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

Tese de Doutorado

Provisionamento Automático de Recursos como um Serviço de IaaS

Fábio Jorge Almeida Morais

Campina Grande, Paraíba, Brasil
Agosto de 2017

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

Provisionamento Automático de Recursos como um Serviço de IaaS

Fábio Jorge Almeida Morais

Tese 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 Doutor em Ciência da Computação.

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

Francisco Vilar Brasileiro e Raquel Vigolvino Lopes
(Orientadores)

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

M827p Morais, Fábio Jorge Almeida.
Provisionamento automático de recursos como um serviço de IaaS / Fábio Jorge Almeida Morais. – Campina Grande, 2017.
163 f.: il. color.

Tese (Doutorado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2017.
"Orientação: Prof. Dr. Francisco Vilar Brasileiro, Profª. Drª. Raquel Vigolvino Lopes".
Referências.

1. Computação na Nuvem. 2. Infraestrutura como um serviço. 3. Gerência de Capacidade. 4. Provisionamento Automático de Recursos. I. Brasileiro, Francisco Vilar. II. Lopes, Raquel Vigolvino. III. Título.

CDU 004.738.5(043)

VISIONAMENTO AUTOMÁTICO DE RECURSOS COMO UM SERVIÇO DE IAAS"

FÁBIO JORGE ALMEIDA MORAIS

TESE APROVADA EM 16/08/2017

FRANCISCO VILAR BRASILEIRO, Ph.D, UFCG
Orientador(a)

RAQUEL VIGOLVINO LOPES, Dra., UFCG
Orientador(a)

ANDREY ELÍSIO MONTEIRO BRITO, Dr., UFCG
Examinador(a)

LÍVIA MARIA RODRIGUES SAMPAIO CAMPOS , Dra., UFCG
Examinador(a)

JOSÉ NEUMAN DE SOUZA, Dr., UFC
Examinador(a)

CESAR AUGUSTO FONTICIELHA DE ROSE, Dr., PUC-RS
Examinador(a)

CAMPINA GRANDE - PB

Resumo

O modelo de IaaS proporcionado pelo paradigma de Computação na Nuvem tem como principais características a provisão sob demanda de recursos e a tarifação do uso de recursos a partir de um modelo *pay-as-you-go*, que permitem que o custo de utilização do serviço seja proporcional à quantidade e ao tempo de uso dos recursos. Essas características possibilitam a criação de infraestruturas virtuais elásticas, que podem ser dinamicamente modificadas, em termos da capacidade de recursos, a fim de acomodar as demandas da aplicação que nela executa. Tal elasticidade é principalmente explorada para o provisionamento de aplicações horizontalmente escaláveis, que possuem demandas variáveis no tempo e executam por longos períodos. Idealmente, para aplicações desse tipo, a capacidade da infraestrutura de execução pode ser automaticamente provisionada com base nas demandas da aplicação, de forma a assegurar a QoS da aplicação e ao mesmo tempo minimizar os custos de execução em termos dos recursos adquiridos. Esse cenário de provisionamento automático pode ser expandido para o desenvolvimento de um serviço de provisionamento automático de recursos em IaaS. Desta forma, o responsável pela aplicação pode contratar um serviço que assuma a responsabilidade de dinâmica e eficientemente provisionar a sua aplicação durante a execução desta. No entanto, por questões de privacidade e principalmente generalidade em termos das aplicações provisionadas, espera-se que um serviço desse tipo opere com informações não específicas da aplicação, tais como utilização de CPU, memória, etc., ou seja, de forma não intrusiva. Este trabalho visa investigar a tese sobre a viabilidade de construção de um serviço de provisionamento automático e não intrusivo para diferentes aplicações horizontalmente escaláveis em um ambiente de IaaS. Tal serviço deve ser capaz de manter a QoS da aplicação provisionada em níveis aceitáveis e, havendo variação de carga de trabalho, minimizar os custos de sua execução. Em geral, as atuais soluções de provisionamento automático fazem uso de abordagens de provisionamento que operam de forma reativa ou proativa. Desta forma, o principal objetivo desse trabalho consiste em analisar como soluções de provisionamento, reativas e proativas, podem ser empregadas na construção de um serviço de provisionamento em IaaS, destacando eficiências e limitações destas abordagens e apontando diretrizes para a criação desse serviço.

Abstract

The IaaS model, provided by the Cloud Computing paradigm, is defined by two main features: the on-demand provision of resources; and the pay-as-you-go pricing model, that allows application providers to pay proportionally to the quantity and time of use of the acquired resources. These features are used to build elastic virtual infrastructures, which can have its resources capacity dynamically modified to accommodate demands' fluctuations of the running application. Such elasticity is mainly exploited for running horizontally scalable applications, that executes over a long period of time and have time-varying workloads. Ideally, for such applications, the capacity of the execution infrastructure can be automatically provisioned based on the application demands, to ensure the application QoS and at the same time minimize the execution costs, in terms of the acquired resources. This provisioning context can be expanded to conceive a scenario of auto scaling as a service in IaaS. In this way, the application owner can contract a service that assumes the responsibility of dynamically and efficiently provisioning the application during its execution. However, due to privacy and mainly generality issues in terms of the provisioned applications, this service needs to operate with non-application-specific information, such as CPU utilization, memory, etc., i.e., in a non-intrusive way. This work aims to investigate the thesis on the construction feasibility of a non-intrusive auto scaling service for different horizontally scalable applications in an IaaS environment. Such service must be able to keep the QoS of the provisioned application at acceptable levels and, if there are workload variations, minimize the execution costs. In general, current provisioning solutions use provisioning approaches that operate in a reactive or proactive manner. Thus, the main objective of this work is to analyze how reactive and proactive auto scaling solutions can be used to basis an auto scaling service in IaaS, describing the efficiencies and limitations of these approaches and pointing out guidelines for the service construction.

Agradecimentos

Primeiramente, gostaria de agradecer a minha esposa, Priscylla Lucena, cujo suporte e compreensão foram imprescindíveis para a conclusão desse percurso, e aos meus filhos Alice, que por tantas vezes deu-me forças para seguir sem ao menos se dar conta disto, e Heitor, que mesmo sem ainda estar em meus braços é parte significativa de minha força. Além disso, não poderia deixar de agradecer aos meus pais, Edilson e Nevinha, que sempre estiveram ao meu lado apoiando minhas escolhas, apesar de todas as divergências em torno de minhas convicções. Para os meus orientadores, Fubica e Raquel Lopes, também torno explícito meus agradecimentos, com os quais muito aprendi e cujos traços hoje enxergo em mim, tanto no profissional como na pessoa. Obrigado aos meus amigos e aos amigões que de diferentes formas foram mais que ouvidos a exaurir minhas preocupações, também foram conselheiros e companheiros de jornada a dividir uma cerveja independentemente do lugar. Minha infinita gratidão ao povo brasileiro, que mesmo sem se dar conta proveu os subsídios financeiros para minha formação, sem os quais seria impossível a conclusão desse trabalho. Por fim, aos companheiros e colegas de LSD, com os quais dividi inúmeras horas ao longo desses anos, alegres e tristes, talvez muito mais do que as horas compartilhadas com minha própria família. Novamente, obrigado a todos.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Nomenclatura

AC *Auto-Correlation*

ACF *Auto-Correlation Function*

AR *Auto-Regressive*

ARIMA *Auto-Regressive Integrated Moving Average*

ARMA *Auto-Regressive Moving Average*

AWS *Amazon Web Services*

ECU *EC2 Compute Unite*

EN *Ensemble*

IaaS *Infrastructure as a Service*

KMP *Knuth-Morris-Pratt*

LR *Linear Regression*

LW *Last Window*

MA *Moving Average*

PaaS *Platform as a Service*

PCS *Processor Clock Speed*

PM *Pattern Matching*

QoS *Quality of Service*

QT *Queueing Theory*

RL *Reinforcement Learning*

RPUR Resource Proporcionality Utilization Ratio

SaaS *Software as a Service*

SLA *Service Level Agreement*

SLO *Service Level Objective*

TCP *Transmission Control Protocol*

TI Tecnologia da Informação

TS *Time Series*

vCPU *Virtual Central Processing Unit*

VM *Virtual Machine*

Capítulo 1

Introdução

Este capítulo oferece uma visão geral da tese, indicando o contexto no qual ela se insere, motivações, escopo, relevância, contribuições e objetivos perseguidos. Por fim, também é apresentada a estrutura deste documento. Este trabalho de tese visa investigar como técnicas de provisionamento automático de recursos podem ser utilizadas para a construção de um serviço não intrusivo de provisionamento automático de aplicações em ambientes de IaaS e Computação na Nuvem. Este serviço de provisionamento deve minimizar o custo de execução das aplicações sempre que possível, porém sem degradar o desempenho das mesmas.

1.1 Contextualização e Escopo

O paradigma de Computação na Nuvem consolidou-se nos últimos anos no cenário global de Tecnologia da Informação (TI) [5], proporcionando flexibilidade e escalabilidade na provisão e na utilização de serviços computacionais. Estima-se um crescimento significativo do mercado de Computação na Nuvem para o ano de 2017, com uma desaceleração limitada de investimentos prevista para os próximos cinco anos [26]. Dada a crescente demanda desse mercado, e o natural aumento na heterogeneidade dessas demandas, os provedores de Computação na Nuvem precisam oferecer diferentes modelos para a aquisição de seus serviços computacionais, com o intuito de atender as diferentes necessidades dos seus clientes.

Um modelo de Computação na Nuvem oferecido atualmente, bastante popular, consiste na oferta de infraestruturas como serviço (IaaS, do inglês *Infrastructure as a Service*), que em geral são apresentadas na forma de máquinas virtuais (VM, do inglês *Virtual Machine*)

implantadas em centros de processamento de dados dos provedores de Computação na Nuvem [60]. Na prática, este tipo de serviço consiste na obtenção de recursos computacionais de um provedor e na utilização destes recursos para implantação e execução de aplicações de interesse [59]. Nesse cenário, o usuário do serviço de IaaS não gerencia ou controla a infraestrutura de Nuvem, mas possui o controle sobre sistemas operacionais, armazenamento e aplicações implantadas em VMs adquiridas [41].

As duas principais características do modelo de IaaS consistem na elasticidade de oferta de recursos e no modelo de tarifação dos recursos adquiridos. A elasticidade é a propriedade que permite a provisão sob demanda de recursos computacionais para uma dada aplicação [1, 34]. Graças a esta propriedade, uma aplicação pode requisitar diferentes quantidades de recursos para suprir variações em sua carga de trabalho ao longo do tempo. Para tal, os clientes de IaaS podem adquirir e liberar recursos em curtos períodos de tempo com o intuito de refletir as demandas de suas aplicações em execução na infraestrutura virtual adquirida.

O modelo de tarifação comumente adotado em IaaS permite que o serviço seja "pago conforme utilização" (do inglês *pay-as-you-go*), onde os clientes pagam apenas pelos recursos de fato adquiridos [6]. Esse modelo permite que o custo da infraestrutura virtual adquirida seja proporcional ao tamanho e ao tempo de uso desta infraestrutura. Na prática, essas infraestruturas são compostas por um conjunto de VMs, cujos tipos são previamente definidos e ofertados pelo provedor de IaaS [28]. Normalmente, cada tipo de VM ou instância oferecido pelo provedor é definido em termos de capacidade de recursos computacionais (CPU, memória RAM, disco, etc.) e tarifado com base no período mínimo de uso, que geralmente considera unidades inteiras de tempo (por exemplo, 1 hora), e na capacidade do tipo da VM.

Dadas essas características, os ambientes de IaaS são comumente explorados para a execução de aplicações horizontalmente escaláveis, cujo desempenho pode ser controlado pela quantidade de unidades computacionais que executam a aplicação. Tipicamente, tais aplicações executam por longos períodos de tempo e possuem cargas de trabalho que podem variar ao longo do tempo, o que potencializa o uso de IaaS para a execução desse tipo de aplicação. Desta forma, ao definir a infraestrutura a ser utilizada para executar uma determinada aplicação com essas características, é preciso decidir criteriosamente a quantidade e a configuração das VMs necessárias para manter o custo de provisionamento baixo e ao mesmo tempo suprir adequadamente as demandas por recursos da aplicação em diferentes dimen-

sões (por exemplo recursos de CPU, memória, disco, etc.), mantendo assim seu desempenho e qualidade de serviço (QoS, do inglês *Quality of Service*) em níveis aceitáveis.

Quando a capacidade não é adequadamente definida, podem ocorrer cenários de super ou sub provisionamento. Nos cenários de super provisionamento existe um excedente de recursos e uma consequente elevação nos custos de execução. Já nos cenários de sub provisionamento, o custo é reduzido, porém a quantidade de recursos alocados não é suficiente para executar eficientemente a aplicação. Nesse último caso a aplicação opera com níveis de QoS abaixo do ideal, o que pode impactar o processo de negócio suportado pela aplicação. Ainda, dependendo do tipo de serviço provido pela aplicação, baixos níveis de QoS podem levar à perda de clientes e de operações de clientes devido ao aumento dos tempos de resposta e do número de requisições não respondidas. Além do mais, acredita-se que há uma relação entre o sub provisionamento e perdas no nível do negócio suportado pela aplicação que podem impactar o valor de mercado do serviço provido pela aplicação no longo prazo. Desta forma, é até possível que prejuízos monetários diretos devido ao desperdício de recursos em decorrência de cenários de super provisionamento sejam menos prejudiciais do que as perdas indiretas provenientes da degradação do desempenho da aplicação. Portanto, faz-se necessária alguma abordagem para o provisionamento dinâmico da aplicação, com o intuito de evitar cenários de sub provisionamento porém visando também minimizar o custo de execução da aplicação.

Apesar da flexibilidade oferecida pelos ambientes de IaaS, garantir a execução eficiente de uma aplicação escalável horizontalmente sobre uma infraestrutura elástica não é uma tarefa trivial. Mesmo assumindo-se que o tipo de instância de VM usado para executar a aplicação é previamente definido, uma solução eficiente de provisionamento dinâmico ou automático de recursos em tais ambientes deve ser capaz de: (i) conhecer a capacidade mínima de recursos exigida pela aplicação em um futuro próximo para as diferentes dimensões de recursos; e (ii) decidir quantas instâncias do tipo usado no provisionamento são necessárias para atender às demandas da aplicação no curto prazo. Além do mais, todas essas decisões precisam ser periodicamente reavaliadas para lidar com a variabilidade da carga de trabalho da aplicação provisionada ao longo do tempo. O fato de os provedores de IaaS oferecerem múltiplos tipos de instância de VM só contribui para tornar mais complexa a solução desse problema.

Essa tarefa torna-se ainda mais complexa em um cenário em que o *provisionamento automático de recursos é oferecido como um serviço de IaaS*. Nesse cenário de serviço é possível explorar a elasticidade oferecida por ambientes de IaaS para execução de aplicações horizontalmente escaláveis, de forma que o responsável pela aplicação possa contratar um serviço que assuma a responsabilidade de dinâmica e eficientemente provisionar a sua aplicação durante a execução desta. Para esse tipo de serviço, além de responsabilidades como minimizar custo do provisionamento e manter a QoS da aplicação em um nível aceitável, existem outras relacionadas à privacidade e principalmente à generalidade e escalabilidade.

Desta forma, espera-se que um serviço nesses moldes opere com informações não específicas da aplicação em execução, que possam ser coletadas no nível da infraestrutura de execução, tais como utilização de CPU, memória, etc. O uso de informações não específicas das aplicações, obtidas no nível da infraestrutura, permite ao serviço de provisionamento principalmente generalidade e desacoplamento das aplicações por ele provisionadas. Além disso, é comum que informações específicas da aplicação como taxa de chegada de requisições, tipos de requisições, tamanhos de filas, etc. sejam consideradas sensíveis, não sendo possível compartilhá-las com terceiros, e por isso inviáveis para o uso em um cenário de provisionamento automático como um serviço.

As atuais soluções de provisionamento automático que potencialmente podem ser usadas para compor um serviço de provisionamento, em geral, seguem um dos seguintes modos de operação: reativo ou proativo. A técnica reativa consiste em uma reação programável a mudanças percebidas no sistema provisionado, que corresponde a aplicação em execução e/ou a sua infraestrutura de execução. Particularmente, essa abordagem utiliza um conjunto de regras de provisionamento para decidir quando e em qual quantidade de recursos a aplicação deve ser provisionada [38]. O provisionamento reativo utiliza apenas informações sobre o estado atual da aplicação e do seu ambiente para decidir sobre o provisionamento da aplicação no curto prazo. No entanto, apesar de serem as soluções de provisionamento mais comuns, tendo em vista a sua simplicidade e natureza intuitiva, acredita-se que abordagens reativas não sejam eficientes ao prover aplicações com cargas de trabalho de intensa variabilidade no tempo [21, 38], que são o objeto do estudo de provisionamento deste trabalho.

Esse suposto decorre da natureza reativa e pontual da solução e do fato da configuração das regras de provisionamento serem consideravelmente sensíveis a mudanças e tendências

da carga de trabalho da aplicação, que geram a necessidade de ajustes frequentes mesmo na presença de um especialista na aplicação atuando no processo de configuração [38]. Desta forma, prováveis equívocos na configuração das regras podem provocar situações indesejáveis de sub provisionamento que levam à degradação de QoS da aplicação e possíveis violações de SLO (do inglês, *Service Level Objective*), ou de super provisionamento, com o desperdício de recursos adquiridos do provedor de IaaS e elevação do custo de execução da aplicação. Outro ponto negativo é que as soluções reativas exploradas tanto pelo mercado quanto pela academia necessitam, em geral, de métricas intrusivas específicas da aplicação que não podem ser consideradas por um serviço de provisionamento genérico e automático em IaaS.

Apesar de ser criticada em alguns aspectos, a abordagem reativa ainda é significativamente explorada [38]. Os principais provedores do mercado de Computação na Nuvem e IaaS, como Amazon Web Services (AWS) [3], Rackspace [51] e Microsoft Azure [7], oferecem serviços de provisionamento automático e reativo de recursos[1]. Esta abordagem também é amplamente explorada na literatura através do uso de diferentes conjuntos de métricas de desempenho predominantemente intrusivas, configurações de limiares e ações de provisionamento [9, 11, 12, 23, 27, 35, 40, 55]. Entretanto, até onde se sabe, a literatura é carente de estudos sobre o desempenho de técnicas reativas de provisionamento que atuam com base em métricas não intrusivas aplicadas em um cenário de provisionamento automático de aplicações como um serviço de IaaS.

Por outro lado, o modo de operação proativo busca superar o caráter imediatista da abordagem reativa através da antecipação de mudanças nas características da aplicação por meio de estimativas de demandas futuras com base no histórico de sua carga de trabalho. Desta forma, essas estimativas são utilizadas para a tomada de decisões antecipadas sobre a capacidade da infraestrutura, que desta forma pode ser previamente preparada para acomodar demandas estimadas para um futuro próximo [24]. No contexto do provisionamento de aplicações com variações intensas de carga de trabalho, considera-se que as abordagens proativas são as mais promissoras para efetuar eficientemente o provisionamento automático

[1] A Google [30] oferece um serviço de provisionamento automático que também baseia-se na técnica reativa, mas utiliza um modelo de provisionamento mais sofisticado, não conhecido pelo público, para decidir a quantidade de VMs que devem ser provisionadas a cada intervalo de tempo.

dessas aplicações [38]. Todavia, estimar a demanda futura das aplicações provisionadas não é uma tarefa trivial.

Dentre os diversos métodos com abordagens proativas propostos na literatura para o provisionamento automático de aplicações horizontalmente escaláveis, são predominantes as soluções que fazem uso de métricas específicas da aplicação e são consequentemente tidas como intrusivas [2, 11, 15, 35, 40, 52, 56, 61, 62, 64]. Uma pequena parcela das soluções propostas opera com métricas não intrusivas à aplicação [14, 44], que parece ser a opção mais viável para um serviço de provisionamento oferecido aos clientes de IaaS. Todavia, o desempenho dessas soluções não intrusivas não foi avaliado em um cenário de provisionamento automático como um serviço em IaaS, que considera outros aspectos além do custo de provisionamento e da QoS da aplicação em execução.

Assim, fica evidente a necessidade de um estudo que avalie de forma profunda o desempenho de soluções de provisionamento, reativas e proativas, que utilizam apenas métricas de uso da infraestrutura de execução e, portanto, podem ser utilizadas para implementar um serviço de provisionamento automático em IaaS. Um serviço de provisionamento nesses moldes permite que infraestruturas elásticas oferecidas pelo já consolidado mercado de IaaS e Computação na Nuvem possam ser utilizadas para executar eficientemente aplicações horizontalmente escaláveis. Desta forma, esse estudo pretende avaliar o desempenho de soluções de provisionamento em termos do custo de execução e do nível de QoS da aplicação provisionada, além de analisar o desempenho destas soluções em relação à aplicabilidade da técnica empregada no cenário de provisionamento como um serviço. Como resultado, serão indicadas diretrizes para criação de um serviço de provisionamento que opere de forma desacoplada das aplicações provisionadas por meio de métricas não intrusivas, ao mesmo tempo que não compromete requisitos de segurança e privacidade das aplicações.

Além do mais, independentemente da solução de provisionamento considerada, o processo de decisão do tipo de instância de VM que deve ser usado no provisionamento também não é devidamente abordado pela literatura. Em geral, o foco principal tem sido em suprir as necessidades da aplicação em execução, independente do tipo de instância considerado no provisionamento. Por este motivo, esse estudo também planeja investigar o uso de múltiplos tipos de instância no provisionamento automático com o objetivo de reduzir custos de execução em ambientes de IaaS.

Desta forma, o escopo deste trabalho limita-se ao estudo de soluções de provisionamento automático, reativo e proativo, empregadas na construção de um serviço para o provisionamento automático de aplicações horizontalmente escaláveis em ambientes de IaaS. Tais soluções apresentam-se como potenciais serviços de provisionamento automático contratados pelo proprietário da aplicação para provisionar de forma automática a infraestrutura utilizada para executá-la. Para o provedor de IaaS, o servidor de provisionamento assume o papel de procurador do proprietário da aplicação, e consequentemente de cliente do provedor. Ou seja, ao serviço de provisionamento é atribuída a responsabilidade pela gerência automática da infraestrutura de execução da aplicação em questão, que a realiza por meio da aquisição e liberação dinâmica de recursos virtuais junto ao provedor de Computação na Nuvem e IaaS. Além do mais, também inclui-se nesse escopo o estudo sobre como os recursos são provisionados e como o custo de execução é impactado a partir do uso de múltiplos tipos de instância de VM no provisionamento automático como um serviço.

1.2 Objetivos

O objetivo principal do trabalho consiste em analisar como soluções de provisionamento automático, reativas e proativas, podem ser empregadas na construção de um serviço eficiente de provisionamento automático em ambientes de IaaS que opere de forma não intrusiva. Essa eficiência baseia-se tanto no desempenho do serviço em termos dos objetivos de provisionamento, que consiste na minimização de custos de execução e manutenção da QoS da aplicação em níveis aceitáveis, quanto na sua generalidade de emprego, facilidade de configuração, complexidade de implementação e controle do *trade-off* entre objetivos de provisionamento. A partir desse estudo, pretende-se por à prova a tese sobre a viabilidade de construção de um serviço de provisionamento automático e não intrusivo para diferentes aplicações horizontalmente escaláveis. Este serviço deve ser capaz de manter a QoS da aplicação provisionada em níveis aceitáveis e, havendo variação de carga de trabalho, minimizar os custos de execução da aplicação. Tendo em vista esse objetivo principal, são considerados os seguintes objetivos específicos:

1. Descrever em detalhes a problemática e o cenário de provisionamento automático como um serviço, que atue de forma não intrusiva à aplicação provisionada. Nesta des-

crição deve-se destacar os atores que fazem parte desse cenário de provisionamento, incluindo as relações de prestação de serviço entre os atores e as obrigações impostas a cada um em decorrência destas relações;

2. Avaliar a aplicabilidade e eficiência das técnicas reativas e proativas de provisionamento quando empregadas no cenário de provisionamento automático como um serviço de IaaS para diferentes métricas de provisionamento não intrusivas. Neste contexto, pode-se considerar apenas uma ou múltiplas dimensões de recursos consumidos pela aplicação. Essa avaliação explora tanto a eficiência da técnica em atingir os objetivos de provisionamento quanto a sua capacidade em compor um serviço de provisionamento genérico e não dependente de características específicas da carga de trabalho das aplicações;

3. Investigar a possibilidade de uso de diferentes tipos de VMs durante o provisionamento automático de aplicações com o intuito de reduzir os custos de provisionamento;

4. Destacar pontos de eficiência e limitações das técnicas de provisionamento exploradas nesse estudo e propor diretrizes, fundamentadas nas análises realizadas, que auxiliem a construção de um serviço de provisionamento como descrito neste documento.

1.3 Contribuições

No tocante aos objetivos anteriormente destacados, a principal contribuição do trabalho consiste na evolução do estado da arte quanto à construção de um serviço de provisionamento automático, baseado em métricas não intrusivas de provisionamento de aplicações em ambientes de IaaS a partir de abordagens de provisionamento reativas e proativas. Com um serviço de provisionamento automático nesse formato é possível intensificar o uso da elasticidade oferecida por ambientes de IaaS para a execução eficiente de aplicações horizontalmente escaláveis. Desta forma, esse estudo analisa as particularidades de cada abordagem para esse contexto de serviço em IaaS, principalmente em termos de eficiência e limitações das técnicas para atingir os objetivos de provisionamento e em relação a capacidade destas de compor um serviço de provisionamento genérico e não dependente de características específicas das aplicações e suas cargas de trabalho. Em detalhes, as principais características

analisadas das soluções de provisionamento automático são as seguintes:

- A capacidade de cumprir os **objetivos de provisionamento**, que consistem na manutenção da QoS da aplicação provisionada em níveis aceitáveis e na eficiência dos custos de provisionamento;
- O grau de **eficiência de implementação e configuração** da solução de provisionamento automático para servir em um ambiente de IaaS;
- O nível de **generalidade e independência** de características específicas das aplicações provisionadas, especialmente em relação à carga de trabalho das aplicações e métricas de provisionamento;
- O **controle de objetivos de provisionamento** da solução de provisionamento automático, que denota a capacidade de se explorar o espaço de configurações do serviço de provisionamento para obter diferentes níveis de desempenho em termos do *trade-off* entre métricas finais de custo de execução e nível de QoS da aplicação.

1.4 Organização do Documento

O restante deste documento encontra-se organizado da seguinte forma. No Capítulo 2 os conceitos e definições que envolvem o estudo são fundamentados, o problema em torno do provisionamento automático como um serviço de IaaS abordado por este trabalho é definido e as questões tratadas por esta pesquisa são descritas. Em seguida, no Capítulo 3, uma revisão da literatura é realizada, no que concerne as características de soluções de provisionamento de recursos em ambientes de IaaS, a fim de ter-se o embasamento teórico necessário para a construção de um serviço de provisionamento automático de recursos nos moldes do que fora anteriormente definido. No Capítulo 4 a metodologia utilizada no desenvolvimento desse estudo é descrita, juntamente com uma caracterização dos dados usados para a realização do estudo.

Os estudos sobre o desempenho das soluções de provisionamento automático, reativo e proativo, quanto à satisfação dos objetivos de provisionamento e a capacidade de constituir um serviço de provisionamento automático como definido neste documento, encontram-se

respectivamente nos Capítulos 5 e 6. Em seguida é apresentado no Capítulo 7 um estudo sobre a potencialização do uso dos recursos alocados para executar a aplicação através do uso de múltiplos tipos de instância de VM durante o provisionamento de aplicações. Por fim, no Capítulo 8 é apresentada uma discussão sobre as implicações inerentes à construção de um serviço de provisionamento automático em ambientes de IaaS a partir de técnicas de provisionamento reativo e proativo, além das conclusões decorrentes desse trabalho juntamente com um direcionamento para trabalhos futuros nessa área de pesquisa.

Capítulo 2

Problema Investigado

Nesse capítulo os conceitos e definições dos temas envolvidos nesta pesquisa são fundamentados. O problema do provisionamento automático não intrusivo como um serviço em IaaS é descrito com o objetivo de delimitar o objeto de estudo desse trabalho. Finalmente, as questões de pesquisa extraídas da problemática exposta são definidas com o intuito de guiar esse estudo.

2.1 Fundamentação Teórica

2.1.1 Infraestrutura como serviço

Infraestrutura como Serviço é um nível de serviço oferecido no paradigma de Computação na Nuvem, tal como *Software* como Serviço (SaaS, *Software as a Service*), Plataforma como Serviço (PaaS, do inglês *Platform as a Service*), dentre outros [41]. IaaS caracteriza-se por oferecer recursos computacionais (poder de processamento, memória, disco, etc.) como um serviço, através da disponibilização imediata de recursos virtualizados, da tarifação baseada na aquisição e consumo de recursos e da ausência de comprometimentos futuros entre cliente e provedor, ou seja, não existe necessariamente duração preestabelecida para a relação de prestação de serviço.

Desta forma, ao utilizar esse serviço o cliente delega ao provedor, por exemplo, as obrigações de compra de servidores físicos e equipamentos e de operação da infraestrutura física, como, gerenciamento de espaço físico, de energia elétrica, do sistema de refrigeração, etc.

Do ponto de vista do cliente, os atuais modelos de IaaS destacam-se principalmente pela elasticidade (poder de redimensionamento de capacidade da infraestrutura contratada) e flexibilidade (diversidade de tipos de instância e sistemas operacionais oferecidos).

IaaS é oferecido através de infraestruturas virtualizadas — máquinas virtuais ou contêineres — que são hospedadas nos servidores do centro de dados do provedor de Computação na Nuvem e que são utilizadas para executar os serviços e aplicações dos clientes do provedor. O provedor define um conjunto de tipos de instâncias de máquinas virtuais que podem ser oferecidas aos clientes do provedor. Cada tipo caracteriza uma máquina virtual em termos de sua capacidade de recursos (CPU, memória RAM, disco, e assim por diante) e possui um preço por período mínimo de uso associado (período de tarifação, do inglês *billing cycle*), que em geral é 1 hora. O valor por período de tarifação da uma instância de um certo tipo é proporcional à quantidade de recursos disponibilizada pelo tipo.

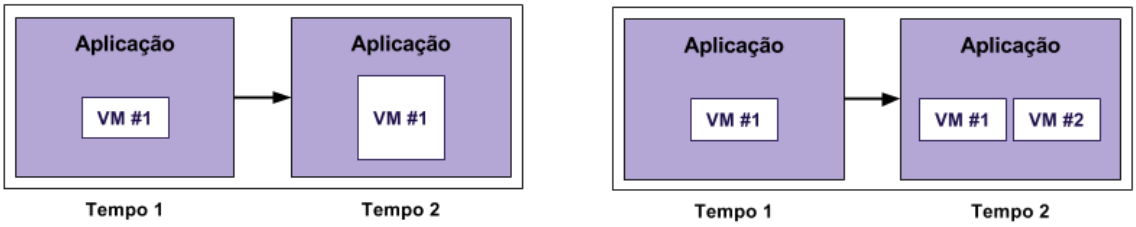
A relação de prestação de serviço estabelecida entre o cliente e o provedor de IaaS é mediada através de um contrato de nível de serviço (SLA, do inglês *Service Level Agreement*) que é responsável por especificar as expectativas do cliente e definir o serviço prestado pelo provedor de Computação na Nuvem [10]. O SLA firmado garante que a disponibilidade, escalabilidade, confiabilidade e segurança do serviço prestado serão mantidas, sendo este composto por objetivos de nível de serviço (SLO, do inglês *Service Level Objective*) que estabelecem os critérios a serem atendidos para assegurar o cumprimento do SLA. Além do mais, é normal que o SLA também defina as penalidades impostas pelo não cumprimento do contrato por qualquer uma das partes.

2.1.2 Provisionamento de recursos a curto prazo

O provisionamento de recursos a curto prazo em ambientes de IaaS preocupa-se com o provisionamento de recursos em um horizonte de alguns minutos. Desta forma, a carga de trabalho de uma aplicação deve ditar a quantidade de recursos necessários para suprir a própria aplicação no próximo horizonte de tempo. Essa carga de trabalho pode ser caracterizada a partir de uma variedade de métricas, como taxa de utilização de recursos, tempos de resposta, tamanhos de filas, dentre outras. As métricas consideradas determinam o nível de intrusão da abordagem de provisionamento. Dado o cenário de provisionamento como um serviço, o foco deste trabalho é em técnicas de provisionamento não intrusivas, que consideram apenas

os dados de utilização dos recursos virtuais obtidos no nível da infraestrutura.

Essencialmente, o provisionamento a curto prazo pode ser realizado de duas formas, como mostrado na Figura 2.1, que são: (a) **vertical**, quando variações na carga de trabalho da aplicação são supridas modificando-se a capacidade das máquinas virtuais que estão executando a aplicação. A modificação da capacidade da VM se dá através do redimensionamento desta em termos de capacidade de recursos; (b) **horizontal**, quando a quantidade de máquinas virtuais alocadas é alterada para adaptar a infraestrutura à carga da aplicação. Nesse caso ocorre um redimensionamento da infraestrutura virtual por meio da aquisição ou liberação de máquinas virtuais. Quando mais de uma VM serve a aplicação é evidente a necessidade de um serviço de balanceamento de carga para distribuir e equilibrar a carga de trabalho da aplicação entre todas as máquinas virtuais alocadas para servi-la.



(a) Elasticidade vertical

(b) Elasticidade horizontal

Figura 2.1: Diferentes tipos de provisionamento de recursos.

O cenário de provisionamento como serviço abordado neste trabalho considera a execução de aplicações horizontalmente escaláveis em ambientes de IaaS onde é empregada a técnica de *provisionamento horizontal*. Quando o termo provisionamento for empregado isoladamente nesse documento, possuirá o sentido de provisionamento horizontal.

2.1.3 Aplicações horizontalmente escaláveis

Uma aplicação é escalável quando seu desempenho melhora com a adição de poder computacional, proporcionalmente à capacidade adicionada. Quando a adição/redução de poder computacional se dá por meio da alocação ou desalocação de máquinas (sejam virtuais ou não) e a carga de trabalho é balanceada entre essas máquinas, as aplicações são consideradas horizontalmente escaláveis. Desta forma, essas aplicações podem ter sua capacidade suprida

por meio de provisionamento horizontal.

Tais aplicações podem apresentar variações intensas e frequentes na carga de trabalho, como será mostrado no Capítulo 4, o que significa que a quantidade de máquinas virtuais necessárias para executá-las, com o nível adequado de desempenho, pode variar com o tempo. Além do mais, a relação de uso entre os tipos de recursos que caracterizam a carga de trabalho das aplicações também pode mudar com o tempo e, por conseguinte, influenciar a decisão do tipo de instância a ser usado em cada momento do provisionamento. Assim, tanto a quantidade de máquinas virtuais quanto o tipo dessas máquinas são passíveis de mudança ao longo do provisionamento.

O nível esperado de QoS da aplicação está relacionado a SLAs firmados entre o provedor da aplicação (que é o cliente IaaS) e os seus usuários. Em um SLA, a QoS esperada da aplicação é definida por um ou mais SLOs. Essencialmente, cada SLO é composto de três partes: a métrica (por exemplo, utilização de CPU), o limiar aceitável (por exemplo 90% de utilização de CPU) e um operador relacional (por exemplo, "menor que"). Assim, nesse exemplo, considera-se que o objetivo exemplificado é cumprido se o percentual de utilização de CPU for mantido abaixo de 90% durante a execução da aplicação e descumprido no caso contrário.

Nesse trabalho, assume-se a existência de um *mapeamento que relaciona a satisfação dos SLOs da aplicação provisionada com o nível de utilização de recursos alocados a ela*. Desta forma, considera-se que os níveis esperados de QoS da aplicação podem ser descritos por SLOs definidos em termos da utilização de recursos da infraestrutura (SLO de utilização). Este mapeamento permite admitir que quanto maior é a utilização de um recurso, menor será a QoS de uma aplicação executada sobre ele, especialmente quando essa utilização viola um SLO de utilização. Por exemplo, considerando a aplicação como um serviço web intensivo em CPU, quanto maior for a utilização de CPU da infraestrutura maior será a disputa por tempo de CPU entre as instâncias do serviço em execução e, por consequência, maior será o tempo de resposta da aplicação, com possíveis perdas de requisições por limitação de tempo de resposta ou rejeição.

Todavia, a decisão sobre a capacidade de recursos necessários para suprir as demandas da aplicação, em termos do tipo e da quantidade de VMs, está atrelada ao modelo de escalabilidade da mesma, que define como ocorre a relação entre a demanda da aplicação, a

capacidade de recursos alocados e o nível de utilização destes. Desta forma, conhecendo-se o modelo de escalabilidade e a demanda por recursos da aplicação é possível definir com precisão a capacidade da infraestrutura minimamente necessária para manter os níveis de utilização abaixo dos limites definidos nos SLOs. A manutenção dos níveis de utilização de recursos abaixo e o mais próximo possível dos limiares estabelecidos tanto garante o menor custo de provisionamento quanto a QoS da aplicação, já que os SLOs presentes nos SLAs firmados entre provedor da aplicação e os seus usuários são satisfeitos.

2.1.4 Provisionamento automático baseado em laço controle

A teoria do controle clássica é comumente aplicada para controlar características específicas de sistemas por meio de modelos de laço fechado de controle com retroalimentação (do inglês, *feedback closed-loop control models*). Nesse caso, o controlador monitora periodicamente o sistema controlado e, com base nas métricas coletadas, ações de controle são realizadas para regular as características do sistema sob controle. Para o controle de sistemas computacionais, como no caso desse trabalho, o processo de controle geralmente envolve a manutenção das características de uma aplicação ou da infraestrutura utilizada para executá-la. A Figura 2.2 representa o modelo padrão de laço fechado de controle com retroalimentação e seus elementos. Hellerstein et al. [32] definem esses elementos da seguinte forma:

- *Sistema*: é o sistema computacional a ser controlado;
- *Saída do sistema*: é uma característica mensurável do sistema controlado, por exemplo, o nível de utilização dos recursos da infraestrutura de execução;
- *Valor de referência*: é o valor desejado para a saída produzida pelo sistema, por exemplo, o limiar estipulado para o nível de utilização dos recursos da infraestrutura;
- *Erro de controle*: é a diferença entre o valor de referência e a saída medida do sistema;
- *Ações de controle*: é uma configuração que influencia o comportamento do sistema controlado e pode ser dinamicamente ajustada, por exemplo, a quantidade de recursos que servem a aplicação em execução;

- *Controlador*: é responsável por determinar as ações de controle necessárias para que a saída do sistema seja a mais próxima possível do valor de referência estipulado. O controlador calcula as ações de controle com base no valor de referência e em valores atuais e/ou do histórico de saídas do sistema;

- *Perturbações*: afetam a maneira como as ações de controle influenciam a saída medida, por exemplo, flutuações na demanda da aplicação podem afetar o nível de utilização da infraestrutura em um dado período de controle;

- *Ruído*: é qualquer efeito que altera a medição da saída produzida pelo sistema controlado, ou seja, um ruído na medição da saída do sistema.

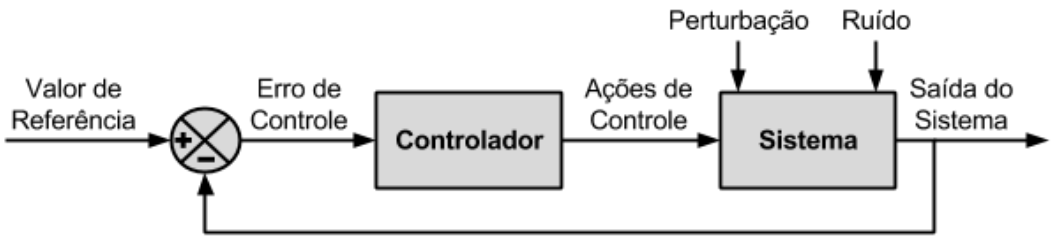


Figura 2.2: Modelo padrão de laço de controle com retroalimentação.

O componente de controle pode ser definido para atender a diferentes finalidades e obter diferentes comportamentos do sistema controlado, que são definidos com base em um ou mais objetivos de controle. Segundo a literatura, tais objetivos de controle são normalmente classificados em [32]:

- *Regulação*: busca assegurar que a saída do sistema é igual (ou próxima) ao valor de referência;

- *Otimização*: atua para a geração de uma saída do sistema próxima ao valor ótimo;

- *Rejeição de perturbações*: tenta assegurar que as perturbações que atuam no sistema não afetem significativamente a saída do mesmo.

A dinâmica dos modelos de controle com retroalimentação pode ser aplicada para provisionar automaticamente aplicações implantadas em ambientes IaaS. Nesse sentido, o modelo de controle atua periodicamente, modificando a quantidade de VMs alocadas à aplicação,

caso necessário, a cada laço de controle do sistema de provisionamento para fornecer os recursos virtuais minimamente necessários para manter os níveis de QoS definidos no SLA da aplicação.

Todavia, a especificação do modelo e das ações de controle que atuam no provisionamento da infraestrutura são derivados da técnica utilizada pela solução de provisionamento empregada. Para técnicas puramente reativas, as ações de controle são definidas a partir de regras de provisionamento previamente especificadas, que estabelecem qual ação de provisionamento deve ser realizada caso um determinado limiar de uma métrica de interesse seja atingido. Por outro lado, as ações de controle baseadas em técnicas proativas ou preditivas são continuamente definidas com base nas estimativas de métricas de interesse produzidas e em um modelo de planejamento de capacidade, que define a capacidade de recursos requerida pela aplicação no futuro próximo.

Independente da técnica de provisionamento utilizada, o objetivo do modelo de provisionamento automático baseado em controle pode ser considerado como uma combinação das categorias acima discriminadas, onde objetiva-se principalmente assegurar o cumprimento dos SLAs da aplicação, atendendo os SLOs associados com a mínima quantidade de recursos. Nesse contexto, o objetivo de *regulação* visa atingir os SLOs da aplicação, o que resulta em um modelo de controle que considera uma métrica de SLO como a saída do sistema e um limiar de SLO como o valor de referência. Além disso, o objetivo de controle como *otimização* é aplicado na tentativa de minimizar os custos de provisionamento através da redução de VMs adquiridas e evitando penalidades derivadas de violações dos SLAs da aplicação decorrentes do não cumprimento dos SLOs. Finalmente, o objetivo de *rejeição de perturbações* consiste em evitar perturbações no sistema devido a variações na carga de trabalho da aplicação, por exemplo através do uso de controladores preditivos que buscam antecipar essas variações de carga.

Em resumo, em um cenário de provisionamento automático baseado em controle com retroalimentação o sistema a ser controlado consiste na aplicação e na infraestrutura sobre a qual ela executa. As ações de controle correspondem a ações de provisionamento realizadas na infraestrutura de execução, que refletem no desempenho da aplicação provisionada. Ou seja, as ações de controle correspondem às modificações na capacidade da infraestrutura, ao longo do tempo, para manter a aplicação com um determinado nível de desempenho,

alocando mais recursos quando a demanda da aplicação aumenta, e liberando recursos tão logo eles não são mais necessários.

2.2 Provisionamento Automático como um Serviço

O cenário de provisionamento automático como um serviço abordado nesse estudo é composto essencialmente por quatro atores: (i) o usuário final da aplicação provisionada que faz uso do serviço prestado pela mesma; (ii) o provedor da aplicação a ser executada no ambiente de IaaS; (iii) o provedor do serviço de provisionamento automático de recursos; e (iv) o provedor de IaaS. A relação entre esses atores está representada na Figura 2.3.

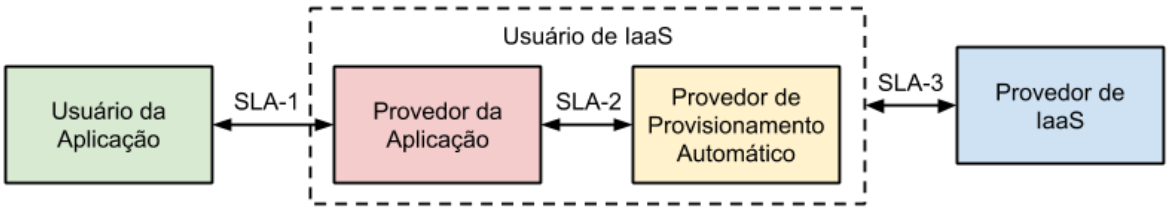


Figura 2.3: Visão geral da relação entre atores do serviço de provisionamento automático de recursos em ambientes de IaaS.

Tipicamente, o provedor de uma aplicação horizontalmente escalável adquire VMs de um provedor de IaaS para executar de forma dedicada a sua aplicação em um ambiente elástico. No contexto desta tese, esta relação com o provedor de IaaS é intermediada pelo provedor de provisionamento automático. Assim, existe um usuário de IaaS que é uma combinação desses dois provedores (da aplicação e do serviço de provisionamento automático), criando-se uma relação de serviço regida por um SLA (SLA-3 na Figura 2.3) que especifica os deveres do provedor de IaaS e direitos do usuário de IaaS. Em termos gerais, este SLA garante que (i) o usuário do provedor de IaaS pode adquirir e criar/terminar VMs e que (ii) essas VMs estejam acessíveis em uma parcela predominante do tempo, sob o risco do ônus de pagamento de penalidades por parte do provedor de IaaS.

A aplicação que executa no ambiente de IaaS é acessada remotamente pelos seus usuários prestando-os um "serviço" que também deve ser regido por um contrato. Nesse caso, o SLA-1 presente na Figura 2.3 indica o contrato estabelecido entre o provedor da aplicação e os usuários dessa aplicação. Esses SLAs são responsáveis por garantir que os usuários da

aplicação tenham acesso a um serviço com o nível de qualidade aceitável, que é resultado de uma execução eficiente da aplicação.

Adicionalmente, o cenário considerado nesse estudo compreende um quarto ator, que consiste em um serviço de provisionamento automático que é responsável por gerenciar os recursos alocados para executar a aplicação no ambiente de IaaS. Esse serviço atua como um procurador do provedor da aplicação junto ao provedor de IaaS, tendo o papel de adquirir e liberar recursos (tipicamente VMs) quando necessário, tornando-se também um usuário do serviço de IaaS por delegação da responsabilidade de gerência da infraestrutura virtual de execução.

O serviço de provisionamento automático opera em um laço de controle e realiza periodicamente o planejamento da capacidade da infraestrutura (quantidade de VMs de um determinado tipo) para acomodar as flutuações da carga de trabalho da aplicação. Desta forma, a capacidade de recursos alocada na infraestrutura de execução pode ser modificada no curto prazo (ordem de minutos) pelo serviço de provisionamento automático. Para tal, esse serviço atua sobre a infraestrutura de execução a cada ciclo de controle, efetivando as decisões de provisionamento tomadas. É importante mencionar que ações de provisionamento, sejam de adição ou remoção de recursos, devem considerar a existência de um tempo para a sua efetivação[1], que corresponde ao tempo de responsividade de provisionamento, da ordem de minutos, segundo estudo apresentado no Apêndice A.

Abordagens reativas estabelecem as ações de provisionamento com base em regras de provisionamento pré-estabelecidas. Estas regras definem condições de disparo de ações. Quando a aplicação monitorada atinge uma dessas condições de disparo, então a ação associada deve ser realizada para o provisionamento da aplicação. Já nas abordagens proativas, a decisão sobre as ações de provisionamento é baseada em métricas de interesse e no modelo de escalabilidade da aplicação. Todavia, independente da abordagem de provisionamento empregada, ações de provisionamento são realizadas com base no tipo de instância de VM estabelecido para provisionar a aplicação.

Para esse trabalho, métricas não intrusivas que são usadas por este serviço de provisionamento correspondem a utilização de recursos alocados. Desta forma, as flutuações da

[1]Tempo necessário para as ações de provisionamento serem efetivadas e refletirem no desempenho da aplicação provisionada.

carga de trabalho da aplicação são vistas pelo serviço de provisionamento como variações nas utilização dos diferentes tipos de recursos (CPU, memória, etc.) executando a aplicação. Assim, para que este serviço de provisionamento automático seja viável, considera-se um sistema de monitoramento que coleta e disponibiliza periodicamente a utilização de diferentes dimensões de recursos das VMs ativas que executam a aplicação. Um exemplo de serviço de monitoramento nesses moldes é o CloudWatch da Amazon [53].

