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

Planejamento de Capacidade Dirigido a Negócios para Aplicações SaaS de Comércio Eletrônico

David Candeia Medeiros Maia

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

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Metodologia e Técnicas da Computação

Raquel Vigolvino Lopes
(Orientadora)

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

M28p Maia, David Candeia Medeiros.
Planejamento de Capacidade Dirigido a Negócios para Aplicações SaaS de Comércio Eletrônico / David Candeia Medeiros Maia. – Campina Grande, 2012.
97 f.: il. color.

Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.
Orientadora: Profa. Dra. Raquel Vigolvino Lopes.
Referências.

1. Computação na Nuvem. 2. Planejamento de Capacidade. 3. *Software* como Serviço. I. Título.

CDU 004.7 (043)

"PLANEJAMENTO DE CAPACIDADE DIRIGIDO A NEGÓCIOS PARA APLICAÇÕES SAAS DE COMÉRCIO ELETRÔNICO"

DAVID CANDEIA MEDEIROS MAIA

DISSERTAÇÃO APROVADA EM 12/07/2012

RAQUEL VIGOLVINO LOPES, D.Sc
Orientador(a)

ANDREY ELÍSIO MONTEIRO BRITO, Dr.
Examinador(a)

ALEXANDRE NOBREGA DUARTE, D.Sc
Examinador(a)

CAMPINA GRANDE - PB

Resumo

Na última década acompanhou-se o crescimento de um novo mercado na computação denominado de Computação na Nuvem. Este mercado tem por base a oferta de três serviços principais: (i) recursos computacionais virtuais sob demanda, por exemplo, processamento e armazenamento – serviço este denominado de *Infraestrutura como Serviço* (do inglês, *Infrastructure as a Service – IaaS*); (ii) plataformas que facilitam o desenvolvimento de novas aplicações – serviço este denominado de *Plataforma como Serviço* (do inglês, *Platform as a Service – PaaS*); (iii) aplicações prontas para uso – serviço este denominado de *Software como Serviço* (do inglês, *Software as a Service – SaaS*). Empresas que ofertam software como serviço (provedores de SaaS) podem planejar e montar sua infraestrutura de Tecnologia da Informação (TI) fazendo uso de recursos computacionais obtidos junto às empresas que ofertam recursos computacionais (provedores de IaaS). O modelo de negócio utilizado por provedores de IaaS prevê que recursos computacionais possam ser reservados com antecedência mediante pagamento de uma taxa de reserva. A reserva de um recurso garante aos contratantes deste serviço que lhes serão cobradas tarifas reduzidas quando do uso dos recursos computacionais. Realizar um planejamento de capacidade de longo prazo para estimar a quantidade de recursos necessários para execução de uma carga de trabalho e, assim, estabelecer bons contratos de reserva junto aos provedores de IaaS é uma das etapas do gerenciamento de infraestruturas de TI a ser realizada pelo provedor de SaaS. Esta dissertação apresenta duas heurísticas para realização do planejamento de capacidade: heurística UT e RF. Tais heurísticas consideram o modelo de utilidade desenvolvido nesta dissertação para estimar a quantidade de recursos a serem reservados. A avaliação realizada utiliza um modelo de simulação que considera a carga de trabalho de uma aplicação de comércio eletrônico. Em todos os cenários avaliados UT e RF apresentaram valores positivos de utilidade, com RF apresentando uma maior margem de utilidade em relação a uma estratégia base que não realiza reserva de recursos. Por fim, foi descoberto que, de acordo com a qualidade da predição da carga de trabalho, pode-se realizar uma escolha mais criteriosa dentre as heurísticas propostas.

Abstract

In the last decade we have seen the advent and growth of the Cloud Computing market. This market is based on offering three main services: (i) virtual on-demand computing resources such as storage and processing – this service is called Infrastructure as a Service (IaaS); (ii) platforms that facilitate the development of new applications – this service is called Platform as a Service (PaaS); (iii) software and data hosted on the cloud – this service is called Software as a Service (SaaS). SaaS providers can plan and build their Information Technology (IT) infrastructure making use of computing resources offered by IaaS providers. The business model used by IaaS providers states that computing resources can be reserved in advance by paying a reservation fee. A reserved resource has the advantage that its usage fee is lower than the usage fee of non-reserved resources. Capacity planning is one step in the IT infrastructure management performed by a SaaS provider that helps to estimate the amount of resources required to execute a future workload and thus establish good reservation contracts with IaaS providers. This dissertation presents two capacity planning heuristics: (i) heuristic based on resources utilization rates – UT; (ii) heuristic based on queue networks – RF. These heuristics consider an utility model based on the SaaS provider profit. The evaluation of such heuristics uses a simulation model that considers an e-commerce workload. In all scenarios evaluated UT and RF presented positive utility values and RF presented the best utility gains compared to a strategy that does not reserve resources. Finally, we discovered that according to the quality of the workload prediction that is used by a SaaS provider a better choice can be done among proposed heuristics.

A vida é uma peça de teatro que não permite ensaios. Por isso, cante, chore, dance, ria e viva intensamente, antes que a cortina se feche e a peça termine sem aplausos.

Charles Chaplin

Agradecimentos

Agradeço primeiramente a Deus. Vários foram os momentos vividos durante estes anos e foi por acreditar nele que encontrei forças para superar os obstáculos e procurar sempre aprender com as mais diversas situações.

Agradeço enormemente aos meus pais, que investiram e acreditaram em mim desde o meu nascimento e que me mostraram que o conhecimento, a educação, o cultivo aos valores pessoais e a busca por ideais movem a vida de uma pessoa. Acho que dificilmente conseguirei retribuir todo o amor e a força que me deram durante todos estes anos e por isso tenho orgulho de dizer que sou filho deles.

Agradeço do fundo do meu coração a minha noiva, Érika. Inúmeros foram os momentos nos quais busquei força em nossas conversas, em nossas brincadeiras, nos sonhos que compartilhamos e no amor que sentimos. Sem dúvida alguma ela é uma peça fundamental para guiar todas as ações que busco tomar em minha vida.

Agradeço a minha orientadora, Raquel Vigolvino Lopes, por me deixar muito à vontade para pesquisar em um tema que fosse do meu interesse, por todos os ensinamentos, por todas as conversas e pela amizade que foi cultivada ao longo destes anos.

Agradeço muito ao meu irmão de sangue, Lucas, aos meus primos (Amanda, Bruno, Herllen, Talytha e Yohanna), aos meus tios (Herbet e Luísa), a minha "sobrinha" (Helena) e aos meus amigos que cultivei ao longo da vida, que são meus irmãos por opção: Anderson Trindade, Ana Karolina, Felipe Nóbrega, Guilherme Nóbrega, Mary Alves, Renan Ribeiro, Rennan Melo, Rodolpho Dias, Paulo José, Thiago Tavares, Vanessa Santos. Gostaria de falar o quão importante foram as mais simples atitudes de todos para comigo, porém isto precisaria de uma quantidade enorme de páginas para retratar uma pequena parte deste carinho.

Gostaria de agradecer a todos os amigos que fazem o Laboratório de Sistemas Distribuídos (LSD). Alguns já terminaram seus trabalhos e partiram, mas possuem um local especial em meu coração. Agradeço particularmente aos que estiveram mais próximos nos últimos dois anos: Marcus Williams, Lília Sampaio, Ricardo Araújo e Paulo Ditarso, com os quais compartilhei a sala Carvalheira, muitas gargalhadas e muitas conversas; Abmar Granjeiro,

Antônio Flávio, Carla Araújo, Edigley Fraga, Fábio Jorge, Francisco Brasileiro, Geraldo Sarmento, Giovanni Farias, João Arthur, Lesandro Ponciano, Lívia Sampaio, Patrick Jourdan, Pryscilla Dora e Thiago Emmanuel, que compartilharam várias conversas e contribuíram de diferentes formas para a conclusão do trabalho.

Agradecer a todos que ajudaram a construir esta dissertação não é tarefa fácil. O maior perigo que se coloca para o agradecimento seletivo é esquecer de mencionar algúem. Então, a meus amigos que não mencionei acima gostaria de expressar minha profunda gratidão.

Todos vocês são co-autores deste trabalho!

Por fim, agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro concedido para a realização desta pesquisa.

Conteúdo

Lista de Símbolos

Tabela de Siglas	
Notação	**Descrição**
BDIM	*Gerência de TI Orientada a Negócios*
CRM	*Gestão de relacionamento com o Cliente, do inglês Customer Relationship Management*
DPS	*Sistema de Provisão Dinâmica de Recursos*
EC2	*Nuvem Elástica de Computação da Amazon, do inglês Amazon Elastic Compute Cloud*
IaaS	*Infraestrutura como Serviço, do inglês Infrastructure as a Service*
PaaS	*Plataforma como Serviço, do inglês Platform as a Service*
PDCA	*Planejar, Executar, Verificar e Agir, do inglês Plan, Do, Check and Act*
PS	*Disciplina de Compartilhamento de Unidade de Processamento, do inglês Processor Sharing*
RF	*Heurística de Planejamento de Capacidade Baseada em Rede de Filas*
SaaS	*Software como Serviço, do inglês Software as a Service*
SLA	*Acordo de Nível de Serviço, do inglês Service Level Agreement*
TI	*Tecnologia da Informação*
UT	*Heurística de Planejamento de Capacidade Baseada em Taxa de Utilização de Recurso*

Tabela de Símbolos Matemáticos

Notação	Descrição
A	*Aplicação ofertada pelo provedor de SaaS*
A^{MIN}	*Disponibilidade mínimia exigida para a aplicação* A
a_s^n	*Contador de consumo de grãos* θ *para um recurso de tipo* s *em um período* n
a_s	*Quantidade do recurso de tipo* s *reservada junto ao provedor de IaaS*
$\alpha(D)$	*Custo do provedor de SaaS em um intervalo* D
c_s	*Custo de uso de recurso do tipo* s *junto ao provedor de IaaS*
C_o	*Conjunto de tuplas que definem tipos e custos de uso para um recurso* ω
$ca(n)$	*Custo do provedor de SaaS gerado pelo uso de recursos junto ao provedor de IaaS em um período* n
$cv(n)$	*Custo do provedor de SaaS gerado pela reserva de recursos junto ao provedor de IaaS em um período* n
$c(n)$	*Custo total do provedor de SaaS em um período* n
$d_{j,m}$	*Demanda de uma requisição em um recurso* $r_{j,m}$
D	*Intervalo de avaliação da aplicação*
E_j	*Conjunto de taxas adicionais a serem pagas pelo uso de recursos de um plano* p_j *além dos limiares definidos no conjunto* L_j
$e_{j,m}$	*Taxa adicional a ser paga por unidade do recurso* $r_{j,m}$*, segundo o plano* p_j*, consumida além do limiar* $l_{j,m}$
f_s	*Taxa de reserva de um recurso de tipo* s *junto ao provedor de IaaS*
I_j^b	*Taxa de configuração do plano* p_j *contratado pelo cliente* u_k
$i_k^b(n)$	*Função que determina a taxa de configuração a ser paga pelo cliente* u_k *em um período* n
I_j	*Taxa de uso paga pelo cliente* u_k*, de plano* p_j*, ao provedor de SaaS a cada período* τ

continua na página seguinte

Tabela 0.1 – continuação da página anterior

Notação	Descrição
$i_k^{\tau}(n)$	*Função que determina a taxa de uso a ser paga pelo cliente u_k em um período n*
$i_k(n)$	*Receita gerada por um cliente u_k para o provedor de SaaS em um período n*
$i(n)$	*Receita gerada por todos os clientes de SaaS para o provedor de SaaS em um período n*
$\iota(D)$	*Receita obtida pelo provedor de SaaS em um intervalo D de avaliação*
γ	*Conjunto de contratos de reserva firmados entre o provedor de SaaS e o provedor de IaaS*
L_j	*Conjunto de limiares de uso dos recursos tarifáveis estabelecidos pelo plano p_j*
$l_{j,m}$	*Limiar de uso do recurso $r_{j,m}$ estabelecido pelo plano p_j*
M_j	*Multa a ser paga pelo provedor de SaaS, em caso de violação do SLA_j, para um cliente de plano p_j*
n_k^b	*Período no qual o cliente u_k SaaS assinou seu contrato junto ao provedor de SaaS*
n_k^e	*Último período no qual o cliente u_k realizou um pagamento para utilizar o sistema*
n^b	*Período de início da avaliação da aplicação*
n^e	*Período de término da avaliação da aplicação*
n_s^b	*Início da vigência de contrato de reserva de recursos*
n_s^e	*Término da vigência de contrato de reserva de recursos*
O	*Conjunto de classes de recursos ofertados pelo provedor de IaaS*
o	*Classe de recurso ofertado pelo provedor de IaaS*
P	*Conjunto de planos ofertados pelo provedor de SaaS*
p_j	*Plano contratado pelo cliente u_k junto ao provedor de SaaS*

continua na página seguinte

Tabela 0.1 – continuação da página anterior

Notação	Descrição
$p(n)$	*Soma de todas as penalidades pagas pelo provedor de SaaS no período* n
Q_k^n	*Conjunto de requisições submetidas pelo cliente* u_k *à aplicação durante um período* n
R_j	*Conjunto de recursos tarifáveis ofertados pelo provedor de SaaS no plano* p_j
$r_{j,m}$	*Recurso tarifável ofertado pelo provedor de SaaS e pertencente ao conjunto* R_j
s	*Tipo de recurso da classe* o
SLA_j	*Acordo de nível de serviço para um plano* p_j
τ	*Intervalo de tempo durante o qual um cliente SaaS pode fazer uso da aplicação mediante um pagamento*
T^{MAX}	*Tempo de resposta máximo aceitável para requisições da aplicação* A
t	*Instante de chegada da requisição aos recursos do provedor de SaaS*
U	*Conjunto de clientes de SaaS*
u_k	*Cliente de SaaS*
$\upsilon(D)$	*Utilidade (lucro) do provedor de SaaS em um intervalo* D
V	*Contrato de reserva de recursos*
θ	*Grão de uso de um recurso tarifável no provedor de IaaS*
Δ	*Quantum durante o qual uma requisição pode fazer uso de uma CPU*

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Capítulo 1

Introdução

Neste capítulo, apresentamos o contexto da pesquisa descrita nesta dissertação discorrendo sobre aspectos relacionados à realização da atividade de planejamento de capacidade considerando o ambiente de Computação na Nuvem. Em seguida, apresentamos o escopo da pesquisa realizada, bem como a especificação do problema. Por fim, o capítulo é encerrado com uma descrição dos demais capítulos que compõem esta dissertação.

1.1 Contexto

O mercado de Computação na Nuvem tem experimentado um rápido crescimento[1] e atualmente muitas empresas fornecem diversos serviços: (i) recursos virtuais sob demanda, por exemplo, poder computacional e armazenamento; (ii) plataformas que facilitam o desenvolvimento de novas aplicações; (iii) aplicações prontas para uso. Esse novo mercado apresenta um modelo de negócio no qual receitas são obtidas de acordo com o uso dos recursos (do inglês, *pay-per-use*), de acordo com quantidades de transações realizadas (do inglês, *pay-per-transaction*) ou ainda de acordo com inscrições (do inglês, *pay-per-subscription*) [Sääksjärvi, Lassila e Nordström 2005] realizadas pelos clientes dos serviços. Aqueles que promovem este mercado afirmam que alguns dos principais benefícios deste modelo de serviço estão relacionados com o fato dos modelos de cobrança utilizados levarem a reduções nos custos de execução das infraestruturas e/ou serviços de Tecnologia da Informação (TI),

[1]Gartner Says Worldwide Cloud Services Market to Surpass $68 Billion in 2010: http://www.gartner.com/it/page.jsp?id=1389313

bem como com a simplificação da gerência de infraestruturas de TI [Tak, Urgaonkar e Sivasubramaniam 2011]. Além disso, outro grande benefício apresentado é a flexibilidade de se aumentar e diminuir a capacidade dos serviços adquiridos. Como a busca para reduzir o custo total de propriedade de componentes de TI deve continuar, pode-se afirmar que se está vivenciando uma mudança importante na forma como as infraestruturas são montadas, configuradas e gerenciadas.

Os serviços oferecidos no mercado de Computação na Nuvem são, tipicamente, classificados em diferentes níveis, porém não há um consenso sobre esta classificação. A classificação proposta em [Vaquero et al. 2008] é o estado da prática e divide os serviços em:

- *Infraestrutura como Serviço* (do inglês, *Infrastructure as a Service – IaaS)* [Vogels 2008]: este tipo de serviço oferece recursos computacionais como processamento e armazenamento de dados para que as empresas possam montar sua infraestrutura de TI. A base para a oferta desses recursos é a utilização das tecnologias de virtualização. Os provedores deste tipo de serviço oferecem a possibilidade de reserva de recursos para um longo prazo (*mercado de reservas*), ou, ainda, a aquisição de recursos no curto prazo (*mercado sob demanda*). Os exemplos de provedores de IaaS mais conhecidos são o **Amazon** (através do serviço **Amazon Elastic Compute Cloud**[2]), **RackSpace**[3], **GoGrid**[4] e **Cloud Sigma**[5];

- *Plataforma como Serviço* (do inglês, *Platform as a Service – PaaS)* [Lawton 2008]: este tipo de serviço oferece ferramentas para auxiliar desde o desenvolvimento de aplicações até a manutenção do ciclo de vida delas. Exemplos de soluções na modalidade PaaS são **Google App Engine**[6] e **Microsoft Azure**[7];

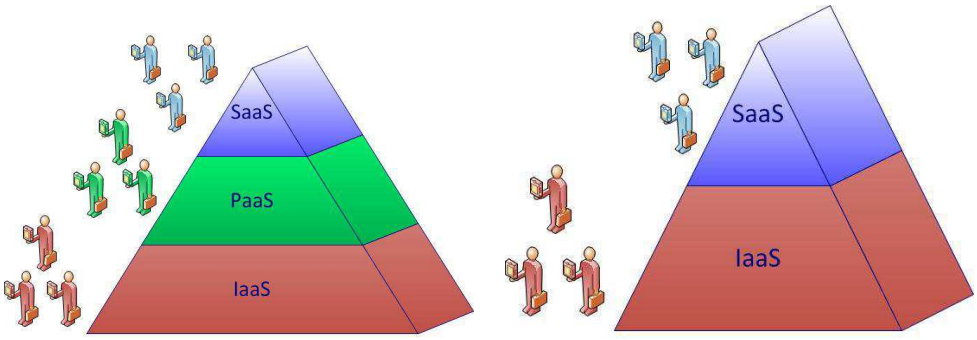
- *Software como Serviço* (do inglês, *Software as a Service – SaaS)* [Jacobs 2005]: este tipo de serviço oferece aplicações prontas para um cliente. Alguns exemplos conhecidos de aplicações que são ofertadas no modelo SaaS são aplicações de *Customer Re-*

[2]http://aws.amazon.com/ec2/
[3]http://www.rackspace.com
[4]http://www.gogrid.com/
[5]http://www.cloudsigma.com
[6]http://code.google.com/appengine/
[7]http://www.microsoft.com/windowsazure/

lationship Management (CRM)[8] [9], editores de texto como o **Google Docs**[10] e soluções de comércio eletrônico desenvolvidas pelas empresas **BigCommerce**[11] e **Volusion**[12].

A oferta de aplicações no modelo SaaS vem atraindo provedores tanto no mundo acadêmico como no mundo comercial. No mundo acadêmico, demandas como processamento de texto e tabelas, envio de mensagens de correio eletrônico, processamento e análise de dados de simulação e transferência de arquivos vêm sendo atendidas, respectivamente, por soluções como **Google Docs**, **Gmail**, **Hotmail**, **R-php**[13] e **Globus Online**[14]. No mundo comercial soluções vêm sendo desenvolvidas por diversas empresas como a **SalesForce**[15] e **SugarCRM**[16], que oferecem soluções de CRM, e por empresas como a **BigCommnerce**[11], que oferece uma solução de comércio eletrônico.

Para oferecer software como um serviço com base nos demais serviços existentes no mercado de Computação na Nuvem, os provedores de SaaS podem organizar sua infraestrutura segundo dois modelos básicos (vide Figura 1.1):



(a) Modelo SaaS, PaaS, IaaS (b) Modelo SaaS, IaaS

Figura 1.1: Modelos de organização de infraestrutura para um provedor de SaaS que deseja utilizar recursos adquiridos no mercado de Computação na Nuvem

[8] http://www.salesforce.com/br/crm/products.jsp
[9] http://crm.dynamics.com/
[10] http://docs.google.com
[11] http://www.bigcommerce.com/
[12] http://www.volusion.com/
[13] http://dssm.unipa.it/R-php/#
[14] http://www.globusonline.org
[15] http://www.salesforce.com/
[16] http://www.sugarcrm.com/crm/

1. *Modelo SaaS + PaaS + IaaS*, Figura 1.1(a) [Buyya, Broberg e Goscinski 2011]: neste modelo uma aplicação SaaS é desenvolvida com base na utilização de serviços ofertados por um provedor de PaaS, delegando para este a gerência dos recursos computacionais;

2. *Modelo SaaS + IaaS*, Figura 1.1(b) [Namjoshi e Gupte 2009]: neste modelo um provedor de SaaS desenvolve sua aplicação utilizando diretamente provedores de IaaS. Neste modelo, o provedor de SaaS é responsável por gerenciar o conjunto de recursos necessários para sua solução.

Especialmente quando o *Modelo SaaS + IaaS* é usado, a forma como o provedor de SaaS adquire recursos de um ou mais provedores de IaaS e a forma como estes recursos são utilizados impactam diretamente no custo de manter o serviço e na qualidade do serviço oferecido, influenciando, por consequência, o lucro do provedor de SaaS. A gerência de quais recursos devem ser adquiridos no provedor de IaaS para executar o serviço é uma decisão do provedor de SaaS que envolve diferentes mercados para aquisição destes recursos e diferentes tipos de decisão.

Os recursos a serem requisitados em um provedor de IaaS podem ser adquiridos em dois mercados: (i) *mercado de reservas*; (ii) *mercado sob demanda*. No *mercado de reservas*, o cliente do IaaS paga inicialmente uma taxa de reserva com certa antecedência (e.g., um ano) para ter os recursos a sua disposição por um longo período futuro (e.g., da ordem de anos). Esta taxa de reserva é proporcional à quantidade de recursos reservados. Os recursos reservados podem ser requisitados pelos clientes quando eles desejarem e, nesta ocasião, uma outra taxa com valor tipicamente muito pequeno, a taxa de uso, é cobrada pelo uso efetivo dos recursos previamente reservados. Uma vez que o recurso foi reservado, o provedor de IaaS não pode negar ao cliente o uso do recurso, caso contrário estará sujeito ao pagamento de grandes penalidades/multas a seus clientes. A **Amazon** é a principal empresa praticante deste mercado através do serviço **Amazon EC2**[17], porém outras empresas, como a **GoGrid**, praticam este *mercado de reservas* com algumas diferenças em relação ao mercado da **Amazon**. Neste serviço, a opção de reserva, se bem usada[18], permite a economia de até 58% em

[17]http://aws.amazon.com/ec2/purchasing-options/

[18]Recursos são bem usados no mercado de reservas quando a taxa efetiva de utilização dos mesmos ul-

relação à utilização de recursos do *mercado sob demanda*[19]. Atualmente as empresas que praticam este *mercado de reservas* permitem apenas a reserva de recursos de processamento.

No *mercado sob demanda*, o cliente do provedor de IaaS solicita instâncias imediatamente antes de usá-las, sem ter a necessidade de realizar uma reserva prévia. Neste mercado a taxa paga pelos clientes em relação ao uso efetivo dos recursos é superior à taxa de uso paga no *mercado de reservas*. O provedor de IaaS tipicamente limita a quantidade de instâncias simultâneas que um dado cliente pode solicitar e mesmo que o número de instâncias solicitadas esteja dentro deste limite, é aceitável que o provedor de IaaS negue recursos ao cliente em um momento de contenção. Isto significa que nem sempre é possível obter as instâncias de recursos desejadas no *mercado sob demanda* no momento desejado.

O *mercado de reservas* oferece benefícios financeiros, bem como benefícios relacionados à qualidade de serviço, contanto que as reservas sejam realizadas para um longo período futuro, tipicamente um ou três anos. À medida que recursos são utilizados para execução da aplicação novas reservas podem ser realizadas para ampliar o total de recursos disponíveis, neste caso múltiplas reservas estariam firmadas entre o provedor de SaaS e o provedor de IaaS.

Devido à existência destes dois mercados, a execução de uma aplicação implica em dois tipos de decisão: (i) o estabelecimento de contratos de reservas para longos períodos futuros; (ii) a requisição de recursos para atendimento da demanda da aplicação. A atividade de planejamento e estabelecimento de contratos de reservas para longos prazos futuros é denominada de planejamento de capacidade de longo prazo. A atividade de avaliação e requisição de recursos para execução da aplicação em um curto prazo é denominada de planejamento de capacidade de curto prazo, escalonamento ou provisão dinâmica de recursos. Por convenção esta dissertação referencia o planejamento de capacidade de longo prazo como planejamento de capacidade e o planejamento de capacidade de curto prazo como provisão dinâmica de recursos.

Ao optar por realizar reservas de longo prazo junto ao provedor de IaaS o provedor de SaaS deve avaliar a viabilidade financeira de se realizar tais reservas e decidir a quantidade

trapassa um certo patamar, que depende dos preços de reserva e uso destes recursos e dos preços de uso no mercado sob demanda. Atualmente, na Amazon, só vale a pena reservar um recurso se ele vier a ser usado durante pelo menos 20% do tempo reservado.

[19]http://aws.amazon.com/ec2/reserved-instances/

de recursos a serem reservados. É importante destacar que o planejamento de capacidade lida com uma incerteza inerente uma vez que se está realizando o planejamento de um longo prazo futuro (e.g., um ano ou períodos maiores) para atender uma carga de trabalho estimada para o futuro. Uma das formas de lidar com esta incerteza é considerar, durante a fase de planejamento, que se a quantidade de recursos reservados não for suficiente para atender a demanda da aplicação em um certo instante, recursos adicionais podem ser adquiridos no *mercado sob demanda*. Ou seja, um planejamento de capacidade pode considerar o uso dos dois mercados acima de forma complementar para a execução da aplicação.

No momento em que a aplicação vai ser efetivamente executada, deve existir um sistema de provisão dinâmica de recursos [Ranjan et al. 2002] [Urgaonkar et al. 2008] [Lee et al. 2010] [Kwok e Mohindra 2008] [Zhu et al. 2011] [Bi et al. 2010] [Chi, Qian e Lu 2011] que decide sobre a quantidade de recursos que deve ser requisitada junto ao provedor de IaaS e, posteriormente, cedida à aplicação em um curto intervalo de tempo para que a demanda atual da aplicação possa ser atendida. Pode-se chamar este sistema de escalonador, ou planejador de capacidade de curto prazo.

O planejamento de capacidade que deve ser realizado pelos provedores de SaaS envolve as seguintes questões: é interessante *para o negócio* (visão do provedor de SaaS) reservar recursos no *mercado de reservas*? Quantos recursos e de que capacidade devem ser reservados? O cenário de Computação na Nuvem requer que o planejamento de capacidade seja realizado com base em métricas de negócio (e.g., lucro obtido pelo provedor de SaaS) e não baseado apenas em métricas técnicas, como tempo de resposta ou taxa de rejeição de serviço. O planejamento de capacidade baseado em métricas de negócio deve conhecer a ponte que faz a ligação entre as decisões técnicas e o impacto destas para o negócio em termos de receitas e perdas.

