UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE CRATEÚS

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ELTON RODRIGUES DA SILVA

VMUALLOCATOR E CBHSELECTION: NOVAS POLÍTICAS VERDE DE ALOCAÇÃO E SELEÇÃO DE MÁQUINAS VIRTUAIS EM UM AMBIENTE DE DATACENTER

CRATEÚS

2023

ELTON RODRIGUES DA SILVA

VMUALLOCATOR E CBHSELECTION: NOVAS POLÍTICAS VERDE DE ALOCAÇÃO E SELEÇÃO DE MÁQUINAS VIRTUAIS EM UM AMBIENTE DE DATACENTER

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Crateús da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Msc.Filipe Fernandes dos Santos Brasil de Matos

CRATEÚS

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S579v Silva, Elton Rodrigues da.
VMUAllocator e CBHSelection : novas políticas verde de alocação e seleção de máquinas virtuais em um ambiente de Datacenter / Elton Rodrigues da Silva. – 2023.
84 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Crateús, Curso de Ciência da Computação, Crateús, 2023.
Orientação: Prof. Me. Filipe Fernandes dos Santos Brasil de Matos.

1. Computação em Nuvem. 2. Política de Alocação. 3. Política de Seleção. 4. Data center. I. Título.
CDD 004

ELTON RODRIGUES DA SILVA

VMUALLOCATOR E CBHSELECTION: NOVAS POLÍTICAS VERDE DE ALOCAÇÃO E SELEÇÃO DE MÁQUINAS VIRTUAIS EM UM AMBIENTE DE DATACENTER

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Crateús da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em:

BANCA EXAMINADORA

Prof. Msc.Filipe Fernandes dos Santos Brasil de Matos (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Msc. Luiz Alberto do Carmo Viana
Universidade Federal do Ceará (UFC)

Prof Msc. Francisco Anderson de Almada Gomes
Universidade Federal do Ceará (UFC)

À minha família, por sua capacidade de acreditar em mim e sempre me apoiar tanto nas horas boas e difíceis. Mãe, todo seu amor seu cuidado e dedicação foi que deram força, e o gás necessário para sempre seguir em frente e nunca desistir dos desafios.

AGRADECIMENTOS

Agradeço primeiramente a minha mãe, as minhas irmãs, e minha namorada, que me deram total apoio e sempre me ajudaram no meu crescimento não só pessoal como acadêmico.

Ao Prof. Msc. Filipe Fernandes dos Santos Brasil de Matos por me orientar, e pela paciência nos momentos mais complicados, sou muito grato.

Ao Prof. Msc. Luiz Alberto do Carmo Viana por todo aprendizado e conceitos através das disciplinas em que participei, onde irei guardar e levar para sempre.

Ao Prof. Msc. Francisco Anderson de Almada Gomes por compartilhar toda sua experiência no estágio supervionado e por me fazer acreditar em meu potencial enquanto desenvolvedor.

A Prof. Msc. Lisieux Marie Marinho dos Santos Andrade por grato pela sua simpatia e paciência ao longo das disciplinas que pude participar, onde se mostrou uma excelente profissional.

Ao Prof. Dr. Rennan Ferreira Dantas por ser um professor e coodenador de curso incrível e extremamente competente, e sempre com sua alegria contagiante.

Ao Prof. Msc. Allysson Allex de Paulo Araújo pelos por todo o aprendizado que me proporcinou na área desenvolvimento de sistemas, e por ter o prazer em trabalhar-mos no SisBarragens.

Ao Prof. Msc. Livio Antonio Melo Freire por todo aprendizado e em especial na disciplina Estrutura de Dados Avançada onde pude me aprofundar no assunto e que irei guardar para sempre.

A Prof. Msc. Lilian de Oliveira Carneiro por tudo pude aprender graças a experiência,dedicação, e simpatia como Professora.

Ao Prof. Msc. Italo Mendes da Silva Ribeiro por grato por compartihar sua expiência em jogos, computação gráfica, e IHc, que me ajudou a evoluir ainda mais no contexto de design em geral.

E aos meus demais colegas de curso e amigos que fiz ao longo da minha tragetória no Campus da UFC de Crateús. Agradeço aos demais funcionários que fazem parte do Campus que sempre trabalharam duro para que o nosso Campus tornasse o Campus incrível que temos hoje!

“Ele é meu Deus, meu refúgio, a minha fortaleza
e nele confiarei.”

(Salmo 91)

RESUMO

A Computação em Nuvem(Cloud Computing) consiste no fornecimento de recursos computacionais, nos quais se destacam serviços de bancos de dados, armazenamento, rede e diversas aplicações através da internet. Os recursos são fornecidos através de provedores, constituídos de diversas máquinas servidoras, formando uma infraestrutura conhecida como *data center*. As entidades que consomem esses recursos possuem máquinas virtuais distribuídas em todas as máquinas servidoras de um *data center*. O crescente aumento das infraestruturas dos data centers, ocasionaram em um grande aumento de consumo de energia elétrica. Por esta razão, diversas políticas de alocação vieram sendo propostas, por que estas políticas visam uma alocação mais otimizada, a fim de proporcionar uma maior redução do consumo energético. Este trabalho apresenta uma nova política de alocação (VMUAllocator) que tem como principal objetivo formar duplas de máquinas virtuais que possam ser alocadas de acordo com a capacidade máxima dos servidores, e uma nova política de seleção (CBHSelection) que analisa as máquinas virtuais e classificam como máquinas "potencialmente problemáticas", ou seja, que causam condições de disputa com outras máquinas dentro do servidor. A política de alocação proposta demonstrou bons resultados se destacando como a melhor política em termos energéticos na maioria dos ambientes. Já a política de seleção proposta obteve bons resultados em relação ao consumo energético, a mesma ficou em segundo lugar em todos os ambientes simulados.

Palavras-chave: Computação em Nuvem. Política de Alocação. Política de Seleção. *Data center*.

ABSTRACT

Cloud Computing consists of providing computational resources, in which database services, storage, network and various applications through the internet stand out. Resources are provided through providers, made up of several server machines, forming an infrastructure known as a Data center. The entities that consume these resources have virtual machines distributed across all server machines in a Data center. The growing increase in Data center infrastructures has led to a large increase in electricity consumption. For this reason, several allocation policies have been proposed, because these policies aim at a more optimized allocation, in order to provide a greater reduction in energy consumption. This work presents a new Allocation Policy (VMUAllocator) whose main objective is to form pairs of virtual machines that can be allocated according to the maximum capacity of the servers, and a new selection policy (CBHSelection) which analyzes the virtual machines and classifies as "potentially problematic"machines, that is, that cause race conditions with other machines within the server. The proposed allocation policy demonstrated good results, standing out as the best energy policy in most environments. The proposed selection policy obtained good results in relation to energy consumption, coming in second place in all simulated environments.

Keywords: Cloud Computing. Allocation Policy. Selection Policy. *Data center*.

LISTA DE FIGURAS

Figura 1 – Componentes da Computação em Nuvem 18
Figura 2 – Tipos dos serviços de Computação 19
Figura 3 – Tipos de nuvem e suas principais características 20
Figura 4 – Virtualização em máquinas físicas 21
Figura 5 – Representação do Problema de problema de Bin Packing 26
Figura 6 – Distribuição aleatória dos itens 27
Figura 7 – Exemplo da redução de recipientes utilizados. 27
Figura 8 – Exemplo de alocação de duplas. 34
Figura 9 – Alocações de duplas por tipo de *host* (VM1 e VM2, VM1 e VM3, VM2 e VM3. 35
Figura 10 – Listas de duplas VMs construídas através da capacidade máxima de cada *host*. 36
Figura 11 – hosts com diferentes porcentagens de utilização. 40
Figura 12 – Ordenação decrescente das VMs e dos hosts em relação ao seus atuais consumos de CPU. 41
Figura 13 – *host* H3 selecionado para receber a VM. 42
Figura 14 – Reordenação dos hosts. 42
Figura 15 – H3 recebendo a VM roxa. 43
Figura 16 – Resultado final da realocação. 43
Figura 17 – Gráfico de dispersão. 47
Figura 18 – Gráfico de dispersão com divisões por limiares. 48
Figura 19 – Consumo energético por ambiente. 62
Figura 20 – Quantidade de migrações por ambiente. 64
Figura 21 – SLATAH por ambiente de simulação. 65
Figura 22 – PDM por ambiente de simulação. 66
Figura 23 – Consumo energético em cada ambiente de simulação. 68
Figura 24 – Quantidade de migrações por ambiente. 69
Figura 25 – SLATAH por ambiente. 71
Figura 26 – PDM por ambiente de simulação. 72
Figura 27 – Consumo energético por ambiente. 73
Figura 28 – Quantidade de migrações por ambiente. 75
Figura 29 – SLATAH por ambiente de simulação. 76
Figura 30 – PDM por ambiente de simulação. 77

LISTA DE TABELAS

Tabela 1 – Tipos de VM e valor associado 48
Tabela 2 – Configurações dos servidores utilizados 54
Tabela 3 – Consumo de energia por carga de trabalho 54
Tabela 4 – Especificações das VMs 55
Tabela 5 – Resumo dos rastreamentos de carga de trabalho do mundo real utilizados . . 56
Tabela 6 – Políticas de Alocação e Seleção adotadas 58

SUMÁRIO

1 INTRODUÇÃO 13
1.1 CONTEXTUALIZAÇÃO 13
1.2 JUSTIFICATIVA 14
1.3 OBJETIVOS 15
1.3.1 Objetivo Geral 15
1.3.2 Objetivos Específicos 15
2 FUNDAMENTAÇÃO TEÓRICA 17
2.1 Computação em Nuvem 17
2.2 Virtualização 21
2.3 Green Cloud Computing 23
2.4 Políticas de Seleção e Alocação 24
2.5 Otimização 25
2.6 Problema de Bin Packing (ou Problema do Empacotamento) 26
2.6.1 Formulação do Problema 27
2.6.2 Soluções para o BPP 28
3 TRABALHOS RELACIONADOS 30
3.1 MSIALLOCATOR: Aplicação da Meta-Heurística ILS para o problema de alocação de máquinas em um data center 30
3.2 Energy-aware virtual machine allocation for cloud with resource reservation 31
3.3 An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment . . 32
4 VMUALLOCATOR E CBHSELECTION 33
4.1 VMUAllocator - *Alocação Inicial* 33
4.2 VMUAllocator - *Realocação* 39
4.3 CBHSelection 46
5 AMBIENTE DE TESTES 51
5.1 O simulador 51
5.2 Configuração Geral do Cenário 54
5.3 Métricas 59

5.3.1 ***Consumo de energia*** . . . 59
5.3.2 ***Quantidade de migrações*** . . . 59
5.3.3 ***Performace Degradation due to Migration*** . . . 60
5.3.4 ***SLA Violation Time per Active Host*** . . . 60
5.4 Configurações da máquina que efetuou a simulação . . . 61
6 RESULTADOS . . . 62
6.1 VMUAllocation (VMU) . . . 62
6.1.1 ***Consumo de energia*** . . . 62
6.1.2 ***Quantidade de migrações*** . . . 63
6.1.3 ***SLATAH*** . . . 65
6.1.4 ***PDM*** . . . 66
6.2 CBHSelection . . . 67
6.2.1 ***Consumo de energia*** . . . 67
6.2.2 ***Quantidade de migrações*** . . . 69
6.2.3 ***SLATAH*** . . . 70
6.2.4 ***PDM*** . . . 72
6.3 VMUAllocation e CHBSelection em comparação as melhores políticas da literatura . . . 73
6.3.1 ***Consumo de energia*** . . . 73
6.3.2 ***Quantidade de migrações*** . . . 74
6.3.3 ***SLATAH*** . . . 76
6.3.4 ***PDM por ambiente de simulação.*** . . . 77
7 CONCLUSÕES E TRABALHOS FUTUROS . . . 79
REFERÊNCIAS . . . 81

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

A Computação em Nuvem pode ser definida como um estilo de computação em que recursos dinamicamente escaláveis e frequentemente virtualizados são fornecidos como serviços pela Internet. Segundo (FURHT; ESCALANTE, 2010) a Computação em Nuvem se tornou uma tendência tecnológica significativa e alguns especialistas como (WANG *et al.*, 2015) e (QIN *et al.*, 2012) esperam que a Computação em Nuvem remodele os processos de tecnologia da informação (TI) e o mercado de TI. O autor ainda fala que com a tecnologia de Computação em Nuvem, os usuários usam uma variedade de dispositivos, incluindo PCs, laptops, smartphones e PDAs para acessar programas, armazenamento e plataformas de desenvolvimento de aplicativos pela Internet, por meio de serviços oferecidos por Provedores de Serviços da Nuvem.

A prestação de serviços para os contratantes é regulamentada através do Acordo de Nível de Serviço(Service Level Agreement ou SLA) onde é estabelecido um contrato com o provedor e especificado todos os serviços adquiridos. E também no mesmo, consta o registro dos valores do serviço prestado, suporte técnico, termos de compromisso entre outros. Com o SLA é possível garantir que não só o Consumidor mas também o Provedor do Serviço possam manter um vínculo transparente, e consequentemente evitando o descomprimimento de quaisquer direitos e deveres para ambos.

Os Provedores de Serviço em Nuvem geralmente usam abordagens baseadas em Virtualização para construir sua pilha. A Virtualização torna possível executar vários sistemas operacionais e vários aplicativos no mesmo servidor ao mesmo tempo. Ele cria uma camada abstrata que esconde a complexidade dos ambientes de trabalho de hardware e software. O paradigma da Computação em Nuvem permite que as cargas de trabalho sejam implantadas e escalonadas rapidamente por meio do rápido provisionamento dos recursos virtualizados. Esta implantação é feita por meio de máquinas virtuais (VMs) (AL-DHURAIBI *et al.*, 2018).

Com a Virtualização é possível fazer a consolidação de servidores, que consiste na alocação de um grande volume de máquinas virtuais em um grupo menor de máquinas físicas, com o intuito de explorar a capacidade máxima dos servidores. Assim ao final do processo é possível verificar quais servidores não possuem máquinas virtuais, para que a mesma possa ser desligada ou colocada em modo de hibernação proporcionando uma grande redução no consumo de energia elétrica.

A consolidação sempre deve ser feita de maneira cuidadosa, pois pode causar problemas como perca de desempenho pelo fato de que, quanto maior o número de máquinas virtuais compartilhando recursos, maior será as condições de disputa entre as VMs dentro do servidor. Outro problema que poderá surgir são as possíveis violações de SLA, caso ocorra uma grande de taxa de consolidação das VMs, os servidores ficam sobrecarregados gerando uma queda de performance.

