

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

**DANIEL SCHEIDEMANTEL CAMARGO**

**CHAVE: CONSOLIDATION WITH HIGH-AVAILABILITY ON  
VIRTUALIZED ENVIRONMENTS**

**JOINVILLE**

**2018**

**DANIEL SCHEIDEMANTEL CAMARGO**

**CHAVE: CONSOLIDATION WITH HIGH-AVAILABILITY ON  
VIRTUALIZED ENVIRONMENTS**

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. Maurício Aronne Pillon

Coorientador: Dr. Charles Christian Miers

**JOINVILLE**

**2018**

Camargo, Daniel Scheidemantel

CHAVE: Consolidation with High-Availability on Virtualized Environments/ DANIEL SCHEIDEMANTEL CAMARGO. – Joinville, 2018-

89 p. : il. (algumas color.) ; 30 cm.

Orientador Dr. Maurício Aronne Pillon

Coorientador Dr. Charles Christian Miers

Dissertação (mestrado) – Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Computação Aplicada, Joinville, 2018.

1. Computação em nuvem. 2. alta disponibilidade. 3. eficiência energética. I. Pillon, Maurício Aronne . II. Christian Miers, Charles. III. Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Computação Aplicada. IV. Título.

CDU 02:121:005.7

**CHAVE: CONSOLIDATION WITH HIGH-AVAILABILITY ON VIRTUALIZED  
ENVIRONMENTS**

por

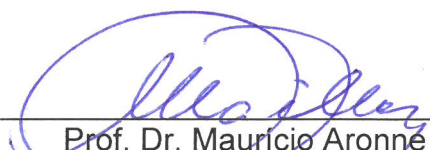
**Daniel Scheidemantel Camargo**

Esta dissertação foi julgada adequada para obtenção do título de

**Mestre em Computação Aplicada**

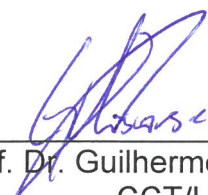
Área de concentração em “Ciência da Computação”,  
e aprovada em sua forma final pelo

**CURSO DE MESTRADO ACADÊMICO EM COMPUTAÇÃO APLICADA  
DO CENTRO DE CIÊNCIAS TECNOLÓGICAS DA  
UNIVERSIDADE DO ESTADO DE SANTA CATARINA.**



---

Prof. Dr. Mauricio Aronne Pillon  
CCT/UDESC (Orientador/Presidente)  
Membro da Banca Examinadora



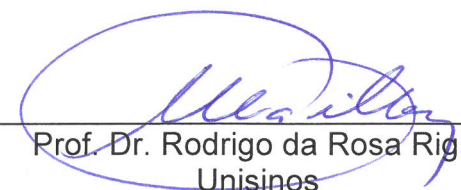
---

Prof. Dr. Guilherme Piêgas Koslovski  
CCT/UDESC  
Membro da Banca Examinadora



---

Prof. Dr. Charles Christian Miers  
CCT/UDESC (Coorientador)



---

Prof. Dr. Rodrigo da Rosa Righi  
Unisinos  
Membro da Banca Examinadora

**Joinville, SC, 31 de outubro de 2018.**

Dedico este trabalho primeiramente a Deus. Aos meus amores Mariela e Helena, por me suportarem durante essa jornada, pois sem vocês nada disso faria sentido. Aos meus pais, Sônia e Arnaldo, por todo amor e auxílio, sem vocês nada disso seria possível. Aos meus orientadores, Maurício, Charles e amigos da banca pela perseverança nesse trabalho e por continuarem apoiando e acreditando, mesmo quando tudo parecia impossível de acontecer. Por fim, aos amigos, colegas e professores que me acompanharam e deram forças nessa magnífica trajetória. A palavra que me define hoje é *#Gratidão*.

## **AGRADECIMENTOS**

Gostaria de agradecer à todos os colegas e amigos do Laboratório de Processamento Paralelo e Distribuído (LabP2D) pela contribuição ao referido trabalho. Agradeço aos meus orientadores, que se empenharam não apenas no desenvolvimento científico, pessoal e profissional, não medindo esforços sempre que os solicitei. Agradeço ao professor Guilherme Piêgas Koslovski e seu orientando Anderson Raugust pela contribuição nas nossas proveitosas reuniões auxiliando em termos teóricos. Em termos práticos, agradeço ao Allan Krueger e Glauber Cassiano Batista pelos estudos sobre o OpenStack e suas tecnologias. Ao amigo, co-autor de artigo e membro da banca Rodrigo da Rosa Righi por suas valiosas observações e contribuições. Aos demais amigos e colegas por suas contribuições, a soma de todo esse auxílio resulta no presente trabalho, que possui um pouco de todos vocês.

“Educação não transforma o mundo.  
Educação muda pessoas.  
Pessoas transformam o mundo.”

Paulo Freire

## RESUMO

As organizações que dependem da continuidade dos negócios baseadas em serviços críticos de Tecnologia da Informação (TI) demandam cada vez mais por alta disponibilidade (HA) para mitigar os prejuízos causados devido a interrupção dos serviços em casos de desastres ou falhas. O paradigma de computação em nuvem fornece meios para que sejam implementados serviços em HA baseado em replicação de serviços críticos através de arquitetura distribuída em múltiplas zonas de disponibilidade (AZs). Todavia, HA possui um inerente custo de execução, relacionado a necessidade de mais equipamentos computacionais em redundância o quê implica no aumento do consumo de energia elétrica. Para mitigar esse impacto financeiro, a literatura especializada aplica a consolidação de máquinas virtuais (MVs) como estratégia para obter eficiência energética, mas que pode causar violações de acordo de nível de serviço (SLA), tais como as políticas de afinidade e anti-afinidade. Assim, CHAVE atua na integração entre a replicação e a consolidação de MVs, executando-as em níveis diferentes. Enquanto a consolidação é aplicada internamente à cada AZ, respeitando a afinidade MVs-AZ, a replicação é executada apenas entre AZs distintas, respeitando a anti-afinidade MV-MVs (crítica-réplica). Nesse sentido, CHAVE especifica a quantidade ideal de réplicas para manter a taxa desejada de HA especificada pelo *tenant*, concebendo a ideia de HA sob demanda. A estratégia de consolidação de MVs baseia-se na heurística *First Fit Decreasing* (FFD), e tem duas variações para o presente trabalho: os algoritmos MAX e *Affinity-Aware* (AA). Testes são aplicados em simulação numérica, e são analisados sob as perspectivas local e global, permitindo a avaliação e correlação dos algoritmos avaliados sob métricas inerentes a cada perspectiva. A aplicação do CHAVE pode permitir a redução de custos com energia e com aquisição de novos servidores, pois a consolidação permite utilizar melhor os recursos disponíveis. Quando o objetivo é fornecer HA, CHAVE mostra-se satisfatório por manter um consumo de energia similar a um cenário base (sem consolidação ou HA). Assim, CHAVE pode ser associado às plataformas de computação em nuvem reais, sejam as públicas, como *Google Cloud Platform* (GCP), *Amazon Web Services* (AWS) ou Azure, ou privadas como Eucalyptus, OpenStack e CloudStack, que necessitem fornecer um mecanismo de HA.

**Palavras-chave:** Computação em nuvem, alta disponibilidade, eficiência energética.



## ABSTRACT

Organizations depending on business continuity based on critical services in Information Technology (IT) increasingly demand high availability (HA) to mitigate the impact caused by disruption of services in the event of disasters. The cloud computing paradigm provides a means for deploying HA services based on replication of critical services through multi-availability zones (AZ) architectures. However, has an inherent execution cost, related to the need for more computational equipment in redundancy, reflecting an increase in the consumption of electric energy. In order to mitigate this financial impact, energy efficiency strategies apply the virtual machine (VM) consolidation cause service level agreements (SLA) violations, infringing mainly the affinity and anti-affinity policies. Thus, CHAVE acts in the integration between the replication and the VM consolidation, executing at different levels. While the consolidation is applied internally to each AZ, respecting the VM-AZ affinity, replication is performed only between distinct AZ, respecting VM-VM (replica-critical) anti-affinity. To measure the HA rate, the present work is based on the probability of independent events, together with the reliability block diagram (RBD) analysis. In this way, CHAVE specifies the ideal amount of replicas to maintain the desired HA rate specified by the tenant, designing the idea of HA on demand. The VM consolidation strategy proposed is based on the First Fit Decreasing (FFD) heuristic, and has two variations for the present work: the MAX and Affinity-Aware (AA) algorithm. Tests are applied in numerical simulation, and are analyzed under local and global perspectives, allowing the evaluation and correlation of the algorithms evaluated under metrics inherent to each perspective. The application of the CHAVE can allow to reduce energy costs and the acquisition of new servers, since the consolidation allows to make better use of available resources. When the objective is to provide HA, CHAVE is shown to be satisfactory by maintaining energy consumption similar to a base scenario (without consolidation or HA). Thus, CHAVE can be associated with real cloud computing platforms, whether public, such as Google Cloud Platform (GCP), Amazon Web Services (AWS) or Azure, or private ones such as Eucalyptus, OpenStack and CloudStack, that need to provide a mechanism of HA.

**Key-words:** Cloud Computing, High-Availability, VM Consolidation

## LISTA DE ILUSTRAÇÕES

Figura 1 – Componentes em série e paralelo. . . . .	29
Figura 2 – Conflito entre consolidação e replicação para eficiência energética e HA. . . . .	35
Figura 3 – Representação da demanda para o Provedor de Serviços em Nuvem (CSP) e CHAVE. . . . .	43
Figura 4 – Arquitetura CHAVE: consolidação e replicação de MVs integrado ao CSP. . . . .	43
Figura 5 – Fluxograma para uma requisição do tipo regular, com fluxo apenas na camada de consolidação de MVs. . . . .	45
Figura 6 – Fluxograma para uma requisição do tipo crítica com consolidação e replicação de MVs. . . . .	46
Figura 7 – Visão geral da replicação em arquiteturas Multi-AZ: <i>tenant</i> requisita HA = 99,999% e CHAVE replica o serviço para mais duas réplicas. .	50
Figura 8 – Árvore de testes . . . . .	57
Figura 9 – Distribuição de carga para AZ1 . . . . .	59
Figura 10 – Distribuição de carga para AZ2 . . . . .	59
Figura 11 – Distribuição de carga para AZ3 . . . . .	60
Figura 12 – Distribuição de carga para AZ4 . . . . .	60
Figura 13 – Distribuição de carga para AZ5 . . . . .	61
Figura 14 – Distribuição de carga para AZ6 . . . . .	61
Figura 15 – Testes em perspectiva local. (a) AZ1 melhores testes: CHA_AA0 e C_MAX. (b) AZ2 melhores testes: CHA_AA0 e C_AA20. . . . .	65
Figura 16 – Testes em perspectiva local. (a) AZ3 melhores testes: CHA_AA0 e empate entre C_AA0 e C_AA20. (b) AZ4 melhores testes: CHA_MAX e empate entre C_AA0, C_AA20 e C_MAX. . . . .	66
Figura 17 – Testes em perspectiva local. (a) AZ5 melhores testes: CHA_MAX e C_AA20. (b) AZ6 melhores testes: CHA_AA0 e C_MAX. . . . .	67
Figura 18 – Testes em perspectiva global. (a) AZ1 melhores testes: CHA_AA0 e C_AA0. (b) AZ2 melhores testes CHA_AA0 e C_AA0. . . . .	68
Figura 19 – Testes em perspectiva global. (a) AZ3 melhores resultados: C_AA0 e grande similaridade entre os três testes CHA. (b) AZ4 melhores resultados . . . . .	69
Figura 20 – Testes em perspectiva global. (a) AZ5 melhores resultados: C_AA0 e CHA_AA0. (b) AZ6 melhores resultados: C_MAX e CHA_AA0. . .	70

## LISTA DE TABELAS

Tabela 1 – Classes de disponibilidade por número de ‘noves’ na taxa. . . . .	22
Tabela 2 – Questões de pesquisa e motivações. . . . .	37
Tabela 3 – Critérios de inclusão e exclusão dos artigos selecionados. . . . .	38
Tabela 4 – Relação entre os trabalhos correlatos e principais critérios. . . . .	39
Tabela 5 – Tabela de notações, definições formais e equações utilizadas. . . .	49
Tabela 6 – Requisitos funcionais . . . . .	55
Tabela 7 – Requisitos Não-funcionais . . . . .	56
Tabela 8 – Informações e configuração dos <i>traces</i> por AZ. . . . .	56
Tabela 9 – Tabela de métricas utilizadas para análise e discussão dos resultados.	64
Tabela 10 – Tabela de métricas utilizadas para análise e discussão dos resultados.	68
Tabela 11 – Relação da execução de base (EUCA) com melhores resultados da execução geral na métrica $ET(W)$ . . . . .	71

## LISTA DE SIGLAS E ABREVIATURAS

**AA** *Affinity-Aware*

**ACPI** *Advanced Configuration and Power Interface*

**API** *Application Programming Interface*

**AZ** zona de disponibilidade

**AWS** *Amazon Web Services*

**COBIT** *Control Objectives for Information and related Technology*

**CSP** Provedor de Serviços em Nuvem

**DC** *data center*

**DVFS** *Dynamic Voltage and Frequency Scaling*

**E/S** entrada e saída

**FFD** *First Fit Decreasing*

**GCP** *Google Cloud Platform*

**GVT** *Global Virtual Time*

**HA** alta disponibilidade

**IaaS** Infraestrutura como Serviço

**ITIL** *Information Technology Infrastructure Library*

**IWGCR** *International Working Group on Cloud Computing Resiliency*

**IV** Infraestrutura Virtual

**LabP2D** Laboratório de Processamento Paralelo e Distribuído

**MV** máquina virtual

**NIST** *National Institute of Standards and Technology*

**PaaS** Plataforma como Serviço

**PBP** Problema do Bin Packing

**PBS** pesquisa bibliográfica sistemática

**PDU** unidade de distribuição de energia

**PUE** *Power Usage Effectiveness*

**QoS** *Quality of Service*

**RBD** diagrama de bloco de confiabilidade

**SaaS** Software como Serviço

**SLA** acordo de nível de serviço

**SO** sistema operacional

**SPoF** ponto único de falhas

**SPI** *Software, Plataforma e Infraestrutura*

**TCO** Custo Total de Propriedade

**TI** Tecnologia da Informação

**UPS** fonte de energia ininterrupta

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>15</b>
1.1	OBJETIVO	16
1.2	CARACTERIZAÇÃO METODOLÓGICA DA PESQUISA	18
1.3	ESCOPO	19
1.4	ESTRUTURA E ORGANIZAÇÃO DO TRABALHO	19
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>20</b>
2.1	COMPUTAÇÃO EM NUVEM	20
2.2	ALTA DISPONIBILIDADE E EFICIÊNCIA ENERGÉTICA EM DATA CENTERS	21
<b>2.2.1</b>	<b>Alta disponibilidade</b>	<b>23</b>
<b>2.2.2</b>	<b>Estratégias de eficiência energética</b>	<b>25</b>
<b>2.2.3</b>	<b>Considerações parciais sobre HA e eficiência energética</b>	<b>27</b>
2.3	REPLICAÇÃO DE MVs	27
2.4	CONSOLIDAÇÃO DE MVs	30
2.5	CONFLITO ENTRE A CONSOLIDAÇÃO E REPLICAÇÃO DE MVs	34
2.6	DEFINIÇÃO DO PROBLEMA	35
2.7	TRABALHOS CORRELATOS	36
<b>2.7.1</b>	<b>Questões de pesquisa</b>	<b>37</b>
<b>2.7.2</b>	<b>Procedimento de seleção dos trabalhos</b>	<b>38</b>
<b>2.7.3</b>	<b>Análise dos trabalhos levantados</b>	<b>38</b>
2.8	CONSIDERAÇÕES PARCIAIS	40
<b>3</b>	<b>CHAVE</b>	<b>42</b>
3.1	ARQUITETURA	43
<b>3.1.1</b>	<b>Casos de uso</b>	<b>44</b>
<b>3.1.2</b>	<b>Requisitos do sistema</b>	<b>46</b>
3.2	FORMALISMO MATEMÁTICO	48
<b>3.2.1</b>	<b>Alta disponibilidade</b>	<b>49</b>
<b>3.2.2</b>	<b>Consolidação de MVs</b>	<b>51</b>
3.3	PROTÓTIPO E AMBIENTE DE TESTES	53
<b>3.3.1</b>	<b>Requisitos vs Implementação</b>	<b>54</b>
<b>3.3.2</b>	<b>Cenário</b>	<b>54</b>
<b>3.3.3</b>	<b>Plano de testes</b>	<b>57</b>
3.4	CONSIDERAÇÕES PARCIAIS	61

<b>4</b>	<b>EXPERIMENTOS E ANÁLISE DOS RESULTADOS . . . . .</b>	<b>63</b>
4.1	PERSPECTIVA LOCAL: <i>SNAPSHOTS</i> DA CONSOLIDAÇÃO . . . .	64
4.2	PERSPECTIVA GLOBAL: SIMULAÇÃO TOTALIZADA . . . . .	67
4.3	COMPARAÇÃO COM O CENÁRIO DE BASE DO EUCALYPTUS . .	71
4.4	CONSIDERAÇÕES PARCIAIS . . . . .	73
<b>5</b>	<b>CONSIDERAÇÕES &amp; TRABALHOS FUTUROS . . . . .</b>	<b>74</b>
5.1	CONTRIBUIÇÕES DO TRABALHO . . . . .	76
5.2	TRABALHOS FUTUROS . . . . .	76
5.3	PUBLICAÇÕES . . . . .	77
	<b>REFERÊNCIAS . . . . .</b>	<b>78</b>

## 1 INTRODUÇÃO

O paradigma de computação em nuvem tem se tornado fundamental para organizações que fornecem e baseiam-se em serviços críticos, uma vez que a interrupção destes serviços pode causar prejuízos financeiros e de reputação (MACIEL, 2016; ALKAWSI; et al, 2015). Nuvens computacionais fornecem recursos computacionais sob demanda, normalmente sob o modelo *pay-as-you-go*, com garantias de disponibilidade (MELL; GRANCE, 2011) estabelecidas em acordo de nível de serviço (SLA) entre o Provedor de Serviços em Nuvem (CSP) e o *tenant* (entidade responsável pela implementação dos serviços). Entretanto, é comum a ocorrência de interrupções causadas por diversos tipos de falhas, que vem ganhando notoriedade entre pesquisadores (ISLAM; MANIVANNAN, 2017; SANTOS; et. al, 2017), transformando-se em um critério de seleção para os *tenants* (MARSTON et al., 2011; AMOON, 2016). Estima-se um impacto financeiro de centenas de milhares de dólares por cada hora de interrupção (ENDO; et. al, 2016), prejudicando não apenas os CSPs, mas também causando má reputação aos *tenants*. Um relatório emitido pelo *International Working Group on Cloud Computing Resiliency* (IWGCR) em 2014 (CÉRIN et al., 2014), apresenta um resumo sobre a disponibilidade dos principais CSP do mercado. São avaliados um total de 38 CSPs e estima-se que o custo por hora varie de US\$ 100 mil a US\$336 mil. No período de sete anos avaliado, o tempo de inatividade causou prejuízos no total de US\$670 milhões. Desse modo, CSPs têm dado ênfase em meios de aprimorar suas infraestruturas e o gerenciamento de estratégias de alta disponibilidade (HA) para prover serviços que suportem os objetivos do SLA.

Organizações que possuem um plano de continuidade de negócios para sua infraestrutura de Tecnologia da Informação (TI) tipicamente gerenciam a recuperação de desastres (DR) (ALSHAMMARI; ALWAN; ALSHAIKHLI, 2016), com o objetivo de minimizar o período de tempo entre o desastre e a sua efetiva recuperação. Estes desastres são muitas vezes imprevisíveis, cujo âmbito de falhas inclui desde as infraestruturas física e virtual, até questões de segurança, como a intrusão de sistemas e infecções por *malwares* (KUMAR; RAJ; JELCIANA, 2018). Com o paradigma de computação em nuvem, diversas organizações migraram sua infraestrutura de servidores privados (*on-premise*) para o modelo de implantação em nuvem híbrida ou pública (IORGA; KARMEEL, 2016). Assim, mesmo que o plano de continuidade de negócios objetive evitar ou prevenir que falhas ocorram, muitas vezes o controle está fora da própria organização, passando a ser de responsabilidade do CSP (ENDO; et. al, 2016). Por outro lado, a interrupção dos serviços pode ser causada internamente à aplicação, por falhas de responsabilidade do *tenant* (e.g., incorreta implementação



ou configuração dos serviços) (BEYER et al., 2016). Para reduzir ou mascarar as interrupções de responsabilidade do CSP, mecanismos de HA baseiam-se em métodos de redundância (MATOS; et. al, 2017), replicação (PETROVIC; SCHIPER, 2012) e tolerância à falhas (KANSO; et. al, 2017), que podem ser implementados a nível de software ou de hardware (PRATHIBA; SOWVARNICA, 2017). Todavia, todos estes mecanismos implicam em uma sobrecarga na infraestrutura física, pois demandam por mais equipamentos e elevam o Custo Total de Propriedade (TCO), implicando em um maior consumo de energia elétrica. Em *data centers* (DCs), a energia é um recurso crítico que corresponde a até 50% dos custos operacionais em um DC (COMERFORD, 2015).

## 1.1 OBJETIVO

O presente trabalho, denominado CHAVE (*Consolidation with High Availability on Virtualized Environments*), tem o objetivo de mascarar as falhas na perspectiva do CSP aplicando a replicação de máquinas virtuais (MVs), ao mesmo tempo em que realiza a consolidação de MVs que considera as restrições da HA. Sua atuação ocorre especificamente na camada de processamento da nuvem (do servidor físico ao hipervisor). A restrição de afinidade de um *pool* de replicação de MVs é definida como “*MVs críticas e suas réplicas não devem ser instanciadas no mesmo ponto único de falhas (SPoF)*” (MACHIDA; KAWATO; MAENO, 2010; MONDAL; MUPPALA; MACHIDA, 2016). Adicionalmente, o presente trabalho baseia-se na concepção de HA sob demanda, em que o *tenant* deve apenas especificar a taxa de HA desejada para cada uma das suas MVs críticas. Assim, CHAVE obtém a quantidade mínima de réplicas suficiente para fornecer a taxa de HA solicitada, reduzindo a probabilidade de falhas com um mecanismo de HA baseado em replicação por *checkpoint* no modo ativo-passivo. A atuação do mecanismo de HA na camada de processamento de uma nuvem baseada em Infraestrutura como Serviço (IaaS), permite uma abordagem agnóstica à aplicação, *i.e.*, é independente dos serviços executados internamente à MV, abstraindo-a como uma caixa-preta.

Para reduzir o impacto do consumo de energia em infraestruturas de computação em nuvem, uma das principais estratégias encontradas na literatura é a consolidação de MVs (BELOGLAZOV; BUYYA, 2012; AHMAD; et. al, 2015; HAMEED; et. al, 2016). Esta estratégia visa alocar todas as MVs no menor número de servidores físicos, possibilitando que equipamentos ociosos ou subutilizados sejam desativados/hibernados, além de permitir uma melhor alocação dos recursos virtuais no DC (MUCHALSKI; MAZIERO, 2014). A consolidação de MVs é considerada uma aplicação prática do Problema do Bin Packing (PBP), de complexidade NP-Difícil (MARTELLO; TOTH, 1990; BELOGLAZOV, 2013). Entretanto, heurísticas permitem aplicar

uma abordagem de consolidação mais rígida, *i.e.*, o mais próximo da configuração ótima, que colateralmente podem implicar em uma série de violações de SLA por infringir requisitos de disponibilidade (NANDURI; KAKADIA; VARMA, 2014; HAMEED; et. al, 2016). Assim, um mecanismo de consolidação deve ser desenvolvido consciente das restrições e propósitos do sistema em que é aplicado. Ao aplicar a consolidação de MVs simultaneamente a um mecanismo de HA, deve-se considerar a possibilidade de alocar uma MV crítica com suas réplicas em um mesmo SPoF afetando a HA desejada. Neste sentido, o presente trabalho aborda a restrição de afinidade da HA arquiteturas multi-zona de disponibilidade (AZ) (UNUVAR; et. al, 2014; DAVID, 2014; BURNS, 2017; MORENO-VOZMEDIANO; et. al, 2017), que consiste em afirmar que duas ou mais MVs de um *pool* de replicação não devem estar posicionadas na mesma AZ. Estima-se que a alta complexidade do gerenciamento de réplicas e consolidação das MV seja um custo para aplicar o CHAVE em um ambiente real por um administrador de CSP.

A contribuição do presente trabalho é fornecer um mecanismo de HA para MVs sob demanda para o *tenant*, vinculado a uma estratégia de consolidação de MVs para reduzir o impacto no consumo de energia elétrica. Baseando-se em arquitetura multi-AZ, ambas as abordagens de consolidação e replicação são separadas em camadas distintas, mas atuam em conjunto para um mesmo objetivo. Isto é, enquanto o mecanismo de HA permite prover serviços contínuos ao *tenant*, a consolidação permite reduzir o impacto no consumo de energia inerente ao mecanismo de HA. Adicionalmente, a aplicação da análise por diagrama de bloco de confiabilidade (RBD) em conjunto com a probabilidade de eventos independentes permite calcular precisamente a taxa de HA com base na taxa de disponibilidade das AZs, resultando na concepção de HA sob demanda. Outra contribuição é a discussão sobre as políticas de afinidade e anti-afinidade, considerando-as como restrições que devem ser devidamente cumpridas conforme estabelecidas em SLA.

Estudos (STROYAN; BROWN, 2013; BLIND; PETERSEN; RIILLO, 2017) indicam que considerar normas e padrões permitem obter um alto grau na eficiência da inovação para a resolução de problemas tecnológicos. Especificamente na área da segurança da informação, as normas da família ISO/IEC 27000, são amplamente adotadas por metodologias de governança e gestão de TI, como *Control Objectives for Information and related Technology* (COBIT), *Information Technology Infrastructure Library* (ITIL) e outras (AKOWUAH et al., 2018). As normas da família ISO/IEC 27000 focam em três pontos:

- (i) integridade, que busca a certeza de que a informação é realmente verdadeira e correta;
- (ii) confidencialidade, que garante que a informação seja acessada somente por

- quem tenha autorização; e
- (iii) disponibilidade, que permite acesso à informação no tempo e no local requerido pelo *tenant*.

Os pontos (i) e (ii) são fortemente correlatos à área de segurança em TI, e portanto, não utilizados no presente trabalho. Assim, apenas o tópico (iii) é objeto de estudos do presente trabalho.

A Norma ISO/IEC 27031 trata especificamente sobre disponibilidade, e objetiva assegurar que os serviços de TI de uma organização estarão aptos a suportar um plano de continuidade de negócios. A continuidade de negócios é a capacidade de uma organização continuar entregando produtos ou serviços em níveis aceitáveis (previamente estabelecidos), após a ocorrência de um incidente disruptivo (CHOUDHARY, 2016). Assim, o plano de continuidade de negócios visa fornecer uma estrutura de métodos e processos para identificar e especificar critérios de desempenho, projeto e implementação. Seu objetivo é garantir para as organizações a alta disponibilidade (HA) através de tolerância à falhas e a recuperação de desastres. Nesse sentido, a recuperação de desastres de uma organização consiste em um conjunto de políticas e procedimentos para assegurar a continuidade e recuperação de sistemas de missão crítica no caso de um evento disruptivo (XIONG; FOWLEY; PAHL, 2016). Adicionalmente, o conceito de tolerância à falhas refere-se a arquiteturas computacionais projetadas para suportarem as funções críticas de negócios, após a ocorrência de uma falha (FAYYAZ; VLADIMIROVA, 2016). Assim, quando um componente falha (hardware ou software), um componente de backup (em redundância) assume imediatamente as operações para que não haja perda de serviço (MACIEL, 2016). Desse modo, a alta disponibilidade em sistemas e serviços computacionais possuem uma forte demanda, através dos planos de continuidade de negócios, consistentemente estabelecidos por normas e padrões.

## 1.2 CARACTERIZAÇÃO METODOLÓGICA DA PESQUISA

Inicialmente, é realizada uma pesquisa exploratória com o objetivo de estabelecer o estado-da-arte dos objetos de estudo, através de levantamento bibliográfico realizado em fontes científicas. O objeto de estudo consiste em dois elementos definidos como conflitantes entre si, que isoladamente constituem a solução do problema do outro, mas em conjunto geram conflitos que impedem que ambas as abordagens estejam simultaneamente associadas. Assim, o objetivo é associar um mecanismo de HA com uma estratégia de eficiência energética, aplicados no contexto de computação em nuvem IaaS. Através da pesquisa exploratória, são definidas as premissas que destacam a replicação de MVs como um mecanismo de HA, e a consolidação de MVs como estratégia de eficiência energética. Nesse sentido, realiza-se uma pes-

quisa bibliográfica sistemática para encontrar os trabalhos correlatos mais relevantes, permitindo definir os parâmetros ao desenvolvimento da solução proposta.

### 1.3 ESCOPO

A presente pesquisa é delimitada a replicação e consolidação de MVs no contexto de nuvens computacionais IaaS. Sob o conceito de responsabilidade compartilhada, define-se que o escopo da solução proposta parte da perspectiva do CSP, *i.e.*, a implementação do CHAVE está no CSP. Assim, o *tenant* é responsável apenas por definir sua demanda de nível de HA e conhecer características de afinidade de servidores.

A implementação da solução proposta é desenvolvida em simulador, sendo os dados coletados são provenientes desta simulação numérica. Com base no escopo inicialmente estabelecido, as entradas desta simulação consistem em traces reais obtidos de uma nuvem privada IaaS. As métricas utilizadas para avaliação dos dados são definidas conforme discussão em trabalhos correlatos e com base no conjunto de equações definidas, que compõem parte da contribuição do presente trabalho.

### 1.4 ESTRUTURA E ORGANIZAÇÃO DO TRABALHO

O presente trabalho está organizado da seguinte forma. O Capítulo 2 aborda a fundamentação teórica utilizada no decorrer do artigo, finalizando com os trabalhos correlatos. A proposta do presente trabalho é desenvolvida no Capítulo 3, que finaliza com a definição do plano de testes. A apresentação e discussão dos resultados ocorre no Capítulo 4. Por fim, as considerações finais estão descritas no Capítulo 5, juntamente com as contribuições e as expectativas para trabalhos futuros.

## 2 FUNDAMENTAÇÃO TEÓRICA

A presente fundamentação teórica tem o objetivo de relacionar os conceitos utilizados no decorrer do presente trabalho. Inicialmente é explanado conceitos de computação em nuvem, alta disponibilidade e eficiência energética, que são direcionados para replicação e consolidação de máquinas virtuais (MVs), apresentando o conflito entre ambas as abordagens. Por fim, uma pesquisa bibliográfica sistemática apresenta os trabalhos correlatos.