De toda forma, quando a estratégia de provisionamento falha, decidindo por uma capacidade menor do que a necessária, o resultado é a degradação da QoS da aplicação e, possivelmente, perdas econômicas para o provedor da aplicação devido ao descumprimento do SLA-1, apesar de reduções no custo de provisionamento. Isto acontece porque os recursos ficam sobre-utilizados. O contrato entre o provedor da aplicação e do serviço automático de provisionamento (SLA-2 na Figura 2.3) deve considerar limiares adequados de uso dos recursos que levam à QoS desejada da aplicação[2]. Quanto maior for a utilização de um recurso, a partir de um determinado limiar de saturação, menor será a QoS da aplicação já que haverá mais competição pelo uso do recurso. Esses limiares definidos no SLA-2, quando descumpridos, levam à situação de saturação e queda da QoS da aplicação mencionada anteriormente, que podem findar em penalidades a serem pagas pelo provedor do serviço de provisionamento ao provedor da aplicação, definidas no SLA-2. A Figura 2.4 exemplifica uma possível relação entre os SLAs 1 e 2, que respectivamente representam o tempo de resposta e o nível de utilização de recursos para um modelo de fila [43]. A partir de um certo limiar de utilização o crescimento do tempo de resposta deixa de ser linear e passa a ser exponencial, de forma que quando a utilização aproxima-se de 100% o tempo de resposta tende a infinito.

Com base na relação entre uso dos recursos e desempenho da aplicação, os SLAs entre os provedores das aplicações e o serviço de provisionamento (SLA-2) definem SLOs com limiares de utilização dos recursos em uso para executar a aplicação. Por exemplo, um SLO pode definir que a utilização de CPU das VMs executando a aplicação não deve ultrapassar 80%. Neste caso, espera-se que depois que a utilização do recurso ultrapassa 80% o tempo de resposta da aplicação cresça, podendo violar o nível aceitável de QoS. Assim, violações dos SLOs definidos no SLA-2 podem gerar degradação no desempenho da aplicação, que

[2]Operações de *benchmarking* sobre as aplicações provisionadas podem ajudar a determinar esses limiares.

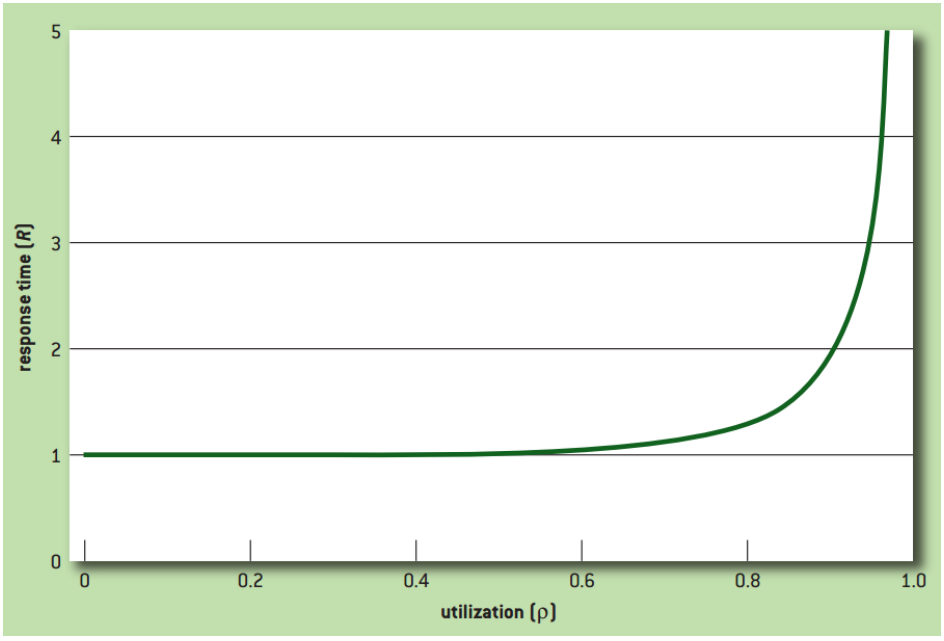


Figura 2.4: Curva demonstrando o tempo de resposta como uma função da utilização para um sistema de fila M/M/m [43].

é possivelmente percebido pelo usuário final da aplicação. Se a utilização de recursos for mantida abaixo, mas o mais próximo possível do limiar estabelecido nos SLOs (do SLA-2), os SLAs da aplicação (SLA-1) são satisfeitos. Além disso, quanto mais próximo dos limiares estabelecidos no SLA-2 estiverem as utilizações recursos que executam a aplicação, menor o custo de execução da aplicação. Ou seja, existe um *trade-off* entre os objetivos de minimização do custo de provisionamento e manutenção da QoS da aplicação. Desta forma, o serviço de provisionamento deve buscar manter os recursos que executam a aplicação com utilizações mais próximas possíveis porém menores que o estabelecido nos SLOs do SLA-2, reduzindo assim a probabilidade de ocorrência de violações do SLA-2 enquanto que o custo de provisionamento é reduzido no longo prazo. Por consequência, espera-se que o SLA-1 seja também cumprido.

Evidentemente, é necessário que exista um certo controle sobre os custos de provisionamento praticados pelo serviço de provisionamento de forma a garantir que não apenas os objetivos de minimização de violações de SLO sejam atingidos. Caso contrário, é natural que o serviço de provisionamento atue de forma conservadora, buscando super provisionar a infraestrutura de execução a fim de evitar penalidades pelo não cumprimento do SLA-2. Esse objetivo de minimização de custos de execução pode ser assegurado na relação entre o pro-

vedor do serviço de provisionamento e o provedor da aplicação de diferentes formas. Uma possibilidade pode ser a aplicação de ganhos para o provedor do serviço proporcionais às economias de custo obtidas em comparação ao custo do provisionamento estático, calculado a partir do histórico observado de demandas da aplicação. Outra abordagem pode basear-se em descontos sobre a tarifação do serviço de provisionamento em função do percentual de recursos desperdiçados. Todavia, a especificação dessa relação entre provedor do serviço e provedor da aplicação com base nos custos de provisionamento não é o foco deste trabalho.

Portanto, o principal objetivo do serviço de provisionamento automático consiste em garantir a QoS desejada para a aplicação horizontalmente escalável ao mesmo tempo que minimiza o custo de execução da mesma em um ambiente de IaaS. Alguns requisitos são fundamentais para permitir que este serviço seja oferecido no contexto de IaaS. Um requisito importante consiste na capacidade do serviço de provisionamento automático de prover recursos automaticamente para diferentes aplicações. Para tal, o serviço deve funcionar com o mínimo grau de intrusividade. Por isso, este serviço usa informações não específicas da aplicação sobre a utilização de recursos alocados, obtidas no nível da infraestrutura virtual. Complementarmente, outros requisitos são importantes, como ser de fácil implementação e eficiente em termos de configuração, além de ser independente de características específicas da carga de trabalho das aplicações. Desta forma, considerando os diferentes requisitos necessários a um serviço de provisionamento automático, é possível que diferentes aplicações horizontalmente escaláveis passem a tirar proveito ou explorem o uso da elasticidade oferecida por ambientes de IaaS para executar eficientemente, com níveis aceitáveis de QoS e fazendo uso otimizado da infraestrutura alocada, a partir de um serviço de provisionamento automático acessível aos usuários do provedor de IaaS.

2.3 Questões de Pesquisa

Diversas questões sobre o processo de provisionamento automático de recursos em ambientes de IaaS revelam-se essenciais para a construção de uma solução de provisionamento de aplicações horizontalmente escaláveis nos moldes do escopo deste trabalho. As principais estão relacionadas ao desempenho das soluções de provisionamento automático, sejam reativas e proativas. Mais especificamente, tratam de como atingir de forma eficiente e con-

trolável os objetivos de provisionamento para diferentes aplicações com base em métricas não intrusivas. Contudo, a questão fundamental que sintetiza essa problemática, e cuja resposta corresponde ao objetivo dessa pesquisa, consiste em:

> **Como realizar eficientemente o provisionamento automático como um serviço em IaaS considerando diferentes aplicações e métricas não intrusivas?** Por eficientemente entende-se: que seja capaz de assegurar níveis aceitáveis de QoS das aplicações e, quando da existência de variação na carga de trabalho, otimizar o uso de recursos e minimizar os custos de provisionamento.

Essa questão constitui o foco principal desta pesquisa, sendo abordada nos Capítulos 5 , 6 e 7. A seguir, decompomos essa questão em sub questões de pesquisa relacionadas com o intuito de objetivar o seu estudo:

1. Como se dá o desempenho das técnicas de provisionamento, reativas e provativas, em atingir os diferentes objetivos de provisionamento, considerando diferentes métricas de interesse uni-e-multidimensionais?

2. Qual é a capacidade das soluções de provisionamento em usar o espaço de configurações para explorar o *trade-off* entre os objetivos de provisionamento para as diferentes aplicações e métricas de interesse?

3. Quão genéricas são as técnicas de provisionamento avaliadas em provisionar eficientemente um conjunto distinto de aplicações? Existe dependência entre o desempenho das soluções e características específicas da carga de trabalho das aplicações provisionadas?

4. Quão eficientes são as técnicas de provisionamento avaliadas em termos de configuração para operar no provisionamento das diferentes aplicações com base em diferentes métricas de interesse?

5. Qual é a complexidade de construção e implementação do serviço de provisionamento com base nas diferentes técnicas de provisionamento, reativas e provativas?

6. Como a escolha do tipo de instância de VM a ser usado no provisionamento pode impactar os custos de execução praticados pelo serviço de provisionamento automático?

Capítulo 3

Trabalhos Relacionados

Neste capítulo são descritos trabalhos relacionados ao tema de gerência de capacidade e provisionamento automático de recursos em ambientes de IaaS. A partir de um levantamento do estado da arte, estes trabalhos são analisados e discutidos segundo a natureza e as características de funcionamento da abordagem de provisionamento. Todavia, o principal objetivo desse levantamento consiste em classificar as atuais soluções de provisionamento automático e horizontal de recursos segundo aspectos e características essenciais para a construção de um serviço de provisionamento automático de aplicações em ambientes de IaaS e Computação na Nuvem.

3.1 Soluções de Provisionamento Automático de Recursos

O objetivo principal das soluções de provisionamento automático consiste usar mecanismos dinâmicos para garantir o nível adequado de QoS das aplicações provisionadas e o uso eficiente de recursos virtuais adquiridos do provedor de Computação na Nuvem e IaaS. [20]. Da perspectiva do usuário de IaaS, o provisionamento dinâmico tem sido principalmente usado para evitar o provisionamento inadequado de recursos para a aplicação sob sua responsabilidade ao mesmo tempo que visa a redução de custos de provisionamento [24]. Desta forma, tais soluções devem ser capazes de lidar com a relação conflitante entre a redução de custos de provisionamento que pode ser alcançada em decorrência da aquisição de recursos do provedor de IaaS, e a manutenção dos níveis esperados de QoS da aplicação em execução na infraestrutura adquirida.

Todavia, da perspectiva do provedor de um serviço de provisionamento automático como abordado nesse trabalho, o desempenho do serviço restringe-se não apenas a sua habilidade de lidar com essa relação conflitante de objetivos de provisionamento, mas também à eficiência da solução quanto a sua generalidade em assegurar tais objetivos para diferentes aplicações, de forma independente dos perfis de carga de trabalho destas. Além disso, o serviço deve operar a partir de métricas não específicas da aplicação provisionada, por questões principalmente de generalidade e desacoplamento das aplicações provisionadas e de privacidade de informações dos usuários do serviço de provisionamento automático.

Desta forma, a partir de um levantamento do estado da arte foi desenvolvida uma taxonomia para classificar as soluções de provisionamento automático e horizontal em um cenário de provisionamento como um serviço. Os diferentes aspectos presentes na taxonomia descrita na Figura 3.1 e usados na classificação das soluções são os seguintes:

- **Modo de operação da abordagem**: consiste em como as ações de provisionamento são realizadas, de maneira reativa ou proativa; Técnicas reativas baseiam-se em informações pontuais de demanda para a tomada de decisão de provisionamento, enquanto que abordagens proativas realizam o planejamento de capacidade da infraestrutura com base em estimativas de demandas baseadas no histórico destas;

- **Técnica de provisionamento**: técnicas usadas como base das decisões de provisionamento realizadas, baseadas em regras de provisionamento ou na análise do histórico de métricas coletadas, através de modelos de teoria das filas (QT, do inglês *Queueing Theory*), de predição de séries temporais (TS, do inglês *Time Series*), de aprendizagem por reforço (RL, do inglês *Reinforcement Learning*) ou outras;

- **Tipo da métrica de provisionamento**: o tipo de informações exigidas pela solução de provisionamento horizontal é definido pela origem de obtenção da informação, que podem ser: a aplicação, a infraestrutura virtual ou a infraestrutura física;

- **Nível de intrusividade**: informações utilizadas pela técnica para decidir sobre as ações de provisionamento, classificadas pelo nível de intrusão que a coleta desta informação proporciona à aplicação provisionada. Apenas informações obtidas no nível da infraestrutura de execução são consideradas não intrusivas e podem ser consideradas por um serviço de provisionamento automático;

- **Multiplicidade de dimensões**: define como o provisionamento deve ser realizado em termos das dimensões de recursos consumidos pela aplicação em execução, considerando uma única dimensão ou múltiplas dimensões de recursos computacionais;

- **Seleção de tipo de instância**: trata sobre qual tipo de instância de VM é utilizado no provisionamento e como esse tipo é definido, se de forma estática no início do processo de provisionamento ou dinamicamente durante o provisionamento em função da demanda por recursos.

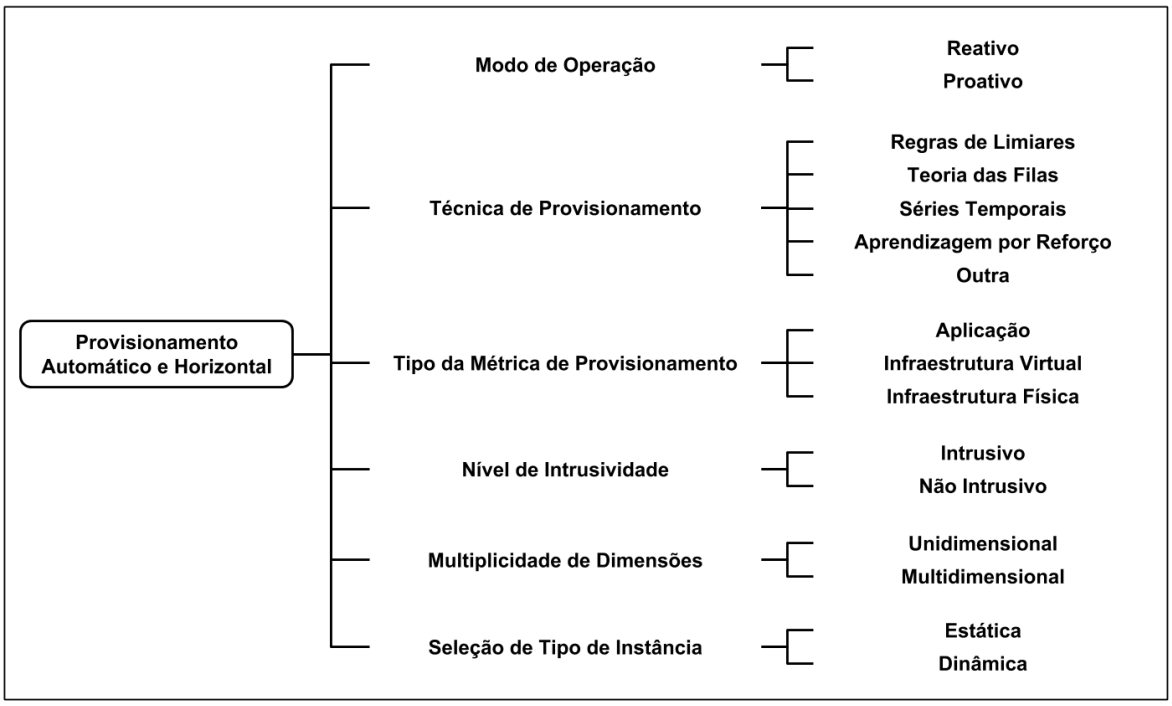


Figura 3.1: Taxonomia para o provisionamento automático em IaaS.

3.1.1 Métodos de provisionamento automático: vertical e horizontal

O provisionamento vertical consiste no aumento ou diminuição, em tempo de execução, da capacidade atribuída a uma instância de máquina virtual em execução. Por outro lado, o provisionamento horizontal é baseado na gerência de capacidade de uma infraestrutura de execução por meio da replicação de recursos virtuais, em que VMs são alocadas ou desalocadas da infraestrutura virtual a fim de modificar sua capacidade. A abordagem de provisionamento vertical é tratada na literatura por diversas pesquisas [22, 29, 47, 57, 58, 64, 67]

principalmente pela sua utilidade em cenários com variações intensas na carga de trabalho, que requerem um menor tempo de responsividade das ações de provisionamento. Ao mesmo tempo, a técnica é limitada em termos da capacidade de recursos que pode ser aumentada, uma vez que uma VM não pode crescer além da capacidade dos servidores físicos. Além disso, essa abordagem não é usualmente oferecida pelos provedores públicos de Computação na Nuvem, principalmente devido à complexidade adicionada ao gerenciamento de recursos do provedor. Desta forma, esse método de provisionamento é mais comum em soluções ou serviços privados de IaaS, como OpenStack [19]. Tal método de provisionamento encontra-se fora do escopo desta pesquisa, que foca em provisionamento automático e horizontal como um serviço em ambientes de IaaS.

O método de provisionamento horizontal está atualmente em uso tanto em provedores de IaaS privados quanto públicos [3, 30, 51]. Esse método também é exaustivamente explorado pela a academia para o desenvolvimento de soluções de provisionamento automático de recursos [2, 8, 9, 11–15, 21, 23, 25, 27, 33, 35, 36, 40, 44, 48, 49, 52, 55, 56, 61–63, 66]. A implementação deste método requer a habilidade de decidir quando e como a capacidade da infraestrutura deve ser modificada em termos da quantidade de recursos necessários em diferentes dimensões para manter a aplicação em execução com níveis aceitáveis de QoS. Além do mais, esse planejamento também deve definir as ações de provisionamento a serem realizadas em função dos tipos de instância de VM oferecidos pelos provedores de IaaS. Este planejamento de capacidade pode operar, de forma reativa, reagindo a mudanças na demanda da aplicação (ou na utilização de recursos) ou de forma proativa, através de estimativas de mudanças na carga de trabalho e da antecipação das ações de provisionamento. Independentemente do modo de operação da solução, reativo ou proativo, o nível e o tipo de informações exigidas por cada técnica a caracteriza segundo o nível de intrusividade da abordagem.

3.1.2 Provisionamento horizontal e reativo

Técnicas reativas intrusivas

Técnicas de provisionamento como modo de operação reativo são bastante comuns, sendo oferecidas pela maior parcela das soluções comerciais de IaaS, tais como aquelas fornecidas pela Amazon AWS [3], Rackspace [51] e Google [30]. Essa popularidade deriva-se da

facilidade de implementação e uso de tais técnicas [38]. As soluções reativas mais comuns na literatura são tidas como intrusivas, que exigem informações específicas da aplicação, tais como, tempos de resposta, taxa de chegada de requisições, comprimentos de fila e demandas de requisições [9, 11, 12, 23, 40, 55]. Na prática, espera-se que soluções intrusivas apresentem melhor desempenho, uma vez que as métricas consideradas no provisionamento estão diretamente relacionadas à QoS da aplicação provisionada e possuem maior potencial de conduzir a melhores decisões de provisionamento. No entanto, o nível de informação requerido por essas abordagens as caracterizam como soluções de provisionamento intrusivas, logo não aplicáveis ao cenário de provisionamento automático como um serviço proposto nesse trabalho.

As soluções propostas por Calcavecchia et al. [11], Fitó et al. [23] e Seung et al. [55] operam em reação ao tempo de resposta de uma aplicação Web em execução na infraestrutura. De forma específica, a primeira solução utiliza um modelo de filas para medir o desempenho da aplicação em termos do tempo de resposta e com base em limiares da taxa de chegada de requisições e da taxa de serviço da aplicação dispara ações de provisionamento da aplicação. A segunda solução, por sua vez, classifica o tempo de resposta da aplicação provisionada em diferentes níveis de intensidade e relaciona cada classe de tempo de resposta com uma ação de provisionamento correspondente, ou seja, para cada limiar de tempo de resposta atingido uma ação de provisionamento é disparada. Por fim, o trabalho de Seung et al. propõe o CloudFlex, que opera com base em um modelo de laço de controle e dispara ações de provisionamento associadas a limiares de tempo de resposta da aplicação provisionada.

Bonvin et al. [9] propõe uma solução de provisionamento reativo, que atua de forma horizontal e vertical, baseada em agentes que executam na mesma infraestrutura da aplicação e são capazes de monitorar o tempo de resposta da mesma. Nesse sentido, sempre que um agente avalia que uma instância da aplicação está com tempo de resposta acima do aceitável uma ação de provisionamento é executada. Calheiros et al. [12] desenvolveu o Aneka, que consiste em uma solução reativa de provisionamento distribuído entre infraestruturas de Nuvem e de grades computacionais. Essa solução considera informações sobre o tempo de execução de tarefas de uma aplicação e a demanda desta aplicação para provisionar recursos computacionais, com o intuito de reduzir o tempo de execução e os custos da aquisição de recursos. Por fim, a solução de Marshall et al. [40] realiza o provisionamento automático e

reativo em função da fila de tarefas da aplicação a serem executadas, onde a quantidade de recursos para execução da aplicação é decidido a partir de métricas computadas dessa fila, como o tempo entre chegadas de tarefas, tempo de espera em fila, etc.

Técnicas reativas não intrusivas

Existem abordagens reativas que baseiam-se apenas em informações obtidas no nível da infraestrutura de execução, i.e. infraestrutura virtual adquirida do provedor de IaaS. Estas métricas apresentam uma relação *indireta* com as métricas de QoS da aplicação e são tidas como não intrusivas. Em geral essas soluções utilizam limiares de utilização de recursos da infraestrutura para disparar ações de provisionamento da aplicação em execução, como proposto por Ghanbari et al. [27], Lim et al. [35], Netto et al. [48] e Righi et al. [21]. No entanto, como demonstrado no decorrer deste trabalho, tais técnicas são limitadas pela complexidade em encontrar configurações para os diferentes cenários de demanda da aplicação, onde configurações padrão nem sempre são suficientes, além de ter sua eficácia questionável para cenários com intensa variação de carga de trabalho [38].

As abordagens de provisionamento desenvolvidas por Ghanbari et al. [27], Lim et al. [35], Netto et al. [48] e Righi [21] baseiam-se em regras de provisionamento definidas em função da utilização de CPU da infraestrutura de execução. A primeira utiliza um sistema de votação associado a regras de provisionamento para definir quando as ações de provisionamento devem ser realizadas para provisionar uma aplicação Web em execução. As demais atuam com base em limiares ou regras de provisionamento adaptativas e diferem entre si em relação ao algoritmo utilizado para estabelecer as novas configurações de regras de provisionamento. Lim et al. fazem uso de um controlador integral que define novos valores de limiares com base na diferença entre os valores de utilização de CPU medidos e o limiar configurado a cada momento. Ao invés de adaptar configurações de limiares de provisionamento, Netto et al. [48] utilizam uma abordagem adaptativa que modifica as configurações da quantidade de recursos provisionados com base na quantidade de VMs alocadas e no percentual de uso da infraestrutura. Por fim, a solução de Righi et al. [21] realiza a configuração dos limiares de provisionamento a partir de um algoritmo inspirado no controle de congestionamento do protocolo TCP (do inglês, *Transmission Control Protocol*), considerando informações de utilização de CPU frequentemente coletadas da infraestrutura de execução.

3.1.3 Provisionamento horizontal e proativo

Técnicas proativas intrusivas

Assim como nas soluções reativas, entre as soluções proativas de provisionamento automático também é predominante o uso de informações intrusivas no processo de provisionamento automático. A maior parcela das soluções de provisionamento proativo consideram informações específicas da aplicação em execução [2, 8, 13, 15, 25, 33, 36, 49, 52, 56, 61, 63, 66]. Em geral, estes trabalhos utilizam métricas obtidas diretamente da aplicação, como tempo de resposta, taxa de chegada de requisições e quantidade de usuários no sistema para decidir ações de provisionamento realizadas sobre a infraestrutura de execução da aplicação de interesse.

Técnicas de provisionamento de modelos de filas O trabalho desenvolvido por Urgaonkar et al. [61] propõe, por meio de análises experimentais, uma técnica para o provisionamento automático de recursos virtuais que opera através da combinação de abordagens proativas e reativas de atuação. A abordagem proativa consiste na produção de estimativas de picos de demanda recebida pela aplicação em execução, em termos da taxa de chegada de requisições, para cada hora. A abordagem reativa corresponde a um modo de atuação conservador que opera para corrigir erros de predição no longo prazo ou em reação a picos de demanda não previstos. Independentemente do modo de operação, o planejamento de capacidade da infraestrutura de provisionamento é realizado com base em um modelo de filas que calcula a quantidade de recursos necessários a partir da taxa de chegada de requisições estimada para manter o tempo de resposta da aplicação conforme o estabelecido no SLA da mesma.

De forma semelhante, a solução de provisionamento proposta por Ali-Eldin et al. [2] opera através de técnicas de provisionamento proativo e reativo, que são combinadas de diferentes formas para decidir sobre as ações de provisionamento a serem realizadas. Todavia, independente do modo de operação empregado, as decisões de provisionamento são tomadas com base na capacidade de serviço da infraestrutura, em termos de taxa de requisições por segundo servidas. Para tal, um modelo de filas também é adotado para modelar a capacidade de recursos necessária para suprir as demandas a partir de um fator de tendência aplicado sobre a taxa de requisições por segundo monitorada.

Casalicchio et al. [15] propõem uma abordagem de provisionamento automático, proativo e reativo, que considera tanto métricas específicas da aplicação quanto métricas de utilização da infraestrutura de execução. As políticas de provisionamento reativo atuam diretamente com base em medições da utilização de CPU da infraestrutura e no tempo de respostas da aplicação, enquanto que as políticas proativas fazem uso de um modelo de filas para estimar a capacidade de recursos necessária para assegurar os níveis esperados de QoS a partir do tempo de resposta e da taxa de chegada de requisições. Apesar do uso de métricas não intrusivas, a necessidade de métricas específicas da aplicação provisionada promove um caráter intrusivo à solução proposta, além de tornar-la fortemente dependente de características do serviço prestado pela aplicação em execução.

Kingfisher, proposto por Sharma et al. [56], é uma solução de provisionamento automático e proativo de recursos que atua com base na replicação, ou provisionamento horizontal, e migração de recursos. A solução tem por objetivo otimizar os custos de provisionamento em termos da quantidade de recursos adquiridos e em relação ao custo operacional de migração da aplicação para uma nova configuração de infraestrutura. O estudo baseia-se na relação de custos entre as duas técnicas de provisionamento consideradas, e por esse motivo considera um preditor perfeito de demanda, em termos da taxa de requisições por segundo. Essas estimativas de demanda são utilizadas pela solução para estimular um modelo de fila que computa a capacidade de recursos requerida pela aplicação no futuro próximo para assegurar os SLAs da aplicação em termos dos tempos de resposta da aplicação. No entanto, a solução também caracteriza-se em uma abordagem intrusiva por necessitar especificamente de métricas coletadas no nível da aplicação em execução na infraestrutura provisionada.

O AutoMAP proposto por Beltrán et al. [8] faz uso de uma abordagem de provisionamento automático que baseia-se em uma rede de filas para realizar o provisionamento de aplicações múltipla camada, interativas ou em lote, onde cada camada é provisionada individualmente. O modelo de filas calcula o tempo de resposta esperado para cada camada com base na taxa de chegada de requisições e na utilização de CPU da infraestrutura. Todavia, a solução também requer informações específicas da aplicação para atuar eficientemente e portanto é considerada como uma abordagem de provisionamento intrusiva à aplicação provisionada.

Técnicas de provisionamento de análise de séries temporais Diferentemente das soluções descritas anteriormente, que utilizam essencialmente técnicas de provisionamento baseadas em modelos de filas, a técnica de provisionamento desenvolvida por Yang et al. [66] utiliza uma técnica de provisionamento baseada em análise de séries temporais, que usa um modelo de regressão linear para estimar a carga de trabalho em termos da taxa de chegada de requisições à aplicação. Desta forma, a estimativa dessa taxa é usada para confrontar limiares de regras de provisionamento pré-definidos e decidir sobre as ações de provisionamento a serem realizadas. O que torna a solução essencialmente intrusiva à aplicação, uma vez que considera apenas métricas específicas sobre a taxa de chegada de requisições. A solução combina um pré provisionamento horizontal com o provisionamento vertical em tempo real, onde a decisão entre os métodos de provisionamento é realizada com base em uma estratégia gulosa que visa obter recursos verticalmente enquanto aloca recursos horizontalmente.

Outra abordagem de provisionamento automático e proativo que baseia-se em análise de séries temporais é proposta por Roy et al. [52]. A solução usa informações sobre a quantidade de diferentes usuários que acessam a aplicação a cada momento do tempo para planejar a capacidade da infraestrutura de execução. O objetivo da solução é minimizar os custos de provisionamento em duas dimensões, na aquisição de recursos e em relação às penalidades devido a violações de SLA, definidas em termos do tempo de resposta da aplicação. Para tal, a solução faz uso de um laço de controle que, a partir de estimativas de demanda geradas por um modelo de predição de auto-regressão com média móvel (ARMA, do inglês Auto-Regressive Moving Average), realiza periodicamente o planejamento de capacidade da infraestrutura de execução da aplicação. Todavia, a solução também configura-se como intrusiva à aplicação em execução, uma vez que requer informações específicas da mesma.

Vadara é um arcabouço proposto por Loff et al. [36] para o provisionamento de elasticidade em ambientes de Nuvem com base em métricas de demanda de tarefas da aplicação a partir de modelos de predição. A solução utiliza uma combinação dos modelos de predição séries temporais para melhorar a acurácia das estimativas de demanda. A técnica é avaliada com base em rastros de aplicações em execução na Google através de predições do número de tarefas a serem executadas a partir da combinação dos modelos de predição: Holt-winters, ARIMA e StructTS. Apesar da técnica apresentar-se como uma solução genérica de provisionamento, as métricas utilizadas pela solução são obtidas no nível da aplicação

em execução e não encontram-se disponíveis para os provedores de computação. Por esse motivo, considera-se a abordagem não aplicável ao cenário de provisionamento automático como um serviço em IaaS.

AGILE, proposto por Nguyen et al. [49], propõe tanto uma técnica de predição de cargas de trabalho a partir de transformadas Wavelet (entre tempo e frequência), usadas para a predição de utilização de CPU e memória, quanto um arcabouço para o provisionamento automático horizontal de recursos para infraestruturas de IaaS. Apesar da técnica de predição ser utilizada com métricas obtidas no nível da infraestrutura virtual, as decisões envolvendo o planejamento de capacidade dependem do perfilamento da aplicação, que é realizado em função do tempo resposta da aplicação. Esse perfilamento é usado para gerar um modelo de pressão (do inglês, *pressure model*), que modela a relação entre recursos alocados e usados pela aplicação para um dado percentual de SLO definido em termos do tempo de resposta, que por sua vez é definido com base na taxa de requisições por segundo e no tipo das requisições. Por esse motivo, a técnica de provisionamento mostra-se intrusiva e não aplicável ao cenário de provisionamento como um serviço proposto nesse trabalho.

O trabalho de Verma et al. [63] descreve um modelo de predição de demanda de recursos a partir de uma classificação de aplicações com base em características de alto nível (por exemplo, número de funcionalidades e status do serviço em execução), em dados de recursos requisitados pela aplicação e na utilização de CPU da infraestrutura de execução. O provisionamento é realizado de forma horizontal com base em VMs com diferentes capacidades de CPU e utiliza dois níveis de predição, uma no curto e outra longo prazo quando a demanda apresenta tendências crescentes. As predições no curto prazo baseiam-se nos modelos de predição de séries temporais ARX e ARMAX, enquanto que modelos de EMA (do inglês, *Exponential Moving Average*) e de sazonalidade são usados para predições no longo prazo. O objetivo principal da técnica é minimizar o tempo de indisponibilidade da aplicação, em que a aplicação fica inacessível (*down-time*).

Técnicas de provisionamento de modelos de filas e análise de séries temporais Entretanto, existem outros tipos de soluções de provisionamento horizontal e proativo cujas abordagens combinam técnicas de provisionamento de modelos de filas e de análise de séries temporais, como os trabalhos de Gandhi et al. [25], Jiang et al. [33] e Calheiros et al. [13].

Gandhi et al. [25] propõem uma solução de provisionamento horizontal com base em modelo de filas e filtros de Kalman para estimar a quantidade de recurso requerida pela aplicação a cada instante de tempo. Desta forma, a solução considera dois níveis de métricas, da infraestrutura virtual (utilização de CPU) e da aplicação (taxa de requisições por segundo e tempo de resposta), assumindo que não é possível capturar todas as características da aplicação sem ser intrusivo ao espaço do usuário do serviço de provisionamento.

Jiang et al. [33] desenvolvem uma solução de provisionamento proativo horizontal que opera com base em métricas intrusivas da taxa de requisições por segundo e modelos de predição baseados em regressões lineares. No processo de predição a escolha do tamanho de histórico de dados considerados baseia-se em coeficientes de autocorrelação calculados com base em dados de requisições por segundo coletados no tempo. Essas predições são utilizadas para alimentar um modelo de fila que avalia a relação entre latência da aplicação e custo, relacionado ao número de VMs e o preço de aquisição, e define a quantidade de recursos necessários para executar a aplicação com o nível esperado de QoS.

Finalmente, o trabalho de Calheiros et al. [13] apresenta uma técnica de provisionamento horizontal baseada em predições de carga de trabalho a partir do modelo de predição ARIMA. A técnica de provisionamento realiza periodicamente predições da taxa de chegada de requisições e utiliza um modelo de fila para computar a quantidade de recursos necessários para provisionar a aplicação em um futuro próximo, mantendo o tempo de resposta da aplicação abaixo de um limiar e considerando a possibilidade de rejeição de requisições, que determinam o nível de QoS da aplicação. Desta forma, as métricas consideradas no processo de provisionamento posicionam a técnica como sendo intrusiva à aplicação provisionada em um cenário de provisionamento automático como um serviço.

Técnicas proativas não intrusivas

Apesar da diversidade de trabalhos que propõem soluções de provisionamento proativo intrusivas, o escopo deste trabalho limita-se à investigação de soluções de provisionamento que atuam de forma não intrusiva, ou seja, que necessitam apenas de informações sobre o nível de utilização de recursos das VMs para tomar decisões de provisionamento. Caron et al. [14] e Vasić et al. [62] propõem soluções proativas que são não intrusivas. No entanto, essas soluções tornam-se não adequadas para um cenário de provisionamento automático como um

serviço, seja respectivamente por necessitar de informações específicas sobre a aplicação na configuração do algoritmo de predição do provisionamento ou por considerar métricas não disponibilizadas na prática pelos atuais provedores de IaaS.

A primeira destas soluções requer informações específicas da aplicação para a configuração do algoritmo de predição de carga de trabalho da aplicação, que baseia-se no histórico do casamento de padrões da carga (do inglês *pattern matching*) [14]. Essa configuração necessita de informações sobre o tempo do processamento de requisições para configurar a técnica de provisionamento. Especificamente, a técnica utilizada consiste em uma adaptação do algoritmo Knuth-Morris-Pratt (KMP) que estima a utilização de CPU futura da infraestrutura com base em padrões de uso do passado. A solução de Vasić et al., por sua vez, propõe um arcabouço de provisionamento automático que atua proativamente através de classificadores de assinaturas de carga de trabalho, derivada de métricas de baixo nível da infraestrutura (por exemplo, taxa de operações por segundo, o uso do cache L2, quantidade de eventos na tabela de páginas, etc.), que não são fornecidos pelos atuais provedores de IaaS [62] e encontram-se disponíveis apenas no nível de infraestrutura física.

Como já destacado, o provisionamento automático como um serviço em IaaS requer que soluções de provisionamento consideradas pelo serviço necessitem apenas de informações que possam ser facilmente obtidas no nível da VM ou da infraestrutura virtual, como exemplo do histórico de utilização de recursos da infraestrutura. Nesse sentido, o arcabouço de provisionamento automático proposto por Morais et al. [44] é a solução proativa que melhor enquadra-se nesse cenário de provisionamento, uma vez que utiliza apenas dados temporais de utilização e alocação de CPU e um limite aceitável de utilização desse recurso, definido pelo usuário do serviço, para tomar decisões de provisionamento horizontal. Para tal, a solução calcula periodicamente estimativas da demanda da aplicação provisionada com base em modelos de predição de séries temporais e confronta estas estimativas com o limite de utilização definido. Contudo, o modelo de predição utilizado nesse processo é periodicamente selecionado dentre um conjunto de preditores pré-definidos, sendo aquele que apresenta o melhor desempenho em termos de custo de provisionamento e manutenção da QoS da aplicação em execução.

3.1.4 Provisionamento horizontal e multidimensional

Como observado nas seções anteriores, as soluções de provisionamento automático abrangem tanto as soluções com modo de operação reativo como as abordagens que operam proativamente. Todavia, independente do modo de operação das soluções, as abordagens de provisionamento conhecidas, em geral, priorizam a estimativa da quantidade de recursos necessários para executar a aplicação provisionada em apenas uma dimensão de recurso, essencialmente utilização de CPU. Além disso, entre essas abordagens de provisionamento horizontal é dominante o uso de um único tipo de instância de VM, não necessariamente o tipo mais economicamente rentável para o provisionamento.

Até onde se sabe, Sharma et al. [56], Yang et al. [66], Beltrán et al. [8], Verma et al. [63] e Vasić et al. [62] propõem abordagens de provisionamento horizontal que abordam o problema de seleção do tipo de instância. A solução Vasić, dentre estas, é a única que utiliza múltiplas métricas no processo de seleção do tipo de instância, que apesar de serem métricas não disponíveis no nível da infraestrutura virtual de execução, estão indiretamente relacionadas ao consumo de CPU e memória, mas não são disponibilizadas pelos atuais provedores de IaaS. Todas as demais soluções de provisionamento usam uma única dimensão de recurso para decidir qual tipo de VM usar, desconsiderando o impacto de outras dimensões de recursos no desempenho da aplicação, o que pode levar a violações de SLO e redução dos níveis de QoS da aplicação [45].

A primeira solução, proposta por Sharma et al., seleciona o tipo de instância mais econômico com base na análise empírica de desempenho dos tipos de instância e no preço relacionado a cada tipo. Yang et al. propõe um modelo de provisionamento horizontal de recursos que considera diferentes tipos de instância para avaliar questões de custo de aquisição de VMs em um cenário de Nuvem pública, todavia as avaliações realizadas fazem uso apenas de informações relacionadas ao consumo de CPU. No trabalho de Beltrán et al. a decisão pelo tipo de VM é realizada com base em um otimizador que decide o tempo de resposta da aplicação com base em tipos de VM com diferentes capacidades de CPU e no custo associado pelo uso de cada um dos tipos. No trabalho de Verma et al. a seleção é executada com o objetivo de minimizar o desperdício de recursos alocados, particularmente em termos de unidades de CPU, desconsiderando custos associados à aquisição de recursos em um ambiente de IaaS.

Desta forma, é importante observar que independentemente da abordagem seguida, diferentes dimensões de recursos devem ser consideradas para incluir a seleção automática de instâncias em uma solução de provisionamento automático existente. Além disso, entender como os recursos alocados são consumidos é fundamental para escolher o tipo de instância a ser usado com a melhor relação custo-benefício. Desta forma, as soluções intrusivas baseadas apenas em métricas específicas da aplicação (tempos de resposta, taxas de chegada, etc.) também devem observar ou estimar métricas de uso de recursos (CPU, memória RAM, Disco, etc.) para executar uma seleção de tipo de instância no processo de provisionamento. As soluções que já reúnem informações relacionadas ao uso de recursos são mais adequadas para incluir um seletor de tipo de instância, uma vez que já monitoram as informações necessárias ao processo de seleção.

3.2 Considerações

A partir da revisão de literatura realizada pode-se observar que o problema de provisionamento automático de aplicações é amplamente explorado pela academia. Diversos são os trabalhos que abordam essa temática a partir de diferentes enfoques e cenários de provisionamento. Percebe-se que a maior parcela das soluções de provisionamento automático propostas na literatura fazem uso de técnicas de provisionamento intrusivas e dependentes de informações específicas da aplicação provisionada. Em geral essas informações são utilizadas para estimar as necessidades da aplicação em termos de recursos computacionais necessários para a sua execução. Todavia, apenas soluções de provisionamento horizontal não dependentes desse nível de informação, ditas não intrusivas, mostram-se adequadas para compor um serviço de provisionamento automático em ambientes de IaaS como o proposto nesse trabalho.

Conforme a revisão da literatura realizada, não observa-se trabalho relacionado com o uso de soluções de provisionamento na construção de um serviço de provisionamento automático em ambientes de IaaS. Até onde se sabe, este é o primeiro trabalho que avalia a eficiência e as limitações do uso de técnicas de provisionamento, reativo e proativo, em um cenário de provisionamento automático como um serviço. Essa avaliação baseia-se tanto em um estudo generalista das técnicas de provisionamento com diferentes modos de operação,

quanto no desempenho de soluções de provisionamento não intrusivas, que utilizam informações obtidas no nível da infraestrutura virtual disponibilizadas pelos provedores de IaaS e Computação na Nuvem.

Adicionalmente, como será apresentado no Capítulo 7, esse trabalho propõe o uso de métricas multidimensionais de utilização de recursos no planejamento de capacidade da infraestrutura de forma a permitir uma seleção de diferentes tipos de instância no processo de provisão da aplicação ao longo do tempo. Isso revela um outro nível de decisão que pode aprimorar a forma como a infraestrutura se adapta às mudanças na carga de trabalho da aplicação em diferentes dimensões e a fim de evitar-se um potencial desperdício de recursos, que causa elevações nos custos de execução e provisionamento da aplicação.

A Tabela 3.1 apresenta um sumário da classificação de trabalhos existentes na literatura e discutidos nesse capítulo sobre o provisionamento automático e horizontal de aplicações em ambientes de IaaS. Essa classificação é conduzida em relação a diferentes aspectos inerentes ao processo de provisionamento automático, desde o modo de operação da técnica de provisionamento até o nível de intrusividade das soluções e o uso de múltiplas dimensões de recursos e de tipos de instancia de VM nas decisões de provisionamento realizadas. Dentre as soluções classificadas, aquelas em destaque na tabela são as que reúnem as características mais adequadas para compor um serviço de provisionamento em IaaS.

Tabela 3.1: Classificação de soluções de provisionamento automático de recursos.

Referência	Método	Operação	Técnica	Tipo de Métrica	Intrusividade	Multidimensional	Tipo de Instância
Vijayakumar et al. [64]	Vertical	Reativo	TS	Aplicação	Intrusivo	Não	—
Shen et al. [57]	Vertical	Proativo e Reativo	TS	Infra. virtual	Não intrusivo	Não	—
Spinner et al. [58]	Vertical	Proativo	TS	Aplicação e Infra. virtual	Intrusivo	Não	—
Dawoud et al. [22]	Vertical	Proativo	TS	Aplicação e Infra. virtual	Intrusivo	Sim	—
Yazdanov et al. [67]	Vertical	Proativo	TS + RL	Aplicação	Intrusivo	Sim	Dinâmico
Nanda et al. [47]	Vertical	Proativo	TS	Aplicação e Infra. virtual	Intrusivo	Sim	—
Gong et al. [29]	Vertical	Proativo	TS	Infra. virtual	Não intrusivo	Não	—
Bonvin et al. [9]	Horizontal e Vertical	Reativo	Regras	Aplicação	Intrusivo	—	Estático
Sharma et al. [56]	Horizontal e Vertical	Proativo	QT	Aplicação	Intrusivo	Não	Dinâmico
Beltrán et al. [8]	Horizontal e Vertical	Proativo	QT	Aplicação e Infra. virtual	Intrusivo	Não	Dinâmico

Tabela 3.1 – Continuação

Referência	Método	Operação	Técnica	Tipo de Métrica	Intrusividade	Multidimensional	Instância
Yang et al. [66]	Horizontal e Vertical	Proativo	Regras e TS	Aplicação	Intrusivo	Sim	Dinâmico
Marshall et al. [40]	Horizontal	Reativo	Regras e QT	Aplicação	Intrusivo	Não	Estático
Seung et al. [55]	Horizontal	Reativo	Regras	Aplicação	Intrusivo	Não	Estático
Fitó et al. [23]	Horizontal	Reativo	Regras	Aplicação	Intrusivo	—	Estático
Calcavecchia et al. [11]	Horizontal	Reativo	Regras	Aplicação	Intrusivo	—	Estático
Calheiros et al. [12]	Horizontal	Reativo	Regras	Aplicação	Intrusivo	—	Estático
Lim et al. [35]	Horizontal	Reativo	Regras e TS	Infra. virtual	Não intrusivo	Não	Estático
Ghanbari et al. [27]	Horizontal	Reativo	Regras	Infra. virtual	Não intrusivo	Não	Estático
Netto et al. [48]	Horizontal	Reativo	Regras	Infra. virtual	Não intrusivo	Não	Estático
Righi et al. [21]	Horizontal	Reativo	Regras	Infra. virtual	Não intrusivo	Não	Estático
Urgaonkar et al. [61]	Horizontal	Proativo e Reativo	Regras e QT	Aplicação	Intrusivo	Não	Estático
Ali-Eldin et al. [2]	Horizontal	Proativo e Reativo	QT	Aplicação	Intrusivo	—	Estático
Casalicchio et al. [15]	Horizontal	Reativo	Regras e QT	Aplicação e Infra. virtual	Intrusivo	Não	Estático
Roy et al. [52]	Horizontal	Proativo	TS	Aplicação	Intrusivo	Não	Estático
Loff et al. [36]	Horizontal	Proativo	TS	Aplicação	Intrusivo	Não	Estático

Tabela 3.1 – Continuação

Referência	Método	Operação	Técnica	Tipo de Métrica	Intrusividade	Multidimensional	Instância
Gandhi et al. [25]	Horizontal	Proativo	TS e QT	Aplicação e Infra. virtual	Intrusivo	Não	Estático
Verma et al. [63]	Horizontal	Proativo	TS	Aplicação e Infra. virtual	Intrusivo	Não	Dinâmico
Nguyen et al. [49]	Horizontal	Proativo	TS	Aplicação e Infra. virtual	Intrusivo	Sim	Estático
Jiang et al. [33]	Horizontal	Proativo	TS e QT	Aplicação	Intrusivo	—	Estático
Calheiros et al. [13]	Horizontal	Proativo	TS e QT	Aplicação	Intrusivo	—	Estático
Caron et al. [14]	Horizontal	Proativo	TS	Infra. virtual	Não intrusivo	Não	Estático
Morais et al. [44]	Horizontal	Proativo	TS	Infra. virtual	Não intrusivo	Não	Estático
Vasić et al. [62]	Horizontal	Proativo	TS	Infra. física	Não intrusivo	Sim	Dinâmico

Capítulo 4

Metodologia

Este capítulo descreve o processo metodológico adotado na realização dos estudos sobre o provisionamento automático como um serviço em IaaS e sobre o uso de diferentes tipos de instância de máquinas virtuais no provisionamento automático de aplicações horizontalmente escaláveis. Detalhadamente, o capítulo apresenta o processo experimental seguido pelas análises realizadas, descreve o modelo de simulação utilizado nessas análises, realiza uma caracterização de dados de utilização de recursos considerados nas simulações e define as métricas consideradas para avaliação do serviço de provisionamento e das técnicas de provisionamento reativas, proativas e baseada em múltiplos tipos de instâncias.

