UNIVERSIDADE FEDERAL DO CEARÁ
DEPARTAMENTO DE COMPUTAÇÃO
CURSO DE CIÊNCIA DA COMPUTAÇÃO

EMANUEL FERREIRA COUTINHO

FOLE: UM FRAMEWORK CONCEITUAL PARA AVALIAÇÃO DE DESEMPENHO DA ELASTICIDADE EM AMBIENTES DE COMPUTAÇÃO EM NUVEM

FORTALEZA, CEARÁ

2014

EMANUEL FERREIRA COUTINHO

FOLE: UM FRAMEWORK CONCEITUAL PARA AVALIAÇÃO DE DESEMPENHO DA ELASTICIDADE EM AMBIENTES DE COMPUTAÇÃO EM NUVEM

Tese submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação.

Área de Concentração: Redes de Computadores

Orientador: Prof. Dr. José Neuman de Souza

Co-Orientador: Prof. Dr. Danielo Gonçalves Gomes

FORTALEZA, CEARÁ

2014

A000z COUTINHO, E. F..

FOLE: Um Framework Conceitual para Avaliação de Desempenho da Elasticidade em Ambientes de Computação em Nuvem / Emanuel Ferreira Coutinho. 2014.

151p.;il. color. enc.

Orientador: Prof. Dr. José Neuman de Souza

Co-Orientador: Prof. Dr. Danielo Gonçalves Gomes

Tese(Ciência da Computação) - Universidade Federal do Ceará, Departamento de Computação, Fortaleza, 2014.

1. Computação em Nuvem 2. Elasticidade 3. Análise de Desempenho I. Prof. Dr. José Neuman de Souza(Orient.) II. Universidade Federal do Ceará– Ciência da Computação(Doutorado) III. Doutor

CDD:000.0

EMANUEL FERREIRA COUTINHO

FOLE: Um Framework Conceitual para Avaliação de Desempenho da Elasticidade em Ambientes de Computação em Nuvem

Tese submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação, da Universidade Federal do Ceará, como requisito para a obtenção do grau de Doutor em Ciência da Computação.

BANCA EXAMINADORA

Prof. Dr. José Neuman de Souza
(Orientador)
Universidade Federal do Ceará – UFC

Prof. Dr. Daniel Gonçalves Gomes
(Coorientador)
Universidade Federal do Ceará – UFC

Prof. Dr. Fernando Antônio Mota Trinta
Universidade Federal do Ceará – UFC

Prof. Dr. Gabriel Antoine Louis Paillard
Universidade Federal do Ceará – UFC

Prof. Dr. Bruno Richard Schulze
Laboratório Nacional de Computação Científica – LNCC

Prof. Dr. Stênio Flávio de Lacerda Fernandes
Universidade Federal de Pernambuco – UFPE

Fortaleza, 03 de novembro de 2014

Aos meus pais Milton e Inêz, minha esposa Carla Ilane, e meu filho Lucas.

AGRADECIMENTOS

Agradeço primeiramente a Deus pela oportunidade da vida.

Agradeço a meu pai Milton e minha mãe Inêz, pois sem eles não seria nada. Sem sombra de dúvida sou o que sou devido a eles.

Obrigado a minha esposa Carla Ilane e meu filho Lucas por terem paciência comigo durante todo esse período da minha vida. Espero que eu também tenha muita paciência com vocês.

Agradeço ao meu orientador José Neuman e co-orientador Danielo pela orientação e apoio durante essa etapa de minha vida.

E finalmente, agradeço pela ajuda, dicas, dúvidas, apoio, patrocínios, viagens, revisões de artigos e publicações a todos os amigos e colegas da UFC: Instituto Universidade Virtual, MDCC, GREat, Ibituruna e UFC Quixadá.

RESUMO

Atualmente muitos clientes e provedores estão utilizando recursos de ambientes de Computação em Nuvem, tais como processamento e armazenamento, para suas aplicações e serviços. Devido à facilidade de utilização, baseada no modelo de pagamento por uso, é natural que a quantidade de usuários e suas respectivas cargas de trabalho também cresçam. Como consequência, os provedores devem ampliar seus recursos e manter o nível de qualidade acordado com os clientes, sob pena de quebras do *Service Level Agreement* (SLA) e consequentes multas. Com o aumento na utilização de recursos computacionais, uma das características principais da Computação em Nuvem tem se tornado bastante atrativa: a elasticidade. A elasticidade pode ser definida como o quanto uma nuvem computacional se adapta a variações na sua carga de trabalho através do provisionamento e desprovisionamento de recursos. Devido à pouca disponibilidade de informação em relação à configuração dos experimentos, em geral não é trivial implementar conceitos de elasticidade, muito menos aplicá-los em ambientes de nuvens computacionais. Além disso, a maneira de se medir a elasticidade não é tão óbvia, e bastante variada, não havendo ainda uma padronização para esta tarefa, e sua avaliação pode ser executada de diferentes maneiras devido às diversas tecnologias e estratégias para o provimento da elasticidade. Um aspecto comum na avaliação de desempenho da elasticidade é a utilização de recursos do ambiente, como CPU e memória, e mesmo sem ter uma métrica específica para a elasticidade, é possível se obter uma avaliação indireta. Nesse contexto, este trabalho propõe o FOLE, um *framework* conceitual para a realização de análise de desempenho da elasticidade em nuvens computacionais de maneira sistemática, flexível e reproduzível. Para apoiar o *framework*, métricas específicas para a elasticidade e métricas para sua medição indireta foram propostas. Para a medição da elasticidade em Computação em Nuvem, propomos métricas baseadas em conceitos da Física, como tensão e estresse, e da Microeconomia, como Elasticidade do Preço da Demanda. Adicionalmente, métricas baseadas em tempos de operações de alocação e desalocação de recursos, e na utilização desses recursos foram propostas para apoiar a medição da elasticidade. Para verificação e validação da proposta, dois estudos de caso foram realizados, um em uma nuvem privada e outro em uma nuvem híbrida, com experimentos projetados utilizando *microbenchmarks* e uma aplicação científica clássica, executados sobre uma infraestrutura baseada em conceitos de Computação Autonômica. Por meio desses experimentos, o FOLE foi validado em suas atividades, permitindo a sistematização de uma análise de desempenho da elasticidade. Os resultados mostram que é possível avaliar a elasticidade de um ambiente de Computação em Nuvem por meio de métricas específicas baseadas em conceitos de outras áreas de conhecimento, e também complementada por métricas relacionadas a tempos de operações e recursos de maneira satisfatória.

Palavras-chave: Computação em Nuvem. Elasticidade. Análise de Desempenho. Metodologia.

ABSTRACT

Currently, many customers and providers are using resources of Cloud Computing environments, such as processing and storage, for their applications and services. Through ease of use, based on the pay per use model, it is natural that the number of users and their workloads also grow. As a result, providers should expand their resources and maintain the agreed level of quality for customers, otherwise breaks the Service Level Agreement (SLA) and the resulting penalties. With the increase in computational resources usage, a key feature of Cloud Computing has become quite attractive: the elasticity. Elasticity can be defined as how a computational cloud adapts to variations in its workload through resources provisioning and deprovisioning. Due to limited availability of information regarding configuration of the experiments, in general is not trivial to implement elasticity concepts, much less apply them in cloud environments. Furthermore, the way of measuring cloud elasticity is not obvious, and there is not yet a standard for this task. Moreover, its evaluation could be performed in different ways due to many technologies and strategies for providing cloud elasticity. A common aspect of elasticity performance analysis is the use of environmental resources, such as CPU and memory, and even without a specific metric, to allow an indirectly assess of elasticity. In this context, this work proposes FOLE, a conceptual framework for conducting performance analysis of elasticity in Cloud Computing environments in a systematic, flexible and reproducible way. To support the framework, we proposed a set of specific metrics for elasticity and metrics for its indirect measurement. For the measurement of elasticity in Cloud Computing, we proposed metrics based on concepts of Physics, such as strain and stress, and Microeconomics, such as Price Elasticity of Demand. Additionally, we also proposed metrics based on resources allocation and deallocation operation times, and used resources, to support the measurement of elasticity. For verification and validation of the proposal, we performed two experiments, one in a private cloud and other in a hybrid cloud, using microbenchmarks and a classic scientific application, through a designed infrastructure based on concepts of Autonomic Computing. Through these experiments, FOLE had validated their activities, allowing the systematization of a elasticity performance analysis. The results show it is possible to assess the elasticity of a Cloud Computing environment using specific metrics based on other areas of knowledge, and also complemented by metrics related to time and resources operations satisfactorily.

Keywords: Cloud Computing. Elasticity. Performance Analysis. Methodology.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

ACM	*ACM Digital Library*
API	*Application Programming Interface*
ARIMA	*Auto Regressive Integrated Moving-Average*
CSV	*Comma-separated values*
GB	*Gbytes*
HPC	*High Performance Computing - Computação de Alto Desempenho*
IaaS	*Infrastructure as a Service* - Infraestrutura como um Serviço
IEEE	*IEEExplore*
IoT	*Internet of Things*
IP	*Internet Protocol*
KB	*Kbytes*
MAPE	*Monitor-Analyze-Plan-Execute*
MB	*Mbytes*
MPI	*Message Passing Interface*
MTC	*Many-Task Computing*
NIST	*National Institute of Standards and Technology*
PaaS	*Platform as a Service* - Plataforma como um Serviço
QoS	*Quality of Service - Qualidade de Serviço*
QP	Questão de Pesquisa
QS	Questão Secundária
SaaS	*Software as a Service* - Software como um Serviço
SD	*ScienceDirect*
ssh	*Secure Shell*
SLA	*Service Level Agreement - Acordo de Nível de Serviço*
SVR	*Support Vector Regression*
TAE	Tempo de Alocação Estabilizada
TASo	Tempo de Alocação Sobreprovisionada
TASu	Tempo de Alocação Subprovisionada
TAT	Tempo de Alocação Transitória
TI	Tecnologia da Informação
TRAE	Total de Recursos Alocados Estabilizados
TRASu	Total de Recursos Alocados Subprovisionados
TRASo	Total de Recursos Alocados Sobreprovisionados
URL	*Uniform Resource Locator* - Localizador Padrão de Recursos
VCPU	*Virtual CPU* - CPU Virtual
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Este documento apresenta um estudo sobre elasticidade em Computação em Nuvem. Aspectos motivacionais, metodológicos e estruturais estão descritos nesta introdução. Na Seção 1.1 deste capítulo o problema é contextualizado e caracterizado. A hipótese e as questões de partida que nortearam o desenvolvimento deste trabalho são apresentadas na Seção 1.2. Na Seção 1.3, os objetivos e as principais metas deste trabalho são elencados. Em seguida, na Seção 1.4, a metodologia utilizada para a elaboração deste trabalho é descrita. O escopo do trabalho é descrito na Seção 1.5. A Seção 1.6 apresenta as principais contribuições deste trabalho e publicações relacionadas. Por fim, a Seção 1.7 encerra o capítulo apresentando a estrutura do restante deste documento.

1.1 Contextualização e Caracterização do Problema

Com o aumento no acesso aos ambientes computacionais em nuvem e sua facilidade de utilização, baseada no modelo de pagamento por uso, é natural que a quantidade de usuários e as respectivas cargas de trabalho também cresçam. Como consequência, os provedores devem ampliar seus recursos e manter o nível de qualidade acordado com os clientes, sob pena de quebras do *Service Level Agreement* (SLA) e decorrentes multas.

Atualmente, muitos clientes e provedores estão utilizando recursos de ambientes de Computação em Nuvem, tais como processamento e armazenamento, para a execução de suas aplicações e disponibilização de serviços. Com o aumento na utilização dos recursos, uma das características principais da Computação em Nuvem tem se tornado bastante atrativa: a elasticidade.

O monitoramento de recursos computacionais, como CPU, memória e largura de banda, se torna essencial tanto para os provedores, os quais disponibilizam os serviços, quanto para seus clientes. Uma maneira de se avaliar um ambiente é monitorando algum aspecto, como utilização de recursos. Muitas vezes a elasticidade está associada a algum recurso do provedor. Na Amazon EC2, por exemplo, o cliente pode estar utilizando uma instância com poucos recursos, e em caso de necessidade de recursos, monitorado através serviço de monitoramento (Cloud Watch), o serviço de escalonamento (Auto Scaling) pode acionar que o serviço de balanceamento de carga (Elastic Load Balance) incremente a quantidade de instâncias.

Uma maneira de monitorar aplicações em nuvem de modo mais efetivo é utilizar mecanismos de Computação Autonômica, através dos quais recursos são adicionados e removidos do ambiente conforme limiares de uso pré-estabelecidos (KEPHART; CHESS, 2003). Um sistema autônomo ou autonômico é composto por um conjunto de elementos autônomos. Um elemento autônomo é um componente responsável pela gestão do seu próprio comportamento em conformidade com políticas, e por interagir com outros elementos autônomos que fornecem ou consomem serviços computacionais (KEPHART; CHESS, 2003). Mecanismos de Computação Autonômica, como *loops* de controle e regras, podem ser empregados no monitoramento de uma nuvem computacional. Assim recursos podem ser adicionados e removidos do am-

biente conforme limiares de uso pré-estabelecidos. Esse tipo de estratégia de monitoramento está diretamente associado a uma das principais características da Computação em Nuvem: a elasticidade. Esse tipo de estratégia de monitoramento está diretamente associada a uma das principais características da Computação em Nuvem: a elasticidade.

A elasticidade é uma característica essencial da Computação em Nuvem. Conforme o *National Institute of Standards and Technology* (NIST), a elasticidade é a habilidade de rápido provisionamento e desprovisionamento, com capacidade de recursos virtuais praticamente infinita e quantidade adquirível sem restrição a qualquer momento (MELL; GRANCE, 2009). Uma definição de elasticidade mais recente proposta por Herbst, Kounev e Reussner (2013) consiste no quanto um sistema é capaz de se adaptar a variações na carga de trabalho pelo provisionamento e desprovisionamento de recursos de maneira autonômica, de modo que em cada ponto no tempo os recursos disponíveis combinem com a demanda da carga de trabalho o mais próximo possível. Considerando que a elasticidade está se tornando uma necessidade cada vez maior em ambientes de nuvens computacionais devido à característica dinâmica das diversas cargas de trabalho impostas, diversos provedores estão disponibilizando serviços de monitoramento de recursos e provimento da elasticidade.

Atualmente diversos provedores de serviço disponibilizam serviços/mecanismos de elasticidade aos usuários, tais como a Amazon EC2 (AMAZONWEBSERVICES, 2013), Microsoft Azure (MICROSOFT, 2014) e HP Cloud Services (HP, 2013). Porém, a maneira de avaliar o desempenho da elasticidade não é uma tarefa fácil, pois diversos aspectos associados ao seu desempenho, tais como alocação, desalocação e utilização de recursos, são implementados de diversas maneiras. Não há um padrão na maneira de se medir a elasticidade. Além disso, existe uma grande quantidade de tecnologias e estratégias para o provimento da elasticidade. Por fim, não é comum a utilização de métricas específicas para a medição da elasticidade de uma nuvem computacional, e sim métricas indiretamente associadas.

O conceito de elasticidade é comum em algumas áreas de atuação diferentes da Computação, tais como a Física, Biologia e Microeconomia. Gambi et al. (2013) e Shawky e Ali (2012) descrevem algumas analogias da elasticidade de outras áreas em comparação à elasticidade em Computação em Nuvem. Todas essas áreas possuem o conceito de elasticidade adequados aos seus domínios. Não é diferente para a Computação em Nuvem, que possui também o conceito da elasticidade seguindo a mesma ideia central: um efeito que surge sobre um material quando este é sujeito a uma força.

Um estudo sobre elasticidade em nuvem foi realizado por Coutinho et al. (2013), com destaque para diversos aspectos como: definições, estado da arte da elasticidade, análise de desempenho, métricas, estratégicas elásticas, *benchmarks*, desafios e tendências na construção de soluções elásticas. Este estudo reforçou a importância da elasticidade atualmente tanto para provedores de acesso quanto para clientes.

Diversas estratégias para a avaliação da elasticidade foram propostas na literatura (ISLAM et al., 2012; SHAWKY; ALI, 2012; HERBST; KOUNEV; REUSSNER, 2013). Porém, a maneira de se medir a elasticidade é bastante variada, não havendo ainda uma padronização para tal tarefa. Um aspecto comum é a utilização de recursos do ambiente, como CPU, me-

mória e *throughput*, para indiretamente se avaliar a elasticidade, mesmo sem ter uma métrica específica para a elasticidade.

Diversas arquiteturas para soluções de provisionamento e manutenção do SLA utilizando recursos de Computação Autonômica em ambientes de Computação em Nuvem têm sido propostas (REGO et al., 2011; TORDSSON et al., 2012; EMEAKAROHA et al., 2012; BUYYA; CALHEIROS; LI, 2012; URIARTE; WESTPHALL, 2014). Porém, devido à pouca disponibilidade de informação acerca de sua instalação e configuração corretas, do ponto de vista experimental em geral não é trivial implementar tais arquiteturas, muito menos aplicá-las em ambientes de nuvens computacionais. Além disso, nota-se uma carência de trabalhos na literatura que avaliem o desempenho destes ambientes de forma metodológica e sistematizada.

Em geral, a análise de desempenho utiliza métricas para a medição de algum aspecto do ambiente, como CPU, memória e rede. Uma métrica para a medição da elasticidade não é fácil de se obter. Um estudo sobre métricas para elasticidade em Computação em Nuvem foi realizado por Coutinho et al. (2013), onde em geral métricas de algum recurso do ambiente são utilizadas para avaliar indiretamente elasticidade do ambiente. Poucas métricas específicas para elasticidade foram identificadas, e não são claras de se aplicar e interpretar.

Diante dos aspectos descritos, surge o problema de como realizar análise de desempenho de maneira flexível e reutilizável em ambientes de Computação em Nuvem para a avaliação da elasticidade. Em geral, trabalhos não são reutilizáveis, não são flexíveis, falta o detalhamento e descrição das atividades, assim como a maneira de utilização e interpretação das métricas, dificultando a comparação e reprodução dos resultados.

Um *framework* ou arcabouço conceitual é um conjunto de conceitos utilizados para a resolução de um problema de um domínio específico, sendo que existem dois tipos: *frameworks* verticais (ou especialistas, confeccionados através da experiência obtida em um determinado domínio específico ou de um especialista, tentando resolver problemas de um determinado domínio de aplicação), e *frameworks* horizontais (podem ser utilizados em diferentes domínios) (CRUZ, 2013).

Na tentativa de diminuir esta lacuna, foi proposto o FOLE, um *framework* conceitual para a resolução de um problema específico de análise de desempenho da elasticidade no domínio das nuvens computacionais. O FOLE suporta um conjunto de métricas para a medição direta e indireta da elasticidade, baseadas em tempos de operações de alocação e desalocação de recursos, quantidade de recursos utilizados, e alguns conceitos de elasticidade da Física e Microeconomia.

1.2 Hipótese e Questões de Partida

Considerando que a avaliação de desempenho de uma nuvem computacional é uma atividade importante para a comparação, evolução e adaptação do ambiente conforme as necessidades dos usuários e serviços, esta tese de doutorado procura testar a seguinte hipótese:

A sistematização da análise de desempenho da elasticidade em Computação em

Nuvem permite uma avaliação de quanto a nuvem se adéqua às cargas de trabalho e contribui para tomada de decisão de maneira eficiente

A partir dessa hipótese quatro questões de partida (QP) foram levantadas:

QP01: *Quais são as principais características que uma nuvem computacional deve possuir para prover elasticidade ?*

QP02: *É possível construir um mecanismo de elasticidade em um ambiente de Computação em Nuvem que permita sua adequação às necessidades dos usuários ?*

QP03: *É possível medir a elasticidade de uma nuvem computacional ?*

QP04: *É possível elaborar uma metodologia para análise de desempenho que permita avaliar a elasticidade de uma nuvem computacional ?*

1.3 Objetivos e Metas

Esta tese de doutorado tem como objetivo principal propor um *framework* conceitual para análise de desempenho da elasticidade em Computação em Nuvem, denominado FOLE, com o intuito de permitir a realização de uma análise de desempenho em ambientes de Computação em Nuvem com foco em elasticidade de maneira sistematizada, flexível e reproduzível. Para o atendimento do objetivo, este foi decomposto em seis metas (META):

META01: *Realizar uma revisão bibliográfica sobre análise de desempenho da elasticidade em ambientes Computação em Nuvem* (QP01);

META02: *Descrever uma arquitetura baseada em conceitos de Computação Autonômica que suporte a avaliação da elasticidade em Computação em Nuvem;* (QP02);

META03: *Definir métricas para a avaliação da elasticidade* (QP03);

META04: *Propor uma estratégia para análise de desempenho específica para elasticidade em Computação em Nuvem* (QP04);

META05: *Desenvolver uma ferramenta que permita o suporte à análise de desempenho na coleta e visualização de dados* (QP03, QP04).

META06: *Analisar o comportamento da elasticidade em um ambiente de Computação em Nuvem* (QP03, QP04).

1.4 Metodologia

A elaboração da tese está orientada à seguinte metodologia:

I. Estudo do referencial teórico necessário ao desenvolvimento do trabalho (conceitos e tecnologias) (META1, META2, META3);

II. Realização de uma revisão bibliográfica através de uma revisão sistemática para identificação de trabalhos relacionados, soluções, métricas e arquiteturas propostas para a análise de desempenho da elasticidade em nuvem (META1, META2, META3);

III. Comparação de trabalhos relacionados a métricas e metodologias para análise de desempenho da elasticidade em nuvem (META1, META3, META4);

IV. Planejamento de uma arquitetura baseada em conceitos de Computação Autonômica para a análise da elasticidade em nuvem, e sua consequente implementação em uma nuvem computacional (META2);

V. Proposta de métricas que auxiliem na avaliação da elasticidade por meio de aspectos variados do ambiente (META3, META6):

   a) Métricas baseadas em tempos de operações de alocação e desalocação de recursos e na utilização de recursos;

   b) Métricas baseadas em conceitos da Microeconomia e Física;

VI. Proposta de um *framework* para análise de desempenho da elasticidade em nuvem (META4);

VII. Desenvolvimento de ferramentas de suporte à análise de desempenho (coleta, automação e visualização) (META2, META5);

VIII. Projeto de experimentos para validação das métricas, arquitetura, ferramentas e *framework* (META2, META3, META4, META5, META6).

1.5 Escopo

Alguns conceitos e aspectos identificados na literatura em trabalhos de Computação em Nuvem não foram abordados neste trabalho. Entretanto, em momentos oportunos eles serão relacionados e comentados sempre que possível. Nosso foco é a proposição de métricas para a elasticidade, descrição de ferramentas de suporte (análise e visualização), construção de um *framework* conceitual para apoiar a análise de desempenho da elasticidade, e experimentos para validação das respectivas métricas e *framework*. Os itens a seguir descrevem o escopo abordado em nosso trabalho sobre alguns desses conceitos.

Computação Autonômica: Os aspectos de Computação Autonômica abordados neste trabalho foram baseados no trabalho de Kephart e Chess (2003). Apenas alguns aspectos de Computação Autonômica foram utilizados na elaboração deste trabalho: *loops* de controle, regras, arquitetura proposta e elementos de auto configuração. Demais elementos do auto gerenciamento (auto otimização, auto cura e auto proteção) não serão considerados neste trabalho.

Segurança: Este aspecto, como não é foco do trabalho, limitou-se às questões de segurança da infraestrutura utilizada, considerando apenas *login* de usuário da rede interna (nuvem privada) e da nuvem pública, conexões ssh (*Secure Shell*) e chaves de segurança. A nuvem privada se localiza na rede de computadores da universidade, e só executa serviços acadêmicos de pesquisa.

A nuvem pública utilizada foi uma conta especial cedida por um provedor, para fins acadêmicos, não possuindo muita liberdade de utilização e acesso, tendo a quantidade de instâncias e capacidade limitadas.

Nuvens Distribuídas: Nuvens distribuídas, ou D-Clouds, consistem em um grande número de pequenos *datacenters* distribuídos por uma área geográfica, sendo atraente para provedores de serviços que já possuem facilidades de desenvolvimento distribuídas interconectadas por redes de alta velocidade (SCOPE, 2011). Nuvens distribuídas possuem características semelhantes aos provedores de nuvem atuais, que além de suas ofertas essenciais, tais como serviços escaláveis, uso sob demanda e pagamento por uso, também tiram proveito da geodiversidade (ENDO et al., 2011). Neste trabalho, nuvens distribuídas não serão utilizadas, limitando-se apenas à utilização de uma nuvem privada e uma nuvem pública para os experimentos, e sua combinação consistindo uma nuvem híbrida.

Qualidade de Serviço: Este trabalho utiliza alguns conceitos de QoS (*Quality of Service*) associados à manutenção da qualidade. Porém seu foco é apenas obter uma referência para os serviços ou cargas de trabalho utilizados. Não é intenção deste trabalho desenvolver estratégias eficientes de manutenção de QoS, apesar de QoS ser um aspecto muito importante em qualquer ambiente computacional, e sim utilizá-las para a validação de algumas atividades.

Provisionamento: Não é intenção deste trabalho desenvolver estratégias de provisionamento de recursos. Apenas utilizar uma estratégia para a validação da proposta. A estratégia de provisionamento pode ser modificada a qualquer momento sem interferir na proposta deste trabalho.

Predição: Não é intenção deste trabalho o desenvolvimento de métodos de predição do comportamento das cargas de trabalho, assim como a utilização de recursos. Apenas utilizar uma estratégia para a validação da proposta. A qualquer momento a estratégia de predição pode ser modificada, sem interferir na proposta deste trabalho.

Custo Financeiro: Este trabalho não considera o aspecto financeiro nos experimentos. Mesmo assim, temos ciência da importância deste aspecto para a avaliação de provedores públicos, onde a precificação é tema contínuo de pesquisa em diversas instituições.

1.6 Contribuições

As principais contribuições desta tese foram:

1. Um *framework* conceitual para análise de desempenho da elasticidade (FOLE);
2. Métricas para a medição da elasticidade baseadas em tempos de operações de alocação e desalocação de recursos, na quantidade de recursos utilizados, e em conceitos da Física e Microeconomia;
3. Uma arquitetura baseada em conceitos da Computação Autonômica para o provimento de soluções elásticas;

4. Uma ferramenta para apoio à avaliação de desempenho da elasticidade, com suporte à visualização e análise.

A Tabela 1.1 apresenta as publicações produzidas até o momento no decorrer do curso de doutorado diretamente relacionadas ao trabalho, tanto publicadas quanto submetidas. O artigo 1 descreve uma métrica para a medição da elasticidade baseada em conceitos da Microeconomia e experimentos sobre nuvens privadas e públicas. O artigo 2 propõe métricas para a medição da elasticidade em Computação em Nuvem baseada em conceitos da Física e da Microeconomia. O artigo 3 apresenta um *survey* sobre o estado da arte da elasticidade em Computação em Nuvem. O artigo 4 apresenta uma métrica para a medição da elasticidade baseada em conceitos da Física. O artigo 5 descreve um *framework* conceitual para análide de desempenho da elasticidade. O artigo 6 descreve uma proposta de arquitetura para o provimento de elasticidade em Computação em Nuvem. O artigo 7 apresenta um conjunto de métricas propostas para a medição da elasticidade baseada em tempos de operações de alocação e desalocação de recursos e quantidade de recursos. O artigo 8 é um minicurso que apresenta os resultados de uma revisão sistemática sobre elasticidade e análise de desempenho em ambientes de Computação em Nuvem.

A Tabela 1.2 exibe as publicações que foram produzidas durante o período do doutorado que auxiliaram no estudo, conhecimento técnico e aplicações, contribuindo indiretamente para o trabalho. Os trabalhos 1, 3, 6, 7, 8, 9 e 10 auxiliaram no estudo de tecnologias relacionadas a Computação em Nuvem, focando em infraestruturas, métricas, *benchmarks* e análise de desempenho. Os trabalhos 2, 4 e 5 são trabalhos que utilizaram Computação em Nuvem com recursos de Computação de Alto Desempenho (HPC-*High Performance Computing*).

Tabela 1.1: Publicações diretamente relacionadas à tese

No.	**Referência**	**Tipo**
1	"On Applying Microeconomics Concepts to Cloud Elasticity Evaluation" - The 30th ACM/SIGAPP Symposium On Applied Computing (SAC2015)	Conferência (submetido)
2	"Metrics for Evaluating Cloud Computing Elasticity" - Computer Networks (COMNET)	*Journal* (aguardando resultado do editor)
3	"Elasticity in Cloud Computing: A Survey" - Annals of Telecommunications	*Journal* (aceito para publicação)
4	"Uma Métrica para Avaliação da Elasticidade em Computação em Nuvem Baseada em Conceitos da Física" - XII Workshop em Clouds e Aplicações (WCGA2014) (COUTINHO et al., 2014)	Conferência
5	"Uma Proposta de Framework Conceitual para Análise de Desempenho da Elasticidade em Nuvens Computacionais" - XII Workshop em Clouds e Aplicações (WCGA2014) (COUTINHO; GOMES; SOUZA, 2014b)	Conferência
6	"Uma Proposta de Arquitetura Autonômica para Elasticidade em Computação em Nuvem" - IV Workshop de Sistemas Distribuídos Autonômicos (WOSIDA2014) (COUTINHO; GOMES; SOUZA, 2014a)	Conferência
7	"An analysis of elasticity in cloud computing environments based on allocation time and resources" - IEEE Latin America Conference on Cloud Computing and Communications (LatinCloud 2013) (COUTINHO; GOMES; SOUZA, 2013)	Conferência
8	"Elasticidade em computação na nuvem: Uma abordagem sistemática" - XXXI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2013) (COUTINHO et al., 2013)	Minicurso

Tabela 1.2: Publicações indiretamente relacionadas à tese

No.	Referência	Tipo
1	"Uma Análise do Impacto da Qualidade da Internet Móvel na Utilização de Cloudlets" - XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2014) (COSTA et al., 2014)	Conferência
2	"Performance Analysis on Scientific Computing and Cloud Computing Environments" - 7th Euro American Association on Telematics and Information Systems (EATIS 2014) (COUTINHO; PAILLARD; SOUZA, 2014)	Conferência
3	"How to deploy a Virtual Learning Environment in the Cloud?" - 7th Euro American Association on Telematics and Information Systems (EATIS 2014) (COUTINHO et al., 2014)	Conferência
4	"High Performance Computing and Cloud Computing Applications at the Discipline of Distributed Applications Development" - International Journal of Computer Architecture Education (IJCAE) (COUTINHO; PAILLARD, 2013b)	*Journal*
5	"Aplicacões de Computação de Alto Desempenho e Computação em Nuvem na Disciplina de Desenvolvimento de Aplicações Distribuídas" - Workshop sobre Educação em Arquitetura de Computadores (WEAC 2013) (COUTINHO; PAILLARD, 2013a)	Conferência
6	"Estratégias para Alocação Dinâmica de Recursos em um Ambiente Híbrido de Computação em Nuvem" - XI Workshop em Clouds e Aplicações (WCGA2013) (REGO; COUTINHO; SOUZA, 2013)	Conferência
7	"Análise de Desempenho com Benchmarks em um Ambiente Público de Computação em Nuvem" - X Workshop em Clouds e Aplicações (WCGA2012) (COUTINHO et al., 2012)	Conferência
8	"Alocação Autonômica de Recursos para Máquinas Virtuais Baseada em Características de Processamento" - II Workshop on Autonomic Distributed Systems (WOSiDA2012) (REGO et al., 2012)	Conferência
9	"Proposta de Workflow para Alocação de Máquinas Virtuais Utilizando Características de Processamento" - IX Workshop em Clouds e Aplicações (WCGA2011) (REGO; COUTINHO; SOUZA, 2011)	Conferência
10	"FairCPU: Architecture for Allocation of Virtual Machines Using Processing Features" - Fourth IEEE International Conference on Utility and Cloud Computing (UCC) (REGO et al., 2011)	Conferência

1.7 Organização do Documento

Este capítulo apresentou a problemática que motiva esta tese de doutorado, a hipótese e as questões de partida envolvidas, os objetivos e as principais metas deste trabalho. Também apresentou a metodologia que será utilizada para o atendimento dos objetivos e metas, além do escopo, contribuições e publicações. O restante desse documento está organizado da seguinte maneira:

Capítulo 2 - Apresenta o referencial teórico, com alguns conceitos necessários para o melhor entendimento do trabalho. Aborda Computação em Nuvem, Análise de Desempenho e Computação Autonômica;

Capítulo 3 - Este capítulo apresenta uma revisão bibliográfica, com a identificação de trabalhos relacionados à elasticidade, análise de desempenho, métricas, ferramentas e soluções elásticas, por meio de uma revisão sistemática. Também são descritos, analisados e comparados trabalhos relacionados;

Capítulo 4 - Métricas propostas para a medição da elasticidade e para a medição de aspectos

relacionados à elasticidade são apresentadas neste capítulo;

Capítulo 5 - O *framework* conceitual FOLE é apresentado, com o detalhamento de suas atividades. Adicionalmente, uma arquitetura para uma solução elástica é proposta, baseada em conceitos de Computação Autonômica, e uma ferramenta para suportar a análise de desempenho é descrita;

Capítulo 6 - Experimentos para a validação do FOLE e das métricas propostas são descritos. Dois experimentos são realizados, utilizando nuvens privadas e públicas.

Capítulo 7 - Este capítulo é dedicado às considerações finais desta tese de doutorado. Nele são descritos os resultados alcançados e uma discussão sobre as questões de partida. Por fim, trabalhos futuros derivados desta tese são apresentados.

Apêndices - Os apêndices apresentam material complementar deste trabalho. Métricas complementares e telas da ferramenta desenvolvida são exibidas.

2 REFERENCIAL TEÓRICO

Neste capítulo são descritos conceitos necessários ao entendimento do trabalho. Aspectos da Computação em Nuvem, Análise de Desempenho e Computação Autonômica são apresentados de maneira geral.

2.1 Computação em Nuvem

A Computação em Nuvem está se tornando uma das palavras chaves da indústria de Tecnologia da Informação (TI). A nuvem é uma metáfora para a Internet ou infraestrutura de comunicação entre os componentes arquiteturais, baseada em uma abstração que oculta a complexidade da infraestrutura. Cada parte desta infraestrutura é provida como um serviço os quais normalmente são alocados em centros de dados, utilizando *hardware* compartilhado para computação e armazenamento (BUYYA et al., 2009).

A infraestrutura do ambiente de Computação em Nuvem normalmente é composta por um grande número (centenas ou milhares) de máquinas físicas ou nós físicos de baixo custo, conectadas por meio de uma rede, como ilustra a Figura 2.1. Cada máquina física pode ter diferentes configurações de *hardware* e *software*, com variações na capacidade em termos de CPU, memória e armazenamento em disco (SOROR et al., 2010). Em cada máquina física existe um número variável de máquinas virtuais, nós virtuais ou instâncias em execução, de acordo com a capacidade do *hardware* disponível na máquina física.

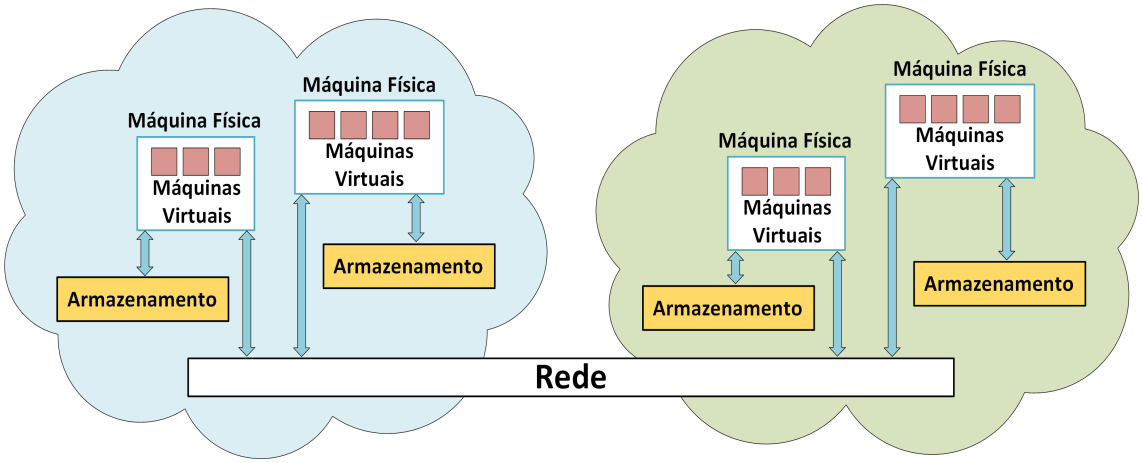


Figura 2.1: Ambiente de Computação em Nuvem

Em 1961, John McCarthy[1] foi a primeira pessoa a sugerir publicamente que a tecnologia de compartilhamento do tempo computacional pode resultar em um futuro no qual o poder de computação e até mesmo aplicações específicas poderiam ser vendidos por meio do modelo de negócio utilitário (como água ou eletricidade). Essa ideia de um computador ou informações de utilidade foi muito popular durante o final da década 60, mas desapareceu em

[1]John McCarthy, em um discurso para celebrar o centenário do MIT em 1961, "*Architects of the Information Society, Thirty-Five Years of the Laboratory for Computer Science at MIT*"

meados da década de 90. No entanto, por volta do ano 2000, a ideia ressurgiu em novas formas, tais como provedores de serviços de aplicações, Computação em Grade e Computação em Nuvem.

A Computação em Nuvem é uma evolução dos serviços e produtos de tecnologia da informação sob demanda, também chamada de *Utility Computing* (BRANTNER et al., 2008). O objetivo da *Utility Computing* é fornecer componentes básicos como armazenamento, processamento e largura de banda de uma rede como uma "mercadoria" através de provedores especializados com um baixo custo por unidade utilizada. Usuários de serviços baseados em *Utility Computing* não precisam se preocupar com escalabilidade, pois a capacidade fornecida é praticamente infinita. A *Utility Computing* propõe fornecer disponibilidade total, isto é, os usuários podem ler e gravar dados a qualquer tempo, sem nunca serem bloqueados, e os tempos de resposta são quase constantes e não dependem do número de usuários simultâneos, do tamanho do banco de dados ou de qualquer parâmetro do sistema. Os usuários não precisam se preocupar com *backups*, pois se os componentes falharem, o provedor é responsável por substituí-los e tornar os dados disponíveis em tempo hábil (BRANTNER et al., 2008).

Uma razão importante para a construção de novos serviços baseados em *Utility Computing* é que provedores de serviços que utilizam serviços de terceiros pagam apenas pelos recursos que recebem, ou seja, pagam pelo uso. Não são necessários grandes investimentos iniciais em TI e o custo cresce de forma linear e previsível com o uso. Dependendo do modelo do negócio, é possível que o provedor de serviços repasse o custo de armazenagem, processamento e de rede para os usuários finais, já que é realizada a contabilização do uso.

Existem diversas propostas para definir o paradigma da Computação em Nuvem, resumidas por Vaquero et al. (2009). O *National Institute of Standards and Technology* (NIST) argumenta que a Computação em Nuvem é um paradigma em evolução e apresenta a seguinte definição: *"Computação em nuvem é um modelo que possibilita acesso, de modo conveniente e sob demanda, a um conjunto de recursos computacionais configuráveis (por exemplo, redes, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente adquiridos e liberados com mínimo esforço gerencial ou interação com o provedor de serviços"* (MELL; GRANCE, 2009). Ainda segundo o NIST, a Computação em Nuvem é composta por cinco características essenciais, três modelos de serviço e quatro modelos de implantação, detalhados a seguir.

2.1.1 Características Essenciais

As características essenciais são vantagens que as soluções de Computação em Nuvem oferecem. Algumas destas características, em conjunto, definem exclusivamente a Computação em Nuvem e fazem a distinção com outros paradigmas. Por exemplo, a elasticidade rápida de recursos, amplo acesso e a medição de serviço são características básicas para compor uma solução de Computação em Nuvem.

- *Self-service sob demanda*: O usuário pode adquirir unilateralmente recursos computacionais, como tempo de processamento no servidor ou armazenamento na rede, na medida

em que necessite e sem precisar de interação humana com os provedores de cada serviço;

- *Amplo acesso*: Recursos são disponibilizados por meio da rede e acessados através de mecanismos padronizados que possibilitam o uso por plataformas do tipo *thin*, tais como celulares, *laptops* e PDAs;

- *Pooling de recursos*: Os recursos computacionais do provedor são organizados em um *pool* para servir múltiplos usuários usando um modelo *multi-tenant* ou multi-inquilino, com diferentes recursos físicos e virtuais, dinamicamente atribuídos e ajustados de acordo com a demanda dos usuários. Estes usuários não precisam ter conhecimento da localização física dos recursos computacionais, podendo somente especificar a localização em um nível mais alto de abstração, tais como o país, estado ou centro de dados;

- *Elasticidade rápida*: Recursos podem ser adquiridos de forma rápida e elástica, em alguns casos automaticamente, caso haja a necessidade de escalar com o aumento da demanda, e liberados, na retração dessa demanda. Para os usuários, os recursos disponíveis para uso parecem ser ilimitados e podem ser adquiridos em qualquer quantidade e a qualquer momento;

- *Serviço medido*: Sistemas em nuvem automaticamente controlam e otimizam o uso de recursos por meio de uma capacidade de medição. A automação é realizada em algum nível de abstração apropriado para o tipo de serviço, tais como armazenamento, processamento, largura de banda e contas de usuário ativas. O uso de recursos pode ser monitorado e controlado, possibilitando transparência para o provedor e o usuário do serviço utilizado.

2.1.2 Modelos de Serviços

O ambiente de Computação em Nuvem é composto de três modelos de serviços. Estes modelos são importantes, pois eles definem um padrão arquitetural para soluções de Computação em Nuvem, conforme descrito na Figura 2.2.

- *Software como um Serviço (SaaS)*: O modelo de SaaS proporciona sistemas de *software* com propósitos específicos que são disponíveis para os usuários por meio da Internet e acessíveis a partir de vários dispositivos do usuário por meio de uma *interface thin client* como um navegador web. No SaaS, o usuário não administra ou controla a infraestrutura subjacente, incluindo rede, servidores, sistema operacional, armazenamento ou mesmo as características individuais da aplicação, exceto configurações específicas. Como exemplos de SaaS podemos destacar os serviços de *Customer Relationship Management* (CRM) da Salesforce e o Google Drive.

- *Plataforma como um Serviço (PaaS)*: O modelo de PaaS fornece sistema operacional, linguagens de programação e ambientes de desenvolvimento para as aplicações, auxiliando a implementação de sistemas de *software*. Assim como no SaaS, o usuário não administra ou controla a infraestrutura subjacente, mas tem controle sobre as aplicações implantadas

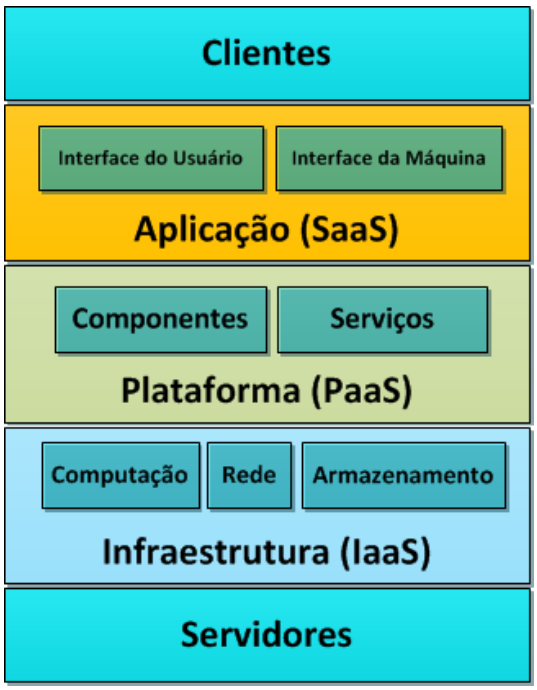


Figura 2.2: Modelo de serviço da Computação em Nuvem (baseado em (VERDI et al., 2010)).

e, possivelmente, as configurações de aplicações hospedadas nesta infraestrutura. Google App Engine (GOOGLE, 2014) e Microsoft Azure (AZURE, 2012) são exemplos de PaaS.

- *Infraestrutura como um Serviço (IaaS)*: A IaaS torna mais fácil e acessível o fornecimento de recursos, tais como servidores, rede, armazenamento e outros recursos de computação fundamentais para construir um ambiente de aplicação sob demanda, que podem incluir sistemas operacionais e aplicativos. Em geral, o usuário não administra ou controla a infraestrutura da nuvem, mas tem controle sobre os sistemas operacionais, armazenamento, aplicativos implantados e, eventualmente, seleciona componentes de rede, tais como *firewalls*. O Amazon Elastic Cloud Computing (EC2) (AMAZONWEBSERVICES, 2013), Eucalyptus (LIU; LIANG; BROOKS, 2007), OpenNebula (OPENNEBULA.ORG, 2012) e OpenStack (OPENSTACK.ORG, 2013) são exemplos de IaaS.

2.1.3 Modelos de Implantação

Quanto ao acesso e à disponibilidade, há diferentes tipos de modelos de implantação para os ambientes de Computação em Nuvem. A restrição ou abertura de acesso depende do processo de negócios, do tipo de informação e do nível de visão desejado, conforme descrito na Figura 2.3:

- *Nuvem Privada*: A infraestrutura de nuvem é utilizada exclusivamente por uma organização, sendo esta nuvem, local ou remota, administrada pela própria empresa ou por terceiros;

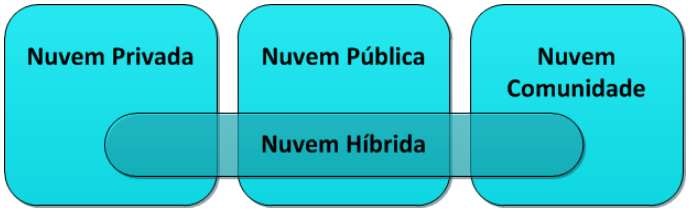


Figura 2.3: Modelos de implantação da Computação em Nuvem.

- *Nuvem Pública*: A infraestrutura de nuvem é disponibilizada para o público em geral, sendo acessada por qualquer usuário que conheça a localização do serviço;
- *Nuvem Comunidade*: Fornece uma infraestrutura compartilhada por uma comunidade de organizações com interesses em comum;
- *Nuvem Híbrida*: A infraestrutura é uma composição de duas ou mais nuvens, que podem ser do tipo privada, pública ou comunidade, e que continuam a ser entidades únicas, mas conectadas por meio de tecnologia proprietária ou padronizada que permite a portabilidade de dados e aplicações.

2.2 Análise de Desempenho

Muitos trabalhos executam atividades para avaliar o desempenho de diversos aspectos, tais como um ambiente, uma aplicação, uma política, um algoritmo, entre outros. Em Computação em Nuvem não é diferente, pois diversas áreas são objeto de avaliações, podendo a análise de desempenho ser utilizada para compreender o nível de qualidade dos serviços e possibilitar o ajuste da elasticidade por meio de políticas de alocação de recursos.

Análise de desempenho envolve coleta de dados para auxiliar clientes e patrocinadores a definir e alcançar seus objetivos. Apresenta várias perspectivas sobre um problema ou oportunidade, determinando o direcionamento para barreiras ou desempenho bem sucedido, e propõe uma solução com base nas descobertas. Inclui: medição, modelagem, estatística, projeto experimental, simulação, teoria das filas, etc. Em geral são identificados três aspectos:

- Sistema: Qualquer conjunto de *hardware*, *software* e/ou *firmware*;
- Métricas: Critérios utilizados para avaliar o desempenho do sistema ou componentes;
- Cargas de trabalho: Requisições realizadas pelos usuários do sistema.

É possível realizar diversas atividades com análise de desempenho, tais como especificar requisitos do desempenho, avaliar alternativas do projeto, comparar dois ou mais sistemas, determinar o valor ótimo de um parâmetro (ajuste do sistema), encontrar gargalos de desempenho, caracterizar a carga de trabalho no sistema, determinar o número e tamanho dos componentes (capacidade planejamento) e predizer o desempenho de futuras cargas. Jain (1991) realiza um estudo detalhado sobre análise de desempenho. Neste estudo, uma abordagem para avaliação de desempenho é descrita:

- Definir metas e o sistema;
- Listar os serviços e seus resultados;
- Selecionar métricas;
- Listar parâmetros;
- Selecionar fatores para estudo;
- Selecionar técnica de avaliação;
- Selecionar carga de trabalho;
- Projetar os experimentos;
- Analisar e interpretar os dados;
- Apresentar os resultados.

Contudo, na abordagem não existe a noção de plano de ação conforme os resultados vão sendo coletados e interpretados, e nem há a idéia do ciclo de vida da análise de desempenho. Além disso, Raj Jain aborda predominantemente o projeto do experimento e a intrepretação dos dados. Segundo Montgomery (2009), um experimento é um teste ou uma série de testes nas quais mudanças intencionais são realizadas para que variáveis de entrada de um processo ou sistema possam ser observadas e identificar as razões para mudanças que podem ser observadas nas respostas de saída. Em geral experimentos são utilizados para estudar o desempenho de processos e sistemas, e estes podem ser representados pelo modelo da Figura 2.4. Geralmente um processo é visualizado como uma combinação de máquinas, métodos, pessoas e outros recursos, que transformam alguma entrada em alguma saída, que possui uma ou mais respostas observáveis. Algumas das variáveis desse sistema são controláveis enquanto outras não. Os objetivos de um experimento podem incluir o seguinte:

- Determinação de quais variáveis influenciam mais na resposta;
- Determinação de onde configurar a variável que mais influencia de maneira que a resposta seja quase sempre próxima ao resultado desejado;
- Determinação de onde configurar a variável que mais influencia de modo que a variação na resposta seja pequena;
- Determinação de onde configurar a variável que mais influencia de modo que os efeitos das variáveis não controladas sejam minimizados.

Em geral, a abordagem de planejar e conduzir o experimento é chamado de estratégia de experimentação. Projeto experimental é uma importante ferramenta para melhorar o desempenho de processos e sistemas, assim como no desenvolvimento de novas aplicações. A aplicação de técnicas de projeto experimental previamente pode resultar em: rendimento do

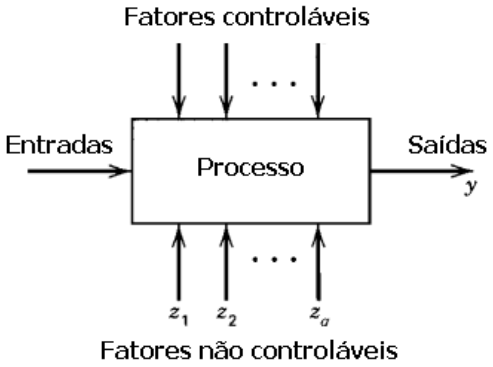


Figura 2.4: Modelo genérico de um processo ou sistema. Fonte: Adaptado de Montgomery (2009).

processo ou sistema melhorado, variabilidade reduzida e conformidade próximos com requisitos ou metas, tempo de desenvolvimento reduzido e redução dos custos globais. Métodos de projeto experimental também ajudam na tomada de decisão, como por exemplo: avaliação e comparação de configurações, avaliação de alternativas, seleção de parâmetros de projeto onde um produto trabalha bem sob uma grande variedade de condições, e determinação de parâmetros chave de projeto que impactam no desempenho do produto. O projeto estatístico de experimentos se refere ao processo de planejamento de maneira que os dados possam ser apropriadamente analisados por métodos estatísticos, resultando em conclusões objetivas e válidas. Montgomery (2009) descreve algumas orientações para o projeto de experimentos:

- Reconhecimento e declaração do problema;
- Escolha dos fatores, níveis e intervalos;
- Seleção das variáveis de resposta;
- Escolha do projeto experimental;
- Realização do experimento;
- Análise estatística dos dados;
- Conclusões e recomendações.

Menasce, Dowdy e Almeida (2004) descreve como as métricas podem ser agrupadas da seguinte maneira: tempo de resposta (tempo que o sistema leva para reagir a uma requisição humana, geralmente medido em segundos), *throughput* (taxa na qual requisições são completadas a partir de um sistema computacional, e medido em operações por unidade de tempo), disponibilidade (fração de tempo que um sistema está ativo e disponível para clientes), confiabilidade (probabilidade de que o sistema funcione adequadamente e continuamente sobre um período fixo de tempo), segurança (combinação de confidencialidade, integridade de dados e não repudiação), escalabilidade (um sistema é dito escalável quando seu desempenho não se

degrada significantemente com o aumento dos usuários ou carga), e extensibilidade (propriedade do sistema de evolução para lidar com os novos requisitos funcionais e de desempenho).

Jain (1991) também agrupa as métricas de maneira semelhante, porém mais simplificada: tempo de resposta, *throughput*, utilização (fração de tempo que o recurso está ocupado atendendo requisições), confiabilidade, disponibilidade (período de tempo em que um sistema está indisponível (*downtime*) e período de tempo em que um sistema está disponível (*uptime*)), aquisição de sistemas (custo e custo/desempenho). Algumas métricas são compartilhadas entre diferentes grupos, dependendo do contexto de utilização, por exemplo: tempo de alocação e de desalocação de recursos são métricas de tempo de resposta, mas também são métricas de escalabilidade.

2.3 Computação Autonômica

Atualmente, os sistemas estão cada vez se tornando mais complexos, seja com a utilização de tecnologias diversas, seja pela complexidade das aplicações (DENKO; YANG; ZHANG, 2009). A Computação Autonômica aborda o projeto e implementação de sistemas computacionais com capacidade de se configurar, proteger, otimizar, tolerar falhas e solucionar problemas automaticamente, respondendo às mudanças do sistema (PARASHAR; HARIRI, 2006). O termo "autônomo" teve origem na Biologia. Baseado nestes conceitos, soluções para auto gerenciamento têm sido propostas.

Segundo Kephart e Chess (2003), um sistema autonômico é composto por um conjunto de elementos autonômicos. Um elemento autonômico é um componente responsável pela gestão do seu próprio comportamento em conformidade com políticas e por interagir com outros elementos autônomos que fornecem ou consomem serviços computacionais. Segundo Hariri et al. (2006), Computação Autonômica é inspirada no sistema nervoso autonômico humano, que tem desenvolvido estratégias e algoritmos para lidar com complexidades e incertezas, e objetiva a construção de sistemas e aplicações capazes de gerenciarem a si mesmos, com o mínimo de intervenção humana.

