VOLTAIC : UM SISTEMA DE GERÊNCIA AUTOMATIZADA DE RECURSOS DE NÓS VIRTUAIS PARA COMPUTAÇÃO EM NUVENS

Hugo Eiji Tibana Carvalho

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Otto Carlos Muniz Bandeira Duarte

Rio de Janeiro
Dezembro de 2012

VOLTAIC : UM SISTEMA DE GERÊNCIA AUTOMATIZADA DE RECURSOS DE NÓS VIRTUAIS PARA COMPUTAÇÃO EM NUVENS

Hugo Eiji Tibana Carvalho

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Otto Carlos Muniz Bandeira Duarte, Dr. Ing.

Prof. Alfredo Goldman vel Lejbman, Dr.

Prof. Artur Ziviani, Dr.

Prof. Luís Henrique Maciel Kosmalski Costa, Dr.

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2012

Eiji Tibana Carvalho, Hugo

VOLTAIC : um Sistema de Gerência Automatizada de Recursos de Nós Virtuais para Computação em Nuvens/Hugo Eiji Tibana Carvalho. – Rio de Janeiro: UFRJ/COPPE, 2012.

XIV, 86 p.: il.; 29, 7cm.

Orientador: Otto Carlos Muniz Bandeira Duarte

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2012.

Referências Bibliográficas: p. 80 – 86.

1. Internet do Futuro. 2. Computação em Nuvens. 3. Qualidade de Serviço. 4. Elasticidade. I. , Otto Carlos Muniz Bandeira Duarte. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

À minha família.

Agradecimentos

Agradeço aos meus pais, avós, tios e primos por toda a dedicação, esforço, apoio e compreensão durante a minha formação acadêmica. Agradeço em especial aos meus pais Marcos de Castro Carvalho e Mauriza Tibana, que sempre foram referências no que tange a humildade, o caráter, a ética, o respeito ao próximo e a dedicação aos estudos.

Agradeço aos meus amigos da pós-graduação Pedro Pisa, Diogo Menezes, Lyno Ferraz, Rodrigo Couto, Pedro Coutinho, Natalia Castro, Daniel Dias, João Vitor, Marcel Silva, André Mendes, Celso Carvalho, Raphael Melo e aos alunos de graduação Alessandra Yoko, Leonardo Cardoso, Daniel Neto, Lucas Henrique, Renan Lage, Filipe Barretto e Victor Pereira pelo apoio, participação e disponibilidade durante o desenvolvimento desta dissertação de mestrado e pelos momentos felizes que vivenciamos no Grupo de Teleinformática e Automação.

Agradeço aos professores do Programa de Engenharia Elétrica (PEE), aos professores do Programa de Engenharia de Sistemas e Computação (PESC) e ao meu orientador Otto Duarte pelas lições de vida vividas e aprendidas. Agradeço aos professores Alfredo Goldman, Artur Ziviani e Luís Henrique pela participação na banca examinadora.

Ao CNPq, CAPES, FUNTELL, FAPERJ, UOL e FINEP pelo financiamento da pesquisa.

Por fim, agradeço a todos que de alguma forma contribuíram na minha formação e em todo o aprendizado ocorrido durante o mestrado.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

VOLTAIC : UM SISTEMA DE GERÊNCIA AUTOMATIZADA DE RECURSOS DE NÓS VIRTUAIS PARA COMPUTAÇÃO EM NUVENS

Hugo Eiji Tibana Carvalho

Dezembro/2012

Orientador: Otto Carlos Muniz Bandeira Duarte

Programa: Engenharia Elétrica

A computação em nuvem fornece acesso sob demanda a recursos computacionais. Um grande desafio do ambiente de nuvens é garantir a elasticidade dos processos que executam na nuvem, evitar violações de acordos de níveis de serviço (*Service Level Agreement* – SLA) e utilizar de forma eficiente os recursos. Esta dissertação de mestrado propõe o VOLTAIC (*Volume Optimization Layer To AssIgn Cloud resources*), um sistema de gerência automatizada de recursos para computação em nuvens. A proposta emprega perfis de uso de elementos físicos e virtuais e tentam garantir uma melhor alocação de recursos. Além disso, a proposta introduz algoritmos que determinam as formas mais adequadas de se alocar elementos na nuvem e de migrar automaticamente elementos virtuais para evitar que a saturação de recursos provoque perda de desempenho. Os resultados obtidos através de uma implementação em um pequeno conjunto de máquinas demonstram que o sistema distribui eficientemente a carga entre os servidores físicos e garante o funcionamento adequado dos elementos virtuais. Um simulador de ambientes virtuais foi desenvolvido para aplicar o funcionamento da proposta em cenários de maior escala. Os resultados mostram que o sistema reduz em mais de 30% a taxa de recusa na oferta de recursos de processamento.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

VOLTAIC: VOLUME OPTIMIZATION LAYER TO ASSIGN CLOUD RESOURCES

Hugo Eiji Tibana Carvalho

December/2012

Advisor: Otto Carlos Muniz Bandeira Duarte

Department: Electrical Engineering

Cloud computing offers on-demand access to computational resources. One of the major challenges in cloud environments is to enforce the elasticity of the processes that execute in the cloud, avoiding Service Level Agreement (SLA) violations and reducing waste with idle resources. We propose an autonomic resource management system for cloud computing, called VOLTAIC (Volume Optimization Layer To AssIgn Cloud resources). The proposal analyzes usage profiles of physical and virtual elements and tries to guarantee an optimized organization of virtual elements. The proposal includes algorithms to determine methods to properly allocate cloud elements and to automatically migrate those elements to avoid decrease in performance due to server saturation. Results obtained through an implementation in a small group of machines show that the system assigns virtual elements efficiently and ensures proper resource allocation to virtual elements. We also have developed a virtual network simulator for cloud environments to attest the applicability of our proposal in broader scenarios. Results show that VOLTAIC reduces in up to 30% the amount of wasted cycles due to correct assignment of virtual elements.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ACPI	*Advanced Configuration and Power Interface* - Interface Avançada de Configuração e Energia, p. 15
AMT	*Active Management Technology*, p. 17
API	*Application Programming Interface*, p. 4, 9, 19, 23
AP	Analisador de Perfis, p. 43, 44, 63
CDF	*Cumulative Distribution Function*, p. 44
CE	Coletor de Estatísticas, p. 43, 44, 63
CLR	*Common Language Runtime*, p. 20
CPU	*Central Processing Unit*, p. 17, 20, 34
CaaS	*Communication as a Service* - Comunicação como Serviço, p. 10, 62
DaaS	*Data as a Service* - Armazenamento como Serviço, p. 10, 62
E/S	Entrada e Saída, p. 24
EP	Elastic Processing, p. 25
HPC	*High Performance Computing*, p. 24
HaaS	*Hardware as a Service* - *Hardware* como Serviço, p. 11, 62
IP	*Internet Protocol*, p. 17
IaaS	*Infrastructure as a Service* - Infraestrutura como Serviço, p. 10, 26, 38, 62
OQ	Orquestrador, p. 43, 63
OSI	*Open Systems Interconnection*, p. 7

PDF *Probability Distribution Function*, p. 44

PO Perfil Oferecido, p. 43, 45, 49

PRESS *PRedictive Elastic ReSource Scaling for cloud systems*, p. 36

PaaS *Platform as a Service* - Plataforma como Serviço, p. 9, 10, 26, 62

QoS *Quality of Service* - Qualidade de Serviço, p. 2, 4

RAM DDR3 *Random Access Memory Double Data Rate 3*, p. 52

SEDF *Scan Earliest Deadline First*, p. 32

SLA *Service Level Agreement* - Acordos de Nível de Serviço, p. 2, 3, 7, 10, 19, 36, 40, 62

SLO *Service Level Objective* - Objetivos de Nível de Serviço, p. 36

SaaS *Software as a Service* - Software como Serviço, p. 8–10, 20, 26, 62

TCP *Transmission Control Protocol*, p. 17

Ti *Tecnologia da Informação*, p. 5

UDP *User Datagram Protocol*, p. 16

VMM *Virtual Machine Monitor* - Monitor de Máquinas Virtuais, p. 13, 14

VOIP Voice Over Internet Protocol, p. 10

VOLTAIC *Volume Optimization Layer To AssIgn Cloud resource*, p. 36, 42, 46, 53

WOL *Wake On Lan*, p. 16

Capítulo 1

Introdução

A computação em nuvem fornece acesso sob demanda a recursos computacionais. O modelo de nuvens se assemelha ao modelo dos computadores de grande porte (*mainframes*) da década de setenta onde um único *pool* ou repositório de recursos, no caso o *mainframe*, é compartilhado por diversos usuários que o acessam através de terminais simples. No contexto atual, existem repositórios de recursos espalhados por *data centers* conectados à Internet e os usuários interagem com esses recursos através da rede. Desta forma, as nuvens podem ser definidas como grandes repositórios de recursos virtualizados (*hardware*, plataformas de desenvolvimento e serviços), facilmente utilizados e acessíveis. A grande vantagem deste modelo é que a alocação desses recursos pode ser feita de forma dinâmica e sob demanda, dependendo da necessidade dos usuários. Esta dinamicidade é definida como elasticidade.

Ao desenvolver serviços novos ou migrar serviços existentes para as nuvens, surgem obstáculos que podem prejudicar o funcionamento das nuvens. Estes obstáculos envolvem a disponibilidade do serviço, a segurança dos dados armazenados, os gargalos de desempenho e de latência, a depuração de sistemas distribuídos e a coerência dos dados. Além desses obstáculos, um grande desafio do ambiente de nuvens é garantir a elasticidade dos processos que executam na nuvem e, ao mesmo tempo, evitar violações de acordos de níveis de serviço (*Service Level Agreement* - SLA) e utilizar os recursos de forma eficiente. Ou seja, a nuvem deve atender a demandas variáveis de recursos dos clientes sem que um cliente interfira e prejudique o desempenho dos outros.

Assim, enquanto a computação tradicional é baseada na oferta de recursos computacionais na forma de produtos de *hardware* e *software* que são negociados com clientes, a computação em nuvem estende este conceito e permite que recursos computacionais sejam oferecidos na forma de serviços. Os clientes da computação em nuvem podem receber a concessão de uso de recursos remotos, sem necessariamente adquirir o recurso. Desta forma, abre-se uma enorme quantidade de possibilidades de computação e de desafios, que serão aprofundados nesta dissertação.

A computação em nuvem introduz, portanto, um novo modelo de fornecimento de infraestrutura tecnológica. Neste modelo, clientes contratam provedores de recursos que podem oferecer dinamicamente processamento, memória, armazenamento e rede. Esta oferta dinâmica de recursos é obtida através da tecnologia de virtualização [1] que implementa uma abstração do *hardware* fornecendo assim flexibilidade na alocação de recursos. Esta maior flexibilidade confere elasticidade no ambiente de nuvens, definida como a capacidade de provisão de recursos sob demanda que garanta a Qualidade de Serviço (*Quality of Service* - QoS) acordada com os clientes [2].

O desenvolvimento de soluções para nuvens de computadores é uma área ativa de pesquisa e grandes empresas já fornecem soluções proprietárias. A EMC [3], a Amazon [4], a Microsoft [5] e o Google [6] são alguns exemplos de empresas que investem em soluções para a computação em nuvens. Ao mesmo tempo, a comunidade *open source* desenvolve soluções abertas que suportam nuvens, como por exemplo o projeto OpenNebula [7] e o Eucalyptus [8]. Para que estas soluções sejam competitivas, os provedores de serviço precisam desenvolver sistemas de nuvem eficientes que evitem o desperdício de recursos, garantam a qualidade do serviço e atendam a demandas variáveis de recursos.

Armbrust *et al.* [9] afirmam que os principais desafios da computação em nuvens são a manutenção da disponibilidade dos serviços e a oferta elástica e eficiente de recursos que escale de acordo com a demanda e reduza custos sem violar os acordos de nível de serviço (*Service Level Agreements* – SLAs). Os SLAs podem ser definidos como contratos formais existentes entre os provedores de serviço e os clientes que estabelecem regras que devem ser cumpridas por ambas as partes para validar a prestação e a utilização do serviço. Como exemplo de SLA, pode-se requerer do provedor que ele garanta a disponibilidade dos servidores em 99,9% do tempo e garanta que o cliente poderá utilizar dois processadores e uma quantidade máxima de memória. O cliente por sua vez não pode utilizar o serviço para prática de atividades ilícitas.

Para os provedores conseguirem garantir os SLAs e ao mesmo tempo evitar o desperdício de recursos, eles precisam gerenciar de forma eficiente os seus recursos. Por exemplo, os provedores podem contratar administradores que monitoram as cargas dos servidores e verificam se os contratos estão sendo cumpridos e em caso de problema, intervenham e modifiquem as configurações das máquinas. Esta abordagem funciona para cenários de pequena escala. As nuvens são compostas por milhares de equipamentos heterogêneos e uma gerência manual não possui o tempo de resposta necessário para corrigir problemas neste tipo de cenário. Desta forma, os administradores de nuvens precisam de um ferramental capaz de auxiliá-los no processo de tomada de decisão nas nuvens, que diminua a carga sobre o operador

humano e diminua o tempo de reação do sistema para minimizar violações nos SLAs.

Uma forma de reação possível no ambiente de nuvens é a utilização da virtualização que permite o remapeamento dinâmico de recursos virtuais sobre recursos físicos, de forma a permitir a adaptação dos sistemas a cargas de trabalho dinâmicas. Essa primitiva de remapeamento é definida como migração e permite a transferência de cargas de trabalho entre diferentes máquinas físicas sem a sua interrupção. A migração normalmente é feita de forma manual pelos gerentes de rede para balancear a carga em *data centers* [10]. Essa forma de realocação manual não é eficiente pois o seu tempo de reação é insuficiente para atender a ambientes dinâmicos com mudanças bruscas de carga de trabalho, como as nuvens de computadores. A migração autônoma é um desafio ainda maior, pois deve-se considerar a variação de múltiplos parâmetros de cada uma das máquinas e estimar demandas futuras de consumo de recursos.

A computação em nuvem é, portanto, uma área desafiadora que apresenta uma série de oportunidades de pesquisa. Torna-se necessário desenvolver mecanismos que automatizem processos de gerência e aumentem a eficiência dos provedores. Para automatizar este processo, desenvolvem-se soluções que tentam controlar os parâmetros do sistema, prever comportamentos futuros e baseados nestas previsões tomar decisões como a de realocar um dado recurso. A solução proposta nesta dissertação tem o objetivo de desenvolver mecanismos autônomos de gerência que aumentam a eficiência dos provedores de nuvens.

1.1 Objetivos

O objetivo desta dissertação é desenvolver soluções para a gerência eficiente de máquinas em servidores de nuvens que permitam a automatização das tarefas de migração de servidores. Esta gerencia automatizada deve garantir o atendimento dos requisitos dos clientes e ao mesmo tempo maximizar a utilização dos recursos disponíveis.

Esta dissertação de mestrado propõe o sistema VOLTAIC que realiza migrações autônomas para prover elasticidade aos recursos oferecidos em um ambiente de nuvens, aumentando a QoS oferecida aos clientes e melhorando o aproveitamento dos recursos disponíveis. Através da análise dos perfis de uso dos elementos físicos e virtuais, o sistema realiza uma alocação dinâmica dos elementos de forma a garantir a elasticidade na provisão de recursos e a alocação eficiente da quantidade de recursos necessária para atender a qualidade de serviço pedida pelos clientes. Os perfis de uso de elementos virtuais são comparados aos perfis de outros elementos virtuais que compartilham o mesmo servidor físico e também com os perfis de recursos que

podem ser oferecidos pelos outros servidores físicos. Busca-se o servidor físico mais adequado para hospedar um dado elemento virtual, levando-se em consideração a heurística de verossimilhança entre o perfil de uso de recursos do elemento virtual com o perfil de recursos oferecido pelos servidores físicos.

O VOLTAIC foi implementado e testado em um ambiente real e utiliza a API Libvirt [11] e, portanto, independe da plataforma de virtualização podendo ser aplicado em ambientes de nuvem heterogêneos como ambientes Xen [12], VMWare [13], KVM [14], etc. Para validar o funcionamento do sistema proposto em ambientes de maior escala, foi desenvolvido um simulador de ambientes virtuais em nuvem. O simulador recebeu o padrão de consumo das máquinas reais da implementação e gerou saídas que validam a modelagem feita para o cenário proposto. Os resultados obtidos mostram que o VOLTAIC é eficiente na provisão de elasticidade e aumenta a disponibilidade de recursos. A utilização da proposta reduz em mais de 10% a taxa de recusa na oferta de recursos de processamento.

1.2 Organização da Dissertação

Este trabalho está organizado da seguinte forma. O Capítulo 2 apresenta os trabalhos relacionados e o estado da arte da pesquisa na área de computação em nuvens. São exploradas as principais propostas de definição de nuvens, as diferentes modalidades de serviço oferecidas na nuvem, o custo de manutenção e um estudo sobre as falhas em ambientes de nuvem. O Capítulo 3 apresenta a área de provisão de elasticidade na nuvem e os trabalhos relacionados à gerência autônoma de recursos em nuvens. Avalia-se o estado da arte na gerência de recursos na nuvem e as tecnologias e algoritmos que permitem a utilização eficiente dos recursos da nuvem. O Capítulo 4 apresenta o funcionamento da lógica nebulosa e como ela é utilizada no escopo desta dissertação. No Capítulo 5 é apresentado o VOLTAIC, o sistema proposto. São apresentados os módulos principais do sistema, o funcionamento de cada um deles e os algoritmos propostos. No Capítulo 6 é apresentado uma análise dos parâmetros de regulagem do sistema e os resultados obtidos. Por fim, o Capítulo 7 apresenta as conclusões e direções futuras do trabalho.

Capítulo 2

Computação em Nuvens

A computação em nuvem está associada a um novo paradigma para o fornecimento de infraestrutura de tecnologia da informação (TI). Este novo paradigma se assemelha ao modelo dos *mainframes* da década de setenta onde um único *pool* ou repositório de recursos, no caso o *mainframe* é compartilhado por diversos usuários que o acessam através de terminais simples. A computação em nuvem retoma este modelo de delegar serviços e tarefas para servidores remotos [15]. Serviços tais como: editores de texto, planilhas, e-mail, calendários, chats, vídeos, agendas, arquivos pessoais, etc. são oferecidos na nuvem aos usuários finais, como mostra a Fig. 2.1. As nuvens atuais permitem que todos estes serviços sejam entregues de forma transparente e através de qualquer dispositivo para os usuários.

Figura 2.1: Serviços oferecidos na nuvem.

A computação na nuvem também estende o conceito de servidores compartilhados, pois além do acesso aos recursos remotos, pode-se alocar dinamicamente recursos de processamento, memória, disco e rede. Este cenário de alocação dinâmica é possível devido principalmente a utilização de tecnologias como a virtuali-

zação de computadores [1] que permite a abstração dos recursos de *hardware*. Para os clientes da nuvem, torna-se fácil o processo de alugar recursos e utilizá-los de acordo com a demanda. Dentre as diversas vantagens de se alugar recursos em uma nuvem podem-se citar a maior facilidade de se instalar, configurar e atualizar sistemas, gerenciar infraestruturas computacionais e controlar dinamicamente a escala e a capacidade computacional fornecida [16]. Uma classificação amplamente aceita de nuvem de computadores é a que decompõe a nuvem em sete características fundamentais. Existem três modelos de serviço e quatro abordagens de implantação [17]. Os modelos são definidos como o provimento de infraestrutura como serviço, plataforma como serviço e *software* como serviço. A implantação dos modelos pode definir uma nuvem como pública, privada, comunitária e híbrida [15].

A computação em nuvem fornece estas funcionalidades e ao mesmo tempo provê desafios que podem limitar a aplicação deste paradigma. Estes desafios são grandes incentivos para a inovação e permitem que os pesquisadores possuam um vasto campo de estudos e desenvolvam modelos de nuvem capazes de lidar com estes problemas de forma inteligente. Dentre os principais desafios das nuvens, pode-se citar a disponibilidade e a gerência de um sistema distribuído complexo, a dificuldade de se padronizar as interfaces entre diferentes provedores de nuvem, a privacidade de dados, a elasticidade eficiente, a auditoria das ações feitas na nuvem e o provimento eficiente de Qualidade de Serviço nas nuvens, como pode ser visto em [18].

2.1 Principais Características das Nuvens

As nuvens de computadores podem ser definidas de formas distintas [19][20][21]. A definição simples e precisa de nuvem ainda é um desafio. Uma definição é que nuvem é um repositório de recursos virtualizados que são alocados dinamicamente para ofertar serviços sob demanda. No entanto, diversos autores exprimem diferentes visões sobre a nuvem. Existem estudos como o visto em Vaquero *et al.* [21] que objetivam uma definição completa sobre o que é uma nuvem de computadores. Vaquero realiza uma análise da literatura sobre nuvens de computadores, avaliando as características mais citadas e os termos utilizados com maior frequência na definição das nuvens. Os autores chegam a uma definição de nuvens que as conceituam como grandes repositórios de recursos virtualizados (*hardware*, plataformas de desenvolvimento e serviços), facilmente utilizados e acessíveis. Estes recursos podem ser dinamicamente reconfigurados para se ajustar a cargas variadas (elasticidade) e permitem uma utilização ótima dos recursos. Este repositório de recursos é tipicamente explorado através de um modelo de pague-pelo-que-foi-utilizado no qual garantias são oferecidas pelo provedor de infraestrutura na forma de acordos de níveis de serviço (SLAs) personalizados. Além disso, os autores definem que o con-

Tabela 2.1: Características das nuvens de computadores (Adaptado de [21])

#	Característica
1	Amigabilidade
2	Virtualização
3	Centralização na Internet
4	Variedade de recursos
5	Escalabilidade ou elasticidade
6	Otimização no uso de recursos
7	Pagamento conforme o uso
8	Acordos de nível de serviço para serviços
9	Acordos de nível de serviço para infraestrutura

junto mínimo de funcionalidades fornecidas e que melhor se assemelham as nuvens são a elasticidade, o modelo de pagamento por demanda e a virtualização. Por fim, os autores enumeram quais são as características que de fato definem a nuvem, conforme pode ser visto na Tabela 2.1. Existe a ideia de que a nuvem é centrada na Internet, apresenta uma interface amigável para os clientes, permite a elasticidade no fornecimento de recursos e precisa garantir acordos de níveis de contratos tanto no nível de infraestrutura quanto no nível de serviços.

2.2 Ontologia de Nuvens e Níveis de Serviço Oferecidos

As nuvens de computadores herdam características de diversas áreas da computação. O estado da arte da pesquisa em redes de computadores demonstra a ausência de um modelo de classificação de sistemas de nuvens, suas correlações e interdependências. Alguns autores propõem o desenvolvimento de uma ontologia que defina a computação em nuvens [22]. A ontologia proposta é baseada no conceito de composição, no qual a computação em nuvem é classificada em camadas hierárquicas e, portanto, uma camada usa os serviços oferecidos das camadas inferiores, de forma semelhante à modelagem em camadas do modelo *Open Systems Interconnection*(OSI) em redes de computadores.

2.2.1 Classificação dos Sistemas de Nuvens

A ontologia para nuvens classifica os sistemas de nuvem em cinco camadas: aplicação, ambientes de *software*, infraestrutura de *software*, *kernel* de *software* e *hardware*. Esta ontologia pode ser observada na Fig. 2.2.

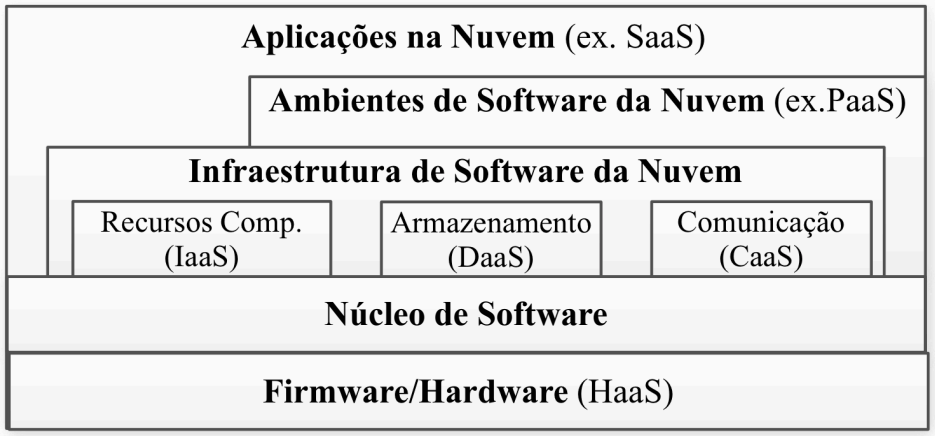


Figura 2.2: Ontologia para definição de computação em nuvens.

Camada de Aplicação

A camada de aplicação é a camada acessível para os usuários finais da nuvem. Estes usuários podem acessar os serviços fornecidos através de portais web e o modelo de negócio desta camada geralmente envolve o pagamento de taxas pela utilização do serviço. Este modelo é atraente para os usuários finais, pois ele reduz a complexidade das tarefas de manutenção de *software* e os custos operacionais e de suporte. O poder computacional associado aos serviços se encontra na nuvem, diminuindo os requisitos de *hardware* das máquinas pelas quais os usuários finais interagem com o serviço. Além disso, pode-se obter um grande poder computacional sem a necessidade de grandes investimentos na infraestrutura local dos usuários. Para os provedores de serviço o *software* está implantado em uma infraestrutura sob seu domínio, de forma que a realização de atualizações pode ser facilitada e a aplicação de direitos de propriedade intelectual são facilmente preservados, quando comparado a um *software* de prateleira. Este modelo também é conhecido como *Software* como Serviço (*Software as a Service* - SaaS). Dentre os exemplos de SaaS, podem ser citados o Google com a plataforma Google Apps, o Dropbox para armazenamento de arquivos e o Zoho para gerência de empresas.

A camada de aplicação apresenta desafios de implementação que dificultam a sua adoção em larga escala. Os desafios principais estão relacionados à segurança e à disponibilidade das aplicações. A segurança concerne principalmente o armazenamento de dados confidenciais nestas aplicações e a autenticação e autorização dos usuários com relação a estes dados. A disponibilidade está relacionada à capacidade da nuvem estar sempre disponível e os usuários finais possuírem conectividade até a nuvem em todos os instantes. Assim, o serviço oferecido na nuvem, ou seja remoto, possui diversos fatores que podem afetar o funcionamento adequado dos ser-

viços, enquanto no modelo de oferta de serviço local, a qualidade do serviço depende exclusivamente das condições locais dos usuários.

Camada de Ambientes de *Software*

A camada de ambientes de *software* é a segunda camada da ontologia. Os usuários dos serviços oferecidos por esta camada, camada superior, são os desenvolvedores de aplicações de nuvem. Os provedores dos serviços a serem usados por esta camada, serviços da camada inferior, oferecem aos desenvolvedores um arcabouço de programação que possui APIs bem definidas que facilitam a interação com os serviços oferecidos pelas camadas inferiores. Estas APIs facilitam as tarefas de elasticidade e escalabilidade destas aplicações, facilitando o desenvolvimento da aplicação e delegando para a API em si a tarefa de escolher onde a aplicação executa, como os recursos são obtidos, dentre outros parâmetros. Esta camada é conhecida também como Plataforma como Serviço (*Platform as a Service* - PaaS). Como exemplos de empresas que fornecem serviços nesta camada, podem ser citados o Google e a Microsoft, que disponibilizam o GoogleAppEngine e o Microsoft Azure, respectivamente. Ao observar a visão dos desenvolvedores, o modelo PaaS oferece uma série de vantagens, como a elasticidade automática, o balanceamento de carga automático e a integração com outros serviços oferecidos pelo PaaS como mecanismos de autenticação e serviços de e-mail. Assim, muitas das dificuldades de implementação são supridas pelo provedor da plataforma.

Camada de Infraestrutura de *Software*

A camada de infraestrutura de *Software* fornece recursos fundamentais para as camadas superiores, que permitem a construção de ambientes PaaS e SaaS. Os serviços oferecidos nesta camada podem ser categorizados como recursos computacionais, de armazenamento de dados e de comunicação.