Apesar da queda de performance a consolidação gera uma economia de energia por concentrar VMs em um grupo de servidores e deixar em hibernação os servidores livres. Visto o contexto apresentado, surge a necessidade da criação de um novo método para otimizar o processo de alocação de máquinas virtuais a fim de reduzir o consumo de energia elétrica por parte dos data centers.

1.2 JUSTIFICATIVA

Os data centers executaram, em 2018, 550% mais aplicativos do que em 2010 segundo dados da Forbes (2020). Tal informação mostra o aumento significativo na procura por soluções em Nuvem observada na última década. Segundo Data Center Frontier (2020) de acordo com a Structure Research empresa de pesquisa e consultoria no mercado de infraestrutura da Internet (Nuvem e data centers), o mercado de data centers para a região Ásia-Pacífico crescerá a uma taxa de anual de 12,2% entre 2018-2024.

Este grande interesse por soluções em Nuvem, culminou em um efeito colateral indesejado: o elevado crescimento no consumo energético dos data centers decorrente, em especial, da grande demanda pelos recursos físicos que compõe um *Data center*: equipamentos de infraestrutura de rede (em especial, servidores e comutadores de pacotes) e sistemas de refrigeração. Em 2017 o consumo de energia dos data centers atingiu cerca de 416 terawatts, representando 40% a mais do que toda a energia consumida pelo Reino Unido, um país industrializado com mais de 65 milhões de pessoas segundo a Forbes (2017).

Os centros de dados tradicionais menores abrigavam 79% das instâncias de computação do mundo em 2010, e em 2018 89% dos serviços de computação eram hospedadas por data centers em Nuvem, tanto em hiperescala quanto em instalações de Computação em Nuvem menores segundo DataCenter Knowledge (2020). Ainda de acordo com o DataCenter Knowledge (2020), em 2018, os data centers mundiais consumiram 205 terawatts-hora de eletricidade, ou cerca de 1% de toda a eletricidade consumida naquele ano em todo o mundo, inclusive em 2010.

Outro estudo recente (PANDA; JANA, 2018) faz referência a um relatório do Conselho de Defesa dos Recursos Naturais (NRDC) que indica que os data centers dos Estados Unidos consumiram aproximadamente 91 bilhões de kWh de eletricidade em 2013. De acordo com Data Center Frontier (2022), entre 2015 e 2021, o tráfego de Internet aumentou 440% e as cargas de trabalho dos data centers subiram em 260%, tais aumentos foram gerados no período da pandemia. E estima-se que os data centers sejam responsáveis por até 3% do consumo global de eletricidade hoje e devem atingir 4% até 2030 segundo DataCentre. (2022).

Diante do atual contexto, é observada a crescente utilização das soluções em nuvem, onde há um alto consumo de energia elétrica. Assim, necessita-se de uma Política de Alocação e uma Política de Seleção de Máquinas Virtuais para os servidores em um ambiente de *Data center*, a fim de reduzir o consumo de energia elétrica e também reduzindo ao máximo o impacto ambiental.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

Sugerir e implementar uma nova Política de Alocação e uma Política de Seleção de Máquinas Virtuais em um ambiente *Data center*, a fim de obter uma redução no consumo de energia elétrica global do mesmo. É importante também observar questões relacionadas a obediência as SLAs dos serviços prestados. Assim, a Política de Alocação proposta neste trabalho também deve impactar minimamente em possíveis violações de SLA. Em resumo, a nova política terá, como objetivo principal (e obrigatório), reduzir o consumo de energia do data centers, e, como objetivo secundário (e desejável), manter o mesmo nível de desempenho dos serviços prestados.

1.3.2 Objetivos Específicos

- Propor uma nova Política de Alocação e uma Política de Seleção de Máquinas Virtuais em um ambiente *Data center* que atenda ao Objetivo geral descrito acima e implementá-la no simulador CloudSim;
- Identificar Políticas de Alocação e Políticas de Seleção já existentes na literatura a fim de confrontar seus desempenhos com o desempenho das políticas propostas neste trabalho;

- Definir os cenários de teste e as métricas para análise de desempenho que proporcionarão um estudo comparativo entre as políticas testadas;
- Analisar os resultados obtidos e identificar os impactos negativos e positivos das novas políticas propostas.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo consta a fundamentação teórica necessária para o entendimento da Política de Alocação e a Política de Seleção propostas neste trabalho. Os assuntos abordados são: a Computação em Nuvem, Virtualização, Green Cloud Computing (Computação em Nuvem Verde), Políticas de Seleção e Alocação, Otimização, e o Problema de Bin Packing (ou Problema do Empacotamento).

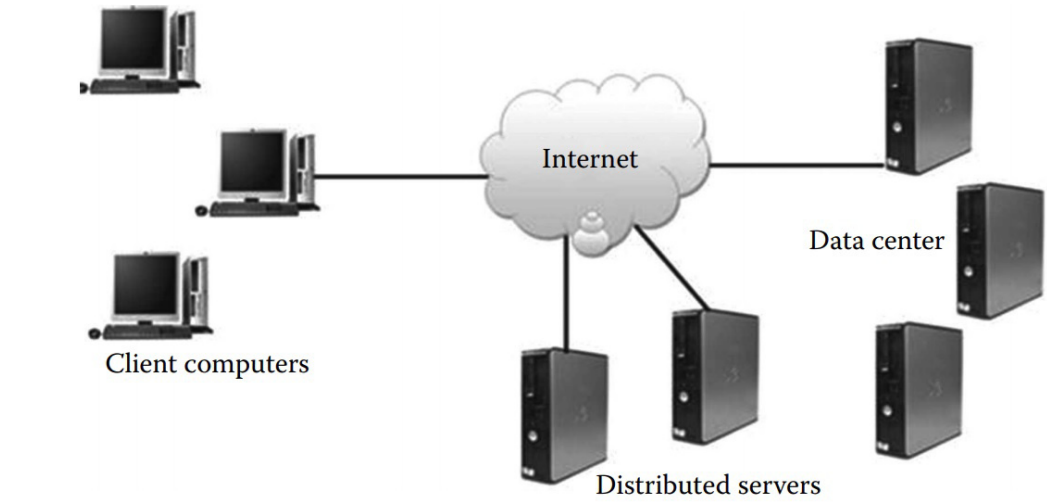
2.1 Computação em Nuvem

Computação em Nuvem consiste na visão da computação como um serviço público. Uma Nuvem é definida como um conjunto de serviços de aplicativo, armazenamento e computação através da Internet, suficientes para suportar as necessidades da maioria dos usuários, permitindo assim que eles possam utilizar em grande medida ou totalmente, de um software local de armazenamento e processamento de dados ou em alguns casos, do próprio aplicativo em execução (COULOURIS *et al.*, 2013).

Organizações de todos os tipos, portes e setores usam a Nuvem para uma grande variedade de finalidades, como por exemplo um imenso repositório para salvar backup de dados, ferramentas para o desenvolvimento e o teste de software, análises de big data e para a disponibilização aplicativos web voltados ao cliente. Por exemplo, as empresas do setor de saúde usam a Nuvem para desenvolver tratamentos mais personalizados para os pacientes. Empresas de serviços financeiros usam a Nuvem como base para detectar e prevenir fraudes em tempo real. Fabricantes de videogames usam a Nuvem para entregar jogos online para milhões de jogadores em todo o mundo(AMAZON WEB SERVICES, 2020).

O centro de processamento e armazenamento de dados em Computação em Nuvem é denominado *Data center*. Em geral, um *Data center* é composto por diversas máquinas físicas, conhecidas como servidores, interconectadas entre si e a Internet através de uma infra-estrutura de rede. A Figura 1 apresenta a arquitetura típica de uma Nuvem em um cenário de Computação em Nuvem. A Nuvem é constituída por um conjunto de clientes que são os Consumidores de serviços oferecidos por ela. O *Data center*, por sua vez, é composto por várias máquinas servidoras, responsáveis por prover serviços a estes clientes, nos quais podem ser destacados os serviços de armazenamento de informações, processamento, entre outros. Clientes acessam o *Data center* através da Internet.

Figura 1 – Componentes da Computação em Nuvem



Fonte: (DEBASHIS DE, 2016).

Diferentemente da comunicação entre usuários e computadores estabelecida de forma local usada nos mainframes (considerados por muitos uma versão inicial da Nuvem), a Computação em Nuvem expandiu essa comunicação a um nível mundial através da Internet, possibilitando uma interação sem uma proximidade física e proporcionando um maior alcance geográfico. Além dessa flexibilidade por conta da localização, a Nuvem também é bastante atrativa por ofertar serviços sob demanda e a Virtualização de recursos.

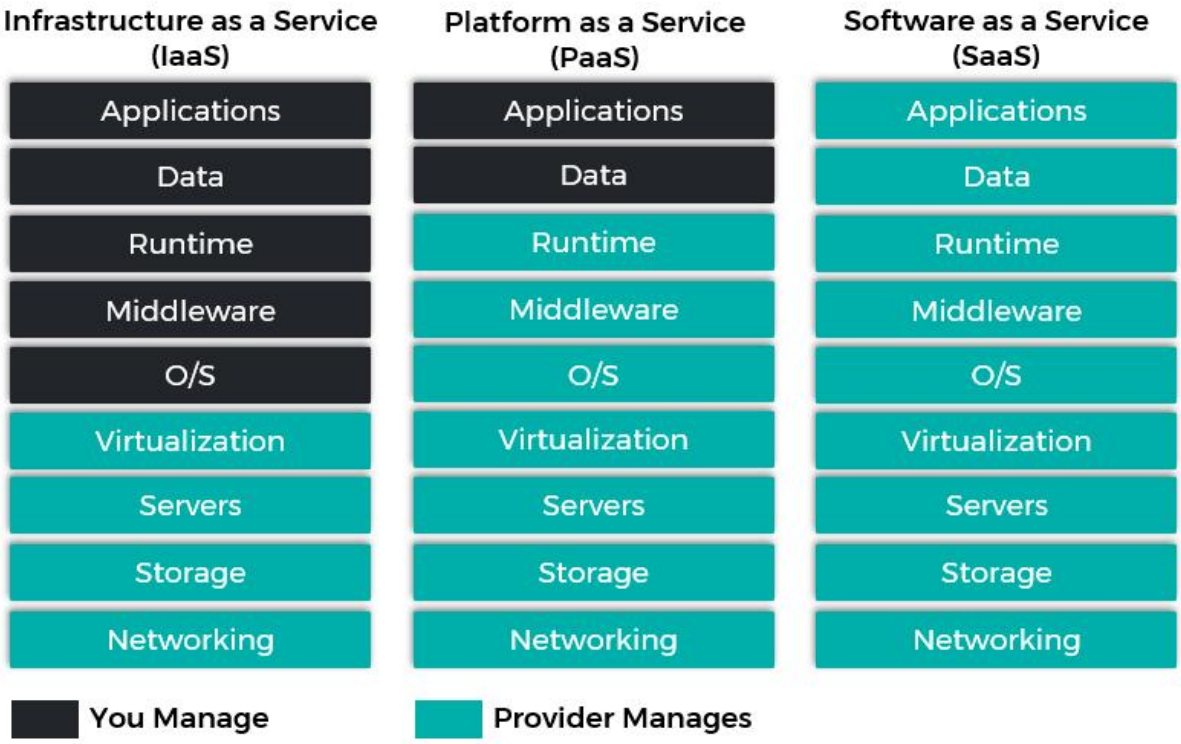
Entre as principais características da Computação em Nuvem podem ser destacadas o compartilhamento de recursos e serviços. Estes podem ser distribuídos entre múltiplos usuários de acordo com a capacidade disponível (em termos de processamento ou armazenamento) e de acordo com a demanda dos usuários. Assim, nessa forma de computação os usuários pagam conforme o uso, de acordo com o modelo *pay-as-use*.

Os serviços oferecidos pela Computação em Nuvem podem ser contratados pelos Consumidores (pessoas físicas ou jurídicas) que são aqueles que utilizam os serviços prestados pelos Provedores (comumente médias e grandes empresas), em que estes oferecem acesso aos serviços. Assim, constitui-se a função do Provedor conceder os serviços a serem executados em seus servidores, como armazenamento, backup de conteúdo, processamento entre outros.

Segundo Microsoft Azure (2020) os modelos mais utilizados nos serviços de Computação em Nuvem se enquadram em três tipos principais, e que cada uma delas define qual o tipo

de recurso a ser disponibilizado, e elas são: Infraestrutura como Serviço (IaaS), Plataforma como Serviço (PaaS) e Software como Serviço (SaaS). A Figura 2 representa como são distribuídos os recursos em forma de pilha em cada um dos tipos de serviço, destacando na cor preta os recursos que são gerenciados pelo cliente, e em ciano os recursos gerenciados e ofertados pelo provedor de serviço.

Figura 2 – Tipos dos serviços de Computação



Fonte: INAP (2020).

- **Infraestrutura como Serviço:** Nesse tipo é oferecido recursos de hardware como serviço. Dessa forma o usuário utiliza os recursos, através da máquina virtual, onde estará configurado o processador, memória e rede, de acordo com as configurações solicitadas por ele. Os principais exemplos de serviço IaaS são a Amazon Web Services (AWS), o Google Compute Engine e o Microsoft Azure.
- **Plataforma como Serviço:** É oferecido um ambiente sob demanda para o usuário, no qual podemos destacar como alguns exemplos, ambientes de desenvolvimento, editores de texto, e ferramentas de controles de versão. Dentre os principais exemplos de serviços PaaS estão o Google App Engine, Heroku, e a Microsoft Azure Cloud Services.

- **Software como Serviço:** Constituída por aplicações completas que podem ser acessadas através da Nuvem, e assim os consumidores podem utilizá-los através da Internet. Os dois principais exemplos desse tipo de serviço são o Google Docs e o Facebook.

Também há classificações quanto ao tipo de Nuvem de acordo com o seu público-alvo. Nesse aspecto a Nuvem pode ser subdividida em três tipos: Públicas, Privadas ou Híbridas. A Figura 3 apresenta os três tipos de Nuvem destacando as suas principais características, repare que a Nuvem Híbrida agrega características da Nuvem Pública e da Nuvem Privada.

Figura 3 – Tipos de nuvem e suas principais características

Fonte: Scurra (2017).