2.3.1 Comportamento dos Elementos Autonômicos

Para White et al. (2004), um elemento autonômico deve ser autogerenciado, ou seja, ele deve ser responsável por se configurar internamente para reparação de falhas internas, otimização de seus próprios comportamentos, e auto proteção contra ataques. Para simplificar o gerenciamento de grandes sistemas, um elemento autonômico deve tratar os problemas localmente sempre que possível. Ele deve ser capaz de estabelecer e manter relacionamentos com outros elementos autonômicos os quais provêm e demandam serviços, induzindo vários requisitos (WHITE et al., 2004). Um elemento autonômico deve descrever seus serviços precisamente de maneira que seja acessível e compreensível por outros elementos autonômicos. Relacionamentos são baseados em acordos, então o elemento autonômico deve entender e manter os termos do acordo, devendo ser capaz de negociar para estabelecer acordos.

Além disso, um elemento autonômico deve gerenciar seu comportamento e relacionamentos de forma a atender obrigações, tanto pelo ajuste ou configuração de parâmetros próprios, quanto pela projeção sobre novos recursos de outros elementos autonômicos. Existem dois tipos de obrigações nas quais um elemento autonômico pode se sujeitar: honrar os termos do acordo, e receber e manter políticas. Um elemento autonômico deve rejeitar qualquer requisição de serviço que viole suas políticas ou acordos, assim como recusar ou propor outras relações e políticas que possam causar violações. Ele deve possuir capacidades analíticas suficientes para suportar estas funcionalidades (WHITE et al., 2004).

2.3.2 Políticas

A habilidade em alto nível de tradução de diretivas para ações específicas a serem tomadas pelos elementos é executada pelo uso de políticas (WHITE et al., 2004). Uma política é a representação de comportamentos desejados ou restrições sobre o comportamento. Gerenciamento baseado em políticas tem sido tópico de pesquisa por mais de uma década (SLOMAN, 1994). Para Computação Autonômica, o foco é especificamente sobre auto gerenciamento baseado em políticas. Kephart e Walsh (2004) destacam três formas de políticas: atuação, objetivo e funções utilitárias.

No nível mais baixo de especificação estão políticas de atuação, as quais estão tipicamente sob a forma de SE (CONDIÇÃO) ENTÃO (AÇÃO). Por exemplo SE (Tempo de Resposta > 2 segundos) ENTÃO (Incremente CPU em 20%). Um elemento autonômico empregando políticas de ação deve medir ou sintetizar os estados quantificados na condição, e deve executar as ações estabelecidas sempre que uma condição for satisfeita. O problema com as políticas de ação é que, quando uma série de políticas de ação é especificada, conflitos podem surgir entre as políticas, sendo difíceis de detectar. Por exemplo, se um servidor *web* e um servidor de aplicação de um sistema multicamadas solicitarem mais recursos, e os recursos disponíveis forem insuficientes para ambos, surge um conflito. Para lidar com tais conflitos, o sistema precisa de políticas de resolução de conflitos adicionais, por exemplo, dando maior prioridade ao servidor de aplicativos. No entanto, como os conflitos aparecem em tempo de execução, é difícil especificar as políticas de resolução de conflitos (SOUSA, 2010).

No nível seguinte, condições são atendidas sem se especificar como atingí-las. Por exemplo: o tempo de resposta não deve exceder 2 segundos. Políticas de objetivo são mais poderosas que políticas de ação pois um humano ou elemento autonômico podem direcionar para outro elemento sem requerer conhecimento detalhado. Elementos autonômicos que empregam políticas de objetivo devem possuir capacidade de planejamento ou modelagem suficientes para traduzir objetivos em ações. Políticas de objetivo liberam o administrador da necessidade de especificar as ações de baixo nível. No entanto, as políticas de objetivo ainda sofrem o problema de conflito, também encontrados com as políticas de ação. Por exemplo, quando o conjunto de estados desejados para atender um objetivo do servidor *web* e um conjunto de estados desejados para cumprir um objetivo de servidor de aplicativos são disjuntos (interseção vazia), surgem conflitos. O problema é encontrar entre os estados indesejáveis o que causa menor prejuízo (SOUSA, 2010).

No nível mais alto estão as políticas de funções utilitárias, que especificam o desejo alternativo de estados alternativos. Isto é executado pela associação numérica de valores ou pela ordenação total ou parcial dos estados possíveis. Funções utilitárias são mais poderosas que políticas de objetivo porque elas automaticamente determinam o objetivo mais valioso em qualquer situação. Elementos autonômicos que empregam políticas de função utilitária devem possuir sofisticadas capacidades de otimização e modelagem suficientes para traduzir funções utilitárias em ações. Por outro lado, as funções utilitárias são difíceis de definir assim como cada aspecto que influencia a decisão desta função que deve ser quantificado (SOUSA, 2010).

2.3.3 Elementos do Auto Gerenciamento

Elementos do auto gerenciamento podem ser utilizados para instituir auto configuração, auto reparação, auto otimização e auto proteção no nível de sistema (WHITE et al., 2004). Esta lista é descrita em Kephart e Chess (2003), White et al. (2004), Sousa (2010). Existem outras estratégias, outros padrões, portanto esta lista de elementos não é exaustiva.

- Auto Configuração: Um sistema autonômico se instala, se configura e modifica seu comportamento automaticamente conforme metas de alto nível que especificam o que é desejado, mas não necessariamente como realizá-las. Na configuração interna pode ocorrer a adição ou retirada de componentes, e em seguida, a auto configuração. Em relação à configuração externa, tem-se a adaptação à infraestrutura global, ambiente e serviços;

- Auto Reparação: Um sistema autonômico detecta problemas, seja de *hardware* ou *software*, realiza o diagnóstico e tenta automaticamente corrigí-los. Se um problema pode ser resolvido automaticamente, o sistema o faz. Para isso, o sistema executa um conjunto de testes para se certificar que a correção não introduziu outros defeitos. Caso não seja possível executar a auto reparação, o problema é relatado para o administrador do sistema. Para realizar a auto reparação, os seguintes passos são executados: detecção, isolamento, correção e reintegração. A detecção consiste em identificar um problema, situação ou comportamento fora dos padrões. Assim, é possível isolar a parte do sistema impactada, de forma a não comprometer o restante do sistema. A correção trata da solução do problema e a reintegração consiste na adição da parte afetada e corrigida ao sistema;

- Auto Otimização: Um sistema autonômico busca continuamente formas para otimizar a utilização dos recursos, a fim de melhorar o desempenho. Assim, em oposição a um comportamento reativo, o sistema autonômico pode proativamente decidir fazer uma alteração no sistema. Na auto otimização, o sistema monitora e gerencia recursos automaticamente em ambientes imprevisíveis, maximizando a utilização sem a intervenção humana;

- Auto Proteção: Um sistema autonômico antecipa, detecta, identifica e se protege de ataques. O sistema prevê falhas de segurança e tenta impedí-los (comportamento proativo). Algumas ações da auto proteção estão relacionados à especificação e gerenciamento do acesso aos recursos, proteção contra acessos não autorizados, detecção intrusões e relato ao administrador do sistema à medida que estes ocorrem.

2.3.4 Considerações Arquiteturais

Segundo Kephart e Chess (2003), sistemas autonômicos são coleções interativas de elementos autonômicos (sistemas individuais constituintes que contém recursos e distribuem serviços para humanos e outros elementos autonômicos). Elementos autonômicos gerenciam seu comportamento interno e relacionamentos com outros elementos autonômicos conforme políticas estabelecidas. O gerenciamento do próprio sistema resulta em diversas interações através de elementos autonômicos do auto gerenciamento dos elementos autonômicos individuais. A Figura 2.5 exibe um elemento autonômico tipicamente composto por um ou mais elementos gerenciados acoplados a um único gerenciador autonômico, que os controla. O elemento gerenciado essencialmente é equivalente ao encontrado em sistemas não autonômicos, embora ele possa ser adaptado para habilitar que o gerente autonômico o controle e monitore. O elemento gerenciado pode ser um recurso de *hardware*, como uma CPU ou impressora, ou um recurso de *software*, como um banco de dados ou um sistema legado.

Cada elemento autonômico é responsável pelo gerenciamento de seus estados internos, comportamento e interações com o ambiente, que consiste de sinais e mensagens de outros elementos e do mundo externo. O comportamento de um elemento interno e seus relacionamentos com outros elementos é orientado pelos objetivos embutidos por seus projetistas, por outros elementos que possuem autoridade sobre ele ou subcontratos com elementos pares

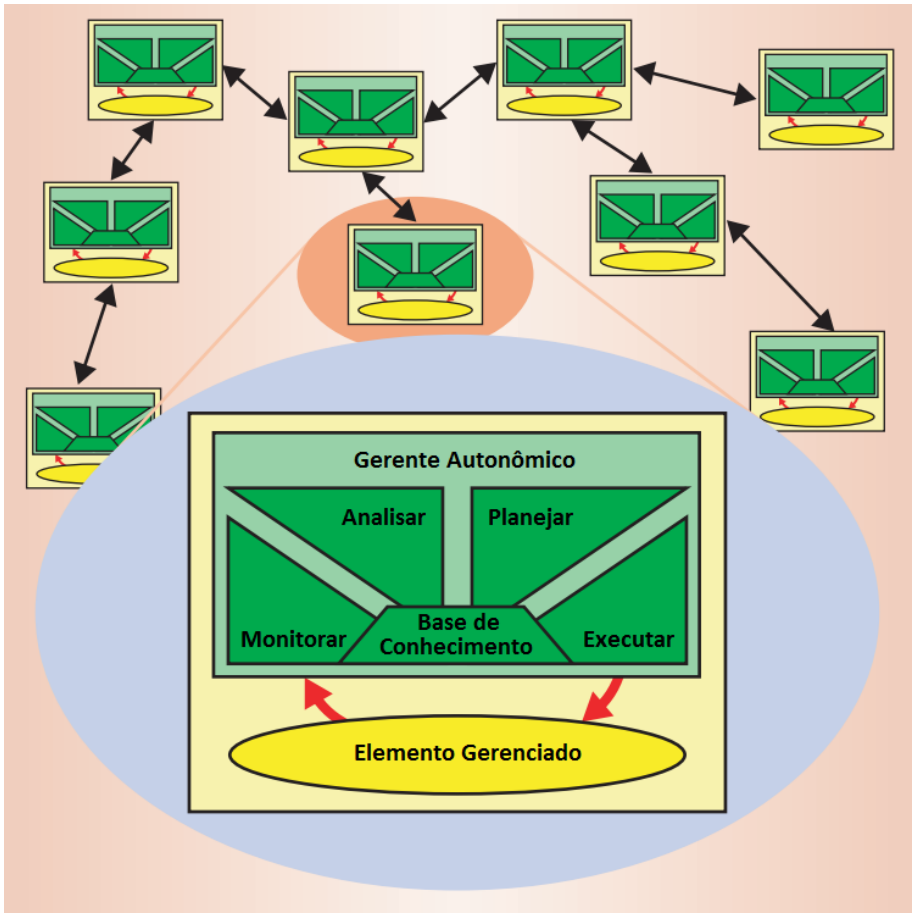


Figura 2.5: Estrutura de um elemento autonômico. Fonte: (KEPHART; CHESS, 2003)

Tabela 2.1: Funções internas de um gerente autonômico

Função	**Definição**
Monitorar	Função do gerente autonômico que coleta, agrega, filtra e reporta detalhes, como métricas e topologias, que foram coletadas de recursos gerenciados.
Analisar	Função do gerente autonômico que correlaciona e modela situações complexas, como previsão de séries temporais ou modelos de filas, para compreender o estado atual do sistema.
Planejar	Função do gerente autonômico que estrutura as ações necessárias para executar metas e objetivos.
Executar	Função do gerente autonômico que muda o comportamento do recurso gerenciado utilizando atuadores, baseado em ações recomendadas pela função de planejamento.

com consentimento explícito. O elemento pode requerer assistência de outros elementos para executar metas. Em caso afirmativo, ele será responsável por obter os recursos necessários de outros elementos e por lidar com casos de exceção, como a falha de um recurso requerido.

Um elemento ou recurso gerenciado é um componente do sistema controlado, podendo ser um recurso único ou uma coleção de recursos, controlado por sensores e atuadores (IBM, 2004). No entanto, ao contrário dos elementos não autonômicos, um elemento gerenciado fornece um conjunto de sensores e atuadores que são utilizados pelo gerente autonômico para monitorar e controlar.

O gerente autonômico é um componente que implementa um *loop* de controle, e realiza ações de: gerenciar a coleta (filtragem e relatório de dados coletados pelos sensores); analisar e aprender sobre o elemento gerenciado; e acumular conhecimento e predizer ações futuras (IBM, 2004). O gerente autonômico também fornece sensores e atuadores para ser utilizado por outros elementos autonômicos em uma infraestrutura distribuída (SOUSA, 2010). Segundo IBM (2005), um evento é uma mudança significante no estado de um recurso do sistema. Um evento pode ser gerado por um problema, para a resolução de um problema ou para a execução de uma tarefa. A Tabela 2.1 exibe ações de um ciclo de vida de um gerente autonômico.

Sensores provêm mecanismos para coletar informações sobre o estado e transições de um elemento (IBM, 2004). Um sensor consiste em: um conjunto de propriedades que expõem informação sobre o estado atual de um recurso gerenciado, acessadas através de operações padrão de consulta; e um conjunto de eventos de gerenciamento (não solicitados, mensagens assíncronas ou notificações) que ocorrem quando o recurso gerenciado sofre alterações de estado que mereça relato (IBM, 2005).

Atuadores são mecanismos que modificam o estado do elemento conforme decisões tomadas pelo sistema autônomo (IBM, 2004). Um atuador consiste em: uma coleção de operações padrão de escrita que permitem alterar o estado do elemento gerenciado; e uma coleção de operações que são implementadas pelos gerentes autonômicos que permite aos recursos gerenciados realizar requisições ao seu gerente (IBM, 2005).

A Figura 2.6 exibe uma arquitetura semelhante à arquitetura proposta por Kephart e Chess (2003), porém com mais detalhes e inclusão dos sensores e atuadores. Em essência, as duas arquiteturas são idênticas, pois possuem as mesmas funcionalidades e objetivos, além das funções de monitorar, analisar, planejar e executar.

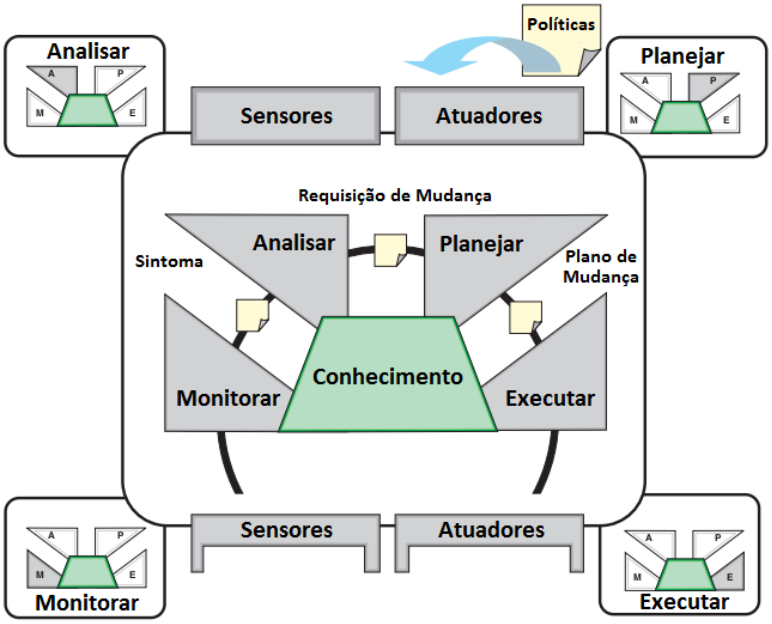


Figura 2.6: Detalhes funcionais de um gerente autonômico. Fonte: (IBM, 2005)

3 TRABALHOS RELACIONADOS

Elasticidade é um ponto chave para a adição de QoS através de serviços em nuvem. Ela permite que provedores adicionem ou removam recursos, sem interrupção e em tempo de execução, para lidar com variações nas cargas de trabalho aplicadas. Esses recursos podem ser adquiridos rapidamente, em alguns casos automaticamente, para atender ao aumento e redução da carga de trabalho (MELL; GRANCE, 2009). Aspectos relacionados à elasticidade têm recebido muita atenção, tais como tempo de resposta, cargas de trabalho e SLA. Estas questões são importantes em ambientes de Computação em Nuvem porque provedores precisam adicionar recursos conforme as cargas de trabalho para prevenir violações de SLA, e remover recursos quando a carga de trabalho diminui.

Elasticidade é comumente associada à escalabilidade de sistemas. Porém existem diferenças. Escalabilidade é definida como a habilidade de um sistema adicionar mais recursos para atender a grandes cargas de trabalho (ISLAM et al., 2012). Elasticidade consiste no crescimento e redução de recursos conforme as cargas de trabalho. A escalabilidade somente considera o crescimento da carga de trabalho, não considera o tempo, e não captura quanto tempo leva para o sistema atender o nível de desempenho desejado. Diferentemente, o tempo para a elasticidade é um aspecto central, o qual depende da velocidade de resposta a uma carga de trabalho.

Embora diversos estudos em Computação em Nuvem tenham explorado o tema elasticidade (COSTA et al., 2011; SHARMA et al., 2011; SULEIMAN et al., 2012; ISLAM et al., 2012; GALANTE; BONA, 2012), o processo de análise do estado da arte para a seleção de trabalhos relacionados necessita de uma metodologia para sua elaboração e execução, e assim evitar algo incompleto ou uma fraca revisão de literatura.

3.1 Adaptação da Revisão Sistemática

Uma revisão sistemática é uma revisão orientada a um protocolo que sintetiza estudos com foco em um tópico de pesquisa ou relacionado à questões chave (RUSSELL; CHUNG; BALK, 2009). Por meio da utilização de um processo controlado e formal de pesquisa bibliográfica, espera-se que os resultados retornem os tópicos mais pesquisados, lacunas, desafios, processos, ferramentas e técnicas. Para a execução da revisão foi utilizado como base o guia para revisão sistemática de Kitchenham (2004) com algumas adaptações (simplificação das atividades, não discussão das vulnerabilidades da revisão e não apresentação dos dados de maneira estatística). A Figura 3.1 exibe a adaptação utilizada neste trabalho.

3.1.1 Atividade 1: Planejamento da Revisão

Descreve as atividades necessárias para o planejamento da revisão.

- **Identificar Necessidade da Revisão**

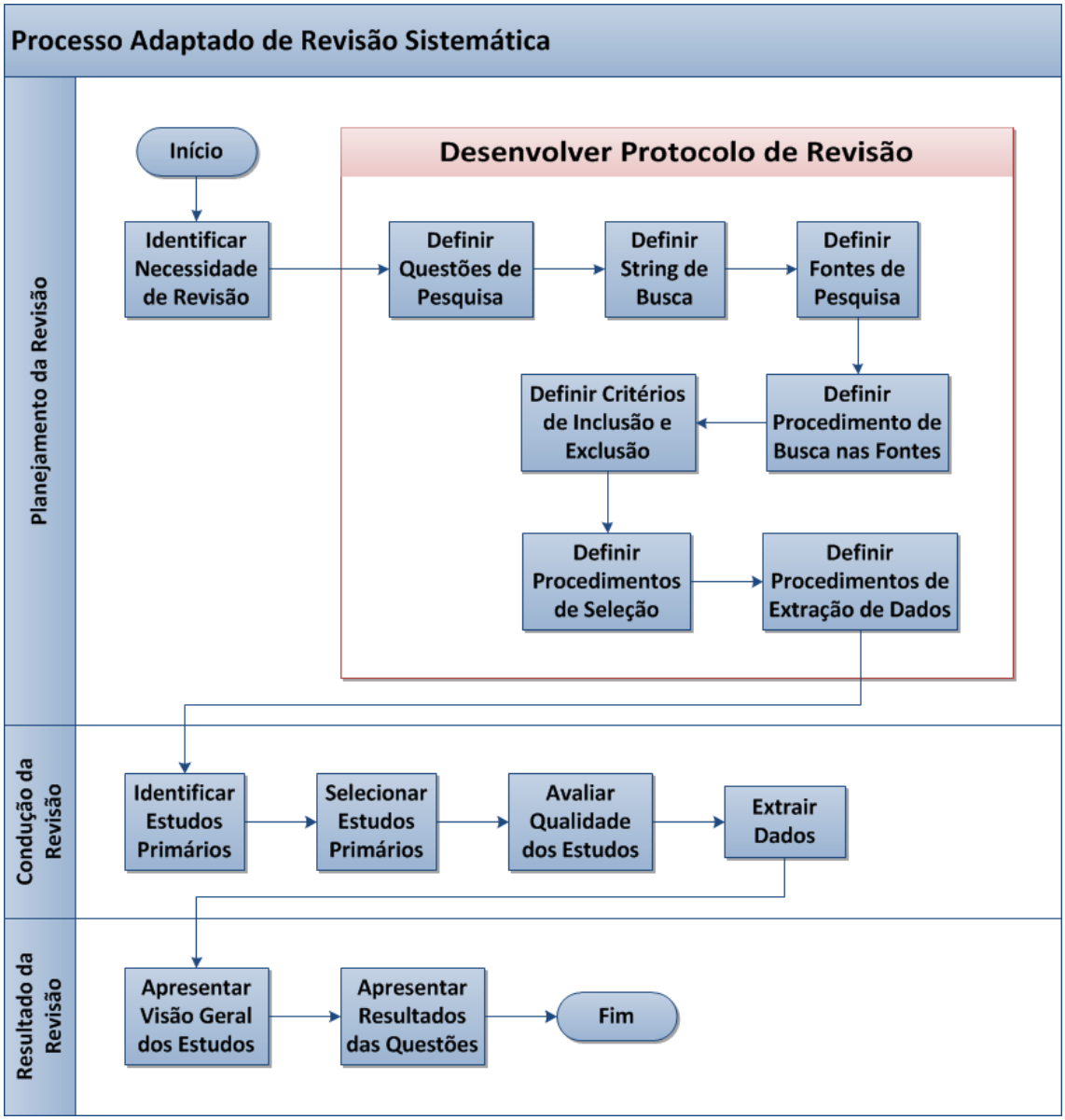


Figura 3.1: Fluxo do protocolo da revisão. Fonte: Adaptado de Kitchenham (2004).

Elasticidade é comumente citada como uma das vantagens da Computação em Nuvem, porém não é fácil encontrar trabalhos que se aprofundem no assunto.

- **Desenvolver Protocolo da Revisão**

  O protocolo de revisão está descrito a seguir em sete subatividades:

  (i) Definir Questões de Pesquisa

  Após a leitura dos artigos selecionados, esta revisão sistemática teve como objetivo responder às seguintes questões:

- Questão Principal: Qual o estado da arte sobre elasticidade em ambientes de Computação em Nuvem?

- Questão Secundária 1 (QS1): Como está sendo realizada análise de desempenho para elasticidade em ambientes de Computação em Nuvem?
- Questão Secundária 2 (QS2): Quais ferramentas, *benchmarks* e cargas de trabalho são utilizados para avaliar a elasticidade de ambientes de Computação em Nuvem?
- Questão Secundária 3 (QS3): Quais métricas são mais utilizadas para avaliar a elasticidade de ambientes de Computação em Nuvem?
- Questão Secundária 4 (QS4): Quais são as tendências de pesquisa em Computação em Nuvem do ponto de vista de elasticidade?
- Questão Secundária 5 (QS5): Quais são as estratégias adotadas no provimento da elasticidade em Computação em Nuvem?

(ii) Definir *String* de Busca

Em todas as consultas foram utilizadas as seguintes palavras chave: *"cloud computing"*, *elasticity*, *"performance analysis"*, *"performance evaluation"*, *metric*, *benchmark* e *tool*. Após alguns refinamentos, a seguinte *string* de busca foi gerada: ("*cloud computing*") AND ("*elasticity*") AND ("*performance analysis*" OR "*performance evaluation*") AND ("*metric*" OR "*benchmark*" OR "*tool*").

(iii) Definir Fontes de Pesquisa

Para esta revisão, os artigos foram pesquisados nos indexadores Science Direct[1], ACM Digital Library[2] e IEEExplore[3].

(iv) Definir Procedimento de Busca nas Fontes

A mesma *string* de busca foi utilizada nos três indexadores por meio do mecanismo de busca avançada existente em cada um.

(v) Definir Critérios de Inclusão e Exclusão

Algumas restrições foram utilizadas para limitar a busca. Foram pesquisados trabalhos do tipo periódico ou conferência e os trabalhos deveriam ter sido publicados entre os anos de 2009 a 2013. A palavra chave "*elasticity*" deve constar no trabalho, já que todos os trabalhos estão em inglês. Novamente, os trabalhos selecionados e não selecionados são verificados para se garantir que nenhum trabalho tenha sido incluído ou excluído erroneamente.

(vi) Definir Procedimento de Seleção

Etapa 1: A estratégia de busca é aplicada nas fontes.

Etapa 2: Para selecionar um conjunto inicial de estudos, os títulos e resumos de todos os artigos obtidos foram lidos e confrontados com os critérios de inclusão e exclusão.

[1] ScienceDirect - http://www.sciencedirect.com/
[2] ACM Digital Library - http://dl.acm.org/
[3] IEEEXplore - http://ieeexplore.ieee.org

Etapa 3: Todos os artigos selecionados na etapa 2 foram lidos por completo e novamente confrontados com os critérios do item (v). Os artigos incluídos são documentados e encaminhados para extração dos dados.

(vii) Definir Procedimento de Extração dos Dados

A extração das informações dos artigos foi realizada com base em um formulário com perguntas direcionadas a obter respostas para as questões de pesquisa da revisão. O formulário consistia de uma planilha com os seguintes itens a serem preenchidos para cada trabalho lido: título, ano de publicação, veículo de publicação, autores, país, grupo, palavras chave, proposta, observações, tipo de análise, métricas, métricas para elasticidade, carga de trabalho, ferramentas, trabalhos futuros e reprodução.

3.1.2 Atividade 2: Condução da Revisão

A condução da revisão consistiu de quatro subatividades, descritas a seguir:

- **Identificar Estudos Primários**

A coleta de informações desta revisão sistemática ocorreu no mês de julho de 2012. A execução das demais atividades ocorreu no segundo semestre de 2012 e atualizada em maio de 2013. Como resultado obtido, na Science Direct foram encontrados 42 trabalhos, na ACM Digital Library 67 e no o IEEExplorer 173 trabalhos, totalizando 282 trabalhos.

- **Selecionar Estudos Primários**

Para um refinamento dos resultados, como critério de inclusão os resumos dos 282 trabalhos foram lidos para que fosse confirmado se eles realmente estavam alinhados com o tema, e se de alguma forma o termo elasticidade era utilizado no trabalho, restando 82 trabalhos divididos em 19 do Science Direct, 21 da ACM Digital Library e 42 do IEEExplorer.

- **Avaliar Qualidade dos Estudos**

A avaliação da qualidade dos artigos dos estudos primários foi feita de forma simplificada verificando apenas a presença ou não de alguma forma de aplicação do conceito elasticidade em Computação em Nuvem, ou métricas e ferramentas específicas para elasticidade.

- **Extrair Dados**

Uma vez definido o conjunto dos trabalhos selecionados para leitura completa, efetuou-se o processo de extração dos dados conforme planejado no item de procedimento de extração de dados. Para isto uma planilha para cada artigo foi preenchida com as informações definidas no item "Definir Procedimento de Extração dos Dados". Esta atividade durou aproximadamente quatro meses para ser concluída.

3.1.3 Atividade 3: Resultado da Revisão

Consistiu de duas atividades, onde são apresentados os resultados da revisão de maneira geral e para cada questão planejada.

- **Apresentar Visão Geral dos Estudos**

Apresenta uma visão geral dos resultados obtidos pela revisão. Estes resultados mostram informações gerais não diretamente associada à elasticidade, mas servem para contextualizar a época da realização da revisão.

- **Apresentar Resultados das Questões**

Apresenta o resultado das questões de pesquisa (questões principal e secundárias).

3.2 Visão Geral dos Estudos

Após a leitura dos artigos selecionados, os resultados foram consolidados conforme o planejamento. Algumas informações foram identificadas antes da consolidação das questões pesquisadas, possibilitando uma visão geral sobre os estudos de elasticidade em Computação em Nuvem.

Segundo o *National Institute of Standards and Technology* (NIST), elasticidade é a capacidade de rápido provisionamento e desprovisionamento, com capacidade de recursos virtuais praticamente infinita e quantidade adquirível sem restrição a qualquer momento (MELL; GRANCE, 2009). Herbst, Kounev e Reussner (2013) propuseram uma definição mais atual para a elasticidade, definida como o grau no qual um sistema é capaz de se adaptar à variações na carga de trabalho pelo provisionamento e desprovisionamento de recursos de maneira autonômica, de modo que em cada ponto no tempo os recursos disponíveis combinem com a demanda da carga de trabalho o mais próximo possível. Nesta revisão sistemática diversas definições de elasticidade foram identificadas, descritas na Tabela 3.1.

A Tabela 3.2 exibe a quantidade de publicações para cada ano, considerando tanto periódicos quanto conferências. O ano de publicação considerado foi o que estava registrado no próprio trabalho como data de publicação. O período considerado foram os anos de 2009 a 2013. No momento da obtenção dos artigos, alguns trabalhos ainda estavam na situação de "*in press*". É provável que ao longo deste trabalho estes artigos já tenham sido publicados.

Percebe-se que houve um acréscimo nas publicações em 2011, alcançando o máximo em 2012 com 37. Talvez este aumento deveu-se ao fato que muitos grupos de trabalho obtiveram resultados de suas pesquisas em Computação em Nuvem neste período de tempo. Em 2013 foram identificados apenas 11 trabalhos publicados, mas este número deve aumentar devido à publicação de artigos em desenvolvimento. Além disso, considerando que os artigos foram selecionados em maio de 2013, provavelmente existirão mais trabalhos ao longo do ano.

Tabela 3.1: Definições de elasticidade identificadas durante a revisão

Autor	**Definição de Elasticidade**
Cooper et al. (2010)	Capacidade de adicionar novas instâncias e distribuir a carga de trabalho para estas instâncias.
Fito, Goiri e Guitart (2010)	Habilidade de adquirir e liberar recursos de granularidades variadas de acordo com a carga da trabalho em um curto intervalo de tempo.
Aisopos, Tserpes e Varvarigou (2011)	Capacidade do provedor alterar dinamicamente a quantidade de recursos de CPU, memória e espaço em disco para uma determinada tarefa.
Espadas et al. (2011)	Habilidade de criar um número variável de instâncias de máquinas virtuais que dependem da demanda da aplicação.
Li et al. (2011)	Habilidade do sistema de se adaptar à mudanças repentinas na carga de trabalho.
Perez-Sorrosal et al. (2011)	Capacidade de aumentar e diminuir a quantidade de réplicas sem interromper o processamento em andamento.
Garg, Versteeg e Buyya (2011, 2012)	Capacidade de um serviço escalar durante períodos de pico caracterizada pelo tempo médio para expandir ou contrair a capacidade do serviço e capacidade máxima do serviço.
Han et al. (2012)	Habilidade de escalar recursos de maneira adaptável para cima e para baixo para atender à variação da demanda das aplicações.
Islam et al. (2012)	Capacidade de provisionar recursos automaticamente e rapidamente.
Pandey et al. (2012)	Habilidade de um sistema expandir e contrair sem problemas.

Tabela 3.2: Quantidade de trabalhos publicados por ano de publicação

Ano	**Quantidade**
2009	2
2010	5
2011	23
2012	37
2013	11
in press até o momento da coleta	4

A consulta foi refeita em julho de 2014, resultando em um acréscimo natural de trabalhos. Porém não foram identificados trabalhos para a definição de métricas específicas para a medição da elasticidade, nem metodologias para sua avaliação.

A maioria dos trabalhos foram originados do periódico *Future Generation Computer Systems* (FGCS)[4], com 13 publicações. Estas publicações são recentes, tendo em vista que a maioria de seus trabalhos são de 2012 (4) e 2013 (6), os quais no momento da seleção, 7 estavam no estado de *in press*. Muitas conferências estão começando a publicar trabalhos em nuvem nos últimos anos e a tendência é que apareçam mais conferências específicas. Os veículos de publicação retornados pela pesquisa foram bem diversificados. A exceção do *Future Generation Computer Systems*, os demais veículos obtiveram uma, duas ou três publicações.

A contabilização dos países foi realizada da seguinte forma: para cada país de cada grupo de pesquisa em uma publicação, incrementou-se a quantidade de publicação do país. Portanto, alguns artigos contabilizaram mais de um país. A Figura 3.2 apresenta as publicações por país. Os países que mais se destacaram nas publicações relacionadas à elasticidade em

[4] *Future Generation Computer Systems* (FGCS) - http://www.journals.elsevier.com/future-generation-computer-systems/

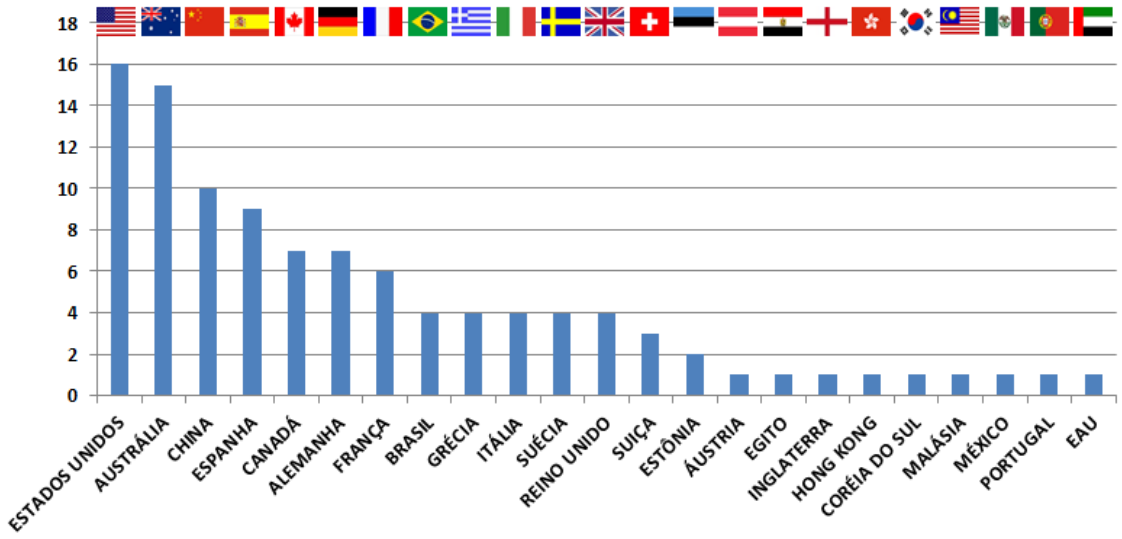


Figura 3.2: Publicações sobre elasticidade em Computação em Nuvem de 2009 a 2013 por país.

Computação em Nuvem foram: Estados Unidos (16), Austrália (15), China (10) e Espanha (9). Apesar dos Estados Unidos possuírem mais publicações, estas foram de diferentes grupos de pesquisas e associadas a empresas.

Ao todo foram identificados 108 grupos de pesquisas distribuídos entre 24 países. Os grupos que apresentaram mais publicações em Computação em Nuvem relacionados à elasticidade foram *Cloud Computing and Distributed Systems (CLOUDS) Laboratory*[5] e *Dept. Arquitectura de Computadores y Automática*[6], que possuem muitas publicações devido a atuação em diversas áreas de pesquisa em Computação em Nuvem. Coincidentemente os autores com o maior número de publicações são de grupos consolidados de pesquisa em Computação em Nuvem. Estes autores são de grupos da Austrália e Espanha, e seus trabalhos não são restritos apenas à elasticidade.

3.3 Resultados das Questões de Pesquisa

O resultado e análise das questões de pesquisa estão descritos a seguir:

Questão Principal: Qual o estado da arte sobre elasticidade em ambientes de Computação em Nuvem?

Em todos os trabalhos os termos elasticidade e análise de desempenho ocorreram. Alguns trabalhos desenvolveram experimentos especificamente para a elasticidade, enquanto que a maioria utilizou recursos de ambientes para a elasticidade ou propuseram soluções elásticas. De maneira geral, a maioria dos trabalhos em Computação em Nuvem relacionados à elasticidade englobaram os seguintes temas:

- Arquiteturas para Computação em Nuvem;

[5] *Cloud Computing and Distributed Systems (CLOUDS) Laboratory - http://www.cloudbus.org*
[6] *Dept. Arquitectura de Computadores y Automática* - http://dsa-research.org

- Estratégias para alocação de recursos;
- Ferramentas para propósito geral;
- Definição da tarifação de serviços na nuvem;
- Aplicações de alto desempenho e *web*;
- Ambientes diversificados (predominantemente Amazon).

De maneira mais específica, algumas categorias de áreas de pesquisa foram identificadas nos trabalhos:

- Elasticidade: Métricas para medir a elasticidade foram propostas por Islam et al. (2012), Li et al. (2011) e Calheiros et al. (2012). Para avaliar a elasticidade, os experimentos realizados utilizaram cargas de trabalho variáveis (TIRADO et al., 2011);
- Alocação de recursos: Estratégias e políticas para a alocação de recursos foram propostas, variando desde modelos matemáticos, análise dos recursos disponíveis, modelos multi-inquilinos e localização geográfica, dentre outros (ESPADAS et al., 2011; PANIAGUA; SRIRAMA; FLORES, 2011; BRYANT et al., 2011; LI; TORDSSON; ELMROTH, 2011; OTTO; STANOJEVIC; LAOUTARIS, 2012; CALHEIROS; RANJAN; BUYYA, 2011). Geralmente métricas estavam associadas às políticas, seja para análise dos recursos, seja para tomada de decisão para a alocação;
- SLA: Os trabalhos pesquisados utilizaram a elasticidade para garantir QoS. Alguns *frameworks* foram identificados para avaliação do SLA: YCSB (COOPER et al., 2010), OPTIMIS (FERRER et al., 2012) e SMICloud (GARG; VERSTEEG; BUYYA, 2011, 2012). Também foram estudados seus efeitos diante de variações em cargas de trabalho distintas para analisar os efeitos sobre a manutenção do SLA;
- Tarifação: As pesquisas em tarifação de serviços na nuvem aparecem ainda de maneira muito variada, geralmente propostas de tarifação baseadas na utilização de recursos (HAN et al., 2012). Um ponto a ser considerado é a minimização dos custos entre os modelos de serviços de provedores (HE et al., 2010; HONG; XUE; THOTTETHODI, 2012);
- *High Performance Computing* (HPC): A Computação de Alto Desempenho surgiu em vários trabalhos, tanto do ponto de vista de simplesmente executar aplicações HPC na nuvem quanto ao desenvolvimento de metodologias (MAUCH; KUNZE; HILLENBRAND, 2012; LU; JACKSON; BARGA, 2010; ZHAI et al., 2011). Raveendran, Bicer e Agrawal (2011) utilizaram aplicações *Message Passing Interface* (MPI) elásticas para um *framework* de nuvem. Moreno-Vozmediano, Montero e Llorente (2011) utilizaram o acesso ubíquo de recursos na nuvem para implantar um *cluster* no topo de uma infraestrututra de nuvem para resolver aplicações do tipo *Many-Task Computing* (MTC);
- Computação Autonômica: Muitos trabalhos indiretamente utilizaram recursos ou princípios de Computação Autonômica, tentando reduzir a intervenção humana (ETCHEVERS

et al., 2011; GHANBARI et al., 2011; NIU et al., 2012; PANDEY et al., 2012). A utilização de agentes para coleta de informações, principalmente métricas associadas a algum recurso computacional do ambiente, para posterior tomada de decisão baseada em regras, foi uma técnica muitas vezes utilizada para promover elasticidade;

- Computação Móvel: A nuvem pode ser utilizada para melhorar o desempenho de aplicações em dispositivos móveis através do processamento e armazenamento de maneira elástica na nuvem (NIEHORSTER et al., 2011). A *Internet of Things* (IoT) deve aumentar a interação entre a nuvem e os diversos dispositivos móveis (PANIAGUA; SRIRAMA; FLORES, 2011; FLORES; SRIRAMA; PANIAGUA, 2011);

- Análise de Desempenho: Avaliações de desempenho orientadas a cargas de trabalho em nuvens privadas e nuvens públicas foram alvos de algumas pesquisas em (GAO et al., 2011; TUDORAN et al., 2012);

- Arquitetura: Abordagens para segurança (LI et al., 2012) e para o gerenciamento de máquinas virtuais (MONTERO; MORENO-VOZMEDIANO; LLORENTE, 2011) foram propostas. Muitos trabalhos utilizaram aspectos de integração entre ambientes, como GRID e nuvens. Alguns *frameworks* e arquiteturas foram propostos para gerenciar a alocação de recursos utilizando balanceadores de carga e *brokers* (TORDSSON et al., 2012). Dawoud, Takouna e Meinel (2011) propuseram uma arquitetura de escalonamento para máquinas virtuais elásticas. Etchevers et al. (2011) utilizaram um modelo arquitetural de aplicação e protocolo de auto configuração que automatiza a distribuição de aplicações distribuídas legadas. Lucas-Simarro et al. (2012) propuseram uma arquitetura baseada em diferentes estratégias de escalonamento através de várias nuvens utilizando critérios de otimização, restrições do usuário e do ambiente.

3.3.1 Questão Secundária 1 (QS1): Como está sendo realizada análise de desempenho para elasticidade em ambientes de Computação em Nuvem?

Neste trabalho foram identificados três tipos de experimentos, conforme as definições de Jain (1991): experimentação, modelagem analítica e simulação. Além disso, esses tipos podem ocorrer de maneira combinada. A Tabela 3.3 descreve o quantitativo dos tipos de experimentos por publicação. Dos 82 artigos pesquisados, 6 eram do tipo *survey* e não possuiam nenhum experimento. Estes trabalhos não estão inclusos na Tabela 3.3.

A maioria dos trabalhos faz algum tipo de experimentação, seja experimentação simples ou combinada com outra técnica. É comum que os experimentos em elasticidade em Computação em Nuvem ocorram na Amazon EC2. Diversos artigos fizeram a experimentação combinando tipos de experimentos. Isso implica em uma qualidade maior dos trabalhos, pois eles tornam-se mais completos e é possível a comparação entre trabalhos iguais mas com visões diferentes. Os trabalhos utilizaram o CloudSim (CALHEIROS et al., 2011) ou traços computacionais (*traces*) para realizar simulações.

Apesar dos experimentos serem bem variados, existem deficiências no projeto dos experimentos. Os objetivos, serviços e resultados dos experimentos são informados. Contudo,

Tabela 3.3: Tipos de experimentos identificados nas publicações

Tipo de Experimento	SD	ACM	IEEE	Total por Tipo de Experimento
Experimentação	7	11	15	33
Modelagem Analítica	0	2	3	5
Simulação	0	0	7	7
Experimentação e Modelagem Analítica	8	2	9	19
Simulação e Modelagem Analítica	2	3	5	10
Experimentação e Simulação	0	1	1	2
Total	17	19	40	76

o planejamento não é detalhado nas publicações. Em alguns trabalhos, a justificativa para a seleção das métricas, dos parâmetros, e ferramentas utilizadas na configuração dos experimentos não são informadas. Em relação aos resultados, muitas vezes estes são apenas dispostos em tabelas ou gráficos e os comentários relacionados são superficiais.

O projeto experimental utilizado nos experimentos está diretamente associado à carga de trabalho ou *benchmark* utilizados, o que significa que se não possuírem algum mecanismo de configuração da carga, o experimento fica muito dependente das ferramentas. Em geral os experimentos dos trabalhos identificados utilizaram o SPECweb2005 (WEE; LIU, 2010; PEREZ-SORROSAL et al., 2011; BRYANT et al., 2011) ou o TPC-W (KOSSMANN; KRASKA; LOESING, 2010; ISLAM et al., 2012; HAN et al., 2012), que permitem configurações das cargas de trabalho.

A maneira na qual os experimentos foram conduzidos requer um projeto mais detalhado, pois está diretamente associado ao esforço e custo do experimento. A análise e interpretação dos dados em vários trabalhos foi bastante completa, porém em alguns casos foi meramente descritiva e geral, o que pode deixar conclusões em aberto. Todos esses fatores prejudicam a reprodução dos experimentos e consequente comparação entre trabalhos.

Por outro lado, a quantidade de 31 trabalhos com técnicas de avaliação combinadas proporciona uma qualidade maior dos trabalhos. Essa combinação de técnicas de avaliação permite que se possa ter uma visão teórica, muitas vezes do ponto de vista matemático, e prática dos experimentos.

3.3.2 Questão Secundária 2 (QS2): Quais ferramentas, *benchmarks* e cargas de trabalho são utilizados para avaliar a elasticidade de ambientes de Computação em Nuvem?

Para facilitar a análise dos resultados desta questão, os seguintes grupos foram considerados: infraestrutura, *benchmarks*, cargas de trabalho, aplicações, traços computacionais, simulação, *solvers*, servidores *web*, banco de dados e linguagens de programação.

Para a construção e configuração de um ambiente de Computação em Nuvem são necessárias várias ferramentas. Consideramos neste trabalho, apenas para fim de categorização, ambientes e seus diversos produtos/serviços/ferramentas como ferramentas.

Algumas ferramentas são utilizadas para a construção de um ambiente de nuvem, ou provisão de serviços na nuvem. Exemplos de ambientes identificados foram Amazon EC2,

Tabela 3.4: Principais ferramentas identificadas para provisionamento de serviços na nuvem

Descrição	Quantidade	Descrição	Quantidade
Amazon EC2	27	Google APP ENGINE	3
Amazon S3	7	Amazon EBS	2
OpenNebula	6	Amazon Elastic Load Balancer	2
Amazon RDS	4	Amazon SIMPLE DB	2
ElasticHosts	4	FlexiScale	2
Microsoft Azure	4	Ganglia	2
Amazon AutoScaling	3	IBM Cloud	2
Amazon CloudWatch	3	Nimbus	2
Aneka	3	Rackspace	2
Eucalyptus	3	Rightscale	2
GoGrid	3	Scicloud	2

OpenNebula, Microsoft Azure, Eucalyptus, Aneka, ElasticHosts, Google App Engine e Rightscale. Todos esses ambientes foram identificados em pelo menos dois trabalhos na pesquisa. A Tabela 3.4 exibe os ambientes e ferramentas que mais se destacaram nesta pesquisa.

Especificamente algumas ferramentas possuem produtos que têm sido largamente utilizados atualmente, não somente para serviços de instâncias. Amazon EC2 foi o ambiente mais utilizado, e possui vários serviços de diversas áreas: instâncias (EC2), escalonamento (AutoScaling), monitoramento (CloudWatch), banco de dados (S3, MySQL, EBS, RDS, SimpleDB), e balanceamento de carga (Elastic Load Balancing), entre outros. RackSpace possui serviços de instâncias e dados. Microsoft Azure também surge como uma opção de provedor, com alguns estudos que utilizaram instâncias, banco de dados e monitoramento.

O Amazon CloudWatch[7] oferece monitoramento de recursos em nuvem. Tanto desenvolvedores quanto administradores do sistema podem utilizá-lo para coletar e monitorar métricas, que podem ser utilizadas para reagir imediatamente à situações e manter serviços funcionando de acordo com o SLA. Assim é possível obter uma visibilidade geral da utilização de recursos e do desempenho dos serviços e aplicativos. O Auto Scaling[8] é um serviço que permite escalar a capacidade da instância para mais ou para menos de forma automática. Média de utilização de CPU, rede e utilização de disco são exemplos de métricas utilizadas para escalar um serviço. Assim, o número de instâncias utilizadas aumenta facilmente durante picos de demanda para manter o desempenho, e diminui automaticamente durante a redução da demanda para minimizar custos. O Auto Scaling é especialmente útil para aplicativos que experimentam variabilidade de uso por hora, dia ou semana, e é ativado pelo Amazon CloudWatch. O Elastic Load Balancing[9] distribui automaticamente o tráfego de entrada dos aplicativos em várias instâncias do Amazon EC2. Ele permite que se atinja uma maior tolerância a falhas nos serviços, fornecendo uma capacidade de equilíbrio entre a carga de trabalho e a resposta ao tráfego de entrada dos aplicativos. Ele detecta instâncias com problemas de integridade dentro de um conjunto e redireciona automaticamente o tráfego para instâncias íntegras até que as instâncias com problemas sejam restauradas. Além disso é possível utilizar o Elastic Load Balancing entre

[7] Amazon CloudWatch - http://aws.amazon.com/pt/cloudwatch/

[8] Auto Scaling - http://aws.amazon.com/pt/autoscaling/

[9] Elastic Load Balancing - http://aws.amazon.com/pt/elasticloadbalancing/

Tabela 3.5: *Benchmarks* e cargas de trabalho identificadas

Ferramenta	**Quantidade**
TPC-W	9
SPECweb2005	8
YCSB	6
Blast, Cloudstone, Embarrassingly Distributed (ED), NAS Grid Benchmarks (NGB), NAS Parallel Benchmarks (NPB)	4
BT-IO, Httperf, Intel MPI Bench, IOR, JMeter, TPC-C	3
ALS, Badabing, Bonnie, Bonnie++, Hdparm Tool, HPCC, Iperf, Linpack, LMBench, Lublin99, PostMark, SPECjvm 2008, TPC-E, TPC-H, UBENCH, Whetstone, WSTest	2

zonas de disponibilidade variadas, melhorando a disponibilidade.

Aplicações de *proxy* e balanceadores de carga também foram utilizadas: HAProxy (3), Amazon Elastic Load Balancer (2), Nginx (2), e Squid e Amoeba com somente uma ocorrência cada. A maioria dessas ferramentas foi utilizada para prover alguma solução autonômica, onde alguma estratégia de balanceamento de carga foi utilizada.

Trabalhos relacionados à análise de desempenho utilizaram *benchmarks* e cargas de trabalho nos experimentos. Muitos *benchmarks* e cargas de trabalho foram identificados na revisão, descritos na Tabela 3.5. Entre eles destacaram-se o TPC-W (HAN et al., 2012; KOSSMANN; KRASKA; LOESING, 2010; ISLAM et al., 2012) e SPECweb2005 (FITO; GOIRI; GUITART, 2010; WEE; LIU, 2010; PEREZ-SORROSAL et al., 2011; BRYANT et al., 2011).

YCSB é um *framework* para facilitar comparações de desempenho de sistemas de serviços de dados na nuvem (COOPER et al., 2010). CloudStone auxilia avaliações de desempenho na nuvem, e implementa um modelo real de aplicações *web* a ser implantado em um IaaS (ACETO et al., 2013). Embarrassingly Distributed (ED) e NAS Grid Benchmarks (NGB) estavam entre os mais utilizados *benchmarks* (MORENO-VOZMEDIANO; MONTERO; LLORENTE, 2009, 2011; MONTERO; MORENO-VOZMEDIANO; LLORENTE, 2011; TORDSSON et al., 2012). NAS Parallel Benchmarks (NPB) são um pequeno conjunto de programas projetados para auxiliar na análise de desempenho de computadores paralelos. Particularmente Embarrassingly Distributed (ED) and BT-IO são parte do NPB, mas eles são tão aplicados que justificou sua utilização separada.

Message Passing Interface (MPI) foi destacado em Zhai et al. (2011). O MPI é um padrão de comunicação de dados em computação paralela e sua aplicação em conjunto com Computação em Nuvem foi identificada, não só relacionado à elasticidade, mas à sua utilização em nuvens computacionais. Alguns *benchmarks* para MPI foram identificados na pesquisa, como o Intel MPI Benchmark e MPI-Blast, ambos em Zhai et al. (2011).

Algumas dessas ferramentas foram utilizadas para testes de carga em aplicações, mostrando-se eficientes para experimentos com cargas de trabalho em ambientes de Computação em Nuvem: Httperf (FITO; GOIRI; GUITART, 2010; BRYANT et al., 2011; LUCAS-SIMARRO et al., 2012), JMeter (ESPADAS et al., 2011; ISLAM et al., 2012) e Tsung (FLORES; SRIRAMA; PANIAGUA, 2011).

Microbenchmarks são alternativas para *benchmarks* e cargas de trabalho. Eles são

benchmarks simplificados. Geralmente são programas escritos e executados para a geração de cargas de trabalho e coleta de métricas. Os *microbenchmarks* foram bem variados nas áreas de atuação: *benchmarks* sintéticos (TUDORAN et al., 2012), *microbenchmarks* (ZHAI et al., 2011), problemas de ordenação (LUCAS-SIMARRO et al., 2012), renderização (AISOPOS; TSERPES; VARVARIGOU, 2011), teste de carga (NIEHORSTER et al., 2011), *Bag-of-Tasks* (BoT) e GRID (CALHEIROS; RANJAN; BUYYA, 2011), cargas de trabalho simuladas (LI et al., 2012) e cargas de trabalho sintéticas (NIE; XU, 2009).

Uma forma de analisar o comportamento de um ambiente é por meio da execução de aplicações. Tais aplicações eram das mais variadas áreas: computação gráfica (EMEAKAROHA et al., 2012), redes sociais (OTTO; STANOJEVIC; LAOUTARIS, 2012), aplicações *web* (LUCAS-SIMARRO et al., 2012) e funções matemáticas (GAO et al., 2011). A maioria das aplicações apareceu apenas uma vez na pesquisa. As aplicações que obtiveram mais de uma ocorrência na pesquisa foram: A-Brain (RAVEENDRAN; BICER; AGRAWAL, 2011; TUDORAN et al., 2012), GASOLINE (KREBS; MOMM; KOUNEV, 2012; LI et al., 2013), NAMD (LI et al., 2012; WONG; GOSCINSKI, 2012), WCD (LI et al., 2013, 2012) e Zookeeper (CRUZ et al., 2013; DAS; AGRAWAL; ABBADI, 2013).

Traços computacionais são dados coletados de aplicações ou sistemas. A utilização de traços em Computação em Nuvem permite que diferentes aspectos do ambiente possam ser analisados, como serviços, disponibilidade do ambiente, utilização dos recursos, estudo de padrões e escalabilidade das aplicações. Nesta pesquisa, os traços foram utilizados em vários experimentos. Muitas vezes foram extrações de dados de sistemas, como *logs* de aplicação. Algumas aplicações utilizaram traços para experimentos na simulação de algum aspecto do ambinte. A maioria dos traços apareceram apenas uma única vez neste estudo. Os traços com maior ocorrência foram: Grid Workloads Archive (AuverGrid), Wikipedia, Fifa 98 e Google. A Tabela 3.6 descreve os traços computacionais identificados, nas mais diversas áreas: provedores de Computação em Nuvem, multimídia, educação, infraestrutura, *web sites* e redes sociais.