A primeira forma de serviço é a oferta de recursos computacionais através da virtualização. As máquinas virtuais são a forma mais comum de oferta de recursos computacionais aos usuários da camada de infraestrutura de *Software*. Nas máquinas virtuais, os usuários possuem uma maior flexibilidade e controle da camada, pois geralmente são entregues máquinas virtuais com privilégios de administrador, onde os usuários desenvolvem suas próprias pilhas de *software* acima do sistema operacional fornecido. Esta camada é conhecida como Infraestrutura como Serviço (*Infrastructure as a Service* - IaaS). A utilização das máquinas virtuais como IaaS é possível devido ao desenvolvimento da paravirtualização e da virtualização assistida por *hardware*. Estas tecnologias atenuam problemas de falta de isolamento entre máquinas virtuais e dificultam que uma dada máquina virtual interfira no desem-

penho das outras, mesmo que todas compartilhem recursos comuns. Pode-se citar a Amazon *Elastic Compute Cloud* [4] e a infraestrutura da Enomalism [23] como exemplos comerciais de IaaS. Das tecnologias de código aberto de IaaS, pode-se citar o Eucalyptus [8].

A segunda forma de serviço é o armazenamento de dados. Este serviço permite que usuários armazenem seus dados em discos remotos, acessíveis de qualquer lugar e a qualquer instante. Um nome comum para este serviço é o Armazenamento como Serviço (*Data-storage as a Service - DaaS*. Uma das funcionalidades principais do DaaS é facilitar o aumento de escala de aplicações na nuvem, que não precisam ficar restritas ao espaço fornecido nos servidores da aplicação. Os usuários dos DaaS esperam que o armazenamento siga uma série de requisitos desejáveis, como a alta disponibilidade, confiança, desempenho, replicação e consistência dos dados. Prover todos estes requisitos é um grande desafio que envolve o processamento e controle distribuídos e a gerência e sincronismo de múltiplas réplicas de dados, de forma que os provedores de DaaS geralmente oferecem serviços que atendem somente alguns deles. Como exemplo de DaaS, pode-se citar o Amazon S3 [4] e o *EMC Storage Managed Service* [24].

A terceira forma de serviço é a comunicação. Os provedores de nuvem precisam prover qualidade de serviço e desta forma a comunicação e a rede que interconecta os elementos de nuvem se tornam fatores críticos na infraestrutura de nuvem. Os provedores devem fornecer capacidade de comunicação que seja configurável, escalável, previsível e confiável. Neste contexto, surge o termo Comunicação como Serviço (*Communication as a Service* - CaaS), que deve atender aos requisitos de redes. Dentre estes requisitos, pode-se pensar em segurança de redes, provisão dinâmica de redes sobrepostas *overlays*, isolamento de tráfego e garantias de banda e atraso e monitoramento de rede. O modelo CaaS ainda é pouco conhecido, mas existem soluções de CaaS como o *Connected Service Framework* (CSF) da Microsoft [5]. Como exemplo de aplicações que dependem de CaaS, pode-se citar sistemas VoIP (*Voice Over Internet Protocol*), conferências de vídeo e sistemas mensageiros.

Camada de Núcleo de *Software*

A quarta camada, de núcleo de *software*, oferece *softwares* de gerência dos servidores físicos que constituem as nuvens. O núcleo pode ser implementado como um sistema operacional, um hipervisor, um monitor de máquinas virtuais ou até um *middleware* de um sistema de clusters.

Camada de *Hardware*

A última camada é a camada de *hardware*. Esta camada representa o *hardware* físico das máquinas e os comutadores que formam o *backbone* da nuvem. Os usuários desta camada são normalmente grandes empresas com grandes demandas em TI, que precisam de alguma forma alugar *hardware* na forma de um serviço, ou seja, *Hardware* como Serviço (*Hardware as a Service* - HaaS). Para isso, o provedor de HaaS opera, gerencia e atualiza o *hardware* para atender aos seus consumidores, durante todo o período de aluguel. Este modelo é interessante para os usuários, pois eles não precisam se preocupar em investimentos de infraestrutura e gerência de infraestrutura, pois estas tarefas são atribuídas ao provedor de HaaS.

2.3 Vantagens da Computação em Nuvem

Como visto anteriormente a nuvem fornece um agregado de recursos para um grande número de entidades. Dependendo da entidade envolvida na nuvem (provedor de nuvem, consumidor ou usuário final), pode-se verificar uma série de vantagens. Para as entidades que consomem os recursos da nuvem, abstrai-se a gerência dos recursos e obtém-se uma ilusão de infinitos recursos computacionais sob demanda. Se a entidade for uma *startup*, por exemplo, pode-se alugar recursos na nuvem e estender os recursos oferecidos conforme a necessidade, evitando um grande investimento inicial em infraestrutura de TI, o que poderia desmotivar ou até coibir o surgimento da empresa. Além disso, os consumidores da nuvem delegam a disponibilidade do serviço e as questões relacionadas à infraestrutura e escalabilidade para o provedor de nuvem. Para o meio acadêmico, pode-se utilizar a nuvem para aumentar a capacidade de cálculo como forma de acelerar a pesquisa e a obtenção de resultados científicos.

As nuvens seguem o modelo de "pague pelo que usar" (*pay-as-you-use*). Assim, para os clientes, alugar um servidor por mil horas ou alugar mil servidores por uma hora [9] representam o mesmo custo monetário, enquanto a possibilidade de um cliente dispor de uma infraestrutura com mil servidores requer um investimento muito alto. Este modelo de pagar pelo uso é um dos maiores atrativos da nuvem. Desta forma grandes quantidades de dados podem ser processadas em paralelo e os cientistas podem obter seus resultados com uma velocidade muito maior, sem grandes investimentos em infraestrutura. Os usuários finais podem se beneficiar da disponibilidade dos serviços em qualquer instante de tempo e em qualquer lugar. Dado que o serviço e as informações estão na nuvem, torna-se possível acessá-los e compartilhá-los com outros usuários, desde que exista conectividade dos usuários com a nuvem.

Outra grande vantagem já citada é a elasticidade. A elasticidade é definida como a capacidade da nuvem adaptar a quantidade de recursos fornecidos a cada consumidor da nuvem sob demanda. Desta forma, um cliente que precisa de um súbito aumento de poder de processamento pode alocar dinamicamente mais processadores e atender a um determinado pico de demanda. Uma área ainda pouco explorada nas nuvens é o desenvolvimento de aplicações interativas que executam em dispositivos de baixa capacidade computacional como *tablets* e *smartphones*. Dado que as capacidades destes dispositivos são limitadas, o dispositivo pode funcionar como uma interface que utiliza a rede para interagir com servidores de alta capacidade, de forma semelhante a um terminal burro. Já existem empresas com propostas semelhantes, que fornecem servidores que processam jogos com um grande processamento gráfico e os usuários finais, mesmo que possuam um computador simples, podem interagir com o jogo através da tela e da rede.

Os *data centers* tradicionais são formados por um conjunto de máquinas de alto desempenho e de preço e manutenção elevados. Os provedores de nuvem por sua vez utilizam *data centers* com um modelo que emprega uma enorme quantidade de computadores de baixo custo para prover a infraestrutura. Esta nova forma de se implantar *data centers* fornece para os provedores uma série de vantagens como a economia na manutenção dos servidores, na administração de sistemas, na instalação e configuração de novos elementos da infraestrutura, dentre outros [25]. Além disso, estudos recentes demonstram que os *data centers* possuem diversos sistemas de redundância para atender as instabilidades, mas que na prática, as máquinas físicas destas infraestruturas podem ficar ociosas por volta de 90% do tempo. Isto se deve principalmente a dificuldade de se encontrar o ajuste ideal entre as demandas reais e as fornecidas, a incerteza sobre demandas futuras, o modelo de comprar em escala para redução de custos, gerência de risco, dentre outros [26]. Devido à elasticidade fornecida pelas tecnologias que viabilizam as nuvens (que serão detalhadas na Seção 2.4), pode-se desligar servidores subutilizados e migrar a carga destes para outros servidores, economizando grandes quantidades de energia. Além disso, podem-se utilizar mecanismos de ajuste de frequência de operação e controle inteligente de voltagem para diminuir o consumo em cenários de baixa carga computacional.

2.4 Tecnologias que Viabilizam as Nuvens

A computação em nuvem se tornou economicamente viável graças a uma série de tecnologias que permitem uma gerência eficiente, um controle fino dos recursos e um sistema de contabilidade (*accounting*). Esta seção demonstra algumas destas tecnologias e como elas permitem que as nuvens funcionem.

2.4.1 Virtualização

A virtualização de computadores é uma técnica que permite a abstração de recursos computacionais [1], implementada através de uma camada de *software*, podendo ser também auxiliada por *hardware*. Desta forma, um computador físico executa diversos computadores lógicos ou máquinas virtuais em paralelo, como pode ser visto de forma ilustrativa na Fig. 2.3. Na figura, diversos computadores lógicos executam em paralelo, dentro de um mesmo computador físico. Esta camada de *software* que permite a abstração de recursos é chamada de Monitor de Máquinas Virtuais (*Virtual Machine Monitor* - VMM ou hipervisor) e fornece às máquinas virtuais interfaces similares a interfaces reais de *hardware*. Desta forma, cada máquina virtual possui a impressão de estar executando em um ambiente isolado e dedicado, quando na verdade ela está compartilhando os recursos físicos com diversas outras máquinas através do hipervisor, como pode ser visto na Fig. 2.4. Assim, pode-se fazer um mapeamento entre quais recursos físicos são entregues a cada máquina virtual e, através do hipervisor, é permitido modificar dinamicamente a distribuição dos recursos existentes.

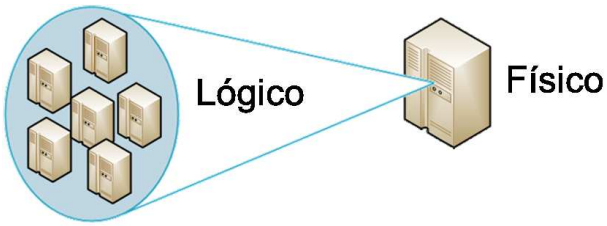


Figura 2.3: Abstração de recursos físicos computacionais na virtualização.

Através desta tecnologia é possível implementar técnicas de elasticidade para adequar as máquinas virtuais às cargas de serviço a elas submetidas. Nuvens de computadores estão geralmente associadas à virtualização, mas nem todos os provedores de nuvem utilizam a virtualização. Alguns provedores como o GoogleAppEngine que fornece a nuvem no modelo Plataforma como Serviço (PaaS) utilizam mecanismos próprios e infraestruturas proprietárias baseadas em clusters de computadores para fornecer a ilusão de nuvem, sem necessariamente utilizar máquinas virtuais.

A virtualização baseada em VMMs pode ser implementada de duas formas distintas. A virtualização total e a para-virtualização. A virtualização total oferece uma cópia do *hardware* para as máquinas virtuais, que atuam como se elas estivessem interagindo diretamente com o *hardware* subjacente. A vantagem desta abordagem é que os sistemas operacionais podem ser executados diretamente em um ambiente virtual, sem modificações. Apesar desta vantagem, a virtualização

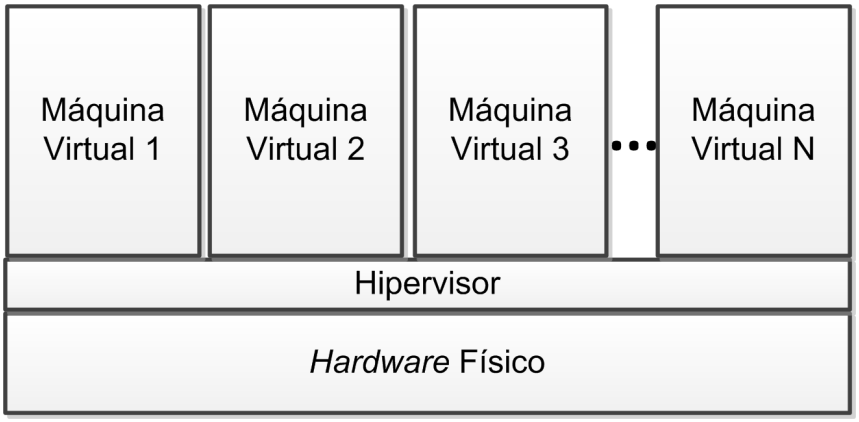


Figura 2.4: Virtualização de Recursos Computacionais.

total possui algumas desvantagens. Os sistemas operacionais virtualizados não são conscientes da virtualização e tentam executar instruções diretamente no *hardware* físico. O VMM precisa então interceptar todas as instruções para verificar se as instruções são sensíveis[1] e em caso positivo, precisam alterar dinamicamente as instruções para adequá-las ao cenário virtualizado fornecido pelo VMM. Este processo é denominado tradução binária (*binary translation* [27]) e adiciona uma sobrecarga significativa de processamento no VMM. Na para-virtualização, diferentemente da virtualização total, o sistema operacional virtualizado sabe que está executando em um ambiente virtualizado. O sistema operacional é modificado e são criados *drivers* para-virtualizados que estão preparados para lidar com o VMM, retirando a necessidade da tradução binária. Assim, o VMM recebe sempre instruções esperadas e não precisa consumir recursos de processamento para validar cada uma das instruções executadas pela máquina virtual. A desvantagem desta abordagem é que os sistemas virtualizados precisam sofrer modificações, que podem ser complicadas quando ocorre a virtualização de plataformas proprietárias como os sistemas operacionais da Microsoft.

Além da virtualização através do uso do VMM, existe ainda a virtualização de processo. Esta virtualização cria um contêiner que possui um ambiente de execução padronizado para que as aplicações virtualizadas encontrem um ambiente único que abstrai especificidades existentes em cada uma das plataformas de sistema operacional. Este contêiner executa como um processo em um sistema operacional que deve se responsabilizar por intermediar a comunicação deste processo com o exterior. Como um exemplo deste tipo de virtualização de processo, pode-se citar a máquina virtual Java, que fornece uma interface de programação independente de plataforma

[1]Instruções sensíveis são instruções de máquina que só podem ser executadas de forma privilegiada, pois elas envolvem acessos a partes críticas do sistema, exigindo que o VMM verifique se as máquinas virtuais podem executá-las de fato ou se elas precisam sofrer modificações antes da execução.

e o ambiente Java se responsabiliza por interpretar as funcionalidades abstratas do ambiente virtualizado e adequá-las a realidade do sistema operacional que executa o processo. Desta forma, uma aplicação desenvolvida em Java pode executar em qualquer sistema operacional existente, desde que ele possua uma implementação da máquina virtual Java.

Uma funcionalidade fornecida por algumas plataformas de virtualização como o KVM e o Xen [28], de código aberto e utilizado pela nuvem da Amazon, é a migração ao vivo de máquinas virtuais. Esta funcionalidade permite que máquinas virtuais sejam transferidas entre diferentes servidores físicos sem que os serviços que executam dentro da máquina sejam interrompidos [29]. Graças a esta funcionalidade, pode-se migrar máquinas virtuais e agregá-las em servidores mais eficientes, desligando servidores ociosos e economizando energia.

2.4.2 Economia de Energia

Além da virtualização, existem técnicas que permitem uma grande economia de energia sem que seja necessário desligar de fato as máquinas [30]. Nedevschi *et al.* [31] apresentam uma série de estudos sobre as técnicas de controle de frequência e de colocar para dormir (*sleep*) os componentes. A técnica de colocar para dormir (*sleep*) permite que alguns subcomponentes dos sistemas sejam desligados e, quando necessário, rapidamente ligados para atender a uma determinada demanda. As técnicas de controle de frequência permitem que os componentes eletrônicos executem em uma frequência de operação menor, diminuindo o desempenho quando os recursos estão ociosos e economizando energia.

Ao se pensar em desligamento de recursos de processamento, uma especificação muito utilizada é a especificação ACPI (*Advanced Configuration and Power Interface*, ou interface avançada de configuração e energia [32]). O ACPI é um padrão aberto para configurar dispositivos e gerenciar a energia em sistemas operacionais.

As especificações da ACPI definem estados de desempenho ou performance `P` e estados de execução do processador, ou estados `C`. Os estados `P` fornecem maneiras de se escalar a frequência e a voltagem com que o processador executa e a quantidade de estados depende do modelo do processador. Eles são ordenados de acordo com o desempenho. Desta forma, o estado `P0` é o estado de maior desempenho, seguido do estado `P1`, e assim por diante. Os estados `P` são conhecidos por nomes diferentes dependendo do fabricante. Como exemplos, cita-se a tecnologia Intel *SpeedStep*, AMD *PowerNow!* e *PowerSaver* em processadores VIA. Conforme o número do estado `P` aumenta, o controle de voltagem e frequência é mais rígido e economiza-se mais energia. Os estados `C` [33] permitem a modificação do estado de execução dos processadores. Os estados `C` são ordenados de forma que quanto maior o valor de `C`,

maior é a quantidade de recursos desligados para economizar energia. Devido a isso, quanto maior o valor de `C`, maior é o tempo necessário para se sair do estado e voltar a um estado normal de execução. A descrição abaixo indica algumas informações úteis sobre os quatro estados `C` que geralmente são encontrados nos processadores modernos.

1. `C0:` O estado `C0` é o estado normal de execução onde todos os componentes do processador estão ligados;

2. `C1:` O estado `C1` é conhecido como *Halt* e é um estado onde o processador está inapto a processar instruções. Apesar disso, ele pode retornar ao estado de execução em tempo quase instantâneo. Qualquer interrupção feita leva o processador automaticamente para o estado `C0`;

3. `C2:` O estado `C2` é chamado de *Stop-Clock* e é um estado semelhante ao estado `C1`. Um diferencial é que este estado desliga o *clock* do processador através do envio de um sinal para o pino do processador chamado de STPCLK. Além disso nesse estado todos os sinais de *clock* são desligados, enquanto que no estado `C1` alguns sinais de *clock* são mantidos ativos para permitir um retorno mais rápido ao estado de execução `C0`.Assim, a latência desta estado é maior. Ou seja, dado que uma interrupção acontece, este estado demora mais para voltar ao estado `C0`;

4. `C3:` Nos estados `C1` e `C2`, duas unidades internas do processador são mantidas ativas: o APIC (*Advanced Programmable Interface Controller*) e a interface com o barramento externo, para permitir o tratamento de interrupções e requisições de processamento que cheguem ao processador pelo barramento externo. O estado `C3` é chamado de *Sleep* desliga completamente estas duas unidades, de forma que requisições e interrupções importantes não podem ser respondidas de forma reativa. O estado é mudado para o estado `C0` automaticamente e possui uma latência maior que `C2`;

A Tabela 2.2 apresenta a economia obtida ao se utilizar os diferentes estados `C`.

Tabela 2.2: Consumo energético em diferentes estados C para um processador da série Intel Core i5 (adaptada de [34])

Estado C	Economia em relação a C0	Tempo de Transição
C0	0%	-
C1	70%	10 ns
C2	75%	100ns
C3	80%	50 μs

Outra forma de se desligar e ligar recursos dependendo da demanda são os protocolos como o *Wake-On-LAN* (WOL). O WOL é uma tecnologia que combina *hardware* e *software* para acordar sistemas computacionais que estão no estado dormindo (*sleep*) através do envio de pacotes de rede especiais para máquinas que possuam suporte a esta tecnologia. Ao receber este pacote, a máquina é ligada [35] e pode passar a funcionar normalmente. Dependendo do cenário onde é aplicado, o WOL pode não ser adequado. Por depender do endereço físico (MAC) das máquinas, o funcionamento deste pacote é restrito a redes locais. Além disso, o pacote mágico é um pacote UDP, que pode ser descartado prioritariamente em redes congestionadas.

Uma tecnologia mais recente que permite a gerência do *hardware* é a *Intel Active Management Technology* (AMT) [36]. Esta tecnologia está presente nas placas mais recentes da Intel e é uma tecnologia baseada em *hardware* e no *firmware* que permite que as informações sobre o *hardware* sejam armazenadas em uma região especial de memória não volátil. Desta forma, torna-se possível verificar as informações das máquinas mesmo que estas estejam desligadas. Além disso, o Intel AMT permite o controle energético do computador, o redirecionamento do console da máquina para máquinas remotas e a instalação e configuração de sistemas operacionais. A tecnologia executa em cima da pilha TCP/IP e permite a utilização de criptografia para proteger o canal de comunicação de ataques.

2.5 O Custo das Nuvens

A computação em nuvem executa sobre *data centers* dedicados a tarefa de prover serviços em nuvem. Um ponto fundamental para analisar a viabilidade de se alugar uma nuvem, ou até construir uma nuvem proprietária deve levar em conta a análise de custo das nuvens. Greenberg *et al.* realizam um estudo que define os principais problemas de custo associados as nuvens e aos seus *datacenters* [26]. Eles concluem que a maior parte do custo monetário das nuvens está associado ao *hardware* físico propriamente dito, os servidores. Em seguida, observa-se os sistemas de arrefecimento e os custos de energia elétrica. E por fim, os custos relacionados a rede, como enlaces e equipamento de rede.

Como exemplo da ordem de grandeza desses custos, os autores estimam o custo anual dos servidores de um grande *data center* com 50 mil servidores como aproximadamente 52,5 milhões de dólares, o custo de infraestrutura como 18,4 milhões de dólares e o custo energético como 9,3 milhões de dólares.

Estes resultados podem ser observados na Tabela 2.3.

Outro fator importante na implantação de um *data center* é a sua localização geográfica. Pode-se observar que lugares distintos podem apresentar custos energéticos

Tabela 2.3: Distribuição de Custos em um *Data Center*(adaptado de [26]).

% Custo	Componente	Subcomponentes
45%	Servidores	CPU, memória, disco
25%	Infraestrutura	Distribuição e refrigeração
15%	Energia	Custo energético
15%	Rede	Enlaces e equipamentos de rede

muito diferentes, como pode ser visto na Tabela 2.4.

Tabela 2.4: Custo do kilowatt-hora de energia em diferentes regiões (adaptado de [9]).

Preço (cents de Dólar)	Lugar	Motivos
3.6	Idaho	Hidrelétrica próxima
10.0	California	Distância da fonte de energia elétrica
18.0	Havaí	Precisa importar combustível

Desta forma, percebe-se que a implantação de um provedor de nuvem é muito onerosa, de forma que o custo para criar e manter um provedor de nuvem de baixa escala pode se tornar inviável na medida em que grandes provedores conseguem diluir seus custos operacionais e seus investimentos em bens de capital ao comprar equipamentos em grande escala. Além disso, deve-se levar em conta a localização dos clientes para se estabelecer um compromisso entre a latência com o usuário e o custo de manutenção do *data center*.

2.6 Modelos de Implantação de Nuvem

Dada a definição dos modelos de serviço, podem-se definir também quatro modelos de implantação de nuvens: Nuvem privada, comunitária, pública e híbrida [15]. As nuvens privadas são aquelas onde a infraestrutura é operada por uma dada organização particular. As nuvens comunitárias são compartilhadas entre várias organizações que compartilham interesses comuns. As nuvens públicas são nuvens onde os serviços são publicamente disponíveis. Por fim, a nuvem híbrida une dois ou mais modelos de implantação, por exemplo, em um cenário onde a nuvem privada não possui mais recursos, ela pode interagir com uma nuvem pública para auxiliar uma determinada tarefa.

2.7 Ambientes de Computação em Nuvem

Esta seção apresenta alguns ambientes de computação em nuvem existentes no mercado, de acordo com a forma de serviço oferecido.

2.7.1 Infraestrutura como Serviço (IaaS)

OpenNebula

O OpenNebula [7] é um projeto de código aberto que fornece um ambiente para gerência de infraestruturas de *data centers*. Ele não é de fato um sistema para prover Infraestrutura como Serviço e sim um sistema para orquestrar *data centers* que utiliza plataformas de virtualização que fornecem máquinas virtuais. A interface de gerência do OpenNebula permite o armazenamento de imagens de disco pré-configuradas para facilitar a instanciação de novas máquinas virtuais e a aplicação de primitivas de virtualização como a migração, pausa, reinício e desligamento de máquinas. Uma das vantagens do OpenNebula é que ele fornece mecanismos de controle de acesso e autenticação, o que permite que a nuvem seja dividida em regiões com diferentes níveis de acesso e gerência, permitindo um controle fino de quais recursos são acessíveis por quais usuários.

Em relação à rede, o OpenNebula utiliza o Open vSwitch e permite que máquinas virtuais sejam associadas a VLANs diferentes, permitindo que sejam criados domínios de rede distintos dentro de uma mesma nuvem.

Eucalyptus

O Eucalyptus [8] é um arcabouço de código aberto de nuvens que fornece o modelo de IaaS para os seus clientes. A nuvem implementada permite o agendamento e a gerência de máquinas virtuais, o armazenamento de dados e de imagens, interfaces de administração e a definição e a execução de SLAs [15]. O Eucalyptus é compatível com o hipervisor Xen e o hipervisor KVM, permitindo a utilização destas plataformas de virtualização.

Amazon Web Services

A Amazon fornece máquinas virtuais baseadas em Xen para seus clientes. A Amazon permite que esta máquina virtual execute diferentes sistemas operacionais e seja configurada com diferentes parâmetros de *hardware* como a configuração dos processadores, o número de processadores, a memória e disco disponível. Ainda pode-se optar por configurações específicas que permitem estabelecer redes de sobrecamada entre máquinas virtuais.

Rackspace

A Rackspace é uma empresa que fornece infraestrutura de nuvem para o público geral. A tecnologia de virtualização utilizada é baseada no Xen. Uma das vantagens do Rackspace é que a interface com o cliente oferece muitas opções de gerência

que permitem que o cliente rapidamente escale sua máquina virtual e aloque mais recursos ou menos recursos para a máquina.

2.7.2 Plataforma como Serviço (PaaS)

GoogleAppEngine

O GoogleAppEngine permite que os desenvolvedores utilizem a infraestrutura do Google para o desenvolvimento de aplicações. Desta forma, através de uma API própria, os desenvolvedores podem programar seus aplicativos para executarem dentro da nuvem do Google. Desta forma, eles não precisam se preocupar com o balanceamento ou com a elasticidade, que é provida de forma transparente pela plataforma de desenvolvimento do Google.

Microsoft Azure

O Microsoft Azure utiliza uma API proprietária da Microsoft que fornece uma plataforma para desenvolvimento de aplicações na nuvem da Microsoft. Ele é baseado no Microsoft *Common Language Runtime* (CLR), que é um ambiente virtualizado que executa em um ambiente gerenciável. Ele também oferece o balanceamento de carga automático, de forma semelhante ao GoogleAppEngine.

2.7.3 *Software* como Serviço (SaaS)

Dropbox

O Dropbox é uma aplicação de armazenamento de dados pessoais na nuvem. Através da interface do Dropbox é possível sincronizar pastas e documentos na rede e acessá-los de qualquer lugar. Também é possível compartilhar pastas e arquivos com outros clientes do Dropbox. Um aspecto interessante é que o Dropbox utiliza a infraestrutura como serviço provida pela Amazon, de forma que o Dropbox é um serviço da nuvem que utiliza serviços de outro provedor de serviços da nuvem. O Dropbox utiliza máquinas virtuais da Amazon para prover a interface com o sistema e criptografar os dados dos usuários. E utiliza a parte de armazenamento da Amazon (S3) para guardar os dados dos usuários.

Google Docs

O Google Docs oferece um conjunto de serviços de escritório para os usuários. Os usuários podem ter seus documentos, planilhas e apresentações sincronizadas na nuvem e eles podem criar, modificar e remover estes arquivos através de uma interface web semelhante as suítes de escritório oferecidas no modelo de *software* de

prateleira como o Microsoft Office. Além disso, a suíte permite o desenvolvimento colaborativo de documentos, que podem ser compartilhados entre todos os usuários da plataforma.

Zoho

O Zoho fornece uma suíte de aplicativos de produtividade para empresas, como aplicações de reunião virtual, listas de discussões, ferramentas de recrutamento, calendários compartilhados, etc.

A lista de ambientes em nuvem é extensa e cresce a cada dia. Aqui foram citados apenas alguns dos principais ambientes.