### 2.1 COMPUTAÇÃO EM NUVEM

O termo computação em nuvem é amplamente utilizado para descrever uma categoria de auto-serviço computacional sob demanda (BUYYA; BROBERG; GOSCINSKI, 2011). Uma das definições mais abrangentes e utilizadas no meio científico (KESHAVARZI; HAGHIGHAT; BOHLOULI, 2017) é estabelecida pelo *National Institute of Standards and Technology* (NIST) (MELL; GRANCE, 2011). São especificados conceitos de configurabilidade, acessibilidade sob demanda e provisionamento rápido de recursos, além de definir cinco características essenciais, quatro modelos de implantação e três modelos de serviços. As cinco características essenciais são auto-serviço sob demanda, acesso amplo a rede, conjunto de recursos, elasticidade e serviços mensuráveis. Os quatro modelos de implantação especificam como uma infraestrutura de nuvem é construída, gerenciada e acessada, sendo classificada em privada, pública, comunitária e híbrida. Os três modelos de serviços referem-se ao nível dos serviços computacionais que o *tenant* tem acesso, também conhecido como modelo SPI:

- Software como Serviço (SaaS), que proporciona sistemas de propósitos específicos, em que os *tenants* tem acesso apenas a camada de aplicação de um sistema;
- Plataforma como Serviço (PaaS), aplicado para o desenvolvimento de aplicações, com ferramentas de compilação/interpretação de linguagens de programação, testes integrados e banco de dados gerenciados; e
- Infraestrutura como Serviço (IaaS), oferece acesso aos recursos computacionais virtualizados de uma MV, como processamento, armazenamento e rede, em que o *tenant* é responsável pelo gerenciamento desses recursos e de segurança.

O escopo da fundamentação teórica consiste no modelo IaaS, devido a possibilidade de ser uma base para o desenvolvimento de SaaS e PaaS, além das características de isolamento e gerenciamento de recursos virtualizados. O presente trabalho

é independente dos modelos de implantação e considera as cinco características essenciais como requisitos de base para uma plataforma de computação em nuvem.

A infraestrutura física de uma plataforma de computação em nuvem é tipicamente constituída por um ou mais *data centers* (DCs) (WU, 2017; YAO; PAPAPANAGIOTOU, 2017), que são instalações especializadas em abrigar equipamentos computacionais e de suporte. Quando essa infraestrutura é globalmente distribuída, os DCs são denominadas de zonas de disponibilidade (AZs) quando há o isolamento de recursos (computacionais e suporte) e suprimentos (energia e rede) entre si (UNUVAR; et.al, 2014). Alguns casos, um mesmo DC pode possuir mais de uma AZ, desde que cumpra o requisito de isolamento. Cerca de 50% dos custos operacionais de um DC são provenientes do consumo de energia desta infraestrutura (COMERFORD, 2015). Para fornecer serviços tolerante a falhas, tipicamente baseados em redundância e replicação, há um aumento na demanda por mais equipamentos, devido a maior carga causada pela replicação. Segundo relatório anual da Uptime Institute (ASCIERTO; PEARL, 2018), no momento de alocar suas cargas de trabalho, as organizações com DCs privados encontram dois fatores conflitantes: reduzir o custo e manter a disponibilidade. Essa pesquisa indica ainda que a medida em que organizações implementam estratégias de eficiência energética, principalmente para reduzir o *Power Usage Effectiveness* (PUE), aumenta-se as interrupções de serviços. Assim, as linhas de pesquisa relacionadas a eficiência energética e mecanismos de alta disponibilidade (HA) apresentadas em um contexto amplo, e após são focadas nas abordagens adotadas para o presente trabalho.

## 2.2 ALTA DISPONIBILIDADE E EFICIÊNCIA ENERGÉTICA EM DATA CENTERS

O conceito de disponibilidade é amplamente utilizado para indicar o percentual de tempo de execução de um serviço ou sistema, durante um período de tempo determinado (WU; BUYYA, 2015). Por definição, este termo refere-se a probabilidade de um serviço ou sistema estar operando quando requisitado para uso (STAPELBERG, 2009; BAUER; ADAMS, 2012). As interrupções durante a operação de um serviço podem ser causadas por dois principais motivos: manutenção (preventiva ou preditiva) e ocorrência de falhas aleatórias (não programadas) (CRITCHLEY, 2014). A disponibilidade inerente a sistemas distribuídos (WEIBULL, 2007; STAPELBERG, 2009; BAUER; ADAMS, 2012) relaciona apenas o tempo de atividade do sistema e o tempo de manutenção corretiva. Genericamente, a disponibilidade é especificada pela razão entre o tempo de atividade ( $T_a$ ) e o tempo total observado ( $T_{to}$ ). O tempo de manutenção corretiva ( $T_{mc}$ ) refere-se ao período necessário para recuperar totalmente um serviço ou sistema desde a detecção de sua falha até a comutação para o componente substituto. De forma genérica, para obter a disponibilidade  $\mathbb{A}$  para um único componente, a Equação

ção 2.1 é aplicada em um período determinado de tempo  $t$ , sendo  $T = t_{final} - t_{inicial}$ .

$$\mathbb{A} = \frac{T_a}{T_{to}} = \frac{T_a}{T_a + \sum_{i=0}^n T_{mc_i}} \quad (2.1)$$

Na Equação 2.1, o tempo de atividade ( $T_a$ ) e o tempo total de manutenção corretiva ( $T_{to} = \sum_{i=0}^n T_{mc_i}$ ) para  $n$  falhas em um período  $t$  podem ser obtidos por análise histórica (ZHOU; SUN; LI, 2017; MORENO-VOZMEDIANO; et. al, 2017) ou por indicadores probabilísticos (KOSLOVSKI et al., 2010). Adicionalmente, observa-se na Equação 2.1 que a taxa de disponibilidade  $\mathbb{A}$  tende a ser maior quando o número de falhas ( $n$ ) e o tempo de manutenção corretiva ( $T_{mc}$ ) são minimizados. Como é inviável reduzir o número de falhas, pois elas são aleatórias e imprevisíveis, mecanismos de tolerância a falhas minimizam o tempo de manutenção corretiva do sistema.

A literatura especializada e a indústria utilizam uma classificação geral para a taxa de disponibilidade através da quantidade de dígitos ‘nove’ que compõe a taxa. A Tabela 1, organiza essas taxas de disponibilidade em sete classes, com termos representativos, e o tempo de indisponibilidade ao ano ( $\bar{\mathbb{A}}/\text{ano}$ ). Na coluna que enumera as

Tabela 1 – Classes de disponibilidade por número de ‘noves’ na taxa.

Classe	Termo	Taxa de Disp.	$\bar{\mathbb{A}}/\text{ano}$
1	Não gerenciado	90%	36,5 dias
2	Gerenciado	99%	3,7 dias
3	Bem gerenciado	99,9%	8,8 horas
3+	SLA de nuvens públicas	99,95%	4,4 horas
4	Tolerante à falhas	99,99%	52,6 minutos
5	Alta disponibilidade	99,999%	5,3 minutos
6	Muito-alta disponibilidade	99,9999%	34,7 segundos
7	Ultra-alta disponibilidade	99,99999%	3,5 segundos

Fonte: Modificado de Critchley (2014).

classes de disponibilidade na Tabela 1, a linha 3+ refere-se uma classe tipicamente adotada em acordo de nível de serviço (SLA) de nuvens públicas, como a *Amazon Web Services* (AWS), *Google Cloud Platform* (GCP) e *Microsoft Azure*, equivalente a 4,4 horas de interrupções ao ano. Em termos do presente trabalho, as taxas definidas acima da classe 5 (99,999%) já são categorizadas como HA, *i.e.*, uma taxa igual ou superior a cinco noves. Observa-se que a quantidade de noves presentes na taxa de disponibilidade implica em uma variação de dias até segundos de indisponibilidade ao ano. Entre a ‘Taxa de Disp.’ e ‘ $\bar{\mathbb{A}}/\text{ano}$ ’, é mais comum a utilização em percentual. Porém, utilizar o tempo de indisponibilidade permite estabelecer o conceito de HA sob demanda, através da questão: “*Quanto tempo o sistema pode ficar offline em um ano?*”, tendo a taxa em percentual como resposta.

### 2.2.1 Alta disponibilidade

Segundo (CRITCHLEY, 2014), define-se alta disponibilidade (HA) como a capacidade de um sistema fornecer serviços sempre que requisitado, em qualquer ponto no tempo e durante um período de tempo determinado. Outra definição utilizada para HA, é uma taxa de disponibilidade igual ou superior a cinco noventa e nove (99,999%) (ENDO; et. al, 2016; MORENO-VOZMEDIANO; et. al, 2017), como observado na Tabela 1. Para o presente trabalho, são adotados ambos os conceitos para HA, porém abstraindo-se a quantidade de noventa e nove da presentes na taxa para desenvolver a concepção de HA sob demanda. Especificamente para computação em nuvem IaaS, estratégias de HA podem ser organizadas conforme o critério de responsabilidade do *tenant* e do Provedor de Serviços em Nuvem (CSP).

Ao implementar a arquitetura do sistema e dos dados de serviços críticos, o *tenant* é responsável pelo uso de boas práticas de segurança (KUMAR; RAJ; JELCIANA, 2018). Além de prover integridade e confidencialidade, essas boas práticas são essenciais ao plano de continuidade de negócios (LI; LI, 2018). Um dos objetivos é mitigar os problemas causados pela interrupção de serviços referente à ataques externos (IORGA; KARMEL, 2016; BARONA; ANITA, 2017). A virtualização habilita o desenvolvimento de aplicações com arquiteturas de redes virtuais públicas e privadas, permitindo instanciar serviços críticos em camadas distintas (BORGOLTE et al., 2018). Considerados nativos para nuvem, os serviços *stateless* utilizam-se de estratégias estáticas ou baseadas na persistência de dados, *i.e.*, sem manter dados em memória. Arquitetura multi-AZs, permite desenvolver serviços *stateless* elásticos e escaláveis em HA, utilizando balanceadores de carga para distribuir as requisições entre instâncias efêmeras distribuídas entre diferentes AZs. Instâncias efêmeras são MVs que podem ser automaticamente substituídas por outras em caso de falhas, pois o balanceador de cargas redireciona as requisições apenas para instâncias saudáveis, enquanto são lançadas novas MVs com as imagens de base (tipicamente *snapshot* estável do sistema). Adicionalmente, soluções de monitoramento de estado de serviço auxiliam no acompanhamento do serviço em tempo de execução, se está ativo ou inativo (TCHANA; BROTO; HAGIMONT, 2012). Destaca-se o *heartbeat* por ser comumente utilizado como um sensor de estado devido seu baixo impacto na rede. Uma desvantagem de o *tenant* implementar as suas estratégia de HA é o desperdício de recursos de rede, enquanto o monitoramento das MVs (bem como outras estratégias) podem ser mais eficientes quando implementado pelo CSP.

Plataformas de computação em nuvem tipicamente oferecem funcionalidades de gerenciamento dos quesitos de segurança (WOLSKI; BREVIK, 2014). Tecnicamente, MVs são um conjunto de processos gerenciados pela camada de virtualização (SMITH; NAIR, 2005), portando para o CSP gerenciar MVs, ele estará gerenci-



ando processos e seus recursos associados. Da perspectiva do CSP, estratégias de HA podem ser melhor implementadas, devido ao acesso direto às MVs por meio dos hipervisores (TCHANA; BROTO; HAGIMONT, 2012). Tipicamente, o CSP coleta informações sobre o estado das MVs e do estado dos servidores, permitindo implementar soluções mais precisas conforme a sua degradação (TCHANA; BROTO; HAGIMONT, 2012). Adicionalmente, o CSP pode identificar a origem das falhas com maior precisão, para a tomada de decisão mais assertiva. A habilidade de detectar rapidamente a falha e recuperar automaticamente o serviço para um componente redundante (YANG et al., 2011) exige mecanismos automatizados para tolerância a falhas. A arquitetura de gerenciadores de nuvem conta com um nível adequado de detecção, isolamento e recuperação de falhas, através da habilitação de sincronização distribuída de dados. Uma das estratégias utilizadas consiste em reiniciar a MV falhada para outro servidor diferente, dependendo da natureza da falha. Assim, nos mecanismos mais básicos (*e.g.*, reinício forçado), as informações do estado da MV não são preservadas (BAUER; ADAMS, 2012), sendo indicados apenas a serviços *stateless*, pois pode causar a perda de dados. Em serviços *stateful*, são utilizados mecanismos de HA mais avançados, que consistem na replicação dos dados transientes na hierarquia de memória de aplicações não-deterministas (MONDAL; MUPPALA; MACHIDA, 2016). Quando uma falha é detectada pelo mecanismo de HA, a réplica passa a assumir o controle imediatamente para minimizar a interrupção (o  $T_{mc}$  da Equação 2.1).

Também são aplicadas estratégias de HA para redundância dos componentes internos do hardware, como processador, disco, memória e interface de rede (JAIN, 2017), e na redundância de equipamentos computacionais e de suporte. Especificamente em sistemas distribuídos, a estratégias de HA baseadas na redundância de servidores, são a base para replicação de MVs (CULLY; et. al, 2008; DONG; et. al, 2013; SILVA, 2015) Nesse sentido, alocar estes servidores em ambientes sem ponto único de falhas (SPoF) correlatos, como as AZs, consiste em afirmar que a causa da falha de um servidor local não será a causa da falha do servidores remotos (CRITCHLEY, 2014; ENDO; et. al, 2016; GONCALVES et al., 2016-06), *i.e.*, são eventos independentes. Adicionalmente, a redundância nos equipamentos de rede e de suporte tornam-se primordiais (WANG; LI, 2015; SUH et al., 2017; CASELLAS et al., 2017; LEVALLE, 2018), garantindo que a comunicação, refrigeração e energia dos servidores não sejam afetados.

As estratégias destacadas para a presente Seção, requerem um alto nível de gerenciamento e automação, devido ao incremento na quantidade de componentes. Esse incremento nos componentes exigem um elevado consumo de energia, considerando que para replicar no mínimo um serviço para uma réplica, exigirá praticamente o dobro do consumo de energia, sem considerar as peculiaridades do impacto do geren-

ciamento. Nesse sentido, existem técnicas e estratégias que visam obter a eficiência energética em DCs de computação em nuvem.

### 2.2.2 Estratégias de eficiência energética

A literatura especializada relata técnicas e estratégias que objetivam a eficiência energética em diversos níveis de um DC (FELLER et al., 2012; YOU et al., 2017; CAMARGO et al., 2017). A nível de instalação, são relacionados os equipamentos de suporte, destacando-se a refrigeração (CAMARGO et al., 2016; CAMARGO; MIERS, 2016), e os sistemas de fonte de energia ininterrupta (UPS) e unidade de distribuição de energia (PDU) (RIEKSTIN et al., 2017). A nível de hardware dos equipamentos computacionais, são relacionados principalmente os servidores da camada de processamento e equipamentos de rede (ABAUNZA; HAMERI; NIEMI, 2018). Os equipamentos computacionais têm destaque na demanda do consumo de energia, devido seu alto desempenho em processamento de dados (RIEKSTIN et al., 2017). A nível de software, destaca-se o escalonamento de recursos através da camada de virtualização (DU; HE; MENG, 2014), que sob o domínio do CSP alcançam maior impacto de redução, quando comparadas as abordagens sob domínio do *tenant*. A literatura indica que linhar estratégias de eficiência energética a nível de hardware, como gerenciamento de servidores, e de software com o escalonamento de recursos, tem fornecido resultados mais expressivos do que individualmente.

A nível de hardware há duas principais estratégias encontradas na literatura (ZELKOWITZ, 2011): desativação dinâmica de equipamentos e escala dinâmica de desempenho (BOUDJADAR, 2017). A desativação dinâmica de equipamentos (TAKOUNA; MEINEL, 2014; BOUDJADAR, 2017) consiste em utilizar o modo *Advanced Configuration and Power Interface* (ACPI) para alterar seu estado entre ativo/inativo. A literatura especifica cinco estados de um servidor (BELOGLAZOV et al., 2010; TENG et al., 2017; DAYARATHNA; WEN; FAN, 2016):

- Ativo-ocioso: há apenas o processamento mínimo necessário aos módulos essenciais do sistema operacional (SO) ou *firmware*.
- Ativo-executando: há qualquer carga de processamento além do SO;
- Inativo-desligado: o equipamento não mantém dados em memória, não possui qualquer consumo de energia e demanda mais tempo para passar ao estado ativo;
- Inativo-Hibernando, mantendo todos os seus estados de memória persistidos, possui um mínimo consumo de energia para permitir que a troca para o estado ativo demande menos tempo; e
- Inativo-Suspenso, com alguns dos dados persistidos e outros em memória, consome pouco mais de energia do que quando hibernando, mas seu tempo de

troca para o estado ativo é ainda menor.

Naturalmente, modificar os estados de inativo para ativo exige um tempo, além de ocorrerem picos no consumo de energia durante a inicialização dos servidores (ZELKOWITZ, 2011). Tipicamente esse tempo é maior na mudança de estado desligado para ativo, do que quando em suspensão para ativo. Assim, é viável utilizar a abordagem conforme indicativos (preditivos ou estocásticos) do tempo entre as demandas de carga de trabalho (ZELKOWITZ, 2011). Nesse sentido, a desativação dinâmica é referenciada como uma das bases para obter a eficiência energética em métodos de escalonamento (ABDELSAMEA; et. al, 2017), como a consolidação de MVs.

O processador é o recurso que mais consome energia em servidores, podendo chegar a 60% para a família Intel Xeon (DAYARATHNA; WEN; FAN, 2016), por exemplo. Quando o objetivo é manter o equipamento ativado, mas adequar seu desempenho à carga recebida, utiliza-se a escala dinâmica de desempenho. Tipicamente, esta é uma estratégia que ajusta as características dos recursos de hardware (processador e memória) em relação ao seu consumo, que impacta em seu desempenho final. Entre as abordagens, destaca-se o *Dynamic Voltage and Frequency Scaling* (DVFS) (BOUDJADAR, 2017) que altera dinamicamente a escala de frequência e corrente, ora elevando o desempenho quando há demanda, ora reduzindo quando ocioso. De forma similar ocorre com a RAM, alterando sua frequência de barramento para reduzir o consumo de energia (TAKOUNA; MEINEL, 2014), ou utilizando o conceito de memória compartilhada (BOUDJADAR, 2017). Todavia, a literatura indica que as abordagens de escala dinâmica de desempenho resultam em eficiência energética menor quando comparadas à desativação dinâmica (AHMAD; et. al, 2015; YOU et al., 2017). Assim, quando estratégias utilizam ambas as abordagens de escala dinâmica de desempenho e desativação dinâmica, possibilita obter resultados ainda melhores (TENG et al., 2017; MALIK et al., 2015).

Em geral, sistemas distribuídos utilizam técnicas de escalonamento de tarefas (SCARPINITI; et. al, 2018), como o balanceamento de carga e a consolidação de servidores/MVs. O escalonamento objetiva a máxima utilização dos recursos, através da atribuição mais apropriada das tarefas aos recursos disponíveis (processamento, memória e armazenamento) (DU; HE; MENG, 2014). O balanceamento de carga é utilizado para melhorar a distribuição de cargas de trabalho entre processadores, tipicamente proporcionando uma utilização equilibrada de todos os recursos disponíveis (PADOIN, 2016). Neste caso, objetiva-se que todas as tarefas sejam igualmente distribuídas entre os recursos existentes, evitando que alguns recursos sejam sobrecarregados, enquanto outros sejam subutilizados. Por outro lado, a consolidação visa minimizar a quantidade de recursos disponíveis, combinando todas as cargas de tra-

balho na menor quantidade possível de recursos, permitindo que estes permaneçam em modo de consumo mínimo de energia. Esse consumo mínimo de energia ocorre ao aplicar as estratégias a nível hardware (BELOGLAZOV; BUYYA, 2012), como a desativação dinâmica de equipamentos (e.g., alterar estados entre ativo/inativo).

### 2.2.3 Considerações parciais sobre HA e eficiência energética

Em suma, observa-se uma grande variedade de abordagens de HA e eficiência energética analisadas separadamente. O escopo do presente trabalho consiste em nuvens computacionais IaaS, com foco no gerenciamento de MVs sob a perspectiva do CSP, relativo a responsabilidade compartilhada. Assim, elege-se como mecanismo de HA a replicação de MV, pela possibilidade de obter melhores resultados quando comparados à redundância de equipamentos. Entre as estratégias de eficiência energética, destaca-se a consolidação de MVs alinhada com a estratégia de desativação dinâmica de equipamentos, fornecendo maior capacidade em obter maior economia de energia. Adicionalmente, ambas abordagens são amplamente utilizadas na literatura (SCARPINITI; et. al, 2018; SECINTI; OVATMAN, 2018; SHARMA et al., 2017; HE; et. al, 2016), indicando consenso entre pesquisadores.

## 2.3 REPLICAÇÃO DE MVs

A literatura estabelece dois modos de replicação para MVs (KANZENBACH, 2014; HE; et. al, 2016; MILANI; NAVIMIPOUR, 2016b): o passivo e o ativo. No modo ativo, as requisições são executadas por todos os servidores, em que as MVs réplicas estão executando em paralelo as requisições dos usuários (ENDO; et. al, 2016), destacando-se o benefício da resposta rápida para minimizar o  $T_{mc}$ . No modo passivo, as MVs réplicas são atualizadas com os estados da hierarquia de memória da MV primária, após executar as requisições dos usuários, pode ser aplicado para serviços *stateful* não-deterministas, mas tem maior  $T_{mc}$ . Naturalmente, a replicação ativa demanda maior complexidade no gerenciamento e uso de recursos, quando comparada com a replicação passiva (FISCHER; MITASCH, 2006; ZHOU; SUN; LI, 2017). A replicação impacta diretamente na carga de trabalho, pois ao acrescentar mais réplicas, exige-se mais equipamentos, e demandam maior consumo de energia elétrica. Um exemplo prático de replicação ativa consiste em manter as MVs réplicas em *lock step* (CRITCHLEY, 2014), comparando a consistência dos dados entre si. Assim é possível detectar erros de execução, tipicamente através de consenso/votação dos resultados de maioria correta (DONG; et. al, 2013). Outras estratégias para replicação passiva baseiam-se em *checkpointing* (CAO et al., 2014; ZHOU; SUN; LI, 2017), ao armazenar regularmente os estados da MV ativa em réplicas passivas. O controle da

frequência dos *checkpoints* (SILVA, 2015) permite adequar o consumo de recursos com os requisitos de cada aplicação.

Após estabelecido o tipo de replicação de acordo com os requisitos do sistema crítico, o próximo passo consiste em estabelecer a quantidade de réplicas ideal para manter a HA. Além da quantidade de réplicas, estabelecer o local onde estas réplicas serão alocadas é primordial para estabelecer uma taxa de HA. A análise por diagrama de bloco de confiabilidade (RBD) é amplamente adotada para calcular a disponibilidade de sistemas complexos (STAPELBERG, 2009; SANTOS; et. al, 2017; MATOS; et. al, 2017), verificando se os componentes estão em série ou em paralelo. Quando componentes estão em série, *i.e.*, no mesmo SPoF, a disponibilidade combinada é o produto da disponibilidade ' $\mathbb{A}$ ' de seus  $P$  componentes. Em série, o resultado será disponibilidade menor do que a de seus componentes individuais e, portanto, não aplicado para HA. Para componentes em paralelo, a disponibilidade combinada é baseada no produto da indisponibilidade ( $\overline{\mathbb{A}} = 1 - \mathbb{A}$ ) de seus  $P$  componentes (COULOURIS et al., 2011). Quando os  $P$  componentes não compartilham o mesmo SPoF, então não haverá dependência entre as falhas que geram sua indisponibilidade. Ou seja, a ocorrência de uma falha no componente ativo não será a responsável pela falha no componente replicado, e portanto, a probabilidade de todos os componentes falharem ao mesmo tempo é reduzida (FISCHER; MITASCH, 2006; FORD; et. al, 2010; MORENO-VOZMEDIANO; et. al, 2017). Este axioma é proveniente da probabilidade de falhas de eventos independentes. Assim, quando em paralelo, a disponibilidade combinada é maior do que a disponibilidade média dos seus componentes individuais.

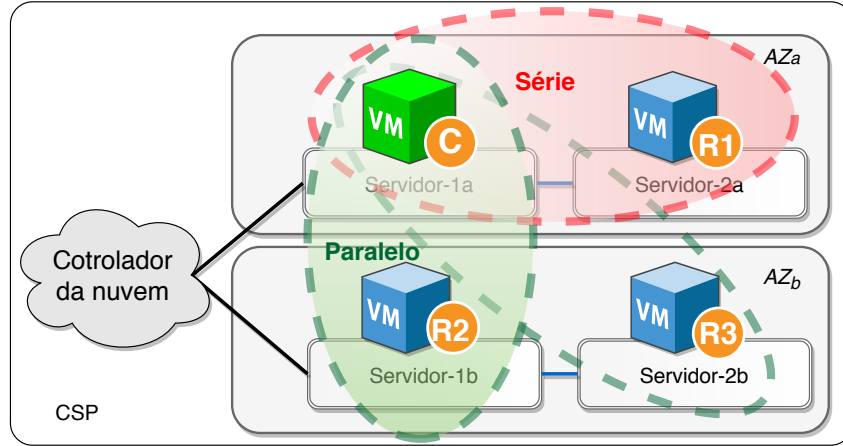
É possível aplicar o modelo de diagrama de blocos tanto para o cálculo da confiabilidade quanto da disponibilidade. Seus componentes são interconectados em série ou em paralelo (Figura 1), seguindo duas regras básicas:

- Se a falha de um componente faz com que o sistema representado torne-se inoperável, deve-se considerar as partes operando em série.
- Se a falha de um componente conduz a um outro componente assumir as operações da parte falhada, então deve-se considerar as partes operando em paralelo.

Dois componentes são considerados em série, se a falha de apenas uma das partes resultar em uma falha no sistema. Conforme a Figura 1, o sistema possui menor disponibilidade se houver falha no domínio da  $AZ_a$ , fazendo com que o  $Servidor_{1a}$  com a  $VM_{Crítica}$  e o  $Servidor_{2a}$  com a réplica  $VM_{R1}$  também falhem simultaneamente. Desse modo, a Equação  $\mathbb{A} = A_x \cdot A_y$  resulta em disponibilidade sempre menor do que a disponibilidade de seus componentes individuais, quando combinada por dois ou mais componentes em série.

Por outro lado, dois componentes são considerados em paralelo, quando uma das partes falhar e o sistema continuar operacional. Assim, o sistema em paralelo terá

Figura 1 – Componentes em série e paralelo.



Fonte: Elaborado pelo autor.

em uma falha no sistema apenas quando todos os componentes em paralelo falharem. Na Figura 1, se todo o domínio da  $AZ_a$  tiver uma falha, e o  $Servidor_{1a}$  com a  $VM_{Crítica}$  ficar indisponível, então o  $Servidor_{1b}$  com a réplica  $VM_{R2}$  ou o  $Servidor_{2b}$  com a réplica  $VM_{R3}$  continuarão disponíveis. No contexto de computação em nuvem, este conceito se traduz para um 'sistema que continua minimamente operacional, se um subconjunto das réplicas estiverem disponíveis'. A taxa de disponibilidade de cada AZ é determinada pelo símbolo  $\mathbb{A}$ , enquanto a taxa de indisponibilidade é seu complemento, e determinado pelo símbolo  $\overline{\mathbb{A}}$ , de tal modo que  $\overline{\mathbb{A}} = 1 - \mathbb{A}$ . Assim, a equação da disponibilidade final ( $\mathbb{H}\mathbb{A}$ ) de um sistema em paralelo é dada pelo complemento do produto da indisponibilidade de  $P$  componentes. Para considerar diferentes taxas  $\mathbb{A}$  que cada AZ pode assumir, aplica-se o produtório na Equação 2.2 com índices correspondentes ao número de paralelos  $P$ . Desse modo, considera-se a variabilidade que um conjunto de componentes  $\{\mathbb{A}_1, \mathbb{A}_2, \mathbb{A}_3, \dots, \mathbb{A}_k\}$  de características heterogêneas pode assumir.

$$\mathbb{H}\mathbb{A} = 1 - \prod_{i=0}^P (1 - \mathbb{A}_{(AZ_i)}) \mid P = \# \text{ de componentes} \quad (2.2)$$

A Equação 2.2 implica que a disponibilidade final obtida por executar  $P$  componentes em paralelo será maior do que a disponibilidade de seus componentes individuais ( $\mathbb{H}\mathbb{A} \geq \mathbb{A}_i$ ). A quantidade de componentes  $P$ , consiste na união do componente crítico que está sendo replicado com as suas  $r$  réplicas. A relação entre  $r$  e  $P$  é que  $P$  é o número de paralelos, que inclui o próprio componente crítico, enquanto  $r = P - 1$ , *i.e.*, apenas o número de réplicas. Em sistemas complexos, normalmente há diversos componentes dispostos em série e em paralelo, e devem ser calculados

decompondo-os por partes.

Considerando sistemas em que a interrupção de um subconjunto de componente leva a uma redução parcial da disponibilidade, mas mantém-se em níveis satisfatórios de operação, denomina-se disponibilidade parcial ou sistemas  $k$ -em- $P$  (SMIDT-DESTOMBES; HEIJDEN; HARTEN, 2004; DAVIES; DEMBIŃSKA, 2018). Assim, no arranjo  $k$ -em- $P$ , pelo menos  $k$  componentes devem estar operantes (de um total de  $P$  componentes) para que o sistema opere satisfatoriamente. Na literatura especializada (YEOW; WESTPHAL; KOZAT, 2010) também são utilizados este conceito para sistemas distribuídos. No contexto de computação em nuvem, a disponibilidade parcial de um serviço pode ser definida considerando o percentual de usuários afetados pela interrupção. Aplicações *Web stateless* replicados são exemplos deste tipo de sistema, considerando parcialmente disponível se 30% dos assinantes forem afetados, ou seja, de um total de 10 MVs, então há 3 de 10 com falha. Neste caso, as combinações são expressadas na Equação 2.3 através do coeficiente binomial, conceito baseado na teoria da probabilidade.

$$\mathbb{H}\mathbb{A}_{(n,k)} = \sum_{i=0}^k \left( \frac{n!}{i!(n-i)!} \right) \mathbb{A}^{n-i} (1 - \mathbb{A})^i \quad (2.3)$$

Considerando um sistema com  $n$  componentes, onde o sistema é considerado disponível quando pelo menos componentes  $n - k$  estão disponíveis (ou seja, não mais do que os  $k$  componentes podem falhar). Salienta-se que esta abordagem somente pode ser aplicada quando todos os componentes possuem a mesma taxa de disponibilidade  $\mathbb{A}$ .

