Neste trabalho, propõe-se Escada, um novo modelo para previsão de largura de banda de rede necessária para guiar o provisionamento de máquinas virtuais em nuvens IaaS. O modelo é agnóstico de aplicação e adapta-se dinamicamente à evolução do perfil de carga da máquina virtual. A partir da monitoração dos fluxos de rede, é produzida uma estimativa da largura de banda total necessária a uma máquina virtual em determinado período, e um algoritmo de previsão é aplicado sobre o histórico de estimativas para prever a largura de banda necessária futura da máquina virtual.

Orientador: Rafael Rodrigues Obelheiro

Joinville, 2017

ANO
2017

JONATAS ADILSON MARQUES | UM MODELO PARA PROVISIONAMENTO DE REDE DE MÁQUINAS VIRTUAIS EM NUVENS IAAS

UNIVERSIDADE DO ESTADO DE SANTA CATARINA – UDESC
CENTRO DE CIÊNCIAS TECNOLÓGICAS – CCT
PROGRAMA DE PÓS GRADUAÇÃO EM COMPUTAÇÃO APLICADA

DISSERTAÇÃO DE MESTRADO

UM MODELO PARA PROVISIONAMENTO DE REDE DE MÁQUINAS VIRTUAIS EM NUVENS IAAS

JONATAS ADILSON MARQUES

JOINVILLE, 2017

UNIVERSIDADE DO ESTADO DE SANTA CATARINA - UDESC
CENTRO DE CIÊNCIAS TECNOLÓGICAS - CCT
MESTRADO EM COMPUTAÇÃO APLICADA

JONATAS ADILSON MARQUES

UM MODELO PARA PROVISIONAMENTO DE REDE DE MÁQUINAS VIRTUAIS EM NUVENS IAAS

JOINVILLE

2017

JONATAS ADILSON MARQUES

UM MODELO PARA PROVISIONAMENTO DE REDE DE MÁQUINAS VIRTUAIS EM NUVENS IAAS

Dissertação submetida ao Programa de Pós-Graduação em Computação Aplicada do Centro de Ciências Tecnológicas da Universidade do Estado de Santa Catarina, para a obtenção do grau de Mestre em Computação Aplicada.

Orientador: Dr. Rafael Rodrigues Obelheiro

JOINVILLE

2017

M357u Marques, Jonatas Adilson
Um modelo para provisionamento de rede de máquinas virtuais em nuvens IaaS / Jonatas Adilson Marques. – 2017.
64 p. : il. ; 30 cm

Orientador: Rafael Rodrigues Obelheiro
Bibliografia: p. 61-63
Dissertação (mestrado) – Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Computação Aplicada, Joinville, 2017.

1. Computação em nuvem. 2. Infrastructure-as-a-Service. 3. Provisionamento de recursos. 4. Previsão de demanda de rede. I. Obelheiro, Rafael Rodrigues. II. Universidade do Estado Santa Catarina. Programa de Pós-Graduação em Computação Aplicada. III. Título.

CDD 004.6782 – 23.ed.

Um modelo para Provisionamento de Rede de Máquinas Virtuais em Nuvens IaaS

por

Jonatas Adilson Marques

Esta dissertação foi julgada adequada para obtenção do título de

Mestre em Computação Aplicada

Área de concentração em "Ciência da Computação,
e aprovada em sua forma final pelo

CURSO Mestrado Acadêmico em Computação Aplicada
CENTRO DE CIÊNCIAS TECNOLÓGICAS DA
UNIVERSIDADE DO ESTADO DE SANTA CATARINA.

Banca Examinadora:

Prof. Dr. Rafael Rodrigues Obelheiro
CCT/UDESC (Orientador/Presidente)

Prof. Dr. Luciano Paschoal Gaspary
UFRGS

Prof. Dr. Mauricio Aronne Pillon
CCT/UDESC

Joinville, 03 de março de 2017.

Dedicado a Adilson, Eliane e Gabriele.

“But of that day and hour no one knows, not even the angels of the heavens, but [my] Father alone.”

Matthew 24:36 (DARBY)

RESUMO

Em nuvens computacionais do tipo Infrastructure-as-a-Service (IaaS), clientes alugam recursos virtuais (processador, memória, rede) fornecidos por provedores de nuvem, pagando pela capacidade reservada, independente da efetiva utilização dos recursos. Nesse contexto, é de interesse dos clientes reservar recursos com capacidade suficiente para que suas aplicações atinjam um bom desempenho, buscando ao mesmo tempo minimizar gastos com capacidade ociosa. O provisionamento adequado dos recursos também é útil aos provedores, pois estes buscam maximizar o nível de uso dos recursos de sua infraestrutura física, para, por exemplo, evitar o gasto com a manutenção de *hardware* de alta capacidade (*e.g.*, servidores, roteadores, *switches*) ociosos.

Neste trabalho, propõe-se Escada, um novo modelo para previsão de largura de banda de rede necessária para guiar o provisionamento de máquinas virtuais em nuvens IaaS. O modelo é agnóstico de aplicação e adapta-se dinamicamente à evolução do perfil de carga da máquina virtual. A partir da monitoração dos fluxos de rede, é produzida uma estimativa da largura de banda total necessária a uma máquina virtual em determinado período, e um algoritmo de previsão é aplicado sobre o histórico de estimativas para prever a largura de banda necessária futura da máquina virtual.

Os resultados da avaliação experimental mostram que a estimação de Escada é útil para guiar a previsão da largura de banda necessária, pois provê estimativas com exatidão suficiente mesmo quando a rede está subprovisionada. Escada ajusta seu algoritmo de previsão em menor tempo que o modelo do estado da arte e é mais robusto na previsão durante o período de ajuste.

Palavras-chaves: computação em nuvem, Infrastructure-as-a-Service, provisionamento de recursos, previsão de demanda de rede.

ABSTRACT

In Infrastructure-as-a-Service (IaaS) clouds, customers lease virtual resources (*e.g.*, CPU, memory, network) offered by cloud providers, paying for the allocated capacity of resources, regardless of their effective use. In this context, it is in the interest of the customers to reserve resources with sufficient capacity so that their applications achieve good performance whilst, at the same time, minimizing their expenses with idle capacities. The correct provisioning of resources is also invaluable to providers, who seek to maximize the resource usage of their physical infrastructures, avoiding spending maintenance costs with unused hardware such as high capacity servers and networking devices.

We introduce Escada, a novel model for network bandwidth prediction to guide the provisioning of virtual machines. Escada is application-agnostic and dynamically adapts to changes in a virtual machine's workload profile. The network flows of a virtual machine are monitored and used to estimate its total required bandwidth over a time period. A prediction algorithm is then applied on the entire history of estimates to predict the virtual machine's future required bandwidth.

Our experimental evaluation shows that the estimates provided by Escada are useful for guiding the prediction of network bandwidth requirements, providing accurate bandwidth estimates even when the network is under-provisioned. Furthermore, Escada correctly adjusts its prediction algorithm faster than the state-of-the-art model, and predicts more reliably during the adjusting period.

Key-words: cloud computing, Iaas, resource provisioning, network demand prediction.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

MV máquina virtual

LB largura de banda de rede

IaaS *Infrastructure as a Service*

MMV monitor de máquinas virtuais

FR registro de fluxo

ARMA *AutoRegressive Moving Average*

ARIMA *AutoRegressive Integrated Moving Average*

TCP *Transmission Control Protocol*

UDP *User Datagram Protocol*

API *Application Programming Interface*

HTTP *Hypertext Transfer Protocol*

HTTPS *HTTP Secure*

SYN *SYNchronize*

ACK *ACKnowledgement*

FIN *FINalize*

MPE Erro Percentual Médio

ERP Erro Relativo Percentual

SSD Solid State Drive

HDD Hard Disk Drive

RAM Random-Access Memory

RUBiS *Rice University Bidding System*

SUMÁRIO

1 INTRODUÇÃO

A computação em nuvem é um modelo que permite o acesso a recursos computacionais configuráveis (como redes, servidores, armazenamento, aplicações e serviços) por seus clientes (MELL; GRANCE, 2011), que pagam conforme a capacidade reservada dos recursos, independente da efetiva utilização destes (SULEIMAN et al., 2012). Neste contexto, clientes de nuvens computacionais buscam reservar a capacidade mínima de recursos sem impactar negativamente o desempenho das aplicações que os usam, para evitar pagar por recursos reservados mas não utilizados ou degradar a experiência de uso das aplicações pelos usuários finais (PFITSCHER; PILLON; OBELHEIRO, 2013).

De modo geral, em nuvens do modelo *Infrastructure as a Service* (IaaS) a rede é compartilhada entre os clientes, sem garantia da largura de banda de rede (LB) disponível para as máquinas virtuais. Essa ausência de garantia prejudica não apenas os clientes, pois o desempenho de muitas aplicações é sensível ao desempenho da rede e este se torna imprevisível (BALLANI et al., 2011), mas também os provedores, que podem arcar com custos decorrentes de violações de acordos de nível de serviço (*service level agreements*, SLAs) e deixar de atrair clientes que demandem previsibilidade da rede (MOGUL; POPA, 2012). Abstrações mais refinadas, tais como infraestruturas virtuais (ANHALT; KOSLOVSKI; PRIMET, 2010), permitem reservar LB para enlaces virtuais, mas exigem que o cliente especifique a capacidade desejada, o que não é uma tarefa trivial, ainda mais considerando que a capacidade necessária às aplicações varia ao longo do tempo (LACURTS et al., 2014; XIE et al., 2012). Portanto, um mecanismo que seja capaz de estimar continuamente a LB necessária para enlaces virtuais seria útil tanto para os clientes, que poderiam procurar serviços com garantia da capacidade de rede, quanto provedores, pois facilitaria a oferta de serviço garantido com bom custo-benefício ao permitir uma utilização otimizada das infraestruturas físicas com mínimo risco de problemas de desempenho.

O dimensionamento de enlaces virtuais para ambientes de nuvem ainda é pouco explorado na literatura. Alguns trabalhos focam em mecanismos para oferecer garantias de LB em nuvens, com base em demandas especificadas pelos clientes (POPA et al., 2012a; ANASTASI et al., 2016; MARCON et al., 2016). Em (PFITSCHER; PILLON; OBELHEIRO, 2013) é proposto um modelo de diagnóstico de provisionamento de rede que identifica se a LB está subprovisionada, superprovisionada ou adequada, mas que, nos casos de sub- e superprovisionamento, não fornece uma estimativa de LB adequada. Cicada (LACURTS et al., 2014) propõe um mecanismo para previsão de LB entre pares de máquinas virtuais (MVs) considerando a demanda média ou de pico em um dado intervalo, mas não trata o tráfego entre a MV e *hosts* na Internet e nem é capaz de estimar a LB necessária quando a rede está subprovisionada.

1.1 OBJETIVOS

O objetivo principal deste trabalho é fornecer previsões suficientemente acuradas da largura de banda de rede necessária de máquinas virtuais em nuvens IaaS para provisionar adequadamente as aplicações nessas implantadas.

Para alcançar o objetivo principal alguns objetivos específicos são elencados:

- Identificar uma abstração para caracterizar o tráfego de rede que possua atributos que indiquem ou sugiram a real demanda de capacidade de rede;
- Propor um modelo para estimação da largura de banda de rede necessária em quaisquer níveis de provisionamento de rede (*i.e.*, subprovisionamento, superprovisionamento e provisionamento adequado);
- Propor ou identificar um modelo para previsão da largura de banda futura;
- Avaliar experimentalmente o impacto do nível de provisionamento de rede sobre a estimação e a previsão;
- Avaliar experimentalmente a qualidade das previsões do modelo.

1.2 ABORDAGEM E CONTRIBUIÇÕES

Esta dissertação foca no projeto e análise do Escada, um novo modelo projetado para melhorar as previsões de demanda de rede de aplicações implantadas em nuvens computacionais IaaS. Tanto quanto sabemos, Escada é o primeiro modelo que, além de identificar a problemática de que a capacidade provisionada de rede pode limitar a demanda observada, vai na direção de tratá-la para estimar e prever melhor as demandas de rede independentemente do nível de provisionamento da rede resultantes de sua previsão.

O Escada monitora a ocorrência de fluxos nas interfaces de rede de uma máquina virtual para identificar o volume de dados das transações de todas as aplicações, sem a necessidade de conhecê-las. Algoritmos de aprendizado de máquina são usados para identificar grupos de fluxos com demandas de rede semelhantes, e o tráfego de rede é classificado (por um novo algoritmo) segundo esses grupos. Essa classificação permite identificar demandas de rede não atendidas e estimar a largura de banda necessária real para a máquina virtual. As estimativas de largura de banda necessárias passadas são usadas por um algoritmo de *expert-tracking* para prever as LBs necessárias futuras. Resultados experimentais mostram que Escada fornece estimativas de períodos passados com exatidão suficiente mesmo quando a rede está subprovisionada, e em função disso consegue prever as LBs necessárias futuras acuradamente requerendo um período menor de ajuste automático quando comparado a um modelo que não trata a problemática do subprovisionamento.

1.3 ESTRUTURA DA DISSERTAÇÃO

Este trabalho está organizado em 5 capítulos. No Capítulo 1 é apresentada a introdução do problema, são descritos os objetivos do trabalho e dá-se uma visão geral do modelo proposto. No Capítulo 2 define-se os principais termos usados no trabalho e faz-se uma revisão teórica sobre os conteúdos e importantes ao entendimento deste trabalho, tais como, conceitos de computação em nuvem, provisionamento de rede em nuvens computacionais, monitoração de fluxos de rede, e *cluster analysis*. Ainda nesse, são discutidos os trabalhos relacionados ao contexto e problemática do presente trabalho. O Capítulo 3 detalha o modelo Escada, apresentando sua arquitetura, seu fluxo de execução e seu procedimento. No Capítulo 4 realiza-se uma avaliação do Escada, analisando seu comportamento em diferentes níveis de provisionamento, sua capacidade de oferecer estimativas passadas e prever valores futuros da largura de banda necessária. Por fim, no Capítulo 5 relata-se as considerações finais do trabalho e as perspectivas de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este trabalho propõe um modelo para provisionamento de rede de máquinas virtuais em nuvens IaaS. O presente capítulo revisa os conceitos e trabalhos relacionados necessários para o entendimento desse modelo e seu contexto. Inicialmente a Seção 2.1 apresenta as características e definições sobre computação em nuvem. Na sequência, a Seção 2.2 discute o provisionamento de rede em ambientes de computação em nuvem. A Seção 2.3 examina aspectos da monitoração de fluxos de rede para análise de tráfego. A Seção 2.4 descreve o processo de *cluster analysis* para a identificação e agrupamento de objetos semelhantes. Por fim, a Seção 2.5 discute os trabalhos relacionados ao contexto e problemática do presente trabalho.

2.1 COMPUTAÇÃO EM NUVEM

A definição mais reconhecida de computação em nuvem foi proposta pelo *National Institute of Standards and Technology* em (MELL; GRANCE, 2011):

> "Computação em nuvem é um modelo para permitir o acesso ubíquo, conveniente, sob demanda e remoto a um pool compartilhado de recursos computacionais configuráveis (por exemplo, redes, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente provisionados e liberados com um mínimo de esforço de gerenciamento ou interação com o provedor de serviços."

Esta definição faz referência a uma característica essencial às nuvens computacionais e de interesse para o presente trabalho, a elasticidade de provisionamento. Esta característica expressa que as capacidades provisionadas dos recursos podem ser ajustadas (para mais ou para menos), ao longo do tempo, de forma a atender apropriadamente novas demandas de recursos do cliente. Desta forma, clientes veem as capacidades de provisionamento como ilimitadas, podendo atender a qualquer demanda em qualquer momento (MELL; GRANCE, 2011).

A elasticidade do provisionamento de recursos pode ser classificada em horizontal ou vertical (LORIDO-BOTRAN; MIGUEL-ALONSO; LOZANO, 2014). Na elasticidade horizontal, réplicas da máquina virtual (MV) que executa a aplicação do cliente são adicionadas ou removidas conforme a demanda da aplicação. Em contraste, a elasticidade vertical permite o ajuste da capacidade dos recursos das MVs existentes, por exemplo, pode-se ajustar o poder de processamento ou a quantidade de memória. O presente trabalho tem enfoque na elasticidade vertical, pois objetiva-se ajustar a largura de banda de rede disponível às MVs de um cliente de nuvem.

A computação em nuvem utiliza um modelo de negócio orientado a serviço, onde recursos são oferecidos como serviços sob demanda. Múltiplos tipos de recursos podem ser

oferecidos, sendo que neste trabalho o recurso de interesse é rede (largura de banda dos enlaces e interfaces de rede). Os modelos de negócio das nuvens computacionais podem ser classificados em três modelos de serviço, de acordo com o tipo de recurso oferecido: *software*, plataforma de desenvolvimento, ou infraestrutura (ZHANG; CHENG; BOUTABA, 2010; MELL; GRANCE, 2011).

Este trabalho é voltado ao modelo de serviço que oferece infraestruturas, denominado *Infrastructure as a Service* (IaaS). Neste modelo, cada usuário tem acesso a seus recursos na forma de infraestruturas configuráveis e gerenciáveis. Sobre estas infraestruturas, é oferecida a possibilidade de provisionar instâncias de máquinas virtuais com capacidade de processamento, armazenamento e comunicação entre si e com a Internet. Todos estes recursos são virtualizados sobre recursos físicos (*e.g.*, servidores, roteadores, enlaces) do provedor da nuvem. Os clientes acessam estes recursos através de interfaces de configuração e mecanismos de acesso remoto, possibilitando implantar suas aplicações e serviços (MELL; GRANCE, 2011).

Além da classificação quanto ao modelo de serviço, as nuvens computacionais podem ser categorizadas por suas finalidades (ZHANG; CHENG; BOUTABA, 2010; MELL; GRANCE, 2011):

- Privada: A infraestrutura da nuvem é provisionada para uso exclusivo por uma única organização. Esta infraestrutura pode pertencer, ser gerenciada e operada pela própria organização, uma organização terceira, ou alguma combinação destas; e estar situada dentro ou fora das dependências da organização. Esta categoria de nuvem oferece o maior grau de controle sobre o desempenho, a confiabilidade e a segurança;

- Comunitária: A infraestrutura é provisionada para uso exclusivo por uma comunidade de usuários de organizações com objetivos compartilhadas (*e.g.*, pesquisadores em diferentes universidades). Esta infraestrutura pode pertencer, ser gerenciada e operada por uma ou mais das organizações, uma organização terceira, ou alguma combinação destas; e estar situada dentro ou fora das dependências das organizações;

- Pública: A infraestrutura é provisionada para uso do público em geral. Esta infraestrutura pode pertencer, ser gerenciada e operada por uma organização lucrativa, acadêmica ou governamental, ou alguma combinação destas; e deve estar situada dentro das dependências do provedor;

- Híbrida: Infraestruturas compostas de duas ou mais infraestruturas distintas (privadas, comunitárias ou públicas) que permanecem entidades singulares, mas que são interligadas por tecnologias que permitem a portabilidade de dados e aplicações.

A proposta deste trabalho tem aplicação e traz potenciais benefícios a todas as finalidades de nuvens computacionais, conforme será discutido na Seção 2.2.

Terminologia Adotada neste Documento

Neste trabalho os termos **nuvem computacional**, **nuvem IaaS** (*Infrastructure-as-a-Service* ou simplesmente **nuvem** se referem à infraestrutura disponibilizada segundo o modelo de computação em nuvem. A entidade responsável pela gerência de uma nuvem computacional é denominada **provedor**. **Clientes**, ou **usuários de nuvem**, são entidades que executam suas aplicações em nuvens computacionais. As entidades que usam as aplicações dos clientes são, por sua vez, chamados **usuários de aplicação**.

Um conjunto de programas que realizam uma tarefa são descritos como sendo uma **aplicação**. Por exemplo, um cliente que deseja instanciar uma aplicação de comércio eletrônico comumente executa programas como servidores HTTP/HTTPS e bancos de dados. Durante sua execução, uma aplicação usa os recursos computacionais (*e.g.*, processador, memória, interface de rede) das MVs nas quais está implantada para atender a sua **carga de trabalho**. Essa carga de trabalho normalmente apresenta variação de intensidade ao longo do tempo, que se traduz na variação das capacidades dos recursos computacionais necessárias à aplicação.