2.8 Desafios e Tendências dos Ambientes em Nuvem

Esta seção apresenta alguns dos maiores desafios para o crescimento das nuvens de computadores, de acordo com pesquisadores de Berkeley [9]. Um pequeno resumo destes desafios pode ser visualizado na Tabela 2.5, adaptada de [9].

2.8.1 Disponibilidade do serviço

Uma das maiores preocupações sobre as nuvens de computadores é a questão da disponibilidade do serviço. Os clientes da nuvem esperam que os serviços fornecidos estejam disponíveis em qualquer lugar e em qualquer instante de tempo. Dado a escala e a complexidade dos sistemas de nuvens existentes, percebe-se a dificuldade de se garantir a disponibilidade. Como exemplo deste problema, em abril de 2011, os serviços da Amazon sofreram uma pane e ficaram indisponíveis por um período de 36 horas [37], afetando o negócio de grandes empresas como o Foursquare, New York Times, Reddit e Quora. Além disso, são conhecidos alguns exemplos de falhas de programação na GoogleAppEngine que tornou o serviço indisponível por um período de cinco horas em julho de 2008 [9]. Uma pequena tabela agregando informações de [9] [38] [39] apresenta um resumo de algumas falhas conhecidas que aconteceram em sistemas de nuvem e pode ser vista na Tabela 2.6.

Percebe-se que a disponibilidade é um ponto crítico na nuvem, pois se a nuvem ficar indisponível, milhões de clientes terão seus serviços afetados pela pane. Observa-se que os problemas ocorrem devido a falhas humanas, falhas de *software* e até devido a problemas da localização dos dados, como o caso da Amazon e da Microsoft onde a nuvem se encontrava fisicamente próxima de uma tempestade elétrica que danificou os servidores. Para tentar mitigar este problema, devem-se utilizar

Tabela 2.5: Obstáculos e Oportunidades nas Nuvens

#	Obstáculo	Oportunidade de Pesquisa
1	Disponibilidade do serviço	Utilizar múltiplos provedores de nuvem; Utilizar elasticidade para prevenir DDOS
2	Aprisionamento dos dados	Padronização das APIs; Desenvolvimento de *softwares* compatíveis entre si
3	Confidencialidade e auditoria dos dados	Desenvolver criptografia, VLANs, firewalls, áreas de armazenamento geograficamente distribuídas
4	Gargalos de transmissão de dados	Enviar dados por correio; Backup e arquivamento de dados; Comutadores de maior capacidade
5	Predição do consumo	Memória flash; Políticas de escalonamento
6	Armazenamento escalável	Desenvolver técnicas de escalonamento escalável
7	Falhas em sistemas distribuídos	Desenvolver sistemas de debug eficientes em sistemas distribuídos
8	Elasticidade	Desenvolver escaladores automáticos, sistema de replicação eficiente
9	Reputação dos provedores de nuvem	Desenvolver serviços voltados para garantias de direitos e serviços de garantia de reputação
10	Licença de *software*	Elaborar modelos de pagamento mais eficientes
11	Coerência e sincronismo entre *data centers*	Novos algoritmos de coerência

múltiplos provedores de nuvem com réplicas do serviço. Além disso, a disponibilidade pode ser afetada por ataques de negação de serviço distribuídos feitos por *botnets*. Dado que a nuvem deve escalar de forma eficiente e permitir a elasticidade, pode-se pensar em um cenário onde a nuvem é capaz de absorver os ataques. Esta técnica poderia desmotivaria os atacantes, que ameaçam provedores de serviço e cobram cifras da ordem de dezenas de milhares de dólares para não lançar o ataque e prejudicar o serviço oferecido [9], através de um mecanismo de ameaças. Desta forma, mesmo sob ataque, os provedores conseguiriam atender as demandas dos clientes mesmo sob o ataque.

Tabela 2.6: Falhas em provedores de nuvem.

Serviço Afetado	Duração Média	Data
Amazon : Indisponibilidade do serviço (Netflix, Instagram e Pinterest)	algumas horas	06/12
Microsoft Office 365: Indisponibilidade do serviço	algumas horas	09/11
Google Docs: Indisponibilidade do serviço	1 h	09/11
Microsoft e Amazon: Indisponibilidade da nuvem hospedada em Dublin devido a tempestade elétrica	algumas horas	08/11
Microsoft BPOS: Indisponibilidade do serviço	2 h	05/11
Microsoft BPOS: Atrasos no envio de emails	24 h	05/11
Microsoft BPOS: Interrupções e atrasos na entrega de mensagens	72 h	05/11
Yahoo Mail: Indisponibilidade do serviço	algumas horas	04/11
VMware Cloud Foundry: Perda de conectividade com o serviço por falha humana	24 h	04/11
Amazon EC2: Pane nos serviços da Amazon	36 h	04/11
Intuit: Falha na hospedagem dos serviços	120 h	04/11
Gmail: Desaparecimento temporário dos emails de 150 mil usuários	96 h	02/11
Microsoft Live Hotmail: Deleção temporária dos emails de 17 mil usuários	48 h	12/10
Amazon S3: sobrecarga no serviço de autenticação que levou a indisponibilidade do serviço	2 h	02/08
Amazon S3: Falha de programação que levou a faltas severas no funcionamento do protocolo utilizado	6-8 h	07/08
Gmail: Sítio indisponível devido a erro no sistema de contatos	1.5 h	08/08
GoogleAppEngine: Erro de programação	5 h	06/08

2.8.2 Aprisionamento dos Dados

A réplica de servidores em múltiplas nuvens está relacionada a outro desafio, que foi denominado como aprisionamento dos dados. Dado que um provedor de nuvem foi contratado, torna-se necessário adequar o *software* as APIs fornecidas e desenvolver aplicações e serviços seguindo as normas de programação ou as restrições impostas pelo provedor de nuvem. Assim, caso torne-se necessário migrar o serviço ou replicá-lo em outra nuvem, o serviço precisa passar por uma etapa de adaptação, pois as APIs e as restrições mudam. Desta forma, uma ótima oportunidade de pesquisa se relaciona com o foco em padronizar as APIs das nuvens de diferentes empresas, permitindo que os clientes extraiam e migrem seus serviços entre nuvens de forma fácil e transparente. Desta forma, o mercado de nuvens se tornaria mais competitivo e os clientes poderiam aumentar a robustez dos seus serviços, pois os serviços poderiam executar em diferentes instâncias de diferentes provedores de nuvem.

2.8.3 Confidencialidade e Auditoria dos Dados

Um desafio fundamental das nuvens, principalmente as nuvens públicas, é a questão da segurança dos dados. Os dados e informações colocados na nuvem são replicados em diversos servidores e estão disponíveis o tempo todo. Desta forma, o sistema fica exposto a ataques e, além disso, esta disponibilidade pública e a dificuldade de se registrar todos os acessos realizados cria um fator limitante relativo à adoção de nuvens em grandes corporações. Pode-se mitigar estes problemas utilizando ferramentas já conhecidas como armazenamento criptografado, VLANs e *middleboxes* como filtros de pacotes e *firewalls*. Quanto à questão da auditoria, pode-se pensar em desenvolver uma camada entre as máquinas virtuais e o hipervisor, responsável por fazer a auditoria dos dados. Outro desafio é a localidade dos dados devido a leis específicas de cada país. Como exemplo de contorno deste problema, a Amazon fornece serviços fisicamente localizados nos Estados Unidos e na Europa, permitindo que os clientes escolham onde seus dados serão guardados geograficamente [9].

2.8.4 Gargalos de Transmissão de Dados e Predição do Consumo

A grande quantidade de dados transmitidos, especialmente em casos onde aplicações científicas enviam grandes quantidades de dados brutos para serem processados na nuvem, pode tornar-se um gargalo de transmissão de dados. Acredita-se que este problema pode ser atenuado com a evolução das tecnologias de rede e a provável redução dos custos de utilização dos enlaces. Porém, mesmo que esta banda aumente, encontra-se um problema mais sério que está relacionado à tecnologia de virtualização que permite o funcionamento das nuvens.

Sabe-se que a virtualização apresenta limitações no compartilhamento de dispositivos de entrada e saída (E/S) [40], de forma que este gargalo é um dos grandes problemas das nuvens. Pode-se pensar no desenvolvimento de mecanismos mais eficientes de compartilhamento de dispositivos de E/S, tanto para rede quanto para o disco. Algumas pesquisadores [9] acreditam que a adoção de armazenamento baseado em memória *flash* pode atenuar o problema de E/S de disco, pois esta tecnologia de memória consegue lidar com um número muito maior de operações de E/S por segundo.

Outro desafio relacionado diz respeito ao escalonamento eficiente das máquinas, dado que a demanda é muito variável e existem aplicações como as aplicações de computação de alto desempenho (*High Performance Computing* - HPC) que requerem o sincronismo de diversos *threads* do programa. A forma que o escalonamento das máquinas virtuais é executado não oferece garantias de sincronismo, o que é uma

ótima oportunidade de pesquisa, para se oferecer um escalonamento sincronizado.

A predição do consumo está associada a capacidade do sistema se adaptar a demandas variáveis e conseguir atender aos padrões de utilização de recursos dos clientes da nuvem.

2.8.5 Armazenamento Escalável

As nuvens lidam com um volume de dados enorme e um grande conjunto das aplicações nas nuvens exige a utilização da nuvem como uma área de armazenamento. Desta forma, um grande desafio para ambientes em nuvem é como armazenar enormes quantidades de dados de forma eficiente. Pode-se pensar em diferentes modelos de bancos de dados ou diferentes estruturas de dados voltadas para as nuvens.

2.8.6 Falhas em Sistemas Distribuídos

Uma das dificuldades das nuvens é a detecção e a recuperação de erros, pois as nuvens são sistemas distribuídos de grande escala. Desta forma, a ocorrência de falhas dificilmente pode ser replicada em configurações restritas, de forma que a depuração deve ocorrer em ambientes de escala de *data centers*. Uma oportunidade neste cenário é a utilização de máquinas virtuais, que permite um maior controle sobre cada elemento de *software* e a adoção de mecanismos de controle e réplica de máquinas virtuais e dados para aumentar a tolerância a falhas desses sistemas.

2.8.7 Reputação dos Provedores de Nuvem

A manutenção de uma boa reputação é um grande desafio para os provedores de nuvem. Por exemplo, um dado cliente pode utilizar a nuvem de forma maliciosa para gerar spam, disseminar vírus, dentre outros. Desta forma, os IPs associados ao provedor de nuvem podem ser inseridos em listas negras e prejudicar o serviço de usuários bem intencionados. Neste cenário, pode-se pensar em mecanismos similares ao *trusted email* que utiliza assinaturas e autenticação mútua para rastrear possíveis problemas. Além disso, existem questões legais sobre a responsabilidade das ações. Deve existir algum mecanismo para garantir que a responsabilidade da ação maliciosa é do cliente que a fez e não do provedor da nuvem. Neste contexto, pode-se desenvolver soluções de monitoramento e sistemas de reputação para nuvens que poderiam medir o grau de confiança em cada provedor de nuvem ou que pudesse migrar os serviços para ambientes de maior reputação.

2.8.8 Licença de *Software*

O modelo de negócios de venda de *softwares* de prateleira tradicional não é adequado ao ambiente de nuvens. As licenças tradicionais relacionam os *softwares* aos computadores físicos onde eles podem executar. Ao estender o *software* para a nuvem, existe a oportunidade de se pensar em diferentes modelos de negócio e cobrança adequados a nuvem. Como exemplo, a Microsoft oferece licenças para o Windows Server e o Windows SQL Server no modelo "pague pelo que usar" em servidores Microsoft armazenados na nuvem EC2 da Amazon. Pode-se pensar em desenvolver metodologias e sistemas que consigam garantir o controle sobre quem tem permissão para executar cada serviço na nuvem e como estes clientes podem ser cobrados por isso.

2.8.9 Coerência e Sincronismo entre *Data Centers*

As nuvens de computadores realizam uma série de réplicas de serviços e estas réplicas podem ficar espalhadas geograficamente para atender a uma grande quantidade de usuários de forma eficiente, por exemplo, diminuindo a latência entre o cliente e a nuvem ao se utilizar uma réplica mais próxima. Ao mesmo tempo em que esta diversidade geográfica visa atender os usuários de forma mais eficiente, surge um problema muito sério e comum em sistemas distribuídos, que é a coerência dos dados. Torna-se necessário que as réplicas dos serviços mantenham um alto grau de coerência entre si, de forma que um dado modificado recentemente possa ser acessado o mais rápido possível em outros lugares da nuvem, sem perda de consistência. Existem algumas soluções comerciais de cache coerente como o disponibilizado pela EMC, o VPLEX [3] que prometem baixa latência e alto grau de coerência entre *data centers* com grandes distâncias geográficas entre si.

2.8.10 Elasticidade

Uma das áreas mais importantes de pesquisa para as nuvens está relacionada a elasticidade, ou seja, a capacidade de dinamicamente adicionar e remover recursos dos clientes, para atender as suas demandas. Alguns provedores de plataforma como serviço oferecem mecanismos de elasticidade automática, mas um grande desafio é como fazer isso de forma a respeitar os contratos existentes e sem prejudicar os contratos de outros clientes da nuvem. Dustdar *et al.* realizam estudos sobre a viabilidade e os desafios da elasticidade em nuvens [2] e propõem o conceito de processos elásticos (EP - *Elastic Process*). Os EPs levam em consideração a elasticidade dos recursos, o custo para prover esta elasticidade e como a elasticidade é provida e o seu impacto na qualidade do serviço oferecido.

A área de elasticidade é o tema principal desta dissertação e será abordado com maior profundidade nos próximos capítulos.

2.9 Principais Observações

As nuvens de computadores fornecem um repositório de recursos computacionais como *hardware*, plataformas de desenvolvimento e serviços de forma fácil e acessível. Dentre as características principais das nuvens, podem-se citar a utilização da virtualização, a adaptação automática de acordo com a demanda, o melhor aproveitamento de recursos, a amigabilidade e a centralização das nuvens na Internet.

Existem uma série de grandes empresas que oferecem serviços na nuvem em diferentes modelos de serviço como o modelo de infraestrutura como serviço (IaaS) fornecido pela Amazon, o modelo de plataforma como serviço (PaaS) oferecido pelo Google e sua plataforma GoogleAppEngine e o modelo de *software* como serviço (SaaS) como o serviço de armazenamento de arquivos pessoais fornecido pelo Dropbox. A tendência é que surjam cada vez mais empresas que se beneficiam da nuvem, tanto no provimento de serviços como na utilização dos serviços oferecidos por estas grandes empresas para o desenvolvimento de serviços próprios. Outro fator interessante é que cada vez mais surgem novos provedores de nuvem, que tentam otimizar e desenvolver mecanismos próprios de virtualização para aumentar o número de máquinas virtuais que executam em simultâneo sobre uma mesma máquina física e que melhorem a virtualização de dispositivos de entrada e saída, que são um dos grandes empecilhos no desenvolvimento de aplicações intensivas em E/S em ambientes virtualizados.

Mesmo que as nuvens apresentem uma série de vantagens como a disponibilidade, a escalabilidade e o modelo de pagar somente pelo que foi consumido, as nuvens de computadores encontram uma série de obstáculos que podem reduzir a sua taxa de crescimento. Dentre estes obstáculos, podem-se citar a dificuldade de se desenvolver mecanismos eficientes de elasticidade, a predição do consumo, a falta de padronização das interfaces fornecidas por nuvens de diferentes provedores de nuvem, a privacidade dos dados, o estabelecimento de licenças de *software* e a depuração de falhas em sistemas distribuídos de alta complexidade. Além disso, como foi demonstrado neste capítulo, a localização dos *data centers* e a forma como os dados são replicados entre eles é uma questão estratégica e jurídica. Primeiramente, o custo energético de cada localização pode variar muito dependendo da proximidade de geradores de energia como hidrelétricas. Além disso, a localização dos dados possui um impacto significativo na latência experimentada pelos clientes da nuvem. Por fim existe a questão legal, pois a localidade dos dados pode implicar na aplicação de leis regionais que podem ferir a privacidade ou os requisitos de contrato estabe-

lecidos pelos clientes da nuvem. Como exemplo, pode-se citar a Amazon que possui servidores nos Estados Unidos e na Europa, e o cliente pode escolher a localidade de seus dados e réplicas para atender as suas demandas.

Conclui-se que as nuvens oferecem uma série de novas possibilidades e modelos de serviços. Este paradigma pode tanto beneficiar grandes empresas consolidadas como fornecer excelentes oportunidades de negócio para empresas emergentes. As nuvens apresentam uma série de desafios de pesquisa que poderão melhorar ainda mais a forma como os usuários lidam com as nuvens. No próximo capítulo, é abordada a elasticidade de recursos e o desenvolvimento de mecanismos de automatização da gerência das nuvens. O capítulo aprofunda os conceitos fundamentais da área como a migração de processos, a migração de máquinas virtuais e as primitivas de alocação que podem ser manipuladas com elasticidade. Além disso são apresentados os trabalhos relacionados de elasticidade e gerência de nuvens de computadores.

Capítulo 3

Elasticidade e Alocação Dinâmica de Recursos

O capítulo anterior apresentou uma análise do funcionamento, dos desafios e das restrições das nuvens de computadores. A elasticidade foi apresentada como um das maiores qualidades e ao mesmo tempo um dos maiores desafios para este modelo de computação. Este capítulo aprofunda a análise sobre o aspecto da elasticidade da nuvem e da alocação dinâmica de recursos, que herdam muitas características das técnicas de migração de processos. São apresentadas as definições de processos, migração de processos, elasticidade, como ela pode ser obtida através da virtualização, o estado da arte dos mecanismos de controle para prover elasticidade e os trabalhos relacionados à alocação automatizada de recursos.

3.1 Alocação Dinâmica de Recursos

Para abordar a alocação dinâmica de recursos é conveniente começar pela migração de processos. Um processo é uma abstração do sistema operacional que representa uma instância de um programa de computador [41]. A migração de processos é definida como o ato de transferir um processo entre duas máquinas durante a execução do processo. Esta migração de processos gerou uma série de implementações para sistemas operacionais distintos como o MOSIX, o Accent, o Mach e o Sprite [42].

A técnica de migração de processos permite distribuir cargas dinamicamente, migrando processos de máquinas sobrecarregadas para máquinas menos carregadas, garantir resiliência ao migrar processos de máquinas que apresentam falhas parciais e melhorar a administração dos sistemas ao migrar processos de máquinas que serão desligadas e otimizar a localidade dos processos, migrando os processos para localizações mais próximas dos dados que o processo está operando. Apesar destas

vantagens, a migração de processos ainda não é adotada em alguns sistemas operacionais, devido à complexidade de se adicionar mecanismos de migração transparentes aos sistemas já existentes.

Os objetivos de se utilizar a migração de processos estão sempre vinculados ao tipo de aplicação que utilizará a migração. Os objetivos principais da migração de processos são (adaptado de [41]):

- **acesso a maior capacidade de processamento:** A utilização da capacidade de processamento das máquinas de um *data center* é variável no tempo. Desta forma, pode-se utilizar a migração de processos para aproveitar de forma mais eficiente os recursos disponíveis, migrando processos para máquinas que possuam capacidade de processamento ociosa;

- **exploração da localidade de recursos:** Em determinadas aplicações, a localidade dos dados manipulados exerce papel fundamental no desempenho das aplicações. Desta forma, através da migração de processos, pode-se mover um processo para uma máquina próxima aos dados que estão sendo operados, reduzindo a latência de acesso e a utilização de recursos de rede;

- **resiliência a falhas:** A resiliência a falhas pode ser melhorada através da migração de processos de nós que apresentam falhas parciais, ou em situações onde aplicações que executam por longos períodos de tempo, quando diferentes tipos de falha como falhas em algum disco ou memória ou falhas na rede podem ocorrer. Um exemplo possível de falhas parciais é a utilização das tecnologias S.M.A.R.T dos discos rígidos que podem indicar falhas nos discos rígidos antes que eles parem de funcionar;

- **administração do sistema:** A migração de processos facilita o processo de administração dos sistemas computacionais. Podem-se transferir temporariamente processos de uma máquina que precisará ser desligada para a manutenção e após a manutenção o processo é migrado de volta para a máquina de origem;

- **computação móvel:** A computação móvel pode ser beneficiada com a utilização da migração pois as aplicações podem migrar dinamicamente de forma a acompanhar o usuário da aplicação independentemente de sua localização física.

Além disso, devem ser considerados os tipos de aplicação que podem obter os maiores benefícios da migração de processos (adaptado de [41]):

- **aplicações paralelizáveis:** As aplicações paralelizáveis podem ser inicializadas em um conjunto inicial de nós e posteriormente migradas para outros nós, de acordo com a demanda de cada um dos processos envolvidos;

- **aplicações de longa duração:** As aplicações que executam por longos períodos de tempo como dias e semanas podem sofrer diferentes interrupções como a falha parcial de máquinas ou desligamentos de origem administrativa. A migração de processos pode realocar estas aplicações de forma transparente para evitar as interrupções;

- **múltiplas cargas de trabalho genéricas:** Em alguns ambientes como os ambientes das universidades, um conjunto de cargas de trabalhos distintas e não previsíveis pode ser executado ao mesmo tempo. Desta forma, conforme os usuários submetem tarefas e executam aplicações, a carga em cada um dos nós individuais dos sistemas varia muito. A migração de processo permite a distribuição destas cargas entre todos os nós;

- **aplicações de rede:** As aplicações de rede são umas das aplicações que mais podem se beneficiar da migração de processos. Agentes móveis voltados para a mobilidade podem utilizar a migração para aproximar os dados dos usuários ou para acompanhar os usuários conforme eles se movimentam na rede.

3.1.1 Migração de Máquinas Virtuais

A migração ao vivo de máquinas virtuais [29] se assemelha em muito à migração de processos. A principal diferença é que a migração ao vivo de máquinas virtuais envolve a migração de um sistema operacional completo, incluindo todas as aplicações em execução, para outra máquina física. Desta forma, todos os processos em execução no sistema operacional migrado são transferidos junto com o sistema operacional para outra máquina física, de forma transparente, como pode ser visto na Fig. 3.1. Na figura, a máquina virtual três é migrada de uma máquina física para outra. Como o *hardware* disponibilizado para as máquinas virtuais é uma abstração do *hardware* real, a máquina física que recebe a máquina virtual migrada pode recriar esta abstração, mesmo que as configurações das máquinas físicas sejam distintas, dado as devidas restrições como por exemplo garantir que a máquina de destino possua memória suficiente para abrigar a máquina virtual migrada. Assim a migração de máquinas virtuais estende a migração de processos e permite que aplicações que não foram desenvolvidas com suporte nativo a migração sejam migradas de forma transparente e eficiente.

A migração ao vivo em ambientes virtualizados pode acontecer de duas formas principais: a migração ao vivo baseada em pré-cópia e a migração ao vivo baseada

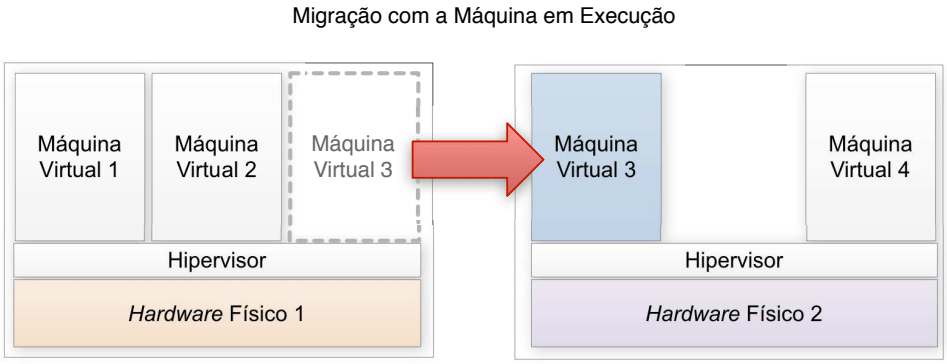


Figura 3.1: Migração ao vivo de máquinas virtuais.

em pós-cópia [43]. A migração com pré-cópia é a migração adotada na maioria das plataformas de virtualização, como o Xen, KVM e o VMWare. No processo de pré-cópia, a memória da máquina virtual é totalmente replicada no destino da migração antes que a execução de fato da máquina passe a ocorrer no destino. Ao iniciar o processo de pré-cópia, o algoritmo de migração segue os seguintes passos. Inicialmente executa-se um monitor de páginas sujas (*dirty pages*), que monitora as páginas de memória que são modificadas durante a execução da máquina virtual. Em seguida, toda a memória da máquina virtual é copiada para o destino. Como a máquina virtual ainda executa na origem, páginas de memória são sujas na origem. Estas páginas sujas precisam ser retransmitidas para o destino. Quando o algoritmo decide que a quantidade de páginas sujas remanescentes é menor que um limiar determinado, a máquina virtual tem sua execução suspensa e as páginas restantes são copiadas para o destino. Após este passo, a máquina pode ser resumida no destino. Pode-se perceber que esta forma de migração apresenta alguns problemas. Primeiramente, o tempo de migração depende da quantidade de memória de cada máquina, da frequência de utilização de memória da máquina virtual e da banda disponível para a migração. Em cenários extremos, se a taxa de atualização de páginas de memória for maior que a capacidade da rede de transferir as páginas atualizadas para o destino, a migração pode não acabar [44]. A estimação do tempo de migração do mecanismo pré-cópia é definida por

$$TempoTotal = \frac{TamanhoDaMemória}{Banda} + \beta, \tag{3.1}$$

onde β é um parâmetro que reflete a taxa de atualização de memória por unidade de tempo. Dependendo do valor de β, o tempo total pode variar muito. Em casos extremos este valor pode inviabilizar a conclusão da migração.

A segunda forma de realizar a migração ao vivo é através do mecanismo de pós-cópia [45]. O mecanismo de pós-cópia parte do pressuposto que é mais eficiente

transferir as páginas de memória após a reiniciação da máquina virtual no destino. O algoritmo funciona da seguinte forma. Ao inicializar o processo de migração, a máquina é suspensa na origem. O conteúdo dos registradores dos processadores virtuais e os estados dos dispositivos virtuais são copiados para o destino. Em seguida, a máquina virtual é reiniciada no destino sem nenhuma página de memória. Caso a máquina virtual necessite de alguma página de memória ainda não copiada, a máquina virtual é pausada temporariamente, as páginas requisitadas são copiadas e a máquina é reiniciada. Além deste mecanismo, existe um processo que executa em paralelo que realiza a cópia das páginas de memória da origem para o destino. Este mecanismo leva em consideração estatísticas de utilização de páginas para transferir inicialmente as páginas mais frequentemente atualizadas. A grande vantagem desta modalidade de migração é que a migração pode ocorrer com maior agilidade e, além disso, o tempo máximo de migração pode ser estimado. O tempo total é dado por

$$TempoTotal = \frac{TamanhoDaMemória}{Banda}, \tag{3.2}$$

que depende somente do tamanho total da memória e a banda disponível. Desta forma, pode-se estimar precisamente o tempo máximo para concluir a migração. Um ponto importante de ressaltar é que nesta modalidade de migração, se uma determinada página de memória for requisitada e ela não estiver presente na máquina física, o tempo de resposta será prejudicado, pois ele sofrerá um aumento proporcional ao tempo requisição e entrega da página de memória que está na máquina de origem.

3.1.2 Primitivas de Alocação de Recursos

A virtualização e a camada de abstração permitem a alocação dinâmica de recursos em uma mesma máquina virtual. Uma mesma máquina virtual, em tempo de execução, pode receber novos processadores virtuais, mais memória e disco.

