratégias de dormência Sobreaviso e Hibernação são responsáveis por 65.53% da economia de energia obtida na infraestrutura do domínio administrativo. Além disso, na Seção 6.2, mostramos que a latência desses estados impactou o atraso gerado no tempo de resposta das aplicações. Naquela avaliação usamos valores obtidos na literatura como referência para os dois principais parâmetros dessas estratégias: a potência e latência. Faz-se necessário identificar como variações nesses valores impactam os resultados obtidos. Deste modo, esta seção tem por objetivo apresentar uma análise da sensibilidade que visa estabelecer o impacto da variação dos valores da potência e da latên-

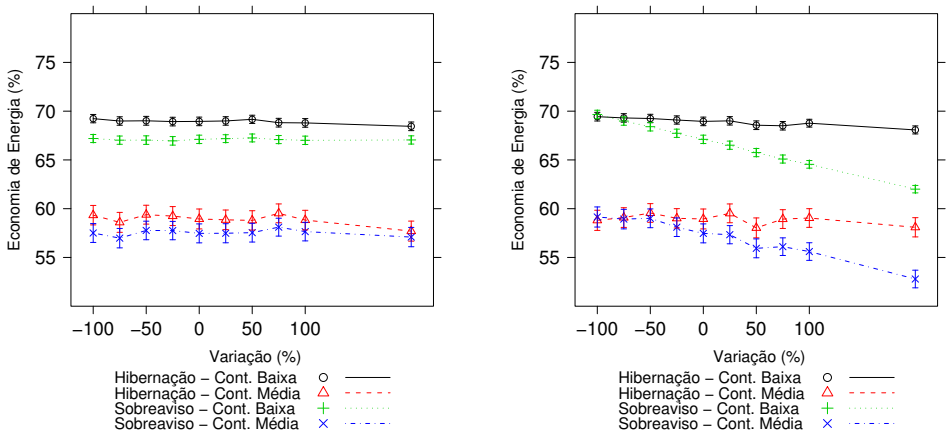
cia das estratégias Sobreaviso e Hibernação na economia de energia e no atraso no tempo de resposta das aplicações em uma grade computacional entre-pares.

Como definimos na Seção 5.5, neste experimento, consideramos que as potências e as latências das estratégias de dormência podem variar em proporções de -100% a 200% em relação aos valores de referência. Os valores de referência são potência de $3,33$ Watts e latência de $2,5$ segundos, para a estratégia Sobreaviso, e potência de $0,7$ Watts e latência de 55 segundos para a estratégia Hibernação. Apresentamos resultados para dois cenários de contenção de recursos que chamamos de média contenção (80 máquinas) e baixa contenção (220 máquinas).

6.4.1 Economia de Energia na Infraestrutura

Em todas as figuras apresentadas nesta seção o eixo das abscissas indica o percentual de variação em relação aos valores de referência, portanto, a variação é de 0% quando são usados os valores de referência. O eixo das ordenadas indica a economia de energia gerada pela configuração. Deste modo, os resultados apresentados na Figura 6.12 mostram a economia de energia no cenário em que a potência permanece constante e a latência varia e, também, que a latência varia e a potência permanece constante. Os resultados indicam que variações na latência não geram impacto significativo na economia de energia. Contudo, a redução da potência aumenta a economia de energia provida pela estratégia Sobreaviso e o aumento reduz essa economia. Os valores das economias de energia providas pelas estratégias se aproximam à medida que a potência reduz. Isso ocorre tanto em média quanto em baixa contenção.

Os resultados apresentados na Figura 6.13 mostram a economia de energia nos cenários em que os valores da potência e da latência variam nas mesmas proporções e em que variam em proporções inversas. O resultado apresentado na Figura 6.13(a) indica que reduções nos valores da potência e da latência aumentam a economia de energia e o aumento desses valores reduz essa economia. De outro modo, no cenário em que a latência e a potência apresentam crescimentos inversos (Figura 6.13(b)), com o uso da estratégia Sobreaviso, tem-se um aumento da economia de energia quando há redução da potência. Esse aumento ocorre mesmo com o aumento do valor da latência. Isso indica que a potência tem maior impacto na economia de energia do que a latência quando se analisa a estratégia de dormência Sobre-



(a) Potência fixa no valor de referência e a latência varia

(b) Latência fixa no valor de referência e a potência varia

Figura 6.12. Sensibilidade da economia de energia às variações na potência ou na latência. Variações ocorrem entre -100% e 200% em relação aos valores de referência. A variação de 0% refere-se ao uso dos valores de referência.