2.2 PROVISIONAMENTO DE REDE

Provisionamento, em ambientes de computação em nuvem, consiste na alocação e distribuição de recursos a máquinas virtuais (MVs). O nível de granularidade do provisionamento depende das estratégias de distribuição e apreciação de recursos do provedor da nuvem. A rede pode ser alocada por unidades de interface de rede (granularidade grossa) ou por largura de banda (LB) (granularidade fina) (LAGAR-CAVILLA et al., 2009).

Considerando uma configuração inicial de capacidades de recursos e MVs determinada por um usuário de nuvem, o ajuste de provisionamento pode ser realizado de dois modos:

- Estático – a realocação dos recursos não é permitida em tempo de execução das MVs. Caso o cliente identifique que necessita alterar a capacidade de seus recursos, deve desligar, reconfigurar e reiniciar a MV;
- Dinâmico – em oposição ao estático, este permite que recursos sejam realocados em tempo de execução da MV.

No caso de rede, os modos de ajuste oferecidos ao usuário dependem das tecnologias de alocação e distribuição implantadas pelo provedor. Por exemplo, no monitor de máquinas virtuais (MMV) Xen (BARHAM et al., 2003) é possível definir a LB de uma MV em sua criação, mas não é possível ajustá-la durante a execução da MV. Para realizar o ajuste dinâmico de LB, um provedor pode optar por usar políticas de controle de filas nas interfaces de rede que limitem a taxa máxima de transmissão nestas através de ferramentas como tc (HUBERT, 2017).

No presente trabalho, a taxa de tranmissão (ou capacidade) de um enlace ou uma interface de rede é referida como **largura de banda de rede** (LB); e expressa em bits por segundo (*e.g.*, kbps, Mbps, Gbps). **LB necessária** de uma aplicação refere-se à taxa de transmissão *efetiva* (em contraste com a nominal) necessária para que ela atinja um desempenho adequado. A rede é dita **superprovisionada** quando sua capacidade (LB nominal) é superior à suficiente para a aplicação atingir sua LB necessária (é possível reduzir a capacidade provisionada sem afetar significativamente o desempenho da aplicação) (PFITSCHER; PILLON; OBELHEIRO, 2013). Além disso, o aumento da capacidade não representa melhora relevante no desempenho. O **subprovisionamento** de rede significa que a capacidade provisionada é inferior à suficiente para que a aplicação atinja sua LB necessária (o aumento da capacidade representa uma melhora relevante no desempenho). O **provisionamento adequado** representa *um* estado em que a redução de capacidade afetaria significativamente o desempenho, e o seu aumento não representaria uma melhora relevante no desempenho. Da definição acima, ressalta-se que o provisionamento adequado pode ser obtido em uma faixa de diferentes valores, e não em um único valor específico. A taxa de transmissão nominal disponível a uma MV segundo requisição do cliente é denominada **LB reservada** ou **LB alocada**.

A expressão **estimar a LB** – e derivadas como **estimativa de LB** – refere-se ao ato de calcular um valor aproximado da LB necessária de uma MV em um período *passado*, considerando dados coletados por monitoração. Subestimar a LB significa estimar valor inferior à LB necessária real. Superestimar significa estimar valor superior à LB necessária real. Usa-se o termo estimar em vez do termo medir, pois a simples medição da taxa de transmissão não representa a real necessidade da MV (principalemente em cenários de subprovisionamento de rede). Já as expressões **prever ou predizer a LB** – e derivadas como **previsão de LB** – refere-se ao ato de tentar antecipar o valor da LB necessária de um período *futuro*, considerando *estimativas* de períodos passados. Subprever indica prever valor inferior a real LB necessária futura. Superprever indica prever valor superior a real LB necessária futura.

Em (VAN; TRAN; MENAUD, 2009) argumenta-se que uma desvantagem proveniente da elasticidade é o aumento da complexidade do gerenciamento da nuvem realizado pelo provedor. No presente trabalho, considera-se que a explicitação da LB necessária às MVs auxilia no tratamento desta complexidade, permitindo aos provedores uma alocação e distribuição mais informada de recursos virtuais sobre o substrato físico da infraestrutura da nuvem. Isto é de bastante valia, por exemplo, para nuvens privadas onde a capacidade de recursos virtuais necessária costuma se aproximar da capacidade física, e o mapeamento adequado das instâncias de MVs sobre os recursos físicos torna-se um fator essencial para garantir o bom desempenho das aplicações da organização e boa utilização dos recursos.

Permitir, aos usuários de nuvem, a determinação de LBs aos seus enlaces virtuais e dar uma garantia sobre estas é um fator chave para que as aplicações dos usuários atinjam um desempenho previsível. A previsibilidade do desempenho facilita o provisionamento adequado de

recursos por parte dos usuários da nuvem. O desempenho imprevisível, por sua vez, leva usuários a provisionarem em excesso suas aplicações, na tentativa de obter um desempenho aceitável na situação de receber capacidade aquém da esperada. Este superprovisionamento de recursos resulta em um custo supérfluo aos usuários, que poderia ser investido em outras áreas do seu negócio. E além disso, causa também prejuízos ao provedor, que tem maior dificuldade em manter seus recursos físicos com alto nível de utilização (PRIMET; ANHALT; KOSLOVSKI, 2009; POPA et al., 2012b).

2.3 MONITORAÇÃO DE FLUXOS DE REDE

Um fluxo de rede é uma sequência de pacotes que compartilham os mesmos endereços IP de origem e destino, portas de origem e destino, e protocolo IP (LUCAS, 2010). Um registro de fluxo (*flow record* – FR) é um sumário de informações sobre um fluxo, registrando que hospedeiro se comunicou com que outro hospedeiro, quando esta comunicação ocorreu, como o tráfego foi transmitido, entre outras informações básicas. Portanto, FRs resumem todas as conexões de uma rede ou hospedeiro. A monitoração de fluxos de rede consiste na observação de pacotes em uma rede (ou hospedeiro) e da geração e armazenamento de FRs.

Um detalhe implícito na definição de fluxo é que um FR descreve o tráfego em apenas um sentido de uma comunicação. Por exemplo, quando um cliente se conecta a um servidor *web* e recebe um arquivo (*i.e.*, uma sessão TCP), dois fluxos são observados: um saindo do cliente em direção ao servidor, e outro do servidor para o cliente. Esses dois fluxos apresentam endereços IP e portas invertidas, ou seja, o endereço IP e a porta de origem do primeiro fluxo são o endereço IP e a porta de destino do segundo, e vice-versa.

Bro (HALL, 2017), Argus (BULLARD, 2017), flow-tools (FULLMER, 2017) e softflowd (MILLER, 2017) são exemplos de ferramentas capazes de realizar a monitoração de fluxos (BEJTLICH, 2013). No presente trabalho, optou-se pelo uso do Argus. Esta ferramenta apresenta uma peculiaridade, pois opera por padrão com fluxos bidirecionais, combinando os fluxos correspondentes aos dois sentidos de uma comunicação em um único FR bidirecional. A Tabela 1 exemplifica FRs resultantes da monitoração com o Argus. Cada FR apresenta separadamente a quantidade de *bytes* recebidos (*SourceBytes*) e enviados (*DestBytes*) pela máquina de endereço 1.1.1.2. A Seção 3.2.1 descreve os atributos dos fluxos usadas por Escada.

A monitoração de fluxos contrasta com a monitoração de pacotes, onde todos pacotes de interesse são capturados e analisados (LUCAS, 2010). Uma vantagem da monitoração de pacotes é (no caso de sessões não cifradas) a possibilidade de reconstruir toda a comunicação de um usuário com um servidor. Por exemplo, pode-se descobrir quais os *websites* visitados pelo usuário; quais os arquivos transferidos; e quais as informações enviadas ao servidor, como nomes de usuário e senhas. Os FRs da monitoração de fluxos, por sua vez, não contêm esses dados, apenas informações como o endereço IP usado pelo usuário, quantas conexões ao servidor

Tabela 1 – Exemplo de registros de fluxo fornecidos pelo Argus após a monitoração.

Id	StartTime	LastTime	SourceAddress	SourcePort	DestAddress	DestPort	...
1	0.000731	3.391281	1.1.1.3	1240	1.1.1.2	80	...
2	0.006170	4.014569	1.1.1.4	1105	1.1.1.2	443	...
3	0.008109	2.132169	1.1.1.3	1215	1.1.1.2	443	...
4	0.008126	7.181482	1.1.1.5	1243	1.1.1.2	80	...

Protocol	Flags	SourcePackets	SourceBytes	DestPackets	DestBytes
TCP	FSPA_FSPA	292	19349	339	1772647
TCP	FRPA_FPA	4	277	2	169
TCP	FSRPA_FSPA	79	5646	89	408348
TCP	SPA_SPA	125	8327	152	760303

Fonte: autoria própria

foram realizadas pelo usuário, e o volume de dados transmitido no fluxo.

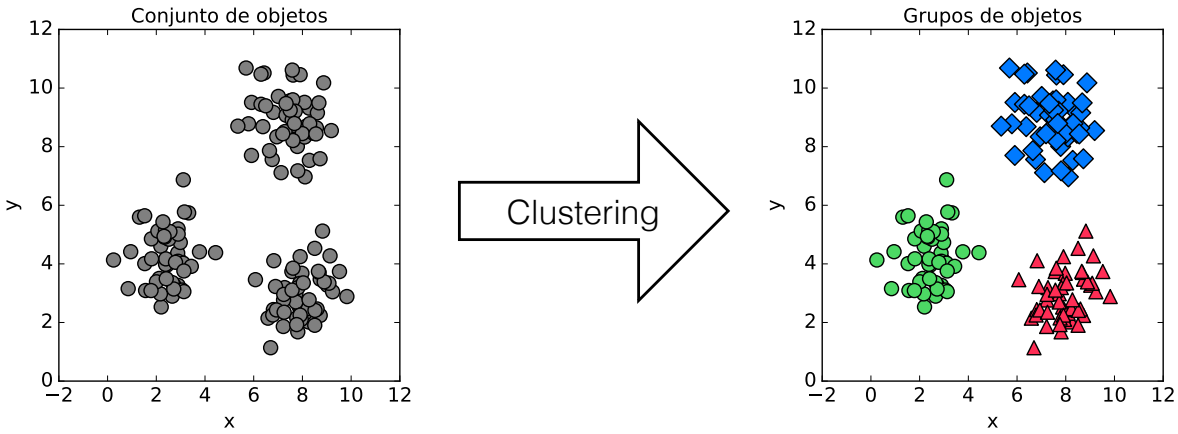
A monitoração de fluxos, entretanto, possui como vantagem uma sobrecarga menor sobre os sistemas monitorados quando comparada à monitoração de pacotes, visto que esta primeira não processa e nem armazena o conteúdo dos pacotes de rede (LUCAS, 2010). Além disso, o espaço ocupado por FRs também é menor em comparação com o usado para armazenamento de pacotes completos, pois as informações apresentam granularidade mais grossa (*i.e.*, a nível de fluxo). No contexto do presente trabalho, essas vantagens tornam a monitoração de fluxos factível.

Um benefício de abstrair o tráfego de rede através de fluxos é que estes representam bem as transações das máquinas virtuais sendo monitoradas, isso sem a necessidade de conhecer ou compreender o funcionamento das aplicações em execução. Dessa forma é possível estimar métricas de tráfego como a demanda típica de cada transação, a frequência de início de transações e entre outras, úteis para caracterizar e compreender o tráfego. O modelo Escada, proposto neste trabalho e descrito no Capítulo 3, considera métricas como estas para estimar a largura de banda de rede necessária.

2.4 *CLUSTER ANALYSIS*

Cluster Analysis (ou *clustering*) é o processo de agrupar um conjunto de objetos em grupos de forma que os objetos de um mesmo *cluster* (ou grupo) sejam mais similares entre si (segundo alguma métrica) do que similares aos objetos de outros grupos (JAIN, 1991). Existem diversos algoritmos projetados para realizar o *clustering* de dados (KABACOFF, 2015). Tipicamente, estes algoritmos recebem um conjunto de dados que representam os objetos que deseja-se agrupar e parâmetros de ajuste (*e.g.*, número de grupos); e retornam para cada objeto um rótulo indicando a qual grupo este pertence.

Figura 1 – Exemplo de *clustering* de um conjunto de objetos com os dados x e y, e com o parâmetro número de grupos igual a três.



Fonte: autoria própria

A Figura 1 exemplifica o processo de *clustering*. Do lado esquerdo apresenta-se um conjunto de objetos representados por pares (x, y); deste conjunto poderiam ser formados três conjuntos menores, agrupando os objetos com valores próximos de x e y e separando-os daqueles com valores menos próximos. O lado direito da figura ilustra um possível agrupamento (conformação de grupos) formado por algoritmo de *clustering* qualquer. Nesse lado da figura os elementos do mesmo grupo possuem mesmo símbolo e cor. Percebe-se que a variação dos valores de x e y para objetos dentre de um mesmo grupo (distância *intra-cluster*) é menor do que a variação entre objetos pertencentes a grupos diferentes (distância *inter-cluster*).

A aplicação da *cluster analysis* está normalmente associada às tarefas de análise, descrição e representação de conjuntos de objetos (TAN; STEINBACH; KUMAR, 2005). Os algoritmos de *clustering* buscam a identificação de características que permitam dividir os objetos em grupos significativos (classes). Em função disso, a *cluster analysis* é por vezes referida como uma técnica de classificação não supervisionada capaz de capturar estruturas implícitas dos conjuntos de objetos, e auxiliar na criação de taxonomias que orientam a compreensão (análise e descrição) de objetos.

Quanto à representação, alguns algoritmos de *clustering* permitem a caracterização de cada grupo formado através de um *protótipo de grupo*, que é um objeto (real ou sintético) representativo dos outros objetos do grupo (TAN; STEINBACH; KUMAR, 2005). Muitas técnicas de análise de dados possuem complexidade de tempo e espaço igual ou superior à quadrática (em relação ao número de objetos), tornando-as impraticáveis para conjuntos de dados grandes. Com o *clustering*, entretanto, em vez de aplicar uma técnica sobre todo o conjunto de objetos, esta pode ser aplicada sobre um conjunto pequeno de protótipos de grupo. Os resultados de uma

análise – dependendo do seu tipo, do número de protótipos e da representatividade dos protótipos – podem ser comparáveis aos que seriam obtidos se todos os objetos do conjunto fossem considerados (JAIN, 1991).

No presente trabalho, o modelo Escada aplica o processo de *cluster analysis* para agrupar fluxos de rede com demandas de envio e recebimento de dados semelhantes e, assim, identificar as demandas de rede típicas das transações realizadas por aplicações em execução em uma máquina virtual. A finalidade da *cluster analysis* neste trabalho pode ser resumida, portanto, como a caracterização do tráfego de rede de uma máquina virtual e suas aplicações. A Seção 3.2.3 descreve em maiores detalhes a utilização da *cluster analysis* pelo Escada. Na Seção 4.1 é apresentada uma avaliação comparativa para identificar o algoritmo de *clustering* mais adequado para o contexto deste trabalho.

2.5 TRABALHOS RELACIONADOS

Nesta seção são discutidos os principais trabalhos relacionados ao provisionamento de recursos computacionais e a previsão de demanda de rede concernentes ao contexto e proposta do presente trabalho.

Em (PFITSCHER; PILLON; OBELHEIRO, 2013) é proposto um modelo para diagnóstico do provisionamento de rede capaz de identificar se a largura de banda (LB) está subprovisionada, superprovisionada ou adequada. Esse modelo é baseado na monitoração do consumo de banda e da fila das interfaces de rede de uma máquina virtual (MV). Apesar do modelo ser capaz de identificar o estado de provisionamento de LB, ele não estima a LB nos casos de sub- e superprovisionamento. Escada (aqui proposto) vale-se da monitoração de fluxos para estimar a LB necessária em uma MV. Essa estimativa pode ser comparada com a LB alocada para diagnosticar o provisionamento de rede.

Cicada (LACURTS et al., 2014) é um mecanismo para predição da LB de enlaces virtuais sob o modelo *pipe* (MOGUL; POPA, 2012), em que são especificadas as capacidades dos enlaces entre pares de MVs. A monitoração do tráfego se baseia na análise dos fluxos entre MVs em um *datacenter*, sendo explicitamente desconsiderado o tráfego entre MVs e a rede externa (*hosts* na Internet). A matriz de tráfego é observada a intervalos regulares, e a LB prevista é dada por uma combinação linear de observações anteriores, com pesos que vão sendo ajustados automaticamente em função dos erros de predição. O mecanismo pode predizer tanto a LB média quanto a LB de pico durante um dado período. O trabalho adota a premissa de que a rede está suficientemente provisionada durante o período de monitoração, não tratando o caso em que ela está subprovisionada. Cicada também incorpora um algoritmo para mapeamento dos recursos virtuais na infraestrutura física. Escada tem um escopo mais estreito, concentrando-se no problema de estimação e previsão da LB necessária para uma dada MV, mas considera o tráfego externo e é capaz de fornecer estimativas acuradas mesmo quando a

rede está subprovisionada. Adaptar a abordagem do Escada para o modelo *pipe* seria uma tarefa trivial, bastando separar os fluxos de acordo com os seus pares origem-destino.

Alguns trabalhos, como (GONG; GU; WILKES, 2010; ENGELBRECHT; GREUNEN, 2015; NGUYEN et al., 2013; MOREIRA JR; OBELHEIRO, 2016), propuseram estratégias de predição do nível de utilização de recursos. Esses trabalhos são baseados em algoritmos de regressão (*e.g.*, auto-regressão, auto-correlação, ARMA, ARIMA, suavização exponencial) que consideram históricos de medições de utilização de recursos com o objetivo de prever valores futuros. Em contraste com o presente trabalho, esses trabalhos focam no provisionamento adequado de processador e memória. Além disso, nenhuma dessas estratégias considera o impacto do subprovisionamento na medição da demanda por capacidade dos recursos. O modelo Escada poderia ser usado em conjunto com uma das estratégias propostas nesses trabalhos para melhorar a qualidade do histórico da demanda de rede para previsão de necessidades futuras.

(POPA et al., 2012a; ANASTASI et al., 2016; MARCON et al., 2016) propuseram mecanismos e estratégias para oferecer garantias de LB em nuvens. Esses trabalhos assumem que os clientes de nuvens especificam a LB desejada no momento da reserva de recursos. Nossa proposta é ortogonal a esses trabalhos e os complementa auxiliando os clientes de nuvem na tarefa de determinar qual a LB necessária para que suas aplicações atinjam bom desempenho sem que a reserva incorra em custos supérfluos.

A Tabela 2 resume as características dos trabalhos relacionados de interesse ao presente trabalho. A coluna *objetivo* indica se o trabalho busca diagnosticar o provisionamento de recursos, prever as capacidades necessárias futuras, ou reservar capacidades dos recursos do provedor para atender a requisições de reserva dos clientes. A coluna *recursos* indica o recurso de interesse do trabalho e, no caso dos trabalhos que buscam a previsão, também o recurso do qual foram coletados dados que representem sua variação temporal. A coluna *método de previsão* indica os algoritmos ou mecanismos usados pelos trabalhos que buscam a previsão das capacidades necessárias.

2.6 CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo foram revisados os conceitos e trabalhos relacionados necessários ao entendimento do trabalho e seu contexto. A computação em nuvem foi conceitualizada como um modelo para permitir o acesso ubíquo, conveniente, sob demanda e remote a um *pool* compartilhado de recursos computacionais configuráveis que podem ser rapidamente provisionados e liberados com um mínimo de esforço de gerenciamento ou interação com o provedor (MELL; GRANCE, 2011) e ressaltou-se a possibilidade aos clientes de provisionar e ajustar a capacidade dos recursos computacionais segundo suas necessidades ao longo do tempo.

Em seguida, alguns termos importantes para as discussões seguintes do texto foram definidos. LB necessária foi apresentada como referindo-se a taxa de transmissão efetiva ne-

Tabela 2 – Resumo dos Trabalhos Relacionados.

