RODRIGO PEREIRA BRAGA

GERÊNCIA DE ENERGIA EM AGRUPAMENTOS DE SERVIDORES NA INTERNET

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Belo Horizonte

20 de janeiro de 2006

Resumo

A maioria dos grandes servidores disponíveis atualmente na Internet são realmente sistemas distribuídos para melhoria de escalabilidade, entretanto tais servidores consomem uma quantidade de energia significativa (que deve incluir também a infra-estrutura de resfriamento). Esse custo pode ser reduzido pelo ajuste do número de servidores ligados baseado na capacidade de processamento necessária para servir a carga observada ao longo do tempo, por exemplo, reduzindo o número de servidores ligados quando a carga decresce. Neste trabalho apresentamos uma infra-estrutura transparente para permitir tal gerência de energia. Aplicamos essa infra-estrutura em três serviços de Internet diferentes: um servidor *Web* de conteúdo estático, uma máquina de busca e um servidor de comércio eletrônico multi-camadas. Nossos resultados mostram que o consumo de energia pode ser reduzido em até 30%, em alguns casos, com um pequeno efeito sobre o desempenho percebido pelo usuário.

Abstract

Most of the servers currently on the Internet are actually distributed systems to improve scalability, however such cluster-based servers consume significant amounts of energy (which must include the cooling infrastructure). These costs can be reduced by adjusting the number of powered nodes based on the processing capacity needed to serve the load perceived as time varies, for example, reducing the number of powered machines as the load decreases. In this work we present a transparent architecture for enabling such power management. We applied this architecture to three different Internet services: a Web server, a Search Engine and an Multi-tiered E-Commerce Server. Our results show that energy consumption may be reduced by as much as 30% in some cases,with little effect on performance perceived by the user.

A Jesus Cristo e a minha noiva Ellen

"Todo o homem tem o desejo natural de saber, mas que vale a ciência sem o temor de Deus?...Se eu possuísse toda a ciência do mundo e não tivesse caridade, que me aproveitaria aos olhos de Deus que me há de julgar segundo as minhas obras?"

Autor desconhecido. Imitação de Cristo. Liv. I, cap. II, par. I.

Agradecimentos

"Senhor, nosso Deus, tu és digno de receber a glória, a honra e a majestade, porque criastes todas as coisas: elas existem e foram criadas por tua vontade" (Apo 4, 11). Tenho certeza que fui criado por ti, Senhor. Esta dissertação foi escrita (criada) em dupla: os erros foram feitos por minhas próprias mãos e a parte louvável pela inspiração do nosso Senhor Jesus Cristo. Agradeço a ti por ser o Amor eterno que me sustenta hoje e sempre.

Agradeço a minha noiva e futura esposa Ellen pelo seu amor, que me acolheu em todos os momentos durante o mestrado. Ela me incentivou a atravessar barreiras e dificuldades; e buscar refúgio no colo do Senhor. É a minha companheira de todos os dias, meu presente de Deus. Lindona, TE AMO.

Agradeço todo amor, paciência e atenção dos meus pais, Volmar e Vilma, e da minha irmã, Priscila. Extendo meu agradecimento também a família da Ellen (Milton, Magali e Lidiane) por terem me "aguentado" todo este tempo.

Ao Prof. Dorgival Guedes Neto, agradeço toda sabedoria, apoio e paciência como orientador. Sua presença sempre será um exemplo de professor, pesquisador e amigo para mim.

Agradeço a ajuda, durante toda a graduação, nas iniciações científicas e no início do mestrado ao Prof. Wagner Meira, culminando com a participação na banca examinadora. Obrigado ao Prof. Renato pela ajuda para finalização desse trabalho durante a defesa de dissertação e preparação final desse texto. Ao Prof. Ricardo Bianchini, eu agradeço a orientação durante todo o trabalho na *Rutgers University*, que foi essencial para a concretização deste mestrado, e por ter me acolhido durante estes 6 meses como parte da sua

família nas celebrações e reuniões.

Não poderia de esquecer vocês amigos de toda hora, Alysson, Fê Estela, Lucas, Lorena, Leandrinho e Paulinha, em quem sempre (logo depois da Ellen) despencava o meu desabafo. Ao Padre Danilo, Luis Pimenta e toda comunidade da paróquia Nossa Senhora Mãe da Igreja, agradeço a oração sempre presente e os encontros. A Comunidade Verbo Eterno que, através de todos os seus membros, me adotou e acolheu nos momentos e correções finais. Aos amigos Diniz, Tiago, Diego, Tassni, AndréC, Brut e turma do Laboratório e-Speed vai o meu muito obrigado e desculpas por toda bagunça que fiz no laboratório para a execução desses experimentos. Ao Gama, Taliver, Fábio, Eduardo Pinheiro, Chris e laboratórios DarkLab/Panic-Lab, agradeço pela amizade e apoio técnico durante os 6 meses na *Rutgers Univeristy*. Obrigado Prof. Luiz Carlos Sizenando por toda ajuda na Faculdade Estácio de Sá nos momentos que tive que correr pra UFMG para finalizar o mestrado.

Agradeço a Cristiane Prado pelo apoio, escuta e oração durante os momentos mais difíceis da minha "ansiedade".

Agradeço ao Mosteiro São Geraldo (especialmente a Cela São José) e ao Mosteiro Nossa Senhora das Graças, pelo acolhimento que recebi desses irmãos durante praticamente toda escrita desse trabalho, em especial ao Ir. João Batista por toda oração e atenção. "Fui hóspede e me recebestes" (RB 53, 1). Pax et bonnum!

Gostaria de agradecer também ao Prof. Flávio Vasconcelos do Departamento de Engenharia Elétrica da UFMG pelo empréstimo do medidor de potência *Yokogawa* usado nessa dissertação.

Ao Google por ser sempre exato nas suas buscas que tanto enriqueceram esse trabalho meu muito obrigado. Sem a ajuda do mesmo, nunca seria possível a conclusão desse texto.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Os serviços oferecidos na Internet aumentam significativamente o número de acessos a cada dia. De acordo com estatísticas, o tráfego na Internet duplica a cada ano e tem um crescimento exponencial no número de usuários previsto para os próximos anos [Odlyzko, 2003]. Portanto, esses serviços precisam de uma estrutura que suporte expansão de capacidade de maneira transparente e simples. A maioria dos grandes serviços de Internet são baseados em *cluster*, que são nada menos que agrupamentos de servidores em uma rede local de alto desempenho que possuem a mesma funcionalidade ou distribuem as tarefas de um serviço entre si. Com estes servidores baseados em *clusters*, a expansão de capacidade pode ser alcançada pelo acréscimo de uma nova máquina (ou nodo), desde que haja uma maneira de redistribuir as requisições entre os nodos disponíveis (através de um distribuidor de requisições centralizado ou distribuído). Além disso, esses serviços devem tolerar falhas, excluindo nodos inoperantes ou não-confiáveis, cuja funcionalidade deverá ser provida por outro nodo do *cluster*. Essas características tornam *clusters* de servidores extremamente úteis para os serviços oferecidos na Internet, viabilizando a implantação de sistemas que demandam alto desempenho.

Os *data centers*[1] e os provedores são os grandes usuários de *clusters*, normalmente utili-

[1]Um *data center* é um repositório centralizado, físico ou virtual, para armazenamento, gerência e disseminação de dados e informação organizados sobre uma área do conhecimento em particular ou pertencente a uma organização.

zam o que é chamado de computação de alta densidade [Warren *et al.*, 2002], que implica na utilização de um grande número de servidores em um espaço físico relativamente reduzido. Além disso, os *data centers* e os provedores precisam manter o seu serviço funcionando continuamente, 24 horas por dia, 7 dias por semana (o que é normalmente identificado pela convenção "24x7"). Podemos utilizar como exemplo o sistema de computação de uma máquina de busca do mercado que utiliza um *data center* com 15.000 servidores [Barroso *et al.*, 2003] em um mesmo local. A utilização de grandes *clusters* de alta densidade exigem infra-estruturas de resfriamento para manter o processamento efetivo. Tanto o processamento de alto desempenho quanto o sistema de resfriamento consomem uma quantidade de energia significativa e esse alto consumo é refletido em altas contas de eletricidade. De fato, cada metro quadrado de um *rack* de servidores baseados em processadores de última geração consome em torno de 7,6 kW [Barroso *et al.*, 2003]. Como esse serviço precisa estar ativo sempre, seu consumo de energia gira em torno de 66,7 MWh por ano por cada metro quadrado de espaço de *rack*. Esse consumo de energia torna imperativo para *data centers* e provedores considerar soluções de gerência de *clusters* considerando o consumo de energia [Mitchell-Jackson, 2001].

Diante deste cenário, soluções que visam economia de energia são necessárias por razões econômicas. O estudo destas soluções de gerência de energia tornam viável o contínuo crescimento dos serviços de Internet baseados em *clusters*.

1.1 Objetivos

O objetivo deste nosso trabalho é desenvolver um sistema de gerência de energia para ser aplicado a *clusters* de servidores na Internet, sem a necessidade de re-desenvolvimento do serviço e independendente do *hardware* ou *software* que estão sendo utilizados.

1.2 Abordagem

As soluções já existentes de gerência de energia sempre incluem *hardware* especializado ou mudanças no *software* do servidor/aplicação, sistema operacional ou ambos. Como resultado disso, estas soluções levam a um alto custo em re-desenvolvimento de *software* e pouca portabilidade. Na maior parte dos casos, mudanças no código fonte de aplicações servidores são inaceitáveis devido a políticas de segurança do sistema. Vários *data centers* utilizam servidores com *software* proprietário que não podem ser alterados para obtenção de economia de energia. Em outros casos, o pessoal técnico pode não desejar fazer tais alterações ao *software* com receio de comprometer a estabilidade ou segurança do sistema.

Os serviços de Internet que utilizam grandes *clusters* possuem softwares com tolerância a falhas, isto é, possuem um mecanismo de manutenção da disponibilidade do *cluster*. Caso haja alguma falha em um dos nodos, outro nodo assume as requisições do nodo indisponível com o mínimo de perda de requisições atendidas. Neste ambiente, a gerência de energia pode ser feita aproveitando da capacidade de expansão (adicionando ou removendo nodos) e a tolerância a falhas existente na própria estrutura do *cluster*, visando economizar energia no momento em que o serviço requisita somente uma fração do *cluster* para manter o mesmo desempenho de horas de pico.

Tendo isso em mente, nosso trabalho busca desenvolver um sistema de gerenciamento de energia em *clusters* transparente e não-intrusivo, que possa ser utilizado em qualquer *cluster* em funcionamento que suporte tolerância a falhas sem requerer nenhuma mudança na aplicação ou no sistema operacional do servidor. Portanto, o trabalho desta dissertação de mestrado é apresentar um sistema de gerenciamento de *clusters* (chamado *PASys*, isto é, *Power Aware System*) que combina um formato de descrição e um *software* de infra-estrutura que monitora a carga e os efeitos das mudanças de configurações do *cluster*. O formato de descrição pode ser usado por desenvolvedores de aplicação ou administradores que descrevem a aplicação para o *PASys*, para que ele possa intervir sobre o sistema. Este formato será chamado "formato de descrição de serviços para gerência de energia" nos capítulos seguintes. O *software* de infra-estrutura consiste em uma rede de monitores de

carga, instalados em cada nodo e um gerente do *cluster* que utiliza a descrição da aplicação e a informação da carga dos nodos para decidir qual nodo deve estar ligado a cada momento. A infra-estrutura poderia ser facilmente estendida para usar múltiplos gerentes de *cluster* em grandes configurações de *cluster*.

1.3 Contribuições deste Trabalho

A principal contribuição do nosso trabalho é a criação de um sistema para gerenciamento de energia em *cluster* de servidores que gerencia de maneira transparente e não-intrusiva vários serviços. Através dos experimentos podemos verificar que a generalidade do sistema e o não conhecimento específico de cada serviço utilizado não implicou em uma perda de desempenho significativo, apesar de ter gerado grande economia de energia. Podemos também enumerar outras contribuições deste trabalho:

- Desenvolvimento de uma infra-estrutura de monitoração e reconfiguração[2] para gerenciar o consumo de energia de *clusters* de servidores.
- Definição de um formato de descrição de serviços visando a gerência de energia para facilitar a adequação do *PASys* com cada tipo de servidor, podendo ser efetuada por uma pessoa que desconheça conceitos de gerência de energia.
- Adaptação do servidor de máquina de busca com suporte à tolerância a falhas para ser utilizado pelo *PASys*.

1.4 Organização do Texto

O restante deste trabalho está dividido da seguinte maneira: o capítulo 2 aborda os trabalhos correlatos em gerência de energia e os termos utilizados. O capítulo 3 descreve a estrutura do *PASys* para monitoração e reconfiguração (número de nodos) de *clusters*,

[2]Chamamos de reconfiguração o acréscimo ou decréscimo de nodos no *cluster* de servidores com o objetivo de maximizar o desempenho ou minimizar o consumo de energia, respectivamente.

além de definir o formato de descrição de serviços para gerência de energia. O capítulo 4 define os três casos de estudos utilizados: um servidor HTTP de conteúdo estático, um servidor de comércio eletrônico e uma máquina de busca, apresentando suas características, como estas foram exploradas no formato de descrição de serviços. O capítulo 5 apresenta resultados de desempenho e economia de energia. O capítulo 6 descreve as conclusões obtidas dos experimentos e discute trabalhos futuros.

Capítulo 2

Gerência de Energia em *Clusters*

Para entender o problema de gerência de energia em *clusters*, primeiro abordamos aspectos gerais de gerência de energia, percebendo assim a abrangência da economia da energia sob outros aspectos. Em seguida, apresentamos algumas soluções tecnológicas disponíveis para máquinas *standalone* e finalmente discutimos trabalhos de pesquisa relacionados.

2.1 Aspectos da Energia na Computação

A gerência de energia é fator chave para um sistema de computação competitivo e se torna crítica por vários fatores inter-relacionados: consumo de energia, sobrecarga de temperatura e padrão de consumo. Estes fatores são explicados mais detalhadamente nas próximas seções e poderemos perceber como estes estão relacionados com o custo da energia total. Outro fator importante a considerar é a relação entre desempenho e consumo de energia, a qual limita a economia possível.