Assim como nos ambientes Web tradicionais, a atividade de planejamento de capacidade precisa lidar com a incerteza inerente às variações futuras nas cargas de trabalho das aplicações. Porém, o planejamento de capacidade específico para aplicações SaaS que serão executadas sobre recursos de provedores de IaaS possui diferenças em relação ao planejamento realizado para as aplicações Web anteriores que lidavam com as infraestruturas tradicionais. Primeiramente, o ambiente de Computação na Nuvem tem como forte argumento a redução de custos para seus clientes dado que só se paga pelo montante usado dos recursos. Como

cliente do IaaS, o provedor de SaaS deve estar ciente das opções de aquisição de recursos e identificar a compra que trará o melhor benefício para seu negócio. Além de considerar as questões econômicas diretamente associadas à reserva dos recursos, o provedor de SaaS também deve levar em consideração, no momento do planejamento, o risco de não ter, no futuro, seus pedidos por recursos no *mercado sob demanda* aceitos. Em um ambiente tradicional, em que a atividade de planejamento resulta em compra de recursos computacionais, este risco não existia. Além disso, o ambiente tradicional exigia que as infraestruturas fossem superprovidas para que os picos da carga de trabalho fossem atendidos pelos provedores de serviço. No ambiente de Computação na Nuvem a infraestrutura não precisa ser superprovida uma vez que o *mercado sob demanda* permite que recursos sejam requisitados no momento em que forem necessários.

Um outro aspecto importante que deve ser levado em consideração é que os provedores de SaaS seguem um modelo de negócio diferente daquele modelo de aplicações Web tradicionais. Tradicionalmente, uma aplicação era de propriedade da organização que a implantava. Ao implantar e gerenciar a aplicação a organização passava a obter receitas, de acordo com a finalidade da aplicação, a partir de seu uso por parte de usuários finais. Esta receita ditava o total do investimento a ser realizado em infraestrutura para hospedar a aplicação. Uma aplicação que se enquadra em tais características e que apresenta muitos estudos na literatura é a aplicação de comércio de eletrônico. Nesta aplicação em particular, uma empresa poderia vender seus produtos na Internet através de uma solução própria de comércio eletrônico. O lucro que esta aplicação trazia para a empresa era dado pelas vendas realizadas via Internet.

Uma aplicação SaaS é uma aplicação que apresenta *multilocação* e envolve mais atores: (i) o provedor da aplicação SaaS, referenciado nesta dissertação como provedor de SaaS, (ii) os clientes da aplicação SaaS, referenciados como clientes de SaaS e (iii) o provedor de IaaS. Os clientes do provedor de SaaS pagam, periodicamente, a este uma taxa de acordo com um plano escolhido. O lucro do provedor de SaaS é proveniente desta taxa. Isto significa que, mesmo que outros planejadores de capacidade tenham sido pensados para aplicações Web tradicionais [Sauve et al. 2006] [Daniel e Virgilio 2000] e, ainda que eles tenham tentado otimizar métricas de negócio para a tomada da decisão de planejamento de capacidade, o modelo de receitas e custos era diferente do modelo atual que envolve provedores e clientes de Computação na Nuvem.

1.2 Escopo

A partir dos modelos apresentados na Figura 1.1, que podem ser utilizados por um provedor de SaaS para composição de sua infraestrutura (*Modelo SaaS + PaaS + IaaS* e *Modelo SaaS + IaaS*), esta dissertação irá considerar o modelo no qual provedores de SaaS montam sua infraestrutura a partir de recursos computacionais adquiridos junto a um provedor de IaaS [Namjoshi e Gupte 2009]. Tal escolha se baseia no fato deste modelo oferecer maior flexibilidade ao provedor de SaaS sobre as tecnologias utilizadas no desenvolvimento da aplicação, bem como oferecer flexibilidade na composição da infraestrutura fazendo uso de recursos de diferentes tipos. Existem indícios de que um grande consumo de recursos de diferentes provedores para ofertar uma aplicação não é interessante em virtude dos custos gerados [Tak, Urgaonkar e Sivasubramaniam 2011]. Outro ponto importante é que a escolha por um provedor de IaaS base para aquisição de recursos pode ser feita a partir de uma avaliação prévia [Li et al. 2010]. Contudo, devido aos riscos de negação de serviço por parte dos provedores de IaaS pode-se fazer uso de múltiplos provedores de IaaS de modo que, quando um determinado provedor venha a negar serviço possa-se requisitar os recursos necessários em outro provedor de IaaS (vide ferramenta TapInSystems[20]). Dados estes pontos, considera-se que o provedor de SaaS utiliza unicamente um provedor de IaaS, selecionado previamente, para obtenção dos recursos necessários.

Um provedor de IaaS tipicamente oferece um portfólio de diferentes recursos (e.g., recursos de processamento e recursos de transferência de dados) que podem ser adquiridos por um provedor de SaaS. Além disto, cada recurso pode ser ofertado em diferentes modalidades. Por exemplo, um recurso de processamento pode ser ofertado sob algumas opções: um recurso de processamento com dois núcleos, e cada núcleo com um *clock* de 1 GHZ; um recurso de processamento com um núcleo e um *clock* de 2 GHZ; etc. Ao montar sua infraestrutura, um provedor de SaaS pode considerar essa variedade de recursos e analisar qual a melhor composição de recursos que atende a sua demanda e que traz melhores benefícios aos negócios. Tipicamente os trabalhos que lidam com planejamento de capacidade consideram recursos homogêneos [Bichler, Setzer e Speitkamp 2006] [Sauve et al. 2006] [Li, Casale e Ellahi 2010] dado que a heterogeneidade de recursos torna ainda mais complexo o problema

[20]http://www.tapinsystems.com/

de se planejar uma infraestrutura de TI. Esta dissertação busca realizar uma avaliação de heterogeneidade de recursos ao permitir que as estratégias de planejamento de capacidade propostas avaliem os benefícios obtidos para o negócio e, assim, possam em um mesmo plano de reservas realizar a reserva de diferentes tipos de recursos. É interessante destacar que nenhuma das heurísticas propostas busca realizar a conversão entre recursos de tipos diferentes (e.g., um recurso de tipo A utilizado por pouco tempo dentro do período de planejamento corresponde a um recurso de tipo B utilizado por mais tempo dentro do período de planejamento) de modo a aperfeiçoar um plano de reservas. Ainda em relação ao provedor de IaaS, é importante destacar que o risco de que recursos previamente reservados ao serem requisitados para uso podem não ser obtidos não foi modelado e considerado nesta dissertação.

A aplicação SaaS considerada no estudo de caso deste trabalho é uma aplicação de comércio eletrônico. Aplicações de comércio eletrônico são muito estudadas na literatura e vários trabalhos [Arlitt, Krishnamurthy e Rolia 2001] [Menascé et al. 2003] [Menascé et al. 2000] [Zhang, Cherkasova e Smirni 2007] promovem a caracterização da carga de trabalho deste tipo de aplicação. Por conta desta ampla caracterização considerar este tipo de aplicação permite, ainda, o uso de geradores de carga de trabalho bem consolidados [Kant, Tewari e Iyer 2001]. A caracterização de um modelo de negócio no molde de SaaS praticado pela empresa **BigCommerce**[21] fornece dados para um modelo de negócio deste tipo de aplicação. Além disso, o planejamento de capacidade estudado para este tipo de aplicação [Sauve et al. 2006] [Daniel e Virgilio 2000] utilizando as infraestruturas tradicionais considerava um modelo de receitas e custos diferente do modelo atual que envolve provedores e clientes de Computação na Nuvem.

1.3 Definição do Problema

Fazendo uso do modelo *Goal, Question, Metric* [Caldiera e Rombach 1996] o problema investigado nesta dissertação pode ser definido segundo a Tabela 1.1:

[21] http://www.bigcommerce.com/

Analisar o planejamento de capacidade
com a intenção de propor e avaliar técnica(s) de planejamento
com respeito a seu retorno para o negócio
do ponto de vista do gerente da infraestrutura
no contexto de execução de aplicações SaaS de comércio eletrônico

Tabela 1.1: Definição de problema segundo modelo *Goal, Question, Metric*

1.4 Objetivos

1.4.1 Objetivo Geral

O objetivo deste trabalho é avaliar o planejamento de capacidade de uma infraestrutura de TI para ofertar uma aplicação SaaS de comércio eletrônico fazendo uso de recursos obtidos junto a um provedor de IaaS. Heurísticas de planejamento de capacidade foram propostas considerando um modelo de utilidade (aqui representada pelo lucro do provedor de SaaS) que captura aspectos de negócio da aplicação em foco.

1.4.2 Objetivos Específicos

De modo a atingir o objetivo geral descrito acima foram definidos os seguintes objetivos específicos:

1. Buscar a caracterização do comportamento da carga de trabalho de uma aplicação de comércio eletrônico. Neste sentido, foi realizada uma análise com base em estudos que caracterizam a carga de trabalho de uma aplicação de comércio eletrônico e com base em modelos de negócio atualmente em vigência para aplicações SaaS de comércio eletrônico;

2. Elaborar um modelo de utilidade para aplicações SaaS. O modelo elaborado captura aspectos de negócio (i.e, custos, receitas e lucro) envolvidos na oferta de uma aplicação por parte de um provedor de SaaS;

3. Propor heurísticas de planejamento de capacidade que consideram o modelo de utilidade desenvolvido para determinar os recursos que devem ser reservados junto ao provedor de IaaS;

4. Avaliar através de um modelo de simulação as heurísticas de planejamento propostas. A avaliação das heurísticas se baseia na análise de aspectos de negócio considerados no modelo de utilidade desenvolvido.

1.5 Metodologia

De modo a cumprir os objetivos específicos e, assim, alcançar o objetivo geral proposto as seguintes atividades foram desenvolvidas:

1. Revisão da literatura sobre aplicações SaaS. Através deste estudo buscou-se entender melhor a variedade de aplicações SaaS atualmente em oferta no mercado de Computação na Nuvem. Devido a grande variedade de aplicações encontradas (e.g., aplicações de *streaming*, aplicações de redes sociais, aplicações de comércio eletrônico, etc.), com suas particularidades em relação à carga de trabalho, optou-se por selecionar a aplicação de comércio eletrônico como a aplicação alvo para o estudo;

2. Revisão da literatura para caracterização da carga de trabalho de uma aplicação de comércio eletrônico. O entendimento do funcionamento de cargas de trabalho de comércio eletrônico e o embasamento para a caracterização da carga de trabalho a ser utilizada nas simulações foram os alvos desta etapa;

3. Revisão da literatura e do estado da prática sobre modelos de utilidade para aplicações SaaS. Nesta etapa foram analisados os modelos de negócio que vem sendo utilizados para a comercialização de diversas aplicações SaaS [22] [23] [24] [25] [26] [27], bem como aspectos de negócio relevantes para um provedor de SaaS;

[22] http://www.bigcommerce.com/
[23] http://www.deskaway.com
[24] http://crm.dynamics.com
[25] http://www.salesforce.com/br/
[26] http://www.surveymonkey.com
[27] http://www.volusion.com/

4. Elaboração de um modelo de utilidade para aplicações SaaS. A partir da revisão da literatura realizada foi elaborado um modelo de utilidade que considera conceitos de negócio envolvidos com a oferta de uma aplicação SaaS utilizando recursos de um provedor de IaaS: (i) receitas obtidas junto aos clientes do provedor de SaaS; (ii) quebra de contratos firmados entre o provedor de SaaS e seus clientes; (iii) custos gerados pelo uso de recursos junto ao provedor de IaaS. A quebra de contratos de reserva entre o provedor de IaaS e o provedor de SaaS não foi considerada nesta dissertação;

5. Revisão da literatura sobre planejamento de capacidade. Foram avaliados estudos sobre planejamento de capacidade para aplicações Web e para aplicações de comércio eletrônico utilizando as infraestruturas tradicionais e o mercado de Computação na Nuvem. Além disso, foram analisados alguns trabalhos de provisão dinâmica de recursos para ambientes Web de modo a se obter uma visão ampla sobre o gerenciamento de infraestruturas de TI;

6. Elaboração de heurísticas de planejamento de capacidade. A análise da literatura permitiu a proposição de heurísticas de planejamento de capacidade que consideram conceitos presentes na área como Teoria das Filas [Menasce et al. 2004] e taxas de utilização de recursos. Esta atividade apresentou diversos ciclos, conjuntamente com a revisão da literatura, uma vez que heurísticas foram elaboradas e testadas em cenários de controle de modo a avaliar a utilidade obtida por um provedor de SaaS;

7. Avaliação das heurísticas de planejamento propostas através de simulações. A avaliação das heurísticas teve por base o uso do modelo de utilidade proposto e a definição de uma métrica de comparação entre as heurísticas. A avaliação foi conduzida considerando uma variedade de cenários elaborados a partir da análise de um projeto experimental do tipo fatorial $2^k r$.

1.6 Contribuições e Resultados

Nesta dissertação o planejamento de capacidade baseado em métricas de negócio para aplicações de comércio eletrônico foi avaliado. Heurísticas foram propostas a fim de determinar

a quantidade de recursos a serem reservados em um provedor de IaaS, bem como os tipos destes recursos.

As principais contribuições alcançadas ao longo deste trabalho são:

- **Proposição e avaliação de heurísticas de planejamento de capacidade**: duas heurísticas que realizam o planejamento de capacidade de uma infraestrutura de TI considerando aspectos de negócio foram propostas: (i) Heurística baseada na taxa de utilização dos recursos; (ii) Heurística baseada em Redes de Filas. Tais heurísticas foram avaliadas frente a estratégias mais simples que são referenciadas na literatura.

- **Desenvolvimento de um modelo de negócio para receita e custo de um provedor de SaaS**: as heurísticas propostas para o planejamento de capacidade foram avaliadas de acordo com a utilidade obtida pelo provedor de SaaS ao utilizar os planos de reserva elaborados por tais heurísticas. De modo a avaliar esta utilidade foi desenvolvido um modelo de receita e um modelo de custos obtidos pelo provedor de SaaS. Tais modelos tiveram como base a avaliação de modelos de provedores de aplicações SaaS existentes atualmente no mercado;

Os principais resultados alcançados nesta dissertação são:

- Os resultados demonstraram que as heurísticas que utilizam informações sobre a carga de trabalho da aplicação aliadas ao modelo de utilidade desenvolvido, considerando que o *mercado sob demanda* pode ser utilizado de forma complementar ao *mercado de reservas*, obtiveram as maiores utilidades;

- Considerar o risco de negação de serviço no *mercado sob demanda* por parte do provedor de IaaS também contribuiu para a obtenção de melhores valores de utilidades;

- Superprovisionar a infraestrutura de TI conduziu a valores de utilidade inferiores aos valores de utilidade obtidos por todas as outras heurísticas e estratégias avaliadas;

- A heurística baseada na taxa de utilização dos recursos se apresentou mais propícia a lidar com erros de predição que subestimam a carga de trabalho ao passo que a heurística baseada em rede de filas se apresentou mais propícia a lidar com erros de predição que superestimam a carga de trabalho;

- A heurística baseada em rede de filas teve um ganho médio de $5,993\%$, para cenários com superestimativa da carga de trabalho, e de $1,18\%$, para cenários com subestimativa da carga de trabalho, em relação a estratégia que ignora o *mercado de reservas*. Tais ganhos percentuais representam, respectivamente, valores médios de $\$4.102,84$ e $\$869,183$ considerando conjuntos de 50 e 100 clientes de SaaS;

- A heurística baseada na taxa de utilização dos recursos teve um ganho médio de $1,783\%$, para cenários com superestimativa da carga de trabalho, e de $3,82\%$, para cenários com subestimativa da carga de trabalho, em relação a estratégia que ignora o *mercado de reservas*. Tais ganhos percentuais representam, respectivamente, valores médios de $\$1.262,44$ e $\$2.611,813$ considerando conjuntos de 50 e 100 clientes de SaaS;

1.7 Estrutura da Dissertação

O conteúdo desta dissertação está distribuído ao longo de sete capítulos, assim estruturados:

- **Capítulo 2 – Fundamentação Teórica e Trabalhos Relacionados**. Neste capítulo os principais conceitos necessários para a realização de um planejamento de capacidade conforme o proposto nesta dissertação são apresentados. Outros trabalhos sobre planejamento de capacidade são apresentados e discutidos;

- **Capítulo 3 – Modelo de Utilidade**. Neste capítulo o modelo de utilidade desenvolvido de modo a capturar os aspectos de negócios considerados é discutido. Tal modelo tem por base modelos de receita e custos que foram desenvolvidos considerando provedores de SaaS e de IaaS;

- **Capítulo 4 – Heurísticas de Planejamento de Capacidade**. Neste capítulo as heurísticas de planejamento de capacidade propostas são apresentadas. Tais heurísticas se baseiam no uso de dados sobre a carga da aplicação, bem como no uso do modelo de utilidade apresentado no Capítulo 3;

- **Capítulo 5 – Modelo de Simulação**. Neste capítulo são discutidas as principais características da modelagem do sistema realizada para execução dos experimentos de

simulação. Os principais componentes e características do sistema de comércio eletrônico são apontados e discutidos;

- **Capítulo 6 – Apresentação e Análise dos Resultados**. Os resultados obtidos na avaliação realizada são apresentados neste capítulo. A relação encontrada entre as heurísticas consideradas a partir da métrica proposta, bem como a quantificação da utilidade obtida por cada uma das estratégias, são discutidas;

- **Capítulo 7 – Conclusão e Trabalhos Futuros**. Neste capítulo são apresentadas as conclusões, bem como as limitações do trabalho. Além disso, as questões que ainda se encontram em aberto e ideias para trabalhos futuros são apresentadas.

Capítulo 2

Fundamentação Teórica e Trabalhos Relacionados

Neste capítulo são apresentados conceitos básicos relacionados ao gerenciamento de infraestruturas de Tecnologia de Informação orientado a negócios, ao planejamento de capacidade e ao uso de funções de utilidade. Em seguida, são discutidos os principais trabalhos que estão relacionados com a temática de planejamento de capacidade. Tais trabalhos focam tanto em aplicações Web tradicionais, que utilizavam recursos locais, como em aplicações que fazem uso do mercado de Computação na Nuvem. As principais diferenças em relação ao trabalho apresentado nesta dissertação são destacadas.

2.1 Fundamentação Teórica

Ao realizar o gerenciamento de infraestruturas de Tecnologia de Informação (TI) provedores de serviços buscam oferecer seus serviços de uma forma eficiente, focando em uma boa relação entre custos e receitas. Além da preocupação com a eficiência, existe uma preocupação com os objetivos do negócio e a governança corporativa uma vez que tais serviços agregam valor aos negócios. A Gerência de TI Orientada a Negócios (do inglês, *Business-driven IT Management* – BDIM) [Sauvé et al. 2006] [Moura, Sauvé e Bartolini 2007] argumenta que basear as decisões de gerência de TI apenas em aspectos técnicos (e.g., disponibilidade do serviço, tempo de resposta obtido pelo usuário final, etc.) conduz a um maior prejuízo financeiro nos negócios e uma menor satisfação do cliente [Sauvé et al. 2006] em relação à

gerência que baseia suas decisões considerando aspectos de negócio. O objetivo das práticas de BDIM é que os serviços de TI agreguem de fato valor ao negócio.

O desenvolvimento da área de BDIM trouxe o conceito de que as práticas de gerência deveriam ocorrer seguindo o ciclo PDCA (do inglês, *Plan, Do, Check and Act*) inicialmente proposto por W. Shewhart na década de 30 [Shewhart 1939]. Neste ciclo existe uma fase de planejamento na qual se define e estuda o problema de gerência e o sistema em questão, bem como os objetivos de negócios a serem alcançados. A partir do cumprimento desta etapa existe um conjunto de configurações da infraestrutura de TI a serem efetivadas buscando aperfeiçoar o sistema em questão.

Planejamento de Capacidade de uma infraestrutura da TI [Almeida e Menascé 2002] é uma das atividades de gerência que se adequa à fase de planejamento do ciclo PDCA. Nesta atividade busca-se entender o sistema em funcionamento e caracterizar a carga da aplicação [Arlitt, Krishnamurthy e Rolia 2001] [Menascé et al. 2003] [Menascé et al. 2000] [Zhang, Cherkasova e Smirni 2007] de modo a, utilizando modelos de predição, analisar o comportamento futuro do sistema diante de diferentes configurações da infraestrutura de TI [Almeida e Menascé 2002]. Segundo a literatura da economia[1] existem dois eixos de classificação das práticas de planejamento de capacidade. O primeiro eixo está relacionado com o período de planejamento, ou seja, para quanto tempo se está planejando a infraestrutura. Este eixo divide as técnicas de planejamento em: (i) planejamento de longo prazo; (ii) planejamento de curto prazo. O planejamento de longo prazo envolve estudar e planejar a infraestrutura de TI para um longo período de tempo (e.g., da ordem de dezenas de meses). O planejamento de curto prazo, ou provisão dinâmica de recursos (do inglês, *Dynamic Provisionning System* – DPS) [Ranjan et al. 2002] [Urgaonkar et al. 2008] [Lee et al. 2010] [Kwok e Mohindra 2008] [Zhu et al. 2011] [Bi et al. 2010] [Chi, Qian e Lu 2011], é responsável por controlar a a quantidade de recursos disponíveis na infraestrutura para um curto período de tempo (e.g., da ordem de algumas horas ou minutos). Este controle pode envolver adicionar ou remover recursos da infraestrutura. O segundo eixo está relacionado com a forma através da qual recursos são adquiridos e adicionados à infraestrutura: (i) *lead strategy*, busca adicionar recursos de forma muito antecipada; (ii) *lag strategy*, adiciona recursos apenas quando a infraestrutura está com sua capacidade esgotada; (iii) *match strategy*, busca adicionar recursos

[1] http://www.zarate-consult.de/kosvet3/m4/KEET_M4_LU6_L1/long_and_short_term_capacity_planning.html

em pequenas quantidades em resposta às variações da carga de trabalho.

Considerando que o *mercado de reservas* exige que as reservas de recursos sejam efetuadas para um longo período de uso futuro (e.g., um ou três anos) um planejamento de capacidade que vise reservar recursos pode ser entendido como um planejamento de longo prazo. O sistema de provisão dinâmica de recursos considerado, por outro lado, é responsável por monitorar a carga da aplicação e, quando necessário, adquirir recursos junto ao provedor de IaaS. Independentemente dos recursos adquiridos serem oriundos do *mercado de reservas* ou do *mercado sob demanda*, o sistema de provisão dinâmica adquire, ou libera os recursos, de acordo com a carga de trabalho. Logo, o sistema de provisão dinâmica considerado funciona segundo uma abordagem *match*.

Funções de Utilidade. Segundo a microeconomia, agentes (i.e., provedores de serviço ou clientes) possuem preferências por certas condições e estas preferências determinam seu comportamento [Wilkes 2008]. Tais preferências podem ser mapeadas em valores de utilidade, onde um maior valor de utilidade indica uma maior preferência. Este mapeamento acontece através do uso de funções matemáticas denominadas de funções de utilidade que determinam para um determinado conjunto de condições um valor de utilidade para o agente.

Uma função de utilidade pode ser definida em função de diversos aspectos de um sistema como, por exemplo [Wilkes 2008]: custos com equipamentos, custos com pessoal, tempo de resposta ofertado ao usuário, vazão do sistema, etc. Capturando os valores de cada um destes aspectos a função determina um valor de utilidade que indica a preferência do agente em relação a tais aspectos. Por exemplo, uma função de utilidade que realize um mapeamento de custo para utilidade deve ser capaz de produzir valores mais altos de utilidade para custos mais baixos, indicando que o agente em questão está visando redução de custos.

Funções de utilidade vem sendo utilizadas de modo a combinar diversos aspectos particulares de sistemas em um valor final de utilidade [Strunk et al. 2008] [Maciel et al. 2011]. [Strunk et al. 2008], por exemplo, busca capturar aspectos como receita gerada pelo atendimento de requisições, custo por *downtime* e custo por perda de *dataset* e, assim, obter um valor de utilidade que permita a um agente de computação determinar a quantidade de recursos necessária para oferecer um determinado serviço de armazenamento.

Um planejamento de capacidade orientado a negócios tem a possibilidade de avaliar diversos aspectos de negócio como [Sauvé et al. 2006]: lucros, receitas, custos, perdas finan-

ceiras, nível de produção, rotação de estoque, tempo de colocação de produto no mercado, etc. A avaliação de mais de um aspecto pode ser realizada combinando tais aspectos através do uso de uma função de utilidade. Esta função indica a preferência de um agente de planejamento por uma certa configuração da infraestrutura de TI a ser utilizada na oferta de um determinado serviço.

2.2 Trabalhos Relacionados

De acordo com quais métricas são avaliadas, pode-se dizer que o planejamento de capacidade lida com uma de duas grandes preocupações [Menascé e Ngo 2009]: (i) apenas preocupações técnicas, que estão relacionadas à qualidade do serviço oferecido aos clientes, por exemplo, tempo de resposta ou disponibilidade; (ii) preocupações de negócio, que envolvem aspectos do negócio como, por exemplo, lucro ou perdas. É interessante ressaltar que alguns trabalhos denominam suas técnicas como técnicas de planejamento de capacidade sem especificar que lidam com o problema de planejamento de capacidade de curto prazo, ou seja, provisão dinâmica de recursos [Bichler, Setzer e Speitkamp 2006]. Apesar dos planejamentos de curto e longo prazo serem problemas distintos conceitos bases apresentados para avaliação do sistema e tomadas de decisão em um sistema de provisão dinâmica podem ser avaliados para um planejamento de capacidade de longo prazo.

Planejamento de capacidade considerando apenas métricas técnicas. Algumas propostas sobre como deveria se realizar um planejamento de capacidade de aplicações Web estavam preocupadas em garantir que determinadas metas técnicas fossem alcançadas. Tais metas estavam relacionadas, por exemplo, à disponibilidade alcançada pela solução [Janakiraman, Santos e Turner 2003], ao tempo de resposta percebido pelos clientes ao utilizarem a aplicação [Cherkasova, Tang e Singhal 2004] ou, ainda, à combinação destas métricas [Woolley 2000] [Menascé, Barbará e Dodge 2001]. O planejamento de capacidade proposto era responsável por avaliar diferentes configurações da infraestrutura de TI utilizando predições da carga de trabalho de modo que certos níveis estimados de disponibilidade e tempo de respostas fossem alcançados.

Planejamento de capacidade orientado a negócios. Considerar apenas metas técnicas no planejamento de infraestruturas pode conduzir a situações que sejam economicamente

inviáveis para os provedores, uma vez que nenhuma ponderação sobre custos e retornos ao negócio é realizada [Sauvé et al. 2006]. Por exemplo, sem considerar aspectos de negócio um provedor não consegue avaliar se é viável para o negócio o investimento necessário para se ampliar a disponibilidade de uma certa aplicação de 99,0% para 99,9%. A Gerência de TI Orientada a Negócios buscou alargar o horizonte de gerência de modo que outros aspectos, além dos técnicos, fossem considerados [Sauvé et al. 2006]. Procurou-se ponderar como capturar o impacto nos negócios causado pelas escolhas técnicas, bem como quais métricas técnicas deveriam ser consideradas durante a avaliação de planejamento [Almeida e Menascé 2002] [Almeida 2002].