4.1 Processo Experimental de Avaliação

Os estudos realizados nesse trabalho sobre o provisionamento automático como um serviço em ambientes IaaS são conduzidos com base em experimentos de simulação. Esses experimentos simulam a operação de um serviço de provisionamento automático por meio de um modelo de laço de controle, que periodicamente decide sobre ações de provisionamento a serem realizadas sobre a infraestrutura de execução com base em dados de utilização de recursos dessa infraestrutura. Particularmente, as ações de provisionamento, adição ou remoção de VMs, são implementadas e simuladas. Essas ações são decididas pelas diferentes técnicas de provisionamento avaliadas, reativas e proativas. Além disso, os mecanismos de provisionamento baseados em múltiplos tipos de instância de VM também são analisados por meio de experimentos simulação, realizados em conjunto o com a execução do modelo

de provisionamento.

O modelo de simulação das técnicas de provisionamento automático e dos mecanismos de provisionamento baseados em múltiplos tipos de instância de VM foram implementados da linguagem de programação R [17]. A linguagem foi escolhida por dar suporte a análises estatísticas e a técnicas de visualização de dados, além de possuir um vasto ferramental sobre aprendizagem de máquina que objetiva o estudo de abordagens de provisionamento proativas por meio de modelos de predição de séries temporais. As simulações foram alimentadas com dados de utilização de recursos de aplicações reais e as análises foram realizadas com base em diferentes métricas de eficiência do serviço de provisionamento automático concebido nesse documento, que são diretamente relacionadas com os níveis de QoS das aplicações provisionadas e os custos de provisionamento decorrentes.

4.2 Modelo de Simulação

Os experimentos de simulação são alimentados com cargas de trabalho de aplicações reais representando diferentes cenários de provisionamento automático de aplicações horizontalmente escaláveis em ambientes IaaS. Desta forma, o modelo de simulação implementado na linguagem de programação R [1] simula a interação entre o sistema de provisionamento automático ou componente de controle mantido pelo provedor do serviço de provisionamento, a infraestrutura virtual adquirida do provedor de IaaS para executar a aplicação e os componentes de monitoramento e atuação normalmente disponibilizados pelos provedores de IaaS (por exemplo, a ferramenta de monitoramento Cloud Watch [53] e a interface EC2 de atuação sobre a infraestrutura [54] da Amazon). Uma visão geral da interação entre esses elementos baseada em um modelo de laço de controle pode ser encontrada na Figura 4.1.

O *sistema de provisionamento automático* atua como um componente de controle da infraestrutura virtual de execução, adicionando VMs quando necessário e removendo quando oportuno. Para tal, o sistema utiliza métricas do uso de recursos da infraestrutura obtidos a partir de um *monitor*, responsável por coletar essas informações no nível da infraestrutura virtual. As ações de provisionamento decididas pelo sistema de controle, com base nas

[1] O código fonte do simulador encontra-se publicamente disponível em `https://github.com/fabiomorais/ASaaS`.

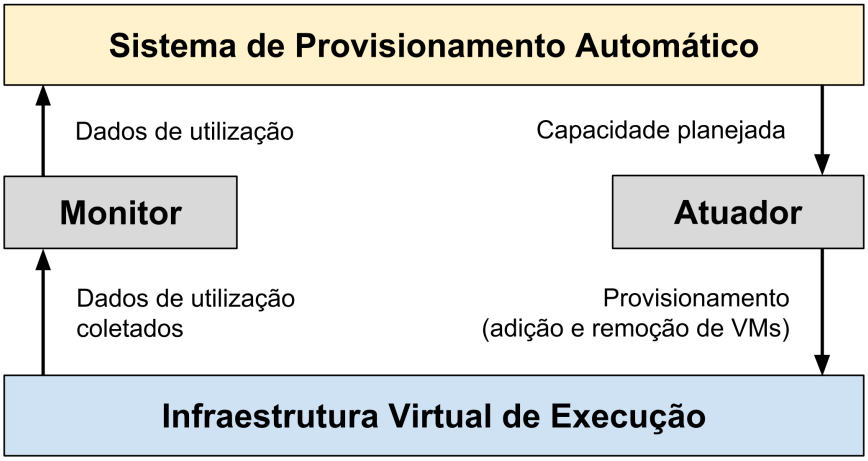


Figura 4.1: Visão geral da interação entre elementos do serviço de provisionamento automático em ambientes de IaaS.

métricas monitoradas, são executadas por um *atuador* que interage com a infraestrutura de execução, sendo capaz de adicionar ou remover VMs dessa infraestrutura. Essas decisões são tomadas com base no modelo de escalabilidade da aplicação provisionada, que modela a relação entre a quantidade de recursos alocados e o percentual de uso da infraestrutura para diferentes dimensões de recursos. Desta forma, a cada laço de controle o sistema e provisionamento avalia métricas da infraestrutura para provisionar a infraestrutura virtual alocada para executar a aplicação.

Assim, os possíveis efeitos de ações de provisionamento sobre a infraestrutura de execução também são simulados, como a relação entre utilização de recursos e capacidade da infraestrutura de execução descrita pelo modelo de escalabilidade da aplicação. Ou seja, simula-se como a utilização de recursos da infraestrutura responde às ações de provisionamento realizadas, seja pela adição ou remoção de VMs. Por questão de simplicidade, o modelo de simulação considera que essa relação ocorre de forma linear na simulação dos efeitos do provisionamento [2]. Além do mais, dado o viés não intrusivo da solução de provisionamento, o componente de controle opera desassociadamente da aplicação provisionada, de tal forma que as interações com as aplicações restringem-se unicamente à alocação ou desalocação de VMs (realizada pelo atuador) e à coleta de dados de utilização de recursos

[2]Experimentos realizados, a partir de aplicações intensivas em CPU, mostram que essa relação pode ser explicada através de um modelo de regressão linear para níveis de utilização de CPU abaixo de um determinado limiar, que representa o ponto de saturação da aplicação.

no nível da infraestrutura virtual alocada (realizada pelo monitor).

Os dados de utilização usados no processo de simulação consistem em séries temporais dos rastros de utilização de recursos, onde cada item da série corresponde à utilização média e a alocação de um dado recurso r (CPU, memória, etc.) para um intervalo de tempo t em questão. Especificamente, cada item no rastro consiste em uma tupla $\langle t, u, r, a \rangle$, onde, para o intervalo de tempo t, u é a utilização média do recurso r ($u \in \mathbb{R}, 0 \leq u \leq 1$) e a é a quantidade de unidades desse recurso alocados à aplicação nesse mesmo intervalo de tempo ($a \in \mathbb{R}^+$), quando o rastro de utilização da aplicação foi originalmente coletado. Desta forma, a demanda média original da aplicação (d) para o recurso r no intervalo de tempo t, em termos de número de unidades do recurso de interesse (por exemplo, núcleos de CPU ou gigabytes de memória), é computada como o produto da utilização média u e da alocação a do recurso r para o intervalo de tempo t, $d = u \times a$ ($d \in \mathbb{R}$).

O sistema de provisionamento opera com periodicidade configurável. No fim de cada período (ou laço) de controle pelo menos um novo item do rastro é lido e a demanda correspondente para um dado recurso é computada. Então, as demandas por recursos são usadas para simular a utilização real de cada um dos recursos da infraestrutura de execução com base nas VMs alocadas durante a simulação de provisionamento para esse período de controle. Por questão de simplicidade, o modelo de simulação assume que a infraestrutura executando a aplicação sempre apresenta uma composição homogênea de instâncias, i.e. uma infraestrutura que possui VMs de apenas um tipo de instância provendo a aplicação a cada intervalo de tempo. Além disso, consideramos um balanceamento de carga perfeito. Logo, cada VM alocada na infraestrutura apresenta a mesma média de utilização para cada um dos recursos em um mesmo intervalo de controle.

Considerando-se que uma iteração do laço de controle ocorre a cada intervalo de tempo t, tem-se que a utilização real simulada μ de cada VM para um dado recurso r no intervalo de tempo t é computada como sendo o mínimo entre 100% e $\frac{d}{\gamma}$, onde d é a demanda real da aplicação computada a partir nos dados originais do rastro para esse intervalo e γ é a capacidade alocada na simulação para o recurso r. A capacidade alocada γ é dada pelo produto entre o número de VMs alocadas n para executar a aplicação nesse intervalo durante o experimento de simulação e a capacidade disponível c no tipo de instância em uso para o recurso r considerado, $\gamma = n \times c$ ($\gamma \in \mathbb{R}$). Desta forma, se a utilização média simulada μ de

um recurso r no intervalo de tempo t é maior do que o limite de utilização l do recurso, como definido no SLA entre o provedor da aplicação e o serviço de provisionamento, então ocorre uma violação de SLO, indicando que a capacidade do recurso r atribuída não foi suficiente para lidar com a demanda da aplicação naquela dimensão específica de recurso.

Finalmente, com base nos dados de utilização de recursos simulados a partir da alocação realizada, o sistema de provisionamento realiza o planejamento de capacidade da infraestrutura de execução para o próximo intervalo de tempo. Esse planejamento é realizado a partir de ações de provisionamento baseadas na técnica de provisionamento adotada pelo sistema de provisionamento, seja ela reativa ou proativa, que definem a quantidade de recursos que deve ser alocada ou desalocada da infraestrutura de execução. O Algoritmo 1 mostra uma visão genérica das etapas realizadas no provisionamento automático baseado apenas em uma única dimensão de recurso.

Algoritmo 1: Etapas do processo de simulação do provisionamento automático com base em uma única métrica de recurso.

Entrada: rastro de utilização do recurso r e limite utilização l

```
para cada intervalo de controle t faça
    Ler um item do rastro de utilização referente ao recurso r (valores de u e a);
    Calcular a demanda original d da aplicação pelo recurso r (d = u × a);
    Calcular a capacidade alocada na simulação γ para o recurso r (γ = n × c);
    Calcular a utilização média simulada, μ = min(100%, d/γ);
    se μ ⩾ l então
        Computar violação de SLO do recurso r;
    fim
    Planejar a capacidade da infraestrutura e realizar provisionamento no curto prazo;
fim
```

No caso de uma abordagem reativa, a alocação de recursos é dirigida pelas regras de provisionamento predefinidas. Essas regras especificam a condição de disparo de uma ação de provisionamento considerando limiares de utilização de recursos, e a quantidade de recursos que devem ser provisionados, adicionados ou removidos, caso a ação seja disparada. Essa quantidade de recursos a serem adicionados ou removidos é dada em termos da quantidade

de VMs de um determinado tipo. Assim, a utilização simulada de cada recurso da infraestrutura é utilizada para confrontar os limiares definidos nas regras de provisionamento e decidir quando e quais ações de provisionamento serão realizadas.

Por outro lado, para abordagens de provisionamento proativo, o planejamento de capacidade é baseado em predições de cada uma das métricas de recursos coletadas da infraestrutura virtual e no tipo de instância considerado no provisionamento. Ou seja, o planejador avalia com base nas estimativas de demanda para o futuro próximo e no modelo de escalabilidade da aplicação quais são os requisitos de capacidade de recursos impostos pela aplicação no curto prazo para assegurar os níveis esperados de QoS da aplicação. Desta forma, a capacidade da infraestrutura é definida como o conjunto de instâncias do tipo selecionado necessário para prover a aplicação no futuro próximo, em todas as dimensões de recursos, com um desempenho aceitável e sem violações de SLO.

Assim, independente da abordagem de provisionamento empregada, a alocação de recursos ocorre sempre que necessário e em conformidade com as demandas simuladas da aplicação que são identificadas e possivelmente supridas a partir da técnica de provisionamento adotada. Em contrapartida, ao realizar a desalocação de recursos da infraestrutura o simulador considera o modelo de IaaS operado pela Amazon AWS, em que as VMs são tarifadas por hora, desta forma mesmo que o sistema de provisionamento decida pela remoção de VMs da infraestrutura em um dado intervalo de tempo, o desligamento de uma VM só é de fato efetuado se a decisão pela desalocação ocorrer em sincronia com horas completas de uso dessa VM. Caso contrário, ocorrem desalocações de VMs já tarifadas que poderiam suprir demandas futuras não esperadas.

4.3 Dados de Utilização de Recursos

Os experimentos de simulação realizados nesse trabalho são baseados em rastros de utilização de CPU e memória de aplicações reais pertencentes a dois conjuntos distintos de dados. O primeiro conjunto de dados, utilizado prioritariamente nas análises das técnicas de provisionamento reativo e proativo, corresponde a rastros de utilização de 30 diferentes aplicações com duração média de 8 meses. Cada um desses rastros consiste na medição de utilização de recursos de CPU e memória produzida por uma única aplicação, onde cada item do ras-

tro corresponde à média de utilização dos recursos a cada intervalo de 5 minutos. Esses dados são provenientes de uma parceria realizada entre Laboratório de Sistemas Distribuídos da UFCG e a HP[3] para o desenvolvimento de soluções de provisionamento automático de recursos em ambientes de Computação na Nuvem. No entanto, devido a questões de confidencialidade esses dados não encontram-se públicos e não podem ser disponibilizados.

Por outro lado, o segundo conjunto é composto por dados públicos da utilização de recursos de CPU e memória de aplicações executando em um *cluster* da Google durante um período de 29 dias [65]. Para formar esse conjunto foram consideradas apenas aplicações (*Jobs* no contexto dos dados) que são compostas de mais de uma tarefa. Essa filtragem é necessária para fortalecer a premissa de que as aplicações consideradas no estudo possuem a característica de escalabilidade horizontal. Após esta filtragem, o conjunto de dados considerado é composto por 956 diferentes aplicações, que corresponde a aproximadamente 40% do total. Esses rastros possuem dados sobre a utilização média e máxima de CPU e memória a cada intervalo de 5 minutos.

Todavia, diferentemente do primeiro conjunto de dados, as informações referentes à real capacidade dos servidores que formam o *cluster* não estão disponíveis e a utilização de recursos é relativa a uma máquina de referência com quantidades de recursos não conhecidas. Além disso, a duração dos rastros de utilização desse conjunto de dados não caracteriza a execução de aplicações horizontalmente escaláveis, que tipicamente executam por longos períodos de tempo. Por estes motivos, esse conjunto de dados é considerado unicamente para o estudo de mecanismos de provisionamento que utilizam múltiplos tipos de instância de VM para a execução da aplicação, onde essencialmente a métrica de proporção de utilização de recursos é considerada. As características dessa proporção para as diferentes aplicações do conjunto de dados da Google serão analisadas no Capítulo 7, referente ao estudo do uso de múltiplos tipos de instância no provisionamento automático.

Devido a não publicidade do primeiro conjunto de dados, foi realizada uma caracterização dos dados de utilização a fim de evidenciar a sua relevância no estudo do provisionamento automático de aplicações horizontalmente escaláveis em ambientes de IaaS. A Figura 4.2 mostra a função de distribuição acumulada (FDA) dos dados de utilização considerados para ambas as métricas. É possível observar que os rastros de utilização apresentam

[3]Projeto desenvolvido pela UFCG, HP Brasil e HP Fort Collins (EUA).

uma ampla diversidade de padrões e distribuições de utilização de CPU e memória, com níveis distintos de utilização entre os rastros. Além disso, os dados de CPU apresentam grande variação de amplitudes de utilização para uma mesma aplicação, enquanto que os dados de memória apresentam baixa variabilidade de utilização nesse caso.

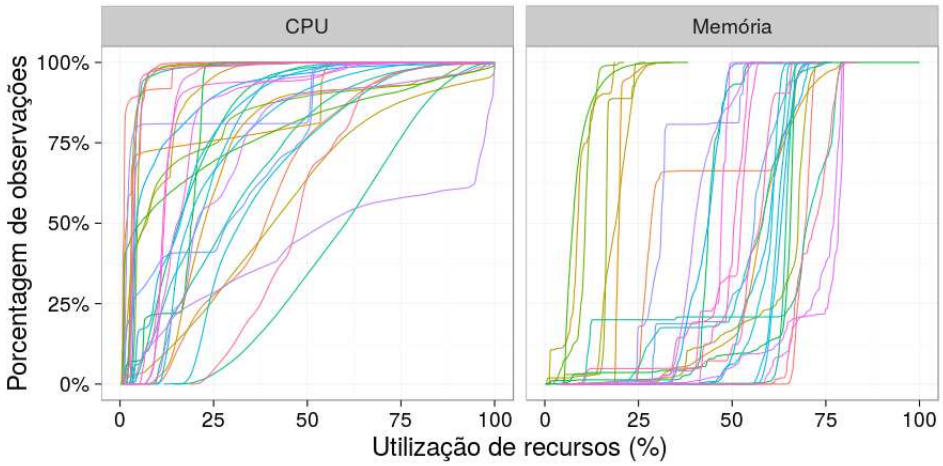


Figura 4.2: FDA da utilização de CPU e memória para os 30 arquivos de rastros de utilização das aplicações.

A variabilidade de amplitude de utilização de recursos para as diferentes métricas também pode ser observada nos diagramas de caixa do desvio padrão das utilizações por aplicação apresentados na Figura 4.3. A partir desses dados é possível verificar que existe uma maior variabilidade de utilização de CPU para uma mesma aplicação, com mediana do desvio padrão em torno de 15%, enquanto para os dados de utilização de memória a mediana é de cerca de 5% para as aplicações consideradas.

Complementarmente, com a aplicação da FDA para as variações de utilização de recursos, definidas como a diferença entre as utilizações no intervalo de tempo t e o intervalo de tempo $t + 1$, para todos os valores de t, obtém-se a caracterização dos rastros segundo a variação de utilização de recursos entre intervalos de tempo do rastro. Os resultados da FDA evidenciam que cada um dos rastros apresenta uma grande diversidade de pequenas variações de utilização de CPU e um pequeno percentual de variações intensas de utilização, positivas e negativas, enquanto que para memória os resultados reafirmam o que foi observado na análise anterior, que os dados de utilização de memória apresentam baixa variabilidade para uma mesma aplicação. Esses resultados podem ser observados nas Figuras 4.4 e 4.5,

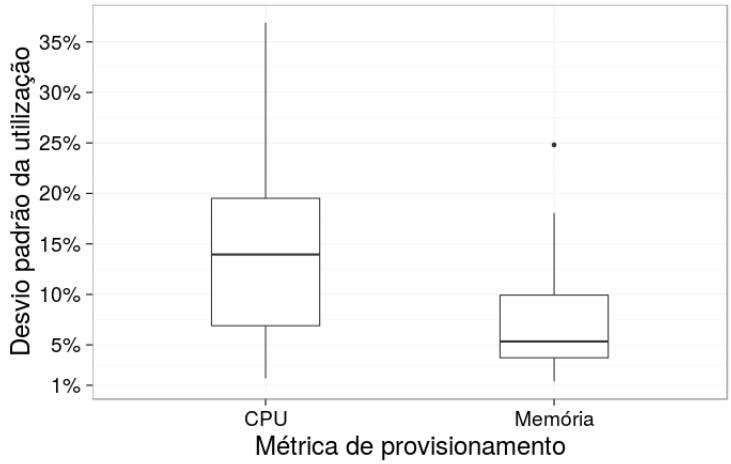


Figura 4.3: Diagrama de caixa do desvio padrão da utilização de recursos por aplicação considerada.

respectivamente.

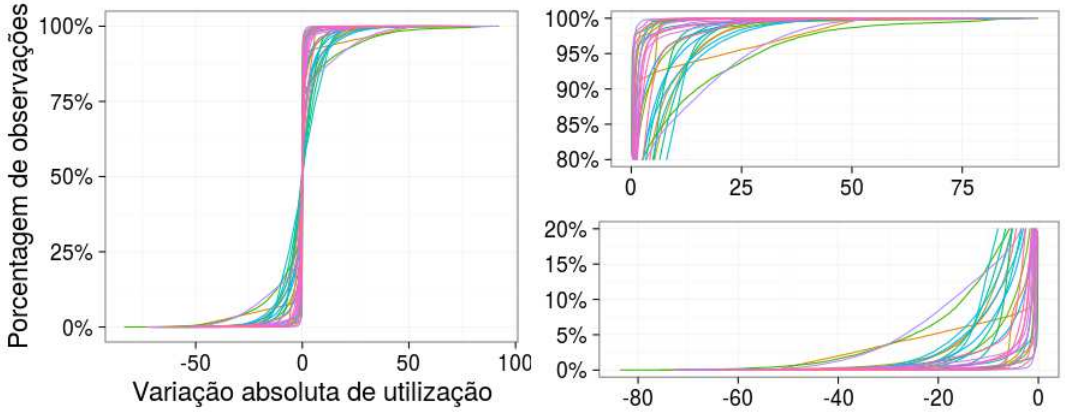


Figura 4.4: FDA da variação absoluta de utilização de CPU para 30 arquivos de rastros de utilização das aplicações.

Com base nessa caracterização dos dados de utilização de CPU e memória das 30 aplicações do conjunto de dados da HP é possível atestar que os rastros usados para alimentar os experimentos de simulação são de fato representativos para os estudos desenvolvidos nesse trabalho. Isso se deve principalmente pelos diferentes níveis de utilização de recursos apresentados entre as aplicações e a variabilidade das utilizações de CPU e memória para uma mesma aplicação, com variação mais significante para a métrica de CPU. Desta forma,

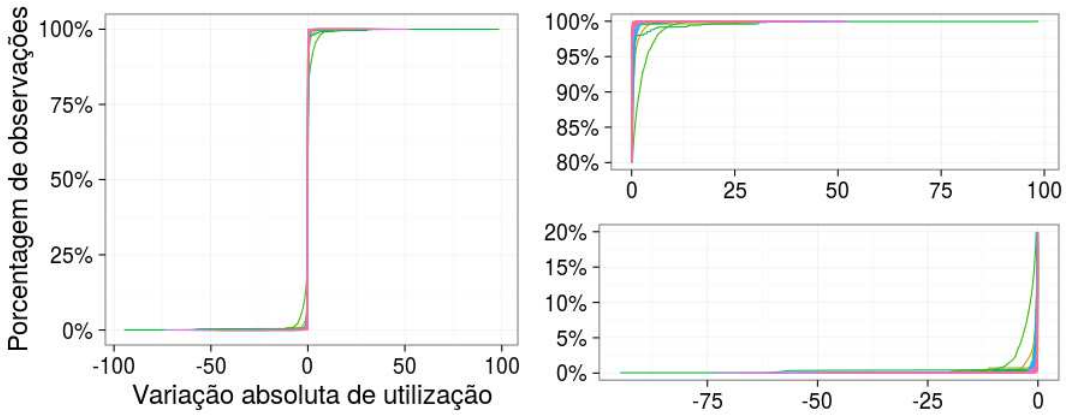


Figura 4.5: FDA da variação absoluta de utilização de memória para 30 arquivos de rastros de utilização das aplicações.

considera-se que esse conjunto de dados forma uma amostra significativamente diversificada de aplicações horizontalmente escaláveis em termos das características de suas cargas de trabalho.

4.4 Métricas de Avaliação

Durante os experimentos de simulação realizados neste estudo foram consideradas duas métricas independentes de avaliação de desempenho que estão diretamente relacionadas à avaliação dos objetivos de provisionamento. Uma dessas métricas visa medir se a QoS da aplicação provisionada se mantém em níveis aceitáveis ao longo da execução da aplicação. A outra métrica tem por objetivo avaliar a redução dos custos de provisionamento por conta da ação do sistema de provisionamento. Especificamente, a primeira consiste no número de violações de SLO durante o provisionamento, que corresponde ao número de intervalos de tempo do provisionamento onde a capacidade alocada de um recurso não foi suficiente para suprir a demanda do serviço para este recurso. Na prática, os intervalos de tempo em que a utilização de pelo menos um recurso alocado foi maior ou igual ao limite de utilização l para esse recurso, como definido no SLA entre o provedor da aplicação e o provedor do serviço de provisionamento.

A segunda métrica, por outro lado, corresponde especificamente ao custo de provisionamento e é calculada a partir do número de ciclos de tarifação de máquina (comumente de

1 hora) que foram necessários para executar a aplicação. No cenário em que o provisionamento faz uso de um ambiente público de IaaS, em que a aquisição de VMs é tarifada por ciclo de tarifação, esse custo de provisionamento pode ser monetizado em função dos preços estabelecidos pelo provedor de IaaS para o uso de uma VM de um determinado tipo por um ciclo de tarifação. Em geral, para um contexto de provisionamento automático, objetiva-se que esses custos de provisionamento sejam no mínimo inferiores aos praticados por um cenário de super provisionamento estático perfeito. Esse cenário consiste em uma abordagem não realista em que a infraestrutura é previamente provisionada com uma quantidade estática de recursos tal que permita que o limite de utilização de cada um dos recursos (l), definidos no SLA do serviço de provisionamento, seja respeitado, porém usando a quantidade mínima de recursos possível.

Também é considerado neste estudo um segundo conjunto de métricas de desempenho que estão relacionadas à eficiência do serviço de provisionamento em si, apesar de obtidas em função do desempenho das técnicas de provisionamento avaliadas em termos dos objetivos de provisionamento. Essas métricas tem por objetivo avaliar a capacidade das técnicas de provisionamento em compor um serviço de provisionamento automático que permita a diferentes aplicações explorar a elasticidade oferecida por ambientes de IaaS para uma execução eficiente. De forma específica, essas métricas correspondem: (i) à eficiência de implementação e configuração das técnicas de provisionamento automático reativas e proativas; (ii) o grau de generalidade e independência de características da carga de trabalho das aplicações provisionadas; e (iii) à capacidade de controle apresentada pelo serviço de provisionamento, que visa permitir o ajuste de configuração da abordagem de provisionamento para obter diferentes níveis de desempenho em termos dos objetivos de provisionamento.

Capítulo 5

Provisionamento como um Serviço Automático e Reativo

O objetivo desse capítulo consiste em avaliar o desempenho de soluções reativas como um serviço de provisionamento automático em termos de métricas finais de custo de execução e manutenção de QoS da aplicação provisionada. Além do mais, o desempenho destas soluções é analisado em relação a aplicabilidade da técnica reativa no cenário de provisionamento como um serviço, apontando diretrizes para criação de um serviço de provisionamento nesses moldes [1].

5.1 Introdução

O provisionamento automático e reativo é uma abordagem que atua na gerência de recursos virtuais alocados para uma aplicação. Essa abordagem modifica a alocação de recursos dedicados a uma aplicação em reação a mudanças no estado da aplicação provisionada e na infraestrutura virtual de execução da aplicação. Esse caráter reativo é derivado do fato desse tipo de abordagem utilizar apenas informações sobre o *estado atual* do sistema de provisionamento no processo de tomada de decisão sobre o provisionamento da aplicação, sem o uso ou manutenção de histórico de dados sobre o sistema. Essencialmente, essas abordagens utilizam um conjunto de regras de provisionamento para decidir quando e em qual quantidade

[1] Resultados parciais dessa análise, com base em métricas de utilização de CPU, foram publicadas na 35ª edição do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2017) [46].

de recursos virtuais a aplicação deve ser provisionada [38].

Essas regras são baseadas em limiares (do inglês, *threshold-based*) de métricas de desempenho do sistema de provisionamento (por exemplo, taxa de chegada de requisições, tempo médio de resposta, utilização de CPU, etc.) usados para definir a faixa desejável de valores que as métricas em questão devem assumir [20, 38]. Assim, a configuração de cada regra é composta de duas partes essenciais, que consistem na condição de disparo de uma ação de provisionamento e na ação de provisionamento associada propriamente dita. A parte condicional é composta por um conjunto de triplas ⟨métrica, limiar, operador condicional⟩ que definem as condições para o disparo de ações de provisionamento (por exemplo, uma ação é disparada se a métrica de utilização média de CPU for maior igual ao limiar de 70%). Enquanto que a segunda parte consiste na ação de provisionamento associada a condição de provisionamento definida (por exemplo, adicionar mais uma VM à infraestrutura de execução). Ou seja, se uma condição de provisionamento for satisfeita então uma ação de provisionamento associada, alocação ou desalocação de recursos, será disparada.

Idealmente, o provisionamento de uma aplicação é gerido por pelo menos duas regras, uma responsável por adicionar recursos à infraestrutura de execução quando necessário e outra por remover recursos que não estão mais em uso. Particularmente, para um ambiente de IaaS a definição de uma ação de provisionamento consiste na configuração de uma quantidade fixa de VMs de um determinado tipo de instância que deve ser adicionada ou removida da infraestrutura alocada para executar a aplicação caso uma condição de disparo de ação seja satisfeita. Esse tipo de solução é normalmente associado a modelos de laço de controle que periodicamente avaliam as condições de disparo das ações de provisionamento a partir de métricas de interesse monitoradas. Desta forma, a infraestrutura usada para executar a aplicação pode ser dinâmica e automaticamente provisionada por meio de regras de provisionamento reativo previamente definidas. Em um cenário de provisionamento automático como um serviço, a configuração dessas regras ao invés de ser realizada pelo responsável pela aplicação é realizada pelo provedor do serviço de provisionamento.

A técnica reativa consiste na principal solução de provisionamento automático oferecida pelos principais provedores do mercado de Computação na Nuvem de IaaS, como Amazon Web Services (AWS) [3], Rackspace [51] e Microsoft Azure [7], possivelmente pela simplicidade de construção de regras para automatizar o provisionamento de uma aplicação em

específico [38]. A Google [30] oferece um serviço de provisionamento automático que também baseia-se na técnica reativa, mas utiliza um modelo de provisionamento aparentemente mais sofisticado, não conhecido pelo público, para decidir a quantidade de VMs que devem ser alocadas ou desalocadas para manter os valores da métrica de interesse próximos de um objetivo predefinido. No entanto, essas soluções de provisionamento automático são oferecidas pelos provedores de Computação na Nuvem como uma ferramenta auxiliar para a otimização do uso de recursos alocados para a aplicação em execução, e não como um serviço a ser contratado por um potencial usuário e responsável pela aplicação implantada no ambiente de IaaS, que consiste no cenário de provisionamento como um serviço abordado por esse estudo. Ou seja, o provedor de Computação na Nuvem não assume obrigações com o provedor da aplicação executada sobre a manutenção da QoS da aplicação durante seu provisionamento, que mostram-se naturais para uma relação de prestação de serviço nesses moldes.

A abordagem reativa também é amplamente explorada na literatura através do uso de diferentes conjuntos de métricas de desempenho, configurações de limiares e ações de provisionamento [9, 11, 12, 23, 35, 40, 42, 64]. No entanto, para o provisionamento de aplicações com cargas de trabalho com significativa variabilidade no tempo considera-se que soluções dessa natureza apresentam desempenho inaceitável [37, 38]. Isso decorre da natureza reativa e pontual da solução e do fato da configuração das regras serem consideravelmente sensíveis a mudanças e tendências da carga de trabalho da aplicação, que geram a necessidade de ajustes frequentes mesmo quando um especialista na aplicação atua no processo de configuração [38]. Desta forma, equívocos na configuração das regras podem provocar situações de sub provisionamento, com degradação de QoS da aplicações e possíveis violações de SLO, ou de super provisionamento, com o desperdício de recursos adquiridos do provedor de IaaS.

Além dessas características, as soluções reativas, exploradas tanto pelo mercado quanto pela academia, em geral também baseiam-se em métricas intrusivas para realização do provisionamento, que são específicas da aplicação provisionada (como tempo de resposta, tamanho de fila, taxa de chegada e tipos de requisições) [9, 11, 12, 23, 40, 55]. Todavia, em um cenário de provisionamento como um serviço é complexo obter e usar informações específicas da aplicação. Isso deve-se principalmente a questões de generalidade e desacoplamento das aplicações provisionadas, dado que uma solução de provisionamento deve ser capaz de

funcionar adequadamente e de forma genérica para aplicações com diferentes características de demanda. Mas o impedimento de usar tais métricas também vem da necessidade de privacidade de informações sensíveis à aplicação provisionada. Desta forma, é fundamental o uso de métricas não intrusivas e disponíveis para qualquer aplicação em execução em um ambiente de IaaS. Essas métricas podem ser coletadas no nível da infraestrutura de execução, que idealmente encontra-se sob a gerência do provedor do serviço de provisionamento (como utilização de CPU, memória, disco, etc.).

Nesse capítulo é apresentado um estudo sobre o como realizar provisionamento automático e reativo como um serviço de Computação na Nuvem para aplicações com variações na carga de trabalho. O principal objetivo desse estudo consiste em avaliar o desempenho de soluções de provisionamento automático e reativo no que diz respeito a: (i) capacidade de manter a QoS da aplicação em nível adequado e à minimização dos custos de provisionamento, particularmente em comparação a um cenário de super provisionamento estático perfeito, em que não ocorrem violações de SLO; (ii) eficiência de configuração das regras de provisionamento e seleção de limiares; (iii) grau de generalidade e independência de características da carga de trabalho da aplicação provisionada; e (iv) capacidade de controle apresentada pela solução de provisionamento, que permite o ajuste de configuração das regras para obter diferentes níveis de desempenho em termos de métricas finais de custo de execução e QoS da aplicação.

A seguir, serão apresentadas diferentes análises do provisionamento reativo em diferentes cenários de provisionamento: (i) provisionamento *perfeito*, i.e. livre de erros de provisionamento; (ii) provisionamento prático a partir de métricas individuais de utilização de recursos, onde ocorrem potenciais situações de erro de planejamento de capacidade da infraestrutura de execução; e (iii) provisionamento baseado em múltiplas dimensões de recursos, quando diferentes métricas são usadas em conjunto na tomada de decisão sobre o provisionamento de uma aplicação.

5.2 Análise do Provisionamento Reativo Perfeito

A partir dos dados de utilização de recursos de CPU e memória é possível realizar uma análise de viabilidade de construção de uma solução de provisionamento reativo que atue

de forma perfeita durante o processo de execução da aplicação. Nesse cenário, informações sobre as demandas futuras da aplicação em termos dos recursos, para ambas as métricas, são previamente conhecidas. Desta forma, a cada ciclo de controle um planejador de capacidade perfeito decide e aloca, em função de uma das métricas, uma quantidade de VMs, com 1 núcleo de CPU e 1GB de memória, para manter a QoS da aplicação no próximo intervalo de tempo com o mínimo custo de alocação. O provisionamento perfeito simulado considera uma única métrica de utilização de recursos por execução.

Além do mais, nesse processo de provisionamento perfeito considerou-se que a QoS esperada para a aplicação é definida por SLOs que limitam a utilização máxima de recursos a 100%. Este limite de utilização foi usado para garantir que ao final do provisionamento perfeito de cada aplicação obtém-se um rastro de alocação de VMs no tempo por métrica de provisionamento considerada. Cada rastro corresponde ao provisionamento, com base em umas das métricas, com o menor custo possível de execução sem a ocorrência de violações dos SLOs. A descrição dos parâmetros considerados nesse cenário de provisionamento perfeito pode ser encontrada na Tabela 5.1

Tabela 5.1: Parâmetros utilizados na execução do provisionamento perfeito.

Parâmetro	**Valor**
Quantidade de rastros de utilização	30 arquivos
Duração média dos rastros de utilização	8 meses
Periodicidade de provisionamento	5 minutos
Quantidade mínima de VMs alocadas	1
Limite de utilização para violações de SLO	100%
Tamanho da fatia da tarifação	60 minutos
Métrica de provisionamento	Individualmente CPU e memória

Com base nesses dados de alocação no tempo é possível determinar, a partir de uma análise *post-mortem*, as configurações de regras de provisionamento reativo que seriam necessárias, a cada intervalo de provisionamento, para gerar um provisionamento perfeito da aplicação para cada uma das métricas individualmente. Ou seja, para cada ação de provi-

sionamento efetuada no provisionamento perfeito baseado em um dado recurso infere-se a regra reativa correspondente, limiar de utilização do recurso e quantidade de VMs alocadas ou desalocadas, que seria responsável por gerar a ação de provisionamento naquele intervalo de tempo no caso do provisionamento perfeito da aplicação.

Nesse processo, considera-se um tempo necessário para que o provisionamento horizontal de recursos faça efeito. Este *tempo de provisionamento* corresponde ao tempo de criação/remoção de VMs, de implantação e configuração da aplicação e balanceamento da carga entre os recursos alocados. Desta forma, para uma ação de provisionamento perfeito esteja efetivada no intervalo de tempo t a configuração de provisionamento necessária para resultar nessa mudança deve ser aplicada no intervalo $t - 1$ com base em dados obtidos no intervalo de tempo $t - 2$. Especificamente, utilizou-se um tempo de provisionamento de 5 minutos em conformidade com a periodicidade disponibilizada nos rastros de utilização disponíveis. Além disso, 5 minutos também é um tempo razoável para a efetivação de uma ação de provisionamento horizontal, tempo este verificado através de análise com experimentos de medição (ver Apêndice A).

Nas próximas seções será apresentada uma visão geral das características de configuração de regras da abordagem reativa perfeita no provisionamento automático. Essas características serão abordadas a partir de análises sobre: (a) a predominância das configurações; (b) a frequência de mudança de limiares de provisionamento; e (c) a relação entre a carga de trabalho das aplicações e a variabilidade das configurações de provisionamento.

5.2.1 Predominância de configuração de regras de provisionamento

A partir das configurações inferidas do processo de provisionamento perfeito descrito anteriormente é possível ter a informação de todas as diferentes configurações de regras de provisionamento reativo que foram necessárias para gerar o provisionamento perfeito de todas as aplicações e de cada uma das aplicações individualmente, para cada uma das métricas consideradas. No entanto, como os valores dos limiares inferidos encontram-se no domínio dos $\mathbb{R}$ ocorre um aumento considerável do universo de possibilidades de configuração, com uma variedade significativa de configurações de provisionamento, o que reduz a significância de qualquer análise sobre a predominância e frequência de uso de configurações provisionamento reativo. Para lidar com essa questão, os valores dos limiares inferidos a partir do

provisionamento perfeito foram discretizados em faixas de valores com tamanho de 5% (por exemplo, um limiar de 31% passa a pertencer faixa do limiar de 31 a 35%). Desta forma, os valores de limiares originalmente inferidos foram categorizados em 20 grupos de limiares, com valores entre [1, 5%], [6, 10%], até [95, 100%].

Frequência de uso de configurações de provisionamento

Com base nos dados derivados dessa discretização de limiares de provisionamento reativo foi realizada uma análise da frequência de uso desses limiares no provisionamento perfeito de cada uma das aplicações, considerando separadamente diferentes dimensões de recursos (utilização de CPU e memória). A Figura 5.1 apresenta o diagrama de caixa da maior frequência de uso de um limiar para cada uma das 30 aplicações consideradas nesse estudo, agrupadas por métrica e ação de provisionamento. Os resultados revelam que para 50% das aplicações provisionadas com base em memória o limiar mais predominante apareceu em 45% das vezes para adição de VMs e em 60% das vezes para remoção de VMs. Enquanto que para o provisionamento baseado em CPU, a mediana de frequência dos limiares mais predominantes foi de aproximadamente 17% considerando os dois tipos de ação de provisionamento.

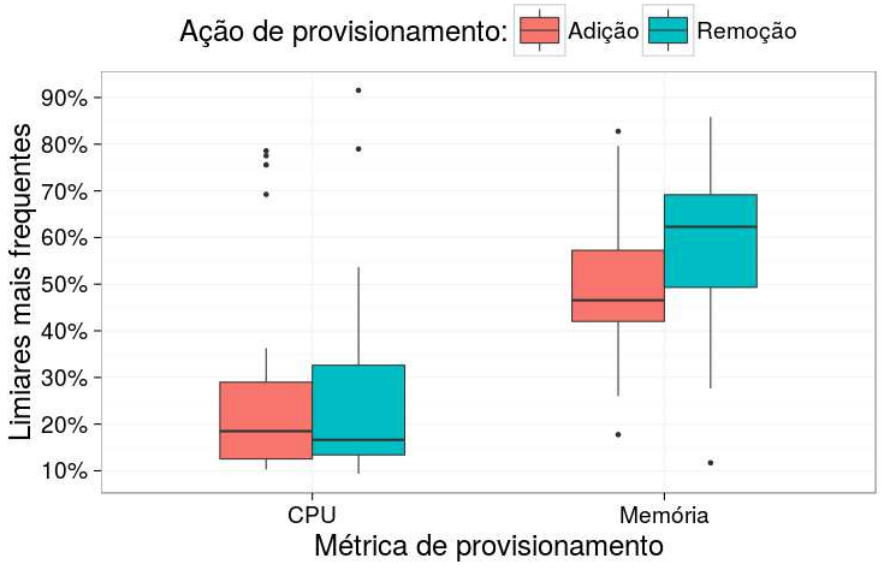


Figura 5.1: Maior frequência de uso dos limiares de provisionamento reativo para cada uma das aplicações e métricas.

Os resultados mostram que para o provisionamento baseado em memória observa-se uma predominância significativa do uso de limiares de provisionamento. Considerando ambos os tipos de ação de provisionamento, a média de uso dos limiares mais predominantes por aplicação gira em torno de 50%, o que favorece a eficiência de configuração da solução de provisionamento para esse caso. Por outro lado, essa constatação não se aplica para o provisionamento baseado na métrica de CPU, no qual não foram encontrados limiares que predominaram de forma significativa. Desta forma, hipotetiza-se sobre a possibilidade de que a maior predominância de limiares para o provisionamento baseado em memória esteja relacionada com o fato das rastros de utilização de memória apresentarem menor variabilidade por aplicação do que os dados de CPU [2].

A quantidade de regras necessárias para realizar o provisionamento perfeito considerando CPU é muito maior do que a quantidade de regras para realizar o provisionamento perfeito com base em memória. A Figura 5.2 apresenta o diagrama de caixa dessas quantidades, agrupadas por métrica e ação de provisionamento. Os resultados mostram que a quantidade média de diferentes configurações por aplicação para o provisionamento baseado em CPU e memória é de cerca de 17 e 9 configurações, respectivamente. Ou seja, de fato a solução reativa mostra-se mais eficiente em termos de configuração de limiares para o provisionamento baseado em memória, se comparado com o provisionamento baseado em CPU, que necessita de uma quantidade maior de diferentes configurações de limiares por aplicação no provisionamento perfeito.

Por outro lado, considerando não apenas os limiares de provisionamento no cálculo da frequência de uso de configurações mas também a quantidade de VMs provisionadas por ação efetuada, observa-se a necessidade de mais regras de provisionamento, o que reduz a eficiência de configuração desse tipo de sistema. Na Figura 5.3 é possível ver o diagrama de caixa da maior frequência de uso de regras completas de provisionamento (composta de limiar e ação) para cada uma das 30 aplicações utilizadas nesse estudo. Nesse caso, não se observa predominância de regras de provisionamento por aplicação. As regras mais usadas são observadas em menos de 10% dos casos. Ou seja, não é possível afirmar que existe uma regra padrão que possa ser usada de forma predominante por aplicação provisionada [3].

Assim, apesar da frequência considerável de uso de um mesmo limiar no provisiona-

[2]O estudo dessa relação entre carga de trabalho e variabilidade de configurações de provisionamento será

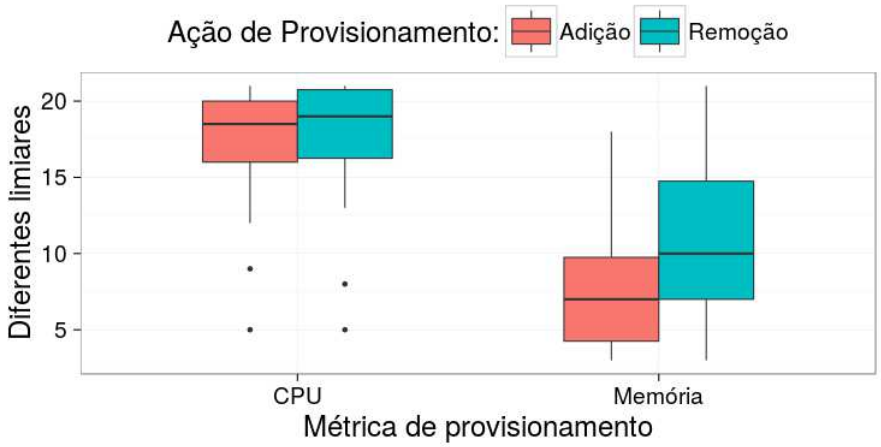


Figura 5.2: Quantidade de diferentes limiares de provisionamento reativo por aplicação e métrica considerada.

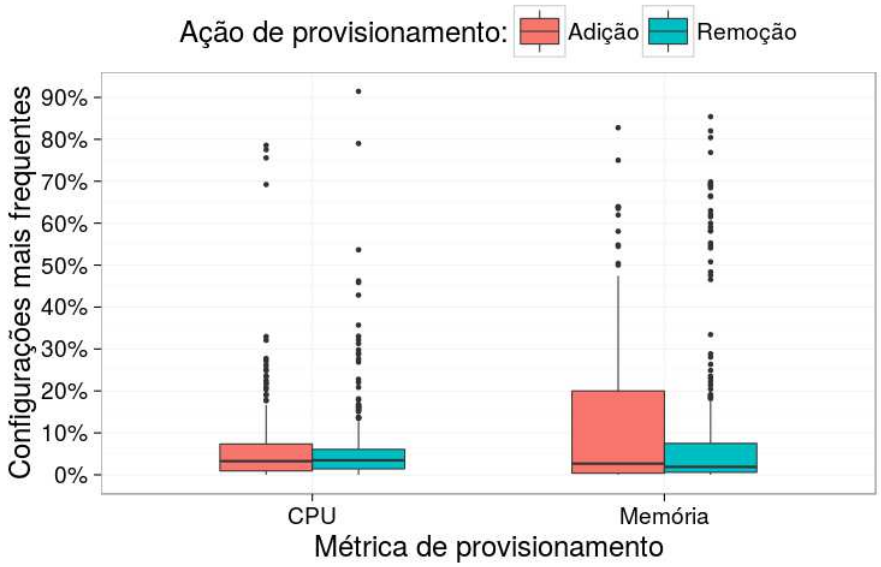


Figura 5.3: Maior frequência de uso das configurações de provisionamento reativo, limiares e quantidade de VMs provisionadas, para cada uma das aplicações e métricas.

mento baseado em memória, não foi observada predominância significativa de configuração

abordada em outro ponto desse capítulo, na Seção 5.2.3.

[3] A questão sobre a escolha da quantidade de VMs a ser provisionada por ação de provisionamento é abordada em maiores detalhes na Seção 5.2.1.

de limiares no provisionamento baseado em CPU e de regras de provisionamento para uma mesma aplicação, muito menos as para diferentes aplicações consideradas. Além disso, os resultados sobre a quantidade de diferentes limiares por aplicação reiteram a ineficiência de configuração da abordagem para o provisionamento reativo perfeito dessas aplicações. Desta forma, no que se refere à predominância de configuração de regras de provisionamento, o uso de uma configuração padrão no provisionamento reativo perfeito mostra-se inviável.

Distribuição de configuração de VMs provisionadas

Nesta seção é realizada uma análise sobre a quantidade de diferentes ações de provisionamento (configurações de quantidades de VMs a serem adicionadas/removidas segundo as regras de provisionamento perfeito das aplicações). Considerando todas as regras de provisionamento usadas para provisionar as aplicações, a quantidade de VMs usadas no provisionamento apresenta variabilidade não significativa, com aproximadamente 85% das ações realizando o provisionamento, adição ou remoção de recursos, de apenas 1 VM por ação. Além disso, em praticamente todas as ações de provisionamento, cerca de 99%, são provisionadas menos de 5 VMs. Apenas 15% de todas as ações de provisionamento realizadas fazem uso de mais de 1 VM no provisionamento, com valores de quantidade de VMs provisionadas concentradas em uma faixa restrita de valores entre duas e 4 VMs. Apesar disso ocorrer em apenas 15% das regras, traz para o sistema de provisionamento uma variabilidade considerável em termos de configuração das regras.

A variabilidade de configuração mostra-se mais acentuada ao considerar a quantidade de VMs provisionadas por limiar de provisionamento, quando agrupado por aplicação provisionada. Nesse caso, observa-se que são utilizadas em média duas diferentes configurações de VMs provisionadas por limiar, como pode ser percebido no diagrama de caixa da Figura 5.4. Nessa figura, as quantidades de VMs a serem adicionadas/removidas são apresentadas, agrupadas por métrica e tipo de ação de provisionamento. Além do mais, apesar do máximo de VMs provisionadas por limiar ser inferior quando baseado em memória, as medianas e o 75-percentil tendem a ser iguais para ambas as métricas e as ações de provisionamento ficam em torno de duas diferentes configurações de quantidade de VMs provisionadas. Isso indica que a maioria das regras de provisionamento utilizadas por aplicação apresentam ações diferentes para a mesma configuração de limiar de provisionamento.