- **Nuvem Pública:** Em uma Nuvem Pública, os serviços ofertados por ela são disponibilizados para o público em geral, ou seja, não há restrições de acesso, tornando esse tipo de Nuvem inviável para disponibilizar serviços com um alto grau de segurança.
- **Nuvem Privada:** Os serviços implantados nesse tipo de Nuvem possuem mecanismos de segurança mais elevados, por conta da sua utilização para armazenamento de informações e gerência de aplicações pertencentes a empresas com uso particular.

- **Nuvem Híbrida:** Representa a junção das principais características dos dois tipos de Nuvem citadas anteriormente, a Nuvem Pública e a Nuvem Privada. A vantagem desse tipo de nuvem é a possibilidade de trazer segurança para os dados privados e a disponibilidade de dados não pertinentes ao público.

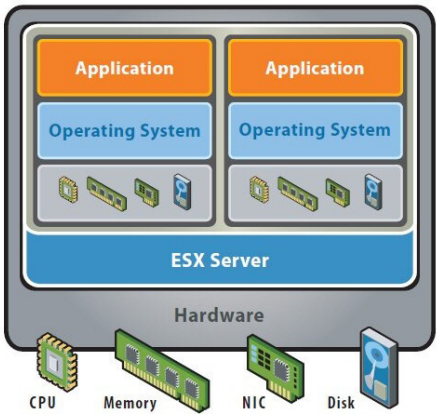
A Política de Alocação e a Política de Seleção de máquinas virtuais que foi desenvolvida neste trabalho é direcionada para Nuvens Privadas, que disponibilizam serviços do tipo IaaS. Uma vez abordada as principais características sobre a Computação em Nuvem, a seguinte seção apresentada o conceito de Virtualização e a sua utilização em servidores pertencentes a um *Data center*.

2.2 Virtualização

A Virtualização é um dos principais componentes da Computação em Nuvem, que trata do compartilhamento de recursos através da criação de instâncias lógicas da estrutura real da máquina física onde a mesma foi criada. Tais instâncias são comumente chamadas de máquinas virtuais (Virtual Machine ou VM) e podem assumir diferentes características obedecendo às limitações de hardware da máquina real (host).

Uma VM é um ambiente computacional que pode ser constituído por sistemas operacionais, e diversos programas, funcionando como uma cópia isolada de uma máquina física. Ao ser criada, uma VM pode ser configurada em termos de memória, CPU, disco e placa de rede. A Figura 4 ilustra duas máquinas virtuais hospedadas em um *host*. Note que cada máquina virtual possui seus próprios componentes de hardware (criados e disponibilizados de forma lógica) e software.

Figura 4 – Virtualização em máquinas físicas



Fonte: Oracle E-Business Suite Technology (2006).

A Virtualização baseada em hipervisor convencional tem sido a tecnologia de fato usada durante a última década para implementar Virtualização e isolamento de servidor. Os hipervisores operam no nível do hardware - isto é, construindo hardware virtual personalizável e drivers de dispositivo virtual - suportando assim máquinas virtuais autônomas (VMs) que são independentes e isoladas do sistema *host* subjacente. Em cada instância de VM, um sistema operacional (SO) completo é normalmente instalado em cima do hardware virtualizado, gerando grandes imagens de VM. Além disso, a emulação de dispositivos de hardware virtuais e drivers relacionados produz uma sobrecarga de desempenho não desprezível (MORABITO *et al.*, 2018).

Um conjunto *N* de VMs podem ser alocadas dentro de um mesmo *host*, porém essa quantidade *N* é limitada pelo recursos físicos do *host*. Com a utilização do *Sandboxing* é possível a garantida da criação e manutenção de várias VMs por conta da aplicacão de um isolamento que proporciona uma separação lógica das mesmas. Além de assegurar que as atividades de cada uma permaneça isolada dentro do seu ambiente, ela proporciona uma maior segurança contra atividades maliciosas ou não entre as VMs.

Essa forma de alocação traz benefícios como a Consolidação de servidores, porém ela pode ocasionar uma série de malefícios. Dentre os problemas podem acontecer concorrências entre as VMs por recursos do *host* proporcionado uma grande queda de desempenho e posteriormente possíveis violações de SLA.

A seção seguinte apresenta Green Cloud Computing(Computação em Nuvem Verde) que aborda a utilização da consolidação de servidores de forma inteligente a fim de proporcionar uma maior economia de energia. Também é destacado os pontos negativos em relação a consolidação e a importância da Green Cloud Computing para a Computação em Nuvem.

2.3 Green Cloud Computing

Com o rápido crescimento da Computação em Nuvem ao redor do mundo, notou-se um grande aumento no consumo de energia elétrica por parte da infraestrutura da Nuvem (em especial, os data centers), juntamente com grandes taxas de emissão de carbono. Com o aparecimento desses efeitos negativos, os provedores começaram a buscar técnicas que trouxessem mais eficiência quanto ao consumo de energia e o mínimo de impacto ambiental possível. Essa prática ficou conhecida como Green Cloud Computing (Computação em Nuvem Verde).

O estudo sobre a alocação e programação eficiente de nós de recursos em data centers, reduz os custos de operação e manutenção da plataforma em Nuvem, de forma a minimizar o consumo de energia e geração de calor, o que é de grande significado teórico e prático para a Computação em Nuvem Verde (LU; SUN, 2019). A consolidação de servidores é uma abordagem popular para reduzir o consumo de energia dos data centers em Nuvem, minimizando o número de máquinas físicas ativas (HAN *et al.*, 2017).

Contudo, a consolidação de servidores possui seus pontos positivos e negativos. O ponto positivo é ao consolidar os servidores é possível ter uma grande economia de energia, porque a consolidação consiste e alocar várias VMs em um grupo especifico de servidores. E ao deixar outros servidores livres, os mesmos podem ser desligados ou colocados em modo de hibernação posteriormente.

O ponto negativo ao consolidar os servidores é a ocorrência no aumento de disputas entre as VMs por recursos dos servidores. Essa disputa por recursos podem causar a degradação de desempenho dos serviços, e consequentemente levando a possíveis violações de SLA. Podendo causar posteriormente uma série de penalidades para o Provedor de serviço.

A Computação em Nuvem Verde que tem com objetivo minimizar o consumo energético dos data centers, otimizando a utilização de recursos dos *hosts* através da Virtualização e Consolidação. A proposta apresentada neste trabalho, compartilha do mesmo objetivo através de uma nova Política de Alocação e uma nova Política de Seleção de VMs. A seguinte seção apresenta a definição e os objetivos das Políticas de Alocação e Seleção, juntamente com suas principais diferenças.

2.4 Políticas de Seleção e Alocação

As Políticas de Alocação e Políticas de Seleção são técnicas que auxiliam no gerenciamento das VMs distribuídas nos hosts pertencentes a um *Data center*. Possibilitando uma melhor organização e gestão do consumo dos recursos, e consequentemente proporcionando uma maior eficiência energética. O que as torna técnicas essenciais e de extrema importância na consolidação de servidores.

A Política de Seleção tem como principal papel selecionar uma ou mais VMs para que sejam migradas para outros *hosts*, tendo como base um ou mais critérios predefinidos. De acordo com (BELOGLAZOV; BUYYA, 2011), as Políticas de Seleção comumente utilizam como critério o nível de utilização de CPU. Já a Política de Alocação é uma técnica que consiste na escolha do *host*, que irá receber as VMs retornadas da Políticas de Seleção e alocá-las, considerando os recursos(por exemplo, CPU, memória principal, ou memória secundária).

(BELOGLAZOV; BUYYA, 2011) tratou a problemática da consolidação levando em consideração vários *hosts* e várias VMs, utilizando *hosts* heterogêneos e limitando a capacidade máxima de CPU que cada VM pode consumir. O autor utilizou como base os dados do histórico de utilização de recursos de VM, e dividiu as etapas da consolidação dinâmica em quatro partes, definidas através de limiar da utilização total dos recursos no *host*.

A primeira parte do modelo de consolidação proposto por (BELOGLAZOV; BUYYA, 2011), consistia em verificar se o *host* está sobrecarregado, o que exige a migração de uma ou mais VMs. Na segunda parte verifica-se se o *host* está com carga insuficiente, o que ocasiona na migração de todas a VMs e deixando o mesmo em modo de suspensão. A terceira parte consiste na seleção das VMs que devem ser migradas dos *hosts* que foram classificados como sobrecarregados. Na ultima parte, consiste na alocação das VMs selecionadas pela migração.

(BELOGLAZOV; BUYYA, 2011) fez uma analogia a alocação de VMs ao um problema de lixeiras com variados tamanhos e preços de depósito, onde as mesmas são os nós físicos(hosts) e as VMs seriam os itens a serem alocados na lixeira. O tamanho das lixeiras correspondem a capacidade de CPU, e preço corresponde a energia consumida pelos nós, e a partir disso o autor classificou com problema não determinístico em tempo polinomial(NP-Difícil). Ele também propôs um o algoritmo modificado de Redução do Melhor Ajuste(Best Fit Decreasing ou BSD), onde ela classifica todas as VMs em ordem decrescente de acordo com a sua utilização de CPU, e em seguida alocar as VMs para o *host* de forma gerar o menor aumento possível no consumo de energia.

2.5 Otimização

Otimização se refere à análise e estudo de problemas complexos a fim de encontrar a melhor solução possível para o problema em questão. A busca pela melhor solução a um determinado problema é feita através de métodos que têm por objetivo selecionar a melhor opção (relativamente a alguns critérios pré-definidos) entre um conjunto de alternativas disponíveis. Dependendo do problema em questão, a sua solução pode estar sujeita a busca por uma minimização ou maximização.

Um problema de minimização se refere a um problema no qual dado um certo conjunto de valores ou grandezas em geral, é feita a busca pelo menor grandeza possível dado um determinado conjunto de restrições (Ex.: Minimizar o valor gasto em compras em um supermercado). Tal qual o problema de minimização, um problema de maximização exige um conjunto de restrições para a resolução do problema, porém ele consiste em maximizar uma grandeza (Ex.: Pegar 3 moedas de um conjunto de moedas de forma a ter o maior valor possível).

Os problemas de otimização combinatória possuem resoluções em dois tipos, no qual pode ser por métodos exatos ou aproximativos (PAPADIMITRIOU C. H; STEIGLITZ, 1998). O método exato assegura que a solução final é a melhor solução possível, também chamada de solução ótima global. Já em um método aproximativo a solução final não seria necessariamente seria a uma solução ótima global, mas seria uma solução ótima local.

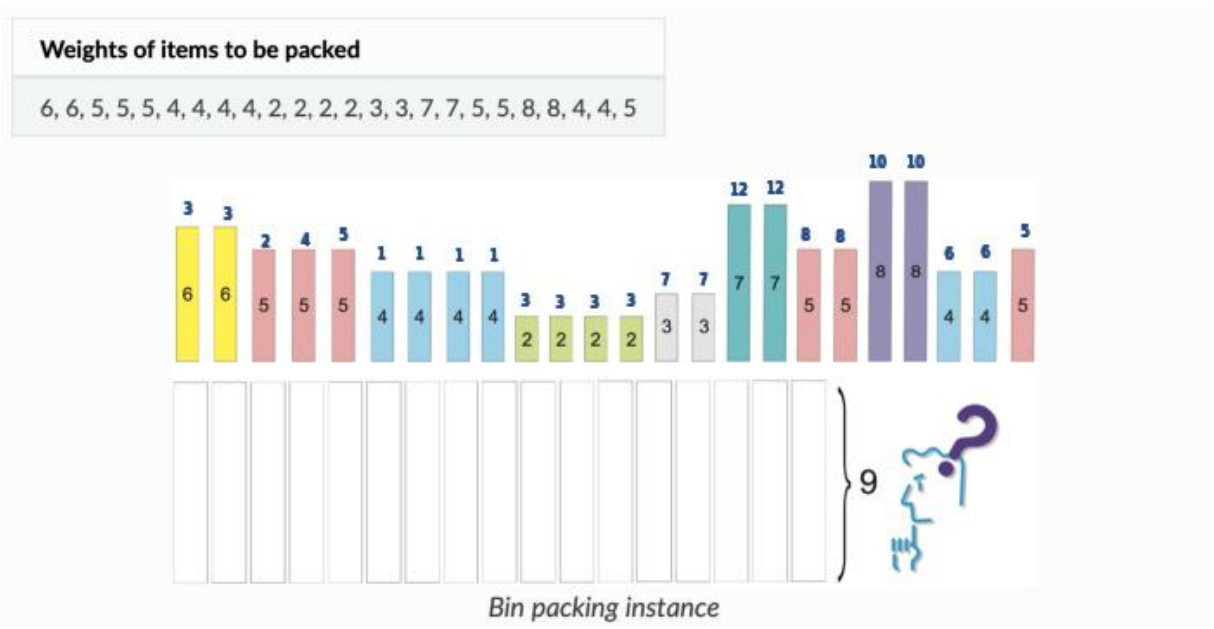
No uso de métodos exatos, pode ser utilizado um modelo de Programação Linear ou Não Determinístico, que se trata de uma técnica de otimização para encontrar um valor ótimo global, podendo ser máximo ou mínimo, nos quais podem estar sujeitos a um conjunto de restrições (PRADO, 2010). As implementações mais comuns de métodos exatos utilizam da técnica da “força bruta”, onde é analisado todo o conjunto de possíveis soluções, e em seguida é retornada a melhor solução do conjunto (CARVALHO, 1999).

Com relação aos métodos aproximativos, existem dois tipos que são as Heurísticas e as Meta-Heurísticas(SALHI, 2014). Uma Heurística busca resolver um problema rapidamente, porém sem a garantia de encontrar a melhor solução possível para um problema, é bastante utilizada para resolver um problema em específico. Já as Meta-Heurísticas, assim como a Heurística, ela busca resolver o problema com menor tempo computacional, sem garantir que irá encontrar a melhor solução possível. No entanto, uma Meta-Heurística por ter um caráter genérico a mesma pode ser utilizada em vários tipos de problemas, sendo necessário apenas modificar os parâmetros da mesma (SALHI, 2014).

2.6 Problema de Bin Packing (ou Problema do Empacotamento)

O Problema de Bin Packing (Bin Packing Problem ou BPP) clássico segundo (KANG; PARK, 2003) é definido por um conjunto n de recipientes de capacidades máximas iguais e finitas, e um conjunto m de itens ou objetos com diferentes valores e pesos. O objetivo deste problema é distribuir os m itens no menor número possível dos n recipientes. É importante destacar que a capacidade máxima dos recipientes deve ser sempre respeitada, assim, o somatório dos pesos dos objetos alocados em um determinado recipiente nunca deve ultrapassar a capacidade máxima deste último. Outro ponto destacável está relacionado ao valor do item, que pode ser representado na forma de um custo específico do item.