Uma das possíveis formas de lidar tanto com aspectos técnicos como com aspectos de negócio se baseia na utilização de funções multiobjetivo [Li, Casale e Ellahi 2010] que capturam estes diferentes mundos. O uso de uma função multiobjetivo [Li, Casale e Ellahi 2010] permite relacionar aspectos de desempenho, como tempo de resposta e utilização, com aspectos de negócio, como custo de uma infraestrutura. A partir de um modelo deste tipo pode-se realizar uma otimização que busque maximizar e/ou minimizar o resultado da combinação de aspectos técnicos e de negócio, e não apenas um dos aspectos em particular.

Procurando eliminar dados técnicos dos modelos de negócios e, assim, facilitar a discussão gerencial sobre infraestruturas de TI, alguns trabalhos passaram a desenvolver modelos de avaliação a serem utilizados durante o planejamento de capacidade e a provisão dinâmica que tivessem como resposta apenas aspectos de negócio. Modelos que consideraram o custo de infraestruturas [Bichler, Setzer e Speitkamp 2006] [Stage, Setzer e Bichler 2009] [Wu, Garg e Buyya 2011] e as perdas resultantes do não atendimento de requisições [Marques, Sauvé e Moura 2006] [Sauve et al. 2006] ou, ainda, modelos que consideraram o lucro obtido com certa infraestrutura de TI [Lopes, Brasileiro e Maciel 2010] [Maciel et al. 2008] [Ardagna, Panicucci e Passacantando 2011] foram propostos. Nestes trabalhos metas técnicas estão incluídas como parte do modelo de negócio (i.e., modelo de utilidade) e, assim, o planejamento de capacidade foca apenas em minimizar ou maximizar o aspecto de negócio considerado como alvo do modelo proposto. Além de facilitar a discussão gerencial, alguns trabalhos mostram que, ao se utilizar um modelo de negócios, contribuições significativas são obtidas, por exemplo, através da economia de valores vultuosos [Sauve et al. 2006].

Ao focar no mercado de Computação na Nuvem esta dissertação considera diferentes

mercados de aquisição de recursos (i.e., *mercado de reservas* e *mercado sob demanda*) que não eram considerados nos trabalhos que tratavam do planejamento de capacidade para infraestruturas de TI compostas por recursos locais [Janakiraman, Santos e Turner 2003] [Sauve et al. 2006]. Os novos mercados trazem uma maior flexibilidade para a realização do planejamento de capacidade uma vez que recursos podem ser requisitados a qualquer momento junto a um provedor de IaaS, e esta flexibilidade deve ser considerada durante o planejamento. Além disso, esta dissertação considera o planejamento de capacidade orientado a negócios, tendo como base o uso de um modelo de utilidade que realiza um mapeamento entre utilidade e o lucro alcançado por um determinado provedor de SaaS. Este modelo foi desenvolvido a partir da observação de modelos de negócios de aplicações SaaS que estão atualmente em vigência no mercado. Tais modelos tipicamente não foram retratados nos trabalhos anteriores que consideraram um ambiente de Computação na Nuvem [Lopes, Brasileiro e Maciel 2010] [Maciel et al. 2008] [Stage, Setzer e Bichler 2009] [Ardagna, Panicucci e Passacantando 2011].

Capítulo 3

Modelo de Utilidade

Neste capítulo é feita a apresentação do modelo de utilidade desenvolvido para nortear a atividade de planejamento de capacidade orientada a negócios. Neste modelo o conceito de utilidade representa o lucro obtido por um provedor de SaaS. Este modelo foi elaborado com base em modelos de receitas e modelos de custos obtidos a partir de empresas que atualmente oferecem aplicações SaaS no mercado de Computação na Nuvem.

O modelo de utilidade desenvolvido nesta dissertação tem por base a avaliação do lucro obtido por um provedor de SaaS ao oferecer certo serviço. Para calcular o lucro de provedores de SaaS precisa-se, inicialmente, especificar: (i) a receita do provedor, tipicamente proveniente de pagamentos periódicos realizados pelos clientes de SaaS; (ii) os custos associados com aquisição de recursos junto aos provedores de IaaS para execução da aplicação.

3.1 Modelo de Receita de Aplicações SaaS

Algumas aplicações SaaS atualmente em oferta no mercado foram avaliadas, dentre elas: **BigCommerce** (`http://www.bigcommerce.com/`), **DeskAway** (`http://www.deskaway.com`), **Microsoft CRM Dynamics** (`http://crm.dynamics.com/`), **ServiceCloud** (`http://www.salesforce.com/br/crm/products.jsp`), **SurveyMonkey** (`http://www.surveymonkey.com`), **Volusion Ecommerce Software** (`http://www.volusion.com`), **Evernote** (`http://www.evernote.com/`), **Flickr** (`http://www.flickr.com`), **Linkedin** (`http://www.linkedin.com/`), **doattend** (`http://doattend.com/`) e **CampaignMonitor** (`http://www.`

campaignmonitor.com/). Foi constatado que os provedores atuais oferecem diferentes modelos de tarifação pelo uso da aplicação SaaS. Dentre os diferentes modelos de cobrança existentes podem ser destacados os seguintes:

- **Modelo de Plano Único:** Neste modelo o provedor de SaaS oferece um único plano, que deve ser aceito pelos clientes que desejam obter o serviço. Tal plano dita direitos e deveres dos clientes e do provedor de SaaS. Tipicamente estes provedores são especializados em aplicações B2C (*Business to Consumer*) ofertando serviços a consumidores individuais. Como exemplos de aplicações ofertadas neste modelo pode-se citar o **Evernote**, o **Flickr** e o **Microsoft CRM Dynamics**;

- **Modelo de Planos Múltiplos:** Neste modelo o provedor de SaaS oferece diferentes planos como uma tentativa de atender aos interesses específicos de classes diferentes de clientes de SaaS. Existem planos que permitem ao cliente de SaaS o uso dos serviços contratados em maior ou menor quantidade de acordo com a demanda do cliente. Comumente tais planos são denominados de *Bronze*, *Gold* e *Platinum*, ou ainda *Professional*, *Enterprise* e *Unlimited*. Como exemplos de aplicações ofertadas neste modelo pode-se citar o **Linkedin**, **DeskAway**, **ServiceCloud**, **SurveyMonkey**, **Volusion Ecommerce Software** e o **BigCommerce**;

- **Modelo com ausência de plano:** Neste modelo os provedores de SaaS realizam uma cobrança por transação bem sucedida realizada pelo cliente de SaaS. Este modelo se baseia no conceito de que os clientes de SaaS não possuem um padrão específico de uso da aplicação, então se tornaria mais vantajoso a cobrança de acordo com o uso da aplicação. Como exemplos de aplicações ofertadas neste modelo pode-se citar o **doattend** e o **CampaignMonitor**.

O modelo de negócio proposto nesta dissertação considera apenas provedores de SaaS que oferecem planos ao cliente de SaaS, seja através de um sistema de múltiplos planos, seja através de um sistema de plano único. Com o apoio das avaliações realizadas junto aos provedores de SaaS reais, foi definido um modelo de receitas que busca abranger as principais características encontradas nos provedores investigados: (i) tipicamente, os planos encontrados possuem cobranças mensais ou anuais; (ii) de acordo com a aplicação ofertada

diferentes restrições referentes ao uso da aplicação são impostas aos clientes de SaaS nos contratos; (iii) os contratos firmados por um cliente de SaaS junto ao provedor de SaaS determinam regras sobre como funcionam as políticas de ressarcimento dos provedores de SaaS.

Um determinado provedor de SaaS desenvolve e oferece uma aplicação A para um conjunto de clientes de SaaS $U = \{u_1, u_2, \ldots, u_{|U|}\}$. De modo a oferecer este serviço, o provedor de SaaS elabora um portfólio de planos $P = \{p_1, p_2, \ldots, p_{|P|}\}$, no qual cada plano busca atender padrões específicos de comportamento de uma classe de clientes, logo é uma tendência que $|P| < |U|$. Cada cliente u_k, pertencente a U, escolhe e assina um contrato referente a algum dos planos p_j, pertencente a P, para, assim, fazer uso da aplicação A.

Escolhido um plano de consumo p_j, um cliente u_k tem o direito de usar uma certa aplicação A por um período de duração τ (por exemplo, se o plano for semestral, então, $\tau = 6$ meses). Por simplicidade, considera-se que todos os planos do provedor de SaaS estão associados ao mesmo período de uso, assim, a linha do tempo da aplicação tem granularidade τ e o período n de valor 1 marca o lançamento da aplicação A. Também por questão de simplicidade considera-se que os novos clientes só podem ser inseridos no sistema imediatamente antes de um novo período de uso n iniciar. Com a passagem do tempo, n é incrementado indicando o número do período de uso corrente.

Um cliente pode permanecer no sistema por vários períodos de uso. Pode-se estabelecer que n_k^b indica o número do período em que o cliente entrou no sistema e n_k^e indica o número do período em que o último pagamento foi realizado. Desta forma, um cliente u_k contrata um plano p_j durante um intervalo de tempo dado por $[n_k^b, n_k^e]$.

No momento em que o contrato é assinado, ou seja em n_k^b, o provedor de SaaS deverá configurar e "implantar" a aplicação A para atender ao novo cliente do plano específico. Para tal, o cliente de SaaS deve pagar uma taxa I_j^b de configuração da aplicação para seu propósito. Esta taxa depende do plano p_j contratado pelo cliente. Durante a vigência de um plano, bem como nos períodos de vigência subsequentes nos quais o cliente renovar o contrato, a taxa de configuração I_j^b não será mais cobrada. Uma nova cobrança desta taxa de configuração ocorrerá caso o cliente de SaaS opte por mudar de plano. A função $i^b : \mathbb{N}^+ \Rightarrow \{0, I_j^b\}$ definida como:

$$i_k^b(n) = \begin{cases} I_j^b & \text{se } n = n_k^b \\ 0 & \text{c.c.} \end{cases} \tag{3.1}$$

determina se existe ou não uma taxa de configuração a ser paga pelo cliente u_k em um dado período n.

Para ter o direito de usar efetivamente a aplicação A o cliente deve pagar, a cada período n, uma taxa de uso I_j ao provedor. Esta taxa deve ser suficiente para arcar com os custos do provedor de SaaS para adquirir os recursos necessários para executar a porção da aplicação SaaS que esteve disponível para o cliente u_k no período de uso correspondente. Esta taxa de uso I_j depende do plano p_j que foi contratado previamente pelo cliente u_k. Em reassinaturas subsequentes do contrato associado ao mesmo plano, apenas esta taxa deve ser paga pelo cliente. A função $i^\tau : \mathbb{N}^+ \Rightarrow \{0, I_j\}$ definida como:

$$i_k^\tau(n) = \begin{cases} I_j & \text{se } n_k^b \leq n \leq n_k^e \\ 0 & \text{c.c.} \end{cases} \tag{3.2}$$

representa a taxa de uso paga por um cliente u_k em um dado período n.

Cada plano p_j contempla um conjunto de recursos que podem ser utilizados pelo cliente de SaaS e que são tarifáveis naquele plano. Por exemplo, é comum permitir que durante um período de uso n o cliente, ao usar a aplicação A, possa consumir uma determinada quantidade de espaço de armazenamento em disco e uma determinada quantidade de transferência de dados pela rede. Um plano p_j está, então, associado a um conjunto $R_j = \{r_{j,1}, r_{j,2}, \ldots, r_{j,|R_j|}\}$ de recursos tarifáveis.

O plano do cliente deve, também, estar associado a um conjunto L_j que define limites de uso para cada recurso tarifável $r_{j,m}$ pertencente a R_j. Cada limite $l_{j,m}$ pertencente a L_j define a quantidade do recurso $r_{j,m}$ que um determinado cliente tem direito de usar dentro da franquia de seu plano p_j, *i.e.*, durante um período de tamanho τ.

Sempre que, durante um período n, o cliente u_k ultrapassar os limites de uso definidos em L_j, taxas extra de uso devem ser cobradas. O conjunto E_j define os valores destas taxas extras a serem cobradas pelo uso adicional de cada recurso tarifável $r_{j,m}$ pertencente a R_j. Assim, cada elemento $e_{j,m}$ pertencente a E_j indica o valor monetário extra a ser pago por unidade de uso do recurso $r_{j,m}$ que ultrapassar os limites de uso $l_{j,m}$ definidos no plano p_j.

Finalmente, um plano p_j deve estar associado a um acordo de nível de serviço, aqui representado por SLA_j. Por simplicidade, um SLA é definido nesta dissertação pela tupla $< A^{MIN}, T^{MAX} >$, onde A^{MIN} representa a disponibilidade mínima exigida da aplicação A para o cliente durante um período de uso e T^{MAX} representa um percentil de tempo de resposta aceitável para as requisições servidas aos clientes de SaaS em um período de uso n (e.g., 95% das requisições devem ser atendidas em até 8 segundos). Cada plano p_j possui um acordo de nível de serviço uma vez que, de acordo com as avaliações de negócio realizadas pelo provedor, pode ser viável oferecer uma maior qualidade de serviço para os planos de maior receita. Além disso, uma vez violado este acordo, existe uma função $M_j(n)$ que indica, para um dado período de uso n, quanto o provedor de SaaS deve pagar de multa ao cliente prejudicado. O valor da multa é proporcional à intensidade da violação e pode ser definida de forma diferenciada entre os planos ofertados. É importante destacar que a modelagem do pagamento de penalidades/multas pagas pelo provedor de SaaS está incluída como parte do modelo de custos apresentado na Seção 3.2.

Considerando o que foi discutido acima, um plano de consumo p_j oferecido por um provedor de SaaS a um cliente u_k pode ser representado pela tupla p_j=$< I_j^0, I_j, L_j, E_j, SLA_j, M_j >$.

Estabelecida a caracterização dos planos ofertados pelo provedor de SaaS faz-se necessário estabelecer como é feita a cobrança junto a cada cliente u_k. Durante um período de uso n, o provedor deve realizar a contabilidade do consumo de recursos por parte de cada cliente e, assim, saber identificar requisições de clientes que estão sendo atendidas dentro dos limites estabelecidos no conjunto L_j de seu plano e requisições que estão sendo atendidas fora destes limites. No fim de cada período n, um conjunto Q_k^n de requisições deve ter sido enviado à aplicação A em nome do cliente u_k. Cada requisição de um cliente u_k com plano p_j é representada por uma tupla $< t, \{d_{j,m}, \ldots, d_{|P|,|R_j|}\} >$, em que t representa o momento em que a requisição chegou aos servidores do provedor de SaaS e $\{d_{j,m}, \ldots, d_{|P|,|R_j|}\}$ representa o conjunto de demandas da requisição em cada recurso $r_{j,m}$ pertencente a R_j.

Em qualquer período n, é possível calcular a receita obtida pelo provedor de SaaS proveniente de certo cliente u_k usando a seguinte equação:

$$i_k(n) = i_k^b(n) + i_k^\tau(n) + \sum_{r_{j,m} \in R_j} \left[\max \left(\left(\sum_{\substack{d_{j',m'} \in \{d_{j',m'},\ldots,d_{|P|,|R_j|}\}, \\ <t,\{d_{j',m'},\ldots,d_{|P|,|R_j|}\}> \in Q_k^n, \\ j'=j, m'=m}} d_{j',m'} \right) - l_{j,m}, 0 \right) \cdot e_{j,m} \right] \tag{3.3}$$

Avaliando todo o conjunto U de clientes que contrataram serviços junto ao provedor de SaaS, pode-se calcular a receita total obtida pelo provedor em um dado período n, associada à aplicação A, pela seguinte equação:

$$i(n) = \sum_{\substack{k=1, \\ u_k \in U}}^{k=|U|} i_k(n) \tag{3.4}$$

A receita obtida pelo provedor de SaaS proveniente dos pagamentos realizados pelos clientes em um intervalo de períodos de uso $D = [n^b, n^e]$, onde $n^e \geq n^b$, é dada pela função $\iota : [n^b, n^e] \Rightarrow \mathbb{R}^+$ com $n^b, n^e \in \mathbb{N}^+$:

$$\iota(D) = \sum_{n=n^b}^{n^e} i(n) \tag{3.5}$$

O modelo de receitas apresentado pode ser utilizado tanto para computar a receita obtida em um intervalo D passado ou, ainda, para estimar a receita em um intervalo D futuro. Neste caso, faz-se uso da caracterização atual do conjunto de planos P, bem como de estimativas e predições da carga de trabalho futura a ser submetida por um conjunto U estimado.

3.2 Modelo de Custos dos Provedores de Infraestrutura

A partir da avaliação de alguns provedores de IaaS atualmente em operação no mercado [1,2,3] foram coletadas as principais características dos modelos de tarifação utilizados: (i) recursos computacionais (e.g., computação, armazenamento, memória) são tarifados de acordo com o uso dos mesmos; (ii) recursos podem ser reservados junto aos provedores de IaaS, fornecendo vantagens financeiras aos clientes de IaaS.

[1] http://aws.amazon.com/ec2/

[2] http://www.cloudsigma.com

[3] http://www.rackspace.com

Ao montar sua infraestrutura de TI sobre um provedor de IaaS para oferecer sua aplicação, um provedor de SaaS passa a gerenciar os seguintes custos: (i) custos de uso dos recursos adquiridos; (ii) custo de reserva de recursos. Além destes custos, uma outra fonte de despesas para o provedor de SaaS está relacionada às violações dos acordos de serviço (i.e., SLA), que ocasionam o pagamento de penalidades/multas aos clientes de SaaS.

Os custos de uso dos recursos adquiridos junto aos provedores de IaaS podem ser divididos por recurso ofertado. Cada provedor de IaaS possui um conjunto O de classes de recursos ofertados, sendo $|O| \geq |R_j|$. O conjunto O pode ser, por exemplo, formado pelos seguintes elementos: (i) recursos de processamento; (ii) recursos de memória; (iii) recursos de armazenamento; (iv) recursos de transferência.

Cada classe de recurso ofertado, $o \in O$, deve estar associado a um conjunto de tuplas $C_o = \{< s, c_s >\}$ onde s indica o tipo do recurso e c_s indica o custo de uso de um grão θ do recurso s. Este grão θ pode assumir diferentes unidades de acordo com a classe o do recurso. Por exemplo, considerando o serviço **Amazon EC2**[1] percebe-se que para a classe de processamento (o = processamento) existem diferentes tipos de recursos, por exemplo: o tipo *small* (s = *small*) possui 1 unidade de processamento; o tipo *large* (s = *large*) possui 4 unidades de processamento, etc. Seguindo o modelo de tarifação do **Amazon EC2**, cada um destes recursos pode ser utilizado por um grão θ, cujo valor é de 1 hora, pelos seguintes custos: (i) $c_{small} = \$0,085$; (ii) $c_{large} = \$0,34$, etc.

De acordo com a classe o do recurso a cobrança pode ser sempre feita em função de múltiplos inteiros de θ, ao passo que para outras classes de recursos pode-se avaliar uma fração da grandeza θ e tarifar um valor proporcional ao valor c_s. Por exemplo, para os recursos de processamento do serviço **Amazon EC2** o grão θ é igual a 1 hora e os recursos são tarifados como múltiplos de hora, ou seja, mesmo que um determinado recurso seja utilizado apenas por 10 minutos o mesmo será tarifado por uma hora inteira de uso. Para recursos de transferência no serviço **Amazon EC2** o grão θ é igual a 1 GB e os recursos são tarifados de uma forma menos rígida. Para estes recursos cobra-se $\$0,12$ por cada gigabyte transferido. Caso um total de 1,5 GB seja transferido será cobrado um valor igual a $\$0,18$ (1,5 GB x $\$0,12$), portanto proporcional ao montante transferido.

O conjunto C_o pode ser entendido como metainformação a respeito dos diferentes recursos que podem ser adquiridos em provedores de IaaS e de seus custos. Por questões de

simplicidade, para cada classe o de recurso ofertado todos os tipos s desta classe possuem o mesmo grão base de uso θ. Esta simplificação está de acordo com o que acontece atualmente no mercado, no qual todos os recursos concedidos para uso pelos provedores de IaaS possuem valores de grãos uniformes. Por exemplo, no serviço **Amazon EC2** todos os recursos de processamento são tarifados pela quantidade de horas utilizadas e os recursos de transferência são tarifados pela quantidade de gigabytes transferida. Para os recursos cuja unidade de θ está relacionada com o tempo, a relação entre τ e θ é definida de tal modo que τ é muito maior que θ, na ordem de centenas ou milhares de vezes.

Definidas as metainformações de custo é necessário que o provedor de IaaS possua um sistema de contabilidade que armazene as quantidades consumidas de cada recurso adquirido, bem como seus respectivos tipos, para cada período n. Esta contabilidade deve, então, indicar, para um dado período de uso n e para cada tipo de recurso s, quantos grãos θ do recurso foram consumidos. Neste caso, para cada s e n existirão contadores a_s^n que serão incrementados sempre que um grão θ do recurso s for consumido dentro do período de uso n. Por exemplo, suponha que durante o primeiro período contabilizado, $n = 1$, um recurso de processamento de tipo $s =$ *large* tenha sido utilizado durante 10 horas. Neste caso, o contador a_{large}^1 possuiria um valor igual a 10.

O custo do provedor de SaaS associado ao uso de recursos, sejam estes obtidos no *mercado de reservas* ou no *mercado sob demanda*, em um período n é dado pela função $ca : \mathbb{N}^+ \Rightarrow \mathbb{R}^+$ definida como:

$$ca(n) = \sum_{o \in O} \left[\sum_{<s,c_s> \in C_o} a_s^n \cdot c_s \right] \tag{3.6}$$

Mesmo o uso de recursos que foram previamente reservados será contabilizado na equação acima, desde que tais recursos reservados estejam devidamente representados por um tipo s e um custo c_s que representem as tarifas de uso do *mercado de reservas*.

Além dos custos relacionados ao uso efetivo dos recursos, existe um outro custo para o provedor de SaaS relacionado ao ato de realizar reservas prévias junto ao provedor de IaaS. A reserva de um recurso de classe o do tipo s deve estar sempre associada a uma quantidade de tal recurso que foi reservada (a_s), à taxa de reserva cobrada pelo provedor de IaaS (f_s) e ao período no qual o recurso reservado estará disponível para uso. Desta forma pode-se definir um contrato de reserva como sendo:

$$V =< o, s, a_s, f_s, n_s^b, n_s^e >$$

onde n_s^b e n_s^e indicam, respectivamente, o período a partir do qual o recurso reservado está disponível para uso e o período limite para uso do recurso reservado. É importante destacar que este período de validade do contrato de reserva deve ser definido com base nos períodos n nos quais o provedor de SaaS usará recursos para ofertar sua aplicação. O conjunto γ representa o conjunto dos contratos de reserva firmados entre o provedor de SaaS e o provedor de IaaS. É importante relembrar que os provedores de IaaS atuais oferecem a possibilidade de reserva apenas para recursos de processamento, todavia o modelo de custos aqui elaborado é flexível para contemplar outros recursos que possam vir a serem disponibilizados para reserva no futuro.

As taxas de reserva pagas pelo provedor de SaaS podem ser amortizadas ao longo de todos os períodos de uso dos recursos reservados e, assim, a cada período n tem-se um componente do custo que está relacionado com a amortização dos contratos de reserva definidos no conjunto γ. Este componente pode ser computado de acordo com a função $cv : \mathbb{N}^+ \Rightarrow \mathbb{R}^+$ definida como:

$$cv(n) = \begin{cases} \sum_{<o,s,a_s,f_s,n_s^b,n_s^e>\in\gamma} \frac{f_s \cdot a_s}{n_s^e - n_s^b} & \text{se } n_s^b \leq n \leq n_s^e, \\ 0 & \text{caso contrário.} \end{cases} \tag{3.7}$$

Definidos os componentes do custo relacionados ao uso, $ca(n)$, e à reserva, $cv(n)$, o custo total de um provedor de SaaS em um período de uso n é dado pela função $c : \mathbb{N}^+ \Rightarrow \mathbb{R}^+$ definida como:

$$c(n) = ca(n) + cv(n) + p(n) \tag{3.8}$$

onde $p(n) = \sum_{u_k \in U} M_j(n)$, ou seja, $p(n)$ representa o conjunto de todas as penalidades pagas pelo provedor de SaaS em um período de uso n. Uma penalidade deve ser paga a um cliente u_k sempre que o SLA_j estabelecido em seu plano p_j for violado. A violação de um SLA, conforme mencionado na Seção 3.1, está relacionada à violação de restrições de disponibilidade e tempo de resposta estabelecidas no plano p_j contratado junto ao provedor de SaaS.

Finalmente, é possível avaliar o custo total de um provedor de SaaS em um intervalo de períodos de uso $D = [n^b, n^e]$, onde $n^e \geq n^b$, através da função $\alpha : [n^b, n^e] \Rightarrow \mathbb{R}^+$ com $n^b, n^e \in \mathbb{N}^+$:

$$\alpha(D) = \sum_{n=n^b}^{n^e} c(n) \tag{3.9}$$

O modelo de custos apresentado pode ser utilizado tanto para computar o custo obtido em um intervalo D passado ou, ainda, para estimar o custo em um intervalo D futuro. Neste caso, trabalha-se com estimativas e predições do conjunto de requisições em cada período n futuro e com um plano de uso estimado dos recursos a serem adquiridos junto ao provedor de IaaS.

3.3 Função de Utilidade: Lucro do provedor de SaaS

Conforme mencionado anteriormente, a função de utilidade proposta nesta dissertação, e que serve como base para um agente de planejamento de capacidade, é definida em função do lucro alcançado por um provedor de SaaS. Uma vez definidos o modelo de receita, Seção 3.1, e o modelo de custo, Seção 3.2, a função de utilidade do provedor de SaaS em um intervalo de períodos de uso $D = [n^b, n^e]$, onde $n^e \geq n^b$, é dada pela função $\upsilon : [n^b, n^e] \Rightarrow \mathbb{R}$ com $n^b, n^e \in \mathbb{N}^+$:

$$\upsilon(D) = \iota(D) - \alpha(D) \tag{3.10}$$

Além de guiar a atividade de planejamento de capacidade, esta função pode ser utilizada, também, para avaliar quão positivas foram as escolhas realizadas por um provedor de SaaS ao montar sua infraestrutura. Durante o planejamento de capacidade a função permite estimar uma utilidade e, assim, avaliar dentre vários planos de reserva qual será mais vantajoso para o negócio. Após montar a infraestrutura a função permite ao provedor avaliar qual a utilidade que de fato foi obtida pelo negócio.

Capítulo 4

Heurísticas de Planejamento de Capacidade

Este capítulo tem por objetivo descrever as heurísticas propostas para serem utilizadas por um agente de planejamento de capacidade na construção dos planos de reservas a serem firmados com o provedor de IaaS. As heurísticas apresentadas neste capítulo são o foco do estudo conduzido nesta dissertação.

As heurísticas propostas a seguir utilizam dados históricos, ou estimativas destes dados, que caracterizam a carga de trabalho submetida ao provedor de SaaS durante um intervalo de tempo igual ao intervalo de tempo para o qual se deseja planejar a infraestrutura. Por exemplo, suponha que se deseje realizar o planejamento de capacidade de uma infraestrutura de TI para um período de um ano. As heurísticas propostas podem realizar tal planejamento com base em dados históricos ou estimativas de um ano da carga de trabalho da aplicação em foco.

De posse da caracterização da carga de trabalho da aplicação, as heurísticas buscam avaliar a reserva de recursos de processamento em provedores de IaaS tendo por base o modelo de utilidade apresentado no Capítulo 3. É importante destacar que essa avaliação é um processo de otimização que busca maximizar a utilidade obtida pelo provedor de SaaS. Ao final desta avaliação as heurísticas elaboram um plano de reservas de recursos junto ao provedor de IaaS especificando:

- As classes de recursos a serem reservados;

- A quantidade de recursos a serem reservados, por tipo de recurso da classe específica.