1. **Processamento:** Os sistemas de virtualização utilizam escalonadores que permitem escalonar o acesso ao processador pelas diversas máquinas virtuais que coexistem na mesma máquina física. Desta forma, pode-se configurar dinamicamente este escalonador para dar mais prioridade a determinadas máquinas virtuais, ou para incluir ou excluir máquinas virtuais de sua fila de execução. Na ferramenta de virtualização Xen, por exemplo, pode-se optar por diferentes tipos de escalonadores como o *Borrowed Virtual Time*, o *Scan Earliest Deadline First* e o *SMP Credit*, que é o escalonador padrão do Xen [46]. No escalonador padrão, dois parâmetros podem ser ajustados no escalonador [47]. O primeiro deles é o `peso` (*weight*). O `peso` define o peso de cada máquina virtual e o escalonador dá prioridade a máquinas virtuais que possuem mais

peso, em situações de escassez de processamento. O segundo parâmetro é o `cap`. O `cap` define um limite rígido de utilização de processamento e indica um percentual máximo de tempo de processamento que é dado a cada máquina virtual. Assim, através deste controle, podem-se alocar dinamicamente recursos de processamento nas máquinas virtuais;

2. **Memória:** Alguns sistemas de virtualização como o Xen permitem que a memória das máquinas virtuais seja dinamicamente alterada. Na ferramenta Xen este procedimento de alocação dinâmica é feito através de um módulo do *kernel* chamado de *Balloon driver*. Este mecanismo permite que a máquina virtual requisite memória para o hipervisor (inflar) e retorne memória para o hipervisor (desinflar). O nome do driver vem da ideia de um balão, que pode inflar e desinflar. Assim, a memória das máquinas virtuais também pode ser alocada dinamicamente;

3. **Disco:** De forma semelhante a alocação de memória, os sistemas de virtualização permitem que sejam criados discos dinâmicos, que podem ser expandidos conforme a demanda das máquinas. O único problema dos discos dinâmicos é que a alocação dinâmica do disco gera perda de desempenho e facilita a fragmentação dos dados.

3.1.3 Elasticidade

A elasticidade é uma das maiores qualidades das nuvens de computadores. A elasticidade pode ser definida como a capacidade da nuvem prover recursos dinamicamente, de acordo com a demanda dos clientes. O NIST [48], ou instituto nacional de padrões e tecnologia dos Estados Unidos define a elasticidade da seguinte forma:

"Capacidades podem ser rapidamente e elasticamente providas, em alguns casos de forma automática, para rapidamente aumentar a capacidade e rapidamente diminuir a capacidade. Para o consumidor, as capacidades disponíveis aparentam ser ilimitadas e podem ser adquiridas em qualquer quantidade e a qualquer instante de tempo."

Ou seja, a elasticidade envolve o processo de conseguir alocar e desalocar recursos de forma dinâmica. Ao aplicar este conceito em nuvens de computadores, definimos elasticidade como a capacidade de prover recursos computacionais como processamento, memória e armazenamento de forma dinâmica e de migrar máquinas virtuais. E a utilização da elasticidade precisa envolver de alguma forma a virtualização que desacopla a configuração da máquina do cliente do *hardware* físico.

3.2 Alocação Dinâmica em Nuvens

Diversos trabalhos abordam o problema de alocação de elementos virtuais em substratos físicos, porém muitas destas propostas focam no desafio de controlar a admissão de elementos virtuais, sem considerar a possibilidade do consumo de recursos variar e, consequentemente, requerer uma realocação dinâmica destes elementos. Fajjari *et al.* desenvolveram um sistema de admissão de elementos baseado em meta-heurísticas de colônias de formigas para resolver este tipo de problema [49]. Alkmin *et al.* desenvolveram algoritmos de mapeamento que minimizam a utilização de recursos no ambiente de redes virtuais [50].

O SandPiper [10] é um sistema que monitora máquinas virtuais com o objetivo de detectar pontos quentes (*hotspots*) ou gargalos nos servidores físicos e atuar de forma a combatê-los. Os pontos quentes são definidos como indisponibilidade de recursos em servidores físicos que podem afetar o serviço das máquinas virtuais que compartilham estes servidores. Os resultados apresentados pela proposta demonstram que pontos quentes singulares conseguem ser detectados e eliminados em menos de 20 segundos e que a proposta pode ser estendida para ambientes de grandes *data centers*. Além disso, a proposta sugere duas abordagens possíveis para o monitoramento. A primeira abordagem é a abordagem caixa-preta, onde o monitoramento ocorre de forma independente do sistema operacional e das aplicações que executam nas máquinas virtuais. A segunda abordagem é a abordagem caixa-cinza, que explora a execução dos processos e aplicações dos sistemas operacionais das máquinas virtuais. Pode-se realizar esta segunda abordagem através da instalação de ferramentas de monitoramento em cada uma das máquinas virtuais.

O SandPiper realiza o monitoramento de máquinas virtuais na plataforma de virtualização Xen e utiliza equações de predição para estimar os pontos quentes. As equações de predição levam em consideração os valores de medidas anteriores, o valor médio da série temporal que representa o consumo de recursos e um parâmetro ϕ que representa variações nas séries temporais.

Ao detectar estes pontos quentes, o SandPiper aplica um algoritmo iterativo que ordena todos os servidores em função do seu volume, definido como

$$Vol = \frac{1}{1 - cpu} \cdot \frac{1}{1 - memória} \cdot \frac{1}{1 - rede}, \tag{3.3}$$

que é um valor gerado pelo sistema para indicar o volume de recursos consumidos na máquina em função do consumo de processamento, memória e rede. Os parâmetros CPU, memória e rede correspondem à utilização destes recursos normalizada pelo número de processadores existentes, uso de rede e memória disponível.

Em seguida, o algoritmo ordena, dentro de um mesmo servidor, os elementos

virtuais que consomem mais recursos. Desta forma, o sistema aloca iterativamente os elementos virtuais pertencentes a servidores de maior volume em servidores de menor volume, até que os pontos quentes sejam mitigados. Nesta dissertação é proposto o sistema VOLTAIC (*Volume Optimization Layer To AssIgn Cloud resources*). A alocação de recursos do VOLTAIC é significativamente diferente da do SandPiper, pois ela leva em consideração a compatibilidade dos perfis de uso das máquinas físicas e dos elementos virtuais. Além disso, o VOLTAIC utiliza uma métrica de volume diferente, que pode atribuir diferentes pesos a cada parâmetro e levar outros parâmetros em consideração (como a temperatura das máquinas, existência de mecanismos de tolerância a falhas, etc.) e indicar a necessidade de executar os algoritmos de gerência antes que a situação se torne crítica. O VOLTAIC permite também, em casos extremos, a utilização de uma adaptação do mecanismo de punição adaptativa proposto em [51] para garantir que os algoritmos do VOLTAIC consigam o processamento necessário para a realização de seus objetivos.

O Violin [52] apresenta uma proposta semelhante ao SandPiper. O Violin objetiva fornecer um ambiente capaz de escalar recursos automaticamente e migrar elementos de forma transparente para os usuários. O foco da proposta reside na utilização do mecanismo de *ballooning* de memória, que permite a alocação dinâmica de memória e de escalonamento de processadores na plataforma de virtualização Xen para entregar aos elementos virtuais os recursos que eles de fato podem vir a precisar. O ambiente proposto é composto de máquinas virtuais conectadas através de uma rede virtual, que permite a separação da configuração e da administração do Violin da configuração e administração da infraestrutura física. Baseado na ideia de federação de recursos computacionais como a infraestrutura compartilhada e distribuída do PlanetLab [53], o Violin permite que diversas instâncias sejam criadas para diferentes usuários que podem acessar recursos heterogêneos de forma isolada.

Cada instância da proposta recebe um ambiente computacional isolado de máquinas virtuais interconectadas por redes virtuais. Na visão do usuário das instâncias, o ambiente computacional fornecido é um *cluster* de máquinas dedicadas. O usuário não tem consciência da localização física real de cada uma das suas máquinas virtuais.

O Violin é dividido em dois componentes principais:

1. **Mecanismos de habilitação:** O mecanismo de habilitação inclui os ambientes virtuais fornecidos aos usuários e os processos de monitoramento de recursos das máquinas físicas. Estes processos monitoram o consumo de processamento e memória através de chamadas ao hipervisor para detectar a disponibilidade de recursos;

2. **Gerente de adaptação:** O gerente de adaptação se comunica com os proces-

sos de monitoramento para formar uma visão global dos recursos disponíveis. Esta visualização envolve as informações de monitoramento de todas as instâncias do Violin.

A proposta não foca na alocação ótima de elementos virtuais, mas em um mecanismo baseado em políticas de realocação que verifica se um elemento virtual pode ter suas políticas atendidas no nó físico corrente. Em caso negativo, ele migra o elemento virtual para outro nó físico em um algoritmo semelhante ao utilizado no SandPiper [10], onde as máquinas virtuais que consomem mais recursos são realocadas em máquinas que possuem recursos disponíveis.

Gong *et al.* desenvolveram o PRESS (*PRedictive Elastic ReSource Scaling for cloud systems*) [54]. O sistema surge para atender as demandas de computação em nuvem, onde a elasticidade deve minimizar os custos dos provedores ao mesmo tempo em que garante objetivos de níveis de serviço (*Service Level Objectives* - SLOs), que são definidos como elementos chave dos SLAs estabelecidos entre provedores e clientes. Os SLOs são acordados como uma forma de medir o desempenho do provedor de serviço, de forma que o provedor e o cliente possam atestar se os contratos estabelecidos foram cumpridos [55]. O grande desafio da elasticidade é decidir quantos recursos alocar, o que é um problema não trivial pois as demandas das aplicações variam no tempo e assim a sua caracterização em tempo de execução é muito complexa. Além disso, a elasticidade precisa alocar recursos antes de eles serem requisitados para evitar a violação de SLOs. O objetivo do PRESS é, portanto, desenvolver um esquema de predição sem a necessariamente envolver mecanismos complexos de perfil de uso, calibração de modelos ou entendimento profundo do comportamento de cada aplicação.

Para prever a demanda de recursos, o PRESS utiliza duas técnicas complementares. A primeira delas é a utilização de técnicas de processamento de sinais para identificar padrões repetitivos, chamados de assinaturas, que são utilizados na predição. Se nenhuma assinatura for detectada por estas técnicas, o PRESS utiliza uma abordagem estatística orientada a estados para capturar padrões de curto prazo, e utiliza cadeias de Markov para predizer o futuro próximo do consumo de recursos. A proposta evita a subestimação e tolera sobrestimação de recursos pois a subestimação apresenta uma maior possibilidade de gerar violações de SLO.

O sistema PRESS foi desenvolvido em um ambiente que utiliza a plataforma Xen e foi testado com sequências de amostras obtidas de dados de clusters de processamento do Google. O sistema apresenta bons resultados de predição, antevendo a necessidade de se migrar determinados elementos virtuais. Uma importante contribuição dos autores é a ideia de assinaturas de uso, que levam em conta a variação do perfil das máquinas no tempo como uma métrica para saber o quão confiável é utilizar o perfil atual para estimar demandas futuras. O PRESS apresenta bons re-

sultados na predição. Porém, ao contrário do mecanismo VOLTAIC proposto nesta dissertação, o PRESS não implementa mecanismos de realocação automática.

Hirofuchi *et al.* desenvolveram um mecanismo de migração baseado em pós-cópia para otimizar a migração em ambientes de nuvens [56]. Hirofuchi argumenta que os mecanismos atuais de migração e de otimização de nuvens são baseados em migrações com algoritmos de pré-cópia, que podem requerer uma grande quantidade de tempo para concluir os processos de migração. Desta forma, dependendo do padrão de consumo de memória da máquina virtual, este processo pode demorar um tempo difícil de ser estimado. Desta forma os autores dizem que a migração de máquinas com uma alta dinâmica e velocidade, feita de forma a atender a mudanças bruscas de carga seria prejudicial ao desempenho das aplicações. O sistema proposto é baseado em pós-cópia. Desta forma, a máquina não precisa ser pausada para a cópia final de memória e o tempo total da migração pode ser estimado, pois dependerá somente da capacidade do enlace que conecta a máquina de origem a máquina de destino e o tamanho da memória da máquina virtual. Se alguma página de memória não transferida for requisitada no destino, esta página é transferida para o destino com uma maior prioridade.

Após a definição e implementação da migração com pós-cópia, Hirofuchi *et al.* desenvolvem um mecanismo que utiliza a pós-cópia e um algoritmo de consolidação para *data centers* implementado com suporte a tecnologia de virtualização KVM. O sistema de consolidação é formado pelo monitor de cargas, o planejador de alocação e os módulos de controle das máquinas virtuais.

1. **Monitor de cargas:** O monitor de cargas recebe estatísticas de uso de cada servidor. Esta informação é extraída do `/proc/` do sistema Linux hospedeiro e da interface de monitoramento do KVM. As estatísticas são armazenadas em um banco de dados SQLite;

2. **Planejador de alocação:** O planejador de alocação recupera as informações do banco de dados, determina se existem servidores sobrecarregados e calcula um plano de re-alocação;

3. **Controlador de máquina virtual:** O controlador executa a migração ao vivo de acordo com o planejador de alocação. As máquinas virtuais são controladas através de chamadas XML-RPC. O controlador de máquinas virtuais também pode suspender e reiniciar máquinas físicas. Quando uma máquina física não abriga máquinas virtuais, o controlador coloca o servidor no modo dormindo *sleep*. Quando o planejador de alocação indica que a máquina física precisa ser acordada, o controlador ativa a máquina física.

Para validar o sistema, os autores consideram que uma máquina virtual precisa

de um processador físico e 1.7 GB de memória. Este valor foi definido de acordo com o serviço IaaS fornecido pela Amazon EC2, que possui um núcleo virtual e 1.7GB de memória. No sistema proposto, cada servidor pode pertencer a duas modalidades diferentes. Os servidores compartilhados e os servidores dedicados. As máquinas virtuais são migradas entre estas duas modalidades, dependendo do seu padrão de uso de recursos. O *hardware* das duas modalidades de servidor são semelhantes, sendo a única diferença a quantidade de memória, que é maior nos servidores compartilhados. Esta quantidade de memória é suficiente para hospedar muitas máquinas virtuais ociosas, porém, o número de processadores não é suficiente para alocar exclusivamente um núcleo físico para cada máquina virtual.

Quando uma máquina virtual utiliza mais do que os servidores compartilhados podem oferecer, o sistema acorda uma máquina dedicada e migra a máquina virtual para ela. O algoritmo de consolidação funciona da seguinte forma. Inicialmente todas as máquinas virtuais estão alocadas em servidores compartilhados. Quando a média de utilização de processamento de algum servidor compartilhado atinge 90%, a máquina virtual que está consumindo mais processamento é migrada para um servidor dedicado. Após uma dada migração, a máquina migrada não pode ser movida por 20 segundos para evitar reações desnecessárias no sistema. A máquina migrada continua no servidor dedicado até que sua média de processamento atinja 50%. Se isso acontecer, o sistema tenta encontrar um servidor compartilhado que consiga comportar a máquina virtual. O sistema apresenta bons resultados e dá foco na comparação da migração com pré-cópia e pós-cópia. A proposta em si não consegue prever com eficiência a demanda de recursos e utiliza dois limiares, um de saturação e um de ociosidade para migrar máquinas entre servidores compartilhados e dedicados. Desta forma, podem ocorrer situações onde não existem servidores dedicados suficientes para determinadas configurações e nesses casos uma realocação intra servidores dedicados ou intra servidores compartilhados poderia atenuar o problema, de forma semelhante à proposta VOLTAIC desta dissertação, que não diferencia os servidores em categorias distintas.

Houidi *et al.* apresentam um mecanismo adaptativo para provisão de recursos voltado para o cenário de redes virtuais [57]. Os autores acreditam que da mesma forma que máquinas virtuais podem ser alocadas de forma dinâmica em diferentes máquinas físicas, redes virtuais também podem ser dinamicamente alocadas em diferentes máquinas físicas. Ao estender esta alocação para o cenário de redes, deve-se levar em consideração uma série de novos parâmetros, como a conectividade entre cada elemento virtual, os atrasos de cada enlace virtual e a topologia de rede virtual. O mecanismo adaptativo tem o objetivo de realocar dinamicamente redes virtuais em resposta a requisições de criação de novas redes. O sistema proposto é distribuído e baseado em agentes que monitoram os elementos físicos. Estes agentes

também detectam falhas de enlace e podem realizar mudanças de acordo com estas falhas e com mudanças no padrão de consumo de recursos de cada rede virtual.

Houidi *et al.* propõem um cenário de provisão de recursos adequado ao mecanismo por eles proposto. Este cenário é composto por provedores de redes virtuais, que atuam como *brokers* que requisitam, negociam e adquirem recursos virtuais de provedores de infraestrutura física em nome de provedores de serviço, operadores de redes virtuais e clientes finais. Desta forma, os provedores de redes virtuais atendem a diferentes tipos de clientes e negociam a alocação de recursos desses clientes com provedores de infraestrutura física. Para este cenário funcionar, inicialmente os provedores de infraestrutura física precisam descrever e ofertar os recursos físicos que eles desejam disponibilizar. Esta oferta de recursos físicos envolve atributos funcionais como o tipo de enlace e parâmetros não funcionais como a capacidade do enlace atual. Dado a requisição de algum cliente, os provedores de redes virtuais realizam uma etapa de descoberta e verificação (*matching*) que consiste na busca e detecção de candidatos aptos a fornecerem as demandas do cliente que atendam aos requisitos feitos pelo cliente. Os provedores de infraestrutura são organizados de acordo com a oferta de recursos e um algoritmo de classificação associa a demanda dos clientes aos provedores de infraestrutura que possuem uma oferta de recursos o mais semelhante possível com a necessidade do cliente. Dado a demanda de todos os clientes, os provedores de redes virtuais desejam alocar o maior número possível de redes virtuais em cima de um mesmo substrato, que reduziria o custo de cada cliente e aumentaria o lucro dos provedores.

Dado que as redes virtuais estão alocadas, Houidi analisa a questão da provisão dinâmica dos recursos. A proposta não considera que a realocação de redes virtuais deve ocorrer de acordo com o comportamento de cada rede. A proposta requer que as mudanças das redes virtuais ocorram mediante a requisição dos administradores da rede, que podem desejar aumentar a rede, diminuir o tamanho da rede ou modificar atributos e valores da rede virtual. Ou seja, se uma determinada rede sofre uma variação de carga que passe a exigir mais recursos, a rede permanecerá sem estes recursos até que o administrador da rede opte por solicitar a alocação de mais recursos. Dado que alguma mudança foi solicitada pelo administrador da rede virtual, como um aumento do número de nós de uma rede virtual, o provedor de rede virtual busca nas vizinhanças da rede do cliente algum provedor de infraestrutura que possua uma oferta de recursos compatível. Caso nenhum provedor próximo seja encontrado a busca é estendida. Um outro caso considerado é quando algum elemento da rede virtual apresenta alguma falha ou degradação de desempenho relacionada a falhas no provedor de infraestrutura. Nesse caso, o mecanismo proposto seleciona novos nós ou enlaces e remapeia a rede dos clientes para contornar o problema. O estado dos elementos do sistema são mantidos através de mensagens do tipo *keep*

alive. No caso de falhas, executa-se um algoritmo que verifica se o provedor atual possui recursos para contornar as falhas. Em caso positivo, o mapeamento é feito de forma automática. Caso contrário, a busca é estendida a outros nós próximos.

Por fim, Hugo Carvalho, Natalia Fernandes e Otto Duarte propõem um mecanismo de controle adaptativo de SLAs para ambientes de redes virtuais [51] [58]. O mecanismo se baseia em punições adaptativas para elementos virtuais para garantir o cumprimento dos SLAs e o melhor uso dos recursos ociosos. O mecanismo proposto utiliza o perfil de máquinas virtuais somente para verificar se elas ultrapassam as SLAs contratadas e se limita à distribuição de recursos dentro de um único servidor físico. Outra contribuição é a adoção de uma métrica nebulosa para medir o grau de saturação de recursos físicos de cada nó físico, definida como carga do sistema. Esta carga do sistema realiza uma combinação de funções de inferência e regras nebulosas [59] [60] para gerar uma nota de carga do sistema, que reflete parâmetros como o uso de processador, memória e rede. Esta métrica foi adaptada ao VOLTAIC como critério para execução dos mecanismos de alocação dinâmica. Quando esta nota de carga do sistema passa de um determinado limiar, os algoritmos são invocados. Além disso, o VOLTAIC permite o controle de recursos em um nível mais amplo, pois permite a utilização da primitiva de migração de máquinas virtuais para melhorar a provisão de recursos das máquinas virtuais. Desta forma, o controle de recursos não é restrito a uma máquina física e permite a alocação de recursos em um grande número de máquinas em um ambiente de nuvens. Os perfis não são usados para verificar violações de SLA e sim para compreender a variação e a correlação do comportamento das máquinas em relação a elas mesmas em função do tempo, pois este é um dos critérios utilizados na escolha de candidatos a migração do sistema VOLTAIC proposta nesta dissertação.

O sistema VOLTAIC proposto é um gerente autônomo de recursos de ambientes de computação em nuvem, que aloca de forma inteligente os elementos virtuais e aumenta a qualidade de serviço oferecida aos clientes ao evitar o desperdício de recursos computacionais. O sistema utiliza a `libvirt` para interagir com as plataformas de virtualização e gerenciar servidores físicos. Como contribuições, a proposta aplica controladores nebulosos para detectar a saturação de máquinas físicas e propõe algoritmos de realocação automática que levam em consideração o perfil de consumo de cada máquina virtual e o perfil de recursos oferecidos por cada máquina física, tentando garantir que uma máquina virtual seja alocada na máquina física que forneça os recursos necessários de forma mais efetiva.

Além disso, o VOLTAIC é compatível com quaisquer plataformas de virtualização que suportem a `libvirt` tais como Xen, VMWare, KVM, etc., diferente das propostas existentes que funcionam apenas em ambientes virtualizados específicos [10], [52], [54] e [51]. Ao adotar a utilização de uma interface única para gerência

de plataformas virtuais, amplia-se a aplicabilidade do sistema proposto, mas isso acarreta algum prejuízo quanto ao tipo de dado que pode ser extraído pela plataforma devido a limitações da `libvirt`. Apesar disso, os resultados demonstram que a falta de algumas informações específicas das plataformas de virtualização não oneram sobremaneira o desempenho da proposta. As plataformas que podem se servir do sistema VOLTAIC devem prover mecanismos de migração ao vivo de máquinas.

3.3 Considerações Importantes

A elasticidade é uma técnica que permite a alocação dinâmica de recursos em ambientes de nuvens de computadores. Esta técnica é uma das maiores novidades da computação em nuvens, pois permite que os usuários utilizem os recursos necessários de forma dinâmica, evitando o desperdício de recursos e altos custos iniciais. Além disso, a elasticidade permite um maior controle sobre os serviços oferecidos, pois se pode ampliar o poder computacional alocado a cada usuário, mesmo que este aumento não tenha sido previsto. Para utilizar a elasticidade é necessário utilizar algumas técnicas que permitem a alocação dinâmica de recursos. A primeira técnica necessária é a virtualização. A virtualização implementa uma camada de abstração entre o *hardware* e o *software* que fornece ao *software* uma visão padronizada de *hardware*. Assim, um mesmo *software* pode executar em diferentes máquinas físicas, independente de variações no *hardware*. Ao utilizar a virtualização, pode-se controlar dinamicamente as políticas de escalonamento de processamento, acesso a memória e acesso ao disco. Além disso, a migração de máquinas virtuais permite que os serviços, em execução, sejam transferidos entre máquinas físicas diferentes de forma transparente.

Capítulo 4

Lógica Nebulosa

No contexto desta dissertação, a utilização da lógica nebulosa permite o desenvolvimento de controladores flexíveis que auxiliam a tomada de decisões de acordo com regras qualitativas definidas pelos gerentes de nuvens[1]. Desta forma, torna-se necessário entender os conceitos básicos e fundamentos da lógica nebulosa para a completa compreensão da proposta desta dissertação.

A lógica nebulosa é uma ferramenta que lida com problemas de tomada de decisão que envolvem incertezas, dados imprecisos e informações qualitativas. Diferentemente da lógica booleana clássica, onde um elemento pertence ou não pertence a um dado conjunto, a pertinência nebulosa de um dado elemento a um conjunto é dada através de um grau de pertinência, ou valor nebuloso, que está contido no intervalo $[0, 1]$.

Um exemplo de pertinência na lógica clássica pode ser visto na Fig. 4.1. Neste exemplo, torna-se trivial classificar elementos como pertencentes ou não ao conjunto que representa os seres vivos. Neste caso, o cálculo da pertinência é simples e é dado por $\mu(x) : X \rightarrow \{0, 1\}$. Ou seja, o elemento X pertence totalmente ou não pertence ao conjunto dos seres vivos.

A lógica clássica, porém, não trata alguns casos específicos. Ao considerar a mesma classificação de ser vivo, existe a dificuldade de se classificar um vírus. Como um vírus possui diversas propriedades que o enquadram tanto como um ser vivo quanto como um ser não vivo, a lógica clássica não consegue representar esta configuração. Ao observarmos este mesmo problema sob a ótica da lógica nebulosa porém, percebe-se que é possível classificar o vírus como um ser vivo com um dado grau de pertinência, resolvendo o problema de classificação. Este exemplo pode ser visto na Fig. 4.2.

O grau de pertinência de um elemento está relacionado à intensidade com que este elemento se relaciona com um dado conjunto e esta intensidade é obtida através

[1] A definição e os conceitos relativos à lógica nebulosa são adaptadas do artigo [51] e do projeto de conclusão de curso do autor desta dissertação [61].

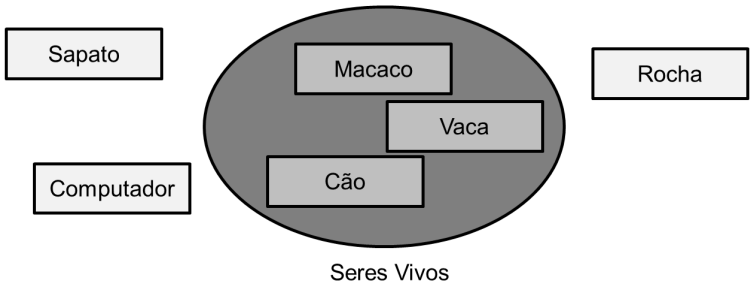


Figura 4.1: Pertinência em conjuntos da lógica clássica. Um elemento pertence ou não pertence a um dado conjunto.

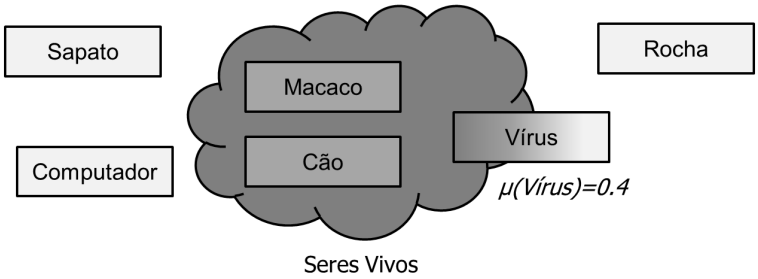


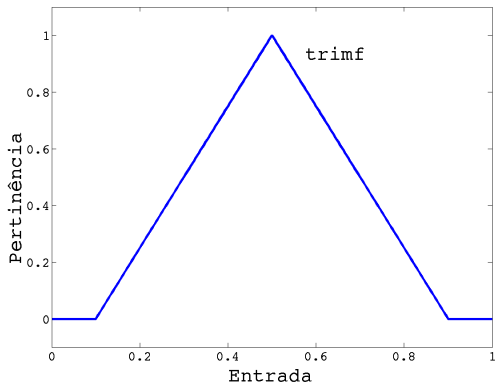
Figura 4.2: Pertinência em conjuntos da lógica nebulosa.

da aplicação de funções de pertinência.

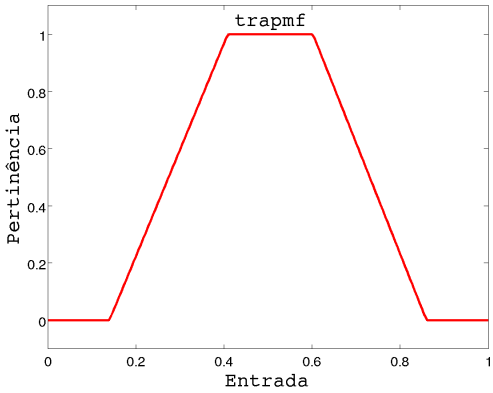
4.1 Funções de Pertinência

As funções de pertinência são funções que mapeiam variáveis de entrada reais em variáveis nebulosas, contidas no intervalo $[0, 1]$. Os sistemas nebulosos geralmente lidam com dados imprecisos e informações qualitativas, portanto as funções de pertinência exercem a tarefa de transformar estes dados imprecisos em informações quantitativas passíveis de serem utilizadas na tomada de decisão.

