CHAVE: Resource Consolidation with High Availability on Virtualized Environments

Daniel S. Camargo
Pós-Graduação em Computação Aplicada
Universidade do Estado de Santa Catarina
E-mail: daniel@colmeia.udesc.br

Maurício A. Pillon
Pós-Graduação em Computação Aplicada
Universidade do Estado de Santa Catarina
E-mail: mauricio.pillon@udesc.br

Charles C. Miers
Pós-Graduação em Computação Aplicada
Universidade do Estado de Santa Catarina
E-mail: charles.miers@udesc.br

Guilherme P. Koslovski
Pós-Graduação em Computação Aplicada
Universidade do Estado de Santa Catarina
E-mail: guilherme.koslovski@udesc.br

Rodrigo R. Righi
Pós-Graduação em Computação Aplicada
Universidade do Vale do Rio dos Sinos
E-mail: rrrighi@unisinos.br

***Abstract*—In order to meet the growing demand for business continuity, the adoption of cloud computing platforms is growing to keep its critical services. However, availability rates established in service level agreements (SLA) by cloud service providers (CSP) does not always meet their demand for high-availability (HA). Services replicated in multi-AZs architecture result in high costs due to the inherent increase in the load of the physical servers, resulting in higher consumption of energy. In this way, virtual machine (VM) consolidation stands out as an energy efficiency strategy based on virtual resource scheduling, allowing to reduce energy consumption as well as improve the organization of fragmented resources. However, when consolidation is applied in conjunction with HA mechanism, there is a risk of violating affinity (MV-server) and anti-affinity (MV-AZ) constraints, thereby violating SLA requirements. Thus, CHAVE presents an on-demand HA mechanism based on Multi-AZ replication, which simultaneously performs a VM consolidation strategy isolated for each AZ, considering its inherent constraints. The numerical results by real-trace driven simulations show that CHAVE, meets 20% of HA requests with energy consumption similar to a CSP that does not apply to consolidation with replication. Additionally, CHAVE does not cause any SLA violations such as overcommiting, or rejection of critical requests.**

***Keywords*—Cloud Computing, High-Availability, VM Consolidation**

I. Introdução

A computação em nuvem tem se tornado uma base fundamental para organizações que dependem da continuidade de negócios e serviços críticos, uma vez que a interrupção destes serviços pode causar prejuízos financeiros e de reputação [1]. Neste sentido, o paradigma de computação em nuvem destaca-se por fornecer recursos computacionais sob demanda [2], com garantias de disponibilidade estabelecidas em acordo de nível de serviço (SLA) entre o Provedor de Serviços em Nuvem (CSP) e seus inquilinos. Entretanto, em alguns casos, as taxas de disponibilidade não correspondem às expectativas de serviços críticos, nesse caso a ocorrência de falhas na ordem de segundos podem ter consequências irrecuperáveis [3]. Este fato vem ganhando notoriedade entre pesquisadores [4] e indústria, que buscam tratar tanto a causa, quanto a resolução das falhas [5]. Pesquisas [4] estimam que o impacto financeiro destas interrupções alcance prejuízos de centenas de milhares de dólares por hora, para o CSPs e seus inquilinos.

CSPs que possibilitam arquitetar serviços de alta disponibilidade (HA), normalmente utilizam mecanismos baseados em tolerância à falhas, redundância e replicação. Assim, o presente trabalho baseia-se no mecanismo de replicação de máquinas virtuais (MVs) aliado ao conceito de HA sob demanda, utilizando conceitos de diagrama de bloco de confiabilidade (RBD) e probabilidade de eventos independentes através de uma arquitetura distribuída em múltiplas zonas de disponibilidade (AZs). Todavia, mecanismos de HA inerentemente implicam em aumento nos custos operacionais do CSP [6]. Este custo está relacionado ao aumento na demanda por mais equipamentos, elevando o custo total de propriedade (TCO), que por conseguinte, impacta no aumento do consumo de energia, recurso que corresponde a até 50% dos custos operacionais do CSP. Desse modo, quanto maior a taxa de HA, maior o número de réplicas e maior a demanda por energia elétrica. Neste sentido, a consolidação de MVs destaca-se como estratégia amplamente utilizada na literatura, na redução do consumo de energia alocando as MVs no menor número de servidores, desativando equipamentos ociosos [7]. Além de reduzir o consumo de energia, possibilita manter o uso eficiente de recursos de CPU e memória, evitando a fragmentação destes recursos em sistemas distribuídos.

A solução CHAVE (*Resource Consolidation with High-Availability on Virtualized Environments*) visa fornecer um mecanismo de HA e, simultaneamente, reduzir o consumo de energia causado pelo seu incremento de carga. A principal contribuição do presente trabalho está em conciliar o conflito entre replicação e consolidação ao considerar suas restrições de afinidade e antiafinidade, apresentando uma solução baseada em arquitetura da infraestrutura de nuvem. Adicionalmente, é apresentado o conceito de HA sob demanda, com uma equação que permite definir o número de MVs réplicas com base na necessidade do inquilino e na taxa de disponibilidade do CSP. Por fim, como CHAVE parte da perspectiva do CSP, pode ser colocado em prática por nuvens públicas ou privadas, tanto na questão arquitetural quanto com os algoritmos propostos individualmente para HA e consolidação.

O presente artigo está organizado da seguinte forma. A Seção II aborda a motivação e fundamentação teórica utilizada no decorrer do artigo, para auxiliar na definição da solução proposta, desenvolvida na Seção III. O plano de testes é descrito na Seção IV, com a discussão dos resultados na Seção V. O levantamento e discussão dos trabalhos correlatos são desenvolvidos na Seção VI, encerrando com as considerações finais e trabalhos futuros na Seção VII.