2.1.1 Consumo de energia

O agregado de consumo de energia da infra-estrutura dos serviços na Internet como um todo é uma parte significativa no consumo de energia mundial e sua demanda vem crescendo rapidamente [Gupta & Singh, 2003]. Duas razões elevam a importância do

fator consumo de energia: o custo da energia e o impacto da sua geração sobre o meio ambiente. O custo da energia nesse caso é o necessário para manter os servidores em operação, assim como o serviço de refrigeração. Previsões do consumo de energia elétrica nos *data centers* americanos para 2003 eram da casa de 22 TWh, considerando o que é gasto com servidores, armazenamento de dados, equipamentos de rede, energia de segurança (baterias) e ar-condicionado [Mitchell-Jackson, 2001]. Esta energia custaria em torno de dois bilhões de dólares anuais. Este cálculo é feito sem levar em consideração que há sobretarifação no preço da energia na hora de pico. O consumo de energia em grande escala implica em um aumento da geração, o que pode ter um grande impacto sobre o meio ambiente: nos EUA, por exemplo, onde a energia elétrica é de origem termo-elétrica, a geração de 22 TWh produz doze milhões de toneladas de gás carbônico (CO_2) dispersadas na atmosfera [Mitchell-Jackson, 2001]. No Brasil, a energia elétrica é de origem principalmente hidrelétrica (em torno de 85%), entretanto, artigos atuais provam que estas usinas elétricas também geram poluição [Fearnside, 1997] - a usina de Tucuruí gerou, em 1990, de sete a dez milhões de toneladas de CO_2.

2.1.2 Sobrecarga de temperatura

Os clusters de servidores estão sujeitos a sobrecargas de temperatura devido a vários fatores: falhas no sistema de ar-condicionado, condições externas ou elevação imprevista de carga nos servidores. Essas sobrecargas de temperatura exigem uma estrutura mais eficiente e potente de ar-condicionado, o que implica em aumento do custo da energia para manutenção da climatização dos ambientes dos *clusters*. O custo dos sistemas de ar-condicionado acaba excedendo o custo para manter os dispositivos de computação em operação [Sharma *et al.*, 2003, Patel *et al.*, 2002, Bellosa *et al.*, 2003, Brooks & Martonosi, 2001].

2.1.3 Padrão de consumo

O padrão de consumo é a manutenção do perfil de consumo e inclui gerência da capacidade extra de energia. A manutenção do perfil de consumo, isto é, a definição de um consumo médio, é bem visto pelas empresas de energia e recompensado sob a forma de descontos. A ANEEL[1], por exemplo, cobra esse gerenciamento das geradoras de energia e estas, por sua vez, precisam sabê-lo de seus clientes (no nosso caso, os *data centers* e provedores). As fornecedoras de energia oferecem melhor preço na potência gasta se o cliente puder reduzir seu consumo sob demanda. A cobrança do gerenciamento desta energia extra por parte da ANEEL e das geradoras de energia acontece devido a casos emergenciais de falta de energia, isto é, *blackouts*.

2.1.4 Relação entre consumo de energia e desempenho

A economia de energia pode ser alcançada se fixarmos o consumo de energia no mínimo possível. Entretanto, como a carga imposta sobre serviços de Internet tem comportamento fractal, em um momento de alta carga, o serviço ficará comprometido devido a fixação da potência consumida. Apesar da economia de energia ser muito importante, essa economia não pode ser feita em detrimento do desempenho dos sistemas de computação, pois o objetivo principal continua sendo prestar serviços através da Internet usando um *cluster* de computadores para aumentar o desempenho geral do sistema. Temos que considerar o compromisso entre a economia de energia e as duas métricas de desempenho do sistema, como taxa de requisições atendidas e tempo de execução de cada requisição. A taxa de requisições atendidas é uma consideração chave para sistemas como servidores de redes modernos, cujo objetivo é servir o maior número de requisições no menor tempo possível; o tempo de execução de cada requisição no servidor é normalmente uma pequena fração diante da latência da comunicação cliente-servidor em redes de longa distância. O tempo de execução de cada requisição é chave para sistemas como servidores baseados em tarefas em lote (*batch*), já que requisições podem estar sujeitas a atrasos na execução dos serviços

[1] Agência Nacional de Energia Elétrica

solicitados.

2.2 Arquitetura de Computadores e Gerência de Energia

Os desenvolvedores das arquiteturas dos computadores usados em *clusters*, há mais de uma década definem um suporte a economia de energia implícita na sua arquitetura, entretanto hoje são padronizações pouco dependentes da arquitetura [APM, 1996, ACPI, 1999, PMU, 1996]. Inicialmente, estes padrões foram criados para computadores portáteis, mas atualmente todos os computadores os suportam. Os principais mecanismos padronizados de economia de energia em dispositivos são os seguintes:

- **DPMS** — Gerência de Energia de Telas por Sinalização que define estado de consumo de energia para monitores ou placas de vídeo.
- **ATA** — Anexos AT, também conhecidos como discos rígidos, tem uma especificação para economia de energia que diminuem a velocidade de rotação do disco de acordo com o uso.

Apesar das várias opções de mecanismos de economia de energia, é necessária uma interface para coordenação geral do sistema e dos vários mecanismos presentes nos dispositivos, visando economia de energia em todo sistema de computação. Esse papel é cumprido pelas interfaces de gerência de energia: *APM*, *PMU* e *ACPI*.

2.2.1 APM

A primeira interface de configuração voltada para gerência de energia é a Gerência de Energia Avançada (*APM*). Esse mecanismo controla a energia através de estados de consumo e acrescenta suporte ao desligamento automático[2]. Além disso, em computadores

[2]No suporte ao desligamento automático é considerado a existência um comando que realmente desliga a máquina, o que em um ambiente sem gerência somente finaliza o sistema operacional, mantendo o consumo de energia da máquina.

portáteis analisa o consumo da bateria e a conexão com a rede elétrica, o que não é o enfoque desse trabalho. A interface *APM*, configurada no *hardware* via *BIOS*[3] disponibiliza quatro estados possíveis para o sistema: pronto, em espera, suspenso e desligado. Cada estado respectivamente economiza um pouco mais de energia e segue uma especificação pré-determinada pela definição da *APM*. Os dispositivos mais comuns com este suporte são o processador, a placa mãe e algumas placas de rede. Considerando este último dispositivo, existe o suporte *wake-on-lan* que, caso a máquina esteja no terceiro estado da *APM* ("suspenso") e a placa de rede suporte esta característica, muda o estado da máquina para "pronto". A interface de configuração *APM* está entrando em desuso por suas limitações: todas as decisões de economia de energia são tomadas sem o conhecimento do que o usuário está fazendo, somente baseado em tempos de espera definidos pela *BIOS* e falta de suporte para dispositivos novos (dispositivos USB, PCMCIA e outros).

2.2.2 ACPI

A próxima interface é a Interface de Configuração Avançada de Gestão de Energia (*ACPI*) que disponibiliza um sistema completo para gerência de energia baseado em uma configuração controlada por *software*. Os estados não são mais definidos para o computador, mas para cada dispositivo, permitindo assim economizar a energia de um dispositivo que não está sendo usado no momento mas manter os outros dispositivos em sua capacidade total.

A especificação do ACPI permite que o Sistema Operacional possa controlar os estados dos dispositivos através de uma interface chamada *OSPM* (Gerência de Energia por Sistema Operacional). Além do desligamento automático do computador, a interface *ACPI* permite o desligamento de monitores (suporte à especificação *DPMS* para monitores, mencionada anteriormente), desligamentos e estados intermediários de economia de energia de dispositivos ociosos, diminuição da velocidade de discos (suporte à especificação *ATA*), despertar o computador remoto através de dispositivos parcialmente desligados e outros

[3]Sistema Básico de Entrada/Saída

recursos (inclusive o suporte a *wake-on-lan* mencionado na interface de gerência de energia anterior).

O processamento consome parte significativa da energia em um computador. Buscando uma economia de energia neste dispositivo, desenvolvedores de processadores definem escalas de voltagem de alimentação do processador que são suportadas, permitindo que a cada uma destas escalas, o processador opere com uma freqüência diferente. Quanto maior a freqüência do processador, maior desempenho e maior consumo de energia e vice-versa. Estas escalas de voltagens são representadas na interface *ACPI* através dos estados do dispositivo processador. Esta técnica de mudança do relógio do processador se chama Escala de Freqüência do Processador (*CPU Frequency Scaling*) [Weiser *et al.*, 1994]. Nesta técnica de Escala de Freqüência do Processador, existem dois modelos de variação de voltagem:

- **Coordenada** — A variação de voltagem coordenada (CVS - *Coordinated Voltage Scaling*), deixa que o controle desta voltagem possa ser feito a nível de usuário por um *software* que decide quando é melhor alterar a freqüência do processador.

- **Independente** — A variação de voltagem independente (IVS - *Independent Voltage Scaling*) é feita de acordo com a utilização do próprio processador pelo sistema operacional. Se o processador alcançou o máximo de desempenho com a freqüência atual e tem mais processos novos entrando em sua fila de escalonamento, o próprio sistema operacional (que controla o IVS) decide aumentar a freqüência do processador.

Não há necessidade de nenhuma configuração na *BIOS* para o suporte *ACPI*.

2.2.3 PMU

Esta interface é chamada Unidade de Gerência de Energia (*PMU*) que gerencia a energia para computadores da arquitetura *PowerPC*. Atualmente suporta também a especificação do ACPI por questões de compatibilidade.

2.3 Trabalhos Correlatos

O consumo de energia sempre tem sido um ponto crítico no desenvolvimento de computadores portáteis e dispositivos de mão (PDAs [4]), porque estes dispositivos geralmente executam usando uma bateria de carga limitada e não são conectados à rede elétrica [Chiasserini & Rao, 2000, Flinn & Satyanarayanan, 1999]. Várias pesquisas foram direcionadas ao consumo de energia e à conservação de energia na computação móvel [Halfhill, 2000, Weiser *et al.*, 1994, Lebeck *et al.*, 2000, Douglis & Krishnan, 1995, Mini *et al.*, 2005, Lorch, 1995].

Nosso trabalho, por outro lado, foca na economia de energia em computadores não-portáteis e nesta área, podemos listar vários esforços feitos para utilização de técnicas de economia de energia em sistemas de computação em várias áreas. Estes esforços são padrões de economia de energia utilizados mundialmente [ACPI, 1999, Semeraro *et al.*, 2002, Douglis *et al.*, 1994, Weiser *et al.*, 1994] na área de Arquitetura de Computadores. Nas áreas de Compiladores e Algoritmos e Estrutura de Dados, vemos resultados de economia de energia na alteração de alguns programas [Heath *et al.*, 2002], assim como na maneira que são compilados [Yang, 2004], entretanto esta alteração não pode ser usada em nosso sistema já que assumimos como premissa que os programas dos servidores não podem ser alterados. O mesmo acontece nas áreas de Sistemas Operacionais e Redes de Computadores, todas as mudanças necessitam de hardware específico ou mudanças no sistema operacional [Chase & Doyle, 2001, Lu *et al.*, 2002] ou no protocolo de comunicação [Singh & Raghavendra, 1999].

Analisando soluções de gerência de energia para um servidor ou computador pessoal *standalone*, existem algumas estratégias baseadas em modificações do hardware ou software de servidores. Nas soluções baseadas na alteração do hardware, podemos incluir a variação da freqüência da CPU (variando a freqüência do processador de acordo com o desempenho esperado) [Chiasserini & Rao, 2000, Semeraro *et al.*, 2002, Weiser *et al.*, 1994], desligando discos rígidos ociosos, ou trocando tais discos por memórias *flash* ou discos de computadores

[4] Personal Digital Assistant

portáteis [Carrera *et al.*, 2003]. Estratégias baseadas em software incluem: tentativas de reduzir o número de ciclos utilizados na execução de uma aplicação [Heath *et al.*, 2002] ou utilização de protocolos diferentes para reduzir o consumo de energia durante tarefas de comunicação [Flinn & Satyanarayanan, 1999]. Todas estas estratégias requerem alterações nos software da aplicação nas máquinas para ter acesso a estas facilidades.

Voltando a atenção para o ambiente de *clusters*, um trabalho anterior em conservação de energia em cluster de servidores focalizou seus esforços na comparação das diferentes técnicas de economia de energia: desligar/ligar nodos, técnica de modo coordenado ou independente da variação da voltagem do processador e algumas técnicas híbridas, este concluiu que a economia de energia é alcançada de maneira mais eficiente desligando e ligando nodos em um *cluster* [Elnozahy *et al.*, 2002]. Outros pesquisadores descreveram uma solução para cluster usando algoritmo de alocação de recurso baseado em funções de lucro e renda [Chase *et al.*, 2001]. Entretanto, este último trabalho é baseado em um sistema operacional específico (Muse) e testado com um servidor web adaptado, conseqüentemente a solução proposta não é nem portável e nem transparente.

Estes trabalhos descrevem como alcançar um balanceamento e concentração de carga em um servidor web distribuído em um cluster de máquinas [Pinheiro *et al.*, 2001]. Esses artigos já discutem em detalhes o uso do controlador com realimentação PID (Proporcional-Integral-Diferencial) como base do algoritmo de decisão de reconfiguração do cluster [Pinheiro *et al.*, 2002, Pinheiro *et al.*, 2003, Carrera & Bianchini, 2001]. Estes últimos trabalhos aplicam as mesmas técnicas da área de Teoria de Controle utilizadas nesse trabalho.

Capítulo 3

PASys

O *PASys* é o sistema de gerência de energia para *clusters* de servidores que foi desenvolvido nesse trabalho para contemplar características que os sistemas de gerências de energia discutidos nos trabalhos correlatos (Seção 2.3) não possuem. Com o objetivo de economizar energia em *clusters*, nossa abordagem de gerência, utilizando algum suporte provido pelos mecanismos de sistemas atuais, deve atender os seguintes requisitos:

- *Transparência* — a arquitetura não deve depender de modificações nas aplicações do servidor; isso visa evitar custos de re-desenvolvimento e mudanças em *software* proprietário;
- *Portabilidade* — a solução deve ser independente do sistema operacional utilizado, especialmente devido ao fato de que nodos com diferentes funcionalidades podem requisitar diferentes sistemas operacionais ou arquiteturas de *hardware*; e
- *Não-Intrusividade* — sistemas conscientes do consumo de energia não devem interferir significativamente na carga de cada nodo, pois isso implicaria em perda de desempenho. Entretanto, o sistema necessitará de alguma informação sobre o estado do serviço em cada nodo.