Referência	Objetivo	Recursos	Método de Previsão
(PFITSCHER; PILLON; OBELHEIRO, 2013)	Diagnóstico	Rede CPU Memória	-
(LACURTS et al., 2014)	Previsão	Rede	Algoritmo de *Expert-Tracking*
(GONG; GU; WILKES, 2010)	Previsão	CPU	Transforma rápida de Fourier Cadeia de Markov
(ENGELBRECHT; GREUNEN, 2015)	Previsão	CPU Memória	Suavização Exponencial
(NGUYEN et al., 2013)	Previsão	CPU	Transformada de Wavelet
(MOREIRA JR; OBELHEIRO, 2016)	Previsão	CPU Memória	Suavização Exponencial ARIMA
(POPA et al., 2012a)	Reserva	Rede	-
(ANASTASI et al., 2016)	Reserva	Rede	-
(MARCON et al., 2016)	Reserva	Rede	-

cessária para que uma aplicação atinja desempenho adequado. A taxa de transmissão nominal disponível a uma MV foi denominada LB reservada. Os conceitos de subprovisionamento, superprovisionamento e provisionamento adequado foram descritos. Resumidamente, o subprovisionamento indica que o aumento da reserva de LB representa uma melhora relevante no desempenho da aplicação; o superprovisionamento indica folga excessiva de LB, a LB reservada pode ser diminuída sem impactar relevantemente o desempenho da aplicação; e o provisionamento adequado representa um meio termo entre sub- e superprovisionamento, onde o aumento da LB reservada não representa uma melhora relevante no desempenho e a diminuição prejudica o desempenho.

Outro conteúdo abordado neste capítulo foi a monitoração de fluxos de rede, que consiste em coletar e registrar atributos referentes aos fluxos de pacotes transitando nas interfaces de rede de uma máquina virtual, ou na rede de um modo geral. O estudo deste tipo de monitoração é justificado pela abstração por ela proporcionada, que representa bem as transações em rede das aplicações de uma máquina virtual, e que ao mesmo tempo não é intrusiva as aplicações, pois não requere a modificação das aplicações e tampouco o entendimento do funcionamento destas. Outros benefícios desse tipo de monitoração são evidenciados no capítulo a seguir.

Concluindo a parte de revisão de conceitos, é apresentado o processo de *cluster analysis*, que consiste sumariamente em agrupar um conjunto de objetos em múltiplos grupos de forma que os objetos de um mesmo grupo apresentem maior similaridade entre si do que a objetos de outros grupos. No presente trabalho a *cluster analysis* é aplicada com a finalidade caracterizar o tráfego de rede de uma máquina virtual e suas aplicações. O uso da *cluster analysis* no presente trabalho é detalhada no capítulo seguinte.

Na última seção deste capítulo foram discutidos os principais trabalhos relacionados ao provisionamento de recursos computacionais e a previsão de demanda de rede que contextualizam o presente trabalho dentro do estado da arte. O próximo capítulo apresenta o modelo

Escada para provisionamento de rede em máquinas virtuais em nuvens IaaS.

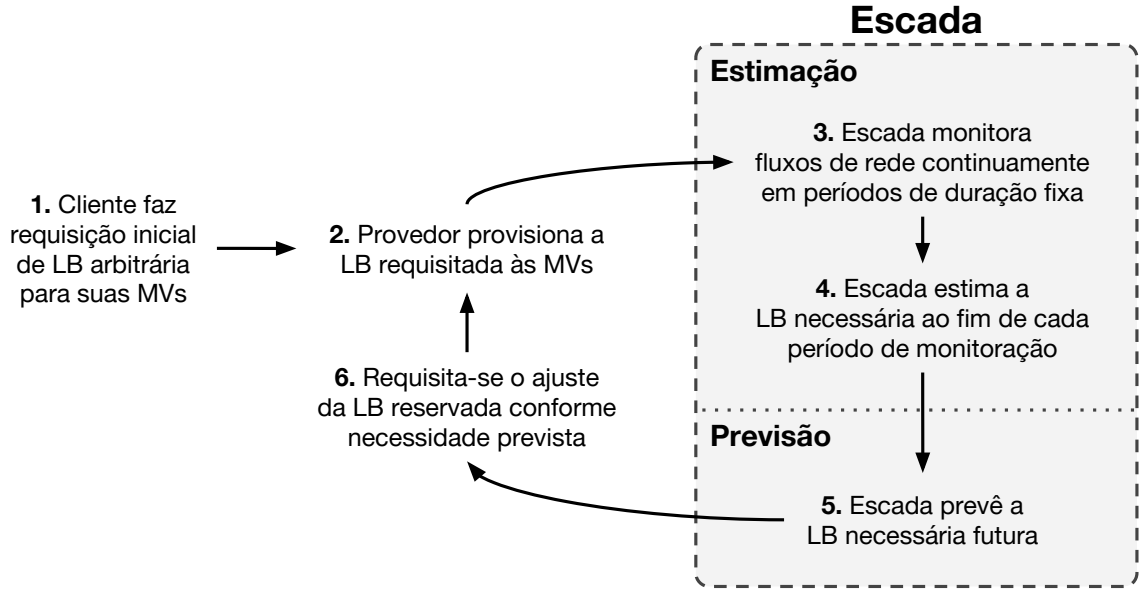
3 PROPOSTA

Neste capítulo é apresentada a proposta de um modelo para o provisionamento de rede de máquinas virtuais hospedadas em nuvens IaaS, principal objetivo do presente trabalho. A Seção 3.1 apresenta uma visão geral da abordagem, e as Seções 3.2 e 3.3 detalham os modelos de estimação e de previsão, respectivamente.

3.1 VISÃO GERAL

A arquitetura de alto nível do modelo proposto – denominado Escada – e seu fluxo de execução são apresentados na Figura 2.

Figura 2 – Arquitetura do modelo proposto Escada.



Fonte: autoria própria

O fluxo de execução do modelo começa com a requisição inicial do usuário por LBs arbitrárias para suas máquinas virtuais (MVs). Em seguida, o provedor atende a requisição e provisiona a rede das MVs com a largura de banda (LB) requisitada. A partir do momento que uma MV é instanciada, Escada monitora os fluxos de rede transitando em suas interfaces de rede virtuais continuamente em períodos de duração fixa (*i.e.*, um novo processo de monitoração é iniciado no fim de cada período de monitoração anterior). Para cada período de monitoração encerrado, Escada estima sua LB necessária e atualiza o histórico de estimativas do tráfego. Sempre que o histórico é atualizado, Escada prevê a LB necessária para o período seguinte. De posse da previsão, requisita-se o ajuste da LB reservada para o valor da LB necessária prevista

(normalmente acrescido de uma folga para suportar eventuais rajadas do tráfego, conforme discutido nas Seções 3.2 e 4.2) através de APIs oferecidas pelo provedor. Após essa requisição, o provedor reajusta a LB reservada da MV conforme requisitado e o fluxo de execução continua neste laço de repetição até que a instância da MV seja finalizada.

Técnicas e mecanismos para provisionar as MVs garantindo a LB reservada, assim como APIs para atribuição e ajuste de LB às MVs, são consideradas problemas ortogonais ao modelo proposto e, portanto, fora do escopo deste trabalho.

A arquitetura do Escada é dividida em duas partes: estimação e previsão, detalhadas nas Seções 3.2 e 3.3.

3.2 ESTIMAÇÃO DE TRÁFEGO

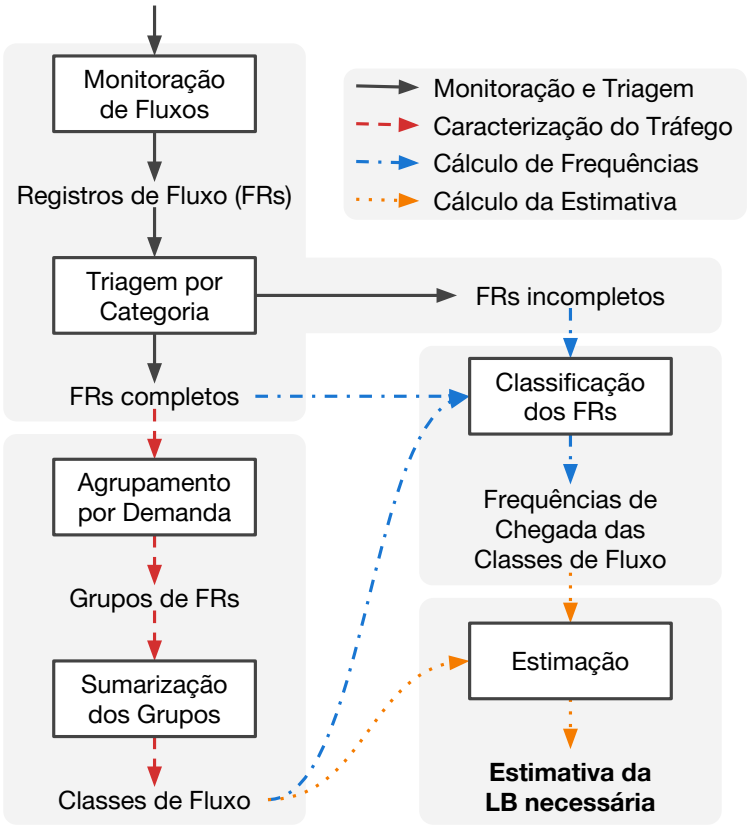
O modelo de estimação do Escada objetiva a estimação da LB necessária para a carga aplicada sobre uma MV em um dado período. Sua abordagem permite que sejam estimados valores próximos à LB necessária real em todos os níveis do provisionamento de rede. Como discutido anteriormente na Seção 2.2, a estimação de tráfego não fornece uma previsão da LB necessária no futuro, mas sim um histórico do tráfego que é posteriormente usado para a previsão. No presente trabalho, argumenta-se que gerar um histórico acurado do tráfego é essencial para prever acuradamente a demanda do tráfego futuro. As etapas da parte de estimação do Escada são mostradas na Figura 3 e descritas nas seções subsequentes.

3.2.1 Monitoração de Fluxos

O fluxo de execução da parte de estimação inicia pela monitoração de fluxos nas interfaces de rede da máquina virtual (MV) da qual deseja-se estimar a largura de banda (LB) necessária. Neste trabalho, essa monitoração é realizada usando o Argus (BULLARD, 2017). Após um período de monitoração, o Argus fornece um relatório do tráfego que consiste em um conjunto de registros de fluxos (*flow records* – FRs), sendo registrado um FR para cada fluxo (bidirecional) observado. Um FR consiste em um registro que traz diversas informações sobre um fluxo (vide Seção 2.3). As informações consideradas pelo modelo são:

- os instantes inicial e final em que o fluxo foi observado;
- os endereços de origem e destino das aplicações comunicantes (*i.e.*, endereço IP e porta);
- o protocolo de rede (*e.g.*, UDP, TCP);
- as *flags* que foram observadas em pelo menos um dos cabeçalhos dos pacotes do fluxo (*e.g.*, SYN, ACK, FIN); e
- a quantidade de bytes transmitidos em cada sentido do fluxo.

Figura 3 – Etapas da estimação de tráfego por Escada.



Fonte: autoria própria

Realizada a monitoração de fluxos, pode-se então realizar a caracterização do tráfego das aplicações. Caracterizar o tráfego consiste em identificar quais são as diferentes demandas de rede típicas dos fluxos das aplicações. A demanda de rede de um fluxo i é expressa pelo par:

$$\text{Demanda}_i = \{\text{quantidade de bytes enviados}_i, \text{quantidade de bytes recebidos}_i\} \tag{3.1}$$

formado pelos valores da quantidade de bytes transmitida em seus dois sentidos. A hipótese do Escada é que conhecendo as demandas típicas dos fluxos e a frequência com que fluxos de cada demanda ocorrem na MV, pode-se estimar a LB necessária da rede para minimizar a ocorrência de filas nas interfaces de rede da MV. O enfileiramento causa atraso na transmissão (e eventualmente a perda) de pacotes, prejudicando o desempenho da aplicação.

3.2.2 Triagem por Categoria

Os registros de fluxo (FRs) gerados pela monitoração podem ser categorizados como completos ou incompletos. Um FR é dito completo quando traz informações referentes a um fluxo que iniciou e terminou dentro do intervalo de monitoração. Quando um fluxo ocorre (em parte) fora do intervalo de monitoração (*i.e.*, iniciou antes e/ou terminou após o período de

monitoração), o FR que traz informações sobre este fluxo é dito incompleto, pois durante o período de monitoração observou-se apenas parte de sua demanda de rede. Para a finalidade de caracterização do tráfego apenas os FRs completos são de interesse, visto que somente esses possuem informações totais sobre a demanda de rede dos fluxos.

A quantidade de FRs referentes a fluxos observados parcialmente é influenciada por dois principais fatores. O primeiro, nível de provisionamento da rede, tem relação direta com a duração de um fluxo com demanda de rede fixa. Por exemplo, um fluxo com volume de dados de 10 MB levaria 8 segundos para ser transmitido em um enlace com largura de banda (LB) igual a 10 Mbps, e 80 s (ou 1 min 20 s) para ser transmitido em outro de capacidade igual a 1 Mbps. Portanto, considerando um período de monitoração de duração fixa (e uma carga estática), quanto mais subprovisionada a rede, maior a quantidade de fluxos observados parcialmente.

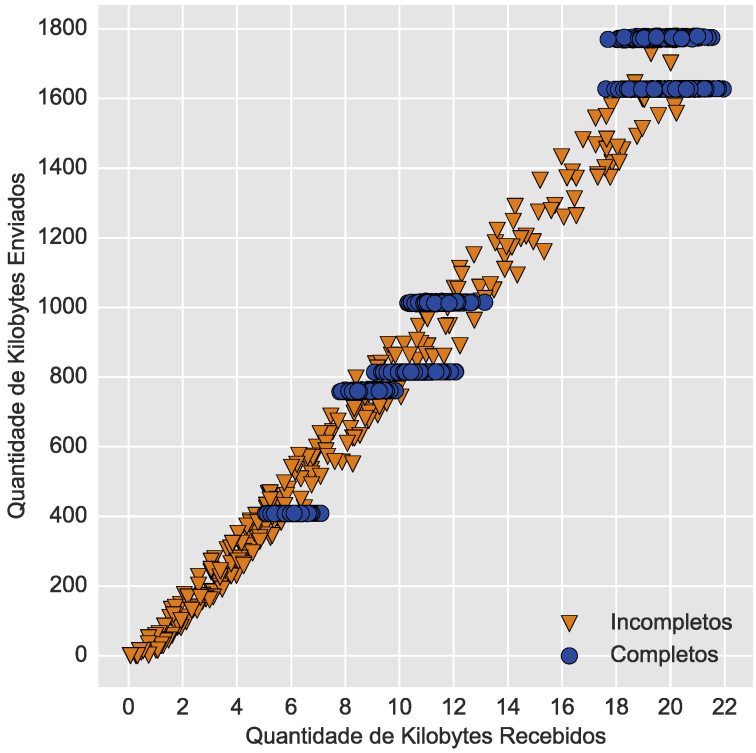
O segundo fator influenciador é a duração do período de monitoração. Se o período de monitoração for excessivamente curto (*i.e.*, menor que a duração típica de uma parcela significante do fluxos), os fluxos com duração mais longa que o período serão observados parcialmente, gerando FRs incompletos. Portanto, sugere-se ao usuário do modelo que escolha uma duração para o período de monitoração que ele espere ser suficiente longa para conter a maioria dos fluxos de sua aplicação. Por exemplo, um cliente de nuvem implementando um servidor HTTP tipicamente espera que sua aplicação não demore um pouco mais que alguns segundos para responder (completamente) a uma requisição de página. Neste caso, intui-se que um período de monitoração de alguns minutos deveria ser suficiente para garantir que uma maioria dos fluxos serão observados completamente. A investigação da duração adequada para o período de monitoração de forma a atender aplicações arbitrárias, com possivelmente fluxos de longa duração, é classificado como um trabalho futuro.

Quanto mais curto o período de monitoração, maior a dinamicidade na identificação da variação do tráfego. Entretanto, períodos excessivamente curtos podem impedir que alguns fluxos que *naturalmente* tenhma duração típica mais longa que o período sejam observados completamente (mesmo situações de provisionamento adequado ou superprovisionamento).

A etapa de triagem categoriza os FRs e os separa entre completos e incompletos. Os FRs de fluxos TCP são categorizados pelas *flags* observadas nos pacotes do fluxo. Caso as *flags* SYN (que indica o início de uma sessão TCP) e FIN (que indica o fim de uma sessão TCP) tenham sido usadas em alguns dos pacotes do fluxo, esse é categorizado como completo. Noutro caso, em que apenas uma ou nenhuma das duas *flags* seja observada, o fluxo TCP é categorizado como incompleto. Se a *flag* SYN não foi observada, conclui-se que o fluxo teve início anterior ao período de monitoração. Se a *flag* FIN não foi observada, conclui-se que o fluxo continuou além do período de monitoração ou foi abortado. Por sua vez, os FRs de fluxos UDP são todos categorizados como completos, visto que esse protocolo não apresenta sessões e todo datagrama transmitido representa uma transação/mensagem completa.

A Figura 4 apresenta a categorização do processo de triagem sobre um conjunto de FRs. Essa figura é referente a um período de subprovisionamento. Observa-se que os FRs completos formam grupos bem caracterizados, enquanto os incompletos estão espalhados ao longo do espaço entre a origem do gráfico e os grupos de FRs completos. Em virtude desta observação, a fase de caracterização do tráfego considera apenas os FRs completos.

Figura 4 – Observações de demanda de rede dos registros de fluxo categorizados pela etapa de triagem.



Fonte: autoria própria

3.2.3 Agrupamento por Demanda

Os programas voltados ao atendimento de requisições (*e.g.*, servidores HTTP, bancos de dados) normalmente possuem um conjunto finito de requisições que são realizadas regularmente. A fase de caracterização do tráfego tem por objetivo identificar as demandas de rede dessas requisições observando unicamente os fluxos transitando nas interfaces de rede, sem a necessidade de entender o funcionamento dos programas ou o que cada requisição representa para a aplicação do cliente. Essa metodologia permite ao Escada ser agnóstico aos programas executando na máquina virtual (MV).

Por experimentação, observou-se que requisições idênticas para uma mesma aplicação podem apresentar variações de quantidade de bytes recebidos e enviados (Figura 4). Isso ocorre

devido a fatores como o encapsulamento de mensagens de aplicação em segmentos TCP, perdas, reordenação de pacotes e ACKs atrasados (*delayed ACKs*). Essas variações, no entanto, não têm magnitude suficiente para descaracterizar a demanda das requisições.

Para administrar essas variações de demanda para requisições idênticas, Escada aplica a *cluster analysis* (Seção 2.4) para identificar grupos de fluxos (que representam requisições) com demandas de rede próximas. A etapa de agrupamento busca identificar grupos de fluxos com demandas de rede diferentes para cada um dos programas executando na MV. Essa etapa é realizada em duas partes. Primeiramente, os registros de fluxo (FRs) são agrupados quando possuem exatamente os mesmos endereços IP e portas de destino e protocolos de rede. Para Escada, cada tripla única $\{\text{endereço IP de destino}, \text{porta de destino}, \text{protocolo de rede}\}$ representa um "programa" diferente. Por exemplo, um servidor Apache HTTPD[1] com um *virtual host* configurado para atender à porta 80 e outro para atender à porta 443 é considerado pelo Escada como dois programas diferentes, pois utiliza portas diferentes.

Após o agrupamento por programa, cada grupo de fluxos de cada programa é subdividido (por um algoritmo de *clustering*) em grupos de fluxos com demanda de rede semelhante. Dessa forma, para cada requisição de cada programa busca-se identificar indiretamente um grupo de fluxos. O algoritmo de *clustering* escolhido para implementação do protótipo de avaliação do Escada foi o Mini Batch K-Means, disponível através da biblioteca SciKit-Learn (PEDREGOSA et al., 2011). Quando comparado com outros cinco algoritmos, Mini Batch K-Means apresentou o melhor compromisso entre complexidade de ajuste, complexidade de tempo e representatividade de agrupamento. Esse comparativo é apresentado na Seção 4.1. O número de grupos usado no algoritmo é selecionado automaticamente, de forma iterativa: o algoritmo é executado várias vezes com diferentes números de grupos, e o melhor agrupamento – aquele que minimiza a variância da distância intragrupo – é o escolhido. A Figura 5 mostra os grupos identificados automaticamente pela etapa de agrupamento dos FRs da Figura 4.