## 2.4 CONSOLIDAÇÃO DE MVs

De modo geral, a consolidação de servidores é uma estratégia que consiste em selecionar processos (serviços) distribuídos em múltiplos servidores subutilizados, concentrando-os em poucos servidores (NATHUJI; SCHWAN, 2007). É tipicamente usada para reduzir custos operacionais com energia elétrica (DOW, 2016; MAZUMDAR; PRANZO, 2017). Porém, a computação em nuvem com a orquestração e virtualização de recursos propicia o isolamento entre MVs oportunizando o gerenciamento dos serviços (PROCACCIANTI, 2015). Assim, a consolidação de MVs (MACHIDA; KAWATO; MAENO, 2010; VARASTEHE; et al., 2017) consiste em migrar todas as MVs para a menor quantidade possível de servidores. Segundo a literatura especializada, a estratégia da consolidação de MVs possui um dos resultados mais significativos (BELOGLAZOV; BUYYA, 2012; MEDINA; GARCÍA, 2014; AHMAD; et. al, 2015; MADNI et al., 2016; MILANI; NAVIMIPOUR, 2016a; PATEL; VAGHELA, 2016; HAMEED; et. al, 2016; XU; TIAN; BUYYA, 2016), pois podem alcançar até 40% de redução no consumo

de energia. Assim, a consolidação de MVs é considerada como uma das principais práticas para prover a redução do consumo de energia em ambientes virtualizados (DAYARATHNA; WEN; FAN, 2016; SCARPINITI; et. al, 2018).

Plataformas de computação em nuvem tipicamente tem sua carga de trabalho dinâmica, variando constantemente a quantidade de requisições para alocar e desalocar MVs (BELOGLAZOV; BUYYA, 2012). Nesse sentido, a fragmentação de recursos pode levar à uma configuração em que servidores ficam subutilizados (TSO; JOUET; PEZAROS, 2016) enquanto outros permanecem sobreutilizados. Assim, a consolidação de MVs pode basear-se em duas abordagens que podem ocorrer em momentos distintos: o posicionamento inicial (ZHAO et al., 2017) e a migração de MVs (ABDEL-SAMEA; et. al, 2017). O posicionamento inicial consiste em selecionar o servidor mais adequado para instanciar uma nova MVs e só ocorre uma vez no ciclo de vida de uma MV. Já a migração visa a re-otimização, e consiste em mover uma MV em execução para outro servidor, visando aprimorar o objetivo geral da consolidação, podendo ocorrer diversas vezes após o posicionamento inicial. Aplicar apenas o posicionamento inicial dificilmente resultará uma boa aproximação do ótimo, tendo em vista a dinamicidade das desalocações (XU; TIAN; BUYYA, 2016; SHEN; et. al, 2017). Por outro lado, a re-otimização deve considerar a sobrecarga gerada pelas migrações (SCARPINITI; et. al, 2018).

Não há um consenso na literatura especializada sobre qual o melhor momento de consolidar (IHARA; LOPEZ-PIRES; BARAN, 2015; ZHAO et al., 2017; DHANOA, 2017), nem qual a ordem (se re-otimiza antes ou posiciona a MV). De fato, conforme a distribuição das MVs em execução e das características da carga da nuvem, pode-se obter resultados distintos. Assim, para cada proposta é importante a análise destas possibilidades para auxiliar na tomada de decisão. Como uma restrição à consolidação, a literatura estabelece a política de afinidade MV-servidor (SU; et. al, 2015). Esta política consiste em instanciar uma determinada MV em um servidor e esta MV não deve ser migrada durante sua execução (JACOBS; et. al, 2017). Adicionalmente, a afinidade MV-AZ é importante quando o *tenant* estabelece que uma determinada MV deve manter-se em execução apenas no domínio de uma AZ. Como a afinidade desabilita a migração da MV, impossibilita que, caso subutilizado, o seu servidor seja desativado (MORENO-VOZMEDIANO; et. al, 2017). Segundo relatório de uma organização (*tenant*) que utiliza o GCP (KOROTKOV, 2018), cerca de 20% de suas instâncias (contêineres) são preemptivas e restringem afinidade MV-Servidor, *i.e.*, 80% das instâncias podem ser migradas. Isso permite ao escalonador selecionar os servidores mais adequados para reduzir os custos operacionais, repassando custos menores para o *tenant*. Violar as políticas de afinidade implica em violar o SLA, pois é um requisito especificado pelo *tenant*.



O critério para selecionar um servidor para o posicionamento inicial é definido pelo algoritmo de escalonamento da nuvem. A exemplo do OpenStack, uma plataforma de computação em nuvem IaaS de *open source*, estão nativamente disponíveis três algoritmos de escalonamento: Round-Robin (circular), Random (aleatório) e Greedy (guloso). Portanto, a depender dos objetivos do posicionamento, se é para eficiência energética ou para balancear a carga, é que se define o algoritmo.

Na prática, quando executa-se a migração de uma MV, há a transferência de um conjunto de processos, com seus volumes de armazenamento e dados da hierarquia de memória, entre os servidores de origem e de destino. Com o uso de um sistema de armazenamento distribuído (ZHANG; GADDAM; CHRONOPOULOS, 2015; KONG; LUO, 2016), seu procedimento consiste apenas em migrar dados da hierarquia de memória, que permite menor impacto. De modo geral, a literatura especializada (MONIL; RAHMAN, 2016; NGENZI; SELVARANI; SUCHITHRA, 2016; DHANOA, 2017) destaca dois principais tipos de migração: ao vivo (*live* ou *online*) e fria (*cold* ou *offline*). A migração *live* apresenta menor interrupção ao usuário final durante sua execução, mas demanda mais recursos físicos (processamento e uso de entrada e saída (E/S)) e mais tempo para completar a sincronização final (NGENZI; SELVARANI; SUCHITHRA, 2016). Por outro lado, a migração *cold* exige que a MV seja desligada/-suspensa para que a migração seja realizada, o que implica em um tempo maior de interrupção, mas utiliza menos recursos para executar (FORSMAN et al., 2015). Para a consolidação de MVs, segundo (BELOGLAZOV, 2013), um processo de migração possui quatro etapas básicas:

- (i) Selecionar uma ou mais servidores que estão sobrecarregados;
- (ii) Selecionar as MVs destes servidores sobrecarregados;
- (iii) Selecionar um ou mais servidor mais apropriados para receber as MV; e
- (iv) Realizar a migração das MVs pela rede.

Quando consolidação de MVs é aplicada em servidores homogêneos, diversos autores a consideram como um exemplo prático do Problema do Bin Packing (PBP), de complexidade NP-Difícil (TENG et al., 2017; ZHAO et al., 2017; ZHOU; SUN; LI, 2017). Quando aplicada em servidores heterogêneos, outros autores consideram como o problema da mochila (*Knapsack Problem*) (SHEN; et. al, 2017), da mesma classe de complexidade do PBP. Essa classe de complexidade refere-se a questão de otimização: “Qual a menor quantidade de servidores possível para instanciar um determinado número de MVs?” A resposta a esta questão é considerado como valor ótimo, denominado *OPT*. Para evitar lidar com a complexidade destes problemas, a literatura propõe heurísticas para obter uma aproximação do valor ótimo (MARTELLO;

TOTH, 1990; RIECK, 2010; ZHANG et al., 2017). Estas heurísticas variam desde abordagens simples como os algoritmos de *Fit* (RIECK, 2010) até meta-heurísticas, como as bioinspiradas em colônias de formigas (FERDAUS et al., 2014) e baseadas em inteligência artificial. Nesse sentido, destaca-se a necessidade de selecionar uma heurística que seja compatível com as restrições do ambiente do problema. Uma boa prática é observar a relação custo-benefício de obter a resposta mais aproximada do valor ótimo em um tempo aceitável (SCARPINITI; et. al, 2018).

Uma análise realizada por (RIECK, 2010), consiste na execução de *benchmarks* específicos para o PBP, para verificar a aproximação dos resultados com o valor ótimo  $OPT$  e o tempo de execução dos algoritmos. São utilizadas seis principais heurísticas, com e sem otimizações de código, como o *Max-Rest*, *Next-Fit*, *Next-Fit-Decreasing*, *Best-Fit*, *First-Fit* e o *First Fit Decreasing* (FFD). Os resultados mostram variação na aproximação do valor ótimo e no tempo de execução entre essas heurísticas. Entre estas opções, o FFD destaca-se por apresentar os valores mais próximos do ótimo em um tempo de execução aceitável, além de ser amplamente utilizado na literatura para o contexto da consolidação de MVs (USMANI; SINGH, 2016; LOPEZ-PIRES; BARAN, 2015; AKHTER; OTHMAN, 2016). Devido sua necessidade de ordenar o vetor de MVs antes de instanciá-los nos servidores físicos, a versão genérica do algoritmo FFD é caracterizado como *offline*, *i.e.*, precisa ter conhecimento sobre todas as variáveis para a tomada de decisão. Na análise de otimalidade, em geral o FFD apresenta, no pior caso, um comportamento  $FFD(L) \leq (11/9) * OPT(L) + c$  (RIECK, 2010; DÓSA et al., 2013), sendo que  $c$  é uma constante que pode variar conforme as técnicas de implementação. Por exemplo, dado um valor ótimo  $OPT(L) = 9$ , então o FFD pode fornecer, no máximo, um  $FFD(L) = 11 + c$ . as técnicas de implementação refere-se ao procedimento de ordenação inicial, que pode ser otimizado com as estratégias *HeapSort* ou *CountingSort* (RIECK, 2010; ALAHMADI et al., 2014).

Devido sua necessidade de ordenar o vetor de MVs antes de instanciá-los nos servidores, o algoritmo FFD é caracterizado como um algoritmo *offline*. Porém, como o FFD possui uma visão global para a sua tomada de decisões, permite-se obter melhores resultados quando comparado com uma abordagem de perspectiva mais restrita, como ocorre com os algoritmos *online*. Observando o presente cenário de aplicação do FFD: plataformas de nuvem IaaS, observa-se a necessidade de um algoritmo *online* para atender imediatamente cada uma das requisições. Todavia, deve-se observar que quando há diversas requisições que ocorrem no mesmo instante de tempo, obrigatoriamente será criada uma fila de requisições. Assim, em uma abordagem híbrida, o FFD tem como entrada a fila instantânea de requisições, executada continuamente conforme as requisições *online*. Esta abordagem híbrida permite unir os benefícios de um algoritmo *online* que é atender *quasi*-instantaneamente as requisições, e *offline*

que é encontrar os melhores resultados. Deste modo, o FFD deve ter como entrada uma lista  $\mathcal{V}$  com  $n$  requisições e considerar a ocupação do servidores atualmente em execução.

## 2.5 CONFLITO ENTRE A CONSOLIDAÇÃO E REPLICAÇÃO DE MVs

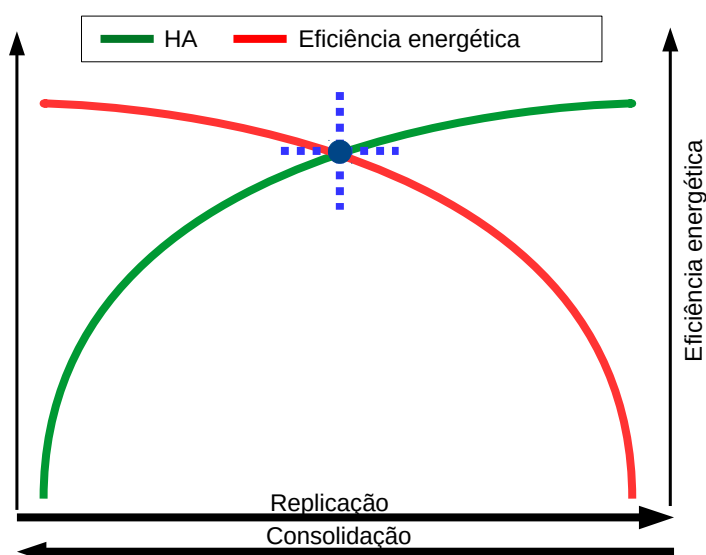
A discussão isolada dos mecanismos de HA e das estratégias de eficiência energética mostram que prover HA afeta a eficiência energética. Com base na fundamentação teórica, a replicação de MVs causa uma sobrecarga de trabalho nos servidores, e em alguns cenários, necessitam-se mais servidores para suprir esta demanda. Nesse sentido, a consolidação de MVs possibilita reorganizar essas MVs no menor número de servidores, possibilitando a redução no impacto energético. Todavia, utilizar ambas as abordagens concomitantemente no mesmo espaço de busca é conflitante devido suas inerentes restrições.

A discussão sobre este *trade-off*, tem princípio na inerente restrição de anti-afinidade entre uma MV crítica e suas respectivas réplicas, pois estas não devem ser alocadas no mesmo SPoF (JAMMAL; KANSO; SHAMI, 2015; DOW, 2016; ALLYBOKUS; et. al, 2018; ROSENDO; et. al, 2018). Manter uma MV crítica com suas réplicas no mesmo servidor ou na mesma AZ reduz drasticamente a taxa de disponibilidade final do sistema e inviabiliza determinar a taxa de HA do serviço que estas MVs fornecem. Isso ocorre devido ao conceito de probabilidade de eventos independentes, que pela análise por RBD as réplicas estão em série com as possíveis falhas da AZ. Assim, ao desconsiderar a anti-afinidade para a consolidação e a replicação de MVs simultaneamente em uma AZ, o algoritmo de consolidação pode migrar a MV crítica e suas réplicas para o mesmo servidor.

Ainda que a estratégia de consolidação da AZ considere a justaposição do *pool* de replicação no mesmo servidor, adicionar esta restrição na estratégia de consolidação pode impedir resultados mais próximo do ótimo (*OPT*). Adicionalmente, quando a replicação e a consolidação ocorrem a nível de uma região, tipicamente composta por duas ou mais AZs, viola-se a restrição de afinidade (MV-AZ) determinada pelo *tenant* em SLA. Nesse sentido, o *tenant* afirma que sua MV deve ser alocada em uma determinada AZ, e não pode ser migrada pelo procedimento de consolidação. Assim, abordagens tradicionais que possuem o *trade-off* entre HA e eficiência energética buscam um ponto de equilíbrio, de modo a não fornecer a máxima eficiência energética, nem a máxima HA. Esse *trade-off* é base da pesquisa de (SHARMA et al., 2016), cujo ponto em azul na Figura 2, representa o ponto de equilíbrio entre HA e eficiência energética.

Observa-se na Figura 2 à proporção em que se aumenta o número de répli-

Figura 2 – Conflito entre consolidação e replicação para eficiência energética e HA.



Fonte: Adaptado de Sharma et al. (2016).

cas, aumenta-se a taxa de HA e se reduz a eficiência energética. Essa redução da eficiência é devido a necessidade de mais servidores ativos para fornecer o mesmo serviço, mas em HA. Por outro lado, à medida em que aumenta-se a consolidação, *i.e.*, resultados mais próximos do ótimo, a tendência é reduzir a HA, em parte devido a menor quantidade de réplicas, em parte devido as violações de anti-afinidade. Assim, de modo genérico, as linhas em verde e vermelho que representam HA e eficiência energética possuem comportamento antissimétrico entre si.

## 2.6 DEFINIÇÃO DO PROBLEMA

Na prática, infraestruturas de nuvem públicas como a AWS e GCP são organizadas por regiões, em que cada região possui ao menos três AZs, sendo algumas regiões com mais de três AZs<sup>1</sup>. Para garantir que as MVs do *pool* de replicação não sejam instanciadas no mesmo SPoF (mesma AZ), é estabelecida a restrição de anti-afinidade (MV-MV) (MORENO-VOZMEDIANO; et. al, 2017; ALLYBOKUS; et. al, 2018; WANG et al., 2018), mas a nível de AZ. Arquiteturas multi-AZ também são uma realidade para o modelos de implantação de nuvens privadas, principalmente as que possuem serviços críticos. Comumente, organizações com filiais geograficamente distribuídas possuem suas infraestruturas de Tecnologia da Informação (TI) (*on-premise*) isoladas das demais filiais (MATLOOBI; ZOMAYA, 2016). Infraestruturas de TI híbri-

<sup>1</sup> Fonte das organização AWS: <<https://aws.amazon.com/pt/about-aws/global-infrastructure>> e GCP <<https://cloud.google.com/compute/docs/regions-zones>>.

das, que distribuem as cargas de trabalho entre nuvem privada e em nuvem pública também contam com os benefícios da arquitetura multi-AZ (WANG et al., 2014).

Entretanto, no meio científico, mecanismos de HA são tipicamente aplicados sem considerar arquiteturas multi-AZ, o que dificulta a análise da disponibilidade final, quando este é objetivo de pesquisa (MORENO-VOZMEDIANO; et. al, 2017). Nesse sentido, a literatura especializada (ENDO; et. al, 2016) tem indicado que existem diferentes mecanismos de HA que produzem resultados distintos. Assim, sua aplicabilidade considera diversos critérios como tipo de aplicação do *tenant* e ambiente/infra-estrutura. Nesse sentido, são utilizadas métricas para avaliar o impacto do mecanismo de HA, que podem ser pertinentes tanto ao *tenant* quanto ao CSP.

Sob a perspectiva do *tenant*, contabilizar as violações de SLA (SLAV) permite mensurar a consistência do objetivos firmados entre o CSP e o *tenant*, como a *Quality of Service* (QoS), taxa de disponibilidade, políticas de (anti)afinidade e a rejeição de processamento de recursos reservados (GHOBAEI-ARANI; JABBEHDARI; POURMINA, 2016). O CSP avalia o impacto no uso de recursos, que causa o aumento no consumo de energia, afetando seu custo operacional. Assim, as estratégias de eficiência energética são utilizadas para reduzir esse impacto, normalmente reacionados a problemas complexos de escalonamento, como é o caso da consolidação. Entretanto, heurísticas são utilizadas para reduzir essa complexidade (RIECK, 2010; BELOGLAZOV; BUYYA, 2012; FERDAUS et al., 2014; MONIL; RAHMAN, 2016), permitindo obter resultados próximos do ótimo, mas em tempo aceitável de execução.

A definição da problemática auxilia na definição do escopo da presente pesquisa, permitindo organizar os conceitos utilizados em critérios, constituindo uma base comparativa pra trabalhos correlatos. Nesse sentido, soluções que tratam simultaneamente o conflito entre HA e eficiência energética, podem ser relacionadas com os seguinte critérios:

- O mecanismo de HA que é utilizado;
- As estratégias de eficiência energética;
- As heurísticas desenvolvidas e/ou comparadas;
- As métricas usadas para avaliar o desempenho das ações;
- A solução utiliza o conceito de multi-AZ?
- A solução contabiliza as violações de SLA? e
- A solução considera as políticas de afinidade/antiafinidade?

## 2.7 TRABALHOS CORRELATOS

Para a execução da pesquisa bibliográfica sistemática (PBS), são utilizadas as metodologias apresentados por (KITCHENHAM, 2004) e (CONFORTO; AMARAL;

SILVA, 2011), cujo objetivo é prover a reprodutibilidade, consistência, e integridade das discussões. Como o presente trabalho baseia-se nas abordagens de eficiência energética e mecanismos de HA em computação em nuvem, então estabelece-se que os trabalhos correlatos devem abordar ambas as linhas. PBS existentes (SHARMA et al., 2016; SHISHIDO; ESTRELLA, 2017) relatam poucos trabalhos que abordam simultaneamente estratégias de eficiência energética com HA, quando comparadas com os temas isoladamente. Assim, para encontrar esses trabalhos correlatos, são utilizados três mecanismos de buscas: IEEE Explore, ACM Digital Library e Scopus. Estes mecanismos são elencados por (BUCHINGER; CAVALCANTI; HOUNSELL, 2014), pelos critério de acessibilidade dos trabalhos e funcionalidades avançadas de busca. A pesquisa é delimitada apenas aos trabalhos mais recentes, com data de publicação em um período de dois anos, *i.e.*, entre janeiro de 2016 e junho de 2018. São definidas as questões de pesquisa, as estratégias de busca, o procedimento de seleção dos trabalhos e a análise dos trabalhos selecionados.

### 2.7.1 Questões de pesquisa

Estabelecer as questões de pesquisa auxilia na definição das palavras-chave, permitindo tornar a estratégia de busca reprodutível. Esta etapa também é utilizada para o refinamento da seleção dos trabalhos encontrados. Com base nas duas linhas de pesquisa: consolidação e replicação de MVs, são estabelecidas cinco questões, com suas respectivas motivações.

	<b>Questão</b>	<b>Motivação</b>
$QP_1$	Quais as abordagens utilizadas para prover alta disponibilidade para MVs em computação em nuvem?	Encontrar os mecanismos mais utilizados pelas pesquisas.
$QP_2$	Quais as abordagens utilizadas para obter a eficiência energética em um ambiente de computação em nuvem?	Encontrar as estratégias que permitem reduzir o consumo de energia.
$QP_3$	Quais as principais métricas utilizadas para analisar os resultados de uma estratégia de eficiência energética?	Encontrar métricas relacionadas às violações de SLA, ao consumo de energia e de recursos computacionais.
$QP_4$	Quais as restrições devem ser consideradas ao aplicar HA e eficiência energética?	Estabelecer uma base que limite a aplicabilidade de ambas as linhas de pesquisa.
$QP_5$	Quais são as características dos ambientes em que são aplicadas a HA e eficiência energética?	Definir as arquiteturas dos ambientes de nuvem em que são aplicadas as pesquisas.

Tabela 2 – Questões de pesquisa e motivações.

Na Tabela 2, as duas primeiras questões genéricas, relativas HA e eficiência

energética, e três últimas são relativas a ambas as linhas de pesquisa. Embora sejam de simples formulação, elas definem o escopo necessário para discutir os trabalhos, conforme a motivação apresentada pelas questões de pesquisa. Com base nas questões de pesquisa propostas na Tabela 2, é selecionado um conjunto de palavras-chave que compõem a *string* de busca: Para o refinamento das buscas, são estabelecidas as *strings* de busca: (“*cloud computing*”  $\wedge$  “*virtual machine*”  $\wedge$  “*energy efficiency*”  $\wedge$  “*high availability*”).

### 2.7.2 Procedimento de seleção dos trabalhos

A metodologia usada para selecionar ou excluir os trabalhos encontrados pela estratégia de busca ocorre com a definição de critérios de inclusão e exclusão. Os resultados iniciais da busca normalmente retornam uma quantidade considerável de trabalhos, sendo necessário reduzir esse número para alcançar os trabalhos mais consistentes. Deste modo, a Tabela 3 relaciona os critérios de inclusão e exclusão usados para este filtro.

Critérios de Inclusão		Critérios de Exclusão	
$CI_1$	Implementa em infraestrutura de nuvem	$CE_1$	Considera outros ambientes que não seja nuvem
$CI_2$	Considera eficiência energética com HA	$CE_2$	Há apenas uma ou outra linha de pesquisa
$CI_3$	Publicado com revisão aos pares	$CE_3$	Publicado em locais não verificáveis
$CI_4$	O trabalho está publicamente acessível	$CE_3$	Publicação indisponível em um dos mecanismos de busca
$CI_5$	Os trabalhos devem ser únicos	$CE_5$	Trabalhos duplicados ou similares entre os mecanismos de busca

Tabela 3 – Critérios de inclusão e exclusão dos artigos selecionados.

Mesmo que estes critérios de exclusão possam inicialmente impedir a adição de um trabalho de outras fontes para ser avaliado, em caso de haver grande coerência com os objetivos da presente pesquisa. Isso permite que, mesmo com o rigor de uma pesquisa sistemática, trabalhos externos poderão ser reconsiderados.

### 2.7.3 Análise dos trabalhos levantados

Após retornar um total de 36 trabalhos entre os mecanismos de busca, são aplicados os critérios de inclusão e exclusão, que consistem em conter os sete critérios definidos anteriormente na Seção 2.6. Após a coleta e tabelamento dos dados, todos os artigos passaram por uma separação de unicidade (critério de exclusão  $CE_5$ ), sendo removidos um total de doze trabalhos duplicados, encontrados em mais de um

mecanismo de busca. Como os mecanismos de busca permitem adicionar um filtro de data de publicação, então este critério já é validado durante a fase inicial. Nesse sentido, aplicar estes critérios em análise nos títulos, *keywords* e resumo dos artigos indexados, resultaram em sete artigos para avaliação completa.

A Tabela 4 apresenta uma relação entre os trabalhos correlatos, estabelecendo como base comparativa os sete critérios da Seção 2.6. As quatro primeiras colunas são constituídas por questões abertas, devido a variedade de abordagens de HA, eficiência energética (EE), heurísticas implementadas e comparadas e as métricas. As três últimas colunas são questões fechadas, com respostas variando entre Não e Sim, sendo Sim\* indicativo que não é utilizado especificamente o termo, mas sim o conceito. As palavras em destacadas em negrito são as abordagens consideradas para implementação no presente trabalho.

Tabela 4 – Relação entre os trabalhos correlatos e principais critérios.

Referência	HA	EE	Heurística	Métricas	Multi-AZ	SLAV	Afinidade
Enokido e Takizawa (2016)	Replicação	Balanceamento de carga	Round-Robin, LB	Consumo de energia, tempo de transmissão	Não	Não	<b>Sim*</b>
Sahoo e Goswami (2017)	<b>Replicação</b>	Ativação e Passivação	Algoritmo Genético	Probabilidade, custo	<b>Sim</b>	Não	Não
Li et al. (2017)	Backup	Tempo compartilhado	N-armed bandit	Nº backups, tempo ocioso	Não	Não	Não
Shen e et. al (2017)	Realocação	Posicionamento Inicial	FF, BF, WF, CSP, MTHM, Water-Filling	Eficiência de Empacotamento	Não	Não	<b>Sim</b>
Spinnewyn e et al. (2017)	<b>Replicação</b>	Posicionamento	Programação Dinâmica	Probab. de falha e tempo médio para perda de dados	Uma AZ	<b>Sim</b>	<b>Sim*</b>
Sharma et al. (2017)	<b>Replicação</b> de serviços	<b>Consolidação</b>	BFD, LB	Confiabilidade, consumo/perda de energia, lifespan	Uma AZ	Não	Não
Secinti e Ovatman (2018)	Recuperação reativa	<b>Consolidação</b>	Não específica	Consumo de energia, Nº migrações	Não	<b>Sim</b>	Não

Fonte: Elaborado pelo autor.

Na Tabela 4, a coluna HA indica o mecanismo utilizado para prover tolerância a falhas e EE as estratégias de eficiência energética para redução de custos. Enquanto três trabalhos (SAHOO; GOSWAMI, 2017; SPINNEWYN; et al., 2017; SHARMA et al., 2017) utilizam a replicação de MVs, apenas dois trabalhos (SHARMA et al., 2017; SECINTI; OVATMAN, 2018) utilizam a consolidação de MVs. Porém, apenas um trabalho (SHARMA et al., 2017) aplica simultaneamente a consolidação de MVs e replicação, porém é a replicação de serviços, que ocorre internamente a uma MV, *i.e.*, ocorre sob a perspectiva do *tenant*. A coluna das heurísticas e das métricas apresentam variação de adoção, pois naturalmente isso ocorre devido a abordagem que o pesquisador dá ao problema. Para os critérios fechados, arquitetura Multi-AZ é pouco utilizado entre os trabalhos levantados com uma ocorrência juntamente com SLAV que possui duas



ocorrências. Por fim, três ocorrências relativas às restrições de (anti)afinidade são consideradas, mesmo que em dois casos não sejam efetivamente utilizados o termo 'afinidade', sendo identificado em análise.

Com auxílio da PBS aplicada na análise dos trabalhos correlatos, permite-se afirmar que os critérios estabelecidos na problemática da presente pesquisa não são efetivamente tratados entre os trabalhos correlatos. Destaca-se o fato destes trabalhos correlatos não utilizarem consolidação de MVs e replicação de MVs devido seu inerente conflito. Adicionalmente, conceitos de multi-AZ e afinidade são pouco explorados entre esses trabalhos, mas comumente abordados em pesquisas especializadas em tolerância à falhas (MORENO-VOZMEDIANO; et. al, 2017; UNUVAR; et. al, 2014; DAVID, 2014; BORGOLTE et al., 2018). Finalmente, não foram encontrados nos trabalhos relacionados, indícios do conceito de HA sob demanda, compreendida no contexto de análise por RBD ou probabilidade de eventos independentes. Assim, mostram-se indícios de uma lacuna de pesquisa ao relacionar todos os conceitos apresentados na fundamentação teórica, que constituem a presente proposta.

## 2.8 CONSIDERAÇÕES PARCIAIS

Com base nas premissas adotadas no decorrer da fundamentação teórica, a replicação de MVs destaca-se como um mecanismo para prover HA em ambientes virtualizados, sendo objeto de pesquisa na academia com aplicações na indústria. A replicação possibilita reduzir o tempo de manutenção de um sistema falhado para outro, principalmente no contexto de computação em nuvem, devido sua capacidade de orquestração automatizada de recursos. Com o objetivo de calcular a taxa de disponibilidade resultante da replicação, a literatura indica o uso da análise por RBD aplicado a probabilidade de eventos independentes. Assim, réplicas devem estar em SPoF diferentes para que seja calculada a taxa de HA final de um sistema replicado.

Em contrapartida ao impacto causado pela replicação de MVs, estão as abordagens de gerenciamento de MVs que permitem reduzir o consumo de energia. Assim, a consolidação de MVs destaca-se por fornecer a redução no consumo de energia, além de reduzir a fragmentação de recursos. Sua ampla adoção em pesquisas relacionadas com eficiência energética implica em diversas estratégias, que variam conforme a aplicação de algoritmos (tipicamente heurísticos, como o FFD) que resultam em efeitos adversos.

Entre esses efeitos adversos, encontram-se as violações de SLA, relacionados com as restrições de afinidade e anti-afinidade por efeito da reorganização da relação VM-Servidor e VM-VM, respectivamente. Nesse sentido, define-se o *trade-off* entre a replicação e a consolidação de MVs. Ou seja, ao considerar um mesmo

ambiente virtualizado, quanto mais consolidação, mais violações de SLA, que resulta em menor capacidade de replicar MVs entre servidores diferentes (e manter essa configuração). Assim, a literatura indica que deve-se haver o equilíbrio entre as duas abordagens ou então separá-las em camadas diferentes, com o auxílio de arquiteturas multi-AZs.

Trabalhos correlatos focados em publicações recentes indicam que a replicação e a consolidação são normalmente aplicadas em separado, sendo poucos trabalhos que aplicam ambas em conjunto. A discussão destes trabalhos auxiliam para a definição de métricas, e trazem uma contribuição para análise das diferentes estratégias utilizadas. Desse modo, conceitos de multi-AZ e afinidade são pouco explorados entre esses trabalhos, mostrando que há uma lacuna de pesquisa, explorada pelo presente trabalho.