Visando cumprir o requisito de não-intrusividade, a infra-estrutura é constituída por módulos que se comunicam usando a rede local para coletar as informações dos nodos e a

partir desses dados reconfigurar o *cluster*, sem exigir comunicação ou consultas à aplicação. A portabilidade do código é garantida pela infra-estrutura, pois no nodo da aplicação, o programa é um coletor de informações portável. O requisito de transparência é completado por um formato de descrição de serviços, pois este descreve as características da aplicação no *cluster* para o *PASys*, já que esse último não pode depender da aplicação para essas informações.

A habilidade de adicionar ou remover nodos de um *cluster* com nenhum ou pouco trabalho de reconfiguração é um importante avanço nos tempos atuais. No *PASys*, assumimos que todos as aplicações são tolerantes a falhas, replicando todos os dados e funcionalidades requeridos através dos múltiplos nodos do *cluster* e podendo assim também cumprir com o requisito de transparência. Em tais servidores, um novo nodo pode ser adicionado para o serviço do *cluster* simplesmente adicionando-o a rede local e atualizando a lista de nodos ativos ao distribuidor de requisições. A remoção de nodos envolve operações similares. Portanto, nós podemos ligar (adicionar) e desligar (remover) servidores e confiar que o sistema original será capaz de se ajustar à nossa configuração do *cluster*.

3.1 Infra-estrutura

A infra-estrutura do *PASys* consiste em dois módulos: o Monitor dos nodos e o Gerente, responsável pela tomada de decisões com base nos dados do *cluster*. Estes dois módulos devem, em conjunto, obter informações do *cluster* e reconfigurá-lo de acordo com a carga imposta ao serviço. A comunicação entre a aplicação do servidor, o sistema operacional, o módulo Monitor e o módulo Gerente é ilustrado na Figura 3.1 e discutido a seguir.

Para ligar um nodo usamos uma característica chamada *wake-on-lan* oferecida pelo suporte da placa-mãe a ACPI (seu funcionamento foi explicado na Seção 2.2.2). Através de uma pacote especial enviado a interface de rede *ethernet* do nodo (que continua parcialmente ligada - estado "suspendido" pela definição da interface de controle de energia utilizada), a máquina liga automaticamente. Para desligar os nodos, nós usamos o suporte da placa a APM (explicado na Seção 2.2.1), já que o suporte ACPI para desligamento na

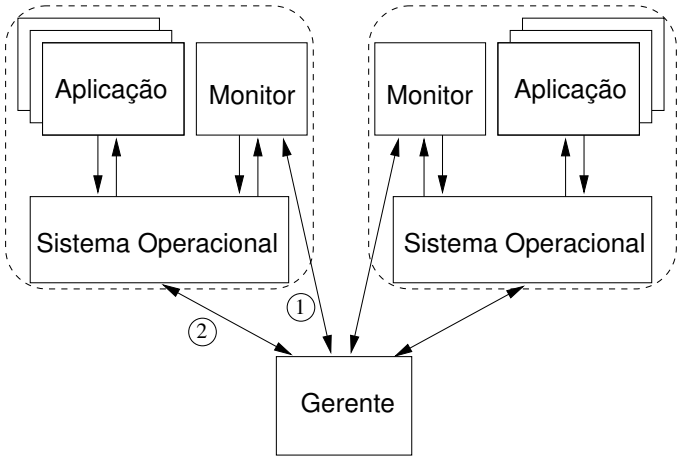

Figura 3.1: Infra-estrutura do *PASys*

nossa configuração não se mostrou muito estável durante alguns testes.

3.1.1 Monitor

O módulo Monitor é um processo *daemon*[1] executado em cada nodo do *cluster* gerenciado. Esse módulo tem por objetivo coletar de forma transparente informações sobre o estado do nodo e da aplicação para repassar ao módulo Gerente. Esse monitor não pode ser intrusivo com relação aos processos do servidor, por isso deve coletar informações disponíveis na interface do sistema operacional que possam ser usadas para caracterizar a carga no nodo. A informação pode incluir a utilização de CPU, disco e rede, por exemplo.

Na implementação atual, feita no sistema operacional *Linux*, nós coletamos estas três informações através do sistema de arquivos `/proc`. A API Win32 denominada `WinMain()` pode ser usada para a coleta das mesmas informações em sistemas operacionais da família *Windows*.

Toda a informação obtida pelo módulo Monitor é enviada ao módulo Gerente como é ilustrado pela seta 1 na Figura 3.1.

[1]Processo executado em *background* do mesmo modo que um serviço do sistema operacional.

3.1.2 Gerente

O módulo Gerente é o processo que recebe a informação dos monitores e toma a decisão baseado nestes valores. A decisão pode ser, por exemplo, desligar um nodo ativo a fim de reduzir a carga imposta ao sistema, ou ligar outro nodo para lidar com um crescimento da carga observada.

Na Figura 3.1, o módulo Gerente recebe informações do Monitor como indicado pela seta 1 e envia uma requisição de ligação ou desligamento para o sistema operacional do nodo (indicado pela seta 2) através das interfaces de controle de energia (ACPI e APM) explicadas na Seção 2.2, de acordo com a decisão daquele módulo. O Gerente pode ser executado em um dos nodos servidores, entretanto isto pode ser um problema caso a carga do serviço possa afetar o seu comportamento (por exemplo, um servidor com tráfego de rede intenso pode atrasar a recepção da informação vinda dos monitores de maneira inaceitável). Para evitar tais problemas, o Gerente deve ser mantido em um nodo dedicado, se possível.

É essencial observar o compromisso entre o desempenho e consumo de energia, como discutido na Seção 2.1: melhorar o desempenho significa adicionar mais nodos (ligando-os), o que significa consumir mais energia. Usando o argumento oposto, para reduzir o consumo de energia devemos desligar nodos, o que pode reduzir o desempenho geral. O Gerente deve lidar com esse compromisso de acordo com a carga imposta: quando é necessário suportar uma sobrecarga dos clientes, o gerente ligará mais nodos, até o máximo da capacidade do *cluster* para garantir o desempenho satisfatório, se preciso. Por outro lado, se perceber uma redução da carga, poderá desligar nodos enquanto o desempenho não for afetado significativamente, até o ponto em que somente um nodo esteja ligado. Como essa decisão é tomada em tempo de execução (*online*), a medição do desempenho e a avaliação de opções devem ser feitos enquanto as requisições estão sendo tratadas.

O algoritmo de decisão de reconfiguração assume que já existe um mecanismo responsável pelo balanceamento e desbalanceamento da carga no *cluster* já que o sistema é tolerante a falhas. A aplicação com tolerância a falhas deve prever a necessidade da redistribuição da carga quando um novo nodo é ligado e do balanceamento quando se decide

desligar um nodo e remover as requisições (ou esperar o seu término) em execução na "vítima" escolhida antes de desligá-la.

3.2 Princípio de Funcionamento

A solução adotada no *PASys* para decidir a alteração da configuração utilizando a carga dos nodos é considerar o *cluster* em questão como um sistema de controle com realimentação como ilustrado na Figura 3.2. O comportamento do sistema de controle pode ser descrito da seguinte maneira: o usuário do PASys, que é o administrador da aplicação, pretende manter o comportamento do sistema em equilíbrio, para isso, define o comportamento esperado para o sistema em termos do valor desejado para uma grandeza mensurável (esse valor é denominado *setpoint*, SP) — por exemplo, a latência máxima aceitável; o sistema de controle (ou controlador) mede o valor real da variável escolhida (denominado variável do processo , PV) e calcula o erro, $e = SP - PV$. Com base nesse erro, o controlador deve calcular o valor de atuação sobre o sistema (saída do controlador, CO) que será aplicado sobre o sistema a fim de tentar reduzir o erro, aproximando PV de SP [Franklin *et al.*, 1997, Seborg *et al.*, 1989].

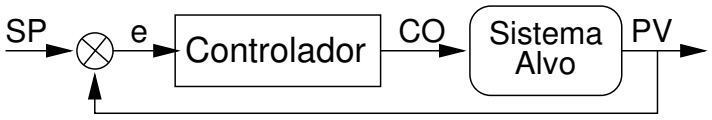

Figura 3.2: Controle com realimentação

Essa solução é utilizada na Teoria de Controle quando a relação entre a variável de processo (PV) e a grandeza de atuação sobre o sistema (CO) não é clara, não sendo possível prever a saída do sistema simplesmente calculando-se um valor de entrada. Nesse caso, o comportamento interno do controlador busca ajustar a entrada dinamicamente, em função da realimentação oferecida pelo erro, até que a saída desejada seja alcançada. A forma tradicional de se conseguir esse comportamento é através do que é chamado um controlador Proporcional-Integral-Derivativo (PID). Nesse caso, a saída do controlador e o

erro calculado são relacionados pela equação:

$$CO = K_p e + K_i \int_0^t e.dt + K_d \frac{de}{dt}$$

As constantes do controlador PID (K_p, K_i e K_d) definem o peso das três componentes do erro e devem ser determinadas por um processo de sintonia realizado na instalação do sistema [Clarke, 1984, Ogata, 1997].

No *PASys* o objetivo é reduzir o consumo de energia tanto quanto possível, mantendo a qualidade do serviço dentro de níveis aceitáveis. A qualidade de serviço instantânea é medida no *cluster* através da coleta de informações de carga pelo módulo Monitor, estas são as métricas: utilização de CPU (U_c), utilização de disco (U_d) e utilização de rede (U_n). Na implementação atual do sistema, esses valores são calculados periodicamente com base em contadores mantidos pelo sistema operacional e limites de capacidade obtidos para os dispositivos:

- Utilização de CPU: razão entre os ciclos contabilizados pelo sistema operacional como tempo de usuário e tempo de sistema (*user+system time*) e o total de ciclos contabilizados no período;

- Utilização de disco: razão entre a taxa de transferência de dados média entre o disco e a memória durante o período de monitoração e a capacidade máxima do disco, conforme indicado pela ferramenta `hdparm`; a taxa média é obtida como o número de bytes transferidos contabilizados pelo sistema durante o intervalo de monitoração dividido pela duração do mesmo;

- Utilização de rede: de forma semelhante à utilização de disco, é determinada em função do número de bytes que trafegam pela interface e da capacidade máxima da mesma, no caso dos experimentos, *Ethernet* Rápida (100Mbps).

Para definir a variável do processo (PV), precisamos de algum dado que informe o desempenho atual do sistema e possa ser coletado instantâneamente e a qualquer momento. Como as informações de carga das máquinas, na maioria das vezes, refletem o consumo de

energia, as medições de cargas dos nodos do *cluster* podem ser utilizados como PV visando a gerência de energia. Portanto esses valores medidos são usados na definição da variável do processo (PV) utilizada:

$$PV = max(U_c, U_d, U_n)$$

Uma possível definição de PV seria uma análise composta dos três tipos de utilização, mas isso tornaria a decisão de reconfiguração do *PASys* mais complexa, sem uma garantia de ganho no resultado final. A utilização do máximo dos valores de carga medidos como variável do processo é a proposta de tratar a informação de carga que se encontra mais próxima do seu limite.

No caso do controlador PID usado no *PASys*, o erro será a diferença entre a utilização do recurso medido (escolhido pela fórmula de PV anterior) e o valor estabelecido (SP). Se o valor medido é menor (taxa mais baixa) que o limite estabelecido, isso indica que o sistema opera com folga e o controlador pode decidir pela redução do consumo, reduzindo algum recurso (desligando alguns dos servidores). Por outro lado, se o valor medido é maior do que limite estipulado (taxa de atendimento alta) o controlador deve adicionar recursos ao sistema (ligando mais nodos) a fim de melhorar o desempenho.

As ações a tomar para controlar o consumo de energia em *clusters* podem ser de três tipos, conforme discutido anteriormente: ligar e desligar nodos, controlar diretamente a freqüência do processador (Escalamento de freqüência coordenado, CVS) e utilizar um controle independente da freqüência (Escalamento independente de freqüência, IVS). Trabalhos anteriores na literatura [Elnozahy *et al.*, 2002] indicam que soluções de controle coordenado de freqüência não apresentam ganhos significativos se comparadas a soluções de controle independente associadas à ligação e desligamento de nodos, sendo que esta última técnica é responsável pelos melhores resultados. Além disso, o controle coordenado implica em sistemas de controle menos transparentes e mais complexos. Sendo assim, o *PASys* considera apenas a reconfiguração baseada na adição e remoção de nodos, o que é feito utilizando-se recursos das interfaces de controle de energia discutidas anteriormente (APM e ACPI, Seção 2.2). Dessa forma, o sistema de controle pode operar em paralelo

às soluções de controle de freqüência independentes já existentes nos sistemas operacionais modernos.

A implementação do módulo Gerente proposto exige a regulagem do controlador pela determinação das constantes do controlador PID. As constantes utilizadas neste trabalho foram aquelas definidas por [Pinheiro *et al.*, 2001, Carrera & Bianchini, 2001]. Todas as constantes do controlador são transparentes para o administrador do *cluster*, que deve apenas indicar o valor desejado para a carga do sistema (discutido na Seção 3.3) para manter limites de desempenho aceitáveis. Esse valor definido como ideal para a carga será o *setpoint* (SP). O serviço também não precisa ter conhecimento das características da política de decisão do módulo Gerente. Para garantir a estabilidade, o controlador não pode reagir muito rápido a mudanças do *cluster*; deve aguardar o tempo que leva para um sistema carregar e inicializar o novo servidor (ou desligar o servidor) antes de reavaliar a condição do *cluster*. No nosso *cluster*, isto consome em torno de três minutos (180 segundos) e esse tempo é chamado de tempo de reconfiguração.

3.3 Formato de Descrição de Serviços

Como discutido anteriormente, as soluções atuais de economia de energia assumem posições extremas:

- **Conscientes do Serviço** — solução baseada completamente na aplicação ou sistema operacional e precisa de alterações no *software* para determinar aspectos específicos da aplicação ou da estrutura do *cluster*. Qualquer modificação feita na aplicação ou no *cluster* implica em uma modificação dessa solução de gerência de energia.