978-1-5386-9233-2/18/$31.00

II. REVISÃO DE LITERATURA

Mecanismos de tolerância à falhas objetivam fornecer HA para serviços críticos, e normalmente baseiam-se em redundância e/ou replicação [4]. Especificamente para replicação de MVs, o tempo de recuperação da falha é reduzido, pois consiste na comutação entre a MV falhada para uma de suas réplicas. A literatura especializada define alta disponibilidade (HA) como a capacidade de um sistema fornecer serviços quando requisitado, em qualquer ponto no tempo e durante um período de tempo determinado [8]. Adicionalmente, HA é descrita como uma taxa de disponibilidade igual ou superior a cinco noves $(0,99999)$ [9]. Na prática, a relação entre as taxas de disponibilidade e HA é definida conforme a capacidade do CSP em manter o SLA. Tipicamente, CSPs estabelecem uma taxa de disponibilidade para cada uma de suas AZs. Todavia, estabelecer a taxa ideal de HA para serviços críticos, depende da demanda dos inquilinos ao observar o grau de criticidade desses serviços [4], *e.g.*, determinar quanto segundos de interrupção um sistema pode ter ao ano.

AZs são constituídas por equipamentos (suporte e computacionais) e suprimentos (energia e rede) isolados das demais. Isso permite que uma AZ não seja afetada na ocorrência de falhas nas outras AZs, permitindo que, mesmo em caso de uma catástrofe, as demais AZs continuem operacionais. Esta característica habilita desenvolver arquiteturas de serviços *stateless*, em que as aplicações multicamadas são replicadas em MVs efêmeras e escalonadas por balanceadores de carga. Os serviços *stateful* dependem de estados transientes de memória [10], e demandam mecanismos de replicação mais complexos.

A análise por RBD é amplamente adotada para calcular a disponibilidade de sistemas complexos [5], considerando componentes (AZs) em série/paralelo. Quando componentes estão em série, *i.e.*, no mesmo ponto único de falhas (SPoF), a disponibilidade combinada é o produto da disponibilidade '$\mathbb{A}$' de seus P componentes, que sempre resulta em uma disponibilidade menor do que a de seus componentes individuais e, portanto, não aplicado para HA. Para componentes em paralelo, a disponibilidade combinada é o produto da indisponibilidade $(1-\mathbb{A})$ dos seus P componentes. Considera-se, que quando os P componentes não compartilham o mesmo SPoF, então não haverá dependência entre as falhas. Este é um axioma proveniente da probabilidade de eventos independentes, expresso pela Equação 1. Assim, a taxa $\mathbb{HA}$ é sempre maior do que a disponibilidade média das AZs ($\mathbb{A}_{(AZ_i)}$, para $i \in P$ componentes). Para garantir que as MVs do *pool* de replicação não sejam instanciadas no mesmo SPoF, é estabelecida a restrição de antiafinidade ($\text{MV}_{critica}$-$\text{MV}_{replica}$) [9], definida pelo presente trabalho.

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

De modo geral, sistemas tolerantes à falhas causam uma sobrecarga na execução do sistema [11]. Para reduzir o consumo de energia e melhorar a utilização de recursos em Data Centers (DCs), a consolidação de MVs visa alocar todas as MVs na menor quantidade possível de servidores, desativando os servidores ociosos. Além de reduzir o consumo de energia, a consolidação reorganiza os espaços contíguos e evita a fragmentação de recursos [12]. A consolidação de MVs, quando aplicada em servidores homogêneos, é um exemplo prático do problema do empacotamento (*Bin Packing*), de complexidade NP-Difícil. Em *benchmarks* específicos para o Problema do Bin Packing (PBP), são comparadas heurísticas que variam entre si na aproximação do valor ótimo (OPT) e no tempo de execução. Entre estas opções, o *First Fit Decreasing* (FFD) destaca-se aproximar-se do OPT em tempo de execução aceitável, sendo $FFD(L) \leq (11/9) * OPT(L) + c$, sendo que c é uma constante [13].

A consolidação pode ocorrer em dois momentos distintos: ao instanciar uma MV (posicionamento inicial), ou através da migração das MVs após instanciadas (re-otimização) [14]. Aplicar apenas o posicionamento inicial dificilmente resultará uma boa aproximação do ótimo, pois a frequência das desalocações pode ser diferente das alocações em um dado momento. Por outro lado, a re-otimização deve considerar a sobrecarga gerada pelas migrações [15], além da restrição de afinidade MV-servidor, que impossibilita algumas MVs de serem migradas. Como uma restrição à consolidação, a literatura estabelece a política de afinidade (MV-servidor) [16], que consiste em instanciar uma determinada MV em um servidor e esta MV não deve ser migrada durante sua execução. Como a afinidade desabilita a migração da MV, impossibilita que, quando subutilizado, o seu servidor seja desativado [9]. Adicionalmente, ocorre a restrição de afinidade MV-AZ, quanto o inquilino especifica que uma MV deve ser executada apenas em uma determinada AZ, não podendo ser migrada. Violar as políticas de afinidade ou antiafinidade implica em violar o SLA, pois são requisitos especificados pelo inquilino.

É conflitante realizar a consolidação de MVs concomitante a um mecanismo de HA baseado em replicação, devido a possibilidade de violar as restrições de afinidade e antiafinidade [16]. Ainda que o escalonador da AZ evite a justaposição do pool de replicação no mesmo servidor, restrições na estratégia de consolidação podem impedir de obter resultados próximos do ótimo. Isto se deve pelo fato de adicionar-se a restrição de afinidade, a já existente, complexidade da consolidação. Na prática, plataformas de nuvem públicas como a *Amazon Web Services* (AWS) e *Google Cloud Platform* (GCP) são constituídas por regiões, comumente cada região possui entre duas a três AZs. Assim, quando aplicada a consolidação de MVs, elimina-se o risco de violar a antiafinidade, por conseguinte violar o SLA. Portanto, a justificativa do trabalho é fornecer HA sob demanda, com base em arquitetura Multi-AZ, sem violar as restrições de afinidade e antiafinidade, permitindo que a consolidação de MVs reduza a carga gerada pela replicação.

III. CHAVE

A solução proposta denominada CHAVE visa fornecer um mecanismo de HA sob demanda e simultaneamente reduzir seu impacto de custo, aplicando a estratégia de consolidação de 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 SLA firmado com o inquilino, CHAVE baseia-se em conceitos de arquitetura Multi-AZ, que atualmente constituem um padrão amplamente adotado nas infraestruturas de CSPs. Desse modo, CHAVE consiste em duas etapas que

são executadas em níveis diferentes dessa arquitetura: (i) a replicação de MVs ocorre entre as AZs de uma mesma região, enquanto (ii) a consolidação ocorre internamente a cada AZ. São utilizadas algumas equações e definições formais para a descrição do CHAVE, listadas na Tabela I.