### 3 CHAVE

A solução proposta, denominada CHAVE, visa fornecer um mecanismo de alta disponibilidade (HA) sob demanda e simultaneamente reduzir seu impacto de custo, aplicando a estratégia de consolidação de máquinas virtuais (MVs) para obter eficiência energética e de uso de recursos. Porém, para alcançar estes objetivos sem que um mecanismo interfira na eficácia do outro, e sem que ambas as estratégias violem o acordo de nível de serviço (SLA) firmado com o *tenant*, CHAVE baseia-se em conceitos de arquitetura multi-zona de disponibilidade (AZ), que atualmente constituem um padrão adotado nas infraestruturas de Provedores de Serviços em Nuvem (CSPs) (UNUVAR; et.a l, 2014). CHAVE traz uma proposta para o *trade-off* HA-energia através de uma solução baseada em arquitetura de sistemas distribuídos, composta de múltiplas AZs. Desse modo, duas abordagens são executadas em níveis diferentes dessa arquitetura: a replicação de MVs ocorre entre AZs distintas, enquanto a consolidação ocorre internamente a cada AZ.

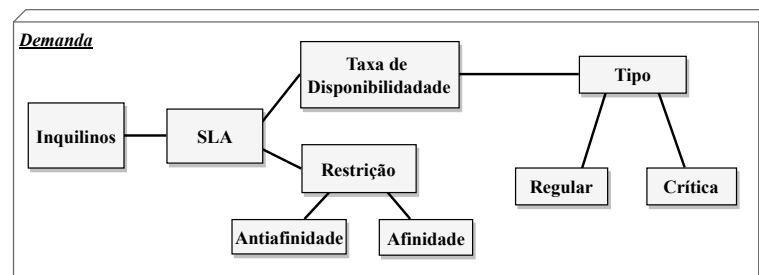
A restrição de anti-afinidade constitui uma das bases para que o mecanismo de HA do CHAVE forneça o conceito de HA sob demanda. Este conceito baseia-se na teoria da probabilidade de eventos independentes, em conjunto com a análise por diagrama de bloco de confiabilidade (RBD) para verificar se os componentes críticos estão em série ou paralelo. Desse modo, habilita-se calcular a taxa de HA de um determinado serviço que está sendo paralelamente executado em duas ou mais instâncias em um *pool* de replicação. Na prática, a HA sob demanda ocorre quando necessita-se de uma taxa de disponibilidade específica e que seja maior que a taxa regularmente oferecida pelo CSP (tipicamente a taxa de uma AZ) cujo valor é estabelecido em SLA. Assim, CHAVE calcula a quantidade ideal de réplicas com base nas taxas de disponibilidade das AZs disponíveis para cumprir com a demanda de HA.

A estratégia de consolidação de MVs tem sua atuação limitada internamente a cada AZ, *i.e.*, as etapas de posicionamento inicial e a re-otimização por migração consideram apenas os recursos da AZ em que as MVs já estão alocadas. Desse modo, evita-se que a consolidação seja restringida pela replicação, permitindo que a heurística selecionada atue de modo a encontrar resultados mais próximos do ótimo. Assim, desvincular as abordagens de consolidação e replicação de MVs em camadas distintas permite-se obter uma solução agnóstica às heurísticas adotadas, permitindo que sejam adotados distintos algoritmos de consolidação de MVs. O presente capítulo apresenta as principais características da solução CHAVE. Inicialmente é realizada uma especificação da arquitetura, com fluxogramas, casos de uso, e os requisitos funcionais e não-funcionais. Apresentado o formalismo utilizado no decorrer do trabalho.

### 3.1 ARQUITETURA

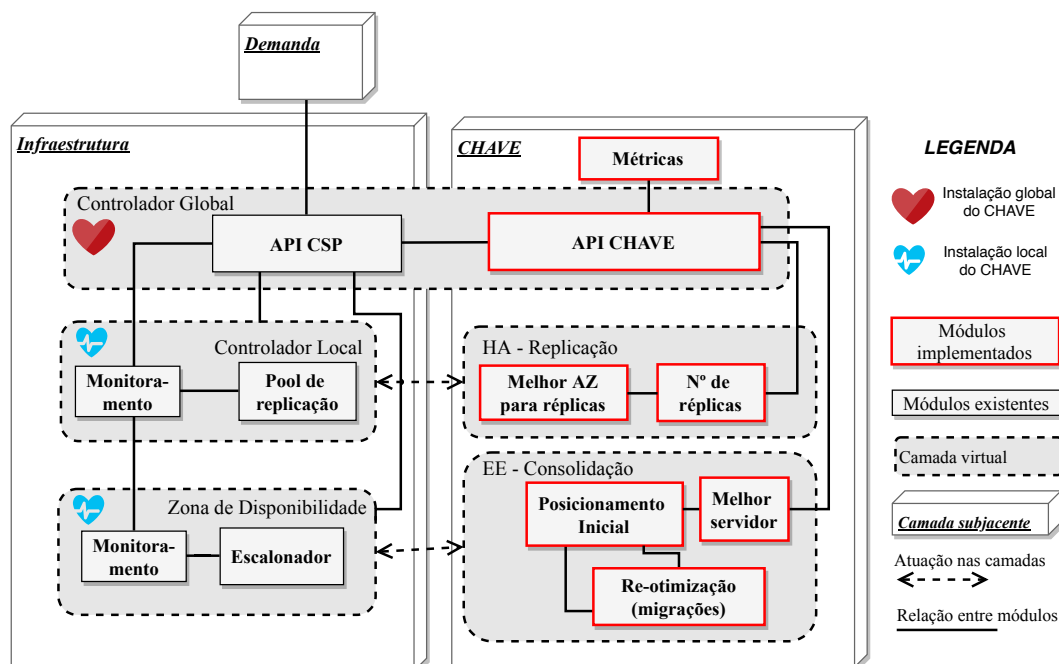
CHAVE é especificado por arquitetura modular, cujo objetivo é permitir que as suas funcionalidades sejam implementadas de acordo com as respectivas camadas da infraestrutura do CSP. A demanda gerada pelos *tenants* é considerada como um componente externo à arquitetura, pois constitui a entrada para as API do CSP e do CHAVE, referindo-se às requisições para criar MVs (Figura 3). As necessidades do *tenants* e as características do CSP são mapeadas em um SLA, que para o escopo do CHAVE destacam-se as taxas de disponibilidade, taxa de HA para as MVs críticas e políticas de afinidade. Cada nova requisição é formada por uma MV, com sua demanda por recursos computacionais (*sabor/flavor*) e por uma taxa de HA. Essa demanda é classificada pelos tipos: crítica se a requisição demanda HA, ou regular caso contrário.

Figura 3 – Representação da demanda para o CSP e CHAVE.



Fonte: Elaborado pelo autor.

Figura 4 – Arquitetura CHAVE: consolidação e replicação de MVs integrado ao CSP.



Fonte: Elaborado pelo autor.

A arquitetura representada pela Figura 4, é organizada em duas camadas subjacentes, que consistem na relação entre CHAVE e a infraestrutura do CSP. As cama-

das virtuais, constituem um agrupamento entre os módulos do CHAVE que atuam nos módulos da infraestrutura do CSP. As funcionalidades do CHAVE estão agrupadas em três principais camadas virtuais: o controle global, a replicação de MVs referente a HA, e consolidação de MVs referente a eficiência energética (EE). Conforme a Figura 4, os símbolos de coração indicam que CHAVE está instalado nas três camadas virtuais, subjacentes ao CSP. Enquanto que no controlador global mantém-se em comunicação direta com a *Application Programming Interface* (API) do CSP, realizando nas demais camadas o controle local e monitoramento da replicação e consolidação. Os módulos grifados em vermelho, que estão sobre a camada subjacente do CHAVE, são implementações do presente trabalho. Entretanto, este é um exemplo de arquitetura, que pode ser readequada quando implementada na prática, pois a arquitetura é modular para prover portabilidade para diversas infraestruturas de nuvem.

Com o objetivo de manter-se agnóstico ao CSP, CHAVE é desenvolvido para permitir integração à qualquer CSP via API, de modo que o controle e orquestração do CSP devem ser acessíveis em nível administrativo. Entre as funcionalidades desta API, destaca-se o módulo de métricas que atua em nível global, responsável tanto obter dados do monitoramento da infraestrutura, quanto das ações do CHAVE, persistindo estes dados em uma base de dados. A camada de replicação é responsável por definir o número ideal de réplicas para as MVs críticas, bem como selecionar as AZs mais adequadas para instanciar cada réplica.

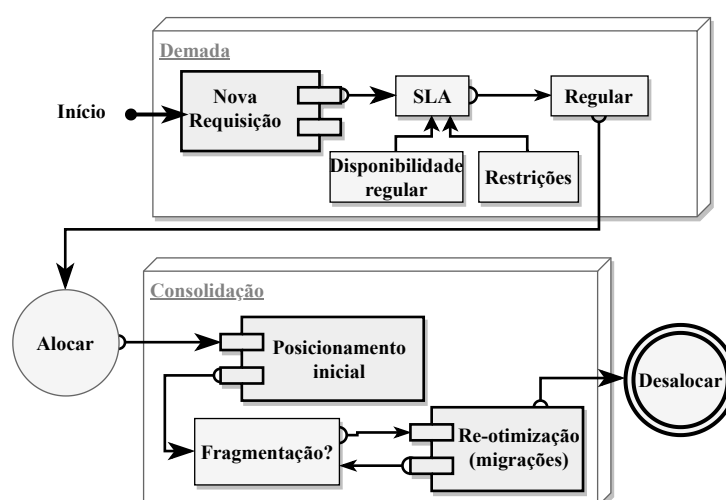
Habitualmente, o CSP organiza essas réplicas em um pool de replicação, monitoradas para atuarem de acordo com a técnica de replicação adotada. Na camada de consolidação, em um primeiro momento cada requisição é analisada para que seja selecionado o servidor mais adequado, objetivando que o posicionamento inicial seja otimizado. A cada mudança de configuração (relação MV-servidor) de cada AZ, CHAVE analisa se é possível obter alguma otimização. Quando positivo, é acionado o gatilho para re-otimizar a AZ através de migrações de MVs, mantendo-se nesse ciclo. O escalonador do CSP é responsável pelo controle do ciclo de vida das MVs, e tem seu algoritmo de escalonamento controlado pelo CHAVE, principalmente pela tomada de decisão para a escolha do melhor servidor.

### 3.1.1 Casos de uso

Genericamente, CHAVE possui dois principais casos de uso, que variam conforme o tipo de requisição: MV regular ou MV crítica. Para ambos os casos, são definidas as restrições e um SLA conforme a taxa de disponibilidade. Esta taxa pode ser regular (a mesma da AZ selecionada), e segue com o fluxo para requisições do tipo regular, ou uma taxa de HA, e segue para o fluxo de requisições críticas. Primeiramente, o caso de uso de uma MV regular, representado pela Figura 5, ocorre quando

já está especificada a AZ em que será instanciada, mantendo-se apenas no fluxo dos módulos referentes a consolidação. Após ser alocada, uma MV regular é inicialmente posicionada no servidor mais adequado, selecionado pelo CHAVE, e sempre que há algum gatilho que indique fragmentação da AZ, então esta pode ser migrada para outro servidor, até que seja desalocada no fim do seu ciclo de vida.

Figura 5 – Fluxograma para uma requisição do tipo regular, com fluxo apenas na camada de consolidação de MVs.

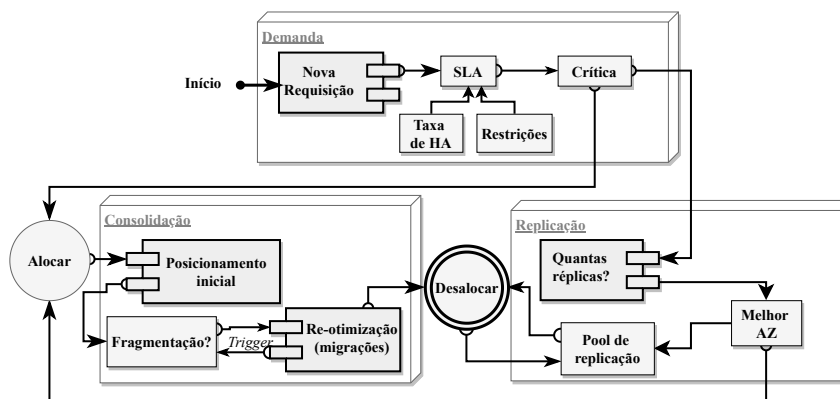


Fonte: Elaborado pelo autor.

Já as requisições de MVs críticas, tem seu fluxo definido na Figura 6. Inicialmente, na camada de replicação é estabelecido o número ideal de réplicas para atender a demanda de HA especificada em SLA. São selecionadas as AZs mais adequadas para serem instanciadas as réplicas, que são organizadas em um pool de replicação. Nesse instante, a MV crítica e suas réplicas são alocadas em cada AZ, cujo a camada de consolidação realiza o posicionamento inicial, e ficam no ciclo de re-otimização até que sejam desalocadas. Por fim, consulta-se o *pool* de replicação para notificar todas as réplicas a serem desativas, liberando recursos para a cada AZ.

Salienta-se que utilizar a taxa de HA das MVs críticas para a tomada de decisão sobre o número de réplicas a serem instanciadas destaca-se como uma contribuição do presente trabalho, constituindo o conceito de HA sob demanda. As melhores AZs para alocar as réplicas são selecionadas conforme algoritmo de balanceamento de carga, e tem como critério de desempate a AZ com maior taxa de disponibilidade. Para o contexto do CHAVE o pool de replicação é utilizado para rastrear a localização das réplicas, mas na prática é utilizado para orquestrar as MVs réplicas visando otimizar os recursos de rede. Considera-se que uma MV crítica ou regular é sempre alocada em uma AZ específica definida pelo *tenant*, conforme estabelecido em SLA.

Figura 6 – Fluxograma para uma requisição do tipo crítica com consolidação e replicação de MVs.



Fonte: Elaborado pelo autor.

As requisições do tipo crítica, regular ou réplicas são alocadas, utilizando o mesmo algoritmo de posicionamento inicial, cujo objetivo é encontrar o melhor servidor e manter o ambiente consolidado, evitando ativar um novo servidor. Porém, caso os servidores ativados não apresentem os recursos necessários para atender a demanda, então é necessário ativar um novo servidor. Após atender as tarefas de alocação ou desalocação, é verificado se há fragmentação de modo que possibilite realizar o procedimento de re-otimização dos posicionamentos das MVs nos servidores através da migração de MVs. Assim, no momento que as MVs críticas forem desalocadas (no final de seu tempo de vida), são desalocadas as réplicas no pool de replicação, instanciadas em outras AZs.

### 3.1.2 Requisitos do sistema

Para manter consistência do CHAVE em relação aos conceitos de projeto, há alguns requisitos que devem ser considerados. O escopo da aplicabilidade do CHAVE ocorre apenas através da perspectiva do CSP e é independente do modelo de implantação (pública, privada, comunitária ou híbrida) Porém, o CSP deve ter o modelo de serviços Infraestrutura como Serviço (IaaS) como base devido ao isolamento proporcionado pela virtualização, além da possibilidade de executar Plataforma como Serviço (PaaS) e Software como Serviço (SaaS) em cima de um modelo IaaS. Para a consistência do conceito de probabilidade de eventos independentes e da análise por RBD, é indispensável que a infraestrutura do CSP tenha arquitetura multi-AZ.

A especificação dos requisitos do CHAVE estão organizadas em requisitos funcionais e não-funcionais. Estes requisitos estão categorizados em três níveis de prioridades, ficando o estabelecido:

- **Essencial:** Deve ser implementado para que o sistema funcione conforme esperado;
- **Importante:** Sem este requisito o sistema pode funcionar, mas não da maneira esperada; e
- **Desejável:** Sua falta não compromete o funcionamento do sistema, mas é interessante considerá-lo.

Este documento é usado como referência em todas as fase de desenvolvimento do CHAVE, visando que o bom gerenciamento do projeto possibilite aumentar a qualidade da realização dos testes e discussão dos resultados, bem como prover manutibilidade do sistema.

Os requisitos funcionais tem o objetivo de apresentar o que um sistema deve fazer. Assim, são apresentados os dez requisitos funcionais mais relevantes para o desenvolvimento do CHAVE.

- RF-1: Considerar metas estabelecidas em SLA:** O sistema deve operar de acordo com as metas e objetivos estabelecidos em SLA. Prioridade: essencial.
- RF-2: Coletar métricas:** Durante a execução do CHAVE, devem ser coletadas e armazenadas métricas como: consumo de energia, carga, rejeições, violações de SLA e número de migrações. Prioridade: desejável.
- RF-3: Consultar as métricas::** O sistema deve permitir que as métricas coletadas pelo CHAVE sejam consultadas e visualizadas. Prioridade: desejável.
- RF-4: Consolidar intra AZs:** O sistema deve ter uma estratégia de consolidação de MVs com escopo limitado internamente a cada AZ. Prioridade: essencial.
- RF-5: Replicar entre AZs:** O sistema de ter um mecanismo de replicação de MVs que seja limitado apenas entre AZs. Prioridade: essencial.
- RF-6: Eliminar conflito:** Consolidação deve executar juntamente com a replicação, de modo que uma abordagem não interfira na outra. Prioridade: essencial.
- RF-7: Organizar as réplicas:** O conjunto de réplicas de uma MV crítica devem ser organizadas em um pool de replicação. Prioridade: importante.
- RF-8: Quantificar réplicas sob demanda:** O sistema deve especificar o número ideal de réplicas ideal para entregar a taxa de HA desejada pelo *tenant*. Prioridade: Essencial.
- RF-9: Selecionar AZ mais adequada (replicação):** Para cada réplica, o sistema deve selecionar a AZ mais adequada para instanciar cada réplica, considerando a taxa de disponibilidade e considerar a sua taxa de ocupação. Prioridade: importante.
- RF-10: Selecionar o servidor mais adequado (posicionamento inicial):** Para MVs regulares, críticas ou réplicas, o sistema deve encontrar o servidor mais adequado para o posicionamento inicial, visando manter a AZ consolidada. Prioridade: essencial.



Os requisitos não-funcionais apresentam como um sistema deve desempenhar uma determinada função. Contudo, são apresentados os oito requisitos não-funcionais mais relevantes para o desenvolvimento do CHAVE.

- RNF-1: Interoperabilidade:** Para ser agnóstico à plataforma de nuvem, o sistema deve estabelecer comunicação através de APIs. Prioridade: importante.
- RNF-2: Configurabilidade:** Fornecer meios de parametrizar o sistema, permitindo adequá-lo às características da simulação. Prioridade: importante.
- RNF-3: Escalabilidade:** O sistema deve se adaptar a CSPs de qualquer dimensão, utilizando-se de configurabilidade e adaptabilidade à quantidade de requisições, de AZs e de regiões. Prioridade: importante.
- RNF-4: Desempenho/Eficiência:** Devido as características *on line*, o sistema deve ter a tomada de decisões no menor tempo possível e com resultados o mais próximo do ótimo. Prioridade: importante.
- RNF-5: Confiabilidade:** Por fornecer HA, o sistema deve estar isento de erros, possibilitando estar totalmente disponível para gerenciar a plataforma de nuvem. Prioridade: essencial.
- RNF-6: Requisito de entrega:** Durante sua execução, o sistema deve prover relatórios parciais de operação. Prioridade: desejável.
- RNF-7: Requisitos de padrões de implementação:** O sistema deve ser implementado de modo a permitir o reuso de código, bem como conceitos de herança e modularização de código. Prioridade: desejável.
- RNF-8: Éticos/legais:** Permanecem os conceitos correlatos à responsabilidade compartilhada, porém a solução tem base na perspectiva do CSP. Prioridade: desejável.

CHAVE é implementado com base nestes requisitos, que auxiliam delimitar o escopo do projeto e permitindo verificar sua consistência no estudo de caso. Esses requisitos também auxiliam no processo de tomada de decisão, para eventuais modificações do sistema.

### 3.2 FORMALISMO MATEMÁTICO

Para a representação formal do CHAVE, são definidas os conjuntos e as equações utilizadas nas seções subsequentes, como a experimentação e análise dos resultados. A Tabela 5 é organizada entre as definições relativas aos conjuntos de recursos e as equações. Para o conjunto de recursos, o formalismo é definido no contexto de teoria de conjuntos, amplamente utilizada para exemplificar o uso das equações. As equações representam as funcionalidades do CHAVE, que em geral, auxiliam na tomada de decisão sobre HA e EE. Salienta-se que as Equações 3.1, 3.2, 3.3, 3.4, 3.5 e 3.6 são contribuições do presente trabalho, e foram desenvolvidas para o contexto do CHAVE, enquanto a Equação 3.7 é uma relação genérica de eficiência.

Tabela 5 – Tabela de notações, definições formais e equações utilizadas.

<b>Conjuntos de recursos</b>	
$\mathcal{V}'_n = \mathcal{V}^s_i \cup \mathcal{V}^c_i \cup \mathcal{V}^r_i = \{v'_1, v'_2, \dots, v'_n\}$	Conjunto de todas as $n$ MVs composta por regulares ( $\mathcal{V}^s_i$ ), críticas ( $\mathcal{V}^c_i$ ) réplicas ( $\mathcal{V}^r_i$ ).
$\mathcal{P}_n = v^c_i \cup \mathcal{V}^r_i \mid i \in P$	<i>Pool</i> de replicação composta por uma $v^c_i$ e $\mathcal{V}^r_i$ .
$\mathcal{H}_m = \{h_1, h_2, h_3, \dots, h_m\}$	Conjunto de $m$ servidores de uma AZ.
$\mathbb{H}\mathbb{A}$	Taxa de alta disponibilidade.
$\mathbb{A}$	Taxa de disponibilidade da AZ.
<b>Equações</b>	
$\mathcal{R} = \text{ceil}\left(\frac{\log_{10}(1-\mathbb{H}\mathbb{A})}{\log_{10}(1-\mathbb{A})} - 1\right)$ (3.1)	Quantidade de réplicas para fornecer a taxa de HA.
$\mathcal{U} = (m * \sum_{i=0}^m \mathcal{H}_i(\#CPU))^{-1}$ (3.2)	Carga unitária, 1 unidade de CPU em relação à AZ.
$\mathcal{F} = \sum_{i=0}^m \frac{\sum_{j=0}^n \mathcal{H}_i(CPU\_idle_j)}{m * \mathcal{H}_i(\#CPU)}$ (3.3)	Fragmentação de recursos de CPU da AZ.
$\mathcal{F}_{min} = m^{-1}$ (3.4)	Fragmentação mínima, equivale a 1 servidor da AZ.
$\mathcal{C} = \frac{\sum_{j=0}^n vCPU_j}{\sum_{i=0}^m \mathcal{H}_i(\#CPU)}$ (3.5)	Carga da AZ, sendo $m$ a quantidade de servidores e $n$ o total de MVs instanciadas na AZ.
$\mathcal{E} = \sum_{i=0}^m \sum_{j=0}^n \text{Consumo}(MV_j \text{ no } \mathcal{H}_i)$ (3.6)	Consumo total de uma AZ, $m$ é a quantidade de servidores e $n$ as MVs de cada servidor.
$\mathcal{E}_r = 1 - \frac{\mathcal{E}_{final}}{\mathcal{E}_{inicial}}$ (3.7)	Redução do consumo em relação à medição anterior.

Fonte: Elaborado pelo autor.

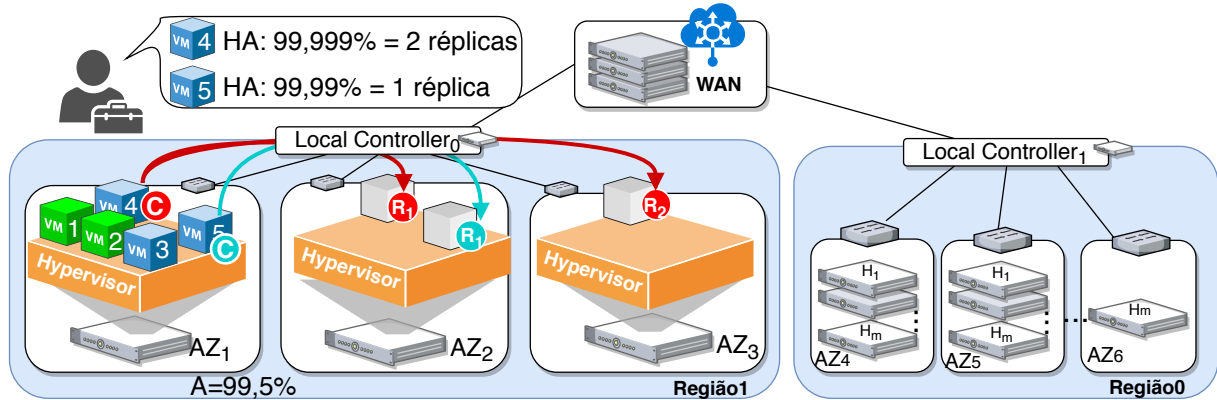
### 3.2.1 Alta disponibilidade

Para fornecer HA através de um mecanismo de tolerância à falhas, CHAVE baseia-se em replicação de MVs, instanciando cada elemento do *pool* de replicação em AZs distintas, obedecendo a restrição de anti-afinidade ( $MV_{critica} - MV_{replica}$ ) entre os elementos do *pool*. Esta abordagem é embasada na análise por RBD e fundamentada pela probabilidade de eventos independentes (Equação 2.2). Normalmente, a arquitetura da infraestrutura de CSPs é constituída por regiões geograficamente distribuídas, e cada região é composta por duas ou mais AZs independentes. Quando próximas, essas AZs são interconectadas por enlaces de rede de baixa latência, permitindo que a replicação de MVs entre as AZs ocorra sem causar impactos na qualidade da rede.

Através de SLA, a demanda gerada pelo *tenant* deve ter especificada a região e a AZ que serão instanciadas cada uma de suas MVs. Adicionalmente, serviços críticos demandam uma taxa de disponibilidade maior do que a taxa padrão estabelecida em SLA, sendo que cada MV crítica pode ter uma taxa de HA específica. Esta taxa de HA também pode ser representada pelo tempo de indisponibilidade, ou seja, a quantidade de tempo que uma MV pode ficar indisponível durante um determinado período, habitualmente estabelecida entre minutos ou segundos ao ano. Nesse sentido, uma

das características do CHAVE é concepção de HA sob demanda, que objetiva atender as necessidades específicas de cada *tenant*, estabelecendo o numero ideal de réplicas através da Equação 3.1. Todas as réplicas  $\mathcal{R}$  são instanciadas em diferentes AZs de uma mesma região. Desse modo, a Equação 3.1 visa assegurar a taxa  $\mathbb{HA}$  requerida. Esta equação é baseada na reformulação da Equação 2.2, em que é isolado o índice  $\mathcal{P}$  do produtório, para obter a quantidade de réplicas  $\mathcal{R}$ . É fornecida a menor taxa de disponibilidade  $\mathbb{A}$  entre todas as AZs e a taxa  $\mathbb{HA}$  requisitada pelo *tenant*. Como o valor de  $\mathcal{P}$  e  $\mathcal{R}$  devem ser discretos, utiliza-se a função *ceil*, que corresponde ao menor valor inteiro maior ou igual a  $\mathcal{R}$ . O número de réplicas  $\mathcal{R}$  é limitado pela quantidade máxima de AZs (índice  $\mathcal{P}$ ) de cada região, que também é um limitante para o tamanho do *pool* de replicação. Da relação entre  $\mathcal{P}$  e  $\mathcal{R}$ , decorre que  $\mathcal{P}$  é o número de elementos do *pool* de replicação, que inclui o próprio componente crítico, enquanto  $\mathcal{R}$  é o número de réplicas, *i.e.*,  $\mathcal{R} = \mathcal{P} - 1$ . Assim, ao aplicar o exemplo prático da Figura 7, é definido que a MV crítica  $v_4^c$  deve ter  $\mathbb{HA} = 99,999\%$ , em um CSP que estabelece por padrão, uma disponibilidade  $\mathbb{A} = 99,5\%$  para todas as suas AZs.

Figura 7 – Visão geral da replicação em arquiteturas Multi-AZ: *tenant* requisita  $\mathbb{HA} = 99,999\%$  e CHAVE replica o serviço para mais duas réplicas.



Fonte: Elaborado pelo autor.

Assim, aplicando a Equação 3.1, tem-se que  $\mathcal{R} = \frac{\log_{10}(1-0,99999)}{\log_{10}(1-0,995)} - 1 = 1,17$ , e quando aplicado  $\text{ceil}(\mathcal{R}) = 2$  resulta em duas réplicas. A restrição de anti-afinidade consiste em considerar um *pool* de replicação  $\mathcal{P}_i$  com a MV crítica  $v^c$  e suas réplicas  $v_1^r$  e  $v_2^r$ , de modo que quando  $v^c$  for instanciada na  $AZ_i$ , então suas respectivas réplicas deverão ser instanciadas em  $AZ_{j'}$  e  $AZ_{j''}$  |  $j \neq j' \neq j''$  e  $j \leq P$ . Assim, ao realizar a estratégia de consolidação de MVs para cada uma das  $\mathcal{P}$  AZs, esta abordagem de distribuição de réplicas restringe alocar quaisquer elementos de  $\mathcal{P}$  no mesmo ponto único de falhas (SPoF).