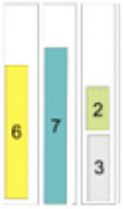
Figura 5 – Representação do Problema de problema de Bin Packing



Fonte: The Startup (2021).

O objetivo do BPP é alocar todos os itens em recipientes de forma a minimizar ao máximo a quantidade de recipientes a serem utilizados. Utilizando a ilustração acima pode ser apresentado o seguinte exemplo, utilizando quatro itens e três recipientes de tamanho 9 ilustrados na Figura 6.

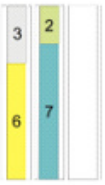
Figura 6 – Distribuição aleatória dos itens



Fonte: O próprio autor.

No exemplo os quatro itens foram distribuídos inicialmente de forma aleatória nos recipientes. No primeiro recipiente foi alocado um item de peso 6, no segundo um item de peso 7, e no terceiro dois itens, um de peso 2 e outro de peso 3. A ideia do BPP é reduzir a quantidade de recipientes a serem utilizados, onde uma das formas é explorar a capacidade máxima de cada recipiente para que ocorra uma maior incidência de recipientes vazios. Na Figura 7 temos uma alocação mais otimizada dos itens nos recipientes, o que reduziu a quantidade de recipientes utilizados de 3 para apenas 2 recipientes.

Figura 7 – Exemplo da redução de recipientes utilizados.



Fonte: O próprio autor.

No contexto do presente trabalho o BPP é análogo ao problema de alocação de VMs, onde as VMs são os itens, e os recipientes são os *hosts*. O peso de um item é equivalente ao consumo de CPU (em MIPS) das VMs, e o valor de um item é equivalente ao consumo energético da VM. A capacidade máxima de um recipiente é análoga à capacidade máxima de um *host* referente ao quanto de CPU ele pode processar. E assim como no BPP o problema que destacamos no presente trabalho consiste em reduzir a utilização de recipientes, ou seja, reduzir a quantidade de *hosts* utilizados para que haja uma maior economia de energia no ambiente *Data center*.

2.6.1 Formulação do Problema

O BPP pode ser definido da seguinte maneira: Dado um conjunto de itens com seus pesos e um conjunto de caixas de tamanho fixo, encontre o número mínimo de caixas que

podem conter todos os itens. Matematicamente, a formulação do BP pode ser expressa como um problema de programação inteira da seguinte forma:

$$\min \quad \sum_{i=1}^{n} y_i \tag{2.1}$$

$$\text{s.t} \quad \sum_{j=1}^{n} w_j x_{ij} \leq c y_i i \in N = \{1,...,n\}, \tag{2.2}$$

$$\sum_{i=1}^{n} x_{ij} = 1,\ j \in N, \tag{2.3}$$

$$y_i \in \{0,1\},\ i \in N, \tag{2.4}$$

$$x_{ij} \in \{0,1\},\ i \in N,\ j \in N, \tag{2.5}$$

Fonte: (ABDEL-BASSET *et al.*, 2018)

O BPP trata-se de um problema de minimização, por que no problema em questão busca-se minimizar a quantidade de recipientes ao alocar um certo conjunto de itens(linha 2.1). O problema está sujeito às seguintes restrições:

- Se o item *j* estiver alocado(*xij*) em um recipiente *i*, o peso do item(*wj*) deve ser menor ou igual a capacidade do recipiente(*yi*) e os itens associados ao recipiente *i* não podem ter seu peso total superando a capacidade *c* do recipiente *i* (linha 2.2);
- Ao final do processo todos os itens devem estar alocados(linha 2.3);
- *yi* é uma variável binária que indica se o recipiente *i* contém(1) ou não(0) itens (linha 2.4);
- *xij* é uma variável binária que indica se se o item *j* é atribuído ou não ao recipiente *i* (linha 2.5).

2.6.2 Soluções para o BPP

O Problema de Bin Packing é bem conhecido por ser NP-difícil (MICHAEL *et al.*, 1972). E apesar do BPP ser NP-difícil, as melhores soluções encontradas inicialmente para instâncias do problema, foram adquiridas através de algumas heurísticas onde as mesmas foram definidas e analisadas assintoticamente por (ULLMAN, 1971): First Fit(FF), Best Fit(BF). Mais tarde (MICHAEL *et al.*, 1972) definiram mais duas heurísticas a First Fit Decreasing (FFD) e

Best Fit Decreasing(BFD) e forneceram análises mais detalhadas das quatro heurísticas.

O FF é um algoritmo de aproximação no qual ele recebe os itens em ordem arbitrária e para cada item é feita a tentativa de colocá-lo no primeiro recipiente que pode recebê-lo. Se não for encontrado nenhum recipiente, é criado um novo recipiente para receber o item. O BF funciona analogamente ao FF, ao invés de colocar o item no primeiro recipiente onde o mesmo pode se encaixar, o item é alocado em um recipiente de maior capacidade, onde o item pode se encaixar. O FFD também funciona analogamente ao FF, porém antes de dar início a alocação dos itens, os mesmos são classificados em ordem decrescente em relação aos seus tamanhos. E o BFD é análogo ao BF porém antes de dar início a alocação, os itens são reorganizados em ordem descrente de acordo com seu tamanho.

3 TRABALHOS RELACIONADOS

Nesta seção, foram selecionados trabalhos que abordam a mesma problemática apresentada no presente trabalho, juntamente com sua respectivas soluções propostas. Foram selecionados trabalhos publicados a partir de 2016, com qualis igual ou superior a B2, com algorítimo presente no trabalho ou no repositório e com a utilzação do CloudSim como ambiente de testes.

3.1 MSIALLOCATOR: Aplicação da Meta-Heurística ILS para o problema de alocação de máquinas em um data center

(FILHO, 2018) propõe uma nova política de alocação de máquinas virtuais (a MSIAllocator) que, através de uma metaheurística chamada ILS (Iterated Local Search), escolhe, dentre os servidores candidatos, aqueles mais aptos a receber máquinas virtuais durante a alocação inicial ou realocação. A política proposta considera a capacidade máxima de processamento e consumo de energia do servidor, com o objetivo de minimizar o consumo energético dos data centers.

A MSIAllocator é dividida em duas partes principais: a primeira compreende a alocação inicial das máquinas virtuais no ambiente de *Data center*. Como ainda não é possível determinar o consumo real das máquinas virtuais (apenas aquilo que é previsto via SLA), esta etapa gera uma solução inicial fazendo uma alocação aleatória válida das máquinas virtuais nos servidores, respeitando a capacidade máxima de hardware de cada um. A segunda parte oxorre com o ambiente em execução e, dessa forma, é possível determinar as demandas computacionais reais das máquinas virtuais, e a política de alocação MSIAllocator seleciona os servidores que poderao receber as VMs selecionadas pela migração.

A política proposta por (FILHO, 2018) consiste na geração de uma solução inicial válida para os servidores, uma perturbação que gera mais soluções diferentes da solução inicial, seguida por uma busca local que visa refinar a solução anterior. Após a busca local, é executada a função de aceitação que retorna o melhor servidor que poderá receber a máquina virtual na iteração corrente. E por último, o critério de parada que segundo Filho(2018) foi definido como o número de iterações num total de 25, definida através de observações empíricas.

3.2 Energy-aware virtual machine allocation for cloud with resource reservation

(ZHANG *et al.*, 2019) aborda a alocação de máquinas virtuais como uma interação evolutiva onde, inicialmente ocorre um mapeamento geral das máquinas virtuais para as máquinas físicas através de alguns métodos clássicos escolhidos pelo autor: as heurísticas de primeiro a ajuste (First Fit ou FF) e com a Heurística de Diminuição de Melhor Ajuste Modificada (Modified Best Fit Decreasing ou MBFD). A heuristíca MBFD ajusta as VMs em ordem decrescente de acordo com sua demandas de CPU, faz a seleção de um *host* que atenda as demandas de recursos da VM migrada, e leve ao menor consumo de energia. Já a heurísca FF consiste em atribuir a VM ao primeiro *host* dentre a lista de *hosts* que atenda aos requisitos de recursos de CPU por parte da VM.

A busca pela solução ideal é dividida em três evoluções onde cada uma contém *n* pesquisas a fim de explorar o melhor mapeamento. Na execução de uma interação evolutiva ocorre o agrupamento em pares de máquinas físicas de forma aleatória, para que posteriormente seja feita operações de crossover. As operações de crossover consistem na troca de VMs entre os pares de *hosts*, considerando-se que as VMs a serem trocadas tem a menor sobreposição de execução em relação as VMs do mesmo *host*. Caso alguma VM seja rejeitada por conta de alguma restrição de recurso de CPU, a mesma será realocada ao final na iteração utilizando o FF.

Segundo (ZHANG *et al.*, 2019) o agrupamento em pares é utilizado para otimizar o mapeamento de máquinas virtuais para as máquinas físicas. Dividindo-se aleatoriamente as máquinas físicas não ociosas em pares para que posteriormente seja feita os ajustes de alocação das VMs entre os *hosts*.

A operação de crossover é utilizada para melhorar localmente a adequação entre as máquinas físicas onde são considerados quatro cenários com em três tipos de ações de alocação: 1) o clone onde é duplicado o par de alocação do *host* orginal; 2) o move onde ocorre a migração de uma VM a partir de um *host* para outro; 3) o exchange uma ação de troca de VMs entre *hosts*. Cada cenário corresponde a um tipo de alocação, exceto o move que é equivalente a dois cenários, por exemplo ao migrar uma VM *a* de um h1 para um *host* h2 ou migro uma VM *b* do *host* h2 para o *host* h1.

3.3 An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment

(ABDEL-BASSET *et al.*, 2019) utilizou como base um algoritmo meta-heurístico chamado Whale Optimization Algorithm(WOA), onde o mesmo é baseado na estratégia de alimentação de baleias. A estratégia de alimentação, consiste na formação de uma rede de bolhas, formadas através do nado em espiral das baleais em torno da sua presa. Assim como na meta-heurística as posições da baleias são aleatórias inicialmente e conforme elas encontram a melhor posição para a captura da presa a mesma atualiza sua posição.

O autor aplica a distribuição de Lévy na meta-heurística WOA, por que segundo ele a WOA apresentava alguns problemas de desempenho com relação ao espaço de busca contínua proporciona por uma certa deficiência na aplicação de prolemas combinatórios. (ABDEL-BASSET *et al.*, 2019) explicada que Lévy, é uma distribuição matemática que apresenta uma média e variância infinitas causando um movimento muito mais longo de sua posição atual.

(ABDEL-BASSET *et al.*, 2019) também faz a utilização de mapas caóticos que são funções transitivas que retornam uma sequência ilimitada de vários números aleatórios em relação a condição definida inicialmente. Onde na proposta apresentada pelo autor ela é utilizada para determinar o valor de uma probabilidade de comutação.

O fluxo inicial do algoritmo proposto consiste na seguinte forma; i) Primeiramente ocorre geração de uma população inicial de baleias; ii) Em seguida ocorre uma conversão da população inicial para uma sequência de permutação através da função Largest Order Value (LOV), que é utilizada para o mapeamento entre a solução contínua e a ordem decrescente de valores discretos; iii) Em seguida é selecionada a melhor solução entre os valores permutados; iv) Neste passo ocorre as atualizações nas posições das baleias utilizando a distribuição de Lévy juntamente com a utilização do mapa caótico.

As operações seguintes a estas, consistem em atualizações de soluções de forma em espiral como é definida no WOA, ou na escolha de soluções aleatórias. Na última parte do algoritmo ocorre a conversão da população para o LOV, seguida de possíveis mutações ou não.

4 VMUALLOCATOR E CBHSELECTION

Este capítulo visa apresentar a proposta deste trabalho: as novas políticas de alocação (VMUAllocator) e de seleção (CBHSelection) de máquinas virtuais que tem, como objetivo principal, minimizar o consumo de energia elétrica de um *Data center*. As novas políticas apresentadas trazem uma estratégia monocritério com base na CPU. Segundo (DAYARATHNA *et al.*, 2016), o consumo de CPU está intimamente relacionado às ações da memória principal, o que a torna um dos principais componentes do *host* e um dos recursos computacionais que consomem mais energia elétrica, o que justifica a escolha desse recurso como critério de avaliação.

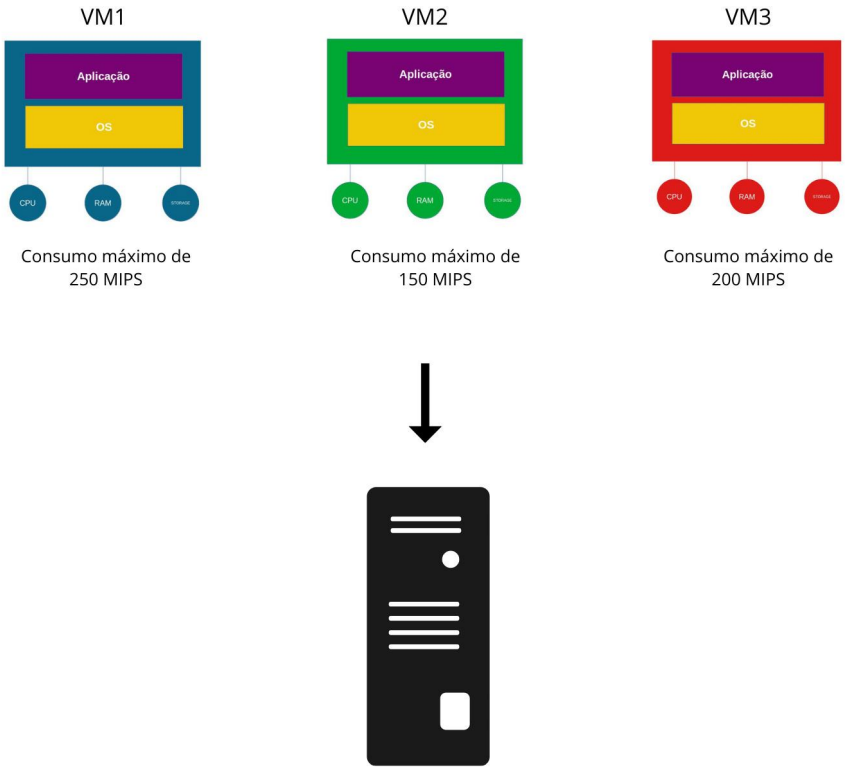
As políticas propostas nesse documento buscam otimizar o processo de alocação e de seleção de VMs, enquanto reduz a quantidade de hosts ativos através da consolidação de máquinas virtuais. Com menos hosts ativos, é possível economizar energia elétrica do *Data center*, porém com um maior número de VMs por *host*, aumenta-se a disputa por recursos computacionais, o que pode aumentar as violações de SLA. Assim, também é objetivo dessas políticas tentar manter uma boa Qualidade de Serviço (QoS), ou seja, procurar manter o número de violações de SLA em um nível aceitável.