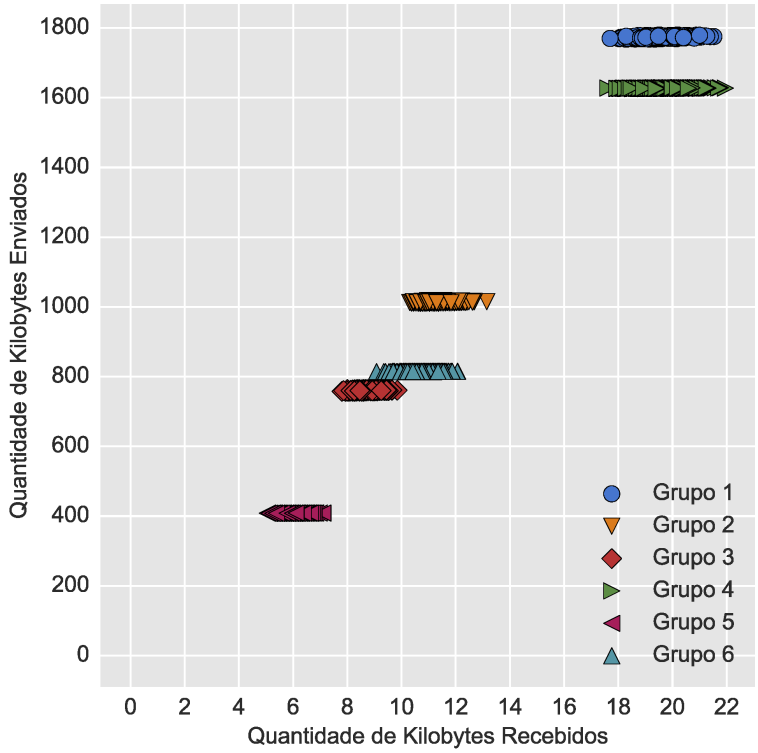
Como comentado anteriormente, essa etapa considera apenas os registros de fluxo categorizados como completos. A consideração dos incompletos acarretaria na formação de grupos que não representam adequadamente as demandas totais típicas da aplicação. Considerar somente os registros de fluxo completos (durante o agrupamento) é importante principalmente em casos onde a rede está subprovisionada, pois nesses, o percentual de registros de fluxo incompletos dentre todos os observados costuma ser maior do que em casos de provisionamento adequado e superprovisionamento. Logo, a triagem proposta nesse trabalho é essencial para obter grupos de fluxos com demandas representativas e boa acurácia na estimação de LB.

3.2.4 Sumarização dos Grupos

Esta última etapa da fase de caracterização do tráfego tem por objetivo criar protótipos para os grupos de registros de fluxo (FRs) identificados pela etapa de agrupamento. Para cada

[1] https://httpd.apache.org/

Figura 5 – Grupos de registros de fluxo identificados pelo agrupamento por demanda de rede.



Fonte: autoria própria

grupo de FRs é criado um protótipo que o resume. Um protótipo, ou doravante chamado classe de fluxo, é composto de sete informações, sendo as três primeiras a tripla de identificação de um programa: endereço IP de destino, porta de destino e protocolo de rede. As outras quatro informações expressam a demanda de rede típica da classe de fluxo, são estas: a quantidade média de bytes enviados em cada fluxo, a margem de erro da quantidade de bytes enviados em cada fluxo (com nível de confiança de 99%), a quantidade média de bytes recebidos em cada fluxo e a margem de erro da quantidade de bytes recebidos em cada fluxo (com nível de confiança de 99%).

A Tabela 3 exemplifica o resultado da etapa de sumarização e da fase de caracterização do tráfego sobre os grupos de FRs da Figura 5. A partir dessa tabela (e das classes por ela apresentada) conclui-se que existem dois programas em execução na MV sendo monitorada, uma operando na porta 8000 e outra na porta 9000. O primeiro programa (porta 8000) atende a requisições com demandas de envio típicas de aproximadamente 761 KB, 1 MB e 1,8 MB. O outro programa (porta 9000) atende a requisições com demandas de envio típicas de aproximadamente 410 KB, 818 KB e 1,63 MB.

Tabela 3 – Classes de Fluxo geradas pela etapa de sumarização dos grupos.

Classe	Endereço IP de Destino	Porta de Destino	Protocolo de Rede	Média Bytes Enviados ± Margem de Erro	Média Bytes Recebidos ± Margem de Erro
1	1.1.1.2	8000	TCP	761.070 ± 352	9.727 ± 213
2	1.1.1.2	8000	TCP	$1.013.472 \pm 600$	12.040 ± 238
3	1.1.1.2	8000	TCP	$1.775.569 \pm 960$	22.170 ± 562
4	1.1.1.2	9000	TCP	409.906 ± 167	7.330 ± 150
5	1.1.1.2	9000	TCP	817.707 ± 461	13.326 ± 410
6	1.1.1.2	9000	TCP	$1.633.532 \pm 404$	25.436 ± 379

Fonte: autoria própria

3.2.5 Classificação dos Registros de Fluxo

As etapas descritas nas subseções anteriores ajustam parcialmente o modelo de estimação através das classes de fluxo que caracterizam o tráfego dos programas em execução na MV. Para estimar a LB necessária é essencial também identificar com que frequência fluxos de cada uma das classes iniciam nas interfaces de rede. Normalmente, a taxa de chegada de fluxos de cada classe de uma máquina virtual varia mais frequentemente que a demanda de rede das classes dessa máquina virtual, visto que é comum a taxa de chegada variar ao longo de um dia (*e.g.*, popularidade das páginas) enquanto a variação de demanda de rede por fluxo costuma estar atrelada à modificação do programa (*e.g.*, criação de nova página em um servidor de páginas web). Escada realiza uma etapa de classificação sobre todos os registros de fluxo (FRs) observados pela monitoração às classes para identificar a taxa de chegada de cada classe de fluxo.

Como comentado na Seção 3.2.2, os FRs fornecidos no relatório da monitoração de fluxos podem ser categorizados como completos ou incompletos. Para a finalidade de caracterização do tráfego, apenas os FRs considerados completos são observados, visto que somente estes contêm informações totais de demanda de rede. Já para a finalidade de identificar as taxas de chegada das classes de fluxo, ambas as categorias dos FRs devem ser consideradas (ainda que de formas diferentes). A classificação dos FRs completos associa cada um deles à classe de fluxo (de mesmo programa, considerando os campos de identificação de programa) com demanda de rede mais próxima segundo a Distância Euclidiana, calculada através da Equação 3.2:

$$d_{i,a} = \sqrt{(R_a - R_i)^2 + (E_a - E_i)^2} \tag{3.2}$$

Sendo $d_{i,a}$ a distância entre um FR i e uma classe de fluxo a; R_a e E_a a quantidade média de bytes recebidos e enviados, respectivamente, da classe a; R_i e E_i a quantidade de bytes recebidos e enviados, respectivamente, pelo fluxo i.

A classificação dos FRs incompletos, por sua vez, não pode ser baseada na distância euclidiana entre as demandas de um fluxo e das classes, pois os FRs incompletos contêm

uma demanda de rede parcial do fluxo (apenas a quantidade de bytes observada durante a monitoração). Por exemplo, é possível que um fluxo com demanda total igual a $[1\,\mathrm{MB}, 10\,\mathrm{KB}]$ seja observado apenas pela metade, resultando em um FR incompleto com demanda parcial de aproximadamente $\{500\,\mathrm{KB}, 5\,\mathrm{KB}\}$. Se o tráfego for caracterizado por três classes com demandas médias iguais a (a) $\{100\,\mathrm{KB}, 1\,\mathrm{KB}\}$, (b) $\{600\,\mathrm{KB}, 6\,\mathrm{KB}\}$ e (c) $\{1\,\mathrm{MB}, 10\,\mathrm{KB}\}$, não é possível determinar categoricamente que quando observado por completo o FR pertenceria à classe (b) ou (c). Entretanto, como a demanda parcial registrada no FR tem valor superior à demanda da classe (a), conclui-se que o fluxo completo não pertenceria a essa classe.

Diante dessa problemática, Escada classifica FRs incompletos usando três cenários (otimista, pessimista e intermediário), e faz uma composição das três classificações para chegar à classificação final. Essa abordagem busca mitigar possíveis distorções no tráfego causadas, por exemplo, pelo subprovisionamento da rede e pela competição entre tráfegos UDP e TCP. No primeiro cenário, cada FR incompleto é classificado como pertencente à classe de mesmo programa com demanda de rede imediatamente superior à demanda parcial do fluxo. Esse cenário otimista representa a LB mínima que seria necessária para transmitir completamente todos os fluxos observados parcialmente, pois cada fluxo é associado à classe de menor demanda dentre as classes com demanda superior à do fluxo. Nesse cenário os FRs incompletos f_1, f_2 e f_3 da Figura 6 seriam atribuídos às classes c_a, c_b e c_c, respectivamente.

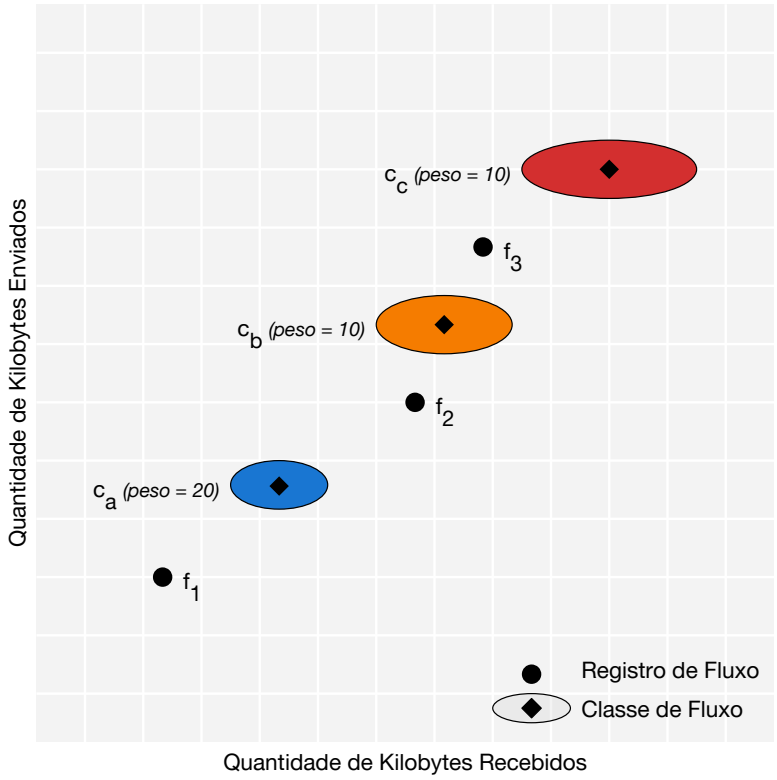
O segundo cenário considerado classifica todos os FRs incompletos como pertencentes à classe de maior demanda da mesmo programa. Esse cenário pessimista representa a LB máxima que seria necessária para transmitir completamente todos os fluxos observados parcialmente, pois todo fluxo é considerado como pertencente à classe de demanda de rede máxima do programa. Neste cenário todos os FRs incompletos da Figura 6 seriam atribuídos à classe c_c.

O terceiro (e último) cenário considerado associa parcelas dos FRs incompletos às classes. A parcela referente a cada classe é distribuída segundo uma ponderação feita sobre a taxa de chegada de fluxos completos de cada classe:

$$p_a = \frac{\lambda_a^c}{\sum_{i \in C_>} \lambda_i^c} \tag{3.3}$$

Sendo p_a o peso da classe a; λ_a^c a taxa de chegada de fluxos completos da classe a; $C_>$ o conjunto de todas as classes de fluxo com demanda superior à do fluxo sob classificação. Essa ponderação busca atribuir um peso maior para as classes que se observou um número maior de fluxos completos. Neste cenário, à classe c_a da Figura 6 seria atribuído $1/2$ do fluxo f_1; à classe c_b seria atribuído $1/4$ do fluxo f_1 e $1/2$ do fluxo f_2; e à classe c_c seria atribuído $1/4$ do fluxo f_1, $1/2$ do fluxo f_2 e todo o fluxo f_3. Esse cenário intermediário representa a LB que seria necessária para transmitir completamente todos os fluxos observados parcialmente caso a proporção entre as taxas de chegada observadas das classes fosse a mesma para os fluxos completos e para os incompletos.

Figura 6 – Exemplo de cenários de classificação de registros de fluxo incompletos.



Fonte: autoria própria

As taxas de chegada das classes de fluxo considerando os FRs incompletos são calculadas pela média aritmética das taxas calculadas nos três cenários discutidos. Como comentado anteriormente na Seção 3.2.3, o percentual de FRs incompletos dentre todos os observados costuma ser significativo nos casos de subprovisionamento. Portanto, a classificação correta dos FRs incompletos (além dos completos) é fundamental para estimar a LB necessária nos casos de subprovisionamento de rede. As taxas de chegada totais das classes de fluxo são calculadas somando as taxas calculadas considerando os FRs completos e as calculadas considerando os incompletos. A Tabela 4 mostra as taxas de chegada de fluxos calculadas para as classes da Tabela 3 pela etapa de classificação dos fluxos. A complexidade de tempo da etapa de classificação, em termos de pior caso de execução (todos os FRs são de uma única aplicação), é expressa por $O(nk)$, sendo n o número de fluxos registrados e k o número de diferentes classes identificadas (*i.e.*, todas as classes de fluxo são investigadas para classificar cada FR).

3.2.6 Estimação

Após o ajuste completo do mecanismo de estimação do Escada através dos processos de caracterização do tráfego e de cálculo das taxa de chegada das classes de fluxo, estima-se a

Tabela 4 – Frequências de chegada das classes de fluxo do exemplo.

Classe	Fluxos Completos	Fluxos Incompletos	Taxa de Chegada (fluxos/s)
1	436	28,85	3,10
2	719	17,94	4,91
3	697	51,88	4,99
4	1.318	70,13	9,25
5	419	25,15	2,96
6	1.206	132,72	8,92

Fonte: autoria própria

LB de envio necessária para a aplicação em execução na MV segundo a Equação 3.4:

$$LB_E = \sum_{i \in C} \lambda_i E_i \tag{3.4}$$

Sendo LB_E a LB de envio necessária; C o conjunto de todas as classes de fluxo; λ_i a taxa de chegada de fluxos da classe i; e E_i a quantidade média de bytes enviados por fluxos da classe i. De forma análoga, estima-se a LB de recepção necessária segundo a Equação 3.5:

$$LB_R = \sum_{i \in C} \lambda_i R_i \tag{3.5}$$

Sendo R_i a quantidade média de bytes recebidos por fluxos da classe i. A LB de necessária para atender as classes da Tabela 4 é estimada como 295,3 Mbps para envio e 3,6 Mbps para recepção.

Para garantir o bom desempenho das aplicações, durante o provisionamento de rede da MV recomenda-se a adição de uma folga à LB necessária para tolerar oscilações naturais da taxa instantânea de transmissão das aplicações. Neste trabalho recomenda-se – após observação experimental (Seção 4.2) e conforme (LACURTS et al., 2014) – provisionar a rede de forma que a LB necessária use em média entre 75% e 85% da capacidade. Por exemplo, para a estimativa de 295,3 Mbps, a LB a ser provisionada está entre 347,4 Mbps (85%) e 393,7 Mbps (75%). Essa folga pode ser ajustada pelo cliente de nuvem de modo a privilegiar o custo (reduzindo a folga) ou o desempenho (aumentando a folga).

3.3 PREVISÃO DE TRÁFEGO

O presente trabalho faz uso do algoritmo de *expert-tracking* proposto em (LACURTS et al., 2014) para prever valores futuros de LB necessária. Esse algoritmo foi avaliado anteriormente sobre um conjunto de dados coletado da HP Cloud Services. Um diferencial desse conjunto de dados é a coleta de métricas de usuários reais em uma nuvem do modelo IaaS; tornando-o mais representativo para o contexto do presente trabalho que dados de outros tipos

de redes de *datacenter*. A avaliação citada mostrou que o algoritmo é capaz de prever acuradamente a demanda de rede de aplicações em nuvens IaaS quando o histórico de tráfego é representativo das demandas reais passadas.

Esse algoritmo de previsão recebe, como entrada, uma série de matrizes de tráfego observadas anteriormente $(M_1, M_2, ..., M_n)$ que constituem o histórico de tráfego. O valor na linha i e coluna j de uma matriz expressa o número de *bytes* que a máquina virtual (MV) i enviou para a MV j em um determinado período. Essas matrizes apresentam valores coletados em intervalos passados igualmente espaçados. Cada matriz representa dados agregados para um período de T minutos, e os dados são coletados continuamente. Isso significa que, assumindo o instante t_0 como o inicial da coleta de dados, a matriz M_1 representa os dados agregados no período de t_0 até $t_0 + T$; a matriz M_2 representa os dados agregados de $t_0 + T$ até $t_0 + 2T$; e uma matriz M_i representa os dados agregados de $t_0 + (i-1) \cdot T$ até $t_0 + i \cdot T$.

O algoritmo de *expert-tracking* baseia-se na ideia de "identificar os melhores especialistas" proposta inicialmente em (HERBSTER; WARMUTH, 1998). Para prever a matriz de tráfego do período $n+1$, todas as matrizes de tráfego $(M_1, M_2, ..., M_n)$ observadas anteriormente são consideradas. O Algoritmo 1 apresenta um pseudo-código para este algoritmo. Segundo nossa análise, esse algoritmo apresenta complexidade $O(nm^2)$, sendo n o número de matrizes de tráfego fornecidas e m a dimensão de cada matriz (*i.e.*, o número de MVs sob monitoração). Nesse algoritmo cada uma das matrizes observadas anteriormente opera como um "especialista", recomendando que M_i é o melhor previsor para o período $n+1$. O algoritmo calcula a matriz previsão $\hat{M}_{n+1}$ como uma combinação linear ponderada das matrizes passadas:

$$\hat{M}_{n+1} = \sum_{i=1}^{n} w_i(n) \cdot M_i \tag{3.6}$$

sendo que $w_i(n)$ denota o peso dado a M_i para prever o período $n+1$; e o somatório dos pesos é 1, $\sum_{i=1}^{n} w_i(n) = 1$.

Algoritmo 1 Algoritmo para previsão de LB necessária (LACURTS et al., 2014).

Entrada: $M_1, M_2, ..., M_n$, as matrizes observadas anteriormente

1: $W = \{w_i\}$, a sequência de $n-1$ pesos existentes (usados para calcular $\hat{M}_n$)
2: **for** $w_i \in W$ **do**
3: $\quad w_i = w_i \cdot e^{-L(i,n)}$
4: $W = \{min(w_i) \cdot 0.5\} + W$
5: $Z = \sum_{i=1}^{n} w_i$
6: **for** $w_i \in W$ **do**
7: $\quad w_i = w_i / Z$
8: $\hat{M}_{n+1} = \sum_{i=1}^{n} w_i \cdot M_i$

Saída: $\hat{M}_{n+1}$, matriz de previsão de tráfego

O algoritmo ajusta os pesos *online*, não há a noção de dados de treinamento e testes. A

cada passo, os pesos são atualizados de acordo com a seguinte regra:

$$w_i(n+1) = \frac{1}{Z_{n+1}} \cdot w_i(n) \cdot e^{-L(i,n)} \tag{3.7}$$

sendo que $L(i,n)$ denota a "perda de especialista" i para o período n; e $Z_n + 1$ normaliza a distribuição para que a soma dos pesos seja igual a 1. O termo perda significa a discrepância (ou erro) entre o especialista i e o valor verdadeiro.