Duas heurísticas foram propostas e avaliadas nesta dissertação:

- Heurística baseada na taxa de utilização dos recursos – UT;
- Heurística baseada em rede de filas – RF.

4.1 Conceitos Gerais

Os recursos computacionais a serem reservados pelas heurísticas de planejamento de capacidade propostas são recursos de processamento. O *mercado de reservas* praticado pelo serviço **Amazon EC2** define que a reserva de um recurso só é viável para um cliente de SaaS quando o recurso possui uma taxa de utilização acima de determinado limiar. O conceito de taxa de utilização de um recurso de processamento está relacionado à porção de tempo, por exemplo, tempo de CPU, do recurso que é efetivamente utilizada em relação a um intervalo de tempo em que o recurso esteve disponível para uso.

Avaliando o modelo de custos elaborado na Seção 3.2 percebe-se que os recursos são tarifados em função da quantidade de grãos θ consumidos. Para os recursos de processamento, este grão é tipicamente definido em função de uma hora de CPU. Avaliando os componentes do modelo de custo definidos na Seção 3.2 percebe-se que o custo de um recurso pode ser composto de duas formas:

- A partir do uso de grãos θ do recurso quando este é oriundo do *mercado sob demanda*, cujo custo por grão θ é aqui referenciado como c_s^{on};
- A partir do uso de grãos θ do recurso quando este é oriundo do *mercado de reservas*, cujo custo por grão θ é aqui referenciado como c_s^{v}, e a partir do custo de reserva do recurso, aqui referenciado como f_s.

Suponha uma quantidade X_s^o de grãos θ consumidos do recurso de classe o de tipo s. Caso esta quantidade tenha sido consumida no *mercado sob demanda*, o custo total do recurso seria dado por

$$X_s^o * c_s^{on}$$

Caso esta quantidade tenha sido consumida no *mercado de reservas* o custo total do recurso seria

$$f_s + X_s^o * c_s^v$$

Logo, pode-se verificar a viabilidade da reserva de um recurso que consumiu uma quantidade X_s^o de grãos θ a partir da inequação:

$$X_s^o \geq \frac{f_s}{(c_s^{on} - c_s^v)} \tag{4.1}$$

A inequação 4.1 permite avaliar os preços praticados no *mercado de reservas* e no *mercado sob demanda* de modo a determinar qual a quantidade mínima de grãos θ de um recurso que devem ser utilizados para que a reserva deste recurso se torne financeiramente viável. Por exemplo, suponha um recurso de processamento ($o =$ processamento), cujo grão $\theta = 1$ hora, que tenha sido utilizado por uma quantidade $X_s^{processamento}$ de horas. Caso o valor de $X_s^{processamento}$ seja igual ou superior ao valor dado por $\frac{f_s}{(c_s^{on} - c_s^v)}$ a reserva deste recurso junto ao provedor de IaaS seria financeiramente viável para o provedor de SaaS.

4.2 Heurística baseada na taxa de utilização dos recursos – UT

A heurística UT tem como base a avaliação da viabilidade de reserva de recursos a partir da inequação 4.1. Logo, descrita a inequação pode-se especificar o procedimento realizado pela heurística para elaboração de um plano de reservas. Uma vez que o agente de planejamento de capacidade conhece todo o funcionamento do sistema, a heurística UT simula a execução da carga de trabalho da aplicação durante um período cuja duração é igual à duração do período que se deseja planejar. Esta simulação utiliza apenas recursos oriundos do *mercado sob demanda* que são requisitados dinamicamente ao provedor de IaaS por um sistema de provisão dinâmica de recursos para atender a demanda esperada. Ou seja, a heurística UT considera que o consumo de recursos de processamento realizado é um consumo adequado para execução da aplicação em termos de atendimento das requisições.

Durante a simulação, a heurística UT mantém para cada classe o e para cada tipo de recurso s o rastro de quantos grãos θ, horas de CPU, são requisitados e consumidos a cada

hora de simulação junto ao provedor de IaaS. A quantidade de grãos θ consumidos a cada hora é um indicativo de quantos recursos foram requisitados e utilizados durante aquela hora junto ao provedor de IaaS. Ao final da simulação a heurística UT realiza um conjunto de avaliações, apontadas no Algoritmo 1, para elaborar o plano de reservas de recursos. Inicialmente, a heurística atualiza para cada classe o e para cada tipo s o consumo de grãos θ considerando que se um total de Y recursos foram consumidos durante 10 horas, $Y - 1, Y - 2, \ldots, 1$ recursos também foram consumidos durante as mesmas 10 horas (linha 5 do Algoritmo 1).

Após esta atualização de consumo a heurística UT obtém para cada classe de recurso o e para cada tipo s um conjunto de dados que indica para cada quantidade de recursos o total de horas durante as quais esta quantidade de recursos foi utilizada. De posse destes dados a heurística busca, para cada classe o e para cada tipo s, a maior quantidade de recursos Y_s^o cujo consumo X_s^o de grãos θ seja maior ou igual a $\frac{f_s}{(c_s^{on} - c_s^v)}$ (linha 6 do algoritmo). Encontradas as quantidades Y_s^o de recursos cujos consumos atingiram os limiares definidos pela inequação 4.1, a heurística UT define a quantidade de recursos a serem efetivamente reservados. A quantidade de recursos da classe o de tipo s a serem reservados é dada por (linha 7 do algoritmo):

$$reserva_s^o = \lceil \frac{Y_s^o * X_s^o}{\frac{f_s}{(c_s^{on} - c_s^v)}} \rceil$$

Algoritmo 1 Heurística baseada na taxa de utilização dos recursos

1: **procedure** PLANEJAMENTODECAPACIDADEUT ($historico$) ▷ mapa de consumo dos recursos a cada hora
2: **for all** recurso $o \in O$ **do**
3: **for all** tipo s **do**
4: Calcular limiar $\frac{f_s}{(c_s^o n - c_s^v)}$
5: Para cada Y recursos consumidos por X horas atualiza consumo de $Y-1$, $Y-2, \ldots, 1$ recursos adicionando X horas
6: Busca maior quantidade Y_s^o de recursos tal que seu consumo obedeça a inequação $X_s^o \geq \frac{f_s}{(c_s^{on} - c_s^v)}$
7: $reserva_s^o = \lceil \frac{Y_s^o * X_s^o}{\frac{f_s}{(c_s^{on} - c_s^v)}} \rceil$
8: Adiciona $reserva_s^o$ ao plano de reservas
9: **end for**
10: **end for**
11: **end procedure**

É importante destacar que a heurística UT não realiza aproximações e conversões entre tipos de recurso de modo a buscar um melhor plano de reserva. As reservas são realizadas considerando, de forma separada, os tipos s utilizados durante a simulação de execução da carga de trabalho. Um recurso de tipo s_1, de grande poder computacional, que tenha sido utilizado durante um curto período de tempo dentro do período de planejamento não é aproximado e convertido de modo que recursos de um tipo s_2, de menor poder computacional, sejam reservados.

Como exemplo do algoritmo acima, suponha que uma determinada carga de trabalho tenha sido submetida à aplicação A durante um período de um ano. Suponha que esteja sendo utilizado o serviço **Amazon EC2**[1] para obtenção de recursos de processamento para execução da aplicação. Ao simular a execução da aplicação a heurística UT coletou as seguintes informações de uso de recursos: (i) foram utilizados três recursos do tipo *small* da classe de processamento *standard* e um recurso do tipo *large* da classe de processamento *standard*; (ii) três recursos do tipo *small* foram utilizados durante 876 horas; (iii) dois recursos do tipo

[1] http://aws.amazon.com/ec2/

small foram utilizados durante 2628 horas; (iv) um recurso do tipo *small* foi utilizado durante 876 horas; (v) um recurso do tipo *large* foi utilizado durante 876 horas.

Utilizando as tarifas praticadas pelo **Amazon EC2** para o tipo *small* ($c^{on}_{small} = 0,085$, $c^{v}_{small} = 0,03$ e $f_{small} = 227,50$) a heurística UT calcula o limiar como sendo $4136,36$ horas (linha 3 do algoritmo). Utilizando os valores praticados pelo **Amazon EC2** para o tipo *large* ($c^{on}_{large} = 0,34$, $c^{v}_{large} = 0,12$ e $f_{large} = 910$) UT calcula o limiar como sendo $4136,36$ horas. Definidos os limiares de consumo, a heurística UT atualiza os dados de consumo obtidos no histórico (linha 5 do algoritmo). Com esta atualização temos que: (i) três recursos do tipo *small* estiveram ativos durante 876 horas; (ii) dois recursos do tipo *small* estiveram ativos durante 3504 horas; (iii) um recurso do tipo *small* esteve ativo durante 4380 horas; (iv) um recurso do tipo *large* esteve ativo durante 876 horas. Em seguida, a heurística busca pela maior quantidade de recursos que pode ser reservada na classe *standard* para o tipo *small* e para o tipo *large*:

- Avaliando o tipo *small*, percebe-se que apenas o recurso que esteve em uso durante 4380 horas foi utilizado por um período superior ao definido pelo limiar. Efetuando o cálculo presente na linha 7 do Algoritmo 1 a heurística UT obtém que $reserva^{standard}_{small} = \lceil 1,058902 \rceil$ recursos. A partir deste valor uma reserva para dois recursos do tipo *small* é adicionada ao plano de reservas;

- Avaliando o tipo *large*, percebe-se que o único recurso requisitado esteve em uso durante 876 horas sendo, portanto, utilizado por um intervalo de tempo inferior ao limiar. Devido à baixa utilização deste recurso nenhuma reserva para recursos do tipo *large* é adicionada ao plano de reservas.

Ao final da avaliação, o plano de reservas elaborado recomenda que sejam reservados dois recursos de processamento do tipo *small* para o próximo período de um ano de execução da aplicação.

4.3 Heurística baseada em Redes de Filas – RF

Uma alternativa muito recorrente na literatura para se avaliar o desempenho de um sistema é o uso dos conceitos de Teoria das Filas [Almeida 2002] [Urgaonkar et al. 2005] [Sauve et

al. 2006] [Marques, Sauvé e Moura 2006] [Liu, Heo e Sha 2005]. Por exemplo, [Urgaonkar et al. 2005] e [Liu, Heo e Sha 2005] apresentam alternativas de como se modelar uma aplicação Web utilizando os conceitos de Teoria das Filas. Tais conceitos permitem que se elabore uma representação de uma infraestrutura de TI, na qual existem recursos que devem executar requisições e filas que são formadas de acordo com a carga de trabalho submetida à infraestrutura. Os recursos que podem ser modelados segundo estes conceitos podem ser de tipos variados, por exemplo, recursos de processamento, armazenamento ou transferência. A partir da avaliação das chamadas Redes de Filas pode-se calcular, dentre outros valores:

- A vazão da rede, ou seja, a quantidade de requisições atendidas por intervalo de tempo;
- O tempo médio de serviço, ou seja, o intervalo de tempo médio necessário para se atender uma requisição em um recurso;
- O tempo de resposta, ou seja, o intervalo de tempo necessário para se atender por completo uma requisição;
- A quantidade de requisições que não puderam ser atendidas dado o estado de contenção do sistema.

A avaliação de uma Rede de Filas pode acontecer de duas formas: (i) através do uso das chamadas Leis Operacionais [Menasce et al. 2004], que permitem a obtenção de dados sobre o sistema sem a necessidade de considerar alguma hipótese sobre as distribuições que determinam o padrão de chegada das requisições e o tempo de serviço das requisições; (ii) através de simulação da infraestrutura modelada monitorando as variáveis de interesse da Rede de Filas durante a simulação. A heurística RF, aqui proposta, busca realizar uma avaliação de vários planos de reservas possíveis, para recursos de processamento, de modo a escolher um plano a ser efetivado. Por conta desta característica, optou-se por realizar uma avaliação utilizando as chamadas Leis Operacionais uma vez que a simulação de vários planos de reserva em longos períodos de planejamento (e.g., um ano) demanda um tempo de execução avaliado como não factível para os propósitos da pesquisa.

Considerando o conjunto de dados históricos, ou estimativas destes dados, que caracterizam a carga de trabalho da aplicação a heurística RF inicialmente realiza um pré-processamento destes dados. Este pré-processamento é realizado para cada hora do período

de avaliação (linha 2 do Algoritmo 2) de modo a obter: (i) a taxa média de chegada de requisições por segundo, grandeza esta comumente representada por $\lambda = \frac{\#req}{3600}$; (ii) o tempo médio de serviço, grandeza esta comumente representada por S; (iii) o número de usuários que estão utilizando a aplicação, grandeza esta aqui representada por N; (iv) o tempo médio de espera de um usuário (do inglês, *think time*) para submeter uma nova requisição, grandeza esta comumente representada por Z.

Realizado o pré-processamento da carga de trabalho, a heurística RF considera o conjunto de recursos que são ofertados pelo provedor de IaaS e elabora os possíveis planos de reservas financeiramente viáveis a serem firmados junto ao provedor (linhas 4 a 9 do Algoritmo 2). Neste processo a heurística RF considera a quantidade mínima, $\frac{f_s}{(c_s^{on} - c_s^v)}$, de grãos θ, horas de CPU, que deveria ser consumida para cada tipo s de recurso de modo a tornar uma reserva viável, bem como uma estimativa do total T de grãos θ dos recursos de processamento que seriam consumidos junto ao provedor de IaaS. Este total T é o total de grãos θ requisitados pelo sistema de provisão dinâmica de recursos ao provedor de IaaS para atender à demanda da aplicação. Através da divisão destes valores, $MAX_s^o = \lfloor \frac{T}{\frac{f_s}{(c_s^{on} - c_s^v)}} \rfloor$, obtém-se para o tipo s específico a quantidade máxima de recursos cuja reserva seria financeiramente viável. De posse do máximo de recursos cuja reserva seria viável para cada tipo s a heurística RF elabora todos os possíveis planos de reserva considerando todas as combinações de reserva dentro das quantidades máximas de recursos calculadas. Por exemplo, suponha que para uma determinada carga de trabalho $MAX_{small}^{processamento} = 10$ e $MAX_{large}^{processamento} = 3$ a heurística RF elabora todos os possíveis planos de reserva que tenham entre 0 e 10 recursos do tipo *small* e entre 0 e 3 recursos do tipo *large*.

Elaborados os possíveis planos de reservas, a heurística busca determinar uma estimativa da utilidade que poderia ser alcançada por cada plano (linhas 10 a 16 do Algoritmo 2 e linhas 17 a 25 do Algoritmo 3) utilizando o modelo apresentado no Capítulo 3. Inicialmente a heurística RF considera a taxa de chegada de requisições λ e distribui, inicialmente, as requisições entre os recursos disponíveis no plano de reservas sob análise no momento. Esta distribuição acontece de maneira proporcional à quantidade de núcleos presentes em cada recurso reservado.

Em seguida, a heurística calcula quantas requisições são atendidas por cada recurso reservado a cada unidade de tempo, ou seja, a vazão de cada recurso. Por exemplo, suponha

um cenário no qual um recurso J possui 1 núcleo, um recurso I possui 2 núcleos e o sistema apresenta uma taxa total de chegada $\lambda = 6$ requisições/segundo, com tempo médio de serviço igual a 1 segundo. O recurso J receberá uma taxa de chegada $\lambda_1 = 2$ requisições/segundo, ao passo que o recurso I receberá uma taxa $\lambda_2 = 4$ requisições/segundo. Por possuir um núcleo o servidor J apresenta uma vazão de 1 requisição/segundo, ao passo que o servidor I apresenta uma vazão de 2 requisições/segundo.

Avaliando a vazão total obtida junto aos recursos reservados e o percentil de tempo de resposta aceitável T^{MAX} a heurística determina se a taxa λ pode ser processada apenas pelos recursos reservados sem violação do SLA. Caso existam requisições que não poderiam ser atendidas pelos recursos reservados, a heurística RF assume que seriam utilizados recursos oriundos do *mercado sob demanda* para o processamento destas requisições. A heurística considera, ainda, um risco de negação de serviço no *mercado sob demanda* do provedor de IaaS (linha 14 do algoritmo). Esta negação de serviço pode fazer com que uma determinada quantidade de requisições que deveriam ser atendidas no *mercado sob demanda* não sejam atendidas. Tal negação pode ocorrer por dois motivos: (i) um limite máximo de uso de recursos sob demanda ter sido alcançado; (ii) o provedor de IaaS não dispõe de recursos sob demanda disponíveis para serem fornecidos ao cliente SaaS [2] [3].

Feita a contabilidade de quantas requisições são atendidas pelos recursos reservados e quantas são atendidas pelos recursos sob demanda, a estratégia estima a quantidade de horas de CPU que seriam requisitadas junto ao provedor de IaaS (linhas 16 a 22 do Algoritmo 3) em cada mercado. Os recursos requisitados no *mercado sob demanda* são recursos de um tipo específico pré-definido pelo gerente da infraestrutura. A quantidade T de horas de CPU requisitadas ao provedor de IaaS depende do funcionamento do sistema de provisão dinâmica de recurso. Um sistema de provisão dinâmica tipicamente deverá requisitar uma quantidade de horas de CPU que seja igual ou superior à demanda da aplicação. Requisitar uma quantidade de horas de CPU superior à demanda pode ser uma ação preventiva tomada na provisão dinâmica de modo a evitar que requisições não sejam atendidas. Dado que o total de horas requisitados ao provedor de IaaS depende do sistema de provisão dinâmica em uso, a heurística RF considera que o modo de funcionamento do sistema de provisão dinâmica é

[2] http://aws.amazon.com/ec2/purchasing-options/

[3] Seção *Detailed Description*: http://aws.amazon.com/ec2/

conhecido e com isso tem-se uma estimativa de quantas horas de CPU são requisitadas ao provedor de IaaS para cada hora de CPU demandada pela carga da aplicação. Esta estimativa é aqui representada pela variável $FATOR\_AJUSTE$. Um $FATOR\_AJUSTE = 1,0$ indica que o sistema de provisão dinâmica de recursos requisita ao provedor de IaaS um total T de horas de CPU que corresponde exatamente ao total demandado pela carga da aplicação.

A heurística avalia, ainda, o tempo médio de resposta utilizando as Leis Operacionais. Tal valor é utilizado para computar a quantidade de requisições que violam o SLA estabelecido no plano de cada um dos clientes de SaaS e que acarretam penalidades para o provedor de SaaS. Finalmente, a heurística calcula uma estimativa de lucro, ou estimativa de utilidade, $E[\upsilon(D)]$ (linha 23 do Algoritmo 3). Este valor se baseia na estimativa de custos gerados pelo consumo de recursos reservados (calculado a partir da estimativa do total de CPU horas consumidas neste mercado e das taxas de reserva pagas por recurso), em uma estimativa de custos gerados pelo consumo de recursos do *mercado sob demanda* (calculado a partir da estimativa do total de CPU horas consumidas neste mercado), em uma estimativa de penalidades pagas aos clientes de SaaS (calculada a partir das requisições que não foram atendidas e das requisições que violaram o SLA) e, por fim, em uma estimativa de receitas obtidas pela cobrança dos planos de todos os clientes de SaaS.

Ao final, linha 25 do Algoritmo 2, o plano de reservas com maior valor de utilidade é escolhido para ser implantado.

Algoritmo 2 Heurística baseada em Redes de Filas: Parte 1

1: **procedure** PLANEJAMENTODECAPACIDADERF ($historico$) ▷ historico representa logs da carga de trabalho recebida na infraestrutura de TI
2: sumário $\leftarrow$ calcula λ, S, Z e N para cada intervalo de uma hora com base no histórico
3: estima $T = FATOR\_AJUSTE * \sum S_m$
4: **for all** classe o in O **do**
5: **for all** tipo s **do**
6: $MAX_s^o = \lfloor T / \frac{f_s}{(c_s^{on} - c_s^v)} \rfloor$
7: **end for**
8: **end for**
9: elabora possíveis planos de reserva, para cada tipo s da classe o, contendo de 0 a MAX_s^o recursos
10: **for all** planoReserva **do**
11: utilidade[planoReserva] $\leftarrow$ 0
12: **for all** hora in sumário **do**
13: avalia requisições atendidas pelos recursos reservados
14: avalia requisições atendidas pelos recursos sob demanda e quantidade de requisições rejeitadas devido ao risco de negação de serviço
15: avalia quantidade de requisições cujo tempo de resposta violou o SLA_j

Algoritmo 3 Heurística baseada em Redes de Filas: Parte 2

16: **for all** classe o in planoReserva **do**
17: **for all** $< s, c_s >$ in o **do**
18: $consumoReservado_s^o = \lceil$ # de requisições atendidas pelos recursos reservados $* \bar{S} * FATOR\_AJUSTE \rceil$
19: **end for**
20: **end for**
21: $consumoSobDemanda = \lceil$ # de requisições atendidas pelos recursos sob demanda $* \bar{S} * FATOR\_AJUSTE \rceil$
22: **end for**
23: utilidade[planoReserva] $\leftarrow E[\upsilon(D)]$
24: **end for**
25: **return** planoReserva que maximiza o valor de utilidade[planoReserva]
26: **end procedure**

Suponha que tenham sido coletados dados sobre uma carga de trabalho constante submetida à aplicação A durante um período de um ano. Através do pré-processamento destes dados (linha 2 do algoritmo 2) a heurística RF obtém que: $\lambda = 30$ requisições/segundo, $Z = 5$ segundos, $S = 500$ ms e $N = 50$ usuários finais. Suponha, ainda, que o sistema de provisão dinâmica requisita ao provedor de IaaS apenas a quantidade de horas de CPU para executar a demanda, ou seja, $FATOR\_AJUSTE = 1$, e que o SLA defina um $T^{MAX} = 1$ segundo. A variável T (linha 3 do algoritmo), carga total, é de 131.400 horas. Considerando que o provedor de IaaS ofereça recursos da classe de processamento *standard* do tipo *small* e que uma reserva torna-se viável a partir de um consumo de $4136,36$ horas do recurso. Desta forma $MAX_{small}^{standard} = 31$ recursos (linhas 4 a 8 do algoritmo).

A heurística RF elabora os possíveis planos de reserva contemplando a reserva de 0 recursos até a reserva de 31 recursos do tipo *small*. Para cada um destes planos RF irá calcular uma estimativa de utilidade. Por exemplo, considere o plano de reserva com 14 recursos reservados. Considerando que um recurso *small* possui um núcleo de processamento e que $T^{MAX} = 1$ segundo, cada recurso consegue executar 2 requisições/segundo (linha 13). Um total de 2 requisições/segundo não podem ser executadas pelos recursos reservados sem vi-

olar o T^{MAX} e são encaminhadas para recursos sob demanda (linha 14). Considerando que não houve negação de serviço pelo provedor de IaaS e que as requisições foram atendidas sem violar o SLA (linha 15) segue-se o cálculo da quantidade de horas de CPU requisitadas em cada mercado. No *mercado de reservas* tem-se que $consumoReservado = 14$ horas de CPU e no *mercado sob demanda* tem-se que $consumoSobDemanda = 1$ hora de CPU (linhas 18 e 21).

Após percorrer todo o período de avaliação (i.e., um ano) a heurística RF estima a utilidade obtida com o plano sob avaliação (linha 23). Considerando que 8.640 horas de CPU foram consumidas no *mercado sob demanda* com um custo por hora de CPU de $c^{on}_{small} = \$0,08$ e que 89.040 horas foram consumidas no *mercado de reservas* com $c^{v}_{small} = \$0,03$, com taxa de reserva de $f_{small} = \$100$ e que o provedor de SaaS não pagou nenhuma penalidade a seus clientes, estima-se $\alpha(D) = \$4.762,4$. Considerando que 15 clientes de SaaS contrataram o provedor de SaaS, sendo responsáveis pela carga de trabalho acima, e que a mensalidade I_j de cada cliente é de $\$400$ a receita total estimada é de $\iota(D) = \$6000$. Uma vez que não ocorreram violações de SLA a estimativa de utilidade é $\upsilon(D) = 6.000 - 4.762,4 - 0,0 = \$1237,6$. Após estimar a utilidade dos 32 planos de reserva a heurística RF seleciona o plano com maior estimativa de utilidade (linha 25).

Capítulo 5

Modelo de Simulação

Neste capítulo são apresentados os componentes e características do modelo de simulação usado para avaliar as heurísticas de planejamento de capacidade propostas no Capítulo 4 considerando o modelo de negócio definido no Capítulo 3. Em especial são discutidos os dois principais componentes deste modelo: (i) o agente de planejamento de capacidade; (ii) o agente de provisão dinâmica de recursos. O modelo aqui apresentado foi utilizado para obtenção dos resultados apresentados no Capítulo 6.

5.1 Visão Geral do Sistema

O cenário base considerado na pesquisa envolve um provedor de SaaS que oferece sua aplicação a um grupo de clientes de SaaS. O conjunto de aplicações SaaS existentes atualmente é bastante amplo e envolve, por exemplo: aplicações de *Customer Relationship Manager* (CRM)[1,2], aplicações de comércio eletrônico[3,4], aplicações para realização de pesquisas através de questionários[5], aplicações de gerência de gastos[6] e de tarefas[7], aplicações para análise de dados de simulação[8] e aplicações para transferência de arquivos[9]. Cada uma des-

[1] http://crm.dynamics.com
[2] http://www.salesforce.com/br/
[3] http://www.bigcommerce.com/
[4] http://www.volusion.com/
[5] http://www.surveymonkey.com
[6] http://www.coupa.com/solutions/
[7] http://www.deskaway.com
[8] http://dssm.unipa.it/R-php/
[9] https://www.globusonline.org/

tas aplicações possuem características diferentes umas das outras (e.g., taxas de chegada de requisições, demandas de processamento) o que torna complexa a tarefa de se analisar um planejamento de capacidade para todo o conjunto de aplicações SaaS.

Dentro desta ampla variedade de aplicações possíveis de serem estudadas, foi escolhida a aplicação de comércio eletrônico. Tal escolha se deu: (i) devido à grande quantidade de estudos a respeito da caracterização de aplicações de comércio eletrônico [Arlitt, Krishnamurthy e Rolia 2001] [Menascé et al. 2003] [Menascé et al. 2000] [Zhang, Cherkasova e Smirni 2007]; (ii) devido ao fato da empresa **BigCommerce** ter assumido um papel considerável no mundo de aplicações SaaS, apresentando mais de 20.000 clientes[10] e arrecadações de recursos cada vez maiores para seu negócio [Hawkins 2011], o que a torna uma das grandes empresas de aplicações SaaS na atualidade e um exemplo de estudo interessante.

A aplicação considerada nesta dissertação é uma aplicação do tipo *e-service*. Tal tipo de aplicação de comércio eletrônico possui demandas sobre três grandes classes de recursos: (i) recursos de processamento e memória; (ii) recursos de armazenamento de dados; (iii) recursos de transferência de dados. Aplicações deste tipo são comumente organizadas em um modelo de N-camadas, tipicamente três camadas, onde as duas primeiras camadas lidam essencialmente com as demandas de processamento e a última camada é responsável pelas demandas de armazenamento e persistência de dados. Destas três classes de recursos, os provedores atuais de IaaS oferecem possibilidade de reserva para recursos de processamento. Logo, o grande foco do planejamento de capacidade avaliado nesta dissertação envolve a elaboração de um plano de reservas de recursos de processamento. Todavia, as demandas relacionadas à transferência e ao armazenamento de dados são consideradas no modelo de utilidade e também são avaliadas no estudo.