- **Conscientes do *Hardware*** — solução baseada simplesmente no consumo de energia e em características do *hardware*, o que torna esta solução transparente para a aplicação.

O *PASys* gerencia a energia do *cluster* de acordo com a estrutura da aplicação sem deixar de cumprir o requisito de transparência discutido anteriormente. Para obter in-

formações da aplicação, do *cluster* ou do sistema operacional de maneira transparente se faz necessário um formato para descrever essas informações. As informações necessárias de serviços baseados em *clusters* de acordo com a nossa análise foram as seguintes: número de nodos do *cluster*, valor máximo para métrica de desempenho, disponibilidade dos nodos e divisão dos nodos em multi-camadas. O formato utilizado deve exprimir todas estas informações.

Com o objetivo de tornar a arquitetura transparente para a aplicação, a maneira do administrador do sistema descrever os papéis das múltiplas máquinas no *cluster* e como elas podem ser tratadas pelo algoritmo de decisão de reconfiguração é através de um formato de descrição. Nosso formato de descrição de serviços para gerência de Energia descreve os nodos e suas características para o módulo Gerente.

O administrador do sistema que utilize o formato de descrição não precisa de conhecimentos sobre gerência de energia e muito menos de conhecimentos específicos sobre estrutura interna da aplicação; basta saber especificar a estrutura do *cluster*.

O arquivo de configuração utilizado pelo módulo Gerente é um conjunto simples de definições, uma linha para cada nodo do *cluster*. Cada nodo é definido usando estes quatro elementos, na seguinte ordem:

- **Identificação do Nodo** — identifica o nodo de uma forma que o módulo Gerente possa usar para acessá-lo para ordenar a adição do nodo (ligação) ou remoção (desligamento). Na nossa implementação, cada nodo é identificado pelo seu único endereço *IP* na rede local;

- **Identificação de Grupo** — permite ao administrador do sistema identificar grupos de nodos que compartilham um mesmo papel e funcionalidade. O módulo Gerente trata todas as máquinas em um dado grupo de forma separada de outros grupos. Por exemplo, quando um serviço baseado em arquitetura multi-camada (*multi-tier*) é considerado, cada camada é marcada com um identificador de grupo diferente;

- **Disponibilidade** — indica quando um nodo pode ser considerado um candidato para ser desligado pelo módulo Gerente. Isto permite ao administrador do sistema

identificar máquinas que não podem ser desligadas, talvez por serem responsáveis por algum serviço essencial, como DNS ou roteamento, ou serem o único servidor em um grupo;

- **Valor da Grandeza** — define um limiar (*threshold*) para a carga preferida para aquele nodo e é representado por uma porcentagem de carga. Este limiar é o *setpoint* para o controlador PID (explicado na seção anterior). Isso permite refinar o algoritmo para um serviço específico, indicando quão alta pode ser a carga do nodo antes que ele seja considerado sobrecarregado (o que pode resultar na ativação de outro nodo). O administrador do sistema tem o poder de alterar a resposta ao sistema de economia de energia se achar conveniente de uma maneira fácil, visando aumentar a economia de energia ou o desempenho do sistema. Se o administrador preferir, o campo pode ser deixado em branco e o valor padrão é usado pelo algoritmo de decisão de reconfiguração, obtido empiricamente. Um estudo do impacto deste limiar para diferentes serviços é apresentado no capítulo 6.

Um exemplo de um conjunto simples de definições é:

```
192.168.0.230|1|Y|0.7
192.168.0.209|1|N|-
```

Na definição acima, percebemos dois nodos identificados através dos IPs de cada um deles. Em cada linha, se encontra a informação inerente a cada nodo separado pelo caractere da barra vertical (*pipe*). O primeiro número se refere à camada em que o nodo se encontra em um *cluster* multi-camada; no caso, os dois nodos pertencem a uma única camada. O segundo item identifica a disponibilidade do nodo: o primeiro nodo é desligável, enquanto o segundo não. O terceiro item é o parâmetro de valor da grandeza para o administrador do sistema e indica que o primeiro nodo usará, como limiar para o controlador PID (SP), 70% de carga do recurso. Para os casos de estudo da próxima seção, os arquivos de definição do *cluster* para cada serviço se encontram no Apêndice A.

Capítulo 4

Casos de Estudo

Para demonstrar a aplicabilidade de nossa abordagem a diferentes tipos de serviços baseados em *clusters*, nós utilizamos o sistema descrito no capítulo anterior, o *PASys*, em uma máquina de busca, um servidor *web* de conteúdo estático e um serviço de comércio eletrônico organizado em três camadas. Esses serviços cobrem um vasto leque de características e representam alguns dos serviços mais comuns usados atualmente na Internet.

É importante conhecer a estrutura dos serviços utilizados baseados em *clusters* de servidores e seu suporte a tolerância a falhas, antes de analisar o impacto do sistema de gerenciamento de energia *PASys* sobre esses.

4.1 Máquina de Busca

Serviços de Busca na *Internet* se baseiam em um modelo de recuperação de informação como estratégias de busca de documentos presentes na rede que são relevantes para uma consulta submetida à máquina de busca. A recuperação de informação é dividida em dois processos distintos: processo de indexação e processo de recuperação, conforme ilustrado na Figura 4.1.

O processo de indexação consiste nas seguintes etapas: a partir dos documentos coletados (no caso de máquinas de busca na Internet, estes documentos são todos aqueles

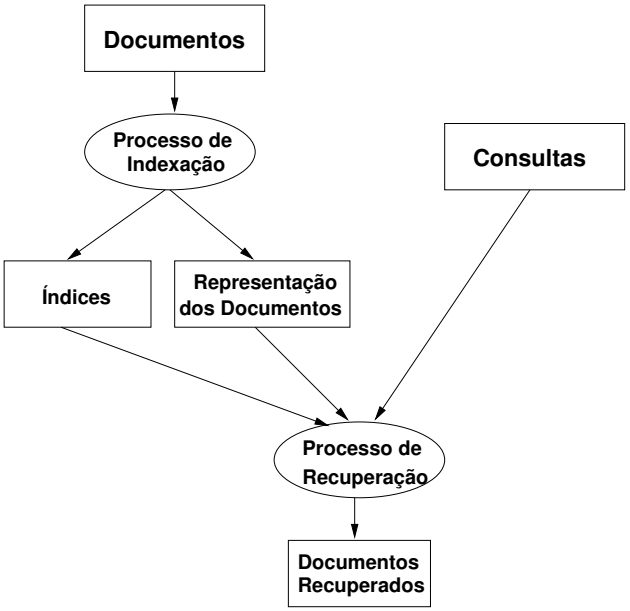

Figura 4.1: Recuperação de Informação em Máquina de Busca

alcançáveis na rede), cria-se um índice que permite a busca de palavras-chaves nestes documentos e uma representação desses documentos para composição da resposta dada ao usuário do serviço, conforme Figura 4.1. O índice pode ser feito de duas maneiras: mapeando para cada documento quais são as palavras-chaves (termos) presentes neste documento ou o inverso, mapeando para cada termo os documentos que o possuem. Este segundo método é mais eficiente, portanto mais utilizado, e é chamado lista invertida. A representação do documento segue um modelo de recuperação de informação.

O processo de recuperação consiste em determinar os documentos que satisfazem a consulta, isto é, aqueles que possuem alguns dos termos da consulta. Com o resultado da busca, é preciso ordenar os documentos por relevância, para isso se usa um método de classificação (*ranking*). Como modelos clássicos de recuperação para classificação de documentos temos os seguintes, conforme [Baeza-Yates & Ribeiro-Neto, 1999]:

- Modelo Vetorial — basea-se na representação de documentos e consultas sob a forma de vetores, onde cada termo é uma dimensão do gráfico e a freqüência de ocorrência deste termo é a coordenada do gráfico na dimensão. Para ordenar as respostas, usa-se uma comparação vetorial entre o vetor que representa a consulta e o vetor que representa o documento que possui os termos da consulta (chamado cálculo de

similaridade) [Salton & Bucckley, 1988]. Este é o modelo usado neste trabalho.

- Modelo Booleano — basea-se na interseção/união/exclusão (de acordo com a expressão booleana definida na consulta - usando os possíveis conectivos lógicos) das listas de documentos recuperados por termo de indexação [Salton, 1989].

- Modelo Probabilístico — basea-se na representação da presença e ausência de termos com pesos binários e os documentos são recuperados com base na probabilidade de que um documentos seja relevante para uma consulta [Van Rijsbergen, 1979].

A arquitetura da máquina de busca distribuída em *cluster* segue a Figura 4.2. Entre os clientes do serviço de busca e os servidores de busca existe um intermediador denominado *Broker*. O *Broker* recebe as requisições dos clientes (a consulta), gera a requisição para os vários nodos processadores responsáveis pelos termos da consulta e, por último, estrutura a página de resposta aos clientes e a envia aos que requisitaram a consulta.

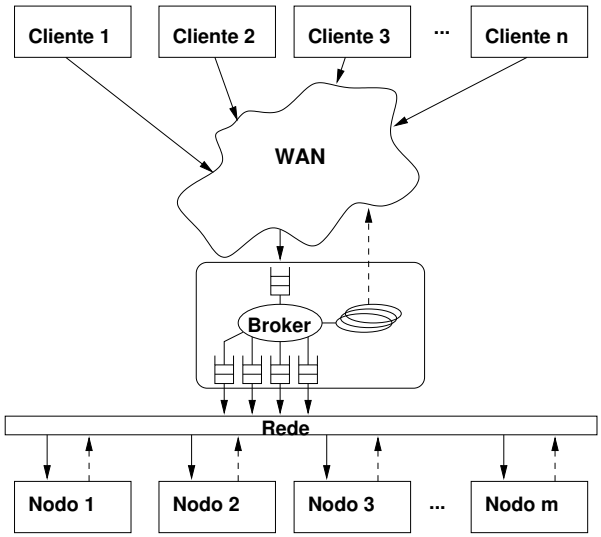

Figura 4.2: Infra-estrutura da Máquina de Busca

A máquina de busca criada neste trabalho utiliza um método de classificação baseado em um modelo vetorial com índices usando lista invertida e tem como base a máquina de busca de outro trabalho [Badue *et al.*, 2001]. Com o enfoque em gerência de energia, acrescentamos à máquina de busca original o suporte a tolerância a falhas, através da replicação do índice em todos os nodos. O *Broker* possui um temporizador *heartbeat* que

permite detectar se uma máquina foi desligada, quando então aquela máquina é removida da lista de nodos disponíveis e, portanto, para de enviar requisições a esse nodo.

4.2 Servidor de Conteúdo Estático

Um serviço de conteúdo Web estático não é nada mais que um servidor de disponibilização de arquivos utilizando o protocolo *HTTP*. O servidor de conteúdo estático usado nesses experimentos é o `press` [Carrera & Bianchini, 2001], um servidor portável e consciente de localidade. Nossa escolha do `press` foi devida ao seu alto desempenho e ao fato de que alguns trabalhos em gerência de energia para *cluster* o utilizam. A estrutura do servidor de conteúdo estático se encontra na figura 4.3.

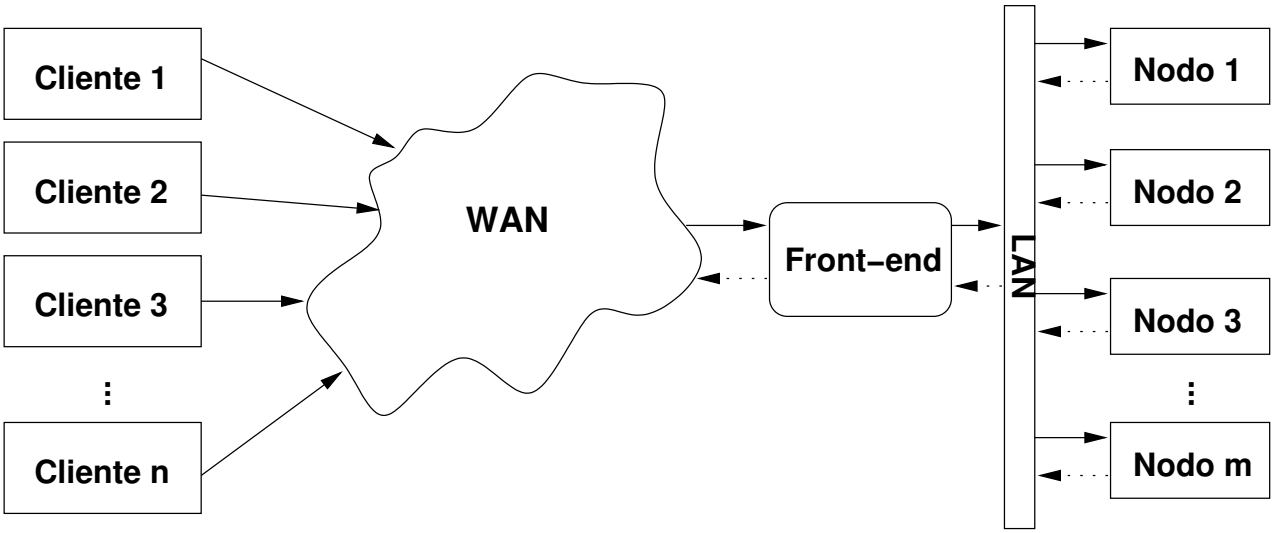

Figura 4.3: Infra-estrutura do Servidor de Páginas Estáticas

O servidor de conteúdo estático distribuído funciona da seguinte maneira: o cliente requisita a página ao *front-end* do *cluster* através do protocolo *HTTP*; o *front-end* (provavelmente um *hardware* distribuidor de requisições) envia a requisição a um dos nodos do *cluster*; e esse nodo envia o conteúdo ao cliente. Como na máquina de busca, todo conteúdo é replicado em todos os nodos do *cluster*. Isso implica em tolerância a falhas, pois o *front-end* pode distribuir as requisições seguindo uma distribuição *round-robin* apenas entre os nodos ativos, sem repassar requisições a um nodo que esteja desligado.