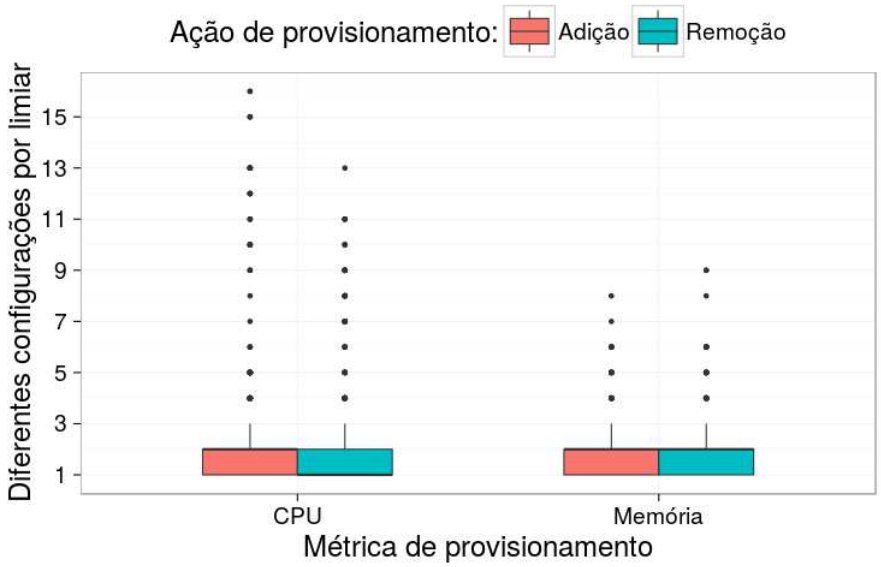


Figura 5.4: Quantidade de diferentes configurações de VMs provisionadas por aplicação e limiar de provisionamento.

Se regras com a mesma condição de gatilho com ações diferentes são necessárias, torna-se impraticável configurar um sistema de provisionamento que funcione próximo ao perfeito. Desta forma, a configuração das ações disparadas por cada regra, mesmo que isoladamente por limiar, apresenta-se como mais um fator de ineficiência de configuração para a solução de provisionamento reativo buscada nesse estudo.

5.2.2 Frequência de transição entre configurações de limiares

Um outro ponto relevante da análise de eficiência de configuração da abordagem reativa consiste na variabilidade temporal das configurações de regras de provisionamento usadas. Além da necessidade de decidir quais as regras que devem ser utilizadas para cada aplicação, também deve-se conhecer como ocorrem as mudanças nessas regras durante o tempo de provisionamento automático da aplicação. A Figura 5.5 mostra o percentual de ocorrência de transições entre configurações de limiares de utilização diferentes, que consiste na mudança de limiares entre intervalos de tempo consecutivos em que ações de provisionamento de um mesmo tipo foram realizadas.

Mesmo para um cenário menos restritivo de configuração, baseado apenas nos limiares

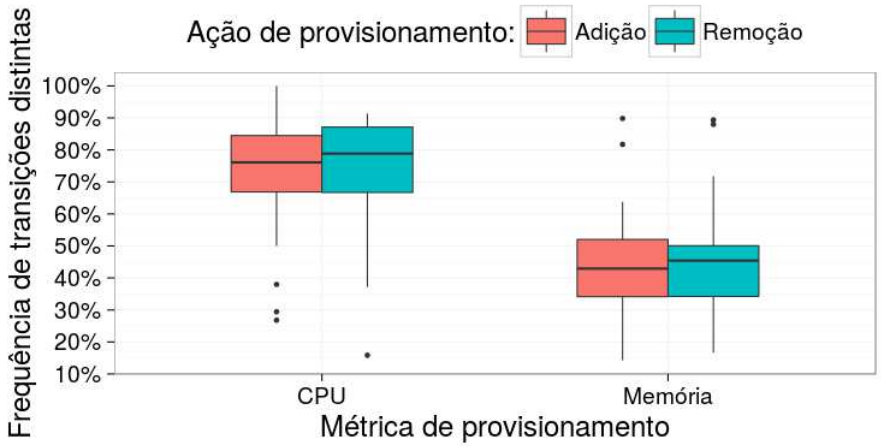


Figura 5.5: Frequência de transições entre diferentes limiares de utilização por aplicação e métrica de provisionamento.

de utilização e sem considerar a quantidade de VMs provisionadas, o percentual de transição entre diferentes limiares dentre as transições efetuadas é considerado significativo. Os resultados mostram que para o provisionamento baseado em CPU a mediana do percentual de transição é de 77%, enquanto que para o provisionamento baseado em memória esse percentual é inferior, em torno de 44%, considerando as diferentes ações de provisionamento. O que remonta à hipótese de uma relação entre a carga de trabalho das aplicações e a variabilidade de configuração das regras de provisionamento reativo.

Ao observar o diagrama de caixa da frequência da transição mais predominante por aplicação na Figura 5.6 é possível ter uma visão mais objetiva do impacto da variabilidade de configuração no provisionamento automático perfeito. Os dados mostram que para todos os cenários observados, com diferentes aplicações, métricas e ações de provisionamento, 90% das transições mais frequentes entre diferentes limiares ocorrem em menos de 12% do total de transições realizadas. Ou seja, mesmo para limiares bastante utilizados no provisionamento de cada aplicação o uso consecutivo deste limiares ocorre em uma frequência não significativa. Com base nisso, é possível atestar que o percentual de ocorrência de uma mesma transição entre limiares desfavorece a eficiência de configuração de uma solução de provisionamento automático e reativo para essas aplicações, principalmente para o provisionamento baseado em CPU onde a variabilidade das configurações mostra-se mais intensa.

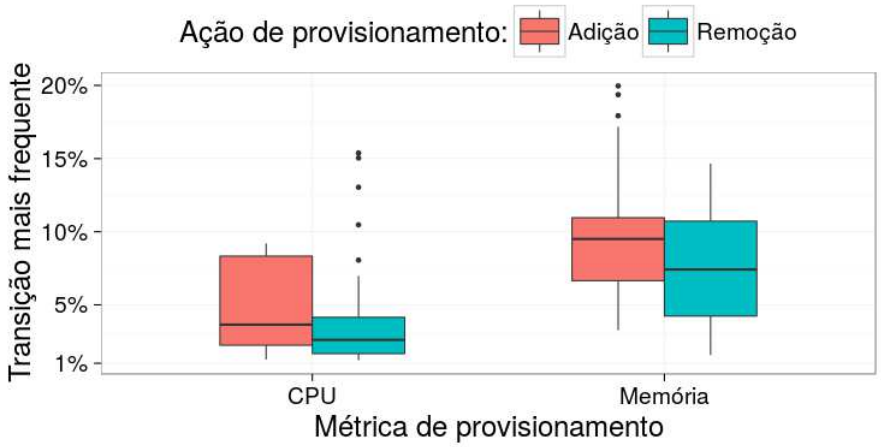


Figura 5.6: Frequência das transições com maior ocorrência no provisionamento das aplicações para diferentes métricas e ações de provisionamento.

5.2.3 Relação entre carga de trabalho e configuração de regras

Foi observada uma relação entre a variabilidade carga de trabalho das aplicações para as diferentes dimensões de recursos e a necessidade de diferentes configurações para o provisionamento perfeito de uma mesma aplicação. Isso confirma a hipótese de existir relação entre a carga de trabalho da aplicação e a configuração de provisionamento reativo perfeito, levantada em seções anteriores. Essa relação foi primeiramente verificada pela análise de correlação entre o desvio padrão do uso de recursos por aplicação, que consiste na quantidade de recursos demandados por intervalo de tempo (por exemplo, número de núcleos de CPU), e a quantidade de operações de provisionamento necessárias para gerar o provisionamento perfeito dessas aplicações. A Tabela 5.2 mostra os valores desses coeficientes calculados a partir dos métodos de correlação de *Spearman* e *Kendall* considerando as diferentes métricas de provisionamento utilizadas nesse trabalho.

Tabela 5.2: Correlação entre a variabilidade de carga de trabalho das aplicações e a quantidade de operações de provisionamento necessárias ao provisionamento perfeito baseado em diferentes métricas de provisionamento.

Métrica de provisionamento	**Correlação de Spearman**	**Correlação de Kendall**
CPU	0.91	0.75
memória	0.76	0.57

Como pode ser observado, existe uma correlação forte entre a variação da carga de trabalho da aplicação e o número de ações de provisionamento realizadas por aplicação, para ambas as métricas de provisionamento, que demonstra uma relação forte entre a necessidade de realizar ações de provisionamento e a variação da carga de trabalho da aplicação. Em outras palavras, quanto maior for a variação da carga de trabalho da aplicação mais operações de provisionamento são necessárias para provisionar perfeitamente e reativamente a aplicação. Além do mais, essa relação mostra-se ainda mais forte para o provisionamento baseado em CPU, cujas cargas de trabalho apresentam maior variabilidade de utilização de recursos para uma mesma aplicação.

Além disso, também existe uma correlação significativa entre a variabilidade de carga de trabalho e o número de diferentes configurações de limiares usados nas condições de provisionamento. A Tabela 5.3 apresenta os coeficientes de correlação, considerando diferentes métodos, entre o desvio padrão do uso de recursos de CPU e memória e a quantidade de diferentes configurações de limiares de provisionamento usadas no provisionamento perfeito de cada aplicação considerada. Os resultados demonstram a existência de uma relação positiva moderada entre as características da carga de trabalho da aplicação e a configuração de limiares de provisionamento, hipótese levantada em análises realizadas em seções anteriores. Ou seja, quando maior for a variação na carga de trabalho da aplicação maior é a necessidade de configurar diferentes limiares no provisionamento perfeito.

Tabela 5.3: Correlação entre a variabilidade de carga de trabalho das aplicações e a quantidade de diferentes configurações de limiares necessárias ao provisionamento perfeito baseado em diferentes métricas de provisionamento.

Métrica de provisionamento	**Correlação de Spearman**	**Correlação de Kendall**
CPU	0.61	0.45
memória	0.66	0.49

Essa relação é potencializada ao considerar-se a configuração da quantidade de VMs provisionadas da regra de provisionamento para o cálculo dessa correlação. Nesse caso, existe uma relação positiva forte entre a variabilidade da carga de trabalho e a configuração de regras de provisionamento, compostas da configuração de limiares e da quantidade de VMs provisionadas. Aplicações com maiores variações na sua carga de trabalho necessitam de um número maior de diferentes configurações de regras de provisionamento no cenário perfeito. Os coeficientes de correlação observados entre o desvio padrão do uso de recursos e a quantidade de diferentes regras de provisionamento estão descritos na Tabela 5.4.

Tabela 5.4: Correlação entre a variabilidade de carga de trabalho das aplicações e a quantidade de diferentes configurações de regras de provisionamento para o provisionamento perfeito baseado em diferentes métricas.

Métrica de provisionamento	**Correlação de Spearman**	**Correlação de Kendall**
CPU	0.88	0.71
memória	0.80	0.63

Desta forma, assume-se que tanto a configuração dos limiares de provisionamento reativo quanto a configuração de regras de provisionamento completas (limiares e quantidades de VMs) são dependentes de características específicas da carga de trabalho das aplicações provisionadas, para as diferentes métricas de provisionamento. Esse resultado limita significativamente o desempenho da abordagem reativa em relação à eficiência e à capacidade de generalidade de configuração para um conjunto heterogêneo de aplicações.

5.2.4 Sumário de resultados sobre provisionamento reativo perfeito

A análise do provisionamento reativo perfeito, em que o uso da infraestrutura de execução é otimizado sem a ocorrência de violações de SLO, foi realizada com base na eficiência de configuração das regras de provisionamento reativo, em termos da predominância e generalidade da configuração de limiares e da quantidade de recursos provisionados a cada ação de provisionamento. Como resultado, a abordagem reativa mostrou-se bastante ineficiente nesse sentido e consideravelmente dependente de características específicas de consumo de recursos de cada uma das aplicações consideradas.

Verificou-se principalmente a ausência de configuração comum a todas (ou pelo menos maior parte) das aplicações. Para cada uma das aplicações e métricas de provisionamento consideradas, não há predominância de configuração de regras, seja para a configuração de limiares ou para a quantidade de VMs provisionadas a cada ação de provisionamento disparada. Em geral, cada aplicação requer um grande número de configurações de regras para o provisionamento perfeito. Percebe-se que as configurações de regras mudam não apenas pela diversidade aplicações, mas pela mudança da carga de trabalho destas, sendo necessária a mudança periódica das configurações durante o provisionamento de uma mesma aplicação.

Além do mais, a generalidade da solução de provisionamento é limitada, uma vez que existe relação entre a configuração das regras de provisionamento e as características específicas da carga de trabalho das aplicações. Existe uma relação forte entre o perfil de variação da carga de trabalho das aplicações e as configurações necessárias ao provisionamento perfeito. Desta forma, configurações padrão mostram-se inviáveis tanto para o provisionamento de aplicações específicas quanto para o provisionamento de um conjunto diverso de aplicações.

Assim, considera-se ser não factível a construção de um serviço de provisionamento automático e reativo que seja eficiente em termos de custo de provisionamento e da ocorrência de violações de SLO, e que ao mesmo tempo seja genérico e de fácil configuração, especialmente se for esperado um desempenho pelo menos semelhante ao obtido no provisionamento perfeito da infraestrutura de execução. Todavia, a seguir serão realizadas análises práticas de desempenho dessa abordagem em termos de métricas de custo de execução e QoS da aplicação provisionada.

5.3 Análise Prática do Provisionamento Reativo

Os objetivos de provisionamento automático de aplicações em ambientes de IaaS envolvem um *trade-off* entre a redução do custo de provisionamento e a redução do número de violações de SLO. No cenário de provisionamento perfeito abordado na seção anterior, esses objetivos conflitantes são otimizados, gerando uma execução da aplicação sem violações de SLO com o menor custo possível de execução. Em uma abordagem de provisionamento prática, o controle dos diferentes objetivos dessa relação deve ser realizado a partir da configuração dos limiares de provisionamento da abordagem reativa. Nesse sentido, o controle se dá da seguinte forma:

- Prioriza-se a redução de custos de provisionamento ao elevar-se o limiar de provisionamento, seja ele de adição ou remoção. Com limiares maiores maior é probabilidade de remoção de VMs e menor é a probabilidade de adição de novas VMs. Por consequência, esse cenário aumenta as chances de ocorrência de violações de SLO devido à insuficiência de recursos;

- Prioriza-se a redução da ocorrência violação de SLOs ao reduzir-se o limiar de provisionamento, seja ele de adição ou remoção. Nesse caso VMs são adicionadas ao primeiro sinal de aumento de utilização e só são removidas quando a utilização está suficientemente baixa. Desta forma, ocorrem reduções no número de violações de SLO ao ônus da elevação de custos derivados da adição de recursos;

Nesta seção é realizada uma análise de desempenho da abordagem reativa em termos dos objetivos de provisionamento: o percentual de violações de SLO obtido no provisionamento e do custo de provisionamento. O custo de provisionamento é analisado em relação ao custo do provisionamento perfeito (abordado na seção anterior) e do super provisionamento estático perfeito para cada aplicação. Esta última abordagem consiste no provisionamento prévio da infraestrutura com uma quantidade estática de recursos capaz de suprir os picos de demanda da aplicação ao longo do tempo, de forma a garantir que não ocorram violações de SLO com o menor custo possível. Para tal, o modelo de simulação foi utilizado para simular o provisionamento reativo das aplicações considerando um variedade de configurações de limiares de provisionamento para adição e remoção de VMs, com base individualmente em métricas de utilização de CPU e memória.

Nessa simulação de provisionamento, a capacidade da infraestrutura utilizada para executar a aplicação é dinâmica e periodicamente modificada com base no disparo de ações de provisionamento associadas a limiares de utilização de recursos, previamente definidos em regras de provisionamento reativo. Adicionalmente, com base no desempenho obtido por cada configuração de provisionamento, é possível avaliar o desempenho da abordagem reativa quanto ao grau de controle apresentado pela solução, ou seja, em termos da capacidade das regras de provisionamento a serem configuradas de forma a atender os diferentes e conflitantes objetivos de provisionamento. Nas próximas seções os resultados dessas análises são apresentados e discutidos.

5.3.1 Objetivos conflitantes e desempenho do provisionamento

O *trade-off* entre os objetivos de provisionamento fica evidente ao analisar o provisionamento reativo considerando diferentes configurações de limiares. Foram simulados cenários de provisionamento com limiares de adição de VMs variando de 20% a 90%, em passos de 10%, e com limiares de remoção variando de 10% a 80%, também em passos de 10%, para ambas as métricas de provisionamento consideradas [4]. Desta forma, o simulador foi configurado com cada um dos pares possíveis de configuração de limiares de adição e remoção, formando um experimento fatorial completo com remoção de cenários inválidos de configuração de limiares. Além disso, as ações de provisionamento foram configuradas para adicionar ou remover 1 VM por operação[5], uma vez que essa mostrou-se ser a configuração, de quantidade de VMs provisionadas, predominante no provisionamento reativo perfeito das aplicações consideradas.

O desempenho em termos do número de violações de SLO foi computado como sendo o percentual de intervalos de tempo em que o nível de utilização da infraestrutura atingiu o limite de 100%, para ambas as métricas observadas. Enquanto que o custo é definido pela quantidade de horas-máquina (VMs) utilizadas para executar a aplicação ao longo do tempo, comparado com os custos apresentados pelos cenários base de provisionamento perfeito, abordado na seção anterior, e de super provisionamento estático perfeito. No processo de simulação, com o intuito de respeitar o tempo de provisionamento horizontal e para estar

[4] Limiares de adição são sempre configurados com valores maiores que os dos limiares de remoção.

[5] As VMs provisionadas possuem 1 núcleo de CPU e 1GB de memória.

em conformidade com a periodicidade dos dados de utilização considerados, assume-se que ações de provisionamento necessitam de um tempo de provisionamento de 5 minutos para serem efetivadas. Desta forma, cada ação de provisionamento disparada no intervalo de tempo t só estará operacional, executando a aplicação, no início do intervalo de tempo $t + 1$ intervalo.

As Figuras 5.7 e 5.8 apresentam os resultados de desempenho das configurações de provisionamento para o provisionamento reativo baseado individualmente em CPU e memória, respectivamente. Os gráficos apresentam o desempenho das configurações em termos do percentual de violações de SLO e do custo do serviço relativo aos cenários base. Em relação ao cenário super provido, custos negativos correspondem a economias de custo por parte do provisionamento reativo simulado. No eixo X são apresentados os limiares de adição usados nas simulações e as cores de cada barra indicam os limiares de remoção considerados (ver as legendas acima do gráfico).

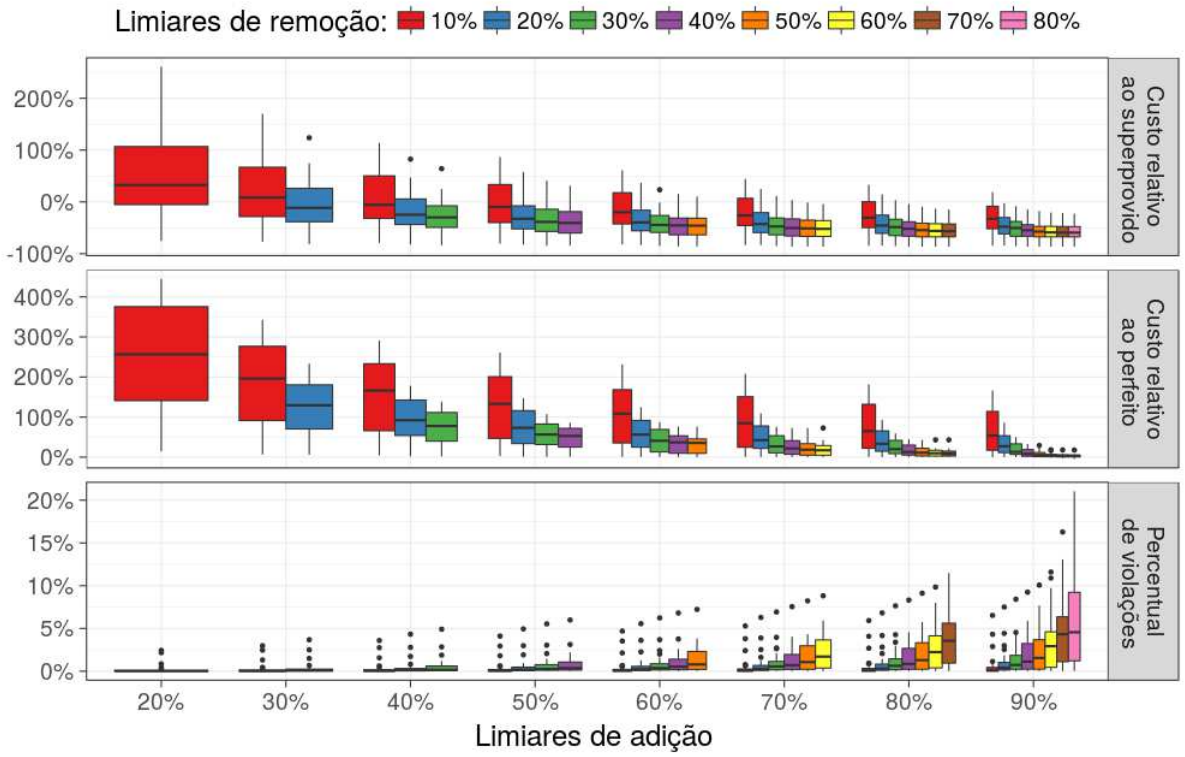


Figura 5.7: Desempenho da abordagem de provisionamento reativo baseada em utilização de CPU em termos do custo de provisionamento e do percentual de violações de SLO.

O *trade-off* fica evidente nos resultados para as duas métricas de provisionamento: para os menores limiares de adição e remoção tem-se os maiores custos e consequentemente os menores percentuais de violações de SLO. E o oposto também é visto, os maiores limiares de adição e remoção conduzem aos menores custos e a maiores percentuais de violações de SLO. Evidentemente, ao controlar-se esses limiares controla-se indiretamente a quantidade aceitável de violações de SLO e o custo aceitável a ser pago. Esse *trade-off* é mais acentuado para o provisionamento baseado em CPU, com potencialização extrema dos objetivos de provisionamento, a exemplo das configurações de limiares de adição em 20% e 90%. No primeiro caso, os custos chegam a ser 400% superiores aos apresentados pelo provisionamento perfeito com uma ocorrência mínima de violações de SLO, próximas de 0%. Enquanto que para o segundo caso, os custos aproximam-se aos praticados pelo provisionamento perfeito com ônus de percentuais significativos de violações de SLO, com valor mediano em torno de 5% de violações para o caso mais extremo.

Por outro lado, para o provisionamento baseado em memória, essa relação entre objetivos é menos evidente, devido à baixa sensibilidade do percentual de violações em função de mudanças nos limiares de provisionamento considerados. Mesmo para o cenário mais conservador em termos de custo (limares de adição em 90%), o percentual mediano de violações de SLO para o caso mais extremo é de aproximadamente 0.45%, enquanto que para o cenário menos conservador de custo (limiares de adição em 20%) as violações de SLO não ocorrem no provisionamento de 80% das aplicações. Para o provisionamento baseado em memória, apenas os custos mostram-se sensíveis à configuração dos limiares, com valores médios de elevação do custo em relação ao provisionamento perfeito extremando entre algo em torno de 425% a 18%, respectivamente da configuração de menor para a de maior limiar de adição. Acredita-se que isso seja decorrente do perfil de variabilidade da carga de trabalho de memória, que apresenta baixa variação entre percentuais de utilização do recurso.

Apesar da baixa sensibilidade do número de violações de SLO no provisionamento baseado em memória a mudanças nas configurações dos limiares de provisionamento, no geral essa relação de controle da configuração de limiares sobre o desempenho da abordagem de provisionamento é presente para as duas métricas de provisionamento observadas. Contudo, resta observar o nível de controle que pode ser obtido através dessas configurações e o quão aplicável é esse controle para manifestar os limites de desempenho desejados para o provisi-

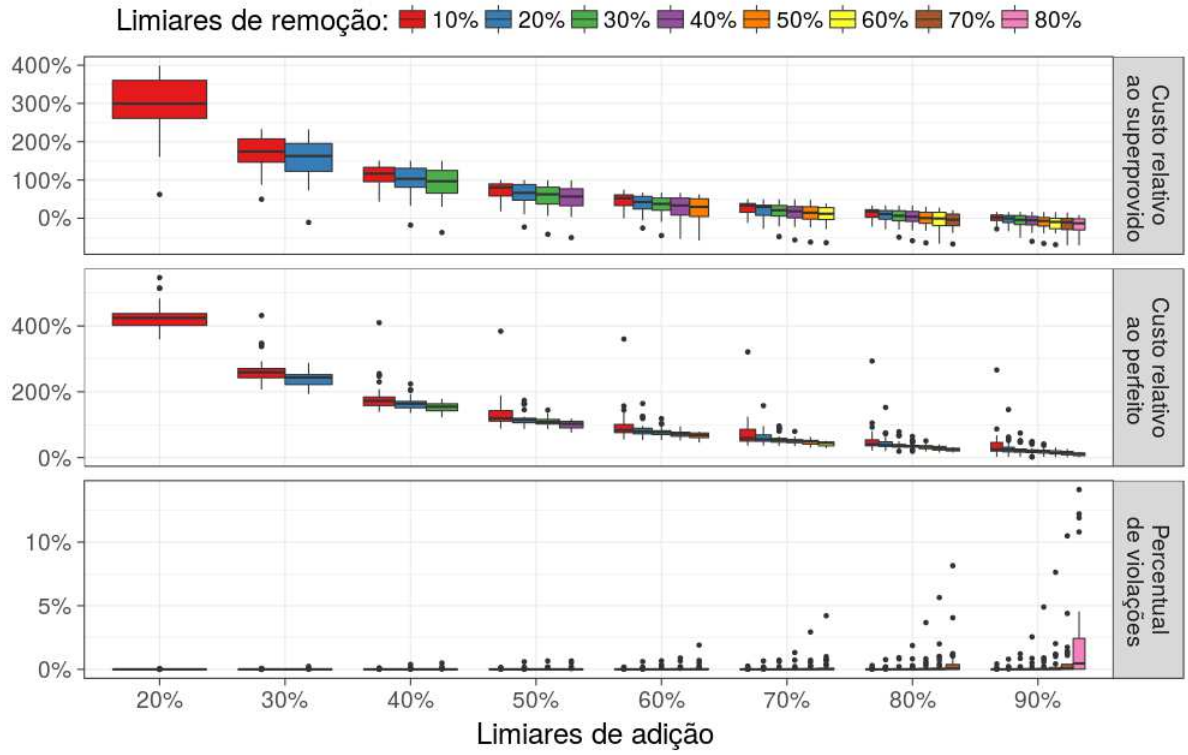


Figura 5.8: Desempenho da abordagem de provisionamento reativo baseada em utilização de memória em termos do custo de provisionamento e do percentual de violações de SLO.

onamento, em termos de custo de provisionamento e QoS da aplicação em execução.

5.3.2 Controle de objetivos de provisionamento

Apesar do *trade-off* de objetivos de provisionamento mostrar-se sensível às configurações de limiares de provisionamento, o controle desses objetivos conflitantes a partir dos limiares não é uma tarefa trivial. Idealmente, deseja-se manter a QoS desejada para a aplicação provisionada com o menor custo possível. Espera-se ainda que esse custo seja inferior ao proporcionado pelo super provisionamento estático perfeito da infraestrutura, que garante a ausência de violações SLO com o mínimo custo possível a partir de uma infraestrutura de capacidade estática. Ou seja, busca-se valores de limiares de adição e remoção que permitem a execução de diversas aplicações a partir de um limite de violações SLO com economias de custo em relação ao cenário de super provisionamento estático perfeito. Nesse sentido, os

resultados de provisionamento foram agrupados conforme o percentual de violação de SLOs observado e considerando a capacidade da configuração em manter os custos inferiores aos do cenário base de super provisionamento. Três classes foram definidas para os limites de violações de SLO: 0% (sem violações), $\leq 0,1\%$ e $\leq 1\%$, de forma que seja possível ter uma avaliação do desempenho da abordagem quanto a sua capacidade de respeitar esses limites e de reduzir custos de provisionamento para diferentes aplicações.

A partir da Figura 5.9 é possível identificar a quantidade de aplicações cujas violações de SLO enquadram-se nas diferentes classes de violações de SLO com custos inferiores aos praticados pelo super provisionamento perfeito. Esses resultados consideram as diferentes configurações analisadas para o provisionamento baseado apenas em CPU. Observa-se que nenhuma das configurações de provisionamento simuladas foi suficiente para atingir os objetivos de custo e QoS de todas as aplicações, mesmo para o cenário mais flexível com um limite máximo de 1% de violações de SLO, que na prática assume que a aplicação em execução possui disponibilidade próxima a 99%, considerada insatisfatória para alguns tipos de serviços. Nesse cenário o percentual médio de aplicações cujo limite de violações foi respeitado com economias de custo é de cerca de 54%. Além do mais, restringindo-se o limite de violações para 0,1% ocorre uma redução da capacidade da solução reativa em atingir os objetivos de custo e QoS da aplicação no provisionamento baseado em CPU, com uma média de sucesso para aproximadamente 20% das aplicações. Além do mais, para o cenário mais conservador em que violações de SLO não são aceitas, a meta é atingida apenas para uma aplicação dentre as 30 consideradas.

Considerando o provisionamento baseado apenas em memória, que apresenta baixa sensibilidade ao número de violações de SLO em função das configurações de controle, o principal fator que limita a obtenção dos objetivos definidos é a economia de custo em relação ao cenário base. Nesse caso, os resultados de provisionamento apresentados na Figura 5.10 demonstram que apenas os limiares de adição mais elevados (superiores a 70%) conseguem atingir os objetivos para um número significativo de aplicações, com um percentual médio aproximado de 46% das aplicações com objetivos atingidos. Para as demais configurações de limiares de adição, o percentual de aplicações em que os objetivos de custo e QoS foram atingidos mostra-se não representativo, com média em torno de 11%.

Para casos mais restritivos, com o limite de violações de SLO definido em 0,1% a capa-

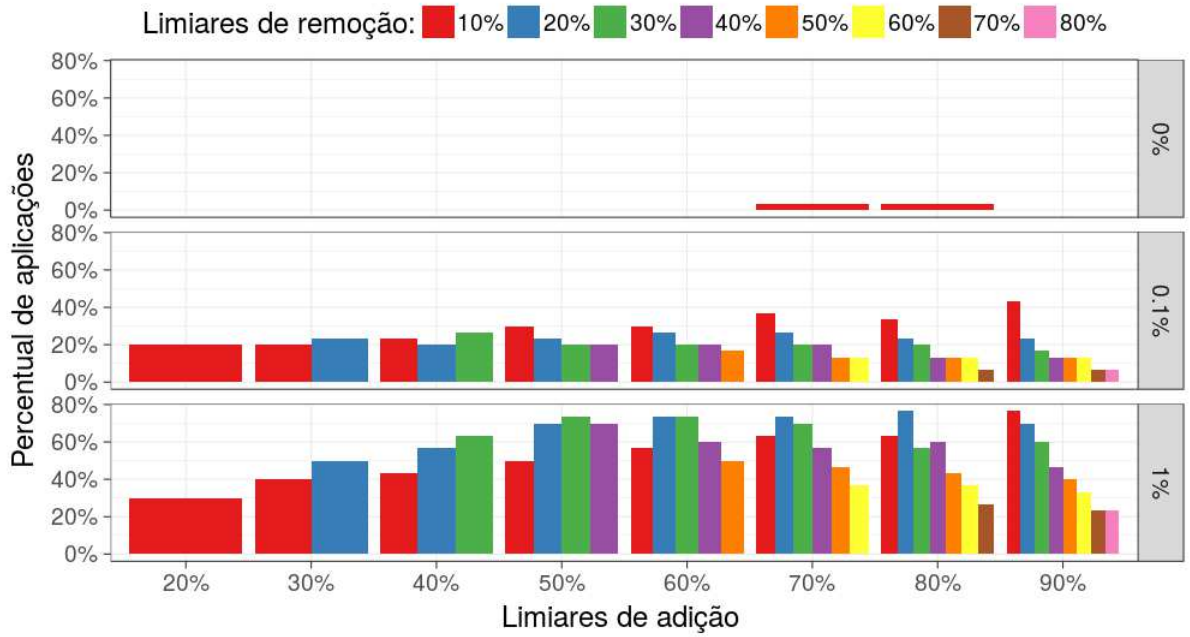


Figura 5.9: Análise do percentual de aplicações em que foi possível atingir os objetivos de custo e QoS no provisionamento baseado em CPU.

cidade média de atingir os objetivos de provisionamento é reduzida para aproximadamente 21% das aplicações. Todavia, quando o objetivo é eliminar violações de SLO no provisionamento baseado em memória o percentual médio de aplicações em que esse objetivo é atendido é de 5%, e de 10% no melhor caso.

Como esperado, para um cenário de objetivos de provisionamento conflitantes, o cumprimento dos objetivos de QoS da aplicação estão associados a custos de provisionamento, de forma que quanto mais restritivo é o limite aceitável de violações de SLO maior será a o custo de provisionamento necessário para atingi-lo. Isso pode ser observado na Figura 5.11, que apresenta a avaliação do custo incorrido para cada uma das classes de violações de SLO. O diagrama de caixa mostra, para cada aplicação cujos objetivos de QoS foram satisfeitos sem restrições de custo, o menor custo de provisionamento relativo ao provisionamento perfeito e ao super provisionamento perfeito. Como consequência, observa-se que grande parte da incapacidade de obtenção dos objetivos de provisionamento é devido a elevação nos custos de provisionamento pela restrição dos objetivos de SLO, que geram custos superiores aos apresentados pelo super provisionamento estático perfeito (percentuais positivos na primeira

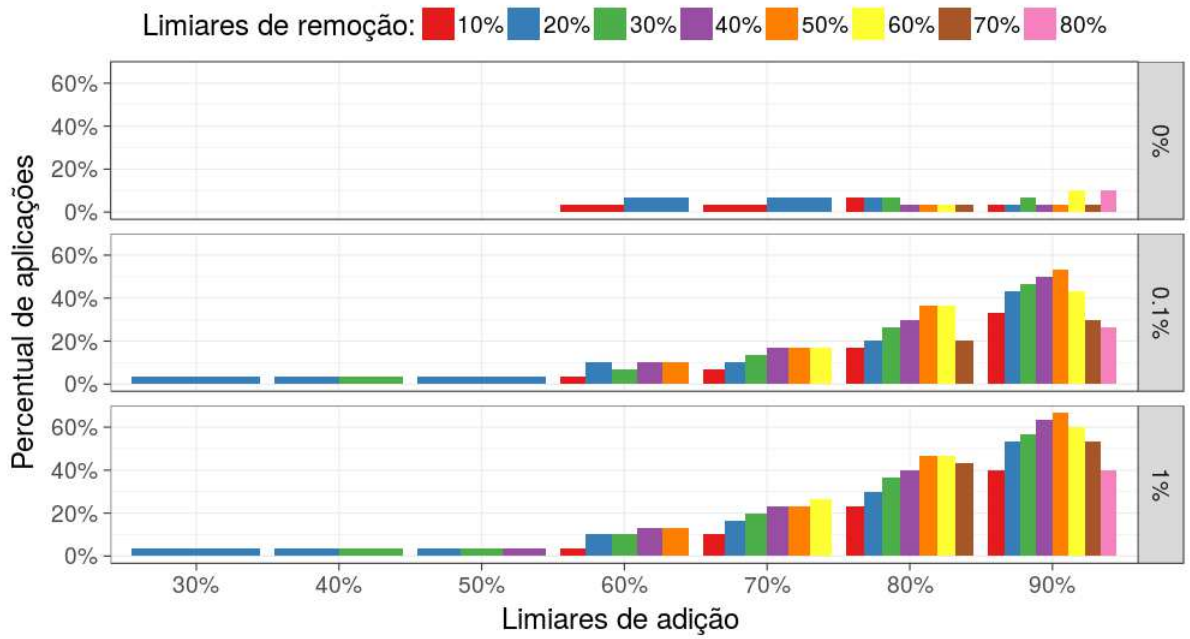


Figura 5.10: Análise do percentual de aplicações em que foi possível atingir os objetivos de custo e QoS no provisionamento baseado em memória.

linha do gráfico). Ou seja, a redução da quantidade de aplicações cujos objetivos de custo e QoS foram obtidos deve-se tanto a restrição dos objetivos de SLO quanto ao ônus de custo associado à obtenção de objetivos de SLO restritivos, para ambas as métricas.

Para a classe limitada a 0% de violações de SLO, metade das aplicações que atingiram esse objetivo apresentaram custo 180% e 27%, respectivamente para o provisionamento baseado em CPU e memória, maior que o custo do provisionamento perfeito correspondente. Por outro lado, para um limite máximo de $0,1\%$ de violações de SLO a mediana do custo é aproximadamente 71% e 14% maior que o obtido no provisionamento perfeito, respectivamente para o provisionamento com base em CPU e memória. Apenas no cenário mais flexível, em que o limite de violações de SLO é de 1%, os custos de provisionamento da abordagem reativa para ambas as métricas apresentam reduções significativas em relação ao cenário de super provisionamento e aproximam-se dos custos obtidos pela abordagem de provisionamento perfeita, com uma elevação de no máximo 6% e 12% de custo para metade das aplicações cujo objetivo foi atingido, considerando CPU e memória respectivamente.

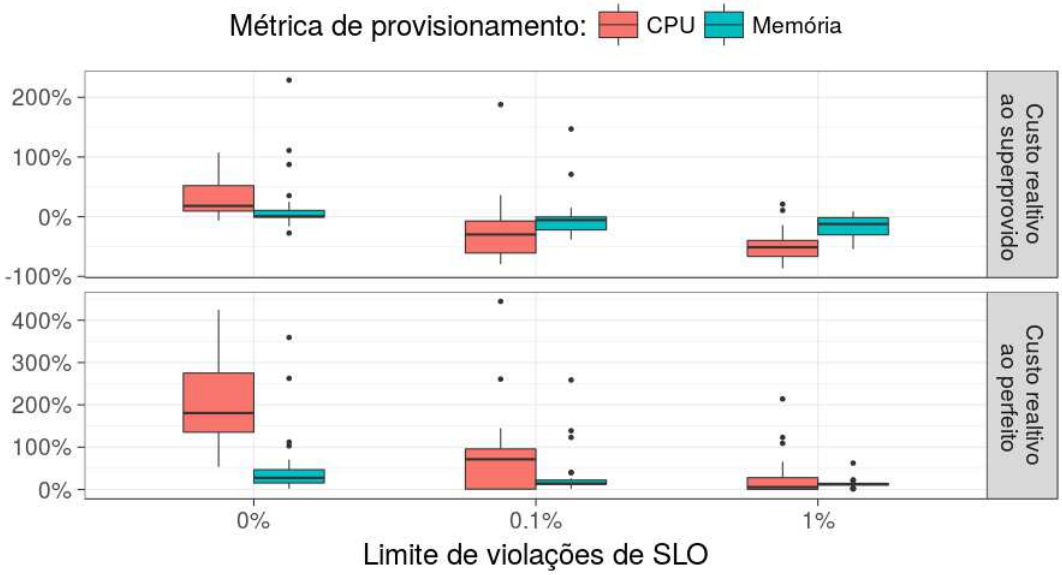


Figura 5.11: Análise de custos relativos ao provisionamento perfeito e ao super provisionamento perfeito para diferentes cenários de limites de violações de SLO.

5.3.3 Eficiência de configuração por objetivo de provisionamento

Além do desempenho da abordagem reativa quanto a sua capacidade de atingir os objetivos de provisionamento, um outro fator de desempenho consiste na eficiência de configuração dos limiares da abordagem para atingir tais objetivos para as aplicações consideradas. Como já mencionado, o provisionamento automático de aplicações em ambientes IaaS tende a priorizar a manutenção da QoS da aplicação provisionada, garantindo a otimização do uso da infraestrutura e a consequente minimização dos custos de execução da aplicação. Desta forma, a análise de eficiência de configuração trata da seleção de configurações que permitam a obtenção dos objetivos de QoS com o menor custo possível. Para uma mesma aplicação, pode existir mais de uma configuração de provisionamento que seja capaz de assegurar os requisitos mínimos de QoS, todavia dentre estas existe uma parcela mais eficiente que consegue atingir o objetivo de QoS com o menor custo entre as demais. Assim, a eficiência de configuração consiste na análise de predominância de um conjunto mínimo de configurações de regras de provisionamento eficientes, capaz de atingir os objetivos de QoS das aplicações com os menores custos possíveis.

Devido ao percentual não significativo de aplicações cujo objetivo de eliminação de violações de SLO foi atingido, essa análise de eficiência de configuração mostra-se não aplicável para a classe com limite de 0% de violações de SLO. Já para a classe menos restritiva (até 1% de violações), dentre as configurações que atingiram os objetivos de SLO com o menor custo de provisionamento, a configuração com maior predominância de uso para esse fim foi responsável pelo provisionamento de em média 34% das aplicações e no máximo 46%, considerando junto os cenários com base em CPU e memória. Enquanto que para a classe com até 0,1% de violações de SLO a configuração mais predominante foi suficiente para garantir os objetivos de QoS e minimização de custo para apenas 25% dessas aplicações em média e no máximo 34%, para ambas as métricas. Desta forma, observa-se que quando da busca pelo provisionamento respeitando os limites de SLO com o menor custo de execução não há um conjunto representativo de configurações de provisionamento que sejam capazes de provisionar todas as aplicações, o que elemina a possibilidade de uma configuração padrão que seja capaz de atingir esses objetivos para as diferentes aplicações consideradas.

5.3.4 Sumário de resultados da abordagem prática

O desempenho da abordagem reativa prática de provisionamento foi verificado através de uma varredura dos parâmetros de configuração de regras de provisionamento e da análise final das métricas do número de violações de SLO e custo de provisionamento, que apresentam-se como objetivos conflitantes para as soluções de provisionamento automático para ambas as métricas consideradas. A partir dos resultados, verificou-se que apesar da abordagem reativa apresentar um grau de controle satisfatório entre objetivos de provisionamento, a depender principalmente dos limites de SLO definidos, a obtenção desses objetivos torna-se infactível para uma parcela considerável das aplicações analisadas, mesmo considerando as duas métricas de provisionamento e as configurações de limiares que apresentam melhor desempenho.

Em um primeiro momento observou-se a incapacidade das configurações de provisionamento reativo em atingir os objetivos de provisionamento para todas as aplicações, mesmo para limites de SLO menos restritivos onde é aceitável que em até 1% dos intervalos de tempo de execução a utilização de infraestrutura esteja em 100%. Essa incapacidade é potencializada com a restrição dos limites de SLO, com valores aproximando-se de 0%, onde

a taxa de aplicações cujos objetivos são atingidos é mínima, principalmente para o provisionamento baseado em CPU, com apenas uma aplicação com objetivos satisfeitos para o caso mais restritivo. Esse resultado é proveniente da relação antagônica entre QoS e custo de provisionamento, dado que a busca pela satisfação dos objetivos de QoS provocam elevações significativas nos custos de provisionamento, que por sua vez chegam a superar os custos de uma abordagem de super provisionamento estático perfeito. Essa relação foi demonstrada a partir dos custos obtidos para cada uma das classes de limites de SLO definidas para o provisionamento automático das aplicações.

Complementarmente, foi analisada a eficiência de configuração das regras de provisionamento para a satisfação dos objetivos de provisionamento definidos. Para classes de limites de SLO menos conservadores e restrições de custo apenas relacionadas ao custo do cenário super provido, existem configurações de provisionamento que mostram-se minimamente predominantes para atingir tais objetivos para as aplicações, com representatividade para mais de 60% das aplicações. Todavia, ao buscar-se os objetivos de QoS juntamente com a minimização dos custos de provisionamento essa representatividade é reduzida para menos da metade das aplicações, considerando ambas as métricas de provisionamento. Ou seja, não existe um conjunto padrão de configurações que seja suficientemente eficiente para atingir os objetivos de custo e QoS para a maior parcela das aplicações.

Desta forma, verifica-se que a solução reativa de provisionamento apresenta desempenho não satisfatório para o provisionamento de diferentes aplicações quanto à satisfação dos objetivos de provisionamento de custo e QoS da aplicação. Além do mais, considerando a eficiência de configuração das regras de provisionamento considera-se que não há predominância de configuração, com configurações que sejam capazes de satisfazer os diferentes objetivos abordados no estudo para as diferentes aplicações, mesmo considerando separadamente as duas métricas de provisionamento. Contudo, na próxima seção será realizada uma análise de desempenho da abordagem de provisionamento reativo com base em métricas multidimensionais, onde as regras de provisionamento consideram simultaneamente a utilização de CPU e memória para efetuar o provisionamento da aplicação ao longo do tempo.

5.4 Provisionamento Reativo baseado em Múltiplas Dimensões de Recursos

Durante o provisionamento de aplicações a alocação de uma quantidade insuficiente de recursos pode gerar gargalos em diferentes dimensões de recursos computacionais. Tais gargalos ocasionam reduções no desempenho da aplicação porque não há recursos de CPU ou memória suficientes para executar a aplicação com os níveis de QoS esperados. Assim, apesar das análises anteriores abordarem o provisionamento reativo sob a ótica de diferentes métricas de utilização de recursos individualmente, na prática o provisionamento automático de uma aplicação requer que alocações e deslocações ocorram em observância à utilização de recursos simultaneamente em diferentes dimensões. Com base nisso, nessa seção aborda-se uma análise do provisionamento reativo de aplicações horizontalmente escaláveis com base em múltiplos recursos, especificamente utilização de CPU e memória.

Nesse sentido, foram realizadas simulações de provisionamento reativo para cada uma das 30 aplicações anteriormente descritas a partir de configurações de regras de provisionamento para as duas métricas de utilização de recursos. Para isso, o modelo de simulação foi adaptado para considerar o provisionamento com base em duas métricas: (i) uma ação de alocação de recursos é realizada quando pelo menos um dos limiares de adição de uma das métricas é atingido; (ii) a remoção de recursos ocorre apenas se ambos os limiares de remoção das métricas forem atingidos. Foi realizada uma varredura de parâmetros de configuração, com limiares de adição para ambas as métricas variando de 20% a 90%, em passos de 10%, e com limiares de remoção variando de 10% a 80%, em passos de 10%, também para ambas as métricas[6]. Desta forma, cada configuração de regra é composta por dois limiares de adição, um para CPU e outro para memória, e dois limiares de remoção, que consiste em um experimento fatorial completo com remoção de configurações inválidas. Para cada limiar de provisionamento foi associada a ação de alocar ou desalocar uma VM com 1 núcleo de CPU e 1GB de memória à infraestrutura de execução[7], semelhante ao realizado na análise

[6]Limiares de adição de uma métrica são sempre configurados com valores maiores que os limiares de remoção da mesma métrica.

[7]Com base no modelo de tarifação da Amazon AWS, as VMs são de fato removidas da infraestrutura se o disparo da ação de remoção ocorrer em sincronia com horas completas de uso completo das VMs a serem desalocadas.

prática da seção anterior.

Desta forma, para cada aplicação considerada foram executadas 1296 testes de simulações de provisionamento com base em diferentes configurações de limiares de adição e remoção de recursos para ambas as métricas em conjunto. Os resultados de provisionamento das configurações foram agrupados por objetivos básicos de QoS e custo. Objetivos de QoS correspondem à capacidade de limitar o percentual de violações de SLO obtidas ao longo do provisionamento, por aplicação e para ambas as métricas [8]. Objetivos de custo visam reduzir os custos de provisionamento em relação a um cenário de super provisionamento estático perfeito, que também serviu de base comparativa para as análises da seção anterior. Foram consideradas três classes de limites de violações de SLO: 0% (sem violações), $\leq 0,1\%$ e $\leq 1\%$, a fim de contemplar diferentes graus de conservadorismo quanto ao objetivo de QoS da aplicação.