Tabela I. TABELA DE NOTAÇÕES E EQUAÇÕES UTILIZADAS.

Conjuntos de recursos	
$\mathcal{V}'_n = \mathcal{V}^s_i \bigcup \mathcal{V}^c_i \bigcup \mathcal{V}^r_i = (v'_1, v'_2, \ldots, v'_n)$	Conjunto de todas as n MVs composta por regulares ($\mathcal{V}^s_i$), críticas ($\mathcal{V}^c_i$) réplicas ($\mathcal{V}^r_i$).
$\mathcal{P}_n = v^c_i \bigcup \mathcal{V}^r_i \mid i \in P$	*Pool* de replicação composta por uma v^c_i e $\mathcal{V}^r_i$.
$\mathcal{H}_m$	Conjunto de m servidores de uma AZ.
$\mathbb{HA}$	Taxa de alta disponibilidade
$\mathbb{A}$	Taxa de disponibilidade da AZ
Equações	
$\mathcal{R} = \text{ceil}\left(\frac{log_{10}(1-\mathbb{HA})}{log_{10}(1-\mathbb{A})} - 1\right)$ (2)	Quantidade de réplicas para fornecer a taxa de HA.
$\mathcal{U} = (m * \sum_{i=0}^{m} \mathcal{H}_i(\#CPU))^{-1}$ (3)	Carga unitária, 1 unidade de CPU em relação à AZ.
$\mathcal{F} = \sum_{i=0}^{m} \frac{\sum_{j=0}^{n} \mathcal{H}_i(CPU\_idle_j)}{m * \mathcal{H}_i(\#CPU)}$ (4)	Fragmentação de recursos de CPU da AZ.
$\mathcal{F}_{min} = m^{-1}$ (5)	Fragmentação mínima, equivale a 1 servidor da AZ.
$\mathcal{C} = \frac{\sum_{j=0}^{n} vCPU_j}{\sum_{i=0}^{m} \mathcal{H}_i(\#CPU)}$ (6)	Carga da AZ, sendo m a quantidade de servidores e n o total de MVs instanciadas na AZ.
$\mathcal{E} = \sum_{i=0}^{m} \sum_{j=0}^{n} Consumo(MV_j)$ (7)	Consumo total de uma AZ, m é a quantidade de servidores e n as MVs de cada servidor.
$\mathcal{E}_r = 1 - \frac{\mathcal{E}_{final}}{\mathcal{E}_{inicial}}$ (8)	Redução do consumo em relação à medição anterior.

A. Alta disponibilidade para máquinas virtuais

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 antiafinidade ($\text{MV}_{critica}$-$\text{MV}_{replica}$) entre os elementos do *pool*. Esta abordagem é embasada na análise por RBD e fundamentada pela probabilidade de eventos independentes (Equação 1). Normalmente, a arquitetura da infraestrutura de CSPs é constituída por regiões geograficamente distribuídas, e cada região é composta por duas ou mais AZs independentes, mas interconectadas por enlaces de rede de baixa latência. Esta característica permite que a replicação entre as AZs de cada região ocorra sem causar os transtornos inerentes a redes de longa distância (WAN).

Através de SLA, o inquilino especifica qual a região e em qual AZ será instanciada cada uma de suas MVs. Adicionalmente, é requisitado uma taxa de HA para executarem os serviços críticos, então para cada MV crítica pode ser estabelecida uma taxa de HA específica. Nesse sentido, uma das características do CHAVE é concepção de HA sob demanda, que objetiva atender as necessidades específicas de cada inquilino. Assim, o algoritmo responsável define a quantidade ideal de R MVs réplicas que são instanciadas em diferentes AZs da mesma região. Para HA sob demanda, a Equação 2 especifica o número de réplicas necessárias para assegurar a taxa $\mathbb{HA}$ requerida. Esta equação é uma reformulação da Equação 1, em que é isolado o índice P do produtório, para obter a quantidade de réplicas R. Também é fornecida a menor taxa de disponibilidade $\mathbb{A}$ entre todas as AZs e a taxa $\mathbb{HA}$ requisitada pelo inquilino. Como o valor de P e R devem ser discretos, utiliza-se a função *ceil*, que corresponde ao menor valor inteiro maior ou igual a R. O número de réplicas R é limitado pela quantidade máxima de AZs (índice P) de cada região, que também é um limitante para o tamanho do *pool* de replicação. Da relação entre P e R, decorre que P é o número de elementos do *pool* de replicação, que inclui o próprio componente crítico, enquanto R é o número de réplicas, *i.e.*, $R = P - 1$. Assim, ao aplicar o exemplo prático da Figura 1, é definido que a MV crítica v^c_4 deve ter $\mathbb{HA}$=99,999%, em um CSP que estabelece por padrão, uma disponibilidade $\mathbb{A}$=99,5% para todas as suas AZs.

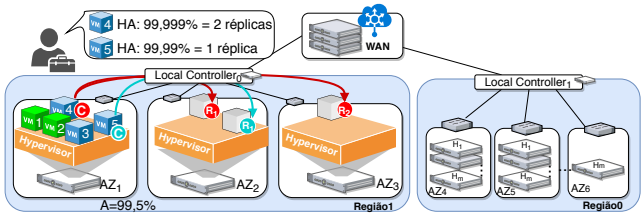


Figura 1. Visão geral da replicação em arquiteturas Multi-AZ: inquilino requisita $\mathbb{HA} = 99,999\%$ e CHAVE replica o serviço para mais duas réplicas.

Assim, aplicando a Equação 2, tem-se que $R = \frac{log_{10}(1-0,99999)}{log_{10}(1-0,995)} - 1 = 1,17$, e quando aplicado $ceil(R) = 2$ resulta em duas réplicas. A restrição de antiafinidade consiste em considerar um *pool* de replicação $\mathcal{P}_i$ com a MV crítica v^c e suas réplicas v^r_1 e v^r_2, 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 P AZs, esta abordagem de distribuição de réplicas restringe alocar quaisquer elementos de $\mathcal{P}$ no mesmo SPoF.