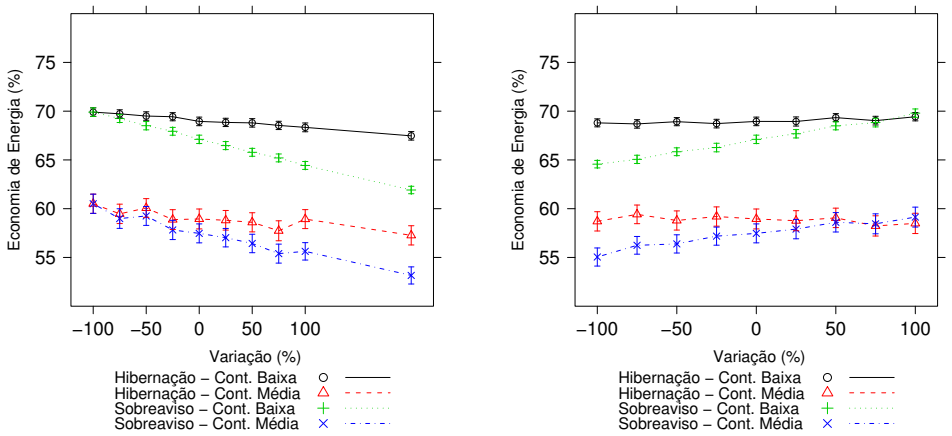
aviso. Com a estratégia de dormência Hibernação, por sua vez, a variação da potência e da latência em proporções inversas gera um impacto desprezível na economia de energia.

Os resultados apresentados nesta seção indicam que as variações na potência e na latência podem gerar uma diferença de até $6,8\%$ na economia de energia provida pelas estratégias Sobreaviso e Hibernação. Em todos os cenários avaliados Hibernação apresentou economia de energia igual ou superior a Sobreaviso. Portanto, não identificamos uma mudança na ordem de eficiência das estratégias.

6.4.2 Tempo de Resposta das Aplicações

Nesta seção apresentamos os resultados da avaliação da sensibilidade do tempo resposta às variações na potência e na latência dos estados Sobreaviso e Hibernação. Em todas as figuras apresentadas nesta seção, semelhante às figuras da economia de energia apresentadas na seção anterior, o eixo das abscissas indica o percentual de variação em relação aos valores de referência e o eixo das ordenadas indica o atraso gerado no tempo de resposta das aplicações.

Os resultados apresentados na Figura 6.14 mostram o atraso no cenário em que o valor da potência permanece constante e o valor latência varia e em que o valor da latência permanece constante e o valor da potência varia. A Figura 6.14(b) mostra que as variações na



(a) Latência e potência variam nas mesmas proporções

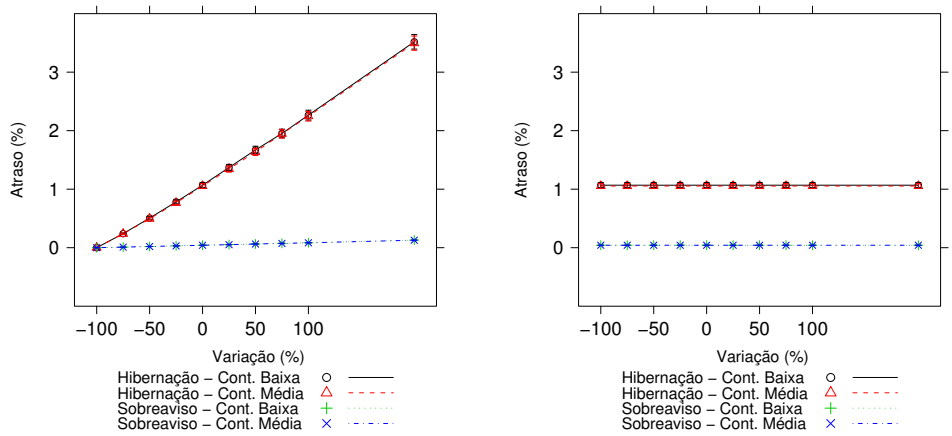
(b) Latência e potência variam em proporções inversas. A latência segue o eixo das abscissas.