### 3.2.2 Consolidação de MVs

A carga de um ambiente de nuvem varia com o tempo, recebendo uma sequência de requisições para alocar ou desalocar as MVs. A estratégia de consolidação de MVs do CHAVE, ativa e desativa os servidores sob demanda, evitando que servidores ociosos consumam energia. Considera-se que CHAVE atua de modo eficiente para a redução do consumo de energia, pois não aplica *overcommitting* de recursos e, portanto, não afeta o desempenho do sistema. A consolidação realizada pelo CHAVE baseia-se em duas etapas distintas: o posicionamento inicial e as migrações de MVs. O posicionamento inicial consiste em selecionar os servidores mais adequados para instanciar as novas requisições, *i.e.*, no momento de alocar novas MVs são selecionados os servidores ativos, caso não haja recursos para tal alocação, então é ativado um novo servidor, conforme descrito no Algoritmo 1. Porém, para esse caso, a entrada relativa ao conjunto de MVs possui tamanho 1 e usa a função de alocação na Linha 07. O custo do procedimento da etapa de posicionamento inicial é o mesmo de qualquer outro algoritmo de escalonamento, pois basta selecionar o primeiro servidor (*First-Fit*) de uma lista ordenada. A etapa das migrações ocorre periodicamente em um processo denominado de re-otimização. A re-otimização é acionada por um gatilho de fragmentação da AZ, descrito pela Equação 3.3, e esse gatilho é quem executa o Algoritmo 1. Nesse caso, a entrada relativa ao conjunto de MVs possui todas as MVs da AZ em questão, usando a função de migração na Linha 07. Realizar migrações consiste em mover uma MV em execução de um servidor para outro, causando uma carga adicional na rede e na camada de controle da nuvem. Estas duas etapas são distintas e independentes, podendo as migrações ocorrerem antes ou depois de um posicionamento inicial. Porém, para a execução dos testes (Capítulo 4), observou-se que a fragmentação de recursos ocorre sempre após atender as requisições de alocação e desalocação de MVs.

```

Algoritmo 1: FFD()
Entradas: [<Hosts>:H, <MVs>:V]
Saída: Conjunto de hosts consolidados
01: ordena_decrescente(V)
02: ordena_decrescente(H)
03: Para cada elemento i de V até n faça
04:   Para cada elemento j de H até m, faça
05:     Se i.CPU couber no host j, então
06:       Se i.RAM couber no host j, então
07:         Aloque ou Migre a MV i para host j
08:         Atualiza RAM e CPU disp. no host j
09:       Quebra o loop e analisa próxima MV (linha 03)
10:     Fim Se
11:   Fim Se

```

```

12:  Fim Para
13:  Se MV  $i$  não couber em nenhum host ativo, então
14:    Ative um novo host ( $j+1$ ) e aloque  $i$ 
15:  Fim Se
16: Fim Para
17: Para cada elemento  $j$  de  $H$ 
18:  Se  $j$  estiver ocioso, então DesativeHost( $j$ )
19: Retorne <Hosts>  $H$ 

```

A função de fragmentação de recursos da AZ (Equação 3.3) verifica a quantidade de CPUs ociosas ( $CPU\_idle$ ) entre os servidores ativos, de tal modo que o somatório dessa fragmentação seja equivalente aos recursos de um servidor ( $\mathcal{F}_{min}$ ), que será desativado após a migração. Assim, se houver fragmentação igual ou superior ao valor mínimo ( $\mathcal{F}_{min}$  na Equação 3.4), então há a possibilidade de realizar a consolidação. A fragmentação mínima corresponde ao percentual dos recursos de CPU de um servidor, em relação ao total de recursos de CPU da AZ. No CHAVE, a fragmentação é verificada após atender as requisições, pois sempre que uma MV é alocada ou desalocada a configuração da AZ se modifica. A função  $\mathcal{F}$  é definida pela razão entre o somatório dos recursos das  $n$  CPUs ociosas de cada um dos  $m$  servidores da AZ, e produto do número de servidores ativos pelos  $n$  núcleos de cada servidor.

A heurística *First Fit Decreasing* (FFD) é selecionada para uso no algoritmo de consolidação do CHAVE devido a sua relação entre os critérios de otimalidade e tempo de resposta. Quando o FFD é aplicado no posicionamento inicial, ele comporta-se como um algoritmo *online*, tratando cada nova requisição somente com base nas informações da instância (número de vCPUs) e, a ordenação nos servidores, de acordo com os recursos ociosos. Por outro lado, o FFD na etapa de migração, mantém suas características *offline*, *i.e.*, organiza todas as MVs em ordem decrescente, com base na quantidade de vCPUs de cada MV. Assim, CHAVE apresenta uma abordagem híbrida: *online* para atender instantaneamente as novas requisições, e *offline* para encontrar os melhores resultados de consolidação.

O caso mínimo em que a consolidação na etapa de migração pode ser acionada, é quando há dois servidores em execução, cada servidor com uma MV de uma vCPU, *i.e.*, duas vezes a carga unitária ( $2 * \mathcal{U}$ , conforme Eq. 3.2). Adicionalmente, há os casos de falso-positivo quando acionado o gatilho de fragmentação, causado por dois motivos: (i) o tamanho de todas as MVs é maior do que a fragmentação individual de cada servidor; e (ii) o desempenho do FFD não é o ideal para aquela configuração. Já na etapa do posicionamento inicial, o melhor caso ocorre quando a nova requisição consegue ser alocada em um servidor ativo, enquanto o pior caso é a rejeição, *i.e.*, quando não há novos servidores para serem ativados, pois a AZ está com carga a

100%. A rejeição é especificada pelo CHAVE como uma quebra de SLA, pois representa a recusa em instanciar as MVs regulares, críticas e réplicas.

Além de executar a estratégia de consolidação máxima, CHAVE possui uma segunda estratégia que considera as restrições de afinidade MV-MV, estabelecida pelo *tenant*. Uma contribuição do presente trabalho, é o algoritmo *Affinity-Aware* (AA) que consiste em manter as MVs configuradas com a restrição para o servidor em que é inicialmente alocado na etapa de posicionamento inicial, evitando que as MVs restritas sejam migradas. Em sequência, as MVs marcadas com esta afinidade também bloqueiam de serem desativados os servidores em que estão instanciadas. Adicionalmente, o algoritmo AA tem a funcionalidade de realizar consolidação com o mínimo de migrações, pois são selecionadas as MVs dos servidores com a menor quantidade de MVs, para serem migrados para os servidores que possuem a restrição ou com maior quantidade de MVs. Este procedimento interfere na capacidade da consolidação em desativar servidores, porém considera uma restrição que ocorre na prática em diversos CSPs. Deste modo, é possível realizar a consolidação, apresentando resultados similares.

### 3.3 PROTÓTIPO E AMBIENTE DE TESTES

Como o CHAVE é concebido sob a perspectiva do CSP, um requisito fundamental é que sua execução deve ser em uma infraestrutura de computação em nuvem com arquitetura multi-AZ. Porém, devido a indisponibilidade de acesso a um ambiente real com esse requisito, o protótipo do CHAVE é desenvolvido para ser executado em um ambiente simulado. Objetivando manter a consistência dos requisitos levantados durante o desenvolvimento do CHAVE, estabelece-se que este ambiente simulado deve cumprir com três requisitos básicos relativos ao CSP:

- **Consolidação de MVs:** Fornecer uma API com nível de acesso ao controle do escalonamento e orquestração das MVs, bem como alterar o estado *Advanced Configuration and Power Interface* (ACPI) dos servidores;
- **Replicação de MVs:** Arquitetura distribuída e organizada em multi-AZ, com um mecanismo de replicação entre AZs; e
- **Mensurar métricas:** Fornecer meios de monitoramento de recursos computacionais e de consumo de energia, compatíveis com as necessidades do CHAVE.

O protótipo deve abranger todas funcionalidades levantadas na análise de requisitos, embora algumas funcionalidades específicas careçam de maior completude para aproximar-se de um ambiente real. Nesse sentido, é utilizado o simula-

dor EAVIRA<sup>1</sup>. EAVIRA é aplicado em pesquisas relacionadas ao escalonamento de Infraestruturas Virtuais (IVs) no contexto de computação em nuvem IaaS (RUCK et al., 2017; RÜCK et al., 2017). Para aplicação do CHAVE, o EAVIRA foi estendido para uma versão que possibilite também realizar o gerenciamento de MVs, de modo a suportar uma arquitetura multi-AZ, que permita a replicação de MVs entre as AZ. Funcionalidades como alocação/desalocação, migração e consolidação de recursos já eram implementações do EAVIRA. Por fim, mantém-se o modelo preexistente de consumo de energia dos servidores, que é guiado pelo modelo de compartilhamento proporcional (HINZ et al., 2018). Porém, são adicionados novos valores escala de consumo de servidores, para suportar servidores com até 32 núcleos, estendendo o limite superior para 209 kWh. O algoritmo do simulador é implementado em linguagem Python (portado para versão 3.6). Todos os testes foram executados em uma MV do OpenStack configurada com 16 vCPUs e 32 GB de memória RAM, sistema operacional Debian Stretch, e hospedada por servidores com processador Intel Xeon E5-2620.

### 3.3.1 Requisitos vs Implementação

Com o objetivo de explicar o modo como os requisitos descritos na Seção 3.1.2 são satisfeitos, é realizada uma relação entre o especificado e o implementado. Ressalta-se que todos os requisitos funcionais e não-funcionais foram devidamente implementados. Nesse sentido, as Tabelas 6 e 7 relacionam os requisitos levantados com o que foi efetivamente implementado.

Salienta-se que a descrição de alguns requisitos são referentes ao desenvolvimento da solução proposta, orientado ao simulador. Assim, mais detalhes são especificados na definição do cenário, com as características relevantes a aplicação da solução.

### 3.3.2 Cenário

Para que o EAVIRA execute o CHAVE com maior proximidade de um ambiente real, a entrada de dados com requisições em séries temporais discretas são constituídas por *traces* reais de cargas de trabalho de um CSP IaaS. Em pesquisa não-exaustiva na literatura especializada, destacam-se os *traces* da plataforma Eucalyptus<sup>2</sup>, por serem avaliados em outros trabalhos científicos para análise de distribuição de dados (WOLSKI; BREVIK, 2014; YAO; PAPAPANAGIOTOU, 2017) e em estratégias de escalonamento e consolidação (PUCHER et al., 2015). Estes *traces* consistem em seis conjuntos de dados que são AZs distintas de organizações privadas. Cada requisição refere-se a uma alocação ou desalocação (START/STOP),

<sup>1</sup> Disponível em: <[https://github.com/DanielFloripa/EAVIRA\\_CHAVE](https://github.com/DanielFloripa/EAVIRA_CHAVE)>.

<sup>2</sup> Disponíveis em <<https://www.cs.ucsb.edu/~rich/workload/>>

Tabela 6 – Requisitos funcionais

Requisito	Implementação
<b>RF-1</b> Metas do SLA	As metas estabelecidas em SLA compõem a entrada de dados.
<b>RF-2</b> Coletar métricas	Dados das métricas são coletadas em tempo de execução e persistidas em uma base de dados.
<b>RF-3</b> Consultar métricas	Dados persistidos são consultados para análise e geração de gráficos.
<b>RF-4</b> Consolidar intra AZs	Consolidação é realizada com os dados isolados de cada AZ.
<b>RF-5</b> Replicar entre AZs	Replicação é feita em dados compartilhados entre AZs, referente as réplicas.
<b>RF-6</b> Eliminar conflito	O isolamento dos objetos que compõe a infraestrutura são orquestrados separadamente.
<b>RF-7</b> Organizar as replicas	As réplicas são organizadas em um <i>pool</i> de replicação, instanciadas a cada unidade de tempo.
<b>RF-8</b> Demanda de HA	Utilizando a Equação 3.1 quantifica-se as réplicas necessárias para entregar a taxa de HA.
<b>RF-9</b> AZ adequada	Para instanciar as réplicas, seleciona-se as AZs conforme critério de balanceamento de carga.
<b>RF-10</b> Servidor adequado	Servidor mais adequado é selecionado conforme algoritmo FFD.

Fonte: Elaborado pelo autor.

tempo virtual, identificação da MV, identificação do servidor em que a MV foi alocada e quantidade de núcleos reservados. Como não há informações sobre a taxa de utilização de vCPU, considera-se para os testes que todas as MVs estão a 100% de uso de seus recursos de vCPU, e que estes recursos estão reservados, pois não é considerado *overcommitting*. Também não é especificada a quantidade de vRAM para cada MV, então considera-se que a vRAM é proporcional à quantidade de vCPUs de 1 : 4 (4GB de RAM para cada vCPU). Cada AZ é constituída por servidores homogêneos quanto ao número de CPUs. Em suma, as principais características de cada *trace* constam na Tabela 8, indicando a quantidade de operações, número de servidores e quantos núcleos possui cada servidor.

As colunas “ $\max(P(x))$ ” e “ $\mathcal{C}$ ” resumem as probabilidades e respectivas cargas definidas no plano de testes (Seção 3.3.3), usadas como base para selecionar os *snapshots* analisados na Seção 4.1. A coluna “ $2 * \mathcal{U}$ ” refere-se a carga mínima para que possa ocorrer uma consolidação (Eq. 3.2), utilizada como critério de corretude ao selecionar cada *snapshot*.

Os *traces* não possuem informações sobre a taxa de disponibilidade  $\mathbb{A}$  das AZs, nem a demanda pela taxa  $\mathbb{HA}$  para cada instância, portanto esses dados são



Tabela 7 – Requisitos Não-funcionais

Requisito	Implementação
<b>RNF-1</b> Interoperabilidade	As APIs são abstraídas através de conceitos de hierarquia classes (orientação a objetos).
<b>RNF-2</b> Configurabilidade	As configurações são descritas em um arquivo de entrada, parametrizando o simulador.
<b>RNF-3</b> Escalabilidade	Definida como um dos parâmetros da configuração, e.g., nº de servidores, AZs, regiões e controladores.
<b>RNF-4</b> Desempenho	Os algoritmos baseados em heurísticas para fornecer os resultados mais adequados no menor tempo.
<b>RNF-5</b> Confiabilidade	São utilizados conceitos de integração e entrega contínua, e desenvolvimento guiado por testes (TDD).
<b>RNF-6</b> Entrega	A solução fornece as métricas em tempo de execução, bem como o tempo decorrido/estimado.
<b>RNF-7</b> Implementação	Utilizada orientação a objetos, com classes e herança de classes.
<b>RNF-8</b> Éticos/legais	A definição da responsabilidade compartilhada é um guia para o desenvolvimento da solução.

Fonte: Elaborado pelo autor.

Tabela 8 – Informações e configuração dos *traces* por AZ.

	#Operações	# $\mathcal{H}$	#CPU	$\max(P(x))$	$\mathcal{C}$	$2 * \mathcal{U}$	
$R_0$	AZ1	18346	13	24	15,1%	25,321%	0,641%
	AZ2	1800	7	12	12%	28,571%	2,381%
	AZ3	5200	7	8	7,8%	3,571%	3,571%
$R_1$	AZ4	2872	12	8	18%	8,33%	2,083%
	AZ5	16912	31	32	4,7%	71,472%	0,202%
	AZ6	8314	31	32	7,8%	64,648%	0,195%

Fonte: Elaborado pelo autor.

sintetizados com base na distribuição uniforme gerada pelo método de Monte Carlo. Monte Carlo é usado para limitar em 20% as requisições dos *traces* que solicitam HA, sendo 15% uma demanda por HIA equivalente a uma réplica, e 5% equivalente a duas réplicas. Adicionalmente, para os testes realizados com a restrição de afinidade, aplicando o algoritmo AA, é também usado o método de Monte Carlo para estabelecer uma restrição de afinidade para 20% das MVs.

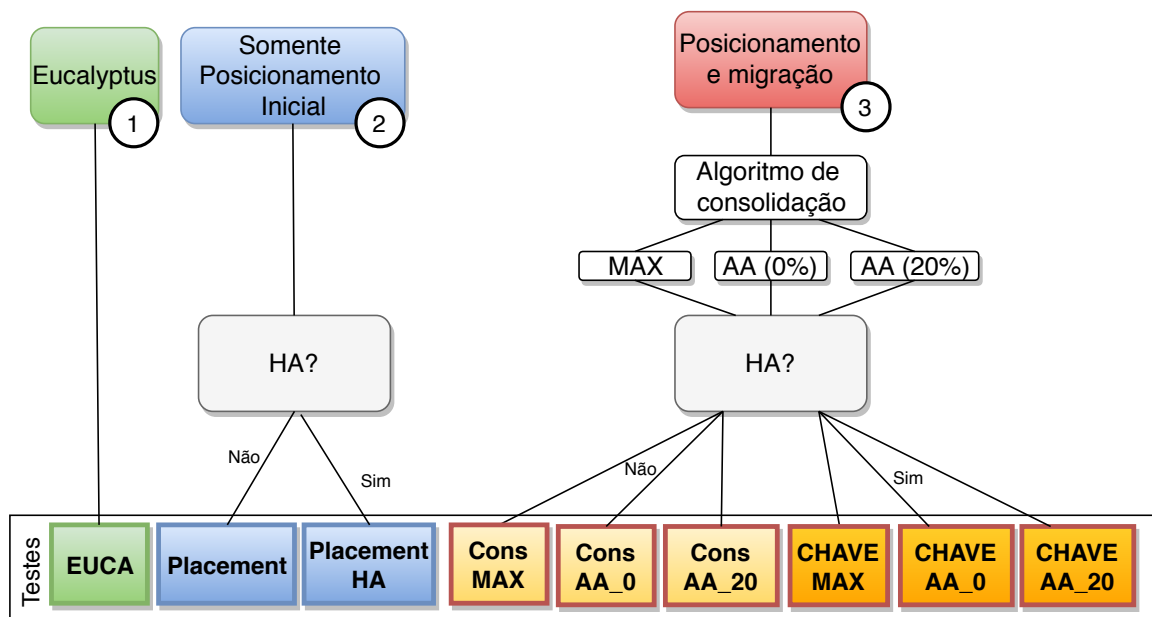
A arquitetura da plataforma de nuvem, em relação a distribuição das AZs por distintas regiões, segue a mesma estrutura definida na Figura 7. Os *traces* estão organizados em duas regiões, de modo que AZ1, AZ2 e AZ3 pertencem à região  $R_0$  e AZ4, AZ5 e AZ6 pertencem à região  $R_1$ . Desse modo, as requisições para replicação de qualquer AZ de  $R_0$ , serão instanciadas apenas entre suas AZs. Os algoritmos

do *Local Controller* de sua respectiva região, selecionam a melhor AZ para receber a réplica através de balanceamento de carga, *i.e.*, seleciona a que possui a menor carga, conforme Equação 3.5. Para sincronizar a execução simultânea de todas as AZs, é utilizado o conceito de *Global Virtual Time* (GVT) síncrono. Neste sentido, todas as requisições são realizadas respeitando um único tempo global, em uma faixa entre as unidades de tempo mínimo (zero) e máximo (mais de 24 milhões) de todas as AZs. Para contabilizar corretamente a métrica de consumo de energia em kWh, considera-se que a unidade de tempo seja data em horas. A medida em que as AZs chegam em seu tempo final de execução, são removidas da execução do GVT para não contabilizar em alterações para os cálculos de consumo de energia.

### 3.3.3 Plano de testes

A definição do plano de testes objetiva a consistência dos resultados, auxiliando na didática da análise dos dados. Com base no protótipo do CHAVE e na definição do cenário, são definidos nove testes, organizados em três conjuntos, organizados em árvore conforme Figura 8.

Figura 8 – Árvore de testes



Fonte: Elaborado pelo autor.

O primeiro teste consiste usar o algoritmo padrão de escalonamento do Eucalyptus, permitindo avaliar a distribuição de carga e observar o comportamento de cada AZ. No segundo conjunto, são aplicados dois testes que executam apenas a etapa do posicionamento inicial, *i.e.*, sem a migração, diferenciando-se entre ter ou não as replicações habilitadas. No terceiro conjunto de testes é adicionada a funcionalidade de migração, que constitui o processo completo de consolidação, com os três algoritmos de consolidação: MAX, AA com 0% de afinidade e o AA com 20% de

afinidade. O parâmetro HA é binário, que pode ativar ou desativar o mecanismo de replicação das MVs críticas. Quando o mecanismo de HA está ativado, então os testes são denominados CHAVE, em conformidade com a proposta de replicar e consolidar simultaneamente. É estabelecido uma distribuição de 20% para o algoritmo de consolidação AA (AA20) com base em uma demanda real (KOROTKOV, 2018), e o teste com 0% de restrições (AA0) é utilizado para minimizar a quantidade de migrações.

O teste EUCA é inicialmente realizado utilizando seu algoritmo padrão de escalonamento (colocando as MVs nos servidores descritos nos traces). Para justificar a seleção das amostras usadas nas Seção 4.1, é realizada a análise da probabilidade de ocorrência destas amostras ( $P(x)$ ), com base na carga da AZ, utilizando-se do método de inferência estatística. Para a análise de carga, são definidos os histogramas referentes a distribuição de carga em cada AZ (Figuras de 9 a 14), de modo a identificar quais as cargas que possuem maior probabilidade de ocorrerem durante sua execução. Em cada AZ, é selecionada a carga  $\mathcal{C}$  com a maior probabilidade de ocorrência de ocorrer ( $\max(P(x))$ ), de modo que esta seja maior ou igual a carga mínima para migração  $2 * \mathcal{U}$ . Nesse sentido, considerar as cargas de maior probabilidade de ocorrência, permite justificar que a seleção das amostras dos demais testes também são comuns de ocorrerem.

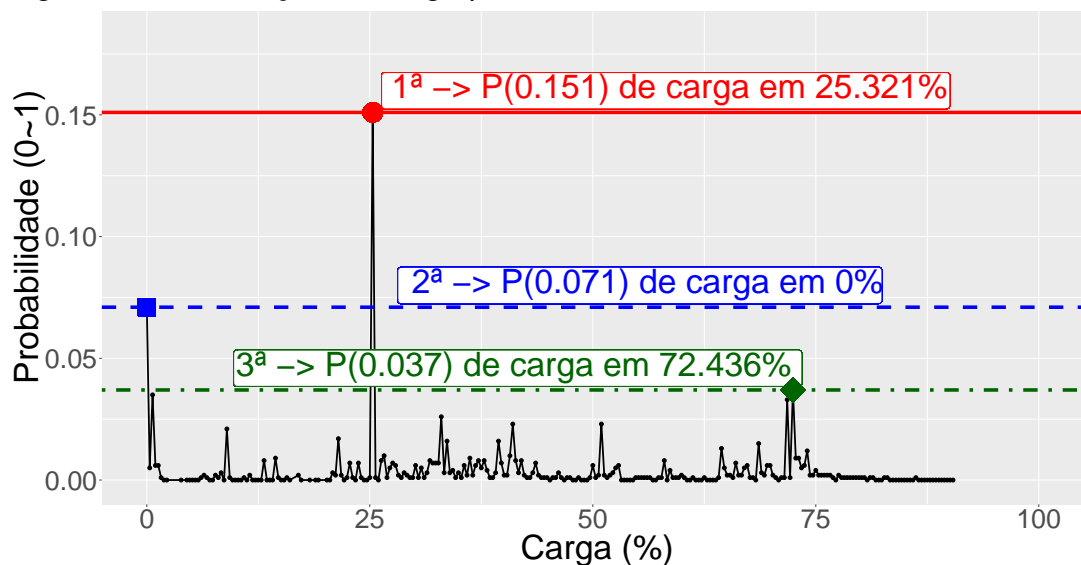
As Figuras de 9 a 14 mostram a análise de probabilidade de frequência das cargas para cada AZ. Para ser acionada a consolidação, é necessário considerar um caso mínimo em que a etapa de migração pode ser acionada. Desse modo, a menor carga possível para cada AZ é estabelecida na Tabela 8, na coluna ' $2 * \mathcal{U}$ ', que indica a carga equivalente a duas vezes o valor mínimo. Assim, para os casos em que a maior probabilidade indica uma carga abaixo do mínimo especificado, são avaliadas as próximas opções de maior probabilidade. Em cada Figura, as linhas horizontais (em vermelho, azul e rosa) indicam as três maiores probabilidades (eixo  $y$ , limitado a 0.2) de ocorrerem cada carga (eixo  $x$ , de 0% até 100%).

A Figura 9 representa a distribuição de carga para AZ1, cujo valor em 25,321% é selecionado devido a probabilidade de 15,1% de sua ocorrência. Neste caso, há uma nítida diferença entre a primeira e a segunda opção, estabelecidas em 7,1 e 3,7% de probabilidade de ocorrerem as cargas de 0% e de 72,436%.

A Figura 10 representa a distribuição de carga para AZ2, em que é selecionada a primeira opção devido a probabilidade de 12% de ocorrer uma carga de 28,571%. Neste caso, a segunda e a terceira opção possuem probabilidades próximas 6,6% e 6,3%, de ocorrerem as cargas de 29,762 e 17,857% respectivamente.

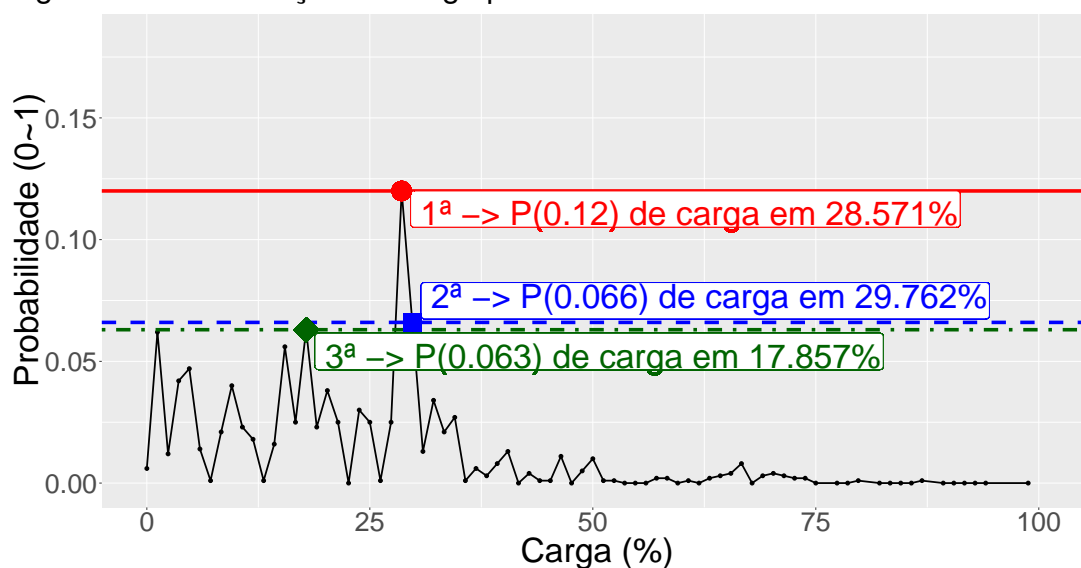
Para o caso da Figura 11, a primeira opção tem uma probabilidade de 15,8% de ocorrer uma carga de 0%, o que permite afirmar que a AZ3 fica ociosa a maior parte do tempo, *i.e.*, sem a execução de nenhuma MV. Ao verificar a carga mínima para ativar a consolidação na Tabela 8, nesta AZ constata-se que deve ser superior a 3,571%. A segunda opção com probabilidade de 11,9%, indica uma carga de 1,78%,

Figura 9 – Distribuição de carga para AZ1



Fonte: Elaborado pelo autor.

Figura 10 – Distribuição de carga para AZ2



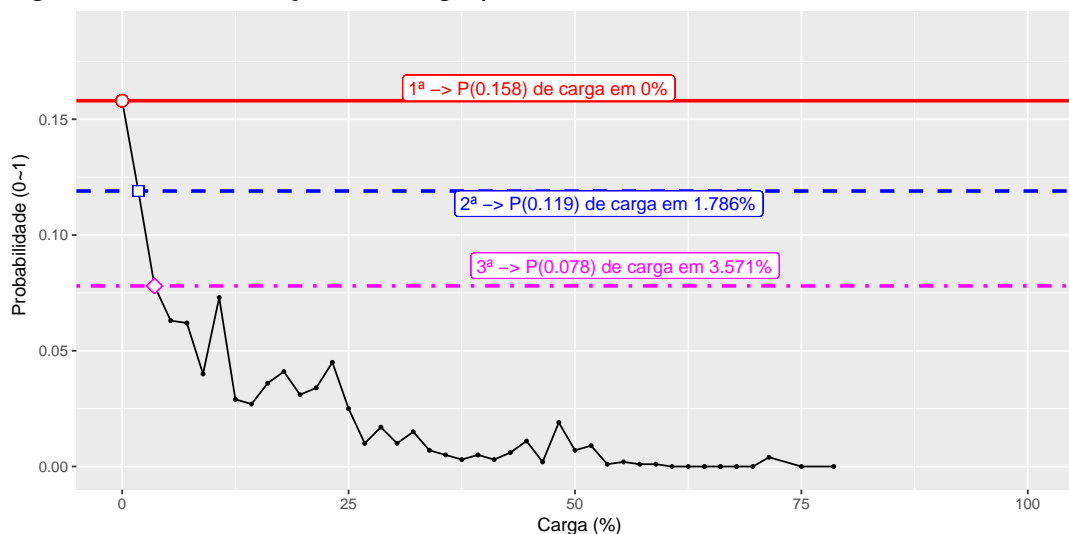
Fonte: Elaborado pelo autor.

que também é descartada devido a discordância com esse critério. Por fim, a terceira opção é selecionada com uma probabilidade de 7,8% de ocorrer uma carga de 3,571%, que equivale ao valor mínimo especificado.

Na Figura 12, é selecionada a primeira opção com a probabilidade de 18% de ocorrer uma carga de 8,33% na AZ4. A segunda opção é uma probabilidade de 14,5% para uma carga de 0%, enquanto a terceira opção indica uma probabilidade de 8,5% de ocorrer uma carga de 33,3%.

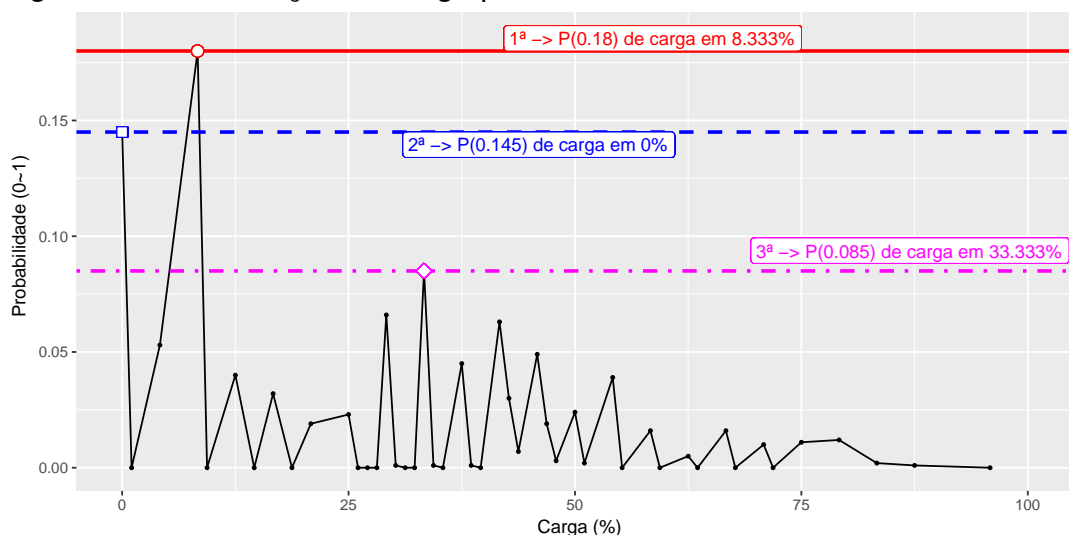
A Figura 13 da AZ5, indica um comportamento de elevada carga em relação às demais AZs, sendo selecionada a primeira opção com a probabilidade de 4,7% de ocorrer uma carga de 71,472%. A segunda e a terceira opção indicam valores próximos entre si, com uma probabilidade de 3,6% e 3,4% de ocorrerem as cargas de 67,44%

Figura 11 – Distribuição de carga para AZ3



Fonte: Elaborado pelo autor.