Por questões de simplicidade foi considerado que a aplicação em questão possui uma única camada. Esta única camada pode representar, teoricamente, a agregação de múltiplas camadas em uma abstração de uma única camada ou, ainda, uma única camada propriamente. Ao considerar múltiplas camadas agregadas em uma única camada, pode-se considerar que se está realizando o planejamento de capacidade para a demanda total da aplicação. Reservados os recursos durante o planejamento, um sistema de provisão dinâmica de recursos (DPS) [Ranjan et al. 2002] [Urgaonkar et al. 2008] [Bi et al. 2010] pode decidir quantos

[10] http://www.bigcommerce.com/about.php

recursos são necessários em um intervalo curto de tempo e, em sequência, alocar tais recursos às camadas da aplicação considerando aspectos como lucro ou satisfação do usuário [Chen et al. 2011] ou de acordo com o desempenho de cada camada ao utilizar um devido recurso [Lee, Chun e Katz 2011]. Ao considerar uma única camada de fato, pode-se realizar o planejamento para cada uma das três camadas de forma independente e, quando da execução da aplicação, um sistema de provisão dinâmica pode decidir quantos recursos são necessários para cada camada em questão e requisitar os recursos para cada camada de forma independente considerando as reservas realizadas para cada camada.

A camada de recursos sobre a qual a aplicação é executada, pode ser composta por um conjunto de recursos heterogêneos adquiridos junto ao provedor de IaaS. Tipicamente, os recursos de processamento apresentam esta heterogeneidade. A heterogeneidade destes recursos pode ocorrer devido às diferentes quantidades de núcleos de processamento disponíveis em cada recurso, bem como devido às diferentes velocidades de processamento oferecidas por cada núcleo. Nesta dissertação a heterogeneidade considerada foca nas diferentes quantidades de núcleos presentes nos recursos de processamento. A aplicação é considerada como sendo horizontalmente escalável, uma vez que é possível alterar sua capacidade ao se adicionar e remover recursos de processamento de acordo com a demanda exigida pela carga de trabalho. O uso de recursos heterogêneos na infraestrutura depende das decisões tomadas pelo sistema de provisão dinâmica de recursos e das reservas realizadas pelas heurísticas de planejamento de capacidade.

Em um dado instante de tempo t existe um conjunto de recursos de processamento que estão disponíveis para execução da aplicação. Um balanceador de carga recebe as requisições dos clientes de SaaS e, para cada requisição, decide qual o recurso que será utilizado para atendimento de cada requisição (vide Figura 5.1). A política adotada por este balanceador de carga obedece uma ordem circular (do inglês, *round-robin*) e considera a capacidade de processamento de cada recurso (quantidade de núcleos de processamento disponíveis). Um recurso que possuir dois núcleos de processamento receberá uma quantidade de requisições igual a duas vezes o número de requisições encaminhadas para um recurso que possuir apenas um núcleo de processamento.

Assume-se que cada recurso de processamento possui instalada uma versão de algum

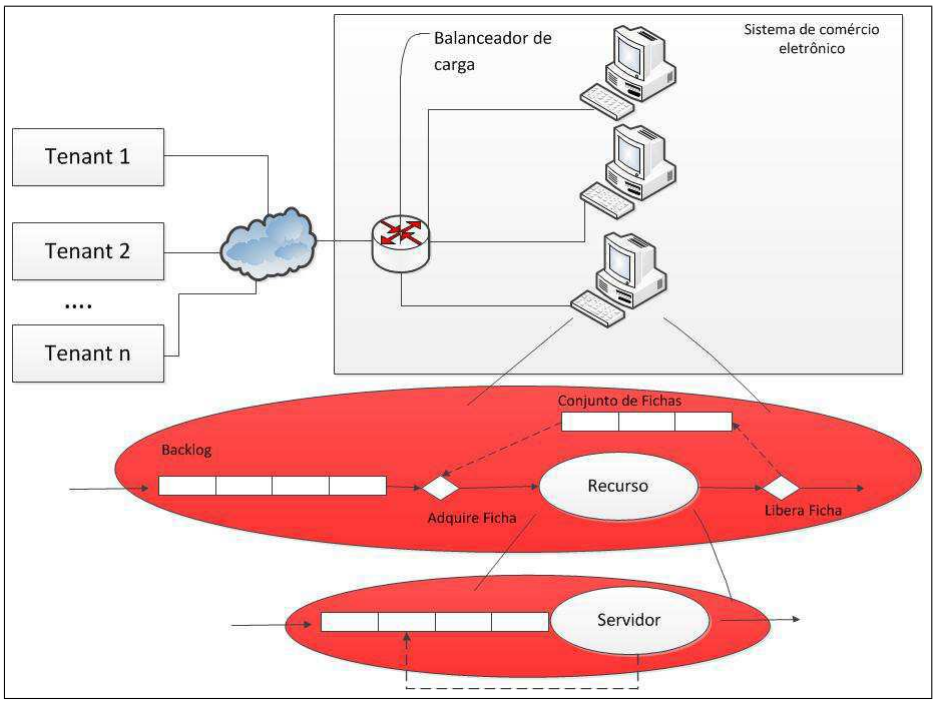


Figura 5.1: Modelo do Sistema: Visão geral sobre filas e recursos de processamento

servidor disponível no mercado, por exemplo, Apache HTTP Server[11], JBoss[12] ou Apache Tomcat[13]. Tais servidores executam múltiplos processos ou múltiplas linhas de execução (do inglês, *threads*) que processam as requisições recebidas. Cada processo/linha de requisição realiza a computação de uma única requisição por vez. Cada um destes servidores permite a configuração de um número máximo de processos/linhas de execução e, assim, tem-se um nível máximo de concorrência atingido pelo servidor. Seguindo esta lógica, cada recurso de processamento apresenta um conjunto de m fichas que representam os processos/linhas de execução disponíveis (vide Figura 5.1). Uma requisição precisa adquirir uma ficha para ser processada e esta fica indisponível até que a requisição libere o recurso. Requisições que chegam em um determinado recurso quando todas as fichas estão indisponíveis esperam em uma fila chamada de *backlog* até que fichas sejam liberadas. A fila de *backlog* tem uma capacidade limitada e obedece uma disciplina de "primeiro a chegar, primeiro a ser servido" (do inglês, *first-come, first-served* – FCFS). Requisições que chegam ao recurso quando a fila de *backlog* está cheia são rejeitadas. Este modelo segue a proposição de [Menasce et al.

[11] http://httpd.apache.org/

[12] http://www.jboss.org/

[13] http://tomcat.apache.org/

2004].

Quando uma requisição está de posse de uma ficha ela é recebida em um sistema de filas que realiza o processamento da mesma. Na prática este sistema seria uma rede de filas composta por recursos de hardware e software. Entretanto, não está no escopo deste trabalho identificar gargalos e caracterizações deste nível de execução. Busca-se, apenas, capturar a influência que a existência de múltiplas requisições possui sobre os tempos de resposta percebidos pelo cliente de SaaS. O processamento que é realizado obedece uma disciplina de compartilhamento de CPU (do inglês, *Processor Sharing* – PS), pois esta é a disciplina mais próxima da disciplina aplicada pelos sistemas operacionais. Uma determinada requisição possui o direito de utilizar a CPU por um determinado *quantum* Δ, tipicamente bem pequeno, e, em seguida, a CPU é realocada para uma outra requisição que esteja esperando para utilizar o recurso. Desta forma, todas as requisições progridem simultaneamente, porém uma requisição que dura em média z *quantuns* para ser processada quando está sozinha na fila passa a ser processada em $z \times n$ *quantuns* quando n requisições iguais a ela estão disputando o acesso ao recurso.

Além da demanda de processamento, cada requisição recebida pelo balanceador de carga apresenta uma demanda de transferência de dados. O atendimento desta demanda por parte do provedor de IaaS acontece sem a necessidade de uma negociação ou escolha prévia de tipos de recursos a serem utilizados, como acontece com os recursos de processamento. Além disso, cada cliente de SaaS apresenta uma demanda de armazenamento relacionada à hospedagem de sua página e ao cadastro do conjunto de produtos e usuários finais que estão relacionados ao cliente de SaaS. O atendimento desta demanda também acontece sem a necessidade da escolha de tipos de recursos a serem utilizados e do estabelecimento de contratos de reserva.

Para o tipo de aplicação que está sendo considerada, a taxa de chegada de requisições em certo instante de tempo pode ser extremamente variável, conforme demonstram os estudos sobre cargas Web [Barford e Crovella 1998] [Crovella e Bestavros 1996] e comércio eletrônico [Arlitt, Krishnamurthy e Rolia 2001] [Menascé et al. 2003]. Para lidar com essas variações de carga de trabalho em um curto prazo (e.g., uma hora) existe um sistema de provisão dinâmica de recursos (DPS) [Ranjan et al. 2002] [Urgaonkar et al. 2008] [Lee et al. 2010] [Kwok e Mohindra 2008], já mencionado anteriormente. O DPS é responsável

por decidir sobre a quantidade de recursos que deve ser requisitada junto ao provedor de IaaS para atender a carga de trabalho em um próximo intervalo curto de tempo. Este processo, conforme apontado na Figura 5.2, se dá através de um monitoramento da aplicação em execução de modo a coletar dados que sirvam como base para a tomada de decisão do agente de provisão dinâmica. Uma vez decidida a quantidade de recursos necessária para o próximo intervalo de tempo, estes recursos são requisitados junto ao provedor de IaaS e, posteriormente, cedidos à aplicação para que a demanda da aplicação possa ser atendida.

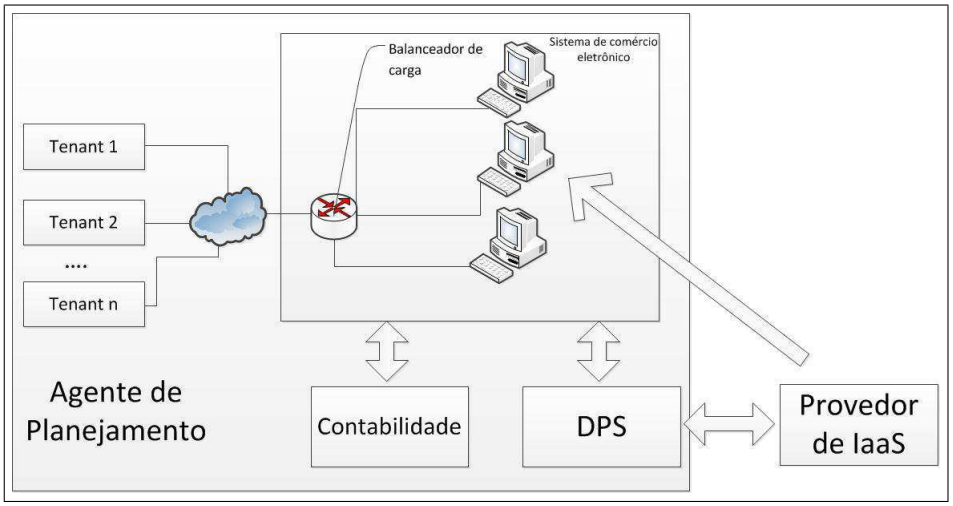


Figura 5.2: Modelo do Sistema: Visão geral sobre DPS, Agente de Planejamento e Sistema de Contabilidade

Um outro componente importante neste sistema, vide Figura 5.2, é o componente de Contabilidade. Este componente é responsável por, também, monitorar a aplicação em execução de modo a coletar a quantidade de recursos consumidos por cada cliente do provedor de SaaS, bem como coletar a quantidade de requisições submetidas e processadas com sucesso. Com base nos dados coletados pelo componente de Contabilidade pode-se, a cada período de uso n junto ao provedor de SaaS, utilizar o modelo de negócio proposto no Capítulo 3 para computar os valores que devem ser pagos pelos clientes de SaaS. Estes valores podem conter a cobrança de taxas extras aos clientes de SaaS quando necessário.

Um último componente deste sistema é o agente de planejamento de capacidade. Como apresentado na Figura 5.2 este agente é um componente mais externo que abrange todo o sistema, tendo conhecimento sobre seu funcionamento, de modo a elaborar planos de reserva

de recursos de processamento a partir de heurísticas de planejamento de capacidade.

5.1.1 Componente de Provisão Dinâmica de Recursos

Um sistema de provisão dinâmica de recursos é um sistema que modifica a quantidade de recursos em uma infraestrutura para se adequar à carga de trabalho corrente. Estes sistemas funcionam segundo um laço de controle com realimentação que atua periodicamente sobre a aplicação para coletar dados e, assim, realizar uma tomada de decisão [Ranjan et al. 2002] [Urgaonkar et al. 2008] [Lee et al. 2010] [Kwok e Mohindra 2008] [Zhu et al. 2011] [Bi et al. 2010] [Chi, Qian e Lu 2011].

Alguns sistemas de provisão dinâmica foram inicialmente verificados [Ranjan et al. 2002] [Lee et al. 2010], porém o uso dos recursos que estava sendo realizado por tais sistemas estava prejudicando a avaliação da qualidade do planejamento de capacidade realizado devido às violações de SLA ocasionadas pelo DPS. Por conta disto foi desenvolvido um DPS, aqui denominado de *DPS Oráculo*, que conhece a carga de trabalho que será submetida à aplicação durante a próxima hora e computa a quantidade de recursos necessária para que nenhuma requisição seja rejeitada. Este cálculo se baseia na ideia de avaliar o instante de chegada de cada requisição, bem como sua demanda, para determinar por quantos intervalos de tempo a requisição estaria sendo processada. Realizada esta avaliação com todas as requisições da próxima hora, e considerando que filas podem ser formadas nos recursos sem violar o SLA, o *DPS Oráculo* avalia a maior quantidade de recursos que deveriam ser utilizados em paralelo para processamento das requisições.

Quando a quantidade de recursos calculada pelo DPS for superior à quantidade de recursos atualmente ativa, acompanhando o crescimento na demanda da aplicação, novos recursos são adquiridos junto ao provedor de IaaS. Quando este valor for inferior à quantidade de recursos atualmente ativa, acompanhando a queda na demanda da aplicação, recursos são liberados junto ao provedor de IaaS. Quando um recurso é liberado, o balanceador de carga deve parar de encaminhar requisições para este recurso.

Ao requisitar novos recursos junto ao provedor de IaaS o DPS verifica a disponibilidade de recursos reservados para, prioritariamente, requisitar este tipo de recurso. Uma vez que todos os recursos reservados já estiverem em uso, o DPS passa a requisitar recursos no *mercado sob demanda*. Tal comportamento se baseia no fato de que o consumo de um recurso

reservado tende a ser mais barato que o consumo de um recurso adquirido no *mercado sob demanda*. A requisição de recursos no *mercado sob demanda* pode levar a uma negação de serviço por parte do provedor de IaaS uma vez que este mercado não oferece garantias de que o pedido por um determinado conjunto de recursos deve sempre ser atendido. O modelo de simulação considera que existe um risco de negação de serviço que indica que para cada recurso requisitado ao provedor de IaaS no *mercado sob demanda* existe uma probabilidade p de que este recurso não seja obtido.

5.1.2 Componente de Planejamento de Capacidade

Um agente de planejamento de capacidade no contexto de um provedor de SaaS que monta sua infraestrutura sobre um provedor de IaaS é responsável por determinar a quantidade, e os tipos de recursos, essencialmente recursos de processamento, que devem ser reservados para a execução de um longo período futuro (e.g., 1 ano) de uma aplicação. Por exemplo, supondo um agente de planejamento que elabore um plano de reservas junto ao serviço **Amazon EC2**, o mesmo pode determinar que para executar a carga de trabalho referente a um ano de certa aplicação de comércio eletrônico é viável que sejam reservados 10 recursos do tipo *large* e 2 recursos do tipo *small* da família *standard*.

Para elaborar um plano de reservas o agente de planejamento precisa conhecer o funcionamento da aplicação em questão, por isso este componente é representado na Figura 5.2 como um componente que engloba todos os componentes já citados do sistema. Uma vez que o sistema é conhecido, o agente de planejamento faz uso de uma caracterização da carga de trabalho da aplicação para, assim, avaliar o comportamento do sistema diante de diferentes configurações de reserva de recursos. Esta caracterização da carga da aplicação pode se dar em função de dados históricos de execução da aplicação ou, caso estes dados não existam ou não sejam acessíveis, em função de estimativas da carga de trabalho futura. Em ambos os casos o agente de planejamento está lidando diretamente com um erro em seu planejamento uma vez que tanto os dados históricos como a predição da carga podem não corresponder exatamente à carga de trabalho que será submetida à aplicação.

A avaliação realizada pelo agente de planejamento para estimar a qualidade de planos de reservas se dá com base no modelo de utilidade desenvolvido no Capítulo 3. O agente de planejamento de capacidade faz uso de estratégias, ou heurísticas, para avaliar os dados

da carga de trabalho e, assim, elaborar os planos de reserva. Diferentes heurísticas podem ser utilizadas conduzindo a diferentes planos de reservas e, consequentemente, diferentes impactos no negócio do provedor de SaaS. As heurísticas de planejamento implementadas no simulador foram apresentadas no Capítulo 4.

5.1.3 Carga de trabalho

A carga de trabalho submetida à aplicação, e por consequência à infraestrutura do provedor de SaaS, é proveniente dos vários clientes de SaaS (do inglês, *tenants*) que contrataram serviços junto ao provedor. A carga de cada um destes clientes de SaaS representa o agregado da carga gerada por vários usuários finais do comércio eletrônico que estão acessando às páginas do cliente de SaaS em busca de realizar a compra de produtos. Por exemplo, se considerarmos que a empresa **Americanas S.A.**[14] deseja contratar um provedor de SaaS para hospedar seu site de comércio eletrônico, a carga de trabalho percebida pelo provedor de SaaS está relacionada aos acessos dos usuários finais que visualizam a página da **Americanas.com** e que realizam compras no referido site.

Cada um dos clientes de SaaS pode estar relacionado a um plano diferente e apresentar diferentes padrões de comportamento quanto à carga de trabalho submetida ao provedor de SaaS. As diferenças nos padrões podem estar relacionadas a diferenças na quantidade de requisições submetidas, a diferenças nas demandas de processamento, armazenamento e transferência, bem como a diferenças no comportamento da carga total ao longo do tempo. Para executar toda esta carga de trabalho recebida pelo provedor de SaaS, uma única infraestrutura é utilizada.

5.2 Execução das Simulações

O planejamento de capacidade pode ser avaliado a partir do uso de modelos analíticos ou de modelos de simulação [Almeida 2002]. O modelo de simulação foi escolhido com o intuito de se capturar com mais detalhes aspectos do funcionamento do sistema de comércio eletrônico estudado. Atualmente existem alguns simuladores para ambientes de Computação na Nuvem disponíveis, tais como o CloudSim [Calheiros et al. 2011]. No entanto, ao avaliar

[14]http://www.americanas.com.br

o uso desta ferramenta para os propósitos da pesquisa alguns problemas foram verificados tais como:

- A ferramenta visa realizar a simulação de ambientes de Computação na Nuvem de forma muita mais detalhista do que o foco desta dissertação permitindo configurações de políticas de consumo de energia, topologias de rede, alocação de máquinas virtuais a recursos físicos, etc. Como consequência teria-se um esforço para caracterização de diversos tópicos que não eram o foco do estado atual da pesquisa;

- Esforço de programação para permitir que o simulador utilizasse a carga de trabalho sintética que foi utilizada nas simulações;

- Esforço de programação para permitir o uso de recursos oriundos dos mercados sob demanda e de reservas simultaneamente. Além disto, a contabilização deste uso para o cálculo de utilidade, conforme o modelo apresentado no Capítulo 3, também exigia uma nova programação.

Considerando os problemas acima e o esforço de programação que seria necessário para desenvolver e testar o novo código necessário, bem como para caracterizar os demais aspectos que não são o foco da pesquisa, optou-se por desenvolver um modelo de simulação baseado em eventos discretos. Este modelo é apresentado no corrente capítulo desta dissertação. O simulador desenvolvido encontra-se disponível através da URL `http://code.google.com/p/saasim-david/`.

De modo geral, podemos pensar no funcionamento da simulação de acordo com a Figura 5.3. O objetivo das simulações é realizar um planejamento de capacidade e, em seguida, avaliar a utilidade que poderia ser obtida por um provedor de SaaS caso tal planejamento fosse utilizado na execução da carga real a qual o provedor será submetido.

Como pode ser percebido na Figura 5.3, a simulação considera uma carga de trabalho sintética em sua entrada. Uma vez que o acesso a registros de execução de aplicações SaaS de comércio eletrônico apresentam restrições de privacidade, optou-se por buscar um gerador de carga de comércio eletrônico. Optando-se pelo uso de um gerador de carga de trabalho surgem duas opções: (i) o uso de um gerador que forneça a carga de cada usuário final do comércio eletrônico; (ii) o uso de um gerador que forneça a carga agregada percebida pela

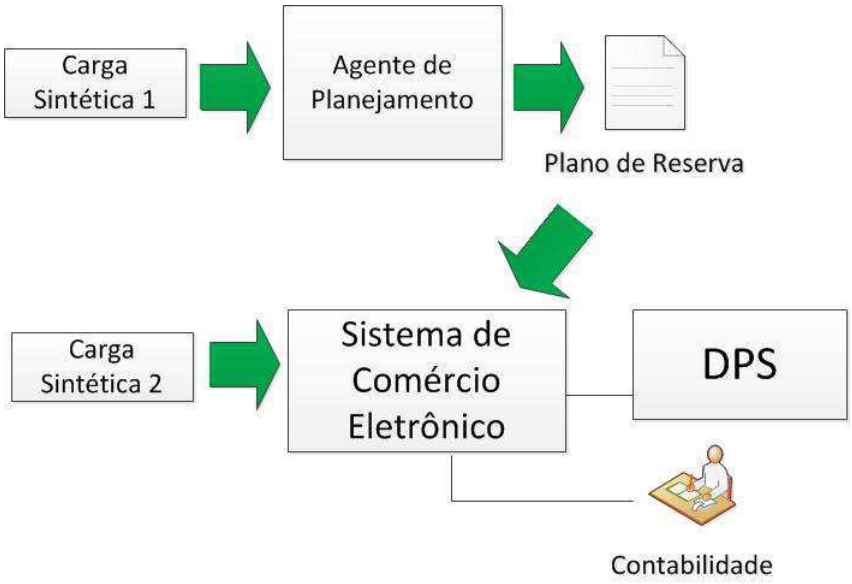


Figura 5.3: Modelo de Simulação

infraestrutura de TI responsável pela execução da aplicação. Quando se busca gerar a carga individual de cada usuário final torna-se difícil controlar o tráfego agregado que é recebido nos recursos uma vez que o envio de uma nova requisição só acontece quando a requisição anterior for satisfeita [Kant, Tewari e Iyer 2001]. Dadas tais características, buscou-se um gerador de cargas que fornecesse a carga de trabalho agregada submetida à infraestrutura de TI.

A ferramenta Geist (do inglês, *Generator of E-Commerce and Internet Server Traffic*) [Kant, Tewari e Iyer 2001] foi proposta em 2002 com o objetivo de permitir a geração da carga de trabalho agregada submetida a uma infraestrutura de TI. A ferramenta foi desenvolvida buscando considerar as principais características dos geradores de carga disponíveis na época. A ferramenta produz um arquivo de rastros (do inglês, *trace*) que fornece, dentre outras informações:

- O instante em que uma requisição chega a infraestrutura de TI para ser processada;
- A quantidade de dados a serem transferidos como entrada (i.e., dados transferidos para o provedor de SaaS) e a quantidade de dados a serem transferidos como saída (i.e., dados transferidos do provedor de SaaS para o cliente).

Quanto à caracterização das demandas de processamento das requisições foram considerados os valores apresentados em [Ranjan et al. 2002] e [Arlitt, Krishnamurthy e Rolia 2001]

uma vez que tais estudos fornecem valores de demanda para as diferentes camadas que compõe uma aplicação de *e-service*, bem como uma distribuição percentual das requisições que apresentam tais valores. Tais valores foram confrontados com valores obtidos a partir de experimentos realizados junto aos recursos de processamento fornecidos pelo serviço **Amazon EC2**.

A simulação ocorre em dois momentos principais (vide Figura 5.3): (i) um momento de planejamento da infraestrutura de TI para um intervalo D de avaliação; (ii) um momento de execução da aplicação durante um intervalo D. O planejamento de capacidade é realizado fazendo uso de uma heurística de planejamento, como as propostas no Capítulo 4, que realiza uma análise sobre a carga de trabalho sintética obtida junto à ferramenta Geist. Feito o planejamento, um arquivo de configuração que determina o plano de reservas firmado junto ao provedor de IaaS é obtido. É importante destacar que a carga de trabalho avaliada pela heurística de planejamento possui uma diferença em relação à carga de trabalho que de fato será submetida à aplicação. Esta diferença, aqui denominada de erro de predição, se deve ao fato de que, no momento do planejamento, o agente possui apenas uma predição, estimativa, da carga de trabalho futura da aplicação. O erro considerado neste trabalho está relacionado com a quantidade de clientes de SaaS que utilizam a aplicação ofertado pelo provedor de SaaS. Por exemplo, suponha que um total de 100 clientes de SaaS submetam requisições ao provedor de SaaS durante o período de 1 ano. Um erro de predição da carga de trabalho de 10% indica que a carga de trabalho percebida no momento do planejamento de capacidade representa um total de 110 clientes de SaaS submetendo requisições ao provedor de SaaS.

De posse do plano de reservas inicia-se a etapa de execução da carga de trabalho. A carga de trabalho é direcionada ao balanceador de carga da camada única existente na simulação, de forma similar ao apresentado na Figura 5.1. O funcionamento do balanceador de carga, bem como a forma como as requisições são processadas nos recursos, obedecendo o sistema de filas, que compõe a infraestrutura de TI segue os conceitos apresentados na Seção 5.1.

Sempre que uma requisição é processada, seja ela processada dentro dos limiares do SLA ou não, o sistema de Contabilidade é responsável por registrar o consumo de recursos realizados pelo cliente de SaaS, bem como se o SLA foi ou não violado. Ao final de cada período n de simulação é realizada a contabilidade dos recursos consumidos por cada cliente do provedor de SaaS, das receitas geradas por cada cliente e das penalidades a serem

pagas pelo de provedor de SaaS. Com base nestes valores a utilidade do provedor de SaaS é computada seguindo a Equação 3.10.

Durante o desenvolvimento da ferramenta de simulação algumas estratégias de Verificação e Validação ($V\&V$) foram utilizadas [Sargent 2005]: (i) uso de gráficos operacionais para avaliação, por exemplo, de custos, receitas e total de horas requisitadas nos mercados sob demanda e de reservas; (ii) comparação da quantidade de máquinas reservadas e do total de horas requisitadas junto ao provedor de IaaS, em cenários de carga controlada, com dados analíticos da carga de trabalho; (iii) testes degenerados alterando o limite máximo de recursos que podem ser obtidos nos mercados de reservas e sob demanda frente uma carga controlada para avaliação de cenários com contenção; (iv) testes de condição extrema, considerando uma carga controlada, que alteraram os valores dos preços dos mercados de reservas e sob demanda de modo a verificar a alternância de escolhas entre reservar ou não recursos; (v) validade de eventos para verificar a correta execução de eventos da simulação como chegada de requisições, inicialização de um recurso, término de execução de requisições, etc.; (vi) validade interna através da análise de custos, receitas e lucros obtidos em cenários com cargas similares; (vii) uso de rastros para acompanhar o comportamento dos principais componentes do sistema em cenários de carga controlada; (viii) uso de testes automáticos durante o desenvolvimento da ferramenta de simulação atrelados ao uso de ferramentas de cobertura de testes.