Figura 6.13. Variação em conjunto da potência e da latência. Nas mesmas proporções (a) e em proporções inversas (b)

potência das estratégias de dormência não impactam o atraso gerado por essas estratégias. De outro modo, a Figura 6.14(b) mostra que variações na latência geram impacto significativo. Quando há redução da latência em 100%, não há qualquer atraso no tempo de resposta da aplicação. Por outro lado, quando se aumenta a latência em 200% ocorre um atraso superior a 3% com o uso da estratégia de dormência Hibernação, independente da contenção da grade. O atraso gerado pela estratégia Sobreaviso é inferior ao atraso gerado pela estratégia Hibernação em todos os cenários. Além disso, com a estratégia Sobreaviso, o maior atraso observado foi de $0,09\%$, no cenário com um aumento de 200% na latência. O atraso gerado no tempo de resposta das aplicações nos cenários em que a potência e a latência variam nas mesmas proporções e em que variam em proporções inversas são os mesmos dos apresentados na Figura 6.14(a), uma vez que variações dos valores da potência não impactam a métrica atraso.

6.5 Considerações Finais

Nesta seção analisamos o desempenho das estratégias de economia de energia em grades computacionais entre-pares. Observamos que as estratégias de dormência Sobreaviso e Hibernação apresentam economia de energia equivalente. Contudo, a estratégia Sobreaviso



(a) Potência fixa no valor de referência e latência variável

(b) Latência fixa no valor de referência e potência variável.

Figura 6.14. Sensibilidade do tempo resposta às variações na potência e na latência dos estados Sobreaviso e Hibernação.

apresenta menor custo associado em termos do atraso gerado no tempo de resposta das aplicações. Quanto ao número de transições realizadas pelas máquinas, os resultados mostraram que o uso dessas estratégias pela grade apresenta menor número de transições do que as que seriam realizadas caso o usuário local utilizasse uma estratégia de dormência nos períodos em que a máquina é disponibilizada para a grade. Portanto, para esse usuário, disponibilizar a máquina para uma grade computacional entre-pares que faz uso de estratégias de dormência não aumenta a depreciação do recurso sob o ponto de vista das transições realizadas pelo disco rígido.

Encontramos que o uso de TI gera variações significativas em todas as métricas analisadas. Principalmente, TI definido como 0 possibilita maior economia de energia tanto quando utilizado em conjunto com a estratégia Hibernação como a estratégia Sobreaviso. No entanto, quando utilizado em conjunto com a estratégia Hibernação, esse valor de TI ocasiona maior atraso no tempo de resposta das aplicações.

Quanto às estratégias de escolha de recursos, mostramos que a estratégia de escolha de recursos EA resulta em maior economia de energia em cenários de baixa contenção de recursos. Essa economia está associada a um aumento no tempo de resposta das aplicações equivalente ao apresentado pelas demais estratégias.

Capítulo 7

Conclusão e Trabalhos Futuros

Nesta dissertação analisamos o consumo de energia em infraestruturas das grades computacionais entre-pares e avaliamos estratégias para reduzir esse consumo. Nesse sentido, usamos um modelo simulado para avaliar o impacto de estratégias de dormência, estratégias de escolha de recursos e diferentes valores de TI na economia de energia, no atraso no tempo de resposta das aplicações e no número de transições realizadas pelas máquinas em uma grade computacional entre-pares. Nossa simulação utilizou um modelo baseado em uma grade computacional entre-pares real e cargas de trabalhos que descrevem o comportamento de grades computacionais reais.