Para cada configuração de provisionamento simulada foi calculado o percentual de aplicações para as quais foi possível atingir os objetivos básicos de custo e QoS a partir da configuração realizada, considerando as diferentes classes de limites de SLO. Observou-se que em nenhum cenário de configuração avaliado foi possível atingir tais objetivos de provisionamento para todas as aplicações provisionadas. Ou seja, também para o provisionamento baseado em múltiplos recursos não existe uma configuração dentre as simuladas que seja capaz de atingir os objetivos de custo e QoS para 100% das aplicações. No cenário em que não é aceito violações de SLO (0% de violações), cada uma das configurações mais eficientes conseguiram provisionar sem a ocorrência de violações apenas 10% das aplicações. O percentual máximo de aplicações cujos objetivos foram atingidos eleva-se com o relaxamento do limite de SLO, assumindo um valor de 50% quando o limite violações é de 0,1%. Esse percentual máximo é de aproximadamente 67% para o cenário menos restritivo, em que aceita-se um percentual de até 1% de violações de SLO para ambas as métricas de provisionamento. Ou seja, mesmo com uma ampla varredura de parâmetros de configuração de regras de provisionamento, baseada em recursos multidimensionais, não foi possível atingir os objetivos de custo e QoS para todas as aplicações.

Além do mais, idealmente o provisionamento busca executar a aplicação com um nível

[8] O percentual de violações de SLO é computado com base no número de intervalos de tempo em que a utilização de pelo menos um tipo de recurso atingiu 100% ao longo da execução da aplicação.

de QoS adequado, correspondente ao limite de violações de SLO, e com o mínimo de custo possível de provisionamento e não apenas com um custo menor do que o apresentado pelo cenário perfeitamente super provido. Nessa caso, busca-se um conjunto de configurações dentre todas simuladas aquelas que são capazes de respeitar os limites de violações de SLO com o menor custo possível de provisionamento para cada uma das aplicações. No entanto, selecionar essas configurações eficientes mostra-se uma tarefa não trivial. A Figura 5.12 apresenta o diagrama de caixa do percentual de configurações eficientes dentre o total de configurações capazes de respeitar os limites de violações e de custo base de provisionamento. Para as classes menos restritivas, com limite de 0,1% e 1% de violações, para 50% das aplicações apenas 2,2% e 1,8% das configurações mostraram-se eficientes em termos de limitar violações ao menor custo possível dentre as configurações consideradas, respectivamente. Apesar do percentual de configurações eficientes mostrar-se mais significativo para o cenário mais restritivo (0% de violações), com média em torno de 35%, na prática esse fato está associado a ineficiência das configurações em eliminar violações de SLO, caracterizada pelo número menor de configurações que são capazes de atingir esses objetivos e pela quantidade não significativa de aplicações cujos objetivos são atendidos.

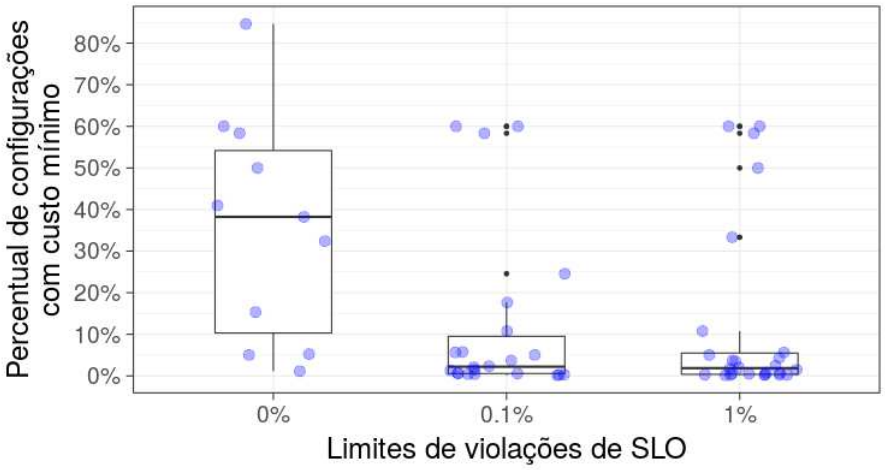


Figura 5.12: Análise do percentual de configurações dentre as configurações que satisfazem os objetivos básicos de provisionamento que o fazem com o mínimo custo de execução.

Associado a essa complexidade de seleção das configurações mais eficiente existe o fator da variabilidade de custo entre as configurações que atendem a ambos os objetivos de provi-

sionamento. Mesmo para configurações que remetem a um custo mínimo dentre as demais configurações existe um elevação de custo em relação ao custo do provisionamento perfeito, que consiste no menor custo de provisionamento sem violações de SLO. Para esses cenários de provisionamento com custo mínimo a elevação média de custo em relação ao provisionamento perfeito das aplicações é de aproximadamente 17%, considerando as diferentes classes de limites de violações de SLO. Por outro lado, considerando todas as configurações em que foi possível atender os objetivos básicos de provisionamento, de QoS e custo, observa-se em média 39% de elevação nos custos de provisionamento em comparação ao cenário de provisionamento perfeito. Esse aumento nos percentuais médios de custos deve-se a um pequeno número de configurações muito conservadoras que causam elevações significativas de custo em relação ao provisionamento perfeito. A Figura 5.13 apresenta o diagrama de caixa dos custos relativos ao cenário perfeito agrupado pelas classes de limites de SLO estabelecidas, onde é possível observar os cenários com custos de provisionamento fora do padrão.

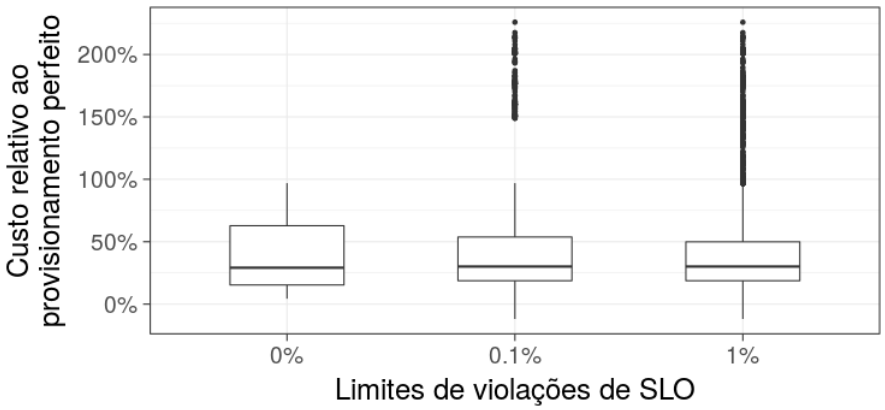


Figura 5.13: Análise de custos relativos ao provisionamento perfeito para diferentes cenários de limites de violações de SLO.

Desta forma, a partir das análises realizadas observa-se que além de não haver configuração que seja eficiente para provisionar todas as aplicações respeitando-se os limites básicos de custo e QoS, a seleção de configurações que minimizam o custo de provisionamento não pode ser considerada uma tarefa trivial. Isso pode ser observado tanto pelo percentual não significativo de configurações que podem atingir tais objetivos de custo mínimo quanto pelo custo incorrido pela seleção inadequada das configurações mais eficientes com relação à minimização do custo. Além disso, o fator de capacidade de controle da solução de

provisionamento é impactado pelo acréscimo de parâmetros de configuração das regras de provisionamento, em decorrência do uso de múltiplas métricas, que torna a eficiência de configuração não satisfatória no contexto de provisionamento automático e reativo como um serviço de IaaS.

5.5 Discussão e Conclusões

Nesse capítulo foi realizada uma análise sobre o uso de abordagem de provisionamento automático e reativo para compor o serviço de provisionamento automático oferecido no contexto de IaaS. Essa análise foi realizada segundo diferentes aspectos relacionados com a construção e implantação da técnica reativa, além de fatores de desempenho que tratam dos principais objetivos do provisionamento automático de recursos. Esse estudo conduz à conclusão de que a abordagem reativa não é adequada para compor um serviço de provisionamento automático não intrusivo em um ambiente de IaaS. As razões que levam a essa conclusão vão desde a não generalidade e eficiência de configuração da técnica até o não controle dos objetivos de provisionamento e a incapacidade de atender esses objetivos para diferentes aplicações.

Primeiramente, a configuração das regras de provisionamento é significativamente sensível às características das cargas de trabalho das aplicações, estando diretamente relacionadas com os perfis de consumo de CPU e memória, o que contesta a generalidade e independência da solução em relação à aplicação provisionada (Questão de pesquisa 3). Um outro ponto de destaque é que as análises realizadas não revelaram a existência de uma configuração de limiar predominante para uma aplicação e métrica de provisionamento, muito menos para várias aplicações consideradas no estudo, o que inviabiliza o uso de uma configuração padrão para o provisionamento de diferentes aplicações em um ambiente de produção do serviço de provisionamento (Questão de pesquisa 4). O senso comum considera que adicionar nós quando se chega a uma utilização de aproximadamente 70% e remover nós quando se chega próximo de 40% de utilização é adequado, quando os resultados obtidos demonstram que isto não é a regra tanto para o provisionamento perfeito quanto para o prático.

Além do mais, a abordagem reativa não é, na prática, bem sucedida em cumprir os objetivos de QoS, em termos de limites de violações de SLO, das aplicações e, quando cumpre,

leva a custos muito elevados, muitas vezes superiores aos praticados pelo super provisionamento estático da infraestrutura (Questões de pesquisa 1 e 2). Isso é observado mesmo quando consideradas diferentes métricas de provisionamento e um provisionamento baseado simultaneamente em múltiplos recursos. Ou seja, além do fato da configuração do sistema de provisionamento não ser uma tarefa trivial, a abordagem reativa também não foi bem sucedida em lidar com os objetivos conflitantes de custo de provisionamento e QoS da aplicação provisionada. Adicionalmente, a solução reativa demonstra que o controle desses objetivos está diretamente relacionado com as características da carga de trabalho da aplicação, o que potencializa a ineficiência da solução quanto à busca pelos objetivos de provisionamento.

Apesar de apresentar desempenho não adequado para o cenário de provisionamento automático como um serviço buscado neste trabalho, a técnica de provisionamento reativo é considerada simples em termos de implementação em um ambiente de IaaS (Questão de pesquisa 5). As atuais soluções e serviços de IaaS (por exemplo Amazon AWS e OpenStack) oferecem ferramentas tanto de monitoramento da infraestrutura, responsáveis por coletar métricas de utilização de recursos, quanto de atuação sobre a infraestrutura, capazes de alocar ou desalocar VMs da infraestrutura usada para executar a aplicação provisionada. Desta forma, a construção da solução reativa consiste essencialmente na implementação de regras programáveis que são disparadas com base em métricas monitoradas e atuam sobre a infraestrutura a partir de ferramentas já disponibilizadas pelos provedores de IaaS.

Em suma, mesmo com a simplicidade de construção da solução reativa e a possibilidade de melhorias no seu desempenho, esta mostra-se não adequada para basear um serviço de provisionamento automático não intrusivo que seja capaz de lidar com os diferentes objetivos de provisionamento e perfis de aplicações em um ambiente real de IaaS, mesmo quando considera-se um cenário de provisionamento multidimensional. Todavia, para que a aplicação dessa técnica fosse de fato viável como um serviço de provisionamento não intrusivo em IaaS seria necessário que a solução desenvolvida fosse capaz de lidar com as variantes existentes nesses cenário de provisionamento, como exemplo das particularidades de carga de trabalho, para cada dimensão de recursos, de cada aplicação provisionada pelo serviço de provisionamento automático e reativo.

Capítulo 6

Provisionamento como um Serviço Automático e Proativo

Esse capítulo aborda diferentes aspectos que envolvem o processo de provisionamento proativo aplicado a um serviço de provisionamento automático em ambientes de IaaS, que opera com base em métricas não intrusivas à aplicação provisionada. Especificamente, é realizado um estudo sobre o desempenho de técnicas de provisionamento proativo em termos do custo de provisionamento da aplicação e manutenção de sua QoS em níveis aceitáveis. Além disso, no que concerne o cenário de provisionamento como um serviço, as técnicas proativas também são avaliadas quanto a sua eficiência de configuração e a controlabilidade dos objetivos de provisionamento, definidos em termos de custo e QoS. Desta forma, o estudo aborda questões sobre desempenho e limitações dessas soluções ao compor um serviço de provisionamento automático como descrito neste trabalho, discorrendo sobre possíveis aprimoramentos e evoluções da técnica proativa baseada em métricas não intrusivas.

6.1 Introdução

O provisionamento automático proativo antecipa variações na carga de trabalho da aplicação provisionada para realizar a gerência automática de recursos virtuais alocados a fim de suprir adequadamente tais demandas no futuro. Desta forma, a base do provisionamento proativo como uma serviço em IaaS consiste na capacidade de produzir estimativas eficientes sobre a futura carga de trabalho da aplicação de interesse. O planejamento de capacidade da in-

fraestrutura de execução, em termos da quantidade e dos tipos de instância de VM a serem provisionadas, é decidido a partir de tais estimativas e considerando os objetivos almejados no processo de provisionamento.

Durante o processo de provisionamento, estimativas acima da utilização real observada (super estimativas) implicam no super dimensionamento da infraestrutura e, por consequência, na alocação desnecessária de recursos, que causa desperdício e elevação dos custos de provisionamento. Por outro lado, estimativas abaixo da demanda real da aplicação (sub estimativas) geram um sub dimensionamento da infraestrutura, com menos recursos do que o requerido pela aplicação, que pode conduzir a aplicação a um estado de degradação de sua QoS. Esta relação entre super dimensionamento e sub dimensionamento está diretamente associada aos objetivos conflitantes de provisionamento, que em geral buscam uma harmonia entre a manutenção da QoS da aplicação provisionada em níveis aceitáveis e a redução de custos de provisionamento, quando da presença de variações na carga de trabalho da aplicação.

É imprescindível, portanto, que a solução de predição usada pelo serviço de provisionamento automático seja capaz de gerar estimativas confiáveis sobre as demandas da aplicação em execução. A técnica de provisionamento proativo é amplamente explorada na literatura através de diferentes abordagens [2, 14, 15, 44, 52, 56, 62]. No entanto, apesar da existência de soluções que baseiam-se apenas em métricas da infraestrutura de execução [14, 44], a maioria das soluções propostas fazem uso de métricas específicas da aplicação para realizar o provisionamento (como tempo de resposta, tamanho de fila, taxa de chegada, tipos de requisições, etc.), que não são acessíveis em um cenário de provisionamento automático como um serviço.

Como já discutido anteriormente, o provisionamento automático e proativo como um serviço deve usar apenas métricas de utilização não intrusivas, disponibilizadas tipicamente para qualquer aplicação em execução em um ambiente de IaaS e, por conseguinte, disponíveis ao provedor do serviço de provisionamento automático (como utilização de CPU, memória, disco, etc.). Com base nisso, nesse capítulo é realizada uma análise do emprego de soluções *preditivas* e *não intrusivas*, baseadas em métricas de utilização de CPU e memória para o provisionamento automático e proativo de aplicações horizontalmente escaláveis em ambientes de IaaS.

O principal objetivo desse estudo consiste em avaliar o desempenho de soluções de provisionamento automático e proativo considerando os seguintes aspectos: (i) capacidade de assegurar a QoS da aplicação provisionada ao mesmo tempo que reduz custos de provisionamento, especialmente em comparação com a abordagem de super provisionamento estático e perfeito da infraestrutura, onde não há violações de SLO; (ii) eficiência de configuração dos modelos de predição e das soluções de provisionamento; (iii) grau de generalidade e independência de características da carga de trabalho da aplicação provisionada; e (iv) capacidade de controle de objetivos de provisionamento apresentada pela solução, com o intuito de permitir uma configuração da solução proativa que explore o *trade-off* entre objetivos, em termos do custo de provisionamento e da QoS da aplicação.

A seguir, serão apresentadas diferentes análises de soluções proativas. Uma abordagem de provisionamento perfeito é inicialmente discutida. Essa análise é considerada não realista por fazer uso de estimativas de demanda sem a presença de erros de predição. Apesar de não realista, é útil para termos de comparação com as técnicas não perfeitas. A discussão continua com a análise de provisionamento em abordagens de provisionamento prático, que são suscetíveis à ocorrência de erros de planejamento de capacidade em função de estimativas de demanda não acuradas. Finalmente, também é analisado o provisionamento baseado em múltiplas dimensões de recursos, que consideram diferentes métricas de utilização de recursos no provisionamento.

6.2 Análise do Provisionamento Proativo Perfeito

O provisionamento proativo perfeito baseia-se na existência de um preditor perfeito, livre de erros de predição. Esse preditor deve ser capaz de gerar estimativas precisas sobre as futuras demandas da aplicação provisionada. Assim, decisões de um planejador de capacidade perfeito, baseadas nessas estimativas, podem promover o provisionamento da aplicação sem violações de SLO e com o menor custo de execução possível. No entanto, até onde se sabe não existe preditor perfeito nesse formato. Isso deve-se principalmente à complexidade de construção de um modelo preditivo perfeito, que envolve operações complexas e não intuitivas sobre o histórico de dados da demanda da aplicação com o intuito de gerar estimativas perfeitas.

Assim, mesmo considerando-se que a relação entre demandas de diferentes dimensões (por exemplo, consumo de CPU e memória) possa ser negligenciada e que essas demandas possam ser individualmente consideradas no provisionamento perfeito, uma técnica preditiva de provisionamento ainda deve prever de forma perfeita a carga de trabalho da aplicação. Para tal, faz-se necessário decompor perfeitamente as séries temporais da carga de trabalho da aplicação em pelo menos três componentes: (i) a tendência da série temporal – isto é, se a série temporal está em tendência de crescimento ou decrescimento e qual é o nível dessa tendência; (ii) o padrão sazonal das demandas da carga de trabalho; e (iii) o ruído ou erro aleatório associado. Além disso, a técnica de provisionamento deve ser capaz de estimar com base nesses componentes o comportamento da carga de trabalho futura no curto e médio prazo (na casa de minutos), de forma a ser eficiente em realizar o planejamento de capacidade da aplicação provisionada.

Essa complexidade é potencializada ao se considerar fatores externos não previsíveis ao processo de provisionamento como um serviço. Diversos são os fatores que podem influenciar nesse processo, como rajadas de demanda de usuários sem padrões de recorrência, instabilidade de desempenho na rede que interliga os componentes da aplicação e seus usuários, serviços em nível gerencial que executam em paralelo à aplicação provisionada (por exemplo o *garbage collector*), heterogeneidade de desempenho entre nós que hospedam as VMs que proveem uma mesma aplicação, etc. Adicionalmente, considerando-se múltiplas dimensões de recursos, diferentes aplicações podem reagir de forma distinta a esses fatores externos em termos da proporcionalidade entre o consumo de recursos de diferentes dimensões. Desta forma, variações na demanda entre dimensões de recursos também podem contribuir para a existência de erros nas estimativas de demanda e por conseguinte no processo de provisionamento preditivo e proativo.

Além do mais, mesmo em um cenário de provisionamento com um preditor sub-ótimo, que apresenta um percentual insignificante de erros e com erros de menor intensidade e amplitude, é possível que em alguns casos esses erros ainda reflitam negativamente no desempenho da aplicação provisionada, de forma a serem percebidos pelos usuários finais da aplicação. Desta forma, argumenta-se sobre a não factibilidade de um preditor livre de erros, ou mesmo de um preditor próximo do perfeito. Por consequência não deve ser possível, na prática, alcançar um serviço de provisionamento proativo sem erros de planejamento de

capacidade e dimensionamento da infraestrutura de execução, de forma a proporcionar o provisionamento perfeito da aplicação em execução. Contudo, na próxima seção é realizada uma análise de provisionamento prático da abordagem proativa.

6.3 Análise Prática do Provisionamento Proativo

A partir do modelo de simulação definido no Capítulo 4 foi realizada uma análise do provisionamento proativo prático a partir de modelos de predição não intrusivos, que consideram apenas métricas de utilização da infraestrutura de execução. Especificamente, foram considerados 3 modelos de predição de séries temporais difundidos na literatura e descritos no Apêndice B: (i) um algoritmo simplista baseado em medições anteriores, que considera no planejamento de capacidade o valor de utilização de recurso observado no intervalo de tempo anterior (LW); (ii) um modelo de predição baseado em auto-regressão do histórico de utilização de recursos (AR); e (iii) uma solução de predição baseada na seleção dinâmica do modelo de predição a ser considerado no provisionamento (Dinâmico), proposta por Morais et al. [44]. Apesar de também considerar métricas não intrusivas no provisionamento, o algoritmo de predição com base em casamento de padrões proposto por Caron et al. [14] não foi aproveitado nesse estudo devido a questões de desempenho, como discutido no Apêndice B.

O provisionamento proativo foi simulado para as diferentes aplicações do conjunto de dados da HP com base individualmente em métricas de demanda de CPU e memória, com o provisionamento baseado apenas em CPU ou apenas em memória. Desta forma, o modelo de simulação foi configurado para simular o controle da aplicação com periodicidade de 5 minutos, em conformidade com a duração de um intervalo de tempo dos rastros de utilização. O horizonte de predição também foi configurado para ter duração de 5 minutos, ou 1 intervalo de tempo, e desta forma a solução de provisionamento opera com uma margem de tempo próxima de 5 minutos para a execução e efetivação do provisionamento, em concordância com a análise sobre tempo de provisionamento realizada no Apêndice A. Assim, ao ser usado no provisionamento de uma aplicação, um modelo de predição é alimentado, a cada laço de controle, com as duas últimas semanas de dados históricos de utilização da aplicação e realiza, com base nesses dados, uma estimativa de utilização de CPU para o horizonte de predição pré-definido.

Para esse estudo, considerou-se que o planejamento de capacidade é realizado em termos da quantidade de VMs necessária para suprir as demandas estimadas da aplicação no curto prazo. Cada VM dispõe de 1 núcleo de CPU e 1GB de memória. Desta forma, o desempenho das técnicas de provisionamento proativo pode ser avaliado segundo métricas de custo de provisionamento e QoS da aplicação provisionada impactadas por erros de estimativa do modelo de predição considerado pela técnica. Especificamente, o custo de provisionamento é computado pela quantidade de horas de máquina alocadas para executar a aplicação ao longo do tempo, enquanto que a QoS da aplicação é mensurada a partir do percentual de intervalos de tempo em que a utilização média da infraestrutura, para o recurso considerado, atingiu o limite de 100% definido como o máximo aceitável no SLO definido entre o provedor da aplicação e o provedor do serviço de provisionamento automático.

Em resumo, nessa seção é realizada uma análise de desempenho da abordagem proativa em termos do percentual de violações de SLO e do custo de provisionamento em comparação aos custos obtidos em cenários de provisionamento automático perfeito e de super provisionamento estático perfeito. Nesta análise são consideradas diferentes métricas de utilização de recursos. Adicionalmente, a partir do desempenho apresentado por cada técnica de provisionamento considerada, é possível realizar uma avaliação da abordagem proativa quanto ao grau de controlabilidade dos objetivos de provisionamento, que corresponde à priorização de objetivos da relação entre a redução de custos ou de violações de SLO, e de sua capacidade em atender diferentes níveis de expectativa desses objetivos para as diferentes aplicações e dimensões de recursos considerados. Os resultados dessas análises são apresentados nas próximas seções.

6.3.1 Objetivos conflitantes e desempenho do provisionamento

Em uma análise geral das abordagens de provisionamento proativo é possível observar o desempenho das técnicas em termos dos objetivos de provisionamento e identificar a relação entre esses objetivos. A Figura 6.1 apresenta os resultados de desempenho das abordagens de provisionamento proativo baseado individualmente em CPU e memória. Os gráficos apresentam o desempenho das abordagens em termos do percentual de violações de SLO e do custo do serviço relativo aos cenários base perfeitos. Em relação a esses cenários, custos negativos correspondem a economias de custo por parte do provisionamento proativo simulado.

A relação entre objetivos também pode ser minimamente observada, onde em geral as abordagens que apresentam maior percentual de violações de SLO são aquelas com menor custo de provisionamento, muitas das vezes menores que os custos obtidos no provisionamento automático perfeito. Isso ocorre devido a erros de sub provisionamento da infraestrutura que reduzem os custos e em contrapartida elevam o percentual de ocorrência de violações de SLO durante a execução da aplicação.

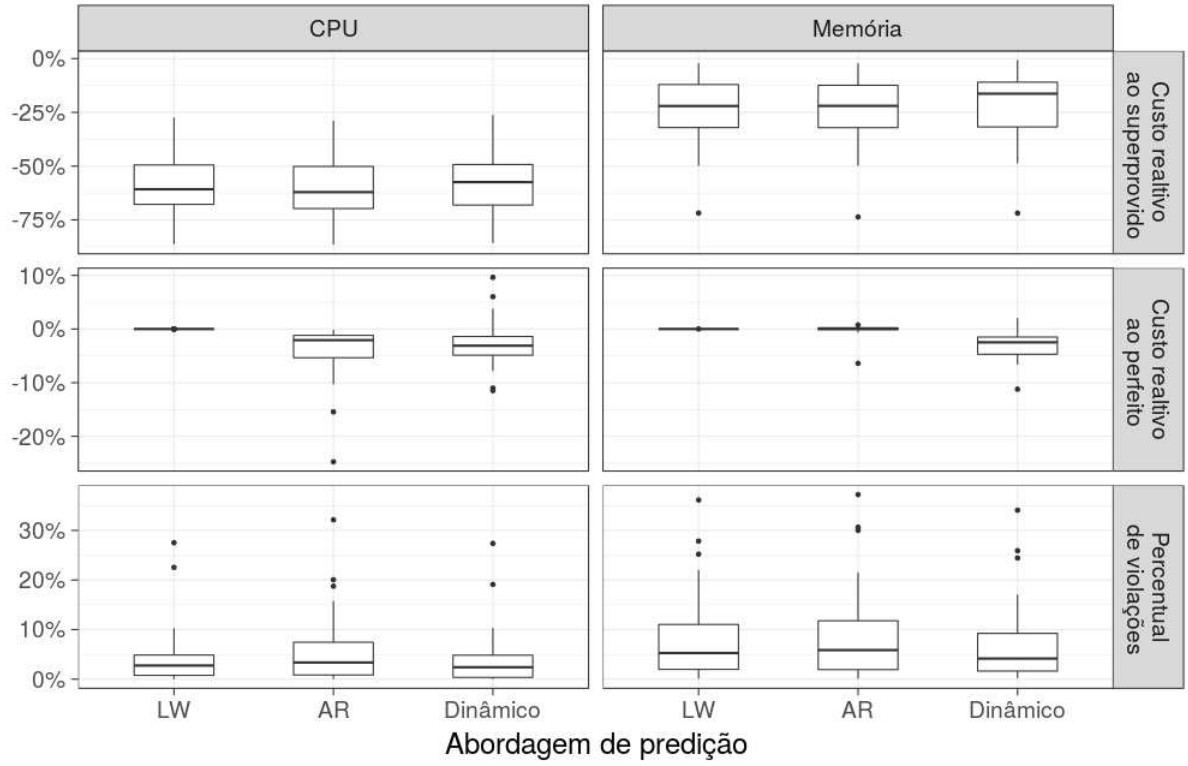


Figura 6.1: Desempenho da abordagem de provisionamento proativo baseada em utilização de recursos em termos do custo de provisionamento e do percentual de violações de SLO.

Os resultados são semelhantes para as diferentes abordagens, considerando ambas as dimensões de recursos, tanto em termos de custo de provisionamento quanto da quantidade de violações de SLO observadas. Em geral, os custos de provisionamento das abordagens proativas são inferiores aos obtidos pelo provisionamento perfeito, em média 2,3% e 1% para o provisionamento baseado em CPU e memória respectivamente. Em contrapartida, os percentuais de violação de SLO são considerados elevados, com aproximadamente 5% para

o provisionamento baseado em CPU e 8% quando baseado em memória. A economia de custo em relação ao cenário perfeito deve-se ao sub provisionamento da infraestrutura de execução, que é mais econômica porém mais suscetível a ocorrência de violações de SLO.

As economias de custo de provisionamento em relação ao cenário de super provisionamento estático perfeito, quando baseado em utilização de memória, são inferiores às economias obtidas no provisionamento guiado por utilização de CPU. Isso se deve à menor variação do nível de utilização e à baixa amplitude dos picos de utilização de memória das aplicações consideradas, que geram uma menor margem de economia para uma abordagem de provisionamento automático. O percentual de violações também é mais intenso no provisionamento baseado em memória. Apesar da variação nas demandas de memória serem menos frequentes, a intensidade dessas variações são suficientes para situar a solução de provisionamento em uma posição de ineficiência para suprir as demandas da aplicação.

Esses resultados evidenciam que o desempenho da abordagem preditiva de provisionamento, de forma semelhante às técnicas reativas, são diretamente dependentes ou relacionadas à variação da carga de trabalho das aplicações provisionadas por um serviço de provisionamento não intrusivo. No entanto, as economias de custo obtidas em relação ao super provisionamento estático perfeito e a proximidade de desempenho de custo com a abordagem de provisionamento perfeito proporcionam uma margem funcional para a solução de provisionamento. Isso significa, que com os índices de custo obtidos, é possível à solução de provisionamento priorizar objetivos de redução de violações de SLO, mesmo com consequentes elevações no custo de provisionamento.

6.3.2 Redução da ocorrência de violações de SLO

Devido à margem funcional proporcionada pela economia de custo obtida pelas abordagens proativas, é possível fazer uso de mecanismos para reduzir a ocorrência de violações de SLO ao realizar o provisionamento automático. Uma técnica amplamente abordada na literatura consiste em aplicar filtros nos dados utilizados para alimentar os modelos de predição, a fim de remover ruídos e pequenas variações que reduzem a acurácia dos preditores de séries temporais. Ruídos ou pequenas variações nos dados de utilização de CPU e memória usados pelos preditores podem alterar o funcionamento dos modelos de predição, prejudicando a detecção de tendências, sazonalidades e periodicidades dos dados de utilização. Desta

forma, essas influências podem causar erros de estimativa de utilização, que podem causar as violações de SLO observadas.

De forma geral, a técnica de filtragem considerada realiza uma conversão dos dados da série temporal, de um uso parcial da capacidade de recursos alocada em uma série de uso inteiro das capacidades. Com essa mudança, é possível mitigar o efeito de pequenas variações da carga de trabalho no desempenho do serviço de provisionamento automático, uma vez que os dados considerados para alimentar os preditores passam a ter menor variabilidade. Para uma série de itens da série temporal da carga de trabalho da aplicação, a técnica de filtragem aplica para cada demanda u, definida em núcleos de CPU e *gigabytes* de memória, a uma função teto $\lceil u \rceil$, resultando na demanda inteira do recurso considerado para cada intervalo de tempo da série. Por exemplo, para uma série temporal com valores de demanda $\{1{,}2;\ 3{,}1;\ 4{,}7;\ 2{,}1;\ 0{,}5\}$ aplica-se a função teto resultando na série $\{2;\ 4;\ 5;\ 3;\ 1\}$.

Em seguida, em formato de janela deslizante, é realizada para cada janela de tempo uma sumariação dos valores pertencentes a janela com o intuito de remover variações entre intervalos de tempo da série. Especificamente, uma janela de tempo com duração fixa de w é deslizada no tempo em saltos de $\frac{w}{2}$ e para cada janela os valores da série temporal são substituídos pelo valor máximo do teto da janela atual. Desta forma, os preditores de demanda passam a ser alimentados com séries temporais compostas pelos valores resultantes da aplicação da função teto e do janelamento subsequente. Por exemplo, considerando uma janela de tempo de tamanho $w = 2$ e a série de demandas inteiras $\{2;\ 4;\ 5;\ 3;\ 1\}$ são realizados 4 janelamentos: o primeiro destes considera os 2 primeiros valores da série de inteiros e gera a série $\{4;\ 4;\ ...\}$; o segundo janelamento desloca a janela em 1 intervalo e resulta em $\{4;\ 5;\ 5;\ ...\}$; o terceiro gera a série $\{4;\ 5;\ 5;\ 5;\ ...\}$; e após o último janelamento obtém-se a série filtrada $\{4;\ 5;\ 5;\ 3;\ 3\}$, que será utilizada pelo modelo de predição.

A avaliação da eficácia da técnica de filtragem foi realizada a partir de uma instanciação do modelo de simulação definido anteriormente. O modelo foi configurado para aplicar o filtro usando janelas de 1 hora de duração, com deslocamentos a cada 30 minutos, em concordância com o tempo mínimo de uso de uma VM considerado na definição do modelo de simulação. É importante não confundir essa janela de 1h com os intervalos de controle previamente definidos, que continuam sendo de 5 min. Para cada cenário avaliado, uma aplicação e as diferentes abordagens de predição com e sem filtro para ambas as dimensões

de recursos, mediu-se o percentual de violações de SLO obtidas e o custo relativo aos cenários de provisionamento perfeito e de super provisionamento estático perfeito para a mesma aplicação. A Figura 6.2 apresente o diagrama de caixa desses resultados de desempenho.

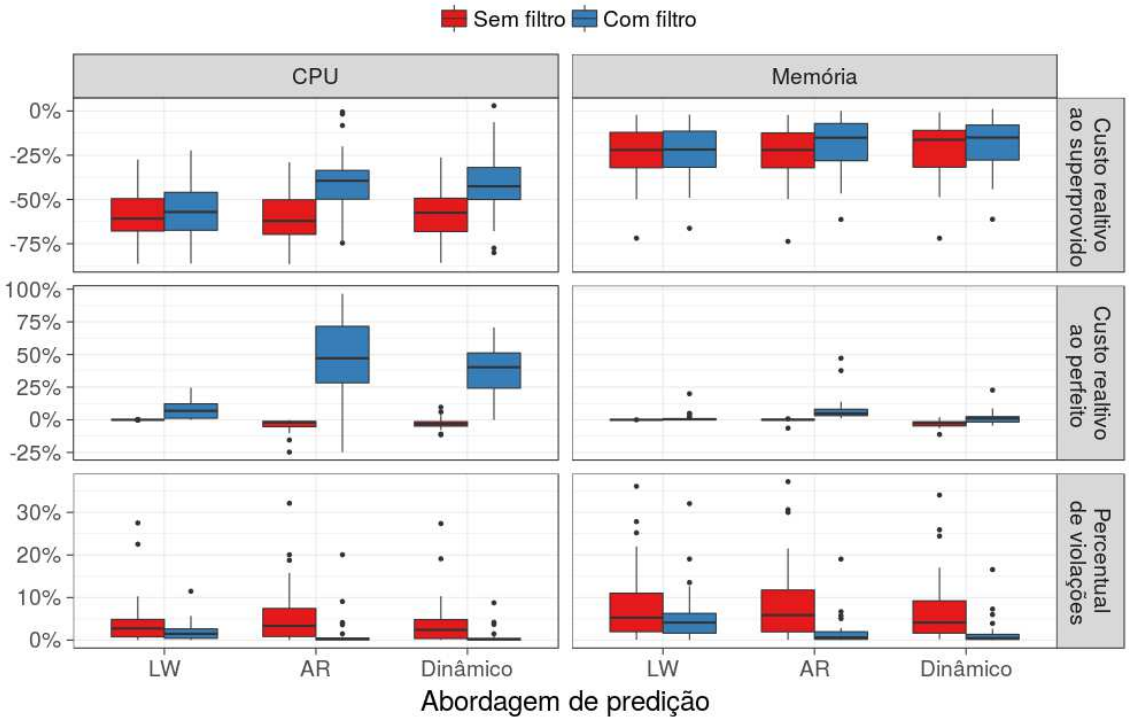


Figura 6.2: Desempenho da abordagem de provisionamento proativa baseada em utilização de CPU e memória a partir da técnica filtragem de dados de predição, em termos do custo de provisionamento e do percentual de violações de SLO.

Como observado, os resultados evidenciam a melhoria na predição devido à técnica de filtragem. Com relação ao número de violações de SLO, a técnica de filtragem apresentou uma redução absoluta no percentual médio de violações de 3,6% e 5% para o provisionamento baseado em CPU e memória, respectivamente. Isso foi obtido mesmo com um percentual médio de redução de custos de aproximadamente 46% para o provisionamento baseado em CPU e de 20% para o provisionamento baseado em memória. Isto evidencia que mesmo com a consequente elevação dos custos de provisionamento, a técnica de filtragem de dados de predição apresenta potencial de reduzir significativamente a ocorrência de violações de SLO durante o provisionamento da aplicação.

6.3.3 Técnicas para o controle dos objetivos de provisionamento

O provisionamento automático de aplicações em ambientes de IaaS possui objetivos conflitantes que correspondem a um *trade-off* entre a redução do percentual de violações de SLO, relacionadas com a QoS da aplicação provisionada, e a redução do custo de provisionamento através da diminuição da capacidade de recursos alocada. Em um cenário de provisionamento reativo, esses objetivos podem ser controlados a partir da configuração dos limiares de utilização de recursos da abordagem de provisionamento reativo e das ações a serem realizadas quando esses limiares são ultrapassados. No entanto, para abordagens proativas consideradas não há configuração que permita esse tipo de controle de objetivos diretamente no nível do modelo de predição considerado pela técnica. Desta forma, faz-se necessário o uso de técnicas auxiliares para proporcionar o provisionamento controlado em função dos objetivos de provisionamento. A seguir são analisadas duas técnicas de controle de objetivos de provisionamento.

Margem de segurança operacional

A primeira técnica desenvolvida para controlar os objetivos de provisionamento consiste em uma abordagem simplista que permite uma inflação controlada dos valores de predição produzidos. De forma específica, a abordagem utiliza uma margem de segurança operacional que permite a redução de sub estimativas por meio da expansão de cada uma das estimativas geradas pelos modelos de predição, que corresponde a um aumento percentual pré-definido dos valores de predição. Essa margem tem o objetivo de gerar um super provisionamento dinâmico e controlado da infraestrutura de execução para lidar com os modelos de predição que produzem resultados com custos reduzidos e com uma ocorrência significativa de violações de SLO, como os que foram avaliados previamente. Evidentemente, a capacidade da técnica de reduzir sub estimativas está atrelada a uma elevação nos custos de provisionamento.

Com a aplicação da margem de segurança, o planejamento de capacidade passa a utilizar predições de utilização aumentadas em um fator fixo (ρ) e o modelo de simulação utilizado anteriormente foi modificado para refletir o novo modo de operação do planejador de capacidade do serviço de provisionamento automático. Desta forma, para cada predição de consumo de recurso da infraestrutura virtual ($\hat{d}$), a capacidade planejada em termos do nú-

mero de instâncias, com 1 núcleo de CPU e 1 *gigabyte* de memória, necessárias para prover a aplicação é dada por $\left\lceil \frac{\hat{d}}{l-\rho} \right\rceil$, onde l corresponde ao limite de utilização de recursos definido no SLO estabelecido pelo provedor da aplicação.

O modelo de simulação foi utilizado com a mesma configuração do experimento realizado na seção anterior, com o limite de utilização de recursos fixado em 100% e a aplicação do filtro nos dados consumidos pelos modelos de predição. No entanto, foram adicionados à configuração do modelo diferentes níveis de margem de segurança operacional, com os valores de ρ variando entre 10% e 50%. O caso base consiste no provisionamento das aplicações consideradas sem o uso da margem de segurança operacional (margem de 0%) e com a aplicação do filtro, apresentado na seção anterior.

A Figura 6.3 mostra os resultados dos experimentos de simulação do provisionamento das aplicações com os diferentes abordagens de predição e configurações de margem de segurança operacional [1]. O resultado dessas simulações permite a análise do provisionamento com relação às métricas de violação de SLO e redução de custo, respectivamente em comparação aos cenários de provisionamento perfeito. Como esperado, o aumento da margem de segurança resulta em reduções significativas do percentual de violações de SLO obtidos pelos diferentes modelos de predição, com redução aproximada da média de violações de 2,25% no cenário sem o fator para 0.06% para o cenário com fator fixado em 50%, apara ambas as métricas de provisionamento. Em contrapartida, essas reduções causam uma elevação no custo de provisionamento. O cenário base apresenta um custo médio 33% inferior ao custo obtido no super provisionamento estático, enquanto que para a configuração com maior margem de segurança o custo médio é 23% superior.

Os resultados evidenciam a capacidade da técnica em promover o controle de objetivos de provisionamento a partir de abordagens proativas, apesar desse controle não mostrar-se linear entre as diferentes configurações de margem de segurança. A redução do número de violações de SLO é mais significativa entre os cenários sem margem de segurança e com uma margem mínima de 10%. Nesse caso, o percentual médio de violações é reduzido de 2,25% para 0,33%, considerando a análise conjunta de ambas as métricas. Isso é obtido mesmo com economias médias de custo em torno de 19% em comparação ao cenário de super provisionamento. Todavia, entre os cenários que fazem uso da margem de segurança a

[1] Por questões de visualização *outliers* dos resultados do percentual de violações de SLO foram omitidos.

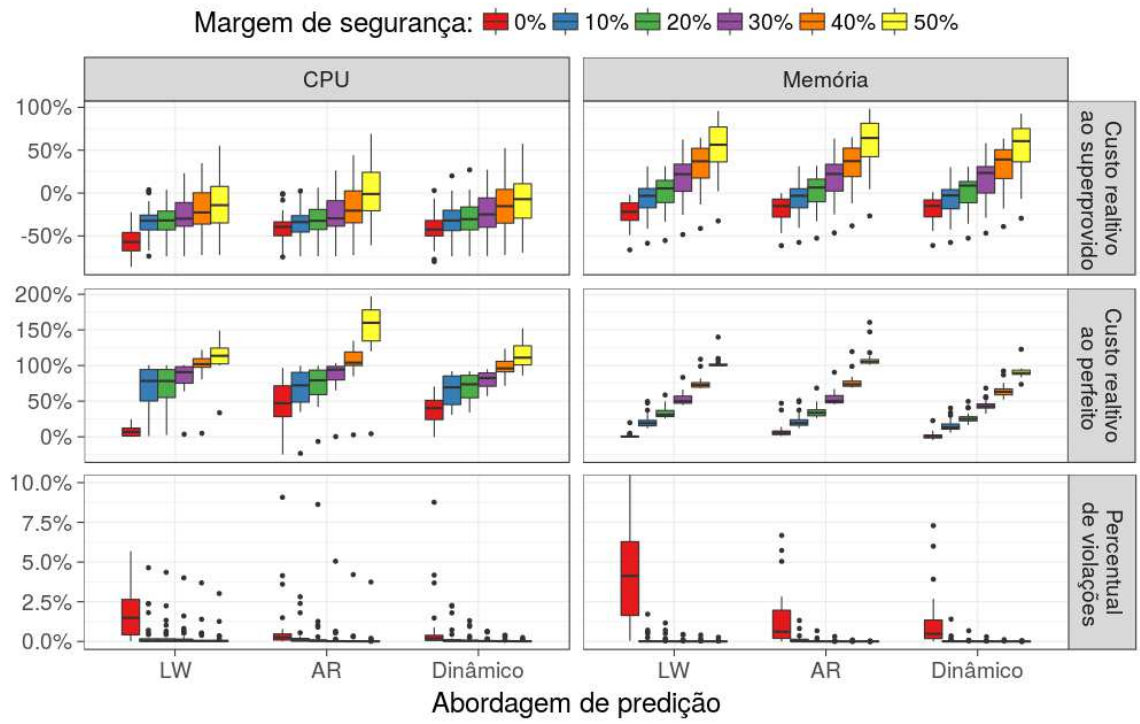


Figura 6.3: Desempenho da abordagem de provisionamento proativa considerando diferentes configurações de margem de segurança operacional em termos dos objetivos de provisionamento.

variação de desempenho da técnica ocorre de forma aproximadamente linear.

Correção de erros de predição

Uma abordagem mais sofisticada para promover o controle de objetivos de provisionamento consiste na utilização de modelos de predição que buscam corrigir possíveis erros de subestimativa. Essa abordagem foi proposta por Morais et al. [44] e baseia-se na análise do histórico de erros que conduzem ao sub provisionamento para corrigir as predições e reduzir a ocorrência de violações de SLO. A abordagem consiste em adicionar uma camada auxiliar aos modelos de predição que é responsável por analisar os padrões de erros capturados e realizar correções das estimativas de utilização produzidas. Ou seja, com base nos modelos iniciais de predição são geradas estimativas de demanda de recursos e estas estimativas são corrigidas com base em erros de predição cometidos anteriormente

O mecanismo de correção calcula o histórico de erros de predição ($e(t) = \hat{d} - d$) para

cada intervalo de tempo t e tenta correlacionar a sequência recente de erros de subestimativa (erros negativos) com sequências de erros de subestimativa no passado, através de uma função de autocorrelação (ACF, do inglês *Auto-Correlation Function*). O ponto no passado em que a correlação entre os erros é máxima é utilizado como um ponto central de uma janela de valores de erro de predição com tamanho configurável. Então, o valor absoluto do maior erro negativo dentro desta janela de correção é utilizado para corrigir a predição seguinte, de forma que essa correção é somada ao valor de utilização estimado. De forma geral, o algoritmo de correção de predição, como ilustrado na Figura 6.4, é dividido em em três etapas: (1) aferição dos erros de predição; (2) determinação da correlação máxima; e (3) cálculo da correção de predição.

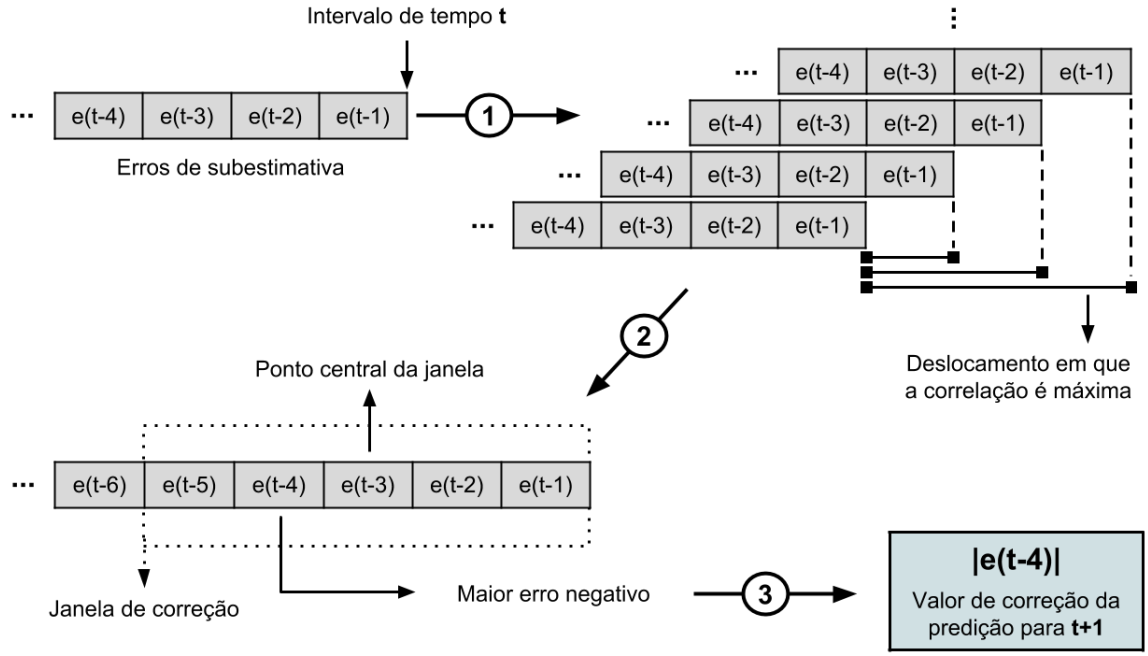


Figura 6.4: Visão ilustrativa do algoritmo de correção de predição baseado na correlação do histórico de erros de subestimativa.