Figura 12 – Distribuição de carga para AZ4

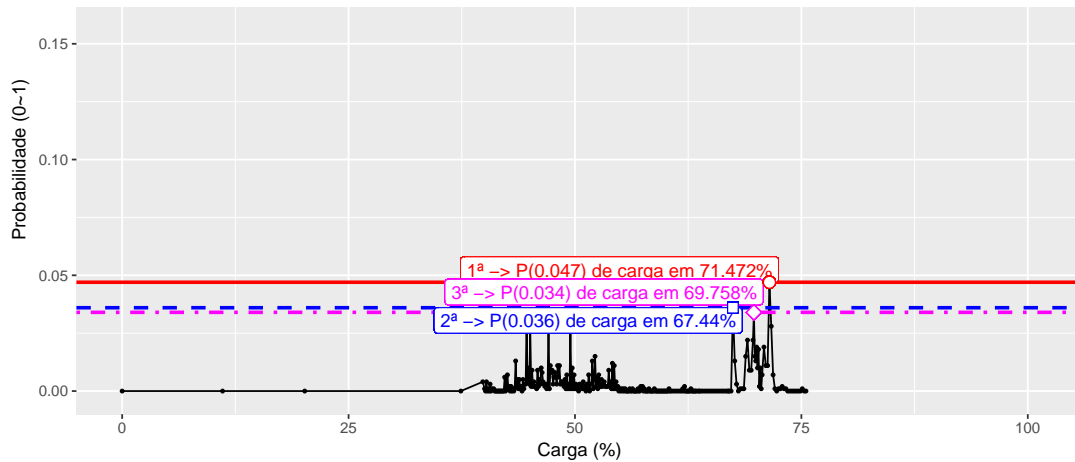


Fonte: Elaborado pelo autor.

e 69,758%, respectivamente. Assim como a AZ5, a Figura 14 da AZ6 mostra que as três opções são próximas entre si, e possuem carga mais elevada, sendo selecionada a primeira opção com carga de 64,648% indicada por uma probabilidade de 7,8%. As demais opções apresentam probabilidade de 7,7 e 5,5% de ocorrer uma carga de 64,844 e 66,992%, respectivamente.

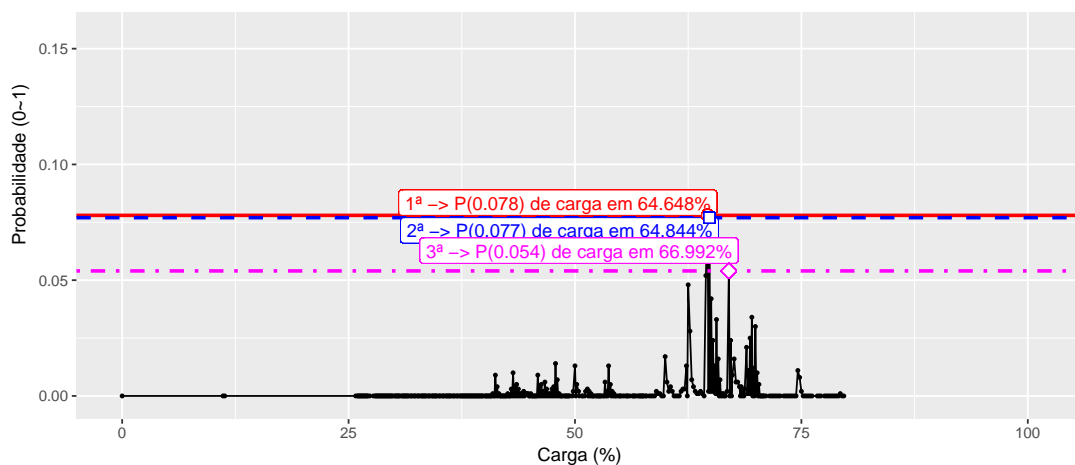
Um resumo com todas as probabilidades selecionadas ( $\max(P(x))$ ) e suas respectivas cargas ( $C$ ) estão descritas na Tabela 8. Observa-se que há uma grande variação entre as cargas selecionadas, sendo os valores extremos representados pela menor carga na AZ3, com 3,571% de carga e a maior carga pela AZ5, com 71,472% de carga. Nesse sentido, é possível estabelecer que essa variabilidade de carga indica que todas as AZs possuem características de demanda diferentes entre si. Com

Figura 13 – Distribuição de carga para AZ5



Fonte: Elaborado pelo autor.

Figura 14 – Distribuição de carga para AZ6



Elaborado pelo autor.

Fonte:

essa heterogeneidade apresentada, viabiliza-se verificar a consistência do CHAVE, e avaliar seu comportamento em diferentes cenários (de baixa ou alta demanda).

### 3.4 CONSIDERAÇÕES PARCIAIS

A presente Seção apresenta a proposta do CHAVE, baseando-se em conceitos de engenharia de software para explicar sua arquitetura de software, e exemplificar os casos de uso através de diagramas de fluxo. Adicionalmente, são estabelecidos os principais requisitos funcionais e não-funcionais para manter a consistência do CHAVE frente aos requisitos inicialmente estabelecidos. O formalismo matemático permite estabelecer as equações utilizadas no decorrer do presente trabalho, bem como as convenções e símbolos. Toda a parte teórica definida no Capítulo 2 é revista sob a perspectiva matemática, e discutindo em conjunto os conceitos de HA e

consolidação de MVs.

A especificação do protótipo e do ambiente de testes ocorre por um cenário baseado em simulação numérica, que tem *traces* reais de uma nuvem IaaS como entrada. Tanto o simulador quanto os *traces* são componentes que já foram anteriormente utilizados em outros projetos científicos. Nesse sentido, visando encontrar resultados consistentes, o plano de testes baseia-se em definir os *snapshots* que serão utilizados para realizar a análise dos resultados. Essa análise baseia-se na distribuição de carga das AZs, permitindo definir os momentos em que a carga é ideal para realizar as devidas análises. Para execução dos experimentos e análise dos resultados, estes *snapshots*, que constituem uma perspectiva local, são comparados com os resultados em perspectiva global.

## 4 EXPERIMENTOS E ANÁLISE DOS RESULTADOS

Com base no plano de testes apresentado na Seção 3.3, os experimentos e análise dos resultados são conduzidos individualmente para cada zona de disponibilidade (AZ) através de duas perspectivas: global e local. Na perspectiva global, cada teste é analisado em relação a execução completa da simulação, considerando um conjunto de métricas com características mais abrangentes. Por outro lado, a perspectiva local refere-se a análise do comportamento dos testes em um momento específico da simulação, *i.e.*, uma análise de um *snapshot* instantâneo ou fotografia da simulação em uma determinada unidade de tempo, desse modo, objetivando uma análise mais precisa. Essa unidade de tempo é selecionada conforme critério de probabilidade de distribuição de carga, estabelecido na Subseção 3.3.3. Organizar os experimentos sob essas duas perspectivas permite uma análise tanto em largura, quanto em profundidade.

Cada perspectiva possui características diferentes, sendo estabelecidas cinco métricas para a perspectiva local e seis métricas para a perspectiva global. As métricas possuem dados multivariados de magnitudes diferentes entre si. Desse modo, os resultados são representados por gráficos de radar, normalizados de 0% a 100%, com base nos valores de mínimo e máximo de cada métrica. Cada gráfico de radar é composto por eixos que representam as métricas (0% no centro e 100% na borda) e seis polígonos que representam os testes realizados.

Para a análise dos gráficos, utiliza-se um método de interpretação visual, referente a área de cada polígono. Assim, interpreta-se que o teste que apresenta o melhor desempenho possui o *‘polígono com a menor área em torno do centro do gráfico’* para as métricas com maior significância, conforme especificação. Todavia, algumas métricas tem o objetivo diferente das demais, sendo interpretadas como: *‘o melhor resultado é o que apresenta o menor valor’*. Para manter a consistência da análise do menor polígono, essas métricas específicas (marcadas com asterisco) tem seu valor invertido no gráfico, ou seja, atribui-se 100% ao centro do eixo e 0% para a borda. Assim, para cada AZ a interpretação dos resultados ocorre de forma visual (pela área do polígono) ao invés de basear-se em análise e interpretação numérica.

Os testes são agrupados por tom de cor e tipo de tracejado. Tons de cor semelhantes (entre claro e escuro) agrupam o mesmo algoritmo, organizados em: *Affinity-Aware* com 0% de afinidade (AA0) para os tons azulados, AA com 20% de afinidade (AA20) para tons avermelhados e algoritmo de consolidação máxima (MAX) em tons esverdeados. Tipos de linha semelhantes (entre tracejadas e contínuas) representam



as estratégias usadas, referente a ter somente consolidação ( $C_{-}$ ) representadas por linhas contínuas ou se possuem alta disponibilidade (HA) adicionalmente à consolidação ( $CHA_{-}$ ), representadas por linhas tracejadas. Desse modo, os polígonos nos gráficos de radar são especificados pela relação de cor e tipo de linha auxiliando no método de interpretação visual. Fica estabelecido que os testes que representam a estratégia proposta pelo CHAVE, que refere-se a conciliar a consolidação com HA, são os que têm o acrônimo  $CHA_{-}$  na legenda, *i.e.*, os polígonos com linha tracejada.

#### 4.1 PERSPECTIVA LOCAL: *SNAPSHOTS* DA CONSOLIDAÇÃO

Para selecionar o instante em que é feita a análise em perspectiva local, utiliza-se como critério a maior probabilidade de distribuição de carga de cada AZ. Para analisar esse comportamento *snapshot*, são definidas cinco métricas referentes a energia, migrações e a carga da AZ. A métrica  $RE(W)$  representa quantidade de potência que foi reduzida devido a ação de consolidação, conforme Equação 3.6, por ser aplicada como ponta de prova em um instante de tempo, sua unidade é considerada como a potência medida em Watts e a maior redução é o melhor resultado. Adicionalmente, a métrica  $RE(\%)$  representa o percentual de redução do consumo de energia (Equação 3.7), aplicada para mensurar mudanças de estado (*e.g.*, antes e depois de ocorrer uma consolidação), o maior percentual é o melhor resultado. A métrica  $T(W)$  refere-se ao consumo total de energia no instante de tempo, mensurada em Watts e o menor valor é o melhor. A métrica  $C(\%)$  é a carga de ocupação dos recursos de CPU de toda a AZ (não dos servidores), especificada conforme a Equação 3.5, o menor valor é o melhor. A métrica  $C(\%)$  possui baixo impacto na interpretação do melhor resultado, pois testes com HA habitualmente vão apresentar maior carga, contudo representa que os gatilhos de consolidação anteriores ao cenário em análise causaram essa diferença. A Tabela 9 resume a relação das métricas em relação a unidade, o objetivo (o asterisco aponta a interpretação inversa) e as equações citadas anteriormente.

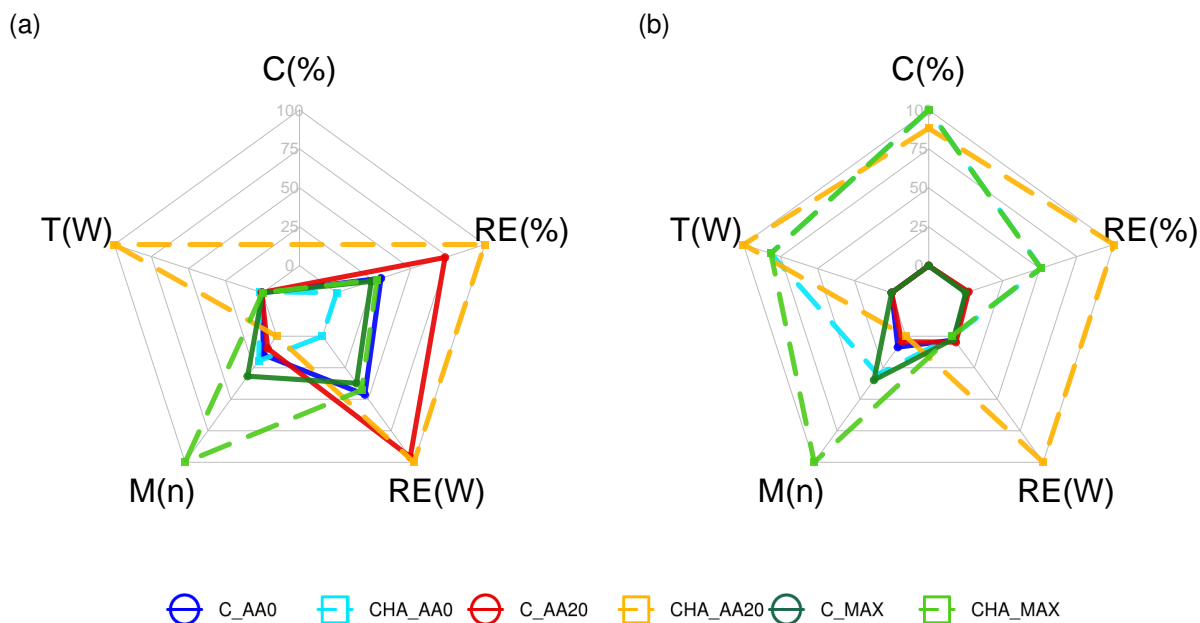
Tabela 9 – Tabela de métricas utilizadas para análise e discussão dos resultados.

Métrica	Descrição	Unidade	Objetivo (melhor)	Equações
$RE(W)$	Redução de consumo de energia	$W$	Maior*	Eq. 3.6
$RE(\%)$	Percentual de redução do consumo	%	Maior*	Eq. 3.7
$M(n)$	Número de migrações	Numérica	Menor	—
$T(W)$	Consumo total de energia	$W$	Menor	—
$C(\%)$	Carga da AZ	%	Menor	Eq. 3.5

Fonte: Elaborado pelo autor.

Para permitir a avaliação dos polígonos pela menor área, as métricas de redução de consumo em  $RE(\%)$  e  $RE(W)$  devem ser interpretadas de forma inversa.

Figura 15 – Testes em perspectiva local. (a) AZ1 melhores testes: CHA\_AA0 e C\_MAX. (b) AZ2 melhores testes: CHA\_AA0 e C\_AA20.



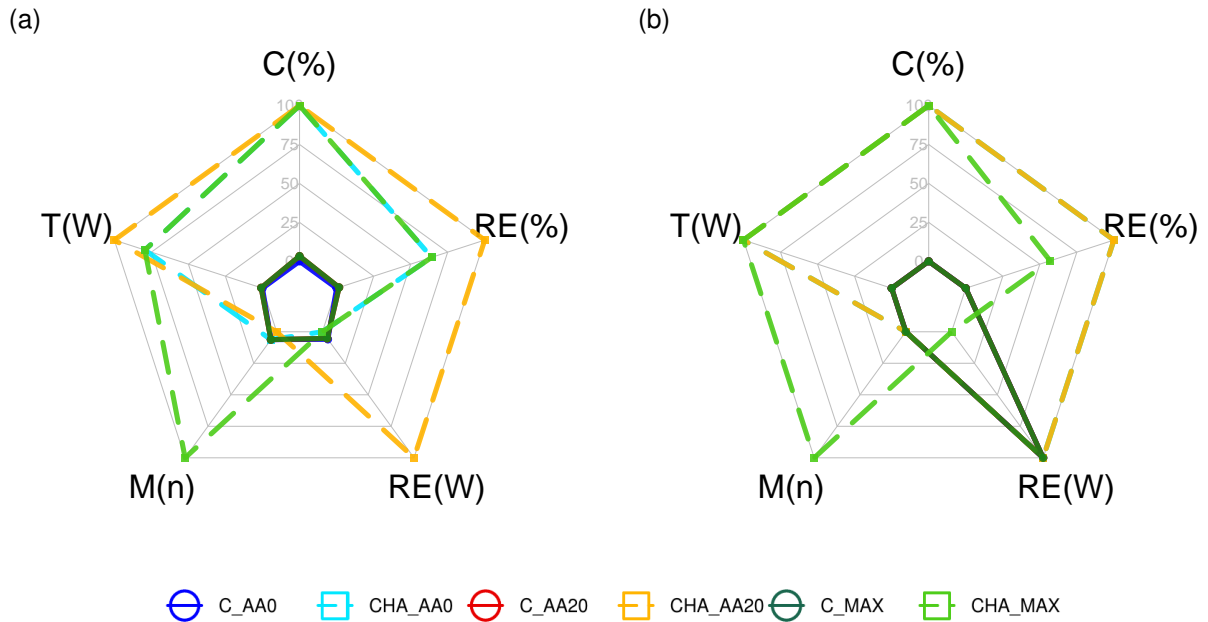
Fonte: Elaborado pelo autor.

Em caso de similaridade de área, considera-se como métricas de menor relevância a carga da AZ ( $C(\%)$ ).

Entre os testes aplicados na AZ1, Figura 15-(a), considerando HA com consolidação destaca-se o teste CHA\_AA0, e apenas consolidação o teste C\_MAX, que apresentam os menores polígonos e por conseguinte, os melhores resultados. Observa-se que o polígono do teste com HA é ainda menor que o teste que não possui essa carga adicional. Todavia, este fato não ocorre na AZ2, Figura 15-(b), em que se destaca o teste C\_AA20, por unanimidade apresentou os menores valores em cada métrica, próximos do teste C\_AA0. Adicionalmente, o teste CHA\_AA0 na AZ2 apresenta os melhores resultados entre os testes que realizam consolidação com HA.

Quando há sobreposição entre as linhas do polígono, indica um empate de valores, que ocorreu nos testes da AZ3 e AZ4. Na AZ3, conforme Figura 16-(a), destaca-se o teste CHA\_AA0 com o menor polígono, empatando-se entre os testes C\_AA0 e C\_AA20, Mas com métricas muito próximas do teste C\_MAX. Na AZ4, a Figura 16-(b) possui diversas sobreposições de linhas. Entre os testes que aplicam apenas a consolidação, ocorreu empate por unanimidade (indistinguível na figura, mas reconfirmado), e entre os testes que consideram HA, destaca-se o CHA\_MAX, cumprindo o objetivo na maioria das métricas. Destaca-se a métrica  $RE(W)$ , em que o CHA\_MAX destaca-se por representar a maior redução de consumo.

Figura 16 – Testes em perspectiva local. (a) AZ3 melhores testes: CHA\_AA0 e empate entre C\_AA0 e C\_AA20. (b) AZ4 melhores testes: CHA\_MAX e empate entre C\_AA0, C\_AA20 e C\_MAX.

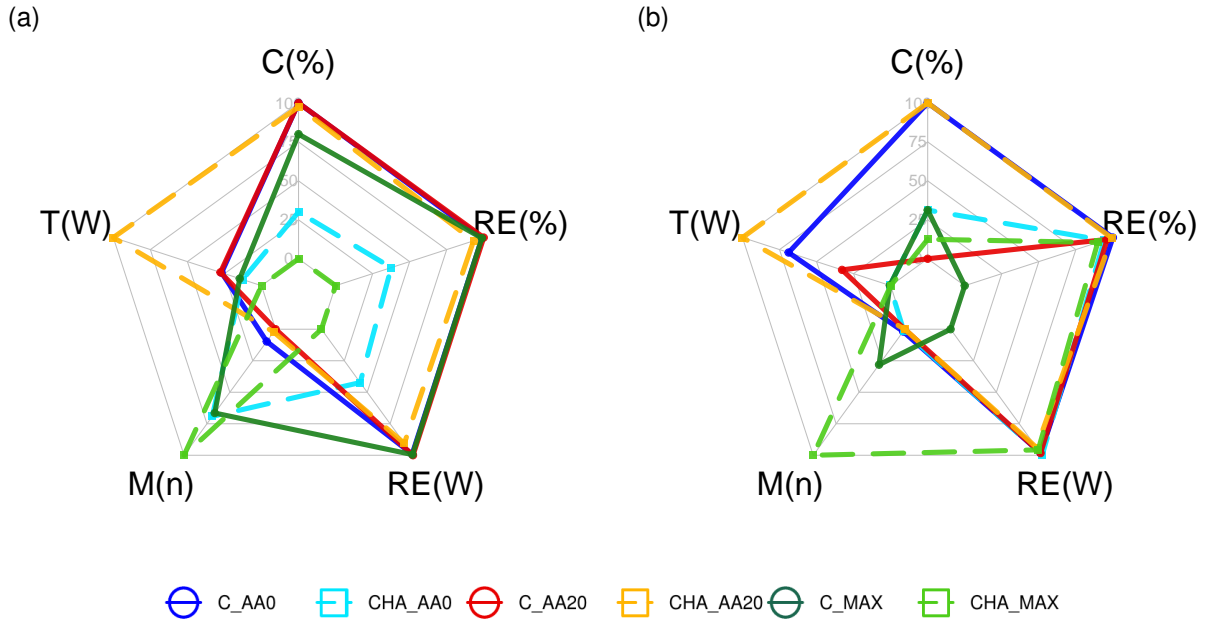


Fonte: Elaborado pelo autor.

Para os testes aplicados na AZ5, Figura 17-(a), destaca-se o teste CHA\_MAX com o menor polígono entre todos os testes, mesmo com a carga adicional de HA. Entre os testes sem HA, destaca-se o C\_AA20 que mesmo com as restrições de afinidade ainda apresentou o menor polígono para este caso. Os testes referentes a AZ6, Figura 17-(b) o menor polígono é representado pelo teste C\_MAX, em seguida do algoritmo CHA\_AA0 apresenta o menor polígono.

Na maioria das AZs, os testes que realizam apenas a consolidação apresentam os menores polígonos. Em todas as seis AZs, destaca-se o alto número de migrações ( $M(n)$ ) atribuído por unanimidade ao teste CHA\_MAX, que deve-se a natureza do algoritmo em realizar um maior número de migrações para cumprir com o objetivo de uma consolidação aproximada do ótimo. Adicionalmente, destaca-se para todas as AZs, os testes avaliados sob as métricas de redução de consumo de energia, tanto  $RE(W)$  quanto  $RE(\%)$ , que por unanimidade apresentam os piores valores para os testes CHA\_AA20. Isso ocorre pois nem sempre uma consolidação é realizada devido a afinidade de 20% das máquinas não puderem ser migradas.

Figura 17 – Testes em perspectiva local. (a) AZ5 melhores testes: CHA\_MAX e C\_AA20. (b) AZ6 melhores testes: CHA\_AA0 e C\_MAX.



Fonte: Elaborado pelo autor.

#### 4.2 PERSPECTIVA GLOBAL: SIMULAÇÃO TOTALIZADA

Em perspectiva global, os valores referem-se a contabilização de cada métrica para toda a execução de cada um dos seis testes definidos anteriormente (Subseção 3.3.3). São utilizadas seis métricas a nível global, que consistem no somatório das de todas as ocorrências. O total de energia consumida  $TE(MWh)$ , indica o quanto a AZ consumiu em cada teste, estabelecida em  $MWh$  devido sua magnitude e natureza de medição (consumo por hora), o menor valor é o melhor resultado. O total de redução de consumo com as consolidações  $RE(Wh)$  representa o somatório das reduções de cada gatilho de consolidação, o maior valor é o melhor. A métrica que contabiliza o total de violações de acordo de nível de serviço (SLA)  $SLAV(n)$ , refere-se a quantas requisições para instanciar novas máquinas virtuais (MVs) (seja ela regular, crítica ou réplica) foram rejeitadas devido a AZ estar em sobrecarga (100% de uso), sendo melhor o menor valor. O número total de migrações  $M(n)$  é o somatório de todas as migrações realizadas, sendo melhor o de menor valor. O total de falso-positivos  $FP(n)$ , contabiliza os casos em que são acionados os gatilhos para realizar uma consolidação, mas nenhum servidor é desativado, o menor valor é melhor. Por fim,  $G(n)$  representa o total de gatilhos de consolidação, *i.e.*, o somatório das ações de consolidação (incluindo os falso-positivos), o menor valor é o melhor. A Tabela 10 resume a relação das métricas em relação a unidade, o objetivo (o símbolo indica uma interpretação inversa) e as equações.

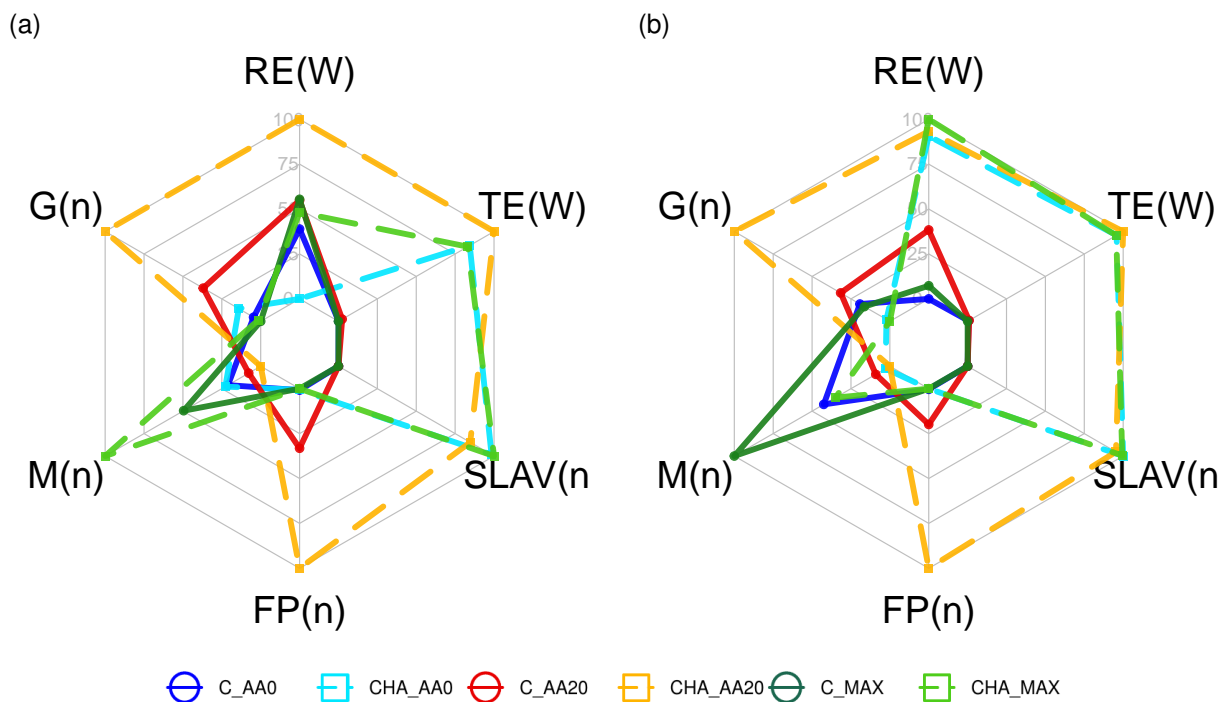
Tabela 10 – Tabela de métricas utilizadas para análise e discussão dos resultados.

Métrica	Descrição	Unidade	Objetivo (melhor)	Equações
$TE(MWh)$	Consumo total de energia	$MWh$	Menor	$\sum_{t=0}^{t_f} const.$
$RE(Wh)$	Redução do consumo	$Wh$	Maior*	$\sum_{t=0}^{t_f} Eq. 3.6$
$M(n)$	Número de migrações	Inteiro	Menor	$\sum_{t=0}^{t_f} migrat.$
$SLAV(n)$	Violações de SLA	Inteiro	Menor	$\sum_{t=0}^{t_f} SLAV_t.$
$FP(n)$	Falso-positivos	Inteiro	Menor	$\sum_{t=0}^{t_f} fpt.$
$G(n)$	Gatilhos de consolidação	Inteiro	Menor	$\sum_{t=0}^{t_f} gt.$

Fonte: Elaborado pelo autor.

Do mesmo modo que na avaliação sob perspectiva local, utiliza-se avaliação dos gráficos de radar conforme área dos polígonos, sendo que a melhor solução é a que resulta no menor polígono centralizado. Para cada AZ, são destacados os resultados em testes que possuem a demanda de HA com a consolidação de MVs paralelamente aos testes que consideram apenas consolidação. Para permitir a análise de menor área, a métrica de redução de consumo ( $RE(Wh)$ ) deve ser interpretada de forma inversa para encontrar o menor polígono. Em caso de similaridade de área, considera-se como métricas de menor peso, o número de falso-positivos ( $FP(n)$ ), e em seguida o número de gatilhos ( $G(n)$ ).

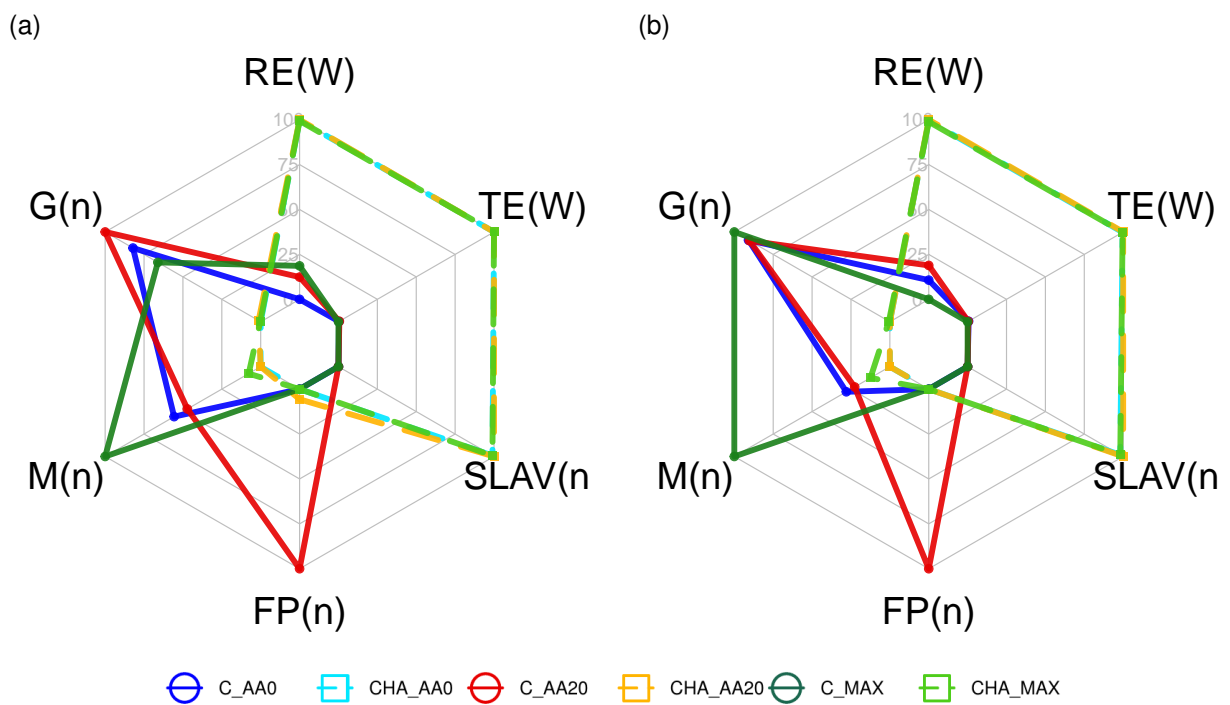
Figura 18 – Testes em perspectiva global. (a) AZ1 melhores testes: CHA\_AA0 e C\_AA0. (b) AZ2 melhores testes CHA\_AA0 e C\_AA0.