Avaliamos três conjuntos de estratégias: de dormência, valores de TI e escolha de recursos. Enfatizamos duas estratégia de dormência que podem ser implementadas em grades computacionais entre-pares: Sobreaviso e Hibernação. Em conjunto com essas estratégias de dormência, avaliamos o uso de valores de TI entre 0 e 1.800 segundos. Esses valores incluem TIs utilizados em outros trabalhos e/ou implementados em sistemas reais [29; 42; 59; 46; 13; 19]. Por fim, foram avaliadas quatro estratégias de escolha de recursos: EA que utiliza informações relativas às características energéticas das máquinas, LRS e MRS que consideram informações sobre o tempo em que as máquinas foram colocadas no estado de dormência e a escolha aleatória, que não considera qualquer informação sobre os recursos.

Os resultados obtidos mostraram as estratégias Sobreaviso e Hibernação podem reduzir o consumo de energia da infraestrutura em $65,53\%$ em cenários de baixa contenção de recursos. Encontramos que valores de TI maiores que 0 reduzem a economia de energia. Quando se utiliza TI definido como 1.800 segundos em um cenário de baixa contenção, tem-

se uma redução de aproximadamente 10% na economia de energia. Por fim, obtivemos que a estratégia de escolha de recursos EA pode aumentar a economia de energia provida pelas estratégias de dormência em até 3% em relação às demais estratégias de escolha de recursos.

Todas as estratégias avaliadas geraram impacto pequeno no tempo de resposta das aplicações. Esse impacto é representado por um atraso de até $3,8\%$ com o uso da estratégia Hibernação e de até 2% com o uso da estratégia Sobreaviso. Ambos os impactos podem ser reduzidos com o uso de um TI de 300 segundos, sob um custo de redução de aproximadamente 2% na economia de energia. O maior impacto gerado pelas estratégias de escolha de recursos foi observado com o uso da estratégia de escolha EA combinada com a estratégia de dormência Hibernação. Esse impacto representa um atraso de $4,5\%$ no tempo de resposta das aplicações.

Avaliamos a quantidade de transições (operações de partidas/paradas) realizadas pelos discos rígidos com o uso das estratégias de dormência. Os resultados mostraram que o uso dessas estratégias pela grade apresenta menor número de transições do que as que seriam realizadas caso o usuário local utilizasse uma estratégia de dormência nos períodos em que a máquina é disponibilizada para a grade. Portanto, para esse usuário, disponibilizar a máquina para uma grade computacional entre-pares que faz uso de estratégias de dormência não aumenta a depreciação do recurso sob o ponto de vista das transições realizadas pelo disco rígido.

Das estratégias avaliadas, destacamos as estratégias de dormência e TI. As estratégias de dormência porque apenas o seu uso foi responsável pela maior parte de economia de energia obtida na infraestrutura. TI porque o seu uso tem impacto significativo em todas as métricas. Esse impacto é importante porque, em situações em que o número de transições ou o atraso no tempo de resposta deve ser evitado, aumentar o valor de TI permite alcançar esses objetivos com um pequeno custo associado em termos da diminuição da economia de energia na infraestrutura.

Trabalhos futuros podem explorar modelos de predição para uma configuração de TI de modo dinâmico de acordo com a perspectiva do tempo de chegada de uma nova tarefa e a contenção da grade. Outro trabalho que pode ser realizado é investigar estratégias híbridas capazes de predizer o tamanho do período de inatividade a fim de decidir dinamicamente entre o uso da estratégia Sobreaviso ou da estratégia Hibernação, dado que mostramos que exis-

tem sessões de disponibilidade que permitem que a estratégia Hibernação apresente maior economia de energia que a estratégia Sobreaviso. Além disso, pode-se ampliar a avaliação que realizamos nesta dissertação para incluir outras cargas de trabalho, tipos e configurações de grades computacionais.