B. 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. Nesse sentido, CHAVE é energeticamente eficiente 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. O custo do procedimento da etapa de posicionamento inicial é o mesmo de qualquer outro algoritmo de escalonamento, pois basta selecionar o primeiro (*First-Fit*). A etapa das migrações ocorre periodicamente, sendo acionada por um gatilho de fragmentação da AZ, descrito na Equação 4. 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. A duas etapas são essencialmente distintas e independentes, podendo as migrações ocorrerem antes ou depois do posicionamento inicial. Porém, para a execução dos testes (Seção IV), observou-se

que a fragmentação de recursos ocorre sempre após atender as requisições de alocação e desalocação de MVs.

A função de fragmentação de recursos da AZ verifica a quantidade de CPUs ociosas (CPU_idle) entre os servidores ativos, de tal modo que o somatório dessa fragmentação corresponda ao montante 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 5), 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 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 possui 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). 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 inquilino. 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 os servidores, os quais estão instanciadas, de serem desativados. 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.

IV. Estudo de caso

CHAVE é avaliado através de análise numérica baseada em simulação. O estudo de caso define os requisitos para que a simulação possa ter maior proximidade a uma infraestrutura de nuvem real, e abranger as funcionalidades do CHAVE. É utilizado o simulador EAVIRA[1], anteriormente aplicado em pesquisas relacionadas ao escalonamento de Infraestruturas Virtuais (IVs) no contexto de computação em nuvem Infraestrutura como Serviço (IaaS) [17]. 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 são implementações do EAVIRA. Mantém-se o modelo preexistente de consumo de energia dos servidores, que é guiado pelo modelo de compartilhamento proporcional [18]. Porém, são atualizados os valores da escala de consumo de servidores, para suportar até 32 núcleos, estendendo o limite superior para 209.1 W. O algoritmo do simulador é implementado em linguagem Python (portado para versão 3.6). Os testes foram executados em uma MV configurada com 16 vCPUs e 32 GB de memória RAM, sistema operacional Debian 9, hospedada na Nuvem Privada Tchê[2].

A. Descrição da entrada de dados: traces reais

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. Em pesquisa não-exaustiva na literatura especializada, destacam-se os *traces* da plataforma Eucalyptus[3], por serem avaliados em outros trabalhos científicos para análise de distribuição de dados [19] e em estratégias de escalonamento e consolidação. Estes *traces* consistem em seis conjuntos de dados (AZs distintas), capturados por *logs* de organizações privadas que utilizam a plataforma Eucalyptus durante o ano de 2014. Cada requisição refere-se a uma alocação ou desalocação (START/STOP), 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 nem quantidade de vRAM, considera-se que todas as MVs têm taxa de utilização de vCPU a 100% e, vRAM proporcional à quantidade de vCPUs, na escala de 1:4 (4GB de RAM para cada vCPU). Além disso, uma vez reservados, estes recursos não podem ser compartilhados entre MVs, pois não é considerado *overcommitting*. Cada AZ é constituída por servidores homogêneos quanto ao número de CPUs.

As principais características de cada *trace* constam na Tabela II, indicando a quantidade de operações, número de servidores e quantos núcleos possui cada servidor.

[1]Disponível em: https://github.com/LabP2D/EAVIRA_CHAVE.

[2]Disponível em: http://labp2d.joinville.udesc.br/

[3]Disponíveis em https://www.cs.ucsb.edu/~rich/workload/

Tabela II. INFORMAÇÕES E CONFIGURAÇÃO DOS TRACES REAIS DO EUCALYPTUS POR AZ.

	AZ	#Op.	$\#\mathcal{H}$	#CPU	$max(P)$	$\mathcal{C}$	$2 * \mathcal{U}$
R_0	AZ1	18346	13	24	15,1%	25,3%	0,641%
	AZ2	1800	7	12	12%	28,5%	2,381%
	AZ3	5200	7	8	7,8%	3,5%	3,571%
R_1	AZ4	2872	12	8	18%	8,3%	2,083%
	AZ5	16912	31	32	4,7%	71,4%	0,202%
	AZ6	8314	31	32	7,8%	64,6%	0,195%

As colunas "$max(P)$" e "$\mathcal{C}$" resumem as probabilidades e respectivas cargas definidas no plano de testes (Seção IV-B), usadas como base para selecionar os *snapshots* analisados na Seção V-A. A coluna "$2 * \mathcal{U}$" refere-se a carga mínima para que possa ocorrer uma consolidação (Eq. 3), utilizada como critério de corretude ao selecionar cada instante. 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 sintetizados com base em uma 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 $\mathbb{HA}$ 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 infraestrutura de nuvem simulada, segue a mesma estrutura definida na Fig 1, com três AZs para duas regiões. Desse modo, as requisições de replicação de qualquer AZ de R_0, serão instanciadas apenas nas AZs desta região. Os algoritmos implementados no *Local Controller* da região, selecionam a AZ mais propícia a receber a réplica através de balanceamento de carga, *i.e.*, seleciona a que possui a menor carga, conforme a Eq. 6. Para sincronizar a execução simultânea de todas as AZs, é utilizado o conceito de *Global Virtual Time* (GVT) síncrono. Para contabilizar corretamente a métrica de consumo de energia em Wh, é considerada que a unidade de tempo seja determinada em horas, evitando valores contínuos.

B. Plano de testes

Como plano de testes, definiu-se nove cenários, organizados em três conjuntos, conforme ilustrado na Figura 2.

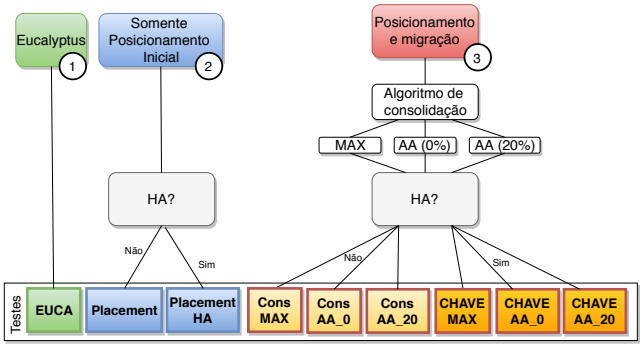


Figura 2. Estrutura de testes: (1) Eucalyptus, (2) somente posicionamento inicial, e (3) consolidação em MAX, AA0 e AA20. (2) e (3) com e sem HA.