O tamanho da sequência de erros de predição e o tamanho da janela de correção são parâmetros do mecanismo de correção de predições. No experimento de avaliação da técnica de controle de objetivos de provisionamento baseada na correção de predições foi utilizado um tamanho de sequência de erros de 2 semanas do passado, que corresponde a 4032 intervalos de tempo com 5 minutos de duração, e valores para o tamanho da janela de correção que

consideram 1%, 25%, 50% e 100% do tamanho dessa sequência de erros e atuam como um *grau de correção*. Por exemplo, para um sequência de erros com 1000 intervalos de tempo e um grau de correção de 50% o algoritmo de correção considera janelas de correção com 500 intervalos de tempo. A atuação do mecanismo foi simulado com base nas cargas de trabalho das 30 aplicações consideradas a partir de métricas de CPU e memória e de diferentes modelos de predição. Também foram utilizados diferentes configurações da margem de segurança operacional associadas ao filtro de dados de predição avaliado anteriormente. As Figuras 6.5 e 6.6 mostram o diagrama de caixa do percentual de violações de SLO e do custo de provisionamento em comparação aos cenários de provisionamento perfeito e de super provisionamento perfeito, respectivamente para o provisionamento baseado em CPU e memória [2].

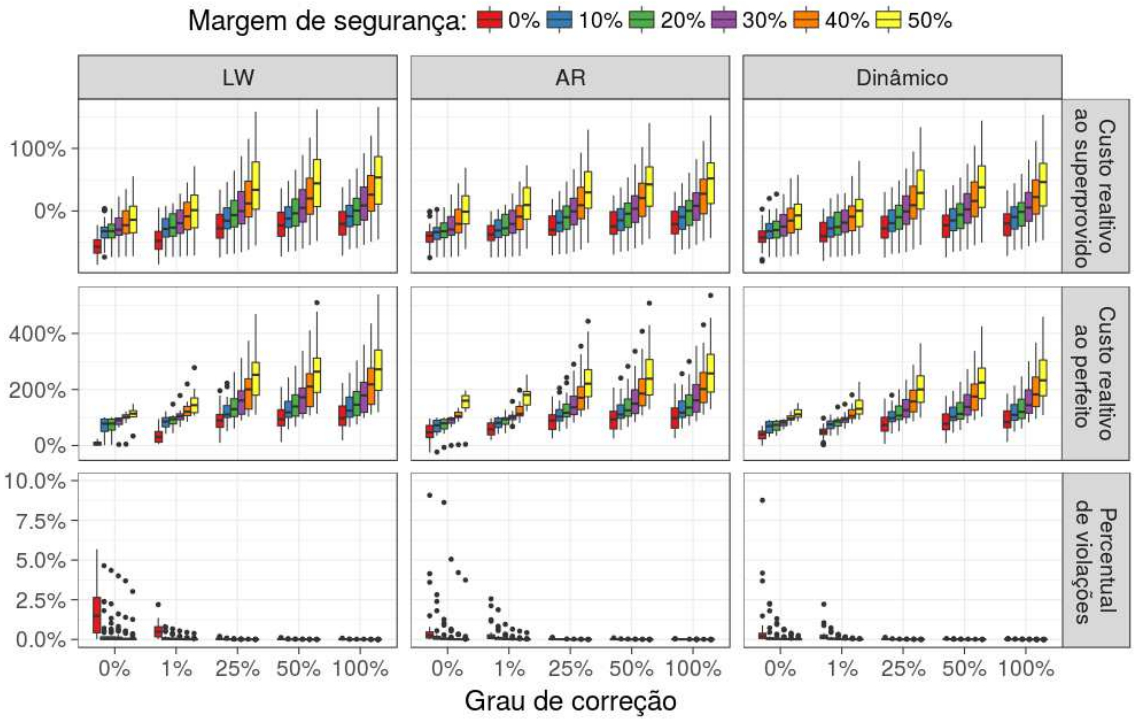


Figura 6.5: Desempenho da abordagem proativa considerando a técnica de correção de predições em termos dos objetivos de provisionamento, para o provisionamento baseado em CPU.

Para ambas as métricas de provisionamento, observa-se uma variação de desempenho da

[2]Por questões de visualização *outliers* dos resultados do percentual de violações de SLO foram omitidos.

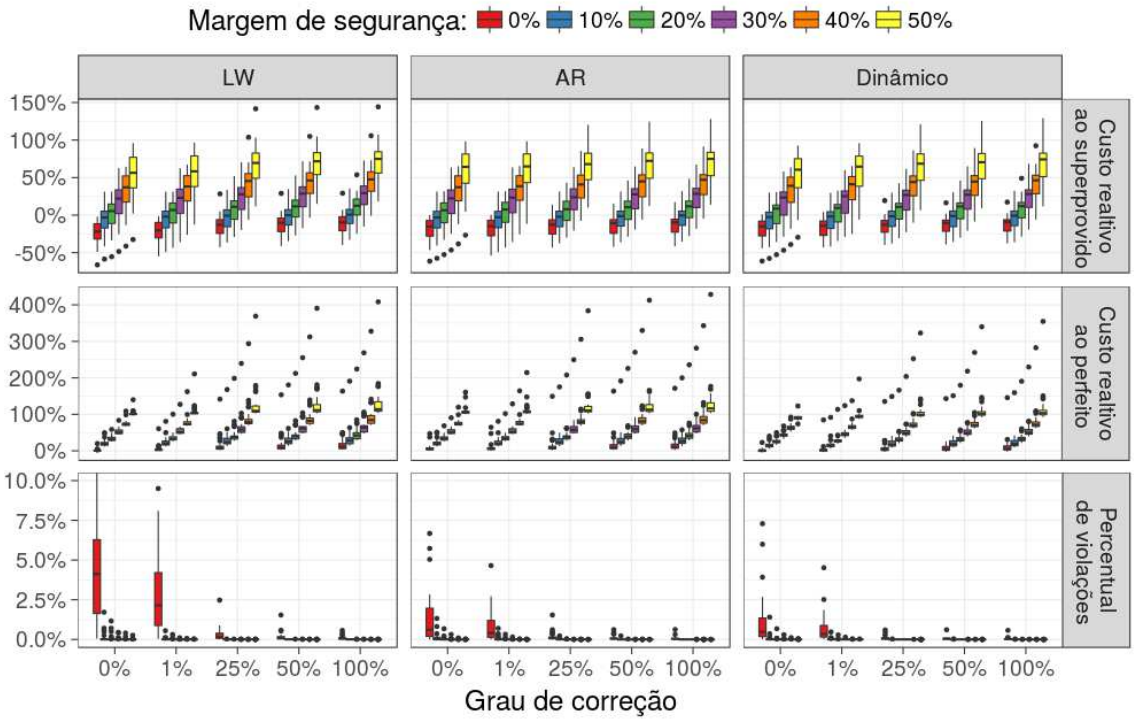


Figura 6.6: Desempenho da abordagem proativa considerando a técnica de correção de predições em termos dos objetivos de provisionamento, para o provisionamento baseado em memória.

abordagem de provisionamento proativa em termos do custo de provisionamento e do percentual de violações de SLO com o aumento do grau de correção. Estes resultados que tornam evidente o potencial da técnica de correção em prover o controle dos objetivos de provisionamento em um cenário de provisionamento como um serviço. Todavia, em comparação com a técnica de margem de segurança operacional, a abordagem de correção apresenta melhor desempenho, com um menor impacto de custo na redução de violações de SLO. A técnica de correção, sem margem de segurança, apresenta um percentual médio de violações de SLO de 0,04% no cenário com o grau de correção de 100%, com uma economia média de custo associada de 15%, em comparação ao super provisionamento estático. Por outro lado, sem o uso da correção, a abordagem com maior margem de segurança (50%) apresenta um percentual de violações semelhante ao da abordagem com correção, de 0,06% em média, no entanto o custo associado é significativamente superior, com sobrecusto de aproximadamente 23% em relação ao cenário super provido. Apesar disso, é possível que combinações das duas

técnicas, ou até mesmo o uso individual de uma delas, promovam um aprimoramento nos resultados quanto ao desempenho em termos dos objetivos de provisionamento.

6.3.4 Controle de objetivos de provisionamento

Como mostrado na seção anterior, as técnicas de margem de segurança e de correção de predições possuem o potencial de explorar o *trade-off* de objetivos de provisionamento. No entanto, é imprescindível que o controle desses objetivos seja eficientemente obtido a partir da configuração das técnicas propostas. Assim, busca-se configurações das abordagens de provisionamento proativo que sejam eficientes em provisionar diferentes aplicações a partir de um limite de violações de SLO e com economias de custo em comparação aos custos obtidos pelo super provisionamento estático perfeito, posto como um limite superior de custo de provisionamento.

A fim de verificar a eficiência de diferentes configurações das técnicas de controle de objetivos de provisionamento e dos modelos de predição considerados, os resultados de provisionamento foram agrupados segundo classes de desempenho. Considerou-se os diferentes cenários de configuração e instanciação do modelo de simulação e mediu-se o percentual de violação de SLOs observado e a capacidade de produzir custos inferiores aos do cenário base de super provisionamento. Foram estabelecidas três classes de limites de violações de SLO: 0% (sem violações), $\leq 0,1\%$ e $\leq 1\%$. Desta forma, é possível avaliar o desempenho das diferentes abordagens proativas e de suas configurações em termos da capacidade de atender os níveis desejáveis de QoS e de promover reduções de custos de provisionamento para diferentes aplicações.

As Figuras 6.7 e 6.8 apresentam o percentual de aplicações para as quais foi possível manter os níveis de QoS desejados e reduzir custos de provisionamento para cada uma das possíveis configurações e instanciações da abordagem de provisionamento proativo. Primeiramente, observa-se que nenhuma das configurações de provisionamento proativo consideradas foi capaz de provisionar eficientemente todas as aplicações, mesmo para limites de QoS mais flexíveis, com um percentual de violações de SLO menor ou igual a 1%. Para este caso, o percentual médio de aplicações para as quais os objetivos de provisionamento foram atingidos corresponde a aproximadamente 56% para o provisionamento baseado em CPU e a 34% para o provisionamento baseado em memória. Esse percentual é reduzido

para aproximadamente 49% e 27%, respectivamente para o provisionamento baseado em CPU e memória, quando o limite de violações é restringido para 0,1%. Para o cenário mais conservador, com um limite de violações de SLO em 0%, os objetivos são atingidos em média apenas para aproximadamente 6% e 5% para o provisionamento baseado em CPU e memória, respectivamente.

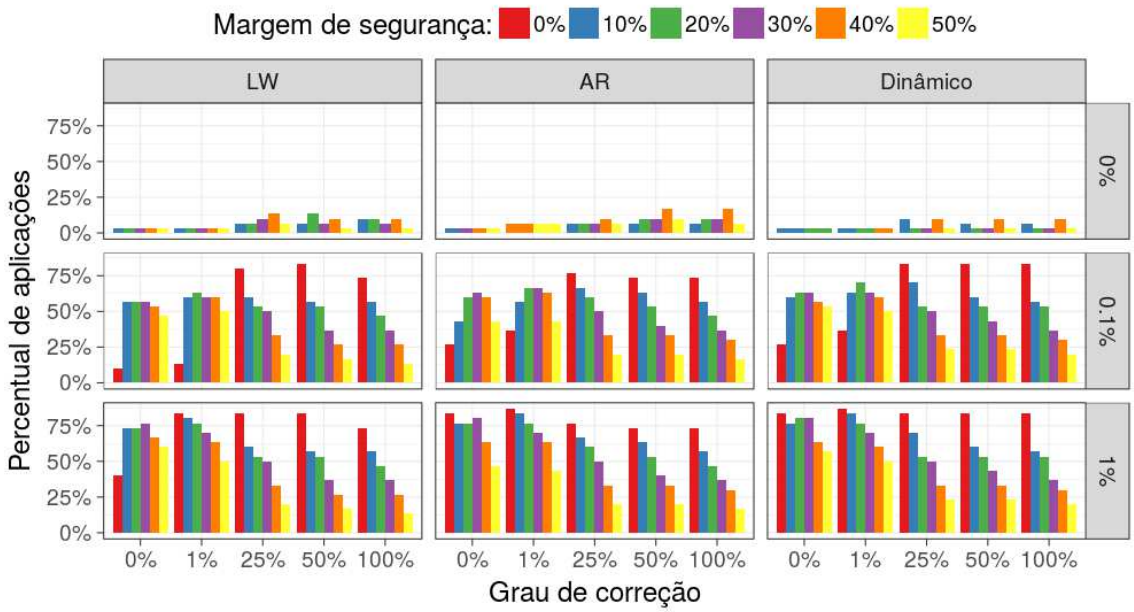


Figura 6.7: Análise do percentual de aplicações em que foi possível atingir os objetivos de custo e QoS no provisionamento proativo baseado em CPU.

Considerando os cenários de configuração mais eficiente, com o maior percentual de aplicações cujos objetivos de provisionamento foram atingidos, para cada classe de limites de SLO, o desempenho da abordagem baseada em métricas de memória consegue assemelhar-se ao desempenho do provisionamento baseado em CPU. Para o cenário menos restritivo de limite de SLO, o percentual de aplicações com objetivos satisfeitos é de aproximadamente 90%. Esse percentual é reduzido para algo em torno de 80% para ambas as métricas, quando o limite de violações é definido em 0.1%. Todavia, para o provisionamento baseado em memória, uma quantidade menor de configurações é capaz de atender os objetivos para um percentual representativo das aplicações. Em geral, quanto maior é a margem de segurança operacional utilizada menor é a quantidade de aplicações cujos objetivos de provisionamento

são atingidos, pois o custo se torna bastante elevado.

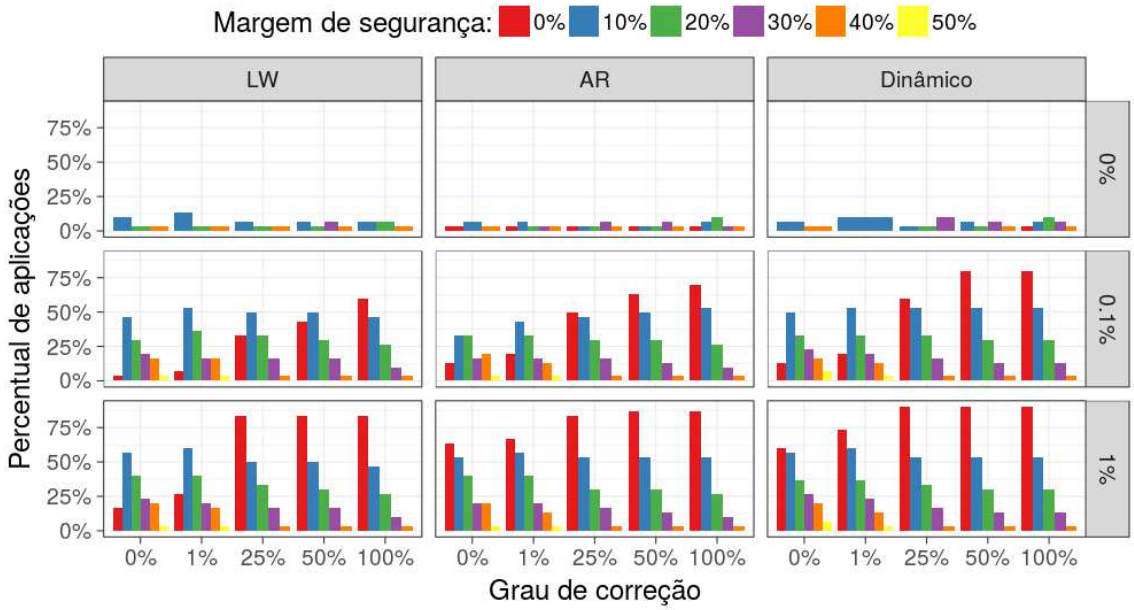


Figura 6.8: Análise do percentual de aplicações em que foi possível atingir os objetivos de custo e QoS no provisionamento proativo baseado em memória.

Em um cenário de objetivos de provisionamento conflitantes, é evidente que o cumprimento dos objetivos de QoS da aplicação está associado a custos de provisionamento, uma vez que quanto mais restritivo for o limite de violações de SLO definido maior será o custo de provisionamento necessário para atingi-lo. Essa relação entre objetivos também pode ser observada na Figura 6.9, que apresenta o menor custo de provisionamento associado à limitação da QoS de cada aplicação para cada classe considerada. O diagrama de caixa mostra, para cada aplicação cujos objetivos de QoS foram satisfeitos sem restrições de custo, o menor custo de provisionamento relativo ao provisionamento perfeito e ao super provisionamento estático perfeito, que consiste na seleção não realista do melhor cenário de configuração da abordagem proativa para obtenção do objetivo de QoS. A partir desses resultados, é possível observar que quanto mais restritivo é o limite de violações de SLO maiores são os custos de provisionamento associados e consequentemente maior é parcela das aplicações cujos objetivos de provisionamento não são atingidos.

Para o cenário mais flexível, em que o limite de violações de SLO é de 1%, os custos

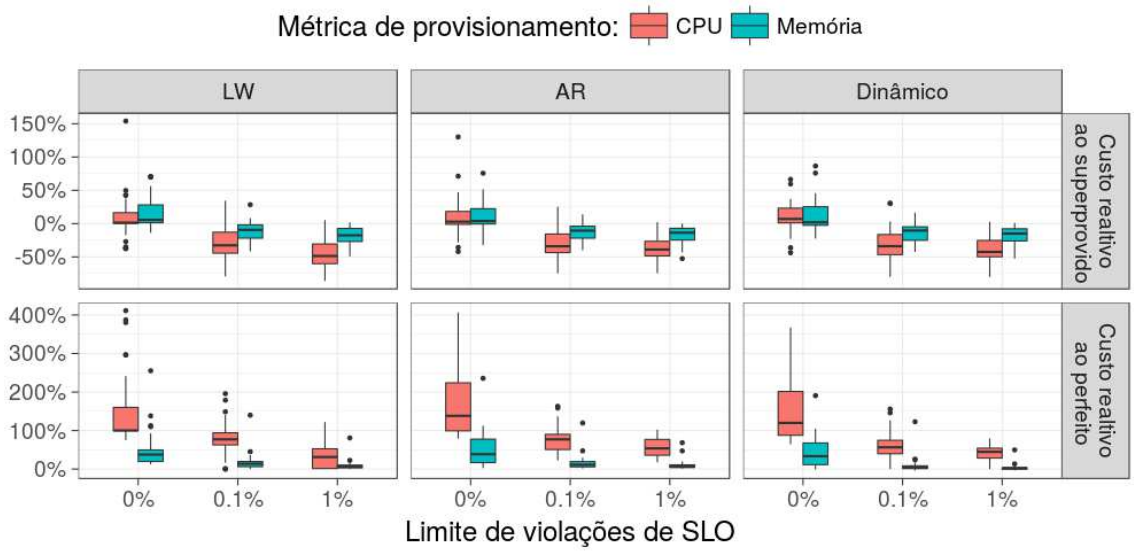


Figura 6.9: Análise de custos relativos ao provisionamento perfeito e ao super provisionamento perfeito para diferentes cenários de limites de violações de SLO.

de provisionamento das abordagens proativas são em média 42% e 7% superiores aos custos da abordagem perfeita, respectivamente para o provisionamento baseado em CPU e memória. Por outro lado, para um limite máximo de 0.1% de violações de SLO o custo médio é aproximadamente 71% e 14% maior que o obtido no provisionamento perfeito, respectivamente para o provisionamento com base em CPU e memória. Para a classe limitada a 0% de violações de SLO, os custos médios de provisionamento são os mais expressivos, aproximadamente 157% e 48% maiores que o custo do provisionamento perfeito correspondente, respectivamente para métricas de CPU e memória. Todavia, independente do limite de violações de SLO, o provisionamento baseado em memória apresenta valores de custo de provisionamento mais próximos ao custos praticados pelo cenário de provisionamento perfeito, possivelmente devido a menor variabilidade presente nas cargas de trabalho de utilização de memória das aplicações consideradas. No entanto, apesar do desempenho em termos de custo, o provisionamento baseado apenas em memória não garante que não existam violações de SLO relacionadas à utilização de CPU. Logo, o provisionamento automático não deve considerar apenas uma única dimensão de recurso, como será abordado em breve.

6.3.5 Eficiência de configuração por objetivo de provisionamento

É importante também estudar a eficiência de configuração das técnicas de provisionamento proativo para atingir os objetivos de provisionamento. Nesse sentido, a eficiência de configuração da abordagem também pode ser verificada pela análise da predominância de um conjunto restrito de configurações, de modelos de predição e técnicas de controle de objetivos, que são capazes de assegurar os objetivos de custo e QoS das aplicações, para diferentes níveis de restritividade de classes de SLO e métricas base do provisionamento. Essa análise é realizada a partir dos resultados obtidos do provisionamento prático considerando todas as diferentes configurações realizadas.

Para essa análise de predominância foram considerados dados referentes às três classes de limites de violações de SLO avaliadas na seção anterior. Entretanto, a classe com limite de 0% de violações de SLO não foi considerada nesse estudo devido ao percentual não significativo de aplicações cujo objetivo de eliminação de violações de SLO foi atingido. Para a classe com um limite 0,1% de violações de SLO a configuração com maior predominância consegue atingir os objetivos de SLO com o custo mínimo para em média 37% das aplicações e no máximo 38%, considerando os cenários com ambas as métricas de provisionamento. Por outro lado, para a classe menos restritiva (até 1% de violações) a configuração mais predominante foi suficiente para garantir os objetivos de QoS e de minimização de custo para em média 75% das aplicações e no máximo 86%, para os cenários com métricas de CPU e memória.

Desta forma, diferentemente da abordagem de provisionamento reativa que mostra-se ineficiente em termos de configuração, a abordagem proativa em alguns cenários menos restritivos alcança elevados índices de predominância de configuração, em geral para elevados graus de correção e não aplicação de margem de segurança. Isso lhe atribui um caráter generalista, apropriado para uma solução que deve compor um serviço de provisionamento automático em ambientes de IaaS.

6.3.6 Sumário de resultados da abordagem prática

Nessa seção foi realizada um análise de desempenho de abordagens de provisionamento proativo, baseado em métricas de CPU e memória, através de instanciações do do serviço de

provisionamento automático com base em diferentes modelos de predição e configurações de técnicas para o controle de objetivos de provisionamento. Especificamente foram considerados três modelos de predição (LW, AR e Dinâmico) e técnicas de controle que fazem uso de uma margem de segurança operacional e de um mecanismo de correção de predições. Além disso, foi considerado um filtro sobre os dados de utilização de recursos com o intuito de reduzir a ocorrência de violações de SLO. A partir dessas instanciações a abordagem de provisionamento proativa foi simulada e o desempenho de cada cenário foi avaliado em termos do custo de provisionamento e do percentual de violações de SLO resultante. Com base nesses resultados foram realizadas análises sobre a capacidade de controle de objetivos de provisionamento da abordagem proativa bem como sobre a sua eficiência em termos de predominância de configuração.

Primeiramente, observou-se que as técnicas de controle de objetivos consideradas permitem o serviço de provisionamento explorar objetivos antagônicos. Apesar da ineficiência no cenário com restrição total de violações de SLO (0% de violações), onde em média apenas de 5% a 6% das aplicações são provisionadas de forma que seus objetivos de provisionamento sejam atingidos, para os demais cenários com limites de violações de SLO menos restritivos o percentual médio de aplicações cujos objetivos de provisionamento são atingidos pode ser considerado significativo, principalmente para o provisionamento baseado em CPU. Nesse caso, em média 53% das aplicações são satisfatoriamente provisionadas, variando de 10% a 86%, enquanto que para o provisionamento baseado em memória esse percentual é de 31%, variando de 3% a 90%. Ou seja, para cenários mais restritivos é possível obter-se o controle de objetivos para um percentual minimamente significativo das aplicações, diferentemente do que o ocorre em cenários mais flexíveis.

Por outro lado, quanto à eficiência de configuração da abordagem proativa, esta mostra-se promissora para diferentes configurações de objetivos de provisionamento, tanto em relação ao percentual máximo de violações de SLO quanto à redução de custo de provisionamento em comparação a um cenário perfeitamente super provido. Para o cenário menos restritivo, onde aceita-se um percentual máximo de 1% de violações de SLO, nos melhores dos casos uma mesma configuração é capaz de assegurar tais objetivos para 86% das aplicações no provisionamento baseado em CPU e para 90% das aplicações no provisionamento baseado em memória. Para um objetivo mais restritivo, com um limite de 0,1% de violações, esses

percentuais ainda são significativos, com objetivos assegurados para 83% e 80% das aplicações, respectivamente para o provisionamento baseado em CPU e memória. No entanto, a abordagem não é completamente eficiente, para objetivos restritivos, sem a ocorrência de violações de SLO (0% de violações), esses percentuais são significativamente reduzidos para menos de 20% das aplicações cujos objetivos são atingidos, considerando ambas as métricas.

Desta forma, considera-se que a abordagem proativa de provisionamento apresenta desempenho não satisfatório para cenários com limites restritivos de violações de SLO, tanto em termos de assegurar os objetivos de provisionamento quanto em relação à eficiência de configuração da abordagem. No entanto, para cenários com uma maior flexibilidade nesse limite, é possível obter resultados de desempenho representativos no uso da abordagem proativa como um serviço de provisionamento automático, em termos de objetivos de provisionamento e eficiência de configuração, quando consideradas métricas individuais no provisionamento. Contudo, na próxima seção é apresentada uma análise do provisionamento proativo como um serviço de IaaS com base em métricas multidimensionais de utilização de CPU e memória.

6.4 Provisionamento Proativo baseado em Múltiplas Dimensões de Recursos

A gerência de capacidade deve ser realizada não apenas em uma dimensão de recursos, mas nas múltiplas dimensões cuja incapacidade de recursos pode gerar impactos no custo de provisionamento e, principalmente, no desempenho da aplicação provisionada. Nesse sentido, foi realizado um estudo sobre o provisionamento automático e proativo como um serviço em IaaS a partir de dados de utilização de recursos de múltiplas dimensões, especificamente CPU e memória. Foram realizados experimentos de simulação do provisionamento proativo das 30 aplicações consideradas em análises anteriores. O modelo de simulação foi configurado a partir de um combinação de modelos de predição LW e AR [3], com aplicação do filtro de dados de predição e com diferentes instanciações das técnicas de controle de objetivos de provisionamento. A margem de segurança operacional foi configurada com valores variando

[3]Especificamente, por questões de custo de execução, a abordagem de seleção dinâmica de modelos de predição não foi considerada nesses estudo.

de 0% (sem margem de segurança) a 50%, em passos de 10%, além disso foram considerados cenários sem correção de predições (0% do histórico de erros) e com correção, fazendo uso de diferentes graus de correção: 1%, 25%, 50% e 100%.

Desta forma, cada cenário de configuração é formado pela tupla $\langle$aplicação, modelo de predição, margem de segurança, grau de correção$\rangle$, o que resulta em 1800 cenários avaliados. Dos resultados de simulação foram selecionados aqueles que apresentam custos de provisionamento inferiores ao custo do provisionamento estático perfeito correspondente e agrupados as três classes de limites de violações de SLO já conhecidas. O percentual de violações de SLO é computado como sendo o percentual de intervalos de tempo em que a utilização de pelo menos um recurso (CPU ou memória) atingiu 100%. Essa classificação foi realizada com o intuito de contemplar diferentes níveis de QoS, em relação ao percentual de violações de SLO obtidas durante o provisionamento, na avaliação dos cenários de provisionamento simulados.

Com base nas diferentes classes de violações de SLO definidas, foi calculado para cada cenário de provisionamento considerado o percentual de aplicações cujos objetivos de provisionamento foram atendidos, de custo e QoS. A partir desse percentual é possível mensurar a eficiência de cada configuração em atingir tais objetivos. Para a classe de limite de violações de SLO mais restritiva, em que não se admite violações, nenhuma das configurações avaliadas foi capaz de evitar violações de SLO para todas as aplicações consideradas. As configurações mais eficientes para essa classe só foram responsáveis pela provisionamento satisfatório de aproximadamente 13% das aplicações. Em um cenário menos restritivo, com um limite de $0,1\%$ de violações, esse percentual eleva-se para a faixa dos 70%, com configurações eficientes no provisionamento de mais da metade das aplicações consideradas. No cenário com um limite de 1% de violações de SLO, o percentual máximo de aplicações cujos objetivos de provisionamento foram atingidos aproxima-se de 87%. Contudo, para nenhuma das classes ou configurações consideradas foi possível satisfazer os objetivos de provisionamento para todas as aplicações no provisionamento multidimensional. Isso evidencia a incapacidade do serviço de provisionar eficientemente todas as aplicações consideradas.

Além da análise de eficiência em atingir os objetivos, é importante avaliar a eficiência de configuração em atingir os objetivos de QoS com o menor custo possível de provisionamento para cada uma das aplicações. Ou seja, avaliar para cada aplicação cujos objetivos foram

atingidos o percentual de configurações que conseguem realizar o provisionamento com o menor custo. A Figura 6.10 apresenta o diagrama de caixa do percentual de configurações mais eficientes dentre o total de configurações capazes de respeitar os limites de violações e de custo base de provisionamento. Esse percentual é mais elevado para a classe de limite de SLO menos flexível (0% de violações), onde em média 38% das configurações que atingem os objetivos o fazem com o menor custo possível. Esse resultado é decorrente do baixo percentual de aplicações cujos objetivos são atendidos e de configurações que conseguem fazê-lo no provisionamento mais restritivo, de forma que as configurações mais rentáveis são predominantes dentre uma pequena quantidade de configurações com baixa eficiência de configuração. Para os casos mais restritivos, com um limite de $0,1\%$ e 1% de violações de SLO, o percentual médio de configurações que atingem os objetivos com o menor custo é reduzido para respectivamente 15% e 13%.

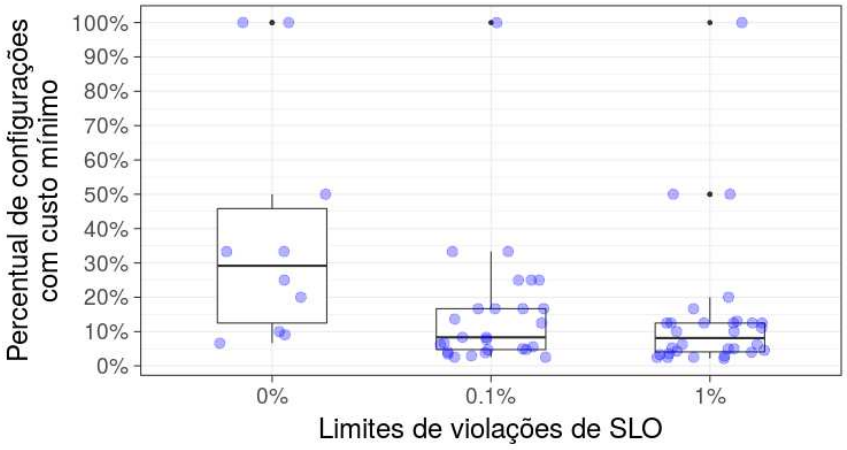


Figura 6.10: Análise do percentual de configurações mais eficientes por aplicação em termos de custo dentre as configurações que satisfazem os objetivos básicos de provisionamento.

Um outro fator importante a ser observado consiste no custo relativo ao cenário de provisionamento perfeito obtido pelas configurações que são capazes de atingir os objetivos do caso base, de custo e do limite de violações de SLO, para as diferentes classes. A Figura 6.11 apresenta o diagrama de caixa dos custos relativos ao cenário perfeito agrupados pelas classes de limites de violações de SLO. A classe mais restritiva (0% de violações) apresenta valores médios de custo relativo semelhantes aos das demais classes, com mediana do custo

relativo ao cenário perfeito em torno de 27%, no entanto, os cenários mais flexíveis proporcionam casos pontuais com grandes impactos de custo. Para esses casos, pelo menos 5% das configurações que atingiram os objetivos de provisionamento apresentam custos 140% superiores aos apresentados pela abordagem de provisionamento perfeito.

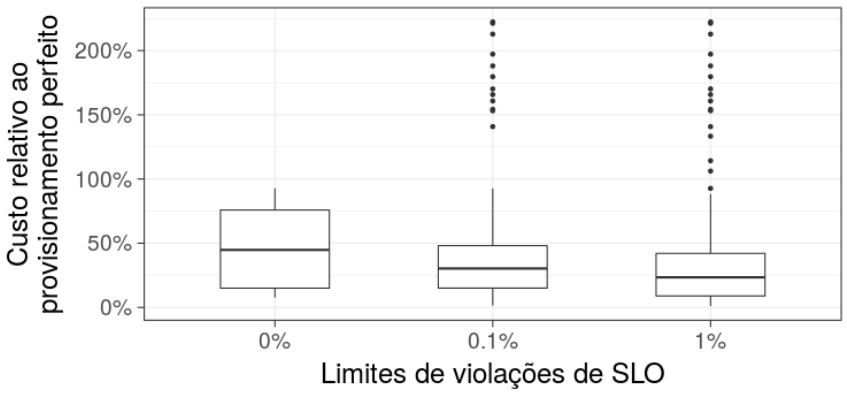


Figura 6.11: Análise de custos relativos ao provisionamento perfeito para diferentes cenários de limites de violações de SLO.

Com base nos resultados obtidos, é possível considerar que a abordagem de provisionamento automático e proativo é eficiente em termos dos objetivos de provisionamento, apesar de não ser possível obter configurações que sejam eficientes para todas as aplicações considerando a classe de limite de violações de SLO mais restritiva. Para os demais cenários, obtém-se níveis de eficiência significativos em garantir os objetivos de provisionamento, com um percentual elevado de aplicações cujos objetivos foram assegurados. Além disso, o custo em relação ao cenário de provisionamento perfeito não apresenta variações significativas para as diferentes configurações em que foi possível atingir os objetivos de provisionamento, apesar da baixa representatividade de configurações que conduzem ao custo mínimo.

6.5 Discussão e Conclusões

O capítulo trata do uso de abordagens de provisionamento proativo, baseadas em modelos preditivos, em um cenário de provisionamento automático como um serviço. Esses modelos são responsáveis por produzir estimativas de demandas da aplicação provisionada, que por sua vez são usadas para antecipar mudanças na carga de trabalho e consequentemente do

planejamento de capacidade necessário para supri-la. Desta forma, foi realizado um estudo sobre o desempenho da técnica proativa em basear um serviço de provisionamento automático nesses moldes. A partir desse estudo é possível obter conclusões sobre a eficiência de abordagens proativas ao compor um serviço de provisionamento automático em ambientes de IaaS para o cenário considerado.

As conclusões baseiam-se no desempenho da abordagem tanto em termos de eficiência de configuração dos modelos preditivos e das técnicas de controle de objetivos de provisionamento quanto na sua capacidade em atingir tais objetivos, considerando métricas singulares e multidimensionais como base do provisionamento. A abordagem proativa requer uma configuração mínima para operar de forma funcional, uma vez que para essa abordagem a expertise do provisionamento é significativamente transferida da configuração da solução para o modelo preditivo considerado pela abordagem (Questão de pesquisa 3). Além disso, a expansão dessa configuração em pelo menos 1 parâmetro, seja de margem de segurança operacional ou de grau de correção, é capaz de assegurar o controle de objetivos antagônicos de provisionamento, relacionados ao custo de provisionamento e a QoS da aplicação provisionada (Questão de pesquisa 2). Por esse motivo, conclui-se que a técnica proativa também apresenta desempenho satisfatório em termos de controle de objetivos.

Adicionalmente, o desempenho da abordagem também é eficiente em relação aos objetivos de provisionamento. Considerando cenários menos restritivos de limites de violação de SLO, onde é permitida um ocorrência mínima de violações, é possível respeitar esses limites mesmo com economias de custo em comparação aos custos da abordagem de super provisionamento estático perfeito, para uma parcela significativa das aplicações consideradas (Questão de pesquisa 1). Além do mais, esse desempenho é representativo para o provisionamento baseado em ambas as métricas de provisionamento, CPU e memória, o que reforça o caráter generalista da abordagem e sua independência de perfis de aplicações e suas cargas de trabalho (Questão de pesquisa 4).

Por outro lado, diferentemente da técnica reativa cuja implementação resume-se ao desenvolvimento de regras programáveis de provisionamento, a abordagem proativa requer o uso de um planejador de capacidade que, além de fazer uso de modelos preditivos, precisa modelar a relação entre as métricas estimadas e as ações de provisionamento a serem realizadas. Além disso, apesar de existirem implementações genéricas de modelos preditivos

disponíveis, a melhoria da acurácia dessas soluções requer o desenvolvimento de técnicas preditivas mais sofisticadas que necessitam de um esforço maior de implementação e adaptação das abordagens existentes. Desta forma, conclui-se que a solução proativa apresenta uma complexidade de implementação superior se comparada à técnica reativa de provisionamento automático (Questão de pesquisa 5).

Em resumo, a partir de uma abordagem de provisionamento mais sofistica, baseada em estimativas de demanda da aplicação provisionada produzidas por modelos preditivos, é possível obter-se uma solução de provisionamento considerada eficiente para um cenário de provisionamento como um serviço, apesar de apresentar maior complexidade de implementação. Essa eficiência é percebida em termos da generalidade da técnica em relação às aplicações provisionadas e métricas de provisionamento consideradas. Além disso, o desempenho da técnica também está associado à eficiência de configuração e a sua capacidade de atingir os objetivos de provisionamento e garantir o controle destes, ao fazer uso de técnicas de margem de segurança e correção de predições. Desta forma, considera-se que a abordagem de provisionamento proativo apresenta um potencial significativo a ser explorado na construção de um serviço de provisionamento automático em ambientes de IaaS, que opere de forma não intrusiva à aplicação provisionada.

Capítulo 7

Múltiplos Tipos de Instância de VM no Provisionamento Automático

Nesse capítulo é realizado um estudo sobre o uso de múltiplos tipos de instância de VM no provisionamento automático de recursos em ambientes de IaaS a partir de métricas não intrusivas de utilização de CPU e memória. Especificamente, o estudo refere-se à escolha do tipo de instância de VM a ser considerado no planejamento de capacidade da infraestrutura executando a aplicação durante a atuação do serviço de provisionamento automático. Principalmente, o trabalho tem como objetivo evidenciar a necessidade do uso de uma técnica de seleção dinâmica de tipo de instância de VM com o intuito de otimizar do uso dos recursos alocados, que por conseguinte promove reduções no custo de provisionamento e permite melhorias no desempenho do serviço de provisionamento proposto. Desta forma, esse capítulo propõe uma abordagem de uso da técnica de seleção dinâmica de tipos de instância de VMs associada ao provisionamento automático e proativo de recursos como um serviço de de IaaS [1].

7.1 Introdução

Como já destacado anteriormente, infraestruturas virtuais providas por empresas de IaaS e Computação na Nuvem são ambientes adequados para execução de aplicações horizontal-

[1] Resultados preliminares desse estudo foram publicados na 8ª edição da *IEEE International Conference on Cloud Computing Technology and Science* (CloudCom 2016) [45].

mente escaláveis e, por este motivo, podem ser potencialmente exploradas na oferta de um serviço de provisionamento automático como descrito nesse trabalho. Na prática, o usuário do provedor de IaaS (provedor do serviço de provisionamento automático nesse contexto) adquire recursos computacionais para executar e provisionar uma aplicação de interesse, que são oferecidos como VMs tipificadas por tamanhos pré-definidos [28] e referenciados por tipos de instância de VM.

Os tipos de instância oferecidos por um provedor de IaaS são definidos em termos da capacidade provida por estes em diferentes dimensões de recursos (por exemplo CPU, memória, disco, etc.). Tipicamente, tipos de instância variam entre si no tamanho da instancia (por exemplo pequeno, médio, grande, etc.) e na proporção entre as capacidades dos diferentes tipos de recursos disponíveis. Ou seja, relativamente mais memória que CPU, relativamente mais CPU que memória, e assim por diante. Cada tipo é associado a um preço por tempo de uso que é definido com base na quantidade de recursos oferecidos pela instância nas diferentes dimensões. Além do mais, independentemente do tipo de instância, as VMs são tarifadas com base no período de alocação e tarifação (por exemplo 1 hora).

Diversas são as soluções de provisionamento automático de recursos propostas na literatura compatíveis com o cenário de provisionamento automático como um serviço de IaaS, sejam operando de modo reativo ou proativo. Como analisado no Capítulo 5, soluções reativas apresentam limitações quando empregadas nesse cenário de provisionamento automático como um serviço e em alguns casos esse tipo de abordagem não é adequada para limitar impactos na QoS da aplicação provisionada. Assim, o estudo desenvolvido nesse capítulo é realizado com base em abordagens de provisionamento proativas, que apresentaram um maior potencial a ser explorado ao oferecer um serviço de provisionamento de recursos em IaaS.

Nesse sentido, a literatura é vasta em termos de soluções para provisionar automaticamente e proativamente uma aplicação horizontalmente escalável [2, 14, 15, 44, 52, 56, 62]. No entanto, como discutido no Capítulo 3, a importância de decidir sobre o tipo de instância de VM a ser usado no processo de provisionamento ao longo do tempo de execução da aplicação tem sido na maioria das vezes negligenciada por esses trabalhos. O objetivo principal tem consistido na estimativa das necessidades de recursos da aplicação, independentemente do tipo de instância de VM considerado.

É evidente que a quantidade exata de VMs necessárias para executar uma aplicação é determinada de acordo com a demanda futura estimada para todas as dimensões de recursos e com o tipo de instância usado para executar a aplicação. Desta forma, o serviço de provisionamento dever ser capaz de decidir sobre o tipo de instância mais compatível com as demandas por recursos observadas. Além do mais, se o aumento e a redução da demanda ao longo do tempo não for proporcional entre todas as dimensões de recurso, então decidir por apenas um tipo de instância de VM durante toda a execução da aplicação pode não ser a decisão mais eficiente em termos de custo de provisionamento. Como será visto na Seção 7.2, uma análise de cargas de trabalho de aplicações de diferentes conjuntos de dados com diferentes durações[2] confirma que pelo menos 75% das aplicações apresentam razões entre CPU/memória substancialmente diferentes ao longo do tempo, o que justifica o uso de mais de um tipo de instância durante o provisionamento.

Nesse capítulo é proposta uma abordagem para realizar a seleção de tipos de instância de VM no contexto do provisionamento automático e proativo, de aplicações horizontalmente escaláveis, como um serviço de IaaS. A abordagem tem como objetivo otimizar o uso da infraestrutura de execução alocada a partir da seleção dinâmica e periódica dos tipos de instância que melhor se adéquam às demandas da aplicação, permitindo que o desempenho do serviço de provisionamento proposto seja aprimorado pela redução dos custos de provisionamento das aplicações.

Dado o contexto de provisionamento como um serviço, espera-se a não intrusividade, limitando as informações necessárias para funcionamento do serviço de provisionamento à utilização de recursos em diferentes dimensões (CPU e Memória). Nesse sentido, a abordagem proativa considerada no estudo prevê a demanda da aplicação para cada dimensão de utilização de recurso e com base nessas estimativas decide o tipo e a quantidade da instância de VM que deve compor a infraestrutura de execução. O tipo de VM com melhor relação custo-benefício considerando as diferentes dimensões de recursos estimados é selecionado dentre os tipos de instância de VM disponibilizados pelo provedor de IaaS. A seguir, serão apresentados diferentes estudos sobre a seleção dinâmica de tipos de instância no serviço de provisionamento automático e proativo de aplicações horizontalmente escaláveis em IaaS.

[2]Dados de aplicações de clientes da HP, utilizado em análises anteriores, e dados de aplicações em execução em um centro de processamento de dados da Google [65].

7.2 Provisionamento Automático com Seleção Dinâmica de Tipos de Instância

No contexto de provisionamento desse trabalho, aplicações horizontalmente escaláveis são executadas em VMs dedicadas, adquiridas de um provedor IaaS. Cada tipo de instância caracteriza uma VM em termos de suas capacidades de recursos (CPU, memória, disco, etc.). Do usuário do provedor de IaaS é cobrada uma taxa previamente acordada por cada intervalo de tempo, de duração inferior ou igual ao mínimo intervalo de tempo considerado (tipicamente de 1 hora), em que a VM esteve alocada, que consiste no ciclo de tarifação praticado pelo provedor de IaaS. Essa taxa a ser paga por ciclo de tarifação é definida com base nas capacidades de recursos oferecidos por cada tipo de instância de VM. Por questões de simplicidade, a abordagem de provisionamento explorada nesse estudo pressupõe que cada período de provisionamento (o tempo entre a aquisição e a liberação de uma VM) ocorre em sincronia com o ciclo de tarifação do provedor, negligenciando o tempo de provisionamento requerido pela solução [3].

Variações das intensidades das utilizações observadas para as diferentes dimensões de recursos resultam em mudanças na quantidade de recursos necessários para executar a aplicação ao longo do tempo. Essas variações podem causar mudanças na relação entre valores de utilização dos recursos, para qualquer par de recursos, e assim influenciar na decisão sobre o tipo adequado de instância de VM a ser usado. Desta forma, além de decidir sobre a quantidade de recursos necessários, o serviço de provisionamento deve considerar o tipo de instância que melhor acomoda essa demanda de recursos. Nesse processo, apesar da possibilidade de uso simultâneo de diferentes instâncias de VM, por questão de simplicidade assume-se aqui que apenas um único tipo de instância de VM é usado durante cada período de provisionamento.

Na presença de falhas na estratégia de provisionamento automático, a aplicação pode atingir um estado de saturação indesejável em que os recursos são super utilizados e os níveis de utilização desses recursos atingem patamares não aceitáveis. Esse cenário pode levar a

[3]Para fins específicos de estudo do provisionamento automático, decisões de provisionamento devem ser antecipadas para levar em conta o tempo de provisionamento. No entanto, o foco estudo consiste no uso de múltiplas dimensões de recursos e na seleção dinâmica de tipos de instância de VM.

violações dos objetivos de nível de serviço da aplicação (SLOs), definidos segundo limites de utilização de recursos, e possivelmente a penalidades (perdas econômicas) ao provedor da aplicação em questão. Desta forma, quanto maior for a utilização de um recurso, menor será a QoS da aplicação em execução nele, especialmente quando a carga de trabalho exceder um determinado limite de utilização que estabelece o ponto de saturação da aplicação. Nesse sentido, assume-se a presença de um sistema de monitoramento que coleta periodicamente a utilização dos recursos das VMs ativas que executam a aplicação, em todas as dimensões de interesse.

Desta forma, para cada dimensão de recurso considerada, existe um SLO que determina o limite máximo aceitável de utilização desse recurso. Como consequência, se a utilização de cada recurso de interesse é mantida abaixo, mas o mais próximo possível do limite estabelecido, os SLOs da aplicação são satisfeitos e a QoS da aplicação desejada é atendida. Além do mais, o custo de execução da aplicação é reduzido pelo fato de que, em qualquer tempo de provisionamento, o número e tipos de instância de VM usados para executar a aplicação são os mais adequados possíveis. Logo, se o serviço de provisionamento automático atua eficientemente na busca desses objetivos de provisionamento, é improvável que ocorram violações de SLO, enquanto que o custo de provisionamento é minimizado no longo prazo.

7.2.1 Evidências da necessidade de múltiplos tipos de instância

Existem evidências da importância da seleção automática de tipo de instância para soluções de provisionamento automático e horizontal, obtidas a partir de análises considerando dois conjuntos de dados de utilização de aplicações executando em ambientes de produção para duas dimensões de recursos: CPU e memória. O primeiro consiste no conjunto de dados de aplicações de clientes da HP com duração média de 8 meses, usado em análises anteriores, e o segundo é formado por dados de aplicações em execução em um centro de processamento de dados da Google durante um período máximo de 1 mês. Foram consideradas todas as 30 aplicações do conjunto de dados da HP, consistindo de dados da utilização média e alocação de recursos no tempo, para as duas dimensões consideradas. No entanto, dos dados da Google foram consideradas apenas aplicações que executaram por 1 mês e que são multi-tarefas. Esta filtragem é necessária para melhorar as chances de capturar aplicações horizontalmente escaláveis de longa duração, resultando em uma seleção de 956 aplicações do conjunto de

dados da Google.

Para os dados da Google, foram usados os valores máximos de utilização de CPU e memória a cada intervalo de tempo de 5 minutos para calcular, respectivamente, a utilização máxima de CPU e memória durante intervalos de tempo maiores, com 1 hora de duração. Semelhantemente, os dados de utilização de aplicações da HP foram sumarizados em intervalos de tempo de 1 hora de duração, considerando as diferentes dimensões de recursos. Todavia, essa sumarização foi realizada a partir da utilização média de recursos a cada intervalo de 5 minutos, resultando em intervalos de 1 hora com o maior valor dentre as médias utilização de cada um dos subintervalos de 5 minutos. Isso fez-se necessário pois as informações sobre a utilização máxima de recursos não é presente no conjunto de dados da HP.