Resumidamente, a nova Política de Alocação agrupa VMs em duplas com base em seu consumo de CPU, para facilitar o processo de alocação inicial. Já no processo de realocação a Política de Alocação proposta aloca as VMs individualmente no *host* que possui um espaço mínimo para recebe-la. A nova Política de Seleção, por sua vez, classifica as VMs em quatro grupos com base na análise dos históricos de consumo de CPU: 1) Comportadas; 2) Potencialmente comportadas; 3) Potencialmente problemáticas; 4) Problemáticas. Dependendo da classe da VM, ela poderá ser escolhida para migração. Nas seções a seguir é apresentada a estratégia da Política de Alocação e da Política de Seleção propostas de forma mais detalhada.

4.1 VMUAllocator - *Alocação Inicial*

O processo de alocação inicial da Política de Alocação VMUAllocator foi construída com base em uma estratégia que consiste na formação de duplas de VMs, onde através da análise da soma do consumo máximo da CPU (definido via SLA) de cada VM, é possível separá-las em duplas que podem ser alocadas de acordo com a capacidade máxima dos hosts. A Figura 8 ilustra como a política cria uma dupla de VMs.

Figura 8 – Exemplo de alocação de duplas.

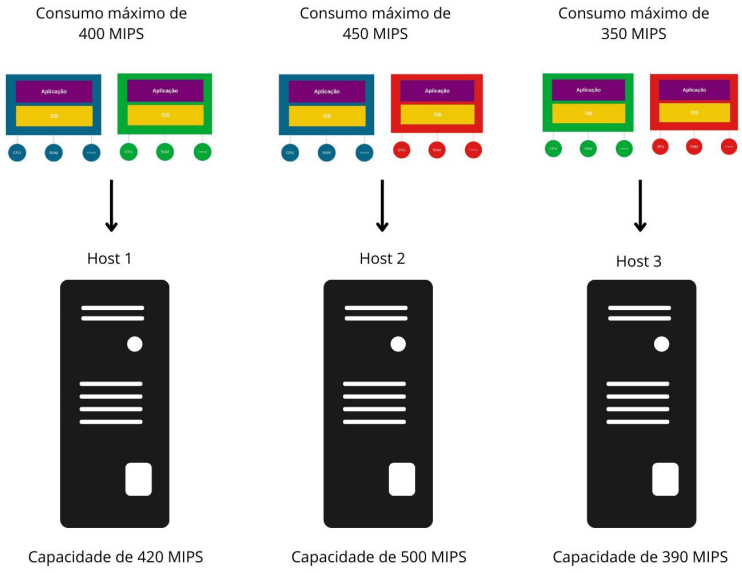


Fonte: O próprio autor

Considere um cenário exemplo onde existem três VMs e um *host*. Dentre as VMs, a VM 1 possui um consumo máximo de CPU de 250 MIPS (*Million Instructions Per Second*), a VM 2 de 150 MIPS e a VM 3 de 200 MIPS, enquanto o *host* possui uma capacidade máxima de 400 MIPS. Note que, para esse *host* em específico, as VMs 2 e 3 podem compor uma dupla, pois a soma dos seus consumos máximos de CPU é igual a 350 MIPS (150 MIPS + 250 MIPS), ou seja, 50 MIPS a menos que a capacidade máxima do *host*. Porém, as VMs 1 e 3 não poderiam ser agrupadas visando esse *host*, pois o somatório de MIPS delas ultrapassa a capacidade do *host* (450 MIPS > 400 MIPS).

Em um *Data center* heterogêneo, há hosts de diferentes tipos, ou seja, com diferentes recursos de hardware e diferentes capacidades máximas de processamento. Então, para cada tipo de *host*, é possível formar diferentes duplas de VMs. Tomando como base o exemplo anterior Figura 8 observe que para cada tipo de *host* presente na Figura 9, é possível criar um grupo diferente de VMs. Por exemplo, a dupla de VMs 1 e 3 que no exemplo anterior não serviam para um *host* de 400 MIPS, agora servem para um *host* de 500 MIPS. É importante deixar claro que a Figura 9 apresenta um exemplo de possíveis duplas, não de duplas formadas para serem alocadas nos servidores. É claro que, uma vez alocada em um grupo, a mesma VM não pode estar em outro grupo. Por exemplo, uma vez a VM 3 está alocada junto com a VM 1 no grupo 2, ela não pode fazer parte do grupo 3 junto com a VM 2.

Figura 9 – Alocações de duplas por tipo de *host* (VM1 e VM2, VM1 e VM3, VM2 e VM3.

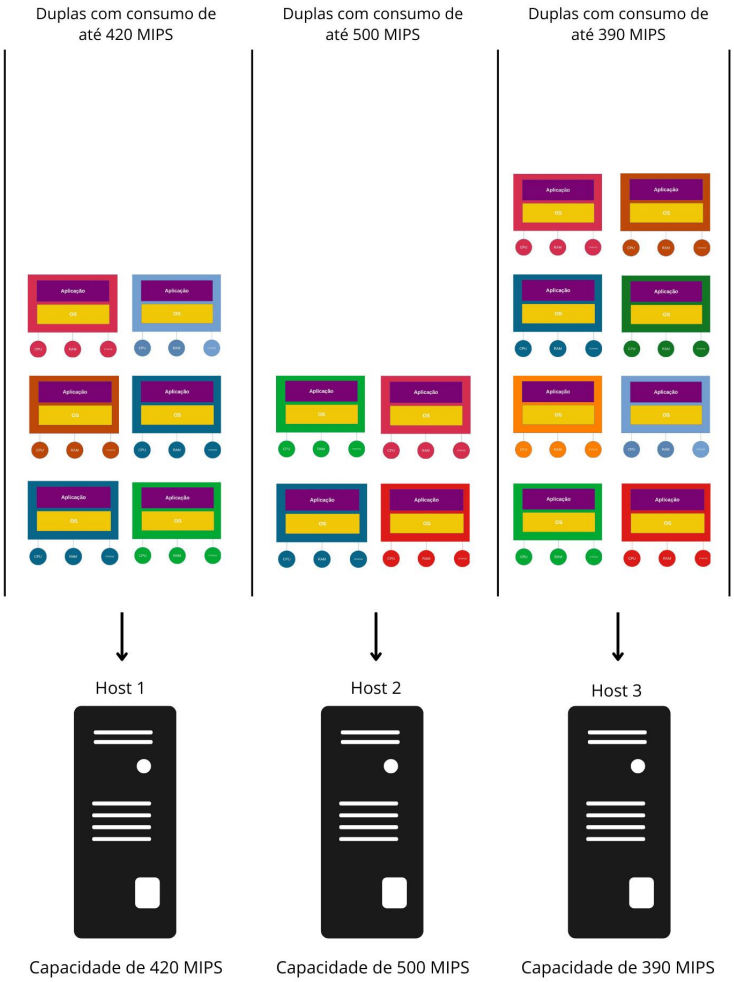


Fonte: O próprio autor

Durante o processo de alocação, existem várias VMs e também muitas possíveis duplas de VMs. Assim, ocasionalmente poderá ocorrer que, para cada tipo de *host*, existam diversas duplas de VMs candidatas a alocação naquele momento. Assim, é necessário um mecanismo que organize essas duplas com o objetivo de facilitar a alocação delas nos hosts apropriados. Dessa maneira, esse trabalho propõe agrupar essas duplas tomando como critério base, a capacidade máxima de CPU da dupla. A Figura 10 ilustra isso ao mostrar três listas de duplas de VMs, onde cada lista é composta por todas as duplas de VMs que possui um consumo máximo de CPU igual ao valor chave determinado na lista. Note que essa estratégia facilita

bastante a alocação das duplas de VMs nos hosts, pois praticamente as mapeia para cada *host* disponível.

Figura 10 – Listas de duplas VMs construídas através da capacidade máxima de cada *host*.



Fonte: O próprio autor

Em termos de implementação, a VMUAllocator apresentada no Pseudocódigo 1 pode ser dividida em três etapas principais: formação das duplas de VMs, reordenação e alocação. Na primeira etapa, as duplas são criadas através do método *generateDuos()* na Linha 1 . Tal método, recebe a lista de VMs a serem alocadas como entrada e implementa em linhas gerais, o processo explicado até agora e exemplificado na Figura 9. O método retorna uma estrutura chave-valor (representada pela variável *hashMap*), onde a chave corresponde a capacidade máxima da dupla de VMs e o valor é uma lista com todas as duplas de VMs cujo a soma dos MIPS é equivalente à chave (semelhante a estrutura mostrada na Figura 10).

Algoritmo 1: VMUAllocator - Initial Allocation

```
   Input: hosts, vms
 1 hashMap ← generateDuos(vms) ;
 2 sortDuos(hashMap) ;
 3 for item in hashMap do
 4     for vms in item.value() do
 5         hostCandidate = findHostForDoubleVms(vms);
 6         if hostCandidate then
 7             for vm in vms do
 8                 hostCandidate.addVm(vm) ;
 9             end
10         end
11     end
12 end
```

Com a estrutura chave-valor (*hashMap*) elaborada, antes de iniciar o processo de alocação, o algoritmo entra na segunda etapa (reordenação) ao ordenar de forma decrescente tal estrutura utilizando as chaves dela como critério (Linha 2). Com a ordenação das chaves da variável *hashMap* é possível definir se a alocação irá iniciar com as duplas de maior consumo(ordem decrescente das chaves) ou menor consumo(ordem crescente das chaves). Explicando melhor, ao final dessa fase a variável *hashMap* estará ordenada de forma decrescente em relação ao consumo máximo de CPU das duplas, ou seja, as listas de duplas que mais consomem MIPS estarão nas primeiras posições, enquanto aqueles que consomem menos MIPS estarão nas últimas.

Após a ordenação, o algoritmo entra na última fase (alocação). Nessa etapa, percorre-se a variável *hashMap* (Linhas 3-12) resgatando de início os grupos, e posteriormente as duplas de VMs de cada grupo (Linhas 4-11) com o intuito de encontrar o *host* mais adequado para receber cada dupla de VM. Para isso, o método *findHostForDoubleVms()* é utilizado (Linha 5). Devido sua relevância para a política, tal método será melhor explicado nos próximos parágrafos.

O código do método *findHostForDoubleVms()* pode ser visualizado no Pseudocódigo 2. Inicialmente, ele recebe como parâmetro a dupla de VMs deverá ser alocada, e de início é feita uma chamada para a função *getHosts()* que traz uma lista com todos os hosts disponíveis, e os salva em uma variável chamada hosts (Linha 1). Em seguida é construída uma Heap Máxima com os hosts que ficará salva em uma variável chamada *heap*, onde o *host* com a maior capacidade

disponível em MIPS ficará na raiz da estrutura, tal abordagem é semelhante a utilizada pela política HeapAllocator (CARVALHO, 2016). O *host* selecionado para a receber a dupla de VMs ficará armazenado na variável *hostAllocable* que é instanciada com valor null (Linha 3)

Algoritmo 2: findHostForDoubleVms

```
   Input: vms
   Output: hostAllocable
 1 hosts ← getHosts();
 2 heap.addAll(hosts);
 3 hostAllocable ← null;
 4 while !heap.isEmpty() do
 5     hostAllocable = heap.poll();
 6     if hostAllocable.isSuitableForVms(vms) then
 7         if isHostOverUtilizedAfterAllocation(hostAllocable, vms) then
 8             continue;
 9         end
10         powerAfterAllocation ← getPowerAfterAllocation(hostAllocable, vms);
11         if powerAfterAllocation ≠ -1 then
12             return hostAllocable ;
13         end
14     end
15 end
16 return hostAllocable; ;
```

O processo de alocação se dará de forma mais simples, porque as duplas com maior consumo serão direcionadas aos hosts com maior capacidade. A cada iteração é removida a raiz da estrutura (Linha 5) e enquanto a mesma não for vazia (Linha 4) o *host* com maior capacidade é direcionado e as iterações que o validam para a receber a duplas de VMs. O *host* escolhido para receber a dupla de VMs ficará salvo na variável *hostAllocable* (Linha 5), e é analisado primeiramente se ele tem recursos suficientes para atender as VMs e se não está com nenhuma falha (Linha 6), através da função *isSuitableForVms()*. E caso o *host* esteja apto, também é necessário verificar se o *host* ficará sobrecarregado ou não (Linha 7) na função *isHostOverUtilizedAfterAllocation()*, e caso ele fique será ignorado e a execução segue para o próximo (Linha 8).

A próxima verificação retorna se a dupla ao ser alocada no *host* excede a quantidade de energia que o *host* suporta (Linha 11), onde a função *getPowerAfterAllocation()* retorna -1 caso isso aconteça. Se a energia não exceder significa que foi encontrado um *host* apto a receber as VMs, e em seguida o mesmo é salvo na variável *hostAllocable* (Linha 13). Ao ser encontrado um *host* apto a receber a dupla de VMs e caso a Heap ainda não esteja vazia as iterações seguem normalmente no laço de repetição (Linha 4), o que pode proporcionar o encontro de um *host* que suporte a alocação da dupla com uma menor capacidade em MIPS já que a ordenação está de forma decrescente, o que é proveitoso porque pode proporcionar uma alocação mais consolidada.

Explicada a função e voltando ao Pseudocódigo 1, o *host* retornado pela função *findHostForDoubleVms()* é armazenado em uma variável chamada *hostCandidate* (Linha 5), em seguida alocará a dupla de VMs (Linha 8) que está em forma de lista (Linha 7). Se não for encontrado nenhum a dupla de VMs serão alocadas separadamente no final do processo da alocação inicial.