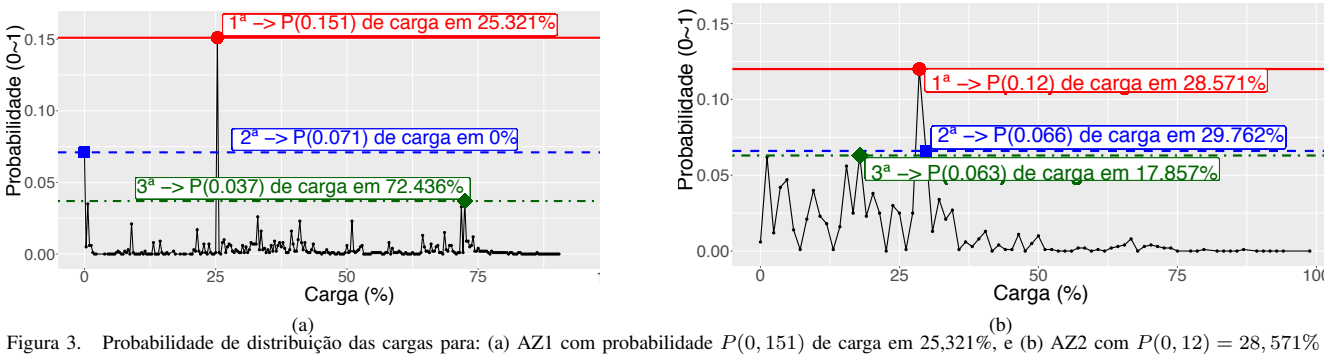
O primeiro cenário consiste em estabelecer uma base de comparação para os demais. É executado o algoritmo padrão de escalonamento do Eucalyptus (*Round Robin* e *Greedy*), alocando e desalocando as requisições conforme estabelecido nos *traces*, permitindo avaliar a distribuição de carga de cada AZ. No segundo conjunto, são aplicados os dois cenários que executam apenas a etapa do posicionamento inicial, *i.e.*, sem a migração, diferenciando-se entre ter ou não HA. No terceiro conjunto, adiciona-se aos cenários, a funcionalidade de migração, que constitui o processo completo de consolidação, com os três algoritmos de consolidação do CHAVE: MAX, AA0 (0% de afinidade) e AA20 (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(20%) com base em uma demanda real informada pelo GCP, e o teste com 0% de restrições é utilizado na tentativa de minimizar a quantidade de migrações.

Para justificar a seleção das amostras usadas nas Seção V-A, inicialmente é realizada uma análise de probabilidade de ocorrência destas amostras ($P(x)$) em relação a carga da AZ, utilizando-se de inferência estatística. São definidos os histogramas referentes a distribuição de carga em cada AZ, 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 tem maiores probabilidades de ocorrerem. Dentre as AZs utilizadas, selecionou-se, para a discussão dos resultados, as AZ1 e AZ2. O comportamento das demais AZs assemelham-se a uma das AZs selecionadas.

Quanto a aplicação da consolidação nas AZs, a mesma só é acionada, se houver alguma possibilidade de migração de MV. Desse modo, a menor carga possível para cada AZ é estabelecida na Tabela II, na coluna '$2 * \mathcal{U}$', que indica a carga equivalente a duas vezes o valor mínimo (uma MV em cada servidor). A probabilidade de frequência de carga pode ser obtida com análise das Figuras 3-(a) e 3-(b), com seu eixo y, indicando probabilidade de ocorrência, e eixo x, carga. As linhas horizontais (vermelho, azul e magenta) destacam as três cargas com maiores probabilidades de ocorrência. Após a análise dos dados referentes ao caso específico de execução dos algoritmos em um instante, são avaliados os dados referentes a uma perspectiva global dos cenários, que refere-se a execução completa dos *traces*.

V. ANÁLISE DOS RESULTADOS

Os resultados são analisados sob duas perspectivas: local e global. A perspectiva local refere-se a análise dos cenários em um caso específico de execução, *i.e.*, os *snapshots* que ocorrem em uma determinada unidade de tempo do GVT. Um *snapshot* consiste em um momento no qual o gatilho de ativação do CHAVE é acionado, sendo que esse momento e as suas consequências são analisados em cada cenário. São apresentados os resultados referentes aos *snapshots* previamente selecionados na análise de probabilidade de distribuição de carga com cinco métricas. A perspectiva global consiste na análise dos cenários referentes a execução de toda a simulação, compreendendo seis cenários relacionados às seis métricas.



(a) (b)

Figura 3. Probabilidade de distribuição das cargas para: (a) AZ1 com probabilidade $P(0,151)$ de carga em 25,321%, e (b) AZ2 com $P(0,12) = 28,571\%$

Mantém-se sucinta a apresentação dos dados com a seleção das AZs AZ1 e AZ2 pelos mesmos critérios. Para comparação entre os cenários, considerando métricas distintas com dados multivariados, os resultados são representados por gráficos de radar, com as métricas normalizadas entre 0% a 100%, com base no valor mínimo e máximo encontrado por métrica. Os eixos representam as métricas e os polígonos são os cenários. Nesse contexto, o cenário com melhor desempenho no conjunto de métricas é aquele que tem o polígono com a menor área em torno do centro do gráfico. O comportamento da AZ é avaliado em seis cenários com algoritmos distintos de consolidação (AA0, AA20 e MAX) e, ativação ou não de HA (CHA_ e C_), para cada algoritmo de consolidação.