Bibliografia

[1] Advanced Micro Devices (AMD). Amd family 10h desktop processor. power and thermal data sheet, 2010. Disponível em: `http://support.amd.com/us/Processor_TechDocs/43375.pdf`. Último acesso em janeiro de 2011.

[2] Advanced Micro Devices, Inc. Magic packet technology, 1995. Disponível em: `http://support.amd.com/us/Embedded_TechDocs/20213.pdf`. Último acesso em janeiro de 2011.

[3] Yuvraj Agarwal, Stefan Savage, and Rajesh Gupta. Sleepserver: a software-only approach for reducing the energy consumption of pcs within enterprise environments. In *USENIXATC'10: Proceedings of the 2010 USENIX conference on USENIX annual technical conference*, pages 22–22, Berkeley, CA, USA, 2010. USENIX Association.

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

[5] Alexandru Iosup, Ozan Sonmez,Shanny Anoep,Dick Epema . The delft grid simulator (dgsim). Disponível em: `http://www.pds.ewi.tudelft.nl/~iosup/dgsim.php`. Último acesso em janeiro de 2011.

[6] David P. Anderson. Boinc: A system for public-resource computing and storage. In *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, GRID '04, pages 4–10, Washington, DC, USA, 2004. IEEE Computer Society.

[7] Nazareno Andrade, Francisco Brasileiro, Walfredo Cirne, and Miranda Mowbray. Automatic grid assembly by promoting collaboration in peer-to-peer grids. *J. Parallel Distrib. Comput.*, 67(8):957–966, 2007.

[8] 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.

[9] J. Baliga, R.W.A. Ayre, K. Hinton, and R.S. Tucker. Green cloud computing: Balancing energy in processing, storage, and transport. *Proceedings of the IEEE*, 99(1):149 –167, 2011.

[10] Luiz A. Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, December 2007.

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

[12] Andreas Berl, Nicholas Race, Johnathan Ishmael, and Hermann de Meer. Network virtualization in energy-efficient office environments. *Comput. Netw.*, 54:2856–2868, November 2010.

[13] Canonical Ltd. Power management in ubuntu. Disponível em: `https://wiki.ubuntu.com/power-management-in-Ubuntu`. Último acesso em janeiro de 2011.

[14] Enrique V. Carrera, Eduardo Pinheiro, and Ricardo Bianchini. Conserving disk energy in network servers. In *Proceedings of the 17th annual international conference on Supercomputing*, ICS '03, pages 86–97, New York, NY, USA, 2003. ACM.

[15] Henri Casanova, Arnaud Legrand, and Martin Quinson. Simgrid: A generic framework for large-scale distributed experiments. In *Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, pages 126–131, Washington, DC, USA, 2008. IEEE Computer Society.

[16] Wu chun Feng and Kirk Cameron. The green500 list: Encouraging sustainable supercomputing. *Computer*, 40(12):50–55, 2007.

[17] 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.

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

[19] 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 2011.

[20] Tathagata Das, Pradeep Padala, Venkata N. Padmanabhan, Ramachandran Ramjee, and Kang G. Shin. Litegreen: saving energy in networked desktops using virtualization. In *USENIXATC'10: Proceedings of the 2010 USENIX conference on USENIX annual technical conference*, pages 3–3, Berkeley, CA, USA, 2010. USENIX Association.

[21] Energy Star. Energy star program requirement for computers (version 5.0). Disponível em: `http://www.energystar.gov/ia/partners/prod_development/revisions/downloads/computer/Version5.0_Computer_Spec.pdf`. Último acesso em novembro de 2010.

[22] Energy Star. Organizations activating power management on the monitor and computer. Disponível em: `http://www.energystar.gov/index.cfm?c=power_mgt.pr_pm_step1`. Último acesso em janeiro de 2011.

[23] D. H. J. Epema, M. Livny, R. van Dantzig, X. Evers, and J. Pruyne. A worldwide flock of condors: load sharing among workstation clusters. *Future Gener. Comput. Syst.*, 12:53–65, May 1996.

[24] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. Power provisioning for a warehouse-sized computer. In *ISCA '07: Proceedings of the 34th annual international symposium on Computer architecture*, pages 13–23, New York, NY, USA, 2007. ACM.