Na análise da seleção dinâmica de tipos, esses valores de utilização sumarizados por hora foram utilizados para calcular a *razão de proporcionalidade do uso de recursos* (RPUR, do inglês *Resource Proporcionality Usage Ratio*). A RPUR é na verdade uma função do tempo, definida como a razão do uso de CPU e memória ($\frac{CPU_{util}}{Mem_{util}}$) e é calculada para cada aplicação sendo provisionada. A metodologia aplicada no estudo baseia-se indiretamente nos valores de RPUR computados para selecionar o tipo de instância de VM que dever ser usado a cada período de provisionamento de uma aplicação. Desta forma, a técnica de seleção deve escolher o tipo de instância com a melhor relação custo-benefício para executar a aplicação a cada momento do tempo.

Diferentemente dos dados de aplicações da HP que disponibilizam tanto informações de utilização quanto de alocação de recursos em múltiplas dimensões no tempo, os dados da Google sobre a capacidade real de recursos alocados em servidores para executar as aplicações não são disponibilizadas. Os valores de utilização disponibilizados são relativos a uma máquina de referência cuja capacidade real não é conhecida. Com o intuito de abordar essa limitação, os valores de RPUR para o conjunto de dados da Google foram calculados com base em duas máquinas de referência, com diferentes proporções de capacidade de CPU e memória: (i) proporção 1:1, onde a máquina possui 1 GB de memória para cada núcleo de CPU presente; e (ii) proporção 1:4, onde a máquina possui 4 GB de memória para cada núcleo de CPU presente. Desta forma, os cenários de dados considerados são denominados "HP", "Google 1:1"e "Google 1:4".

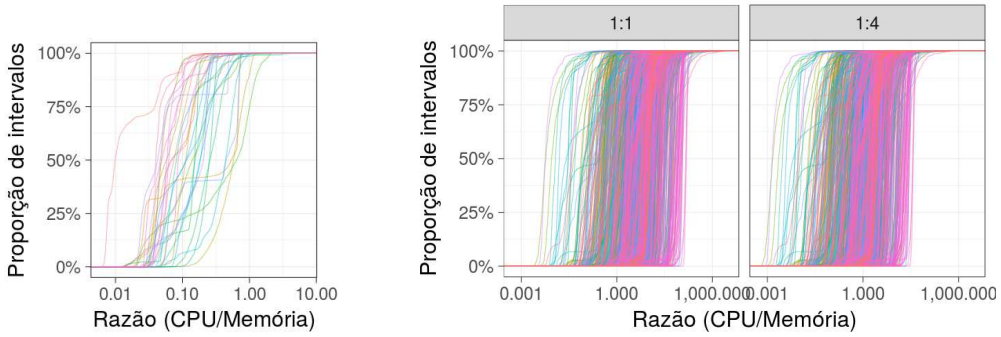
A primeira evidência coletada, sobre a necessidade de um mecanismo de seleção dinâ-

mica de tipos de instância no provisionamento automático de recursos, mostra que o RPUR varia consideravelmente no tempo para uma mesma aplicação. Com o objetivo de quantificar a variação de RPUR por aplicação, considerando os dois conjuntos de dados, foram identificados para cada aplicação o 5^o e 95^o percentis dos valores de RPUR. Os resultados indicam que durante pelo menos 10% de todos os intervalos de 1 hora de duração, as maiores proporcionalidades de consumo de recursos são 5 e 3 vezes maiores do que as menores proporcionalidades para metade das aplicações, respectivamente para dados da HP e da Google. Essa análise reforça a necessidade de seleção dinâmica do tipo de VM ao longo do provisionamento de uma aplicação.

Uma outra evidência da necessidade da técnica de seleção é que os valores de RPUR variam consideravelmente de uma aplicação para outra, independentemente do conjunto de dados considerado e da máquina de referência, para os dados da Google. Essa análise reforça a hipótese de que não existe um único tipo de VM que é o mais adequado para todas as aplicações provisionadas pelo serviço, sendo necessária a seleção dinâmica de tipo de VM por aplicação.

A Figura 7.1 apresenta a função de distribuição acumulada (FDA) dos valores de RPUR para cada aplicação. A partir desses dados é possível observar uma grande variedade de distribuições de RPUR, para os dois conjuntos de dados. Considerando o valor mediano do RPUR para cada aplicação, a diferença entre o RPUR mediano mínimo e o máximo é de cerca de 2 e 5 ordens de grandeza, para o conjunto da HP e Google respectivamente. Ou seja, a variação do consumo de CPU em relação ao consumo de memória pode ser até 10^2 e 10^5 vezes maior para valores medianos, considerando respectivamente dados da HP e Google.

Um ponto importante a ser destacado quanto à distribuição de valores de RPUR, independente da sua intensidade de variação, consiste na presença de diferentes perfis de tendência central entre os dois conjuntos de dados considerados. Para os dados da Google essa tendência mostra uma concentração de valores de RPUR próximos de 1, o que significa que existe um equilíbrio no uso de CPU e memória para as aplicações desse conjunto de dados. Essa característica demonstra que para o provisionamento dessas aplicações devem ser considerados tipos de instância tanto com predominância de capacidade de CPU em relação à capacidade de memória quanto os tipos que favorecem o cenário oposto. Por outro lado, a distribuição de RPUR para os dados da HP mostra que as aplicações desse conjunto são ma-



(a) Dados de utilização da HP

(b) Dados de utilização da Google

Figura 7.1: Distribuição de RPUR para todas as aplicações dos dois conjuntos de dados (HP e Google) em escala logarítmica.

joritariamente predominantes no consumo relativo de memória em comparação ao consumo de CPU, característica destacada pela predominância de valores de RPUR inferiores a 1.

Contudo, os resultados mostram que os valores de RPUR apresentam variação substancial, principalmente entre as aplicações da Google e entre uma mesma aplicação do conjunto de dados da HP. Esses resultados evidenciam a necessidade de um mecanismo de seleção dinâmica de tipos de instância no provisionamento automático e horizontal de recursos em ambientes de IaaS, com o objetivo de minimizar custos de provisionamento.

7.2.2 Serviço de provisionamento automático

O serviço de provisionamento atua na infraestrutura de execução da aplicação a cada intervalo de tempo, com periodicidade pré-definida, que corresponde a uma parcela do tempo total de provisionamento da aplicação. Por questão de simplicidade, considera-se nesse estudo que cada intervalo de tempo possui a mesma duração de um ciclo de tarifação de um provedor de IaaS. Por exemplo, considerando o modelo de preço da Amazon Web Services (AWS), cada intervalo de tempo possui 1 hora de duração e por conseguinte a periodicidade de provisionamento é a mesma. Desta forma, a cada intervalo o serviço de provisionamento, observando diferentes dimensões de recursos, deve (i) definir o tipo de instância com melhor relação custo-benefício para executar a aplicação e (ii) alocar a capacidade de recursos em termos da quantidade de VMs deste tipo. Essa decisão deve ser realizada a fim de manter a QoS da aplicação provisionada em níveis aceitáveis com o mínimo custo possível de

provisionamento.

Assim, o objetivo do serviço de provisionamento automático consistem em assegurar a execução de aplicações horizontalmente escaláveis em ambientes de IaaS com um nível de QoS aceitável e com o menor custo de provisionamento possível. Esse objetivo pode ser atingido assegurando-se que a aplicação provisionada execute, ao longo do tempo, usando a menor quantidade possível de VMs, do tipo de instância com melhor relação custo-benefício, que é suficiente para manter a taxa de violação de SLOs baixa. Esse limiar da taxa de SLO é definido como sendo o percentual máximo aceitável de violações de SLO estabelecido pelo provedor da aplicação como o nível de QoS esperado. O custo de provisionamento é computado como a soma do número de ciclos de tarifação de cada VM usada multiplicado pelo preço do tipo de instância de VM por ciclo de tarifação. Nesse processo, considera-se que apenas um único tipo de instância é usado em cada intervalo de tempo de provisionamento.

No entanto, para que o serviço de provisionamento atue eficientemente, é necessário que o mesmo tenha informações prévias sobre a utilização de recursos em diferentes dimensões para realizar o planejamento de capacidade para cada intervalo de tempo. Obviamente, não existe solução de provisionamento prática que seja capaz de dispor dessas informações de forma ótima, uma vez que não é possível antecipar os níveis de utilização da infraestrutura sem a existência de erros associados. Uma solução não realista de provisionamento nesses moldes — que usa um oráculo capaz de prever com exatidão os valores de utilização para cada intervalo de tempo — é proposta na próxima seção. Esta solução ótima é usada como melhor caso para base de comparação ao avaliar o desempenho de soluções práticas de provisionamento.

7.3 Seleção Ótima de Tipo de Instância de VM

Uma solução de provisionamento ótima é baseada em informações exatas sobre as demandas da aplicação no futuro próximo para cada dimensão de recursos (CPU e memória). Nesse cenário de provisionamento perfeito, o serviço de provisionamento é capaz de alocar ao longo do tempo a quantidade mínima de recursos com o tipo de instância de VM mais econômico e garantir que a aplicação seja executada sem violações de SLO em ambas as dimensões de recursos (ou seja, com um percentual de máximo aceitável de violações igual a 0%). Nessa

seção utilizamos uma solução de provisionamento perfeita simulada a partir de dados dos dois conjuntos de rastros de utilização previamente discutidos.

7.3.1 Modelo de simulação e instanciação

Usando como base o modelo de simulação descrito no Capítulo 4, foi desenvolvido um modelo de simulação do serviço de provisionamento que opera a cada hora e realiza o provisionamento com base em estimativas de demandas de utilização de recursos, como discutido no Capítulo 6, e na seleção dinâmica do tipo de VM a ser usado. O modelo de simulação foi implementado na linguagem de programação R [17], o que permite avaliar soluções de provisionamento automático e horizontal por meio de experimentos de simulação guiados por dados de utilização de recursos no tempo[4]. Nesse cenário, a solução de provisionamento ótima é alcançada quando o componente de predição é um oráculo que antecipa as futuras cargas de trabalho reais da aplicação.

O principal provedor de IaaS do mercado, Amazon AWS, expressa a quantidade de CPU alocada a um tipo de instância em termos da métrica ECU (do inglês *EC2 Compute Unite*), que é tida como a mais apropriada para comparar a capacidade real de CPU entre diferentes tipos de instância de VM [4]. Essa métrica é considerada para o cálculo do preço de uso de uma tipo de instância por ciclo de tarifação. Assim, foi necessária uma conversão dos valores de utilização de CPU dos rastros para valores de utilização de ECU. Essa conversão é realizada a partir da razão $\overline{e}$ entre CPU[5] e ECU de todos os tipos de instância da Amazon, onde o fator e para cada tipo de instância é definido a partir da Equação 7.1. Desta forma, os valores de utilização provenientes dos rastros são multiplicados pelo número de núcleos de CPU alocados nos dados originais e pela velocidade do processador, presente nos dados da HP e considerada neutra para os dados da Google (PCS igual a 1 GHz). Esse produto é então dividido pelo fator $\overline{e}$ calculado com base nos tipos de instância disponibilizados pela Amazon [6], resultando nos valores de ECU para os rastros.

[4]O código fonte do simulador encontra-se publicamente disponível em `https://github.com/fabiomorais/ASaaS/tree/master/multiple_types`.

[5]A capacidade de CPU estipulada pela Amazon corresponde ao produto entre número de núcleos virtuais (vCPU, do inglês *Virtual Central Processing Unit*) e a velocidade de *clock* do processador (PCS, do inglês *Processor Clock Speed*.

[6]Especificamente, o fator $\overline{e}$ consiste na média dos valores de e calculados a partir da Equação 7.1 para cada

$$e = \frac{vCPU \times PCS}{ECU} \tag{7.1}$$

Para instanciar o modelo de simulação foram considerados diferentes tipos de instância disponíveis na Amazon AWS, descritos na Tabela 7.1, uma vez que o tipo de instância com melhor relação custo-benefício não é previamente conhecido para o provisionamento. Foram selecionados tipos de instância que apresentam diferentes proporções de capacidades de ECU e memória, considerando três diferentes famílias: instâncias de propósito geral, otimizadas em computação e otimizadas em memória.

Tabela 7.1: Tipos de instância selecionados.

Referência	Modelo	CPU	ECU	PCS	memória	US$/hora
m3	m3.medium	1	3.0	2.5 GHz	3.75 GB	0.067
m4	m4.large	2	6.5	2.4 GHz	8 GB	0.126
c3	c3.large	2	7.0	2.8 GHz	3.75 GB	0.105
c4	c4.large	2	8.0	2.9 GHz	3.75 GB	0.110
r3	r3.large	2	6.5	2.5 GHz	15 GB	0.175

O provisionamento é realizado periodicamente a cada hora, que é a duração de um intervalo de provisionamento do simulador. Estimativas de demanda do futuro próximo são realizadas e computadas em termos do número de instâncias que podem suprir a demanda prevista. Isso é realizado para todos os tipos de instância considerados pelo modelo de simulação. A estimativa é calculada como o número mínimo de VMs, de cada tipo considerado, necessário para suportar a demanda da aplicação em todas as dimensões de recursos sem violações de SLO. Em uma solução ótima essas estimativas de demanda são exatas e previamente conhecidas. Assim, a informação sobre as demandas é usada para decidir para o próximo intervalo de provisionamento a quantidade exata de VMs de um certo tipo que deve executar a aplicação.

Portanto, cada tupla formada pela aplicação, tipos de instâncias, dimensões de recursos e dado de referência[7] define um cenário de simulação. Em cada simulação determina-se: (i)

tipo de instância disponibilizado.

[7]Combinação entre o conjunto de dados e a máquina de referência, para o conjunto de dados da Google.

o número de intervalos de tempo (de uma hora) em que a utilização dos recursos viola os limiares de utilização de SLO desses recursos, definidos em 100% de utilização para esse estudo[8]; e (ii) o custo da infraestrutura considerando o preço por hora de uso de cada tipo de instância selecionado para uso e o tempo de uso de cada VM alocada. Os experimentos de simulação foram realizados com dois objetivos em mente. Um deles é comparar os diferentes custos resultantes do uso de diferentes tipos de instância durante toda a execução da aplicação. O outro consiste em selecionar o melhor tipo de instância para cada intervalo de tempo e, nesse caso, encontrar o custo mínimo de execução.

7.3.2 Violações colaterais de SLO no provisionamento unidimensional

A partir da solução ótima do modelo de simulação foi realizada uma análise do caso em que apenas CPU ou apenas memória é usado como o recurso base do processo de provisionamento automático. Mais especificamente, a solução elimina violações relacionadas à dimensão de recurso sendo monitorada e o custo de infraestrutura é o mínimo, ie. o sistema decide o tipo de VM mais barato que irá satisfazer a demanda do recurso considerado. No entanto, violações de SLO relacionadas com o recurso não considerado ocorrem com frequência. Este resultado não é uma surpresa, dada a variação de RPUR calculada para as aplicações dos dois conjuntos de dados, e destaca a necessidade de considerar múltiplas dimensões de recursos no provisionamento automático, fato comumente negligenciado na literatura.

A Figura 7.2 apresenta o percentual de intervalos de tempo com violações de SLO quando uma única dimensão de recurso, CPU ou memória, é considerada como métrica base do provisionamento. As barras vermelhas correspondem ao total de violações de SLO de memória quando o provisionamento é conduzido unicamente pela dimensão de CPU, e as barras azuis consistem no cenário oposto.

Como pode ser observado, o provisionamento baseado em uma única dimensão de recurso geralmente leva a um número substancial de violações de SLO. Uma vez que o provisionamento faz uso de um preditor perfeito, todas as violações de SLO estão relacionadas

[8]Em um experimento de simulação esses valores não precisam ser cuidadosamente ajustados, todavia, na prática, parâmetros de referência específicos da aplicação provisionada podem ajudar o usuário do serviço de provisionamento a determinar com precisão estes limiares.

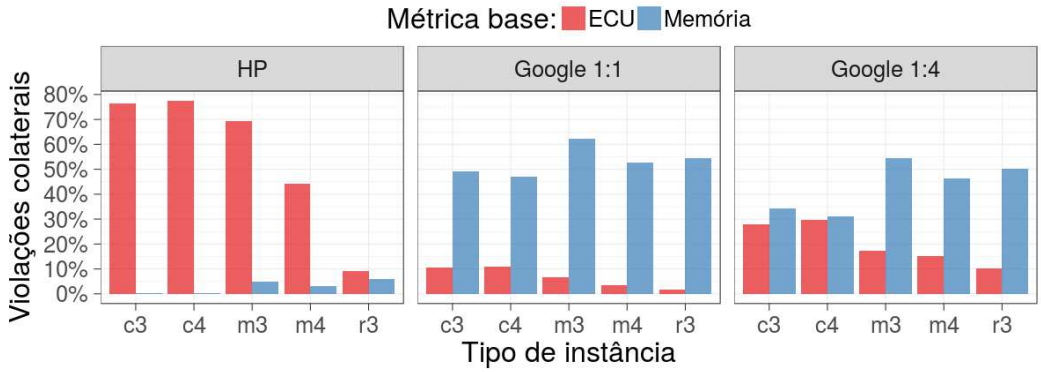


Figura 7.2: Violações de SLO quanto uma única dimensão de tipo de recurso é considerada no provisionamento automático.

às dimensões de recursos não observadas. Para o conjunto de dados da HP, o provisionamento baseado em memória obtém percentuais de violação de SLO para CPU próximos de 0% quando são considerados tipos de instância com maior proporção de capacidade de CPU (c3 e c4). No entanto, esse cenário é desfavorável ao custo de provisionamento, uma vez que os valores de RPUR para as aplicações da HP apresentam maior proporção de consumo de memória e os tipos de instância c3 e c4 apresentam proporção inversa de capacidade de recursos. Para os demais cenários, o percentual médio de violações de SLO para CPU aproxima-se de 5%, o que pode impactar a QoS das aplicações provisionadas.

Para os dados da Google, as menores taxas de violações ocorrem quando a métrica de CPU é usada para guiar o provisionamento, os tipos m3, m4 e r3 são usados e a máquina de referência tem uma proporção de 1:1. Nestes casos, o percentual total de violações SLO para memória é inferior a 10%. Para essas configurações, os tipos de instância têm grande quantidade de memória por núcleo de CPU (as maiores proporções entre os tipos de instâncias considerados) e o provisionamento conduzido por métricas de CPU leva a um número de VMs com memória suficiente para suportar a demanda da aplicação por memória. Esta situação também tem implicações nos custos, como mostrado em breve. Para os outros casos, o provisionamento baseado em uma única dimensão não é uma opção efetiva para manter a QoS das aplicações em níveis aceitáveis.

7.3.3 O impacto de custo ao se evitar violações de SLO

Como demonstrado anteriormente, o mecanismo de provisionamento automático precisa considerar múltiplas dimensões de recursos para evitar completamente violações de SLO. A maneira mais simples de evitar violações de SLO é incluir outras dimensões de recursos na solução de provisionamento automático. A decisão final consiste em uma solução gulosa em que a quantidade de VMs a ser provisionada é o máximo entre todas as quantidades requeridas pelas soluções baseadas em um único recurso. Quando outra dimensão de recursos é incluída no processo de provisionamento, o custo da infraestrutura naturalmente aumenta, uma vez que mais VMs serão alocadas em muitos momentos devido às demandas da segunda dimensão de recursos incluída.

Nesta seção, é realizada uma avaliação deste impacto de custo extra comparando o desempenho de custo do provisionamento automático ótimo baseado em múltiplas dimensões de recursos com a solução baseada em uma única dimensão, discutida anteriormente, para todas as aplicações consideradas dos dois conjuntos de dados. Para isso, o modelo de simulação foi instanciado para operar com base tanto em CPU quanto em memória, usando tipos de instância oferecidos pelo AWS e considerando as diferentes máquinas de referência para os dados de aplicações da Google. O custo total de infraestrutura para executar as aplicações é calculado considerando o preço de cada tipo de instância de VM usado e a duração do provisionamento de cada aplicação.

A Figura 7.3 apresenta o incremento de custo decorrente da adição uma uma outra dimensão de recurso no processo de provisionamento automático. Esse incremento de custo é relativo ao custo de infraestrutura obtido quando uma única dimensão de recurso é usada. As barras vermelhas representam o incremento de custo total relativo ao provisionamento baseado apenas em CPU e as barras azuis mostram o custo incremental comparado ao do provisionamento baseado apenas em memória.

Como esperado, os cenários com uma única dimensão que geraram os menores percentuais de violações de SLO no estudo anterior (c3 e c4 para os dados da HP e m3, m4 e r3 para os dados da Google) são os que apresentam menor aumento no custo total de provisionamento, com valores abaixo de 10%. No entanto, considerando todos os tipos de instância de VM, o aumento no custo de provisionamento foi de até 170%, com média de 50%, para os dados da HP e de até 400%, com média em torno de 100%, para os dados da Google. Esses

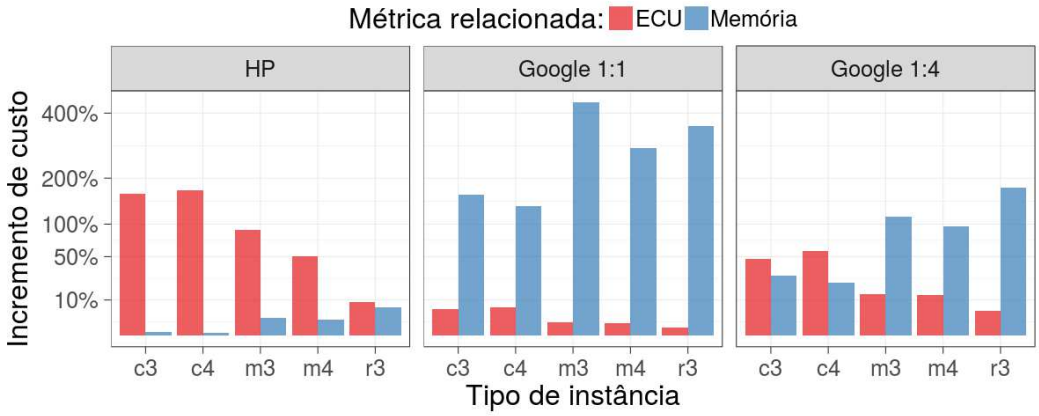


Figura 7.3: Incremento de custo do provisionamento multidimensional em comparação ao provisionamento baseado em uma única dimensão, em escala de raiz quadrada.

resultados determinam o custo de evitar violações sem seleção dinâmica de tipo de instância de VM, que podem ser consideravelmente elevados em muitos casos.

7.3.4 Redução de custos por meio da seleção dinâmica de tipos

Como visto anteriormente, considerar múltiplas dimensões de recursos no processo de provisionamento automático é eficaz em termos de eliminação de violações de SLO, mas não é eficiente quando considera-se o custo de infraestrutura. Nesse contexto, uma abordagem para aprimorar o processo de provisionamento consiste em selecionar dinamicamente o tipo de instância mais apropriado a cada intervalo de tempo de provisionamento. Desta forma, a solução de provisionamento automático considera múltiplas dimensões de recurso, como CPU e memória, e a cada intervalo de tempo seleciona o tipo de instância com melhor relação custo-benefício para executar a aplicação.

O potencial da seleção de tipo de instância na otimização do custo de provisionamento foi verificado pela análise da redução total de custo obtida, para todas as aplicações, por essa solução em comparação com o custo do provisionamento multidimensional baseado em um único tipo de instância, avaliado na seção anterior. A Figura 7.4 apresenta o gráfico de barras dessa economia de custo, que varia de 7% a até aproximadamente 50%. Além disso, para metade dos cenários, as economias de custo obtidas foram superiores a 24%.

A distribuição do número de diferentes tipos de instância usados no provisionamento

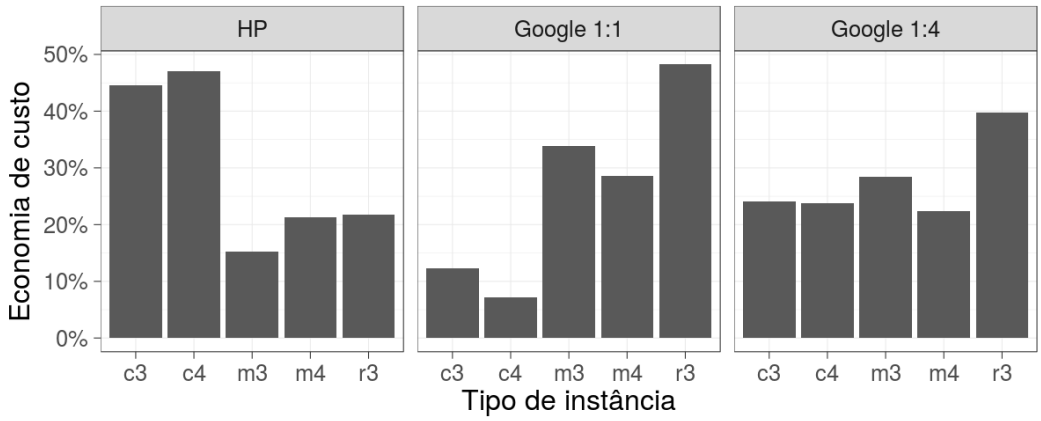


Figura 7.4: Economia de custo total obtida pela seleção dinâmica do tipo de instância mais apropriado no provisionamento automático.

ótimo de cada aplicação é apresentada na Figura 7.5, onde cada barra mostra o percentual de aplicações que usaram a quantidade de tipos discriminadas. No conjunto de dados da Google 14% das aplicações não necessitam de mais de um tipo de instância durante o provisionamento ótimo. Todavia, como os dados para o conjunto da Google tem apenas um mês de duração, não está claro se essas aplicações exigiriam mais tipos de instância se fossem observadas por um período de tempo mais longo. As demais 76% das aplicações da Google beneficiam-se do uso de mais de um tipo de instância durante a execução da aplicação. Além disso, para os dados da HP, cujo período médio de duração dos dados é de 8 meses, o uso de mais de um tipo de instância no provisionamento gera benefícios de redução de custo de infraestrutura para todas as aplicações.

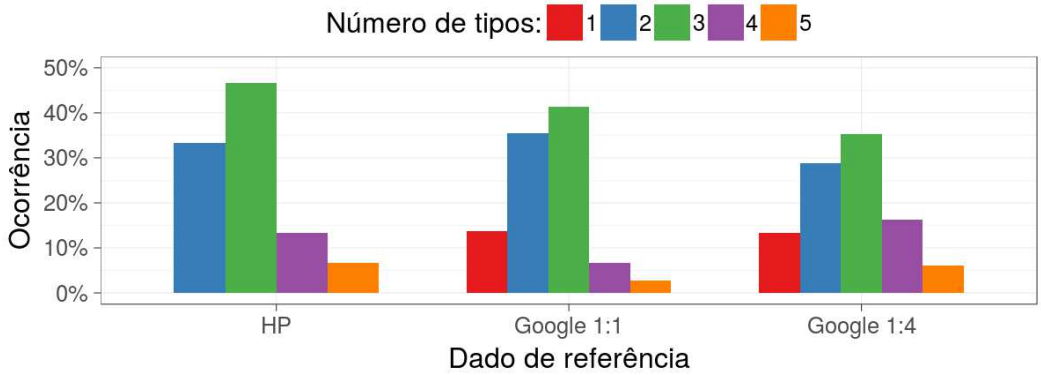


Figura 7.5: Percentual do número de tipos de instância usado por aplicação no provisionamento ótimo.

Considerando o provisionamento ótimo com seleção dinâmica de tipo de todas as aplicações, foi medido o percentual total de intervalos de provisionamento em que um determinado tipo de instância foi selecionado como a opção mais econômica pelo serviço de provisionamento. A Figura 7.6 mostra para cada conjunto de rastros considerado o percentual total de uso de cada um dos tipos de instância considerados. Como pode ser observado, não há um único tipo de instância que é o melhor para todos os casos, considerando diferentes conjuntos de rastros e máquinas de referência. Para os dados da HP, o tipo de instância com maior frequência de seleção (r3.large) é usado em aproximadamente 38% dos intervalos de provisionamento. Por outro lado, o tipo de instância mais promissor (m3.medium), para o conjunto de dados da Google, responde por menos de 35% dos casos, considerando ambas as máquinas de referência. Isso indica que não há um único tipo de instância que destaca-se como o tipo que melhor se adéqua a todas as aplicações.

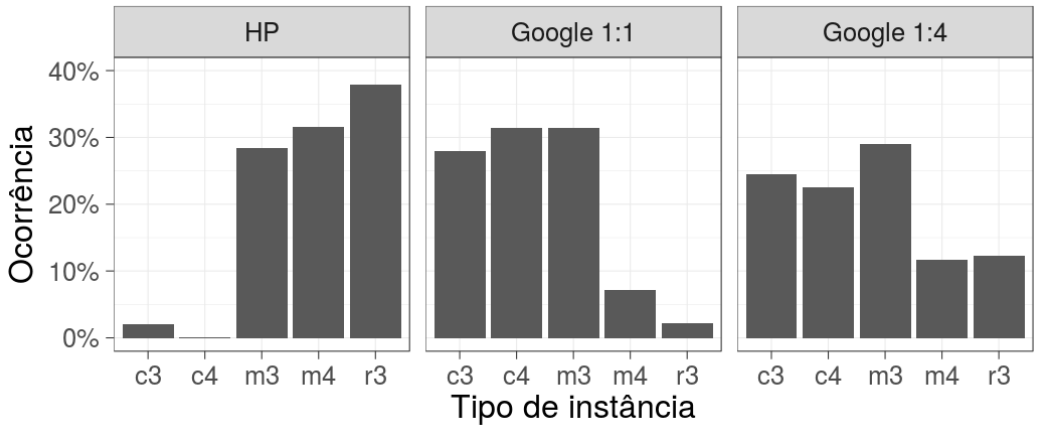


Figura 7.6: Percentual de intervalos de provisionamento em que cada um dos tipos de instância foi selecionado no provisionamento automático ótimo, para diferentes conjuntos de rastos de utilização.

Em resumo, o estudo mostra que em um cenário de provisionamento ótimo é possível uma economia de custo considerável ao adicionar um seletor dinâmico de tipo de instância ao processo de provisionamento. Nas próximas seções, é realizada uma investigação sobre a possibilidade de obtenção de economia de custos em um cenário de provisionamento prático, com estimativas de demanda propensas a erros.

7.4 Seleção Prática de Tipo de Instância de VM

7.4.1 Solução de provisionamento baseada em predição

Nesta seção é realizada a análise de uma solução prática de provisionamento automático que atua proativamente e usa métricas multidimensionais de utilização de CPU e memória. A solução utiliza um modelo de predição de séries temporais baseado em auto-regressão (AR) combinado com a técnica de correção de predições proposta por Morais et al. [44] (vide Seção 6.3.3), para produzir estimativas de demanda de ECU e memória a partir do histórico de utilização de recursos de cada aplicação. Foram consideradas quatro configurações dos modelos de predição baseados em AR, uma sem correção e três que usam porcentagens diferentes de dados históricos: 25%, 50% e 100%. Essas configurações são denominadas "AR", "AR 25", "AR 50"e "AR 100", respectivamente.

O modelo de simulação descrito anteriormente foi adaptado para considerar esse modelo de predição. Para cada predição realizada, o preditor é alimentado com até uma semana de dados históricos (168 intervalos de tempo de 1 hora de duração) e gera uma estimativa de demanda para a próxima hora. Para tal, foram considerados os dois conjuntos de rastros de utilização sumarizados na utilização máxima a cada intervalo de tempo de 1 hora. As estimativas para cada dimensão considerada são usadas para decidir o tipo de instância com melhor relação custo-benefício para executar a aplicação, a partir do número necessário de VMs, de cada tipo, para suportar a demanda da aplicação na próxima hora. Assume-se que o processo de provisionamento proativo é executado em um tempo insignificante se comparado ao tempo de vida da aplicação, portanto, próximo ao final de um intervalo de provisionamento, o serviço pode executar o provisionamento para o próximo intervalo de tempo.

Com base nessa solução prática de provisionamento automático, foi simulado o provisionamento das aplicações dos dois conjuntos de dados, 30 aplicações para o conjunto da HP e 956 para o conjunto da Google, considerando dimensões de ECU e memória e a seleção periódica do tipo de instância de VM a ser usado. Para cada aplicação, foi computado o número de intervalos de 1 hora em que a utilização de pelo menos uma dimensão de recursos atingiu 100%, que corresponde a uma violação de SLO. O custo de provisionamento para cada aplicação foi calculado em termos monetários, com base no preço por hora de uso de

cada tipo de instância usado no provisionamento. Esse custo foi comparado com o custo de provisionamento da solução ótima com seleção dinâmica de tipos de instância, discutido na seção anterior.

A Figura 7.7 apresenta o percentual de variação de custo e o percentual de violações de SLO do provisionamento baseado em AR para essas aplicações. Os custos de provisionamento obtidos pela solução prática sem correção podem ser considerados em média inferiores aos custos obtidos pelo provisionamento ótimo, com base no teste pareado de Wilcoxon com nível de confiança de 95%. Isto se deve aos erros de predição que produzem cenários de sub provisionamento e, consequentemente, reduções de custo de provisionamento e mais violações de SLO. Esse impacto pode ser observado a partir do percentual significativo de violações de SLO obtido, superior em média a 5% e a 15% para o conjunto de aplicações da HP e Google, respectivamente.

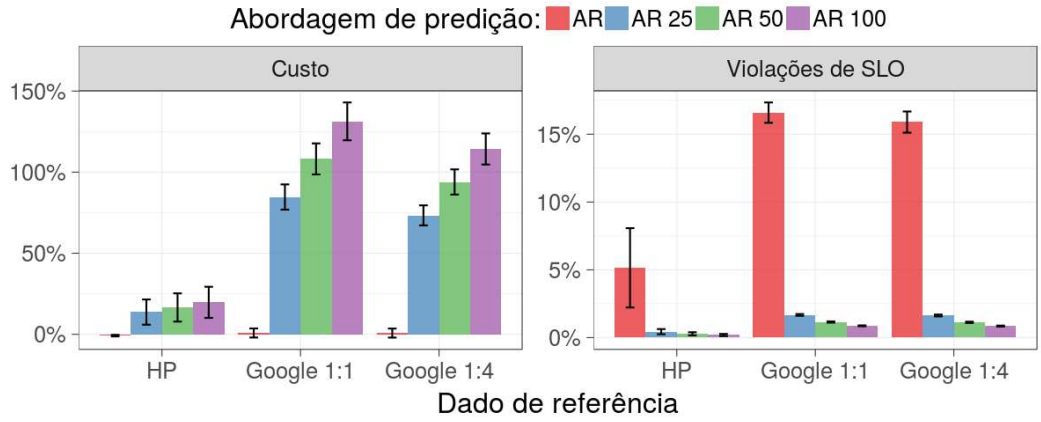


Figura 7.7: O custo de provisionamento da solução baseada em AR em relação ao provisionamento ótimo e a porcentagem de violações de SLO.

As configurações que usam o corretor de predição são mais eficientes em reduzir o percentual de violações de SLO, cuja média, considerando todas as aplicações e conjunto de dados, é de cerca de 1%. No entanto, isso vem com um aumento considerável nos custos de provisionamento, que adicionam em média, em comparação com o seletor de tipo de instância ótimo, de 13% a 20% de nos custos das aplicações da HP e de 73% a 130% nos custos das aplicações da Google. Como esperado, quanto mais conservadora for a configuração do mecanismo de correção, maior será a redução das violações de SLO, mas maior será o custo

de provisionamento associado.

7.4.2 Custo incorrido da redução de violações de SLO

Evidentemente, os custos incorridos pela abordagem de provisionamento ótimo para evitar violações de SLO não são realistas, uma vez que faz uso de informações que não estão disponíveis na prática. Uma abordagem mais realista considerada por provedores de aplicações é o super provisionamento estático, que não tira proveito da elasticidade proporcionada por ambientes de IaaS, uma vez que provisiona estaticamente todas as dimensões de recursos no longo prazo.

Nesse caso, dependendo da variabilidade da demanda ao longo do tempo, a infraestrutura pode ser super provisionada por um período de tempo substancialmente grande, levando a custos de provisionamento desnecessários. Além disso, na prática, o super provisionamento não garante a ausência de violações de SLO, uma vez que se baseia em estimativas de pico de demandas que são propensas a erros. Nesse estudo são considerados cenários onde o provisionamento estático é realizado de forma perfeita, sem erros de estimativa, e cenários nos quais a demanda de pico é superestimada ou subestimada, com estimativas de demanda inferiores à demanda real em 40% e 20% (referenciados como "SP -40"E "SP -20", respectivamente), e superior à demanda real em 20%.

Para cada aplicação, foi calculado o custo de provisionamento resultante da abordagem de provisionamento baseada em AR em relação ao custo das abordagens de super provisionamento estático. Para a abordagem de provisionamento estático, foi considerado, dentre os 5 tipos, o tipo de instância que produz o menor custo de provisionamento para cada aplicação considerada. A Figura 7.8 mostra o diagrama de caixa da comparação de custos de provisionamento, onde os custos negativos significam que o custo da abordagem de provisionamento baseada em AR foi menor do que o custo do provisionamento estático e os custos positivos significam o contrário.

Os resultados mostram que uma solução de provisionamento pode proporcionar reduções substanciais de custo, apesar da desvantagem de apresentar quantidades razoáveis de violações de SLO, como será mostrado em breve. Contudo, as técnicas de correção de predições podem reduzir as taxas de violações de SLO para um nível considerado baixo, com um aumento associado nos custos de provisionamento. A abordagem proposta, baseada em

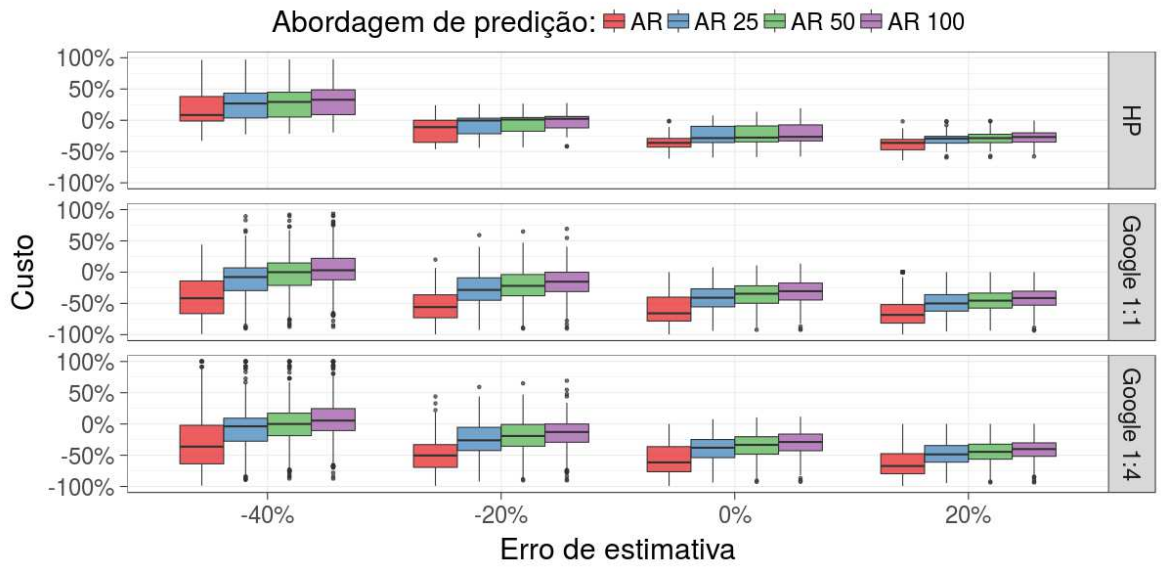


Figura 7.8: O custo de provisionamento de instanciações da solução baseada em AR em relação aos custos dos cenários de provisionamento estático.

AR, gera um custo de infraestrutura que em metade dos casos é 38% inferior ao obtido no provisionamento estático perfeito, para ambos os conjuntos de aplicações. Por outro lado, se comparado ao cenário de provisionamento estático com uma estimativa de demanda máxima 20% maior do que o valor real, a abordagem baseada em AR apresenta em média um custo de provisionamento 49% inferior.

A Figura 7.9 apresenta a comparação da porcentagem de violações de SLO para as abordagens de provisionamento baseadas em AR e os cenários que utilizam o provisionamento estático com sub estimativas de picos de demanda. Quando a estimativa de pico de demanda é 20% menor que a demanda real, a taxa média de violações de SLO é igual a 4,9% para os cenários da seleção dinâmica baseada em AR e de 2,2% para o cenário de provisionamento estático sub estimado em 20% ("SP -20%"). No entanto, o desempenho em termos de custo se inverte entre as abordagens, onde os cenários baseados em AR apresentam um custo em média 26% menor do que o custo do provisionamento estático.

Para erros de estimativa de pico de demanda de -40% ("SP -40"), a relação entre custo e violações de SLO se inverte para o conjunto de dados da HP. Nesse cenário, o custo da abordagem baseada em AR é em média 25% maior do que o apresentado pelo provisiona-

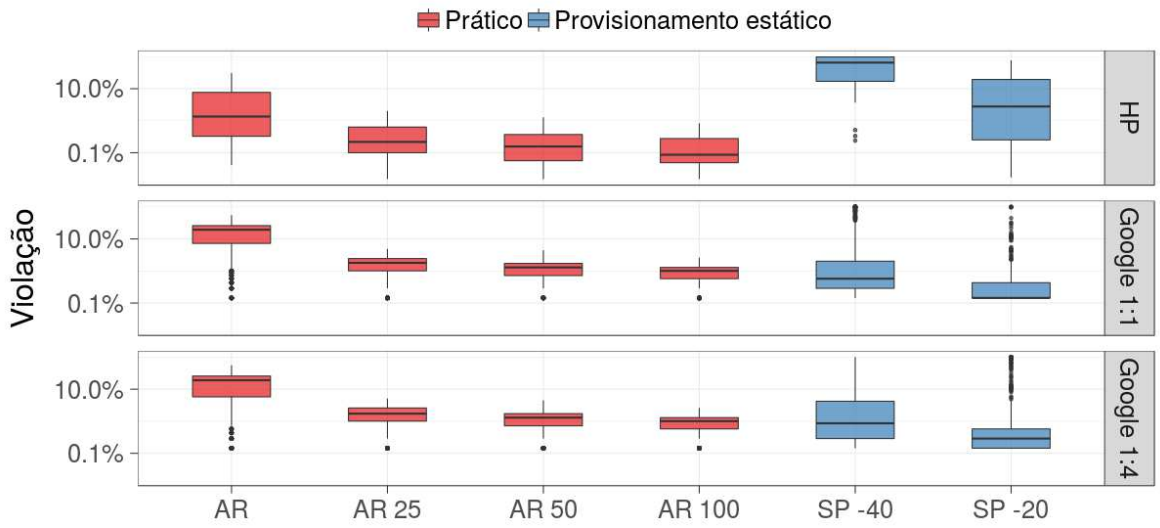


Figura 7.9: Análise de violações de SLO de instanciações da solução baseada em AR e dos cenários de provisionamento estático com subestimativa de pico de demanda.

mento estático, enquanto que o percentual mediano de violações de SLO é de 0,25% para a abordagem prática baseada em AR e de 47,5% para a abordagem de provisionamento estática com sub estimativa é de -40%. Por outro lado, para aplicações da Google, quando a sub estimativa de demanda de pico é de -40%, tanto os custos médios de provisionamento como a porcentagem média de violações de SLO são muito semelhantes entre as abordagens baseadas em AR com correção e de provisionamento estático.

É interessante observar que, principalmente para os dados da Google que em alguns cenários apresentam desempenho semelhante entre abordagem prática e provisionamento estático, as soluções baseadas em AR possuem distribuições de taxas de violação SLO com menor variação e menor ocorrência de valores extremos. Esse comportamento é indesejável para uma solução de provisionamento automático como um serviço, uma vez que impacta a generalidade da solução em provisionar eficientemente aplicações com diferentes perfis de demanda, de forma que quanto mais consistentes forem os resultados para as diferentes aplicações a serem provisionadas, melhor. Assim, para o cenário de seleção dinâmica de tipos de instância, é possível afirmar que a abordagem baseada em AR incorre em um risco menor de altos percentuais de violações de SLO para qualquer aplicação em particular do

que uma abordagem de provisionamento estático prática, propensa a erros de estimativas de demanda.

7.5 Discussão e Conclusões

Nesse capítulo foi proposto um mecanismo de seleção de tipo de instância de VM para um serviço de provisionamento automático de aplicações horizontalmente escaláveis em ambientes de IaaS. O mecanismo de seleção é baseado na análise de métricas de utilização de recursos multidimensionais e opera em conjunto com uma solução de provisionamento que atua sobre a infraestrutura acada de intervalo de tempo com uma hora de duração. Experimentos de simulação foram realizados para avaliar o uso do mecanismo de seleção dinâmica no provisionamento automático com base em rastros de utilização de aplicações reais.

A partir dessas simulações foi demonstrada a importância de considerar diferentes tipos de instância e, consequentemente, métricas multidimensionais durante o provisionamento de aplicações através da análise de desempenho de uma solução de provisionamento automático ótimo e não realista. Evidenciou-se que a seleção de tipo de instância é uma alternativa para reduzir custos de provisionamento (Questão de pesquisa 6) e considerar múltiplas dimensões no provisionamento é essencial para manter a taxa de violações de SLO baixa. Além do mais, a solução de seleção de tipo de instância foi avaliada considerando uma abordagem de provisionamento proativo prática, baseada em um modelo de predição AR combinado com uma técnica de correção de predições.

Apesar do objetivo principal da seleção dinâmica de tipo de instância ser o mesmo buscado pelas soluções de provisionamento automático, que consiste na redução dos custos de provisionamento, existe outro fator que merece atenção no processo de provisionamento: violações de SLO. Na prática, o provisionamento automático provavelmente produz reduções razoáveis de custos de provisionamento com a presença de violações de SLO, devido essencialmente a erros de sub provisionamento. Além disso, abordagens típicas de super provisionamento estático da infraestrutura — alternativas a uma solução de provisionamento automático — conduzem a um elevado custo de provisionamento e também não garantem que violações de SLO sejam evitadas, uma vez que uma estimativa propensa a erros da demanda máxima da aplicação faz-se necessária. Desta forma, violações de SLO são uma

realidade na prática, especialmente para soluções de provisionamento que objetivam reduções do custo de infraestrutura.

Os resultados revelam que quando essa estimativa não é acurada, o provisionamento estático não é eficiente para todas as aplicações. Mesmo quando o desempenho médio da abordagem proativa baseada em AR e do provisionamento estático são semelhantes, em termos de custo e violações de SLO, a variação da taxa de violações é substancialmente maior para a abordagem de provisionamento estático quando considerado o maior conjunto de dados. Além do mais, erros de sub estimativa geram impactos distintos em termos de violações de SLO para os dois conjuntos de dados considerados. Para aplicações da HP, existe uma elevação substancial no percentual de violações de SLO em virtude do aumento do erro de sub estimativas do pico de demanda das aplicações, enquanto que esse impacto de violações é menos evidente para o conjunto de dados da Google. De toda forma, os resultados da abordagem prática proativa mostram que uma limitação da taxa de violações de SLO da aplicação restringe uma redução de custo viável. Todavia, isso não é uma consequência da seleção dinâmica, mas de um modelo de predição impreciso.

Além do mais, quando ambos, custos e QoS, são considerados, incluir o mecanismo de seleção de tipo de instância como parte da abordagem proativa de provisionamento automático aprimora os resultados, principalmente em termos de redução de custos de provisionamento, uma vez que a técnica de seleção não degrada a QoS da aplicação provisionada (Questão de pesquisa 6). As taxas de violação de SLO são tão boas (ou tão ruins) quanto seriam sem o uso da seleção dinâmica de tipos de instância de VM, enquanto que a eficiência da política de provisionamento é melhorada em termos de custo pela inclusão do seletor de tipo de instância. Desta forma, o desempenho do serviço de provisionamento em termos de custo de provisionamento pode ser aprimorado pelo uso do mecanismo de seleção de dinâmica de tipos de instância de VM no provisionamento automático de aplicações.