Considerando as ferramentas para a simulação de ambientes de Computação em Nuvem, destacam-se o CloudSim (CALHEIROS et al., 2011) e o D-Cloud (BANZAI et al., 2010). Outras ferramentas identificadas foram Alea e GridSim.

Alguns estudos fizeram uso de modelagem analítica e utilizaram *solvers* para a resolução de problemas matemáticos. A ferramenta mais utilizada foi o AMPL (LI; TORDSSON; ELMROTH, 2011; LUCAS-SIMARRO et al., 2012; TORDSSON et al., 2012), com 3 ocorrências, seguida pelo CPLEX (TORDSSON et al., 2012), Gurobi (LI; TORDSSON; ELMROTH, 2011), LP Solver (DENG et al., 2012), Matlab (KOUSIOURIS et al., 2012), MINOS (LUCAS-SIMARRO et al., 2012), MKL Library (MAUCH; KUNZE; HILLENBRAND, 2012), Octave (KOUSIOURIS et al., 2012), Optimj (PANIAGUA; SRIRAMA; FLORES, 2011) e SSJ (KREBS; MOMM; KOUNEV, 2012), todos com uma única ocorrência.

Alguns trabalhos utilizaram servidores *web*. O Apache Web Server e Apache Tomcat foram os mais utilizados, com 8 e 7 ocorrências respectivamente. As demais ferramentas, em ordem de ocorrência, foram: Jonas (2), Glassfish (1) e JBOSS (1).

Banco de dados tipicamente são utilizados para armazenamento de informações.

Tabela 3.6: Traços computacionais identificados

Descrição	Descrição
Amazon EC2 (GARG; VERSTEEG; BUYYA, 2011)	RIKEN Integrated Cluster of Clusters (RICC) Japan (XU; LI, 2013)
Clarknet (HONG; XUE; THOTTETHODI, 2012)	Simulação de Vírus (WONG; GOSCINSKI, 2012)
Fifa 98 (GHANBARI et al., 2011; ALI-ELDIN et al., 2012)	Taxa de Requisição Serviço da HP (VDR) (DENG et al., 2012)
Google (ALI-ELDIN et al., 2012; XU; LI, 2013)	Twitter (OTTO; STANOJEVIC; LAOUTARIS, 2012)
Grid Workloads Archive (NIE; XU, 2009; MICHON; GOSSA; GENAUD, 2012)	UC Berkeley (HONG; XUE; THOTTETHODI, 2012)
Intrepid Cluster Argonne National Laboratory (ANL) (XU; LI, 2013)	Vod UUSee (NIU et al., 2012)
Lastfm (TIRADO et al., 2011)	Website Urdue University's College of Engineering (HONG; XUE; THOTTETHODI, 2012)
NASA (HONG; XUE; THOTTETHODI, 2012)	Wikimedia Web Traces (HONG; XUE; THOTTETHODI, 2012)
Preço de Carbono (iPath Global Carbon) (DENG et al., 2012)	Wikipedia (CALHEIROS; RANJAN; BUYYA, 2011; FERRER et al., 2012; CASALICCHIO; SILVESTRI, 2013)
Rackspace (GARG; VERSTEEG; BUYYA, 2011)	Windows Azure (GARG; VERSTEEG; BUYYA, 2011)

Alguns *benchmarks* e geradores de carga de trabalho requerem um banco de dados para o povoamento de *log* de operações. A elasticidade em alguns casos era avaliada pela capacidade de processamento de consultas no banco de dados, geralmente submetida pelos *benchmarks* e geradores de carga de trabalho. Nesta categoria foram incluídos servidores de banco de dados e ferramentas para manipulação de dados. As ferramentas mais utilizadas foram: MySql (9), Hadoop (4), Cassandra (2) e HBase (2). Outras ferramentas identificadas com apenas uma ocorrência foram: JDBC, JoSQL, NoSQL e PostgreSQL.

Outra categoria identificada foram ferramentas relacionadas à linguagens de programação, como APIs, componentes e bibliotecas. A maioria dessas linguagens foi utilizada para a construção de *microbenchmarks* e *scripts* para apoiar tarefas como a coleta de informação e comunicação, não necessariamente diretamente relacionadas à elasticidade, mas a funcionalidades do sistema e suporte. Um exemplo é a linguagem Java, que foi a mais utilizada, com 8 ocorrências. Python obteve 3 ocorrências. Aplicações Android, servlets e *shell script* também foram identificadas. Outras linguagens de programação também foram utilizadas, mas com apenas uma ocorrência: c, c++ and php.

3.3.3 Questão Secundária 3 (QS3): Quais métricas são mais utilizadas para avaliar a elasticidade de ambientes de Computação em Nuvem?

Métricas são sempre utilizadas para analisar o desempenho. Existem diversos tipos e categorias de métricas, e muitas delas podem ser utilizadas para avaliar a elasticidade. Algumas métricas são de propósito geral, enquanto que outras são específicas para elasticidade.

Métricas Gerais

Muitas métricas de propósito geral foram identificadas nessa pesquisa. Estas métri-

cas são encontradas em diversas áreas relacionadas a redes de computadores e sistemas distribuídos, e não são limitadas somente às métricas identificadas neste trabalho.

Raj Jain definiu algumas categorias de métricas: tempo de resposta, *throughput*, utilização, disponibilidade, aquisição, confiabilidade e escalabilidade (JAIN, 1991). A maioria das métricas citadas foram, por categoria: tempo de resposta (latência, alocação, desalocação, acesso, ociosidade, resposta), *throughput* (requisições por segundo, Megabytes por segundo), utilização (recursos, percentagem de CPU), disponibilidade (período de tempo em que um sistema está indisponível e período de tempo em que um sistema está disponível), aquisição (custo, desempenho do custo), confiabilidade (quantidade de violações) e escalabilidade. Escalabilidade está também na classificação de Menasce, Dowdy e Almeida (2004). Nesta categoria as métricas mais citadas foram: demanda, SLA, *overhead*, capacidade total de infraestrutura, custo e energia.

As principais métricas foram orientadas a recursos, como CPU e memória. O percentual de utilização de CPU foi a métrica mais citada, mas utilização de mémória, acesso a disco e violações de QoS também foram identificadas. Métricas de desempenho também foram destacadas, como *throughput*, expresso em Megabytes por segundo.

Algumas métricas são compartilhadas entre grupos diferentes dependendo do contexto de utilização, por exemplo, tempo de alocação/desalocação de recursos pode estar relacionado à métrica de tempo de resposta e escalabilidade. Algumas técnicas estatísticas são muito utilizadas na exposição e interpretação dos resultados de experimentos. As mais encontradas foram média, desvio padrão, variância e erro de utilização de recursos e *throughput*.

O custo e a tarifação são aspectos que muitas vezes estão associados à nuvens públicas devido ao seu caráter de aquisição de recursos. A definição de uma forma de se precificar a utilização de recursos na nuvem ainda não é padronizada, e normalmente cada provedor de serviços possui a sua própria estratégia. Custos totais, custos da infraestrutura, custos por hora são exemplos identificados. A tarifação dos serviços de um provedor muitas vezes está associada à utilização de recursos, como CPU, memória e armazenamento.

O consumo de energia é um dos temas principais em ambientes de Computação em Nuvem. Nos trabalhos analisados foram identificadas as seguintes métricas para medir eficiência energética: *Data Center Infrastructure Efficiency* (DCiE), *Power Usage Efficiency* (PUE) e *Data Center Performance per Energy* (DPPE) (GARG; VERSTEEG; BUYYA, 2012; TUDORAN et al., 2012). DCiE é definido como o percentual da potência total do equipamento de TI (computadores, *storages* e rede). O PUE é inversamente proporcional ao DCiE. DPPE correlaciona o desempenho do *datacenter* com emissões de carbono. O *overhead* de energia foi analisado em um trabalho como uma maneira de se comparar experimentos, com a medição da potência e consumo de energia sendo realizada através de um dispositivo físico (LI et al., 2012).

Considerando a latência como a diferença entre o início de um evento e o início real do evento, onde seus efeitos se tornam perceptíveis, muitos eventos na nuvem utilizam esta métrica como uma maneira de se medir esse tempo. Nesta pesquisa foram identificadas latências relacionadas à migração de máquinas virtuais, atualização e leitura de dados. Aspectos de QoS relacionados à latência das aplicações foram identificados e quantificados, muitas vezes

Tabela 3.7: Métricas específicas para elasticidade

Alocação	*Computing Resource Allocation Meter* (CRAM), Alocação de Recursos
Capacidade	Fornecimento Disponível, Capacidade, Aumento da Capacidade, Capacidade de Computação, Capacidade de Serviço Máxima, Serviço Disponível, *System Capacity Meter* (SCM), *System Effective Capacity Meter* (SEC), *System Load Meter* (SLM), *Total Capacity of Infrastructure*
Custo	Custo / Taxa de Desempenho, Custo da Largura de Banda, Efetividade do Tempo e Custo (Horas $ / instâncias), Custo de Migração, Custo Total de Implantação, Preço Total da Infraestrutura
QoS	% de Violações, *Overhead* de Reconfiguração da Infraestrutura, Ganho de Desempenho, SLA, *System Performance Meter* (SPM)
Utilização de Recursos	Percentual de Violações, Estresse da Nuvem, *Computing Resource Utilization Meter* (CRUM), Demanda, Ociosidade, Aumento da Ociosidade, Número de Máquinas Virtuais Sobreprovisionadas, Número de Máquinas Virtuais Subprovisionadas, Número de Máquinas Virtuais, Taxa de Sobreprovisionamento, Sobreutilização, Subutilização, Utilidade Global, *Performance Resource Ratio* (PRR), Utilização Possível, Número Médio de Servidores
Escalabilidade	*Effective Scalable Range* (ESR), *Effective System Scalability* (ESS), Escalabilidade, *Scale-up*
Tempo	*Boot*, Criação, Deleção, Tempo Médio para Retração da Capacidade de Serviço, Tempo Médio para Expandir a Capacidade de Serviço, Provisionamento, Alocação de Recursos, Desalocação de Recursos, Incialização, Suspensão, Tempo de Reinicialização da Suspensão, Tempo/Recursos no Tempo, Aquisição Total, Liberação Total

associados a um SLA ser a mantido pela aplicação ou ambiente. Geralmente QoS está associado a regras de monitoramento. Por fim, métricas associadas ao *overhead* e confiabilidade foram também identificadas nesta revisão.

Métricas Específicas para Elasticidade

Em relação à medidas específicas para elasticidade, a pesquisa retornou diversas métricas descritas na Tabela 3.7. Muitas dessas métricas são de recursos do sistema associadas à elasticidade, ou seja, para medir ou promover a elasticidade. Algumas são específicas para a elasticidade, como é o caso de métricas relacionadas à sobreutilização e subutilização. Islam et al. (2012) definiram métricas de elasticidade baseadas em penalidades para a sobreutilização e a subutilização dos recursos. Gao et al. (2011) descreveram um conjunto de métricas (CRAM, CRUM, SPM, SLM, SCM, SEC, ESS e ESR) para analisar a escalabilidade e elasticidade em ambientes de SaaS. Estas métricas estão associadas à alocação de recursos, utilização dos recursos, desempenho do sistema, carga do sistema, capacidade efetiva do sistema, escalabilidade efetiva do sistema, e faixa efetiva de escalabilidade.

Garg, Versteeg e Buyya (2012) descreveram uma métrica para elasticidade como o quanto um serviço de nuvem pode ser escalado durante intervalos de pico. Esta métrica pode ser definida por dois atributos: tempo médio para expandir ou contrair a capacidade do serviço e a capacidade máxima do serviço (número máximo de unidades computacionais que podem ser providas em períodos de pico). Bai et al. (2011) descrevem a métrica *Performance Resource Ratio* (PRR), ou Taxa de Desempenho de Recursos (TDR), que reflete o relacionamento entre o desempenho e os recursos utilizados, baseada no tempo de espera, tempo de execução, e alocação dos recursos, como CPU, largura de banda e memória.

Não é fácil definir uma métrica para a medição da elasticidade. Em geral, muitas métricas comuns são utilizadas, como tempo de resposta, ou algum recurso do ambiente, combinados juntos para avaliar algum aspecto do ambiente. As métricas relacionadas à elasticidade foram agrupadas da seguinte maneira: alocação, capacidade, custo, QoS, utilização de recursos, escalabilidade e tempo.

Alocação de recursos possui aspectos e métricas que geralmente descrevem a alocação e liberação dos recursos, como CPU, memória e instâncias de máquinas virtuais no ambiente, e a consolidação dos recursos em servidores mais otimizados ou com menor consumo.

A capacidade de um ambiente pode ser medida por seus recursos e disponibilidade. Como a quantidade de recursos varia no tempo e eles são providos durante um intervalo, isso impacta diretamente nas cargas de trabalho aplicadas ao ambiente, que por si são dinâmicas.

Sempre há um custo associado à utilização dos recursos em algum mecanismo de elasticidade, podendo ser monetário ou operacional. Algumas métricas de custo foram identificadas: custo da utilização de recursos, custo da execução dos serviços, como balanceamento de carga e monitoração, custo por unidade de tempo para manutenção de um dado serviço com limiares pré-estabelecidos de qualidade.

Em relação a QoS, métricas que medem violações de alguma característica do sistema foram identificadas. A medição do SLA em si foi utilizada como uma maneira indireta para mensurar a elasticidade. Assim é possível avaliar se há uma ganho no desempenho da elasticidade. Em outras palavras, para ajustar o ambiente devido à elasticidade, é possível medir o *overhead* da reconfiguração da infraestrutura.

A categoria de utilização de recursos identificou métricas relacionadas ao uso de algum recurso ou característica do sistema. Tipicamente mede ociosidade, subutilização e sobreutilização de recursos. Também descreve a demanda do ambiente e se o nível de utilização é alto ou baixo. Percebe-se que recursos virtuais são utilizados para algum mecanismo de elasticidade. A quantidade de máquinas virtuais e seus estados com respeito à subutilização e sobreutilização também foi identificada.

Muitas vezes escalabilidade foi utilizada como uma estratégia para o provimento da elasticidade. Uma métrica específica para escalabilidade foi identificada, assim como sua faixa de utilização e escalabilidade do sistema (GAO et al., 2011).

Em geral o tempo requerido para alguma operação foi a métrica mais utilizada. Para elasticidade, muitas operações possuem o tempo medido no provimento de algum serviço. O tempo de alocação, desalocação, provimento e desprovimento de recursos foram utilizados para medir o desempenho da elasticidade. Durante esses períodos, o serviço pode estar inativo, o que não é desejável. Também é interessante que esses tempos sejam monitorados.

3.3.4 Questão Secundária 4 (QS4): Quais são as tendências de pesquisa em Computação em Nuvem do ponto de vista de elasticidade?

De maneira geral, muitos trabalhos foram identificados como tendências de pesquisa e lacunas em Computação em Nuvem relacionados à elasticidade, onde se destacam:

- *Benchmarks* para Computação em Nuvem: Na literatura existem muitos *benchmarks* para a geração de diversos testes e cargas de trabalho. Porém, *benchmarks* específicos para ambientes de Computação em Nuvem, onde se possa avaliar a elasticidade de um ambiente para simular cargas variadas com aumento e diminuição da carga ainda não são comuns. Um exemplo de *benchmark* utilizado para medir a elasticidade em um ambiente de Computação em Nuvem é o *framework* Yahoo! Cloud Serving Benchmark (YCSB) (COOPER et al., 2010), que permite comparar o desempenho de serviço de dados na nuvem;

- Métricas específicas para elasticidade: Apesar que alguns trabalhos citaram métricas e modelos específicos para a medição da elasticidade (GAO et al., 2011; GARG; VERSTEEG; BUYYA, 2012; ISLAM et al., 2012), ainda existem poucos trabalhos que realmente medem a elasticidade. É mais comum identificar trabalhos que utilizam ou provêm a elasticidade;

- Computação Autonômica em ambientes de nuvem: Os princípios de Computação Autonômica já são encontrados em trabalhos de elasticidade em nuvem, principalmente pela utilização de agentes para coleta de dados e definição de regras para a tomada de decisão (PANDEY et al., 2012; NIEHORSTER et al., 2011; GHANBARI et al., 2011). Além disso, em geral operações de auto escalonamento de recursos são baseadas em regras. Contudo, este ambiente não possui padronização. A automação do provisionamento de recursos nos provedores, de maneira que não seja necessária a intervenção humana e que se adpate às necessidades dos usuários, é outro aspecto a ser considerado;

- Testes em ambientes de Computação em Nuvem: Testes com o objetivo de verificar se o ambiente de Computação em Nuvem está apto a suportar cargas de trabalho variadas, alterações de cargas em curtos intervalos de tempo, simulação de usuários, o quanto um ambiente é elástico, o quanto ele suporta a elasticidade dos recursos mantendo um nível de serviço adequado ao usuário. Bai et al. (2011) aborda diversos aspectos de testes em ambientes de Computação em nuvem;

- Geração de cargas de trabalho em nuvem: Diversos geradores de carga de trabalho existem na literatura, assim como *benchmarks*. O estudo de mecanismos estatísticos, como distribuições estatísticas, para modelagem de cargas de trabalho, é uma atividade que suportaria o planejamento e acompanhamento da utilização de ambientes;

- Tarifação baseado em elasticidade: A tarifação de serviços na nuvem é alvo de muita pesquisa, e muitos trabalhos propõem estratégias de precificação. A padronização ainda é uma área a ser pesquisada, pois os provedores são variados, os tipos de serviços e modelos adotados são diferentes;

- Computação móvel: Com a facilidade de acesso através de dispositivos móveis faz-se necessário a integração entre os diversos dispositivos e a nuvem, responsável pelo processamento (FLORES; SRIRAMA; PANIAGUA, 2011; MORENO-VOZMEDIANO; MONTERO; LLORENTE, 2011). A elasticidade está no lado servidor das aplicações, ou seja, nos *datacenters*, onde mecanismos que provêm a elasticidade estão em execução;

- Aplicações orientadas ao consumo de energia: Com os *datacenters* investindo em mecanismos de redução do consumo energético, a Computação em Nuvem pode colaborar na migração de aplicações, que atendendo às necessidades dos clientes através da elasticidade, pode evitar o desperdício de uma infraestrutura ociosa ou mal distribuída. A "computação verde" pode ser utilizada junto a Computação em Nuvem. Estratégias de alocação de recursos baseadas no consumo energético podem regular e consolidar servidores, desativando os que não estão em utilização.

3.3.5 Questão Secundária 5 (QS5): Quais são as estratégias adotadas no provimento da elasticidade em Computação em Nuvem?

O gerenciamento de ambientes de Computação em Nuvem envolve a construção de estratégias para torná-lo elástico, garantir QoS e utilização dos recursos de maneira eficiente. Assim, existem diferentes soluções que implementam elasticidade. Essas soluções adicionam e removem recursos conforme políticas baseadas em cargas de trabalho. Neste trabalho, modelos e métodos para a implementação da elasticidade foram identificados, basedos nos mecanismos de escalabilidade apresentados por Vaquero, Rodero-Merino e Buyya (2011) e na classificação de elasticidade apresentada por Galante e Bona (2012). Uma visão geral desta classificação pode ser visualizada na Figura 3.3.

Os métodos identificados estão descritos conforme a seguir:

- Elasticidade Horizontal (Replicação): Consiste em adicionar/remover instâncias a partir

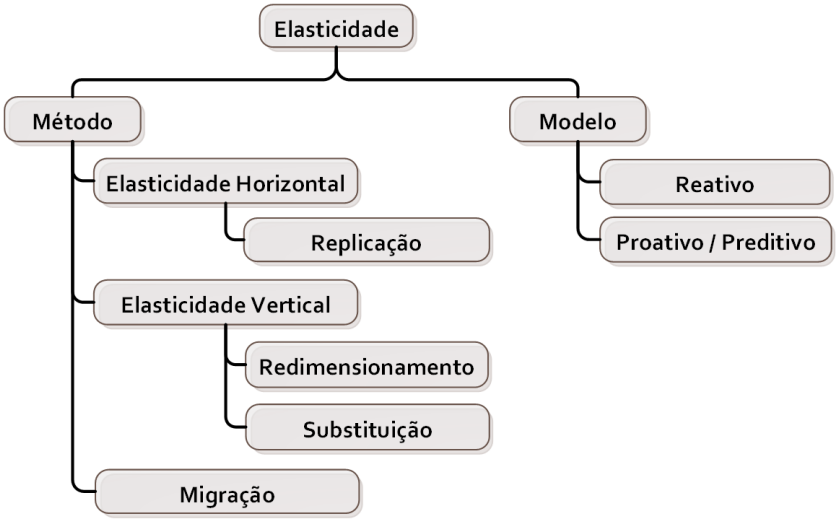


Figura 3.3: Classificação de soluções e estratégias de elasticidade

do ambiente virtual do usuário. Estas instâncias podem ser aplicações, *containers* ou máquinas virtuais. Por exemplo adicionar novos servidores de réplicas e balanceadores de carga para sua distribuição entre todas as réplicas disponíveis. Replicação é atualmente o método mais utilizado no provimento da elasticidade;

- Elasticidade Vertical: Consiste em adicionar/remover recursos. Por exemplo, adicionar capacidade de processamento, memória e armazenamento a partir de uma máquina virtual. Duas diferentes abordagens para executar escalabilidade vertical foram identificadas:

  - Redimensionamento: Consiste na alteração em tempo de execução dos recursos associados à instância em execução (por exemplo associar mais CPU e memória para uma máquina virtual em execução). Esta solução é comum em sistemas operacionais Unix, uma vez que eles suportam mudanças na CPU e memória em tempo real sem a necessidade de reinicialização (por exemplo a utilização de *ballooning* em memória);

  - Substituição: Consiste na adição de servidores mais potentes na substituição de servidores menos potentes, simulando a abordagem de redimensionamento. É comum em provedores públicos de Computação em Nuvem.

- Migração: Consiste na transferência de máquinas virtuais ou aplicações que estão sendo executadas de um servidor físico para outro. Máquinas virtuais podem ser migradas de um servidor para outro para consolidação ou balanceamento de carga.

Os modelos identificados estão descritos conforme a seguir:

- Reativo: Reage à carga de trabalho atual e utiliza limiares da utilização dos recursos ou violações de SLA para disparar a necessidade de capacidade adicional. O sistema reage à mudanças, mas não as antecipa;

- Proativo (preditivo): Utiliza técnicas de predição de cargas de trabalho para determinar quando a carga de trabalho futura irá superar a capacidade atual provisionada e invoca algoritmos de alocação adicional de servidores antes que sua capacidade seja excedida. Técnicas proativas geralmente baseiam suas decisões no histórico da carga de trabalho.

Diversas soluções elásticas foram identificadas neste trabalho. Os trabalhos identificados estão organizados conforme a classificação de elasticidade apresentada.

Replicação Reativa

Vários trabalhos utilizaram uma abordagem reativa na qual é dependente de monitoração. Os serviços de monitoração são mecanismos que permitem aos usuários monitorar e coletar métricas de desempenho do sistema em um baixo nível de granularidade. Tais métricas são utilizadas por um serviço de auto escalonamento para automaticamente adicionar ou remover máquinas virtuais.

Fito, Goiri e Guitart (2010) trabalharam a elasticidade em um ambiente de nuvem com servidores *web*. É apresentada uma solução de gerenciamento de recursos baseada em

SLA que reage às cargas dinâmicas recebidas com réplicas de servidores. Um conjunto de variáveis de economia foram definidas, além de funções para o estudo do rendimento adquirido pelo provedor e como ele pode maximizá-lo. Perez-Sorrosal et al. (2011) apresentaram uma estrutura de cache e replicação que facilita a implantação elástica e escalável de arquiteturas multi-camadas. Bryant et al. (2011) propuseram o Kaleidoscope, uma abordagem de replicação de máquinas virtuais associada à clonagem sob-demanda. Pawluk et al. (2012) introduziram o STRATOS, um serviço de *broker* para facilitar a comunicação entre nuvens, a construção de uma topologia de plataforma e em tempo de execução modificar os objetivos implementados. STRATOS habilita a instanciação inter provedores de réplicas pela resolução de um problema de otimização multi-critério. Em Sousa e Machado (2012), o RepliC é apresentado como uma abordagem para a replicação de banco de dados na nuvem com qualidade de serviço e suporte a multi-inquilinos. Sua estratégia adiciona e remove réplicas rapidamente conforme a carga de trabalho aplicada ao ambiente. Zhao, Sakr e Liu (2013) apresentaram um *framework* para uma camada de replicação de banco de dados baseada no SLA. Este *framework* continuamente monitora a aplicação e o SLA, e automaticamente dispara a execução de ações corretivas quando requerida.

Migração Reativa

Estratégias de migração de máquinas virtuais estão mais concentradas no atendimento do balanceamento de carga entre máquinas físicas sem considerar características específicas das aplicações que estão sendo executadas nas máquinas virtuais.

He, Guo e Guo (2011) propuseram uma solução de gerenciamento de recursos baseada na migração de máquinas virtuais para melhorar a utilização de recursos na nuvem. O desempenho da migração da máquina virtual foi analisado e uma infraestrutura de armazenamento compartilhado para lidar com o problema foi proposta. Mauch, Kunze e Hillenbrand (2012) descreveram técnicas de virtualização e métodos de gerenciamento, como multi-inquilino, provisionamento dinâmico e SLA. Também é apresentado uma abordagem que utiliza altas conexões entre *clusters*, como InfiniBand, em um ambiente de Computação em Nuvem de alto desempenho. Esta abordagem permite migrar rapidamente máquinas virtuais para garantir o desempenho da rede e QoS. Uma abordagem para a melhoria da qualidade do serviço para banco de dados relacionais multi-inquilinos foi proposta em Moreira et al. (2013), utilizando técnicas de migração, monitoramento e uma infraestrutura de Computação em Nuvem, para melhorar o desempenho e reduzir custos. Quando uma violação de SLA é identificada, uma nova máquina virtual é provisionada para atender ao inquilino.

Redimensionamento Reativo

Enquanto as mais comuns implementações de escalabilidade na nuvem utilizam máquinas virtuais como uma unidade de escalonamento, alguns trabalhos propõem soluções em níveis menores para técnicas de elasticidade vertical para a adição ou remoção de recursos como CPU virtuais (VCPU) e Megabytes de memória RAM de uma máquina virtual.

Dawoud, Takouna e Meinel (2011) propuseram uma arquitetura para operações de elasticidade vertical. Uma comparação entre a implementação proposta e uma arquitetura com várias instâncias foi apresentada. A comparação inclui modelagem de filas e implementações de

cada arquitetura. É comentado que a elasticidade vertical implica em menos tempo de resposta e reduz o *overhead* do provisionamento, o que leva a menos violações de SLA.

Replicação Proativa

Além de abordagens reativas, abordagens proativas/preditivas têm sido utilizadas para prover soluções elásticas automatizadas. Estas soluções utilizam heurísticas e técnicas matemáticas para antecipar o comportamento das cargas de trabalho e utilizam estas predições para decidir como escalar os recursos (GALANTE; BONA, 2012).

Calheiros, Ranjan e Buyya (2011) propuseram uma técnica de provisionamento de máquinas virtuais para adaptação dinâmica às mudanças na carga de trabalho e prover garantias de QoS aos usários finais. Utiliza um modelo de filas e informações da carga de trabalho para direcionar decisões de replicação de um provisionador de aplicações. Resultados indicaram que a técnica proposta pode detectar mudanças na intensidade da carga de trabalho (padrões de entrada e demanda de recursos) no tempo e instanciar réplicas conforme os objetivos de QoS das aplicações. Tirado et al. (2011) combinaram um modelo preditivo com replicação para adição e remoção de réplicas com o objetivo de melhorar a utilização de recursos e distribuição da carga de trabalho. A predição da carga de trabalho é executada conforme um modelo de regressão, enquanto melhorias são realizadas baseadas em dados preditos conforme histórico.

Ferrer et al. (2012) apresentaram o OPTIMIS como uma ferramenta para solucionar problemas de gerenciamento de serviços na nuvem, com foco em serviços e otimizar a infraestrutura através de um ciclo de vida: construção, implantação e operação dos serviços. Ele utiliza dados diferentes para prever o impacto energético futuro baseado em estados em execução, padrões de utilização do histórico, e estimativas de futuras demandas. Esta estratégia adiciona e remove máquinas virtuais do sistema. Konstanteli et al. (2012) apresentaram uma abordagem probabilística do problema de alocação ótima de serviços sobre recursos físicos, quando elasticidade horizontal é requerida. O modelo de otimização constitui de um controle de teste de admissão probabilístico. Baseado no conhecimento estatístico, esta abordagem utiliza modelos que são capazes de prever utilização de recursos baseados no monitoramento de dados históricos. Em Santos et al. (2013), uma abordagem baseada na teoria do espaço de escala foi proposta para auxiliar no provisionamento dinâmico de recursos em nuvens computacionais. Esta teoria é capaz de eliminar a presença de informações irrelevantes a partir de um sinal, que pode introduzir erros na tomada de decisão. Nos experimentos confirmou-se que a abordagem melhorou a análise da carga de trabalho e predição.

Migração Proativa

Qin et al. (2012) apresentam um *framework* para o rebalanceamento de carga baseado no modelo *Monitor-Analyze-Plan-Execute* (MAPE), denominado ElastiCat. O *framework* tem como objetivo eliminar desbalanceamentos de carga enquanto minimiza o rebalanceamento dos custos, utilizando dois modelos de predição para prever o tempo de migração dos dados e o impacto no desempenho através de técnicas de máquina de aprendizagem, e então gerar um modelo de custos. Bruneo (2013) apresentou um modelo analítico baseado em *Stochastic Reward Nets* (SRN). Este modelo representa sistemas compostos por milhares de recursos e sua representação física e virtual, explorando conceitos específicos da nuvem como elasticidade,

permitindo predizer os efeitos de uma condição particular de uma carga de trabalho de maneira a analisar a habilidade do sistema reagir a uma situação de sobrecarga. Moreira et al. (2014) propuseram uma abordagem para a melhoria da qualidade do serviço para banco de dados relacionais multi-inquilinos. Utilizou-se técnicas de migração de inquilinos, monitoramento do sistema, estratégias de alocação e predição, tudo em um ambiente de Computação em Nuvem.

Soluções Mistas e Outras

Durante esta revisão não foi possível identificar a utilização de modelo e métodos em alguns trabalhos. Foram identificados quatro trabalhos que somente utilizaram replicação (NIE; XU, 2009; LI et al., 2011; RAVEENDRAN; BICER; AGRAWAL, 2011; MALIK; HUET; CAROMEL, 2012), um que utilizou somente migração (MA et al., 2010), e um que utilizou apenas redimensionamento (KUMAR; SHAE; JAMJOOM, 2012). Ainda foram identificados dois trabalhos que utilizaram a abordagem reativa (DENG et al., 2012; GHOSHAL; RAMAKRISHNAN, 2012) e dois trabalhos que utilizaram a abordagem proativa (NIEHORSTER et al., 2011; PANIAGUA; SRIRAMA; FLORES, 2011), mas métodos que utilizaram ambos não foram identificados.

Adicionalmente, outros trabalhos utilizaram uma solução mista, combinando diferentes modelos e métodos (ETCHEVERS et al., 2011; LI; TORDSSON; ELMROTH, 2011; YU et al., 2012; ALI-ELDIN et al., 2012; DAS; AGRAWAL; ABBADI, 2013). Estas soluções permitiram condições mais complexas, regras e políticas de alocação, podendo contribuir para o aumento da receita dos provedores (TOOSI et al., 2011).

Yu et al. (2012) propuseram um algoritmo de localização de banco de dados baseado em custos através de um plano de migração de restrições de recursos, e preferências do sistema e usuários. A solução reativa é combinada com o plano de migração patra minimizar o custo de migração e maximizar a utilização dos recursos na nuvem. Das, Agrawal e Abbadi (2013) apresentam o ElasTraS, um banco de dados escalável e multi-inquilino para o processamento de transações, e o Albatross, uma técnica de migração de banco de dados. ElasTraS direciona Albatross para um balanceamento de carga elástico pela adição de servidores durante picos de cargas de trabalho e consolida para poucos servidores em reduções na carga de trabalho. Este escalonamento elástico minimiza o custo operacional e garante bom desempenho mesmo na presença de mudanças de cargas de trabalho não previstas. Ali-Eldin et al. (2012) construíram um controlador de elasticidade que modifica a quantidade de máquinas virtuais alocadas a um serviço baseada em ações reativas e predições de cargas futuras. A solução é baseada na teoria das filas e foi utilizada para a construção de um controlador reativo adaptativo híbrido para rapidamente reagir à mudanças de cargas repentinas, considerando a heterogeneidade da carga de trabalho, e prevenir oscilações.

Soluções Elásticas de Provedores

Diversos provedores de Computação em Nuvem atualmente oferecem serviços de escalonamento automático. Tais serviços geralmente confiam em regras pré-determinadas ou políticas reativas para acionar suas operações de ajuste de recursos.

O Amazon Auto Scaling utiliza um método reativo e replicação para promover elas-

ticidade. O comportamento de um escalonamento automático segue um conjunto de regras que devem ser definidas. Uma regra de auto escalonamento possui a forma de um par consistindo de um antecedente e um consequente. A Amazon geralmente oferece condições baseadas em regras para adicionar e remover recursos. Esta abordagem simples tipicamente envolve a criação de regras para determinar quando escalar. Para cada regra o usuário deve definir uma condição baseada em uma variável. Por exemplo, se a carga de CPU exceder 80%, é disparada uma ação pré-determinada de *scaling up* ou *scaling down*.

Similar ao Amazon Auto Scaling, o Windows Azure possui o Windows Azure Application Autoscaling Block, que utiliza regras para definir quando escalar. GoGrid também permite aos usuários configurar um serviço de auto escalonamento em um ambiente *web*. Utilizando o GoGrid's Dynamic Load Balancer e API GoGrid, os usuários podem escalar suas aplicações.

Provedores de Computação em Nuvem sofrem com as limitações inerentes aos métodos de replicação: a utilização de máquinas virtuais como unidade de escalonamento, as quais podem causar sobreprovisionamento; e o balanceador de cargas. Diante dessas limitações, provedores públicos de nuvem disponibilizam aos usuários alguns tipos de métodos de escalonamento vertical. Abordagens de substituição são comuns na Amazon e GoGrid. Por exemplo, os usuários podem modificar o tipo da instância (*small* para *large*) e escalonar máquinas virtuais para mais ou para menos na Amazon EC2. GoGrid também oferece ferramentas de escalonamento vertical de memória RAM, a qual permite o escalonamento fácil de memória.

3.4 Ambientes, Metodologias e Métricas Específicos para Elasticidade

Alguns trabalhos na literatura desenvolveram aplicações e *frameworks* para operação em ambientes de Computação em Nuvem. Outros propuseram abordagens e metodologias para a realização da análise de desempenho em Computação em Nuvem. Esta seção discute alguns trabalhos correlatos ao *framework* proposto. Aqui eles estão organizados em (i) *benchmarks*, (ii) *frameworks*, (iii) arquiteturas e *testbeds*, (iv) metodologias para avaliação de desempenho em nuvens computacionais, e (v) métricas para avaliar a elasticidade.

Em geral, o projeto de experimentos não é claro. Muitos trabalhos não informam detalhes dos experimentos, que auxiliam em sua replicação, assim como detalhes da instalação e configuração dos *frameworks* e *benchmarks* disponibilizados. Outra deficiência comum em trabalhos é a não disponibilização de uma metodologia para a utilização das ferramentas em experimentos. Muitas vezes existem poucos detalhes ou não há uma abordagem sistemática para sua utilização, dificultando a análise de desempenho.

A Tabela 3.8 compara os trabalhos descritos sob o ponto de vista da técnica do experimento utilizada (Tec.), metodologia detalhada (Met.) e detalhamento do projeto do experimento (Proj.). A escala para a técnica do experimento é: medição (M), simulação (S) e modelagem analítica (MA), assim como suas possíveis combinações. Os valores da escala para a metodologia e projeto são: detalhado (D), parcial (P) e ausente (A), e foram baseados no projeto de experimentos descrito por Jain (1991). Ressalta-se que nos trabalhos citados não há

Tabela 3.8: Comparação entre trabalhos relacionados de *benchmarks*, *frameworks* e *testbeds*. Técnica (Tec.): Medição (M), Simulação (S) e Modelagem Analítica (MA); Metodologia (Met.) e Projeto de Experimentos (Proj.): Detalhado (D), Parcial (P) e Ausente (A)

Trabalho	**Objetivo**	**Tec.**	**Met.**	**Proj.**
CloudStone (SOBEL et al., 2008)	*Benchmark* para aplicações *web* 2.0	M	A	P
C-METER (YIGITBASI et al., 2009)	*Framework* para análise de desempenho em nuvem	M	P	P
CloudGauge (EL-REFAEY; RIZKAA, 2010)	*Benchmark* para virtualização em nuvem e medição de cargas de trabalho dinâmicas	M	P	P
LoM2HiS (EMEAKAROHA et al., 2010)	*Framework* para mapear métricas de nível mais baixo para SLA para detecção de violações	S	A	P
MalStone (BENNETT et al., 2010)	*Benchmark* para medir desempenho de computação intensiva de dados	M/S	A	P
CloudCMP (LI et al., 2010)	Comparar desempenho entre provedores de Computação em Nuvem	M	P	P
BenchLab (CECCHET et al., 2011)	*Testbed* para *benchmark web* 2.0	M/S	A	P
FairCPU (REGO et al., 2011)	Arquitetura de provisionamento de recursos baseada em CPU	M	A	P
CloudCrawler (CUNHA; MENDONÇA; SAMPAIO, 2013)	Teste de desempenho de IaaS para auxiliar dimensionamento de recursos	M	A	P
MELA (MOLDOVAN et al., 2013)	*Framework* para monitoração e análise da elasticidade em serviços de nuvem	S	A	P

o detalhamento do projeto, o que não quer dizer que não seguem uma metodologia ou não há projeto de experimentos, sendo em geral muito bem escritos.

Dentre o conjunto de trabalhos sobre *benchmarks*, Sobel et al. (2008) propuseram o CloudStone, um *benchmark* de aplicações *web* 2.0 para medir o custo monetário usuário/mês. O CloudGauge foi apresentado por El-Refaey e Rizkaa (2010) para virtualização em nuvem com o objetivo de medir cargas de trabalho dinâmicas, enquanto o MalStone é voltado para desempenho em aplicações orientadas à CPU (BENNETT et al., 2010). Li et al. (2010) compararam o desempenho entre provedores de nuvem mediante o *benchmark* CloudCMP, considerando aspectos de elasticidade, rede, custo e armazenamento, avaliados de maneira sistemática.

Em relação a *frameworks* e aplicativos para desempenho em nuvem, o C-METER (YIGITBASI et al., 2009) se destacou. Ele é um *framework* portável, extensível, com geração e submissão de cargas de trabalho sintéticas. Seguindo um princípio de ciclo de vida autoadaptável com gerenciamento autonômico, outro *framework* que merece destaque é o LoM2HiS (EMEAKAROHA et al., 2010), cujo objetivo é mapear métricas de infraestrutura (IaaS) para métricas de usuário (SaaS), visando detecção de quebras de SLA. O MELA (MOLDOVAN et al., 2013) foi proposto para o monitoramento e análise de serviços elásticos, utilizando outros conceitos de elasticidade, como espaço e caminho da elasticidade.

Em relação a arquiteturas e *testbeds*, o BenchLab é descrito como um ambiente para *benchmark* com foco na *web* 2.0 (CECCHET et al., 2011), onde alguns requisitos são apresentados, como aplicação e cargas de trabalho reais. Uma arquitetura de provisionamento de recursos baseada em CPU foi apresentada por Rego et al. (2011), baseada na ferramenta

CPULIMIT, que limita a utilização da CPU em máquinas virtuais. Uma aplicação para teste de desempenho em IaaS foi desenvolvida por Cunha, Mendonça e Sampaio (2013), chamada CloudCrawler, para auxiliar o dimensionamento de recursos.

Critérios a serem utilizados na avaliação da elasticidade foram descritos em Herbst, Kounev e Reussner (2013), tais como: processos de adaptação para escalabilidade autonômica; recursos escaláveis para o processo de adaptação; variação da quantidade de recursos alocados; e limite superior da quantidade de recursos que podem ser alocados. Além disso, métricas foram propostas para a medição da elasticidade baseadas em velocidade e precisão. É um trabalho motivacional, não havendo experimentos, porém propõe métricas. No trabalho de Simões e Kamienski (2014), uma avaliação do desempenho de um ambiente híbrido de nuvem foi realizada por meio de simulações. Os resultados mostraram que a escolha de métricas e limiares é fundamental na manutenção do SLA. A relação custo/benefício dos experimentos também foi analisada em relação aos limiares estabelecidos, tempo de resposta e recursos utilizados. Costa et al. (2011) analisaram motivos pelos quais provedores de nuvem tradicionais necessitam impor limites sobre a quantidade de recursos que um usuário pode adquirir simultaneamente. Um modelo matemático foi proposto para representar um provedor de IaaS e a geração de cargas de trabalho sintéticas. Seus experimentos simularam aplicações *Bag of Tasks* (BoT) em relação aos limites impostos por provedores na execução deste tipo de aplicação.

O FOLE procura estabelecer uma metodologia para a realização de análise de desempenho em ambientes de Computação em Nuvem de maneira sistemática. Também destaca o projeto dos experimentos, assim como o projeto de cargas de trabalho, coletas e medições. Ele não define uma arquitetura, mas sugere fortemente aspectos ferramentais que influenciam diretamente em seus componentes, como recursos de Computação Autonômica. Ele não define ou disponibiliza uma API (*Application Programming Interface*) para o desenvolvimento de códigos. Por ser um *framework* conceitual, ele é genérico e independente de tecnologia. Como em alguns trabalhos relacionados, a ideia é ser extensível e adaptável, porém projetando a avaliação de maneira flexível e reutilizável.

Bai et al. (2011) discutiram a métrica *Performance Resource Ratio* (PRR) que reflete os relacionamentos entre desempenho e os recursos utilizados. Esta métrica é baseada nos tempos de espera e execução de recursos alocados, que podem ser CPU, memória, I/O e largura de banda. Gao et al. (2011) descreveram um conjunto de métricas para analisar a escalabilidade e elasticidade em ambientes de SaaS. Estas métricas estão associadas à alocação de recursos, utilização dos recursos, desempenho do sistema, carga do sistema, capacidade efetiva do sistema, escalabilidade efetiva do sistema e faixa efetiva de escalabilidade.

Garg, Versteeg e Buyya (2012) descreveram uma métrica para elasticidade como o quanto um serviço de nuvem pode ser escalado durante picos, que pode ser definido pelo tempo médio para expandir ou contrair a capacidade do serviço e a capacidade máxima do serviço (número máximo de unidades computacionais que podem ser providas em períodos de pico). Maneiras de quantificar elasticidade são discutidas em Islam et al. (2012), considerando aspectos de qualidade e penalidades em caso de não atendimento do serviço. Um modelo matemático baseado em penalidades, recursos e demandas foi descrito. Experimentos foram realizados na Amazon EC2 com o *benchmark* TPC-W e o aplicativo JMeter para geração de cargas de tra-

balho. Um conjunto de métricas relacionadas à elasticidade foi descrito em Li et al. (2012), identificado sobre outros trabalhos, e destacando que avaliar elasticidade em Computação em Nuvem não é uma atividade trivial. As métricas descritas estão agrupadas em tempo de aquisição de recursos (tempo de provisão, de inicialização, de aquisição), tempo de liberação de recursos (tempo de suspensão, de deleção, de liberação), e custo financeiro.

Um conjunto de métricas para elasticidade foi proposto por Herbst, Kounev e Reussner (2013), com a ideia de velocidade e precisão. É um trabalho motivacional, não havendo experimentos, porém propõe uma métrica específica para elasticidade baseada em tempos de operações e recursos. As métricas Elasticidade de *Scaling Up* e Elasticidade de *Scaling Down* foram propostas, e levam em consideração a quantidade de recursos alocados e o período de tempo dispendido durante períodos de subprovisionamento e sobreprovisionamento.

Novas ideias relacionadas ao teste de sistemas computacionais elásticos foram introduzidas por Gambi et al. (2013), onde foram identificados alguns desafios de pesquisa e tendências futuras. Neste mesmo trabalho foi sugerida uma analogia com conceitos de elasticidade definidos na Física (Elasticidade de Materiais) com sistemas computacionais elásticos, tais como: deformação, recuperação, plasticidade e tensão. Também foi elaborada uma analogia com testes mecânicos para a definição de testes adequados a sistemas computacionais. Porém neste trabalho não foram discutidas métricas, apenas a metáfora entre as áreas. Além disso, não foram realizados experimentos.

Uma abordagem para a medição da elasticidade de uma nuvem foi proposta por Shawky e Ali (2012), baseada nos princípios da elasticidade definidos na Física. Nesta definição é abordado o conceito de estresse da nuvem, definido como a taxa de recursos requerida pela quantidade de recursos alocados, e o de tensão da nuvem, definido como a variação nos recursos antes e depois de operações de escalonamento. Neste trabalho, métricas para nuvens computacionais foram propostas, baseadas em capacidade de computação demandada e alocada, largura de banda, e tempo levado para alocar novos recursos. Entretanto, suas métricas estão específicas para largura de banda e foram realizados apenas experimentos de simulação, além de carecer da interpretação das métricas. Para o cálculo do estresse foi utilizado o conceito de capacidade de computação, que é estimada pela quantidade total de dados processados em Gigabytes. O foco para o cálculo da tensão foi em largura de banda.

A Tabela 3.9 exibe um comparativo entre trabalhos que propuseram métricas específicas ou indiretas para medir elasticidade, ou avaliações de desempenho. A nomenclatura utilizada na tabela é: Experimento (Medição, Simulação, Modelagem Analítica, Não possui experimentos); Tipo da Métrica (Específica para elasticidade, Indireta, Não propôs métrica).

Em nosso trabalho, a meta é estudar a elasticidade em ambientes de Computação em Nuvem. Assim como em alguns dos trabalhos citados anteriormente, pretende-se também executar experimentos com medidas reais para o atendimento dos objetivos, propondo métricas para avaliar a elasticidade. Entretanto, a aplicação das métricas propostas é mais simples e direta. Nossas métricas são alternativas para a medição da elasticidade. Seus detalhamentos (descrição das variáveis envolvidas e interpretação dos resultados) são úteis para que um pesquisador possa aplicá-las com mais facilidade na avaliação de desempenho da elasticidade em

Tabela 3.9: Comparação entre trabalhos relacionados de métricas para elasticidade. Experimento (Exp.): Medição (M), Simulação (S), Modelagem Analítica (MA), Não possui experimentos (N); Tipo da Métrica (Tipo): Específica para elasticidade (E), Indireta (I), Não propôs métrica (N)

Trabalho	**Exp.**	**Tipo**	**Observações**
Bai et al. (2011)	N	I	Métrica baseada em tempos de espera e execução de recursos alocados. É um *survey*
Gao et al. (2011)	M	I	Métrica baseada em recursos utilizados em alocação. Mede a capacidade e carga de trabalho. Utilizaram uma nuvem pública
Garg, Versteeg e Buyya (2012)	S/MA	I	Métrica baseada em tempo e capacidade. Utilizou modelagem matemática para um compor um *ranking* de serviços
Islam et al. (2012)	M/MA	E	Métrica baseada em recursos e demandas. Utilizou um modelo matemático para penalidades e qualidade em uma nuvem pública
Li et al. (2012)	N	I	Métricas baseadas em recursos, tempo e custo financeiro. É uma revisão sistemática da literatura
Shawky e Ali (2012)	S	E	Métrica baseada em largura de banda. Utilizou conceitos da Física
Costa et al. (2013)	S/MA	N	Utilizou modelagem matemática para representar um provedor de IaaS
Gambi et al. (2013)	N	N	Descreve analogias com outras áreas de conhecimento
Herbst, Kounev e Reussner (2013)	N	E	Métrica baseada em tempo de operações de alocação e recursos. Utilizou a ideia de velocidade e precisão da elasticidade
Simões e Kamienski (2014)	M	N	Experimentos baseados em tempo de resposta e utilização de recursos. Realizou uma análise do custo/benefício em nuvem híbrida

Computação em Nuvem. Como elas foram baseadas em métricas de elasticidade de outras áreas do conhecimento, então já existe uma referência da sua utilização prática.

As métricas propostas neste trabalho podem ser utilizadas de diversas maneiras: na comparação de soluções, onde se mede o quanto uma solução é elástica, ou qual solução é mais elástica; em comparações de cargas de trabalho, onde se verifica qual carga de trabalho necessita de um ambiente mais ou menos elástico para ser executada corretamente (atendimento ao SLA), ou quais características que uma carga de trabalho deve possuir para avaliar a elasticidade de um ambiente; e apoiar na tomada de decisão.

4 MÉTRICAS PARA ELASTICIDADE EM NUVENS COMPUTACIONAIS

Este capítulo descreve um conjunto de métricas propostas para apoiar a medição da elasticidade em ambientes de Computação em Nuvem. A Seção 4.1 apresenta métricas baseadas em tempos de operações de alocação e desalocação de recursos, e na utilização dos recursos. A Seção 4.2 descreve métricas baseadas em conceitos da Física (tensão e estresse). A Seção 4.3 apresenta métricas para medição da elasticidade baseadas em conceitos da Microeconomia (Elasticidade do Preço da Demanda). Na Seção 4.4, a maneira de se interpretar os resultados das métricas específicas de elasticidade é apresentada. Por fim, a Seção 4.5 descreve alguns benefícios da utilização das métricas propostas.

4.1 Métricas Orientadas a Tempos de Alocação e Utilização de Recursos

Um conjunto de métricas para a medição da elasticidade foi proposto por Herbst, Kounev e Reussner (2013), apresentando as ideias de velocidade e precisão da escalabilidade. Métricas específicas para elasticidade foram propostas baseada em tempos de operações e recursos, denominadas Elasticidade de *Scaling Up* e Elasticidade de *Scaling Down*. Essas métricas consideram a quantidade de recursos alocados e o intervalo de tempo dispendido durante períodos de subprovisionamento e sobreprovisionamento. Entretanto este trabalho é motivacional, e não há a realização de experimentos. Também não foram avaliados os períodos de tempo entre estados sobreprovisionados e subprovisionados.

Baseado nas ideias de Herbst, Kounev e Reussner (2013), propomos métricas orientadas aos tempos de execução de operações de alocação de recursos, e consideramos os períodos de tempo entre estados sobreprovisionados e subprovisionados, denominados estados transitórios, e também propomos métricas para avaliar situações de estabilidade. Adicionalmente propusemos métricas baseadas na utilização de recursos para cada um dos estados de alocação. Não foi proposta uma métrica específica para elasticidade, e sim métricas secundárias, que permitem o apoio e análise da elasticidade em ambientes. Além disso, Herbst, Kounev e Reussner (2013) utilizaram medidas de tempo e recursos, agregando seus valores médios, para compor métricas específicas para a medição da elasticidade, diferentemente de nossa maneira de cálculo. Por fim, não foram realizados experimentos em seu trabalho.

Em nosso trabalho, definimos o termo “estabilizado” como a situação onde a alocação dos recursos permanece a mesma, ou seja, não há adição ou remoção de recursos. O período de em tempo que ocorrer esta situação será considerado como um intervalo de tempo estabilizado. O termo “transitório” foi definido como o tempo entre estados no qual o ambiente ainda está se adaptando às ações de elasticidade, ou seja, ainda está sofrendo os efeitos das ações para que o aumento ou redução de recursos seja efetivado, não estando ainda disponíveis.

Quatro métricas relacionadas ao tempo de execução de operações no ambiente foram propostas: Tempo de Alocação Sobreprovisionada, Tempo de Alocação Subprovisionada, Tempo de Alocação Estabilizada e Tempo de Alocação Transitória. O Tempo de Alocação Sobreprovisionado (*TASo*) corresponde ao tempo utilizado para a execução de operações de

remoção de recursos. O Tempo de Alocação Subprovisionada ($TASu$) corresponde ao tempo utilizado para a execução de operações de adição de recursos. O Tempo de Alocação Estabilizada (TAE) corresponde ao intervalo de tempo onde não há adição ou remoção de recursos, sendo que a alocação permanece a mesma, ou estabilizada. O Tempo de Alocação Transitória (TAT) corresponde ao tempo em que os efeitos da adição ou remoção de recursos estão em progresso.

Também foram considerados os recursos envolvidos nas operações de alocação e desalocação. Três métricas foram propostas: Total de Recursos Alocados Subprovisionados, Total de Recursos Alocados Sobreprovisionados e Total de Recursos Alocados Estabilizados. O Total de Recursos Alocados Subprovisionados ($TRASu$) corresponde à quantidade de recursos alocados em um estado subprovisionado, mas não estabilizado. O Total de Recursos Alocados Sobreprovisionados ($TRASo$) corresponde à quantidade de recursos alocados em um estado sobreprovisionado, mas não estabilizado. O Total de Recursos Alocados Estabilizados ($TRAE$) corresponde à quantidade de recursos alocados em um estado estabilizado.

Especificamente para a métricas orientadas à recursos, consideramos recursos como qualquer recurso utilizado pelo ambiente que possa ser adicionado ou removido para ações de ajuste da capacidade do ambiente. Exemplos de recursos são: máquinas virtuais (instâncias que podem ser adicionadas ou removidas), réplicas (que podem ser geradas e adicionadas e removidas), e unidades de processamento (capacidade de aumentar ou diminuir recursos de infraestrutura, como CPU e memória, a uma quantidade ou unidade pré-determinada). Assim, essas métricas são dependentes da infraestrura, pois vão depender da estratégia empregada.

Para as métricas orientadas a tempos de operações de alocação e desalocação de recursos, a estratégia de alocação e desalocação só influencia na duração, sendo assim, as métricas de tempo independentes da estratégia empregada, bastando apenas a coleta destes tempos. O que diferencia é que, como as estratégias são diferentes, o resultado das métricas podem demorar mais ou menos (variar para mais ou para menos). Um exemplo é uma estratégia que utilize migração de máquinas virtuais, que demoraria mais do que uma estratégia que apenas adiciona uma máquina virtual já pré-existente a um balanceador de carga. Existirão tempos para as 4 métricas, mas alguns variarão mais que outros devido à estratégia utilizada.