A norma euclidiana entre M_i e M_n é usada como função perda L. Tratando ambas essas matrizes como vetores, $\vec{M}_i$ e $\vec{M}_j$, respectivamente, o erro relativo normalizado é:

$$L(i,n) = E_{l^2} = \frac{\|\vec{M}_i - \vec{M}_n\|}{\|\vec{M}_n\|} \tag{3.8}$$

isto é, a norma dos erros individuais dividido pela norma dos dados reais observados. A norma euclidiana de um vetor v com n elementos é calculada segundo a equação:

$$\|v\| = \sqrt{\sum_{i=1}^{n} v_i^2} \tag{3.9}$$

A cada passo, um novo peso deve ser adicionado ao conjunto de pesos: são usados $n-1$ pesos para prever $\hat{M}_n$, mas n para prever $\hat{M}_{n+1}$. Para realizar isso, é inserido um pequeno peso no início do vetor de pesos, correspondente a 50% do menor peso já presente no conjunto (linha 4 do Algoritmo 1). Essa inserção tem o efeito de deslocar todos os outros pesos: w_0 torna-se w_1, w_1 torna-se w_2, e assim por diante. Para entender por que este processo é adequado, considera-se uma indexação negativa alternativa para os pesos: $w_n = w_{-1}, w_{n-1} = w_{-2}, \ldots, w_1 = w_{-n}$. Com essa indexação, w_{-1} representa o peso atribuído à matriz mais recente, w_{-2} representa o peso dado à segunda matriz mais recente, e assim por diante. A inserção adiciona um novo peso para a n-ésima matriz mais recente, ou, em outras palavras, a matriz mais antiga. Como não havia anteriormente uma matriz referente ao n-ésimo período passado (para a previsão de $\hat{M}_n$ havia apenas $n-1$ matrizes), esse período deve começar com um peso pequeno, visto que sua "opinião de especialista" ainda não foi testada.

Nota-se que este algoritmo depende das medições realizadas anteriormente. Portanto, caso essas não representem valores reais da demanda do tráfego – o que normalmente ocorre em períodos de subprovisionamento – suas previsões tendem a não ser acuradas em relação às demandas futuras. Essa observação reforça o argumento de que gerar um histórico acurado do tráfego é essencial para prever acuradamente a demanda do tráfego futuro. O presente trabalho trata essa limitação através de um novo modelo de estimação capaz de oferecer estimativas suficientemente acuradas em todos os níveis de provisionamento (*i.e.*, subprovisionamento, superprovisionamento e provisionamento adequado).

3.4 CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo foi apresentada a proposta de um modelo para o provisionamento de rede de máquinas virtuais em nuvens IaaS. O modelo proposto (Escada) é composto por duas partes: estimação e previsão. Durante a parte de estimação, Escada monitora a ocorrência de fluxos nas interfaces de rede de uma máquina virtual para identificar o volume de dados das transações das aplicações em execução. Os fluxos observados são triados entre completos e incompletos. Um algoritmo de *clustering* é usado para identificar grupos de fluxos completos com demandas de rede semelhantes, e o tráfego de rede é classificado por um novo algoritmo segundo esses grupos. Esse novo algoritmo, aqui proposto, classifica fluxos completos e incompletos diferentemente, permitindo identificar demandas de rede não atendidas dos fluxos incompletos e estimar a LB necessária para a máquina virtual. Em seguida, na parte de previsão, um histórico das estimativas de LB necessária é usado por um algoritmo de *expert-tracking* para prever as LBs necessárias futuras. No capítulo seguinte o modelo aqui proposto é avaliado experimentalmente.

4 AVALIAÇÃO

No presente capítulo são descritos os experimentos realizados para desenvolver e avaliar o modelo Escada. O experimento inicial realiza um comparativo entre algoritmos de *clustering* para o propósito de agrupamento de fluxos de rede. O segundo experimento analisa o comportamento do Escada em diferentes níveis de provisionamento e avalia sua capacidade de oferecer estimativas acuradas da largura de banda necessária. O último experimento compara a qualidade das previsões dadas por Escada com aquelas fornecidas por Cicada, proposto em (LACURTS et al., 2014).

4.1 COMPARATIVO DE ALGORITMOS DE *CLUSTERING*

Este experimento tem por objetivo identificar o algoritmo de *clustering* mais adequado dentre os da biblioteca Scikit-learn (PEDREGOSA et al., 2011) para identificar os diferentes tipos de fluxos transitando em interfaces de rede. Os algoritmos considerados no comparativo são: Agglomerative Clustering, Birch, DBSCAN, K-Means, Mini Batch K-Means, e Mean Shift.

A Seção 4.1.1 considera critérios qualitativos de comparação entre os algoritmos de *clustering*. A Seção 4.1.2 descreve uma métrica de avaliação quantitativa e a aplica sobre os algoritmos em seis cenários diferentes. A Seção 4.1.3 discorre sobre as conclusões que podem ser obtidas a partir dos resultados da aplicação dos critérios e da métrica sobre os algoritmos de *clustering*.

4.1.1 Avaliação Qualitativa

Esta primeira etapa do comparativo tem por objetivo avaliar: a complexidade de ajuste dos algoritmos; e a viabilidade destes para agrupamento de conjuntos de dados com milhares de amostras. Dois critérios são propostos para avaliar a complexidade de ajuste:

1. Quantidade de parâmetros: para ajustar um algoritmo, é necessário considerar diferentes combinações de valores de seus parâmetros. O número de combinações a serem consideradas cresce com complexidade fatorial em função do número de parâmetros. Portanto, quanto *menor* o número de parâmetros, *melhor* a complexidade de ajuste de um algoritmo.

2. Objetividade dos parâmetros: descreve o nível de relação de um parâmetro com o objetivo do agrupamento. Por exemplo, o número de grupos que deseja-se encontrar é considerado um parâmetro altamente objetivo. Já o raio máximo de um grupo é pouco objetivo, pois depende de informações sobre o posicionamento das amostras no espaço. Quanto *maior* o nível de objetividade dos parâmetros, *melhor* a complexidade de ajuste de um algoritmo.

Um critério é proposto para avaliar a viabilidade dos algoritmos:

1. Complexidade de Tempo: Todos os algoritmos sob estudo possuem complexidade de tempo polinomial. Entretanto, a complexidade de tempo quadrática no número de amostras, por exemplo, pode tornar o tempo total de execução excessivamente longo para agrupar milhares de amostras quando uma restrição temporal é imposta sobre o ajuste do modelo de estimação. Quanto *menor* a complexidade de tempo do algoritmo, *maior* a probabilidade deste ser viável para uso pelo modelo.

A Tabela 5 apresenta os valores dos critérios para cada algoritmo. As colunas da tabela apresentam, respectivamente: o algoritmo; a quantidade de parâmetros; o nível de objetividade típica dos parâmetros; e a complexidade de tempo do algoritmo. Os valores para os critérios de complexidade de ajuste foram atribuídos pelos autores do presente trabalho. As funções de complexidade de tempo foram obtidas de (XU; WUNSCH et al., 2005; ZHANG; RAMAKRISHNAN; LIVNY, 1996; CHENG, 1995), sendo n o número de amostras e k o número de grupos.

Tabela 5 – Valores dos critérios qualitativos sobre os algoritmos de *clustering*.

Algoritmo	**Quantidade**	**Objetividade**	**Complexidade de Tempo**
Agglomerative Clustering	3	Média	$O(n^2)$
Birch	2	Baixa	$O(nk)$
DBSCAN	2	Baixa	$O(n\log n)$
K-Means	1	Alta	$O(nk)$
Mean Shift	1	Baixa	$O(kn\log n)$
Mini Batch K-Means	1	Alta	$O(nk)$

Fonte: autoria própria

As principais considerações sobre os dados apresentados na Tabela 5 são:

- K-Means e Mini Batch K-Means são considerados os algoritmos mais facilmente ajustáveis, pois requerem o ajuste de um único parâmetro altamente objetivo, o número de grupos a serem formados;
- Birch, DBSCAN e Mean Shift são os algoritmos mais difíceis de ajustar, em função da baixa objetividade de seus parâmetros;
- DBSCAN possui a menor complexidade de tempo, $O(n\log n)$;
- Agglomerative Clustering possui a maior complexidade de tempo, $O(n^2)$;
- Os demais algoritmos possuem complexidade de tempo adequada para a tarefa de agrupar milhares de amostras.

4.1.2 Avaliação Quantitativa

A segunda etapa do comparativo tem por objetivo avaliar a qualidade dos agrupamentos gerados pelos algoritmos de *clustering* para a estimação do total de kilobytes transmitidos. A infraestrutura do experimento consistiu em duas máquinas virtuais (MVs) (servidora e cliente) provisionadas sobre o MMV Xen, cada uma com: 1 VCPU; 512MB de RAM; 20GB de espaço em disco; interface de rede com largura de banda de saída limitada a 10MB/s; e sistema operacional Ubuntu 14.04. Um servidor de páginas com 25 páginas foi implantando na MV servidora. Os tamanhos das páginas criadas foram sorteados segundo uma distribuição exponencial de probabilidade com média igual a $2,5$MB, e são expostos na Tabela 6.

Tabela 6 – Tamanhos das páginas do servidor com 25 páginas.

Página	Tamanho	Página	Tamanho	Página	Tamanho	Página	Tamanho
1	68KB	8	1,2MB	15	2,4MB	22	5,9MB
2	162KB	9	1,5MB	16	2,4MB	23	6,9MB
3	238KB	10	1,5MB	17	2,6MB	24	7,9MB
4	245KB	11	1,6MB	18	2,9MB	25	9,4MB
5	381KB	12	1,6MB	19	3,3MB		
6	582KB	13	1,9MB	20	3,5MB		
7	682KB	14	2,0MB	21	4,1MB		

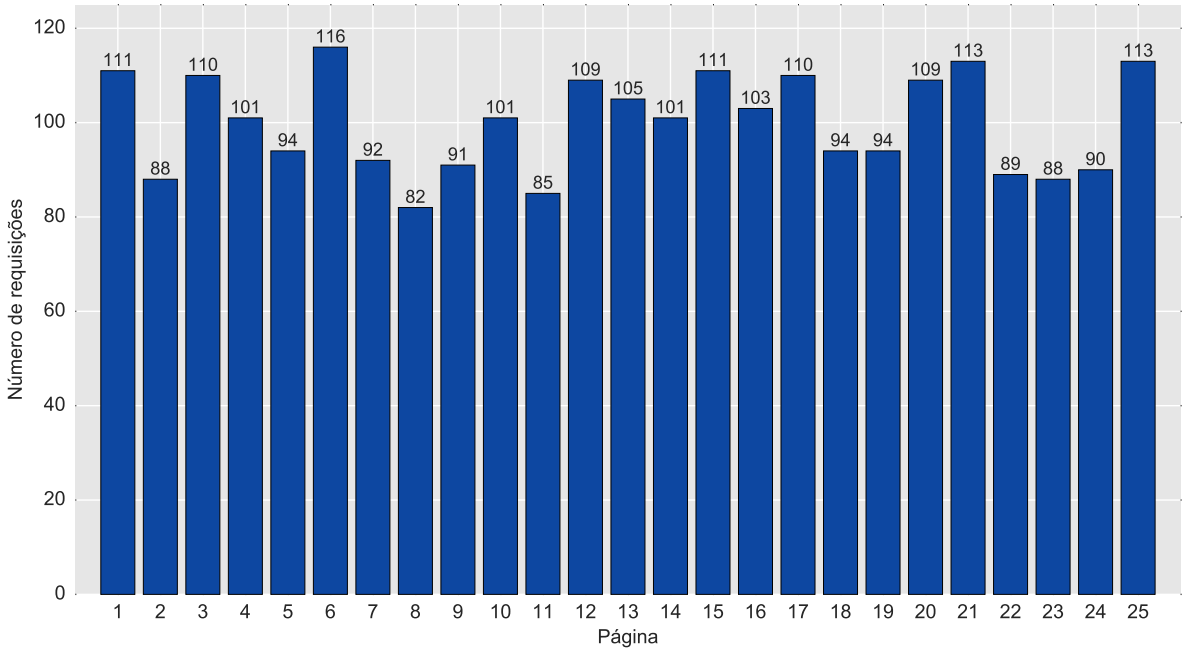
Fonte: autoria própria

Para gerar tráfego de rede e avaliar a representatividade dos grupos de fluxos formados pelos algoritmos de *clustering*, três conjuntos de requisições de páginas (U, E, N) foram criados. Cada conjunto é composto de 2500 requisições, e cada uma destas requisita uma das 25 páginas do servidor. As figuras 7–9 apresentam o número de requisições (ou popularidade) de cada página em cada um dos conjuntos de requisições.

Na criação do conjunto U, para cada requisição sorteou-se um número entre 1 e 25 (representando uma página conforme a Tabela 6), segundo uma distribuição uniforme de probabilidade. Isto resulta em todas as páginas terem aproximadamente o mesmo número de requisições, ou em outras palavras, mesma popularidade (vide Figura 7). No conjunto E, sorteou-se páginas segundo uma distribuição exponencial de probabilidade com média igual a $25/3$; isso faz com que, em geral, quanto menor for o número de uma página, maior seja a sua popularidade (vide Figura 8). No conjunto N, sorteou-se páginas segundo uma distribuição normal de probabilidade com média $25/2$ e desvio-padrão $25/6$, isto resulta em que páginas com números mais próximos da média (e.g. 12 e 13) tenham maior popularidade, e páginas com números mais afastados da média (e.g. 1 e 25) tenham menor popularidade (vide Figura 9).

O processo de avaliação de cada algoritmo consistiu em primeiramente formar os grupos de fluxos usando um dos três conjuntos de requisições e, posteriormente, avaliar a representatividade dos grupos formados usando-os para estimar o total de bytes transmitidos pelos

Figura 7 – Número de requisições a cada página no conjunto U.



Fonte: autoria própria

outros dois conjuntos de requisições individualmente. Cada par ordenado (A,B) de conjuntos de requisições é denominado um cenário de teste, sendo A o conjunto usado para formar os grupos e B o conjunto do qual deseja-se estimar o total de bytes transmitidos. Por exemplo, o par (U,E) representa o cenário em que a obtenção do modelo de carga (*clustering* e sumarização) foi feita com base nas requisições geradas usando a distribuição uniforme e a avaliação foi feita usando o conjunto de requisições gerado com a distribuição exponencial. Ao total tem-se seis cenários de teste: (U,E), (U,N), (E,U), (E,N), (N,U) e (N,E).

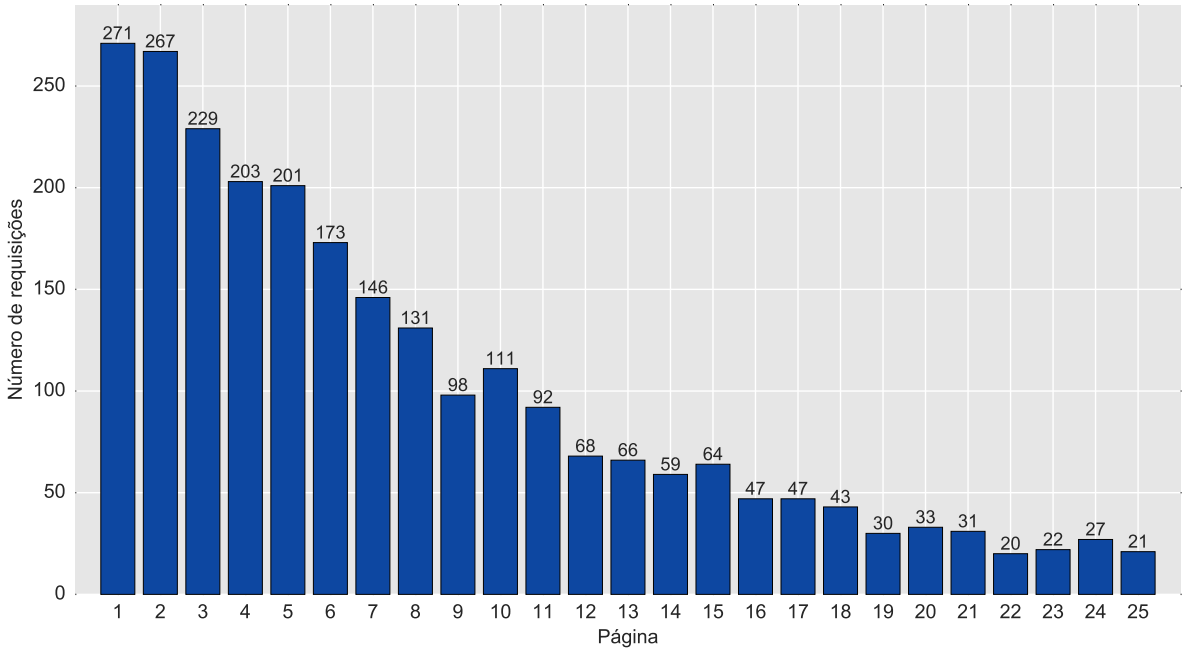
Como a qualidade e representatividade do agrupamento obtido dependem dos parâmetros dos algoritmos, múltiplas combinações de valores de parâmetros foram explorados para cada algoritmo em cada cenário de teste. A métrica escolhida para representar a qualidade dos agrupamentos foi o Erro Percentual Médio (MPE). Esta métrica é calculada obtendo a média dos erros percentuais de estimação das classe de fluxo:

$$\text{MPE} = \frac{100\%}{k} \sum_{i=1}^{k} \frac{a_i - f_i}{a_i} \tag{4.1}$$

Sendo k o número de classes, a_i o total real de bytes transmitidos por fluxos da classe i, e f_i o total estimado de bytes transmitidos pela classe i.

A Figura 10 mostra a qualidade do melhor agrupamento encontrado com cada algoritmo. Como a métrica MPE pode apresentar valores negativos, nesta figura utilizou-se o seu

Figura 8 – Número de requisições a cada página no conjunto E.



Fonte: autoria própria

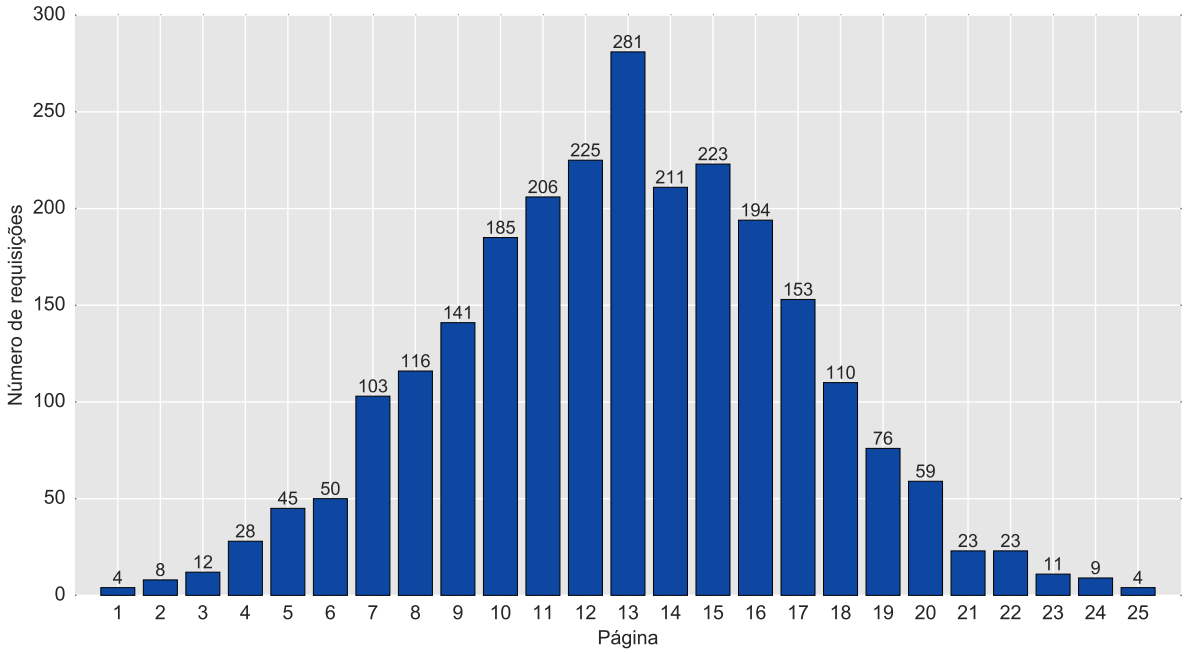
valor absoluto, pois neste estudo tem-se interesse pela magnitude do erro e não pela sua natureza (*i.e.*, superestimativa ou subestimativa). Cada gráfico de barras da Figura 10 compara os melhores agrupamentos encontrados pelos algoritmo para um determinado cenário de teste, segundo descrito em cada subtítulo. Quanto menor a barra, melhor o algoritmo. A escala do eixo de |MPE| é variável entre os gráficos. Os valores de |MPE| para os algoritmos Birch e Mini Batch K-Means nos cenários (E,U) e (N,U), respectivamente, são maiores que zero, mas não é possível visualizar suas barras em função de serem muito pequenas proporcionalmente às escalas dos gráficos.