Fonte: Elaborado pelo autor.

Com base na Figura 18-(a), referente a AZ1, destaca-se o teste CHA\_AA0 com o menor polígono, que reflete a análise em perspectiva local para a mesma AZ, que também havia apresentado o menor polígono entre os testes. Entre os testes que aplicam apenas consolidação, destaca-se C\_AA0 com o menor polígono entre todos os testes na AZ1. Para os testes da AZ2, conforme Figura 18-(b), também destaca-se o C\_AA0 entre os testes que realizam apenas consolidação, com o menor polígono entre todos os testes. Em seguida, entre os testes que aplicam consolidação em ambiente com HA, também destaca-se o teste CHA\_AA0, sendo este um teste que também se destacou na avaliação em perspectiva local. Observa-se que a avaliação global é um reflexo da avaliação do *snapshot* na AZ1 e AZ2, mantendo-se a coerência entre o comportamento dos *snapshots* e o resultado global da simulação.

Figura 19 – Testes em perspectiva global. (a) AZ3 melhores resultados: C\_AA0 e grande similaridade entre os três testes CHA. (b) AZ4 melhores resultados .

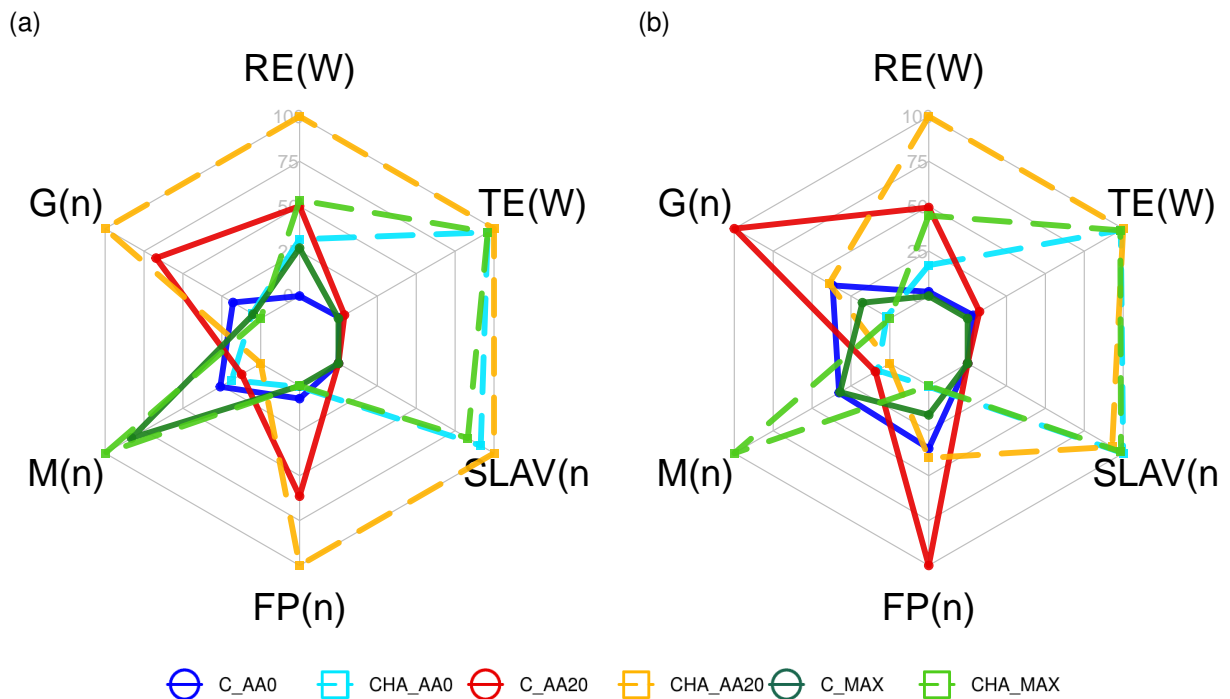


Fonte: Elaborado pelo autor.

Mesmo em se tratando de AZs com características diferentes, os testes executados na AZ3 e na AZ4 possuem comportamentos similares. Assim, a Figura 19-(a), referente a AZ3 e a Figura 19-(b) da AZ4, destaca o menor polígono para o teste C\_AA0 entre os testes que aplicam apenas consolidação. Em ambas as AZs, há uma grande similaridade entre os testes que aplicam consolidação em ambiente com HA, sendo que na AZ3 CHA\_AA0 apresenta área um pouco menor que os demais, en-

quanto na AZ4, ha um claro empate entre os testes CHA\_AA0 e CHA\_AA20. Este comportamento também é um reflexo da análise em perspectiva local, que apresentou os mesmos resultados. Ao observar paralelamente os gráficos da AZ3 e AZ4, nota-se que os testes que consideram consolidação com HA (linhas tracejadas) tendem a ter piores resultados em relação ao consumo de energia ( $TE(W)$ ), violações de SLA ( $SLAV(n)$ ) e redução de energia ( $RE(W)$ ). Enquanto os testes que aplicam apenas consolidação (linhas contínuas) tendem a apresentar piores resultados entre as métricas  $G(n)$ ,  $FP(n)$ , e  $M(n)$ , citadas inicialmente como métricas de menor peso. Adicionalmente, salienta-se que na avaliação em perspectiva local, também houve empates entre testes, tanto para AZ3 quanto para AZ4, indicando que há uma similaridade entre ambas as AZs.

Figura 20 – Testes em perspectiva global. (a) AZ5 melhores resultados: C\_AA0 e CHA\_AA0. (b) AZ6 melhores resultados: C\_MAX e CHA\_AA0.



Fonte: Elaborado pelo autor.

Diferentemente do cenário anterior, os testes aplicados na AZ5 e AZ6 indicam comportamentos distintos, sem empates. Segundo Figura 20-(a), que representa AZ5, destaca-se claramente como menor polígono o teste C\_AA0 e entre os testes que aplicam consolidação com HA, destaca-se o teste CHA\_AA0. Apenas neste caso, a análise em perspectiva local não reflete a análise em perspectiva global, apresentando resultados diferentes. Para a AZ6, segundo Figura 20-(b), destaca-se o teste C\_MAX com o menor polígono entre todos os testes, e o teste CHA\_AA0 entre os tes-

tes que aplicam consolidação em HA. A análise em perspectiva local indica um reflexo nos resultados em perspectiva global, apresentando os mesmos testes como destaque. Ao observar paralelamente os polígonos dos testes da AZ5 e AZ6, que o teste CHA\_MAX apresenta um alto número de migrações ( $M(n)$ ), que é um comportamento característico da estratégia utilizada.

Assim como na perspectiva local, testes que aplicam consolidação de MVs em ambiente com HA (CHA\_\*) apresentam polígonos com áreas maiores do que os testes que aplicam apenas consolidação em ambiente regular (C\_\*). De modo geral, os testes relativos a CHA\_\* tendem a apresentar os piores resultados para as métricas  $RE(W)$ ,  $TE(W)$  e  $SLAV(n)$ . Enquanto os testes relativos a C\_\* tendem aos piores resultados para as métricas  $G(n)$ ,  $FP(n)$  e  $M(n)$ , mesmo que o número de migrações também seja destaque para o teste CHA\_MAX.

#### 4.3 COMPARAÇÃO COM O CENÁRIO DE BASE DO EUCALYPTUS

Considerando todas as métricas discutidas, a única métrica que é comparável com a execução de Eucalyptus é o consumo total de energia  $TE(W)$ . De fato, a execução do Eucalyptus não possui gatilhos ( $G(n)$ ), migrações ( $M(n)$ ), falso-positivos ( $FP(n)$ ) ou violações ( $SLAV(n)$ ), pois como não há consolidação, então não há as métricas de redução do consumo de energia ( $RE(W)$ ). Assim, a Tabela 11 relaciona EUCA com os valores correspondentes aos testes que se destacaram na execução do teste global, para os testes de consolidação com HA (CHA\_) e sem HA (C\_). Adicionalmente, indica-se o percentual de impacto dos testes relacionados, constituindo uma redução (-) ou um acréscimo (+) no consumo de energia total, quando comparado com o teste de base EUCA.

Tabela 11 – Relação da execução de base (EUCA) com melhores resultados da execução geral na métrica  $ET(W)$ .

Região	AZ	EUCA	Resultado C_			Resultado CHA_		
		$ET(W)$	Teste	$ET(W)$	%	Teste	$ET(W)$	%
$R_0$	AZ1	15,8	AA0	7,9	-50,0%	AA0	12,0	-24,1%
	AZ2	23,6	AA0	9,1	-61,4%	AA0	28,9	+22,5%
	AZ3	22,1	AA0	5,5	-75,1%	AA0	29,2	+32,1%
$R_1$	AZ4	13,3	AA0	5,3	-60,2%	AA0	16,9	+27,1%
	AZ5	17,2	AA0	10,9	-36,6%	AA0	17,8	+3,5%
	AZ6	17,9	MAX	12,8	-28,5%	AA0	18,8	+5,0%

Fonte: Elaborado pelo autor.

As unidades de medidas referentes a métrica  $ET(W)$  é determinada em Megawatt-hora devido sua magnitude (embora denotada em Watts), compreendendo o consumo



de energia durante todo o período da simulação. Observa-se na Tabela 11 que os testes AA0 majoritariamente apresentam os melhores resultados, tanto para os cenários CHA\_ quanto C\_. Nesse sentido, ao analisar os testes em que é aplicada apenas a consolidação de MVs, há uma redução de até 75% no consumo de energia total para a AZ3. Ao considerar a análise de probabilidade de cargas (Figura 11), a AZ3 fica ociosa na maior parte do tempo do EUCA. De fato isso mostra o potencial da aplicação da consolidação, principalmente quando aplicado em ambientes em que há pouca utilização dos recursos físicos.

Em contrapartida, quando analisada sob os testes com a carga adicional de HA, a mesma AZ3 apresenta o maior acréscimo no consumo de energia (+32%). Nesse sentido, atribui-se esse efeito ao balanceamento de carga no momento de selecionar a AZ que será instanciada a réplica, entre as AZs de uma mesma região. Assim, a maior parte das réplicas da AZ1 e AZ2 foram instanciadas na AZ3, contribuindo para esse impacto no consumo de energia. Ainda analisando os cenários com a carga de HA, a AZ1 mostra a melhor redução no consumo de energia final, mesmo com a referida carga de HA. Essa redução se deve a carga de instanciar as réplicas da AZ1 em parte na AZ2 e, principalmente na AZ3 (por isso a elevação no consumo), recebendo poucas réplicas das AZ2 e AZ3.

Ao aplicar a mesma discussão na  $R_1$ , observa-se que a AZ4 também apresenta a maior redução (-60%) entre os testes que aplicam apenas a consolidação, em contrapartida com seu comportamento nos testes com HA apresenta o maior acréscimo na métrica (+27%). Usando a mesma relação que anteriormente aplicada na AZ3, quando analisada a probabilidade de distribuição de carga na Figura 12, indica-se uma potencial ociosidade na AZ4, que leva a receber a maior parte das réplicas das demais AZs. Entretanto, a AZ5 e AZ6 mantêm-se em conformidade, indicando que há um pequeno acréscimo no consumo de energia total, mesmo com o acréscimo da carga de HA.

Em geral, destaca-se o algoritmo AA0 entre os testes avaliados sob perspectiva global, apresentando melhores resultados do que os testes com o algoritmo MAX, cujo objetivo inicial era de reduzir ao máximo o consumo de energia. Com base nas discussões apresentadas, pode-se considerar que tão importante quanto a estratégia de consolidação ou replicação, é a estratégia que realiza o balanceamento de carga entre as AZs de uma mesma região. Assim, com base nos resultados, quando avaliada sob a perspectiva global, considera-se que a região  $R_1$  apresenta-se melhor equilibrada, quando comparada com a região  $R_0$ . De modo geral, salienta-se que esse acréscimo no consumo de energia nos testes CHA\_ não é necessariamente um problema, pois já era esperado que uma carga de HA teria esse impacto.

#### 4.4 CONSIDERAÇÕES PARCIAIS

A análise dos resultados é uma tarefa complexa quando trata-se de dados multivariados, pois apresentam escalas e unidades de medida diferentes. Nesse sentido, para a discussão dos resultados, são adotados os gráficos de radar, aplicando-se uma didática de seleção dos melhores testes conforme área do polígono. Desse modo, destacam-se os testes com melhores resultados através de uma análise visual, com base na seleção do menor polígono. De imediato, observa-se entre todos os gráficos, que a maioria dos testes que aplicam consolidação com HA (CHA\_) possuem uma área de gráfico maior quando comparadas com os testes que aplicam apenas a consolidação de MVs sem a carga adicional de HA. De fato, é natural que os testes CHA\_ apresentem pior desempenho em relação às métricas de consumo de energia e de carga da AZ. Todavia, alguns testes mostram que os resultados dos testes com a carga de HA apresentam resultados ainda melhores, como na AZ1 (Figura 15-(a)) e na AZ5 (Figura 17-(a)), quando comparados com testes sem a carga adicional de HA.

No decorrer do presente Capítulo são apresentados e discutidos os resultados dos seis testes em cada AZ sob ambas as perspectivas local e global. Assim, permite-se que sejam aplicadas métricas inerentes a cada perspectiva, objetivando uma análise com maior granularidade. Adicionalmente, as duas perspectivas são posteriormente comparadas, verificando se o comportamento local também se reflete em um comportamento global. De modo geral, observa-se que a maioria dos testes que apresentaram os melhores resultados sob a perspectiva local também se refletem para a perspectiva global, com exceção da AZ5.

Ao comparar os resultados com o cenário de base, com a execução do simulador considerando a alocação padrão do Eucalyptus, a discussão parte para um viés de balanceamento de carga entre as AZs de uma mesma região. Dessa forma, observa-se inicialmente que os testes que não possuem a carga de HA apresentam redução de até 75%, ressaltando que AZs com baixa carga de fato tendem a apresentar melhores resultados. Quando avaliados os testes com a carga de HA, então verifica-se que a região  $R_1$  apresenta um balanceamento mais equilibrado ao alocar as réplicas das demais AZs. A região  $R_1$  indica duas AZs com incremento mínimo no consumo de energia, indicando que o algoritmo CHA\_AA0 permite adicionar um mecanismo de HA com reduzido impacto financeiro.

## 5 CONSIDERAÇÕES & TRABALHOS FUTUROS

Organizações que dependem da continuidade de negócios baseados em serviços críticos de Tecnologia da Informação (TI), demandam cada vez mais alta disponibilidade (HA) para mitigar os prejuízos causados pela interrupção de serviços em caso de desastres. Todavia, este requisito possui um elevado custo de execução, principalmente no que refere-se a redundância de equipamentos, causando impacto no consumo de energia elétrica. Atualmente, Provedores de Serviços em Nuvem (CSPs) permitem implementar técnicas para obter HA baseado em replicação de serviços críticos com base nas características de isolamento de arquiteturas multi-AZ. Por outro lado, para possibilitar a redução do consumo energia em servidores, estratégias de consolidação de máquinas virtuais (MVs) podem causar violações de acordo de nível de serviço (SLA), infringindo principalmente as políticas de afinidade e anti-afinidade. Assim, CHAVE atua na integração entre ambas abordagens, executando-as em níveis diferentes: a consolidação é aplicada apenas internamente à zona de disponibilidade (AZ), respeitando a afinidade MV-AZ; e a replicação é executada apenas entre AZs distintas, respeitando a anti-afinidade MV-MV (crítica-réplica). A estratégia de consolidação de MVs baseia-se no algoritmo *First Fit Decreasing* (FFD), e tem duas modificações aplicadas no presente trabalho: os algoritmos MAX e *Affinity-Aware* (AA). Com base na probabilidade de eventos independentes e em análise por diagrama de bloco de confiabilidade (RBD), CHAVE especifica a quantidade ideal de réplicas para obter a taxa desejada de HA, definindo o número ideal de réplicas e concebendo a HA sob demanda. Nesse sentido, a alta complexidade relativa ao gerenciamento das réplicas e da consolidação de MV, pode ser uma barreira para aplicar a solução CHAVE por um administrador em um CSP real.

Os dados obtidos com a execução dos testes em simulação numérica foram analisados sob duas perspectivas: local e global, permitindo a avaliação dos algoritmos MAX e AA que se destacam entre as métricas inerentes a cada perspectiva. O objetivo do MAX é a máxima consolidação para obter o menor consumo de energia, porém exige um maior número de migrações. AA considera a restrição de afinidade entre MV-servidor antes de decidir se pode migrar. Assim, a execução dos testes mostram que ao considerar um determinado conjunto de métricas, o algoritmo MAX nem sempre apresenta os melhores resultados. Em geral, destacam-se os testes cujo algoritmo AA são configurados em 0% de afinidade, encontrados por análise visual (menor polígono) dos gráficos de radar. Quando é aplicado apenas a consolidação de MVs (sem a carga adicional de HA), CHAVE apresenta redução no consumo de energia para todos os cenários realizados. Adicionalmente, quando comparados os cenários

em uma perspectiva global, observa-se um comportamento similar aos testes em perspectiva local, e portanto, mantendo-se consistente.

Ao comparar os resultados em perspectiva global com a execução de base do Eucalyptus, discorre-se sobre o equilíbrio de carga entre as AZs de uma mesma região. Nesse sentido, os resultados obtidos em simulação, utilizando *traces* reais, mostram que mesmo ao adicionar uma demanda de 20% de HA, implica em um impacto mínimo no consumo de energia. Em contrapartida, testes sem a carga de HA permitem obter uma redução no consumo de energia em até 75%, ressaltando que AZs com baixa carga de fato tendem a apresentar melhores resultados. Quando avaliados os testes com a carga de HA, então verifica-se que a região  $R_1$  está melhor balanceada ao alocar as réplicas das demais AZs. A região  $R_1$  indica duas AZs com incremento mínimo no consumo de energia, indicando que o algoritmo CHA\_AA0 permite a replicação de MVs com reduzido impacto financeiro. Por fim, conclui-se a importância da avaliação da infraestrutura do CSP como um todo, pois também é importante considerar a estratégia que realiza o balanceamento de carga entre as AZs de uma mesma região. Assim, com base nos resultados, quando avaliada sob a perspectiva global, considera-se que a região  $R_1$  apresenta-se melhor equilibrada quando comparada com a região  $R_0$ .

Embora não tenha sido quantificado o impacto financeiro, observa-se que a aplicação do CHAVE pode permitir a redução de custos de energia e evita a aquisição de mais servidores, pois a consolidação permite utilizar melhor os recursos disponíveis. Quando o objetivo é fornecer HA, CHAVE mostra-se satisfatório por manter um consumo de energia similar cenários sem consolidação ou HA. Na prática, considerando um ambiente de computação em nuvem real, a tomada de decisão sobre posicionamento inicial, migração e replicação podem ser realizadas através das *Application Programming Interfaces* (APIs) a nível de administrador, desde que seja fornecida essa possibilidade. Assim, CHAVE pode ser associado às plataformas de computação em nuvem reais, sejam as públicas, como *Google Cloud Platform* (GCP), *Amazon Web Services* (AWS) ou Azure, ou privadas como Eucalyptus, OpenStack e CloudStack, que necessitem fornecer ou não um mecanismo de HA.

Entre as principais dificuldades encontradas, a nível conceitual foi estabelecer que as abordagens de replicação e consolidação devem estar em camadas diferentes para que uma ação não interfira nas restrições de outra. A nível de implementação, foi desenvolver a integração do CHAVE ao simulador EAVIRA e adequá-lo aos requisitos inicialmente estabelecidos. Por fim, encontrar uma abordagem didática para discutir os resultados com métricas em diferentes magnitudes e unidades de medidas, estabelecendo-se os gráficos em radar com análise visual dos polígonos.

## 5.1 CONTRIBUIÇÕES DO TRABALHO

Inicialmente, há um reforço na discussão teórica os temas relativos a eficiência energética e HA. Adicionalmente, há itens pontuais que são desenvolvidos para o contexto do presente trabalho, mas que podem ser reutilizados em outros contextos de pesquisa, como as equações e algoritmos desenvolvidos. Porém, entre as principais contribuições do presente trabalho, destacam-se três itens:

- O uso de uma solução baseada em arquitetura multi-AZ, separando as estratégias de consolidação e replicação em camadas distintas, estabelecendo uma relação entre ambas;
- A discussão sobre as políticas de afinidade e anti-afinidade, considerando-as como restrições que devem ser devidamente cumpridas em SLA; e
- A aplicação da análise por RBD em conjunto com a probabilidade de eventos independentes para o cálculo da taxa de HA, fornecendo a concepção de HA sob demanda.

Conceitualmente, discutir os resultados numéricos multivariados entre diferentes métricas através de gráficos de radar, utilizando apenas análise visual permitiu uma análise diferenciada padrão estabelecido no meio científico.

## 5.2 TRABALHOS FUTUROS

Ainda há diversos requisitos a serem considerados para que CHAVE compreenda todos os recursos de uma nuvem computacional. Um destes requisitos é considerar os recursos de armazenamento e rede para alocação e quantificar o impacto da migração nos recursos de rede. Adicionar novas funcionalidades, como o redimensionamento de recursos para MVs (requisito de elasticidade), e redimensionamento de infraestrutura ao adicionar novos servidores sob demanda. Outra estratégia que permite melhorar a utilização de recursos é considerar o conceito de *overcommitting*. Todas essas novas funcionalidades estão pré-implementadas no simulador EAVIRA, e devem ser testados e analisados como trabalhos futuros.

Especificamente para o CHAVE, pode-se trabalhar com maior profundidade o algoritmo de balanceamento de carga entre as AZ. Pode-se estabelecer que os objetivos deste balanceador de carga entre AZs também possa considerar a taxa de HA individual de cada AZ para tomada de decisão.

Por fim, utilizar outros *traces* com configurações de arquiteturas multi-AZ distintas, evidenciando as características essenciais da solução CHAVE. *Benchmarks*

com outras soluções similares estão em vias de realização, permitindo estabelecer uma relação entre trabalhos correlatos.

Para uma aplicação real, e portanto mais complexa, trabalhos futuros podem compreender outras métricas, como o impacto financeiro da aplicação do CHAVE. A presente solução também pode ser avaliada sob duas perspectivas: a de corretor, atuando em conjunto com o escalonador pré-existente da nuvem, ou um escalonador, sendo este o único responsável pelas operações relativas a HA e eficiência energética. Para ser aplicado em um ambiente real de grande porte, como a AWS, CHAVE deve ser modificado para ser escalável, tornando-se um componente crítico e tolerante a falhas.

### 5.3 PUBLICAÇÕES

Durante o período do presente trabalho, foram publicados quatro artigos científicos relativos às linhas de pesquisa que constam no presente trabalho. Organizados em decrescente quanto a data de publicação:

- CAMARGO D.S.; PILLON, M.A.; MIERS, C.C.; KOSLOVSKI, G.P.; RIGHI, R.R. *CHAVE: Resource Consolidation with High Availability on Virtualized Environments*. In: **37th International Conference of the Chilean Computer Science Society (SCCC)**, 2018, Santiago. Proceedings of SCCC-2018, 2018. p. 1-8.
- CAMARGO D.S.; PILLON, M.A.; MIERS, C.C.; KOSLOVSKI, G.P. *Uma proposta de orquestração de nuvem computacional baseada em consolidação, elasticidade e disponibilidade*. In: **XVII Escola Regional de Alto Desempenho do Estado do Rio Grande do Sul (ERAD/RS)**, 2017, Ijuí/RS. Anais do XVII ERAD/RS. Porto Alegre/RS: SBC, 2017
- CAMARGO D.S.; MIERS, C.C.; PILLON, M. A.; KOSLOVSKI, G.P. *MeHarCEn: Um Método de Harmonização do Consumo de Energia em Data Centers*. In: **Revista de Informática Teórica e Aplicada: RITA**, v. 24, p. 47-70, 2017.
- CAMARGO, D.S.; JUNIOR, A.C.; PILLON, M.A.; PRADO, R.B.S. *Mapeamento Térmico em Data Center de Pequeno e de Médio Porte*. In: **Brazilian Symposium on Computing Systems Engineering (SBESC)**, 2017, Curitiba. Embedded Systems Competition 2017 - WND IoT Challenge, 2017.

## REFERÊNCIAS

- ABAUNZA, F.; HAMERI, A.-P.; NIEMI, T. EEUI: a new measure to monitor and manage energy efficiency in data centers. **International Journal of Productivity and Performance Management**, v. 67, n. 1, p. 111–127, 2018.
- ABDELSAMEA, A.; et. al. Virtual machine consolidation enhancement using hybrid regression algorithms. **Egyptian Informatics Journal**, v. 18, Issue 3, p. 161–170, nov. 2017. ISSN 1110-8665. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1110866516300925>>.
- AHMAD, R. W.; et. al. A survey on virtual machine migration and server consolidation frameworks for cloud data centers. **JNCA**, v. 52, p. 11–25, 2015. ISSN 1084-8045. Disponível em: <<http://dx.doi.org/10.1016/j.jnca.2015.02.002>>.
- AKHTER, N.; OTHMAN, M. Energy aware resource allocation of cloud data center: Review and open issues. **Cluster Comput**, v. 19, n. 3, p. 1163–1182, 2016. ISSN 1386-7857, 1573-7543.
- AKOWUAH, F. E. et al. Standards and Guides for Implementing Security and Privacy for Health Information Technology. **Security and Privacy Management, Techniques, and Protocols**, p. 214–236, 2018.
- ALAHMADI, A. et al. Enhanced First-Fit Decreasing Algorithm for Energy-Aware Job Scheduling in Cloud. 2014 International Conference on Computational Science and Computational Intelligence, v. 2, p. 69–74, mar. 2014.
- ALKAWSI, G. A.; et al. Factors influencing the adoption of cloud computing in SME: A systematic review. In: **(IEEE-ISMSC) 2015**. IEEE, 2015. Disponível em: <<http://ieeexplore.ieee.org/abstract/document/7594056/>>.
- ALLYBOKUS, Z.; et. al. Virtual function placement for service chaining with partial orders and anti-affinity rules. **Networks**, v. 71, n. 2, p. 97–106, mar. 2018. ISSN 1097-0037. Disponível em: <<https://onlinelibrary.wiley.com/doi/abs/10.1002/net.21768>>.
- ALSHAMMARI, M. M.; ALWAN, A. A.; ALSHAIKHLI, I. F. T. Data recovery and business continuity in cloud computing: a review of the research literature. **International Journal of Advancements in Computing Technology(IJACT)**, v. 8, p. 80–94, dez. 2016. ISSN 2005-8039. Disponível em: <<http://www.globalcis.org/ijact/ppl/IJACT3598PPL.pdf>>.
- AMOON, M. Adaptive Framework for Reliable Cloud Computing Environment. **IEEE Access**, v. 4, p. 9469–9478, 2016. ISSN 2169-3536. Disponível em: <<http://ieeexplore.ieee.org/document/7742909/>>.
- ASCIERTO, R.; PEARL, T. **2018 Data Center Industry Survey Results**. Massachusetts, EUA, 2018. 20 p. Disponível em: <<https://uptimeinstitute.com/2018-data-center-industry-survey-results>>.

BARONA, R.; ANITA, E. A. M. A survey on data breach challenges in cloud computing security: Issues and threats. In: **2017 International Conference on Circuit ,Power and Computing Technologies (ICCPCT)**. IEEE, 2017. p. 1–8. ISBN 978-1-5090-4967-7. Disponível em: <<http://ieeexplore.ieee.org/document/8074287/>>.

BAUER, E.; ADAMS, R. **Reliability and Availability of Cloud Computing**. 1 edition. ed. Piscataway, NJ : Hoboken, NJ: Wiley-IEEE Press, 2012. ISBN 978-1-118-17701-3.

BELOGLAZOV, A. **Energy-efficient Management Of Virtual Machines In Data Centers For Cloud Computing**. Tese (PHD Thesis) — The University Of Melbourne, 2013. Disponível em: <<https://minerva-access.unimelb.edu.au/handle/11343/38198>>.

BELOGLAZOV, A.; BUYYA, R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. **Concurrency Computat.: Pract. Exper.**, v. 24, n. 13, p. 1397–1420, 2012. ISSN 1532-0634.

BELOGLAZOV, A. et al. A taxonomy and survey of energy-efficient data centers and cloud computing systems. **Arxiv:1007.0066 [cs]**, 2010. Disponível em: <<http://arxiv.org/abs/1007.0066>>.

BEYER, B. et al. **Site Reliability Engineering**. 1st. ed. Sebastopol, CA: O'Reilly, 2016. ISBN 978-1-4919-2912-4.

BLIND, K.; PETERSEN, S. S.; RIILLO, C. A. F. The impact of standards and regulation on innovation in uncertain markets. **Research Policy**, v. 46, n. 1, p. 249–264, fev. 2017. ISSN 0048-7333. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0048733316301743>>.

BORGOLTE, K. et al. Cloud Strife: Mitigating the Security Risks of Domain-Validated Certificates. In: . Internet Society, 2018. ISBN 978-1-891562-49-5. Disponível em: <<https://www.ndss-symposium.org/wp-content/uploads/sites/25/2018/02/ndss2018-06A-4-Borgolte-paper.pdf>>.

BOUDJADAR, J. An efficient energy-driven scheduling of DVFS-multicore systems with a hierarchy of shared memories. **2017 IEEE/ACM 21st International Symposium on Distributed Simulation and Real Time Applications (DS-RT)**, p. 1–8, out. 2017.

BUCHINGER, D.; CAVALCANTI, G. A. d. S.; HOUNSELL, M. d. S. Mecanismos de busca acadêmica: uma análise quantitativa. **Revista Brasileira de Computação Aplicada**, v. 6, n. 1, p. 108–120, abr. 2014. ISSN 2176-6649. Disponível em: <<http://seer.upf.br/index.php/rbca/article/view/3452>>.

BURNS, S. High availability in cloud computing prevents a SPOF. **TechTarget: SearchITOperations (07/2017)**, jul. 2017. Disponível em: <<http://searchitoperations.techtarget.com/tip/High-availability-in-cloud-computing-prevents-a-SPOF>>.

BUYYA, R.; BROBERG, J.; GOSCINSKI, A. M. (Ed.). **Cloud Computing: Principles and Paradigms**. 1 edition. ed. Hoboken, N.J: Wiley, 2011. ISBN 978-0-470-88799-8.

CAMARGO, D. S.; MIERS, C. C. GreenHop: Open source PUE continuous monitoring for small and medium data centers. **2016 XLII Latin American Computing Conference (CLEI)**, p. 1–12, 2016.



CAMARGO, D. S. et al. GreenHop: Open source environmental monitoring for small and medium data centers. **2016 35th International Conference of the Chilean Computer Science Society (SCCC)**, p. 1–12, 2016.