A Figura 4.1 descreve de forma gráfica e, somente para propósitos didáticos, as métricas propostas relacionadas a tempos de alocação estabilizada, subprovisionada, sobreprovisionada e transitória. Adicionalmente, é possível analisar o comportamento das métricas propostas à quantidade de recursos em estados estabilizados, subprovisionados e sobreprovisionados. Esta figura é um modelo teórico, apenas para ilustar os pontos de controle e coleta dos tempos considerados nas métricas, e os períodos de tempo onde recursos estão alocados. Considerando um modelo em que as unidades de recursos são limitadas e atômicas, como a quantidade de máquinas virtuais, a ideia das métricas se adéqua aos deslocamentos na Figura 4.1. Em situações mais realísticas, a estratégia de elasticidade pode ser horizontal ou vertical, e a forma de se calcular os tempos e a quantidade de recursos varia conforme a estratégia utilizada.

Na situação em azul consideramos como alocação estabilizada, o que significa que não há necessidade de aumentar ou reduzir recursos. É a situação onde os recursos do ambiente

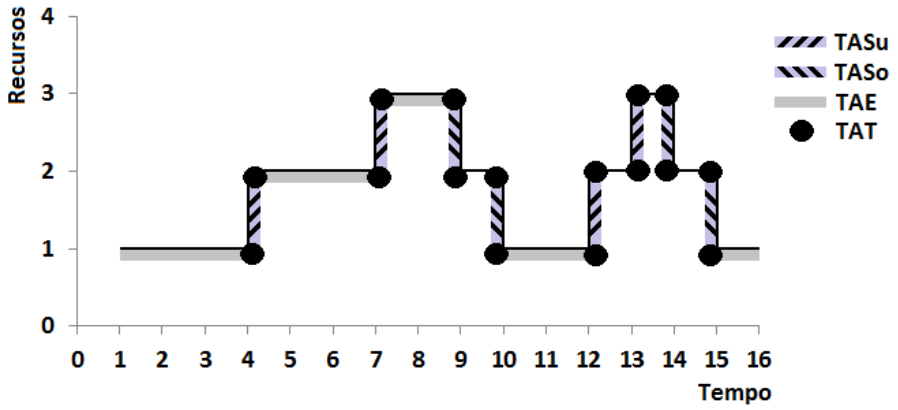


Figura 4.1: Representação das métricas propostas para a análise da elasticidade em nuvem

estão adequados às cargas de trabalho. Na situação em laranja ocorre a necessidade de mais recursos, por isso a capacidade deve aumentar, caso contrário violações de SLA ocorrerão. Na situação em verde ocorre a necessidade de reduzir recursos, por isso a capacidade deve diminuir, caso contrário estará entrado em uma situação de desperdício de recursos. A situação representada por um círculo preto é o intervalo em que a infraestrutura está se adaptando ao que foi adicionado ou removido de recursos. É o tempo que se gasta para que as operações de adição ou remoção de recursos tenham efeito no ambiente. Por isso ela sempre aparece após um momento de aumento e de redução de recursos. As métricas propostas orientadas a tempos de operações de alocação e desalocação de recursos correspondem aos tempos dos estados representados. As métricas propostas orientadas à quantidade de recursos correspondem à quantidade de recursos alocada aos estados representados.

É possível ter períodos em que não há estabilidade entre intervalos de coletas, representados por linhas mais finas na Figura 4.1. Esta é uma situação onde há a necessidade de aumentar ou diminuir ainda mais os recursos, sem ter passado por um período de estabilidade de recursos, e depende do tamanho do intervalo de coleta definido.

TAT é o período de tempo representado pelo círculo preto, e em teoria existe entre todo ponto de mudança de estado de alocação de recursos ou passagem de um estado para outro. Caso *TAT* seja bem curto, o que depende da estratégia de elasticidade projetada para a nuvem, mesmo existindo ele poderá ser ignorado, pois em relação aos demais tempos ele será quase insignificante. Outra situação que pode ser desprezada caso seja relativamente curta são as situações em que não há estabilidade entre intervalos de coletas, podendo este tempo ser considerado pertencente a estados sobreprovisionados ou subprovisionados.

Utilizando um modelo apresentado para a representação de um provedor proposto por Costa et al. (2011), e considerando que os recursos de um provedor estão alocados em intervalos de tempo, então tem-se um ambiente de Computação em Nuvem A, para um período de tempo ΔT. Assim, formula-se a seguinte tupla:

$$A =< R, N, U, O, S, T, U_r, O_r, S_r >$$

Tabela 4.1: Métricas propostas orientadas a tempos de operações e a recursos alocados

Métrica	Descrição
$TASo = \sum_{1}^{m} Ui$	Ui = período de tempo no estado de alocação subprovisionada
$TASu = \sum_{1}^{m} Oi$	Oi = período de tempo no estado de alocação sobreprovisionada
$TAE = \sum_{1}^{m} Si$	Si = período de tempo no estado de alocação estabilizada
$TAT = \sum_{1}^{m} Ti$	Ti = período de tempo no estado de alocação transitória
$TRASu = \sum_{1}^{m} U_r i$	$U_r i$ = quantidade de recursos r no período de tempo i no estado de alocação subprovisionada
$TRASo = \sum_{1}^{m} O_r i$	$O_r i$ = quantidade de recursos r no período de tempo i no estado de alocação sobreprovisionada
$TRAE = \sum_{1}^{m} S_r i$	$S_r i$ = quantidade de recursos r no período de tempo i no estado de alocação estabilizada

onde R corresponde à quantidade de recursos disponíveis no ambiente, N é a quantidade de intervalos de tempo de todo o período de tempo (dividindo o tempo total do experimento em períodos menores), os elementos U, O, S e T correspondem a conjuntos de intervalos referentes às métricas de tempo $TASo$, $TASu$, TAE e TAT, respectivamente, e os elementos U_r, O_r e S_r correspondem a conjuntos de intervalos referentes às métricas de recursos $TRASu$, $TRASo$ e $TRAE$, respectivamente.

A Tabela 4.1 exibe as métricas compostas por somatórios, consideradas métricas de referência, onde cada conjunto de intervalos pode ser vazio. A partir das métricas de referência, métricas secundárias podem ser derivadas a partir de seus valores totais, médios e percentuais, e assim servirem como análise complementar. A Tabela A.1 do Apêndice A exibe métricas complementares baseadas em tempos de alocação e desalocação de recursos, e a Tabela A.2 do Apêndice A exibe métricas complementares baseadas na quantidade de recursos.

A principal diferença entre as métricas propostas nesse trabalho e as propostas por Herbst, Kounev e Reussner (2013) é que existe o complemento de três métricas (TAE, TAT e $TRAE$), devido a uma necessidade de avaliar o ambiente durante períodos de estabilidade e transição, assim como seus recursos.

4.2 Métricas Orientadas a Física

Na Física, segundo Timoshenko e Goodier (1970), todo material estrutural possui uma determinada extensão da propriedade elasticidade. Se forças externas aplicadas ao material produzirem uma deformação na estrutura, não excedendo um determinado limite, a deformação desaparece com a remoção das forças. Para isso, os conceitos estresse e tensão da Física foram estudados e métricas para avaliação da elasticidade em nuvens computacionais foram propostas.

A Tabela 4.2 descreve as variáveis relacionadas às métricas propostas, assim como a nomenclatura a ser utilizada. As métricas associadas a recursos têm seus valores coletados do ambiente, como utilização de CPU e quantidade de máquinas virtuais alocadas. A variável t corresponde ao número de iterações em que são coletados os diferentes parâmetros utilizados, definido pelo intervalo de coleta.

Na Física, o estresse indica o quão forte é um material. Ele é definido como a quantidade de pressão que o material pode suportar, sem sofrer algum tipo de alteração física (TIMOSHENKO; GOODIER, 1970; SHAWKY; ALI, 2012). Considerando que o estresse na Física é uma força que age sobre a área da secção transversal do material, fazendo a analogia com uma nuvem computacional, onde R_D são os recursos demandados impostos pelas cargas de trabalho e R_A são os recursos alocados na atual configuração da infraestrutura, temos que o Estresse da Qualidade da Nuvem (S_{QN}) pode ser aferido pela Equação 4.1:

Tabela 4.2: Lista de variáveis e nomenclatura para métricas baseadas em conceitos da Física

Variável	Descrição
R_D	Recursos Demandados: recursos necessários para o atendimento do SLA
R_A	Recursos Alocados: recursos alocados ao ambiente
t	Total de iterações (intervalos de coleta)
R_DAtu	Recursos Demandados Atuais: recursos demandados pelo ambiente no momento da coleta para atendimento do SLA
R_DAnt	Recursos Demandados Anteriores: recursos demandados pelo ambiente em momentos de coleta anteriores
R_AAtu	Recursos Alocados Atuais: recursos alocados ao ambiente no momento da coleta
R_AAnt	Recursos Alocados Anteriores: recursos alocados ao ambiente em momentos de coleta anteriores
Δ_{RD}	$R_DAtu - R_DAnt$
Δ_{RA}	$R_AAtu - R_AAnt$
S_{QN}	Estresse da Qualidade na Nuvem: taxa na qual os recursos são demandados em relação aos recursos alocados
T_{RD}	Tensão dos Recursos Demandados: taxa que mede a variação dos recursos demandados entre intervalos de coleta (Δ_{RD}) e os recursos demandados ao ambiente no momento da coleta
T_{RA}	Tensão dos Recursos Alocados: taxa que mede a variação dos recursos alocados entre intervalos de coleta (Δ_{RA}) e os recursos alocados ao ambiente no momento da coleta
E_{RDi}	Elasticidade dos Recursos Demandados: relação entre S_{QN} e T_{RD}
E_{RAi}	Elasticidade dos Recursos Alocados: relação entre S_{QN} e T_{RA}
$\overline{E_{RDi}}$	Elasticidade Média dos Recursos Demandados: média aritmética de todas as medições para E_{RDi} calculadas em cada um dos intervalos de coleta t
$\overline{E_{RAi}}$	Elasticidade Média dos Recursos Alocados: média aritmética de todas as medições para E_{RAi} calculadas em cada um dos intervalos de coleta t

$$S_{QN} = \begin{cases} 0, & \text{se } R_A = 0 \\ \frac{R_D}{R_A}, & \text{caso contrário} \end{cases} \tag{4.1}$$

Algumas situações específicas devem ser observadas. No caso em que os recursos demandados e os recursos alocados são zero, considera-se que o estresse é zero. No caso em que recursos demandados são diferentes de zero e os recursos alocados são iguais a zero, o estresse tende a infinito. Esta situação semanticamente significa que há uma demanda onde não há recursos alocados e nada que foi demandado será atendido, e para fim de interpretação, esta situação também implicará em um estresse igual a zero.

A tensão, conforme a definição na Física, é o quanto um objeto está sendo estendido ou distorcido, e pode ser medida através da divisão entre a extensão de um material e seu comprimento após a distorção provocada pela aplicação de forças externas (TIMOSHENKO; GOODIER, 1970; SHAWKY; ALI, 2012). Fazendo uma analogia com a nuvem computacional, a Tensão dos Recursos Demandados (T_{RD}) pode ser medida pela Equação 4.2 e a Tensão dos Recursos Alocados (T_{RA}) pode ser aferida pela Equação 4.3:

$$T_{RD} = \begin{cases} 0, & \text{se } R_D = 0 \\ \frac{\Delta_{R_D}}{R_D Atu}, & \text{caso contrário} \end{cases} \tag{4.2}$$

$$T_{RA} = \begin{cases} 0, & \text{se } R_A = 0 \\ \frac{\Delta_{R_A}}{R_A Atu}, & \text{caso contrário} \end{cases} \tag{4.3}$$

Conforme a Lei de Hook, a elasticidade pode ser medida pela divisão entre o estresse e a tensão (TIMOSHENKO; GOODIER, 1970; SHAWKY; ALI, 2012). A elasticidade dos recursos em um determinado ponto i pode ser calculada de duas maneiras diferentes, dependendo da fórmula de tensão utilizada (T_{RD} ou T_{RA}). A Equação 4.4 considera os recursos demandados, enquanto que a Equação 4.5 considera os recursos alocados.

$$E_{RDi} = \begin{cases} 0, & \text{se } T_{RDi} = 0 \\ \frac{S_{QNi}}{T_{RDi}}, & \text{caso contrário} \end{cases} \tag{4.4}$$

$$E_{RAi} = \begin{cases} 0, & \text{se } T_{RAi} = 0 \\ \frac{S_{QNi}}{T_{RAi}}, & \text{caso contrário} \end{cases} \tag{4.5}$$

Por fim, a elasticidade média dos recursos pode ser calculada tanto para os recursos demandados ($\overline{E_{RDi}}$) quanto para os recursos alocados ($\overline{E_{RAi}}$), por meio das Equações 4.6 e 4.7 respectivamente.

$$\overline{E_{RDi}} = \frac{\sum_{i=0}^{t} E_{RDi}}{t} \tag{4.6}$$

$$\overline{E_{RAi}} = \frac{\sum_{i=0}^{t} E_{RAi}}{t} \tag{4.7}$$

4.3 Métricas Orientadas a Microeconomia

A demanda por um bem ou serviço é definida como a quantidade de um bem ou serviço os quais pessoas estão aptas (dispostas e capazes) para comprar a preços diferentes em um determinado período de tempo, considerando outros fatores além do preço mantidos constante (SALVATORE, 2011). Como exemplo de demanda, considere um indivíduo que compra 3.5 toneladas de milho todo mês com o preço da tonelada a R$ 5. Essa é sua demanda individual. Se existirem 1000 indivíduos no mercado com demandas semelhantes, a demanda por milho é a soma das quantidades que os 1000 indivíduos comprariam neste preço. Então, se existirem 1000 indivíduos dispostos a adquirir 3500 toneladas de milho ao preço de R$ 5 por mês, esta seria a demanda do mercado. Se o preço mudar para R$ 4, a demanda individual se modifica para 4.5 toneladas de milho, e a demanda do mercado seria 4500 toneladas.

Na Microeconomia, a elasticidade pode ser definida como uma medida da capacidade de resposta da quantidade demandada ou da quantidade fornecida a uma alteração em um dos seus determinantes (disponibilidade de substitutos próximos, necessidades *versus* luxos, definição do mercado e horizonte de tempo) (MANKIW, 2012). Assim, a Elasticidade da Demanda mede a extensão em que a quantidade demandada de uma mercadoria aumenta ou diminui em resposta ao aumento ou diminuição de qualquer dos seus determinantes quantitativos (JAIN; OHRI, 2012).

A Elasticidade do Preço da Demanda mede o quanto a quantidade demandada de um bem responde a uma mudança no preço desse bem, calculada como a variação percentual da quantidade demandada dividida pela variação percentual no preço (MANKIW, 2012). A Elasticidade do Preço da Demanda é a relação entre a variação percentual da quantidade e a variação percentual no preço do bem, e todos os outros fatores mantidos constantes (SAMUELSON; MARKS, 2012). Dessa maneira, a Elasticidade do Preço da Demanda pode ser medida pela Equação 4.8, onde Q é a quantidade do produto, P é o preço do produto, Δ_Q é a variação na quantidade, Δ_P é a variação no preço.

$$\text{Elasticidade do Preço da Demanda} = \frac{\%\text{ variação em } Q}{\%\text{ variação em } P} = \frac{\Delta_Q}{\Delta_P} \times \frac{P}{Q} \tag{4.8}$$

Como exemplo de Elasticidade do Preço da Demanda, seguindo no exemplo da demanda apresentado, considere o preço do produto e a quantidade deste produto demandada pelo mercado. Em um período p_0, o preço unitário do produto é R$ 5 e a quantidade demandada é 3500. Em um período p_1, o preço unitário do produto modificou para R$ 4 e quantidade demandada modificou para 4500. Assim, a elasticidade do preço da demanda nos os períodos indicados é -1.42. Devido à quantidade de um produto ser negativamente relacionada a seu preço, o percentual de mudança na quantidade sempre possuirá o sinal oposto ao percentual

Tabela 4.3: Lista de variáveis e nomenclatura para métricas baseadas em Microeconomia

Variável	Descrição
R_D	Recursos Demandados: recursos necessários para o atendimento do SLA
R_A	Recursos Alocados: recursos alocados ao ambiente
t	Total de iterações (intervalos de coleta)
R_DAtu	Recursos Demandados Atuais: recursos demandados pelo ambiente no momento da coleta para atendimento do SLA
R_DAnt	Recursos Demandados Anteriores: recursos demandados pelo ambiente em momentos de coleta anteriores
R_AAtu	Recursos Alocados Atuais: recursos alocados ao ambiente no momento da coleta
R_AAnt	Recursos Alocados Anteriores: recursos alocados ao ambiente em momentos de coleta anteriores
Δ_{RD}	$R_DAtu - R_DAnt$
Δ_{RA}	$R_AAtu - R_AAnt$
E_{Di}	Elasticidade da Demanda: multiplicação da relação entre a variação dos recursos alocados (Δ_{RA}) e recursos demandados (Δ_{RD}) entre intervalos de coleta pela relação entre os recursos demandados e alocados
$\overline{E_{Di}}$	Elasticidade Média da Demanda: média aritmética de todas as elasticidades calculadas em cada um dos intervalos de coleta t

de mudança no preço (MANKIW, 2012). Neste exemplo, o percentual de mudança no preço é negativo (refletindo um decréscimo no preço), e o percentual de mudança na quantidade demandada é positivo (refletindo um crescimento na quantidade). Por este motivo, a elasticidade do preço da demanda algumas vezes é reportado com valores negativos.

Fazendo a analogia com a elasticidade em nuvens computacionais, podemos considerar a quantidade demandada Q como a alocação de recursos ou o quanto de recursos está sendo disponibilizado, e o preço dos bens P como as cargas de trabalho impostas.

A Tabela 4.3 descreve as variáveis relacionadas às métricas propostas, assim como a nomenclatura a ser utilizada. Para o cálculo da elasticidade, fazendo a analogia com a elasticidade em nuvens computacionais, podemos considerar a quantidade demandada como os recursos alocados R_A, e o preço dos bens como os recursos demandados R_D impostos pelas cargas de trabalho. Consequentemente, a variação na quantidade demandada equivale a Δ_{RA}, e a variação no preço do produto equivale a Δ_{RD}. Como o nome da métrica na Microeconomia se chama “Elasticidade do Preço da Demanda", seu equivalente será chamado de Elasticidade da Demanda (E_{Di}), e pode ser medida pela Equação 4.9. Por fim, a elasticidade média da demanda $\overline{E_{Di}}$ pode ser calculada por meio da Equação 4.10.

$$E_{Di} = \begin{cases} 0, & \text{se } R_A = 0 \text{ ou } \Delta_{RA} = 0 \text{ ou } \Delta_{RD} = 0 \\ \frac{\Delta_{RA}}{\Delta_{RD}} \times \frac{R_D}{R_A}, & \text{caso contrário} \end{cases} \tag{4.9}$$

$$\overline{E_{Di}} = \frac{\sum_{i=0}^{t} E_{Di}}{t} \tag{4.10}$$

Algumas situações específicas devem ser tratadas. No caso em que os recursos demandados são zero, considera-se que a elasticidade é zero, já que não há demanda. No caso em que recursos alocados são zero, a elasticidade tende a ser infinita. Esta situação semanticamente significa que há uma demanda onde não há recursos alocados. No caso em que Δ_{RD} for igual a

zero, a elasticidade também será igual a zero, pois implica que não houve variação nos recursos demandados no período.

Conforme a Lei da Demanda, considerando outras variáveis de outros aspectos permanecendo iguais, a demanda por um bem aumenta com a diminuição do preço, e a demanda por um bem diminui com um aumento no preço (SALVATORE, 2011). Esse efeito também ocorre em sistemas computacionais. Se os recursos demandados aumentam, os recursos alocados começam a diminuir no sentido de já estarem sendo utilizados, tornando-se escassos. Se os recursos demandados diminuem, os recursos alocados começam a se tornar mais disponíveis, pois à medida em que vão sendo liberados ou finalizando suas atividades, tornam-se novamente disponíveis.

4.4 Interpretação das Métricas para Elasticidade em Nuvens Computacionais

As métricas propostas têm como objetivo medir a elasticidade de uma nuvem computacional, logo seus resultados possuem o mesmo objetivo. A maneira de interpretar as métricas propostas baseadas em conceitos da Física (E_{RDi}, E_{RAi}, $\overline{E_{RDi}}$ e $\overline{E_{RAi}}$) e na Microeconomia (E_{Di} e $\overline{E_{Di}}$) pode ser realizada da mesma maneira na maioria das situações. Apesar dos gráficos gerados pela duas métricas serem diferentes, e eles exibirem valores em diferentes escalas para mesmos experimentos, suas interpretações estão alinhadas. Além disso, as métricas propostas podem corroborar uma com a outra. As Figuras 4.2 e 4.3 exibem gráficos para as métricas de elasticidade propostas. As linhas azul, vermelha e preta dos gráficos representam respectivamente as métricas E_{RAi}, E_{RDi} e E_{Di}.

A Figura 4.2 exibe situações onde a elasticidade pode ser observada pelas métricas propostas para avaliar a utilização e necessidade de recursos. Existem dois casos: utilização de mais ou menos recursos, e necessidade por mais recursos. Na primeira metade da Figura 4.2 os valores para as duas métricas estão baixos em relação à segunda metade do gráfico. Isto indica que nesse período de tempo do experimento os recursos estão relativamente adequados à carga de trabalho. Já na segunda parte do gráfico, os valores estão bem maiores, indicando a utilização de mais recursos. No segundo caso, na primeira parte do gráfico está sendo indicado que não há tanto a necessidade de mais recursos, enquanto que na segunda parte há a necesssidade por mais recursos.

A Figura 4.3 apresenta situações nas quais as métricas propostas são utilizadas para avaliação de estados de alocação e desalocação de recursos. Existem dois casos: muitos ou poucos eventos de alocação e desalocação de recursos, e maior ou menor estabilidade entre estados subprovisionados e sobreprovisionados. Os valores para as duas métricas estão mais espaçados que na Figura 4.2. Isto indica que muitos eventos de alocação/desalocação de recursos estão ocorrendo, podendo ser imediatamente sequenciais ou não, e em geral sem muitos intervalos de alocação mínima. Em relação à estabilidade, quanto mais espaçado e horizontal nos picos, mais estável a alocação.

Valores absolutos individuais elevados para E_{RDi} e E_{RAi} indicam nesse ponto duas possíveis situações: se o valor coletado for positivo indicam muitos recursos alocados utilizados

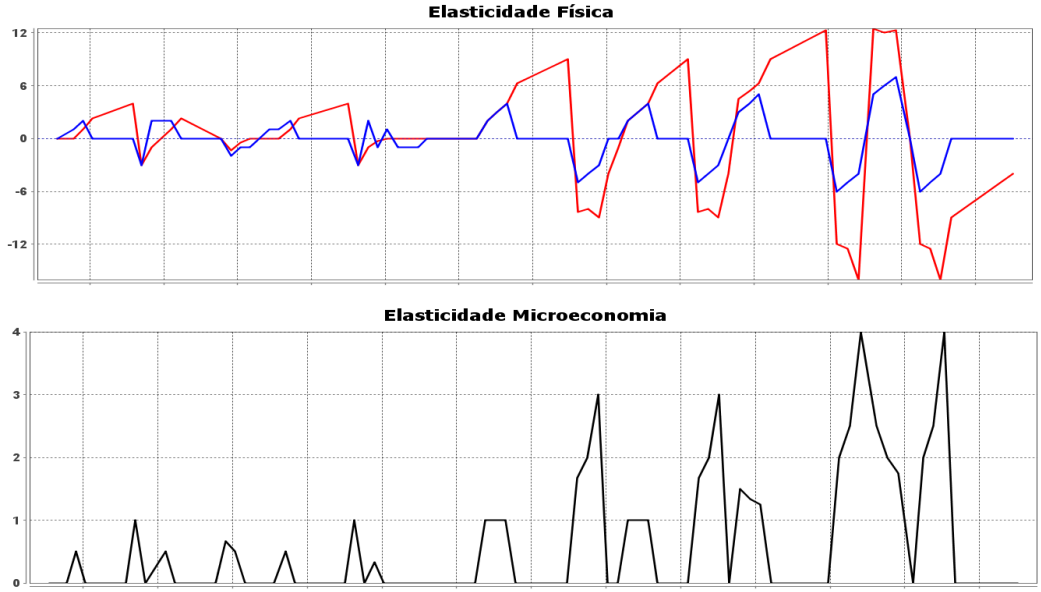


Figura 4.2: Gráfico da elasticidade para as métricas propostas para avaliação da utilização e necessidade de recursos

ou a necessidade de maior quantidade de recursos; e se o valor coletado for negativo, menos recursos alocados são necessários, ou não há a necessidade da quantidade atual de recursos alocados. Em geral, para E_{RDi} e E_{RAi} sempre há picos positivos seguidos de picos negativos correspondentes a eventos de alocação e desalocação de recursos, respectivamente.

Valores individuais elevados para E_{Di} indicam nesse ponto duas possíveis situações: muitos recursos alocados utilizados ou a necessidade de maior quantidade de recursos. Valores individuais baixos para E_{Di} indicam que poucos recursos alocados são necessários, ou não há

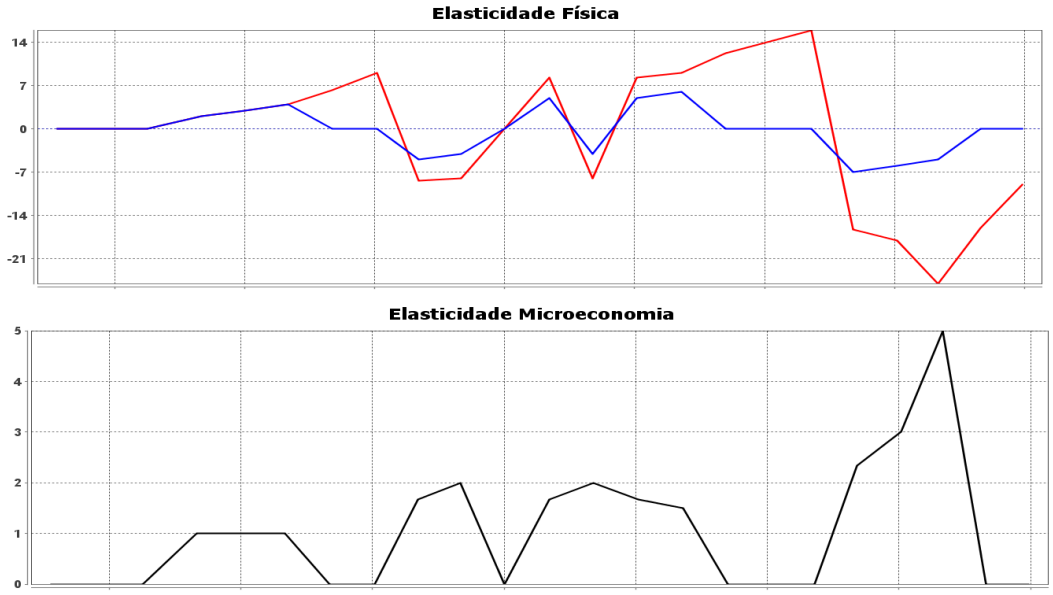


Figura 4.3: Gráfico da elasticidade para as métricas propostas para avaliação de estados de alocação e desalocação de recursos

a necessidade da quantidade atual de recursos alocados. Em geral, para E_{Di} sempre há picos positivos correspondentes a eventos de alocação e desalocação de recursos, respectivamente.

Alguns fatores podem influenciar no resultado obtido no cálculo das métricas. O ponto de início e término de coleta do experimento impacta diretamente nos valores da elasticidade. Vários pontos sem atividade no ambiente são considerados com elasticidade igual a zero, influenciando nos valores médios. Outro aspecto a ser considerado é a estratégia de alocação de recursos a ser utilizada no ambiente. Dependendo da maneira na qual as unidades de alocação são utilizadas, por exemplo máquinas virtuais adicionadas ou removidas, e capacidade de CPU a ser incrementada ou reduzida, os tempos nas operações e a maneira na qual os recursos são alocados influenciarão nos valores das métricas. Um exemplo é uma estratégia de elasticidade horizontal onde as máquinas virtuais são apenas adicionadas e removidas (já estão ativas e são só alocadas ao balanceador de carga), ou no caso em que elas são instanciadas (criadas) e removidas em tempo real (dinamicamente).

4.5 Benefícios da Utilização das Métricas de Elasticidade

Este trabalho propôs métricas para avaliação da elasticidade de maneira indireta (baseada em tempos de alocação e desalocação de recursos e na utilização de recursos) e direta (baseado em conceitos da Física e Microeconomia). A aplicação dessas métricas em geral é mais simples e direta que métricas propostas na literatura, o que facilita sua utilização na prática. O fato de existir uma maneira mais detalhada de sua concepção e interpretação é útil para que um pesquisador ou administrador de um provedor possam aplicá-las com mais facilidade na avaliação de desempenho da elasticidade em Computação em Nuvem.

As métricas orientadas a tempos de alocação e desalocação auxiliam na identificação de períodos de tempo onde os recursos estão sendo mal utilizados, tais como momentos de necessidade, ociosidade e desperdício. Analisando o tempo de estabilidade é possível se identificar um ajuste dos recursos às demandas, mas também é possível que o SLA não esteja sendo atendido. Um exemplo é a situação em que a capacidade máxima de recursos foi atingida, as demandas estão ocorrendo, não há mais como expandir a capacidade ou alocar mais recursos, e consequentemente o SLA não será atendido. Nesse caso, a alocação está estabilizada.

As métricas orientadas à utilização de recursos medem como eles estão sendo distribuídos ao longo do tempo diante de estados de subprovisionamento, sobreprovisionamento e estabilidade. Essa quantidade de recursos também pode indicar a efetividade do mecanismo de alocação e desalocação de recursos e a intensidade de operações de alocação.

As métricas baseadas em conceitos da Física e Microeconomia avaliam a elasticidade de maneira direta. A analogia com elementos dessas áreas de conhecimento tem relação e faz sentido com cargas de trabalho, demanda e oferta. O suporte à análise devido à interpretação baseada nos gráficos também permite uma avaliação mais prática da elasticidade no tempo.

Por fim, mecanismos de predição baseados nas métricas propostas também podem ser desenvolvidos, e assim a construção de soluções elásticas mais eficientes, que evitem o desperdício e sejam mais ágeis na alocação de recursos.

5 FRAMEWORK CONCEITUAL FOLE

Na literatura, metodologias detalhadas para análise de desempenho não são tão comuns de se encontrar. Alguns autores, como Jain (1991) e Menasce, Dowdy e Almeida (2004) detalham diversos aspectos da análise de desempenho de sistemas computacionais.

Avaliar o desempenho de uma nuvem computacional é uma atividade comumente encontrada na literatura. Porém, avaliar a elasticidade não é tão comum. Muitos trabalhos utilizam recursos de elasticidade em seus experimentos, porém existem diversos aspectos a serem considerados da infraestrutura, além da dificuldade em encontrar trabalhos bem detalhados, com reuso dos componentes e flexibilidade suficientes para a reprodução dos experimentos.

Diante dos aspectos descritos, surge o problema de como realizar análise de desempenho de maneira flexível e reutilizável em ambientes de Computação em Nuvem para a avaliação da elasticidade. Em geral, trabalhos não são reutilizáveis, não são flexíveis, falta o detalhamento e descrição das atividades, assim como a maneira de utilização e interpretação das métricas, dificultando a comparação e reprodução dos resultados.

A Seção 5.1 apresenta uma motivação do porquê avaliar a elasticidade em uma nuvem computacional. A Seção 5.2 apresenta o FOLE, um *framework* conceitual para análise de desempenho da elasticidade. A Seção 5.3 descreve cada atividade do *framework*. A Seção 5.4 apresenta detalhes para a aplicação do FOLE. Ao final deste capítulo, na Seção 5.5, uma proposta de arquitetura elástica e uma ferramenta para apoiar a análise e visualização dos dados são apresentadas.

5.1 Elasticidade em Computação em Nuvem

Neste trabalho vamos considerar a definição de elasticidade proposta por Herbst, Kounev e Reussner (2013): “Elasticidade dele ser definida como o grau no qual um sistema é capaz de se adaptar à variações na carga de trabalho pelo provisionamento e desprovisionamento de recursos de maneira autonômica, de modo que em cada ponto no tempo os recursos disponíveis combinem com a demanda da carga de trabalho o mais próximo possível”.

Seguindo o ponto de vista desta definição, alguns aspectos devem ser considerados na análise de desempenho da elasticidade em Computação em Nuvem: Computação Autonômica, provisionamento de recursos e carga de trabalho. Esses elementos possuem total relação com a elasticidade em Computação em Nuvem.

A Computação Autonômica possui elementos de monitoramento, ação, regras, sensores e atuadores, promovendo a adaptação do ambiente conforme a necessidade. O provisionamento de recursos é uma característica arquitetural que influencia na infraestrutura e a estratégia de elasticidade depende dela, podendo ser implementada de diversas maneiras. A carga de trabalho, que pode ser gerada por usuários reais ou *benchmarks*, é quem provoca as ações que devem ocorrer em um ambiente elástico, assim como o consumo dos recursos.

Entende-se por “avaliar o desempenho da elasticidade” como a verificação do com-

portamento da elasticidade de uma nuvem computacional diante de cargas de trabalho aplicadas a esse ambiente.

Um experimento dito mais elástico que outro significa que ele possui um grau composto por diversos aspectos combinados entre si, tais como estratégias de elasticidade, alocação e desalocação de recursos, e que uma vez aplicada uma carga de trabalho sobre o ambiente, este grau avaliará a elasticidade e indicará o quanto os recursos estão sendo adequados às necessidades da carga de trabalho imposta ou do usuário, sinalizando o quanto o ambiente é elástico.

Assim, a avaliação da elasticidade em Computação em Nuvem é uma tarefa que se torna essencial para provedores de acesso e clientes que desejam avaliar um ambiente de nuvem computacional. Essa avaliação considera diversos aspectos, que podem variar conforme a aplicação desejada. A avaliação do desempenho da elasticidade pode ocorrer sob diversos aspectos de uma nuvem computacional: avaliar arquiteturas, conhecer um ambiente, comparar infraestruturas computacionais, adaptar um ambiente, comparar soluções elásticas, identificar falhas de qualidade em um ambiente, avaliar o custo financeiro da utilização de serviços, etc. Também pode ser aplicada na avaliação de elementos que agem sobre o ambiente, tais como: avaliar o efeito de métricas, analisar o comportamento de aplicações, avaliar as cargas de trabalho aplicadas ao ambiente, etc.

Suponha que uma empresa necessite realizar o processamento de um grande conjunto de dados, e para isto utiliza um provedor de Computação em Nuvem para adquirir uma instância com a suposta capacidade para a realização de tal tarefa. Suponha que ao longo do processamento, a capacidade não seja suficiente, e a aplicação corra o risco de ser interrompida. Caso ocorra uma interrupção e o trabalho seja finalizado de maneira indesejada, mesmo os resultados não sendo concluídos, o cliente pagará pela utilização da instância. Uma solução é a utilização de um serviço de elasticidade, disponibilizado por diversos provedores. Um mecanismo de monitoramento, baseado em alguma regra ou métrica, fica observando o estado do ambiente, e em caso de ultrapassar algum limiar pré-estabelecido, aciona o mecanismo de elasticidade para adicionar mais recursos ao ambiente, seja pela adição de novas instâncias, seja pelo aumento da capacidade da instância em uso. Assim, o processamento em andamento pode continuar e encerrar de maneira correta e satisfatória. Após o encerramento da tarefa, o mecanismo de monitoramento observa que os recursos adicionados não são mais necessários, e são removidos do ambiente.

Atualmente existem muitas soluções de elasticidade. Nesse contexto, é interessante que o usuário saiba qual a melhor solução a ser contratada. Assim, um mecanismo que possibilite a avaliação da elasticidade possibilitaria que infraestruturas fossem avaliadas, assim como aplicações utilizadas e suas características, tais como duração dos experimentos e custo financeiro, aspectos comumente encontrados em provedores de Computação em Nuvem.

5.2 O *Framework* Conceitual FOLE

Um *framework* ou arcabouço conceitual é um conjunto de conceitos utilizados para a resolução de um problema de um domínio específico, sendo que existem dois tipos: *fra-*

(a) (b)

Figura 5.1: Exemplos de acordeão (a) e de um fole (b).

meworks verticais (ou especialistas, confeccionados através da experiência obtida em um determinado domínio específico ou de um especialista, tentando resolver problemas de um determinado domínio de aplicação), e *frameworks* horizontais (podem ser utilizados em diferentes domínios) (CRUZ, 2013).

FOLE, o *framework* apresentado neste trabalho, é vertical, pois seu foco é no domínio de Computação em Nuvem, especializado em elasticidade. Nele propomos um conjunto de atividades para a execução de uma análise de desempenho em nuvem de maneira sistematizada. As atividades relacionadas ao planejamento da elasticidade foram elaboradas com base em princípios de Computação Autonômica, na arquitetura definida por Kephart e Chess (2003). Basicamente, o FOLE possui três macroatividades, relacionadas a planejamento, inicialização de serviços e execução, e suas respectivas atividades.

O *framework* tem como nome FOLE devido a uma analogia com o instrumento musical acordeão. O acordeão ou acordeom é um instrumento musical aerofone de origem alemã, composto por um fole, um diapasão e duas caixas harmônicas de madeira. O som do acordeão é criado quando o ar que está no fole passa por entre duas palhetas, que vibram mais grave ou agudo de acordo com a distância entre elas (quando mais distantes, mais grave o som) e seu tamanho (quanto maior, mais grave o som produzido). Quanto mais forte o ar é forçado para as palhetas, mais intenso é o som. O ar é proveniente do fole, que é aberto ou fechado com o auxílio do braço esquerdo. A Figura 5.1 ilustra um acordeão e um fole.

Então, fazendo a analogia com a elasticidade, à medida em que recursos são mais ou menos necessários, o ambiente se adapta às necessidades, assim como o acordeão e o fole trabalhando em conjunto.

5.3 Descrição das Atividades

A Figura 5.2 ilustra o FOLE de maneira integrada e com a ideia de fluxo de operação. A sequência de atividades representada pela linha cheia possibilita a análise de desempenho de maneira genérica, podendo ser aplicada a qualquer ambiente computacional. A linha pontilhada representa sua extensão para um estudo de caso de elasticidade. A ideia de ciclo

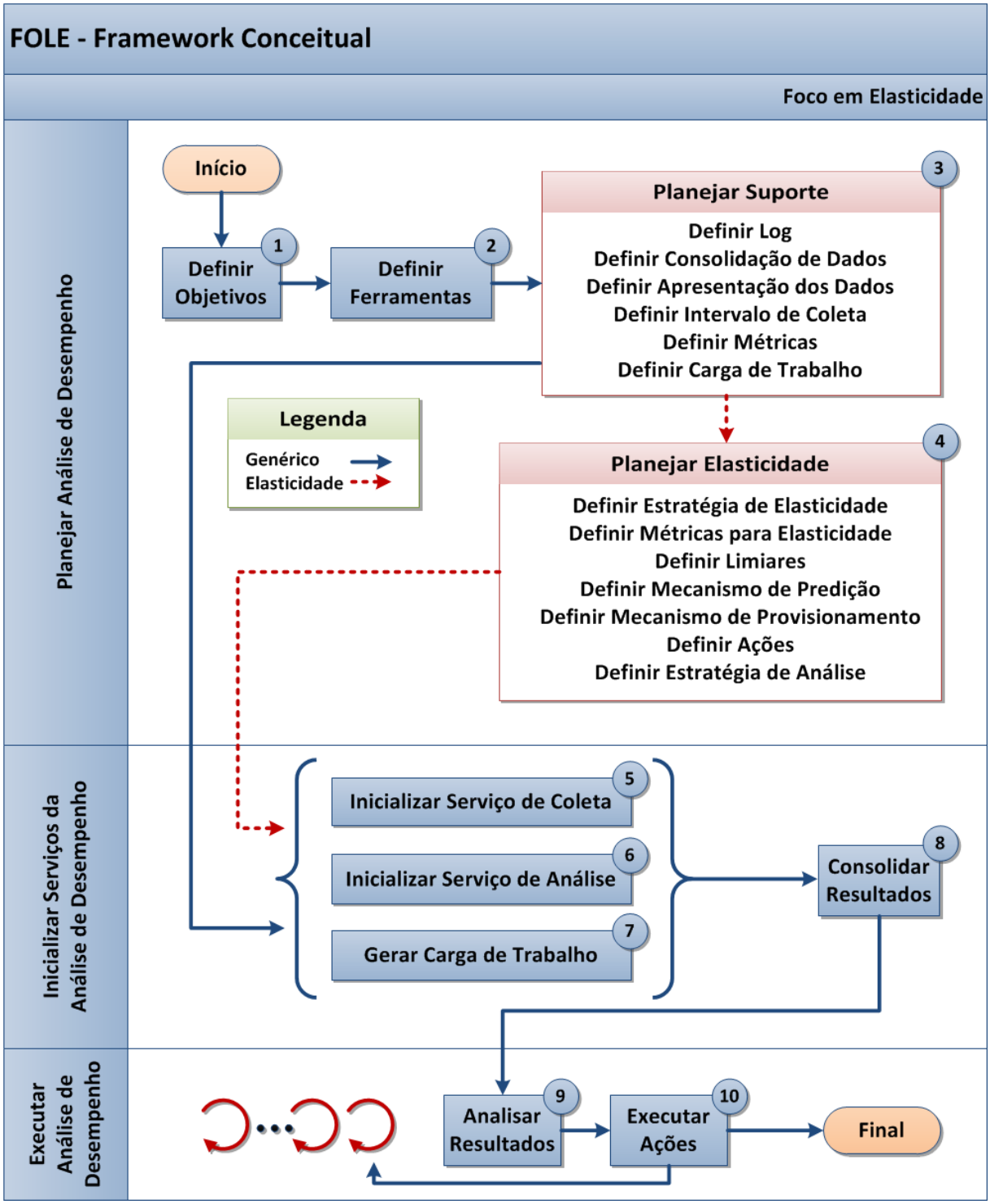


Figura 5.2: Fluxo de atividades do *framework* FOLE

de vida ou fases pode ser aplicada nesta abordagem devido à natureza cíclica da análise de desempenho proposta. Após a atividade "Executar Ações" é possível retornar para um replanejamento, caso necessário, ou repetir a macroatividade "Executar Análise de Desempenho", e obter uma nova avaliação do ambiente.

- **Macroatividade - Planejar Análise de Desempenho**

1. **Definir Objetivos:** descrever o sistema a ser analisado e os objetivos a serem atendidos na análise de desempenho (geral e específicos). Os objetivos da análise de desempenho são bastante variados, podendo ser comparações de tecnologias, resolução de problemas ou investigar algum aspecto do ambiente. Também é possível querer avaliar o desempenho de uma determinada aplicação real, ou simulada, em um ambiente de Computação em Nuvem, e assim verificar o comportamento da elasticidade. Dessa maneira, o objetivo pode ser conhecer o ambiente, verificar o comportamento do ambiente, identificar os efeitos das cargas de trabalho aplicadas ao ambiente, verificar se o ambiente se adapta de maneira satisfatória às necessidades das cargas de trabalho impostas, e por fim verificar o atendimento ao SLA definido.
2. **Definir Ferramentas:** listar ferramentas a serem utilizadas no experimento para operações de infraestrutura, suporte, armazenamento, programação, carga de trabalho e análise.
   - Infraestrutura: ferramentas que provém suporte às funcionalidades de um provedor de Computação em Nuvem, tais como *middlewares* e hipervisores. Mecanismos para alocação e desalocação de recursos também são necessários ser definidos para a infraestrutura, como balanceadores de carga e ferramentas para aumentar e reduzir a capacidade dos recursos;
   - Suporte: em geral *scripts* são construídos para definir o fluxo de operações do ambiente, como a geração de cargas de trabalho, coleta de métricas, consolidação dos resultados, e posterior análise. Além de apoiar serviços do ambiente, a integração de diferentes componentes e subsistemas também pode ser necessária;
   - Armazenamento: ferramentas que proveem funcionalidades de armazenamento das informações do ambiente. O mecanismo de armazenamento pode ocorrer de várias maneiras, tais como: arquivos texto para o armazenamento do *log* de dados, banco de dados relacionais, e mecanismos de manipulação de arquivos XML;
   - Programação: linguagens de programação para construção de ferramentas, *microbenchmarks* e consolidação de dados. Eventuais integrações são necessárias, então as linguagens de programação possibilitam o desenvolvimento de soluções para integração entre partes e padronização;
   - Carga de Trabalho: diversas ferramentas podem ser empregadas na geração de cargas de trabalho, tais como aplicações, *benchmarks* e *microbenchmarks*. Estas ferramentas geralmente possuem algum mecanismo de configuração da carga de trabalho;
   - Análise: em geral são ferramentas que suportam a análise de dados. Podem conter mecanismos estatísticos, gráficos e relatórios. Ferramentas de monitoração também auxiliam na análise de dados, alimentando bancos de dados com as informações a serem utilizadas pelas análises. *Scripts* para automação da análise também podem ser utilizados;

As ferramentas estão diretamente associadas à arquitetura do ambiente. Muitos provedores de Computação em Nuvem possuem diversas ferramentas, seja APIs para manipular máquinas virtuais (coleta de dados, adição, remoção, etc), seja para monitorar dados. Em geral, a utilização dessas ferramentas em provedores públicos é paga, como na Amazon EC2 e Microsoft Azure. Em *middlewares* de código livre, como o OpenNebula e OpenStack, a possibilidade de configuração é maior devido ao código fonte disponível. Além disso, diversas ferramentas, linguagens de programação e utilitários podem ser utilizados pelos usuários e desenvolvedores para suporte às demais atividades, tais como *scripts*.

3. **Planejar Suporte:** definir atividades para o planejamento do suporte às operações gerais do ambiente
   - Definir *Log*: definir o mecanismo de armazenamento do *log* de dados (e.g., banco de dados, arquivo texto ou XML, CSV, separado por “;" ou “|");
   - Definir Consolidação de Dados: definir o mecanismo de consolidação dos dados obtidos das diversas fontes de dados do experimento para poderem ser utilizados em operações do ambiente e tomada de decisão. A consolidação deve considerar uma padronização sempre que possível, pois assim os diversos componentes e serviços do ambiente poderão se comunicar de maneira mais efetiva. Além do mecanismo, uma estratégia de consolidação dos dados através de informações estatísticas (média, mediana, moda, variância, desvio padrão, erro e intervalo de confiança) pode ser utilizada para permitir uma análise mais global;
   - Definir Apresentação dos Dados: identificar formas de apresentação dos dados para análise adequados para análise de dados (gráficos e tabelas). Existem diversos tipos de gráficos na literatura que podem auxiliar na visualização dos dados. Em geral, gráficos de linha e barra são os mais comumente utilizados para a exibição de dados. Gráficos que agregam diversas informações estatísticas, como o *boxplot*, também são formas de apresentação de dados que podem auxiliar na análise de desempenho. Entretanto, cada gráfico possui seus benefícios, sendo necessário a correta seleção para uma melhor visualização e análise. Tabelas também são muito utilizadas. A correta estrutura da tabela permite uma melhor consolidação de resultados. Ferramentas de visualização de dados podem ser utilizadas, como pacotes gráficos, planilhas ou ferramentas desenvolvidas para este fim;
   - Definir Intervalo de Coleta: definir intervalo de tempo em que as ferramentas irão coletar dados dos recursos pré-definidos do ambiente. Deve-se considerar o tamanho do tempo de coleta, o *overhead* e a sincronização dos dados, pois é possível que a frequência de coleta seja maior ou menor que os momentos de consolidação de dados, e acabe interferindo nos resultados, ou eles não estejam sincronizados, havendo uma defasagem. Além disso, o *overhead* imposto pela coleta, armazenamento e análise dos dados também deve ser avaliado, pois sua duração pode ser maior que o tempo definido como intervalo de coleta;
   - Definir Métricas: definir as métricas que irão mapear recursos do ambiente

(e.g., CPU, memória, disco, rede) e recursos das aplicações (tempo de resposta) utilizadas na análise. Em geral métricas de propósito geral são agrupadas nas seguintes categorias: tempo de resposta, utilização, *throughput*, disponibilidade, escalabilidade, aquisição e confiabilidade (JAIN, 1991). Métricas comumente identificadas em muitos trabalhos são o percentual de utilização de CPU, o *throughput* e tempo de resposta de requisições;

- Definir Carga de Trabalho: projetar a carga de trabalho a ser aplicada ao ambiente estabelecido. Alguns aspectos de projeto de experimentos devem ser considerados no projeto de experimento da carga de trabalho, tais como: taxa de repetição, ferramentas envolvidas e quantidade de repetições da carga de trabalho. Outras informações sobre projeto de experimentos podem ser encontradas em Jain (1991) e Montgomery (2009). As cargas de trabalho podem ser geradas por meio de aplicações reais ou sintéticas, *benchmarks* ou *microbenchmarks*, ou traços computacionais. No caso da avaliação de um ambiente em produção (ambiente real com usuários reais), considera-se o volume de trabalho gerado equivalente à carga de trabalho projetada;

4. **Planejar Elasticidade:** definir atividades para o planejamento da análise de desempenho da elasticidade da nuvem computacional
   - Definir Estratégia de Elasticidade: definir qual estratégia de provimento da elasticidade será empregada no experimento. As estratégias podem ser horizontal (replicação), vertical (redimensionamento ou substituição) ou migração;
   - Definir Métricas para Elasticidade: identificar no conjunto de métricas definidas quais serão utilizadas para a medição da elasticidade, considerando a combinação de métricas (indiretas e específicas à elasticidade). Métricas para a análise de desempenho da elasticidade podem ser relacionadas à infraestrutura, como percentual de utilização de CPU e memória, relacionadas à aplicação, como o tempo de resposta das requisições dos usuários, ou métricas especificas para a medição da elasticidade, como as métricas propostas neste trabalho (E_{RDi}, E_{RAi} e E_{Di}). Essas métricas serão utilizadas para monitorar e avaliar a elasticidade do ambiente. Seus resultados avaliarão a adequação da alocação e desalocação dos recursos em relação às cargas de trabalho;
   - Definir Limiares: definir os limiares associados ao SLA que o ambiente deve prover. Muitas vezes os limiares estão associados às políticas definidas para o sistema a ser avaliado, resultando em regras. Essas regras podem ser funções de ação, objetivo ou utilitárias, conforme o nível de especificação desejado. A quantidade de limiares também deve ser considerada, pois quanto maior a quantidade de limiares, mais *overhead* do sistema para sua avaliação. Em geral, deve-se definir um limite inferior e um limite superior, com a possibilidade de limites intermediários. Esses limiares serão considerados como regras que serão monitoradas pelo ambiente para ações de elasticidade. Esses limites podem estar associados a recursos do ambiente, tais como utilização de CPU e memória, ou métricas de aplicações, como a quantidade de requisições executadas por segundo e tempo de resposta;

  - Definir Mecanismo de Predição: identificar mecanismos de predição a serem utilizados no ambiente. Avaliar mecanismos reativos e proativos para análise. Mecanismos reativos reagem à carga de trabalho atual e utilizam políticas para disparar ações. Mecanismos proativos utilizam técnicas de predição para determinar quando a carga de trabalho futura irá superar ou não a capacidade atual, requisitando alocação ou desalocação de recursos;
  - Definir Mecanismo de Provisionamento: definir mecanismos de provimento de recursos para o ambiente (balanceamento de carga, migração de máquinas virtuais, ampliação e redução de capacidade, etc). Esses mecanismos irão atuar conforme a estratégia de elasticidade definida;
  - Definir Ações: definir ações que serão executadas em caso da necessidade do provimento / desprovimento de recursos. Em geral são disparadas por eventos no ambiente baseados nos limiares e regras definidas, que adicionam ou removem recursos conforme a necessidade ou políticas definidas;
  - Definir Estratégia de Análise: definir estratégias para operacionalização da análise do ambiente quanto à elasticidade (por exemplo: estatística descritiva, média móvel e *n* últimos valores coletados), utilizando métricas definidas. A utilização de mecanismos estatísticos auxilia na estratégia definida (média, mediana, moda, variância, desvio padrão, erro e intervalo de confiança);

- **Macroatividade - Inicializar Serviços da Análise de Desempenho**

  5. **Inicializar Serviço de Coleta:** executar o conjunto de atividades para a inicialização dos serviços de coleta de dados (limpeza, coleta e armazenamento). Atividades comuns são: executar *scripts* para limpar arquivos temporários, reinicializar variáveis globais, iniciar a coleta dados das diversas fontes e armazenar os dados coletados em bases de dados.
  6. **Inicializar Serviço de Análise:** executar o conjunto de atividades para a inicialização dos serviços de análise do ambiente baseado nas estratégias definidas. Atividades comuns são: executar *scripts* para leitura de dados e utilizar estratégias de análise conforme técnica definida.
  7. **Gerar Carga de Trabalho:** executar as atividades de geração das cargas de trabalho projetadas (inicialização de *scripts*, leitura de configurações de banco de dados, ferramentas, etc). A carga de trabalho projetada é executada, gerando uma carga sobre o ambiente. Em caso de avaliação de um ambiente de produção, a carga de trabalho é o volume atual de trabalho do ambiente.
  8. **Consolidar Resultados:** executar o conjunto de atividades para a consolidação dos resultados obtidos do ambiente para análise (organização e formatação para a análise). Atividades comuns são: padronizar resultados para leitura por ferramentas, calcular métricas consolidadas, gerar arquivos de *log* e mover arquivos para locais específicos.

- **Macroatividade - Executar Análise de Desempenho**

9. **Analisar Resultados:** avaliar os resultados consolidados conforme objetivos e estratégias definidas e direcionar as ações a serem executadas para tomada de decisão. Analisa as regras definidas pelos limiares e sinaliza a necessidade de aumentar ou reduzir recursos do ambiente (elasticidade).
10. **Executar Ações:** executar ações projetadas para adequação do ambiente ao SLA e atendimento aos objetivos da análise de desempenho. Em geral, *scripts* construídos em atividades anteriores são executados para o ajuste do ambiente às necessidades de recursos ou para uma melhor distribuição dos recursos atuais, seja pela adição ou pela remoção de recursos. Essas ações irão tornar o ambiente mais elástico no sentido de melhor se adaptar às cargas de trabalho submetidas.

5.4 Aplicação do FOLE

O FOLE é um *framework* conceitual. Desse modo, ele é abstrato de forma que para sua utilização, o usuário deve instanciar cada uma de suas atividades, fazendo a correspondência com o ambiente computacional a ser avaliado. Entende-se por instanciar o ato de tornar uma atividade abstrata ou genérica em concreta e personalizada para um determinado ambiente.