A Tabela 7 apresenta a média dos valores absolutos de MPE obtidos pelos algoritmos considerando todos os cenários de teste; quanto menor a média, melhor o algoritmo. Os algoritmos estão ordenados em ordem crescente da métrica de erro.

As principais considerações sobre os resultados da avaliação quantitativa são:

- Todos os algoritmos obtiveram baixo percentual de erro em todos os cenários de teste, sendo o maior valor igual a 1.26%, reforçando a viabilidade do uso de algoritmos de *clustering* para identificar os diferentes tipos de fluxos que costumam transitar nas interfaces de rede de uma máquina virtual;
- Mini Batch K-Means obteve a menor média de |MPE| entre os algoritmos e o menor

Figura 9 – Número de requisições a cada página no conjunto N.



Fonte: autoria própria

Tabela 7 – Algoritmos em ordem crescente da média de |MPE|. Quanto menor a média, melhor o algoritmo.

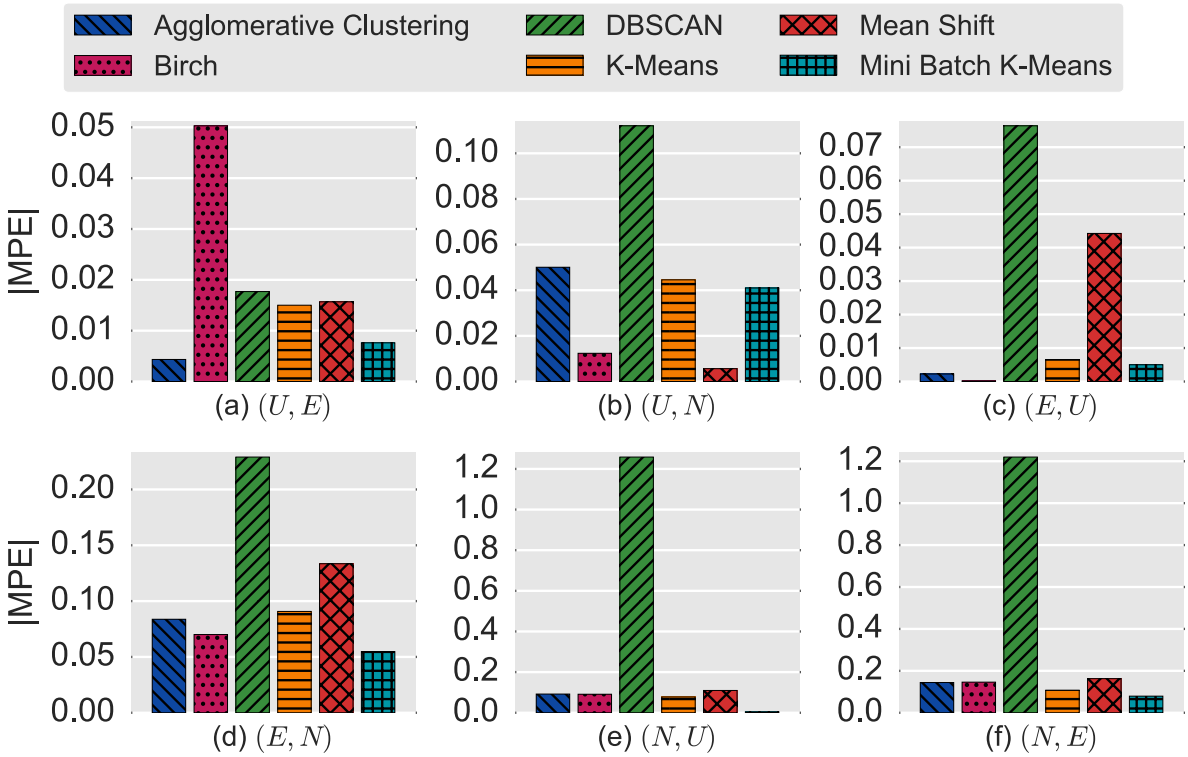
Algoritmo	**Média de |MPE|**
Mini Batch K-Means	0,032
K-Means	0,057
Birch	0,062
Agglomerative Clustering	0,063
Mean Shift	0,079
DBSCAN	0,486

Fonte: autoria própria

|MPE| em metade dos cenários de teste (d–f);

- DBSCAN obteve a maior média de |MPE| entre os algoritmos e o maior |MPE| em cinco dos seis cenários de teste (b–f);
- Os valores de |MPE| do K-Means são maiores que os do Mini Batch K-Means em todos os cenários de teste. Resultado esperado, visto que este último é uma evolução do primeiro.

Figura 10 – |MPE| de cada algoritmo para cada cenário de teste. Quando menor a barra, melhor o algoritmo.



Fonte: autoria própria

4.1.3 Conclusão do Experimento

Considerando os resultados das avaliações qualitativa e quantitativa, o algoritmo Mini Batch K-Means apresenta-se como a melhor opção de algoritmo de *clustering* para o propósito do trabalho. Este algoritmo necessita de um único parâmetro altamente objetivo; possui complexidade de tempo adequada; e apresentou a melhor média de |MPE| para os cenários de teste. O protótipo do Escada desenvolvido para sua avaliação usa o algoritmo de Mini Batch K-Means. Dos resultados desse experimento também conclui-se que o uso do processo de *cluster analysis* é viável para identificar as demandas típicas dos fluxos de rede transitando nas interfaces de rede de máquinas virtuais. Todos os algoritmos estudados foram capazes de gerar agrupamentos bastante representativos em todos os cenários de teste, o maior |MPE| foi de apenas 1.26%.

DBSCAN apresenta-se como o algoritmo menos indicado para o propósito do trabalho. Apesar de apresentar a menor complexidade de tempo, esse algoritmo possui alta complexidade de ajuste, pior qualidade de agrupamento na maioria dos cenários de teste e pior média de |MPE|. Agglomerative Clustering apresentou agrupamentos com qualidade adequada, mas a complexidade de tempo pode tornar seu uso inviável para conjuntos de amostras com milhares

de fluxos. Os demais algoritmos apresentam qualidades de agrupamento pouco piores que Mini Batch K-Means, e podem ser considerados para agrupamento de fluxos de rede. A principal contraindicação ao uso de Birch e Mean Shift está na baixa objetividade dos seus parâmetros, o que significa que ajustar ambos os algoritmos para obter agrupamentos adequados pode ser uma tarefa custosa. K-Means apresenta a melhor complexidade de ajuste (juntamente com Mini Batch K-Means) e complexidade de tempo adequada, mas possui desempenho inferior ao Mini Batch K-Means na tarefa de agrupamento de fluxos de rede, tornando a utilização do último preferencial.

4.2 VARIAÇÃO DO NÍVEL DE PROVISIONAMENTO

Este experimento tem por objetivo avaliar a qualidade das estimativas dadas pelo modelo proposto Escada quando a largura de banda (LB) reservada no período de monitoração varia entre valores subprovisionados, adequados e superprovisionados. Para este experimento foram criadas quatro cargas de trabalho, cada uma com páginas de tamanhos e taxas de requisição diferentes. As páginas das cargas foram implantadas em um servidor Apache HTTPD[1] com dois *virtual hosts*, um HTTP e outro HTTPS. A Tabela 8 apresenta o tamanho, a taxa de requisição e o protocolo de aplicação (HTTP ou HTTPS) de cada página de cada carga de trabalho. Realizou-se também experimentos com cargas de trabalho de outras aplicações – como FTP, SSH, e banco de dados SQL – mas seus resultados não são apresentados por serem semelhantes aos observados no presente experimento. Os tamanhos das páginas foram escolhidos aproximando-se de tamanhos de páginas dos websites mais populares da Internet[2]. As frequências de requisições de cada página foram escolhidas para explorar diferentes distribuições de popularidade (*e.g.*, páginas possuem popularidade uniforme, páginas menores são mais populares).

A infraestrutura do experimento consistiu em duas máquinas físicas (hospedeira e cliente) e uma MV (servidora). Cada máquina física possui 2 processadores Intel Xeon E5520 (4 *cores* por processador), 24 GB de RAM, 64 GB de armazenamento SSD e uma interface de rede Gigabit Ethernet. Ambas estão configuradas com o sistema operacional Debian 7 (*kernel* Linux versão 3.2.0-4), e adicionalmente, a máquina hospedeira executa o monitor de máquinas virtuais Xen versão 4.1. A MV servidora é virtualizada na hospedeira com 4 VCPUs, 4 GB de RAM, 4 GB de armazenamento SSD e LB da interface de rede variando de acordo com o cenário de teste do experimento. A máquina cliente faz as requisições das cargas de trabalho ao servidor HTTPD implantado na MV servidora, usando a ferramenta httperf (MOSBERGER; JIN, 1998).

Para cada uma das cargas de trabalho, o modelo de estimação do Escada foi executado em sete cenários de provisionamento. Em cada um desses cenários o valor da LB reservada

[1] https://httpd.apache.org/
[2] http://www.alexa.com/topsites

Tabela 8 – Cargas de trabalho para o experimento de variação da LB reservada.

Carga 1

Página	Tamanho	λ (req/s)	Protocolo
1	750 KB	3	HTTP
2	1 MB	5	HTTP
3	1,75 MB	5	HTTP
4	400 KB	9	HTTPS
5	800 KB	3	HTTPS
6	1,6 MB	9	HTTPS

Carga 2

Página	Tamanho	λ (req/s)	Protocolo
1	100 KB	25	HTTP
2	1 MB	7	HTTP
3	4 MB	3	HTTP
4	500 KB	10	HTTPS
5	1,5 MB	5	HTTPS
6	2 MB	5	HTTPS

Carga 3

Página	Tamanho	λ (req/s)	Protocolo
1	100 KB	20	HTTP
2	200 KB	25	HTTP
3	500 KB	20	HTTP
4	400 KB	25	HTTPS
5	500 KB	15	HTTPS
6	600 KB	15	HTTPS

Carga 4

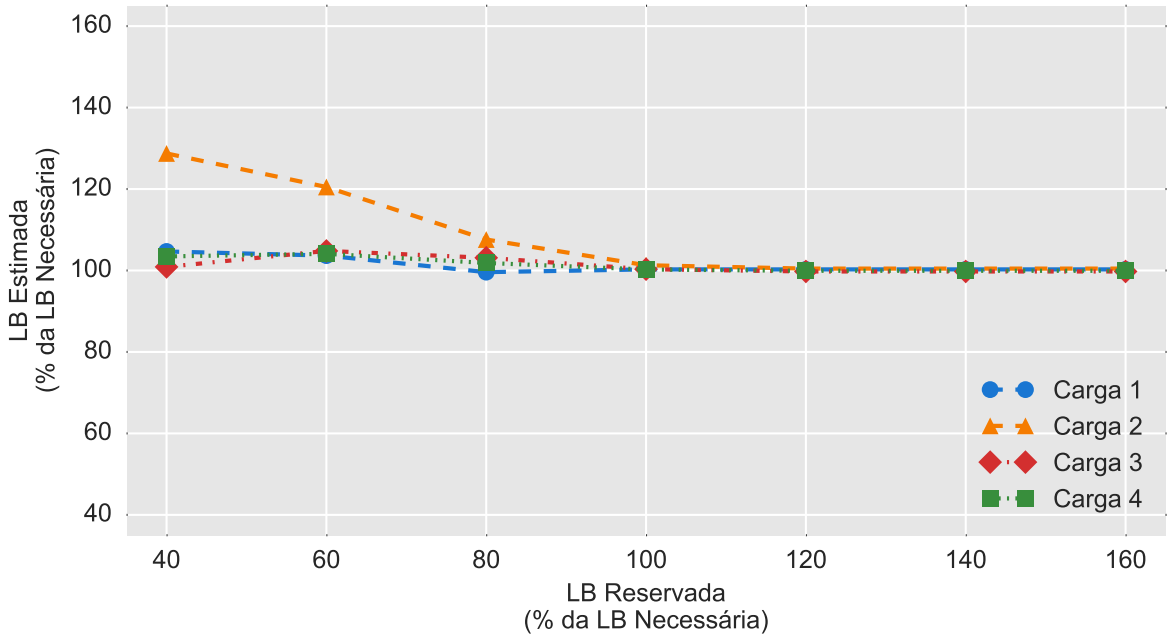
Página	Tamanho	λ (req/s)	Protocolo
1	100 KB	20	HTTP
2	200 KB	20	HTTP
3	500 KB	15	HTTP
4	400 KB	20	HTTPS
5	500 KB	15	HTTPS
6	600 KB	15	HTTPS

Fonte: autoria própria

é um dentre os seguintes: 40%, 60%, 80%, 100%, 120%, 140% e 160% da LB necessária da carga. Para determinar a LB necessária de uma carga, esta foi executada com LB nominal de 1 Gbps disponível (*i.e.*, um cenário de superprovisionamento). Essa variação da LB reservada busca reproduzir os estados de subprovisionamento, provisionamento adequado e superprovisionamento e avaliar a qualidade das estimativas fornecidas pelo Escada. As Figuras 11 e 12 apresentam as estimativas de LB necessária e o tempo médio de resposta para cada carga de trabalho em cada cenário de reserva de LB. Cada curva do gráfico representa os valores para uma carga de trabalho. Cada observação indica a LB estimada pelo modelo proposto ou o tempo médio de resposta das requisições ao servidor HTTPD (eixo y) quando executado sob determinada LB reservada (eixo x). O tempo de resposta reflete a duração de cada transação web, do início da requisição até o fim da resposta. Nos gráficos, os valores de LB são expressos como um percentual da LB necessária (quando a LB estimada ou reservada for igual à LB necessária esse percentual é de 100%).

Na Figura 11, observa-se que as estimativas de LB para os cenários onde a LB reservada é igual ou superior à LB necessária apresentam erros percentuais de pequena magnitude. Constata-se também que os cenários de subprovisionamento (40%, 60% e 80%) resultam em erros maiores, mas ainda menores dos que os que seriam obtidos pela medição da taxa de transmissão da MV. Esse aumento era esperado visto que a rede fica saturada, o que por sua vez distorce o tráfego imprevisivelmente, favorecendo alguns tipos de fluxos em detrimento de outros. O maior erro observado, uma superestimação de aproximadamente 25%, ocorreu para a

Figura 11 – LB necessária estimada variando a LB reservada no período de monitoração.

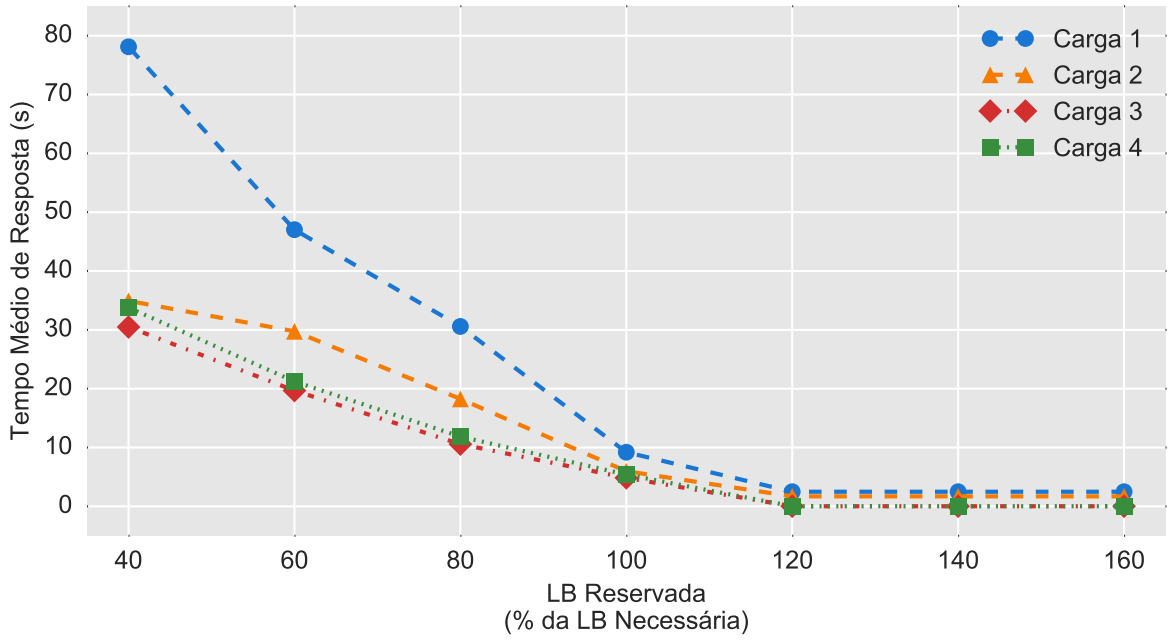


Fonte: autoria própria

Carga 2, no cenário onde a LB reservada é igual a 40% da necessária. Neste cenário, o erro de estimação do método que mede a taxa de transmissão da MV seria de 60%. Mesmo no cenário mais desfavorável, Escada consegue produzir uma estimativa útil para o provisionamento adequado da rede. Portanto, pode-se concluir que Escada é capaz de fornecer boas estimativas da LB necessária independentemente do nível atual de provisionamento da rede.

Considerando o tempo médio de resposta da aplicação (Figura 12), conclui-se que a LB reservada em 120% da LB necessária de cada carga representa um estado de provisionamento adequado. Dá-se essa conclusão pois o aumento da capacidade a partir deste valor não resulta em redução relevante no tempo de resposta; e a redução da capacidade – para, por exemplo, 100% da LB necessária – aumenta relevantemente o tempo de resposta. Observa-se que a LB reservada mais adequada (dentre os valores experimentados) tem valor superior à LB necessária (em 20%). Isso era esperado, pois a LB necessária expressa um valor adequado da taxa de transmissão *efetiva* média da aplicação, enquanto a LB reservada refere-se a taxa de transmissão *nominal* disponível à MV. Nesse contexto, a LB reservada deve apresentar uma folga capaz de tolerar oscilações naturais da taxa de transmissão instantânea, de forma que a aplicação atinja sua taxa de transmissão média necessária.

Figura 12 – Tempo médio de resposta das requisições de páginas quando da variação da LB reservada.