No entanto, este desempenho pode ser impactado pelo custo de mudança do tipo de instância de VM utilizada durante o provisionamento, que não foi considerado nesse estudo devido a simplificações sobre a periodicidade de provisionamento ocorrer em sincronia com o ciclo de tarifação do provedor. Em cenários em que isso não ocorre é possível que existam períodos em que uma nova alocação de tipo de instância ocorra enquanto ainda há recursos alocados de um tipo diferente, não considerados no planejamento de capacidade. Essa aloca-

ção redundante de recursos proporciona, além do super provisionamento, um custo adicional que pode afetar o desempenho final da solução de provisionamento com seleção dinâmica de tipos de instância. Desta forma, evoluções desse estudo sobre soluções de provisionamento automático baseadas na seleção dinâmica de tipos de instância devem considerar tais limitações.

Além disso, é importante ressaltar que independente da abordagem de provisionamento utilizada ou técnica de provisionamento considerada, métricas de uso de recursos devem ser consideradas para incluir a seleção dinâmica de tipos de instância em uma solução de provisionamento automático existente. Entender como os recursos alocados são consumidos é fundamental para escolher o tipo de instância mais econômico a ser usado. Desta forma, soluções intrusivas baseadas apenas em métricas específicas da aplicação também deveriam observar ou estimar métricas de uso de recursos (CPU, memória RAM, disco, etc.) para executar uma seleção de tipos de instância no processo de provisionamento. Soluções de provisionamento, aplicadas a um cenário de provisionamento como um serviço de IaaS, que já reúnem informações relacionadas ao uso de recursos são mais adequadas para incluir um seletor de tipo de instância, uma vez que estas já monitoram as informações necessárias.

Capítulo 8

Considerações Finais

Nesse capítulo, são realizadas as considerações finais desse trabalho por meio de uma discussão sobre os resultados obtidos, descrevendo as conclusões decorrentes destes e destacando possíveis trabalhos futuros.

8.1 Discussão e Conclusões

Esse trabalho de tese abordou a problemática de construção de um serviço de provisionamento automático em ambientes de IaaS, como definido no Capítulo 2. Nessa concepção de serviço, técnicas de provisionamento automático podem ser utilizadas pra explorar a elasticidade proporcionada por infraestruturas virtuais de Computação na Nuvem na execução de aplicações horizontalmente escaláveis. Para isso, tais técnicas devem ser capazes de operar com base em informações não específicas da aplicação, que consistem em métricas não intrusivas de utilização de recursos computacionais, obtidas no nível da infraestrutura virtual e em geral disponibilizadas pelo provedor de IaaS e Computação na Nuvem.

A não intrusividade da técnica é essencial para que um serviço de provisionamento automático proposto seja potencialmente desacoplado da aplicação provisionada e genérico em termos de características e do tipo dessa aplicação. Além disso, o nível intrusão da técnica deve ser tal que assegure que informações sensíveis, na perspectiva do responsável da aplicação, não necessitem ser compartilhadas com o provedor do serviço de provisionamento automático. Desta forma, questões de privacidade e confidencialidade das informações são respeitadas. Adicionalmente, as métricas devem considerar o consumo de recursos em múl-

tiplas dimensões para que o planejamento de capacidade da infraestrutura seja realizado em observância à QoS da aplicação, evitando a degradação do desempenho da aplicação devido à insuficiência de recursos em uma determinada dimensão, além de possibilitar uma otimização do uso de recursos da infraestrutura.

O trabalho realiza um estudo aprofundado sobre o provisionamento automático não intrusivo considerando as duas principais técnicas de provisionamento automático atualmente em discussão na literatura e em uso no mercado de Computação na Nuvem, como destacado na revisão da literatura realizada no Capítulo 3, que baseiam-se em abordagens com modos de operação reativo e proativo. Conforme metodologia descrita no Capítulo 4, esses estudos de avaliação das abordagens de provisionamento em um cenário de provisionamento como um serviço foram realizados a partir de experimentos de simulação com base em dados representativos de utilização de recursos de aplicações reais, descritos nos Capítulos 5 e 6. Essas avaliações consideram desde o desempenho das técnicas em atingir os objetivos de provisionamento, relacionados à QoS da aplicação e ao custo de provisionamento, até como essas técnicas são eficientes ou limitadas em compor um serviço nesses moldes.

Os resultados levam à conclusão de que a abordagem reativa, a mais difundida na área de Computação na Nuvem possivelmente devido ao seu caráter intuitivo e sua simplicidade de implementação e implantação, não é eficiente para compor um serviço de provisionamento automático em ambientes de IaaS, para o conjunto de aplicações consideradas. Apesar da configuração de limiares de provisionamento possibilitar ao usuário do serviço, com mesma finalidade, a capacidade de controlar o *trade-off* entre a priorização da minimização da ocorrência de violações de SLO e do custo de provisionamento, a eficiência da abordagem é limitada em outros aspectos fundamentais a um serviço de provisionamento.

Primeiramente, a abordagem reativa não apresenta desempenho satisfatório em garantir que os objetivos mínimos de provisionamento sejam atendidos para um percentual representativo das aplicações, mesmo considerando métricas individuais de provisionamento, e particularmente quando o limite máximo de violações de SLO é mais restritivo. Além disso, a mesma mostra-se ineficiente em termos de configuração no que diz respeito a não existência de um conjunto restrito de configurações que possa ser usado como padrão do serviço para as aplicações consideradas, ou até mesmo para o provisionamento de uma mesma aplicação. Isso ocorre principalmente pela forte relação entre as características da carga de trabalho das

aplicações e a configurações necessárias para o provisionamento das mesmas.

O processo de configuração da técnica torna-se ainda mais complexo pelo fato de que a configuração de uma regra de provisionamento reativa normalmente é composta, além do limiar de provisionamento, também do número de VMs de um determinado tipo que devem ser alocadas ou desalocadas na ação de provisionamento. Em muitos casos foi observado que para um provisionamento perfeito, o mesmo limiar levava ora a adição de VMs, ora à remoção, ainda com número de VMs diferentes a serem adicionadas/removidas. A dificuldade de configuração é potencializada ao considerar múltiplas dimensões de recursos no provisionamento das aplicações. Nesse cenário, a configuração da técnica de provisionamento deve ser realizada para diferentes dimensões, gerando um incremento substancial de complexidade de configuração, mesmo que para configurações de super provisionamento estático.

Desta forma, considera-se que o uso de soluções reativas baseadas em limiares estáticos não são adequadas para compor um serviço abrangente de provisionamento de aplicações horizontalmente escaláveis em ambientes de IaaS. Todavia, acredita-se que o uso de uma configuração dinâmica, que se adapte aos perfis de consumo de recursos da carga de trabalho da aplicação, é mais indicado para este fim. No geral, o uso da técnica reativa deve estar associado a mecanismos mais sofisticados que sejam capazes de realizar a gerência de configuração do serviço de provisionamento de forma automática.

Em contrapartida, abordagens de provisionamento proativo mostram-se mais adequadas para fundamentar um serviço de provisionamento automático em IaaS. Para esse tipo de abordagem é regra que a complexidade de configuração do serviço seja transferida para o modelo preditivo usado pela técnica. Por esse motivo, soluções proativas apresentam uma eficiência de configuração mais significativa, se comparada a abordagens reativas, devido a redução do espaço de configuração necessário para operar no provisionamento de um conjunto distinto de aplicações com base em diferentes métricas de utilização de recursos. Além disso, o controle dos objetivos de provisionamento pode ser obtido pelo uso de mecanismos adicionais de ajuste do conservadorismo das predições consideradas, ao ônus de configuração de pelo menos um parâmetro adicional.

Além da eficiência de configuração e controle de objetivos, a técnica mostra-se capaz de assegurar os objetivos de provisionamento para uma parcela representativa das aplicações, mesmo considerando diferentes métricas de utilização de recursos e classes de limites de

violações de SLO minimamente flexíveis. No entanto, é possível que o uso de modelos preditivos mais sofisticados torne o serviço eficiente em provisionar uma gama maior de aplicações, apesar dos modelos considerados já apresentarem resultados satisfatórios. Tendo em vista tais constatações, conclui-se sobre a eficiência da abordagem proativa em basear um serviço de provisionamento em IaaS.

Adicionalmente, o objetivo de redução de custos e otimização do uso da infraestrutura pode ser complementarmente atingido pelo uso de diferentes tipos de instância com o melhor custo-benefício, em termos da proporção entre recursos requeridos pela aplicação, a cada intervalo de provisionamento. Isso foi verificado em um estudo realizado no Capítulo 7, que propõe um mecanismo de seleção dinâmica de tipos de instância de VM associado à solução de provisionamento automático e proativo, que avalia periodicamente o tipo de instância mais rentável dentre os disponíveis para executar a aplicação.

Os benefícios do uso desse mecanismo, em termos de custo de execução, foram verificados para uma parcela majoritária das aplicações consideradas, para os dois conjuntos de dados (HP e Google). Além de colaborar com a minimização de custos de provisionamento para aplicações com variação temporal da proporcionalidade do uso de recursos, a seleção dinâmica também permite que configurações padrão equivocadas sobre o tipo de instância do provisionamento sejam automaticamente revisadas, mesmo quando não há variação entre a proporção de recursos consumidos. Desta forma, o mecanismo pode ser facilmente agregado a um serviço de provisionamento automático em IaaS.

Assim, a tese proposta sobre a viabilidade de construção de um serviço de provisionamento automático e não intrusivo para diferentes aplicações horizontalmente escaláveis, pôde ser verificada. Principalmente para as técnicas de provisionamento proativo associadas a mecanismos de controle de violações de SLO, que controlam o conservadorismo da solução proativa e, por consequência, os objetivos de provisionamento buscados. Além do mais, tal serviço pode ser aprimorado pelo uso associado de mecanismos de seleção dinâmica de tipos de instância de VMs, que promovem potenciais reduções nos custos de provisionamento e permitem que configurações iniciais equivocadas do tipo de instância a ser usado no provisionamento sejam naturalmente reavaliadas.

8.2 Ameaças à Validade

Nessa seção são identificadas ameaças à validade dos estudos realizados sobre o desempenho de um serviço de provisionamento automático baseado em abordagens de provisionamento, reativas e proativas, a partir de métricas não intrusivas em um ambiente de IaaS.

Validade de construção Esse tipo de ameaça à validade define a capacidade que um estudo realizado tem em expressar que o que foi medido era o que de fato se pretendia medir. Nesse sentido, uma ameaça à validade de construção a ser destacada é a capacidade do estudo em medir o impacto que violações de SLO observadas no provisionamento apresentam sobre o desempenho da aplicação provisionada. Nos estudos realizados, violações de SLO com intensidades diferentes em termos de demanda são contabilizadas de forma equivalente na medição de desempenho do serviço de provisionamento em termos de violações de SLO. No entanto, a intensidade das violações de SLO pode gerar impactos diferentes na QoS da aplicação provisionada. Além disso, em avaliações baseadas em métricas individuais os níveis de utilização do recurso considerados no SLO podem não representar um impacto real na QoS da aplicação. Por exemplo, espera-se que níveis de utilização de memória próximos de 100% associados a taxas não significativas de escrita e leitura em disco não tenham impacto real no desempenho da aplicação em execução. Essa ameaça não tem implicações significativas no estudo quantitativo sobre o desempenho de serviços de provisionamento em termos de violações de SLO, mas sim em uma possível análise qualitativa sobre como essas violações impactam o desempenho real das aplicações provisionadas.

Validade interna Representa a possibilidade de se estabelecer uma conclusão causal com base no estudo realizado. No contexto desse trabalho, verifica-se uma relação de causa e efeito entre a configuração das técnicas de provisionamento e o desempenho dessas técnicas em um cenário de provisionamento como um serviço, principalmente em termos de QoS da aplicação provisionada e custo de provisionamento. No entanto, outros fatores podem influenciar esse desempenho, como a disponibilidade dos recursos adquiridos do provedor de IaaS, variação no desempenho da infraestrutura virtual alocada à aplicação ou falhas da aplicação não relacionadas à demanda de recursos alocada. Todavia, acredita-se que influências outras, por mais impactantes que sejam sobre a relação entre configuração e desempenho da

abordagem de provisionamento, não são capazes de anular a causalidade entre estas variáveis, devido a variabilidade de desempenho observada a partir da varredura de parâmetros de configuração realizada.

Validade externa Expressa a generalidade dos resultados obtidos para outros contextos. Os estudos realizados apesar de dependentes de características do serviço de IaaS considerado, como a duração do ciclo de tarifação de recursos ou os tipos de instância de VM disponibilizados, apresentam resultados que são considerados extensíveis para outros cenários de estudo de provisionamento como um serviço. Isso deve-se ao fato dessa ser uma ameaça cujo impacto pode ser amenizado ao considerar-se como base de estudo o modelo de IaaS mais utilizado no contexto de Computação na Nuvem, baseado no serviço oferecido pela Amazon AWS. Além disso, a diversidade de aplicações, perfis de carga de trabalho e métricas de provisionamento consideradas nos estudos possibilitam que os resultados obtidos sejam generalizados para outros contextos de provisionamento automático e não intrusivo. Adicionalmente, os estudos consideram métricas gerais de avaliação das técnicas de provisionamento automático que podem ser estendidas para estudos que consideram outros ambientes de IaaS e abordagens de provisionamento não intrusivo.

8.3 Trabalhos Futuros

A partir dos resultados obtidos nesse trabalho observa-se possíveis pontos de evolução e aprimoramento da área de pesquisa abordada por essa tese de doutorado. Desta forma, as seguintes atividades de pesquisa são sugeridas como possíveis trabalhos futuros:

- Avaliar o estudo de abordagens de provisionamento reativo a partir de mecanismos de configuração dinâmica de regras de provisionamento, que busquem se adaptar às variabilidades presentes na carga de trabalho da aplicação em diferentes dimensões de recursos. Como observado nesse trabalho, a configuração estática da abordagem reativa não é eficiente em diferentes aspectos e mecanismos dinâmicos podem ser uma solução de aprimoramento da técnica;

- Evoluir o estudo do provisionamento proativo como um serviço de IaaS considerando soluções proativas mais sofisticadas e diferentes mecanismos de aprimoramento da

acurácia de predição de demandas futuras da aplicação provisionada. Apesar dos resultados promissores, é possível que a combinação de técnicas empregadas em diferentes cenários de provisionamento sejam eficientes em refinar o desempenho da solução proativa, principalmente em termos de satisfação dos objetivos de provisionamento;

- Dar continuidade ao estudo iniciado nesse trabalho sobre a seleção dinâmica de tipos de instância de VM no provisionamento automático e proativo como um serviço de IaaS. Esse estudo apresenta limitações quanto ao custo de mudança do tipo de instância de VM utilizada durante o provisionamento, que consiste essencialmente nos períodos de tempo em que a infraestrutura de execução da aplicação é formada por um conjunto heterogêneo de VMs, onde a alocação redundante de recursos implica em elevações no custo total de provisionamento da aplicação;

- Expandir a abordagem do serviço de provisionamento automático para considerar diferentes métricas de provisionamento, como taxa de leitura e escrita em disco e uso da largura de banda da rede da infraestrutura virtual. Apesar de métricas de utilização de CPU e memória serem representativas para o estudo de provisionamento automático, considerar outras métricas pode aprimorar o desempenho do planejador de capacidade do serviço de provisionamento, de tal forma que a QoS da aplicação possa ser assegurada de forma mais holística e para aplicações com diferentes características de demanda (por exemplo, aplicações de processamento de fluxo de dados, que em geral fazem uso intensivo da infraestrutura de rede);

- Aprimorar o estudo sobre o provisionamento automático e não intrusivo de aplicações horizontalmente escaláveis a partir do impacto proporcional de violações de SLO na QoS da aplicação. Em um cenário simulado é possível medir como as violações de SLO ocorrem em termos de intensidade e tipo de métrica de recurso da violação. Aplicações podem ter diferentes curvas de escalabilidade e, por consequência, podem sofrer impactos diferentes sobre a sua QoS em função de violações de SLO durante o provisionamento. Esse estudo pode permitir que soluções de provisionamento automático não intrusivas sejam avaliadas com base em pesos sobre erros de subestimativa, e consequentes violações de SLO, para diferentes aplicações e métricas de utilização de recursos consideradas;

- Avaliar o serviço de provisionamento automático, proativo e não intrusivo através de experimentos de medição em um ambiente de IaaS (por exemplo, OpenStack [19]). Desta forma, as soluções e técnicas de provisionamento abordadas podem ser experimentadas na prática e o serviço proposto ser validado e prototipado. Assim, outros aspectos não considerados em um cenário de provisionamento simulado podem ser contemplados, como falhas e ausência de recursos virtuais, e o seu impacto sobre o desempenho do serviço de provisionamento proposto pode ser avaliado.

Bibliografia

[1] D. Agrawal, A. Abbadi, S. Das, and A.J. Elmore. Database scalability, elasticity, and autonomy in the cloud. In *Database Systems for Advanced Applications*, volume 6587 of *Lecture Notes in Computer Science*, pages 2–15. Springer Berlin Heidelberg, 2011.

[2] Ahmed Ali-Eldin, Johan Tordsson, and Erik Elmroth. An adaptive hybrid elasticity controller for cloud infrastructures. In *Network Operations and Management (NOMS), 2012 IEEE Symposium on*, pages 204–212. IEEE, 2012.

[3] Amazon. Amazon auto scaling. `http://aws.amazon.com/autoscaling/`. Online; Nov., 2016.

[4] Amazon. Amazon ec2 faqs. `https://aws.amazon.com/ec2/faqs/`. Online; Nov., 2016.

[5] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *ACM Communications*, 53(4):50–58, Apr. 2010.

[6] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, et al. Above the clouds: A berkeley view of cloud computing. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28*, 2009.

[7] Microsoft Azure. Autoscale a cloud service. `https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-how-to-scale`. Online; Nov., 2016.

[8] Marta Beltrán. Automatic provisioning of multi-tier applications in cloud computing environments. *The Journal of Supercomputing*, 71(6):2221–2250, 2015.

[9] N. Bonvin, T.G. Papaioannou, and K. Aberer. Autonomic sla-driven provisioning for cloud applications. In *Cluster, Cloud and Grid Computing, 11th IEEE/ACM International Symposium on*, CCGRID '11, pages 434 –443, Newport Beach, CA, USA, May. 2011.

[10] R. Buyya, Chee Shin Yeo, and S. Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *High Performance Computing and Communications, 10th IEEE International Conference on*, HPCC '08, pages 5 –13, Dalian, China, Sep. 2008.

[11] N.M. Calcavecchia, B.A. Caprarescu, E. Di Nitto, D.J. Dubois, and D. Petcu. Depas: a decentralized probabilistic algorithm for auto-scaling. *Computing*, 94:701–730, 2012.

[12] R.N. Calheiros, C. Vecchiola, D. Karunamoorthy, and R. Buyya. The aneka platform and qos-driven resource provisioning for elastic applications on hybrid clouds. *Future Generation Computer Systems*, 28(6):861 – 870, 2012.

[13] Rodrigo N Calheiros, Enayat Masoumi, Rajiv Ranjan, and Rajkumar Buyya. Workload prediction using arima model and its impact on cloud applications' qos. *IEEE Transactions on Cloud Computing*, 3(4):449–458, 2015.

[14] Eddy Caron, Frédéric Desprez, and Adrian Muresan. Pattern matching based forecast of non-periodic repetitive behavior for cloud clients. *Journal of Grid Computing*, 9(1):49–64, 2011.

[15] Emiliano Casalicchio and Luca Silvestri. Mechanisms for sla provisioning in cloud-based service providers. *Computer Networks*, 57(3):795–810, 2013.

[16] S. Casolari and M. Colajanni. Short-term prediction models for server management in internet-based contexts. *Decision Support Systems*, 48(1):212 – 223, 2009.

[17] John Chambers. The R project for statistical computing. `http://www.r-project.org/`. Online; Nov., 2015.

[18] HP Development Company. httperf homepage. `http://www.hpl.hp.com/research/linux/httperf/`. Online; Nov., 2016.

[19] Rackspace Cloud Computing. Openstack open source cloud computing software. `https://www.openstack.org/`. Online; Nov., 2016.

[20] Emanuel Ferreira Coutinho, Flávio Rubens de Carvalho Sousa, Paulo Antonio Leal Rego, Danielo Gonçalves Gomes, and José Neuman de Souza. Elasticity in cloud computing: a survey. *annals of telecommunications-annales des télécommunications*, pages 1–21, 2015.

[21] Rodrigo da Rosa Righi, Vinicius Facco Rodrigues, Gustavo Rostirolla, Cristiano André da Costa, Eduardo Roloff, and Philippe Olivier Alexandre Navaux. A lightweight plug-and-play elasticity service for self-organizing resource provisioning on parallel applications. *Future Generation Computer Systems*, 2017.

[22] W. Dawoud, I. Takouna, and C. Meinel. Elastic vm for cloud resources provisioning optimization. In *Advances in Computing and Communications*, volume 190 of *Communications in Computer and Information Science*, pages 431–445. Springer Berlin Heidelberg, 2011.

[23] J.O. Fito, I. Goiri, and J. Guitart. Sla-driven elastic cloud hosting provider. In *Parallel, Distributed and Network-Based Processing, 18th Euromicro International Conference on*, PDP '10, pages 111 –118, Pisa, Italy, Feb. 2010.

[24] G. Galante and L.C.E. de Bona. A survey on cloud computing elasticity. In *Utility and Cloud Computing, 5th IEEE/ACM International Conference on*, UCC '12, pages 263 –270, Chicago, IL, USA, Nov. 2012.

[25] Anshul Gandhi, Parijat Dube, Alexei Karve, Andrzej Kochut, and Li Zhang. Adaptive, model-driven autoscaling for cloud applications. In *Automation and Computing, 20th IEEE International Conference on*, volume 14 of *ICAC '14*, pages 57–64, 2014.

[26] Gartner. Forecast alert: It spending, worldwide, 4q16 update. `https://www.gartner.com/doc/3140436`. Online; Jan., 2017.

[27] Hamoun Ghanbari, Bradley Simmons, Marin Litoiu, and Gabriel Iszlai. Exploring alternative approaches to implement an elasticity policy. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 716–723. IEEE, 2011.

[28] Daniel Gmach, Jerry Rolia, and Ludmila Cherkasova. Selling t-shirts and time shares in the cloud. In *Cluster, Cloud and Grid Computing, 12th IEEE/ACM International Symposium on*, CCGRID'12, pages 539–546. IEEE Computer Society, 2012.

[29] Z. Gong, X. Gu, and J. Wilkes. Press: Predictive elastic resource scaling for cloud systems. In *Network and Service Management, 6th International Conference on*, CNSM '10, pages 9 –16, Ontario, Canada, Oct. 2010.

[30] Google. Google cloud autoscaler. `https://cloud.google.com/compute/docs/autoscaler/`. Online; Nov., 2016.

[31] J.D. Hamilton. *Time series analysis*, volume 2. Cambridge Univ Press, 1994.

[32] J. Hellerstein, S. Parekh, Y. Diao, and D.M. Tilbury. *Feedback control of computing systems*. Wiley-IEEE Press, 2004.

[33] Y. Jiang, C. Perng, T. Li, and R. Chang. Self-adaptive cloud capacity planning. In *Services Computing, 9th IEEE International Conference on*, SCC '12, pages 73 –80, Honolulu, HI, USA, Jun. 2012.

[34] F. Leymann and D. Fritsch. Cloud computing: The next revolution in it. *The 52th Photogrammetric Week, Proceedings of*, pages 3–12, 2009.

[35] H.C. Lim, S. Babu, J.S. Chase, and S.S. Parekh. Automated control in cloud computing: challenges and opportunities. In *Automated control for datacenters and clouds, 1st Workshop on*, ACDC '09, pages 13–18, Barcelona, Spain, Jun. 2009.

[36] Joao Loff and Joao Garcia. Vadara: Predictive elasticity for cloud applications. In *Cloud Computing Technology and Science, 6th IEEE International Conference on*, CloudCom '14, pages 541–546. IEEE, 2014.

[37] Tania Lorido-Botrán, José Miguel-Alonso, and Jose Antonio Lozano. Auto-scaling techniques for elastic applications in cloud environments. *Department of Computer*

Architecture and Technology, University of Basque Country, Tech. Rep. EHU-KAT-IK-09, 12:2012, 2012.

[38] Tania Lorido-Botrán, José Miguel-Alonso, and Jose Antonio Lozano. A review of auto-scaling techniques for elastic applications in cloud environments. *Journal of Grid Computing*, 12(4):559–592, 2014.

[39] Ming Mao and Marty Humphrey. A performance study on the vm startup time in the cloud. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 423–430. IEEE, 2012.

[40] P. Marshall, K. Keahey, and T. Freeman. Elastic site: Using clouds to elastically extend site resources. In *Cluster, Cloud and Grid Computing, 10th IEEE/ACM International Conference on*, CCGRID '10, pages 43–52, Melbourne, Victoria, Australia, May. 2010.

[41] Peter Mell and Timothy Grance. The nist definition of cloud computing. Technical report, The National Istitute of Standarts and Technology, 2011.

[42] S. Meng, L. Liu, and V. Soundararajan. Tide: achieving self-scaling in virtualized datacenter management middleware. In *Industrial track, 11th International Middleware Conference on*, Middleware '10, pages 17–22, Bangalore, India, Nov.-Dec. 2010.

[43] Cary Millsap. Thinking clearly about performance. *Queue*, 8(9):10, 2010.

[44] F. Morais, F. Brasileiro, R. Lopes, R. Araújo, W. Satterfield, and L. Rosa. Autoflex: Service agnostic auto-scaling framework for iaas deployment models. In *Cluster, Cloud and Grid Computing, 13th IEEE/ACM International Symposium on*, CCGRID '13, Delft, Netherlands, May. 2013.

[45] F. Morais, R. Lopes, and F. Brasileiro. Instance type selection in proactive horizontal auto-scaling. In *Cloud Computing Technology and Science, 8th IEEE International Conference on*, CloudCom '16, Luxembourg, Dec. 2016.

[46] F. Morais, R. Lopes, and F. Brasileiro. Provisionamento automático de recursos em nuvens iaas: eficiência e limitações de abordagens reativas. In *Anais do XXXV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2017)*. Sociedade Brasileira de Computação (SBC), 2017.

[47] Saurav Nanda, Thomas J Hacker, and Yung-Hsiang Lu. Predictive model for dynamically provisioning resources in multi-tier web applications. In *Cloud Computing Technology and Science, IEEE International Conference on*, CloudCom '16, pages 326–335. IEEE, 2016.

[48] Marco AS Netto, Carlos Cardonha, Renato LF Cunha, and Marcos D Assunçao. Evaluating auto-scaling strategies for cloud computing environments. In *Modelling, Analysis & Simulation of Computer and Telecommunication Systems, 22nd IEEE International Symposium on*, MASCOTS '14, pages 187–196. IEEE, 2014.

[49] Hiep Nguyen, Zhiming Shen, Xiaohui Gu, Sethuraman Subbiah, and John Wilkes. Agile: Elastic distributed resource scaling for infrastructure-as-a-service. In *Automation and Computing, 19th IEEE International Conference on*, ICAC '13, pages 69–82, 2013.

[50] OpenStack. Openstack icehouse. `http://www.openstack.org/software/icehouse/`. Online; Nov., 2016.

[51] Rackspace. Rackspace cloud auto scale. `http://www.rackspace.com/cloud/auto-scale`. Online; Nov., 2016.

[52] N. Roy, A. Dubey, and A. Gokhale. Efficient autoscaling in the cloud using predictive models for workload forecasting. In *Cloud Computing, 4th IEEE International Conference on*, CLOUD '11, pages 500 –507, Washington DC, USA, Jul. 2011.

[53] Amazon Web Services. Amazon cloudwatch documentation. `http://aws.amazon.com/pt/documentation/cloudwatch/`. Online; Nov., 2015.

[54] Amazon Web Services. Amazon elastic compute cloud api reference. `http://docs.amazonwebservices.com/AWSEC2/latest/APIReference/Welcome.html`. Online; Nov., 2016.

[55] Y. Seung, T. Lam, L.E. Li, and T. Woo. Cloudflex: Seamless scaling of enterprise applications into the cloud. In *Computer Communications, 30th IEEE International Conference on*, INFOCON '11, pages 211 –215, Shanghai, China, Apr. 2011.

[56] U. Sharma, P. Shenoy, S. Sahu, and A. Shaikh. A cost-aware elasticity provisioning system for the cloud. In *Distributed Computing Systems, 31st International Conference on*, ICDCS '11, pages 559 –570, Minneapolis, MN, USA, Jun. 2011.

[57] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Cloud Computing, 2nd ACM Symposium on*, SOCC '11, pages 5:1–5:14, Cascais, Portugal, Oct. 2011.

[58] Simon Spinner, Nikolas Herbst, Samuel Kounev, Xiaoyun Zhu, Lei Lu, Mustafa Uysal, and Rean Griffith. Proactive memory scaling of virtualized applications. In *Cloud Computing, 8th IEEE International Conference on*, CLOUD '15, pages 277–284. IEEE, 2015.

[59] I. Sriram and A. Khajeh-Hosseini. Research agenda in cloud technologies. *Computing Research Repository*, abs/1001.3259, 2010.

[60] K. Stanoevska-Slabeva and T. Wozniak. Cloud basics - an introduction to cloud computing. In *Grid and Cloud Computing*, pages 47–61. Springer Berlin Heidelberg, 2010.

[61] B. Urgaonkar, P. Shenoy, A. Chandra, P. Goyal, and T. Wood. Agile dynamic provisioning of multi-tier internet applications. *ACM Trans. on Autonomous and Adaptive Systems*, 3(1):1:1–1:39, Mar. 2008.

[62] N. Vasić, D. Novaković, S. Miučin, D. Kostić, and R. Bianchini. Dejavu: accelerating resource allocation in virtualized environments. In *Architectural Support for Programming Languages and Operating Systems, 17th International Conference on*, ASPLOS '12, pages 423–436, London, England, UK, Mar. 2012.

[63] Manish Verma, GR Gangadharan, Nanjangud C Narendra, Ravi Vadlamani, Vidyadhar Inamdar, Lakshmi Ramachandran, Rodrigo N Calheiros, and Rajkumar Buyya. Dynamic resource demand prediction and allocation in multi-tenant service clouds. *Concurrency and Computation: Practice and Experience*, 2016.

[64] S. Vijayakumar, Q. Zhu, and G. Agrawal. Dynamic resource provisioning for data streaming applications in a cloud environment. In *Cloud Computing Technology and*

Science, 2nd IEEE International Conference on, CloudCom '10, pages 441 –448, Indianapolis, IN, USA, Dec. 2010.

[65] John Wilkes and Charles Reiss. Google clusterdata 2011. `https://github.com/google/cluster-data`. Online; Nov., 2016.

[66] Jingqi Yang, Chuanchang Liu, Yanlei Shang, Bo Cheng, Zexiang Mao, Chunhong Liu, Lisha Niu, and Junliang Chen. A cost-aware auto-scaling approach using the workload prediction in service clouds. *Information Systems Frontiers*, 16(1):7–18, 2014.

[67] Lenar Yazdanov and Christof Fetzer. Vscaler: Autonomic virtual machine scaling. In *Cloud Computing, 6th IEEE International Conference on*, CLOUD '13, pages 212–219. IEEE, 2013.

Apêndice A

Tempo de Responsividade do Provisionamento Horizontal

As ações de provisionamento realizadas para dinamicamente gerenciar a capacidade da infraestrutura de execução de uma aplicação horizontalmente escalável, independente da técnica de provisionamento adotada (reativa ou proativa), devem ser capazes de lidar com tempo intrínseco ao processo de provisionamento horizontal. Nesse formato de provisionamento a capacidade da infraestrutura de execução é dinamicamente ajustada pela adição ou remoção de VMs dessa infraestrutura. Assim, é evidente que esse processo de provisionamento requer um tempo para ser executado e efetivado, e por esse motivo as soluções de provisionamento horizontal devem ter em conta esse tempo ao decidir sobre as ações de provisionamento a serem realizadas.

A responsividade das ações de provisionamento horizontal são regidas por diferentes elementos a depender da ação de provisionamento realizada, adição ou remoção de VMs. O tempo necessário para a alocação efetiva de um nova instância na infraestrutura é limitado principalmente pela soma do tempo de inicialização da VM adicionada e de preparação da aplicação. Além do mais, tem-se o tempo adicional para o balanceamento da carga da aplicação, que consiste no tempo para a redistribuição da carga de trabalho após o provisionamento da infraestrutura de execução. Por outro lado, para o cenário de desalocação de recursos o tempo de provisionamento resume-se apenas ao tempo de balanceamento da carga de trabalho, uma vez que o tempo de remoção de uma VM é tido como insignificante. Desta forma, para ambos os cenários de provisionamento, o tempo de responsividade das ações de pro-

visionamento é de suma importância para o desempenho da abordagem de provisionamento automático utilizada pelo serviço de provisionamento.

Esse tempo de provisionamento ou responsividade de provisionamento foi medido através de experimentos de medição do tempo envolvido em cada ação de provisionamento realizada sobre a infraestrutura de execução. O experimento consistiu em alocar ou desalocar uma VM do conjunto de recursos que proveem uma aplicação Web intensiva em CPU (do inglês *CPU-bound*) e medir os tempos relacionados ao processo de provisionamento. A aplicação utilizada no experimento consiste em um serviço Web que executa operações intensivas em CPU a cada requisição HTTP, cuja intensidade, ou o tempo de CPU requerido para execução, é definida a partir de um parâmetro da própria requisição.[1]. Desta forma, é possível garantir a execução da aplicação com requisições que apresentam intensidades configuráveis de consumo de CPU e com diferentes níveis de utilização de CPU da infraestrutura de execução.

A infraestrutura considerada para a execução da aplicação corresponde a um conjunto configurável de VMs, com instâncias da aplicação em questão, e um balanceador de carga agindo como um escalonador de requisições[2]. O nível de utilização das VMs que executam a aplicação, antes da alocação ou desalocação de uma nova VM, é um fator do experimento. Esse fator é definido em função da taxa de chegada de requisições e da intensidade dessas requisições. A carga da aplicação, em termos de requisições por segundo, é produzida através de um gerador de requisições HTTP [18]. Durante cada cenário do experimento, a injeção de carga é mantida constante, antes e depois do provisionamento, com duração total de 10 minutos. Portanto, a métrica observada é o tempo de provisionamento para os cenários de provisionamento (alocação e desalocação de recursos) de aplicações com diferentes configurações de intensidade de requisição e diferentes níveis de utilização de CPU da infraestrutura de execução. O experimento seguiu um projeto fatorial completo com 4 fatores e 10 repetições, conforme detalhado na Tabela A.1.

Uma instalação do OpenStack [50] (versão *Icehouse*) provê ao experimento uma infraestrutura de Computação na Nuvem similar àquelas disponibilizadas pelo atual mercado de IaaS. Para a sua instalação foi usada uma máquina HP Intel(R) Xeon(R) CPU

[1] A aplicação calcula o valor fatorial de um inteiro passado como parâmetro da requisição.

[2] O balanceador de carga do Apache configurado com o algoritmo de escalonamento por requisição.

Tabela A.1: Projeto experimental de análise de tempo de provisionamento

Fatores primários	Configuração
Capacidade da infraestrutura	1, 2, 3 e 4 VMs
Nível de utilização de CPU da infraestrutura	50, 70 e 90%
Intensidade da requisição	5000 e 10000
Tipo do provisionamento	Alocação e Desalocação

X5550@2.67GHz (16 núcleos), com 20GB de memória RAM, 500GB de disco e o Ubuntu Server 14.04 como sistema operacional. As VMs usadas no experimento foram configuradas com 1 núcleo virtual de CPU, 2GB de memória RAM, 20GB de disco (*m1.small*, tipo padrão do OpenStack) e instalações do Ubuntu Server 12.04 e Apache2 como servidor Web.

Na Figura A.1 é possível observar os resultados dos tempos envolvidos no provisionamento para os cenários operados no experimento, onde o tempo total de provisionamento (barra em verde) é constituído pela soma do tempo de inicialização e de balanceamento de carga. Evidentemente, a segunda linha do gráfico, com os tempos referentes à desalocação de recursos, não apresenta a barra (em vermelho) com o tempo de inicialização por se tratar de um cenário de remoção de VMs da infraestrutura. Todavia, para todos cenários experimentados, o tempo necessário para o provisionamento eficaz da infraestrutura não foi maior que 90 segundos. Além disso, segundo a literatura, para cenários de IaaS em produção verifica-se tempos superiores de inicialização de VMs, com um tempo médio de inicialização de uma VM na Amazon AWS em torno de 100 segundos [39].

Desta forma, a partir dos resultados experimentais e da literatura, conclui-se as técnicas de provisionamento consideradas em um serviço de provisionamento automático devem ser capazes de lidar com um tempo de responsividade de provisionamento na casa de minutos. Em outras palavras, o gerenciador de recursos deve ter em conta, durante o provisionamento de aplicações intensivas em CPU, que ações de provisionamento, principalmente de adição de recursos, demandam em média pelo menos 90 segundos para serem de fato efetivadas e começarem a servir normalmente a aplicação.

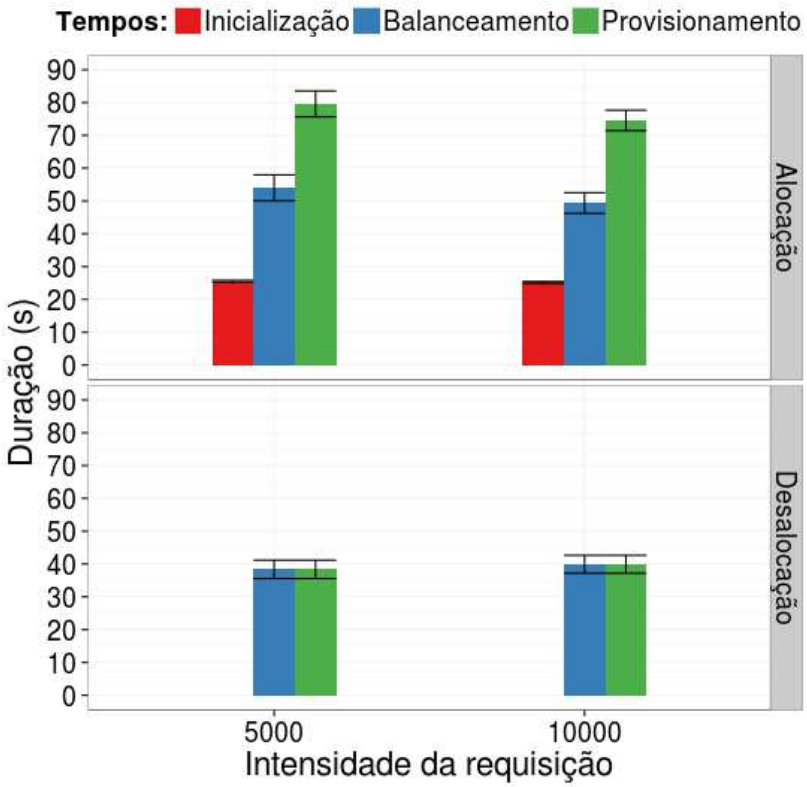


Figura A.1: Análise do tempo responsividade do provisionamento horizontal de aplicações intensivas em CPU.

Apêndice B

Erro de Estimativa de Modelos de Predição de Séries Temporais

A partir da revisão da literatura realizada no Capítulo 3 é possível enumerar uma variedade de soluções de provisionamento automático proativas que usam diferentes técnicas preditivas para estimar a carga de trabalho das aplicações provisionadas. No entanto, dado o cenário de provisionamento automático como um serviço em IaaS proposto nesse trabalho, apenas técnicas preditivas não intrusivas devem ser consideras para compor soluções de provisionamento proativo. Desta forma, apenas os trabalhos de Caron et al. [14] e Morais et al. [44] fazem uso de soluções de provisionamento não intrusivas, baseada em métricas de utilização da infraestrutura, e aplicáveis em um cenário de provisionamento automático como um serviço.

Nesse sentido, este estudo tem como objetivo avaliar o erro de predição obtido a partir dos modelos de predição de séries temporais utilizados pelos estudos acima citados, aplicados especificamente para gerar estimativas futuras sobre utilização de CPU da infraestrutura [1]. A avaliação considera 7 diferentes modelos preditivos:

- Medições anteriores: o algoritmo realiza predições de futuras demandas de utilização através da repetição de valores de utilização coletados do sistema no último intervalo ou janela de tempo observado (LW, do inglês *Last Window*);
- Autocorrelação: a técnica de autocorrelação (AC, do inglês *Auto-Correlation*) visa en-

[1] O erro de predição é calculado a partir do módulo da diferença relativa entre os valores reais e estimados.

contrar um padrão de repetição nos dados de utilização coletados. Para tal, o algoritmo calcula a correlação dos dados de utilização originais com deslocamentos crescentes no tempo destes mesmos dados e utiliza o valor do dado para o deslocamento com maior coeficiente de correlação como valor de predição;

- Regressão linear: O modelo de regressão linear (LR, do inglês *Linear Regression*) estima futuros valores de utilização a partir de uma função derivada através de uma regressão linear dos valores históricos de utilização dos últimos intervalos de tempo;

- Auto-regressão: o modelo de auto-regressão (AR, do inglês *Auto-Regressive*) prevê a utilização da demanda com base em uma combinação linear ponderada dos valores do histórico de utilização recursos [31];

- Auto-regressão com média móvel integrada: as estimativas de utilização desse modelo de auto-regressão com média móvel integrada (ARIMA, do inglês *Auto-Regressive Integrated Moving Average*) são obtidas através de diferenciações da sequência não estacionária de dados de utilização do passado e do ajuste de um modelo ARMA, que é composto pelo modelo AR em conjunto com um modelo de média móvel (MA, do inglês *Moving Average*) [16];

- Combinação de preditores atráves de pesos: o algoritmo de combinação de preditores (EN, do inglês *Ensemble*) utiliza uma estratégia de combinação linear ponderada de um conjunto pré-determinado de preditores para realizar estimativas de demanda, calculado com base nos erros de predição de cada modelo de predição considerado [33];

- Casamento de padrões: o modelo de predição de casamento de padrões (PM, do inglês *Pattern Matching*) proposto por Caron et al. [14] utiliza uma abordagem baseada casamento de cadeia de caracteres (do inglês *String Matching*), especificamente o algoritmo Knuth-Morris-Pratt (KMP), para identificar padrões de utilização de recursos no passado que são similares ao uso de recursos da aplicação no presente.

Os modelos de predição foram utilizados para estimar demandas de CPU de 30 aplicações reais de usuários da HP. A cada intervalo de tempo de 5 minutos os preditores LW, AC, LR, AR, ARIMA e EN foram alimentados com o histórico recente de dados com 2 semanas de

duração para gerar estimativas de utilização de CPU no curto prazo, assim como proposto no trabalho de Morais et al. De forma semelhante, cada estimativa do modelo de predição proposto por Caron et al. realizada com um histórico de 15 dias de dados de utilização de CPU. O preditor EN considerou valores de predições dos outros 5 modelos de predição usados no trabalho de Morais et al. (LW, AC, LR, AR, ARIMA).

O algoritmo de casamento de padrões é baseado no erro instantâneo entre os valores de dois intervalos de tempo e no erro acumulado desses erros pontuais para cada padrão verificado. Todavia, informações sobre a parametrização dos limites aceitáveis para cada tipo de erro não encontram-se disponíveis no trabalho publicado. Desta forma, fez-se necessário uma varredura de parâmetros a fim de avaliar o desempenho da técnica de predição proposta. Para os cenários em que não são encontrados padrões que respeitam esses limites as predições são realizadas através do preditor LW. A varredura considerou uma configuração com limites de erro instantâneo de 1%, 10%, 30% e 50% e de erro acumulado de 70%, 80%, 90% e 100%.

Os valores de predição obtidos pelos diferentes modelos e configurações foram confrontados com os dados reais de utilização de CPU do rastro original. Para cada intervalo de tempo foi calculado o erro relativo de predição em termos percentuais. A Figura B.1 mostra o diagrama de caixa da mediana do módulo do erro relativo de predição para cada aplicação considerada. No entanto, os resultados do algoritmo PM configurado com erro instantâneo maior que 10% foram omitidos por serem significativamente superiores aos demais cenários, com erros medianos maiores que 1000% para mais de 75% dos casos avaliados. Os resultados do algoritmo de casamento de padrão são referenciados por "PM X-Y", onde X equivale ao limite de erro instantâneo e Y ao limite de erro acumulado.

Como pode ser observado, o desempenho das instanciações do modelo de predição PM apresentam erros de predição superiores aos demais preditores. A mediana do erro relativo do modelo PM é de aproximadamente 38%, enquanto que o erro gira em torno 13% para os outros modelos de predição. Além disso, as configurações do preditor PM que conseguiram melhor desempenho, com limite de erro instantâneo igual a 1% e 10%, são os casos em que são encontrados menos padrões de carga que se adéquam aos limites de erros instantâneo e acumulado. Para esses casos, em média em 55% de todas as predições realizadas não foram encontrados padrões e a estimativa de carga foi produzida por outro modelo de predição

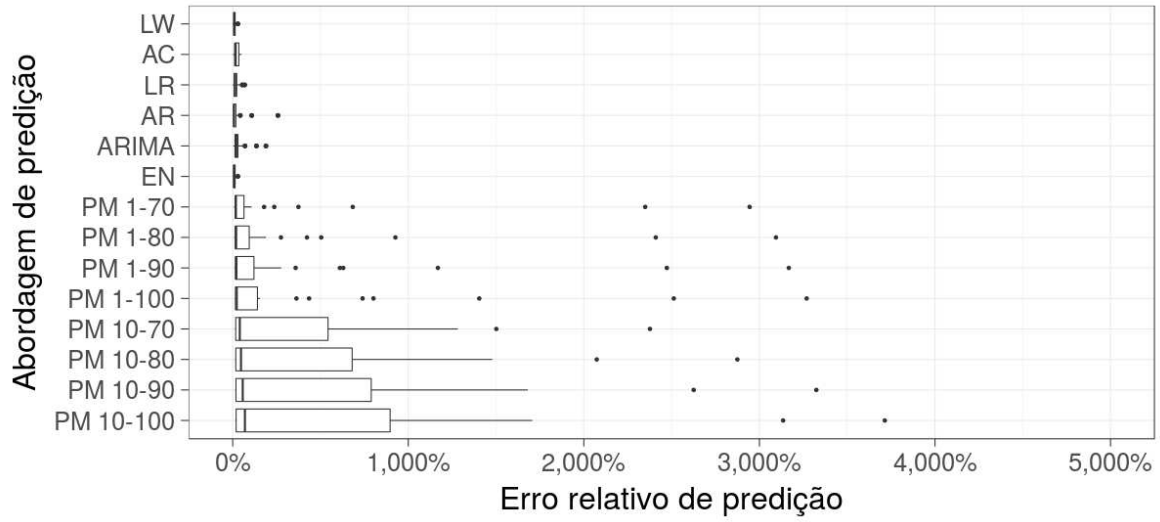


Figura B.1: Análise do erro de predição relativo de modelos de predição de utilização de CPU usados por suluções de provisionamento proativo.

(preditor LW).

Como base na análise dos erros relativos de predição dos modelos avaliados observa-se que dentre estes o modelo PM é o que apresenta pior desempenho. Além do percentual elevado dos erros de predição a abordagem apresenta baixo desempenho em termos do número de casamento de padrões encontrados. Isso possivelmente deve-se ao fato do trabalho original, que propõe esta abordagem, fazer uso de um conjunto de dados de aplicações diferentes daquelas que foram utilizadas nesse estudo. Além dos dados serem coletados com maior frequência, a cada segundo, é possível que as aplicações que consumiram os recursos de CPU possuam características diferentes das aplicações da HP. Desta forma, dado do desempenho dos modelos de predição, considera-se que os modelos de predição mais adequados, dentre os avaliados, para compor uma solução de provisionamento automático não intrusiva, em termos da acurácia da estimativa de demanda, são aqueles utilizados ou propostos no trabalho de Morais et al.