Como o objetivo do FOLE é avaliar a elasticidade de um ambiente de Computação em Nuvem, todas as atividades de elasticidade devem ser instanciadas, e assim, atividades relacionadas à elasticidade descreverão o comportamento do ambiente sob este aspecto. Também é possível que o usuário utilize o FOLE para projetar ou validar uma arquitetura, e assim as atividades definirão o comportamento do ambiente. As demais atividades são tarefas que se adaptariam bem à análise de desempenho de qualquer ambiente computacional, e caso fossem utilizadas para este fim, algumas atividades podem ser optativas, conforme a característica do ambiente.

Na situação em que a infraestrutura já está ativa, com usuários em atividade e cargas de trabalho atuando, as atividades do FOLE devem ser identificadas sobre a infraestrutura atual, e pontos de deficiência podem ser identificados. Dessa maneira, a infraestrutura poderá ser avaliada, assim como as aplicações e experimentos conduzidos, do ponto de vista da elasticidade em Computação em Nuvem. Como consequência de sua utilização temos: a evolução da infraestrutura e das estratégias de elasticidade utilizadas, e adequação dos recursos às cargas de trabalhos aplicadas ao ambiente.

O FOLE possui três macroatividades, cada uma respectivamente com foco em planejamento, serviços e análise. As atividades de planejamento possuem um apelo grande para a definição da infraestrutura, e são atividades voltadas para o projeto de experimentos. As atividades de serviços e de análise são bastante operacionais, e em geral são implementadas por meio de *scripts* desenvolvidos para serem executados sobre a infraestrutura.

Muitas ferramentas estão disponíveis para a instanciação do FOLE, em geral ferramentas de infraestrutura, suporte, armazenamento, programação, carga de trabalho e análise. A seleção das ferramentas a serem utilizadas deve ser cuidadosa, pois elas impactarão na infraestrutura, requerendo certo tempo para estudo e configuração. O ideal é que elas sejam ferra-

mentas de domínio dos usuários e que já estejam em plena utilização. Porém o FOLE permite uma flexibilidade do ponto de vista arquitetural, e ferramentas que possuam a mesma funcionalidade podem ser trocadas ou interagir sem problemas. Para isto, normalmente o usuário deve ajustar as atividades das macroatividades "Inicializar Serviços da Análise de Desempenho" e "Executar Análise de Desempenho".

Existem nuvens computacionais de diferentes tipos: privada, pública, comunidade e híbrida. O FOLE possui suporte a todos esses tipos. Além disso, existe a possibilidade de novas combinações de utilização de nuvens computacionais com outras arquiteturas, como nuvens distribuídas, Computação Móvel, Redes de Sensores sem Fio e Computação de Alto Desempenho (*High Performance Computing* - HPC). Todas essas tecnologias podem se beneficiar da elasticidade. Assim, a análise de desempenho da elasticidade passa por diversos níveis, tendo seus efeitos propagados nos níveis de infraestrutura (IaaS), níveis de desenvolvimento de plataformas (PaaS) e chegando à aplicação do usuário (SaaS).

5.5 Suporte Ferramental

Esta seção descreve o suporte necessário para a validação do *framework* e das métricas propostas. Para isto, uma arquitetura baseada em algumas características de Computação Autonômica foi proposta, sendo seus componentes descritos na Seção 5.5.1. Para apoiar a visualização e análise dos resultados obtidos por meio da coleta de informações do ambiente, uma ferramenta de visualização foi construída, sendo descrita na Seção 5.5.2.

5.5.1 Arquitetura

Com o aumento no acesso aos ambientes computacionais em nuvem e sua facilidade de utilização, baseada no modelo de pagamento por uso, é natural que a quantidade de usuários e suas respectivas cargas de trabalho também cresçam. Como consequência, os provedores devem ampliar seus recursos e manter o nível de qualidade acordado com os clientes, sob pena de quebras do *Service Level Agreement* (SLA) e decorrentes multas.

O monitoramento de recursos computacionais, como CPU, memória, disco e rede, se torna essencial tanto para os provedores, os quais disponibilizam os serviços, quanto para seus clientes. Uma maneira de se avaliar um ambiente é monitorando algum aspecto, como utilização de recursos. Muitas vezes a elasticidade está associada a algum recurso do provedor. Por exemplo, na Amazon EC2, o cliente pode estar utilizando uma instância com poucos recursos, monitorado através serviço de monitoramento (Cloud Watch), e em caso de necessidade de recursos, o serviço de escalonamento (Auto Scaling), através do serviço de balanceamento de carga (Elastic Load Balance), pode incrementar a quantidade de instâncias. Uma maneira de monitorar aplicações em nuvem de modo mais efetivo é utilizar mecanismos de Computação Autonômica, através dos quais recursos são adicionados e removidos do ambiente conforme limiares de uso pré-estabelecidos.

Um sistema autônomo ou autonômico é composto por um conjunto de elementos autônomos. Um elemento autônomo é um componente responsável pela gestão do seu próprio comportamento em conformidade com políticas, e por interagir com outros elementos autônomos que fornecem ou consomem serviços computacionais (KEPHART; CHESS, 2003). Mecanismos de Computação Autonômica, como *loops* de controle e regras, podem ser empregados no monitoramento de uma nuvem computacional. Assim recursos podem ser adicionados e removidos do ambiente conforme limiares de uso pré-estabelecidos. Esse tipo de estratégia de monitoramento está diretamente associado a uma das principais características da Computação em Nuvem: a elasticidade.

Diversas arquiteturas para soluções de provisionamento e manutenção de SLA utilizando recursos de Computação Autonômica em ambientes de Computação em Nuvem tem sido propostas (REGO et al., 2011; TORDSSON et al., 2012; EMEAKAROHA et al., 2012; BUYYA; CALHEIROS; LI, 2012; KOUKI; LEDOUX, 2013; URIARTE; WESTPHALL, 2014). Porém, devido a grande diversidade de tecnologias, pouca disponibilidade de informação acerca de sua instalação e configuração corretas, do ponto de vista experimental em geral não é trivial implementar tais arquiteturas, muito menos aplicá-las em ambientes de nuvens computacionais.

Como solução para o provimento da elasticidade em Computação em Nuvem baseado em conceitos da Computação Autonômica, uma arquitetura foi proposta utilizando principalmente *loops* de controle, coletores, atuadores e regras, conforme Kephart e Chess (2003). Em geral, aspectos de auto configuração foram considerados.

A arquitetura proposta e seus componentes estão descritos na Figura 5.3 juntamente com seus relacionamentos. Em geral, cargas de trabalho são geradas por diferentes fontes, tais como usuários reais, *benchmarks*, traços computacionais e aplicações. Essa carga de trabalho, dependendo de seu objetivo, pode ser aplicada apenas à máquina virtual do balanceador de carga, ou nas demais máquinas virtuais da infraestrutura, gerando assim uma concorrência por recursos. Uma vez que o balanceador de carga recebe requisições, estas são distribuídas às máquinas virtuais alocadas. Localmente em cada máquina virtual os coletores armazenam dados no banco de dados conforme definido no *loop* de controle local. O *loop* de controle global gerencia todo o fluxo da arquitetura, acionando serviços de predição e atuação quando necessários. Ele também é responsável por monitorar se os valores consolidados conforme alguma técnica estão ultrapassando algum dos limites definidos pelas regras, e assim acionar alguma ação pré-definida para ajuste da alocação dos recursos.

1. *Loop* de Controle Global: responsável por organizar todas as atividades no ambiente relacionadas à elasticidade. O *loop* de controle global gerencia o comportamento do sistema como um todo e define como as informações impactarão nas adaptações locais. Ele atua modificando o comportamento atual do ambiente de tal modo que ele possa se adaptar às necessidades impostas pelas cargas de trabalho. É o gerente do ambiente. Dispara eventos de coleta, predição, análise e ações de adição e remoção de recursos;

2. Coletor: mecanismo para coletar dados das máquinas virtuais do ambiente. Recupera dados de diversos recursos e armazena no banco de dados;

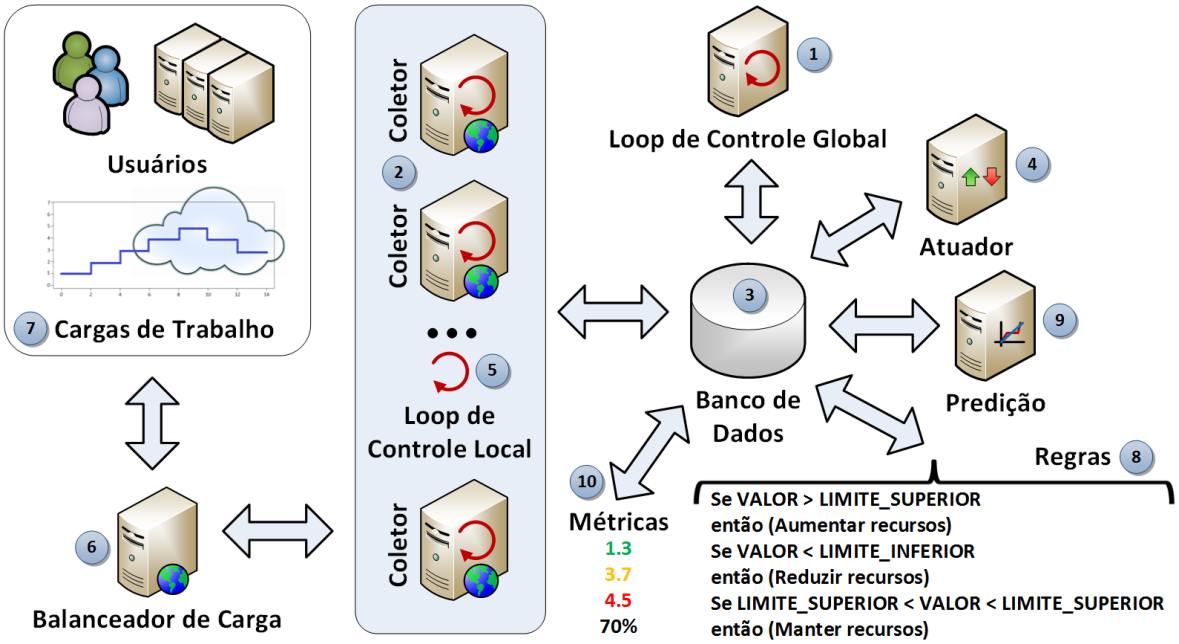


Figura 5.3: Arquitetura autonômica genérica para elasticidade em Computação em Nuvem

3. Banco de Dados: repositório de informações do ambiente. Armazena dados, métricas e configurações do ambiente para posterior consulta pelos demais componentes e serviços;

4. Atuador: responsável por executar ações no ambiente de adição e remoção de máquinas virtuais no balanceador de carga, assim como sua criação / remoção, conforme estratégia de alocação de recursos estabelecida;

5. *Loop* de Controle Local: mecanismo que gerencia as operações locais de coleta e consolidação de dados nas máquinas virtuais. O *loop* de controle local gerencia localmente o comportamento de elementos individuais do sistema. Dessa maneira, ele não possui visibilidade do comportamento global, atendendo apenas a objetivos ou funções locais;

6. Balanceador de Carga: mecanismo que provê a distribuição das requisições entre as máquinas virtuais alocadas ao ambiente. Pode ser uma ferramenta própria de balanceamento de carga ou implementado em alguma linguagem de programação;

7. Cargas de Trabalho: carga de trabalho aplicada ao ambiente, que pode ser gerada por usuários, aplicações, traços computacionais ou por *benchmarks* sintéticos. Podem possuir taxas pré-definidas ou serem aleatórias;

8. Regras: condições que definem limites a serem monitorados de recursos do ambiente, definidos em níveis. Limiares de qualidade podem possuir limites superiores, inferiores e intermediários, onde dependendo dos valores coletados e do nível associado, ações serão executadas sobre a infraestrutura;

9. Predição: mecanismo baseado em alguma técnica, que pode ser estatística, de predição ou mesmo uma situação de interesse para o sistema executado sobre a infraestrutura, como

média móvel e regressão multilinear, que procura prever eventos de adição ou remoção de recursos do ambiente, com a intenção de evitar quebras no SLA e ociosidade.

10. Métricas: medem os recursos do ambiente quanto à utilização. São utilizadas nas regras de definição dos limiares, e obtidas da infraestrutura pelos componentes coletores.

Diversos aspectos desta arquitetura são flexíveis do ponto de vista de implementação. Por exemplo, máquinas virtuais podem ser pré-alocadas ou instanciadas dinamicamente. Situações como essa não afetam a arquitetura, pois o balanceador de carga utiliza as máquinas virtuais disponíveis no ambiente. O que basicamente se modifica é a maneira como o *loop* de controle global gerencia a alocação, criação e remoção de máquinas virtuais de maneira dinâmica, e a duração das operações envolvidas.

Apesar da Figura 5.3 não discriminar o tipo da nuvem, ela pode ser aplicada a qualquer uma delas, ou seja, é independente do tipo da nuvem (privada, pública, comunidade ou híbrida). Dessa maneira, a arquitetura é flexível a ponto de utilizar qualquer tecnologia e infraestrutura de nuvem. A arquitetura descrita na Figura 5.4 exibe uma nuvem privada e nuvem pública em conjunto, demonstrando a aplicação da proposta de arquitetura para uma nuvem híbrida.

Além disso, os eventos utilizados para acionar ações de elasticidade podem utilizar

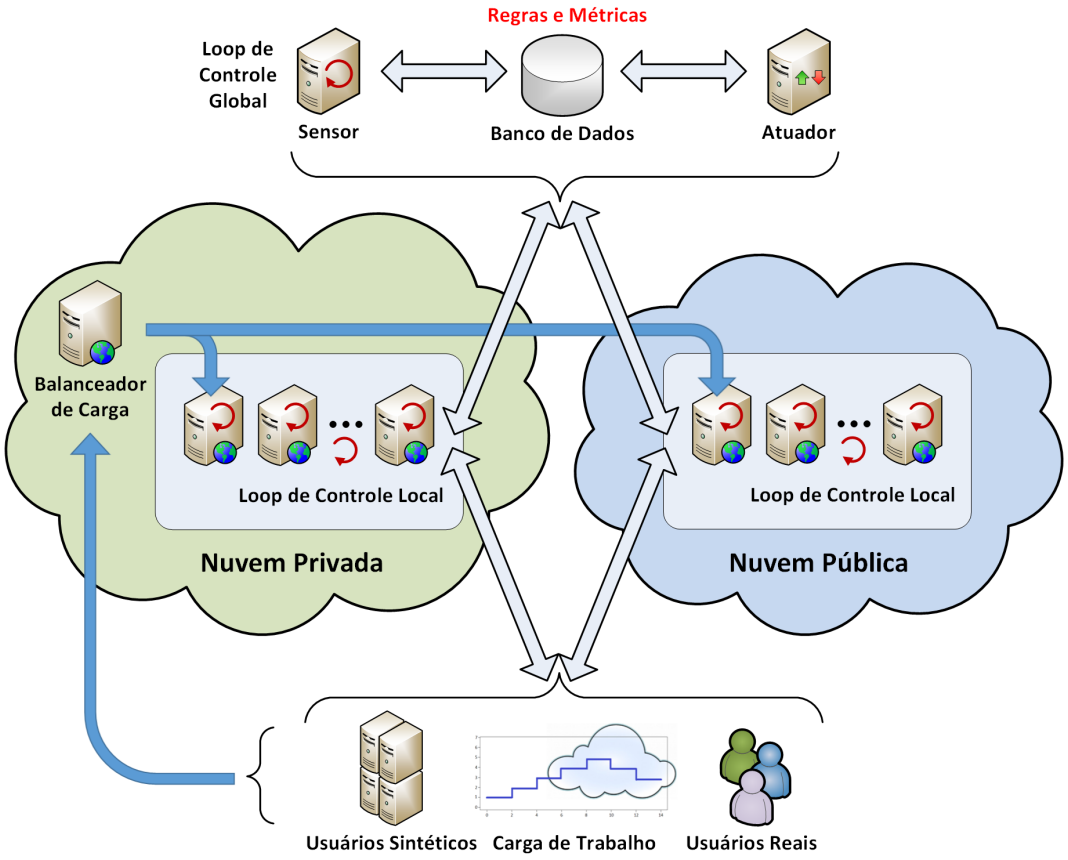


Figura 5.4: Arquitetura autonômica para elasticidade em Computação em Nuvem com a utilização de nuvens privadas e públicas (nuvem híbrida)

qualquer estratégia, como a geração de máquinas virtuais em tempo de execução (dinamicamente), réplicas de máquinas virtuais, ou migração de máquinas virtuais.

Para apoiar a implementação dos componentes da arquitetura proposta, alguns algoritmos em alto nível foram projetados. Considerando a infraestrutura representada pela tupla $I =< M, R, V >$, onde M é o conjunto de métricas a serem monitoradas e utilizadas para a criação de regras, R é o conjunto de regras, e V é o conjunto de valores coletados das métricas. Também considerando para a representação de uma regra a tupla $R =< m, o, f, a >$, onde m é a métrica utilizada na regra, o é o operador aplicado na regra $(=, >, <, \leq, \geq)$, f é o valor de referência utilizado como limiar na regra, e a é a ação a ser tomada em caso da regra ser verdadeira, ou seja, em caso de violação da regra. Por fim, consideramos $V =< m, v >$, onde m é uma métrica e v é o seu valor coletado.

O Algoritmo 1 descreve o pseudocódigo para uma implementação do mecanismo de coleta e *log* de dados. A função $coletarDados(m)$ deve ser implementada para coletar os dados das métricas que serão utilizadas nas regras, como *scripts* para obtenção do percentual de CPU utilizado. A função $formatarDados(dado)$ deve implementar mecanismos para a padronização dos dados, deixando a informação de maneira comum a todos os procedimentos da arquitetura que necessitem dessa informação, como data da coleta da métrica. O procedimento $salvarDados(dadoFormatado, arquivo, tipo)$ deve ser implementado para armazenar os dados no banco de dados (parâmetro *arquivo*), podendo ser, por exemplo, arquivos texto ou um banco de dados relacional (parâmetro $tipo$). O procedimento $logarDados(M)$ pode ser implementado como um agente nas máquinas virtuais.

Algoritmo 1 Procedimento para *log* de dados

```
1: procedure LOGARDADOS(M)
2:     for all m ∈ M do
3:         dado ← coletarDados(m)
4:         dadoFormatado ← formatarDados(dado)
5:         salvarDados(dadoFormatado, arquivo, tipo)
6:         V(m) ← dadoFormatado
7:     end for
8: end procedure
```

O Algoritmo 2 descreve o pseudocódigo para uma implementação do mecanismo de verificação das regras, cujo objetivo é identificar se limites pré-definidos estão sendo violados. A função $identificarValor(r.m, V)$ deve ser implementada para a obtenção do valor calculado da métrica utilizada na regra. A função $verificarViolacao(r, valor)$ deve implementar um mecanismo para a comparação do valor coletado da métrica da regra com o valor de referência, e identificar uma violação do SLA. O procedimento $executarAcao(r.a)$ deve ser implementado para acionar as ações definidas para eventos de adição ou remoção de recursos.

Algumas considerações devem ser feitas em relação aos limites máximo e mínimo de recursos do ambiente. No caso do limite máximo de recursos ser atingido, duas opções existem: (1) parar o incremento de recursos, e não ampliar mais a capacidade porque não existe mais essa possibilidade; (2) utilizar uma solução externa, como uma nuvem pública, híbrida

Algoritmo 2 Procedimento para verificação das regras

```
1: procedure VERIFICARREGRAS(R, M, V)
2:     for all r ∈ R do
3:         valor ← identificarValor(r.m, V)
4:         if verificarViolacao(r, valor) = true then
5:             executarAcao(r.a)
6:         end if
7:     end for
8: end procedure
```

ou migrar máquinas virtuais. No caso do limite mínimo do ambiente ser atingido, ou seja, não tendo mais como diminuir os recursos, a solução é parar a remoção dos recursos, permanecendo na alocação mínima. Ambas situações são potenciais momentos para a ocorrência de violações de SLA, e são dependentes da estratégia de elasticidade empregada.

O Algoritmo 3 descreve o pseudocódigo para uma implementação do gerente global, representado pelo *loop* de controle global. A função $verificarPredicao(V)$ deve implementar um mecanismo para a predição das quebras de SLA baseada nos valores coletados e nas regras, e assim acionar ou não as ações de elasticidade. Neste caso, o procedimento $executarAcao()$ deve ser implementado de forma a agir de maneira a antecipar a adição ou remoção dos recursos. O procedimento $verificarRegras(R,M,V)$ é o mesmo descrito no Algoritmo 2.

Algoritmo 3 Procedimento para controle global

```
1: procedure ACIONARGERENTEGLOBAL
2:     loop
3:         if verificarPredicao(V) = true then
4:             executarAcao()
5:         else
6:             verificarRegras(R, M, V)
7:         end if
8:     end loop
9: end procedure
```

A implementação do gerente global pode levar a um problema de sincronização dos dados. Enquanto a avaliação das regras ocorre, os dados podem ser modificados conforme a carga de trabalho aplicada ao ambiente, já que as máquinas virtuais podem não estar sincronizadas nos tempos de coleta ou terem sido iniciadas em tempos diferentes. Além disso, o envio dos dados para consolidação pode também não estar sincronizado. Estas situações podem tornar a avaliação defasada. Um solução seria a implementação de um gerente global sem eventos de coleta dos dados da infraestrutura ou aplicação, utilizados nas regras, e sim apenas com a consolidação dos dados para ações de predição e validação das regras. A coleta dos dados ficaria nos agentes locais das máquinas virtuais. O gerente global iria apenas capturar as informações armazenadas, e não logar os dados, minimizando problemas de sincronização.

5.5.2 Ferramenta para Visualização Gráfica dos Dados

Para auxiliar na visualização e análise dos dados coletados, uma ferramenta foi construída. Esta ferramenta possibilita a geração de gráficos para as diversas métricas coletadas a partir dos recursos do ambiente. As métricas de elasticidade propostas também podem ser visualizadas por meio da ferramenta, e dessa maneira auxiliar na análise da elasticidade.

5.5.2.1 Estrutura da Aplicação

A aplicação foi desenvolvida com a linguagem de programação Java. Ela é baseada na leitura de arquivos texto com as informações coletadas a partir do ambiente. Assim é possível a leitura de arquivos texto gerados por meio dos mecanismos propostos na arquitetura. Esses arquivos registram dados do ambiente, como utilização de CPU e memória, além de dados de aplicações, como tempo de resposta de requisições.

A ferramenta está dividida em três pacotes: arquivos de *log*, gráficos e utilitários, além da interface gráfica do usuário. Os *logs* de dados coletados do ambiente e das métricas propostas estão distribuídos em um conjunto de arquivos texto, que podem ser configurados ou estendidos sempre que um novo dado precisar ser coletado ou uma nova métrica for criada. Os arquivos são: alocação, elasticidade, média de consumo de CPU, máquinas virtuais (um para cada máquina virtual) e servidor *web* (um para cada máquina virtual). Cada um desses arquivos possui uma estrutura própria lida pela aplicação. O pacote de gráficos possui um conjunto de classes correspondente a cada tipo de gráfico utilizado pela aplicação. Os utilitários são uma biblioteca de funções para manipulação de datas e arquivos texto. A interface gráfica permite que o usuário escolha o experimento a ser lido, correspondente a um conjunto de arquivos de *log*, a seleção da métrica a ser visualizada e os painéis de gráficos disponíveis. Os arquivos de *log* lidos pela ferramenta possuem a seguinte estrutura descrita na Tabela 5.1:

Tabela 5.1: Estrutura dos arquivos de *log* lidos pela ferramenta

Arquivo	**Estrutura**
Alocação	Data da coleta | Mantém (m), Aumenta (a), ou Diminui (d) | Quantidade de máquinas virtuais alocadas
Média de Consumo de CPU	Data da coleta | Média do valor do percentual de CPU em todas as máquinas virtuais
Máquina Virtual	Data da coleta | Percentual de utilização de CPU pelo usuário | Percentual de utilização de CPU pelo sistema | Percentual de CPU livre | Quantidade de memória total | Quantidade de memória utilizada | Quantidade de memória livre | Quantidade de KB lidos em disco | Quantidade de KB escritos em disco | Quantidade de pacotes que chegam em KB | Quantidade de pacotes que saem em KB
Servidor *web*	Data da coleta | IP de origem | URL | Tempo levado para processar a requisição em milissegundos
Elasticidade	Data da coleta | Estresse da Qualidade na Nuvem | Tensão dos Recursos Alocados | Tensão dos Recursos Demandados | Elasticidade dos Recursos Alocados | Elasticidade dos Recursos Demandados | Elasticidade da Demanda

5.5.2.2 Extensibilidade

A ferramenta permite a flexibilização na leitura de arquivos texto de *log*, bastando o usuário implementar uma classe da estrutura definida no arquivo texto. Mecanismos para leitura de arquivos já estão disponibilizados para o usuário, bastando que se projete o arquivo texto baseado em colunas, onde cada coluna possui uma informação coletada do ambiente, e separadas por um "|" (arquivos CSV).

Para a inclusão de uma nova métrica, é necessário projetar um novo arquivo de *log* a ser lido pela ferramenta com dados do ambiente ou com métricas geradas automaticamente pela consolidação de dados. A seguir, uma classe para uma nova métrica deve ser criada, com a leitura do arquivo de *log* correspondente, e configurar os parâmetros do gráfico a ser exibido. Para a criação de novos tipos de gráfico, o usuário deve utilizar gráficos da biblioteca JFreeChart[1], e adaptar a classe correspondente à métrica desejada. Por fim, a disponibilização da métrica na interface do usuário deve ser implementada, para permitir sua seleção.

5.5.2.3 Métricas

A ferramenta já possibilita a visualização de diversas métricas coletadas do ambiente, conforme descritas na Tabela 5.2. Além disso, algumas métricas são disponibilizadas a partir de dados coletados do ambiente, conforme uma fórmula de cálculo pré-definida, como o caso das métricas de elasticidade: Estresse da Qualidade da Nuvem (S_{QN}), Tensão dos Recursos Demandados (T_{RD}), Tensão dos Recursos Alocados (T_{RA}), Elasticidade dos Recursos Demandados (E_{RDi}), Elasticidade dos Recursos Alocados (E_{RAi}) e Elasticidade da Demanda (E_{Di}).

Tabela 5.2: Métricas disponibilizadas pela ferramenta

Métrica	**Descrição**
CPU	Percentual de utilização de CPU do usuário, do sistema e ocioso. Variando de 0% a 100%
Memória	Quantidade de memória livre, quantidade de memória total e quantidade de memória utilizada pelo usuário. Variando de 0 até a quantidade de memória disponível em GBytes
Rede	Pacotes recebidos e pacotes enviados. Em KB/s
Disco	Quantidade de dados lidos e quantidade de dados escritos. Em KB/s.
Tempo de Resposta	Tempo de resposta das aplicações. Em milissegundos (ms)
Média	Média de utilização de CPU de todas as máquinas virtuais, em valores percentuais
Alocação	Alocação das máquinas virtuais ao longo do tempo, conforme necessidade de adição ou remoção de recursos
Elasticidade	Estresse da Qualidade da Nuvem (S_{QN}), Tensão dos Recursos Demandados (T_{RD}), Tensão dos Recursos Alocados (T_{RA}), Elasticidade dos Recursos Demandados (E_{RDi}), Elasticidade dos Recursos Alocados (E_{RAi}) e Elasticidade da Demanda (E_{Di})

Todas as métricas citadas são visualizadas de maneira individual por máquina virtual, podendo o usuário selecionar qual máquina virtual deseja visualizar os dados. Adicionalmente, é possível verificar algumas métricas em paralelo. Para todas as máquinas virtuais é possível se visualizar em paralelo CPU, memória, rede, disco e tempo de resposta. Alocação e

[1] JFreeChart - http://www.jfree.org

média são métricas consolidadas, portanto são do ambiente como um todo. Tensão e estresse são visualizadas em conjunto, para facilitar a análise, assim como E_{RAi}, E_{RDi} e E_{Di}.

5.5.2.4 Telas

A ferramenta desenvolvida possui basicamente um conjunto de painéis de visualização de gráficos, e outro painel para seleção das métricas e seleção dos arquivos de *log* de um determinado experimento, conforme Figura 5.5. Demais telas de utilização da ferramenta, com a visualização das métricas podem ser visualizadas no Apêndice B.

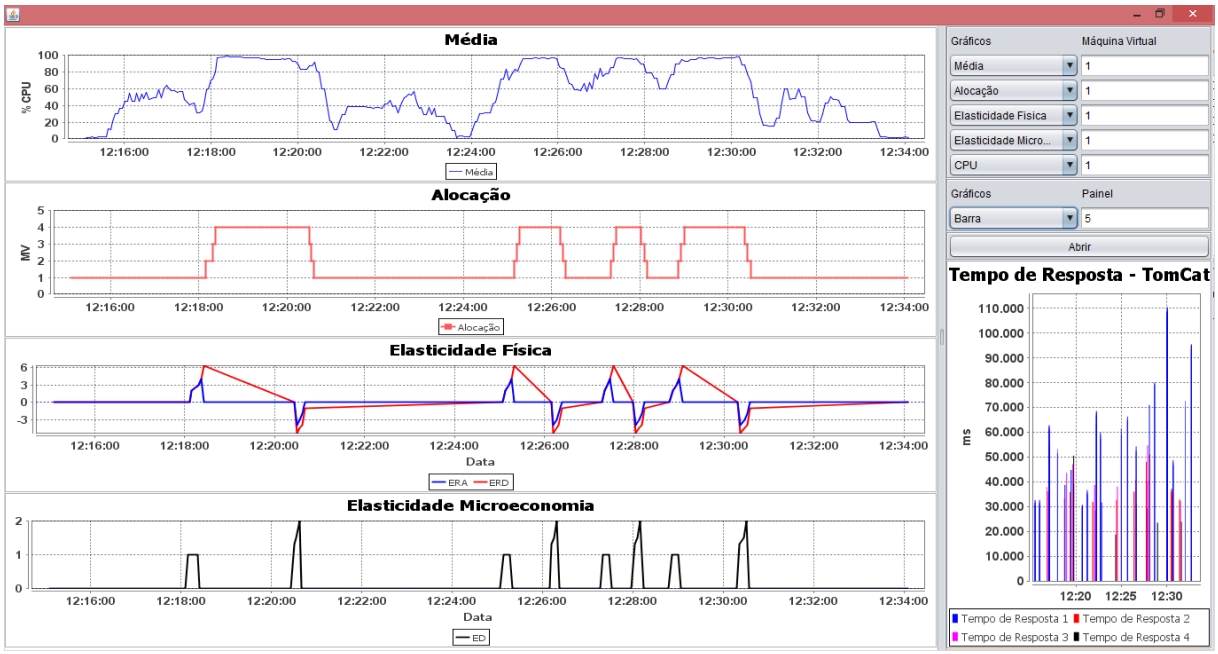


Figura 5.5: Tela típica para uma análise de desempenho com a média de utilização de CPU de todas as máquinas virtuais, a alocação conforme os limiares definidos na infraestrutura, as métricas de elasticidade e o tempo de resposta das requisições de um experimento

6 EXPERIMENTOS

Este capítulo descreve os experimentos realizados para a validação do *framework* FOLE e das métricas propostas. Dois experimentos foram projetados. O primeiro experimento realiza uma análise de desempenho em uma nuvem privada com o apoio do FOLE e utilizando todas as métricas propostas de maneira detalhada (Seção 6.1). O segundo experimento executa uma avaliação da elasticidade em uma nuvem híbrida, também com o apoio do FOLE e com as métricas de elasticidade (Seção 6.2).

6.1 Experimento - Nuvem Privada

O objetivo deste experimento é avaliar o comportamento do ambiente diante de cargas de trabalho dinâmicas e seus efeitos na elasticidade. Este experimento utilizou o FOLE para o projeto de experimentos. A análise de desempenho utilizou todas as métricas propostas e várias métricas relacionadas como complemento.

6.1.1 Material e Métodos

Para a validação do FOLE, dois experimentos foram projetados e executados em um mesmo ambiente de nuvem. O OpenNebula 3.8 foi utilizado para a criação de uma nuvem privada. Não foram utilizados neste trabalho nuvens públicas comerciais. Todas as máquinas físicas são de 5 e 7 núcleos (Ci5 e Ci7), 24 GB de memória RAM, sistema operacional Ubuntu Server 12.04 64 bits e hipervisor KVM. Foram utilizadas quatro máquinas virtuais nos experimentos, todas com 1 VCPU, 1 GB de memória RAM e sistema operacional Ubuntu Server 12.04 64 bits. O Apache Tomcat foi utilizado como servidor *web*, como balanceador de carga o NGINX, e como gerador de cargas de trabalho o HTTPERF. A instanciação do *framework* foi desenvolvida em Java e *shell script*. A representação do *testbed* está descrita na Figura 6.1.

6.1.2 Carga de Trabalho

Destaca-se a geração de cargas de trabalho diretamente na máquina virtual do balanceador de carga, oriundas de navegadores *web* e do HTTPERF, que são distribuídas entre as demais máquinas virtuais, e também cargas de trabalho executadas diretamente nas máquinas virtuais utilizadas pelas aplicações. A ideia foi emular a concorrência pelos recursos em diferentes maneiras de se utilizar um ambiente de Computação em Nuvem. A Figura 6.2 ilustra a carga de trabalho empregada nos dois experimentos sobre a infraestrutura construída. A Tabela 6.1 detalha o projeto, as taxas e quantidade de requisições aplicadas à carga de trabalho.

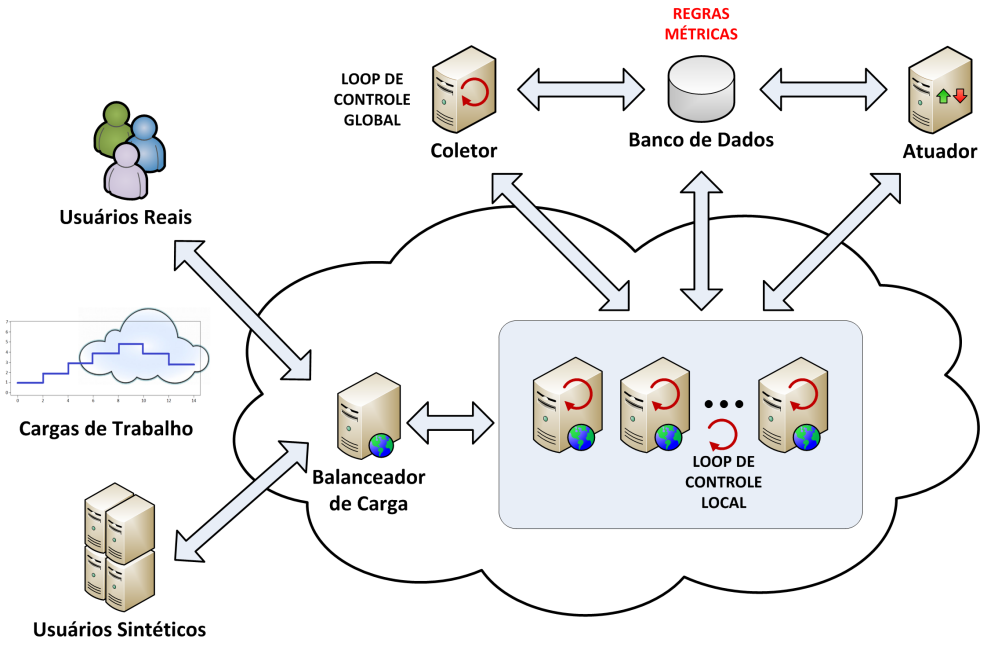


Figura 6.1: Arquitetura e ambiente experimental para o experimento com nuvem privada

6.1.3 Projeto do Experimento - Instanciação do FOLE

O objetivo dos experimentos é avaliar o comportamento de aplicações *web* diante de cargas de trabalho dinâmicas, e como se comportam de maneira autonômica para adaptação a variações de demanda e manutenção do SLA. Para isso, o FOLE será utilizado. Como ele é um *framework* conceitual, suas atividades deverão ser instanciadas para o ambiente e projeto de experimentos. Destaca-se que o objetivo é validar as atividades do FOLE, e não implementar

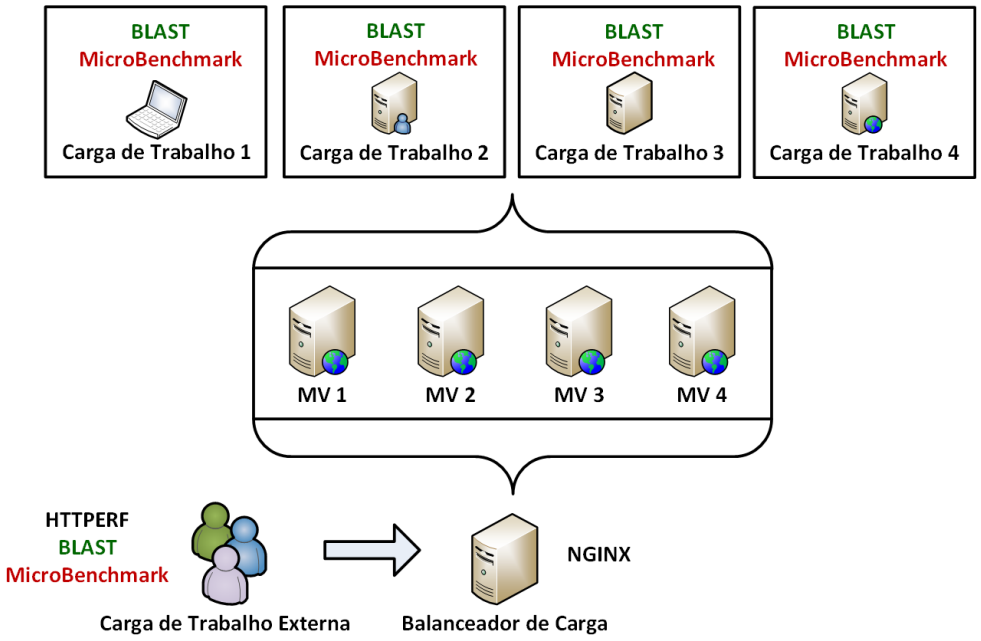


Figura 6.2: Representação da carga de trabalho aplicada ao experimento com nuvem privada

Tabela 6.1: Critérios para avaliação de desempenho para os experimentos

Critérios	**Descrição**
Sistema	Ambiente privado de Computação em Nuvem (OpenNebula)
Métricas	Tempo de resposta das aplicações, percentual de utilização de CPU, percentual de utilização de memória, pacotes recebidos e enviados em KB, KB lidos e escritos em disco; S_{QN}, T_{RD}, T_{RA}, E_{RDi}, E_{RAi}, $\overline{E_{RDi}}$, $\overline{E_{RAi}}$, E_{Di}, $\overline{E_{Di}}$ TAE, $TASu$, $TASo$, TAT, $TRASu$, $TRASo$, $TRAE$; Elasticidade de *Scaling Up* e Elasticidade de *Scaling Down* (HERBST; KOUNEV; REUSSNER, 2013); Relação Custo/Benefício Média (SIMÕES; KAMIENSKI, 2014)
Parâmetros	CPU, memória, sistema operacional, quantidade de máquinas virtuais
Fatores	Configuração do *benchmark* (repetições e tamanho da matriz), configuração do BLAST (tamanho do arquivo, tamanho da consulta), configuração do HTTPERF (quantidade de requisições, taxa de requisições, tempo)
Técnica de Avaliação	Medição
Carga de Trabalho	**Experimento 1: *Microbenchmark* -** Executar multiplicações de matrizes (dimensão 200x200x200) através de um *microbenchmark* construído na linguagem de programação Java, sob a forma de pequenas aplicações *web*, com requisições disparadas através do HTTPERF, com taxas variando de 1 a 5 requisições por segundo e quantidade de conexões variando em 10, 30 e 50; **Experimento 2: BLAST -** Executar uma aplicação científica (BLAST), através de um *microbenchmark* construído na linguagem de programação Java, sob a forma de pequenas aplicações *web*, com requisições disparadas através do HTTPERF, com taxas variando de 1 a 3 requisições por segundo e quantidade de conexões variando em 3, 6 e 9
Projeto de Experimentos	Geração da carga de trabalho com o HTTPERF no servidor *web* para que os recursos dinamicamente se adaptem às necessidades de CPU. O primeiro experimento utiliza um *microbenchmark* e o segundo uma aplicação de computação científica
Análise dos Dados	Interpretação dos resultados descritos nos gráficos de média do percentual de CPU, gráficos de alocação das máquinas virtuais, gráficos de elasticidade e tabelas com métricas coletadas
Apresentação dos Resultados	Gráficos de linha, gráfico de bolhas e tabelas

soluções ótimas. A intenção é executar todas as atividades do *framework*. A Tabela 6.1 detalha itens do projeto experimental, tais como métricas e cargas de trabalho.

As ferramentas foram definidas na infraestrutura. Para a macroatividade "Planejar Suporte", arquivos texto registram o *log* das operações de coletadas. O *log* gerado para cada máquina virtual em arquivo texto contém a data da coleta, valores de utilização de CPU, memória, disco e rede. Também foi gerado um arquivo de *log* para a média de utilização de CPU e alocação de recursos, permitindo sua leitura em uma ferramenta construída especificamente para a visualização gráfica de todas as métricas de maneira individual (métricas coletadas de cada máquina virtual) e em conjunto (métricas coletadas de todas as máquinas virtuais ao mesmo tempo). A consolidação dos dados ocorreu por meio da leitura dos arquivos de *log* e a partir desses dados decisões foram tomadas. O dados coletados são apresentados sob a forma de gráficos de linha, a serem disponibilizados através da aplicação de visualização. O intervalo de coletas definido foi de 1 segundo adicionado do custo da coleta e análise dos resultados.

Para a avaliação da elasticidade, utilizamos métricas diretamente e indiretamente associadas à elasticidade, descritas na Tabela 6.1. Utilizamos métricas baseadas em tempo de execução de operações: Tempo de Alocação Sobreprovisionado (*TASo*), correspondente ao

tempo utilizado em operações de remoção de recursos; Tempo de Alocação Subprovisionada (*TASu*), utilizado para medir o tempo de operações de adição de recursos; Tempo de Alocação Estabilizada (*TAE*), tempo no qual não há adição ou remoção de recursos; e Tempo de Alocação Transitória (*TAT*), correspondente ao tempo em que os efeitos da adição ou remoção de recursos ainda não impactaram no ambiente. Também utilizamos métricas baseadas na utilização de recursos: Total de Recursos Alocados Subprovisionados (*TRASu*), correspondente à quantidade de recursos alocados em um estado subprovisionado, mas não estabilizado; Total de Recursos Alocados Sobreprovisionados (*TRASo*), indicando a quantidade de recursos alocados em um estado sobreprovisionado, mas não estabilizado; e o Total de Recursos Alocados Estabilizados (*TRAE*), que corresponde à quantidade de recursos em um estado estabilizado. Para a medição específica da elasticidade foram utilizadas as métricas propostas neste trabalho, baseadas em conceitos da Física e da Microeconomia. Como análise complementar, utilizamos as métricas Elasticidade de *Scaling UP* e Elasticidade de *Scaling Down*, para avaliar a velocidade na qual um ambiente realiza tais operações, e a relação custo/benefício média.

As cargas de trabalho foram aplicadas diretamente na máquina virtual do balanceador de carga, onde requisições geradas por navegadores *web* e pelo HTTPERF, que são distribuídas entre as demais máquinas virtuais. Além disso, também foram aplicadas cargas de trabalho diretamente nas máquinas virtuais utilizadas pelas aplicações. A ideia é emular a concorrência pelos recursos em diferentes maneiras de se utilizar um ambiente de Computação em Nuvem.

Para a macroatividade "Planejar Elasticidade", um mecanismo baseado na arquitetura de Computação Autonômica proposta foi utilizado. Uma estratégia de elasticidade horizontal foi aplicada, através da qual à medida que recursos são necessários, novas instâncias de máquinas virtuais são adicionadas, por meio de um balanceador de carga, e retiradas caso não sejam mais necessárias. Para a avaliação da elasticidade foram utilizadas as métricas propostas neste trabalho, baseadas em conceitos da Física e da Microeconomia, além das demais métricas que indiretamente avaliam a elasticidade e métricas de trabalhos relacionados, citadas na Tabela 6.1. A métrica utilizada para disparar ações de elasticidade (gerar eventos de adição/remoção de recursos) foi a média do percentual de utilização de CPU das máquinas virtuais. Os limiares utilizados para a execução de alguma ação de elasticidade no ambiente foram: acima de 90% (aloca uma nova máquina virtual), abaixo de 80% (desaloca uma máquina virtual), e entre 80% e 90% (mantém alocação). Esses valores foram definidos considerando a premissa que a situação ideal de consumo de CPU das máquinas virtuais do ambiente deveria ser quase em sua totalidade, maximizando a utilização da capacidade de processamento. O valor a ser comparado com os limiares foi calculado como a média das 10 últimas coletas de utilização de CPU nas máquinas virtuais. Esse valor foi utilizado apenas para prover uma resposta mais rápida para a análise, mas o ideal em ambientes reais é que ele seja maior.

Como mecanismo de predição foi utilizado regressão multilinear sobre valores de utilização de CPU, memória, disco e rede, coletados em experimentos prévios, com cargas de trabalho semelhantes. Esse cálculo é realizado para decidir se uma ação de alocação ou desalocação deve ser executada ou não, antes mesmo do cálculo da média.

Para o provisionamento dos recursos, a estratégia de um balanceamento de carga

foi utilizada, onde máquinas virtuais são adicionadas conforme a necessidade. Um conjunto de *scripts* foi desenvolvido para as ações e estratégias de análise.

As macroatividades “Inicializar Serviços da Análise de Desempenho” e “Executar Análise de Desempenho” foram implementadas através de *scripts* que inicializavam os serviços de coleta, análise, geração das cargas de trabalho (operações manuais com o HTTPERF), consolidação dos dados, além de analisar e disparar ações.

6.1.4 Experimentos

Nesta seção os experimentos projetados são descritos. As Subseções 6.1.4.1 e 6.1.4.2 descrevem experimentos que aplicam diferentes cargas de trabalho ao ambiente. A Subseção 6.1.5 exibe as métricas propostas calculadas e métricas calculadas de trabalhos relacionados. Por fim, uma discussão dos resultados é apresentada na Subseção 6.1.6.

6.1.4.1 Experimento 1 - *Microbenchmark*

A carga de trabalho utilizada neste experimento foi projetada para executar operações de multiplicação de matrizes, com muitas operações de CPU e memória, por meio de um *microbenchmark*. O tempo de resposta das aplicações geradas foi relativamente curto, variando em algumas centenas de milissegundos. A duração deste experimento foi de 12min58s. A Figura 6.3 ilustra o tempo de resposta das requisições para cada uma das quatro máquinas virtuais envolvidas no experimento, a utilização média de CPU de todas as máquinas virtuais e a alocação das máquinas virtuais.

Muitas variações no consumo de CPU ocorreram devido à característica do *microbenchmark*, uma vez que alocações e desalocações eram disparadas quando os limiares eram atingidos. Para memória o comportamento foi quase constante, estando em todas as máquinas virtuais quase sempre em 100%.

Muitas variações no tempo de resposta das aplicações foram identificadas, principalmente na máquina virtual 1, onde vários picos ocorreram. Muitas vezes, estes picos ocorreram devido ao fato que muitas requisições estavam represadas no servidor da máquina virtual 1, e demoraram muito a serem finalizadas. Além disso, o tempo de resposta nas demais máquinas virtuais variou pouco devido à estratégia de balanceamento de carga inicialmente utilizar a máquina virtual 1, enquanto as demais foram utilizadas apenas quando recursos eram necessários. Nas demais máquinas virtuais o comportamento foi mais constante, o que implica em uma distribuição de requisições conforme a necessidade.

Os gráficos de utilização média de CPU e alocação de máquinas virtuais mostram métricas diretamente associadas à elasticidade. Identificou-se que quando o SLA era violado, novas instâncias de máquinas virtuais eram alocadas ao balanceador de carga, e quando a utilização da CPU era inferior ao limite estabelecido, havia uma desalocação de máquinas virtuais.

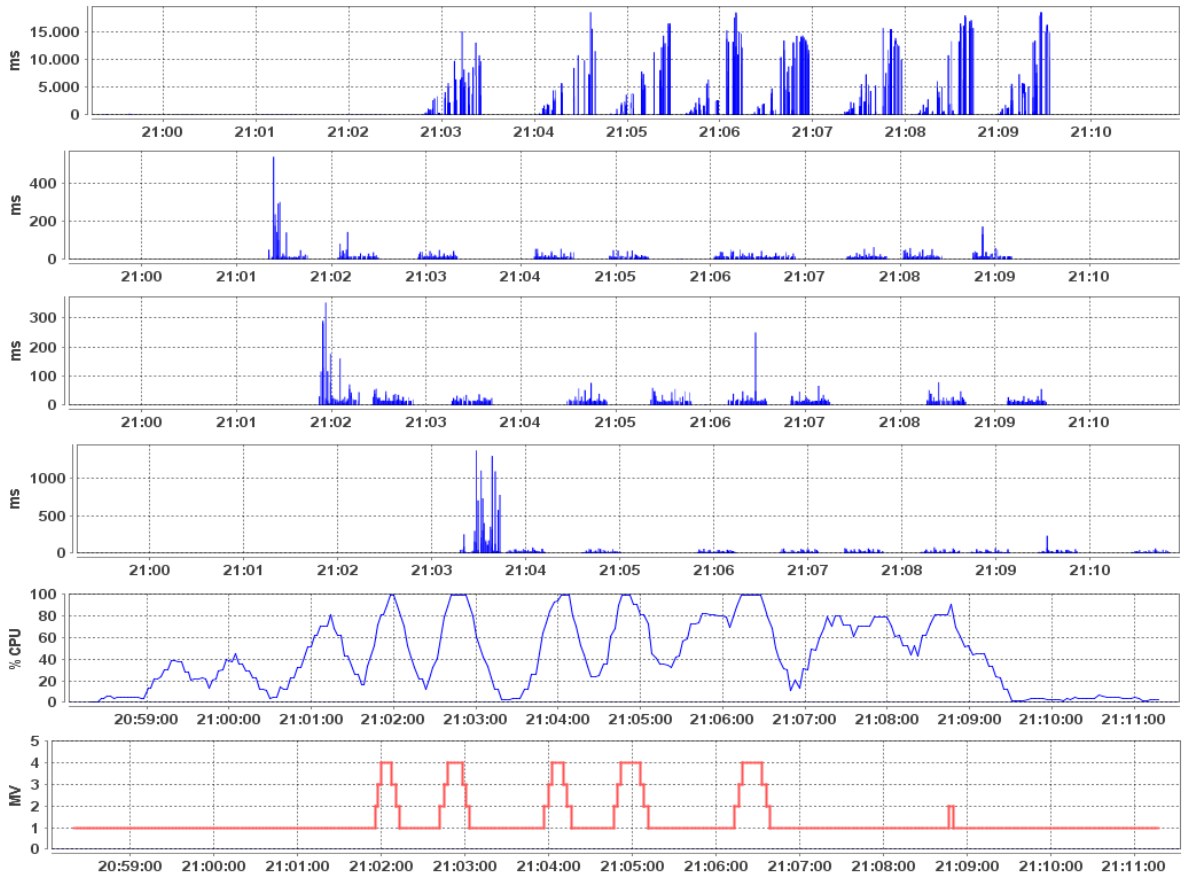


Figura 6.3: Tempo de resposta das requisições para cada uma das quatro máquinas virtuais (milissegundos), utilização média de CPU (%) e alocação/desalocação das máquinas virtuais para o Experimento 1

6.1.4.2 Experimento 2 - BLAST

A carga de trabalho neste experimento foi projetada para executar uma aplicação científica, BLAST (NCBI, 2013), orientada a CPU, memória e disco. O BLAST é uma ferramenta de similaridade muito utilizada para comparação e análise de sequências de proteínas. Ele é um algoritmo otimizado para velocidade utilizado para buscar em bancos de dados sequências de alinhamentos locais de uma consulta. Como o BLAST é uma aplicação de computação científica, é esperado que seu consumo de CPU seja alto. O tempo de resposta das aplicações geradas foi mais longo em relação ao experimento anterior, variando em alguns milhares de milissegundos. A duração do experimento foi de 19 minutos. A Figura 6.4 ilustra o tempo de resposta das requisições para cada uma das quatro máquinas virtuais envolvidas no experimento, a utilização média de CPU e a alocação das máquinas virtuais.

Assim que a primeira máquina virtual executava requisições, sua utilização de CPU aumentava, e ao superar o limite de qualidade estabelecido, novas instâncias foram alocadas. Para memória e acesso a disco o comportamento foi quase constante, porém a memória em todas as máquinas virtuais permaneceu quase sempre em 100%, semelhante ao Experimento 1. Espera-se que em aplicações melhores estruturadas com o BLAST, a leitura em disco seja

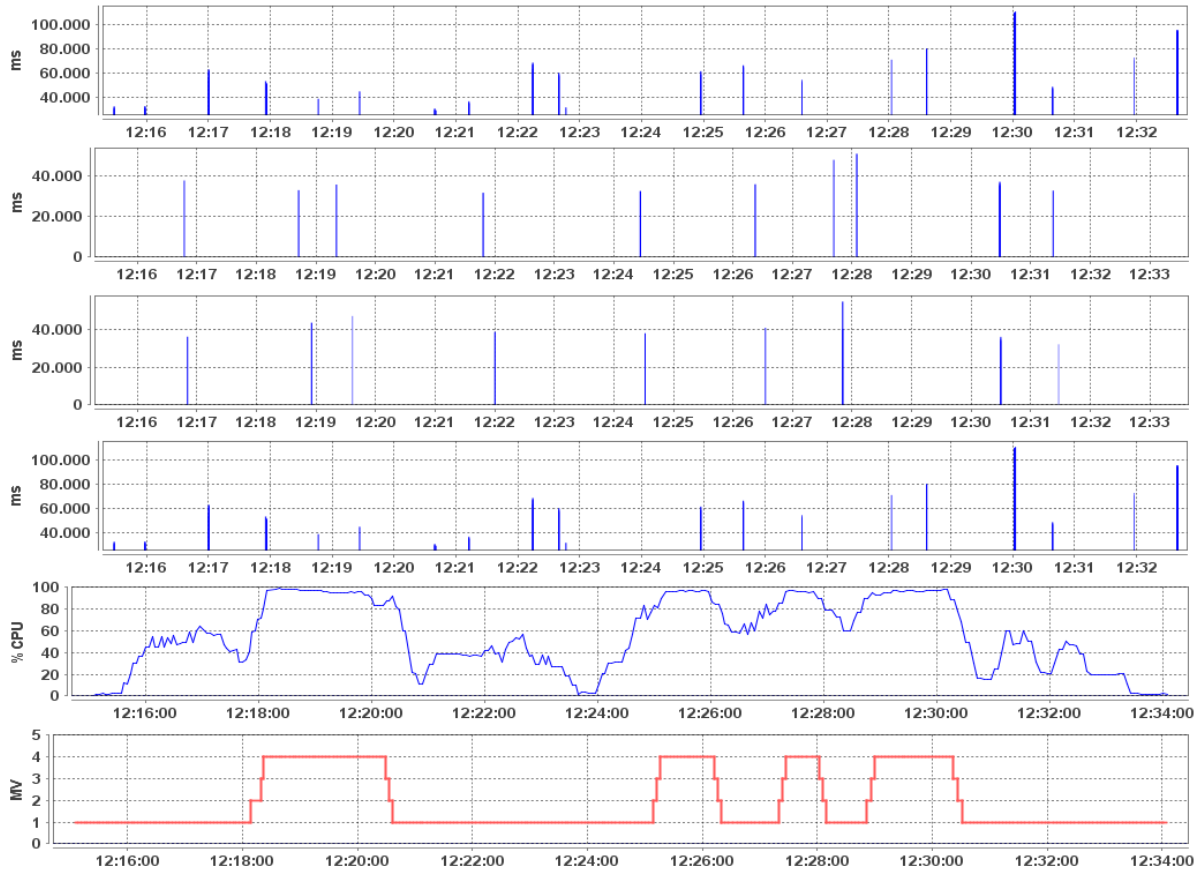


Figura 6.4: Tempo de resposta das requisições para cada uma das quatro máquinas virtuais (milissegundos), utilização média de CPU (%) e alocação/desalocação das máquinas virtuais para o Experimento 2

maior, pois ocorre uma pesquisa em disco.