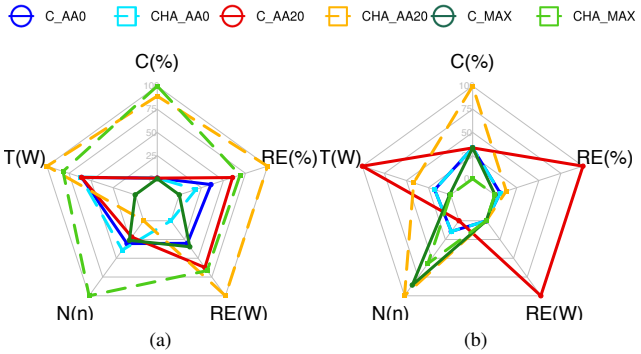
A. *Snapshots da consolidação*

Com o intuito de avaliar a atuação do CHAVE, definiu-se cinco métricas, suas respectivas unidades de medidas e a interpretação dos valores:

- $T(W)$: potência consumida pela AZ (Eq. 7). Por referir-se a um *snapshot* de tempo, ponta de prova, então a unidade é dada pela potência em Watts (W). O menor valor é melhor;
- $RE(W)$: redução do consumo de energia ($T_{final} - T_{inicial}$). O maior valor é melhor;
- $RE(\%)$: percentual de redução do consumo de energia (Eq. 8), aplicada para mensurar mudanças de estado, *i.e.*, antes e depois de ocorrer uma consolidação. O maior valor é melhor;
- $C(\%)$: percentual de carga de uso (Eq. 6), refere-se a taxa de utilização dos recursos de CPU de todos os servidores de cada AZ. O menor valor é melhor; e
- $M(n)$: número de migrações, refere-se a quantidade de migrações para a consolidação. O menor valor é melhor.

A melhor solução ocorre quando todas as métricas tendem aos valores mínimos, resultando no menor polígono centralizado. As métricas de redução de consumo em % e em W devem ser interpretadas de forma inversa para encontrar o menor polígono, pois o maior valor é melhor.

Para os cenários aplicados à AZ1 (Figura 4-(a)) observa-se que o teste C_MAX possui o menor polígono, portanto, melhores resultados. Naturalmente, aplicar a consolidação máxima, sem a carga adicional da HA, resulta na obtenção dos melhores valores. Porém, o comportamento esperado para melhor resultado não se confirmou na AZ2 (Figura 4-(b)).



(a) (b)

Figura 4. Cenários nos *snapshots* para (a) AZ1 e (b) AZ2.

A distribuição de carga solicitada a esta AZ2 não permite a maximização dos recursos, possibilitando a aplicação de HA, sem custo adicional. Observa-se que o cenário MAX do CHAVE obteve melhor desempenho que cenários sem HA.

B. *Perspectiva global dos testes*

Em perspectiva global, são utilizadas seis métricas que contabilizam o total de cada execução.

- $T(Wh)$: total de energia consumida, indica o quanto a AZ consumiu durante sua execução. O menor valor é melhor;
- $RE(Wh)$: total de redução de consumo com as consolidações, representa o somatório das reduções de cada ação de consolidação. O maior valor é melhor;
- $SLAV(n)$: contabiliza o total de violações de SLA, especificamente as rejeições ao instanciar novas MVs (regular, crítica ou réplica) devido a sobrecarga (100% de uso) da AZ. O menor valor é melhor;
- $M(n)$: número total de migrações, representa o somatório de todas as migrações realizadas. O menor valor é melhor;
- $FP(n)$: total de falso-positivos contabiliza os casos em que são acionados os gatilhos para realizar uma consolidação, mas nenhum servidor é desativado. O menor valor é melhor; e
- $Tr(n)$: total de gatilhos de consolidação, *i.e.*, o somatório das ações de consolidação. O menor valor é melhor.

Do mesmo modo que na avaliação dos *snapshots*, para facilitar a avaliação dos gráficos de radar, a melhor solução é a que resulta no menor polígono centralizado. Exceto para a métrica de redução de consumo deve ser interpretada de forma inversa para encontrar o menor polígono. Com base

nas Figuras 5-(a) e 5-(b), observa-se que os cenários sem a carga de HA, vinculado ao algoritmo de consolidação máxima (C_MAX), apresenta o menor polígono, e portanto este é o melhor resultado. De fato, assemelha-se com a avaliação dos demais *snapshots*, em que é aplicado o mesmo algoritmo AA, tanto nos casos com a restrição 0% quanto 20% de afinidade.

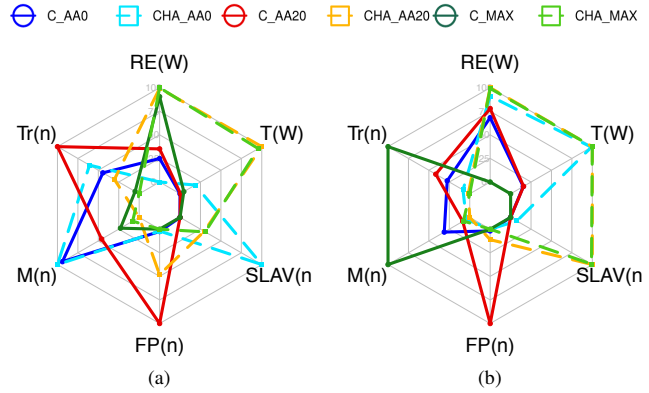


Figura 5. Testes globais para (a) AZ1 e (b) AZ2.

Como apenas duas métricas são compatíveis com a execução de Eucalyptus, (T(MWh) e C(%)) a Tabela III compara os valores correspondentes para o teste global.

Tabela III. RESULTADO DA EXECUÇÃO GERAL

Teste	Métrica	AZ1	AZ2
EUCA	T(MWh)	15799,8	23546,9
	C(%)	37,65	22,14
CHAVE_MAX	T(MWh)	11927,7	28854,3
	C(%)	58,41	84,85

Observa-se (Tabela III) para a AZ1 que mesmo com a adição da carga de HA, CHAVE ainda reduz o consumo de energia em aproximadamente 24,5%. Isso deve-se principalmente ao pequeno incremento na carga média da AZ, que foi de 37% para 58%. Por outro lado, ao quase quadruplicar a carga da AZ2 de 23% para 84%, houve um acréscimo de 22,5% no consumo de energia. Relacionado esses resultados com a Figura 3, as características da AZ1 indicam uma carga mais distribuída, quando comparada com a AZ2. Adicionalmente, no eixo de carga (x) observa-se que a AZ2 apresenta cargas próximas a 100%, indicando que em determinados momentos de sua execução, a AZ2, possui alta demanda de recursos, enquanto, a AZ1, apresenta valores mais próximos a 90%.