4.3 Servidor de Comércio Eletrônico

O servidor de comércio eletrônico foi escolhido por causa de sua organização em camadas, o que não ocorria nos dois casos anteriores. As características interessantes em um serviço multi-camadas são sua capacidade de tolerância a falhas, já que cada camada normalmente tem vários servidores que provêem o mesmo serviço, e sua necessidade de manter pelo menos um nodo de cada serviço específico (isto é, de uma camada) ligado em qualquer instante. A estrutura multi-camada de um *cluster* de comércio eletrônico pode ser observada na figura 4.4. O serviço de comércio eletrônico é organizado em três camadas: servidores de conteúdo estático (*Web*), servidores de aplicação e um servidor de banco de dados. Os três tipos de nodos servidores se encontram na mesma rede local.

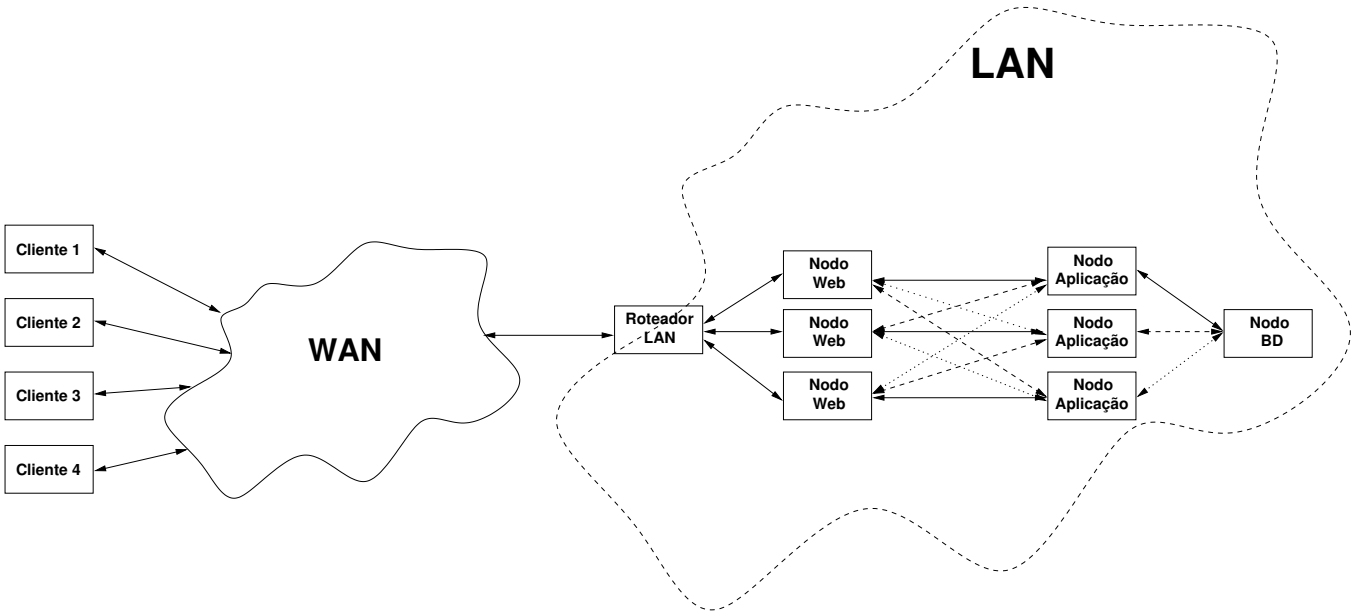

Figura 4.4: Servidor Multicamada em Comércio Eletrônico

Neste caso, ao contrário dos anteriores, não foi usado um *front-end* para distribuição de requisições. Isso é feito aqui através do uso da técnica de DNS *round-robin*: cada cliente que tenta localizar o serviço pode receber o endereço IP de qualquer um dos nodos *Web* existentes.

Um serviço de comércio eletrônico pode servir os seguintes conteúdos:

- Conteúdo estático — imagens, vídeos, páginas estáticas (p. ex.: formas de contato

do serviço de comércio eletônico, explicações sobre o pagamento/funcionamento do serviço);

- Conteúdo dinâmico — páginas geradas com base em dados de um banco de dados ou em dados obtidos durante a conexão.

No caso de conteúdo estático, a conexão acontece da seguinte maneira: o cliente requisita algum conteúdo estático ao nodo servidor *Web* pelo endereço IP; e esse servidor retorna o arquivo estático disponível em seu repositório local. No caso de conteúdo dinâmico, a conexão é mais complexa: o cliente requisita o conteúdo ao servidor *Web*; o servidor *Web* envia a requisição do conteúdo dinâmico ao servidor de aplicação; se a requisição for para uma página dinâmica sem necessidades de dados armazenados no banco de dados, o servidor de aplicação responde ao cliente; caso contrário o servidor consulta o banco de dados e retorna a página dinâmica gerada com base nos dados ao cliente.

O servidor de comércio eletrônico é modelado a partir de uma livraria eletrônica. O código fonte deste serviço é publicamente disponível pelo Projeto DynaServer da Universidade de Rice [DynaServer, 2003]. As camadas utilizam o servidor web de conteúdo estático Apache (versão 1.3.27), o servidor de aplicação *servlets* Tomcat (versão 4.1.18) e o servidor de banco de dados relacional MySQL (versão 4.12), respectivamente. Já que o servidor de gerenciamento de banco de dados não é replicável, nós o descrevemos usando o formato de descrição de serviços (Seção 3.3) como não desligável.

4.4 Sumário

Os três casos de estudos apresentados permitem a análise abrangente do *PASys*. Com a máquina de busca, podemos considerar o comportamento do sistema utilizando um *cluster* que demanda muito do processador. No caso do servidor de conteúdo estático, o enfoque está na grande utilização da interface de rede e do disco. Por último, no servidor de comércio eletrônico, a importância dessa aplicação está no uso de um sistema multicamadas.

No próximo capítulo, discutimos experimentos com os três casos de estudo e analisamos a gerência de energia desempenhada pelo *PASys* nesses casos.

Capítulo 5

Resultados

Através da utilização de três diferentes arquiteturas de servidores discutidos na seção anterior, nós podemos verificar a aplicabilidade do *PASys* a diferentes situações. Os resultados a seguir discutem primeiramente a máquina de busca, seguida do servidor de conteúdo estático, que é discutido em maiores detalhes, e o servidor de comércio eletrônico conclui a análise.

Para muitos dos resultados, nós mostramos figuras com a taxa de requisições percebidas pelo cliente e a energia consumida pelo sistema a cada momento. Para comparação, cada servidor foi simulado e monitorado sem o *PASys*, executando com todos os nodos ligados. Posteriormente, o *PASys* foi habilitado e os *clusters* iniciaram com o mínimo de nodos ligados.

A medição do desempenho foi através da contagem de requisições atendidas por segundo de acordo com o cliente, portanto qualquer atraso na rede foi contabilizado neste desempenho, apesar de que todos se encontravam na mesma rede local como veremos a seguir.

Nossos experimentos foram conduzidos em um *cluster* com 7 nodos com o sistema operacional *Linux* Kernel 2.4.18, com processadores Pentium 4 de 1,80 GHz, *caches* de 512 KB, 1 gigabyte de memória principal, discos IDE de 60 gigabytes e uma chave (*switch*) *Ethernet* Rápida (100Mbps).

Para a medição de energia elétrica consumida durante os experimentos, utilizamos um medidor de potência de qualidade industrial da marca *Yokogawa*, modelo *WT2010*. Esse medidor de potência se destaca em artigos científicos voltados para consumo de energia, pois tem alta precisão, alto número de amostragens por segundo e tempo de resposta pequeno. Como nossas cargas apresentam uma aceleração na sua execução (por serem simulações de cargas correspondentes a dias), necessitamos de um número alto de amostragens por segundo para validar o nosso experimento.

Além do *cluster* de servidores, 4 nodos na mesma rede local foram usadas como clientes para gerar carga para os serviços. Dependendo do serviço, o *cluster* de servidores foi configurado da seguinte maneira:

- *Servidor de Conteúdo Estático*: 4 servidores Web e 4 clientes;
- *Serviço de Comércio Eletrônico*: 2 servidores Web, 3 servidores de aplicação, 1 servidor de banco de dados e 3 clientes;
- *Serviço de Máquina de Busca*: 3 máquinas de busca e 3 clientes.

Todos os nodos usam o núcleo padrão do sistema operacional e todas as aplicações são usadas sem nenhuma modificação.

O cliente utilizado no caso do servidor de comércio eletrônico é um gerador de carga implementado em C e Java que distribui as requisições entre os servidores *web* disponíveis, já suprindo o papel do DNS com distribuição *round-robin*. Para os casos da máquina de busca e o servidor de conteúdo estático onde não há necessidade de conhecimento da sessão, o cliente utilizado é o gerador de carga `sclient` utilizado em trabalhos correlatos [Carrera *et al.*, 2002].

A análise comparativa dos servidores com ou sem gerência de energia é feita considerando a taxa de requisições atendidas e a energia consumida ao longo do tempo, para isso, calculamos a integral das curvas de taxa de requisições atendidas e consumo de energia, o que resulta na área sob cada gráfico, e depois disso, contabilizamos a relação percentual entre as áreas, tanto no caso do desempenho (taxa de requisições atendidas) como na economia de energia (potência consumida).

5.1 Máquina de Busca

O resultado e a metodologia dos experimentos com o serviço de máquina de busca está caracterizado a seguir.

5.1.1 Metodologia

A carga de trabalho utilizada compreende de 20 gigabytes de páginas web coletadas pela máquina de busca *TodoBR* [TodoBR, 1999] a partir da Internet brasileira. O conjunto de consultas foi composto por 100.256 consultas de um *log* parcial de consultas submetido pelos atuais clientes da máquina de busca *TodoBR*, com um total de 37.450 consultas únicas e 27.751 termos únicos [Calado *et al.*, 2003]. Para variar a intensidade da carga, geramos as consultas para a máquina de busca de acordo com uma taxa pré-definida seguindo uma série de escalas, primeiro aumentando a carga e depois diminuindo-a.

5.1.2 Comportamento sob cargas leves

Como mencionado anteriormente, a máquina de busca foi simulada com uma lista de requisições reais de usuários refeitas a várias taxas diferentes, seguindo uma série de degraus crescentes, seguidos por uma série de degraus decrescentes. Apesar do crescimento, a carga não foi suficiente para saturar completamente os servidores, simulando a operação sob carga normal.

A Figura 5.1 mostra o perfil da carga (séries de degraus) e a energia consumida no sistema sem o *PASys* (nesse caso, todos os nodos estavam ligados durante todo o tempo). A alta variação em um pequeno espaço de tempo na energia consumida é devida aos vários ciclos de processador necessários a busca na maioria das requisições, o que causou grandes variações na potência.

Uma vez ligado o sistema de gerência de energia (*PASys*), o comportamento do sistema mudou perceptivelmente, como pode ser visto na Figura 5.2. O *cluster* de servidores começou com um único nodo ligado; com o aumento da carga, o segundo e (posteriormente)

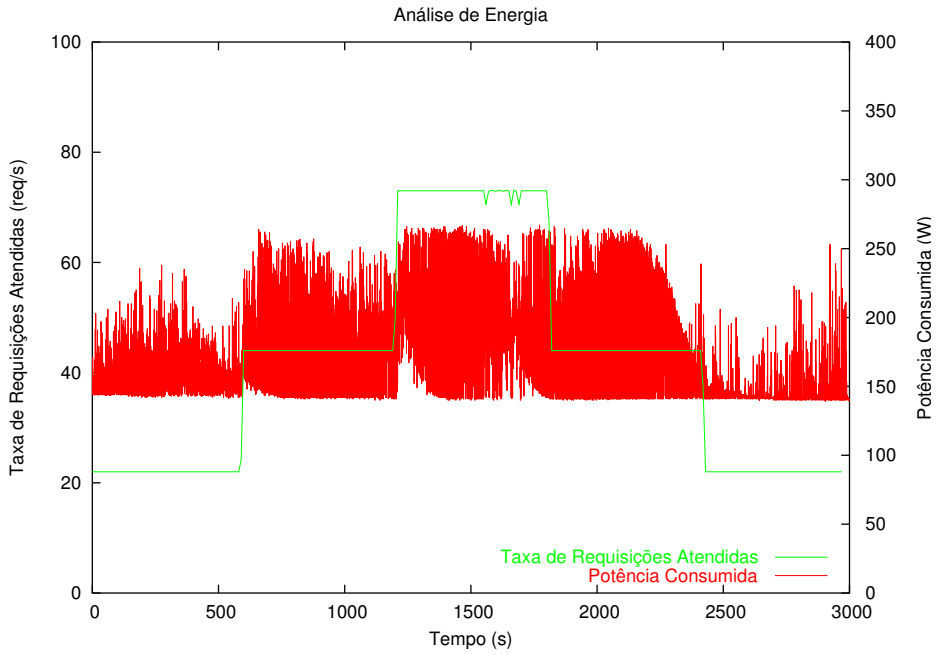

Figura 5.1: Taxa de Requisições e Energia Consumida na Máquina de Busca sem *PASys*

o terceiro nodo servidor foram ligados, provendo quase a mesma taxa de requisições atendidas no caso base. Uma vez que a carga começa a decrescer, o módulo Gerente decide desligar os dois nodos extras quando seu algoritmo de reconfiguração identifica essa possibilidade. O atraso entre o início do degrau e o crescimento do consumo de energia (o que indica uma novo nodo sendo ligado) é devido ao tempo de resposta do módulo Gerente e o refinamento das variáveis do controlador PID.

Quando nós comparamos as curvas de taxas de requisições e a energia consumida para o caso base e o sistema operando com o *PASys*, nós observamos praticamente nenhuma redução na taxa de requisições (menos que 0,1 %), mas a redução no consumo de energia é sensível, 31,6 %, o que confirma o bom funcionamento do *PASys* nesse caso.

5.2 Servidor de Conteúdo Estático

Os experimentos com o servidor de conteúdo estático estão analisados abaixo juntamente com a sua metodologia.