Em relação ao tempo de resposta das aplicações, poucas variações ocorreram nas máquinas virtuais, diferente do Experimento 1, onde a maioria das requisições e maiores tempos de resposta ocorreram na máquina virtual 1. Neste experimento, as requisições e os tempos de resposta foram melhores distribuídos nas máquinas virtuais. Em picos de utilização de CPU as 4 máquinas virtuais foram alocadas, e em geral há uma desalocação logo após. Isso se explica devido à carga de trabalho empregada, que parava de enviar requisições por um intervalo de tempo, período em que as requisições alocadas eram finalizadas.

Em intervalos onde somente havia uma máquina virtual alocada, observou-se algumas requisições nas demais máquinas virtuais, nos gráficos de tempos de resposta. Isto ocorreu por causa da estratégia de elasticidade horizontal aplicada, e todas as máquinas virtuais já estavam funcionalmente ativas. Além disso, a carga de trabalho projetada gerava requisições locais.

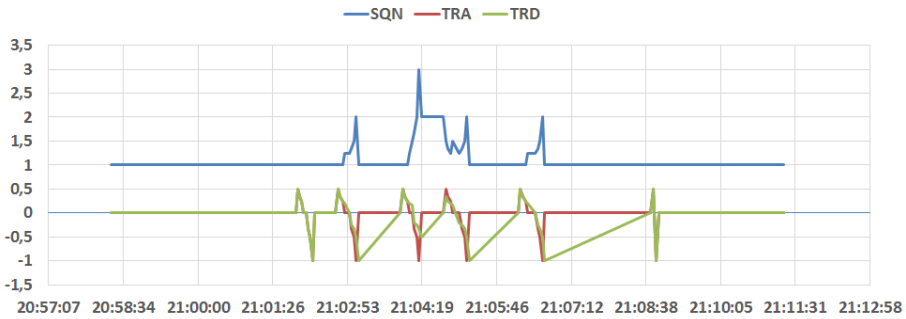
6.1.5 Métricas

O cálculo das métricas propostas foi realizado para os dois experimentos. Todas as métricas propostas foram calculadas e consolidadas em tabelas e gráficos, para suas validações

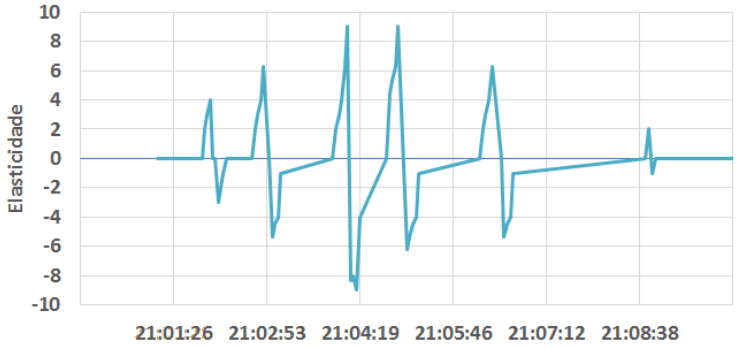
e comparação entre si. Além disso, métricas propostas na literatura também foram calculadas, para fim de comparação e análise complementar. Ao final desta seção, uma discussão mais inter-relacionada dos resultados é apresentada.

6.1.5.1 Métrica 1 - Física

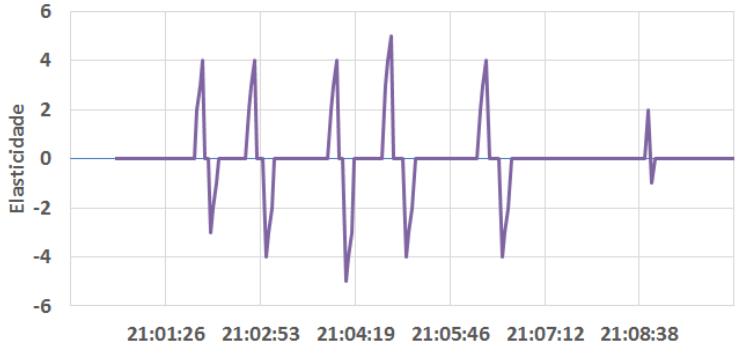
Os valores de S_{QN}, T_{RD}, T_{RA}, E_{RDi} e E_{RAi} calculados para os Experimentos 1 e 2 podem ser verificados nas Figuras 6.5 e 6.6. Durante o Experimento 1, a elasticidade resultou em 12 picos. E_{RDi} obteve valores na faixa de -9 a 9, variando em 18 unidades entre valores



(a) Estresse da Qualidade na Nuvem (S_{QN}), Tensão dos Recursos Demandados (T_{RD}) e Tensão dos Recursos Alocados (T_{RA})

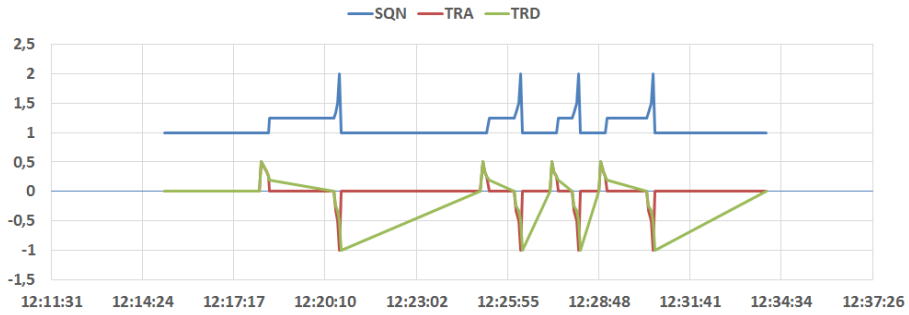


(b) Experimento 1 - E_{RDi}

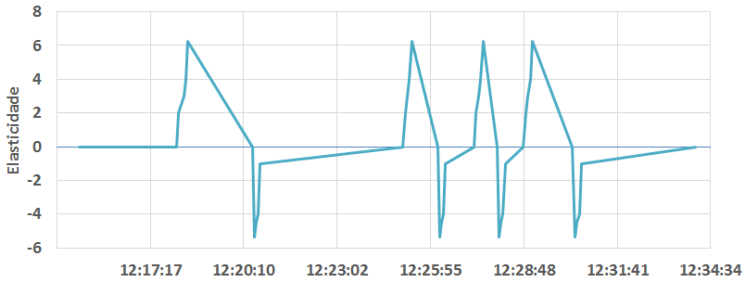


(c) Experimento 1 - E_{RAi}

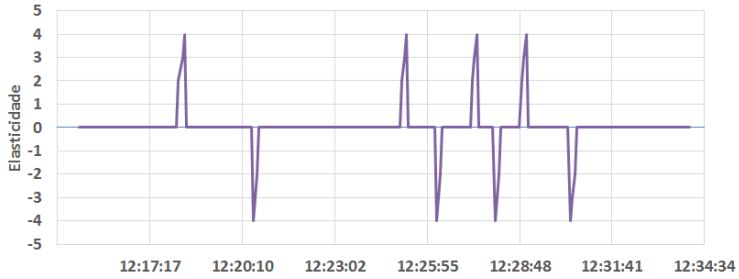
Figura 6.5: Experimento 1 - *Microbenchmark*: Estresse da Qualidade na Nuvem (S_{QN}), Tensão dos Recursos Demandados (T_{RD}), Tensão dos Recursos Alocados (T_{RA}), Elasticidade dos Recursos Demandados (E_{RDi}) e Elasticidade dos Recursos Alocados (E_{RAi}).



(a) Estresse da Qualidade na Nuvem (S_{QN}), Tensão dos Recursos Demandados (T_{RD}) e Tensão dos Recursos Alocados (T_{RA})



(b) Experimento 2 - E_{RDi}



(c) Experimento 2 - E_{RAi}

Figura 6.6: Experimento 2 - Blast: Estresse da Qualidade na Nuvem (S_{QN}), Tensão dos Recursos Demandados (T_{RD}), Tensão dos Recursos Alocados (T_{RA}), Elasticidade dos Recursos Demandados (E_{RDi}) e Elasticidade dos Recursos Alocados (E_{RAi}).

mínimos e máximos. E_{RAi} resultou em valores de -5 a 5, variando em 10 unidades. Para o Experimento 2, a quantidade de picos foi menor, resultando em apenas 8 picos. A variação de E_{RDi} foi de -5.3 a 6.25, variando em 11.5 unidades entre valores mínimos e máximos. Para E_{RAi} a faixa de valores foi de -4 a 4, variando em 8 unidades.

Esse comportamento se deve ao fato que a carga de trabalho do Experimento 1 era composta por pequenas aplicações (menores e mais rápidas), porém maior em quantidade, enquanto que no Experimento 2 as aplicações eram maiores e mais demoradas, e em quantidades menores. Essa variação entre os valores máximos e mínimos indica que se utilizou muitos recursos ou existe a necessidade por mais recursos.

O estresse (S_{QN}) respeita a carga de trabalho aplicada ao ambiente no sentido que variações na carga que impliquem em um consumo de CPU próximas ao SLA estabelecido influenciam no S_{QN}. Há a tendência de aumentar seu valor se a carga de trabalho imposta

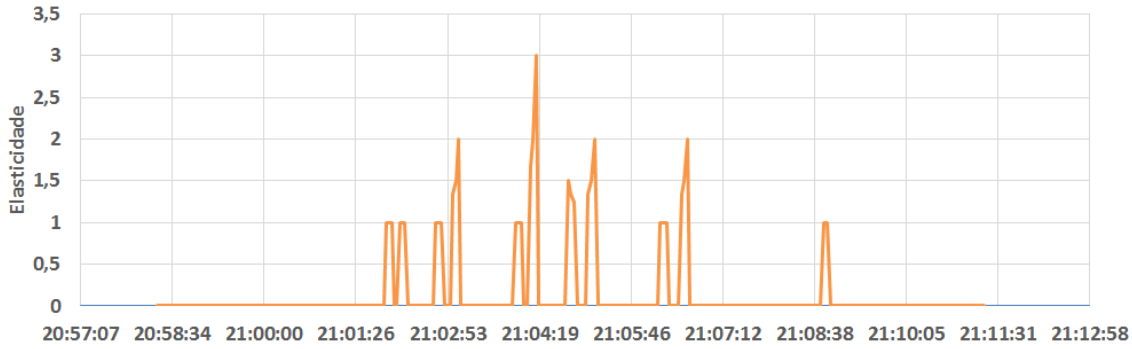
aumentar. Momentos curtos de variação na carga de trabalho e no consumo de CPU não impactam tanto em S_{QN}, implicando em S_{QN} menor e mais constante. Para o contexto definido, S_{QN} sempre será maior ou igual a 1 devido à suposição de sempre haver pelo menos uma máquina virtual alocada. A tensão (T_{RD} e T_{RA}) também não é muito influenciada pelas variações de carga de trabalho, pois mantêm os valores próximos uns dos outros, mas acompanha as variações.

A elasticidade média dos recursos foi calculada, e teve como resultado para $\overline{E_{RAi}}$ e $\overline{E_{RDi}}$ igual a 0.07 e 0.06, respectivamente para o Experimento 1, e 0.00 e 0.03 respectivamente para o Experimento 2. Esse valor $\overline{E_{RDi}}$ igual a 0.00 para o Experimento 2 destacou-se, porém foi calculado da mesma maneira, reforçando que na média o Experimento 1 é mais elástico que o Experimento 2, isto é, mais sensível à mudanças impostas pelas cargas de trabalho.

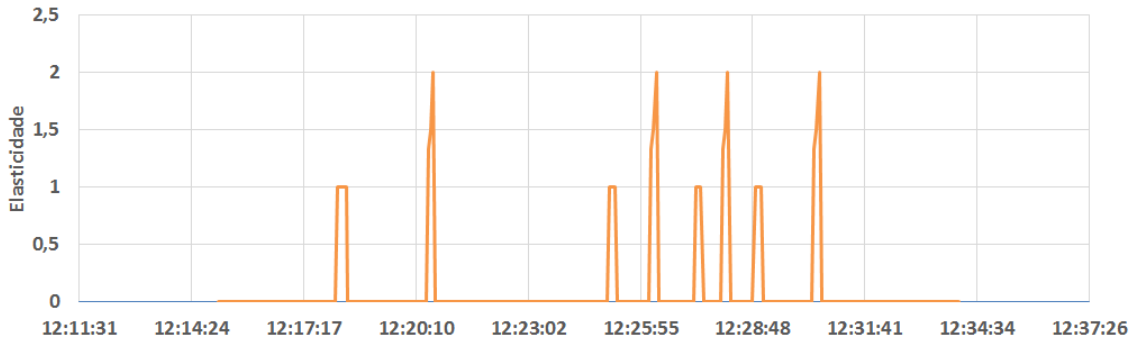
6.1.5.2 Métrica 2 - Microeconomia

Os valores de E_{Di} calculados para os Experimentos 1 e 2 podem ser visualizados na Figura 6.7. E_{Di} para o Experimento 1 possuiu 11 picos grandes de variação, indo da faixa de 0 a 3. Para o Experimento 2, a variação foi menor, de 0 a 2, e possuindo apenas 8 picos.

Sempre que há uma alocação e uma desalocação, existem picos correspondentes em E_{Di}, a exceção do Experimento 1, no qual o último pico ocorreu de maneira isolada. Esse fato coincide com a menor situação de alocação (de 1 para 2 máquinas virtuais), e sua rápida desalocação. O Experimento 1 foi bem mais diversificado que o 2, com variações mais bruscas



(a) Experimento 1 - Microbenchmark



(b) Experimento 2 - Blast

Figura 6.7: Elasticidade dos Recursos Demandados (E_{Di}) para os Experimentos 1 e 2

no consumo de CPU. O Experimento 2, por sua vez, foi mais constante em termos de cargas de trabalho e alocação.

À medida em que os recursos foram demandados, e os limites de SLA estabelecidos eram atingidos, os recursos disponíveis eram alocados, por meio da adição de máquinas virtuais ao balanceador de carga, chegando ao limite do ambiente. Quando havia uma diminuição de requisições, os recursos alocados ocupados eram novamente liberados. Pelo mecanismo de elasticidade, eles eram retirados do balanceador de carga, o que implicava na desalocação das máquinas virtuais.

Na média, os experimentos tiveram valores muito próximos. $\overline{E_{Di}}$ resultou em 0.75 e 0.74, respectivamente para os Experimentos 1 e 2. A carga de trabalho foi disparada da mesma maneira, variando-se apenas a aplicação utilizada e taxas. Esses foram os fatores diferenciais no consumo de CPU.

6.1.5.3 Métricas Orientadas a Tempos de Operações e Utilização de Recursos

Para uma avaliação complementar da elasticidade, um conjunto de métricas foi proposto, divididas em dois grupos: tempo de execução de operações, medida em minutos, e utilização de recursos, medida em quantidade de máquinas virtuais. A Tabela 6.2 contém as métricas calculadas para os dois experimentos realizados.

Para as métricas relacionadas a tempos de operações, ambos experimentos possuíram *TAE* bem longos em relação às demais métricas, ocupando grande parte do tempo total dos experimentos. Essa estabilidade significa que, de certa forma, as alocações e desalocações estavam “estabilizadas" ou “paradas" em termos de operações de elasticidade. No Experimento 2, *TASu* e *TASo* foram maiores que no Experimento 1. Porém, *TASu* e *TASo* tiveram valores próximos, o que pode implicar em capacidades elásticas semelhantes. Para TAT, os valores foram muito próximos nos dois experimentos, reforçando operações de elasticidade executadas com comportamento e duração semelhantes. Porém a duração dos experimentos e a quantidade de mudanças nos estados influencia na análise. Não se pode afirmar com certeza que o Experimento 2 foi pior que o Experimento 1, ou menos elástico, pois em termos de duração foi aproximadamente 6 minutos mais longo.

Em relação às métricas orientadas à utilização de recursos, $TRAE$, $TRASu$ e $TRASo$, ao contrário da maioria das métricas de tempo, foram maiores para o Experimento 1, implicando uma maior utilização de recursos durante ações de alocação e desalocação (máquinas virtuais). É de se esperar que quanto maior forem seus valores, maior seja TAT. Essas métricas analisam indiretamente a elasticidade do ambiente. Se analisarmos os gráficos de elasticidade das Figuras 6.5 e 6.6, verificamos que as variações coincidem com as alocações (Figuras 6.3 e 6.4), consequentemente com os momentos onde são mais utilizados recursos, podendo ser empregados como análise complementar para a elasticidade.

Em ambos experimentos, *TAE* ocupa a maioria do tempo de alocação, conforme esperado. Isto se deve ao fato que em grande parte dos experimentos, a alocação dos recursos é estável, variando apenas quando há incremento ou redução de recursos, refletidos nos tem-

Tabela 6.2: Métricas coletadas para os Experimentos 1 e 2

Métrica	**Experimento 1**	**Experimento 2**
Tempo Total do Experimento (minutos)	12:58	19:00
Tempo de Alocação Estabilizada (TAE)	11:01	16:51
Tempo de Alocação Subprovisionada (TASu)	00:26	00:37
Tempo de Alocação Sobreprovisionada (TASo)	00:27	00:31
Tempo de Alocação Transitória (TAT)	01:03	00:59
Percentual do Tempo de Alocação Estabilizada	84.92%	88.74%
Total de Intervalos de Tempo de Alocação Estabilizada	12	9
Tempo Médio de Alocação Estabilizada	00:55	01:52
Percentual do Tempo de Alocação Subprovisionada	3.45%	3.29%
Total de Intervalos de Tempo de Alocação Subprovisionada	6	4
Tempo Médio de Alocação Subprovisionada	00:04	00:09
Percentual do Tempo de Alocação Sobreprovisionada	2.04%	2.73%
Total de Intervalos de Tempo de Alocação Sobreprovisionada	6	4
Tempo Médio de Alocação Sobreprovisionada	00:04	00:07
Percentual do Tempo de Alocação Transitória	8.10%	5.23%
Total de Recursos Alocados (máquinas virtuais)	132	102
Total de Recursos Alocados Estabilizado (TRAE)	54	42
Média de Recursos Alocados Estabilizado	4.5	4.66
Total de Recursos Alocados Subprovisionados (TRASu)	47	36
Média de Recursos Alocados Subprovisionados	7.8	9
Total de Recursos Alocados Sobreprovisionados (TRASo)	31	24
Média de Recursos Alocados Sobreprovisionados	5.1	6
Elasticidade de *Scaling Up*	0.03	0.01
Elasticidade de *Scaling Down*	0.04	0.02
Relação Custo/Benefício Média	1.98	2.06
Elasticidade Média da Física $\overline{E_{RDi}}$	0.06	0.03
Elasticidade Média da Física $\overline{E_{RAi}}$	0.07	0.00
Elasticidade Média da Microeconomia $\overline{E_{Di}}$	0.75	0.74

pos *TASu*, *TASo* e *TAT*. Apesar de experimentos com diferentes cargas de trabalho, todas as métricas de tempo possuíram valores percentuais em relação ao tempo total do experimento próximos para os dois experimentos, indicando similaridade nos experimentos.

Estabilidade não necessariamente implica em atendimento do SLA. Um valor alto para *TAE* deve ser analisado em conjunto com outros aspectos. O limite superior definido para SLA foi 90%, e ocorreram algumas violações. Isto indica que a capacidade de recursos atingiu o limite. Se houvessem mais recursos, os gráficos de alocação teriam mais degraus, correspondendo a alocações e desalocações de máquinas virtuais. As métricas *TRASu* e *TRASo* aumentariam e *TRAE* diminuiria. Consequentemente *TASu* e *TASo* provavelmente seriam maiores. Diferente da métrica *TAE*, cujo valor é maior que *TASu*, *TASo* e *TAT*, a métrica *TRAE* mostra que um ambiente pode ter grandes intervalos de alocação estável com poucos recursos. Quanto maior a variação na média de consumo de CPU, maior a adição ou remoção de recursos no ambiente. Apesar que no Experimento 1 foram alocados mais recursos às operações de *scaling up* e *scaling down*, na média os valores foram bem próximos. O ideal é que quanto maior a quantidade de intervalos em estado de alocação subprovisionada e sobreprovisionada, maior a capacidade do ambiente em se adaptar às demandas impostas. Os valores médios para

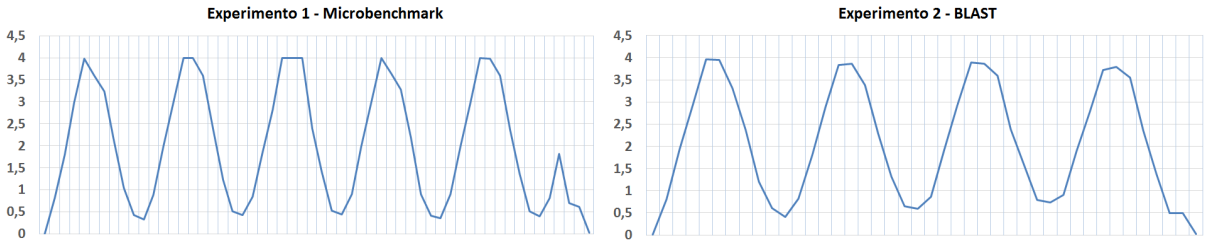


Figura 6.8: Relação custo/benefício da elasticidade para os dois experimentos.

TRASu, *TRASo* e *TRAE* foram bem próximos nos dois experimentos, indicando similaridade nos comportamentos.

Outros autores na literatura propuseram diferentes formas de se medir e avaliar a elasticidade. Para uma comparação e análise mais completa, algumas métricas foram coletadas nos experimentos e dispostas na Tabela 6.2.

Utilizando as métricas propostas por Herbst, Kounev e Reussner (2013), concluiu-se que o Experimento 1 é um pouco mais veloz que o Experimento 2, obtendo 0.03 e 0.01 na Elasticidade de *Scaling Up* respectivamente. Esse valor indica a velocidade na qual recursos são alocados, mudando de um estado subprovisionado para um estado ótimo (estados com alocação igual à demanda) ou sobreprovisionado. Na Elasticidade de *Scaling Down* ocorre o mesmo comportamento em termos de velocidade (0.04 e 0.02), sendo o Experimento 1 também mais veloz. Esse valor indica a velocidade na qual são desalocados recursos de estados sobreprovisionados para estados ótimos ou para estados subprovisionados.

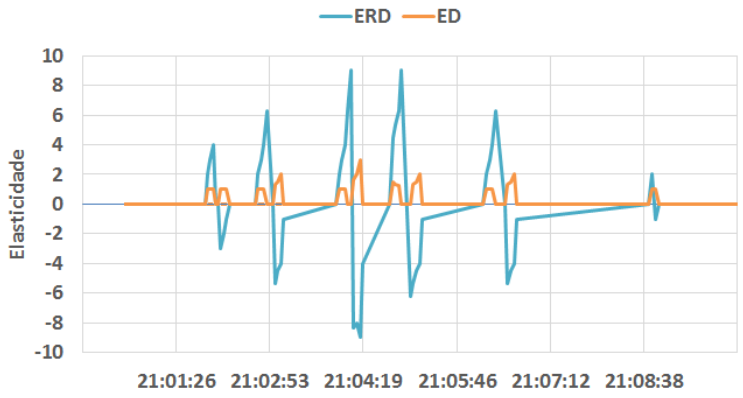
Por fim, uma análise do custo/benefício da elasticidade foi realizada sobre um ambiente de Computação em Nuvem por Simões e Kamienski (2014), utilizando métricas de aplicação. Após uma adaptação, trazendo para o contexto dos experimentos executados, calculamos a relação custo/benefício como a multiplicação entre o percentual de utilização de CPU e a quantidade de máquinas virtuais alocadas em cada ponto de coleta do experimento. Sendo assim, calculou-se a relação custo/benefício média para os dois experimentos, identificado que os resultados foram bastante próximos, mesmo com cargas de trabalho diferentes, com uma leve vantagem para o Experimento 1. Visualmente percebe-se que o Experimento 2 possui uma curva mais suave, o que indica que o ambiente tem comportamento semelhante ao longo do tempo em relação à alocação das máquinas virtuais e utilização de CPU. O ideal é que o custo/benefício seja o mais baixo possível, pois assim indicaria que poucas operações de alocações e desalocações são necessárias para suprir as necessidades de utilização de CPU, ou que eles são proporcionais. Na Figura 6.8 pode-se observar que no final do Experimento 1 ocorreu uma diminuição no valor da relação custo/benefício, o que é desejável.

6.1.6 Discussão dos Resultados

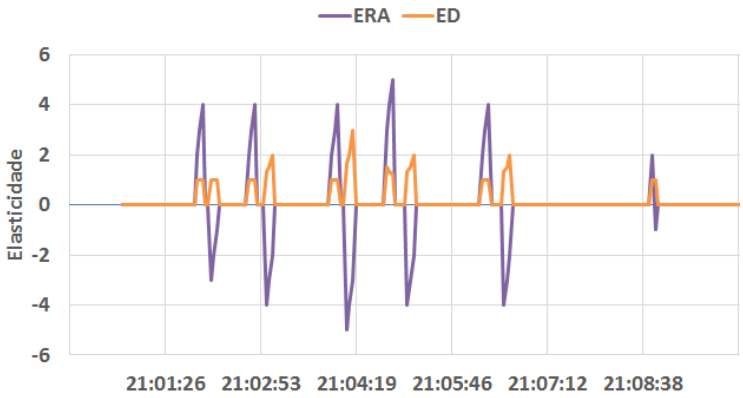
De maneira geral, analisando os dois experimentos, nota-se que em alguns momentos a média de utilização da CPU ficou acima do limite, resultando em violações do SLA. Caso existissem mais máquinas virtuais alocadas ao balanceador de carga, tais violações reduziriam.

Isso é um aspecto diretamente relacionado ao provisionamento de recursos. Nesse caso, o ideal é que instâncias de máquinas virtuais fossem geradas dinamicamente sempre que necessário, e finalizadas quando não mais necessário. O tempo de resposta foi alto em muitos casos, mesmo quando novas máquinas virtuais eram alocadas. Muitos deles devido à latência das operações. Para minizar esses altos valores, possivelmente uma escolha de limiares diferentes seria suficiente para manter a latência sempre dentro de um limite aceitável, além da inclusão de regras que contemplassem o tempo de resposta como critério para ações de elasticidade.

As Figuras 6.9 e 6.10 exibem os valores calculados para E_{RDi}, E_{RAi} e E_{Di} com uma sobreposição nos resultados. Comparando a diferença entre os valores mínimos e máximos de E_{RDi} e E_{RAi} em relação a E_{Di}, houve uma diferença significante. Para o Experimento 1, os valores de E_{RDi} e E_{RAi} foram respectivamente 18 e 10, e E_{Di} foi igual a 3. Para o Experimento 2, 11.5 e 8, respectivamente para E_{RDi} e E_{RAi}, e 2 para E_{Di}. Além disso, E_{Di} não possui valores negativos. O Experimento 1 teve em alguns pontos E_{RAi} atingindo 5, e nesses momentos a alocação estava no máximo, necessitanto talvez de mais recursos, que o ambiente não possuía. Nesse momento, algumas violações de SLA ocorreram. O mesmo ocorreu para o Experimento 2, mas com um pico menor, com E_{RAi} igual a 4. O equivalente ocorreu para E_{RAi} negativos, indicando desalocações. Em relação a E_{RDi}, o mesmo comportamento ocorreu.

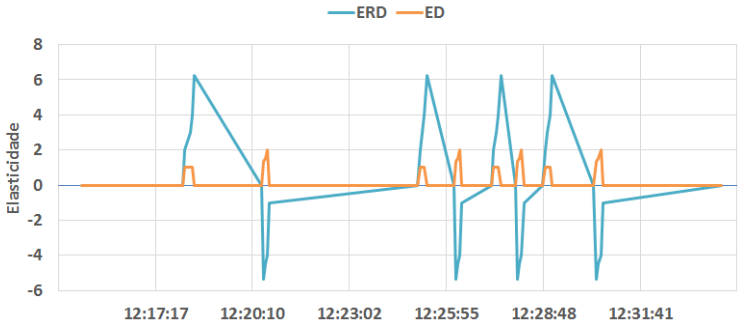


(a) Experimento 1 - *Microbenchmark* - E_{RDi} e E_{Di}

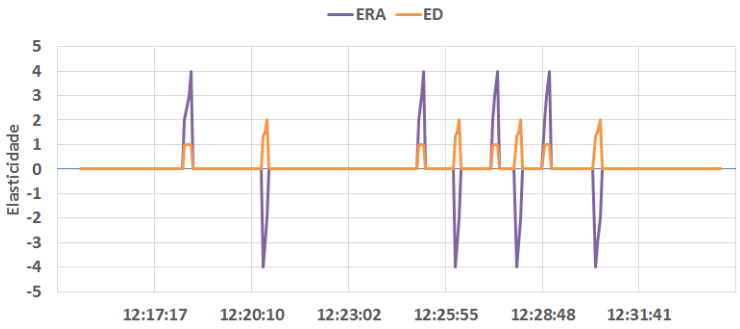


(b) Experimento 1 - *Microbenchmark* - E_{RAi} e E_{Di}

Figura 6.9: Gráficos sobrepostos da elasticidade para o Experimento 1 - *Microbenchmark*.



(a) Experimento 2 - Blast - E_{RDi} e E_{Di}



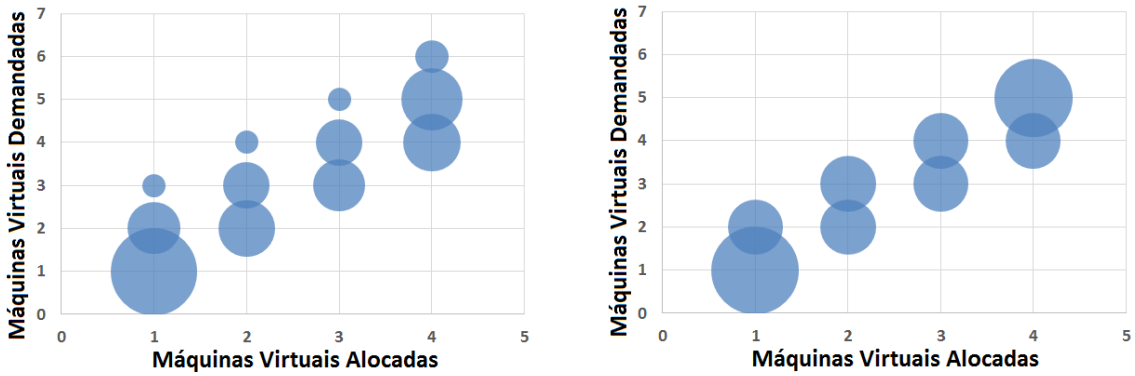
(b) Experimento 2 - Blast - E_{RAi} e E_{Di}

Figura 6.10: Gráficos sobrepostos da elasticidade para o Experimento 2 - Blast.

Os valores para E_{RDi}, E_{RAi} e E_{Di} acompanharam as variações no consumo de CPU e na alocação das máquinas virtuais. Em ambos experimentos, na maioria dos casos, para cada pico de alocação e desalocação, houve um pico em para E_{RAi} e E_{RDi}, e E_{Di}, um correspondente às alocações, e outro correspondente às desalocações. Especificamente para E_{Di}, no Experimento 1, houve apenas um pico, ocupando o tempo de alocação e desalocação. Isto ocorreu justamente ao final do experimento, quando a maioria das máquinas virtuais estavam ficando disponíveis e sendo desalocadas pelo balanceador de carga. Os pontos de pico entre as métricas propostas coincidem, variando apenas seus valores.

Em relação às médias, os valores para os Experimentos 1 e 2 foram bastante próximos. Porém houve uma variação considerável entre as métricas orientadas a conceitos da Física e orientadas a conceitos da Microeconomia. Isso se deve ao fato que apenas a carga de trabalho aplicada foi diferente, permanecendo a mesma infraestrutura, o mesmo mecanismo de elasticidade, e os mesmos limiares. A Elasticidade de *Scaling Up* e Elasticidade de *Scaling Down* tiveram valores bastante próximos a E_{RAi} e E_{RDi}, e sempre o Experimento 1 maior que o Experimento 2, indicando uma coerência entre os resultados. Comparando essas métricas com $\overline{E_{RDi}}$ e $\overline{E_{RAi}}$, com valores iguais a 0.06 e 0.07, respectivamente para o Experimento 1, e 0.03 e 0.00 respectivamente para o Experimento 2, reforça-se que o Experimento 1 é mais elástico, em termos de utilização de CPU e operações com máquinas virtuais.

O gráfico da Figura 6.11 descreve para os dois experimentos a quantidade de máquinas virtuais alocadas (eixo x) e a quantidade de máquinas virtuais demandadas (eixo y). O Experimento 1 alocou mais máquinas virtuais que o Experimento 2, tanto em termos de quanti-



(a) Experimento 1 - Microbenchmark (b) Experimento 2 - Blast

Figura 6.11: Gráfico de bolhas para os dois experimentos indicando máquinas virtuais alocadas por máquinas virtuais demandadas.

dade, quanto em termos de antecipação de recursos. Porém, o Experimento 2, foi mais proporcional em termos de alocação de recursos, pois existem pontos em que a alocação e a demanda estão igualmente dispostas no gráfico. Esses valores podem possuir o mesmo significado das métricas $TRASu$ e $TRASo$, pois assim como elas medem a quantidade de recursos em estados subprovisionados e sobprovisionados, podemos utilizar o gráfico também para avaliar estados sobreprovisionados e subprovisionados, não sob o ponto de vista temporal, mas em quantidade de estados nos quais a demanda é maior ou menor que a oferta.

Verificando os gráficos das Figuras 6.9 e 6.10, percebe-se que há uma diferença na amplitude de E_{RAi} e E_{RDi} para os dois experimentos. O Experimento 1 é mais variado, possuindo picos com maiores variações entre si para ambas as métricas, atingindo os maiores valores próximo à metade do experimento. Porém o Experimento 2 foi mais constante, possuindo os picos com valores próximos. Isto pode ser explicado pela carga de trabalho variada aplicada aos dois experimentos, o que justifica o padrão identificado por E_{RAi} e E_{RDi}. Porém, para E_{Di}, a variação não é tão sensível, não se verificando esse comportamento no gráfico de maneira evidente. Para E_{Di}, verificou-se que na maioria dos casos há um pico referente a um evento de alocação (*scaling up*) e logo em seguida um pico referente a um evento de desalocação (*scaling down*). Esses picos para o Experimento 2 possuíram E_{Di} com valores próximos entre si, ou seja, E_{Di} com valores em picos de alocação próximos entre si, e E_{Di} com valores em picos de desalocação próximos entre si. Porém, este comportamento não foi visto no Experimento 1. De certa forma, esse resultado confirma o que foi verificado no gráfico da Figura 6.11.

A relação custo/benefício indicou que na média, o Experimento 1 foi mais eficiente que o Experimento 2, reforçando o que a Elasticidade de *Scaling Up*, Elasticidade de *Scaling Down*, $\overline{E_{RDi}}$, $\overline{E_{RAi}}$ e $\overline{E_{Di}}$ indicaram. Porém, identificou-se que a diferença nos valores para todas essas métricas foi pequena. Analisando esse resultado em conjunto com o gráfico da Figura 6.11, identifica-se novamente que o Experimento 1 foi mais flexível em termos de ações de elasticidade. Não necessariamente o valor da relação custo/benefício é adequada para um provedor. Seu valor pode ser baixo, mas utilizar muitos recursos, o que para uma nuvem pública onde o usuário paga pela utilização de recursos, o custo pode se tornar inviável. Assim, uma relação custo/benefício com valor maior (pior) pode ser mais adequada para a aplicação em exe-

cução. Sendo assim, se na medida de custo/benefício for adicionado o componente financeiro, a avaliação da elasticidade deverá considerar situações de priorização sobre a economia ou preço gerado pelo experimento sobre a própria elasticidade em si.

A análise de desempenho muitas vezes depende da carga de trabalho utilizada na aplicação, tendo impacto direto sobre o ambiente. Um desafio para ambientes experimentais é projetar uma carga de trabalho que represente a realidade de um provedor, devido à característica dinâmica de cargas de trabalhos reais. Um estudo do comportamento das cargas de trabalho, através de métodos estatísticos, predição, natureza das aplicações e usuários pode auxiliar no projeto. Como os dois experimentos utilizaram o mesmo ambiente, a carga de trabalho foi o principal elemento a provocar a diferença nos resultados.

Algumas limitações neste trabalho foram identificadas. A métrica associada às ações de elasticidade definidas pelos limiares foi apenas o percentual de utilização de CPU. Devido a isso, outros aspectos do sistema e da aplicação foram prejudicados, sendo o principal o tempo de resposta das aplicações executadas. Mesmo atendendo ao SLA na maior parte do tempo dos experimentos, em alguns períodos de tempo dos experimentos os tempos de respostas foram bastante elevados. Caso houvesse uma regra baseada no tempo de resposta, provavelmente ocorreriam menos violações de SLA e as aplicações seriam melhor distribuídas nas máquinas virtuais. Concluiu-se que utilizar apenas métricas associadas à infraestrutura pode não ser adequado às necessidades de SLA da aplicação.

O fato da não criação e destruição das máquinas virtuais empregadas pela estratégia de elasticidade utilizada impacta no resultado de algumas métricas, especialmente nas métricas orientadas a tempos de operações. Isto se deve ao fato que as máquinas virtuais e serviços já estavam imediatamente funcionais, em vez de demorar um determinado tempo para sua criação e destruição. Porém, esse tempo gasto influencia na maneira na qual os recursos são alocados no tempo, influenciando na quantidade de recursos alocados, e consequentemente nos valores das métricas propostas.

A utilização da média como critério de decisão para ações de elasticidade tem impacto apenas nas coletas recentes. Assim, o histórico de consumo de CPU não é considerado para eventos de configuração no ambiente. A correta seleção de métricas também permite uma análise mais objetiva e sem desperdício de esforço na coleta e consolidação dos dados. A utilização de recursos estatísticos para obtenção de parâmetros de decisão é uma estratégia que pode ser facilmente modificada conforme a necessidade. Em um ambiente de nuvem real, onde diversos usuários estão concorrendo por recursos, além das cargas de trabalho dinâmicas, variadas e sem um controle do provedor, os parâmetros definidos para tomada de decisão devem ser ajustados para provocar um melhor efeito sobre o ambiente, de preferência dinamicamente. Uma estratégia a ser empregada neste ambiente poderia ser a utilização inicial de recursos da nuvem privada, por meio de elasticidade vertical (a ser implementado na infraestrutura), seguido de elasticidade horizontal, e em caso de necessidade por mais recursos, avaliar a possibilidade de se utilizar recursos de uma nuvem pública.

Em relação à quantidade de máquinas virtuais utilizadas, o experimento restringiu-se apenas à variação de 1 a 4 instâncias alocadas. Isto deixou o experimento limitado, não

possibilitando uma visão mais abrangente, prejudicando como os diversos parâmetros do sistema, projetados pelo FOLE, se comportariam em relação ao ambiente. Há um impacto na utilização de muitas instâncias. O mecanismo de elasticidade deve possuir a capacidade de criação e remoção de máquinas virtuais. O tempo de resposta das requisições pode diminuir caso a estratégia de balanceamento de carga seja efetiva. Entretanto, se a utilização dessas instâncias for de maneira dinâmica, o tempo para sua criação e ativação pode influenciar no desempenho inicial das aplicações a serem executadas. Para uma visibilidade maior, seria interessante a utilização de simulação para uma quantidade maior de máquinas virtuais, e assim dependendo dos resultados obtidos, ajustar o ambiente real.

Como principais conclusões dos experimentos, destaca-se: (1) as métricas de elasticidade propostas possuem alta relação com o projeto de cargas de trabalho; (2) a elasticidade se comportou bem em relação à utilização de CPU, mas não foi eficiente em relação ao tempo de resposta; (3) métricas de aplicação devem ser utilizadas em ações de elasticidade e em regras para melhorar o desempenho do ambiente; (4) em diversos momentos os resultados obtidos foram semelhantes às métricas para a avaliação da elasticidade propostas na literatura, demonstrando coerência, porém a maneira proposta para o cálculo da elasticidade é mais simples.

6.2 Experimento - Nuvem Híbrida

Este experimento tem como objetivo avaliar a elasticidade de uma nuvem híbrida, onde inicialmente são utilizados recursos de uma nuvem privada, e conforme a necessidade por mais recursos, são adicionados recursos de uma nuvem pública. O FOLE foi utilizado para apoiar a análise de desempenho, e as métricas de elasticidade baseadas em conceitos da Física e Microeconomia foram calculadas, com suporte da ferramenta de visualização de dados.

6.2.1 Material e Métodos

Os experimentos foram projetados e executados em dois ambientes diferentes de nuvens computacionais: uma nuvem privada e uma nuvem pública. Para a nuvem privada, o OpenNebula 3.8 foi utilizado, com todas as máquinas físicas com 5 e 7 núcleos, 24 GB de memória RAM, sistema operacional Ubuntu Server 12.04 64 bits e hipervisor KVM. Cada máquina virtual foi criada com 1 VCPU, 1 GB de memória RAM e sistema operacional Ubuntu Server 12.04 64 bits. Para a nuvem pública, foi utilizada a plataforma da Microsoft Azure, com as instâncias criadas do tipo A1 padrão (1 núcleo e 1.75 GB de memória RAM) e sistema operacional Ubuntu Server 14.04 64 bits. Cada experimento utilizou quatro máquinas virtuais, porém a quantidade de instâncias utilizadas na nuvem pública e privada variou conforme o experimento. Foi utilizado como servidor *web* o Apache Tomcat, balanceador de carga o NGINX, e gerador de cargas de trabalho o HTTPERF. A instanciação do *framework* foi desenvolvida em Java e *shell script*. A Figura 6.12 exibe o *testbed* utilizado, baseado na arquitetura proposta no Capítulo 5.5, Seção 5.5.1.

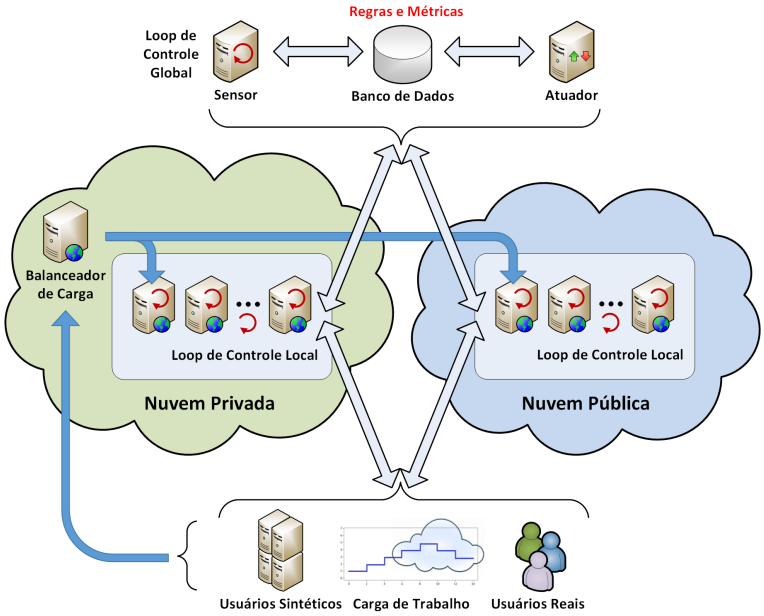


Figura 6.12: Arquitetura e ambiente experimental

6.2.2 Carga de Trabalho

A geração de cargas de trabalho para os experimentos ocorreu de duas maneiras distintas: (1) requisições encaminhadas diretamente na máquina virtual do balanceador de carga, geradas pelo HTTPERF e por navegadores *web*, distribuídas entre as demais máquinas virtuais alocadas; e (2) requisições executadas diretamente nas máquinas virtuais utilizadas pela infraestrutura. Dessa maneira é possível emular a concorrência pelos recursos em um ambiente de Computação em Nuvem. O Experimento 1 pode utilizar a mesma representação de carga de trabalho dos Experimentos 2 e 3, já que ele utiliza apenas uma máquina virtual em cada tipo de nuvem. As Figuras 6.13 e 6.14 ilustram a carga de trabalho empregada respectivamente para os Experimentos 2 e 3 sobre a infraestrutura construída. A Tabela 6.3 detalha informações adicionais sobre a carga de trabalho e demais itens do projeto experimental.

6.2.3 Projeto do Experimento - Instanciação do FOLE

Para o planejamento da análise de desempenho, o FOLE foi utilizado. Suas atividades foram instanciadas para a descrição do projeto do experimento e consequente avaliação da elasticidade. O objetivo do experimento é avaliar a elasticidade de uma nuvem híbrida, onde são utilizados recursos de uma nuvem privada e conforme a necessidade por mais recursos, a utilização de recursos de uma nuvem pública.

A macroatividade “Planejar Análise de Desempenho” teve as ferramentas definidas indiretamente pela infraestrutura. Arquivos texto registram o *log* das operações de coleta das máquinas virtuais e das demais informações consolidadas. O *log* gerado para cada máquina virtual em arquivo texto contém a data da coleta, valores de utilização de CPU, memória, disco, rede e tempo de resposta das requisições, assim como a média de utilização de CPU, alocação

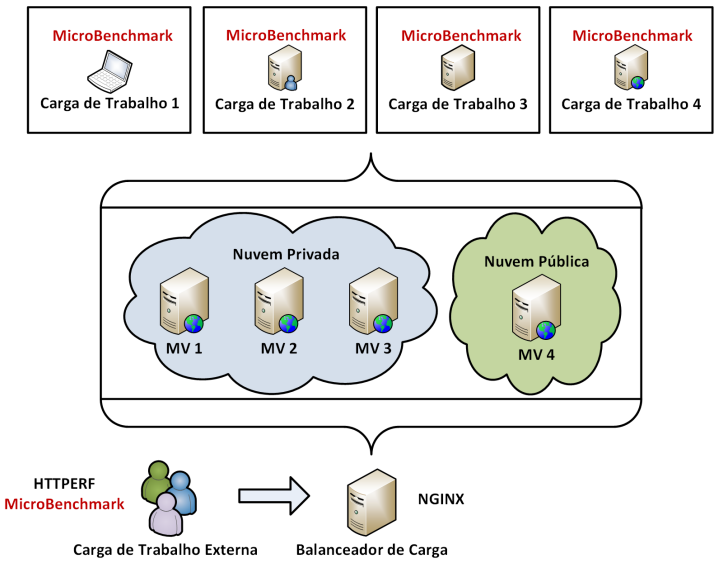


Figura 6.13: Representação da carga de trabalho aplicada ao Experimento 2 com nuvem híbrida

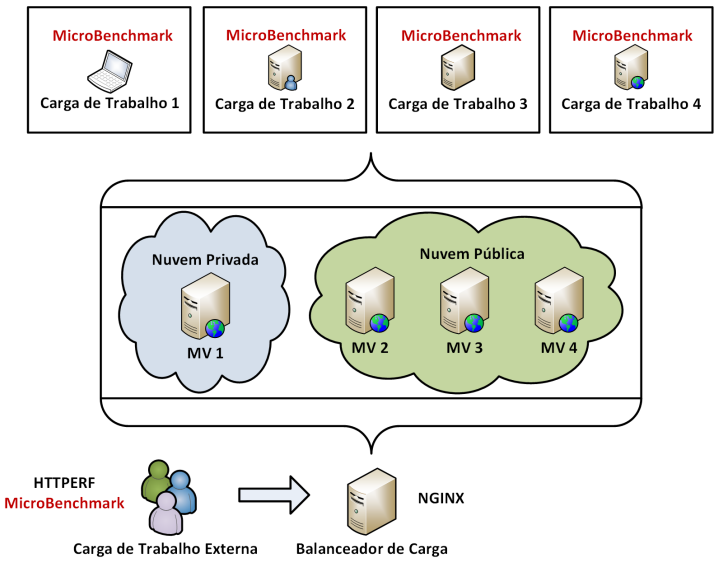


Figura 6.14: Representação da carga de trabalho aplicada ao Experimento 3 com nuvem híbrida

de recursos, e dados sobre a elasticidade. A consolidação dos dados ocorreu por meio da leitura dos arquivos de *log*. O dados coletados e consolidados são apresentados sob a forma de gráficos de linha em uma ferramenta de visualização, onde todas as métricas podem ser visualizadas de maneira individual e em conjunto. O intervalo de coletas definido foi de 1 segundo adicionado do custo da coleta e análise dos resultados. As métricas utilizadas são as armazenadas em arquivos de *log*, adicionada às métricas de elasticidade detalhadas na Tabela 6.3. A carga de trabalho empregada nos experimentos foi descrita na Seção 6.2.2 e na Tabela 6.3.

Na atividade "Planejar Elasticidade", um mecanismo baseado na arquitetura proposta no Capítulo 5.5, Seção 5.5.1, foi implementada. Uma estratégia de elasticidade horizontal foi utilizada, por meio da qual sempre que recursos são necessários, novas máquinas virtuais são adicionadas ao balanceador de carga, e retiradas caso não sejam mais necessárias. Para disparar

Tabela 6.3: Critérios para avaliação de desempenho para os experimentos

Critérios	**Descrição**
Sistema	Ambiente híbrido de Computação em Nuvem (OpenNebula e Microsoft Azure)
Métricas	Tempo de resposta das aplicações, percentual de utilização de CPU, percentual de utilização de memória, pacotes recebidos e enviados em KB, KB lidos e escritos em disco; S_{QN}, T_{RD}, T_{RA}, E_{RDi}, E_{RAi}, $\overline{E_{RDi}}$, $\overline{E_{RAi}}$, E_{Di}, $\overline{E_{Di}}$; TAE, $TASu$, $TASo$, TAT, $TRASu$, $TRASo$, $TRAE$; Elasticidade de *Scaling Up* e Elasticidade de *Scaling Down* (HERBST; KOUNEV; REUSSNER, 2013)
Parâmetros	CPU, memória, sistema operacional, quantidade de máquinas virtuais, tipo de instância
Fatores	Configuração do *benchmark* (repetições e tamanho da matriz), configuração do HTTPERF (quantidade de requisições, taxa de requisições, tempo)
Técnica de Avaliação	Medição
Carga de Trabalho	Executar multiplicações de matrizes (dimensão 200x200x200) através de um *microbenchmark* construído na linguagem de programação Java, sob a forma de pequenas aplicações *web*, com requisições disparadas através do HTTPERF, com taxas variando de 1 a 5 requisições por segundo e quantidade de conexões variando em 10, 30 e 50
Projeto de Experimentos	Geração da carga de trabalho com o HTTPERF no servidor *web* e em cada máquina virtual da infraestrutura para que os recursos dinamicamente se adaptem à quantidade de CPU; **Experimento 1:** 1 máquina virtual na nuvem privada e 1 máquina virtual na nuvem pública; **Experimento 2:** 3 máquinas virtuais na nuvem privada e 1 máquina virtual na nuvem pública; **Experimento 3:** 1 máquina virtual na nuvem privada e 3 máquinas virtuais na nuvem pública;
Análise dos Dados	Interpretação dos resultados apresentados nos gráficos de média do percentual de utilização de CPU, gráficos de alocação das máquinas virtuais, gráficos da elasticidade, *boxplot* e tabela com dados estatísticos
Apresentação dos Resultados	Gráficos de linhas, *boxplot* e tabelas

ações de elasticidade, foi utilizada a média do percentual de utilização de CPU das máquinas virtuais. As métricas utilizadas para a medição da elasticidade foram: S_{QN}, T_{RD}, T_{RA}, E_{RDi}, E_{RAi}, $\overline{E_{RDi}}$, $\overline{E_{RAi}}$, E_{Di} e $\overline{E_{Di}}$. Os limiares utilizados para a execução de ações de elasticidade foram: acima de 70% (aloca uma nova máquina virtual), abaixo de 60% (desaloca uma máquina virtual), e entre 60% e 70% (mantém alocação). Esse valor foi calculado como a média das 10 últimas coletas de utilização de CPU nas máquinas virtuais. Como mecanismo de predição foi utilizado regressão multilinear sobre valores de utilização de CPU, memória, disco e rede. Os valores para os limiares e para o mecanismo de predição foram definidos baseados em experimentos prévios utilizando as mesmas taxas projetadas em cargas de trabalho semelhantes. Para o provisionamento dos recursos, a estratégia de balanceamento de carga foi utilizada, onde máquinas virtuais são adicionadas conforme a necessidade. Um conjunto de *scripts* foram desenvolvidos para as ações e estratégias de análise.

As macroatividades “Inicializar Serviços da Análise de Desempenho” e “Executar Análise de Desempenho” foram implementadas por meio de *scripts* que inicializavam os serviços de coleta, análise, geração das cargas de trabalho, consolidação dos dados, além de analisar resultados e disparar ações.

6.2.4 Experimentos

Três experimentos foram executados, conforme projeto de experimentos, para avaliar a elasticidade em uma nuvem híbrida, com a variação da quantidade de máquinas virtuais envolvidas nas nuvens privada e pública. Ao final desta seção, uma discussão sobre os resultados é apresentada.