Fonte: autoria própria

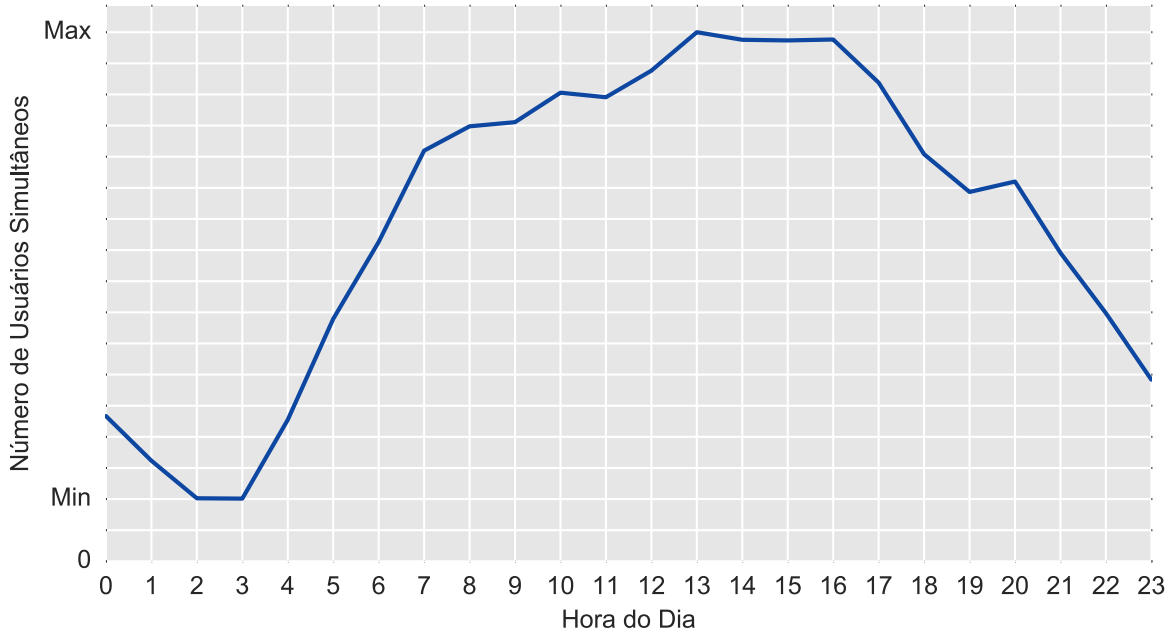
4.3 PROVISIONAMENTO DE REDE

Este experimento tem por objetivo avaliar a qualidade das estimativas e previsões de LB necessária do Escada e compará-las às oferecidas pelo Cicada (LACURTS et al., 2014). A aplicação escolhida para o experimento foi *Rice University Bidding System* (RUBiS), um *benchmark* que simula cargas de requisições realizadas a um site de busca e compra de produtos (similar ao eBay[3]). A variação de demanda de rede ao longo do tempo foi realizada através da variação do número de usuários simultâneos simulados pelo RUBiS. Seguindo trabalhos anteriores (LACURTS et al., 2014; BACKSTROM; KLEINBERG; KUMAR, 2009), a função do número de usuários simulados tem um comportamento fortemente diurnal, seguindo aproximadamente uma onda senoidal ao longo do dia (com um *jitter* aleatório). A Figura 13 exemplifica o comportamento dessa função. Segundo (LACURTS et al., 2014) e nossa avaliação, Cicada é capaz de se ajustar a este tipo de variação de carga quando a LB reservada não limita a taxa de transmissão da aplicação (*i.e.*, superprovisionamento ou provisionamento adequado).

A infraestrutura do experimento consistiu em três máquinas físicas (hospedeira e clientes) e uma MV (servidora). Cada máquina física possui 2 processadores Intel Xeon E5-2630L (6 *cores* por processador, e 2 *threads* por *core*), 32 GB de RAM, 250 GB de armazenamento

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

Figura 13 – Variação do Número de Usuários Simulados ao Longo do Dia.



Fonte: autoria própria

HDD (7200 rpm) e uma interface de rede Gigabit Ethernet. Todas foram configuradas com o sistema operacional Debian 7 (*kernel* Linux versão 3.2.0-4), e adicionalmente, a máquina hospedeira executa o monitor de máquinas virtuais Xen versão 4.1. A MV servidora é virtualizada na hospedeira com 12 VCPUs, 4GB de RAM, 4GB de armazenamento e LB da interface de rede variando de acordo com o caso de teste do experimento. As máquinas clientes simulam os clientes do *benchmark* RUBiS realizando requisições ao servidor RUBiS implantado na MV servidora.

Para avaliar e comparar os modelos Cicada e Escada, cada um deles foi executado por 504 períodos ou 21 ciclos (504 períodos ÷ 24 períodos por ciclo) da função da carga de trabalho. A ideia é que cada período representa uma hora do dia (e cada ciclo um dia). Entretanto, para manter a duração do experimento dentro de um limite factível, cada período teve duração de 2 minutos e 30 segundos. Assim cada ciclo teve duração de 1 hora e a execução completa de cada modelo durou 21 horas. Essa duração dos períodos não deve alterar o comportamento geral dos resultados. A duração do período de monitoração foi também escolhida por ser suficientemente longa para conter a maioria dos fluxos típicos gerados pela aplicação em estudo. Ao final de cada período, anotou-se as LBs necessárias estimada e prevista pelo modelo em execução e ajustou-se a LB reservada para 125% do valor previsto.

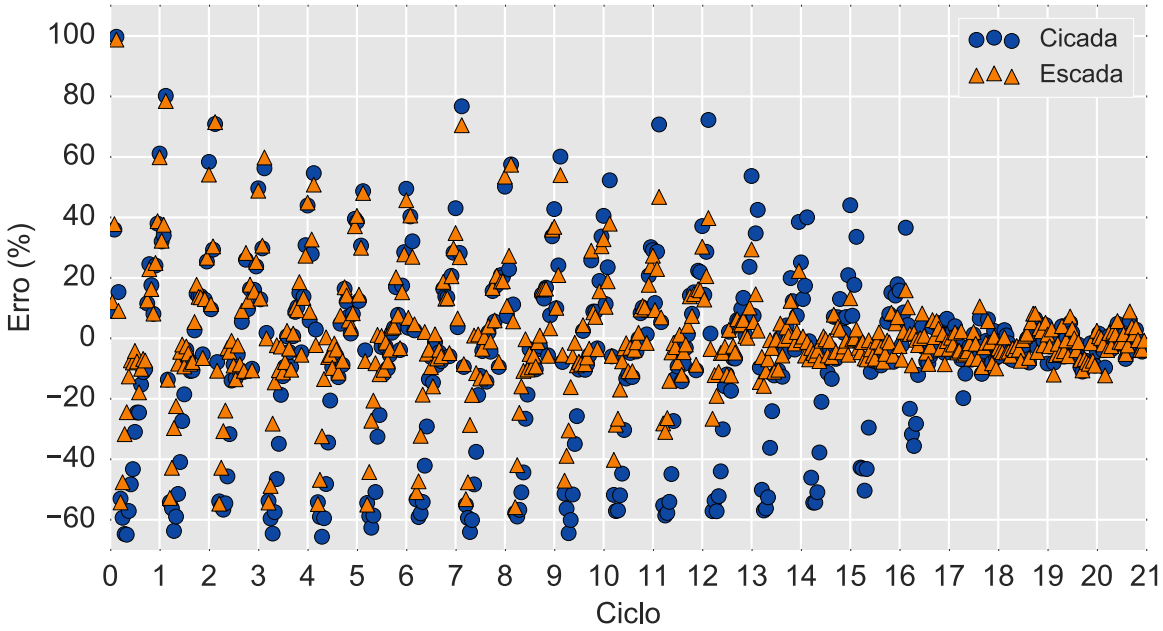
Reservou-se 125% da LB necessária prevista – em vez de 120%, como sugerido pelos resultados da Figura 12 – em conformidade com as avaliações realizadas sobre o modelo Cicada

em (LACURTS et al., 2014), visto que o Cicada é aqui também avaliado. Neste experimento, as previsões foram calculadas considerando todas as estimativas de LB necessária obtidas nos períodos anteriores. Opcionalmente, poderia-se usar uma janela de estimativas passadas, possivelmente descartando períodos antigos que recebam pesos muito baixos (*e.g.*, abaixo de um limiar $\varepsilon = 10^{-6}$).

Para avaliar a acurácia dos valores estimados e previstos pelos modelos, eles são comparados a valores reais de LB necessária obtidos executando a carga de trabalho com LB reservada à MV fixada em 1 Gbps (superior à LB necessária em todos os períodos). A Figura 14 apresenta o erro relativo percentual (ERP) das previsões de LB necessária dos modelos em relação aos valores de referência. Nos 9 ciclos iniciais ($0-8$), Cicada e Escada apresentam ERPs de previsão máximos semelhantes, $+99{,}8\%$ para Cicada e $+98{,}8\%$ para Escada. Com relação aos erros de previsão mínimos, Cicada apresenta ERPs de maior magnitude com $-65{,}6\%$, enquanto Escada chega até $-55{,}8\%$. Isso significa que Cicada chega a subprovisionar a rede em até 10 pontos percentuais além que Escada, limitando mais severamente a taxa de transmissão efetiva da máquina virtual. A partir do décimo ciclo (9), Escada começa a obter ERPs com menor magnitude (do que as observadas até então) e alcança um estado estável – com previsões de ERP de pequena magnitude (valor absoluto menor que 20%) – no décimo quinto (14) ciclo de execução. Cicada, por sua vez, começa a obter ERPs de menor magnitude apenas no décimo sexto ciclo (15) de execução. Esse modelo converge para um estado estável de previsão apenas no décimo nono (18) ciclo de execução do experimento, no qual suas previsões passam a apresentar ERPs absolutos menores que 20%. Escada inicia a convergência para estado estável em 2/3 do tempo de Cicada. Escada converge completamente para um estado estável de previsões acuradas em aproximadamente 79% ($15 \div 19$) do tempo necessário ao Cicada. Intui-se que essa convergência mais rápida do Escada está associada com a qualidade das estimativas fornecidas pelo modelo, como será observado a seguir na Figura 16.

Para compreender melhor o comportamento de convergência para o estado de previsão acurada do Escada, na Figura 15 compara-se os erros de previsão desse modelo com previsões obtidas aplicando os valores de referência ao algoritmo de *expert-tracking*. Essa aplicação permite verificar o comportamento do algoritmo de *expert-tracking* no melhor caso para a carga de trabalho do experimento. Observa-se que Escada apresentou erros semelhantes aos das previsões de referência em todos os períodos de execução do experimento. Na aplicação de referência, o algoritmo de *expert-tracking* demorou as mesmas quantidades de ciclos que Escada para iniciar a convergência e para alcançar o estado estável de previsão. Com base nesses resultados, conclui-se que Escada fornece estimativas de LB necessária tão úteis quanto as medições de referência para o ajuste do algoritmo de *expert-tracking*. Esse é um bom resultado, pois na prática é inviável trabalhar apenas com as medições de referência, haja vista que esses dados só podem ser obtidos com a rede superprovisionada, uma restrição que o Escada busca explicitamente contornar. Algumas modificações que poderiam ser estudadas para permitir que o algoritmo de

Figura 14 – Erros Relativos Percentuais das Previsões de Cicada e Escada.



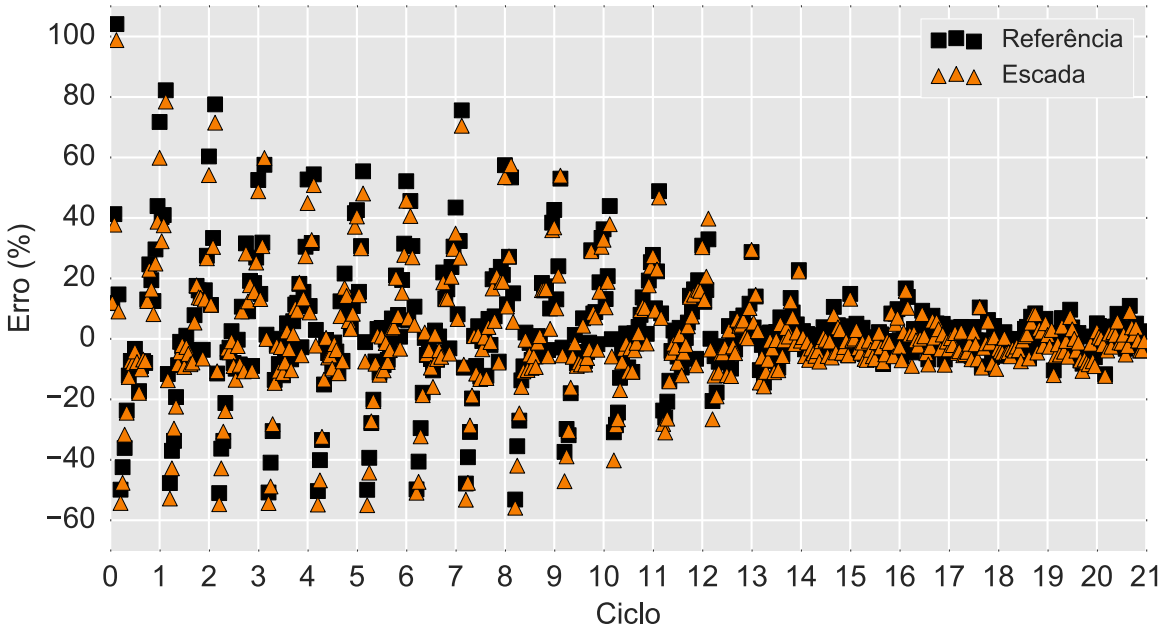
Fonte: autoria própria

expert-tracking convirja mais rapidamente para o estado estável de previsões são descritas na Seção 5.2.

A Figura 16 contrasta o ERP das estimativas de LB necessária dos modelos. Cicada subestima a LB necessária com magnitude significante (valor absoluto maior que 20%) frequentemente. Essas subestimativas estão relacionadas aos períodos de subprovisionamento de rede, visto que o método de estimação de Cicada não é capaz de observar completamente a LB necessária nesses períodos. A partir do décimo sétimo ciclo (18), o algoritmo de previsão do Cicada passa a prever valores mais acurados de LB necessária, evitando o subprovisionamento da rede e permitindo que as estimativas sejam também mais acuradas. A média do ERP desse modelo foi $-8{,}9\%$, e seus valores mínimo e máximo foram $-56{,}1\%$ e $+5{,}0\%$ (respectivamente). Escada gerou estimativas acuradas durante toda a execução do experimento. Os erros de previsão (principalmente os que causam o subprovisionamento da rede) têm um impacto relativamente menor sobre a estimação do Escada. O ERP desse modelo foi de em média $-2{,}0\%$, e variou entre um mínimo de $-13{,}4\%$ e um máximo de $+6{,}8\%$. Dessas observações, conclui-se que Escada possui maior robustez na estimação de LB necessária.

Com o objetivo de auxiliar na compreensão do comportamento dos modelos ao longo da execução do experimento, a Figura 17 mostra as estimativas de LB necessária passadas e as LBs reservadas por Cicada durante os 24 primeiros períodos e as contrasta com os valores de LB necessária medidos na execução de referência. A LB necessária estimada para cada período

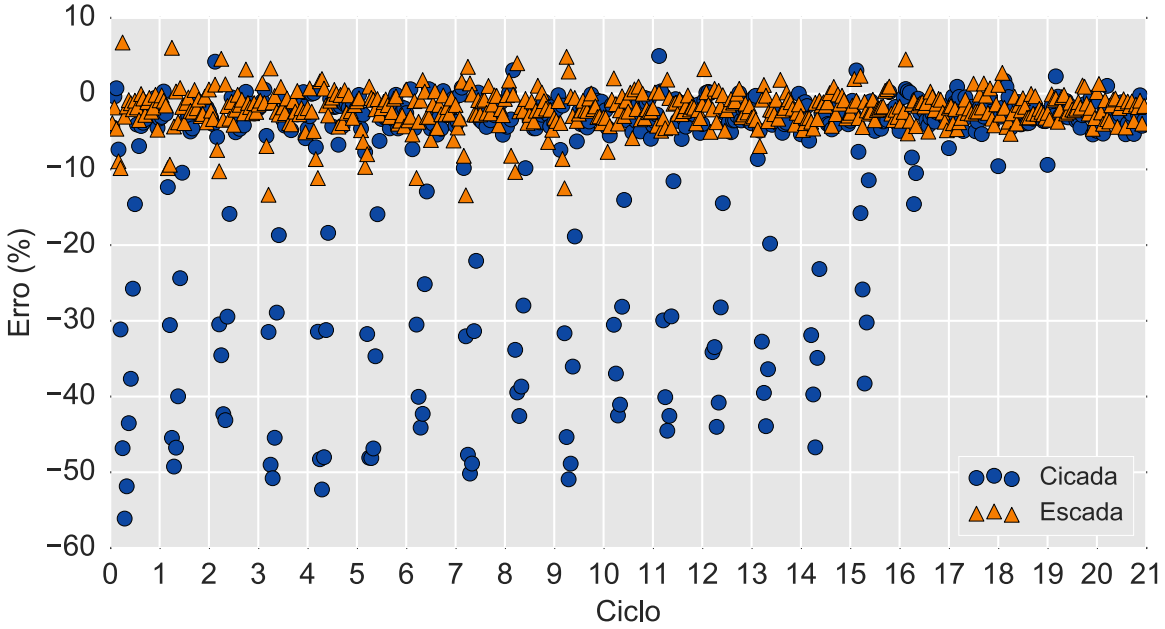
Figura 15 – Erros Relativos Percentuais das Previsões de Escada e do algoritmo de *expert-tracking* com os valores de referência.



Fonte: autoria própria

é obtida ao fim do período. A LB reservada de cada período, por sua vez, é igual a 125% do valor previsto ao fim do período imediatamente anterior, antes que o próprio período inicie. Repara-se que Cicada não prevê (ao fim do período 3) o crescimento da curva de LB necessária com a intensidade correta e acaba por subprovisionar a rede durante o período 4. Esse tipo de erro é esperado no início da execução do experimento, pois o histórico de estimativas ainda é muito pequeno para guiar adequadamente a previsão. Como Cicada não consegue estimar acuradamente a LB necessária quando a rede está subprovisionada, o algoritmo de *expert-tracking* não toma conhecimento do crescimento real da LB necessária. Em função disso, o algoritmo prevê apenas um leve crescimento para o período 5, que é acrescido em 25% no provisionamento da rede de acordo com o método de reserva do experimento. A folga de 25% acrescida à previsão do período 5 permite que Cicada observe um percentual pouco maior da LB necessária, resultando em uma estimativa um pouco maior que a anterior para o período 6. Cicada repete esse padrão até o período 14, aumentando suas previsões segundo uma curva quadrática. No período 14, a LB reservada supera a LB necessária (que alcançara seu platô de aproximadamente 22 Mbps), finalizando uma série de 10 períodos de subprovisionamento. Nota-se que o algoritmo de *expert-tracking* acaba por prever o crescimento da LB necessária, ainda que com comportamento bastante diferente do crescimento real da curva, graças ao método de reserva de LB. Cicada apresenta um comportamento semelhante a esse discutido – de subestimar a LB

Figura 16 – Erros Relativos Percentuais das Estimativas de Cicada e Escada.

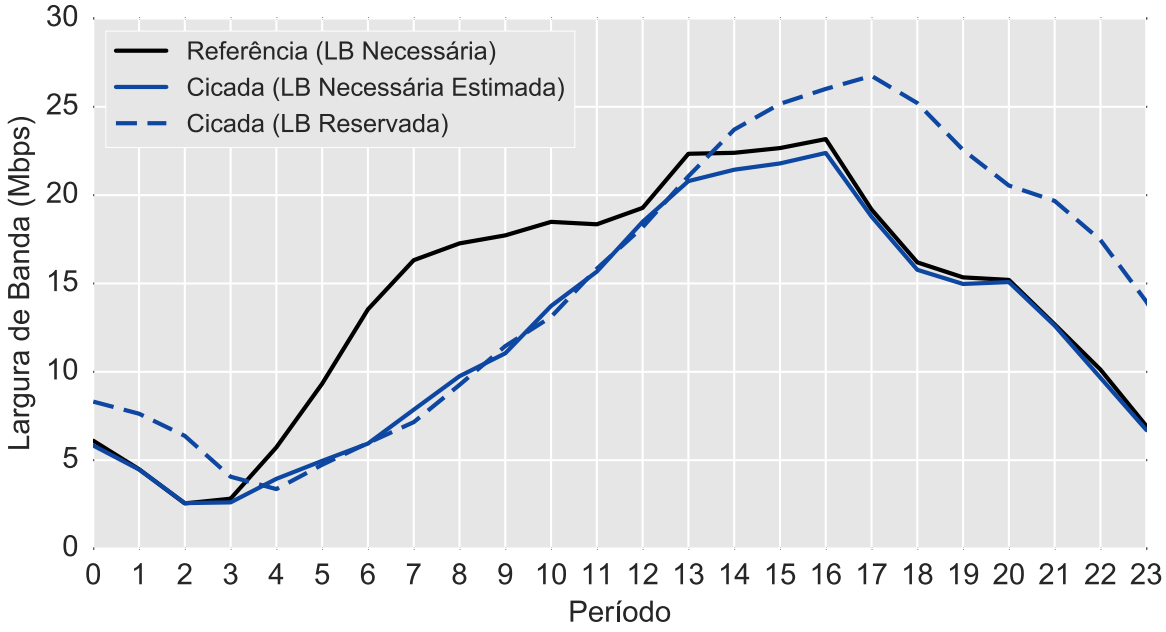


Fonte: autoria própria

necessária e subprovisionar a rede em cerca de 10 dos 24 períodos de um ciclo – em todos os ciclos de execução até o momento em que o algoritmo de *expert-tracking* se ajusta e passa a prever a LB necessária acuradamente.