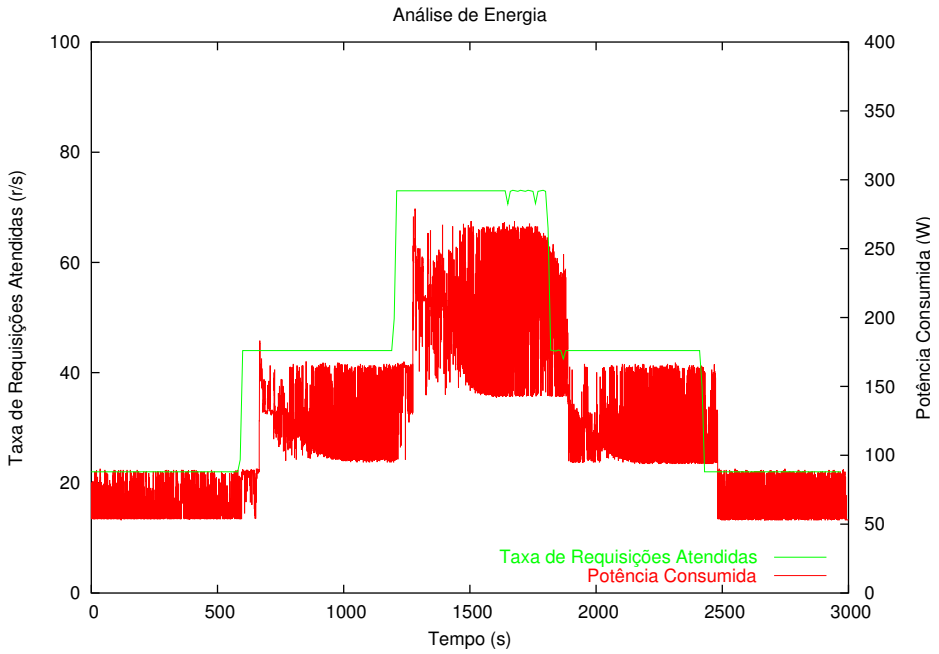

Figura 5.2: Taxa de Requisições e Energia Consumida na Máquina de Busca com *PASys* ativo

5.2.1 Metodologia

Usamos um cliente gerador de carga similar ao usado na máquina de busca. O gerador trabalha como um servidor DNS com distribuição *round-robin* e com um temporizador *heartbeat* que detecta nodos removidos.

A carga de trabalho é composta de dois dias de requisições estáticas (23 e 24 de Junho) do histórico de requisições feitas aos servidores da Copa do Mundo de 98 [Arlitt & Jin, 2000]. A carga de trabalho exibe períodos de carga baixa e alta, então nós respeitamos o tempo das requisições em nossos experimentos com esta carga. Entretanto, nós aceleramos o histórico de requisições, para que os experimentos possam implicar em maior carga imposta sob o *cluster* e serem executados em menos tempo. Esta aceleração da carga poderia ocasionar em falta de precisão em nossos resultados, caso não fizéssemos uma coleta de informações tanto de taxa de requisições atendidas como energia coletada a grão fino.

A análise do servidor *Web* de conteúdo estático foi feita sobre condições de cargas altas: a lista de requisições da Copa do Mundo foi reexecutada a uma taxa mais alta até tornar possível observar que durante uma carga de pico todos os nodos do *cluster* estavam

próximos da sua saturação.

5.2.2 Desempenho sob cargas altas

Podemos observar o comportamento do *cluster* sem o *PASys* sob cargas altas. Os resultados na Figura 5.3 são para o caso com todos os nodos ligados e sem gerência de energia.

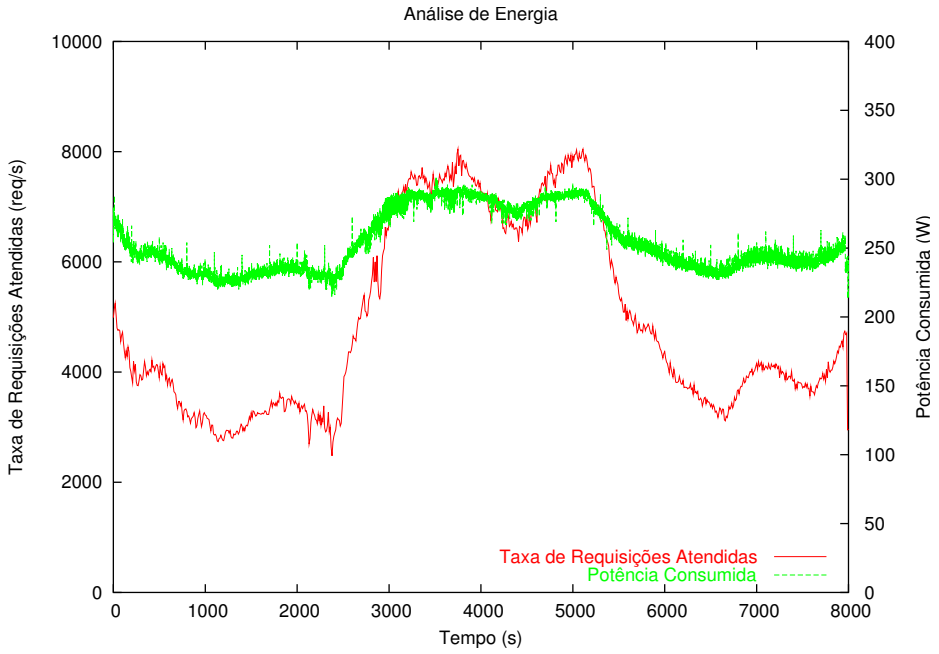

Figura 5.3: Taxa de Requisições e Energia Consumida no Servidor de Conteúdo Estático sem *PASys*

Nesse caso, é bem clara a variação da carga, de períodos de baixa atividade a períodos de carga alta. É interessante observar como a potência varia neste caso, com todos os nodos ligados. Com o aumento da carga, o consumo dos nodos devido à alta intensidade de atividade do processador e acessos a disco resultam em mais energia consumida. Em comparação com o consumo de energia do caso da máquina de busca, um servidor *Web* tem um consumo menos intensivo do processador que uma máquina de busca, portanto o gráfico de consumo de energia é mais estável.

O caso em que o *PASys* é ativado é mostrado na Figura 5.4. Nesse caso, não somente o comportamento da curva de potência não é tão estável como aquele para a máquina de

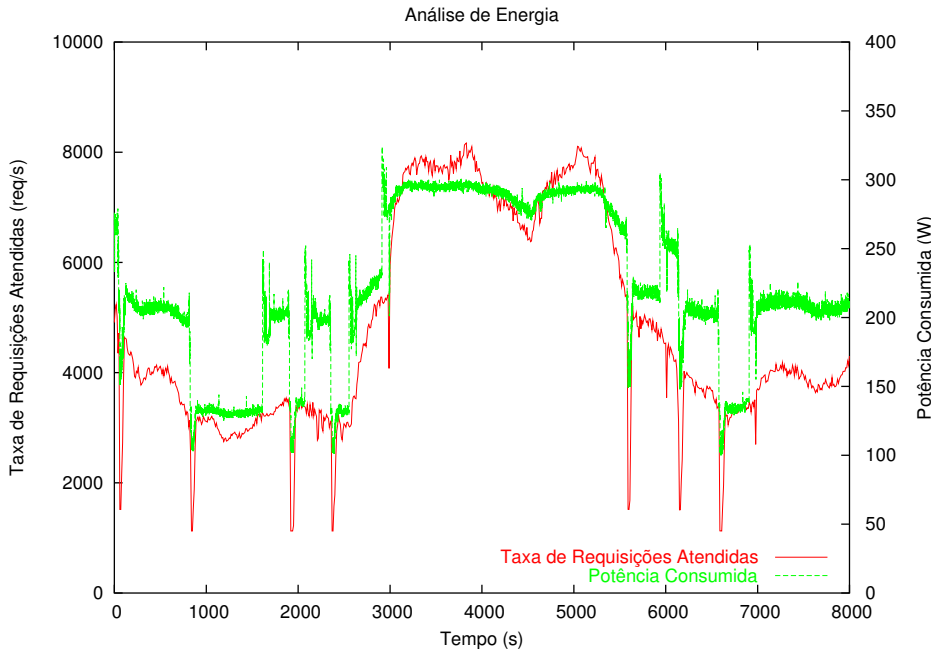

Figura 5.4: Taxa de Requisições e Energia Consumida no Servidor de Conteúdo Estático com *PASys* ativo

busca, mas também a taxa de requisições atendidas não é tão suave como o caso base (sem *PASys*).

O fato da curva de potência não seguir a carga tão de perto quanto no caso da máquina de busca é devido principalmente à natureza mais irregular da carga (derivada de uma lista de requisições reais) e à saturação dos servidores. Em alguns momentos, um crescimento na carga percebida aciona a adição de um novo nodo, entretanto a tendência não continua alta o suficiente para realmente tornar aquele nodo necessário. O controlador *PID*, sendo conservador, opta por ligar o nodo ao invés de arriscar a diminuir muito a taxa de requisições devido a uma falta de capacidade do *cluster* no caso da tendência continuar e nenhum nodo ter sido adicionado ao *cluster*. Tão logo o controlador verifica que o crescimento da taxa não continua como esperado, ele decide desligar o nodo extra novamente.

Existe também o problema dos vales muito profundos na taxa de requisições atendidas associado ao instante em que o nodo é desligado. A razão para isto é principalmente um efeito artificial do cenário de simulação. Já que não foi realmente utilizado um elemento chaveador para a distribuição de requisições, usando um gerador de carga para tal fim,

não temos controle direto sobre o processo de remoção de um nodo da lista de nodos disponíveis. Isto é feito somente quando o gerador de carga detecta (pelo seu temporizador estilo *watchdog*) que o nodo não está respondendo. Isto implica que, devido a esta simplificação, depois que um nodo é desligado o gerador de carga pode ainda considerar, por um tempo, que aquele nodo está disponível, enviando algumas requisições a ele; como não há resposta, isto é registrado pelo cliente como uma queda de desempenho.

Apesar disso, a curva de taxa de requisições atendidas não é exatamente igual ao caso base. Sob cargas altas, o atraso na decisão de ligar um nodo pode levar a uma perda no desempenho do sistema, porque por algum tempo (antes do nodo extra ser ligado e até que ele se torne operacional) o sistema deve trabalhar com uma configuração que não é a melhor configuração para lidar com esta carga.

Uma solução plausível para o caso em que o servidor é ligado e desligado sem necessidade é aumentar o número de nodos no *cluster*, então cada servidor ligado ou desligado não impactará tanto no desempenho total do sistema distribuído. Como discutiremos na próxima seção, este comportamento pode ser refinado pelo administrador do sistema através da configuração de um valor específico de limiar para cada nodo.

Enfim, quando comparamos a taxa de requisições atendidas e a energia consumida pelo caso base e o sistema com o *PASys*, nós observamos uma queda da taxa de requisições de 2,18% com uma economia de energia de 12,62%.

5.2.3 A sensibilidade ao parâmetro limiar

Como discutido previamente na seção 3.3, o elemento *Valor da Grandeza* do formato de descrição define um limiar que pode ser usado para refinar o sistema, de acordo com um tipo específico de servidor.

A Figura 5.5 mostra dois gráficos do comportamento do sistema sob a carga discutida anteriormente, quando variamos o valor do limiar. É fácil observar a diferença na forma das curvas. O primeiro gráfico mostra o comportamento do *PASys* com um limiar de 0,65, enquanto o segundo utiliza 0,90. O efeito geral é que, quanto menor o limiar, existe

a tendência de tomar decisões durante pequenas variações de carga, enquanto limiares maiores tendem atrasar a tomada de decisões até que as variações de carga sejam mais altas.

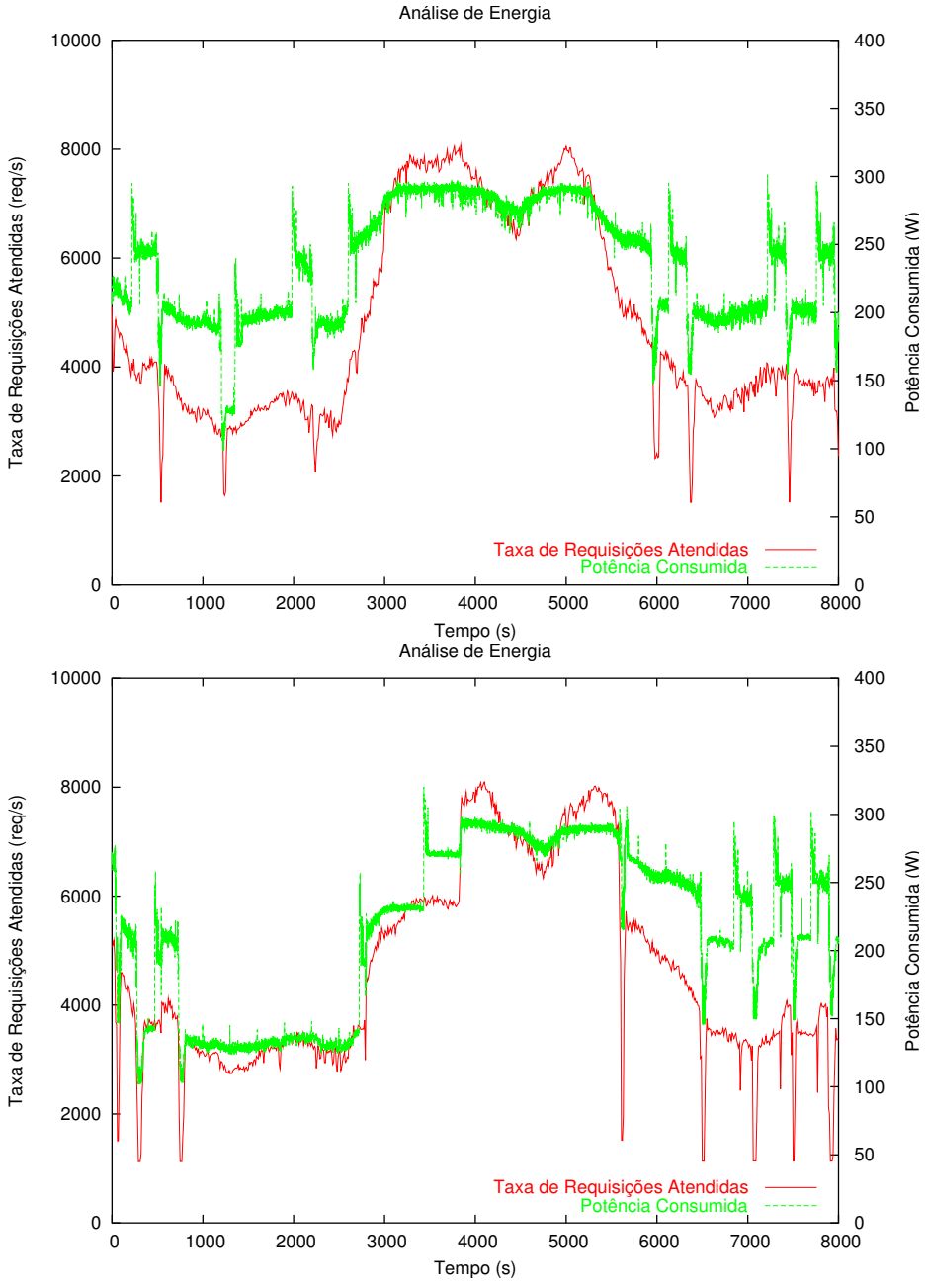

Figura 5.5: Taxa de Requisições e Energia Consumida no Servidor de Conteúdo Estático com *PASys* ativo, diferentes limiares (0,65 e 0,90)

Analisando a variação de limiar em uma aplicação específica percebemos que, como esperado, temos um compromisso entre queda na taxa de requisições e economia de energia;

a razão entre as porcentagens parece ter um ponto ótimo quando o *threshold* é 0,80 no caso do servidor de conteúdo estático (vide Figura 5.6).