6.2.4.1 Experimento 1 - Uma Máquina Virtual na Nuvem Privada e Uma Máquina Virtual na Nuvem Pública

Este experimento teve duração 36min10s. Apenas duas máquinas virtuais estiveram envolvidas neste experimento: uma na nuvem privada e outra na nuvem pública. Dessa maneira foi possível verificar se a infraestrutura construída proveria recursos das duas nuvens, constituindo uma nuvem híbrida, conforme a necessidade gerada pela carga de trabalho. A Figura 6.15 exibe a média de utilização de CPU consolidada em todas as máquinas virtuais, a alocação das máquinas virtuais, as métricas de elasticidade baseadas em conceitos da Física e da Microeconomia e o tempo de resposta das requisições.

A média do percentual de utilização de CPU foi bastante diversificada, chegando em alguns pontos a 100% de utilização. Isto ocorreu devido a muitas requisições e poucas máquinas virtuais para o atendimento. Entretanto, sempre que o limiar de 70% estabelecido como limite superior era ultrapassado, a máquina virtual da nuvem pública era incluída na lista do balanceador de carga, e assim as requisições eram distribuídas, reduzindo o valor do percentual de utilização de CPU. Violações ocorreram por causa da utilização de apenas duas máquinas virtuais no experimento (poucos recursos), e assim que a utilização de CPU estava acima do limite e ambas máquinas virtuais estavam alocadas, não existia mais a possibilidade de alocar mais recursos, restando esperar que as requisições encerrassem. Momentos de alocação das duas máquinas virtuais em geral coincidem com momentos de alta utilização de CPU.

As métricas para elasticidade possuíram valores baixos até cerca da metade do experimento. A partir da metade do tempo do experimento os valores começaram a aumentar, e a utilização de CPU em geral permaneceu elevada, indicando uma necessidade maior de recursos. Na maioria dos casos, momentos de alocação e desalocação coincidiram com os picos nas métricas de elasticidade. O que provocou essa variação foi o SLA definido e a velocidade na qual recursos são alocados e desalocados.

O tempo de resposta das requisições no início do experimento possuiu os maiores valores, e à medida em que o experimento avançava, estes tempos foram reduzindo. Como só haviam duas máquinas virtuais disponíveis para o experimento, assim que as requisições eram distribuídas pelo balanceador de carga, consequentemente o tempo de resposta diminuía. Porém isso não implicava no pleno atendimento do SLA em todo o período do experimento, visivelmente identificado no gráfico de utilização de CPU.

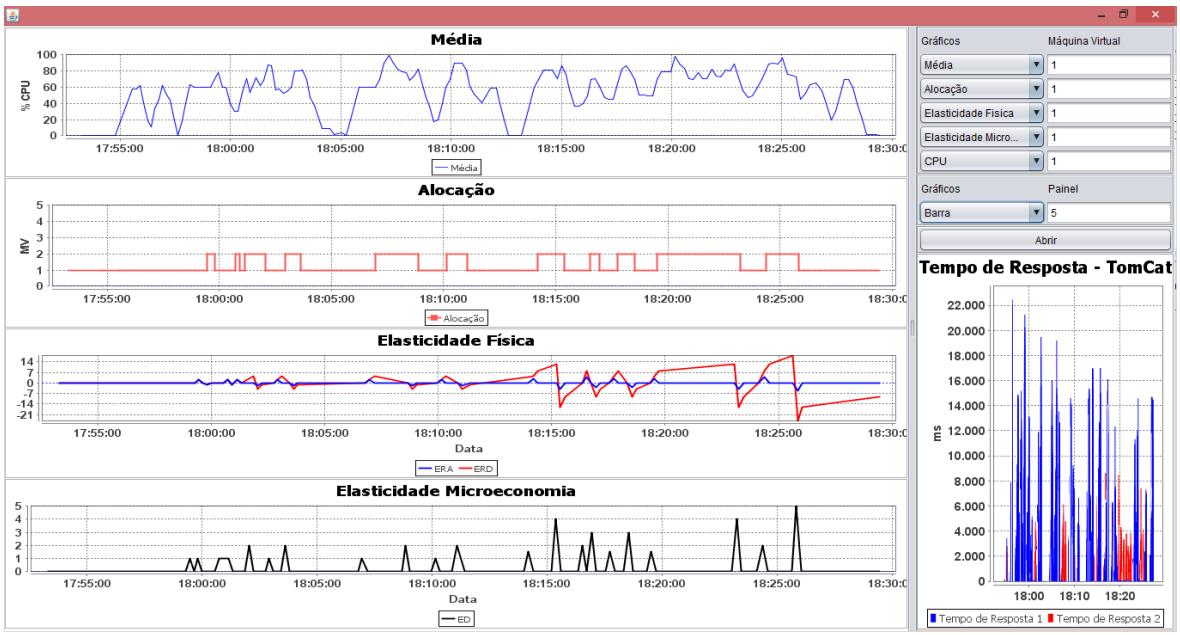


Figura 6.15: Média de utilização de CPU, alocação das máquinas virtuais, métricas de elasticidade baseadas em conceitos da Física e da Microeconomia e tempo de resposta das requisições para o Experimento 1.

6.2.4.2 Experimento 2 - Três Máquinas Virtuais na Nuvem Privada e Uma Máquina Virtual na Nuvem Pública

A duração deste experimento foi de 11min14s. Quatro máquinas virtuais estiveram envolvidas neste experimento: três na nuvem privada e uma na nuvem pública. Assim, foi possível verificar se o desempenho da infraestrutura seria impactado pela adição de uma máquina virtual de uma nuvem pública, e se o SLA seria mantido. A Figura 6.16 exibe a média de utilização de CPU consolidada em todas as máquinas virtuais, a alocação das máquinas virtuais, as métricas de elasticidade baseadas em conceitos da Física e da Microeconomia e o tempo de resposta das requisições.

Com esta configuração, a média do percentual de utilização de CPU foi mais constante. Isto ocorreu devido a uma distribuição com mais máquinas virtuais, e consequentemente mais recursos. Sempre que o limiar de 70% estabelecido como limite superior era ultrapassado, o balanceador de carga adicionava na sequência as duas máquinas virtuais da nuvem privada, e em caso de necessidade por mais recursos, a máquina virtual da nuvem pública era adicionada. Mesmo assim, violações do SLA ocorreram, mas em menor quantidade.

As métricas para elasticidade possuíram valores baixos até metade do experimento, onde se iniciou uma variação mais intensa na amplitude. Isto ocorreu por causa das alocações e necessidade por recursos. As regras impostas de SLA só realmente começaram a ser alcançadas e demandar recursos por volta da metade do experimento, momento onde as alocações se iniciaram com mais intensidade. Na maioria dos casos, momentos de alocação e desalocação coincidiram com os picos nas métricas de elasticidade. Novamente, o que influenciou nesse resultado foi o SLA definido e a velocidade de alocação e desalocação de recursos.

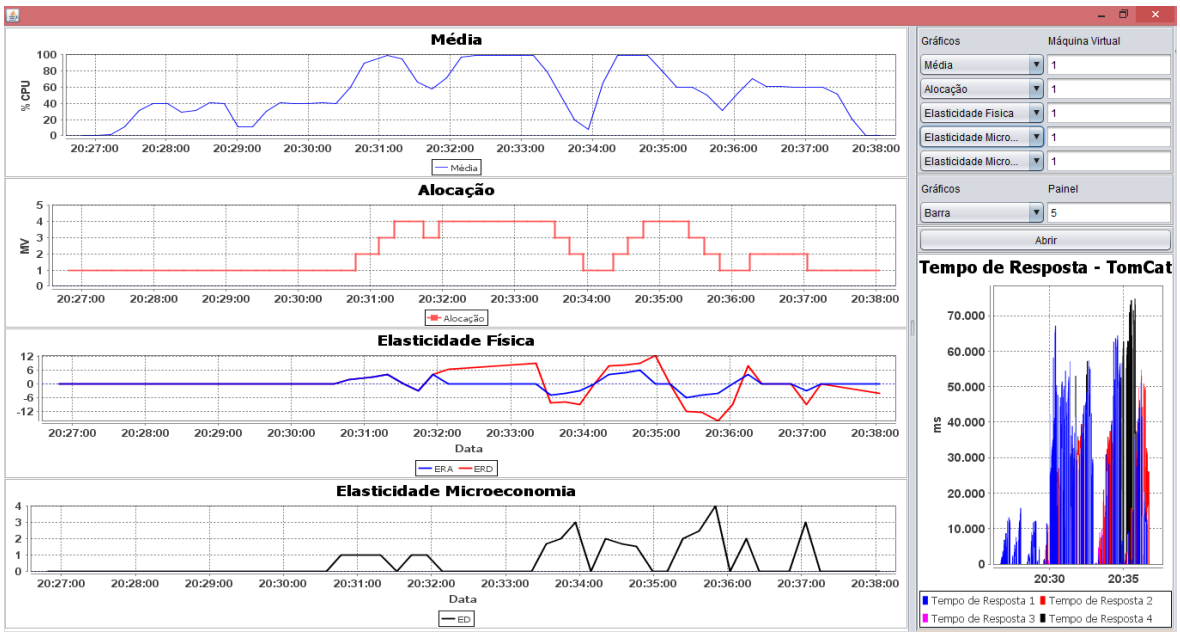


Figura 6.16: Média de utilização de CPU, alocação das máquinas virtuais, métricas de elasticidade baseadas em conceitos da Física e da Microeconomia e tempo de resposta das requisições para o Experimento 2.

O tempo de resposta das requisições inicialmente possuiu valores bem baixos em relação a grande parte do experimento, e praticamente todos alocados à primeira máquina virtual. À medida que a carga de trabalho aumentava, alocações eram realizadas conforme a necessidade. Por volta da metade do experimento, muitas requisições tiveram um alto tempo de duração para sua finalização, mesmo com a alocação das quatro máquinas virtuais. Os maiores tempos de resposta ocorreram na máquina virtual da nuvem pública.

6.2.4.3 Experimento 3 - Uma Máquina Virtual na Nuvem Privada e Três Máquinas Virtuais na Nuvem Pública

A duração deste experimento foi de 14min51s. Quatro máquinas virtuais estiveram envolvidas: uma na nuvem privada e três na nuvem pública. Assim, foi possível verificar se o desempenho da infraestrutura seria impactado pela adição de várias máquinas virtuais de uma nuvem pública, e se o SLA seria mantido. A Figura 6.17 exibe a média de utilização de CPU consolidada em todas as máquinas virtuais, a alocação das máquinas virtuais, as métricas de elasticidade baseadas em conceitos da Física e da Microeconomia e o tempo de resposta das requisições.

A média do percentual de utilização de CPU foi constante em vários pontos, porém com um alto consumo de CPU. Sempre que o limiar de 70% estabelecido como limite superior era ultrapassado, o balanceador de carga adicionava na sequência as três máquinas virtuais da nuvem pública, em caso de necessidade de mais recursos. Mesmo assim, violações do SLA ocorreram, mas em menor quantidade, conforme o gráfico de média de utilização da CPU da Figura 6.17. Entretanto, neste experimento ocorreram períodos de violação maiores que nos

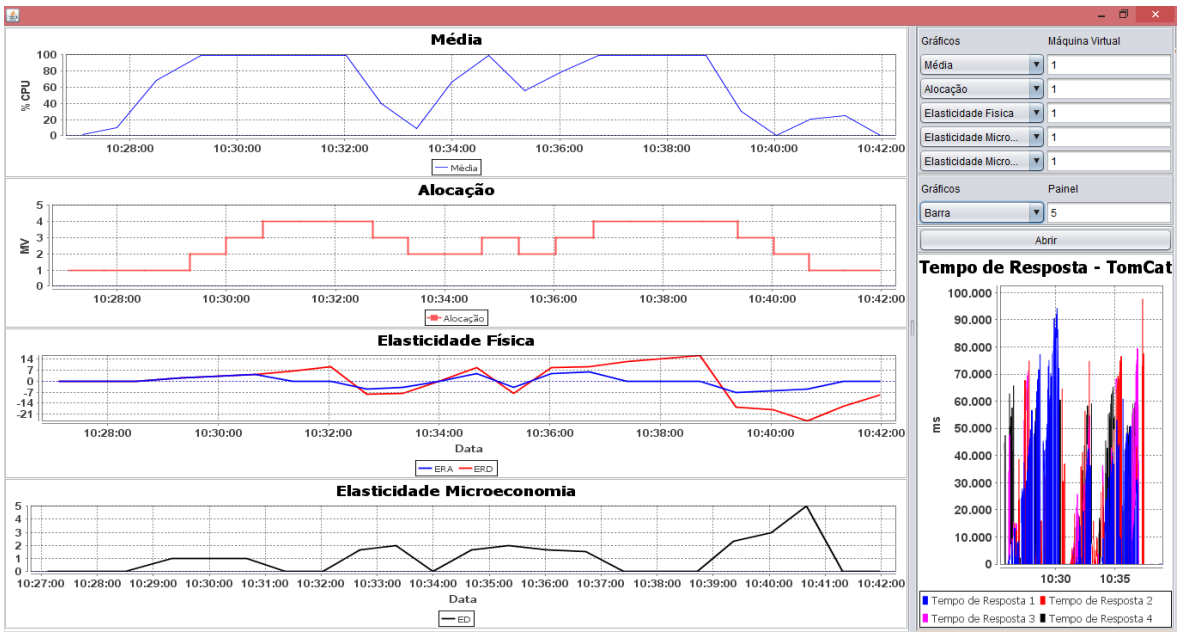


Figura 6.17: Média de utilização de CPU, alocação das máquinas virtuais, métricas de elasticidade baseadas em conceitos da Física e da Microeconomia e tempo de resposta das requisições para o Experimento 3.

experimentos anteriores.

As métricas para elasticidade baseadas em conceitos da Física possuíram valores altos durante grande parte do experimento. Tanto as métricas baseadas em conceitos da Física quanto da Microeconomia possuíram valores distribuídos ao longo de todo o experimento, não tendo muitas regiões com valor igual a zero. Isto ocorreu devido às alocações ocorrerem logo no início do experimento, e por existir alocação de apenas 1 máquina virtual no início e no final do experimento. Entretanto, ocorreram longos períodos de elasticidade estável, o que não implica no atendimento do SLA, e sim estabilidade em estados de alocação.

O tempo de resposta das requisições foi bastante variado e distribuído entre todas as máquinas virtuais. Todas as máquinas virtuais possuíram requisições com altos tempos de resposta.

6.2.5 Discussão do Resultados

Os três experimentos utilizaram a mesma infraestrutura, os mesmos limiares e a mesma estratégia de alocação de recursos. A carga de trabalho foi a mesma, variando apenas a quantidade de vezes que foi aplicada.

A duração das requisições em vários pontos dos três experimentos resultou em altos valores. Em geral uma requisição duraria 2ms, e algumas duraram cerca de 20.000ms, 70.000ms, e 90.000ms, para os três experimentos respectivamente. Essa situação não é interessante, pois é muito tempo para uma requisição. Como o tempo de resposta não foi considerado como uma métrica a ser utilizada para ações de elasticidade, os três experimentos possuíram

esse prejuízo no atendimento das requisições. O ideal é que métricas de aplicação, como o tempo de resposta e vazão (*throughput*), sejam incluídas nas regras para a execução de ações de elasticidade.

A utilização de máquinas virtuais em nuvens públicas não impactou muito nos resultados dos experimentos do ponto de vista de infraestrutura, validando a infraestrutura proposta. Entretanto, a estratégia de empregar o percentual de utilização de CPU como métrica para acionar ações de elasticidade poderia ser mais efetiva, pois facilmente se atingia o limite definido, e assim a ocorrência de violações no SLA e consequentemente altos tempos de resposta das requisições. A latência da rede pode ter sido um fator que influenciou neste tempo de resposta elevado. A latência, entretanto, poderia ter sido utilizada como uma regra a ser aplicada para ações de elasticidade.

A Tabela 6.4 exibe os resultados para as métricas de elasticidade e valores estatísticos. Observou-se que $\overline{E_{RDi}}$ não obteve bons valores. $\overline{E_{RAi}}$ e $\overline{E_{Di}}$ possuíram resultados semelhantes do ponto de vista estatístico, e mais condizentes com os experimentos. A amplitude dos experimentos (diferença entre valores máximos e mínimos) foi bem elevada para $\overline{E_{RDi}}$. Isso exibe uma alta variação entre extremos que indica uma necessidade de recursos maior do que o ambiente possui. Já $\overline{E_{RAi}}$ e $\overline{E_{Di}}$ possuíram valores com diferenças menores, sendo mais próximos entre si. Uma variância baixa indica experimentos com rendimentos semelhantes, sinalizado por $\overline{E_{Di}}$. Desvios padrão baixos indicam uma baixa dispersão (alta concentração) dos valores em torno da média, assim como um desvio padrão alto indica alta dispersão (baixa concentração) dos valores em torno da média. Sob essa perspectiva, $\overline{E_{RAi}}$ e $\overline{E_{Di}}$ possuíram bons resultados. Em relação à mediana e moda, praticamente todos os valores foram zero, o que indica muitos pontos de estabilidade nos três experimentos, seja com alocação mínima ou máxima, refletindo as métricas de elasticidade com valores baixos, quase sempre zero, e que pode refletir também que o SLA não foi atingido.

Os dados da Tabela 6.4 também podem ser visualizados no *boxplot* da Figura 6.18. Podemos visualizar para cada métrica de cada experimento os valores mínimos, primeiro quartil, medianas (segundo quartil), terceiro quartil, valores máximos, média (círculo preto maior) e desvio padrão (losango cinza). Alguns *outliers* foram identificados (valores fora da curva, representados pelos círculos pretos menores), principalmente no Experimento 1, onde ocorreram

Tabela 6.4: Resultados das métricas de elasticidade e valores estatísticos para os três experimentos

	Experimento 1			**Experimento 2**			**Experimento 3**		
Duração	36:10			11:14			14:51		
	E_{RAi}	E_{RDi}	E_{Di}	E_{RAi}	E_{RDi}	E_{Di}	E_{RAi}	E_{RDi}	E_{Di}
Valor Médio	0	-0.08	0.70	-0.03	-0.50	1.04	-0.28	-1.45	1.13
Máximo	4	18	5	6	12,25	4	6	16	5
Mínimo	-5	-25	0	-6	-16	6	-7	-25	0
Amplitude	9	43	5	12	28.25	4	13	41	5
Variância	2.88	52.36	1.36	10.10	55.04	1.33	14.01	124.04	1.69
Desvio Padrão	1.69	7.23	1.16	3.17	7.41	1.15	3.74	11.13	1.30
Mediana	0	0	0	0	0	1	0	0	1
Moda	0	0	0	0	0	0	0	0	1

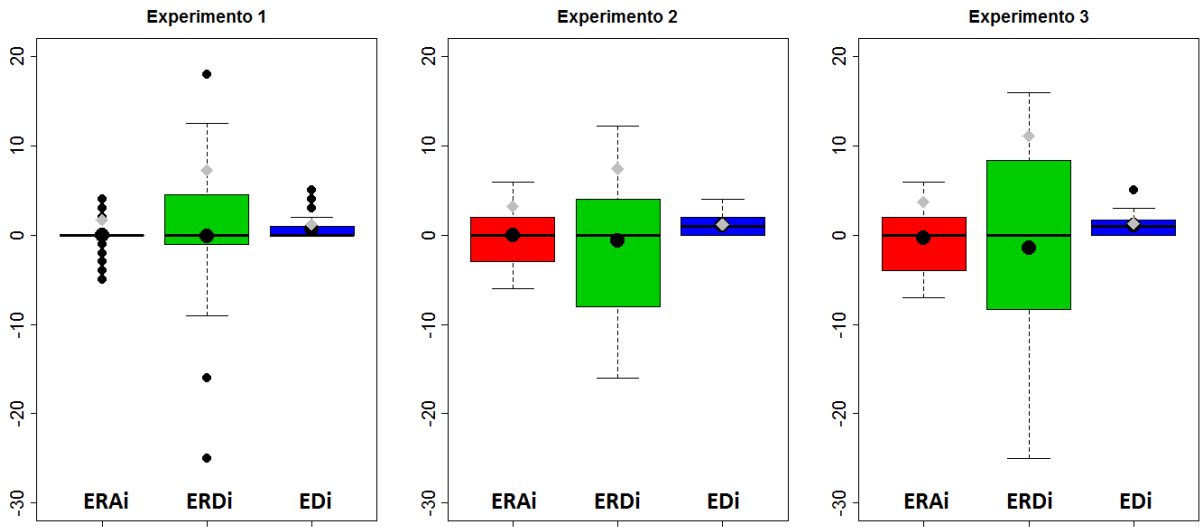


Figura 6.18: *Boxplot* gerado para os três experimentos

a maior quantidade de violações no SLA. Pela figura, é possível visualizar que os Experimentos 2 e 3 possuíram comportamentos relativamente semelhantes.

No Experimento 1, *outliers* ocorreram com maior frequência. $\overline{E_{RAi}}$ foi 0, porém com valores bem próximos a $\overline{E_{RDi}}$ e $\overline{E_{Di}}$. Isto indicou uma tentativa de ajuste dos recursos em relação à carga de trabalho aplicada, com alocações e desalocações, porém não havia recursos suficientes, por isso os picos na elasticidade. O Experimento 2 resultou em médias com variações pequenas para as três métricas de elasticidade. Não foram identificados *outliers*. Este experimento foi o que melhor utilizou os recursos da nuvem híbrida, entretanto apenas uma máquina virtual foi utilizada da nuvem pública, o que minimizou a latência da rede imposta pela nuvem pública. O Experimento 3 também apresentou médias com variações pequenas paras as três métricas de elasticidade. Apenas um *outlier* foi identificado. Esperava-se mais valores fora do comum para a elasticidade devido ao fato de ter sido empregada uma nuvem híbrida nesse experimento.

Em geral, os gráficos das Figuras 6.15, 6.16 e 6.17 exibiram E_{Di} com uma diferença visual principalmente em relação à largura dos picos, sendo os menos espaçados no Experimento 1 e os mais espaçados no Experimento 3, indicando uma estabilidade crescente nas alocações dos experimentos. Para E_{RAi} e E_{RDi}, a partir de cerca da metade do tempo dos experimentos, ocorreu uma maior variação nos valores, indicando uma alocação mais intensa. Já no *boxplot* da Figura 6.18, a distribuição foi semelhante em todos os três experimentos.

A Tabela 6.5 lista as métricas orientadas a tempos de operações de alocação e desalocação de recursos, métrica orientadas a utilização de recursos, e métricas calculadas da literatura para os três experimentos.

O Experimento 1 possuiu maior estabilidade (TAE), mas estabilidade não necessariamente implica em atendimento ao SLA. Esse experimento foi o que mais necessitou de recursos e obteve a maior quantidade de violações. As métricas $TASu$, $TASo$ e TAT também foram as que resultaram nos mais elevados valores, o que indica muitas operações de alocação

Tabela 6.5: Métricas orientadas a tempos de operações de alocação e desalocação de recursos, orientadas a utilização de recursos, e calculadas da literatura para os três experimentos

Métrica	**Exp. 1**	**Exp. 2**	**Exp. 3**
Tempo Total do Experimento (minutos)	36:10	11:14	14:51
Tempo de Alocação Estabilizada (TAE)	27:46	6:23	4:01
Tempo de Alocação Subprovisionada (TASu)	2:15	0:57	2:01
Tempo de Alocação Sobreprovisionada (TASo)	2:05	0:49	1:56
Tempo de Alocação Transitória (TAT)	4:02	3:03	6:51
Total de Recursos Alocados (máquinas virtuais)	91	70	54
Total de Recursos Alocados Estabilizado (TRAE)	58	30	22
Total de Recursos Alocados Subprovisionados (TRASu)	22	24	19
Total de Recursos Alocados Sobreprovisionados (TRASo)	11	16	13
Elasticidade de *Scaling Up*	0.04	0.01	0.00
Elasticidade de *Scaling Down*	0.09	0.02	0.01
Elasticidade Média da Física $\overline{E_{RAi}}$	0.00	-0.03	-0.28
Elasticidade Média da Física $\overline{E_{RDi}}$	-0.08	-0.50	-1.45
Elasticidade Média da Microeconomia $\overline{E_{Di}}$	0.70	1.04	1.13

e desalocação. Em relação à quantidade de recursos alocados, o Experimento 1, mesmo tendo apenas 2 máquinas virtuais envolvidas, realizou mais alocações e desalocações.

O Experimento 2 foi o obteve o melhor desempenho nos tempos, o que reforça os dados dispostos no *boxplot*, com o menor tempo para TAT, indicando que dos três experimentos foi o que menos gastou tempo esperando os efeitos das alocações sobre o ambiente. Em relação à quantidade de recursos alocados, o Experimento 2 possuiu uma distribuição razoável entre os recursos e os estados de alocação.

O Experimento 3 indicou o maior valor para TAT, o que não é bom. Esse valor pode ter sido causado pela latência da rede, já que este experimento utilizou 3 máquinas virtuais em uma nuvem pública, e o tempo gasto para as operações realmente serem efetivadas pode ter prejudicado o experimento, e consequentemente ter elevado os tempos de resposta das requisições. Em relação à quantidade de recursos alocados, o Experimento 3 utilizou a menor quantidade de recursos dos três experimentos, e possuiu uma distribuição de recursos também razoável entre os estados de alocação.

Os Experimentos 2 e 3 possuíram ambos 4 máquinas virtuais envolvidas nos experimentos. Entretanto a variação da quantidade de máquinas virtuais na nuvem privada e pública influenciou em $TASu$ e $TASo$, sendo o Experimento 3 praticamente o dobro dos valores obtidos no Experimento 2. Em compensação, TAE para o Experimento 2 foi maior, mesmo o tempo total do experimento menor, implicando em uma estabilidade proporcionalmente maior.

Em relação à Elasticidade de *Scaling Up* e Elasticidade de *Scaling Down*, os valores para todos os experimentos foram bem próximos. Destaca-se que o Experimento 1 foi o mais veloz, com os maiores valores, entretanto seu desempenho foi o pior, dado a falta de recursos.

Todos os experimentos possuíram tempos relativamente elevados para TAT. Isso indica que a comunicação entre as nuvens está influenciando no desempenho. Em geral, as requisições para os três experimentos obtiveram o tempo de resposta aumentando à medida em

que se utilizava mais máquinas virtuais da nuvem pública. Então a latência da rede é um fator que deve ser considerado, pois está diretamente associado aos tempos das métricas orientadas à tempos de operações de alocação e desalocação, principalmente *TAT*.

Para a melhoria do ambiente, seria interessante a utilização de novas regras para a execução de ações de elasticidade, sendo a mais indicada o tempo de resposta das requisições, e a latência da rede. Também se indica estratégias de alocação e predição mais efetivas, para evitar momentos longos de subprovisionamento e sobreprovisionamento de recursos, e assim evitar ociosidade e desperdício. Uma melhor distribuição dos recursos em nuvens privadas ou públicas, com um melhor balanceamento, também é uma situação a ser analisada. Uma estratégia a ser seguida poderia ser a utilização inicial de recursos da nuvem privada, por meio de elasticidade vertical (a ser implementado na infraestrutura), seguido de elasticidade horizontal, e em caso de necessidade por mais recursos, replicar essa estratégia na nuvem pública. Por fim, a carga de trabalho aplicada sobre a infraestrutura, demandada por usuários ou *benchmarks*, implica diretamente na elasticidade, e deve ser melhor projetada, pois assim a infraestrutura pode ser configurada e adaptada da melhor maneira possível.

Em relação à quantidade de máquinas virtuais utilizadas, o experimento restringiu-se apenas à variação de 1 a 4 instâncias alocadas, e apenas a uma nuvem privada e uma nuvem pública. Isto deixou o experimento limitado, não possibilitando uma visão mais abrangente, prejudicando como os diversos parâmetros do sistema, projetados pelo FOLE, se comportariam em relação ao ambiente. Mesmo com o impacto na utilização de muitas instâncias, uma avaliação com uma quantidade maior de instâncias é interessante para permitir a integração entre nuvens de diferentes provedores, e verificação de seus efeitos sobre a elasticidade. Nesse experimento, uma estratégia de elasticidade horizontal foi utilizada, mas com instâncias já previamente funcionais. Também seria interessante a utilização de máquinas virtuais de maneira dinâmica, ou seja, com a criação e remoção de instâncias, mesmo com o tempo para sua criação e ativação influenciando no desempenho das aplicações a serem executadas. O tempo de resposta das requisições e a latência podem diminuir caso a estratégia de balanceamento de carga seja efetiva. A utilização de simulação para uma quantidade maior de máquinas virtuais e diferentes provedores possibilitaria a experimentação com diversas combinações, e assim dependendo dos resultados obtidos, reajustar o ambiente real.

Como principais conclusões dos experimentos, destaca-se: (1) as métricas de elasticidade propostas possuem alta relação com o projeto de cargas de trabalho; (2) a elasticidade se comportou bem em relação à utilização de CPU, mas não foi eficiente em relação ao tempo de resposta; (3) métricas de aplicação, como tempo de resposta, devem ser utilizadas em ações de elasticidade e em regras para melhorar o desempenho do ambiente; (4) a latência da rede prejudica o desempenho da elasticidade quando se utiliza nuvens híbridas.

7 CONSIDERAÇÕES FINAIS

Este capítulo é dedicado às considerações finais. A Seção 7.1 apresenta os resultados alcançados com este trabalho, e a Seção 7.2 apresenta os trabalhos futuros oriundos desta tese de doutorado.

7.1 Resultados Alcançados

As principais contribuições desta tese foram: (i) FOLE, um *framework* conceitual para análise de desempenho da elasticidade em ambientes de Computação em Nuvem (Capítulo 5); (ii) métricas para a elasticidade baseadas em tempos de operações de alocação e desalocaçao de recursos, e na quantidade de recursos utilizados (Subseção 4.1), nos conceitos da Física de tensão, estresse e elasticidade (Subseção 4.2) e no conceito da Microeconomia de Elasticidade do Preço da Demanda (Subseção 4.3); (iii) uma arquitetura baseada em conceitos da Computação Autonômica para o provimento de soluções elásticas (Subseção 5.5.1); e (iv) uma ferramenta para apoio à avaliação de desempenho da elasticidade, com suporte à visualização e análise (Subseção 5.5.2).

Neste documento foi apresentada uma revisão sistemática (Subseções 3.1, 3.2 e 3.3), o que possibilitou uma uma revisão bibliográfica mais detalhada. Esta revisão identificou ferramentas e métricas utilizadas na análise de desempenho da elasticidade em nuvens computacionais, e estratégias empregadas para o provimento da elasticidade. Além disso, diversas tendências de pesquisa em elasticidade foram identificadas, permitindo novos direcionamentos de pesquisa. Uma análise dos trabalhos permitiu identificar que muitos artigos não detalham a metodologia utilizada (Subseção 3.4). Quando detalham, não explicam muito bem como as métricas devem ser utilizadas e interpretadas.

Nesta tese foi levantada a hipótese que a sistematização da análise de desempenho da elasticidade em Computação em Nuvem permite uma avaliação de quanto a nuvem se adéqua às cargas de trabalho e contribui para a tomada de decisão de maneira eficiente (Seção 1.2). Os resultados deste trabalho confirmaram que esta hipótese foi aceita. As questões de partida (QP) levantadas nesta tese são discutidas a seguir:

QP01: *Quais são as principais características que uma nuvem computacional deve possuir para prover elasticidade ?*

O resultado obtido pela revisão sistemática (Subseções 3.1, 3.2 e 3.3) possibilitou a identificação de várias características que nuvens computacionais devem possuir para o provimento da elasticidade. A elasticidade permite ao ambiente a capacidade do provisionamento e desprovisionamento de recursos conforme a necessidade. Uma nuvem computacional deve possuir essa capacidade para seu provimento. A carga de trabalho exercida pelos usuários da nuvem influencia diretamente no ambiente e na elasticidade. O monitoramento de recursos é fundamental para que mecanismos de elasticidade possam tomar decisões de quando e quanto os recursos deverão ser incrementados ou reduzidos no ambiente. Características de Computa-

ção Autonômica foram identificadas em diversos trabalhos. Tais características, como regras, *loops* de controle, sensores e atuadores, são bastante úteis na implementação de mecanismos de elasticidade. A utilização de algum mecanismo de balanceamento ou distribuição da carga de trabalho é um importante fator que deve ser considerado pelo provedor. Além disso, diversas estratégias podem ser aplicadas no provimento da elasticidade, com diferentes modelos e métodos, conforme identificados na revisão bibliográfica (Subseção 3.3.5).

QP02: *É possível construir um mecanismo de elasticidade em um ambiente de Computação em Nuvem que permita sua adequação às necessidades dos usuários ?*

Uma nuvem computacional, seja privada, pública ou híbrida, pode possuir mecanismos de elasticidade na sua infraestrutura. Um balanceador de carga pode ser utilizado, assim como o aumento e diminuição de recursos de uma máquina virtual. Uma vez que a nuvem possua um mecanismo que permita aplicar conceitos de Computação Autonômica, que consiga captar a utilização dos recursos, e adequar a capacidade conforme a demanda, esta nuvem possuirá um mecanismo de elasticidade. Sendo assim, arquiteturas podem ser projetadas ou ajustadas para poderem utilizar tais mecanismos, e prover elasticidade. Alguns mecanismos para o provimento da elasticidade foram identificados na revisão sistemática, podendo ser ser proativos ou reativos, horizontais ou verticais (Subseção 3.3.5). A arquitetura proposta neste trabalho (Subseção 5.5.1) possibilitou a experimentação de um mecanismo que provê elasticidade de maneira satisfatória em nuvens privadas e híbridas (Capítulo 6).

QP03: *É possível medir a elasticidade de uma nuvem computacional ?*

Conforme estudos bibliográficos realizados (Subseções 3.3.3 e 3.4) e métricas propostas (Capítulo 4), constatou-se que é possível medir a elasticidade, comprovada através de experimentos realizados no ambiente da nuvem computacional. Diversas métricas identificadas não medem a elasticidade diretamente, e sim recursos e operações do ambiente. Porém, por meio delas é possível indiretamente verificar o quanto um ambiente é elástico e se ele atende de maneira adequada às variações nas cargas de trabalho impostas. Algumas das métricas propostas não medem diretamente a elasticidade, e sim recursos do ambiente, como o tempo de operações de alocação e desalocação de recursos, e utilização dos recursos (Subseção 4.1). Adicionalmente, métricas específicas para a medição da elasticidade foram propostas (Subseções 4.2 e 4.3) e validadas por meio de experimentos, possibilitando a avaliação da elasticidade, sua respectiva interpretação, e colaboração para a melhoria do ambiente.

QP04: *É possível elaborar uma metodologia para análise de desempenho que permita avaliar a elasticidade de uma nuvem computacional ?*

Conforme o *framework* conceitual proposto, o FOLE (Capítulo 5), é possível seguir uma metodologia para a realização de uma análise de desempenho da elasticidade em ambientes de Computação em Nuvem, de maneira sistemática. Experimentos realizados permitiram a execução de diversas atividades de projeto, avaliação e análise de uma nuvem computacional, avaliando a elasticidade do ambiente (Capítulo 6). De maneira geral, o FOLE permitiu a realização da análise de desempenho em nuvens computacionais de maneira satisafatória por meio de alguns experimentos que utilizaram uma arquitetura baseada em conceitos de Computação Autonômica, algumas métricas orientadas a tempos de operações de alocação e desalocação de

recursos, e utilização de recursos, e conceitos de outras áreas de conhecimento. Dessa maneira, foi possível analisar o ambiente, e assim propor melhorias de maneira sistematizada.

Como principais conclusões dos experimentos destacamos:

1. As métricas de elasticidade propostas possuem alta relação com o projeto de cargas de trabalho.

   O projeto de carga de trabalho vai definir o comportamento da demanda, e caso o ambiente não consiga prover os recursos necessários, as métricas de elasticidade informarão a necessidade por mais recursos, ou que estes não estão adequados. Caso o projeto da carga de trabalho seja compatível com o perfil de demanda, as métricas refletirão essa situação.

2. A elasticidade se comportou bem em relação à utilização de CPU, mas não foi eficiente em relação ao tempo de resposta.

   As métricas propostas foram baseadas em recursos alocados e demandados do ambiente. Para os experimentos, limiares foram definidos utilizando o percentual de utilização de CPU, que é uma métrica de infraestrutura. Dessa maneira, a estratégia definida pela arquitetura e projetada sobre a infraestrutura foi capaz de realizar ações de elasticidade baseadas no consumo de CPU, se comportando de maneira satisfatória. Entratanto, não foram utilizadas métricas de aplicação, como o tempo de resposta das requisições, nas regras para acionamento de ações de elasticidade. Consequentemente, esse tipo de métrica não foi considerada para a elasticidade, o que prejudicou o desempenho do ambiente.

3. Métricas de aplicação, como tempo de resposta, devem ser utilizadas em ações de elasticidade e em regras para melhorar o desempenho do ambiente.

   Métricas de infraestrutura podem refletir situações globais do ambiente, e não específicas de uma aplicação. Métricas de aplicação, como tempo de resposta de requisições e *throughput*, são mais adequadas para a análise de desempenho de aplicações porque seus resultados podem ser mais significativos para os usuários. Recomenda-se sua utilização em regras para o acionamento de ações de elasticidade em conjunto com métricas de infraestrutura, pois dessa maneira colaborariam ainda mais para a avaliação e melhoria do desempenho de uma nuvem computacional.

4. A latência da rede influencia no desempenho da elasticidade quando se utiliza nuvens híbridas.

   Há um forte indício que avaliações da elasticidade com métricas que utilizam tempos de resposta de operações serão impactadas pela latência da rede entre as nuvens, como é o caso de nuvens híbridas. Nos experimentos realizados, por meio das métricas orientadas a tempos de operações de alocação e desalocação de recursos, percebeu-se que o fator latência influencia diretamente nessas métricas. Seus valores foram maiores quando uma nuvem híbrida foi utilizada. Diversas operações podem ser prejudicadas pela latência, como configurações nas máquinas virtuais, consolidação dos dados e requisições de usuários. A definição da sequência de locais onde os recursos serão buscados, ou estratégias a serem utilizadas, pode minimizar a latência (e.g., iniciar na nuvem privada ampliando capacidade, adicionar novas instâncias, e partir para a nuvem pública).

7.2 Trabalhos Futuros

Esta tese abordou diversos aspectos de arquitetura, infraestrutura, métricas e análise de desempenho. Assim, diversos trabalhos futuros podem ser derivados desta tese. Para uma melhor descrição, os trabalhos futuros foram separados nas seguintes categorias: experimentos, métricas, *framework* FOLE, arquiteturas e ferramentas.

7.2.1 Experimentos

Diversos experimentos podem ser projetados para avaliar o desempenho da elasticidade com o auxílio do FOLE e das métricas propostas, independente da arquitetura e infraestrutura utilizada. A comparação de soluções de elasticidade entre provedores e *middleware* de Computação em Nuvem, assim como a comparação entre soluções de elasticidade em nuvens privadas, híbridas e públicas possibilitarão uma avaliação mais ampla da elasticidade em ambientes de nuvens. Ferramentas de *software* livre como o OpenNebula e OpenStack, e provedores comerciais como a Amazon EC2, HP Cloud Services e Microsoft Azure podem ser utilizados para os experimentos, e terem seus desempenhos comparados e avaliados.

A avaliação da elasticidade em nuvens distribuídas possibilitaria uma visão de um ambiente bem mais complexo, devido às diversas nuvens envolvidas, e os efeitos da elasticidade em relação a outros aspectos, como latência. Assim, seria possível a utilização de regras que utilizassem a latência como uma métrica a ser analisada para as ações de elasticidade.

O projeto de experimento de cargas de trabalho específicas para um determinado domínio, e sua avaliação com o auxílio do FOLE e métricas propostas também são atividades a serem analisadas. A identificação da distribuição estatística que a carga de trabalho segue, sua modelagem e experimentação, seja por simulação, modelagem analítica ou medição, também são potenciais trabalhos futuros. Ferramentas como o HTTPERF[1] e JMeter[2] podem ser utilizadas para aplicação e comparação de cargas de trabalho.

A utilização de poucas máquinas virtuais nos experimentos possibilita uma visão limitada (ambiente relativamente simples e controlado). O ideal seria a utilização de uma quantidade grande de máquinas virtuais, e assim avaliar os efeitos da elasticidade quando aplicada em um ambiente de maior escala e volume de dados, consequentemente mais complexo.

7.2.2 Métricas

Este trabalho propôs um conjunto de métricas para a avaliação do desempenho da elasticidade em Computação em Nuvem. A repetição de seus cálculos em experimentos diversos permitirá uma melhor identificação de padrões nos resultados, e assim a evolução da interpretação e especificação, além de possíveis ajustes na forma do cálculo.

[1] HTTPERF - http://www.hpl.hp.com/research/linux/httperf/
[2] JMeter - http://jmeter.apache.org/

As métricas orientadas a tempos de operações de alocação e desalocação de recursos, e orientadas à quantidade de recursos, foram focadas em máquinas virtuais. A utilização de outras fontes de recursos e estratégias, como aumento da capacidade de processamento e memória, replicação e migração, são experimentos a serem projetados que permitirão avaliar o impacto de diferentes estratégias de elasticidade nas métricas propostas.

As métricas orientadas a conceitos da Física podem ainda utilizar outros conceitos, como a velocidade para dilatação e contração. Outro conceito da Física é a plasticidade, que ocorre quando o material não pode mais recuperar a sua forma original depois de se remover o estresse, podendo muito bem ser aplicado ao contexto de nuvens computacionais. As métricas orientadas a conceitos da Microeconomia possuem diversas situações relacionadas à elasticidade da demanda e da oferta, como interpretação dos resultados baseadas em gráficos da elasticidade. Todos esses conceitos dessas outras áreas do conhecimento podem auxiliar nas análises e justificativas dos resultados.

Para a validação de alguns trabalhos futuros, é interessante a utilização de simuladores para os experimentos, e avaliar a elasticidade em grande escala. Assim, a aplicação das métricas em simuladores como o CloudSim[3] e a execução de experimentos de simulação em grande escala permitirá uma visão mais ampla das métricas.

Além disso, a utilização das métricas em um ambiente com aplicações e usuários reais, e medição da elasticidade com as métricas propostas, ou avaliação do desempenho do provedor neste aspecto, possibilitará uma visão do desempenho de nuvens privadas, públicas ou híbridas de um ponto de vista mais realístico.

A inclusão do conceito de qualidade nas métricas propostas possibilitará sua evolução no sentido em que elasticidade possa ser mais adequada para QoS e SLA, e nas necessidades de qualidade dos clientes e provedores.

Além disso, o fator custo financeiro também poderia ser avaliado em conjunto com as métricas de elasticidade. Em nuvens públicas comerciais há o componente financeiro, não considerado neste trabalho. Entretanto, uma avaliação da elasticidade do ponto de vista financeiro, comparando diferentes provedores de serviços, agregando o cálculo da elasticidade das métricas propostas, pode auxiliar na escolha do melhor provedor para serviços elásticos. Outro trabalho relacionado à tarifação seria avaliar o custo da elasticidade em horários diferentes (diurno ou noturno), assim como a localização geográfica do *datacenter* do provedor.

Por fim, como trabalho futuro, a automação da coleta das métricas propostas, de modo que elas possam ser utilizadas em ferramentas de visualização e análise de maneira mais prática, como geração de gráficos e dados estatísticos em tempo real, possibilitando a análise em relação a outros aspectos do ambiente.

[3]CloudSim - http://www.cloudbus.org/cloudsim/

7.2.3 Framework FOLE

Mesmo com a validação do FOLE, suas atividades ainda podem ser evoluídas. Um detalhamento maior de cada atividade, para que seu uso seja mais intuitivo, e se destaque a importância de cada atividade para o projeto de experimentos e análise de desempenho, é um trabalho a ser desenvolvido. A repetição de sua aplicação permitirá um refinamento das atividades, e consequentemente melhoria na sua execução. Além disso, a aplicação em diferentes contextos também permitirá sua evolução e adequação à possíveis situações específicas da elasticidade que possam surgir.

A criação de orientações para a análise da elasticidade em determinados aspectos, como a interpretação em um gráfico específico, ou sua interpretação baseada em resultados estatísticos também auxiliaria muito na análise de desempenho.

7.2.4 Arquiteturas

Este trabalho tem um forte conteúdo arquitetural. Sendo assim, diversos trabalhos futuros nesse aspecto podem ser identificados.

A estratégia de provimento da elasticidade utilizada neste trabalho foi horizontal. O desenvolvimento e avaliação de estratégias diferentes, como vertical (aumentando a capacidade dos recursos) e mista (horizontal e vertical), permitiria uma análise mais completa de um ambiente de Computação em Nuvem, pois impactaria em diversos componentes arquiteturais.

A arquitetura proposta baseou-se em alguns componentes de Computação Autonômica. Sendo assim, novos níveis de *loops* poderiam ser empregados, com abordagens proativas. Novas regras e limiares poderiam ser propostos, utilizando não somente métricas de infraestrutura, mas métricas da própria aplicação, como tempo de resposta e *throughput*. Demais elementos do auto gerenciamento (auto otimização, auto cura e auto proteção) também poderão ser considerados para a arquitetura proposta neste trabalho como novos componentes e apoiar a elasticidade. Este trabalho utilizou funções de ação para execução de eventos de elasticidade. Como uma expansão deste mecanismo, funções de objetivo e utilitárias poderão ser utilizadas em trabalhos futuros, assim como a construção de novas arquiteturas baseadas no modelo *Monitor-Analyze-Plan-Execute* (MAPE).

O mecanismo de predição utilizado não foi efetivo, pois seu objetivo foi apenas validar uma atividade do FOLE. A sua evolução, por meio da utilização de diferentes estratégias, como classificação, aprendizagem de máquina, SVR (*Support Vector Regression*) e ARIMA (*AutoRegressive Integrated Moving Average*), são trabalhos futuros a serem desenvolvidos.

A Internet das Coisas (*Internet of Things* - IoT) promove um aumento da interação entre usuários por meio da conexão de diversos dispositivos ao seu redor, compondo uma grande nuvem computacional. A nuvem e os diversos dispositivos (móveis ou fixos) podem fazer uso da elasticidade para ajustar o ambiente conforme as necessidades dos serviços utilizados pelos usuários. Assim, a utilização do FOLE e das métricas propostas na avaliação da elasticidade do ambiente pode ser executada.

A virtualização convencional possui agregada o custo do *overhead* da utilização dos recursos (SCHEEPERS, 2014). Virtualização baseada em *container* pode ser uma alternativa para a redução do *overhead* e assim melhorar a utilização dos recursos nos *datacenters*. Linux Containers (LXC) é uma tecnologia capaz de executar uma grande quantidade de processos, cada um em seu próprio ambiente isolado. Esta técnica é chamada de virtualização baseada em *container*. Docker, por exemplo, é uma ferramenta que facilita o empacotamento de aplicações e suas dependências em *containers* (MERKEL, 2014). Docker utiliza um componente também utilizado pelo LXC chamado Control Groups (cgroups), que implementa contabilização e limitação de recursos. Cgroups possibilita Docker a limitar os recursos consumidos pelo *container*, como memória, disco e I/O, e também oferece métricas destes recursos. Soluções elásticas podem ser desenvolvidas utilizando essas tecnologias, já que existe um gerenciamento para a limitação dos recursos e métricas para monitoração, aspectos muito utilizados na elasticidade.

Por fim, como trabalho futuro, a utilização do FOLE em arquiteturas variadas, como plataformas de computação móvel e de Computação de Alto Desempenho, para verificar como a elasticidade se comporta nesses ambientes.

7.2.5 Ferramentas

Trabalhos de análise de desempenho em geral utilizam diversas ferramentas para a geração das cargas de trabalho sobre os ambientes, como os próprios *benchmarks*. Essas cargas de trabalho podem ser geradas a partir de diversas fontes, tais como: *benchmarks*, *micro-benchmarks*, traços computacionais e aplicações. Portanto, diversos experimentos podem ser projetados para a avaliação de desempenho da elasticidade com diferentes cargas de trabalho e ferramentas, sob as mais diversas situações. A revisão sistemática identificou diversas ferramentas, e a utilização de aplicativos e *benchmarks* mais tradicionais, como o TPC-C[4] e YCSB[5], em análises futuras são trabalhos a serem considerados.

Além disso, também pretende-se realizar um projeto da carga de trabalho que se aproxime de situações reais, e estender a validação do FOLE em aplicações e ambientes com aplicações reais, como *e-science*, *web*, aplicações biomédicas, ambientes virtuais de aprendizagem e Computação de Alto Desempenho. A criação de um *benchmark* que gere uma carga de trabalho padrão, realize o cálculo das métricas de elasticidade, e retorne o resultado como uma forma de comparar nuvens computacionais do ponto de vista da elasticidade é uma ferramenta a ser construída, e disponibilizada para a comunidade.

A ferramenta de visualização desenvolvida neste trabalho necessita de diversas melhorias e extensões: automação da coleta das métricas, novos gráficos, novas métricas e estatísticas. Além disso, alguns *microbenchmarks* foram desenvolvidos. Assim, suas melhorias podem auxiliar na análise de desempenho de ambientes computacionais, com foco em aspectos específicos, como processamento, banco de dados, leitura e escrita em disco, e rede.

O desenvolvimento de um *benchmark* orientado à elasticidade é uma atividade a ser

[4]TPC-C - http://www.tpc.org/tpcc/

[5]YCSB - http://labs.yahoo.com/news/yahoo-cloud-serving-benchmark/

futuramente projetada devido à carência na literatura de *benchmarks* específicos para a medição e avaliação da elasticidade. Assim seria possível a comparação entre ambientes computacionais de maneira mais padronizada.

Um outro trabalho futuro relacionado à ferramentas se refere à utilização de de ferramentas para a medição do consumo de energia de sistemas computacionais, tais como o Joulemeter[6], PowerTOP[7], Intel Energy Checker[8] e o VMmark[9], que dependem de dispositivos físicos conhecidos como wattímetros, como o WattsUp[10]. A ideia seria avaliar o custo energético de operações de elasticidade, e a utilização do FOLE e das métricas propostas para apoiar essa avaliação.

[6]Joulemeter - http://research.microsoft.com/en-us/downloads/fe9e10c5-5c5b-450c-a674-daf55565f794/
[7]PowerTOP - https://01.org/powertop/
[8]Intel Energy Checker - https://software.intel.com/en-us/articles/intel-energy-checker-sdk
[9]VMmark - https://www.vmware.com/products/vmmark/
[10]Watts Up? - https://www.wattsupmeters.com/secure/index.php

REFERÊNCIAS BIBLIOGRÁFICAS

ACETO, G. et al. Cloud monitoring: A survey. *Computer Networks*, n. 0, p. –, 2013. ISSN 1389-1286. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1389128613001084>.

AISOPOS, F.; TSERPES, K.; VARVARIGOU, T. Resource management in software as a service using the knapsack problem model. *International Journal of Production Economics*, n. 0, p. –, 2011. ISSN 0925-5273. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0925527311005275>.

ALI-ELDIN, A. et al. Efficient provisioning of bursty scientific workloads on the cloud using adaptive elasticity control. In: *Proceedings of the 3rd workshop on Scientific Cloud Computing Date*. New York, NY, USA: ACM, 2012. (ScienceCloud '12), p. 31–40. ISBN 978-1-4503-1340-7. Disponível em: <http://doi.acm.org/10.1145/2287036.2287044>.

AMAZONWEBSERVICES. *Amazon Web Services (Portuguese (Brazil))*. 2013. Http://aws.amazon.com/pt/. Online; acessado em janeiro-2012.

AZURE. *Microsoft Azure*. 2012. Http://www.microsoft.com/azure/.

BAI, X. et al. Cloud testing tools. In: *Service Oriented System Engineering (SOSE), 2011 IEEE 6th International Symposium on*. [S.l.: s.n.], 2011. p. 1 –12.

BANZAI, T. et al. D-cloud: Design of a software testing environment for reliable distributed systems using cloud computing technology. In: *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*. [S.l.: s.n.], 2010. p. 631 –636.

BENNETT, C. et al. Malstone: towards a benchmark for analytics on large data clouds. In: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM, 2010. (KDD '10), p. 145–152. ISBN 978-1-4503-0055-1. Disponível em: <http://doi.acm.org/10.1145/1835804.1835826>.

BRANTNER, M. et al. Building a database on s3. In: *Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD '08*. New York: ACM Press, 2008. p. 251. ISBN 9781605581026.

BRUNEO, D. A stochastic model to investigate data center performance and qos in iaas cloud computing systems. In: . [S.l.: s.n.], 2013. PP, n. 99, p. 1–1. ISSN 1045-9219.

BRYANT, R. et al. Kaleidoscope: cloud micro-elasticity via vm state coloring. In: *Proceedings of the sixth conference on Computer systems*. New York, NY, USA: ACM, 2011. (EuroSys '11), p. 273–286. ISBN 978-1-4503-0634-8. Disponível em: <http://doi.acm.org/10.1145/1966445.1966471>.

BUYYA, R.; CALHEIROS, R.; LI, X. Autonomic cloud computing: Open challenges and architectural elements. In: *Emerging Applications of Information Technology (EAIT), 2012 Third International Conference on*. [S.l.: s.n.], 2012. p. 3–10.

BUYYA, R. et al. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 25, n. 6, p. 599–616, 2009. ISSN 0167-739X.

CALHEIROS, R.; RANJAN, R.; BUYYA, R. Virtual machine provisioning based on analytical performance and qos in cloud computing environments. In: *Parallel Processing (ICPP), 2011 International Conference on*. [S.l.: s.n.], 2011. p. 295 –304. ISSN 0190-3918.

CALHEIROS, R. N. et al. A coordinator for scaling elastic applications across multiple clouds. *Future Generation Computer Systems*, v. 28, n. 8, p. 1350 – 1362, 2012. ISSN 0167-739X. Including Special sections SS: Trusting Software Behavior and SS: Economics of Computing Services. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X12000635>.

CALHEIROS, R. N. et al. The aneka platform and qos-driven resource provisioning for elastic applications on hybrid clouds. *Future Generation Computer Systems*, n. 0, p. –, 2011. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X11001397>.

CASALICCHIO, E.; SILVESTRI, L. Mechanisms for {SLA} provisioning in cloud-based service providers. *Computer Networks*, v. 57, n. 3, p. 795 – 810, 2013. ISSN 1389-1286. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1389128612003763>.

CECCHET, E. et al. Benchlab: an open testbed for realistic benchmarking of web applications. In: *Proceedings of the 2nd USENIX conference on Web application development*. Berkeley, CA, USA: USENIX Association, 2011. (WebApps'11), p. 4–4. Disponível em: <http://dl.acm.org/citation.cfm?id=2002168.2002172>.