Um exemplo de duas possíveis funções de pertinência pode ser visto na Fig. 4.3. Neste exemplo, observam-se dois modelos de funções de pertinência muito utilizados nos sistemas nebulosos. A função trimf representa uma função triangular enquanto que trapmf representa uma função trapezoidal. O eixo horizontal corresponde ao valor da entrada real do sistema enquanto o eixo vertical representa como este valor de entrada é mapeado na lógica nebulosa.



(a) Função de pertinência triangular.



(b) Função de pertinência trapezoidal.

Figura 4.3: Alguns modelos de funções de pertinência que podem ser utilizados em sistemas nebulosos.

4.2 Controladores Nebulosos

Uma das aplicações mais importantes da lógica nebulosa são os sistemas de inferência nebulosa. Estes sistemas utilizam regras do tipo SE → ENTÃO, que são aplicadas sobre variáveis nebulosas para se representar o conhecimento e as estratégias qualitativas desejadas. No contexto desta dissertação, os administradores das nuvens podem definir regras de forma qualitativa através de descrições linguísticas simples, representadas através destas regras [62].

Um dado conjunto de regras de inferência é chamado de base de regras. Existem duas formas principais de se obter estas regras. A primeira delas é através da captura da experiência de operadores de sistema ou gerentes, enquanto a segunda delas é através do monitoramento e do uso de mecanismos de inteligência computacional para modelar estas regras. No escopo desta dissertação, as regras são obtidas através da primeira forma, com a captura de experiência de operadores de rede. O processo envolvido no mapeamento de variáveis do mundo real em variáveis nebulosas é também chamado de sistema de inferência nebuloso (*Fuzzy Inference System - FIS*).

Um exemplo de sistema de inferência nebuloso pode ser visto na Fig. 4.4. Inicialmente o sistema recebe entradas reais, chamadas de entradas *crisp*, que são de fato as variáveis de entrada capturadas pelo sistema (passo um). Em seguida, são aplicadas funções de pertinência que mapeiam as variáveis de entrada em valores nebulosos, em um processo chamado de *fuzzificação* (passo dois). Dados os valores nebulosos, são aplicadas as regras do tipo SE → ENTÃO que avaliam as saídas do controlador (passo três). Em seguida, verifica-se que regras foram ativadas, agregam-se os resultados e estes então são mapeados novamente em variáveis reais (passo quatro), que podem ser utilizadas para regular o sistema (passo cinco). A defuzzificação do

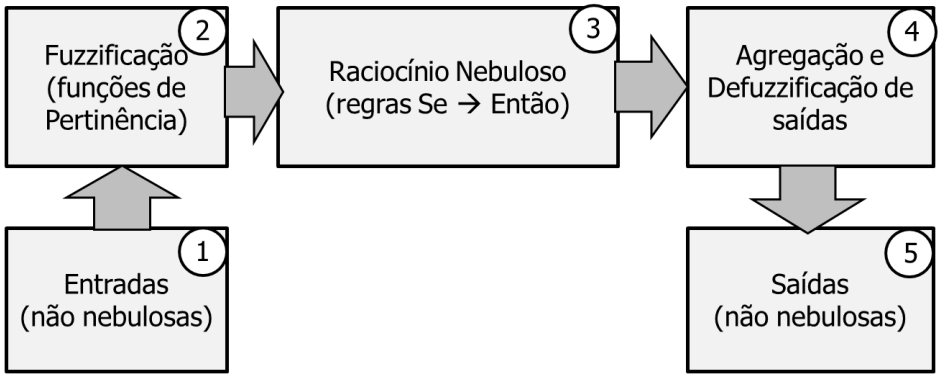


Figura 4.4: Sistema de inferência nebuloso.

passo quatro pode ser realizada de diferentes formas. Por exemplo, pode-se calcular o centro de massa do polígono gerado (centróide), a média dos maiores valores das funções de pertinência de saída (Média dos Máximos), ou simplesmente o maior valor de saída de todas as funções de saída (Maior dos máximos).

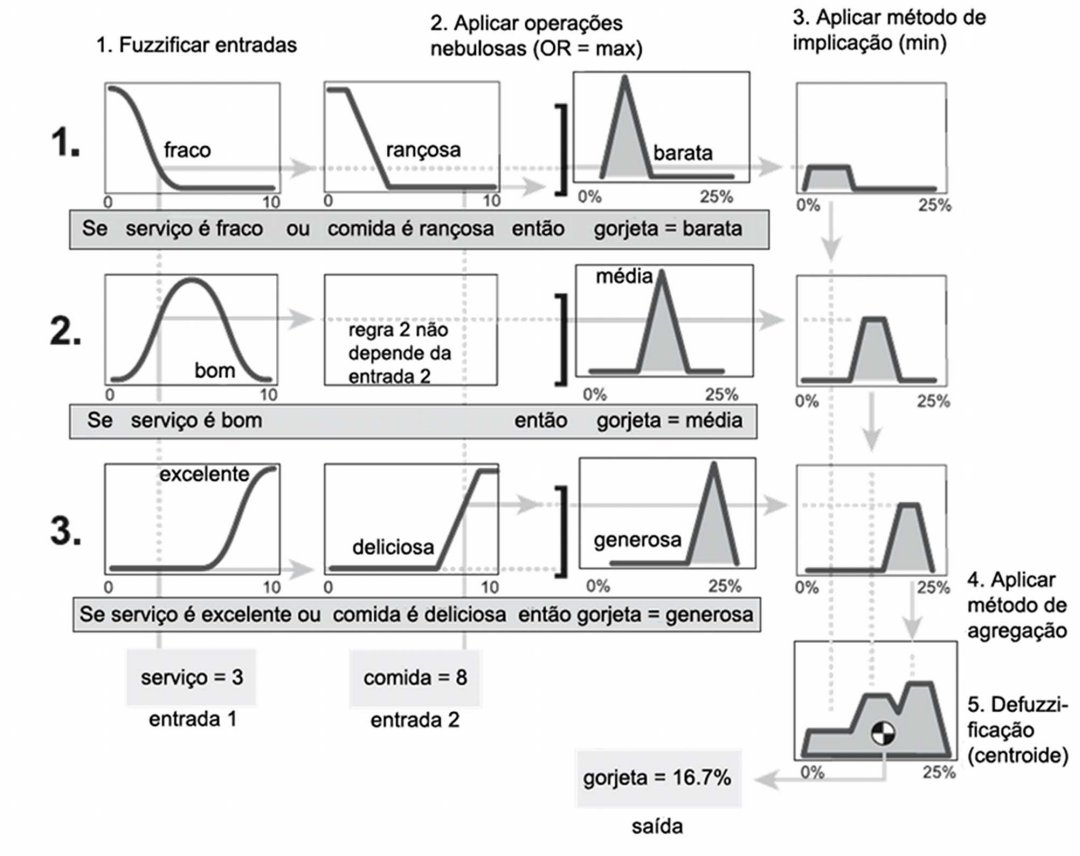


Figura 4.5: Sistema de inferência nebuloso para calcular a gorjeta de um serviço. Adaptado de [63].

Um exemplo muito utilizado para explicar o funcionamento de um controlador nebuloso e que facilita a compreensão dos controladores nebulosos é o exemplo da

gorjeta, conforme pode ser visto na Fig. 4.5. Um dado indivíduo deseja determinar quanto de gorjeta (qual percentual do valor cobrado na nota fiscal) ele deve dar a um garçom de um dado restaurante, dependendo da qualidade do serviço e da qualidade da comida. Para resolver este problema, este indivíduo desenvolveu um sistema nebuloso, que recebe como entradas uma nota entre zero e dez para o serviço e uma nota entre zero e dez para a comida. O indivíduo então adota funções de pertinência que definem se o serviço é fraco, bom ou excelente. Também são determinadas funções de pertinência para definir se a comida está rançosa (péssima qualidade), razoável ou deliciosa. E por fim, são definidas funções de saída, que qualificam a gorjeta como barata, média ou generosa. Após a definição destas funções de pertinência, o indivíduo insere algumas regras SE $\rightarrow$ ENTÃO no sistema de inferência. Estas regras definem que se o serviço é fraco ou a comida é rançosa então a gorjeta é barata. Se o serviço é bom, independentemente da qualidade da comida, a gorjeta é média. E se o serviço é excelente ou a comida é deliciosa então a gorjeta é generosa. Pode-se observar o funcionamento deste sistema na figura Fig. 4.5, para o caso onde a nota dada para o serviço foi três e a nota para a comida foi oito. Inicialmente, estas variáveis são mapeadas em valores nebulosos, no processo de fuzzificação. Pode-se observar que o serviço nota três ativou a função de pertinência de “serviço bom e levemente a função de "serviço fraco". A comida nota oito ativou a função de pertinência de comida deliciosa. Ao aplicar-se as regras de inferência nestas variáveis nebulosas, percebe-se que cada uma das três regras foi ativada com certa intensidade. Em seguida estas intensidades são aglutinadas por um dado método de agregação (por exemplo, desenhar o polígono que comporta a intensidade de cada regra). Finalmente, é utilizado um método de defuzzificação como o centróide que calcula o centro de massa do polígono resultante, que dará qual é a percentagem da gorjeta que deverá ser paga ao garçom.

4.3 Comparação entre Lógica Nebulosa e Outros Mecanismos de Inteligência Artificial

A inteligência artificial reúne um conjunto de mecanismos que visam a representação do conhecimento humano em ambientes de tomada de decisão. Dentre as diversas áreas de estudo, pode-se realizar uma classificação que divide a inteligência artificial em duas abordagens principais. A primeira abordagem é a baseada em aprendizado ou técnicas de aprendizado. A segunda abordagem é a baseada na inserção e no mapeamento de conhecimento humano em algoritmos funcionais. Não existem restrições quanto a relação entre as abordagens, de forma que podem existir abordagens híbridas que utilizam por exemplo, as etapas de treinamento e

aprendizado para extrair conhecimentos e aplicá-los no mapeamento em algoritmos funcionais. A abordagem baseada em aprendizado, como o nome diz, precisa receber uma série de dados de treinamento, que permitirão que o sistema aprenda como agir baseado em eventos passados. Desta forma, existe uma etapa de aprendizado ou treinamento, onde os sistemas recebem um número significativo de pares de entrada e saída, que alimentam o sistema e o preparam para interpretar novos dados. A cada iteração no treinamento, é calculado um erro que indica a diferença entre o valor real da saída e a gerada pelo sistema. Assim, o objetivo da etapa de treinamento é minimizar este erro através de ajustes no sistema a cada iteração. Existem alguns casos onde a obtenção de um conjunto de treinamento pode ser muito onerosa, devido à quantidade de variáveis envolvidas e à quantidade de dados reais que precisam ser fornecidos para que o sistema tenha uma boa taxa de acertos. Além disso, antes de serem utilizados, os sistemas obrigatoriamente precisam passar pela etapa de treinamento, que dependendo da tecnologia e dos tamanhos dos conjuntos de treinamento, pode demorar um tempo significativo. Dentre as técnicas existentes baseadas em treinamento, podem-se destacar as redes neurais, algoritmos genéticos e máquinas de vetor de suporte. A segunda abordagem baseada no mapeamento do conhecimento engloba a lógica nebulosa. A lógica nebulosa funciona de forma eficiente em cenários onde existe o conhecimento humano sobre como resolver determinado problema, porém existe uma dificuldade em se mapear este conhecimento em modelos computacionais existentes. Além disso, a lógica nebulosa, assim como as demais técnicas de inteligência artificial, funciona bem em casos onde a modelagem matemática é muito complexa ou impossível de se obter, por exemplo, em processos não lineares. Por fim, a lógica nebulosa é adequada a cenários onde existe certa imprecisão nas medidas extraídas do ambiente.

Ao assumir que os administradores de rede precisam passar seus conhecimentos para que as redes funcionem de forma adequada, e que cada administrador pode ter soluções qualitativas específicas (por exemplo, dando prioridade a garantir uma maior quantidade de recursos ociosos para aguentar picos de demanda ou permitir uma maior utilização de recursos para alocar mais clientes e aumentar os lucros, sob pena de poder afetar a qualidade do serviço), a adoção de sistemas nebulosos para controlar as redes atuais e futuras se apresenta como uma solução viável. Devido as mudanças constantes e a enorme quantidade de variáveis passíveis de análise em redes de computadores, a adoção de mecanismos de treinamento pode se tornar muito custosa, devido a necessidade de realizar constantemente etapas de treinamento e de se adquirir conjuntos de treinamento que reflitam as condições atuais da rede. Através do desenvolvimento de controladores personalizados, cada administrador consegue administrar domínios de forma autonômica e respeitando suas premissas e conhecimentos e em casos de mudança, o administrador pode modificar

dinamicamente o conhecimento inserido no sistema, sem a necessidade de re-treinar os controladores.

4.4 Aplicação da Lógica Nebulosa no Contexto da Dissertação

Esta dissertação propõe uma solução de gerência autonômica de nuvens de computadores. Esta gerência autonômica envolve o desenvolvimento de mecanismos capazes de monitorar as máquinas físicas presentes nas nuvens e detectar o esgotamento de recursos antes que ele cause degradação na qualidade do serviço oferecido aos clientes. Para detectar este esgotamento de recursos, pode-se monitorar uma série de parâmetros do sistema como a utilização de processamento, memória, utilização das interfaces de rede, temperatura do sistema, etc. Um administrador da nuvem pode portanto mapear a importância de cada um destes parâmetros através de funções de pertinência e de regras de inferência que facilitam o mapeamento do conhecimento qualitativo do administrador. Ao realizar este mapeamento, pode-se gerar uma saída não nebulosa que representa o grau de utilização de recursos de cada máquina física. Esta saída recebe o nome de carga do sistema. Ao monitorar a carga do sistema das diversas máquinas físicas, pode-se detectar as máquinas que estão na iminência de terem seus recursos esgotados. E ao detectar este problema, pode-se atuar pró-ativamente migrando as máquinas virtuais que executam nestes sistemas saturados para máquinas físicas menos saturadas. A proposta VOLTAIC desta dissertação propõe mecanismos que utilizam a análise desta carga do sistema para migrar de forma autonômica as máquinas virtuais em um ambiente de nuvem que segue o modelo de Infraestrutura como Serviço (IaaS).

Capítulo 5

O Sistema VOLTAIC

Os capítulos anteriores apresentaram o estado da arte em nuvens de computadores e também, mais especificamente em elasticidade e alocação dinâmica de recursos em ambientes de nuvens. Neste capítulo é descrita a proposta dessa dissertação, o sistema VOLTAIC (*Volume Optimization Layer To AssIgn Cloud resources*), que é um sistema de gerência automatizada de recursos para computação em nuvens que se aplica ao modelo de Infraestrutura como Serviço (IaaS).

No ambiente de nuvens, existem grandes conjuntos de máquinas físicas e virtuais e a gerência eficiente destas máquinas representa um problema complexo. Cada máquina virtual pode apresentar um padrão diferente de consumo de recursos. Desta forma a alocação correta destas máquinas virtuais dentre as máquinas físicas disponíveis apresenta um grande desafio, fundamental para garantir qualidade de serviço e o atendimento dos SLAs de cada máquina. Neste contexto surge a proposta VOLTAIC.

O nome do sistema proposto vem da natureza, onde diferenças de cargas elétricas entre nuvens podem resultar em uma troca de cargas que se dá através de um arco voltaico. Da mesma forma que as nuvens balanceiam a carga elétrica entre elas, a proposta balanceia e gerencia o consumo de recursos em ambientes de computação em nuvem. A gerência autonômica é caracterizada como o processo de observação e tomada de decisões no ambiente proposto, sem intervenção humana. Desta forma, dado um conjunto de máquinas físicas que executam uma plataforma de virtualização e suportam máquinas virtuais, o gerente autonômico é capaz de entender o comportamento destes elementos, observar se os comportamentos conduzem o sistema a situações de saturação e, nestes casos, tomar decisões para evitar a saturação. Estas decisões podem envolver a poda temporária de recursos [51] ou a migração ao vivo de elementos virtuais para uma máquina física menos sobrecarregada [44].

Um exemplo de funcionamento do VOLTAIC pode ser visto na Fig. 5.1 e na Fig. 5.2. Na Fig. 5.1, temos um exemplo básico de funcionamento do VOLTAIC. Existem duas máquinas físicas A e B. Estas duas máquinas físicas possuem máquinas

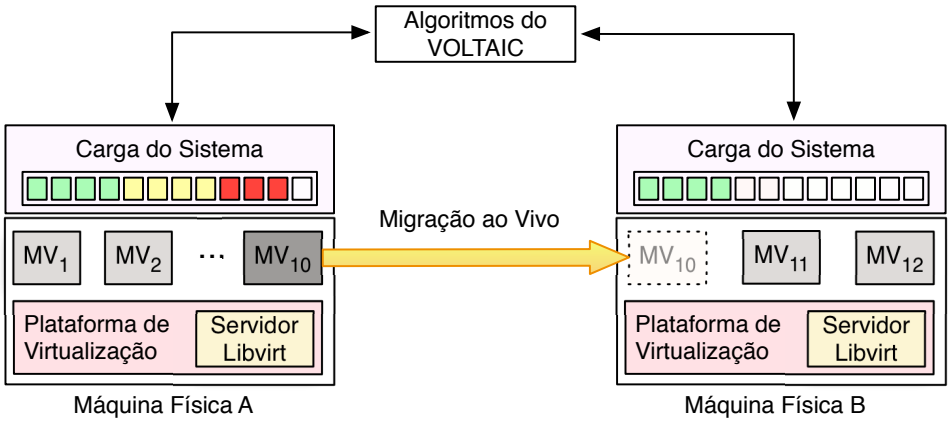


Figura 5.1: O VOLTAIC monitora as máquinas físicas e detecta uma máquina com carga elevada. Ele executa algoritmos de decisão e opta por migrar a máquina virtual 10 da máquina física A para a máquina física B.

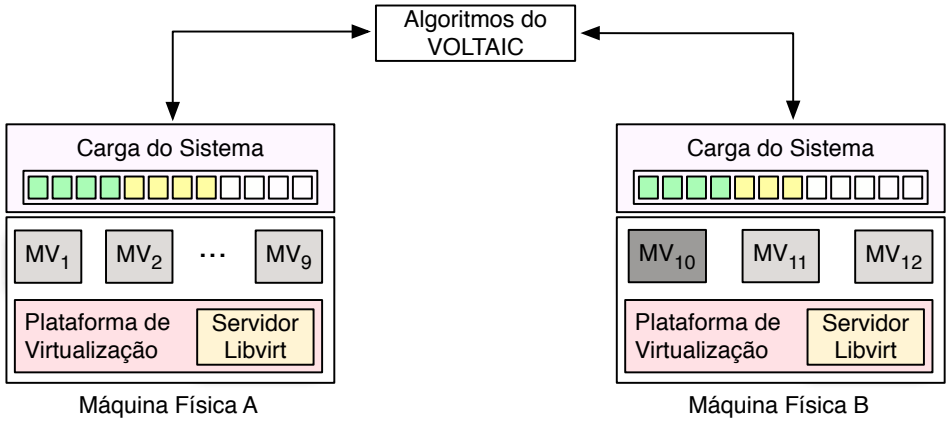


Figura 5.2: Balanceamento das máquinas. Após a migração feita pelo VOLTAIC, a carga do sistema de todas as máquinas físicas é balanceada.

virtuais numeradas de 1 até 12. As máquinas virtuais numeradas de 1 a 10 se encontram na máquina física A e as máquinas virtuais 11 e 12 se encontram na máquina física B. A caixa denominada Carga do Sistema utiliza um dos mecanismos[1] do VOLTAIC para estimar a utilização de recursos de cada máquina física. Um valor muito alto indica que a máquina física está próxima de utilizar todos os seus recursos e, portanto, ela provavelmente começará a degradar a qualidade do serviço das máquinas virtuais que estão alocadas na máquina física. No exemplo dado, a máquina física A está com a carga do sistema muito alta. Ao perceber isso, o sistema VOLTAIC detecta esta carga alta e inicia alguns mecanismos para tentar sanar esta carga alta. Nesta situação, a proposta encontrou uma máquina física B com recursos disponíveis e optou por migrar a máquina virtual 10 da máquina física A para a máquina física B. O resultado desta operação pode ser visto na

[1]Este mecanismo é o mecanismo de carga do sistema baseado em lógica nebulosa e será apresentado em mais detalhes no decorrer deste capítulo.

Fig. 5.2. Após a migração, o consumo de recursos foi balanceado entre as duas máquinas físicas, reduzindo a chance da qualidade de serviço das máquinas virtuais ser degradada.

Para realizar este tipo de operação de balanceamento exemplificado, a modelagem da proposta requer a utilização de diversos módulos. Cada módulo é responsável por uma das etapas envolvidas no processo. Inicialmente, é necessário monitorar as máquinas físicas e virtuais (i). Em seguida torna-se necessário analisar e extrair informações sobre as máquinas e o padrão de consumo (ii). Em seguida, deve-se raciocinar sobre estas informações e elaborar decisões para escolher o posicionamento adequado de cada máquina virtual (iii). Para atender a estas etapas, o sistema é dividido em três módulos principais: o Coletor de Estatísticas (CE), o Analisador de Perfis (AP) e o Orquestrador (OQ).

O Coletor de Estatísticas (CE) se responsabiliza por interagir com a `libvirt`[2] e obter as estatísticas de monitoramento de cada máquina física monitorada e armazenar esta informação em uma base de dados. O segundo módulo é o Analisador de Perfis (AP), que utiliza a informação armazenada pelo Coletor de Estatísticas para extrair conhecimento das plataformas de virtualização. Este conhecimento engloba perfis de consumo dos elementos virtuais, perfis de recursos oferecidos pelos servidores físicos (POs)[3], carga do sistema e o mapeamento dos elementos virtuais nos elementos físicos. O terceiro módulo é o Orquestrador que gerencia as máquinas físicas e virtuais a partir do conhecimento gerado pelo Analisador de Perfis. Ele pode controlar a quantidade de recursos oferecidos a cada máquina virtual e organizar migrações de elementos virtuais para balancear a distribuição de elementos virtuais nas máquinas físicas.

5.1 Arquitetura do VOLTAIC

Os três módulos mencionados anteriormente são instanciados dentro de uma máquina física dedicada à execução do VOLTAIC. Esta máquina pode ser associada a um conjunto de máquinas físicas que serão gerenciadas pela proposta. Pode-se observar esta organização na Fig. 5.3.

O VOLTAIC gerencia estas máquinas físicas, controla os recursos oferecidos em cada uma delas e migra dinamicamente elementos virtuais, evitando a saturação de recursos. Nesta modelagem, a quantidade de máquinas físicas que podem ser gerenciadas por cada máquina que executa o VOLTAIC depende da capacidade de processamento e da quantidade de mensagens de dados de monitoramento que po-

[2] A `libvirt` é uma biblioteca que permite a interação com múltiplas plataformas de virtualização.

[3] O perfil oferecido é o perfil de oferta de recursos de uma dada máquina física, ou seja, o perfil da disponibilidade de recursos da máquina física.

dem ser recebidos por esta máquina. Para utilizar a proposta em uma escala maior, torna-se necessário criar diversas unidades VOLTAIC, cada uma associada a um determinado conjunto de máquinas que precisam ser monitoradas. E os algoritmos e o balanceamento são realizados dentro destes grupos que são formados. Porém, esta utilização em maior escala acarreta um aumento significativo na complexidade do sistema para se realizar a sincroniza entre medidas e decisões.

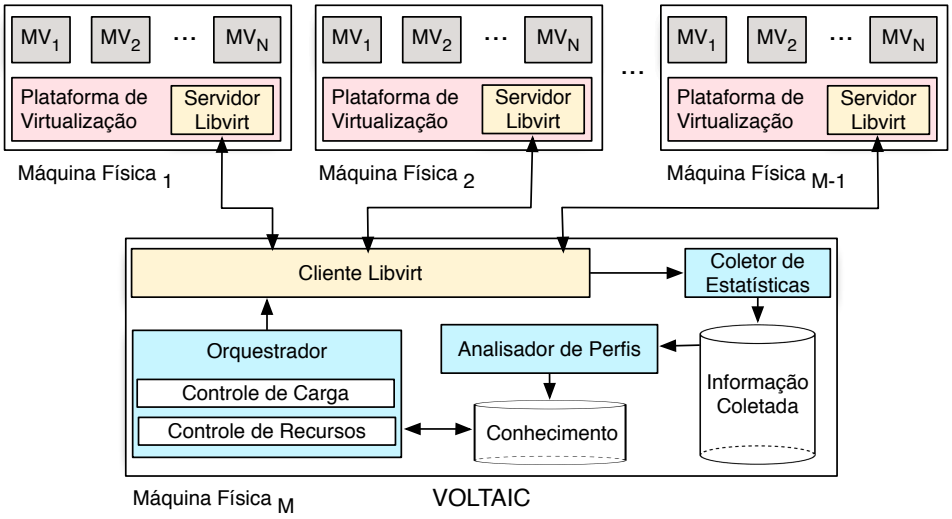


Figura 5.3: Arquitetura do VOLTAIC com um nó VOLTAIC gerenciando $M - 1$ servidores físicos e organizando a distribuição de recursos.

5.1.1 Coletor de Estatísticas

O Coletor de Estatísticas (CE) utiliza a interface da `libvirt` para interagir com as máquinas físicas e obter informações de monitoramento. O CE permite a coleta do uso de processamento, memória alocada e utilização das interfaces de rede da máquina física e das máquinas virtuais. Pode-se ajustar a frequência de monitoramento e a sua distribuição no tempo, de forma a evitar que eventos com periodicidade bem definida possam se sincronizar com períodos de inatividade do CE. A frequência de monitoramento está associada à velocidade de reação da proposta na ocorrência de saturação de recursos. A frequência de monitoramento também está associada à quantidade de trocas de mensagem por unidade de tempo e uma frequência muito alta pode saturar as interfaces de rede com o excesso de mensagens de controle.

5.1.2 Analisador de Perfis

O Analisador de Perfis (AP) processa as informações obtidas pelo Coletor de Estatísticas. A análise envolve a coleta de informações de quais elementos virtuais estão mapeados em quais elementos físicos e a geração de séries temporais que

refletem a relação temporal entre o consumo de recursos. São gerados perfis que utilizam distribuições de probabilidade de uso de recursos, medidas através de funções de distribuição de probabilidade (PDFs) e funções de distribuição de probabilidade cumulativa (CDFs). Os perfis de uso representam o padrão de consumo de recursos de cada máquina virtual. Pode-se observar na Fig. 5.4 e na Fig. 5.5 exemplos de perfis extraídos de uma máquina virtual que executa o protocolo de roteamento RIPv2. Os perfis refletem a forma que a máquina virtual utiliza recursos de processamento durante o tempo.

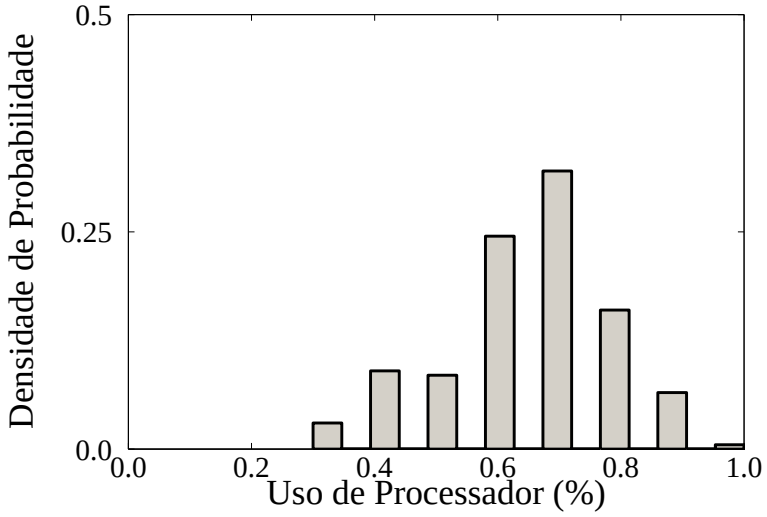


Figura 5.4: Função densidade de probabilidade (PDF) para uma máquina virtual executando o protocolo RIPv2.