CAMARGO, D. S. et al. MeHarCEn: Um Método de Harmonização do Consumo de Energia em Data Centers. **Revista de Informática Teórica e Aplicada**, v. 24, n. 2, p. 47–70, dez. 2017. ISSN 21752745. Disponível em: <<http://seer.ufrgs.br/index.php/rita/article/view/VOL24-NR2-47>>.

CAO, J. et al. Checkpointing as a Service in Heterogeneous Cloud Environments. **arXiv:1411.1958 [cs]**, nov. 2014. ArXiv: 1411.1958. Disponível em: <<http://arxiv.org/abs/1411.1958>>.

CASELLAS, R. et al. Highly available SDN control of flexi-grid networks with network function virtualization-enabled replication. **IEEE/OSA Journal of Optical Communications and Networking**, v. 9, n. 2, p. A207–A215, fev. 2017. ISSN 1943-0620.

CÉRIN, C. et al. **Downtime Statistics of Current Cloud Solutions**. France, 2014. 5 p. Disponível em: <<http://iwgcr.org/wp-content/uploads/2014/03/downtime-statistics-current-1.3.pdf>>.

CHOUDHARY, R. Business Continuity Planning: A Study of Frameworks, Standards and Guidelines for Banks IT Services. v. 5, n. 8, p. 8, 2016.

COMERFORD, T. **How Data Center Operators Can Avoid Energy Price Hikes This Winter**. 2015. Disponível em: <<http://www.datacenterknowledge.com/archives/2015/01/29/data-center-operators-can-avoid-energy-price-hikes-winter/>>.

CONFORTO, E. C.; AMARAL, D. C.; SILVA, S. L. Roteiro para revisão bibliográfica sistemática: aplicação no desenvolvimento de produtos e gerenciamento de projetos. 2011. Disponível em: <<http://vision.ime.usp.br/~acmt/conforto.pdf>>.

COULOURIS, G. et al. **Distributed Systems: Concepts and Design**. 5 edition. ed. Boston: Pearson, 2011. ISBN 978-0-13-214301-1.

CRITCHLEY, T. **High Availability IT Services**. USA: CRC Press, 2014. ISBN 978-1-4822-5591-1.

CULLY, B.; et. al. Remus: High availability via asynchronous virtual machine replication. p. 161–174, 2008. Disponível em: <[https://www.usenix.org/legacy/event/nsdi08/tech/full\\_papers/cully/cully\\_html/](https://www.usenix.org/legacy/event/nsdi08/tech/full_papers/cully/cully_html/)>.

DAVID, B. **Benchmarking, Consistency, Distributed Database Management Systems, Distributed Systems, Eventual Consistency**. Karlsruhe, Germany: KIT Scientific Publishing, 2014. ISBN 978-3-7315-0186-2.

DAVIES, K.; DEMBIŃSKA, A. Computing moments of discrete order statistics from non-identical distributions. **Journal of Computational and Applied Mathematics**, v. 328, p. 340–354, jan. 2018. ISSN 0377-0427. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0377042717303588>>.

DAYARATHNA, M.; WEN, Y.; FAN, R. Data Center Energy Consumption Modeling: A Survey. **IEEE Communications Surveys Tutorials**, v. 18, n. 1, p. 732–794, 2016. ISSN 1553-877X.

DHANO, I. AN EXPERIMENT BASED EXAMINATION OF ENERGY OVERHEAD THROUGH VM MIGRATION. **International Journal of Advanced Research in Computer Science**, v. 8, n. 9, p. 655–660, 2017. ISSN 0976-5697. Disponível em: <<http://ijarcs.in/index.php/ijarcs/article/view/5070>>.

DONG, Y.; et. al. COLO: COarse-grained LOck-stepping virtual machines for non-stop service. **ACM Press**, ACM Press, p. 1–16, 2013. Disponível em: <<http://dl.acm.org/citation.cfm?doid=2523616.2523630>>.

DÓSA, G. et al. Tight absolute bound for First Fit Decreasing bin-packing:  $\text{FFD}(L) \leq 11/9 \text{OPT}(L) + 6/9$ . **TCS 2013**, v. 510, 2013.

DOW, E. M. Decomposed multi-objective bin-packing for virtual machine consolidation. **PeerJ Computer Science**, v. 2, p. e47, fev. 2016. ISSN 2376-5992. Disponível em: <<https://peerj.com/articles/cs-47>>.

DU, G.; HE, H.; MENG, Q. Energy-efficient scheduling for tasks with deadline in virtualized environments. 2014. Disponível em: <<https://www.hindawi.com/journals/mpe/2014/496843/>>.

ENDO, P. T.; et. al. High availability in clouds: systematic review and research challenges. **JoCCASA**, v. 5, p. 16, out. 2016. ISSN 2192-113X. Disponível em: <<https://doi.org/10.1186/s13677-016-0066-8>>.

ENOKIDO, T.; TAKIZAWA, M. An energy-efficient load balancing algorithm for virtual machine environments to perform communication type application processes. **2016 IEEE 30<sup>th</sup> International Conference On Advanced Information Networking And Applications (aina)**, p. 392–399, 2016.

FAYYAZ, M.; VLADIMIROVA, T. Survey and future directions of fault-tolerant distributed computing on board spacecraft. **Advances in Space Research**, v. 58, n. 11, p. 2352–2375, dez. 2016. ISSN 0273-1177. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0273117716304689>>.

FELLER, E. et al. Energy Management in IaaS Clouds: A Holistic Approach. **2012 IEEE Fifth International Conference on Cloud Computing**, p. 204–212, jun. 2012.

FERDAUS, M. H. et al. Virtual Machine Consolidation in Cloud Data Centers Using ACO Metaheuristic. In: **Euro-Par 2014 Parallel Processing**. Springer, Cham, 2014. (Lecture Notes in Computer Science), p. 306–317. ISBN 978-3-319-09872-2 978-3-319-09873-9. Disponível em: <[https://link.springer.com/chapter/10.1007/978-3-319-09873-9\\_26](https://link.springer.com/chapter/10.1007/978-3-319-09873-9_26)>.

FISCHER, W.; MITASCH, C. High availability clustering of virtual machines—possibilities and pitfalls. **12th Linuxtag, Wiesbaden/Germany**, v. 1, 2006. Disponível em: <<https://pdfs.semanticscholar.org/0072/1298260d3458b7d3555fa1e347e8ec11c84c.pdf>>.

FORD, D.; et. al. Availability in Globally Distributed Storage Systems. **9th USENIX**, 2010. Disponível em: <<https://research.google.com/pubs/pub36737.html>>.

FORSMAN, M. et al. Algorithms for automated live migration of virtual machines. v. 101, p. 110–126, 2015. ISSN 01641212. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0164121214002751>>.

GHOBAEI-ARANI, M.; JABBEHDARI, S.; POURMINA, M. A. An autonomic approach for resource provisioning of cloud services. **Cluster Comput**, v. 19, n. 3, p. 1017–1036, 2016. ISSN 1386-7857, 1573-7543.

GONCALVES, G. et al. Risk-based model for availability estimation of SAF redundancy models. In: . IEEE, 2016–06. p. 886–891. ISBN 978-1-5090-0679-3. Disponível em: <<http://ieeexplore.ieee.org/document/7543848/>>.

HAMEED, A.; et. al. A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. **Computing**, v. 98, n. 7, 2016. ISSN 0010-485X, 1436-5057.

HE, M.; et. al. Reverse Replication of Virtual Machines (rRVM) for Low Latency and High Availability Services. **2016 IEEE/ACM 9th International Conference on Utility and Cloud Computing (UCC)**, p. 118–127, 2016.

HINZ, M. et al. A Cost Model for IaaS Clouds Based on Virtual Machine Energy Consumption. **Journal of Grid Computing**, p. 1–20, maio 2018. ISSN 1570-7873, 1572-9184. Disponível em: <<https://link.springer.com/article/10.1007/s10723-018-9440-8>>.

IHARA, D.; LOPEZ-PIRES, F.; BARAN, B. Many-Objective Virtual Machine Placement for Dynamic Environments. **2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)**, dec 2015.

IORGA, M.; KARMEL, A. **Cloud Computing Security Essentials and Architecture**. 1st, chapter 3. ed. Boca Raton: CRC Press, 2016. (Cloud Computing Security: Foundations and Challenges). ISBN 978-1-4822-6095-3. Disponível em: <<https://csrc.nist.gov/publications/detail/book/2016/cloud-computing-security-essentials-and-architecture>>.

ISLAM, T.; MANIVANNAN, D. Predicting Application Failure in Cloud: A Machine Learning Approach. **2017 IEEE International Conference on Cognitive Computing (ICCC)**, p. 24–31, jun. 2017.

JACOBS, A. S.; et. al. Affinity measurement for NFV-enabled networks: A criteria-based approach. **2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)**, p. 125–133, maio 2017.

Nitin K. Jain. **Systems and methods for native network interface controller (NIC) teaming load balancing**. 2017. US9781041B2. Disponível em: <<https://patents.google.com/patent/US9781041B2/en>>.

JAMMAL, M.; KANSO, A.; SHAMI, A. CHASE: Component High Availability-Aware Scheduler in Cloud Computing Environment. **2015 IEEE 8th International Conference on Cloud Computing**, p. 477–484, jun. 2015.

KANSO, A.; et. al. Enhancing OpenStack Fault Tolerance for Provisioning Computing Environments. **2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)**, p. 77–83, jan. 2017.

KANZENBACH, K. Redundancy and Replication. **Masterseminar AKSS im WS 2014/15**, v. 7, 2014. Disponível em: <[http://www4.cs.fau.de/Lehre/WS14/MS\\_AKSS/](http://www4.cs.fau.de/Lehre/WS14/MS_AKSS/)>.

KESHAVARZI, A.; HAGHIGHAT, A. T.; BOHLOULI, M. Adaptive Resource Management and Provisioning in the Cloud Computing: A Survey of Definitions, Standards and Research Roadmaps. **KSII Transactions on Internet and Information Systems**, v. 11, n. 9, p. 4280–4300, set. 2017. Disponível em: <<http://itiis.org/digital-library/manuscript/1788>>.

KITCHENHAM, B. **Kitchenham, 2004 Procedures for Performing Systematic Reviews**. Keele, UK: Keele University Technical Report TR/SE-0401, 2004. ISSN ISSN:1353-7776.

KONG, W.; LUO, Y. Multi-level image software assembly technology based on OpenStack and Ceph. **2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference**, p. 307–310, 2016.

KOROTKOV, F. **Using preemptible VMs to cut Kubernetes Engine bills in half**. 2018. Disponível em: <<https://medium.com/google-cloud/using-preemptible-vms-to-cut-kubernetes-engine-bills-in-half-de2481b8e814>>.

KOSLOVSKI, G. et al. Reliability support in virtual infrastructures. **2010 IEEE Second International Conference on Cloud Computing Technology and Science**, p. 49–58, 2010.

KUMAR, P. R.; RAJ, P. H.; JELCIANA, P. Exploring Data Security Issues and Solutions in Cloud Computing. **Procedia Computer Science**, v. 125, p. 691–697, jan. 2018. ISSN 1877-0509. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1877050917328570>>.

LEVALLE, R. R. **Resilience by Teaming in Supply Chains and Networks**. Cham: Springer International Publishing, 2018. v. 5. (Automation, Collaboration, & E-Services, v. 5). ISBN 978-3-319-58321-1 978-3-319-58323-5. Disponível em: <<http://link.springer.com/10.1007/978-3-319-58323-5>>.

LI, J.; LI, Q. Data security and risk assessment in cloud computing. **ITM Web of Conferences**, v. 17, p. 03028, 2018. ISSN 2271-2097. Disponível em: <<https://articles/itmconf/abs/2018/02/itmconf-wcsn2018-03028/itmconf-wcsn2018-03028.html>>.

LI, X. et al. Optimizing Backup Resources in the Cloud. **CLOUD IEEE**, p. 790–797, 2017.

LOPEZ-PIRES, F.; BARAN, B. Virtual Machine Placement Literature Review. **arXiv:1506.01509 [cs]**, jun. 2015. ArXiv: 1506.01509. Disponível em: <<http://arxiv.org/abs/1506.01509>>.

MACHIDA, F.; KAWATO, M.; MAENO, Y. Redundant virtual machine placement for fault-tolerant consolidated server clusters. **2010 IEEE Network Operations and Management Symposium - NOMS 2010**, p. 32–39, abr. 2010.

MACIEL, P. R. M. Modeling Availability Impact in Cloud Computing. In: **Principles of Performance and Reliability Modeling and Evaluation**. Springer, 2016, (Series

in Reliability Engineering). p. 287–320. ISBN 978-3-319-30597-4 978-3-319-30599-8. Disponível em: <<https://link.springer.com/chapter/10.1007/978-3-319-30599-8:11>>.

MADNI, S. H. H. et al. Recent advancements in resource allocation techniques for cloud computing environment: a systematic review. **Cluster Computing**, dez. 2016. ISSN 1386-7857, 1573-7543. Disponível em: <<http://link.springer.com/10.1007/s10586-016-0684-4>>.

MALIK, A. et al. Cloudnetsim++: A gui based framework for modeling and simulation of data centers in omnet++. **IEEE Transactions On Services Computing**, n. 1, p. 1–1, 2015. ISSN 1939-1374.

MARSTON, S. et al. Cloud computing — The business perspective. **Decision Support Systems**, v. 51, n. 1, p. 176–189, 2011. ISSN 0167-9236. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167923610002393>>.

MARTELLO, S.; TOTH, P. Lower bounds and reduction procedures for the bin packing problem. **Discrete Applied Mathematics**, v. 28, n. 1, p. 59–70, jul. 1990. ISSN 0166-218X. Disponível em: <<http://www.sciencedirect.com/science/article/pii/0166218X9090094S>>.

MATLOOBI, R.; ZOMAYA, A. Y. Managing Performance Degradation of Collocated Virtual Machines in Private Cloud. **2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)**, p. 128–135, 2016.

MATOS, R.; et. al. Redundant Eucalyptus Private Clouds: Availability Modeling and Sensitivity Analysis. **Journal of Grid Computing**, v. 15, n. 1, p. 1–22, mar. 2017. ISSN 1570-7873, 1572-9184. Disponível em: <<https://link.springer.com/article/10.1007/s10723-016-9381-z>>.

MAZUMDAR, S.; PRANZO, M. Power efficient server consolidation for Cloud data center. **FGCS**, v. 70, p. 4–16, 2017. ISSN 0167-739X. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167739X16308093>>.

MEDINA, V.; GARCÍA, J. M. A Survey of Migration Mechanisms of Virtual Machines. **ACM Comput. Surv.**, v. 46, n. 3, p. 30:1–30:33, jan. 2014. ISSN 0360-0300. Disponível em: <<http://doi.acm.org/10.1145/2492705>>.

MELL, P. M.; GRANCE, T. **SP 800-145. The NIST Definition of Cloud Computing**. Gaithersburg, MD, United States, 2011.

MILANI, A. S.; NAVIMIPOUR, N. J. Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends. **Journal of Network and Computer Applications**, v. 71, p. 86–98, ago. 2016. ISSN 10848045. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S1084804516301217>>.

MILANI, B. A.; NAVIMIPOUR, N. J. A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions. **Journal of Network and Computer Applications**, v. 64, p. 229–238, abr. 2016. ISSN 1084-8045. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1084804516000795>>.

MONDAL, S.; MUPPALA, J.; MACHIDA, F. Virtual Machine Replication on Achieving Energy-Efficiency in a Cloud. **Electronics**, v. 5, n. 3, p. 37, jul. 2016. ISSN 2079-9292. Disponível em: <<http://www.mdpi.com/2079-9292/5/3/37>>.

MONIL, M. A. H.; RAHMAN, R. M. Vm consolidation approach based on heuristics, fuzzy logic, and migration control. **Journal Of Cloud Computing**, v. 5, n. 1, p. 8, 2016. ISSN 2192-113X.

MORENO-VOZMEDIANO, R.; et. al. Orchestrating the Deployment of High Availability Services on Multi-zone and Multi-cloud Scenarios. **Journal of Grid Computing**, p. 1–15, 2017. ISSN 1570-7873, 1572-9184. Disponível em: <<https://link.springer.com/article/10.1007/s10723-017-9417-z>>.

MUCHALSKI, F.; MAZIERO, C. Alocação de máquinas virtuais em ambientes de computação em nuvem considerando o compartilhamento de memória. **XII Workshop de Computação em Clouds e Aplicacções - WCGA 2014**, XII WCGA 2014, Florianópolis/SC–Brazil, p. 81–92, may 2014.

NANDURI, R.; KAKADIA, D.; VARMA, V. Energy and SLA aware VM Scheduling. **arXiv:1411.6114 [cs]**, nov. 2014. ArXiv: 1411.6114. Disponível em: <<http://arxiv.org/abs/1411.6114>>.

NATHUJI, R.; SCHWAN, K. VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems. In: **Proceedings of Twenty-first ACM SIGOPS Symposium on Operating Systems Principles**. New York–NY, USA: ACM, 2007. (SOSP '07), p. 265–278. ISBN 978-1-59593-591-5. Disponível em: <<http://doi.acm.org/10.1145/1294261.1294287>>.

NGENZI, A.; SELVARANI, R.; SUCHITHRA, R. Framework of Resource Management using Server Consolidation to Minimize Live Migration and Load Balancing. **Framework**, v. 7, n. 11, 2016. Disponível em: <[https://www.researchgate.net/profile/Alexander\\_Ngenzi/publication/311385728\\_Framework\\_of\\_Resource\\_Management\\_using\\_Server\\_Consolidation\\_to\\_Minimize\\_Live\\_Migration\\_and\\_Load\\_Balancing/links/5843c4e208aeda696815c64e.pdf](https://www.researchgate.net/profile/Alexander_Ngenzi/publication/311385728_Framework_of_Resource_Management_using_Server_Consolidation_to_Minimize_Live_Migration_and_Load_Balancing/links/5843c4e208aeda696815c64e.pdf)>.

PADOIN, E. L. **Energy-aware load balancing approaches to improve energy efficiency on HPC systems**. Tese (Doutorado) — UFRGS, Porto Alegre RS, 2016. Disponível em: <<http://www.lume.ufrgs.br/handle/10183/140401>>.

PATEL, S. D.; VAGHELA, D. A survey paper on load balancing algorithms for resource management in virtualization. **International Journal For Innovative Research In Science And Technology**, v. 2, n. 11, p. 68–70, 2016. Disponível em: <<http://www.ijirst.org/Article.php?manuscript=IJIRSTV2I11013>>.

PETROVIC, D.; SCHIPER, A. Implementing Virtual Machine Replication: A Case Study Using Xen and KVM. In: **Proceedings of the 2012 IEEE 26th International Conference on Advanced Information Networking and Applications**. Washington, DC, USA: IEEE Computer Society, 2012. (AINA '12), p. 73–80. ISBN 978-0-7695-4651-3. Disponível em: <<http://dx.doi.org/10.1109/AINA.2012.50>>.

PRATHIBA, S.; SOWVARNICA, S. Survey of failures and fault tolerance in cloud. **2017 2nd International Conference on Computing and Communications Technologies (ICCCT)**, p. 169–172, fev. 2017.

PROCACCIANTI, G. Energy-Efficient Software. 2015. Disponível em: <<https://iris.polito.it/handle/11583/2609157#.WpaonktrxqM>>.

PUCHER, A. et al. Using Trustworthy Simulation to Engineer Cloud Schedulers. **IEEE**, p. 256–265, mar. 2015. Disponível em: <<http://ieeexplore.ieee.org/document/7092927/>>.

RIECK, B. **Basic Analysis of Bin-Packing Heuristics**. Interdisciplinary Center for Scientific Computing. Heidelberg University, 2010. Disponível em: <[http://bastian.riECK.ru/uni/bin\\_packing/bin\\_packing.pdf](http://bastian.riECK.ru/uni/bin_packing/bin_packing.pdf)>.

RIEKSTIN, A. C. et al. A Survey on Metrics and Measurement Tools for Sustainable Distributed Cloud Networks. **IEEE Communications Surveys Tutorials**, PP, n. 99, p. 1–1, 2017.

ROSENDO, D.; et. al. How to Improve Cloud Services Availability? Investigating the Impact of Power and It Subsystems Failures. **HICSS-2018**, p. 10, jan. 2018. Disponível em: <<http://scholarspace.manoa.hawaii.edu/handle/10125/50080>>.

RUCK, D. et al. EAVIRA: Energy-Aware Virtual Infrastructure Reallocation Algorithm. **Proceedings SBESC 2017**, Cutiba/PR/Brazil, p. 14, nov 2017.

RÜCK, D. B. et al. Consolidação de MVs em nuvem IaaS orientada por operações elásticas e focada em consumo energético. **Workshop em Clouds e Aplicações (WCGA - SBRC)**, v. 15, n. 1/2017, 2017. ISSN 2595-2714. Disponível em: <<http://portaldeconteudo.sbc.org.br/index.php/wcga/article/view/2550>>.

SAHOO, C. N.; GOSWAMI, V. Cost and energy optimisation of cloud data centres through dual VM modes - activation and passivation. **IJCNDs**, v. 18, n. 3/4, p. 371, 2017. ISSN 1754-3916, 1754-3924. Disponível em: <<http://www.inderscience.com/link.php?id=10004670>>.

SANTOS, G. L.; et. al. Analyzing the IT subsystem failure impact on availability of cloud services. p. 717–723, jul. 2017.

SCARPINITI, M.; et. al. Energy performance of heuristics and meta-heuristics for real-time joint resource scaling and consolidation in virtualized networked data centers. **J Supercomput**, p. 1–38, jan. 2018. ISSN 0920-8542, 1573-0484. Disponível em: <<https://link.springer.com/article/10.1007/s11227-018-2244-6>>.

SECINTI, C.; OVATMAN, T. Fault Tolerant VM Consolidation for Energy-Efficient Cloud Environments. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, v. 10967 LNCS, p. 323–333, 2018.

SHARMA, Y. et al. Reliability and energy efficiency in cloud computing systems: Survey and taxonomy. **Journal of Network and Computer Applications**, v. 74, p. 66–85, out. 2016. ISSN 1084-8045. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1084804516301746>>.

SHARMA, Y. et al. Reliable and Energy Efficient Resource Provisioning and Allocation in Cloud Computing. In: **Proceedings of the 10th International Conference on Utility**

**and Cloud Computing**. New York, NY, USA: ACM, 2017. (UCC '17), p. 57–66. ISBN 978-1-4503-5149-2. Disponível em: <<http://doi.acm.org/10.1145/3147213.3147218>>.

SHEN, B.; et. al. High Availability for VM Placement and a Stochastic Model for Multiple Knapsack. **2017 26th International Conference on Computer Communication and Networks (ICCCN)**, p. 1–9, jul. 2017.

SHISHIDO, H. Y.; ESTRELLA, J. C. Bibliometric analysis of workflow scheduling in grids and clouds. **2017 36th International Conference of the Chilean Computer Science Society (SCCC)**, p. 1–9, 2017.

SILVA, M. P. d. Adaptive remus: replicação de máquinas virtuais xen com checkpointing adaptável. 2015. Disponível em: <<http://tede.udesc.br/handle/handle/2046>>.

SMIDT-DESTOMBES, K. S.; HEIJDEN, M. C. van der; HARTEN, A. van. On the availability of a k-out-of-N system given limited spares and repair capacity under a condition based maintenance strategy. **Reliability Engineering & System Safety**, v. 83, n. 3, p. 287–300, mar. 2004. ISSN 0951-8320. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S095183200300228X>>.

SMITH, J. E.; NAIR, R. The architecture of virtual machines. **Computer**, v. 38, n. 5, p. 32–38, maio 2005. ISSN 0018-9162. Disponível em: <<http://ieeexplore.ieee.org/document/1430629/>>.

SPINNEWYN, B.; et al. Cost-effective replica management in fault-tolerant cloud environments. **2017 13th International Conference on Network and Service Management (CNSM)**, p. 1–9, nov. 2017.

STAPELBERG, R. F. **Handbook of Reliability, Availability, Maintainability and Safety in Engineering Design**. USA: Springer Science & Business Media, 2009. ISBN 978-1-84800-175-6.

STROYAN, J.; BROWN, N. **Study on the contribution of standardization to innovation in European-funded research projects**. UK, 2013. 122 p. Disponível em: <<https://www.cencenelec.eu/standards/Education/JointWorkingGroup/Documents/Study-Contribution-Standardization-Innovation-Final2013.pdf>>.

SU, K.; et. al. Affinity and Conflict-Aware Placement of Virtual Machines in Heterogeneous Data Centers. In: . IEEE-ISADS, 2015. p. 289–294. ISBN 978-1-4799-8260-8 978-1-4799-8261-5. Disponível em: <<http://ieeexplore.ieee.org/document/7098274/>>.

SUH, D. et al. Toward Highly Available and Scalable Software Defined Networks for Service Providers. **IEEE Communications Magazine**, v. 55, n. 4, p. 100–107, abr. 2017. ISSN 0163-6804.

TAKOUNA, I.; MEINEL, C. Coordinating VMs' Memory Demand Heterogeneity and Memory DVFS for Energy-Efficient VMs Consolidation. **2014 IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom)**, p. 478–485, set. 2014.



TCHANA, A.; BROTO, L.; HAGIMONT, D. Approaches to cloud computing fault tolerance. In: **2012 International Conference on Computer, Information and Telecommunication Systems (CITS)**. IEEE, 2012. p. 1–6. ISBN 978-1-4673-1550-0 978-1-4673-1549-4 978-1-4673-1548-7. Disponível em: <<http://ieeexplore.ieee.org/document/6220386/>>.

TENG, F. et al. Energy efficiency of VM consolidation in IaaS clouds. **The Journal of Supercomputing**, v. 73, n. 2, fev. 2017. ISSN 0920-8542, 1573-0484. Disponível em: <<http://link.springer.com/10.1007/s11227-016-1797-5>>.

TSO, F. P.; JOUET, S.; PEZAROS, D. P. Network and server resource management strategies for data centre infrastructures: A survey. **Computer Networks**, set. 2016. ISSN 1389-1286. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1389128616302298>>.

UNUVAR, M.; et al. A Predictive Method for Identifying Optimum Cloud Availability Zones. **2014 IEEE 7th International Conference on Cloud Computing**, p. 72–79, jun 2014.

USMANI, Z.; SINGH, S. A Survey of Virtual Machine Placement Techniques in a Cloud Data Center. **Procedia Computer Science**, v. 78, p. 491–498, jan. 2016. ISSN 1877-0509. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1877050916000958>>.

VARASTEY, A.; et al. On Reliability-Aware Server Consolidation in Cloud Datacenters. **ISPD**, p. 95–101, set. 2017. ArXiv: 1709.00411. Disponível em: <<http://arxiv.org/abs/1709.00411>>.

Jun Wang, Jian Zhang, Yixing JIA, Maosheng REN e Wenwu PENG. **System and methods to create virtual machines with affinity rules and services asymmetry**. 2018. US20180046481A1. Disponível em: <<https://patents.google.com/patent/US20180046481A1/en>>.

WANG, Y.; LI, X. Achieve high availability about point-single failures in OpenStack. **2015 4th International Conference on Computer Science and Network Technology (ICCSNT)**, v. 01, p. 45–48, 2015.

WANG, Y. et al. Improving utilization through dynamic VM resource allocation in hybrid cloud environment. **2014 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)**, p. 241–248, dez. 2014.

WEIBULL. Reliability basics: Availability and the different ways to calculate it. **Weibull: Reliability HotWire**, v. 79, 2007. Disponível em: <<http://www.weibull.com/hotwire/issue79/relbasics79.htm>>.

WOLSKI, R.; BREVIK, J. Using Parametric Models to Represent Private Cloud Workloads. **IEEE-TSC**, v. 7, n. 4, p. 714–725, out. 2014. ISSN 1939-1374.

WU, C.; BUYYA, R. **Cloud Data Centers and Cost Modeling: A Complete Guide To Planning, Designing and Building a Cloud Data Center**. Massachusetts, EUA: Morgan Kaufmann, 2015. ISBN 978-0-12-801688-6.

WU, X. Combination Replicas Placements Strategy for Data sets from Cost-effective View in the Cloud. **International Journal of Computational Intelligence Systems**, v. 10, n. 1, p. 521–539, jan. 2017. ISSN 1875-6883. Disponível em: <<https://www.atlantispress.com/journals/ijcis/25866768>>.

XIONG, H.; FOWLEY, F.; PAHL, C. A database-specific pattern for multi-cloud high availability and disaster recovery. **Communications in Computer and Information Science**, v. 567, p. 374–388, 2016.

XU, M.; TIAN, W.; BUYYA, R. A survey on load balancing algorithms for vm placement in cloud computing. **2016 (cs.DC)**, 2016. Disponível em: <<http://arxiv.org/abs/1607.06269>>.

YANG, C. T. et al. On Improvement of Cloud Virtual Machine Availability with Virtualization Fault Tolerance Mechanism. **2011 IEEE Third International Conference on Cloud Computing Technology and Science**, p. 122–129, nov. 2011.

YAO, Z.; PAPAPANAGIOTOU, I. A trace-driven evaluation of cloud computing schedulers for IaaS. In: . **IEEE-ICC**, 2017. p. 1–6. ISBN 978-1-4673-8999-0. Disponível em: <<http://ieeexplore.ieee.org/document/7996822/>>.

YEOW, W.-L.; WESTPHAL, C.; KOZAT, U. Designing and embedding reliable virtual infrastructures. In: **'10 - ACM SIGCOMM**. ACM Press, 2010. p. 33. ISBN 978-1-4503-0199-2. Disponível em: <<http://portal.acm.org/citation.cfm?doid=1851399.1851406>>.

YOU, X. et al. A survey and taxonomy of energy efficiency relevant surveys in cloud-related environments. v. 5, p. 14066–14078, 2017.

ZELKOWITZ, M. **Advances in Computers**. USA: Academic Press, 2011. ISBN 978-0-12-385513-8.

ZHANG, Q. et al. A column generation-based algorithm for two-stage, two-dimensional bin packing problem with a variant variable sized constraint. **2017 36th Chinese Control Conference (CCC)**, p. 2841–2845, jul. 2017.

ZHANG, X.; GADDAM, S.; CHRONOPOULOS, A. T. Ceph Distributed File System Benchmarks on an Openstack Cloud. **2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)**, p. 113–120, nov. 2015.

ZHAO, L. et al. Online Virtual Machine Placement for Increasing Cloud Provider's Revenue. **IEEE Transactions on Services Computing**, v. 10, n. 2, p. 273–285, mar. 2017. ISSN 1939-1374. Disponível em: <<http://ieeexplore.ieee.org/document/7128735/>>.

ZHOU, A.; SUN, Q.; LI, J. Enhancing reliability via checkpointing in cloud computing systems. **China Communications**, v. 14, n. 7, p. 1–10, jul. 2017. ISSN 1673-5447.