4.2 VMUAllocator - *Realocação*

A realocação acontece sempre que um *host* está sobrecarregado ou subutilizado. A política de alocação analisa todos os hosts, e, caso um *host* esteja sobrecarregado, ela realoca VM(s) do *host* em questão para outro(s) *host*(s) enquanto houver sobrecarga. Um procedimento semelhante ocorre com os hosts subutilizados, a diferença é que a política de alocação remove todas as VM(s) hospedadas no *host* subutilizado e o desliga ou o coloca em modo de economia de energia ao final da migração.

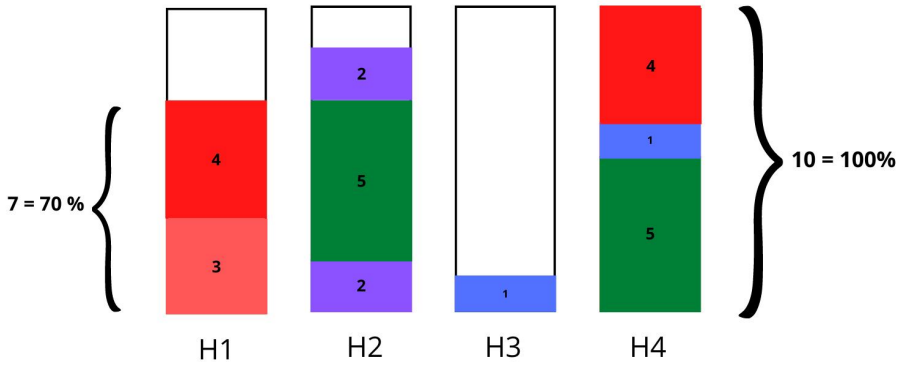
A escolha de quais VMs realocar fica a critério da Política de Seleção. Uma vez selecionadas as VMs, a função de realocação define os hosts mais apropriados para cada uma delas. Como os objetivos da VMUAllocator é economizar energia e reduzir o número de violações de SLA, então o melhor *host* será aquele que resultar em maior economia de energia e uma menor probabilidade de violação de SLA após a realocação.

A estratégia de realocação da política VMUAllocator é diferente da estratégia da alocação inicial, pois na realocação, as VMs são alocadas individualmente. Durante a realocação, a VMUAllocator hospeda cada VM retornada pela Política de Seleção em um *host* que possui o mínimo de CPU disponível para recebê-la, sem gerar sobrecarga. O objetivo da estratégia é consolidar o maior número de máquinas virtuais na menor quantidade de hosts ativos possível, evitando gerar sobrecarga e subutilização nos hosts e mantendo a qualidade do serviço prestado.

Como explicado anteriormente a realocação inicia na identificação de hosts sobrecarregados/subutilizados. No presente trabalho, considerou-se o *host* como sobrecarregado quando o seu nível de uso de CPU ultrapassar 70%. Já os hosts subutilizados são aqueles que atingem uma carga inferior ou igual a metade do mínimo de CPU sobrecarregada, ou seja, 35%. Assim, no presente trabalho, um *host* com consumo normal tem uma porcentagem de utilização de CPU entre 35% e 70%.

Considere o cenário da Figura 12 como exemplo. Nela, há 4 hosts com uma capacidade máxima de CPU de 10 MIPS cada e diferentes níveis de uso momentâneo de CPU. No cenário há também 9 VMs consumindo de 1 à 5 MIPS da CPU do *host* onde ela está hospedada. Para simplificar a visualização, cada VM foi colorida usando o seu nivel de consumo de CPU como critério. Por exemplo, VMs que consomem 2 MIPS foram coloridas em vermelho, enquanto VMs que consomem 5 MIPS foram coloridas em verde.

Figura 11 – hosts com diferentes porcentagens de utilização.

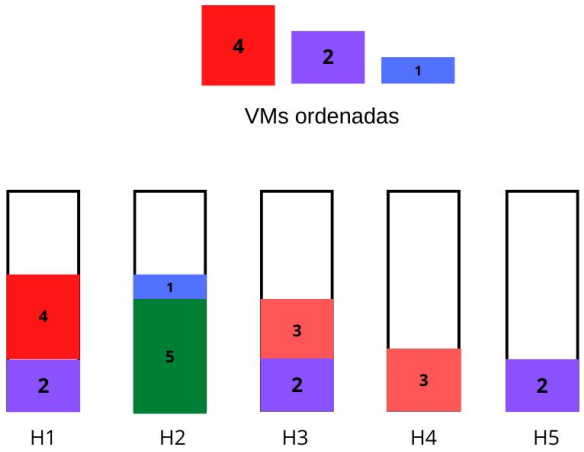


Fonte: O próprio autor

Nos hosts sobrecarregados as VM(s) que são selecionadas para realocação ficarão a critério da Política de Seleção. Nos hosts subutilizados todas as VMs são removidas de imediato para realocação, e em seguida o *host* é desligado ou colocado em modo hibernação.

Considerando um cenário de exemplo (Figura 12), as VM(s) que serão selecionadas para realocação serão as VM(s): vermelha de tamanho 4, roxa de tamanho 2 e uma azul de tamanho 1. Inicialmente, a estratégia de realocação recebe as VMs e as ordena de forma decrescente com base na requisição de consumo atual de CPU de cada uma. Simultaneamente, os hosts que irão a receber VMs também são ordenados de forma decrescente em relação ao seu consumo atual de CPU.

Figura 12 – Ordenação decrescente das VMs e dos hosts em relação ao seus atuais consumos de CPU.



Fonte: O próprio autor

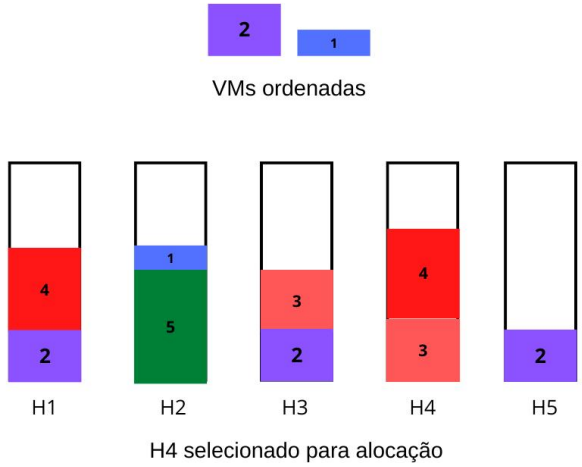
Esse processo visa facilitar o encontro do espaço mínimo para alocar uma VM. As VMs que mais requisitam CPU serão as primeiras a serem realocadas por conta da ordenação decrescente. Assim, cada VM que será realocada irá ser direcionada ao o *host* com maior consumo de CPU, analisando se o mesmo tem um espaço mínimo para alocar a mesma.

Caso o *host* em questão não tenha espaço para alocar a VM, a mesma será direcionada para o próximo *host* da lista, até encontrar um *host* em que seja possível a alocação. Dessa forma o processo de consolidação acontecerá de forma mais facilitada e proporcionará uma maior incidência de hosts subutilizados que posteriormente serão desligados.

Na primeira iteração, ocorrerá a alocação da VM vermelha de tamanho 4 e, para isso, todos os hosts são ordenados de forma decrescente de acordo com seu consumo atual de CPU. No primeiro momento, tenta-se alocar a VM vermelha no *host* H1. Contudo, se realizada a alocação, H1 ficará sobrecarregado por ultrapassar 70% de consumo, o que não torna possível a alocação. Seguindo a sequência, o mesmo ocorrerá ao tentar alocar nos hosts H2 e H3.

A tentativa seguinte é no *host* H4 que possui o espaço requisitado e é automaticamente selecionado para receber a VM vermelha (Figura 13). Como já explicado na estratégia definida, alocar no espaço mínimo disponível. O *host* H5 poderia facilmente receber a VM, porém ficaria com um espaço disponível de 1 MIPS o que não seria uma boa alternativa em termos de consolidação no exemplo.

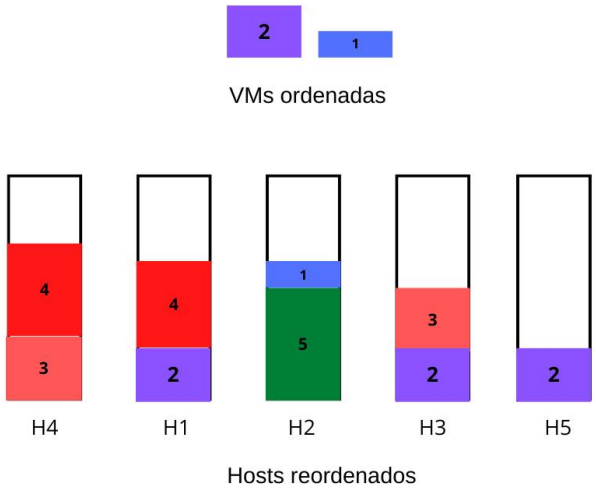
Figura 13 – *host* H3 selecionado para receber a VM.



Fonte: O próprio autor

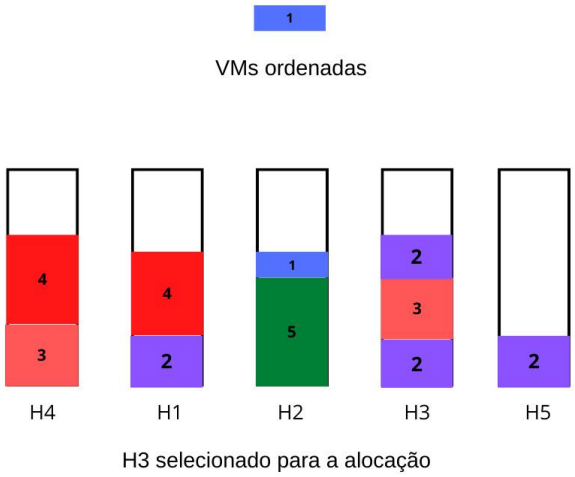
A próxima VM que irá ser alocada será a VM roxa, e antes de iniciar a busca do *host* mais adequado, é feita novamente a reordenação dos hosts em ordem decrescente (Figura 14). Na figura, o primeiro *host* (H4) está sem espaço para receber novas VMs, o que ocorre também no hosts seguintes(H1, H2). Na sequência o *host* H3 é previamente selecionado para alocar a VM roxa (Figura 15).

Figura 14 – Reordenação dos hosts.



Fonte: O próprio autor

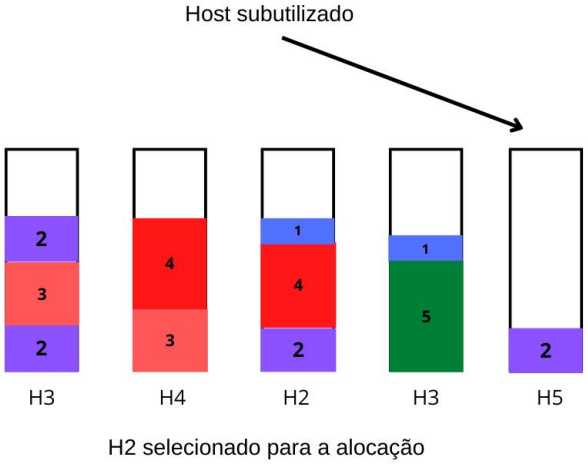
Figura 15 – H3 recebendo a VM roxa.



Fonte: O próprio autor

A VM seguinte que irá ser alocada será a VM azul, e antes do processo ocorre a reordenação dos hosts em ordem decrescente (Figura 16). No *host* H3 e H4 não possuem espaço disponível para a alocação da VM. Já o *host* H2 na sequência possui o espaço disponível para a alocação então o mesmo é selecionado(Figura 16). No final do processo busca-se ter uma boa parte dos hosts consolidados, o que irá promover a ocorrência de hosts subutilizados como ilustrado na figura, e onde posteriormente as VMs restantes desse *host* subutilizado serão migradas e ele será desligado. O que proporciona uma minimização dos hosts ativos, reduzindo o consumo de energia no *Data center*.

Figura 16 – Resultado final da realocação.



Fonte: O próprio autor

A estratégia de realocação a nível implementação é destacada no Pseudocódigo 3, onde a função recebe uma lista de VMs que irão ser alocadas através do parâmetro *vmsToMigrate* e uma lista com os hosts desligados excludeHosts. As VMs inicialmente são reordenadas em ordem decrescente em relação aos seus atuais consumos de CPU em MIPS (Linha 1).

Com as VMs ordenadas inicia-se o processo de alocação, onde é feito o percorrimento por cada VM contida na lista (Linha 2). Para buscar o *host* que irá alocar a VM, é invocada a função *findHostForVm()*, que recebe a VM que deverá ser alocada e a lista com hosts desligados (Linha 3).

Algoritmo 3: VMUAllocator - Relocation

Input: vmsToMigrate, excludedHosts

```
1 vmsToMigrate.sort();
2 for vm in vms do
3     hostAllocable = findHostForVm(vm, excludedHosts) ;
4     if hostAllocable then
5         hostAllocable.addVm(vm) ;
6     end
7 end
8 return hashMap ;
```

O Pseudocódigo 4 contém a lógica utilizada na função *findHostForVm()*, que é parecida com estratégia da alocação inicial em duplas, porém no processo de realocação ocorre com VMs individuais. No início do processo é feita a invocação da função *getHosts()* que acessa a lista com todos os hosts do *Data center*(Linha 1) e os salva em uma variável chamada hosts. Em seguida a lista de hosts é adicionada em uma Heap Máxima que ordena os hosts com as maiores utilizações de CPU em MIPS(Linha 2), identificada pela variável *heap*.

Algoritmo 4: findHostForVm

Input: vm, excludedHosts

Output: host

```
1  hosts ← getHosts() ;
2  heap.addAll(hosts) ;
3  hostCandidate ← null ;
4  while !heap.isEmpty() do
5      hostCandidate = heap.poll() ;
6      if excludedHosts.contains(hostCandidate) then
7          continue;
8      end
9      if hostCandidate.isSuitableForVm(vm) then
10         if isHostOverUtilizedAfterAllocation(hostCandidate, vm) then
11             continue ;
12         end
13         powerAfterAllocation ← getPowerAfterAllocation(hostCandidate, vm) ;
14         if powerAfterAllocation ≠ -1 then
15             return hostCandidate ;
16         end
17     end
18 end
19 return null;
```

Assim como na alocação inicial a raiz da *heap* irá sendo removida e salva (Linha 5) enquanto a heap não for vazia(Linha 4). A raiz é salva em uma variável chamada *hostCandidate*, onde posteriormente será analisada para uma possível alocação para a VM que foi recebida por parâmetro.