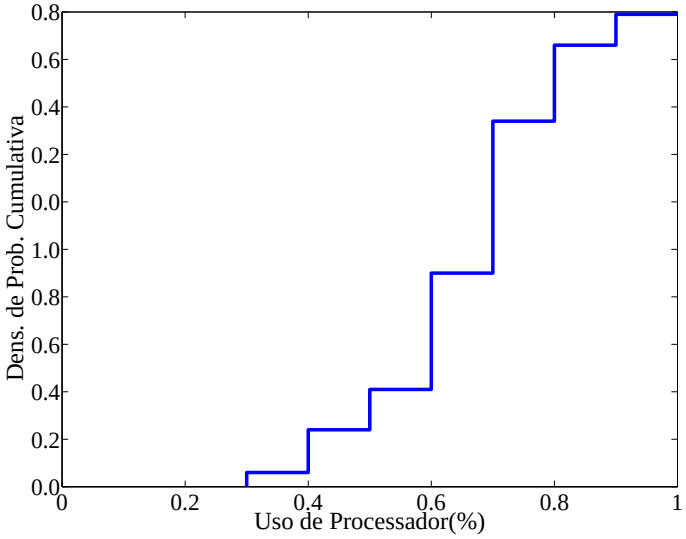


Figura 5.5: Função densidade de probabilidade cumulativa (CDF) para uma máquina virtual executando o protocolo RIPv2.

As funções de distribuição de probabilidade permitem a estimação de qual seria o consumo futuro de recursos de cada máquina. As funções de probabilidade cumulativa permitem estimar a probabilidade de um dado elemento ser atendido em uma dada máquina física. Esta forma de análise de elementos virtuais é baseada em [10]. Além do perfil de consumo, as máquinas físicas também possuem um perfil oferecido (PO), que representa o perfil de disponibilidade de recursos. Este perfil é utilizado durante a escolha da máquina física adequada para receber uma dada máquina virtual que precisa ser migrada.

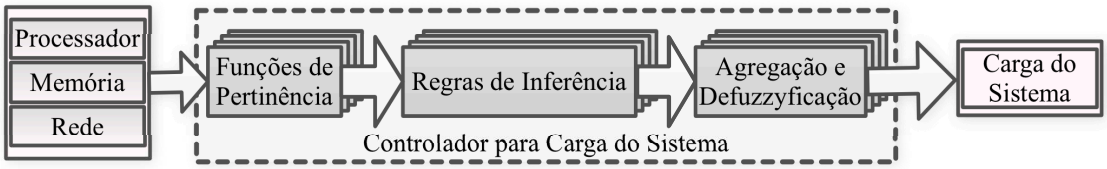


Figura 5.6: Funcionamento do controlador nebuloso para geração da métrica carga do sistema.

Além dos perfis e séries, o Analisador de Perfis gera uma métrica de carga do sistema. Esta métrica é gerada de acordo com [51][4] e reflete uma combinação nebulosa entre múltiplas variáveis do sistema, como processamento, utilização de memória e utilização de rede.

A carga do sistema é uma métrica que determina o nível de carga do *hardware* gerenciado. Graças a ela é possível detectar o esgotamento dos recursos disponíveis nas máquinas físicas. O cálculo da carga do sistema envolve múltiplos parâmetros como o consumo de recursos de processamento, memória e das interfaces de rede. No escopo dos resultados dessa dissertação, foram utilizados estes três parâmetros. Outros exemplos de parâmetros poderiam ser a existência de mecanismos de redundância como fontes extras de energia, a temperatura do sistema, etc. O monitoramento é feito através de medidas coletadas através da `libvirt`. Para gerar a métrica carga do sistema, foi definido o conjunto de funções de pertinência μ_{Proc}, μ_{Mem}, μ_{Net}, correspondentes ao processador, memória, e utilização das interfaces de rede respectivamente, que associa cada um dos recursos ou parâmetros em variáveis *fuzzyficadas*. A combinação destes parâmetros gera uma saída definida como carga do sistema, que possui valores no intervalo $[0, 1]$ e representa a utilização dos recursos físicos do sistema. Este valor pode ser utilizado para estimar o esgotamento de recursos das máquinas físicas. O funcionamento deste sistema baseado em lógica nebulosa pode ser visto na Fig. 5.6. Neste diagrama, pode-se observar o funcionamento do controlador nebuloso. Inicialmente, são recebidos os parâmetros de

[4]A definição e os conceitos relativos a carga do sistema e lógica nebulosa são adaptados do artigo [51] e do projeto final do autor desta dissertação [61]. Um maior detalhamento sobre a lógica nebulosa pode ser visto no Capítulo 4.

entrada do sistema, que são a utilização de processamento, memória e interfaces de rede. Em seguida, são aplicadas funções de pertinência que fuzzyficam as entradas. A fuzzyficação consiste na aplicação das funções de pertinência sobre os parâmetros de entrada. Em seguida, estes valores são avaliados de acordo com regras de inferência definidas pelo administrador da nuvem, em linguagem natural, que mapeiam qualitativamente a tomada de decisão do administrador. Em seguida, o resultado desta operação é defuzzyficado. A defuzzyficação transforma a saída nebulosa do controlador em valores do mundo real, que no caso desta dissertação é o valor carga do sistema.

Regras de Inferência
Se **Processador** (baixo) e **Memória** (baixa) Então **Carga do Sistema** (baixa)
Se **Memória** (médio) e **Processador** (baixa) Então **Carga do Sistema** (baixa)
Se **Processador** (alto) e **Memória** (baixa) Então **Carga do Sistema** (média)
Se **Processador** (médio) e **Memória** (alta) Então **Carga do Sistema** (alta)
Se **Processador** (alto) e **Memória** (alta) Então **Carga do Sistema** (alta)

Tabela 5.1: Exemplo de um conjunto de regras de inferência para processador e memória.

Para determinar como a carga do sistema varia de acordo com os parâmetros avaliados, deve-se elaborar regras de inferência que reflitam o conhecimento do gerente da nuvem. Um exemplo de conjunto de regras de inferência pode ser visto na Tabela 5.1. No conjunto de regras nomeado Pacote de Estratégia, pode-se observar que a carga do sistema é mapeada através de conceitos qualitativos sobre utilização de processamento e de memória. O controlador nebuloso utiliza estas regras para gerar uma superfície de decisão que indica o valor da carga do sistema.

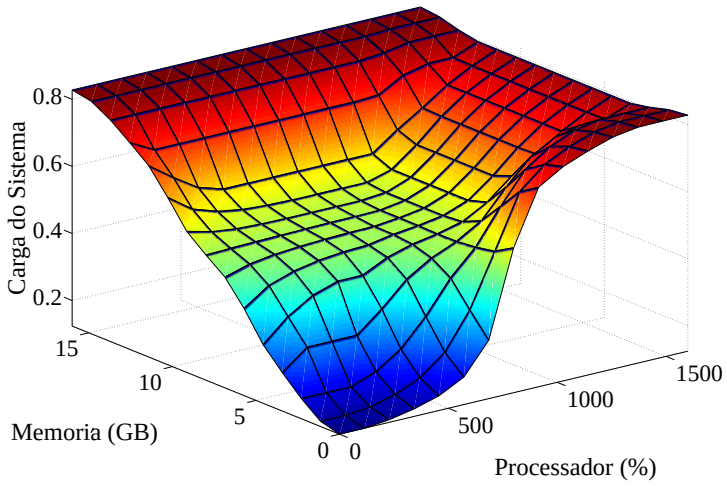


Figura 5.7: Carga do sistema em relação a utilização de processador e memória.

Ao observar a Fig. 5.7, verifica-se como uma configuração de carga do sistema varia em função do processamento e da memória disponível na máquina física. Nesta configuração específica, percebe-se que conforme a escassez de recursos de processador e memória aumenta a carga do sistema aumenta, indicando uma possível situação de sobrecarga. Assim, os gerentes do sistema podem modelar que parâmetros são mais importantes na carga do sistema e como a variação dos parâmetros influencia na tomada de decisão. Através desta métrica e da sua variação no tempo, o sistema detecta máquinas físicas que estão próximas da saturação. Ao detectar este risco, o VOLTAIC atua pró-ativamente para eliminar o problema através da migração autonômica de máquinas virtuais.

5.1.3 Orquestrador

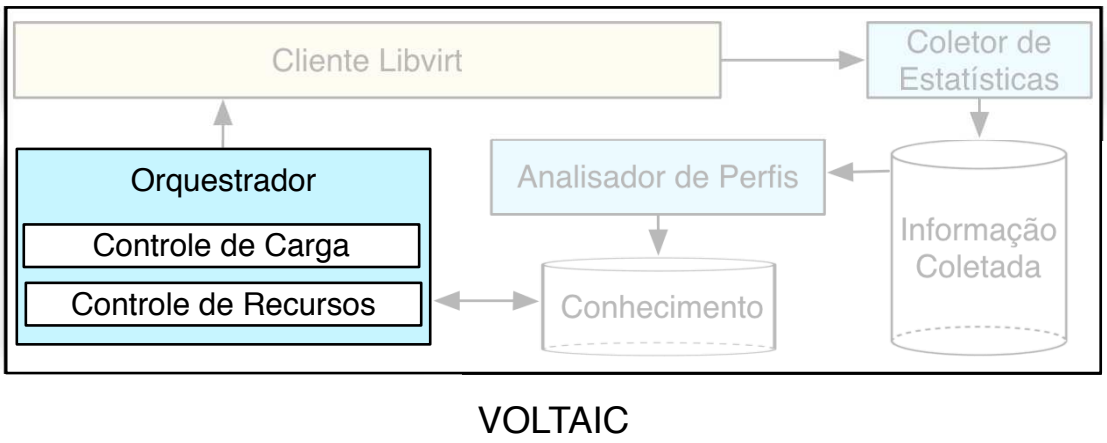


Figura 5.8: O Orquestrador do VOLTAIC, responsável pela gerência das máquinas e pela tomada de decisões.

O Orquestrador (Fig. 5.8) é o módulo principal do VOLTAIC. Ele é responsável pela gerência das máquinas físicas e, consequentemente, pela tomada de decisões. Essas decisões são embasadas na execução de algoritmos de controle de carga e de controle de recursos. O submódulo `controle de carga` analisa a variação da carga dos sistemas físicos e detecta gargalos na oferta de recursos, que podem gerar perdas. A variação da carga é analisada pela sua variação no tempo. Se a média da carga das últimas medidas extrapolar um limiar de segurança, o `controle de carga` inicia os procedimentos de realocação de recursos. O submódulo `controle de recursos` analisa a disponibilidade de recursos em cada máquina física e os perfis de consumo de cada máquina virtual. Este controle de recursos permite que o sistema estime demandas futuras de consumo de recursos e decida qual é a melhor alocação para cada elemento virtual.

Algoritmos de Realocação de Recursos

O submódulo `controle de carga` do Orquestrador monitora a carga das máquinas físicas e detecta se alguma das máquinas físicas sustentou uma carga média maior do que um dado limite de segurança por um período de tempo pré-definido. Caso isso aconteça, os algoritmos de migração são ativados. Na implementação e na simulação, esse período de tempo corresponde ao tempo necessário para efetuar as últimas cinco medidas da carga do sistema. O limite de segurança pode assumir um valor no intervalo $[0, 1]$, que é igual ao intervalo de valores que a carga do sistema pode apresentar. O limite de segurança da carga foi especificado como $0,85$ e a explicação para a utilização deste valor pode ser encontrada na Seção 6.2. O algoritmo, que determina quais máquinas são consideradas em uma situação crítica, ordena as máquinas físicas em função da carga do sistema e da quantidade de máquinas virtuais críticas que cada uma das máquinas físicas possui.

Máquinas virtuais críticas são aquelas presentes em máquinas físicas saturadas e que são responsáveis por uma parcela significativa da carga do sistema da máquina física onde elas residem e cujas variações de perfil no tempo apresentam baixa correlação. O perfil da máquina é calculado a partir da geração de distribuições de probabilidade de uma janela deslizante que representa as últimas informações sobre a utilização de recursos da máquina. Ao analisar essa janela deslizante em diferentes períodos de tempo, pode-se verificar como o perfil de uma máquina varia no tempo.

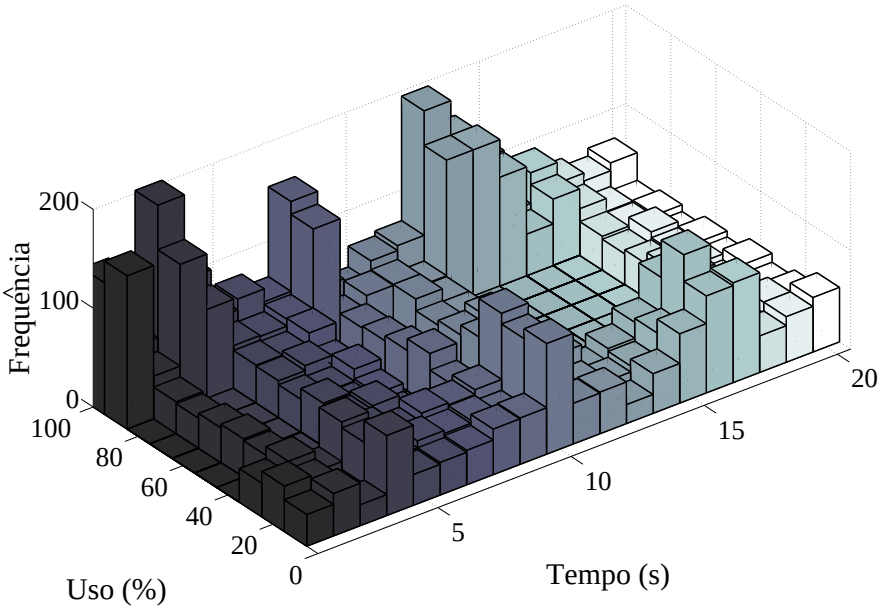


Figura 5.9: Evolução dos perfis no tempo para uma carga de trabalho com baixa correlação.

A variação do perfil no tempo é um reflexo da probabilidade da máquina ter um

comportamento previsível. Se uma máquina apresenta uma alta correlação entre perfis consecutivos, isto é um indicativo que nas iterações futuras esta máquina terá uma probabilidade maior de se comportar de forma previsível. Caso contrário, a máquina apresenta comportamento instável e o algoritmo deseja alocá-la em outro ambiente físico para evitar que ela prejudique o funcionamento das máquinas mais comportadas que compartilham os mesmos recursos físicos. O algoritmo pode ser visto no Alg. 1. No algoritmo, a função `criticidade()` avalia a contribuição da máquina virtual na carga total da máquina física que a hospeda e a variabilidade do seu perfil. A criticidade é dada por

$$criticidade = \alpha \cdot [\ carga\ virtual\ ] + (1 - \alpha) \cdot [1 - \ abs(correlação)\ ] \qquad (5.1)$$

relacionando o impacto da máquina virtual na carga da máquina física e a correlação entre o perfil atual da máquina e perfis anteriores da mesma máquina. O perfil da máquina é armazenado como uma janela deslizante de tamanho fixo. Nos testes, o α adotado foi de $0,5$, dando um peso igual para a carga e para a correlação no cálculo da criticidade. A correlação envolve o cálculo da correlação existente entre os perfis adjacentes de uma mesma máquina conforme este perfil varia no tempo.

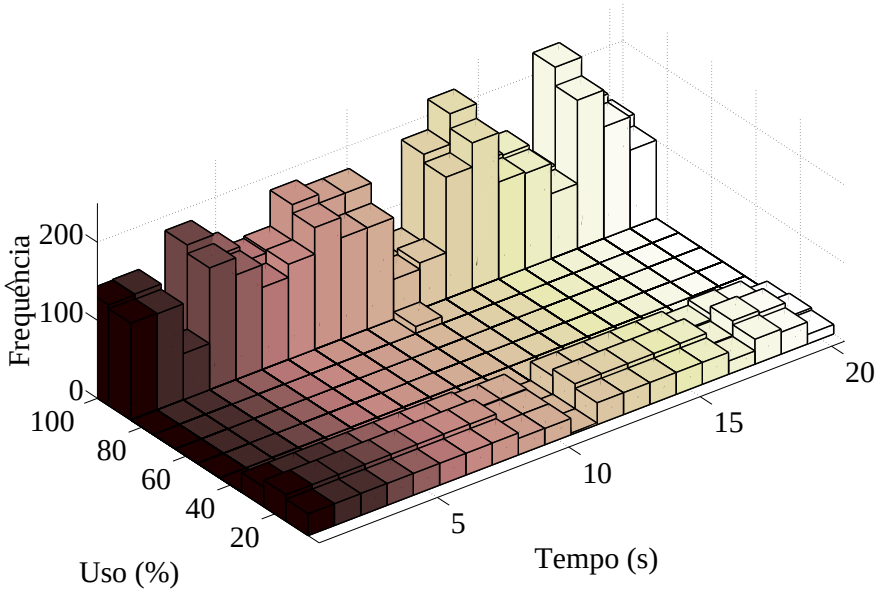


Figura 5.10: Evolução dos perfis no tempo para uma carga de trabalho com alta correlação.

Após calcular a criticidade, a função retorna uma tupla com o valor de criticidade e um valor booleano que indica se este valor ultrapassa o limiar estabelecido. Na

Fig. 5.9, observa-se um exemplo de como um perfil de uma máquina virtual varia no tempo. Neste exemplo, os perfis consecutivos possuem baixa correlação. Pode-se perceber a baixa correlação, pois os perfis de janelas deslizantes consecutivas apresentam padrões de consumo de recursos bem diferentes. Na Fig. 5.10, observa-se uma máquina que possui perfis consecutivos com alta correlação, ou seja, ao observar janelas deslizantes consecutivas o padrão de consumo de recursos não sofre grandes alterações. Desta forma, a tarefa de estimar o padrão de consumo futuro a partir deste perfil é facilitada.

```
   input  : SysChargeList[ ], limiteVirtual
   output: MFsCandidatasMigracao
 1 MFsCandidatasMigracao = [ ];
 2 for MF ∈ SysChargeList do
 3 |   numeroDeVirtuaisCriticas = 0;
 4 |   for MV ∈ MF do
 5 |   |   critica = criticidade(VM, limiteVirtual);
 6 |   |   if critica[0] == True then
 7 |   |   |   numeroDeVirtuaisCriticas+ = 1;
 8 |   |   end
 9 |   end
10 |   Info = (MF, numeroDeVirtuaisCriticas)
   |   MFsCandidatasMigracao.append(Info);
11 end
12 ordenarDeAcordoComMVsCriticas(MFsCandidatasMigracao);
```

Algoritmo 1: Seleção das máquinas físicas críticas.

A função `ordenarDeAcordoComMVsCriticas()` agrupa máquinas físicas com carga próxima e ordena estes grupos em função do número de máquinas virtuais críticas de cada um deles. Desta forma, os primeiros elementos retornados representam as máquinas com maior carga e mais máquinas virtuais críticas.

Após esta etapa, executa-se o algoritmo de seleção de máquinas virtuais candidatas a migração. Este algoritmo, que pode ser visto em Alg. 2, itera sobre as máquinas físicas candidatas e busca por máquinas virtuais críticas. Estas máquinas são ordenadas pelo grau de criticidade. Em seguida, o sistema itera sobre as máquinas virtuais candidatas. Para cada uma delas, observa-se se o perfil de consumo mais recente e analisa-se a correlação entre este perfil e o perfil oferecido (PO) pelas máquinas físicas, ordenadas da menor carga para a maior carga. A correlação utilizada é a correlação de Pearson, obtida através da divisão da covariância pelo produto dos desvios padrão de duas variáveis. Caso o perfil seja correlacionado e o consumo médio da máquina virtual seja menor que o oferecido pela máquina física, a máquina virtual é migrada. Se não for encontrado um perfil compatível, a próxima máquina é avaliada e assim sucessivamente.

```
   input  : MFCandidatasMigracao[ ], limiteFisico, limiteVirtual
   output: VmsCandidatasMigracao[ ]
 1 VmsCandidatasMigracao = [ ];
 2 for MF ∈ MFCandidatasMigracao do
 3 |   if MF.charge ≥ limiteFisico then
 4 |   |   for MV ∈ MF do
 5 |   |   |   distribution = getDist(VM);
 6 |   |   |   if criticidade(MV, limiteVirtual)[0] == True then
 7 |   |   |   |   info = (distribution, VM, PM);
 8 |   |   |   |   VmsCandidatasMigracao.append(info);
 9 |   |   |   end
10 |   |   end
11 |   end
12 end
13 VmsCandidatasMigracao.sort(reverse = True);
```

Algoritmo 2: Seleção das máquinas virtuais candidatas a migração.

5.2 Implementação e Simulação

O VOLTAIC foi implementado na linguagem de programação Python. Ele utiliza a biblioteca *python-libvirt* para se comunicar com as plataformas de virtualização. A implementação foi feita baseada na programação com múltiplos processos leves (*multi-thread*), na qual conjuntos de processos leves (*threads*) monitoram e realizam as operações de gerência. Para cada máquina física, o VOLTAIC cria um processo leve que monitora esta máquina preenchendo uma base de dados de informações sobre as máquinas, que inclui a utilização de memória, processamento e rede de cada máquina física e de cada máquina virtual. Baseado nestas informações, o processo Analisador de Perfis gera os perfis de uso e as cargas do sistema e armazena esta informação em outra base de dados. O Orquestrador captura estas informações e, se necessário, efetua migrações para controlar a alocação das máquinas virtuais.

Para testar uma maior quantidade de parâmetros e realizar testes de migração em maior escala, foi desenvolvido um simulador discreto de eventos para ambientes virtuais. O simulador foi desenvolvido na linguagem de programação Python e permite a configuração de máquinas físicas e virtuais. O simulador[5] fornece as primitivas de criação e destruição de máquinas virtuais, migração de máquinas com um dado custo de processamento e memória e o agendamento de eventos. Neste simulador, cada passo de execução representa uma unidade de tempo de simulação. As principais classes do simulador estão detalhadas abaixo.

O simulador foi desenvolvido de acordo com a modelagem orientada a objetos,

[5]O simulador está disponível para *download* e pode ser encontrado na página http://www.gta.ufrj.br/~hugo/virtsim/.

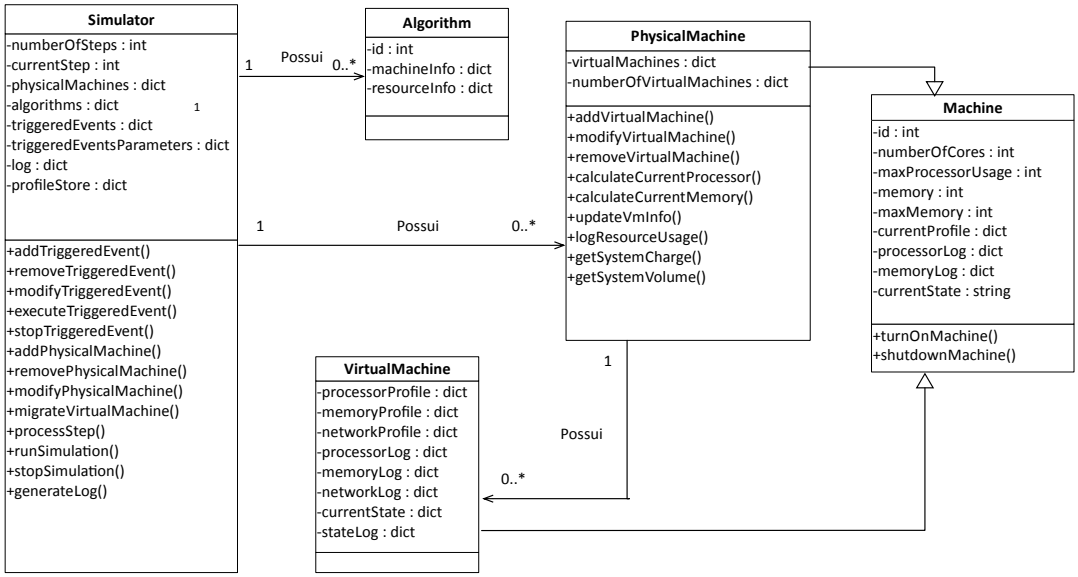


Figura 5.11: Diagrama de classes simplificado do simulador proposto.

e o diagrama de classes simplificado pode ser observado na Fig. 5.11. O diagrama foi modelado de acordo com os conceitos de Engenharia de Software. As setas com ponta preta representam as relações de posse, onde uma determinada entidade pode possuir uma associação com outra entidade. Por exemplo, pode-se analisar a relação entre a entidade Máquina Física (*PhysicalMachine*) e a entidade máquina virtual (*VirtualMachine*). Esta relação indica que para uma máquina virtual existir ela precisa estar associada a uma e somente uma máquina física. O valor 1 na base da seta indica este relacionamento. Por sua vez, uma máquina física pode existir e possuir nenhuma ou um número ilimitado de máquinas virtuais. Isso é representado pelo 0..∗ que indica que na relação a máquina física pode estar associada a um número de zero a infinito de máquinas virtuais. As setas com ponta branca indicam herança. A herança significa que a entidade na base da seta herda os atributos e métodos da entidade da ponta da seta. Por exemplo, as máquinas físicas e virtuais herdam da entidade máquina (*Machine*). Desta forma, as máquinas físicas e virtuais possuem os atributos associados à entidade máquina e os seus métodos, e podem possuir ainda mais atributos e métodos específicos de cada uma dessas entidades.

No diagrama são apresentadas as entidades principais do modelo. O Simulador é a entidade principal do sistema. Ele é responsável por armazenar e processar os passos de simulação. Pode-se configurar o número de passos, o registro das informações dos eventos ocorridos a cada passo, as configurações de cada máquina física, etc. Além disso, pode-se programar o disparo de eventos e executar métodos relacionados à operação do simulador. O Simulador armazena objetos do tipo algoritmo e objetos do tipo máquina física. Os objetos do tipo algoritmo podem conter todos os

algoritmos que são utilizados para se realizar a gerência dos ambientes virtualizados. Os objetos do tipo máquina física representam máquinas físicas reais e armazenam as máquinas virtuais que executam nestas máquinas físicas. Por fim, existe a entidade máquina virtual que representa uma máquina virtual no sistema e o seu perfil de consumo de recursos.

5.2.1 Máquinas Físicas

As entidades do tipo máquina física no simulador possuem configurações semelhantes a uma máquina real. Pode-se definir a quantidade máxima de recursos de processamento (em unidades de processamento por segundo), rede (utilização em megabytes por segundo) e memória (Em megabytes) que podem ser oferecidos a cada passo de simulação e também se podem associar máquinas virtuais a estas máquinas físicas. A implementação também simula uma plataforma de virtualização genérica, que permite a utilização de primitivas como a criação, destruição e migração de máquinas virtuais e a implementação de diferentes escalonadores de recursos. Dependendo da plataforma simulada, o usuário do simulador pode personalizar a simulação, inserindo custo computacional na execução destas primitivas e agendando a execução de tarefas, por exemplo, estimando a quantidade de passos necessários para se concluir uma migração.

O escalonador de processamento implementado faz uma divisão proporcional, caso o somatório de recursos oferecidos for inferior ao requisitado. Desta forma, caso a máquina física seja capaz de fornecer 100 unidades de processamento em um dado passo e duas máquinas desejem usar 100 unidades de processamento neste passo, cada máquina recebe apenas 50 unidades de processamento e o simulador registra que máquinas virtuais perderam recursos nesta iteração e de quanto foi a perda.

5.2.2 Máquinas Virtuais

As entidades do tipo máquina virtual herdam características das entidades máquina física, com a diferença que não existe a implementação da plataforma de virtualização. Podem-se definir limites de processamento, memória e rede. Além disso, a máquina virtual permite a inserção de perfis de consumo de cada um dos recursos, de forma a possibilitar a injeção de um padrão de consumo de recursos real no simulador e observar o comportamento do sistema. Além do carregamento de perfis reais, pode-se optar por definir uma função de distribuição que gere o perfil de uso da máquina. Desta forma, dado que o desenvolvedor possui conhecimento do perfil de tipos de máquina virtual específicos, pode-se simular a forma como ela acessa os recursos mesmo sem a existência de perfis reais.