[25] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. Power provisioning for a warehouse-sized computer. *SIGARCH Comput. Archit. News*, 35:13–23, June 2007.

[26] Anshul Gandhi, Mor Harchol-Balter, Rajarshi Das, and Charles Lefurgy. Optimal power allocation in server farms. In *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems*, SIGMETRICS '09, pages 157–168, New York, NY, USA, 2009. ACM.

[27] Saurabh Kumar Garg and Rajkumar Buyya. Exploiting heterogeneity in grid computing for energy-efficient resource allocation, 2009.

[28] Stavros Harizopoulos, Mehul A. Shah, Justin Meza, and Parthasarathy Ranganathan. Energy efficiency: The new holy grail of data management systems research. In *4th Biennial Conference on Innovative Data Systems Research (CIDR)*, January 2009.

[29] 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 janeiro de 2011.

[30] Intel Corporation. Enhanced intel speedstep technology - how to document. Disponível em: `http://www.intel.com/cd/channel/reseller/asmo-na/eng/203838.htm`. Último acesso em janeiro de 2011.

[31] Intel Corporation and Microsoft Corporation. Advanced power management (apm) bios interface specification, 1996. Disponível em: `http://download.microsoft.com/download/1/6/1/161ba512-40e2-4cc9-843a-923143f3456c/APMV12.rtf`. Último acesso em janeiro de 2011.

[32] 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 *GRID '06: Proceedings of the 7th IEEE/ACM International Conference on Grid Computing*, pages 262–269, Washington, DC, USA, 2006. IEEE Computer Society.

[33] 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*, HPDC '08, pages 97–108, New York, NY, USA, 2008. ACM.

[34] Alexandru Iosup, Ozan Sonmez, and Dick Epema. Dgsim: Comparing grid resource management architectures through trace-based simulation. In *Proceedings of the 14th international Euro-Par conference on Parallel Processing*, Euro-Par '08, pages 13–25, Berlin, Heidelberg, 2008. Springer-Verlag.

[35] Bahman Javadi, Derrick Kondo, Jean-Marc Vincent, and David P. Anderson. Discovering statistical models of availability in large distributed systems: An empirical study of seti@home. *IEEE Transactions on Parallel and Distributed Systems*, 99(PrePrints), 2011.

[36] 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.

[37] Kyong Hoon Kim, Rajkumar Buyya, and Jong Kim. Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters. *Cluster Computing and the Grid, IEEE International Symposium on*, 0:541–548, 2007.

[38] D. Kondo, A. Andrzejak, and D.P. Anderson. On correlated availability in internet-distributed systems. In *9th IEEE/ACM International Conference on Grid Computing*, pages 276 –283, 2008.

[39] 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.

[40] Derrick Kondo, Michela Taufer, Charles L. Brooks III, Henri Casanova, and Andrew A. Chien. Characterizing and evaluating desktop grids: An empirical study. volume 1, page 26b, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[41] Jonathan Koomey, Stephen Berard, Marla Sanchez, and Henry Wong. Implications of historical trends in the electrical efficiency of computing. *IEEE Annals of the History of Computing*, 99(PrePrints), 2010.

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

[43] Top 500 list. Disponível em: `http://www.top500.org/lists`. Último acesso em novembro de 2010.

[44] Yung-Hsiang Lu and Giovanni de Micheli. Adaptive hard disk power management on personal computers. In *Proceedings of the Ninth Great Lakes Symposium on VLSI*, GLS '99, pages 50–50, Washington, DC, USA, 1999. IEEE Computer Society.

[45] 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.

[46] Microsoft Corporation. Windows power management. Disponível em: `http://www.microsoft.com/whdc/archive/winpowmgmt.mspx`. Último acesso em janeiro de 2011.

[47] The Trustees of 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 janeiro de 2011.

[48] Anne-Cécile Orgerie, Laurent Lefèvre, and Jean-Patrick Gelas. Chasing gaps between bursts: Towards energy efficient large scale experimental grids. In *Proceedings of the 2008 Ninth International Conference on Parallel and Distributed Computing, Applications and Technologies*, pages 381–389, Washington, DC, USA, 2008. IEEE Computer Society.