VI. TRABALHOS CORRELATOS

Inicialmente, encontra-se na literatura duas linhas de pesquisa distintas: replicação de MVs relacionada a mecanismos de tolerância a falhas para prover HA, e a consolidação de MVs relacionada com estratégias de eficiência energética. Analisar separadamente ambas permite obter uma perspectiva abrangente sobre cada tema, mas para o CHAVE, as duas abordagens são simultâneas e deve ter seus conflitos tratados. Pesquisas bibliográficas sistemáticas [20], [21] relatam poucos trabalhos que abordam simultaneamente estratégias de eficiência energética com HA quando relacionadas com os temas individualmente. Neste trabalho foi realizado um levantamento bibliográfico em três mecanismos de buscas: IEEE Explore, Scopus e ACM DL. Adicionalmente, os critérios de inclusão estão relacionados com aplicação de consolidação e/ou replicação de MVs no contexto de nuvem computacional IaaS, com data de publicação de até cinco anos. Assim, a Tabela IV resume a discussão sobre as contribuições e problemas não solucionados e que CHAVE utiliza.

Tabela IV. RELAÇÃO DOS TRABALHOS CORRELATOS.

Ref.	Pesquisa	Contribuições	Problemáticas
[7]	Cons	Usa o FFD para simulação	Desconsidera restrições afinidade
[22]	Cons	Correlaciona recursos físicos/virtuais	Utiliza apenas o posicionamento
[23]	Cons	Minimiza migrações	Desconsidera restrições de afinidade
[24]	HA	*Lockstep*: desempenho no uso de recursos	Invasivo: Requer TCP da MV modificado
[25]	HA	Replicação Multi-AZ	Taxonomia
[9]	HA	Multi-AZ/Cloud e restrição de afinidade	Apenas na perspectiva do inquilino
[26]	Cons+HA	Arquitetura *Big-Tree*, Multi-AZ	Sem HA sob demanda do inquilino
[27]	Cons+HA	Foco em não violar SLA	Modelo orientado à aplicação da MV.
[28]	Cons+HA	Replicação 1:M, conceitos probabilísticos	*Snapshots*, viola isolamento e segurança
[29]	Cons+HA	Problema da mochila para alocação k-HA	Realocação de MVs apenas após falha
[30]	Cons+HA	Evita ativar/desativar servidor pois isso reduz sua vida útil	Desconsidera antiafinidade para migrar

Os trabalhos na Tabela IV estão organizados entre apenas consolidação de MVs (Cons.) ou mecanismos de HA e, por fim, os abordam as duas linhas de pesquisa (Cons. + HA). Os trabalhos identificados possuem tanto aplicação em ambientes reais quanto em simulação, sendo comum utilizar simuladores para obter dados numéricos. Observa-se que considerar as restrições de afinidade e antiafinidade ocorre em uma minoria de trabalhos. Assim, a literatura especializada indica diversas abordagens para consolidação e HA, o que indica uma área de potencial demanda para pesquisa e para indústria. Porém, até o momento não foi identificada uma abordagem que concilie a replicação com a consolidação de MVs, tendo como base uma solução baseada em arquitetura e nas restrições de afinidade e antiafinidade, como propostas pelo presente trabalho.

VII. CONSIDERAÇÕES & TRABALHOS FUTUROS

As organizações que dependem da continuidade de negócios baseados em serviços críticos de TI demandam HA para mitigar os prejuízos causados pela interrupção de serviços em caso de desastre. Enquanto CSPs permitem implementar serviços críticos em arquiteturas multi-AZ, gerando a inerente carga da replicação de MVs, estratégias de consolidação de MVs podem ocasionar violações de SLA, infringindo as restrições de afinidade e antiafinidade. Assim, CHAVE atua na integração entre ambas abordagens, executando-as em níveis diferentes: a consolidação é aplicada apenas internamente à AZ (respeitando a afinidade MV-AZ); e a replicação é executada apenas entre AZs distintas (respeitando a antiafinidade $\text{MV}_{critica}$-$\text{MV}_{replica}$). Com base na probabilidade de eventos independentes e em análise por RBD, CHAVE especifica a quantidade ideal de réplicas para obter a taxa desejada de HA, permitindo o conceito de HA sob demanda. A consolidação de MVs é baseada no algoritmo FFD e possui duas estratégias propostas para o presente trabalho: o MAX permite obter o menor consumo de energia, mas exige mais migrações, e o

AA, que considera a restrição de afinidade entre MV-servidor para migrar.

Os resultados obtidos em simulação, utilizando *traces* reais, mostram que mesmo ao adicionar uma demanda de 20% de HA (gerada sinteticamente), se obtém uma redução de 24,5% no consumo de energia ao utilizar a consolidação com a estratégia CHAVE_MAX na AZ1. Esse resultado foi possível devido as características da AZ1 serem diferentes da AZ2, principalmente devido a diferença entre as cargas. 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 local, com e sem a replicação de MVs em *snapshots*, observa-se um comportamento similar, diferenciando-se o melhor cenário entre a AZ1 e AZ2. De fato, a configuração de distribuição das MVs entre os servidores de cada AZ e as diferentes demandas de requisições, permitem comportamentos diferentes. Embora não tenha sido quantificado o impacto financeiro, observa-se que a aplicação do CHAVE permite a redução de custos de energia e de custos aquisição de mais servidores, pois a consolidação permite utilizar melhor os recursos disponíveis. Na prática, a tomada de decisão sobre posicionamento inicial, migração (seleção das MVs de origem e servidor destino) e replicação (seleção da AZ que receberá a réplica) podem ser realizadas através das *Application Programming Interface*s (APIs) a nível de administrador. Assim, CHAVE pode ser associado às plataformas de computação em nuvem reais, sejam as públicas, como GCP, AWS ou Azure, ou privadas como OpenStack e CloudStack, que necessitem fornecer ou não um mecanismo de HA.

Ainda há 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 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.