5.2.3 O VOLTAIC e o Gerente de Simulação

Na simulação o VOLTAIC possui uma entidade própria, capaz de interagir com as máquinas físicas e virtuais. Ela permite a execução das tarefas de monitoramento de máquinas e a execução dos algoritmos de migração e seleção de candidatos propostos. Essa interação é feita por meio do gerente de simulação.

O gerente de simulação coordena todas as atividades de simulação. Pode-se definir o número de rodadas de simulação e quantos passos cada rodada possui. O gerente de simulação permite a criação de conjuntos de máquinas físicas e a criação e associação das máquinas virtuais. O gerente de simulação também auxilia o sistema a realizar a migração de elementos virtuais, através da invocação de métodos de envio e recepção de máquinas virtuais dentro de cada uma das máquinas físicas envolvidas. No final da execução, pode-se gerar um registro de todas as operações que foram feitas.

5.3 Considerações Importantes

Este capítulo apresentou a proposta VOLTAIC para gerenciamento autonômico de máquinas virtuais em ambientes de nuvens de computadores baseados em Infraestrutura como Serviço (IaaS). O VOLTAIC utiliza algoritmos e heurísticas e mecanismos de monitoramento para estimar a carga de cada máquina física na nuvem. Após a análise das cargas das máquinas, o sistema realiza um balanceamento das máquinas virtuais de forma a tentar distribuir de forma eficiente as máquinas virtuais. Esta distribuição tenta balancear a utilização de recursos entre todas as máquinas físicas. O próximo capítulo apresenta os principais resultados obtidos na proposta. São apresentados testes que definem a melhor escolha de parâmetros para a proposta e são apresentados resultados de implementação e simulação.

Capítulo 6

Resultados

Os resultados tanto de implementação quanto de simulação foram obtidos em dois servidores idênticos (máquinas físicas MF1 e MF2), ligados entre si por uma fibra ótica conectada a placas Intel 82599EB 10 Gbps. Os servidores possuem dois processadores Intel Xeon X5570, fornecendo um total de 16 núcleos físicos e 24 GB de memória RAM DDR3 1066 MHz. A plataforma de virtualização utilizada nos resultados é o KVM e a versão da `libvirt` utilizada é a 0.9.6.

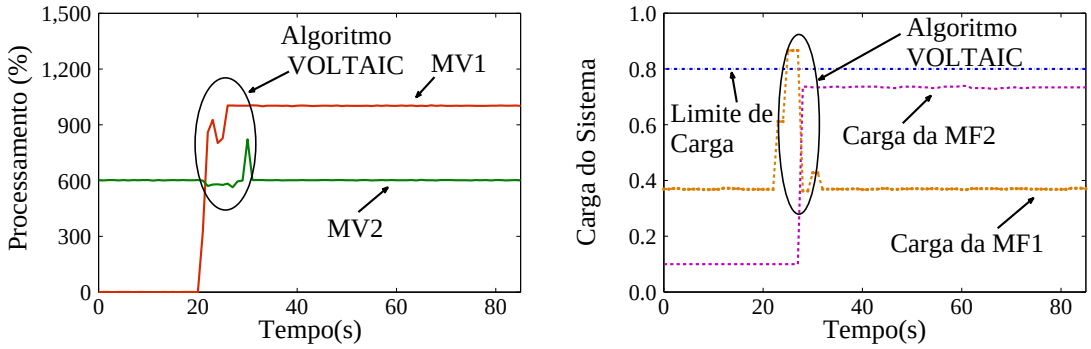
Os resultados se dividem em três etapas principais. A primeira etapa tem o objetivo de apresentar alguns resultados da implementação do VOLTAIC e compará-los com a simulação sob as mesmas condições de forma a validar o simulador. Demonstra-se que o simulador proposto apresenta um comportamento equivalente ao sistema implementado. A segunda etapa tem o objetivo de definir a melhor configuração de parâmetros para o sistema VOLTAIC. São avaliadas a capacidade no oferecimento de recursos de uma dada máquina física e a escolha do tamanho de janela deslizante que melhor atende a diferentes perfis de consumo de recursos. Além disso, são avaliados modelos de consumo de recursos que representam o consumo em *data centers*. Por fim, a terceira etapa avalia a proposta e a sua eficiência na alocação dinâmica de recursos.

6.1 Simulação e Implementação

A metodologia de testes seguiu a evolução do sistema VOLTAIC, com intuito de demonstrar a validade de cada uma das etapas do desenvolvimento da proposta. Inicialmente, são feitos testes que demonstram que a implementação da proposta funciona adequadamente em um cenário restrito e real. Foram capturadas informações de perfil de uso reais neste cenário. O simulador desenvolvido foi calibrado com as informações do cenário real, como o tempo estimado de migração e o perfil de consumo de processamento e memória no tempo. Em seguida, testou-se o comportamento do simulador quando este foi submetido à carga real. Observou-se que o

simulador apresentou um comportamento igual ao comportamento real do sistema físico. Em seguida, foram simulados ambientes com diversas máquinas físicas e virtuais, a fim de avaliar a melhora que o sistema proposto consegue na alocação de recursos.

No primeiro teste da implementação, foram criadas duas máquinas virtuais. Cada máquina virtual pode receber tarefas de processamento e utilizar os 16 núcleos físicos disponíveis. Caso a máquina física não seja capaz de fornecer o processamento necessário, as máquinas virtuais sofrem redução da capacidade de utilização de processamento até atingir um valor que corresponda à capacidade de fornecimento de recursos disponíveis na máquina física.



(a) Variação do processamento.

(b) Variação da carga do sistema.

Figura 6.1: Migração autonômica: a) detecção de um aumento de carga sustentado no sistema e b) migração da máquina virtual para um servidor menos carregado.

Na Fig. 6.1(a), observa-se a utilização do processador de duas máquinas virtuais, MV1 e MV2, em relação ao tempo. A escala do processamento representa a utilização total de recursos de processamento, que pode variar de zero até 1.600%, que representa a utilização total dos 16 núcleos físicos. Inicialmente as duas máquinas virtuais, MV1 e MV2, estão alocadas na máquina física MF1. Observa-se que no instante *t=20* segundos, a máquina virtual MV1 recebe uma tarefa de processamento intensa, que eleva a sua utilização de processamento de 0% para 1.000%. Nesta situação, a máquina física está totalmente saturada e qualquer aumento no uso de processamento gera perdas de desempenho. Ao detectar o aumento da demanda por recursos, o sistema VOLTAIC opta por automaticamente migrar a máquina virtual MV1 que apresenta maiores chances de saturar a capacidade da máquina física MF1 para a máquina física MF2, para distribuir de forma mais justa a carga nas máquinas físicas, como pode ser visto na Fig. 6.1(b). Desta forma, o VOLTAIC realoca máquinas virtuais de acordo com as máquinas físicas que podem atender melhor a demanda das máquinas virtuais.

Para aumentar a escala de testes foi desenvolvido um simulador de ambientes virtuais. Para validar este simulador, o mesmo perfil de consumo de processamento

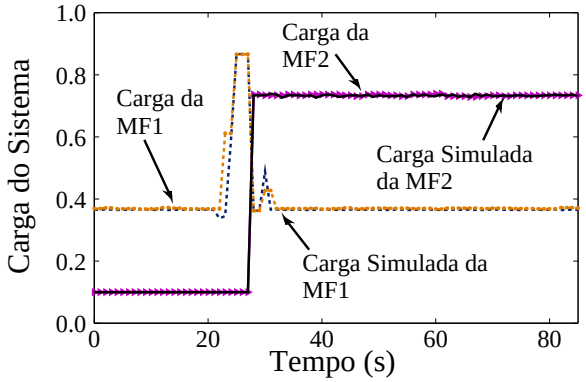


Figura 6.2: A Simulação do ambiente virtualizado com o VOLTAIC demonstra a equivalência entre as saídas geradas pela simulação e as saídas reais.

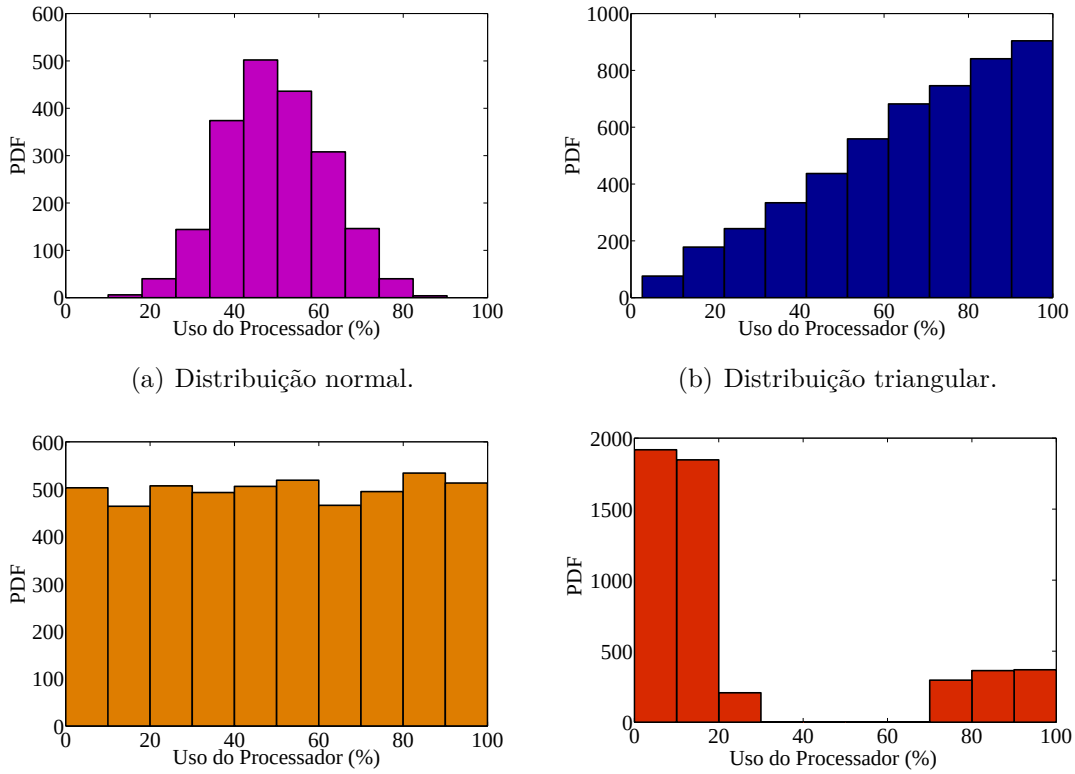
visto na Fig. 6.1(a) e a configuração das máquinas físicas da implementação foram reproduzidos no simulador e foi observada a variação de carga do sistema. Os resultados demonstram que para a mesma carga real, o simulador apresenta o mesmo comportamento na variação do processamento e na variação da carga do sistema, como pode ser visto na Fig. 6.2. Os passos de simulação foram associados aos segundos de execução do sistema real.

6.2 Análise de Perfis, Métricas e Parâmetros

Um desafio encontrado no desenvolvimento da proposta foi a especificação dos valores dos parâmetros do VOLTAIC, como a sensibilidade da carga do sistema e o tamanho adequado da janela deslizante. Para determinar um tamanho adequado de janela deslizante, optou-se por gerar padrões de consumo com distribuições de utilização de recursos distintas e avaliar para estas distribuições qual é a representatividade do tamanho da janela deslizante em relação ao perfil completo da máquina. Ou seja, dada uma janela deslizante, qual é a semelhança entre o perfil extraído para essa janela e o perfil que representa de fato todo o consumo da máquina virtual.

Para realizar esta avaliação, foram selecionadas quatro distribuições de consumo, que podem ser vistas na Fig. 6.3. As distribuições são a distribuição normal, triangular, uniforme e a *race to halt*. A abcissa representa o uso de processador e a ordenada indica a frequência com que o processador foi utilizado com uma dada intensidade. A distribuição *race to halt* representa um perfil de consumo existente em *data centers*[35].

A *race to halt* modela um sistema onde máquinas ficam ociosas a espera de tarefas, e ao receber estas tarefas, as máquinas utilizam o máximo possível de recursos para terminar as tarefas o mais rápido possível. A distribuição pode ser modelada como uma máquina de estados, que é caracterizada por dois estados possíveis, o



(a) Distribuição normal.

(b) Distribuição triangular.

(c) Distribuição uniforme.

(d) Distribuição *race to halt*.

Figura 6.3: Distribuição de probabilidades para diferentes distribuições de uso de processamento.

estado ativo e o estado inativo. Uma máquina no estado ativo consome uma carga de processamento aleatória entre 80% e 100% e uma máquina no estado inativo consome uma carga aleatória entre 0% e 30%. O estado da máquina muda para ativo ou inativo com uma probabilidade de 20% e 80%, respectivamente. Um estado novo tem uma duração aleatória entre 60 e 300 segundos.

A avaliação consiste em gerar dez mil amostras de consumo de processamento para cada uma das distribuições escolhidas. Em seguida, são geradas as janelas deslizantes e avalia-se para as janelas deslizantes que percorrem as amostras qual é a correlação entre elas e a distribuição total, gerada calculando-se a distribuição de probabilidade de todas as dez mil amostras. Esta semelhança entre a distribuição da janela e a total foi definida como representatividade da janela. O objetivo é encontrar se existe um tamanho de janela deslizante mínimo capaz de representar com eficiência o comportamento total da máquina virtual.

Observa-se na Fig. 6.4 o valor da representatividade em função do tamanho da janela deslizante. As medidas possuem um intervalo de confiança de 95%. A representatividade é definida como a correlação entre a janela que está sendo avaliada e o comportamento completo da máquina, de forma que avalia-se o quão correlacionada a janela avaliada é com o comportamento da máquina. Ela assume um valor no

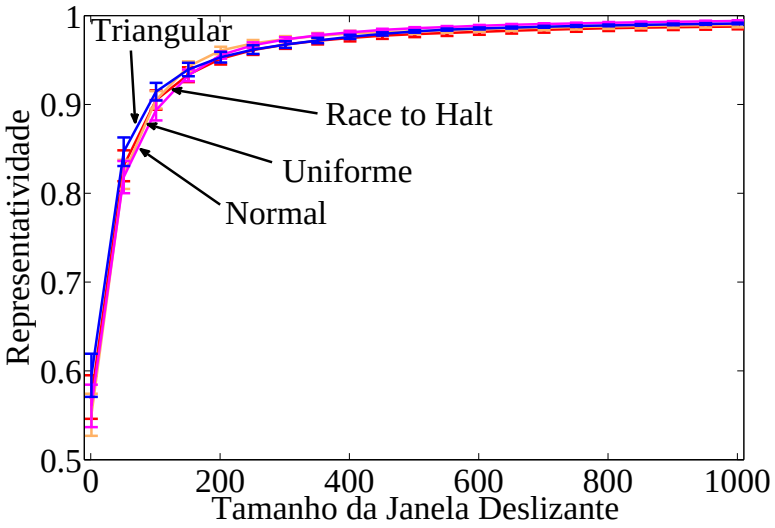


Figura 6.4: Representatividade do perfil da máquina em função do tamanho da janela deslizante.

intervalo $[-1, 1]$, onde o valor -1 indica que a janela representa perfeitamente o inverso da distribuição total. O valor 1 representa a representatividade perfeita, ou seja, a janela representa perfeitamente o perfil total da máquina. Por fim, o valor zero representa a ausência de representatividade, ou seja, a janela não apresenta nenhuma semelhança com o perfil total. Percebe-se que, independentemente da distribuição, o comportamento é assintótico até que o tamanho da janela deslizante seja exatamente todo o tamanho da amostra. Percebe-se também que independentemente da distribuição o comportamento e os valores de representatividade são muito próximos.

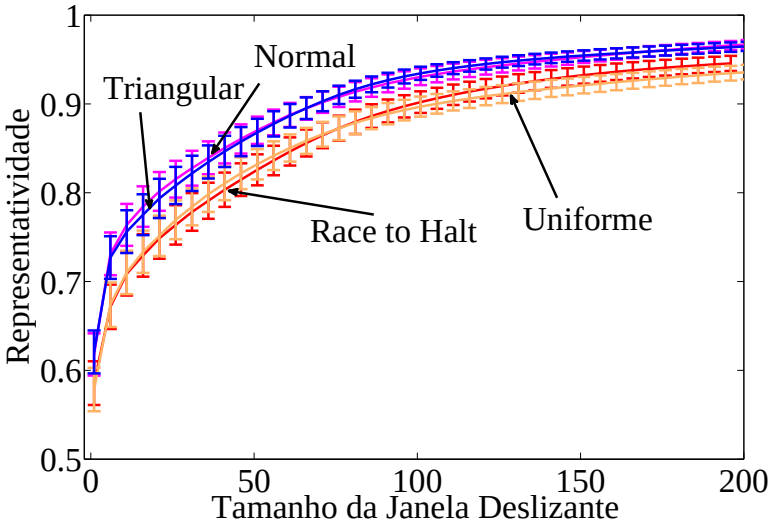


Figura 6.5: Representatividade do perfil da máquina em função do tamanho da janela deslizante. Analise detalhada para janelas de tamanho um a 200.

Na Fig. 6.5, observa-se com uma maior granularidade os tamanhos de janela deslizante entre 1 e 200. A distribuição normal e triangular conseguem obter alta representatividade mesmo com tamanhos menores de janela deslizante, enquanto a distribuição *race to halt* e a uniforme precisam de um tamanho um pouco maior de janela deslizante para obter a mesma representatividade.

A partir dessa análise, pode-se concluir que para todas as distribuições avaliadas, uma janela deslizante de tamanho 100 é suficiente para garantir uma representatividade de 0.9. Se o tamanho de janela deslizante igual a 50 for adotado, é garantido que a janela terá uma representatividade de pelo menos 0.8.

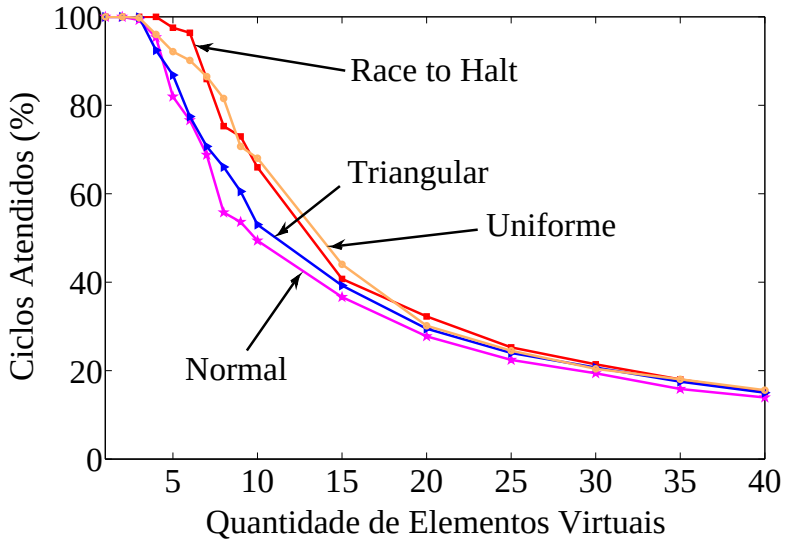
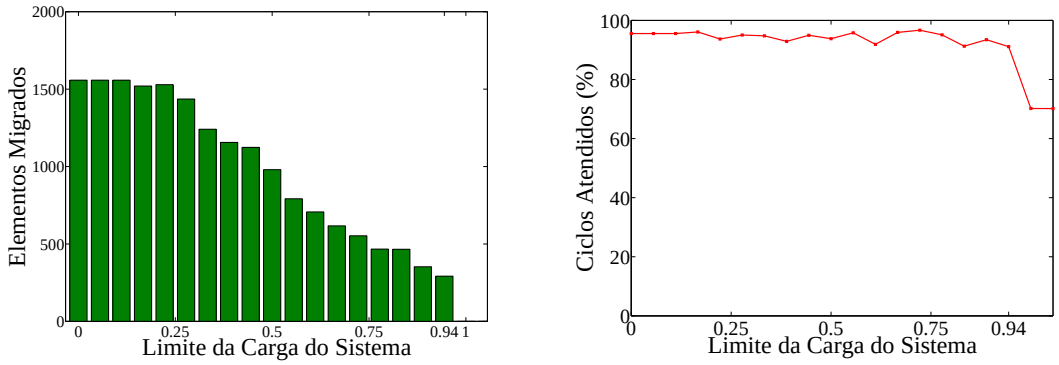


Figura 6.6: Percentual de ciclos atendidos em relação ao número de máquinas virtuais.

Uma segunda análise que pode ser realizada é a verificação de qual é o limite de máquinas virtuais que podem ser suportadas por uma dada máquina física. Para realizar este teste, foram criadas máquinas virtuais com os perfis analisados e elas foram instanciadas em uma única máquina física, que possui 20 GB de memória RAM e dois processadores. Para cada tipo de padrão de consumo, criou-se um número crescente de máquinas virtuais e avaliou-se a quantidade de ciclos de máquinas que foram atendidos. Conforme pode-se observar na Fig. 6.6, a instanciação de máquinas virtuais deprecia rapidamente a qualidade do serviço oferecido. Ao instanciar dez máquinas virtuais, observa-se que o percentual de ciclos atendidos diminui para 60% e que ao instanciar 30 máquinas virtuais este percentual já é reduzido para 20%. E esta queda acontece independentemente do tipo de distribuição avaliado.

Outro parâmetro importante avaliado no sistema é a sensibilidade do parâmetro carga do sistema para o disparo dos algoritmos de migração. A carga do sistema é uma métrica que representa o grau de saturação das máquinas, que é dado em função da utilização dos recursos disponíveis em cada máquina física. A sensibilidade do



(a) Número de migrações realizadas em função da sensibilidade da carga do sistema.

(b) Percentual de ciclos atendidos em função da sensibilidade da carga do sistema.

Figura 6.7: Escolha da sensibilidade do parâmetro carga do sistema do VOLTAIC.

parâmetro carga do sistema está associada ao valor do parâmetro, que pertence ao intervalo $[0, 1]$. No sistema VOLTAIC, o valor limite do parâmetro é chamado de limite de segurança e define a sensibilidade no disparo dos algoritmos de migração autonômica. Ou seja, o limite de segurança indica o valor máximo de carga que uma dada máquina física pode possuir antes que o sistema inicie os procedimentos de migração automática. Para analisar a importância do valor do parâmetro, foi simulada a instanciação de 30 máquinas virtuais que seguem o padrão de consumo de recursos *race to halt* em dez máquinas físicas que possuem 20 GB de memória RAM e dois processadores. Foram executados mil passos de simulação e o número de migrações e a quantidade de ciclos atendidos foram medidos.

Observa-se na Fig. 6.7(a) o número de migrações realizadas em função do valor limite de segurança do sistema. Pode-se perceber que conforme o limite de segurança aumenta, o número de migrações realizadas diminui. Ao utilizar limites muito altos, o número de migrações é reduzido para zero. Isso significa que o valor escolhido de limite é tão elevado que as máquinas físicas não atingiram este valor e os algoritmos de migração autonômica não foram ativados. Na Fig. 6.7(b) observa-se o percentual de ciclos atendidos para cada parâmetro de limite da carga do sistema.

Percebe-se que mesmo para valores de limite diferentes, o algoritmo proposto continua a garantir um mesmo percentual de ciclos atendidos. Somente nos valores extremos de limite que o algoritmo não é ativado. Isto implica em zero migrações, o que inativa o algoritmo e piora os resultados. É interessante analisar que quanto menor o limite da carga do sistema, mais rapidamente o sistema detecta a necessidade de realizar migrações para evitar a sobrecarga. Apesar de limites menores induzirem o sistema a migrar um maior número de máquinas, como o algoritmo sempre move máquinas virtuais para as máquinas físicas mais adequadas, independentemente do limite da carga, o sistema mantém o seu desempenho. Porém, para otimizar o nú-

mero de migrações necessárias, foi escolhido o limite de carga de 0.85 que minimiza o número de migrações realizadas sem prejudicar a ativação dos mecanismos de migração.

6.3 Análise da Proposta

Para demonstrar o funcionamento do VOLTAIC, o simulador foi carregado com máquinas virtuais com perfis de uso de processamento que seguem a modelagem *race to halt*, existente em *data centers*. Cada máquina virtual foi configurada com 256 MB de memória RAM e esta quantidade permanece fixa durante a execução dos testes. Nas simulações não foi considerada a utilização das interfaces de rede.

Uma análise que pode ser realizada é a análise da eficiência dos mecanismos de migração propostos. No início de cada rodada de simulação, a alocação inicial das máquinas virtuais é modificada e é verificada a quantidade de ciclos de processamento que foi perdida no sistema. Os ciclos perdidos são referentes a passos da simulação nos quais a demanda por recursos de uma dada máquina física ultrapassa a capacidade máxima de fornecimento desta mesma máquina física. Ou seja, neste passo, existe pelo menos um conjunto de máquinas virtuais alocadas em uma mesma máquina física e o somatório da quantidade de recursos requisitados ultrapassa a capacidade máxima da máquina física. A Eq. 6.1 demonstra como é feito o cálculo do número de ciclos requeridos na simulação, representado por $C_{totalreq}$.

$$C_{totalreq} = \sum_{passo=1}^{n} \sum_{mv=1}^{m} proc_{mv}^{passo} \tag{6.1}$$

Para avaliar a quantidade de ciclos perdidos, define-se $R^{i}_{dem,mf=j}$ como a quantidade de recursos demandados por todas as máquinas virtuais que executam em uma determinada máquina física j durante o passo i da simulação. Define-se também $R_{ofer,mf=j}$ a quantidade máxima de recursos que pode ser oferecida pela máquina física j em uma dado passo de simulação. Assim, para calcular a quantidade de ciclos perdidos, ou não atendidos, é utilizada a Eq. 6.2. O valor $C_{perdidos}$ reflete o percentual de ciclos perdidos em relação ao total de ciclos que foram demandados. O termo $(R^{p}_{dem,mf} - R^{p}_{ofer,mf})$ representa a diferença entre a quantidade de recursos demandados e os recursos oferecidos. Na Eq. 6.2, se esta diferença for positiva, ciclos foram perdidos e esta quantidade perdida é somada à quantidade de ciclos perdidos. Se a diferença for negativa, ou seja, a máquina física consegue atender a demanda daquele passo, então esta diferença não é adicionada a quantidade de ciclos perdidos. Por fim, divide-se o total de ciclos perdidos pelo total de ciclos demandados para se obter o percentual de ciclos perdidos na simulação. Assim, o valor de $C_{perdidos}$ é

expresso por

$$C_{perdidos} = \frac{100}{C_{totalreq}} * \sum_{p=1}^{n} \sum_{mf=1}^{m} \frac{(R^p_{dem,mf} - R^p_{ofer,mf}) + \left|R^p_{dem,mf} - R^p_{ofer,mf}\right|}{2} \quad (6.2)$$

Após a simulação faz-se uma análise estatística dos dados e compara-se a quantidade de ciclos atendidos em três ambientes diferentes, um com o VOLTAIC, um com o SandPiper e outro sem mecanismos autonômicos de migração.

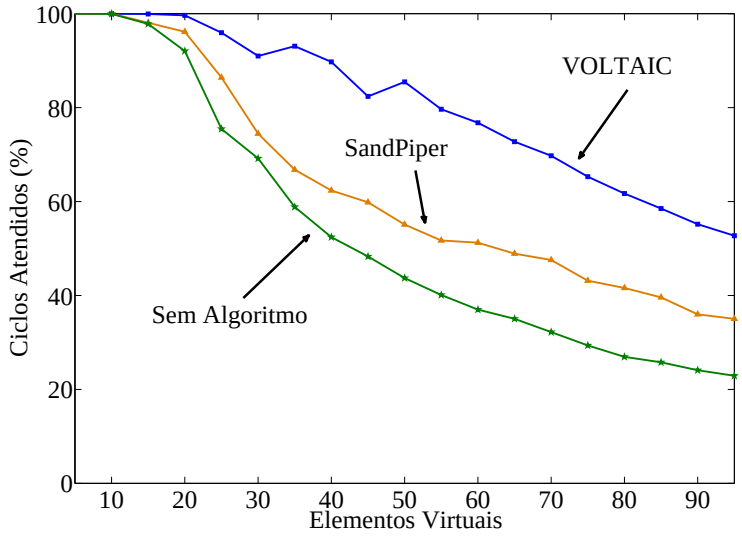


Figura 6.8: Percentual de ciclos atendidos em função do número de máquinas virtuais race to halt. O VOLTAIC reduz em mais de 30% a perda de ciclos em relação ao SandPiper.