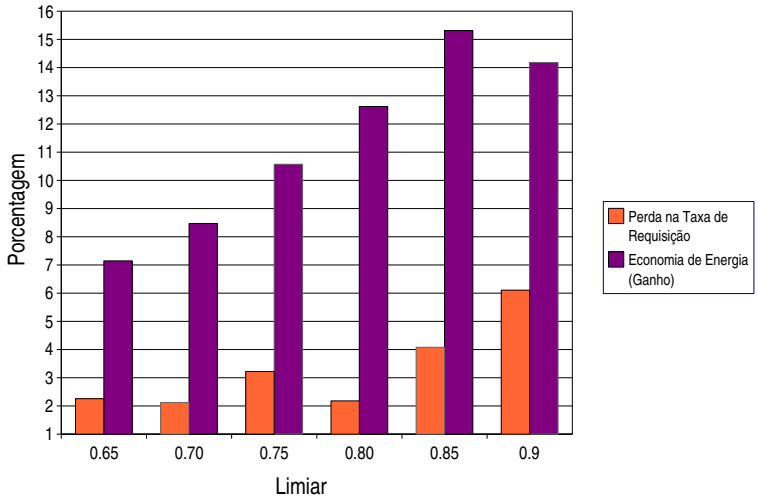

Figura 5.6: Perda na Taxa de Requisições versus Economia de Energia Consumida no Servidor de Conteúdo Estático com diferentes limiares

5.3 Servidor de Comércio Eletrônico

A seguir detalharemos os resultados obtidos no servidor de comércio eletrônico.

5.3.1 Metodologia

A carga de trabalho é baseada no *benchmark* padrão *TPC-W* para sistemas de comércio eletrônico [García & García, 2003]. A intensidade da carga experimentada por nós exibe um forma triangular, cuja carga cresce linearmente por um tempo, estabiliza em uma carga alta e, depois, decresce linearmente. Este formato de carga foi feito para as possíveis reconfigurações (ligação e desligamento de nodos) de um *cluster* serem alcançadas.

5.3.2 O efeito de servidores multicamadas

A Figura 5.7 reflete o caso base para essa aplicação sem utilizar o PASys. A potência segue nos momentos de carga baixa uma variação pequena, entretanto no pico de taxa de requisições aumenta esta variação. Utilizando o *PASys* (Figura 5.8), a taxa de requisições apresenta vários picos e vales demonstrando sobrecarga no servidor sem resposta rápida do sistema.

O servidor de comércio eletrônico apresenta ganhos de economia de energia (22,78%) com queda na taxa de requisições razoáveis (4,64%). Nesse serviço percebemos que o tempo de reconfiguração (no nosso caso, 3 minutos) é um dado bem mais sensível gerando grandes quedas no desempenho. O tempo de reconfiguração, como explicado anteriormente, é o tempo que o *PASys* aguarda até executar o algoritmo de reconfiguração após uma execução prévia. Este tempo é necessário, pois, após um acréscimo de um nodo, é esperada uma estabilização do sistema sobrecarregado.

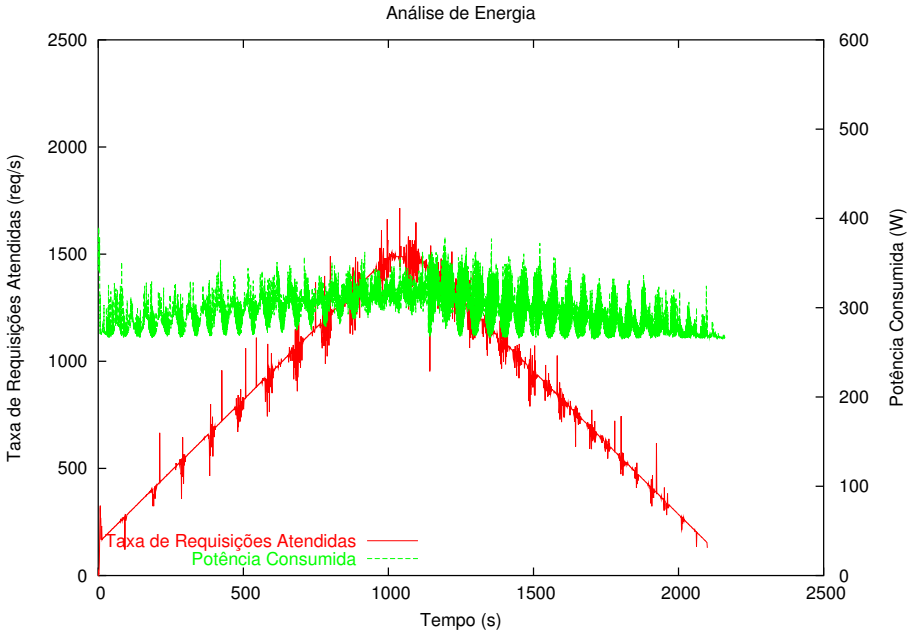

Figura 5.7: Taxa de Requisições e Energia Consumida no Servidor de Comércio Eletrônico sem *PASys*

Uma possível solução para este caso seria aumentar a diferença entre o tempo total do experimento e o tempo de reconfiguração do *cluster*, pois o tempo total desse experimento

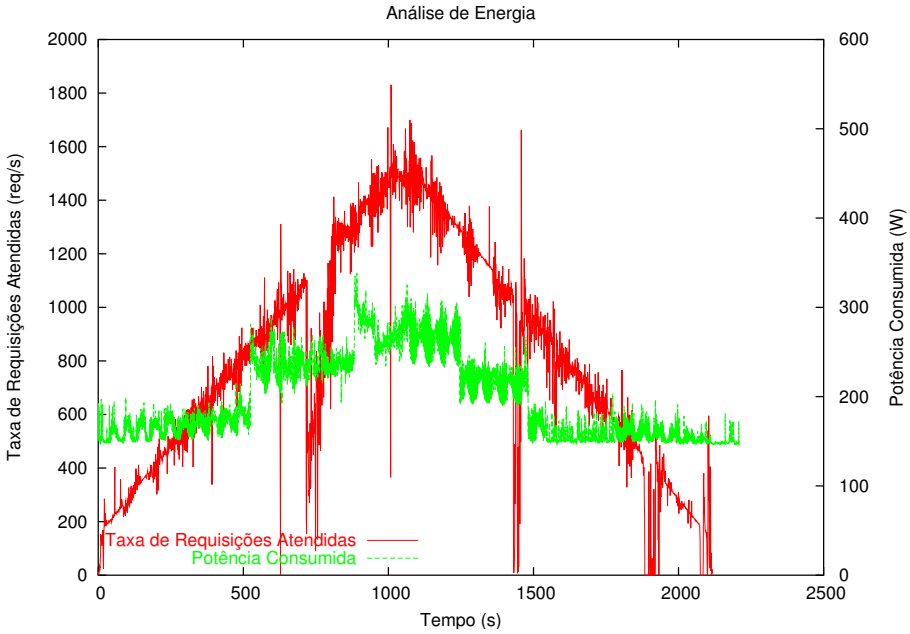

Figura 5.8: Taxa de Requisições e Energia Consumida no Servidor de Comércio Eletrônico com *PASys* ativo

foi 35 minutos e o tempo de reconfiguração foi 3 minutos, praticamente 10% do tempo total. Desse modo, quando um nodo acrescentado estivesse ábil a responder requisições, já teria passado 10% da duração do experimento desde a decisão de sua ligação e as condições do sistema podem ter se degradado excessivamente. A fim de constatar este fato, foi executado um experimento com mais de 2 horas de duração e percebeu-se uma diminuição significativa dos picos e vales (Figura 5.9), entretanto não foi possível coletar informações de potência consumida neste experimento, pois o medidor de potência não estava disponível.

5.4 Sumário

Finalmente, a tabela 5.1 apresenta a perda na taxa de requisições atendidas versus a economia de energia obtida usando o *PASys* em limiares escolhidos, mostrando claramente que o *PASys* com pequenas perdas na taxa de requisições alcança economias de energia significativas. Os limiares escolhidos foram específicos por caso de estudo e seus valores foram obtidos empiricamente.

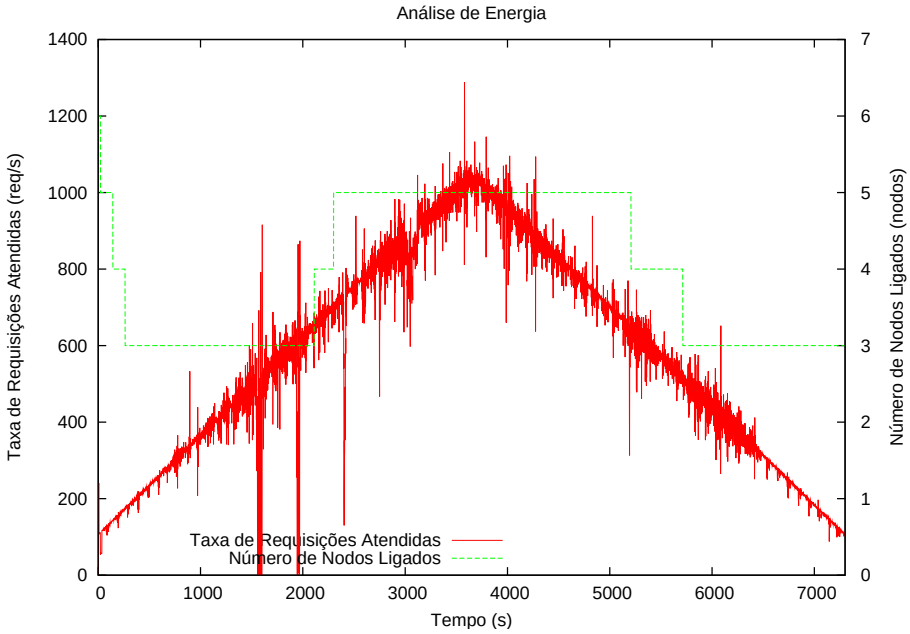

Figura 5.9: Taxa de Requisições e Número de Nodos Ligados no Servidor de Comércio Eletrônico com *PASys* ativo

	Queda na Taxa de Requisições	Economia de Energia
Servidor *Web*	2,18%	12,62%
Máquina de Busca	menos que 0,1%	31,23%
Servidor de *E-Commerce*	4,64%	22,78%

Tabela 5.1: Perda na Taxa de Requisição Atendidas versus Economia de energia em *thresholds* escolhidos

Estes resultados mostram que o *PASys* apresenta ganhos significativos, apesar dos problemas percebidos na atualização de servidores disponíveis pelos geradores de carga, que geraram os vales.

Aceleramos a carga e utilizamos um coletor de potência consumida com uma alta taxa de consultas, entretanto não encontrarmos meios de compactar o tempo de ligação de um nodo. Apesar de reconhecermos que esses experimentos não refletem exatamente a situação real, acreditamos que o controlador PID com realimentação pode ser adaptado para uma carga real em tempo de execução, caso sejam utilizados valores adequados para as constantes do controlador e o limiar (SP).

Capítulo 6

Conclusões e Trabalhos Futuros

Neste capítulo descreveremos mais sobre as conclusões obtidas no estudo de gerência de energia em *clusters* e trabalhos futuros motivados a partir da pesquisa realizada nesta dissertação.

6.1 Conclusões

Neste trabalho foi criado um sistema de gerência de energia transparente, não-intrusivo e portável chamado *PASys*. Este sistema de gerência baseia-se em uma infra-estrutura, formada por dois módulos (Monitor e Gerente) e um formato para descrição de serviços em *clusters*.

O sistema de gerência de energia *PASys* apresentou uma economia de energia significativa, quando analisado em três casos de estudos com características distintas.

O *PASys* aplicado a máquina de busca, um servidor com alto consumo de energia devido ao impacto da carga sob o processador de um nodo, apresentou economia de aproximadamente 31% de energia com um impacto mínimo no desempenho do sistema.

Quando aplicado ao servidor de páginas estáticas, podemos perceber que o desempenho do sistema de gerência sob cargas altas não impactou tanto no desempenho total do sistema, pois para a queda de somente 2,18% de taxa de requisições atendidas, o ganho foi de

aproximadamente 13% de economia de potência consumida.

O servidor de comércio eletrônico em um sistema baseado em *cluster*, apesar de ter uma resposta muito lenta a um acréscimo do nodo ao *cluster*, também teve resultados satisfatórios, pois nas camadas mais externas do *cluster* (camadas que recebem a requisição direto do cliente — primeiro a do servidor *Web* e depois a do servidor de aplicação) há muita ociosidade em grande parte do experimento. Os resultados de economia de energia foram um pouco piores que no caso de páginas estáticas (23% comparado a 13%) com praticamente o dobro de perda na taxa de requisições atendidas, apesar de que 4,64% de queda nessa taxa não seja um valor muito expressivo.

Para todos os casos de estudo analisados, o formato de descrição de serviços se mostrou abrangente e sua definição é tão simples que o administrador utiliza sem conhecimentos prévios na área de gerência de energia.

Portanto, os resultados da utilização do sistema *PASys* com a sua infra-estrutura e formato de descrição de serviços apresentam um grande compromisso no ganho da economia de energia elétrica sem alta perda de taxa de requisição de maneira transparente, não-intrusiva e portável para outras arquiteturas e sistemas.