AGRADECIMENTOS

Os autores agradecem o apoio do Laboratório de Processamento Paralelo e Distribuído (LabP2D) do CCT / UDESC em cooperação com a UNISINOS. Realizado utilizando recursos obtidos através da FAPESC.

REFERÊNCIAS

[1] P. R. M. Maciel, "Modeling Availability Impact in Cloud Computing," in *Principles of Performance and Reliability Modeling and Evaluation*, ser. Series in Reliability Engineering. Springer, 2016, pp. 287–320.

[2] P. M. Mell and T. Grance, "SP 800-145. The NIST Definition of Cloud Computing," NIST, Gaithersburg, MD, United States, Tech. Rep., 2011.

[3] T. A. Nguyen and et. al, "Model-Based Sensitivity of a Disaster Tolerant Active-Active GENESIS Cloud System," in *INIS 2017*. Springer, Cham, Sep. 2017, pp. 228–241.

[4] P. T. Endo and et. al, "High availability in clouds: systematic review and research challenges," *JoCCASA*, vol. 5, p. 16, Oct. 2016. [Online]. Available: https://doi.org/10.1186/s13677-016-0066-8

[5] G. L. Santos and et. al, "Analyzing the IT subsystem failure impact on availability of cloud services," pp. 717–723, Jul. 2017.

[6] L. Parziale and et. al, *End-to-End High Availability Solution for System z from a Linux Perspective*. IBM Redbooks, Oct. 2014.

[7] A. Beloglazov, "Energy-efficient management of virtual machines in data centers for cloud computing," PHD Thesis, The University Of Melbourne, 2013.

[8] T. Critchley, *High Availability IT Services*. CRC Press, Dec. 2014.

[9] R. Moreno-Vozmediano and et. al, "Orchestrating the Deployment of High Availability Services on Multi-zone and Multi-cloud Scenarios," *Journal of Grid Computing*, pp. 1–15, 2017.

[10] M. Machado and et. al, "Prototyping a high availability PaaS: Performance analysis and lessons learned," in *2017 IFIP/IEEE IM*, May 2017, pp. 805–808.

[11] Z. Li, L. Liu, and Z. Tong, "Study on Fault Tolerance Method in Cloud Platform based on Workload Consolidation Model of Virtual Machine," *JESTR*, vol. 10, no. 5, pp. 41–49, 2017.

[12] F. P. Tso, S. Jouet, and D. P. Pezaros, "Network and server resource management strategies for data centre infrastructures: A survey," *Computer Networks*, Sep. 2016.

[13] B. Rieck, "Basic analysis of bin-packing heuristics," 2010.

[14] A. Abdelsamea and et. al, "Virtual machine consolidation enhancement using hybrid regression algorithms," *Egyptian Informatics Journal*, vol. 18, Issue 3, pp. 161–170, Nov. 2017.

[15] M. Scarpiniti and et. al, "Energy performance of heuristics and meta-heuristics for real-time joint resource scaling and consolidation in virtualized networked data centers," *J Supercomput*, pp. 1–38, Jan. 2018.

[16] K. Su and et. al, "Affinity and Conflict-Aware Placement of Virtual Machines in Heterogeneous Data Centers." IEEE-ISADS, Mar. 2015, pp. 289–294.

[17] D. Ruck, C. Miers, M. A. Pillon, and G. Koslovski, "EAVIRA: Energy-Aware Virtual Infrastructure Reallocation Algorithm," in *Proceedings SBESC 2017*, Cutiba/PR/Brazil, nov 2017, p. 14.

[18] M. Hinz, G. P. Koslovski, C. C. Miers, L. L. Pilla, and M. A. Pillon, "A Cost Model for IaaS Clouds Based on Virtual Machine Energy Consumption," *Journal of Grid Computing*, pp. 1–20, May 2018.

[19] Z. Yao and I. Papapanagiotou, "A trace-driven evaluation of cloud computing schedulers for IaaS." IEEE-ICC, May 2017, pp. 1–6.

[20] Y. Sharma, B. Javadi, W. Si, and D. Sun, "Reliability and energy efficiency in cloud computing systems: Survey and taxonomy," *Journal of Network and Computer Applications*, vol. 74, pp. 66–85, Oct. 2016.

[21] H. Y. Shishido and J. C. Estrella, "Bibliometric analysis of workflow scheduling in grids and clouds," in *2017 36th SCCC*, 2017, pp. 1–9.

[22] Q. Zhang, G. Metri, S. Raghavan, and W. Shi, "Rescue: An energy-aware scheduler for cloud environments," *2014 Suscom*, vol. 4, no. 4, pp. 215–224, 2014.

[23] S. Mazumdar and M. Pranzo, "Power efficient server consolidation for Cloud data center," *FGCS*, vol. 70, pp. 4–16, 2017.

[24] Y. Dong and et. al, "COLO: COarse-grained LOck-stepping virtual machines for non-stop service," *ACM Press*, pp. 1–16, 2013.

[25] R. Ranjan, B. Benatallah, S. Dustdar, and M. P. Papazoglou, "Cloud Resource Orchestration Programming: Overview, Issues, and Directions," *IEEE Internet Computing*, vol. 19, no. 5, pp. 46–56, Sep. 2015.

[26] M. Simonin and et. al, "An Autonomic and Scalable Management System for Private Clouds," in *13th CCGrid*, May 2013, pp. 198–199.

[27] C. N. Sahoo and V. Goswami, "Cost and energy optimisation of cloud data centres through dual VM modes - activation and passivation," *IJCNDS*, vol. 18, no. 3/4, p. 371, 2017. [Online]. Available: http://www.inderscience.com/link.php?id=10004670

[28] X. Li, Y. Qi, P. Chen, and X. Zhang, "Optimizing Backup Resources in the Cloud," *CLOUD IEEE*, pp. 790–797, 2017.

[29] B. Shen and et. al, "High Availability for VM Placement and a Stochastic Model for Multiple Knapsack," in *2017 26th ICCCN*, Jul. 2017, pp. 1–9.

[30] A. Varasteh and et al., "On Reliability-Aware Server Consolidation in Cloud Datacenters," *ISPDC*, pp. 95–101, Sep. 2017, arXiv: 1709.00411. [Online]. Available: http://arxiv.org/abs/1709.00411