Os resultados observados na Fig. 6.8 apresentam o percentual de ciclos atendidos em função do número de máquinas virtuais alocadas nas máquinas físicas. Esses resultados foram obtidos na execução de cinco rodadas de 1000 passos de simulação e foram simuladas dez máquinas físicas. Observa-se que o sistema VOLTAIC consegue reduzir a quantidade de ciclos perdidos em aproximadamente 40% quando comparado a ausência de algoritmos de migração autonômica e em aproximadamente 30% quando comparado ao SandPiper. Percebe-se que conforme o número de máquinas virtuais aumenta, torna-se mais difícil a tarefa de se distribuir as máquinas virtuais nas máquinas físicas de forma eficiente. Para até 20 máquinas virtuais, o VOLTAIC consegue garantir praticamente 100% dos ciclos requisitados.

O SandPiper utiliza a métrica chamada de volume que indica a quantidade de recursos consumidos em função do processamento, memória e rede. Esta métrica atribui pesos iguais para cada um destes recursos. Dessa forma, máquinas físicas

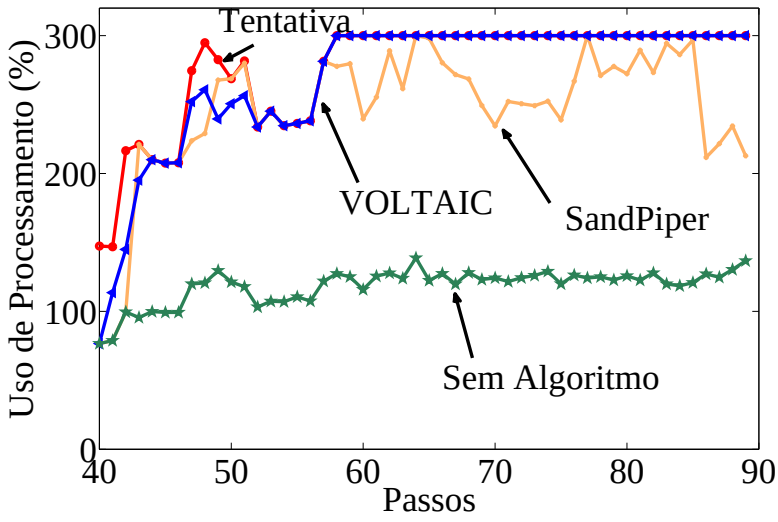


Figura 6.9: Processamento oferecido a uma dada máquina virtual.

que possuem padrões de consumo distintos podem possuir o mesmo volume. Por exemplo, uma máquina com muito recurso de processamento disponível, porém com pouca memória disponível pode possuir o mesmo volume de uma máquina com excesso de memória livre e pouco processamento disponível. Por sua vez, no VOLTAIC a métrica utilizada é a carga do sistema. A carga do sistema pode ser modelada de acordo com os administradores da nuvem e pode atribuir diferentes pesos para as variáveis. Desta forma, se o administrador considera que a quantidade de recursos de processamento é o parâmetro mais importante da carga do sistema, torna-se possível criar uma métrica que leve isso em consideração. Ou seja, apesar da métrica agregar o consumo dos recursos e gerar um valor, este valor é ponderado de acordo com os critérios que o administrador considera mais importante.

Outra diferença existente é quanto ao mecanismo de alocação. O mecanismo de alocação de máquinas do SandPiper move as máquinas virtuais das máquinas de maior volume para as de menor volume. O algoritmo desempenha bem, porém migrar a máquina virtual para uma máquina de menor volume não significa que aquela máquina física, em especial, é a mais adequada à demanda da máquina virtual migrada. No VOLTAIC, ao migrar uma máquina virtual, leva-se em consideração o perfil de consumo de recursos da máquina virtual e da máquina física que vai receber a máquina virtual. Dessa forma, busca-se alocar a máquina virtual em um ambiente que possua um perfil de oferta de recursos mais adequada para o perfil da máquina. Pode-se fazer isso comparando o perfil de consumo de recursos da máquina com o perfil de recursos que podem ser oferecidos pela máquina física. Além disso, o VOLTAIC leva em consideração a correlação entre perfis adjacentes das máquinas virtuais. Dessa forma, as máquinas que são migradas com maior prioridade são

aquelas que possuem um comportamento mais instável, aumentando - as chances da máquina com comportamento elástico possam ter suas demandas atendidas.

Na Fig. 6.9, pode-se verificar a análise do uso de processamento de uma máquina virtual aleatória em uma rodada de execução de testes. Na figura, observa-se a tentativa de consumo da máquina, que representa quanto a máquina utilizaria de recursos se existissem infinitos recursos disponíveis, a quantidade de processador oferecido no VOLTAIC, no SandPiper e na ausência de algoritmos de controle. Conforme o tempo avança, a nossa proposta analisa perfis de consumo da máquina virtual e aprende qual a sua demanda. Assim, ela consegue alocar esta máquina virtual em uma máquina física que melhor a atenda. Após 58 passos da simulação, o VOLTAIC consegue encontrar a máquina física que melhor se adequa a máquina virtual e conseguiu garantir que a demanda de processamento da máquina seja atendida.

Capítulo 7

Conclusão

As nuvens de computadores podem ser definidas como grandes repositórios de recursos virtualizados (*hardware*, plataformas de desenvolvimento e serviços) que são facilmente utilizados e acessíveis. Os recursos disponibilizados na nuvem podem ser reconfigurados dinamicamente para que estes se adequem a cargas variadas de utilização. A adequação a cargas variadas pode ser definida como elasticidade. Um dos maiores desafios do ambiente de nuvens é garantir a elasticidade dos processos que executam na nuvem e evitar a ocorrência de violações de acordos de níveis de serviço (SLAs). Ao mesmo tempo, deseja-se que a utilização de recursos ocorra de forma eficiente. A computação em nuvem introduz um novo modelo de infraestrutura tecnológica, onde clientes contratam ou alugam provedores de recursos que podem oferecer de forma dinâmica processamento, memória, armazenamento e rede. Desta forma, a nuvem tem que ser capaz de atender a demanda dos clientes, evitando que estas demandas não sejam atendidas e ao mesmo tempo evitando o desperdício de recursos computacionais.

No mercado atualmente existem um grande conjunto de soluções proprietárias para nuvem, como a nuvem da EMC, da Amazon, Microsoft e do Google e soluções de código aberto como o Eucalyptus e o OpenNebula. Cada uma das propostas de nuvens oferecidas atende a determinadas especificidades. Pode-se ter o modelo de *Software* como Serviço (SaaS), Plataforma como Serviço (PaaS), Infraestrutura como Serviço (IaaS), Armazenamento como Serviço (DaaS), Comunicação como Serviço (CaaS) e *Hardware* como Serviço (HaaS).

Analisando o modelo de Infraestrutura como Serviço, como o empregado nos projetos de código aberto Eucalyptus e OpenNebula, torna-se evidente a necessidade de se desenvolver mecanismos eficientes de elasticidade. O modelo Infraestrutura como Serviço (IaaS) oferece máquinas virtuais completas a seus clientes e estes clientes podem executar seus sistemas operacionais e executar cargas de trabalho variáveis. Desta forma, caso não existam mecanismos eficientes para gerenciar estas máquinas virtuais, muitos recursos podem ser desperdiçados e a qualidade do serviço

oferecido aos clientes pode ser afetada.

Um mecanismo ideal de gerência deve ser capaz de monitorar as cargas de trabalho das máquinas virtuais e selecionar os ambientes físicos mais adequados para estas máquinas. A seleção da máquina física apropriada reduz a quantidade de recursos desperdiçados e diminui a chance de que uma demanda de consumo de um cliente não possa ser atendida. Para atender a esta demanda de gerência, podem-se empregar primitivas das ferramentas de virtualização. Estas primitivas permitem a alocação dinâmica de recursos de processamento e memória e a realocação de máquinas virtuais através da migração ao vivo de máquinas. Esta migração ao vivo se assemelha a migração de processos e permite que em tempo de execução as máquinas virtuais sejam transferidas para outras máquinas físicas, sem a necessidade de interromper os serviços que nela executam. Além disso, pode-se utilizar a migração para fornecer mobilidade ao migrar as máquinas para mais perto de quem as utiliza.

Para atender esta demanda, este trabalho propõe o sistema VOLTAIC que é uma plataforma de gerência automatizada dos recursos existentes em um ambiente de nuvens de computadores. A plataforma é capaz de monitorar as máquinas físicas e as máquinas virtuais existentes na nuvem e extrair informações sobre o comportamento de cada um destes elementos. Através destas informações, o VOLTAIC é capaz de decidir quando migrar elementos virtuais e para onde migrá-los, de forma a aumentar a chance das demandas individuais de cada máquina virtual serem atendidas. A plataforma de gerência proposta foi pensada com o intuito de gerenciar múltiplos ambientes heterogêneos de nuvens.

Desta forma, o desenvolvimento é compatível com a biblioteca `libvirt` que é uma biblioteca de gerência compatível com a grande maioria das soluções de virtualização existentes (Xen, VMWare, KVM, etc.).

O VOLTAIC se divide em três módulos principais. O primeiro deles é o Coletor de Estatísticas (CE) que interage com a `libvirt` e obtém as estatísticas de monitoramento de cada máquina física monitorada e armazena esta informação em uma base de dados. O segundo módulo é o Analisador de Perfis (AP), que utiliza a informação do Coletor de Estatísticas para extrair conhecimento das plataformas de virtualização. O terceiro módulo é o Orquestrador. O Orquestrador (OQ) utiliza o conhecimento gerado pelo Analisador de Perfis para gerenciar as máquinas físicas e virtuais. Ele pode controlar a quantidade de recursos oferecidos a cada máquina virtual e organizar migrações de elementos virtuais para balancear a distribuição de elementos virtuais nas máquinas físicas.

O módulo Orquestrador executa os algoritmos de realocação de recursos. Os algoritmos analisam o perfil das máquinas virtuais e tentam detectar padrões de comportamento. Baseado na forma com que as máquinas consomem recursos, são escolhidas as máquinas físicas que melhor se adequam a estas máquinas. Além

disso, o monitoramento detecta quando máquinas físicas se encontram em situações de saturação e dispara mecanismos que tentam aliviar a carga das máquinas físicas para evitar a degradação do serviço oferecido aos clientes.

Para regular o sistema, foram desenvolvidos diversos testes com o intuito de detectar qual a melhor escolha de parâmetros para a proposta. Realizaram-se testes com quatro distribuições de padrão de consumo de recursos. Além disso, avaliou-se o valor do parâmetro de segurança e foi escolhido o valor que ao mesmo tempo garante uma alta quantidade de ciclos atendidos e evita migrações desnecessárias.

As heurísticas propostas mostraram eficácia de antever situações de saturação e disparar os algoritmos de migração autônoma. Os algoritmos selecionam as máquinas físicas que mais se aproximam de uma situação de saturação e as máquinas virtuais que são melhores candidatas à realocação. Os resultados demonstram que a aplicação da heurística de disparo da migração e os algoritmos de seleção de máquinas diminuem em mais de 30% a taxa de recusa na oferta de recursos de processamento. Portanto, nos testes realizados, a proposta se mostrou a mais adequada à realidade de *data centers* de computação em nuvens, aumentando de forma significativa a quantidade de máquinas virtuais instanciadas, garantindo que a qualidade de serviço fornecido por cada uma delas seja atendida, evitando o desperdício de recursos e, consequentemente, aumentando os ganhos dos provedores de infraestrutura.

Além da proposta de alocação dinâmica de recursos na nuvem, esta dissertação traz como contribuição a implementação de um simulador de ambientes virtualizados. Este simulador, desenvolvido na linguagem de programação Python, permite a criação de elementos virtuais, reais e de plataformas de virtualização. Pode-se configurar dinamicamente as limitações de consumo de cada um dos elementos existentes assim como o mapeamento de qual elemento virtual está executando em cada elemento real a cada instante de tempo. Assim, torna-se possível recriar de fato um ambiente que apresenta um comportamento semelhante a um conjunto de máquinas físicas que executam plataformas de virtualização que possuem máquinas virtuais com características distintas.

Outra funcionalidade do simulador é o carregamento de padrões de consumo de recursos reais. O carregamento de perfis de processador, memória e rede permite que padrões reais sejam inseridos no simulador. O simulador também permite a execução de diversos testes que não são atrelados necessariamente à migração de máquinas. Por fim, o simulador é extensível e permite o desenvolvimento de novos escalonadores e propostas de virtualização.

Conclui-se que o VOLTAIC é uma proposta eficiente para realocação dinâmica de elementos virtuais em servidores físicos. O estudo de perfis, a confiabilidade dos perfis e a escolha de máquinas físicas que possuem máquinas virtuais críticas

provaram ser capazes de reduzir a quantidade de ciclos de processamento perdidos, reduzindo a perda de ciclos de processamento em situações normais e diminuindo de forma significativa a perda de ciclos em cenários de alta saturação.

Como trabalhos futuros, pode-se realizar uma otimização que leve em consideração a economia de energia. Esta otimização permitiria, por exemplo, que máquinas virtuais fossem consolidadas em um número menor de máquinas físicas para permitir que recursos ociosos sejam desligados com o intuito de economizar energia. Além disso, pode-se realizar um trabalho que meça a economia de energia obtida ao se utilizar as instruções de economia de energia mais avançadas como a tecnologia da Intel presente nos processadores mais novos que permite não somente regular a frequência de operação de cada um dos núcleos do processador como optar por desligar alguns núcleos, ou seja, colocá-los em um estado de consumo muito baixo para economizar ainda mais energia. Além disso, pode-se pensar em algoritmos que levem em consideração, por exemplo, o período do dia para adotar políticas mais ou menos agressivas.

O trabalho desenvolvido durante o período de desenvolvimento desta dissertação deu origem a uma publicação internacional no *3rd International Conference on Information and Communications Systems* (ICICS - 2012) [64] e foi publicado na revista internacional JETWI (Journal of Emerging Technologies in Web Intelligence) [65].

Referências Bibliográficas

[1] FERNANDES, N., MOREIRA, M., MORAES, I., et al. "Virtual networks: Isolation, performance, and trends", *Annals of Telecommunications*, pp. 1–17, 2010. ISSN: 0003-4347.

[2] DUSTDAR, S., GUO, Y., SATZGER, B., et al. "Principles of Elastic Processes", *Internet Computing, IEEE*, v. 15, n. 5, pp. 66–71, 2011.

[3] EMC. "EMC's VPLEX Tecnology". , dez. 2012. Disponível em: <`http://www.emc.com/campaign/global/vplex/`>.

[4] "Amazon Simple Storage Service". , dez. 2012. Disponível em: <`http://aws.amazon.com/s3/`>.

[5] "Microsoft Connected Service Framework". , dez. 2012. Disponível em: <`http://www.microsoft.com/serviceproviders/solutions/connectedservicesframework.mspx`>.

[6] CHEN, Q., DENG, Q. "Cloud Computing and its key techniques", *Jisuanji Yingyong/ Journal of Computer Applications*, v. 29, n. 9, pp. 2562–2567, 2009.

[7] SOTOMAYOR, B., MONTERO, R., LLORENTE, I., et al. "Capacity leasing in cloud systems using the OpenNebula engine", *Cloud Computing and Applications*, v. 2008, pp. 1–5, 2008.

[8] NURMI, D., WOLSKI, R., GRZEGORCZYK, C., et al. "Eucalyptus: A technical report on an elastic utility computing architecture linking your programs to useful systems". In: *UCSB TECHNICAL REPORT*. Citeseer, 2008.

[9] ARMBRUST, M., FOX, A., GRIFFITH, R., et al. "Above the clouds: A berkeley view of cloud computing", *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28*, 2009.

[10] WOOD, T., SHENOY, P., VENKATARAMANI, A., et al. "Sandpiper: Black-box and gray-box resource management for virtual machines", *Computer Networks*, v. 53, n. 17, pp. 2923–2938, 2009.

[11] BOLTE, M., SIEVERS, M., BIRKENHEUER, G., et al. "Non-intrusive virtualization management using libvirt". In: *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 574–579. European Design and Automation Association, 2010.

[12] BOURGUIBA, M., HADDADOU, K., PUJOLLE, G. "Evaluating Xen-based virtual routers performance", *International Journal of Communication Networks and Distributed Systems*, v. 6, n. 3, pp. 268–282, 2011.

[13] VMWARE, E. "Server", *GSX Server, documentação oficial do produto*, 2005.

[14] KIVITY, A., KAMAY, Y., LAOR, D., et al. "KVM: the Linux virtual machine monitor". In: *Proceedings of the Linux Symposium*, v. 1, pp. 225–230, 2007.

[15] JR, A., LAUREANO, M., SANTIN, A., et al. "Aspectos de segurança e privacidade em ambientes de Computação em Nuvem", *Anais do X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, 2010.

[16] HAYES, B. "Cloud computing", *Communications of the ACM*, v. 51, n. 7, pp. 9–11, 2008.

[17] MELL, P., GRANCE, T. "The NIST definition of cloud computing", *National Institute of Standards and Technology*, v. 53, n. 6, 2009.

[18] WANG, Y., HUANG, C., LI, J., et al. "QoSaaS: quality of service as a service". In: *Proceedings of the 11th USENIX conference on Hot topics in management of internet, cloud, and enterprise networks and services*, pp. 6–6. USENIX Association, 2011.

[19] BUYYA, R., YEO, C., VENUGOPAL, S. "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities". In: *High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on*, pp. 5–13. Ieee, 2008.

[20] GEELAN, J. "Twenty one experts define cloud computing. Virtualization", 2008.

[21] VAQUERO, L., RODERO-MERINO, L., CACERES, J., et al. "A break in the clouds: towards a cloud definition", *ACM SIGCOMM Computer Communication Review*, v. 39, n. 1, pp. 50–55, 2008.

[22] YOUSEFF, L., BUTRICO, M., DA SILVA, D. "Toward a unified ontology of cloud computing". In: *Grid Computing Environments Workshop, 2008. GCE'08*, pp. 1–10. IEEE, 2008.

[23] ENOMALY. "Enomaly Cloud Computing Platform". , mar. 2012. Disponível em: <`http://www.enomaly.com/`>.

[24] "EMC Managed Storage Service". , dez. 2012. Disponível em: <`http://emc.com`>.

[25] ZHANG, Q., CHENG, L., BOUTABA, R. "Cloud computing: state-of-the-art and research challenges", *Journal of Internet Services and Applications*, v. 1, n. 1, pp. 7–18, 2010.

[26] GREENBERG, A., HAMILTON, J., MALTZ, D., et al. "The cost of a cloud: research problems in data center networks", *ACM SIGCOMM Computer Communication Review*, v. 39, n. 1, pp. 68–73, 2008.

[27] ADAMS, K., AGESEN, O. "A comparison of software and hardware techniques for x86 virtualization". In: *ACM SIGOPS Operating Systems Review*, v. 40, pp. 2–13. ACM, 2006.

[28] BARHAM, P., DRAGOVIC, B., FRASER, K., et al. "Xen and the art of virtualization". In: *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pp. 164–177. ACM, 2003. ISBN: 1581137575.

[29] PISA, P., FERNANDES, N., CARVALHO, H., et al. "OpenFlow and Xen-Based Virtual Network Migration", *The World Computer Congress 2010 - Network of the Future Conference*, pp. 170–181, 2010.

[30] JIANG, C., XU, X., WAN, J., et al. "Energy Management for Microprocessor Systems: Challenges and Existing Solutions". In: *Intelligent Information Technology Application Workshops, 2008. IITAW'08. International Symposium on*, pp. 1071–1076. IEEE, 2008.

[31] NEDEVSCHI, S., POPA, L., IANNACCONE, G., et al. "Reducing network energy consumption via sleeping and rate-adaptation". In: *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, pp. 323–336. USENIX Association, 2008.

[32] SAKAI, M. “ACPI sleep control”. , jul. 24 2001. US Patent 6,266,776.

[33] MIYOSHI, A., LEFURGY, C., VAN HENSBERGEN, E., et al. “Critical power slope: understanding the runtime effects of frequency scaling”. In: *Proceedings of the 16th international conference on Supercomputing*, pp. 35–44. ACM, 2002.

[34] BOLLA, R., BRUSCHI, R., CARREGA, A., et al. “Theoretical and technological limitations of power scaling in network devices”. In: *Telecommunication Networks and Applications Conference (ATNAC), 2010 Australasian*, pp. 37–42. IEEE, 2010.

[35] HIROFUCHI, T., NAKADA, H., ITOH, S., et al. “Making VM Consolidation More Energy-efficient by Postcopy Live Migration”. In: *CLOUD COMPUTING 2011, The Second International Conference on Cloud Computing, GRIDs, and Virtualization*, pp. 195–204, 2011.

[36] AGARWAL, Y., HODGES, S., CHANDRA, R., et al. “Somniloquy: augmenting network interfaces to reduce PC energy usage”. In: *Proceedings of the 6th USENIX symposium on Networked systems design and implementation*, pp. 365–380. USENIX Association, 2009.

[37] AGUIARI, V. “Servidores da Amazon falham e ficam 36h offline”. , dez. 2012. Disponível em: <`http://info.abril.com.br/noticias/ti/servidores-da-amazon-falham-e-ficam-36h-offline-25042011-6.shl`>.

[38] CRN-BRASIL. “Dez Maiores Falhas na Nuvem 2011, publicação em mídia online”. , dez. 2012. Disponível em: <`http://crn.itweb.com.br/27644/as-10-maiores-falhas-na-nuvem-em-2011-ate-agora/`>.

[39] CLOUDWAYS. “Five Serious Cloud Failures and Disasters of 2011, publicação em mídia online”. , 2012. Disponível em: <`http://www.cloudways.com/blog/cloud-failures-disasters-of-2011/`>.

[40] FERNANDES, N. C., DUARTE, O. C. M. B. “XNetMon: Uma Arquitetura com Segurança para Redes Virtuais”. In: *Anais do X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg'10*, pp. 339–352, Fortaleza, CE, Brazil, out. 2010.

[41] MILOJIČIĆ, D., DOUGLIS, F., PAINDAVEINE, Y., et al. “Process migration”, *ACM Computing Surveys (CSUR)*, v. 32, n. 3, pp. 241–299, 2000.

[42] DOUGLIS, F., OUSTERHOUT, J. "Transparent process migration: Design alternatives and the Sprite implementation", *Software: Practice and Experience*, v. 21, n. 8, pp. 757–785, 1991.

[43] HINES, M., GOPALAN, K. "Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning". In: *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, pp. 51–60. ACM, 2009.

[44] CLARK, C., FRASER, K., HAND, S., et al. "Live migration of virtual machines". In: *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pp. 273–286. USENIX Association, 2005.

[45] HIROFUCHI, T., NAKADA, H., ITOH, S., et al. "Enabling instantaneous relocation of virtual machines with a lightweight VMM extension". In: *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, pp. 73–83. IEEE, 2010.

[46] ROSSI, F. "Alocação dinâmica de Recursos no Xen", *Master's thesis, Dissertação (Mestrado em Ciência da Computação)-Faculdade de Informática-PUCRS, Porto Alegre*, 2005.

[47] COUTO, R., CAMPISTA, M., COSTA, L. "XTC: a throughput control mechanism for Xen-based virtualized software routers". In: *IEEE Global Communications Conference (GLOBECOM'2011)-aceito para publicaçao, Houston, Texas, EUA*, 2011.

[48] DUSTIN OWENS, B. "Securing elasticity in the cloud", *Communications of the ACM*, v. 53, n. 6, 2010.

[49] FAJJARI, I., AITSAADI, N., PUJOLLE, G., et al. "VNE-AC: Virtual network embedding algorithm based on ant colony metaheuristic". In: *Communications (ICC), 2011 IEEE International Conference on*, pp. 1–6. IEEE, 2011.

[50] ALKMIN, G. P., BATISTA, D. M., FONSECA, N. L. S. "Mapeamento de Redes Virtuais em Substratos de Rede". In: *SBRC 2011*, may 2011.

[51] CARVALHO, H. E. T., FERNANDES, N. C. F., DUARTE, O. C. M. B. "Um Controlador Robusto de Acordos de Nível de Serviço para Redes Virtuais Baseado em Lógica Nebulosa". In: *SBRC 2011*, may 2011.

[52] RUTH, P., RHEE, J., XU, D., et al. "Autonomic live adaptation of virtual computational environments in a multi-domain infrastructure". In: *Autonomic Computing, 2006. ICAC'06. IEEE International Conference on*, pp. 5–14. IEEE, 2006.

[53] PETERSON, L., ROSCOE, T. "The design principles of PlanetLab", *ACM SIGOPS Operating Systems Review*, v. 40, n. 1, pp. 11–16, 2006.

[54] GONG, Z., GU, X., WILKES, J. "PRESS: Predictive elastic resource scaling for cloud systems". In: *Network and Service Management (CNSM), 2010 International Conference on*, pp. 9–16. IEEE, 2010.

[55] YANG, W., LUDWIG, H., DAN, A. "Compatibility analysis of wsla service level objectives". In: *Workshop on the Design of Self-Managing Systems. Supplemental*, 2003.

[56] HIROFUCHI, T., NAKADA, H., ITOH, S., et al. "Reactive consolidation of virtual machines enabled by postcopy live migration". In: *Proceedings of the 5th international workshop on Virtualization technologies in distributed computing*, pp. 11–18. ACM, 2011.

[57] HOUIDI, I., LOUATI, W., ZEGHLACHE, D., et al. "Adaptive virtual network provisioning". In: *Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures*, pp. 41–48. ACM, 2010.

[58] CARVALHO, H. E. T., FERNANDES, N. C. F., DUARTE, O. C. M. B., et al. "SLAPv: A Service Level Agreement Enforcer for Virtual Networks". In: *International Conference on Computing, Networking and Communications, Internet Services and Applications Symposium (ICNC'12 - ISA)*, pp. 713–717, Maui, Hawaii, USA, jan. 2012.

[59] KECMAN, V. *Learning and soft computing: support vector machines, neural networks, and fuzzy logic models*. The MIT press, 2001. ISBN: 0262112558.

[60] ZADEH, L. "Fuzzy sets as a basis for a theory of possibility", *Fuzzy sets and systems*, v. 1, n. 1, pp. 3–28, 1978. ISSN: 0165-0114.

[61] CARVALHO, H. E. T., DUARTE, O. C. M. B. "Controle de Acordos de Nível de Serviço em Redes Virtuais Baseado em Lógica Nebulosa". In: *Projeto final de graduação da UFRJ, Engenharia de Computação e Informação*. GTA - Grupo de Teleinformática e Automação, 2011.

[62] NETO, A., COSTA, A., NETO, A., et al. “Adaptive Call Admission Control in Long Term Evolution System Based on Fuzzy Logic”. In: *II Workshop de Redes de Acesso em Banda Larga, SBRC 2012*, may 2012.

[63] JANG, J., GULLEY, N. “MATLAB Fuzzy Logic Toolbox-User’s Guide Version 1”, .

[64] CARVALHO, H. E. T., DUARTE, O. C. M. B. “VOLTAIC : Volume Optimization Layer To AssIgn Cloud resources”. In: *3rd International Conference on Information and Communications Systems, (ICICS’12 - ISA)*, Irbid, Jordan, 2012.

[65] CARVALHO, H. E. T., DUARTE, O. C. M. B. “Elastic Allocation and Automatic Migration Scheme for Virtual Machines”, *Accepted for publication in the Journal of Emerging Technologies in Web Intelligence (JETWI)*, 2012.