Capítulo 6

Apresentação e Análise dos Resultados

Este capítulo tem por objetivo discutir como foi conduzida a avaliação das heurísticas de planejamento de capacidade apresentadas no Capítulo 4 e discutir os resultados obtidos. A avaliação das heurísticas propostas foi realizada através da condução de experimentos considerando o modelo de simulação apresentado no Capítulo 5.

6.1 Heurísticas base para comparação

De modo a avaliar os benefícios gerados pelas heurísticas propostas, em relação à utilidade (lucro) do provedor de SaaS, as mesmas serão avaliadas em comparação a três estratégias de referência:

- Estratégia que não realiza reserva de recursos – ON;
- Estratégia que realiza superprovisionamento da infraestrutura de TI – SUPER;
- Estratégia de planejamento ótimo.

A estratégia ON é uma estratégia aversa ao uso do *mercado de reservas*, ou seja, esta estratégia confia plenamente no *mercado sob demanda*. A estratégia se baseia apenas no uso de recursos adquiridos no *mercado sob demanda* para execução da aplicação e, portanto, não elabora um plano de reservas de recursos. A heurística ON também não considera que o provedor de IaaS pode não atender por completo suas requisições por recursos computacionais,

o que pode, na prática, ocasionar a negação de serviço aos clientes de SaaS e a cobrança de penalidades ao provedor de SaaS.

A estratégia SUPER é uma estratégia que realiza um superprovisionamento da infraestrutura de TI. Este tipo de heurística é comumente discutido na literatura [Armbrust et al. 2009] [Sauve et al. 2006] [Chapman et al. 2010]. Segundo este conceito, uma infraestrutura de TI é planejada considerando a maior quantidade de recursos em paralelo que é necessária para atender a demanda da aplicação dentro do período a ser planejado. A partir disto, a estratégia SUPER avalia um conjunto de dados históricos, ou uma estimativa destes dados, que caracteriza a carga de trabalho da aplicação e calcula a maior quantidade de recursos que seria necessária para atender a demanda. A estratégia SUPER considera que uma infraestrutura superprovida apresenta uma taxa de utilização de seus recursos em torno de 20% [Armbrust et al. 2009]. Como consequência, a heurística reserva 20% do total de máquinas superestimado. A análise dos dados de simulação demonstraram que 20% do total de máquinas superestimado representa um valor próximo do total de recursos reservados por RF e UT (vide Seção 6.5.4). De modo a avaliar qual o impacto produzido por um superprovisionamento no tipo dos recursos buscou-se manter esta quantidade de 20%, porém utilizando recursos com elevado poder computacional. Mais especificamente, foram considerados recursos da classe *standard* do tipo *large* ofertados pelo serviço **Amazon EC2**.

A estratégia de planejamento ótimo é uma estratégia que conhece a carga de trabalho futura que será submetida a aplicação, bem como a quantidade de recursos requisitados pelo DPS, e avalia todos os possíveis planos de reserva para esta carga. Estes possíveis planos contemplam a reserva de quantidades de recursos que variam da menor quantidade de recursos consumida pelo DPS até a maior quantidade consumida pelo DPS. Para cada um destes possíveis planos de reserva a estratégia calcula a utilidade do provedor de SaaS e, ao final, escolhe o plano de reserva que fornece a maior utilidade.

6.2 Métrica de Avaliação

Avaliar a utilidade (lucro) obtida por um provedor de SaaS ao executar uma aplicação durante um intervalo D fornece um indicativo da contribuição para o crescimento do negócio. Porém, de modo a quantificar as relações entre as heurísticas propostas em diferentes cenários de

avaliação, foi elaborada uma métrica aqui denominada de **ganho** ($v_A(D)$**,** $v_B(D)$**)**. Esta métrica busca relacionar as utilidades obtidas pelo provedor de SaaS utilizando as heurísticas **A** e **B** e, assim, determinar, percentualmente, quanto a utilidade obtida pela heurística **A** foi maior que a utilidade obtida pela heurística **B**. Desta forma, a métrica **ganho** ($v_A(D)$**,** $v_B(D)$**)** pode ser definida segundo a Equação 6.1:

$$ganho(v_A(D), v_B(D)) = 100 * (v_A(D) - v_B(D))/|v_B(D)| \tag{6.1}$$

A métrica acima considera o módulo da utilidade da heurística **B** uma vez que, de acordo com o cenário de avaliação considerado, pode ocorrer que as heurísticas apresentem valores de utilidade negativos.

6.3 Projeto Experimental

De modo a avaliar as heurísticas propostas no Capítulo 4 e validar as hipóteses elaboradas foi realizado um levantamento de fatores que, a princípio, deveriam influenciar a qualidade do planejamento realizado. Tais fatores foram:

- O número de clientes de SaaS que estão submetendo requisições ao provedor de SaaS;
- O erro de predição da carga de trabalho no momento do planejamento de capacidade;
- O risco de negação de serviço por parte do provedor de IaaS quando da requisição de recursos no *mercado sob demanda*.

A partir deste conjunto de fatores realizou-se um projeto experimental do tipo fatorial $2^k r$ de modo a capturar se os fatores acima eram de fato relevantes para o estudo do problema em questão, bem como para capturar tendências no comportamento das heurísticas UT e RF em diferentes cenários de avaliação. Os níveis utilizados para o projeto experimental foram:

- **Número de clientes de SaaS**: 10, 100. Os clientes de SaaS foram distribuídos uniformemente em três planos distintos (explicitados na Seção 6.4) uma vez que não se obteve dados claros sobre a composição da carteira de clientes de um provedor de SaaS;

- **Erro de predição da carga de trabalho**: 15%, 40%. Estes valores foram estabelecidos com base em uma análise de variação do mercado de comércio eletrônico apresentada em [UEMA e LAZZARI 2008]. A partir desta análise optou-se por avaliar um projeto experimental com superestimativas da carga de trabalho (15% e 40%) e outro projeto experimental com subestimativas da carga de trabalho (-15% e -40%) uma vez que se entende que superestimativas e subestimativas conjuntamente não afetam a variável de resposta de forma linear. Um erro de predição da carga de trabalho está relacionado com um erro na predição da quantidade de clientes de SaaS que estarão utilizando os serviços oferecidos pelo provedor de SaaS. Logo, um erro de 40% implica que durante o planejamento de capacidade um total de 40% a mais de clientes de SaaS foram considerados submetendo requisições ao provedor de SaaS;

- **Risco de negação de serviço no mercado sob demanda**: 1% e 15%. Estes níveis foram utilizados de modo a capturar a presença do risco de negação de serviço, porém estimando que os provedores atuais de IaaS não devem possuir riscos muito maiores que tais valores dado que isto representaria uma qualidade de serviço muito baixa oferecida pelo provedor de IaaS.

Cada um dos cenários acima foi repetido três vezes no projeto experimental, ou seja, $r = 3$. Cada replicação representou o uso de uma carga de trabalho diferente obtida junto à ferramenta Geist. Através da análise de variância e da análise do coeficiente R-quadrado foi verificado que os fatores **número de clientes de SaaS** e **risco de negação de serviço** foram os fatores significativos. Além disso, pôde-se perceber que, apesar do fator **erro de predição da carga de trabalho** não ter sido apontado como um dos fatores significativos, o comportamento das heurísticas UT e RF diante de superestimativas e de subestimativas da carga de trabalho foi diferenciado. Logo, optou-se por considerar o erro de predição da carga de trabalho na realização dos experimentos abrangendo um valor de erro para representar uma superestimativa e um valor de erro para representar uma subestimativa. Uma análise preliminar do fator **risco de negação de serviço** é apresentada na Seção 7.2 que discorre sobre Trabalhos Futuros.

6.4 Instância do Modelo

Para descrever os cenários utilizados na avaliação das heurísticas propostas, primeiramente é interessante especificar quais as variáveis existentes nos experimentos. As variáveis dependentes, ou seja, aquelas que são capturadas como resultados dos experimentos são: (i) a utilidade (lucro) obtida pelo provedor de SaaS ao executar uma aplicação; (ii) a receita obtida pelo provedor de SaaS; (iii) os custos relacionados ao uso de recursos de processamento pelo provedor de SaaS no *mercado sob demanda* e no *mercado de reservas*; (iv) os custos obtidos pelo provedor de SaaS devido às transferências e armazenamento de dados. As variáveis independentes dos experimentos são as variáveis que caracterizam o provedor de IaaS, o provedor de SaaS, a carga de trabalho e configurações adicionais da simulação.

As variáveis independentes que caracterizam o *mercado sob demanda* e o *mercado de reservas*, bem como os custos de transferência e armazenamento junto ao provedor de IaaS são apresentados na Tabela 6.1. Tais dados representam os valores praticados pelo serviço **Amazon EC2**, um dos maiores provedores de infraestrutura em operação atualmente, no ano de 2011. Considera-se que uma vez que um recurso de processamento é adquirido junto à **Amazon** o mesmo necessita de um intervalo de tempo para que a máquina virtual seja inicializada, bem como para que o serviço Web que é responsável por atender as requisições dos clientes de SaaS seja inicializado. Um intervalo de 5 minutos [Wu, Garg e Buyya 2011] foi utilizado nas simulações para este propósito. O fator **risco de negação de serviço** utilizado foi de 10%. Acredita-se que este valor esteja dentro do que é praticado no mercado atualmente de modo que a imagem da empresa não seja prejudicada junto aos seus clientes.

Para a caracterização das variáveis independentes relacionadas ao provedor de SaaS foram escolhidos os dados de mercado da empresa **BigCommerce**. Esta empresa publica que atualmente conta com mais de 20.000 clientes, o que a torna uma dos grandes provedores de SaaS. A Tabela 6.2 apresenta a caracterização dos três planos ofertados pelo provedor de SaaS que foram considerados para os experimentos, segundo valores praticados no ano de 2011. Optou-se por escolher planos distintos de modo que diferentes perfis de clientes de SaaS utilizassem a aplicação e, assim, a carga de trabalho submetida ao sistema fosse mais realista. Uma margem de contribuição de 30% foi escolhida para cada plano. Esta escolha teve como base uma pesquisa sobre margens de contribuição e taxas de lucro praticadas por

Tabela 6.1: Caracterização dos parâmetros relativos ao provedor de IaaS

Variável	**Valor**
Provedor de IaaS	**Amazon EC2**
Custo de uso de um recurso de processamento sob demanda	*Small* – $0,085$\$/hora; *Large* – $0,34$\$/hora; *XLarge* – $0,68$\$/hora
Custo de uso de um recurso de processamento reservados	*Small* – $0,03$\$/hora; *Large* – $0,12$\$/hora; *XLarge* – $0,24$\$/hora
Custo de reserva de um recurso de processamento por 1 ano	*Small* – $\$227,50$; *Large* – $\$910$ *XLarge* – $\$1.820$
Custo de monitoramento dos recursos de processamento	$0,15$\$/hora
Custo de transferência de dados para **Amazon**	$0,00$\$/GB
Custos de transferência de dados a partir da **Amazon**	• até 1 GB/mês: $0,00$\$/GB • até 10 TB/mês: $0,12$\$/GB • até 50 TB/mês: $0,09$\$/GB • até 150 TB/mês: $0,07$\$/GB • a partir de 150 TB/mês: $0,05$\$/GB
Tempo para inicialização de serviço em máquina virtual	5 min
Risco de negação de serviço no *mercado sob demanda*	10%

empresas de computação[1]. Tais empresas tipicamente trabalham com 10% de taxa de lucro e incluem na margem de contribuição gastos com pessoal, administração, etc.

Definidos os planos praticados pelo provedor de SaaS a serem considerados nos experimentos buscou-se caracterizar a carga de trabalho de um cliente de SaaS para cada plano escolhido. Inicialmente foi feita uma análise considerando os objetos que compõe as pági-

[1]http://biz.yahoo.com/p/sum_qpmd.html

Tabela 6.2: Caracterização dos parâmetros relativos ao provedor de SaaS

Parâmetro	**Valor**
Planos do **BigCommerce** utilizados	*Bronze* • Limite de transferência: 2 GB • Limite de armazenamento: 200 MB • Mensalidade: $\$24,95$ *Gold* • Limite de transferência: 5 GB • Limite de armazenamento: 500 MB • Mensalidade: $\$79,95$ *Diamond* • Limite de transferência: 45 GB • Limite de armazenamento: 3 GB • Mensalidade: $\$299,95$
Custo por transferência fora do plano	$0,005$\$/MB
Custo de processamento fora do plano	$0,52$\$/hora
Margem de contribuição por plano	30%

nas de empresas de comércio eletrônico como **Lojas Americanas S.A**[2], **Magazine Luiza**[3], **Electronics City**[4], **Pride Nutrition**[5], bem como os objetos gerados pela ferramenta Geist [Kant, Tewari e Iyer 2001] e que são requisitados na carga sintética. Em média os objetos

[2]http://www.americanas.com.br/

[3]http://www.magazineluiza.com.br/

[4]http://www.electronicscity.com/

[5]http://www.pridenutrition.com/

alvo de uma requisição apresentam $13,44$ KB. A partir deste valor considerou-se os limites de transferência de dados nos planos *Bronze*, *Gold* e *Diamond* para, assim, calcular o total de requisições que poderiam ser atendidas dentro dos limites de cada plano. Considerando este valor limite e a margem de contribuição de 30% vista acima, foram definidas as taxas de chegada de requisições de um dia de carga típica de uma semana de carga típica para cada um dos planos, conforme a Tabela 6.3.

Além de um valor típico de taxa de chegada, a carga de trabalho foi caracterizada de acordo com estudos existentes na literatura [Menascé et al. 2003] [Arlitt, Krishnamurthy e Rolia 2001]. Tais estudos demonstram que a carga de trabalho das aplicações de comércio eletrônico apresentam variações dentro de intervalos de tempo de um dia, uma semana e um mês. Em cada dia existe um período no qual a quantidade de requisições é mais elevada, tipicamente de 9:00 AM até as 9:00 PM. Em cada semana existem dias que apresentam uma carga mais elevada (aqui referenciado como dia de pico) que o padrão da semana (aqui referenciado como dia típico). Além disso, existem dias que apresentam uma carga mais baixa (aqui referenciado como dia de menor carga). Pensando em intervalos de meses, existem meses que apresentam datas comemorativas especiais (aqui referenciado como semanas de pico), como Dia dos Pais ou Natal, que contribuem para um aumento na carga de trabalho submetida à aplicação. Tal aumento ocorre de maneira gradativa nos dias que antecedem o período comemorativo (este período de aumento gradativo é aqui referenciado como semana de transição). Segundo [Arlitt, Krishnamurthy e Rolia 2001] este aumento na carga de trabalho em períodos de pico é em torno de duas vezes o valor médio encontrado.

De posse desta análise foi definida a taxa de chegada de um dia típico de uma semana típica para cada um dos planos em função do cálculo com base nos limites definidos nos planos. Independentemente da semana, seja ela típica, de transição ou de pico, os dias de menor carga possuem uma taxa de chegada que corresponde a 50% da taxa do dia típico, e os dias de pico possuem uma taxa que corresponde a duas vezes a taxa do dia típico. As semanas de pico são definidas de modo que o dia típico da semana de pico possua duas vezes a carga do dia típico de uma semana típica. As semanas de transição são definidas de modo que o dia típico desta semana possua um acréscimo de 50% em sua carga em relação a um dia típico de uma semana típica. A Tabela 6.3 apresenta um resumo das taxas de chegada consideradas na simulação para cada plano ofertado pelo provedor de SaaS.

Tabela 6.3: Caracterização da taxa média de chegada de requisições por plano do provedor de SaaS

Tipo do Plano	**Taxas de Chegada**
Bronze	• Semana Típica – Dia típico: $0,058$ requisições/segundo; – Dia de pico: $0,117$ requisições/segundo; – Dia de menor carga: $0,029$ requisições/segundo; • Semana de Transição – Dia típico: $0,087$ requisições/segundo; – Dia de pico: $0,175$ requisições/segundo; – Dia de menor carga: $0,0435$ requisições/segundo; • Semana de Pico – Dia típico: $0,117$ requisições/segundo; – Dia de pico: $0,234$ requisições/segundo; – Dia de menor carga: $0,058$ requisições/segundo;

Continua na página seguinte

Tabela 6.3 – continuação da página anterior

Tipo do Plano	**Taxas de Chegada**
Gold	• Semana Típica – Dia típico: $0,176$ requisições/segundo; – Dia de pico: $0,35$ requisições/segundo; – Dia de menor carga: $0,09$ requisições/segundo; • Semana de Transição – Dia típico: $0,264$ requisições/segundo; – Dia de pico: $0,525$ requisições/segundo; – Dia de menor carga: $0,135$ requisições/segundo; • Semana de Pico – Dia típico: $0,35$ requisições/segundo; – Dia de pico: $0,7$ requisições/segundo; – Dia de menor carga: $0,176$ requisições/segundo;

Continua na página seguinte

Tabela 6.3 – continuação da página anterior

Tipo do Plano	Taxas de Chegada
Diamond	• Semana Típica – Dia típico: $0,65$ requisições/segundo; – Dia de pico: $1,3$ requisições/segundo; – Dia de menor carga: $0,325$ requisições/segundo; • Semana de Transição – Dia típico: $0,975$ requisições/segundo; – Dia de pico: $1,95$ requisições/segundo; – Dia de menor carga: $0,4875$ requisições/segundo; • Semana de Pico – Dia típico: $1,3$ requisições/segundo; – Dia de pico: $2,6$ requisições/segundo; – Dia de menor carga: $0,65$ requisições/segundo.

Por fim, faz-se necessário definir as variáveis independentes que estão relacionadas aos aspectos gerais da simulação. A Tabela 6.10 apresenta os níveis considerados para estas variáveis. A tabela demonstra que as simulações foram realizadas considerando o planejamento e a execução de um ano de carga de trabalho da aplicação de comércio eletrônico. Os erros de previsão da carga de trabalho considerados tiverem como base valores típicos obtidos a partir de uma análise do crescimento do mercado de comércio eletrônico [UEMA e LAZZARI 2008]. Foram realizadas 70 repetições de cada cenário de avaliação, onde cada repetição representou uma carga de trabalho diferente gerada pela ferramenta Geist a partir das mesmas taxas médias de chegada.

Tabela 6.4: Caracterização dos parâmetros gerais utilizados na simulação

Parâmetro	Valor
Número de Clientes de SaaS	$10, 50, 100$
Estratégia de Provisão Dinâmica	*DPS Oráculo*
Estratégia de Escalonamento	*Round-Robin* por núcleo de processamento
Período de Simulação	365 dias
Duração do período de uso (τ)	1 mês
Erro de previsão da carga de trabalho	-40%, $+40\%$
Número de Repetições	70

6.5 Análise dos Resultados

As utilidades coletadas e os **ganhos** calculados a partir destas utilidades, vide Equação 6.1, foram avaliados segundo o teste de Shapiro-Wilk [Shapiro e Wilk 1965] e gráficos Q-Qplot [Jain 2008]. Os resultados demonstraram que as utilidades seguem uma distribuição normal, porém os ganhos não se comportam segundo uma normal. Avaliando as utilidades coletadas em todos os cenários de avaliação percebe-se que utilidades positivas foram obtidas para a grande maioria dos cenários, exceto para os cenários da estratégia SUPER com 10 clientes de SaaS.

6.5.1 Viabilidade do planejamento de capacidade

O primeiro ponto de avaliação das heurísticas propostas no Capítulo 4 consiste em verificar a viabilidade da realização de um planejamento de capacidade para um intervalo D de avaliação. Para esta avaliação foram elaboradas as seguintes hipóteses:

- H1.0: $\upsilon_{SUPER}(D) = \upsilon_{ON}(D) = \upsilon_{UT}(D) = \upsilon_{RF}(D)$, ou seja, as utilidades obtidas pelas heurísticas no intervalo avaliado podem ser avaliadas como similares;
- H1.1: Ao menos uma heurística apresenta utilidades diferentes das demais heurísticas.

Inicialmente foram considerados os cenários com 100 clientes de SaaS, 10% de risco de negação de serviço no *mercado sob demanda* e erros de predição da carga de trabalho de

$+40\%$ e -40%. Um vez que o projeto experimental apontou que o comportamento das heurísticas UT e RF é diferenciado quando lida-se com erros positivos e negativos de predição de carga de trabalho, foram conduzidos testes separados para o erro de predição positivo e para o erro de predição negativo. Diante deste cenário tem-se um fator único (heurística de planejamento) e deseja-se checar o impacto dos níveis deste fator. Para isto foram realizadas análises de variância (ANOVA) [Jain 2008], uma para cada erro de predição, com as utilidades obtidas pela simulação das heurísticas.

Avaliando os resultados da análise de variância pode-se rejeitar a hipótese nula (H1.0) e conclui-se que ao menos uma heurística apresenta utilidades que podem ser tidas como diferentes das demais. Focando na análise da viabilidade do planejamento de capacidade o primeiro passo da análise *post hoc* consistiu em verificar as seguintes hipóteses:

- H2.0: $\upsilon_{ON}(D) = \upsilon_{UT}(D)$
- H3.0: $\upsilon_{ON}(D) = \upsilon_{RF}(D)$
- H4.0: $\upsilon_{ON}(D) = \upsilon_{SUPER}(D)$

Foi realizado um conjunto de testes-t [Jain 2008], considerando a correção de Bonferroni, para verificar tais hipóteses. Analisando os p-valores de tais testes, considerando um nível de confiança de 95%, chega-se a conclusão de que UT, RF e SUPER não apresentam utilidades iguais às utilidades apresentadas pela heurística ON. Um novo conjunto de testes-t foi realizado considerando como hipóteses nulas o fato das heurísticas UT, RF e SUPER apresentarem utilidades maiores ou igual, ou ainda menores ou igual, que as utilidades de ON. Avaliando os p-valores destes testes conclui-se que: (i) $\upsilon_{UT}(D) > \upsilon_{ON}(D)$; (ii) $\upsilon_{RF}(D) > \upsilon_{ON}(D)$; (iii) $\upsilon_{SUPER}(D) < \upsilon_{ON}(D)$. Ou seja, realizar o planejamento de capacidade considerando o modelo de utilidade proposto no Capítulo 3 apresentou melhorias de utilidade em relação a heurística que não realiza planejamento de capacidade. Além disso, realizar um superprovisionamento da infraestrutura de TI conduziu a utilidades inferiores às utilidades alcançadas por todas as outras heurísticas.

Os testes conduzidos acima foram repetidos para os cenários com 50 e 10 clientes de SaaS e as conclusões acima também foram verificadas.

6.5.2 Importância Prática

Demonstrado que as heurísticas RF e UT apresentam utilidades maiores que as utilidades apresentadas pela heurística ON e demonstrado que a estratégia SUPER apresenta utilidades inferiores às utilidades apresentadas por ON, é interessante quantificar tais melhorias e perdas. Neste sentido foram, inicialmente, considerados os cenários com 10, 50 e 100 clientes de SaaS, 10% de risco de negação de serviço no *mercado sob demanda* e erros de predição da carga de trabalho de $+40\%$ e -40%. As Tabelas 6.5 e 6.6 apresenta os intervalos de confiança das melhorias/perdas de utilidade obtidas pelas heurísticas UT, RF e SUPER nos cenários considerados. Estes valores representam os intervalos de confiança da diferença entre as utilidades obtidas por UT, RF e SUPER em relação à utilidade obtida por ON. A unidade desta diferença é o $.

Tabela 6.5: Intervalos de 95% de confiança para melhorias/perdas: erro de predição de -40%

Número de clientes	**Intervalos de confiança**		
	SUPER	**UT**	**RF**
10	$[-3.700,27 : -3.089,35]$	$[1.203,34 : 1.316,38]$	$[259,21 : 292,36]$
50	$[-6.690,48 : -6.125,59]$	$[2.086,31 : 2.197,87]$	$[543,55 : 609,74]$
100	$[-10.426,02 : -9.838,21]$	$[3.023,22 : 3.161,13]$	$[1.120,71 : 1.218,84]$

Tabela 6.6: Intervalos de 95% de confiança para melhorias/perdas: erro de predição de +40%

Número de clientes	**Intervalos de confiança**		
	SUPER	**UT**	**RF**
10	$[-12.851,03 : -11.730,06]$	$[60,87 : 178,29]$	$[887,982 : 1.000,77]$
50	$[-32.110,84 : -30.396,30]$	$[905,34 : 995,37]$	$[3.285,72 : 3.398,62]$
100	$[-53.935,84 : -51.908,66]$	$[1.545,83 : 1.649,35]$	$[4.818,71 : 4.959,68]$

Analisando os intervalos de confiança percebe-se que a estratégia SUPER apresenta intervalos com valores negativos uma vez que a utilidade desta heurística é sempre inferior às utilidades apresentadas pela heurística ON. Além disto, pode-se perceber que as melhorias/perdas apresentadas variam conforme a quantidade de clientes de SaaS que estão contra-

tando serviços junto ao provedor de SaaS. Uma maior quantidade de clientes de SaaS gera uma maior receita para o provedor de SaaS, bem como implica em um maior consumo de recursos junto ao provedor de IaaS. Erros e acertos realizados durante o planejamento de capacidade produzem maiores efeitos nas utilidades obtidas pelo provedor de SaaS quando da existência de uma maior quantidade de clientes de SaaS. Analisando as Tabelas 6.5 e 6.6 percebe-se que as heurísticas UT e RF apresentam os melhores resultados. A Tabela 6.7 apresenta os intervalos de confiança das melhorias de utilidade obtidas por RF e UT considerando um erro de predição de 0% e 100 clientes de SaaS. Tais intervalos representam os melhores resultados alcançados por estas heurísticas.

Tabela 6.7: Intervalos de 95% de confiança para melhorias/perdas: erro de predição de 0%

Número de clientes	**Intervalos de confiança**	
	UT	**RF**
100	$[5.322,900 : 5.515,692]$	$[5.237,582 : 5.442,380]$

De modo a avaliar o quanto tais melhorias/perdas representam da utilidade alcançada por cada heurística, foram calculados os intervalos de confiança para a métrica **ganho**, definida pela Equação 6.1, em relação à estratégia ON. Tais valores são apresentados nas Tabelas 6.8 e 6.9.

Tabela 6.8: Intervalos de 95% de confiança para **ganhos**: erro de predição de -40%

Clientes	**Intervalos de confiança**		
	SUPER	**UT**	**RF**
10	$[-966,44 : -424,87]$	$[173,25 : 407,69]$	$[37,05 : 86,03]$
50	$[-14,18 : -13,41]$	$[4,55 : 4,72]$	$[1,19 : 1,28]$
100	$[-10,03 : -9,57]$	$[2,95 : 3,03]$	$[1,09 : 1,15]$

Tabela 6.9: Intervalos de 95% de confiança para **ganhos**: erro de predição de +40%

Clientes	**Intervalos de confiança**		
	SUPER	**UT**	**RF**
10	$[-4.269,9 : -1.786,8]$	$[6,7 : 27,9]$	$[135 : 329]$
50	$[-69,26 : -66,2]$	$[1,98 : 2,09]$	$[7,17 : 7,31]$
100	$[-51,9 : -50,5]$	$[1,51 : 1,59]$	$[4,69 : 4,77]$

Os cenários com 10 clientes de SaaS apresentam uma grande variação nos valores do **ganho**, bem como **ganhos** diferentes dos obtidos com outras quantidades de clientes, dada uma maior variação na composição da carga de trabalho devido ao baixo número de clientes. Nestes cenários existem cargas de trabalho nas quais o sorteio dos planos dos clientes de SaaS conduziu, por exemplo, a uma composição majoritariamente de clientes *Diamond*, enquanto em outras cargas a composição é majoritariamente de clientes *Bronze*. Para quantidades maiores de clientes de SaaS, 50 e 100 clientes, o sorteio tende a uma maior igualdade de clientes em cada plano e, por consequência, uma maior estabilidade nos **ganhos** obtidos. Uma quantidade maior de repetições, superior às 70 repetições utilizadas nos experimentos, conduziria a intervalos de confiança de menor tamanho.