Com relação ao Escada, a Figura 18 contrasta suas estimativas de LB necessária e suas LBs reservadas durante os 24 primeiros períodos do experimento com os valores de LB necessária medidos na execução de referência. Semelhantemente à execução do Cicada, Escada acaba por subprovisionar a rede no período 4, pois o histórico de estimativas era muito pequeno para prever o crescimento da curva. Entretanto, Escada tem como diferencial a capacidade de estimar acuradamente a LB necessária mesmo durante períodos de subprovisionamento. Nota-se que as estimativas do Escada nos períodos de subprovisionamento são próximas da LB necessária (e superiores à LB reservada). O histórico acurado de estimativas fornecido ao algoritmo de *expert-tracking* permite que Escada deixe de subprovisionar a rede após 4 períodos. Escada estima acuradamente a LB necessária em todos os períodos (em contraste com 14 dos 24 períodos de Cicada) e subprovisiona a rede em 4 dos 24 períodos (em contraste com 10 dos 24 períodos de Cicada). Escada apresenta um comportamento de estimação semelhante ao discutido em todos os ciclos de execução, não importando a saúde do ajuste do algoritmo de *expert-tracking* (conforme também observado na Figura 16).

Figura 17 – Estimativas de LB necessária e LBs reservadas por Cicada nos primeiros 24 períodos de execução do experimento.



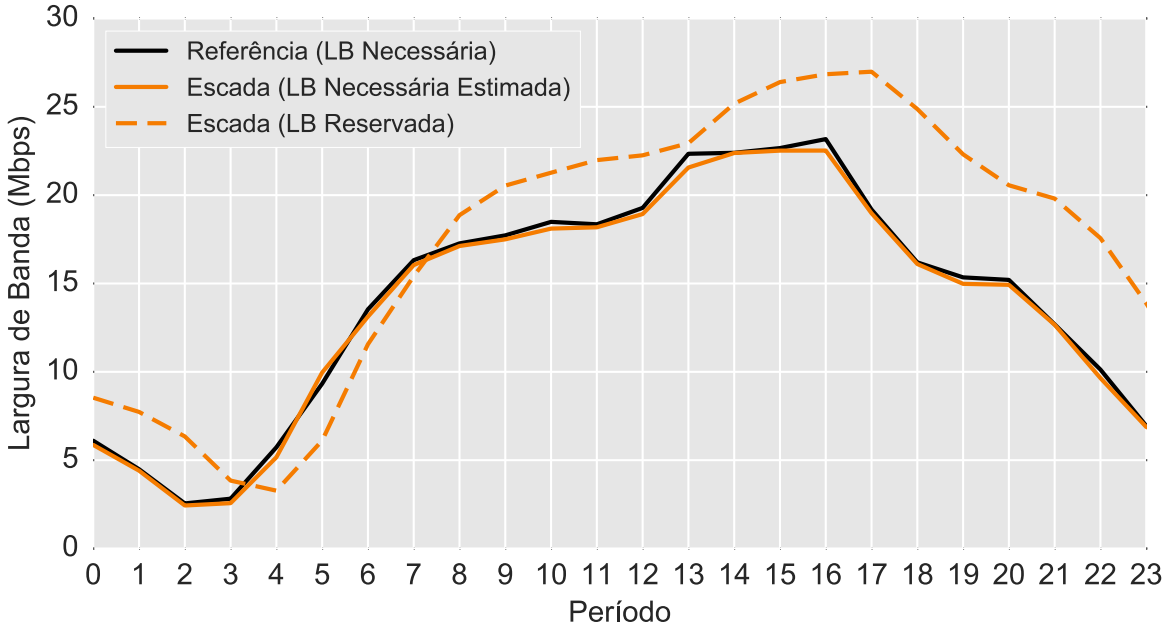
Fonte: autoria própria

4.4 CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo foram descritos experimentos realizados para desenvolver e avaliar a proposta de um modelo para provisionamento de rede de máquinas virtuais em nuvens IaaS. O primeiro experimento foi elaborado com o intuito de verificar a viabilidade do uso da *cluster analysis* para o propósito de agrupamento de fluxos e rede e também identificar o algoritmo de *clustering* mais adequado, segundo critérios qualitativos e quantitativos. Concluiu-se que Mini-Batch K-Means é a melhor opção de algoritmo para o propósito do trabalho, pois ele necessita de apenas um parâmetro que é altamente objetivo, possui complexidade de tempo adequada, e apresentou a melhor média de |MPE| durante o comparativo quantitativo. Como todos os algoritmos foram capazes de gerar agrupamentos bastante representativos em todos os cenários de teste, considerou-se viável o uso do processo de *cluster analysis* para o propósito do trabalho.

O segundo experimento foi realizado com o objetivo de avaliar a qualidade das estimativas dadas pelo Escada quando o provisionamento de rede no período de monitoração varia entre níveis de subprovisionamento, superprovisionamento e provisionamento adequado. Observou-se que mesmo no cenário mais desfavorável testado, um (sub)provisionamento de LB com apenas 40% da necessidade real, Escada consegue produzir estimativas úteis para levar ao provisionamento adequado da rede. O modelo proposto é capaz de fornecer boas estimativas da

Figura 18 – Estimativas de LB necessária e LBs reservadas por Escada nos primeiros 24 períodos de execução do experimento.



Fonte: autoria própria

LB necessária independentemente do nível de provisionamento da rede durante o processo de estimação.

O último experimento foi conduzido com a finalidade de avaliar a qualidade das previsões de LB necessária fornecidas por Escada e compará-las às do Cicada (LACURTS et al., 2014). Segundo os resultados obtidos, Escada precisa de aproximadamente 79% do tempo necessário ao Cicada para alcançar um estado estável de previsões acuradas. Para iniciar a convergência para o estado estável, Escada demanda cerca de 2/3 do tempo necessário ao Cicada. Durante os períodos de instabilidade e ajuste do algoritmo de *expert-tracking*, Escada tende a subprovisionar com menor frequência que Cicada (4 períodos por ciclo para Escada contra 10 períodos para Cicada). As estimativas de LB necessária fornecidas pelo modelo de estimação do Escada permitem que o algoritmo de previsão de *expert-tracking* convirja com a mesma velocidade que no caso de uso ideal do algoritmo. Reiterando os resultados do experimento anterior, observou-se que os erros de previsão (e provisionamento) têm menor impacto sobre a estimação do Escada do que sobre Cicada. Escada apresentou maior robustez na estimação de LB necessária, estimando-a acuradamente em todos os períodos de execução do experimento.

5 CONCLUSÃO

5.1 CONSIDERAÇÕES FINAIS

O modelo de serviço *Infrastructure as a Service* (IaaS) da computação em nuvem oferece a seus clientes o acesso a recursos computacionais configuráveis (capacidade de processamento, armazenamento e comunicação), normalmente instanciados na forma de máquinas virtuais (MVs). Através do conceito de elasticidade, clientes podem requisitar o ajuste das capacidades dos recursos das MVs ao longo do tempo para atender variações de demanda de suas aplicações, e pagam conforme a capacidade reservada dos recursos, independentemente da efetiva utilização desses. Nesse contexto, clientes de nuvens buscam reservar a capacidade mínima de recursos sem impactar negativamente o desempenho das aplicações que os utilizam, evitando gastos supérfluos e a degradação da experiência de uso das aplicações.

No presente trabalho, buscou-se auxiliar os clientes de nuvem na tarefa de provisionar adequadamente a capacidade da rede disponível a suas aplicações. O objetivo deste trabalho foi propor um modelo para fornecer previsões da largura de banda de rede necessária a máquinas virtuais hospedadas em nuvens IaaS, de forma que clientes possam provisionar adequadamente suas máquinas virtuais e as aplicações nelas implantadas. O modelo proposto, denominado Escada, monitora a ocorrência de fluxos de rede nas interfaces de uma máquina virtual para identificar o volume de dados das transações típicas de todas as aplicações em execução, sem a necessidade de conhecê-las. Um algoritmo de *clustering* é usado para identificar grupos de fluxos com demandas de rede semelhantes, e o tráfego de rede é classificado segundo esses grupos. Um novo algoritmo de classificação foi proposto para permitir identificar demandas de rede não atendidas e estimar a largura de banda necessária real para a máquina virtual. Essas estimativas de LB necessária são usadas, por fim, por um algoritmo de previsão para auxiliar no provisionamento futuro da rede.

Resultados experimentais mostram que algoritmos de *clustering* são adequados para o agrupamento de fluxos de rede. Nos cenários investigados, todos os algoritmos estudados foram capazes de gerar agrupamentos com baixas médias de erros percentuais. Isso significa que, os grupos formados pelos algoritmos representam bem a totalidade dos fluxos de rede observados.

A avaliação do modelo de estimação do Escada mostrou que este é capaz de gerar estimativas de largura de banda de rede aproximadas das reais e úteis para o provisionamento adequado da rede. No pior caso observado, o modelo superestimou em apenas 25% a demanda de rede quando esta estava provisionada em somente 40% da capacidade necessária. Uma melhora em relação ao que seria obtido pela medição da taxa de transmissão da máquina virtual, que subestimaria a demanda em 60%.

Da avaliação do provisionamento de rede realizado por Escada e da sua comparação com o Cicada (LACURTS et al., 2014) conclui-se que o modelo aqui proposto demanda menos tempo para ajustar o algoritmo de previsão de LB necessária de forma que esse preveja acuradamente durante um ciclo completo. Escada também é menos suscetível a que erros de previsão passados impactem nas previsões futuras direta ou indiretamente. Uma outra conclusão importante é que Escada é capaz de prever as demandas de rede futuras tão bem quanto o algoritmo de *expert-tracking* em seu caso de uso ideal, quando o histórico de estimativas contém as LBs necessárias reais de uma máquina virtual e sua aplicação.

Diante dos resultados acima descritos, observa-se que o principal objetivo deste trabalho – fornecer previsões suficientemente acuradas da LB necessária a máquinas virtuais em nuvens IaaS para provisionar adequadamente as aplicações nessas implantadas – foi alcançado. Os fluxos de rede foram identificados como uma abstração para caracterizar o tráfego de rede que possui atributos que indicam ou sugerem a real demanda de capacidade de rede. Foi proposto um modelo que estima suficientemente bem a LB necessária em quaisquer níveis de provisionamento de rede (Seção 3.2). Adaptou-se o algoritmo de *expert-tracking* (LACURTS et al., 2014) para prever as LBs necessárias futuras (Seção 3.3). Analisou-se o impacto do nível de provisionamento de rede sobre a estimação e a previsão do Escada (Seções 4.2 e 4.3). E, finalmente, avaliou-se experimentalmente a qualidade das previsões do modelo proposto Escada (Seção 4.3). Portanto, todos os objetivos principal e específicos traçados foram atingidos.

5.2 TRABALHOS FUTUROS

Alguns trabalhos futuros podem ser elencados para continuidade do presente trabalho. Uma proposta seria a modificação do algoritmo de *expert-tracking* (LACURTS et al., 2014) para atribuir peso à matriz de tráfego mais antiga segundo alguma regra que considera a saúde das previsões realizadas até então. Em sua proposta original, o novo peso atribuído à matriz mais antiga é igual à metade do menor peso existente. Esse método é justificado pelo fato de que não havia matriz referente ao novo período mais antigo na previsão mais recente, e portanto sua "opinião de especialista" ainda não foi testada. Para acelerar o ajuste do algoritmo de *expert-tracking*, o novo peso poderia considerar o estado atual de ajuste, caso o algoritmo esteja bem ajustado os novos pesos podem ser pequenos, no caso, entretanto, de o algoritmo ainda estar se ajustando e apresentar erros significativos de previsão os novos pesos podem ser maiores. A ideia é que, apesar de a "opinião de especialista" da matriz não ter sido testada, as opiniões das outras consideradas até então não se mostraram totalmente confiáveis.

Uma segunda proposta seria aliar o modelo de estimação do Escada com outros métodos de previsão como, por exemplo, redes neurais e *support vector machines*. Entende-se que, assim como o algoritmo de *expert-tracking*, esses métodos poderiam se beneficiar de um histórico de estimativas com melhor qualidade para gerar previsões mais acuradas.

REFERÊNCIAS

ANASTASI, G. F. et al. QoS guarantees for network bandwidth in private clouds. **Procedia Computer Science**, v. 97, p. 4–13, 2016.

ANHALT, F.; KOSLOVSKI, G.; PRIMET, P. V.-B. Specifying and provisioning virtual infrastructures with HIPerNET. **Int. J. Network Management**, John Wiley & Sons, Inc., v. 20, n. 3, p. 129–148, 2010.

BACKSTROM, L.; KLEINBERG, J.; KUMAR, R. Optimizing web traffic via the media scheduling problem. In: **Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2009. (KDD '09), p. 89–98. ISBN 978-1-60558-495-9. Disponível em: <http://doi.acm.org/10.1145/1557019.1557036>.

BALLANI, H. et al. The price is right: Towards location-independent costs in datacenters. In: **10th ACM Workshop on Hot Topics in Networks (HotNets)**. [S.l.: s.n.], 2011.

BARHAM, P. et al. Xen and the art of virtualization. In: **Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles**. New York, NY, USA: ACM, 2003. (SOSP '03), p. 164–177. ISBN 1-58113-757-5. Disponível em: <http://doi.acm.org/10.1145/945445.945462>.

BEJTLICH, R. **The practice of network security monitoring: understanding incident detection and response**. [S.l.]: No Starch Press, 2013.

BULLARD, C. **ARGUS – Auditing Network Activity**. 2017. Disponível em <http://www.qosient.com/argus/>.

CHENG, Y. Mean shift, mode seeking, and clustering. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 17, n. 8, p. 790–799, 1995.

ENGELBRECHT, H.; GREUNEN, M. van. Forecasting methods for cloud hosted resources, a comparison. In: IEEE. **Network and Service Management (CNSM), 2015 11th International Conference on**. [S.l.], 2015. p. 29–35.

FULLMER, M. **flow-tools – Tool set for working with Netflow data**. 2017. Disponível em <https://code.google.com/archive/p/flow-tools/>.

GONG, Z.; GU, X.; WILKES, J. PRESS: Predictive elastic resource scaling for cloud systems. In: IEEE. **2010 International Conference on Network and Service Management**. [S.l.], 2010. p. 9–16.

HALL, S. **The Bro Network Security Monitor**. 2017. Disponível em <https://www.bro.org/index.html>.

HERBSTER, M.; WARMUTH, M. K. Tracking the best expert. **Machine Learning**, v. 32, n. 2, p. 151–178, 1998. ISSN 1573-0565. Disponível em: <http://dx.doi.org/10.1023/A:1007424614876>.

HUBERT, B. **tc(8) man-page**. 2017. Disponível em <http://man7.org/linux/man-pages/man8/tc.8.html>.

JAIN, R. **The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling**. New York, NY: Wiley, 1991.

KABACOFF, R. **R in action: data analysis and graphics with R**. [S.l.]: Manning Publications Co., 2015.

LACURTS, K. et al. Cicada: Introducing predictive guarantees for cloud networks. In: **6th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 14)**. [S.l.: s.n.], 2014.

LAGAR-CAVILLA, H. A. et al. Snowflock: rapid virtual machine cloning for cloud computing. In: ACM. **Proceedings of the 4th ACM European conference on Computer systems**. [S.l.], 2009. p. 1–12.

LORIDO-BOTRAN, T.; MIGUEL-ALONSO, J.; LOZANO, J. A. A review of auto-scaling techniques for elastic applications in cloud environments. **Journal of Grid Computing**, Springer, v. 12, n. 4, p. 559–592, 2014.

LUCAS, M. **Network flow analysis**. [S.l.]: No Starch Press, 2010.

MARCON, D. S. et al. PredCloud: Providing predictable network performance in large-scale OpenFlow-enabled cloud platforms through trust-based allocation of resources. **Computer Communications**, Elsevier, v. 91, p. 44–61, 2016.

MELL, P. M.; GRANCE, T. **The NIST Definition of Cloud Computing**. Gaithersburg, MD, United States, 2011.

MILLER, D. **softflowd – fast software NetFlow probe**. 2017. Disponível em <http://www.mindrot.org/projects/softflowd/>.

MOGUL, J. C.; POPA, L. What we talk about when we talk about cloud network performance. **ACM Computer Communication Review**, v. 42, n. 5, p. 44–48, out. 2012.

MOREIRA JR, D. A.; OBELHEIRO, R. R. Previsão de demanda de recursos em nuvens IaaS usando séries temporais. In: **VI Simpósio Brasileiro de Engenharia de Sistemas Computacionais (SBESC)**. João Pessoa, PB: [s.n.], 2016.

MOSBERGER, D.; JIN, T. httperf—a tool for measuring web server performance. **ACM SIGMETRICS Performance Evaluation Review**, ACM, v. 26, n. 3, p. 31–37, 1998.

NGUYEN, H. et al. Agile: Elastic distributed resource scaling for infrastructure-as-a-service. In: **Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13)**. [S.l.: s.n.], 2013. p. 69–82.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PFITSCHER, R. J.; PILLON, M. A.; OBELHEIRO, R. R. Diagnóstico do provisionamento de recursos para máquinas virtuais em nuvens IaaS. **31o. Simpósio Brasileiro de Redes de Computadores (SBRC)**, p. 599–612, 2013.

POPA, L. et al. FairCloud: Sharing the network in cloud computing. In: **Proc. ACM SIGCOMM**. [S.l.: s.n.], 2012. p. 187–198.

POPA, L. et al. Faircloud: sharing the network in cloud computing. In: ACM. **Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication**. [S.l.], 2012. p. 187–198.

PRIMET, P. V.-B.; ANHALT, F.; KOSLOVSKI, G. Exploring the virtual infrastructure service concept in grid'5000. In: **20th ITC Specialist Seminar on Network Virtualization, Hoi An, Vietnam**. [S.l.: s.n.], 2009.

SULEIMAN, B. et al. On understanding the economics and elasticity challenges of deploying business applications on public cloud infrastructure. **Journal of Internet Services and Applications**, v. 3, p. 173–193, 2012.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. 1st. ed. [S.l.]: Pearson, 2005.

VAN, H. N.; TRAN, F. D.; MENAUD, J.-M. Sla-aware virtual resource management for cloud infrastructures. In: IEEE. **Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference on**. [S.l.], 2009. v. 1, p. 357–362.

XIE, D. et al. The only constant is change: Incorporating time-varying network reservations in data centers. **SIGCOMM Comput. Commun. Rev.**, ACM, New York, NY, USA, v. 42, n. 4, p. 199–210, ago. 2012. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/2377677.2377718>.

XU, R.; WUNSCH, D. et al. Survey of clustering algorithms. **Neural Networks, IEEE Transactions on**, Ieee, v. 16, n. 3, p. 645–678, 2005.

ZHANG, Q.; CHENG, L.; BOUTABA, R. Cloud computing: state-of-the-art and research challenges. **Journal of Internet Services and Applications**, Springer-Verlag, v. 1, n. 1, p. 7–18, 2010. ISSN 1867-4828. Disponível em: <http://dx.doi.org/10.1007/s13174-010-0007-6>.

ZHANG, T.; RAMAKRISHNAN, R.; LIVNY, M. Birch: an efficient data clustering method for very large databases. In: ACM. **ACM Sigmod Record**. [S.l.], 1996. v. 25, n. 2, p. 103–114.