[49] Anne-Cécile Orgerie, Laurent Lefèvre, and Jean-Patrick Gelas. Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems. pages 171–178, 2008.

[50] OurGrid Community. Ourgrid web status http://status.ourgrid.org/, 2010.

[51] Venkatesh Pallipadi and Alexey Starikovskiy. The ondemand governor: past, present and future. In *Proceedings of Linux Symposium*, volume 2, pages 223–238, 2006.

[52] Jean-Francois Pineau, Yves Robert, and Frédéric Vivien. Energy-aware scheduling of bag-of-tasks applications on master-worker platforms. *Concurrency and Computation: Practice and Experience*, 23(2):145–157, 2011.

[53] Eduardo Pinheiro, Ricardo Bianchini, and Cezary Dubnicki. Exploiting redundancy to conserve energy in storage systems. *SIGMETRICS Perform. Eval. Rev.*, 34:15–26, June 2006.

[54] 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.

[55] Lesandro Ponciano, Francisco Brasileiro, Jaindson Santana, Marcus Carvalho, and Matheus Gaudencio. Usando as estratégias sobreaviso e hibernação para economizar energia em grades computacionais oportunistas. *Revista Brasileira de Redes de Computadores e Sistemas Distribuídos*, 2011.

[56] Lesandro Ponciano, Jaindson Santana, Marcus Carvalho, Matheus Gaudencio, and Francisco Brasileiro. Análise de estratégias de computação verde em grades computacionais oportunistas. In *Anais do XXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 307–320, Porto Alegre, Brasil, may 2010. SBC.

[57] David Przybyla and Mahmoud Pegah. Dealing with the veiled devil: eco-responsible computing strategy. In *Proceedings of the 35th annual ACM SIGUCCS fall conference*, SIGUCCS '07, pages 296–301, New York, NY, USA, 2007. ACM.

[58] Asfandyar Qureshi, Rick Weber, Hari Balakrishnan, John Guttag, and Bruce Maggs. Cutting the electric bill for internet-scale systems. *SIGCOMM Comput. Commun. Rev.*, 39:123–134, August 2009.

[59] Joshua Reich, Michel Goraczko, Aman Kansal, and Jitendra Padhye. Sleepless in seattle no longer. In *Proceedings of the 2010 USENIX conference on USENIX annual*

technical conference, USENIXATC'10, pages 17–17, Berkeley, CA, USA, 2010. USENIX Association.

[60] Seagate Technology LLC. Product manual: St31000524as, st3750525as, st3500413as, st3320413as, st3250312as, st3160316as, 2010. Disponível em: `http://www.seagate.com/www/en-us/support/document_library`. Último acesso em dezembro de 2010.

[61] Kamal Sharma and Sanjeev Aggarwal. Energy aware scheduling on desktop grid environment with static performance prediction. In *Proceedings of the 2009 Spring Simulation Multiconference*, SpringSim '09, pages 105:1–105:8, San Diego, CA, USA, 2009. Society for Computer Simulation International.

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

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

[64] United States Environmental Protection Agency. Energy star – the power to protect the environment through energy efficiency, 2010. Disponível em: `http://www.energystar.gov/index.cfm?c=about.ab_history`. Último acesso em janeiro de 2011.

[65] Vasanth Venkatachalam and Michael Franz. Power reduction techniques for microprocessor systems. *ACM Comput. Surv.*, 37:195–237, September 2005.

[66] Tao Xie and Yao Sun. Understanding the relationship between energy conservation and reliability in parallel disk arrays. *J. Parallel Distrib. Comput.*, 71:198–210, February 2011.

[67] Shu Yin, Xiaojun Ruan, A. Manzanares, and Xiao Qin. How reliable are parallel disk systems when energy-saving schemes are involved? In *Cluster Computing and Workshops, 2009. CLUSTER '09. IEEE International Conference on*, pages 1 –9, 2009.