6.2 Trabalhos Futuros

Como trabalhos futuros podemos citar a melhoria dos resultados do *PASys* com o aumento de nodos no *cluster* para suavizar o impacto do acréscimo de um nodo ao desempenho do serviço.

Os resultados com servidores multi-camadas são promissores, porém a resposta do serviço após uma reconfiguração do *cluster* é muito lenta. Seria interessante fazer um estudo mais profundo sobre os tempos de atualização da lista de servidores disponíveis no *cluster* multi-camada, aumentando assim a resposta após reconfiguração.

Como cargas simuladas de servidor de comércio eletrônico e de máquina de busca podem desconsiderar algumas anomalias reais, pretendemos utilizar o *PASys* com cargas reais ou definir um perfil para as cargas utilizadas a partir de caracterizações para estes serviços.

Máquinas de busca com tolerância a falhas presentes no mercado (conforme [Barroso *et al.*, 2003]) utilizam um índice particionado em grupos de nodos e cada grupo mantém uma replicação do índice local, isto é, usam um esquema multi-camada em máquinas de busca. Portanto o *PASys* pode explorar a capacidade de suporte a multi-camadas em um servidor organizado desta maneira.

Em *Data Centers* e provedores, o gerenciamento de energia pode ser feito utilizando a temperatura como fator de decisão na reconfiguração do *cluster* [Sharma *et al.*, 2003]. O *PASys* pode utilizar valores de temperatura como parâmetros de base para a reconfiguração.

Analisar a ampliação do formato de descrição de serviços para uma linguagem para estender o *PASys* para *clusters* heterogêneos, descrevendo as características diferentes dos nodos através dessa linguagem de descrição.

Como o sistema de controle PID com realimentação utilizado no *PASys* foi sintonizado de acordo com constantes obtidas empiricamente por trabalhos correlatos [Pinheiro *et al.*, 2001, Carrera & Bianchini, 2001] para servidores de conteúdo estático, podemos estudar métodos de sintonia de PID automáticos de acordo com o serviço utilizado.

Referências Bibliográficas

[ACPI, 1999] Advanced configuration and power interface (acpi). Intel, Hewlett-Packard, Microsoft, Phoenix e Toshiba. *http://www.acpi.info* visitado em 13/01/2006., 1999.

[APM, 1996] Advanced power management (apm) - bios interface specification, revision 1.2. Intel e Microsoft. *http://www.microsoft.com/whdc/archive/amp_12.mspx* visitado em 13/01/2006., 1996.

[Arlitt & Jin, 2000] M. Arlitt e T. Jin. Workload characterization of the 1998 world cup web site. *IEEE Network*, 14(3):30–37, junho 2000.

[Badue *et al.*, 2001] C. S. Badue, R. Baeza-Yates, B. Ribeiro-Neto, e N. Ziviani. Distributed query processing using partitioned inverted files. In *Proceedings of the 8th String Processing and Information Retrieval Symposium (SPIRE'01)*. IEEE Computer Society, 2001.

[Baeza-Yates & Ribeiro-Neto, 1999] Ricardo A. Baeza-Yates e Berthier A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[Barroso *et al.*, 2003] Luiz André Barroso, Jeffrey Dean, e Urs Hölzle. Web search for a planet: The google cluster architecture. *IEEE Micro*, 23(2):22–28, 2003.

[Bellosa *et al.*, 2003] Frank Bellosa, Andreas Weissel, Martin Waitz, e Simon Kellner. Event-driven energy accounting for dynamic thermal management. In *Proceedings of the Workshop on Compilers and Operating Systems for Low Power (COLP'03)*, setembro 2003.

[Brooks & Martonosi, 2001] David Brooks e Margaret Martonosi. Dynamic thermal management for high-performance microprocessors. In *HPCA '01: Proceedings of the 7th International Symposium on High-Performance Computer Architecture*, página 171, Washington, DC, USA, 2001. IEEE Computer Society.

[Calado *et al.*, 2003] Pável Calado, Berthier Ribeiro-Neto, Nivio Ziviani, Edleno Moura, e Ilmério Silva. Local versus global link information in the web. *ACM Trans. Inf. Syst.*, 21(1):42–63, 2003.

[Carrera & Bianchini, 2001] Enrique V. Carrera e Ricardo Bianchini. Efficiency vs. portability in cluster-based network servers. In *PPoPP '01: Proceedings of the eighth ACM SIGPLAN symposium on Principles and practices of parallel programming*, páginas 113–122, New York, NY, USA, 2001. ACM Press.

[Carrera *et al.*, 2002] Enrique V. Carrera, Srinath Rao, Liviu Iftode, e Ricardo Bianchini. User-level communication in cluster-based servers. In *HPCA*, página 275, 2002.

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

[Chase & Doyle, 2001] Jeff Chase e Ron Doyle. Balance of power: Energy management for server clusters, maio 2001.

[Chase *et al.*, 2001] Jeffrey S. Chase, Darrell C. Anderson, Prachi N. Thakar, Amin M. Vahdat, e Ronald P. Doyle. Managing energy and server resources in hosting centers. In *SOSP '01: Proceedings of the eighteenth ACM symposium on Operating systems principles*, páginas 103–116, New York, NY, USA, 2001. ACM Press.

[Chiasserini & Rao, 2000] Carla-Fabiana Chiasserini e Ramesh R. Rao. Energy efficient battery management. In *INFOCOM*, páginas 396–403, 2000.

[Clarke, 1984] D. W. Clarke. Pid algorithms and their computer implementation. *Transactions of the Institution of Measurement and Control*, 6(6):305–316, 1984.

[Douglis & Krishnan, 1995] Fred Douglis e P. Krishnan. Adaptive disk spin-down policies for mobile computers. *Computing Systems*, 8(4):381–413, outubro 1995.

[Douglis *et al.*, 1994] Fred Douglis, P. Krishnan, e Brian Marsh. Thwarting the power-hungry disk. In *USENIX Winter*, páginas 292–306, 1994.

[DynaServer, 2003] Dynaserver project - rice university. *http://www.cs.rice.edu/CS/Systems/DynaServer/index.html* visitado em 13/01/2006, 2003.

[Elnozahy *et al.*, 2002] E.N. (Mootaz) Elnozahy, Michael Kistler, e Ramakrishnan Rajamony. Energy-efficient server clusters. In *Proceedings of Power-Aware Computer Systems*, páginas 179–196, 2002.

[Fearnside, 1997] P.M. Fearnside. Greenhouse-gas emissions from amazonian hydroelectric reservoirs: The example of brazil's tucuruí dam as compared to fossil fuel alternatives. *Environment Conservation*, 24(1):64–75, 1997.

[Flinn & Satyanarayanan, 1999] Jason Flinn e M. Satyanarayanan. Energy-aware adaptation for mobile applications. In *SOSP '99: Proceedings of the seventeenth ACM symposium on Operating systems principles*, páginas 48–63, New York, NY, USA, 1999. ACM Press.

[Franklin *et al.*, 1997] Gene F. Franklin, David J. Powell, e Michael L. Workman. *Digital Control of Dynamic Systems*. Addison-Wesley, 1997.

[García & García, 2003] Daniel F. García e Javier García. Tpc-w e-commerce benchmark evaluation. *Computer*, 36(2):42–48, 2003.

[Gupta & Singh, 2003] Maruti Gupta e Suresh Singh. Greening of the internet. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architec-*

tures, and protocols for computer communications, páginas 19–26, New York, NY, USA, 2003. ACM Press.

[Halfhill, 2000] Tom R. Halfhill. Transmeta breaks x86 low power barrier. *Microprocessor Report*, 14(2):1, 9–18, 2000.

[Heath *et al.*, 2002] Taliver Heath, Eduardo Pinheiro, Jerry Hom, Ulrich Kremer, e Ricardo Bianchini. Application transformations for energy and performance-aware device management. In *IEEE PACT*, páginas 121–130, 2002.

[Lebeck *et al.*, 2000] Alvin Lebeck, Xiaobo Fan, Heng Zeng, e Carla Ellis. Power aware page allocation. *ACM SIGPLAN Notices*, 35(11):105–116, novembro 2000.

[Lorch, 1995] J. Lorch. A complete picture of the energy consumption of a portable computer. Dissertação de Mestrado, University of California at Berkeley, 1995.

[Lu *et al.*, 2002] Yung-Hsiang Lu, Luca Benini, e Giovanni De Michelli. Power-aware operating systems for interactive systems. *IEEE transactions on very large scale integration (VLSI) systems*, 10(2), abril 2002.

[Mini *et al.*, 2005] R. Mini, M. Machado, A. A. Loureiro, e B. Nath. Prediction-based energy map for wireless sensor networks. *The Ad Hoc Networks Journal PWC 2003 Special Issue*, 3(2):235–253, 2005.

[Mitchell-Jackson, 2001] J. Mitchell-Jackson. Energy needs in an internet economy: A closer look at data centers. Dissertação de Mestrado, University of California at Berkeley, julho 2001.

[Odlyzko, 2003] A. Odlyzko. Internet traffic growth: Sources and implications. In *Proceedings of ITCOM 2003*, 2003.

[Ogata, 1997] Katsuhiko Ogata. *Modern control engineering (3rd ed.)*. Prentice-Hall Inc., Upper Saddle River, NJ, USA, 1997.

[Patel *et al.*, 2002] C. D. Patel, R. Sharma, C. E. Bash, e A. Beitelmal. Thermal considerations in cooling large scale high compute density data centers. In *Proceedings of 8th ITHERM Conference*, maio 2002.

[Pinheiro *et al.*, 2001] E. Pinheiro, R. Bianchini, E. V. Carrera, e T. Heath. Load balancing and unbalancing for power and performance in cluster-based systems. In *Proceedings of the Workshop on Compilers and Operating Systems for Low Power COLP'01*, setembro 2001.

[Pinheiro *et al.*, 2002] E. Pinheiro, R. Bianchini, E. Carrera, e T. Heath. Dynamic cluster reconfiguration for power and performance. In M. Kandemir L. Benini e J. Ramanujam, editores, *Compilers and Operating Systems for Low Power*. Kluwer Academic Publishers, 2002.

[Pinheiro *et al.*, 2003] E. Pinheiro, R. Bianchini, E. V. Carrera, e T. Heath. *Dynamic Cluster Reconfiguration for Power and Performance*, capítulo 5, páginas 75–93. Kluwer Academic, setembro 2003.

[PMU, 1996] Power management unit (pmu), 1996.

[Salton & Bucckley, 1988] G. Salton e C. Bucckley. Term-weighting approaches in automatic retrieval. *Information Processing & Management*, 24(5), 1988.

[Salton, 1989] G. Salton. *Automatic Text Processing: The Transformation, Analisys, and Retrieval of Information by Computer*. Addison Wesley, 1989.

[Seborg *et al.*, 1989] D. E. Seborg, T. F. Edgar, e D. A. Mellichamp. *Process Dynamics and Control*. John Wiley & Sons, 1989.

[Semeraro *et al.*, 2002] G. Semeraro, G. Magklis, R. Balasubramonian, D. Albonesi, S. Dwarkadas, e M. Scott. Dynamic frequency and voltage control for a multiple clock domain microarchitecture. In *Proceedings of International Symposium on Microarchitecture (MICRO)*, páginas 356–367, 2002.

[Sharma *et al.*, 2003] Ratnesh K. Sharma, Cullen E. Bash, Chandrakant D. Pateland, Richard J. Friedrich, e Jeffrey S. Chase. Balance of power: Dynamic thermal management for internet data centers. Relatório Técnico HPL-2003-5, HP Labs, fevereiro 2003.

[Singh & Raghavendra, 1999] S. Singh e C. Raghavendra. PAMAS: Power aware multi-access protocol with signalling for ad hoc networks. In *Proceedings of ACM Computer-Communications Review*, 1999.

[TodoBR, 1999] Todobr. Google América Latina. *http://www.todobr.com.br visitado em 13/01/2006.*, 1999.

[Van Rijsbergen, 1979] C. J. Van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

[Warren *et al.*, 2002] Michael S. Warren, Eric H. Weigle, e Wu-Chun Feng. High-density computing: a 240-processor beowulf in one cubic meter. In *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, páginas 1–11. IEEE Computer Society Press, 2002.

[Weiser *et al.*, 1994] Mark Weiser, Brent Welch, Alan Demers, e Scott Shenker. Scheduling for reduced CPU energy. In USENIX, editor, *Proceedings of the First USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, páginas 13–23, Berkeley, CA, USA, novembro 1994. USENIX.

[Yang, 2004] Hongbo Yang. *Power-Aware Compilation Techniques for High Performance Processors*. Tese de Doutorado, University of Delaware - USA, 2004.

Apêndice A

Arquivos de Definição

Para os experimentos, os arquivos do formato de descrição dos serviços estão a seguir com suas respectivas explicações.

As três linhas do arquivo de descrição representam as três máquinas de busca. Todas pertencem a mesma camada e são desligáveis. O *setpoint* para refinamento do experimento feito foi 70% de carga.

```
192.168.64.2|1|Y|0.7
192.168.64.3|1|Y|0.7
192.168.64.4|1|Y|0.7
```

No próximo arquivo, existiram quatro servidores de conteúdo estático, todos pertencentes a mesma camada, desligáveis e o *setpoint* foi o mesmo de 70% de carga.

```
192.168.64.2|1|Y|0.7
192.168.64.3|1|Y|0.7
192.168.64.4|1|Y|0.7
192.168.64.5|1|Y|0.7
```

No último arquivo, no caso do servidor de comércio eletrônico, existem três camadas. Na primeira camada, existem dois servidores de conteúdo estático e todos desligáveis. Na segunda camada, existem três servidores de aplicação e todos desligáveis. Na terceira camada, existe um servidor de banco de dados que não é desligável. Foi usado o mesmo *setpoint* para todos os nodos deste *cluster*, apesar da terceira camada não precisar de valor de *setpoint*, já que não tem nodo desligável.

```
192.168.64.2|1|Y|0.7
192.168.64.3|1|Y|0.7
192.168.64.4|2|Y|0.7
192.168.64.5|2|Y|0.7
192.168.64.6|2|Y|0.7
192.168.64.7|3|N|0.7
```