A primeira verificação na busca de um *host* válido para a alocação de uma VM, é analisar se o mesmo está contido na lista dos hosts desligados (Linha 6). Caso esteja, a execução segue para o próximo *host* contido na heap, se não o *host* segue para as próximas validações. A próxima verificação analisa se o *host* em questão tem recursos suficientes para atender a VM, e se não está com nenhuma falha através da função *isSuitableForVm()*(Linha 9). Caso seja válido, é verificado também se o *host* não ficará sobrecarregado após uma possível alocação(Linha 10),

caso esteja a execução segue para o próximo *host* (Linha 11). A última verificação retorna se a VM ao ser alocada no *host* excede a quantidade de energia que o mesmo suporta através da função *getPowerAfterAllocation()* (Linha 13). Caso exceda a função retorna -1, caso contrário, foi encontrado um *host* válido para a alocação e em seguida o mesmo é retornado pela função.

Explicada a função e voltando ao Pseudocódigo 4, caso seja encontrado um *host*(Linha 5) a VM é alocada no mesmo. Caso não seja encontrado um *host*, a VM continua no *host* de origem do qual foi selecionada pela a Política de Seleção.

4.3 CBHSelection

A Política de Seleção desenvolvida visa classificar VMs com o objetivo de escolher a mais indicada para ser migrada de um servidor para outro. A classificação é feita através da análise do histórico de consumo de CPU das VMs (em MIPS) e considera, no máximo, as trinta execuções mais recentes para identificar se cada VM possui uma média e uma variância de consumo alta ou baixa. Assim, cada VM pode ser classificada em um dos seguintes grupos: 1) Comportadas (Variância Baixa - Média Baixa); 2) Potencialmente comportadas (Variância Baixa - Média Alta); 3) Potencialmente problemáticas (Variância Alta - Média Baixa); 4) Problemáticas (Variância Alta - Média Alta).

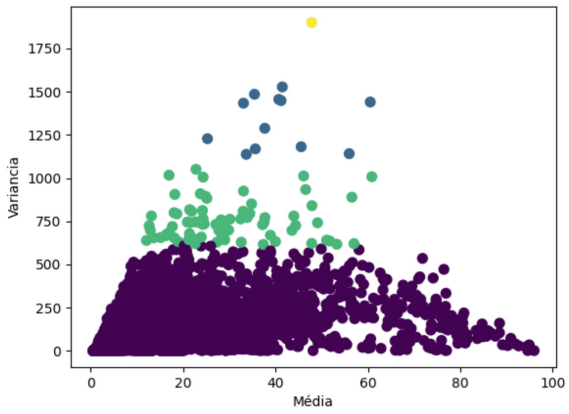
Com base nos valores da variância e da média de consumo das VMs é possível identificar aquelas que são mais ou menos problemáticas para o servidor que as hospeda. As VMs mais problemáticas são aquelas que, ou consomem uma grande quantidade de recursos computacionais (em média) ou possuem alta variância no consumo desses mesmos recursos. Acredita-se que VMs com tais características podem, potencialmente provocar mais condições de disputa no servidor, pois em servidores muito consolidados as VMs mais problemáticas podem abruptamente solicitar uma quantidade significativa de recursos computacionais, o que pode ocasionar em falta de recursos para as demais VMs. Com uma maior disputa pelos recursos do servidor, consequentemente a tendência é observar um maior número de violações de SLA. As VMs mais problemáticas são aquelas classificadas nos grupos 2, 3 e 4 supracitados.

Já as VMs menos problemáticas ou potencialmente comportadas têm o comportamento contrário, ou seja, possuem baixo consumo de recursos e baixa variância, o que proporciona maior controle sobre o consumo delas e uma maior garantia de que não ocorrerão graves violações de SLA. VMs menos problemáticas são aquelas classificadas no grupo 1 supracitado.

Para distribuir as VMs nos grupos supracitados e classificá-las como mais ou menos

problemáticas, é necessário definir limitares que separem as médias e as variâncias de consumo das VMs em alta e baixa. Para isso, foi realizado o seguinte procedimento: Inicialmente, escolheu-se dois valores que representassem um intervalo de possíveis médias e mais dois valores que representassem o intervalo de possíveis variâncias com base na densidade de dados contidos no gráfico da Figura 17, gerado com base nos *traces* disponibilizados pelo próprio CloudSim. A natureza desses traces serão melhores explicados em parágrafos posteriores. Os intervalos escolhidos foram de 0 a 40 (média) e de 0 a 500 (variância).

Figura 17 – Gráfico de dispersão.



Fonte: O próprio autor

Os limiares foram então determinados empiricamente tomando como base os intervalos escolhidos previamente. Assim, tais intervalos foram usados em testes preliminares, onde os limiares (média e variância) que provocassem um menor consumo de energia na simulação e a menor violação de SLA seriam os escolhidos para a definição dos grupos.

Os valores dos intervalos da média e variância foram testados um a um, ou seja, um valor x pertencente ao intervalo da média e um valor y pertencente ao intervalo da variância. Foi testada cada possibilidade dos intervalos nas simulações salvando os resultados simultaneamente. A simulação que gerou o menor consumo de energia e a menor violação de SLA foi com os limares 40 e 350, média e variância respectivamente.

Através do gráfico foi possível observar onde havia VMs mais dispersas ou mais concentradas em certos intervalos, o que proporcionou a criação de limiares a fim de definir VMs com variância baixa e média alta, ou com variância alta e média baixa e assim por diante. Essa análise possibilitou a criação de 4 tipos de VMs como ilustrado na Tabela 1.

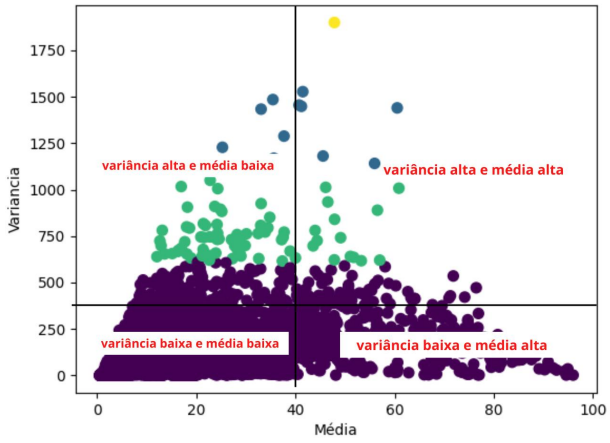
Tabela 1 – Tipos de VM e valor associado

Média	Variância	Tipo de VM
baixa	baixa	0
baixa	alta	1
alta	baixa	2
alta	alta	3

Fonte: O próprio autor

Na ilustração da Figura 18 foi utilizado dois limiares para a classificação, o primeiro limiar é 40 e o segundo é 350. Com isso, foi possível a criação de "quadrantes" onde estavam as VMs com variância alta e variância alta e média baixa, variância alta e média alta, variância baixa e média baixa e variância baixa e média alta. Através da análise gráfica e com os valores associados a cada tipo de VM foi feita a escolha de seleção de VMs do tipo 3 (variância alta e média alta), por se tratar de dados mais dispersos em relação aos outros tipos e também por indicar uma grande oscilação em relação ao consumo de CPU por parte desse tipo de VM.

Figura 18 – Gráfico de dispersão com divisões por limiares.



Fonte: O próprio autor

Se identificada uma única VM como problemática, então ela prontamente é retornada pela Política de Seleção como a VM a ser migrada. Porém, no servidor, pode haver mais de uma VM desse tipo. Nesse caso específico, quando mais de uma VM é classificada como problemática, a política seleciona a VM com o menor consumo de CPU mais recente no histórico. Optou-se por essa estratégia porque ao remover uma VM com esse perfil, aumenta a probabilidade da mesma

ser alocada em outro *host* ao ser migrada, pois ela consome menos recursos computacionais naquele momento específico. Se não existir nenhuma VM do tipo 3 não ocorrerá migração de VMs.

No Pseudocódigo 5 é apresentada a estratégia aqui descrita. Lembrando que no CloudSim a Política de Seleção é responsável apenas em escolher as VMs a serem migradas de hosts sobrecarregados e a realocação efetiva dessas VMs em outros hosts é de responsabilidade da Política de Alocação.

Algoritmo 5: CBHSelection

```
   Input: host
   Output: vm
1  vmSelected ← null;
2  vmsProblematics ← [ ];
3  for vm in host.getVmList() do
4  |  classification = classifier(vm);
5  |  if classification == 3 then
6  |  |  vmsProblematics.add(vm);
7  |  end
8  end
9  if vmsProblematics.size > 0 then
10 |  vmsProblematics.sort();
11 |  vmSelected ← vmsProblematics[0];
12 end
13 return vmSelected;
```

Na Linha 1, é instanciado um atributo *vmSelected* que irá armazenar a VM selecionada para a migração. Na Linha 2 também é instanciada uma lista vazia que irá receber todas as VMs problemáticas identificada como *vmsProblematics*. De início, o algoritmo percorre a lista de VMs contida no servidor (Linhas 3 à 8) e, para cada VM, é feita uma chamada a função *classifier()* que retorna a classificação da VM (Linha 4). Os detalhes da função *classifier()* se encontram no Pseudocódigo 6 e serão discutidos no próximo parágrafo. Caso a VM analisada seja problemática, ou seja, do tipo 3 (Linha 5), ela é salva na lista *vmsProblematics* (Linha 6). Ao final, é analisado se o servidor em questão possui VMs do tipo 3 ou não (Linha 9). Caso positivo, as VMs são ordenadas de forma crescente com base nos seus consumos de CPU em

MIPS (Linha 10). Com as VMs ordenadas, é selecionada a primeira VM lista que, obviamente, é aquela que possui o menor consumo de MIPS (Linha 11). Finalmente, a mesma é retornada pela função (Linha 12).

Algoritmo 6: classifier

Input: vm

Output: 0, 1, 2, 3

```
1  historicMips ← vm.getHistory();
2  threshold_1 ← x;
3  threshold_2 ← y;
4  mean = calcMean(historicMips);
5  variance = calcVariance(historicMips);
6  if mean <= threshold_1 and variance <= threshold_2 then
7  |   return 0;
8  end
9  if mean <= threshold_1 and variance > threshold_2 then
10 |   return 1;
11 end
12 if mean > threshold_1 and variance <= threshold_2 then
13 |   return 2;
14 end
15 if mean > threshold_1 and variance > threshold_2 then
16 |   return 3;
17 end
```

A função *classifier* destacada no Pseucódigo 6 recebe a VM em análise e resgata o histórico de execuções em MIPS dela em uma variável chamada *historicMIPS* (Linha 1), onde posteriormente será feito o cálculo da média e variância nas Linhas 4 e 5, respectivamente. Nas Linhas 2 e 3 é definido a dupla de limiares inteiros para que seja feita a classificação da VM no trecho com base nas estruturas condicionais entre as Linhas 6 a 16.

5 AMBIENTE DE TESTES

Após a apresentação das Políticas de Alocação e seleção propostas nesse trabalho, é necessário avaliar o desempenho delas em um ambiente de *Data center*. Assim, foram conduzidos diversos experimentos simulados que compararam as políticas *VMUAllocator* e *CBHSelection* à algumas das principais Políticas de Alocação e seleção de máquinas virtuais disponíveis na literatura. Esse capítulo apresenta uma descrição detalhada sobre os testes realizados, com foco especial em: 1) Introdução sobre o simulador, 2) Configurações gerais do cenário, 3) Definição da métricas de avaliação, 4) Configuração da máquina utilizada para executar o simulador.

5.1 O simulador

Todos os experimentos foram conduzidos através de simulações realizadas no CloudSim 3.0.3 (CALHEIROS *et al.*, 2011), uma ferramenta bastante utilizada pela literatura para simular ambientes de Computação em Nuvem e avaliar o desempenho de novas Políticas de Alocação e/ou de seleção de máquinas virtuais e, recentemente, também de instâncias virtuais mais leves como containers. A versão 3.0.3 do simulador foi adotada devido ao trabalho de (MANN; SZABÓ, 2017). Esse trabalho foi escolhido como referência porque ele já possuia Políticas de Alocação implementadas que poderiam ser adotadas nos testes comparativos, além de conduzir testes em um cenário com especificações semelhantes a um *Data center* convencional, como as configurações das VMs, hosts e tarefas com perfis de consumo real.

Em um ambiente típico no CloudSim há cinco componentes principais:

- ***Data center***: Segundo (CISCO, 2022), um *Data center* é uma instalação física onde as empresas hospedam aplicativos e dados essenciais. No simulador, cada instância desse componente é composto por uma lista de hosts, que podem ser heterogêneos entre si ou não.
- ***Hosts***: Trata-se de uma máquina física, conectada a uma rede e oferecendo recursos, informações e/ou serviços variados aos usuários. No simulador, um *host* hospeda de zero à N VMs, onde N é limitado pela recursos disponibilizados pelo *host*, ou seja, o *host* aceitará VMs enquanto ele for capaz de suportar as demandas das VMs pelo hardware dele. Dependendo do valor de N, o *host* pode entrar em um estado de sobrecarga ou subutilização de recursos
- ***VM***: A VM é uma abstração de uma máquina física, trabalhando de forma

isolada dentre de um *host*. No simulador, uma VM é responsável por receber tarefas (Cloudlets) e por simular a execução delas no *host*. Para criar uma VM no emulador, deve-se configurar a sua capacidade de armazenamento (memória principal e secundária) e de processamento (poder computacional da CPU em MIPS), além do identificador do broker relacionado a VM, entre outros. Para simular a execução de uma Cloudlet a VM possui um método (CloudletScheduler) para agendar a execução de cada Cloudlet. Ao executar as Cloudlets as VMs podem apresentam uma porcentagem de consumo de CPU entre os valores 0 e 1.

- ***Cloudlet***: A Cloudlet representa uma tarefa requisitada por um usuário para ser processada por uma VM. Dependendo da configuração inserida na implementação do simulador, diferentes parâmetros serão considerados durante a sua instanciação, como prazo, duração e hora de chegada. No simulador, cada instância é responsável por gerenciar a execução de uma tarefa na VM onde ela está alocada. Por exemplo, uma Cloudlet é responsável por definir a quantidade de recursos requisitados para simular a computação da tarefa em um determinado momento (CALHEIROS *et al.*, 2011).
- ***Broker***: O Broker atua como um intermediário encaminhando as solicitações dos usuários e o datacenter. Para cada usuário no datacenter existe um Broker, ou seja, a quantidade de usuários e Brokers são equivalentes. No simulador, o Broker distribui as Cloudlets entre as VMs disponíveis, além de submeter as VMs para serem executadas no Datacenter.