COOPER, B. F. et al. Benchmarking cloud serving systems with ycsb. In: *Proceedings of the 1st ACM symposium on Cloud computing*. New York, NY, USA: ACM, 2010. (SoCC '10), p. 143–154. ISBN 978-1-4503-0036-0. Disponível em: <http://doi.acm.org/10.1145/1807128.1807152>.

COSTA, P. et al. Uma análise do impacto da qualidade da internet móvel na utilização de cloudlets. In: *XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2014)*. [S.l.: s.n.], 2014.

COSTA, R. et al. Sobre a amplitude da elasticidade dos provedores atuais de computação na nuvem. In: *SBRC 2011*. [S.l.: s.n.], 2011.

COSTA, R. et al. Analyzing the impact of elasticity on the profit of cloud computing providers. *Future Generation Computer Systems*, n. 0, p. –, 2013. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X13000058>.

COUTINHO, E.; GOMES, D. G.; SOUZA, J. D. Uma proposta de arquitetura autonômica para elasticidade em computação em nuvem. In: *IV Workshop de Sistemas Distribuídos Autonômicos - WOSIDA2014*. [S.l.: s.n.], 2014.

COUTINHO, E.; GOMES, D. G.; SOUZA, J. D. Uma proposta de framework conceitual para análise de desempenho da elasticidade em nuvens computacionais. In: *XII Workshop em Clouds e Aplicações (WCGA2014)*. Florianópolis - Brasil: [s.n.], 2014.

COUTINHO, E.; PAILLARD, G. Aplicacões de computação de alto desempenho e computação em nuvem na disciplina de desenvolvimento de aplicações distribuídas. In: *WSCAD-WEAC 2013*. Porto de Galinhas: [s.n.], 2013.

COUTINHO, E.; PAILLARD, G. High performance computing and cloud computing applications at the discipline of distributed applications development. In: *International Journal of Computer Architecture Education (IJCAE)*. [S.l.: s.n.], 2013.

COUTINHO, E. et al. Análise de desempenho com benchmarks em um ambiente público de computação em nuvem. In: *X Workshop em Clouds e Aplicações (WCGA2012)*. [S.l.: s.n.], 2012.

COUTINHO, E. et al. Uma métrica para avaliação da elasticidade em computação em nuvem baseada em conceitos da física. In: *XII Workshop em Clouds e Aplicações (WCGA2014)*. Florianópolis - Brasil: [s.n.], 2014.

COUTINHO, E. et al. Elasticidade em computação na nuvem: Uma abordagem sistemática. In: *XXXI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2013) - Minicursos*. [S.l.: s.n.], 2013.

COUTINHO, E. F.; GOMES, D. G.; SOUZA, J. N. d. An analysis of elasticity in cloud computing environments based on allocation time and resources. In: *Cloud Computing and Communications (LatinCloud), 2nd IEEE Latin American Conference on*. Maceió, Brazil: [s.n.], 2013. p. 7–12.

COUTINHO, E. F. et al. How to deploy a virtual learning environment in the cloud? In: *Proceedings of the 7th Euro American Conference on Telematics and Information Systems*. New York, NY, USA: ACM, 2014. (EATIS '14), p. 25:1–25:4. ISBN 978-1-4503-2435-9. Disponível em: <http://doi.acm.org/10.1145/2590651.2590675>.

COUTINHO, E. F.; PAILLARD, G.; SOUZA, J. N. de. Performance analysis on scientific computing and cloud computing environments. In: *Proceedings of the 7th Euro American Conference on Telematics and Information Systems*. New York, NY, USA: ACM, 2014. (EATIS '14), p. 5:1–5:6. ISBN 978-1-4503-2435-9. Disponível em: <http://doi.acm.org/10.1145/2590651.2590656>.

CRUZ, F. *Scrum e PMBOK unidos no Gerenciamento de Projetos*. [S.l.]: Brasport Livros e Multimídia Ltda., 2013. ISBN 978-85-7452-594-5.

CRUZ, F. et al. Met: workload aware elasticity for nosql. In: *Proceedings of the 8th ACM European Conference on Computer Systems*. New York, NY, USA: ACM, 2013. (EuroSys '13), p. 183–196. ISBN 978-1-4503-1994-2. Disponível em: <http://doi.acm.org/10.1145/2465351.2465370>.

CUNHA, M.; MENDONÇA, N.; SAMPAIO, A. Cloud crawler: Um ambiente programável para avaliar o desempenho de aplicações em nuvens de infraestrutura. In: *XXXI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2013)*. [S.l.: s.n.], 2013.

DAS, S.; AGRAWAL, D.; ABBADI, A. E. Elastras: An elastic, scalable, and self-managing transactional database for the cloud. *ACM Trans. Database Syst.*, ACM, New York, NY, USA, v. 38, n. 1, p. 5:1–5:45, abr. 2013. ISSN 0362-5915. Disponível em: <http://doi.acm.org/http://dx.doi.org/10.1145/2445583.2445588>.

DAWOUD, W.; TAKOUNA, I.; MEINEL, C. Elastic vm for rapid and optimum virtualized resources' allocation. In: *Systems and Virtualization Management (SVM), 2011 5th International DMTF Academic Alliance Workshop on*. [S.l.: s.n.], 2011. p. 1 –4.

DENG, N. et al. Adaptive green hosting. In: *Proceedings of the 9th international conference on Autonomic computing*. New York, NY, USA: ACM, 2012. (ICAC '12), p. 135–144. ISBN 978-1-4503-1520-3. Disponível em: <http://doi.acm.org/10.1145/2371536.2371561>.

DENKO, M.; YANG, L. T.; ZHANG, Y. *Autonomic Computing and Networking*. [S.l.]: Springer, 2009. ISBN 978-0-387-89828-5.

EL-REFAEY, M.; RIZKAA, M. Cloudgauge: A dynamic cloud and virtualization benchmarking suite. In: *Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), 2010 19th IEEE International Workshop on*. [S.l.: s.n.], 2010. p. 66 –75. ISSN 1524-4547.

EMEAKAROHA, V. C. et al. Low level metrics to high level slas - lom2his framework: Bridging the gap between monitored metrics and sla parameters in cloud environments. In: SMARI, W. W.; MCINTIRE, J. P. (Ed.). *Proceedings of the 2010 International Conference on High Performance Computing & Simulation, HPCS 2010, June 28 - July 2, 2010, Caen, France*. [S.l.]: IEEE, 2010. p. 48–54. ISBN 978-1-4244-6828-7.

EMEAKAROHA, V. C. et al. Towards autonomic detection of sla violations in cloud infrastructures. *Future Generation Computer Systems*, v. 28, n. 7, p. 1017 – 1029, 2012. ISSN 0167-739X. Special section: Quality of Service in Grid and Cloud Computing</ce:title>. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X11002184>.

ENDO, P. et al. Resource allocation for distributed cloud: concepts and research challenges. *Network, IEEE*, v. 25, n. 4, p. 42–46, July 2011. ISSN 0890-8044.

ESPADAS, J. et al. A tenant-based resource allocation model for scaling software-as-a-service applications over cloud computing infrastructures. *Future Generation Computer Systems*, n. 0, p. –, 2011. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X1100210X>.

ETCHEVERS, X. et al. Automated configuration of legacy applications in the cloud. In: *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*. [S.l.: s.n.], 2011. p. 170 –177.

FERRER, A. J. et al. Optimis: A holistic approach to cloud service provisioning. *Future Generation Computer Systems*, v. 28, n. 1, p. 66 – 77, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X1100104X>.

FITO, J.; GOIRI, I.; GUITART, J. Sla-driven elastic cloud hosting provider. In: *Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on*. [S.l.: s.n.], 2010. p. 111 –118. ISSN 1066-6192.

FLORES, H.; SRIRAMA, S. N.; PANIAGUA, C. A generic middleware framework for handling process intensive hybrid cloud services from mobiles. In: *Proceedings of the 9th International Conference on Advances in Mobile Computing and Multimedia*. New York, NY, USA: ACM, 2011. (MoMM '11), p. 87–94. ISBN 978-1-4503-0785-7. Disponível em: <http://doi.acm.org/10.1145/2095697.2095715>.

GALANTE, G.; BONA, L. C. E. de. A survey on cloud computing elasticity. In: *Proceedings of the 5th IEEE/ACM International Conference on Utility and Cloud Computing (UCC '12)*. [S.l.: s.n.], 2012.

GAMBI, A. et al. Testing elastic computing systems. *Internet Computing, IEEE*, v. 17, n. 6, p. 76–82, 2013. ISSN 1089-7801.

GAO, J. et al. Saas performance and scalability evaluation in clouds. In: *Service Oriented System Engineering (SOSE), 2011 IEEE 6th International Symposium on*. [S.l.: s.n.], 2011. p. 61 –71.

GARG, S.; VERSTEEG, S.; BUYYA, R. Smicloud: A framework for comparing and ranking cloud services. In: *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*. [S.l.: s.n.], 2011. p. 210 –218.

GARG, S. K.; VERSTEEG, S.; BUYYA, R. A framework for ranking of cloud computing services. *Future Generation Computer Systems*, n. 0, p. –, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X12001422?v=s5>.

GHANBARI, H. et al. Exploring alternative approaches to implement an elasticity policy. In: *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. [S.l.: s.n.], 2011. p. 716 –723. ISSN 2159-6182.

GHOSHAL, D.; RAMAKRISHNAN, L. Frieda: Flexible robust intelligent elastic data management in cloud environments. In: *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:*. [S.l.: s.n.], 2012. p. 1096–1105.

GOOGLE. *Google App Engine*. 2014. Https://appengine.google.com/. Online; acessado em julho-2014.

HAN, R. et al. Enabling cost-aware and adaptive elasticity of multi-tier cloud applications. *Future Generation Computer Systems*, n. 0, p. –, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X12001148?v=s5>.

HARIRI, S. et al. The autonomic computing paradigm. *Cluster Computing*, Kluwer Academic Publishers, v. 9, n. 1, p. 5–17, 2006. ISSN 1386-7857. Disponível em: <http://dx.doi.org/10.1007/s10586-006-4893-0>.

HE, Q. et al. Case study for running hpc applications in public clouds. In: *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. New York, NY, USA: ACM, 2010. (HPDC '10), p. 395–401. ISBN 978-1-60558-942-8. Disponível em: <http://doi.acm.org/10.1145/1851476.1851535>.

HE, S.; GUO, L.; GUO, Y. Real time elastic cloud management for limited resources. In: *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. [S.l.: s.n.], 2011. p. 622–629. ISSN 2159-6182.

HERBST, N. R.; KOUNEV, S.; REUSSNER, R. Elasticity in cloud computing: What it is, and what it is not. In: *Proceedings of the 10th International Conference on Autonomic Computing(ICAC 2013), San Jose, CA*. [S.l.]: USENIX, 2013. p. 23–27.

HONG, Y.-J.; XUE, J.; THOTTETHODI, M. Selective commitment and selective margin: Techniques to minimize cost in an iaas cloud. In: *Performance Analysis of Systems and Software (ISPASS), 2012 IEEE International Symposium on*. [S.l.: s.n.], 2012. p. 99 –109.

HP. *HP Cloud Services: Now in Private Beta | HP Cloud Services*. 2013. Http://hpcloud.com/. Online; acessado em fevereiro-2013.

IBM. *A Pratical Guide to the IBM Autonomic Computing Toolkit Toolkit*. [S.l.], 2004. Disponível em: <http://www.redbooks.ibm.com/abstracts/sg246635.html?Open>.

IBM. *An Architectural Blueprint for Autonomic Computing*. [S.l.], 2005.

ISLAM, S. et al. How a consumer can measure elasticity for cloud platforms. In: *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*. New York, NY, USA: ACM, 2012. (ICPE '12), p. 85–96. ISBN 978-1-4503-1202-8. Disponível em: <http://doi.acm.org/10.1145/2188286.2188301>.

JAIN, R. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. 1st. ed. [S.l.]: John Wiley and Sons, INC, 1991. 685 p. ISBN 1-471-50336-3.

JAIN, T.; OHRI, V. K. *Introductory Microeconomics and Macroeconomics*. Vk Publications, 2012. ISBN 9788187344216. Disponível em: <http://books.google.com.br/books?id=4W68ooHGK18C>.

KEPHART, J. O.; CHESS, D. M. The vision of autonomic computing. *Computer*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 36, n. 1, p. 41–50, 2003. ISSN 0018-9162.

KEPHART, J. O.; WALSH, W. E. An artificial intelligence perspective on autonomic computing policies. In: *POLICY*. IEEE Computer Society, 2004. p. 3–12. ISBN 0-7695-2141-X. Disponível em: <http://dblp.uni-trier.de/db/conf/policy/policy2004.html#KephartW04>.

KITCHENHAM, B. *Procedures for performing systematic reviews*. [S.l.], 2004.

KONSTANTELI, K. et al. Admission control for elastic cloud services. In: *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. [S.l.: s.n.], 2012. p. 41–48. ISSN 2159-6182.

KOSSMANN, D.; KRASKA, T.; LOESING, S. An evaluation of alternative architectures for transaction processing in the cloud. In: *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. New York, NY, USA: ACM, 2010. (SIGMOD '10), p. 579–590. ISBN 978-1-4503-0032-2. Disponível em: <http://doi.acm.org/10.1145/1807167.1807231>.

KOUKI, Y.; LEDOUX, T. Scaling: Sla-driven cloud auto-scaling. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. New York, NY, USA: ACM, 2013. (SAC '13), p. 411–414. ISBN 978-1-4503-1656-9. Disponível em: <http://doi.acm.org/10.1145/2480362.2480445>.

KOUSIOURIS, G. et al. Dynamic, behavioral-based estimation of resource provisioning based on high-level application terms in cloud platforms. *Future Generation Computer Systems*, n. 0, p. –, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X12001057>.

KREBS, R.; MOMM, C.; KOUNEV, S. Metrics and techniques for quantifying performance isolation in cloud environments. In: *Proceedings of the 8th international ACM SIGSOFT conference on Quality of Software Architectures*. New York, NY, USA: ACM, 2012. (QoSA '12), p. 91–100. ISBN 978-1-4503-1346-9. Disponível em: <http://doi.acm.org/10.1145/2304696.2304713>.

KUMAR, D.; SHAE, Z.-Y.; JAMJOOM, H. Scheduling batch and heterogeneous jobs with runtime elasticity in a parallel processing environment. In: *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2012 IEEE 26th International*. [S.l.: s.n.], 2012. p. 65–78.

LI, A. et al. Cloudcmp: comparing public cloud providers. In: *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2010. (IMC '10), p. 1–14. ISBN 978-1-4503-0483-2. Disponível em: <http://doi.acm.org/10.1145/1879141.1879143>.

LI, J. et al. Cyberguarder: A virtualization security assurance architecture for green cloud computing. *Future Generation Computer Systems*, v. 28, n. 2, p. 379 – 390, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X1100063X>.

LI, M. et al. A scalable and elastic publish/subscribe service. In: *Parallel Distributed Processing Symposium (IPDPS), 2011 IEEE International*. [S.l.: s.n.], 2011. p. 1254 –1265. ISSN 1530-2075.

LI, W.; TORDSSON, J.; ELMROTH, E. Modeling for dynamic cloud scheduling via migration of virtual machines. In: *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*. [S.l.: s.n.], 2011. p. 163 –171.

LI, Z. et al. On a catalogue of metrics for evaluating commercial cloud services. In: *Proceedings of the 2012 ACM/IEEE 13th International Conference on Grid Computing*. Washington, DC, USA: IEEE Computer Society, 2012. (GRID '12), p. 164–173. ISBN 978-0-7695-4815-9. Disponível em: <http://dx.doi.org/10.1109/Grid.2012.15>.

LI, Z. et al. On evaluating commercial cloud services: A systematic review. *Journal of Systems and Software*, n. 0, p. –, 2013. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121213000915>.

LIU, S.; LIANG, Y.; BROOKS, M. Eucalyptus: a web service-enabled e-infrastructure. In: *CASCON '07: Proceedings of the 2007 conference of the center for advanced studies on Collaborative research*. New York, NY, USA: ACM, 2007. p. 1–11.

LU, W.; JACKSON, J.; BARGA, R. Azureblast: a case study of developing science applications on the cloud. In: *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. New York, NY, USA: ACM, 2010. (HPDC '10), p. 413–420. ISBN 978-1-60558-942-8. Disponível em: <http://doi.acm.org/10.1145/1851476.1851537>.

LUCAS-SIMARRO, J. L. et al. Scheduling strategies for optimal service deployment across multiple clouds. *Future Generation Computer Systems*, n. 0, p. –, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X12000192>.

MA, R. et al. A stack-on-demand execution model for elastic computing. In: *Parallel Processing (ICPP), 2010 39th International Conference on*. [S.l.: s.n.], 2010. p. 208 –217. ISSN 0190-3918.

MALIK, S.; HUET, F.; CAROMEL, D. Racs: A framework for resource aware cloud computing. In: *Internet Technology And Secured Transactions, 2012 International Conferece For*. [S.l.: s.n.], 2012. p. 680–687.

MANKIW, N. G. *Principles of Microeconomics*. 6th. ed. [S.l.]: CENGAGE Learning, 2012. ISBN 978-0538453042.

MAUCH, V.; KUNZE, M.; HILLENBRAND, M. High performance cloud computing. *Future Generation Computer Systems*, n. 0, p. –, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X12000647>.

MELL, P.; GRANCE, T. The nist definition of cloud computing. *National Institute of Standards and Technology*, NIST, v. 53, n. 6, p. 50, 2009. Disponível em: <http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc>.

MENASCE, D. A.; DOWDY, L. W.; ALMEIDA, V. A. F. *Performance by Design: Computer Capacity Planning By Example*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2004. ISBN 0130906735.

MERKEL, D. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, Belltown Media, Houston, TX, v. 2014, n. 239, mar. 2014. ISSN 1075-3583. Disponível em: <http://dl.acm.org/citation.cfm?id=2600239.2600241>.

MICHON, E.; GOSSA, J.; GENAUD, S. Free elasticity and free cpu power for scientific workloads on iaas clouds. In: *Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on*. [S.l.: s.n.], 2012. p. 85–92. ISSN 1521-9097.

MICROSOFT. *Azure: Plataforma em Nuvem da Microsoft*. 2014. Https://azure.microsoft.com/pt-br/. Online; acessado em julho-2014.

MOLDOVAN, D. et al. Mela: Monitoring and analyzing elasticity of cloud services. In: *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*. [S.l.: s.n.], 2013. v. 1, p. 80–87.

MONTERO, R. S.; MORENO-VOZMEDIANO, R.; LLORENTE, I. M. An elasticity model for high throughput computing clusters. *Journal of Parallel and Distributed Computing*, v. 71, n. 6, p. 750 – 757, 2011. ISSN 0743-7315. Special Issue on Cloud Computing. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0743731510000985>.

MONTGOMERY, D. C. *Design and analysis of experiments*. 7th. ed. [S.l.]: John Wiley and Sons, INC, 2009. 656 p. ISBN 9780470128664.

MOREIRA, L. O. et al. Towards improvements on the quality of service for multi-tenant rdbms in the cloud. In: *Data Engineering Workshops (ICDEW), 2014 IEEE 30th International Conference on*. [S.l.: s.n.], 2014. p. 162–169.

MOREIRA, L. O. et al. A live migration approach for multi-tenant rdbms in the cloud. In: *Simpósio Brasileiro de Banco de Dados (SBBD)*. [S.l.: s.n.], 2013.

MORENO-VOZMEDIANO, R.; MONTERO, R.; LLORENTE, I. Multicloud deployment of computing clusters for loosely coupled mtc applications. *Parallel and Distributed Systems, IEEE Transactions on*, v. 22, n. 6, p. 924 –930, june 2011. ISSN 1045-9219.

MORENO-VOZMEDIANO, R.; MONTERO, R. S.; LLORENTE, I. M. Elastic management of cluster-based services in the cloud. In: *Proceedings of the 1st workshop on Automated control for datacenters and clouds*. New York, NY, USA: ACM, 2009. (ACDC '09), p. 19–24. ISBN 978-1-60558-585-7. Disponível em: <http://doi.acm.org/10.1145/1555271.1555277>.

NCBI, N. C. for B. I. *National Center for Biotechnology Information*. 2013. Http://www.ncbi.nlm.nih.gov/. Online; October-2013.

NIE, L.; XU, Z. An adaptive scheduling mechanism for elastic grid computing. In: *Semantics, Knowledge and Grid, 2009. SKG 2009. Fifth International Conference on*. [S.l.: s.n.], 2009. p. 184 –191.

NIEHORSTER, O. et al. Autonomic resource management with support vector machines. In: *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*. Washington, DC, USA: IEEE Computer Society, 2011. (GRID '11), p. 157–164. ISBN 978-0-7695-4572-1. Disponível em: <http://dx.doi.org/10.1109/Grid.2011.28>.

NIU, D. et al. Quality-assured cloud bandwidth auto-scaling for video-on-demand applications. In: *INFOCOM, 2012 Proceedings IEEE*. [S.l.: s.n.], 2012. p. 460 –468. ISSN 0743-166X.

OPENNEBULA.ORG. *.:: OpenNebula: The Open Source Toolkit for Data Center Virtualization ::.* 2012. Http://opennebula.org/. Online; acessado em janeiro-2012.

OPENSTACK.ORG. *OpenStack Open Source Cloud Computing Software*. Março 2013. Http://openstack.org/. Online; acessado em março-2013.

OTTO, J.; STANOJEVIC, R.; LAOUTARIS, N. Temporal rate limiting: Cloud elasticity at a flat fee. In: *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*. [S.l.: s.n.], 2012. p. 151 –156.

PANDEY, S. et al. An autonomic cloud environment for hosting ecg data analysis services. *Future Generation Computer Systems*, v. 28, n. 1, p. 147 – 154, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X11000732>.

PANIAGUA, C.; SRIRAMA, S. N.; FLORES, H. Bakabs: managing load of cloud-based web applications from mobiles. In: *Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services*. New York, NY, USA: ACM, 2011. (iiWAS '11), p. 485–490. ISBN 978-1-4503-0784-0. Disponível em: <http://doi.acm.org/10.1145/2095536.2095636>.

PARASHAR, M.; HARIRI, S. *Autonomic Computing: Concepts, Infrastructure, and Applications*. [S.l.]: CRC Press, 2006.

PAWLUK, P. et al. Introducing stratos: A cloud broker service. In: *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. [S.l.: s.n.], 2012. p. 891–898. ISSN 2159-6182.

PEREZ-SORROSAL, F. et al. Elastic si-cache: consistent and scalable caching in multi-tier architectures. *The VLDB Journal*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 20, n. 6, p. 841–865, dez. 2011. ISSN 1066-8888. Disponível em: <http://dx.doi.org/10.1007/s00778-011-0228-8>.

QIN, X. et al. Elasticat: A load rebalancing framework for cloud-based key-value stores. In: *High Performance Computing (HiPC), 2012 19th International Conference on*. [S.l.: s.n.], 2012. p. 1–10.

RAVEENDRAN, A.; BICER, T.; AGRAWAL, G. A framework for elastic execution of existing mpi programs. In: *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*. [S.l.: s.n.], 2011. p. 940 –947. ISSN 1530-2075.

REGO, P. et al. Faircpu: Architecture for allocation of virtual machines using processing features. In: *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*. [S.l.: s.n.], 2011. p. 371–376.

REGO, P. et al. Alocação autonômica de recursos para máquinas virtuais baseada em características de processamento. In: *II Workshop de Sistemas Distribuídos Autonômicos - WOSIDA2012*. [S.l.: s.n.], 2012.

REGO, P.; COUTINHO, E.; SOUZA, J. D. Proposta de workflow para alocação de máquinas virtuais utilizando características de processamento. In: *IX Workshop em Clouds e Aplicações (WCGA2011)*. [S.l.: s.n.], 2011.

REGO, P.; COUTINHO, E.; SOUZA, J. D. Estratégias para alocação dinâmica de recursos em um ambiente híbrido de computação em nuvem. In: *XII Workshop em Clouds e Aplicações (WCGA2013)*. [S.l.: s.n.], 2013.

RUSSELL, R.; CHUNG, M.; BALK, E. *Issues and Challenges in Conducting Systematic Reviews to Support Development of Nutrient Reference Values*. [S.l.], 2009.

SALVATORE, D. *Managerial Economics in a Global Economy*. 7th. ed. [S.l.]: Oxford University Press, USA, 2011. ISBN 978-0199811786.

SAMUELSON, W. F.; MARKS, S. G. *Managerial Economics*. 7th. ed. [S.l.]: JOHN WILEY & SONS, INC., 2012. ISBN 978-1118041581.

SANTOS, G. et al. Scale-space filtering for workload analysis and forecast. In: *Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference on*. [S.l.: s.n.], 2013. p. 677–684.

SCHEEPERS, M. Virtualization and containerization of application infrastructure: A comparison. In: . [S.l.]: University of Twente, 2014. v. 21, n. June 23.

SCOPE. *SCOPE Alliance Telecom Grade Cloud Computing*. [S.l.], 2011. Disponível em: <http://scope-alliance.org/>.

SHARMA, U. et al. A cost-aware elasticity provisioning system for the cloud. In: *International Conference on Distributed Computing Systems (ICDCS)*. [S.l.: s.n.], 2011. p. 559–570.

SHAWKY, D. M.; ALI, A. F. Defining a measure of cloud computing elasticity. In: *Systems and Computer Science (ICSCS), 2012 1st International Conference on*. [S.l.: s.n.], 2012. p. 1–5.

SIMÕES, R.; KAMIENSKI, C. Gerenciamento de elasticidade em nuvens privadas e híbridas. In: *XII Workshop em Clouds e Aplicações (WCGA2014)*. Florianópolis - Brasil: [s.n.], 2014.

SLOMAN, M. Policy driven management for distributed systems. *Journal of Network and Systems Management*, v. 2, p. 333–360, 1994.

SOBEL, W. et al. *Cloudstone: Multi-platform, multi-language benchmark and measurement tools for web 2.0*. 2008.

SOROR, A. A. et al. Automatic virtual machine configuration for database workloads. *ACM Trans. Database Syst.*, ACM, New York, NY, USA, v. 35, n. 1, p. 1–47, 2010. ISSN 0362-5915.

SOUSA, F.; MACHADO, J. Towards elastic multi-tenant database replication with quality of service. In: *Utility and Cloud Computing (UCC), 2012 IEEE Fifth International Conference on*. [S.l.: s.n.], 2012. p. 168–175.

SOUSA, F. R. Computação autônoma. *Monografia de Qualificação de Doutorado*, 2010.

SULEIMAN, B. et al. On understanding the economics and elasticity challenges of deploying business applications on public cloud infrastructure. *J. Internet Services and Applications (JISA)*, v. 3, n. 2, p. 173–193, 2012.

TIMOSHENKO, S. P.; GOODIER, J. *Theory of Elasticity*. 3rd. ed. [S.l.]: McGraw-Hill Education, 1970. 608 p. ISBN 0070642702.

TIRADO, J. et al. Predictive data grouping and placement for cloud-based elastic server infrastructures. In: *Cluster, Cloud and Grid Computing (CCGrid), 2011 11th IEEE/ACM International Symposium on*. [S.l.: s.n.], 2011. p. 285 –294.

TOOSI, A. et al. Resource provisioning policies to increase iaas provider's profit in a federated cloud environment. In: *High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on*. [S.l.: s.n.], 2011. p. 279–287.

TORDSSON, J. et al. Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future Generation Computer Systems*, v. 28, n. 2, p. 358 – 367, 2012. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X11001373>.

TUDORAN, R. et al. A performance evaluation of azure and nimbus clouds for scientific applications. In: *Proceedings of the 2nd International Workshop on Cloud Computing Platforms*. New York, NY, USA: ACM, 2012. (CloudCP '12), p. 4:1–4:6. ISBN 978-1-4503-1161-8. Disponível em: <http://doi.acm.org/10.1145/2168697.2168701>.

URIARTE, R. B.; WESTPHALL, C. B. Panoptes: A monitoring architecture and framework for supporting autonomic clouds. In: *Network Operations and Management Symposium (NOMS), 2014 IEEE*. [S.l.: s.n.], 2014. p. 1–5.

VAQUERO, L. M.; RODERO-MERINO, L.; BUYYA, R. Dynamically scaling applications in the cloud. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 41, n. 1, p. 45–52, 2011. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/1925861.1925869>.

VAQUERO, L. M. et al. A break in the clouds: towards a cloud definition. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 39, n. 1, p. 50–55, 2009. ISSN 0146-4833.

VERDI, F. L. et al. Novas arquiteturas de data center para cloud computing. In: ____. Gramado, RS: SBC, 2010. in Minicursos do XXVIII Simpósio Brasileiro de Redes de Computadores - SBRC2010, p. 103–152.

WEE, S.; LIU, H. Client-side load balancer using cloud. In: *Proceedings of the 2010 ACM Symposium on Applied Computing*. New York, NY, USA: ACM, 2010. (SAC '10), p. 399–405. ISBN 978-1-60558-639-7. Disponível em: <http://doi.acm.org/10.1145/1774088.1774173>.

WHITE, S. R. et al. An architectural approach to autonomic computing. In: *ICAC*. IEEE Computer Society, 2004. p. 2–9. ISBN 0-7695-2114-2. Disponível em: <http://dblp.uni-trier.de/db/conf/icac/icac2004.html/#WhiteHWCK04>.

WONG, A. K.; GOSCINSKI, A. M. A vmd plugin for namd simulations on amazon ec2. *Procedia Computer Science*, v. 9, n. 0, p. 136 – 145, 2012. ISSN 1877-0509. Proceedings of the International Conference on Computational Science, ICCS 2012. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050912001366>.

XU, H.; LI, B. A study of pricing for cloud resources. *SIGMETRICS Perform. Eval. Rev.*, ACM, New York, NY, USA, v. 40, n. 4, p. 3–12, abr. 2013. ISSN 0163-5999. Disponível em: <http://doi.acm.org/10.1145/2479942.2479944>.

YIGITBASI, N. et al. C-meter: A framework for performance analysis of computing clouds. In: *Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on*. [S.l.: s.n.], 2009. p. 472 –477.

YU, T. et al. Intelligent database placement in cloud environment. In: *Web Services (ICWS), 2012 IEEE 19th International Conference on*. [S.l.: s.n.], 2012. p. 544–551.

ZHAI, Y. et al. Cloud versus in-house cluster: evaluating amazon cluster compute instances for running mpi applications. In: *State of the Practice Reports*. New York, NY, USA: ACM, 2011. (SC '11), p. 11:1–11:10. ISBN 978-1-4503-1139-7. Disponível em: <http://doi.acm.org/10.1145/2063348.2063363>.

ZHAO, L.; SAKR, S.; LIU, A. A framework for consumer-centric sla management of cloud-hosted databases. *IEEE Transactions on Services Computing*, IEEE Computer Society, Los Alamitos, CA, USA, v. 99, n. PrePrints, p. 1, 2013. ISSN 1939-1374.

APÊNDICE A – MÉTRICAS COMPLEMENTARES

Tabela A.1: Lista de variáveis e nomenclatura para métricas complementares baseadas em tempos de alocação de recursos

Variável	Descrição
T_{Total}	Tempo Total do experimento
$\%TASo = \frac{TASo}{T_{Total}}$	Percentual do tempo de alocação sobreprovisionado
$\%TASu = \frac{TASu}{T_{Total}}$	Percentual do tempo de alocação subprovisionado
$\%TAE = \frac{TAE}{T_{Total}}$	Percentual do tempo de alocação estabilizado
$\%TAT = \frac{TAT}{T_{Total}}$	Percentual do tempo de alocação transitório
I_{TASo}	Total de intervalos de tempo de alocação sobreprovisionado
I_{TASu}	Total de intervalos de tempo de alocação subprovisionado
I_{TAE}	Total de intervalos de tempo de alocação estabilizada
I_{TAT}	Total de intervalos de tempo de alocação transitório
$\overline{T_{TASo}} = \frac{TASo}{I_{TASo}}$	Tempo médio de alocação sobreprovisionado
$\overline{T_{TASu}} = \frac{TASu}{I_{TASu}}$	Tempo médio de alocação subprovisionado
$\overline{T_{TAE}} = \frac{TAE}{I_{TAE}}$	Tempo médio de alocação estabilizado
$\overline{T_{TAT}} = \frac{TAT}{I_{TAT}}$	Tempo médio de alocação transitório

Tabela A.2: Lista de variáveis e nomenclatura para métricas complementares baseadas em quantidade de recursos

Variável	**Descrição**
R_{Total}	Total de recursos alocados
I_{TRASo}	Total de intervalos de tempo de alocação sobreprovisionada
I_{TRASu}	Total de intervalos de tempo de alocação subprovisionada
I_{TRAE}	Total de intervalos de tempo de alocação estabilizada
$\overline{R_{TRASo}} = \frac{TRASo}{I_{TRASo}}$	Média de recursos alocados sobreprovisionados
$\overline{R_{TRASu}} = \frac{TRASu}{I_{TRASu}}$	Média de recursos alocados subprovisionados
$\overline{R_{TRAE}} = \frac{TRAE}{I_{TRAE}}$	Média de recursos alocados estabilizados

APÊNDICE B – FERRAMENTA

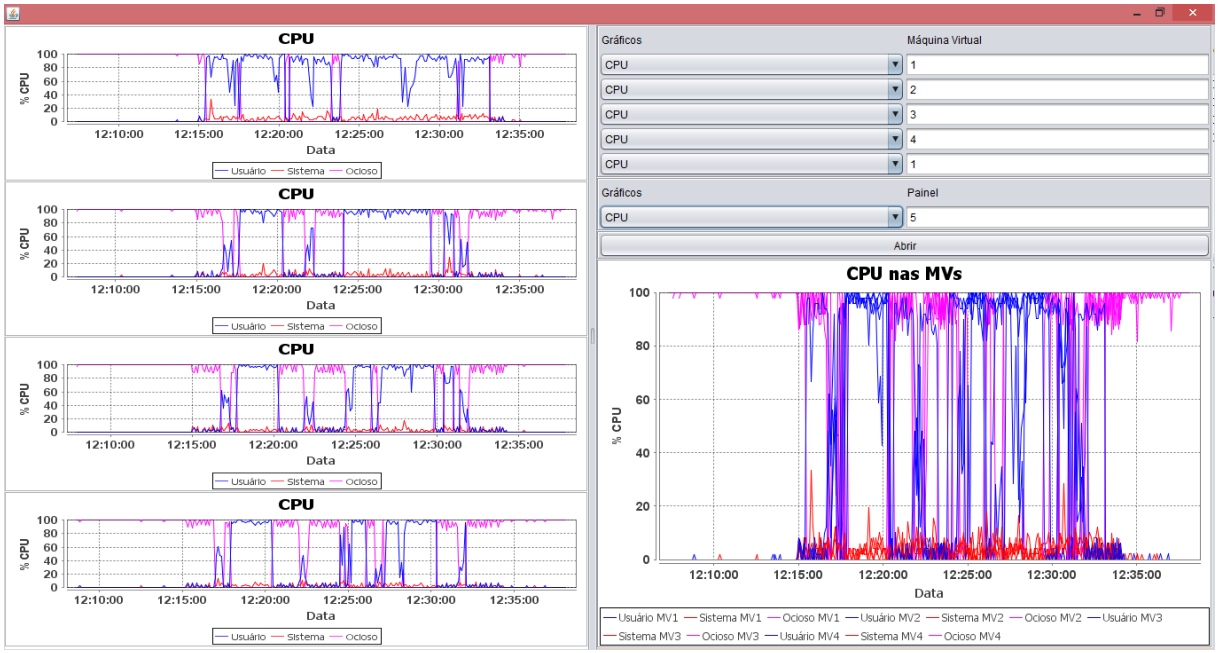


Figura B.1: Tela com o consumo de CPU de um experimento individual por máquina virtual e com todas as máquinas virtuais em paralelo

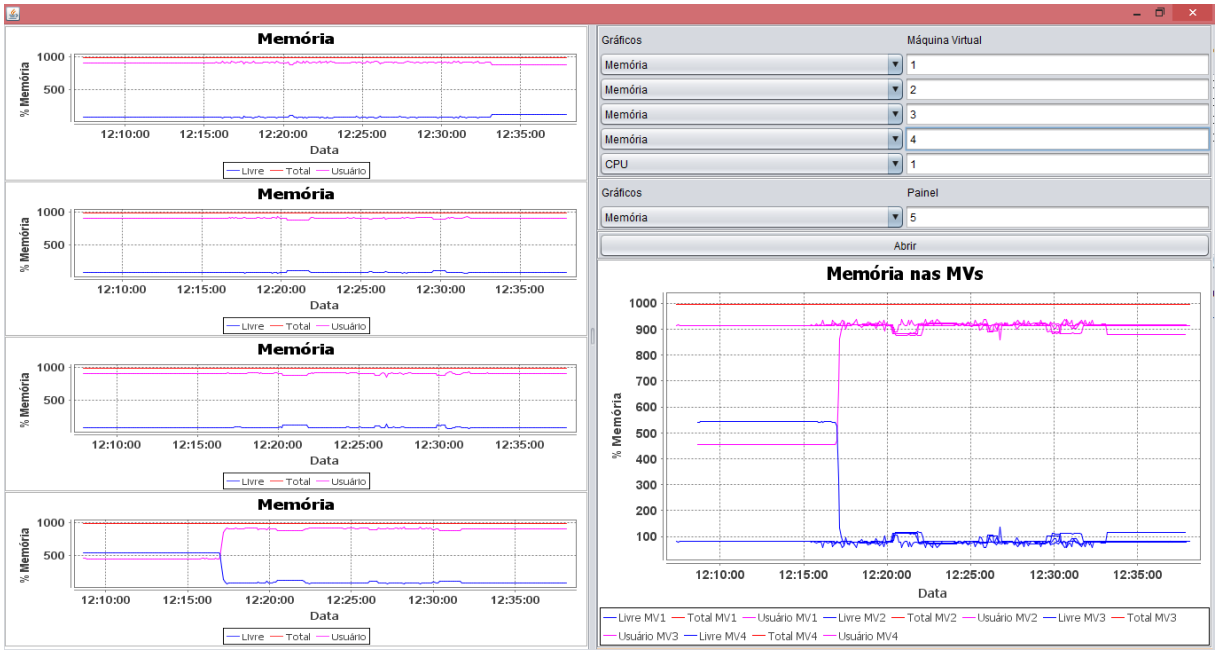


Figura B.2: Tela com o consumo de memória de um experimento individual por máquina virtual e com todas as máquinas virtuais em paralelo

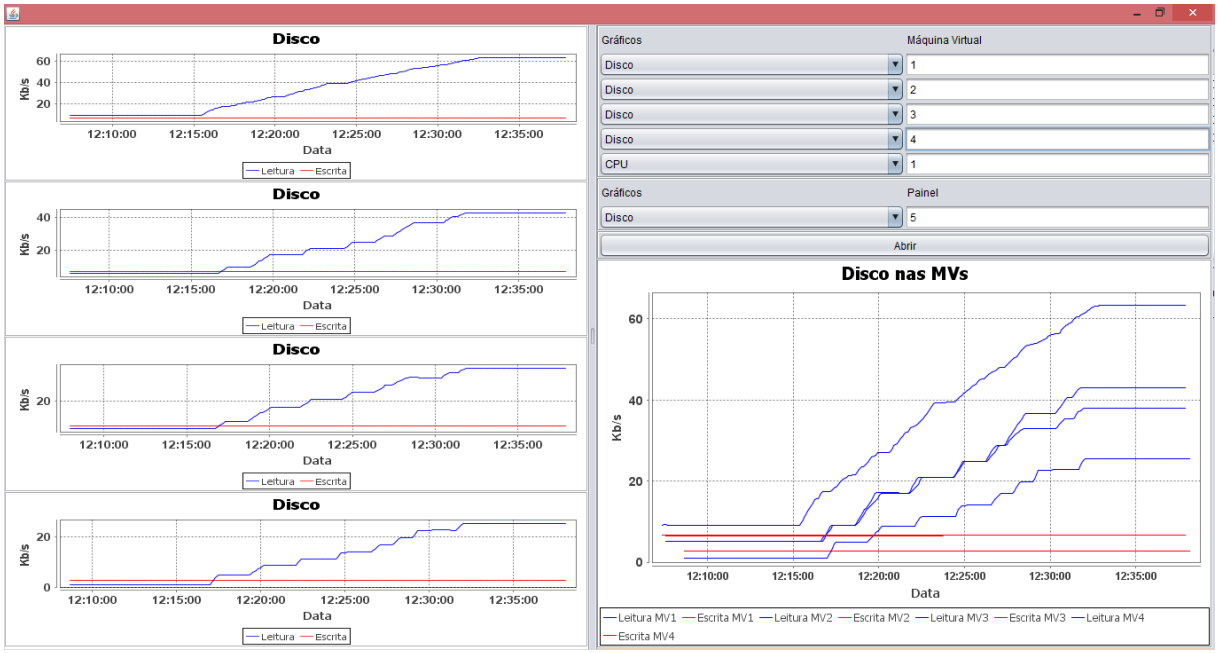


Figura B.3: Tela com a leitura e escrita em disco de um experimento individual por máquina virtual e com todas as máquinas virtuais em paralelo

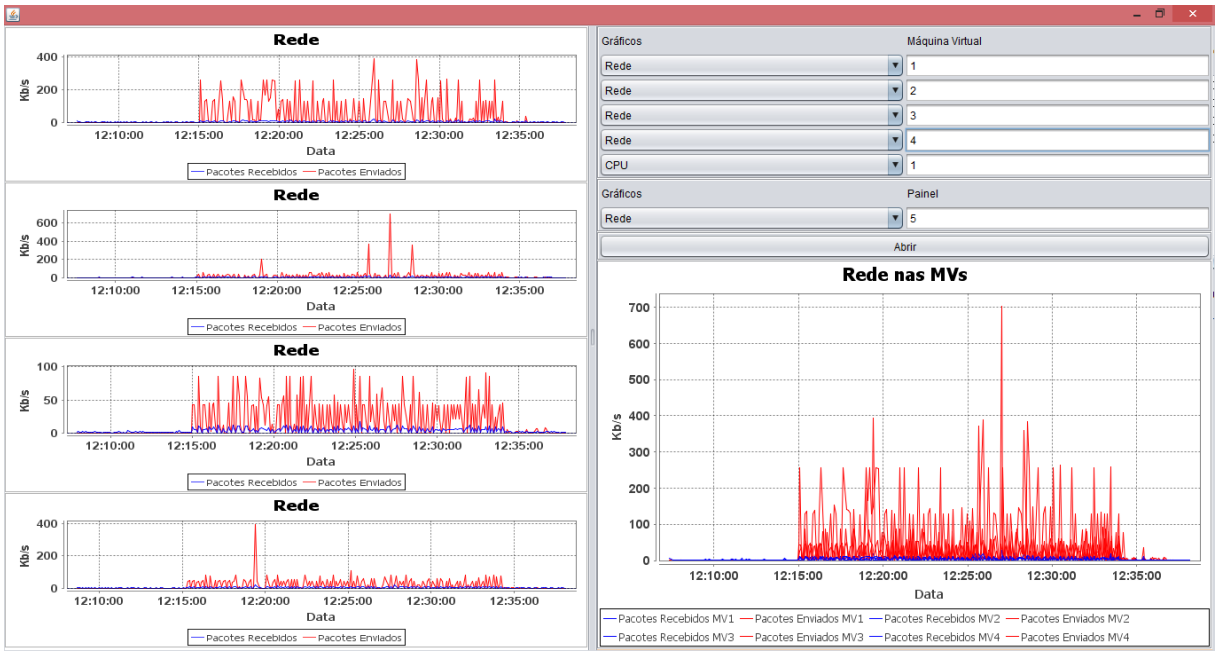


Figura B.4: Tela com pacotes enviados e pacotes recebidos de um experimento individual por máquina virtual e com todas as máquinas virtuais em paralelo

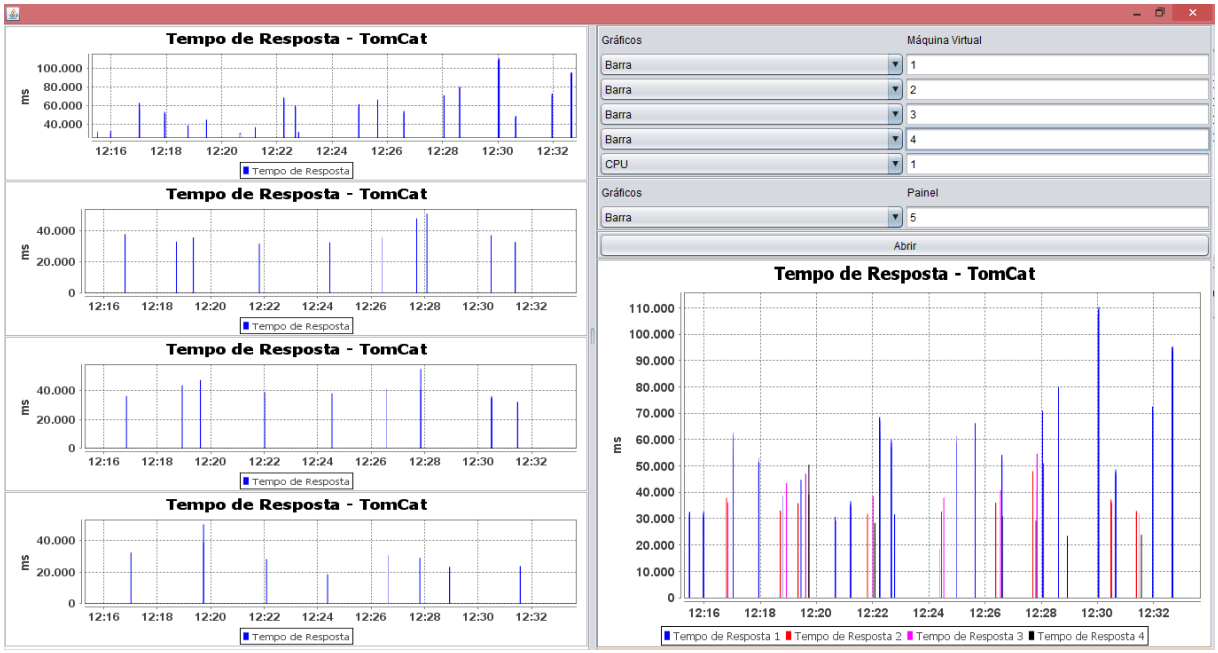


Figura B.5: Tela com tempo de resposta de um experimento individual por máquina virtual e com todas as máquinas virtuais em paralelo

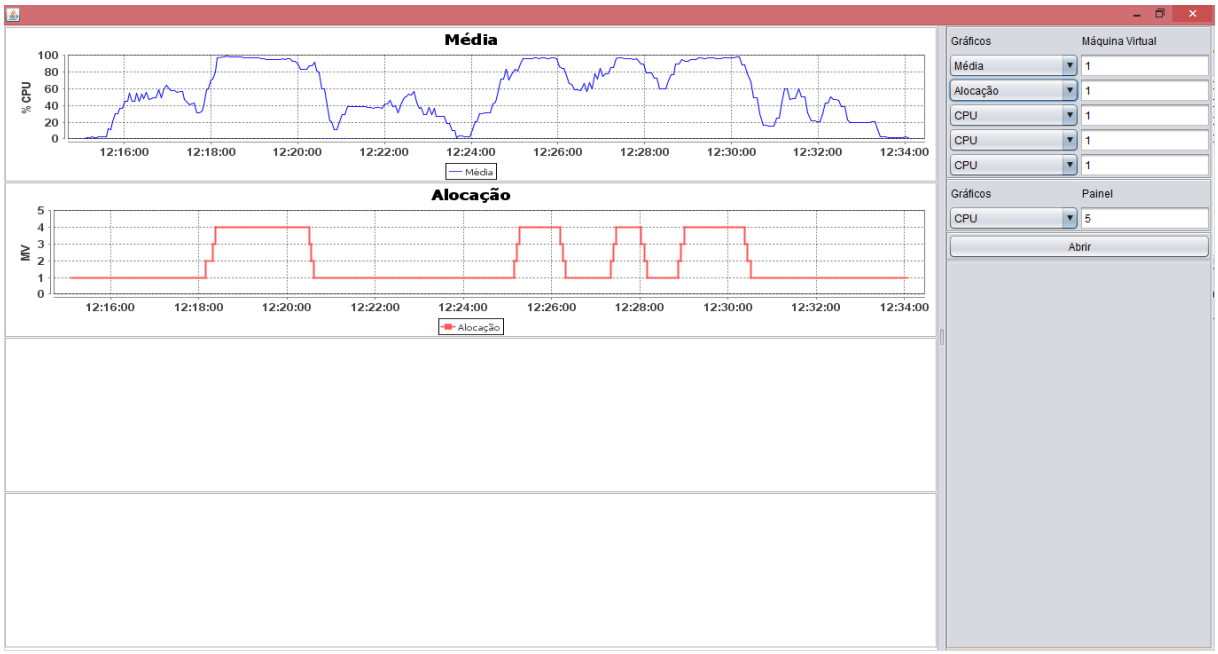


Figura B.6: Tela com a média de utilização de CPU de todas as máquinas virtuais e a alocação das máquinas virtuais de um experimento

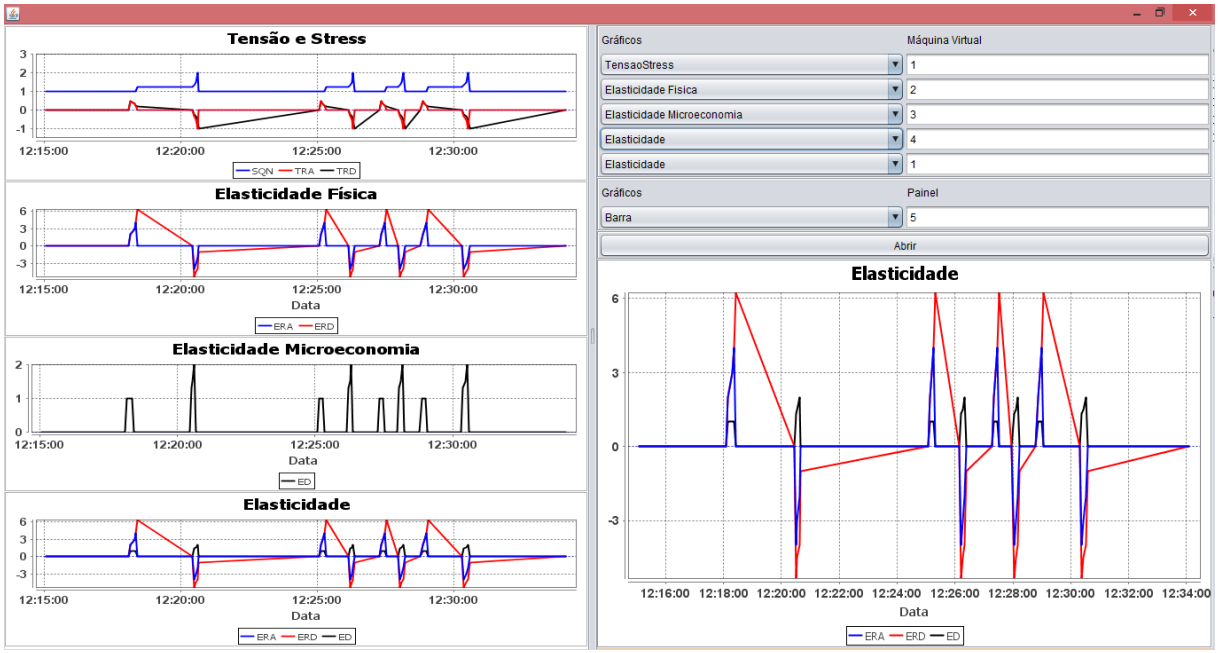


Figura B.7: Tela com as métricas de elasticidade de um experimento

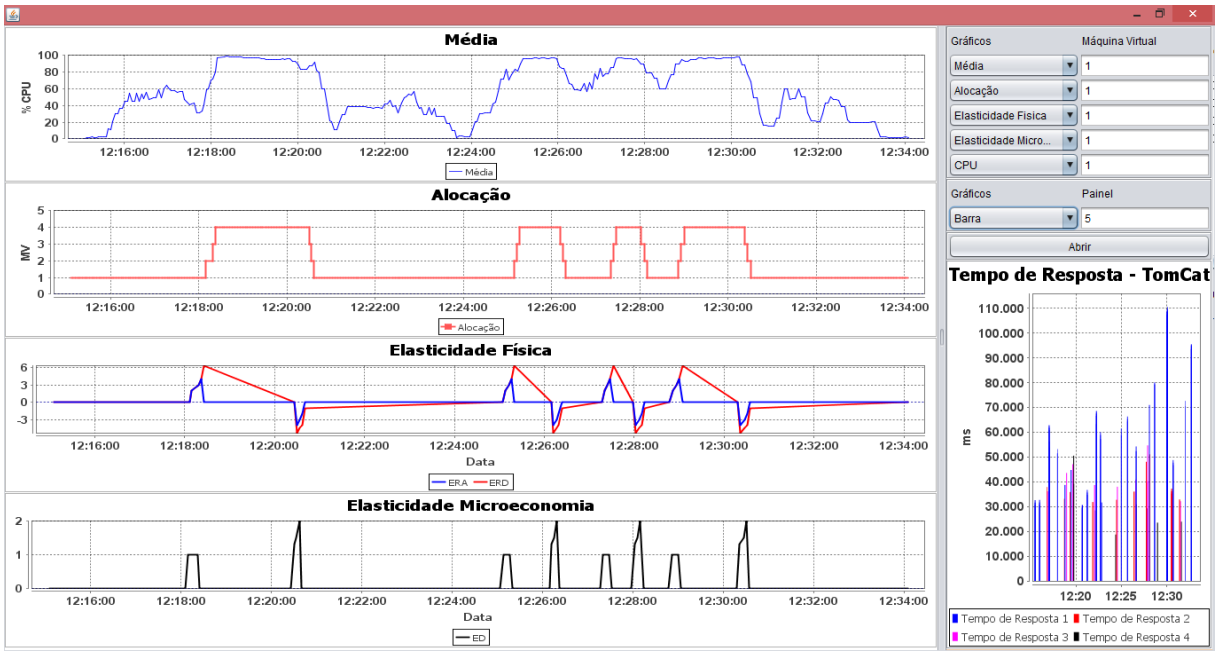


Figura B.8: Tela típica para uma análise de desempenho com a média de utilização de CPU de todas as máquinas virtuais, a alocação conforme os limiares definidos na infraestrutura, as métricas de elasticidade e o tempo de resposta das requisições de um experimento