A Tabela 6.10 apresenta os intervalos de confiança dos **ganhos** obtidos por UT e RF nos cenários nos quais as heurísticas apresentaram as maiores utilidades (erro de predição de 0%). Mesmo tais **ganhos** apresentam valores baixos se comparados com os ganhos obtidos pelo planejamento ótimo apresentados na Tabela 6.11. O planejamento ótimo obtém melhores resultados em relação às heurísticas UT e RF ao aperfeiçoar a quantidade de recursos reservados. Em relação aos planos de reserva das heurísticas UT e RF nos cenários de subestimativas da carga de trabalho o planejamento ótimo reserva uma maior quantidade de recursos. Em relação aos planos de reserva das heurísticas UT e RF nos cenários de superestimativas da carga de trabalho o planejamento ótimo reserva uma menor quantidade de recursos.

Avaliando conjuntamente as Tabelas 6.6, 6.5, 6.9 e 6.8 percebe-se que apesar dos **ganhos** serem baixos o provedor de SaaS passa a obter valores em $ maiores conforme se aumenta o número de clientes de SaaS. Desta forma, os valores em $ passam a se tornar mais signi-

ficativos, e por consequência o planejamento de capacidade, quanto maior for a quantidade de clientes assinando planos junto ao provedor de SaaS.

Tabela 6.10: Intervalos de 95% de confiança para ganhos: erro de predição de 0%

Número de clientes	**Intervalos de confiança**	
	UT	**RF**
100	$[5,21 : 5,26]$	$[5,15 : 5,18]$

Tabela 6.11: Intervalos de 95% de confiança para ganhos: planejamento ótimo

Clientes	**Intervalos de confiança**
	Ótimo
50	$[15,748 : 16,026]$
100	$[11,527 : 11,624]$

6.5.3 Análise de Sensibilidade: Erro de predição

O projeto experimental realizado apontou o **número de clientes de SaaS** como grande fator significativo. Avaliando os dados das simulações percebeu-se que uma quantidade muita pequena de clientes de SaaS gerou resultados quantitativos diferentes dos resultados obtidos a partir de 50 clientes. Além disso, já no projeto experimental percebeu-se que as heurísticas UT e RF se comportam de forma diferenciada mediante erros positivos e negativos de predição da carga de trabalho.

De modo a avaliar melhor o comportamento das heurísticas UT e RF mediante diferentes erros de predição e, assim, verificar se o efeito do fator erro de predição não foi mascarado no projeto experimental pelos níveis escolhidos para os demais fatores foi realizada uma análise de sensibilidade considerando o fator erro de predição. O resultado desta análise é apresentado na Figura 6.1.

A partir da análise da Figura 6.1 tem-se confirmada a hipótese levantada durante o projeto experimental de que as heurísticas UT e RF possuem comportamentos distintos mediante erros positivos e erros negativos de predição da carga de trabalho. Para erros positivos de predição a heurística RF apresentou melhores resultados por se aproximar melhor da quantidade

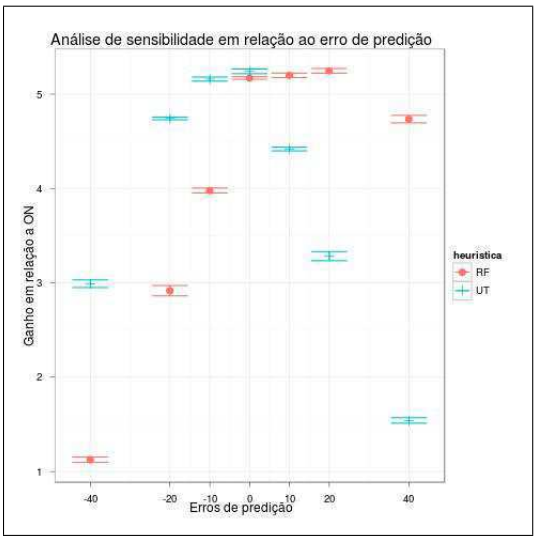


Figura 6.1: Análise de Sensibilidade: Variação do erro de predição da carga de trabalho

de recursos necessários para execução da carga de trabalho, ao passo que UT superestimou a quantidade de recursos necessários. Para subestimativas da carga de trabalho ambas as heurísticas reservaram uma quantidade de recursos de modo que o custo de uso dos recursos reservados foi mais barato que o custo de uso de recursos sob demanda. Por outro lado, RF reserva uma pequena quantidade de recursos, quantidade esta inferior à quantidade reservada por UT. Por não utilizar uma maior quantidade de recursos reservados em detrimento de recursos sob demanda RF obtém um custo total superior ao custo obtido por UT.

Seja para superestimativas ou subestimativas da carga de trabalho, tem-se que quão melhor for a predição da carga de trabalho futura melhor serão os resultados alcançados pelas heurísticas UT e RF.

6.5.4 Análise das heurísticas

Confirmadas as hipóteses de que o planejamento de capacidade utilizando as heurísticas propostas no Capítulo 4 propicia melhores utilidades para o provedor de SaaS, e avaliados os **ganhos** obtidos pelas heurísticas, faz-se necessário analisar o desempenho das heurísticas UT, RF e SUPER. Uma nova hipótese foi formulada de modo a se buscar estabelecer uma relação entre as heurísticas UT e RF:

- H5.0: $\upsilon_{UT}(D) = \upsilon_{RF}(D)$, ou seja, as utilidades obtidas pelas heurísticas UT e RF

podem ser avaliadas como similares;

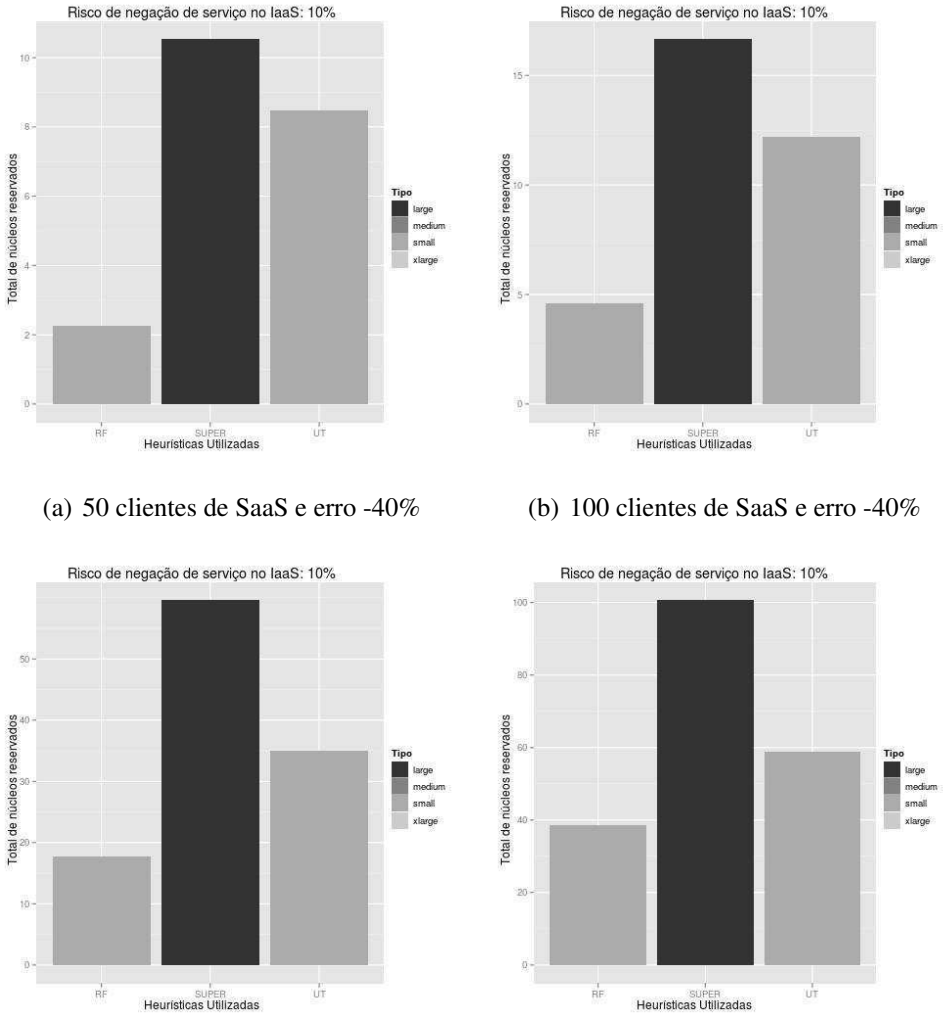
- H5.1: As heurísticas UT e RF não apresentam utilidades similares.

Foram considerados os cenários com 10, 50 e 100 clientes de SaaS, 10% de risco de negação de serviço no *mercado sob demanda* e erros de predição da carga de trabalho de $+40\%$ e -40%. Analisando os p-valores obtidos através da realização de testes-t pode-se rejeitar a hipótese nula. De posse desta conclusão um novo conjunto de testes-t foi realizado considerando como hipóteses nulas o fato da heurística UT apresentar utilidades maiores ou igual, ou ainda menores ou igual, que as utilidades apresentadas por RF. Avaliando os p-valores destes novos testes chega-se a duas conclusões:

1. $\upsilon_{UT}(D) > \upsilon_{RF}(D)$ para os cenários com erro de predição da carga de trabalho de $+40\%$;

2. $\upsilon_{UT}(D) < \upsilon_{RF}(D)$ para os cenários com erro de predição da carga de trabalho de -40%.

Para buscar explicações que justifiquem as relações encontradas entre as heurísticas se faz necessário analisar as quantidades de recursos que foram reservados por cada heurística, bem como avaliar o consumo destes recursos ao longo do intervalo D de avaliação. A Figura 6.2 apresenta a média do total de núcleos reservados por cada heurística para os cenários com 50 e 100 clientes de SaaS. Os cenários com 10 clientes de SaaS não foram apresentados uma vez que as relações encontradas também estão presentes em tais cenários e dado que os cenários de 10 clientes apresentou grandes variações nos **ganhos** devido às variações na composição da carga de trabalho (conforme explicado na Seção 6.5.2).



(a) 50 clientes de SaaS e erro -40%

(b) 100 clientes de SaaS e erro -40%

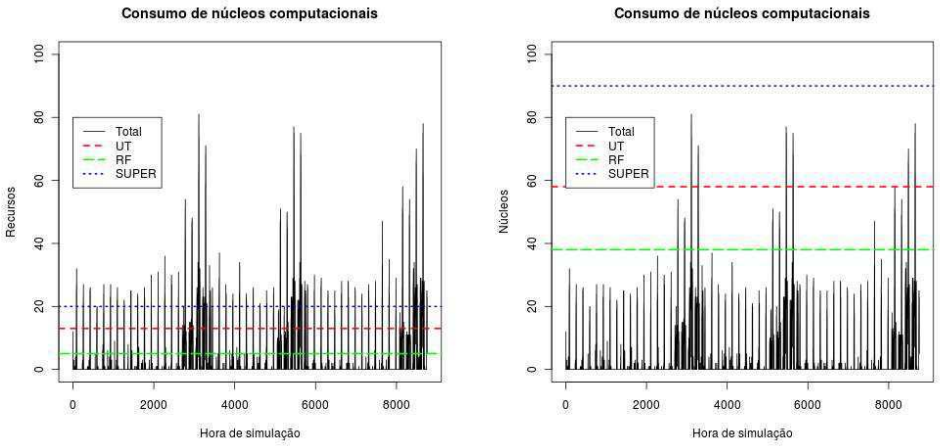
(c) 50 clientes de SaaS e erro +40%

(d) 100 clientes de SaaS e erro +40%

Figura 6.2: Quantidade média de núcleos reservados por cada heurística

Em relação à estratégia SUPER duas características devem ser destacadas: (i) pensando em um superprovisionamento da infraestrutura a estratégia SUPER utilizou recursos com maior poder computacional, ou seja, recursos *large* da classe *standard* que possuem 2 núcleos e, assim, uma capacidade de processamento superior à capacidade dos recursos *small* da mesma classe; (ii) a heurística elabora um plano de consumo de recursos ao longo do tempo, de acordo com a chegada de requisições e as demandas destas requisições, de modo a calcular a maior quantidade de recursos em paralelo que poderia ser utilizada e reserva 20% deste total.

Avaliando o total de núcleos reservados pela estratégia SUPER percebe-se que tal valor é superior ao total reservado por todas as outras heurísticas. De modo a verificar quanto os núcleos reservados representam do total requisitado, a Figura 6.3 apresenta o total de núcleos consumidos ao longo do intervalo D e a quantidade de núcleos que foi disponibilizado no mercado de reservas a partir do plano de reservas elaborado por cada heurística para um cenário com 100 clientes de SaaS. Percebe-se que em termos da quantidade de núcleos a estratégia SUPER reserva uma quantidade bem superior às quantidades reservadas por UT e RF.



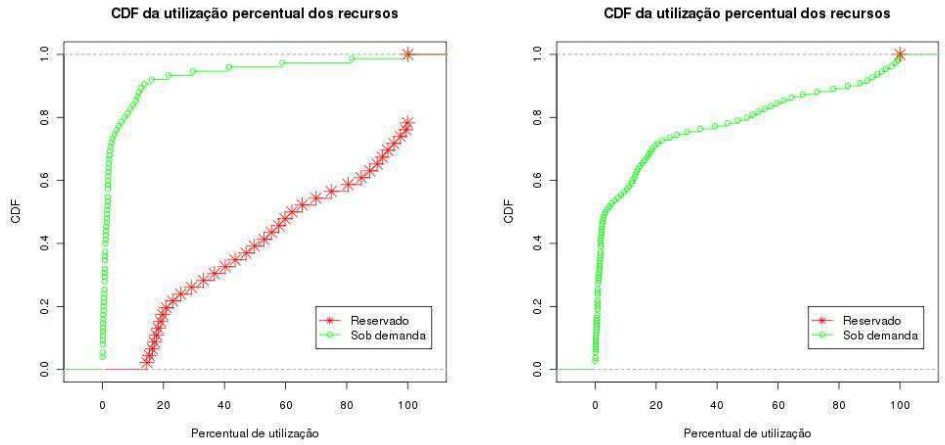
(a) Consumo Heurísticas com erro -40% (b) Consumo Heurísticas com erro +40%

Figura 6.3: Consumo de núcleos computacionais ao longo do intervalo D de avaliação

Avaliando-se a relação entre o total de recursos consumidos e o total de recursos disponibilizados pela estratégia SUPER percebe-se que nos cenários de superestimativas da carga de trabalho (Figura 6.3(b)) a quantidade de núcleos reservados é bem maior que a quantidade tipicamente consumida. Como consequência, existem recursos que são muito utilizados, ao passo que outros recursos são pouco utilizados. Para os cenários de subestimativas da carga de trabalho (Figura 6.3(a)) a quantidade de núcleos reservados se aproxima mais da quantidade de núcleos tipicamente utilizada, melhorando a utilização dos recursos.

A Figura 6.4 apresenta a função de densidade acumulada (CDF) da taxa de utilização dos recursos reservados por SUPER ao longo do intervalo D, ou seja, o percentual do intervalo D durante o qual os recursos foram utilizados. Nos cenários de superestimativas da carga, cerca de 30% dos recursos foram utilizados por no máximo 50% do intervalo D. Nos cenários de

subestimativas as taxas de utilização ficam em torno de 100%, dado o menor número de recursos reservados. É importante destacar que a taxa de utilização apresentada implica que o recurso foi utilizado e não necessariamente que todos os núcleos do recurso foram utilizados. Analisando os rastros da simulação percebe-se que tipicamente um núcleo de processamento é utilizado. Além disso, analisando o preço de uso de um recurso *small* sob demanda, utilizado pela heurística ON, e o preço de uso e de reserva de um recurso *large*, utilizado por SUPER, percebe-se que não existe uma taxa de utilização que torne os recursos do tipo *large* mais baratos que os recursos *small* do *mercado sob demanda*. Como consequência, quão mais baixa for a taxa de utilização dos recursos maiores são as perdas em relação a heurística ON. Logo, nos cenários com superestimativa da carga de trabalho SUPER apresenta as maiores perdas de utilidade.



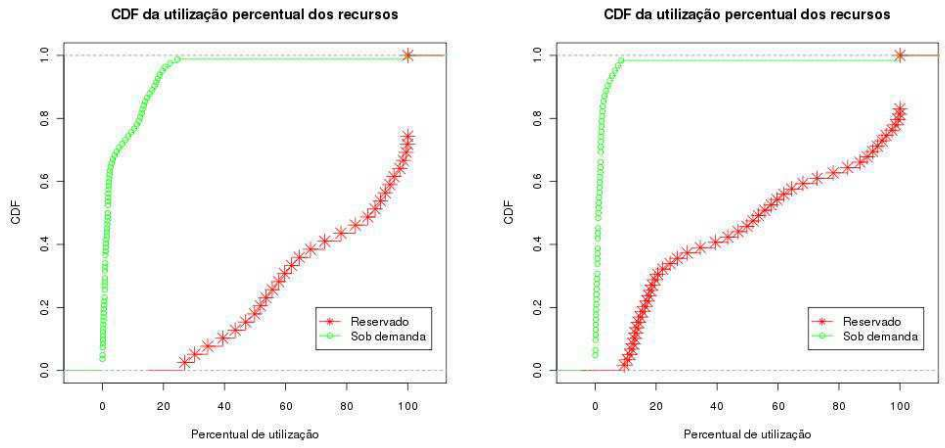
(a) Estratégia SUPER: erro de predição +40% (b) Estratégia SUPER: erro de predição -40%

Figura 6.4: CDF do percentual de utilização dos recursos

Considerando que a estratégia SUPER reservasse a mesma quantidade de recursos, porém considerando agora recursos do tipo *small* ao invés de recursos do tipo *large*, o total de recursos reservados seria um valor intermediário entre os valores das heurísticas RF e UT. Esta alteração conduziria a utilidades inferiores a RF, e superiores a UT, para superestimativas da carga de trabalho devido à existência de recursos com baixa utilização. Para subestimativas da carga de trabalho as utilidades seriam superiores às utilidades de RF e inferiores às utilidades de UT devido ao fato dos recursos passarem a apresentar taxas de utilização condizentes com os preços dos mercados de reserva e sob demanda.

Avaliando o total de reservas realizadas pelas heurísticas RF e UT, Figura 6.2, percebe-se que a heurística RF realiza reservas mais conservadoras que as reservas feitas por UT, ou seja, RF possui uma menor quantidade de recursos reservados disponíveis para uso. A heurística RF avalia estatísticas médias da taxa de chegada de requisições e da demanda de requisições buscando fazer com que cada recurso reservado seja utilizado por no máximo 75% do intervalo D. Por outro lado, apesar da heurística UT utilizar informações mais detalhadas sobre tempo de chegada e demandas das requisições, a heurística considera que cada recurso reservado venha a ser utilizado, durante o intervalo D, por uma quantidade de horas que corresponde a quantidade mínima de horas a serem consumidas de modo que o uso de recursos reservados seja mais barato que o uso de recursos sob demanda. Para o mercado da **Amazon EC2** este valor tipicamente representa uma taxa de utilização em torno de $47,2\%$.

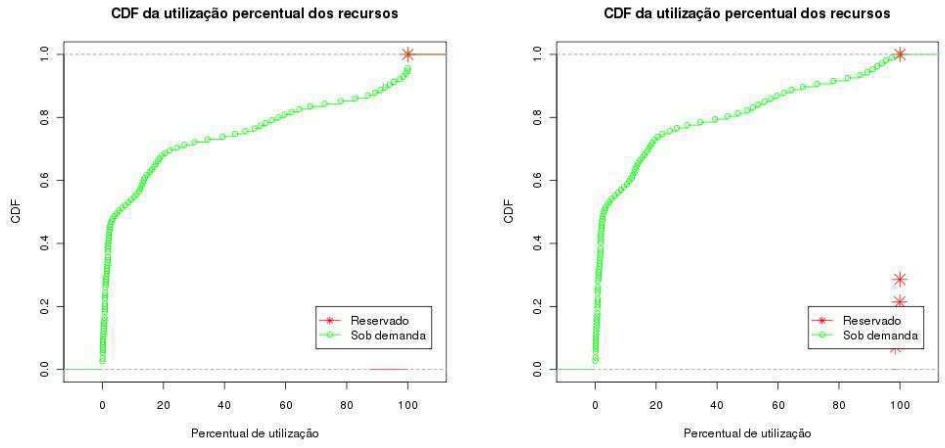
Como a heurística UT reserva recursos no limiar de utilização para que o uso de um recurso reservado seja mais barato que o uso de um recurso sob demanda, superestimativas da carga de trabalho fazem com que recursos reservados por UT apresentem taxas de utilização inferiores ao limiar que torna a reserva financeiramente viável. A Figura 6.3(b) demonstra que a quantidade de recursos reservados por UT é bem superior a quantidade de recursos tipicamente consumida. Para estes cenários, o uso de alguns recursos reservados se torna mais caro que o uso de recursos sob demanda. Já a heurística RF, por reservar uma quantidade inferior de recursos, faz com que as taxas de utilização dos recursos reservados sejam maiores, barateando o uso dos recursos e contribuindo para que a utilidade obtida seja superior à utilidade de UT. A Figura 6.5 apresenta a CDF das taxas de utilização de recursos para as heurísticas UT e RF nos cenários de superestimativas da carga de trabalho.



(a) Heurística RF (b) Heurística UT

Figura 6.5: CDF do percentual de utilização dos recursos: erro de predição de +40%

Por outro lado, subestimativas da carga de trabalho fazem com que tanto os recursos reservados por UT como os recursos reservados por RF apresentem taxas de utilização que tornam viável a reserva de recursos, vide Figura 6.6. Por reservar uma quantidade superior de recursos, e de que recursos que são tipicamente requisitados (vide Figura 6.3(a)), UT consegue realizar uma economia no uso de recursos reservados maior que a economia realizada pela heurística RF. Logo, as utilidades obtidas por UT são superiores às utilidades obtidas por RF.



(a) Heurística RF (b) Heurística UT

Figura 6.6: CDF do percentual de utilização dos recursos: erro de predição de -40%

6.5.5 Lições aprendidas

As versões das heurísticas UT e RF apresentadas no Capítulo 4 foram resultado de iterações que buscaram melhorias realizadas em cada uma das heurísticas. Em relação à heurística RF a primeira versão desenvolvida considerava unicamente a demanda da aplicação para elaborar o plano de reservas. Além disso, esta primeira versão calculava, a cada hora, o total de requisições atendidas nos mercados sob demanda e de reserva, porém o cálculo do total de horas de CPU consumidas em cada mercado era realizado a partir do agregado de requisições atendidas em cada mercado durante todo o período de avaliação. Esta versão reservava uma pequena quantidade de recursos e, assim, não conseguia tirar proveito da economia propiciada pelo uso de recursos reservados. Em alguns cenários a utilidade obtida chegou a ser semelhante à utilidade obtida pela heurística ON.

De modo a aumentar a quantidade de recursos reservados por RF, e assim melhorar a utilidade obtida pela heurística, duas alterações foram realizadas. O cálculo do total de horas de CPU requisitadas em cada mercado passou a ser realizado a cada hora em função do total de requisições alocadas para cada mercado e, além disso, este cálculo passou a considerar o $FATOR\_AJUSTE$ do DPS. Como consequência a quantidade de recursos reservados por RF foi aumentada e a utilidade da heurística passou a ser superior à utilidade obtida por ON em todos cenários.

Em relação à heurística UT a primeira versão desenvolvida coletava as utilizações dos recursos e reservava os recursos que fossem utilizados por uma quantidade de horas superior ao limiar definido por $\frac{f_s}{(c_s^{on}-c_s^v)}$. Entretanto, esta versão apresentou a dificuldade de que, a menos que o DPS escolha de forma criteriosa quais recursos devem ser desligados junto ao provedor de IaaS quando a carga de trabalho for reduzida, recursos dificilmente obtém o mínimo de utilização que torna a reserva financeiramente viável. Como consequência, uma quantidade muito pequena de recursos era reservada e a heurística não conseguia tirar proveito da economia propiciada pelo uso de recursos reservados.

A segunda versão de UT passou a coletar a quantidade de recursos requisitados em cada hora de simulação e, a partir desta coleta, estimar a utilização dos recursos. Esta versão é apresentada no Capítulo 4. Além disso, esta versão passou a agregar o consumo de todos os recursos que foram utilizados por intervalos superiores ao limiar definido por $\frac{f_s}{(c_s^{on}-c_s^v)}$ e, assim, calcular o total de recursos a serem reservados. Como consequência uma maior

quantidade de recursos passou a ser reservada melhorando a utilidade obtida por UT em relação à primeira versão desta heurística. Entretanto, para os cenários de superestimativas da carga de trabalho esta nova versão passou a reservar uma quantidade de recursos maior do que o necessário. Buscando aperfeiçoar a quantidade de recursos reservados nestes cenários uma terceira versão da heurística UT foi desenvolvida. Esta terceira versão apresentava duas etapas: (i) a elaboração de um primeiro plano de reservas, conforme a versão de UT apresentada no Capítulo 4; (ii) o aperfeiçoamento do plano de reserva através da utilização da técnica de subida de encosta (do inglês, *hill climbing*). Esta terceira versão de UT reduziu as reservas realizadas pela segunda versão de UT, aperfeiçoando as utilidades obtidas por UT nos cenários de superestimativas da carga de trabalho. Entretanto, a queda na quantidade de recursos reservados reduziu as utilidades obtidas por UT nos cenários de subestimativas da carga de trabalho. Avaliando a utilidade obtida por UT em todos os cenários de simulação percebeu-se uma queda da utilidade em relação à utilidade obtida pela segunda versão da heurística.

6.5.6 Recomendações de Uso

Realizando uma comparação das utilidades e ganhos obtidos pelas heurísticas UT e RF no conjunto de cenários simulados tem-se que: (i) a heurística RF apresenta utilidades superiores às utilidades apresentadas por UT; (ii) a heurística RF apresenta **ganhos** superiores aos **ganhos** apresentados por UT.

Diante de um cenário no qual o gerente da infraestrutura de TI de um provedor de SaaS não possui conhecimento prévio e detalhado a respeito da qualidade das predições da carga de trabalho que são utilizadas pelo provedor tem-se como recomendação a utilização da heurística RF para o planejamento de capacidade. O risco existente neste cenário é que a empresa tipicamente lide com predições da carga de trabalho que subestimem a carga de trabalho real e, assim, passe a apresentar um ganho pequeno em relação ao ganho que poderia ser obtido caso existissem maiores subsídios para a escolha da heurística de planejamento.

Por outro lado, um gerente de infraestrutura de TI que já tem um melhor conhecimento das predições da carga de trabalho realizadas para planejar sua infraestrutura pode obter melhores resultados no planejamento ao aliar este conhecimento com o conhecimento a respeito do comportamento do mercado. Nestas condições o gerente pode possuir a percepção se tipi-

camente as predições da carga de trabalho superestimam ou subestimam a carga de trabalho real. De posse destes conhecimentos o gerente pode fazer uma escolha mais criteriosa entre as heurísticas apresentadas no Capítulo 4 e, assim, ampliar a utilidade do provedor de SaaS. Caso as predições tipicamente superestimem a carga de trabalho real é recomendável a escolha da heurística RF. Caso as predições tipicamente subestimem a carga de trabalho real é recomendável a escolha da heurística UT.

Ainda é possível que gerentes ao realizarem predições da carga de trabalho futuro atuem de forma conservadora buscando reservar uma quantidade de recursos elevada que busque evitar ao máximo que requisições não sejam atendidas, ou ainda sejam atendidas violando o SLA. Nestes cenários o gerente estaria lidando com uma superestimativa, intencional, da carga de trabalho e sua melhor escolha também seria a heurística RF.

6.6 Considerações Finais

Neste capítulo foi analisado o comportamento das heurísticas UT e RF frente o comportamento de três estratégias tidas como base para avaliação: ON, SUPER e ótimo. De modo geral, todas as heurísticas geraram utilidades positivas para o provedor de SaaS, exceto a estratégia SUPER para os cenários com 10 clientes de SaaS. Utilizar informações mais detalhadas sobre o modelo de utilidade do provedor de SaaS permitiu que as heurísticas UT e RF apresentassem as maiores utilidades dentre as quatro heurísticas avaliadas, fazendo com que o planejamento de capacidade se mostrasse viável para o provedor de SaaS. De modo geral, a heurística RF apresentou as melhores utilidades.

A heurística RF demonstrou ser mais conservadora que a heurística UT no momento de realização do planejamento de capacidade. Como consequência RF apresentou melhores valores de utilidade nos cenários com superestimativas na predição da carga de trabalho. Por outro lado, a heurística UT apresentou apresentou melhores valores de utilidade nos cenários com subestimativas na predição da carga de trabalho. Foi demonstrado, ainda, que uma predição mais acurada da carga de trabalho contribui para melhores resultados tanto para a heurística UT como para a heurística RF.

A avaliação da estratégia SUPER, que realiza um superprovisionamento da infraestrutura de IT, demonstrou que a escolha do tipo dos recursos reservados tem um grande impacto na

utilidade obtida pelo provedor de SaaS. A quantidade de recursos reservados por SUPER foi uma quantidade intermediária entre as heurísticas RF e UT. De modo geral o uso destes recursos superprovidos se tornou mais caro que o uso de recursos de menor capacidade do *mercado sob demanda*. Como consequência, a estratégia SUPER apresentou os piores resultados dentre as quatro heurísticas avaliadas.

As simulações apresentadas envolveram um máximo de 100 clientes de SaaS submetendo requisições à infraestrutura do provedor de SaaS. As utilidades médias obtidas por cada uma das heurísticas propostas nos cenários com 100 clientes de SaaS foram de: (i) UT: $\$106.803,729$; (ii) RF: $\$107.182,498$. Considerando que a empresa **BigCommerce**[6] afirma que possui um total de mais de 20.000 clientes pagantes uma estimativa otimista da utilidade total que seria obtida pela empresa utilizando cada uma das heurísticas é de: (i) UT: $\$21.360.746$; (ii) RF: $\$21.436.500$.

[6]http://www.bigcommerce.com/about.php

Capítulo 7

Conclusões e Trabalhos Futuros

Neste capítulo são apresentadas as conclusões do trabalho e trabalhos futuros que podem ser desenvolvidos para complementar o trabalho apresentado nesta dissertação.

7.1 Conclusões

Nesta dissertação foi avaliado o planejamento de capacidade de longo prazo de uma infraestrutura de TI utilizando recursos adquiridos junto a provedores de Computação na Nuvem que ofertam *Infraestrutura como Serviço*, ou seja, provedores de IaaS. Foi considerado um cenário no qual um provedor de *Software como Serviço*, provedor de SaaS, utiliza os recursos adquiridos junto ao provedor de IaaS para ofertar sua aplicação. Os recursos adquiridos junto ao provedor de IaaS foram obtidos a partir de dois mercados: o *mercado de reservas* e o *mercado sob demanda*. O provedor de IaaS oferece diferentes tipos de recursos em cada mercado e combinações destes tipos podem ser utilizados na composição da infraestrutura.

A aplicação ofertada pelo provedor de SaaS escolhida como estudo de caso foi uma aplicação de comércio eletrônico. Aplicações de comércio eletrônico são aplicações muito estudadas na literatura com vários trabalhos promovendo a caracterização deste tipo de carga de trabalho [Arlitt, Krishnamurthy e Rolia 2001] [Menascé et al. 2003] [Menascé et al. 2000] [Zhang, Cherkasova e Smirni 2007]. Por conta desta ampla caracterização considerar este tipo de aplicação permite, ainda, o uso de geradores de carga de trabalho bem consolidados [Kant, Tewari e Iyer 2001]. Além disto, tem-se disponível a caracterização de um modelo de

negócio de um provedor de SaaS consolidado no mercado, **BigCommerce**[1], que oferta uma aplicação SaaS de comércio eletrônico.

Neste contexto foi desenvolvido um modelo de utilidade para o provedor de SaaS que calcula a utilidade do provedor com base na receita, nos custos e nas penalidades/multas pagas pelo provedor. Foram propostas duas heurísticas para realização do planejamento de capacidade: (i) Heurística baseada na taxa de utilização dos recursos – UT; (ii) Heurística baseada em rede de filas – RF. As heurísticas foram avaliadas com base em um modelo de simulação que utilizou cargas sintéticas geradas a partir da ferramenta Geist [Kant, Tewari e Iyer 2001] e que considerou o modelo de negócio praticado pela **BigCommerce** como o modelo de SaaS e o modelo de negócio praticado pelo serviço **Amazon EC2**[2] como o modelo de IaaS. As heurísticas propostas foram confrontadas com três outras estratégias de planejamento de capacidade: (i) estratégia que realiza um superprovisionamento da infraestrutura de TI – SUPER; (ii) estratégia que não realiza reserva de recursos – ON; (iii) estratégia que realiza um planejamento de capacidade ótimo.

Os resultados obtidos apontam que a heurística RF apresenta os melhores resultados com uma utilidade média de $\$84.905,805$ e um **ganho** médio de $3,7702\%$. A heurística UT apresenta uma utilidade média de $\$84.513,344$ e um **ganho** médio de $3,1983\%$. Tanto a heurística RF como a heurística UT obtém utilidades superiores às utilidades apresentadas pelas estratégias ON e SUPER em todos os cenários simulados. Analisando os resultados obtidos com o planejamento ótimo percebe-se que existe espaço para melhorias nos resultados obtidos pelas heurísticas UT e RF. A análise dos dados demonstrou, ainda, que a heurística RF reserva uma quantidade de recursos inferior à quantidade reserva por UT. Como consequência, RF apresentou melhores valores de utilidade nos cenários com superestimativas na predição da carga de trabalho. Por outro lado, a heurística UT apresentou melhores valores de utilidade nos cenários com subestimativas na predição da carga de trabalho.

[1]http://www.bigcommerce.com/

[2]http://aws.amazon.com/ec2

7.2 Trabalhos Futuros

7.2.1 Análise preliminar do fator risco

O projeto experimental realizado apontou o **número de clientes de SaaS** e o **risco de negação de serviço no mercado sob demanda** como fatores significativos para avaliação nos experimentos. Realizadas as análises apresentadas nas Seções 6.5.1, 6.5.2 e 6.5.4 buscou-se realizar uma análise preliminar do impacto do risco de negação de serviço nas utilidades e ganhos obtidos pelas heurísticas. Três níveis foram considerados para o fator de risco: 5%, 10% e 15%. As seguintes hipóteses nulas foram inicialmente elaboradas:

- H6.0: $v_{UT}^{10\%}(D) = v_{UT}^{5\%}(D) = v_{UT}^{15\%}(D)$, ou seja, a utilidade obtida pela heurística UT não é alterada com a variação do risco;
- H7.0: $v_{RF}^{10\%}(D) = v_{RF}^{5\%}(D) = v_{RF}^{15\%}(D)$, ou seja, a utilidade obtida pela heurística UT não é alterada com a variação do risco;
- H8.0: $v_{SUPER}^{10\%}(D) = v_{SUPER}^{5\%}(D) = v_{SUPER}^{15\%}(D)$, ou seja, a utilidade obtida pela heurística UT não é alterada com a variação do risco;

Uma análise de variância e um conjunto de testes-t foram realizados considerando 10, 50 e 100 clientes de SaaS e erros de predição de $+40\%$ e -40%. Foi verificado que a alteração do fator risco afeta os valores de utilidade obtidos pelas heurísticas. Hipóteses similares às hipóteses H6.0, H7.0 e H8.0 foram elaboradas em relação ao **ganho** obtido por cada heurística. Através da realização de um conjunto de testes de postos de sinais de Wilcoxon [Jain 2008] conclui-se que a alteração do risco afeta os **ganhos** obtidos pelas heurísticas. Dado que o risco afeta as utilidades e **ganhos** das heurísticas, a avaliação realizada nas Seções 6.5.1, 6.5.2 e 6.5.4 foi realizada para os riscos de 5% e 15% e percebeu-se que as conclusões obtidas nas seções anteriores permanecem válidas.

Em seguida buscou-se entender como a alteração do risco afeta as utilidades e **ganhos** das heurísticas. Em relação à utilidade a expectativa inicial era de que o aumento do risco conduzisse a uma redução da utilidade das heurísticas. Todavia, analisando os resultados dos testes-t realizados descobriu-se que o aumento do risco conduziu a um aumento da utilidade nos cenários avaliados. Este aumento é decorrente da natureza conservadora do *DPS Oráculo*

utilizado nas simulações. O *DPS Oráculo* requisita uma quantidade elevada de recursos junto ao provedor de IaaS de modo a evitar que o SLA dos clientes de SaaS seja violado. Como consequência recursos sob demanda tendem a serem utilizados durante curtos intervalos de tempo. O aumento do fator risco contribui para que recursos sob demanda pouco utilizados não sejam obtidos e, assim, o provedor de SaaS acaba por reduzir seu custo relativo ao uso de recursos sob demanda. Para um DPS que requisite uma quantidade de recursos próxima à demanda da aplicação o aumento do risco deve conduzir ao não atendimento de requisições, ou ainda a violações do SLA, e, por consequência, a uma conclusão contrária a conclusão observada com o *DPS Oráculo*.

Em relação ao **ganho** esperava-se que o aumento do risco reduzisse a utilidade da heurística ON, dado que esta heurística depende unicamente do *mercado sob demanda*, e não conduzisse a um prejuízo da utilidade das heurísticas SUPER, UT e RF. Logo, esperava-se que o **ganho** fosse ampliado. Analisando os resultados dos testes de postos de sinais de Wilcoxon descobriu-se que, para os cenários avaliados, o aumento do risco de negação de serviço conduz a uma redução do **ganho** obtido pelas heurísticas. Novamente o comportamento conservador do *DPS Oráculo* conduz a estes resultados. Como a heurística ON consome uma quantidade muito elevada de recursos sob demanda, a economia no uso de recursos sob demanda e posterior melhoria na utilidade obtida pela heurística ON é superior a melhoria alcançada pelas demais heurísticas. Logo, o **ganho** é reduzido. Para um DPS que requisite uma quantidade de recursos próxima à demanda da aplicação o aumento do risco deve conduzir ao não atendimento de requisições, ou ainda a violações do SLA, por ON e, por consequência, a uma conclusão contrária a conclusão observada com o *DPS Oráculo*.

7.2.2 Próximos Passos

Como perspectivas de trabalhos futuros uma primeira etapa seria realizar uma avaliação criteriosa de sistemas de provisão dinâmica de recursos [Ranjan et al. 2002] [Urgaonkar et al. 2008] [Lee et al. 2010] [Kwok e Mohindra 2008] [Zhu et al. 2011] [Bi et al. 2010] [Chi, Qian e Lu 2011] no contexto de provedores SaaS que oferecem aplicações de comércio eletrônico. A partir desta avaliação torna-se possível escolher um sistema de provisão que apresente um bom desempenho, em termos de negócio, na execução da aplicação. Tal sistema de provisão passaria a substituir o *DPS Oráculo* que foi considerado nesta dissertação.

Escolhido um novo sistema de provisão dinâmica faz-se necessário reavaliar as utilidades e **ganhos** obtidos pelas heurísticas propostas neste novo cenário mais realista. Além disso, faz-se necessária uma avaliação criteriosa do impacto dos riscos de negação de serviço no *mercado sob demanda* no comportamento das heurísticas propostas de modo a verificar se as hipóteses que inicialmente foram levantadas em 7.2.1 são verdadeiras.

Os resultados apresentados demonstraram que os ganhos obtidos pelas heurísticas UT e RF chegam a 7% em relação a estratégia ON em alguns cenários. Analisando o planejamento ótimo percebe-se que ganhos na faixa de 11% a 16% podem ser obtidos. Diante disto, é interessante a investigação de estratégias de melhoria das heurísticas aqui propostas no sentido de buscar aperfeiçoar os ganhos obtidos por cada heurística.

Devido à grande variedade de aplicações SaaS atualmente presentes no mercado de Computação na Nuvem, é interessante a expansão do trabalho para outros tipos de aplicações que não foram consideradas nesta dissertação. Avaliar o planejamento de capacidade para aplicações populares como, por exemplo, redes sociais e *streaming* de áudio e vídeo seria outro passo natural a ser realizado.

Por fim, uma vez que o planejamento de capacidade envolve a reserva de recursos computacionais para um longo período futuro e considerando que os provedores de SaaS ofertam diferentes planos aos seus clientes, outros aspectos de negócio podem ser avaliados durante o planejamento. Outros aspectos do negócio como o tempo de retorno do investimento (do inglês, *payback*) realizado para efetivação da reserva e o valor presente líquido (VPL) dos valores pagos ao provedor de IaaS podem ser considerados. A avaliação de tais aspectos aproxima o planejamento de capacidade da avaliação de carteiras de investimentos tipicamente realizada na economia e pode trazer maiores benefícios para o provedor de SaaS.

Referências Bibliográficas

[Almeida 2002]ALMEIDA, V. Capacity planning for web services techniques and methodology. *Performance Evaluation of Complex Systems: Techniques and Tools*, Springer, p. 293–302, 2002.

[Almeida e Menascé 2002]ALMEIDA, V.; MENASCÉ, D. Capacity planning an essential tool for managing web services. *IT professional*, IEEE, v. 4, n. 4, p. 33–38, 2002.

[Ardagna, Panicucci e Passacantando 2011]ARDAGNA, D.; PANICUCCI, B.; PASSACANTANDO, M. A game theoretic formulation of the service provisioning problem in cloud systems. In: ACM. *Proceedings of the 20th international conference on World wide web*. [S.l.], 2011. p. 177–186.

[Arlitt, Krishnamurthy e Rolia 2001]ARLITT, M.; KRISHNAMURTHY, D.; ROLIA, J. Characterizing the scalability of a large web-based shopping system. *ACM Transactions on Internet Technology*, v. 1, n. 1, p. 44–69, 2001.

[Armbrust et al. 2009]ARMBRUST, M. et al. Above the Clouds : A Berkeley View of Cloud Computing Cloud Computing : An Old Idea Whose Time Has (Finally) Come. *Computing*, p. 07–013, 2009.

[Barford e Crovella 1998]BARFORD, P.; CROVELLA, M. Generating representative web workloads for network and server performance evaluation. In: ACM. *ACM SIGMETRICS Performance Evaluation Review*. [S.l.], 1998. v. 26, n. 1, p. 151–160.

[Bi et al. 2010]BI, J. et al. Dynamic provisioning modeling for virtualized multi-tier applications in cloud data center. In: IEEE. *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*. [S.l.], 2010. p. 370–377.

[Bichler, Setzer e Speitkamp 2006]BICHLER, M.; SETZER, T.; SPEITKAMP, B. Capacity planning for virtualized servers. In: *Workshop on Information Technologies and Systems (WITS), Milwaukee, Wisconsin, USA*. [S.l.: s.n.], 2006. v. 1.

[Buyya, Broberg e Goscinski 2011]BUYYA, R.; BROBERG, J.; GOSCINSKI, A. *Cloud computing: Principle and Paradigms*. [S.l.]: Wiley Online Library, 2011.

[Caldiera e Rombach 1996]CALDIERA, V.; ROMBACH, D. The goal question metric approach. *Encyclopedia of Software Engineering*, v. 2, p. 528–532, 1996.

[Calheiros et al. 2011]CALHEIROS, R. et al. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, Wiley Online Library, v. 41, n. 1, p. 23–50, 2011.

[Chapman et al. 2010]CHAPMAN, C. et al. Elastic service definition in computational clouds. In: IEEE. *Network Operations and Management Symposium Workshops (NOMS Wksps), 2010 IEEE/IFIP*. [S.l.], 2010. p. 327–334.

[Chen et al. 2011]CHEN, J. et al. Tradeoffs between profit and customer satisfaction for service provisioning in the cloud. In: ACM. *Proceedings of the 20th international symposium on High performance distributed computing*. [S.l.], 2011. p. 229–238.

[Cherkasova, Tang e Singhal 2004]CHERKASOVA, L.; TANG, W.; SINGHAL, S. An sla-oriented capacity planning tool for streaming media services. In: IEEE. *Dependable Systems and Networks, 2004 International Conference on*. [S.l.], 2004. p. 743–752.

[Chi, Qian e Lu 2011]CHI, R.; QIAN, Z.; LU, S. A heuristic approach for scalability of multi-tiers web application in clouds. In: IEEE. *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2011 Fifth International Conference on*. [S.l.], 2011. p. 28–35.

[Crovella e Bestavros 1996]CROVELLA, M.; BESTAVROS, A. Self-similarity in world wide web traffic: evidence and possible causes. In: ACM. *ACM SIGMETRICS Performance Evaluation Review*. [S.l.], 1996. v. 24, n. 1, p. 160–169.

[Daniel e Virgilio 2000]DANIEL, A.; VIRGILIO, A. Scaling for e-business: technologies, models, performance & capacity planning. 2000.

[Hawkins 2011]HAWKINS, L. *Fast-growing BigCommerce raises $15 million*. 2011. `http://www.statesman.com/business/fast-growing-bigcommerce-raises-15-million-1678495.html`. Acessado em 21 de Maio de 2012.

[Jacobs 2005]JACOBS, D. Enterprise software as service. *Queue*, ACM, v. 3, n. 6, p. 36–42, 2005.

[Jain 2008]JAIN, R. *The art of computer systems performance analysis*. [S.l.]: John Wiley & Sons, 2008.

[Janakiraman, Santos e Turner 2003]JANAKIRAMAN, G.; SANTOS, J.; TURNER, Y. Automated multi-tier system design for service availability. In: CITESEER. *Proceedings of the First Workshop on Design of Self-Managing Systems*. [S.l.], 2003.

[Kant, Tewari e Iyer 2001]KANT, K.; TEWARI, V.; IYER, R. Geist: Generator of ecommerce and internet server traffic. In: *Proc. of Int. Symposium on Performance Analysis of Systems and Software*. [S.l.: s.n.], 2001.

[Kwok e Mohindra 2008]KWOK, T.; MOHINDRA, A. Resource calculations with constraints, and placement of tenants and instances for multi-tenant saas applications. *Service-Oriented Computing–ICSOC 2008*, Springer, p. 633–648, 2008.

[Lawton 2008]LAWTON, G. Developing software online with platform-as-a-service technology. *Computer*, IEEE, v. 41, n. 6, p. 13–15, 2008.

[Lee, Chun e Katz 2011]LEE, G.; CHUN, B.; KATZ, R. Heterogeneity-aware resource allocation and scheduling in the cloud. *Proceedings of HotCloud*, 2011.

[Lee et al. 2010]LEE, Y. et al. Profit-driven service request scheduling in clouds. In: IEEE COMPUTER SOCIETY. *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. [S.l.], 2010. p. 15–24.

[Li et al. 2010]LI, A. et al. Cloudcmp: comparing public cloud providers. In: ACM. *Proceedings of the 10th annual conference on Internet measurement*. [S.l.], 2010. p. 1–14.

[Li, Casale e Ellahi 2010]LI, H.; CASALE, G.; ELLAHI, T. Sla-driven planning and optimization of enterprise applications. In: ACM. *Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*. [S.l.], 2010. p. 117–128.

[Liu, Heo e Sha 2005]LIU, X.; HEO, J.; SHA, L. Modeling 3-tiered web applications. In: IEEE. *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on*. [S.l.], 2005. p. 307–310.

[Lopes, Brasileiro e Maciel 2010]LOPES, R.; BRASILEIRO, F.; MACIEL, P. Business-driven capacity planning of a cloud-based it infrastructure for the execution of web applications. In: IEEE. *Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*. [S.l.], 2010. p. 1–8.

[Maciel et al. 2011]MACIEL, P. et al. Business-driven short-term management of a hybrid it infrastructure. *Journal of Parallel and Distributed Computing*, Elsevier, 2011.

[Maciel et al. 2008]MACIEL, P. et al. On the planning of a hybrid it infrastructure. In: IEEE. *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*. [S.l.], 2008. p. 496–503.

[Marques, Sauvé e Moura 2006]MARQUES, F.; SAUVÉ, J.; MOURA, A. Business-oriented capacity planning of it infrastructure to handle load surges. In: IEEE. *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*. [S.l.], 2006. p. 1–4.

[Menasce et al. 2004]MENASCE, D. et al. *Performance by design: computer capacity planning by example*. [S.l.]: Prentice Hall, 2004.

[Menascé et al. 2000]MENASCÉ, D. et al. In search of invariants for e-business workloads. In: ACM. *Proceedings of the 2nd ACM conference on Electronic commerce*. [S.l.], 2000. p. 56–65.

[Menascé et al. 2003]MENASCÉ, D. et al. A hierarchical and multiscale approach to analyze e-business workloads. *Performance Evaluation*, Elsevier, v. 54, n. 1, p. 33–57, 2003.

[Menascé, Barbará e Dodge 2001]MENASCÉ, D.; BARBARÁ, D.; DODGE, R. Preserving qos of e-commerce sites through self-tuning: A performance model approach. In: ACM. *Proceedings of the 3rd ACM conference on Electronic Commerce*. [S.l.], 2001. p. 224–234.

[Menascé e Ngo 2009]MENASCÉ, D.; NGO, P. Understanding cloud computing: Experimentation and capacity planning. In: *2009 Computer Measurement Group Conference*. [S.l.: s.n.], 2009. p. 11.

[Moura, Sauvé e Bartolini 2007]MOURA, A.; SAUVÉ, J.; BARTOLINI, C. Research challenges of business-driven it management. In: IEEE. *Business-Driven IT Management, 2007. BDIM'07. 2nd IEEE/IFIP International Workshop on*. [S.l.], 2007. p. 19–28.

[Namjoshi e Gupte 2009]NAMJOSHI, J.; GUPTE, A. Service Oriented Architecture for Cloud Based Travel Reservation Software as a Service. *2009 IEEE International Conference on Cloud Computing*, Ieee, p. 147–150, set. 2009.

[Ranjan et al. 2002]RANJAN, S. et al. Qos-driven server migration for internet data centers. In: IEEE. *Quality of Service, 2002. Tenth IEEE International Workshop on*. [S.l.], 2002. p. 3–12.

[Sääksjärvi, Lassila e Nordström 2005]SÄÄKSJÄRVI, M.; LASSILA, A.; NORDSTRÖM, H. Evaluating the software as a service business model: From cpu time-sharing to online innovation sharing. In: *Proceedings of the IADIS International Conference e-Society*. [S.l.: s.n.], 2005. p. 177–186.

[Sargent 2005]SARGENT, R. Verification and validation of simulation models. In: WINTER SIMULATION CONFERENCE. *Proceedings of the 37th conference on Winter simulation*. [S.l.], 2005. p. 130–143.

[Sauve et al. 2006]SAUVE, J. et al. Optimal design of e-commerce site infrastructure from a business perspective. In: IEEE. *System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on*. [S.l.], 2006. v. 8, p. 178c–178c.

[Sauvé et al. 2006]SAUVÉ, J. et al. An introductory overview and survey of business-driven it management. In: IEEE. *Business-Driven IT Management, 2006. BDIM'06. The First IEEE/IFIP International Workshop on*. [S.l.], 2006. p. 1–10.

[Shapiro e Wilk 1965]SHAPIRO, S.; WILK, M. An analysis of variance test for normality (complete samples). *Biometrika*, JSTOR, v. 52, n. 3/4, p. 591–611, 1965.

[Shewhart 1939]SHEWHART, W. *Statistical method from the viewpoint of quality control*. [S.l.]: Dover Publications, 1939.

[Stage, Setzer e Bichler 2009]STAGE, A.; SETZER, T.; BICHLER, M. Automated capacity management and selection of infrastructure-as-a-service providers. In: IEEE. *Integrated Network Management-Workshops, 2009. IM'09. IFIP/IEEE International Symposium on*. [S.l.], 2009. p. 20–23.

[Strunk et al. 2008]STRUNK, J. et al. Using utility to provision storage systems. In: USENIX ASSOCIATION. *Proceedings of the 6th USENIX Conference on File and Storage Technologies*. [S.l.], 2008. p. 21.

[Tak, Urgaonkar e Sivasubramaniam 2011]TAK, B.; URGAONKAR, B.; SIVASUBRAMANIAM, A. To move or not to move: The economics of cloud computing. *Proceedings of the Third USENIX Workshop on Hot Topics in Cloud Computing (HOTCLOUD 2011)*, p. 1–5, 2011.

[UEMA e LAZZARI 2008]UEMA, E.; LAZZARI, C. D. O crescimento do e-commerce no brasil: estudo de caso do:"submarino". *Revista Científica da Faculdade das Américas, Ano II*, n. 2, p. 3, 2008.

[Urgaonkar et al. 2005]URGAONKAR, B. et al. An analytical model for multi-tier internet services and its applications. In: ACM. *ACM SIGMETRICS Performance Evaluation Review*. [S.l.], 2005. v. 33, n. 1, p. 291–302.

[Urgaonkar et al. 2008]URGAONKAR, B. et al. Agile dynamic provisioning of multi-tier internet applications. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, ACM, v. 3, n. 1, p. 1, 2008.

[Vaquero et al. 2008]VAQUERO, L. et al. A break in the clouds: towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, ACM, v. 39, n. 1, p. 50–55, 2008.

[Vogels 2008]VOGELS, W. Head in the clouds—the power of infrastructure as a service. In: *First workshop on Cloud Computing and in Applications (CCA'08)(October 2008)*. [S.l.: s.n.], 2008.

[Wilkes 2008]WILKES, J. Utility functions, prices, and negotiation. *Market-Oriented Grid and Utility Computing*, Wiley Online Library, p. 67–88, 2008.

[Woolley 2000]WOOLLEY, R. Web performance measurement & capacity planning: Briefing paper. *State Information Architect*, v. 1, 2000.

[Wu, Garg e Buyya 2011]WU, L.; GARG, S.; BUYYA, R. Sla-based resource allocation for software as a service provider (saas) in cloud computing environments. In: IEEE. *Cluster, Cloud and Grid Computing (CCGrid), 2011 11th IEEE/ACM International Symposium on*. [S.l.], 2011. p. 195–204.

[Zhang, Cherkasova e Smirni 2007]ZHANG, Q.; CHERKASOVA, L.; SMIRNI, E. A regression-based analytic model for dynamic resource provisioning of multi-tier applications. In: IEEE. *Autonomic Computing, 2007. ICAC'07. Fourth International Conference on*. [S.l.], 2007. p. 27–27.

[Zhu et al. 2011]ZHU, Z. et al. Sla based dynamic virtualized resources provisioning for shared cloud data centers. In: IEEE. *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. [S.l.], 2011. p. 630–637.