Em geral, para executar uma simulação no CloudSim é necessária realizar algumas ações prévias para configurar os componentes supracitados:

1. Inicialmente deve-se instanciar os hosts. Para isso, é necessário criar uma lista de hosts, onde cada *host* é definido com base em parâmetros como a capacidade máxima de Disco (em *Gigabytes*), de RAM (em *Megabytes*) e de CPU (em MIPS), bem como o padrão de consumo energético da máquina simulada;
2. Em seguida, deve-se criar as VMs. Para isso, é exigido definir configurações de cada VM, como a capacidade máxima de CPU (em MIPS), de Disco (em *Gigabytes*), de RAM (em *Megabytes*), entre outros.
3. Em seguida, deve-se inicializar as *Cloudlets*. Inicialmente, são definidas algumas características de cada *Cloudlet*, como o prazo para o término de execução, a duração, a hora de

chegada e o número de CPUs que serão utilizadas para executá-la.

4. Em seguida, é necessário criar o *Broker* que mediará a comunicação entre o provedor de serviço (representado pelo *Datacenter*) e os usuários. Para isso, é necessário informar ao *Broker* todas as VMs e *Cloudlets* que foram criadas nos dois passos anteriores ao *Broker*.
5. O passo seguinte consiste em associar *Cloudlets* e VMs para que as VMs possam simular as execuções das tarefas solicitadas pelo usuário. Essa ação é realizada internamente pelo *Broker*.
6. Para adicionar Políticas de Alocação de seleção de VMs, inicialmente, deve-se desenvolver a Política de Seleção, e, para isso, deve-se criar uma classe que estende a classe *PowerVmSelectionPolicy* da biblioteca do CloudSim. Ao estender essa classe, a nova classe deverá implementar todos os métodos abstratos necessários para construir uma Política de Seleção. Em seguida é feita a criação da sua instância. Essa monografia, contribui com a literatura ao implementar a classe *CHBSelection*.
7. No passo seguinte, deve-se desenvolver a Política de Alocação, onde é preciso criar uma classe que estende *PowerVmAllocationPolicy* presente na biblioteca do CloudSim. Assim, a Política de Alocação irá implementar todos métodos necessários para criação da Política de Alocação. A sua instância receberá a lista de VMs e a lista de Hosts, junto a Política de Seleção criada e a porcentagem de utilização máxima de um *host* pra identificar possíveis sobrecargas. Essa monografia, contribui com a literatura ao implementar a classe *VMUAllocation*.
8. O último passo consiste em criar o *Data center*. A sua instância irá receber a Política de Alocação, e outras características do data center como sistema operacional, arquitetura do sistema entre outros.

5.2 Configuração Geral do Cenário

O Passo 1 requer a inicialização de uma lista de servidores (hosts). Assim, foram configurados 800 servidores, sendo metade deles do tipo HP ProLiant G5 e a outra metade do tipo HP ProLiant G4. Os dois tipos de servidores de uma forma geral têm as mesmas especificações, exceto em relação a capacidade máxima de processamento por núcleo, onde o HP ProLiant G5 possui 2660 MIPS e o HP ProLiant G4 possui 1860 MIPS. A Tabela 2 apresenta de forma detalhada as especificações para cada tipo de servidor.

Tabela 2 – Configurações dos servidores utilizados

Recurso	HP ProLiant G4	HP ProLiant G5
Número de Núcleos	2	2
Processamento	1860 MIPS	2660 MIPS
Memória Primária	4 GB	4 BG
Memória Secundária	1 TB	1 TB
Largura de Banda	1 Gbit/s	1 Gbit/s
Qtde de servidores	400	400

Fonte: (BELOGLAZOV; BUYYA, 2012)

Além da definição das configurações de hardware dos servidores, também é necessário determinar o consumo energético de cada um. O consumo energético é computado através da carga de trabalho da CPU em um determinado período, ao passo que o nível de consumo de CPU é mensurado pela porcentagem de trabalho. A Tabela 3 apresenta a relação da porcentagem da carga de trabalho e consumo de energia para cada tipo de servidor adotado nos experimentos. Por exemplo, com base na Tabela 3, é possível identificar que o consumo energético de um servidor HP G4 que apresentou uma carga de trabalho de 20% é de 92.6 Watts.

Tabela 3 – Consumo de energia por carga de trabalho

Servidor	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP G4 (W)	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP G5 (W)	93.7	97	101	105	110	116	121	125	129	133	135

Fonte: (BELOGLAZOV; BUYYA, 2012)

Contudo, a Tabela 3 não apresenta os valores de consumo energético para cargas de trabalho intermediárias, como 15%. Para calcular o consumo energético nesses casos, adota-se a Fórmula 5.1 que através da interpolação dos valores mais próximos ao valor desejado, calcula o consumo energético correspondente.

$$y = y_1 + \left[\left(\frac{x - x_1}{x_2 - x_1} \right) (y_2 - y_1) \right] \tag{5.1}$$

Fonte: (BELOGLAZOV; BUYYA, 2012)

Para utilizar a fórmula, primeiro obtém-se o percentual da carga de trabalho alvo (X). Feito isso, são selecionados os percentuais de carga de trabalho mais próximos à X. Dessa maneira, obtém-se: X_1 (a porcentagem de carga de trabalho menor e mais próxima a X) e X_2 (a porcentagem de carga de trabalho maior e mais próxima a X). Também são obtidos os consumos energéticos referentes a X_1 e X_2 (Y_1 e Y_1) respetivamente. De posse desses valores, aplica-se a Fórmula 5.1 e obtém-se Y, o valor do consumo de energia que está sendo buscado em Watts.

Considere como exemplo da aplicação da Fórmula 5.1 a busca do consumo energético de um servidor HP G4 que apresentou uma carga de trabalho de 25%. Inicialmente, identifica-se X como 25%. Em seguida, obtém-se os valores de carga de trabalho (X_1 = 20% e X_2 = 30%) e de consumo de energia (Y_1 = 92,6 Watts e Y_2 = 96 Watts) mais próximos a 25%. Encontrados os valores, aplica-se a fórmula citada e encontra-se o valor de 94,3 Watts como consumo energético relacionado a carga de trabalho de um servidor HP G4 com uma carga de trabalho de 25%.

Configurados os servidores do *Data center*, o próximo passo (Passo 2) é criar N máquinas virtuais que serão hospedadas por eles. Assim, foram utilizadas quatro instâncias de VMs disponibilizadas pela empresa Amazon EC2[1]: Instância Média Otimizada Para Computação (*High-CPU Medium Instance*), Instância Extra Grande (*Extra Large Instance*), Instância Pequena (*Small Instance*) e Microinstância (*Micro Instance*). As especificações das VMs adotadas são apresentadas na Tabela 4.

Tabela 4 – Especificações das VMs

Recurso	High-CPU Medium	Extra Large	Small	Micro
Qtde. de núcleos	1	1	1	1
Processamento	2500 MIPS	2000 MIPS	1000 MIPs	500 MIPS
Memória Primária	870 MIPS	1.7 MIPS	1.7 MIPs	613 MIPS
Tamanho da VM	2.5 GB	2.5 GB	2.5 GB	2.5 GB
Largura de banda	10 Mbits/s	10 Mbits/s	10 Mbits/s	10 Mbits/s

Fonte: (BELOGLAZOV; BUYYA, 2012)

O próximo passo na configuração do ambiente simulado (Passo 3) é definir a carga de trabalho de cada Cloudlet durante a simulação. Foram usados três *traces* de dados com o comportamento de aplicações reais em um ambiente de Nuvem:

1. ***PlanetLab***: Seus dados foram extraídos de um *cluster* que continha cargas de trabalho de dez dias de simulação, onde para cada dia ocorria uma alteração na quantidade de VMs utilizadas. Para os experimentos realizados nesse trabalho, foram considerados somente o primeiro dia desses dez dias;
2. ***BitBrains***: Seus dados foram extraídos de um *cluster* que continha cargas de trabalho de quatro meses de execução. Para a simulação das políticas propostas foi utilizado as cargas de trabalho desse período de simulação.
3. ***GoogleCluster***: Apresenta cargas de trabalho de execução de 29 dias de simulação. Para a simulação das políticas propostas foi utilizado as cargas de trabalho desse período de simulação.

Algumas especificações dos ambientes citados são apresentados na Tabela 5. Cada ambiente possui características particulares em seus *traces* de carga. A característica comum a todos eles é o processador, nos outros casos apenas alguns fazem o rastreio em disco e memória. Para a realização dos experimentos, apenas os dados de CPU são relevantes para a simulação. Por conta disso, os dados dos demais recursos (Memória RAM e Disco) foram desconsiderados da análise.

Tabela 5 – Resumo dos rastreamentos de carga de trabalho do mundo real utilizados

Origem	Virtualizado	CPU	Memória	Disco
PlanetLab	sim	sim	não	não
Google	não	sim	sim	sim
Bitbrains	sim	sim	sim	não

Fonte: O próprio autor

Finalmente, o último passo para a configuração do cenário (Passo 4) é definir as Políticas de Alocação e de seleção que serão utilizadas durante a alocação e a realocação das máquinas virtuais no *Data center*. Além das políticas definidas nessa monografia (VMUAllocator e CBHSelection), também foram selecionadas algumas Políticas de Alocação e de seleção já propostas pela literatura.

Para execução das simulações e análise de desempenho da Política de Alocação proposta (VMUAllocator): 1) Definiu-se (FILHO, 2018) como fonte de Políticas de Alocação porque foi executado no mesmo cenário utilizado do presente trabalho e apresenta os resultados das Políticas de Alocação da literatura. 2) Foram selecionadas duas Políticas de Alocação que apresentaram os menores consumos energéticos no trabalho de (FILHO, 2018):*GuazzoneBFD* (GUAZZONE *et al.*, 2012) e *AbsoluteCapacity* (SHI *et al.*, 2013). A proposta de (FILHO,

2018), a *MSIAllocator*, também foi selecionada em um primeiro momento. Contudo, devido ao seu tempo de execução bastante elevado (chegando a 16 horas em um único cenário), ela foi desconsiderada nos experimentos. 3) Também foi desenvolvida uma política chamada *RandomAllocation* (próprio autor) para ser utilizada nas análises comparativas de desempenho como um *baseline*. A política *baseline* servirá como uma base em relação ao seu desempenho como a política mais simples. A Tabela 6 apresenta todas as políticas que foram utilizadas nos testes. Segue uma breve explicação sobre as Políticas de Alocação concorrentes:

1. ***AbsoluteCapacity* (SHI *et al.*, 2013)**: É uma Política de Alocação que ordena os servidores em ordem decrescente em relação a sua capacidade total de processamento, retornando sempre o servidor de maior capacidade.
2. ***GuazzoneBFD* (GUAZZONE *et al.*, 2012)**: Política de alocação que utilizou a heurística *BFD* (Best Fit Decreasing) para definir a alocação de VMs. Na estratégia, os servidores ativos são priorizados e são ordenados de forma decrescente quanto a capacidade de CPU livre. Assim, a política sempre retorna o servidor ativo que possui a CPU mais ociosa.
3. ***RandomAllocation* (próprio autor)**: Essa Política de Alocação seleciona um servidor aleatório apto receber a VM a ser alocada.

Para análise de desempenho da Política de Seleção (CBHSelection) proposta nesse trabalho foi utilizada uma metodologia semelhante a análise da Política de Alocação (VMU-Allocation): 1) Definiu-se (MORAIS, 2019), como fonte das Políticas de Seleção, porque foi executado no mesmo cenário utilizado do presente trabalho e apresentava os resultados das Políticas de Seleção da literatura. 2) Do trabalho de (MORAIS, 2019), foram escolhidas as duas Políticas de Seleção que obtiveram melhor desempenho em termos de consumo de energia, *AHPSelection* (MORAIS, 2019) e *Minimum Utilization* (BELOGLAZOV; BUYYA, 2010), para serem utilizadas para comparativo de desempenho. 3) Também foi utilizada uma Política de Seleção elementar, a *RandomSelection* (BELOGLAZOV; BUYYA, 2012), como *baseline*.

1. ***MinimumUtilization* (BELOGLAZOV; BUYYA, 2010)**: Política de seleção que escolhe a VM com menor uso de CPU, buscando minimizar o aumento potencial de energia e as violações de SLA.
2. ***AHPSelection* (MORAIS, 2019)**: Política de seleção que estrutura o processo de escolha de uma VM para migração como problema de tomada de decisão, onde é aplicada uma técnica chamada *Analytic Hierarchy Process* (AHP).
3. ***RandomSelection* (BELOGLAZOV; BUYYA, 2012)**: Política de seleção que escolhe

aleatoriamente uma VM para ser migrada de um servidor.

Tabela 6 – Políticas de Alocação e Seleção adotadas

Nome da política	Sigla	Referência
AHPSelection	AHP	(MORAIS, 2019)
AbsoluteCapacity	ABS	(SHI et al., 2013)
CBHSelection	CBH	(próprio autor)
GuazzoneBFD	GUA	(GUAZZONE et al., 2012)
Minimum Utilization	MU	(BELOGLAZOV; BUYYA, 2010)
RandomAllocation	RA	(próprio autor)
RandomSelection	RS	(BELOGLAZOV;BUYA; 2012c)
VMUAllocation	VMU	(próprio autor)

Fonte: O próprio autor

Para a executar a simulação de uma Política de Alocação no CloudSim é necessária a definição de uma Política de Seleção para executarem juntas. O mesmo vale para a Política de Seleção, ou seja, ela deverá ser executada junto a uma Política de Alocação.

Para permitir a execução das Políticas de Alocação da literatura e a Política de Alocação proposta neste trabalho, foi definida uma Política de Seleção única para executar em conjunto com cada uma delas. Tal estratégia foi definida para manter uma simulação mais homogênea entre as Políticas de Alocação utilizando a mesma Política de Seleção com as Políticas de Alocação. A Política de Seleção escolhida foi a política *AHPSelection* que teve o melhor desempenho em termos de consumo em energético do trabalho de (MORAIS, 2019) para proporcionar simulações com o melhor desempenho possível. A mesma estratégia foi utilizada para executar as Políticas de Seleção, onde foi definida uma Política de Alocação única para executar junto as Políticas de Seleção avaliadas. A Política de Alocação selecionada foi a *GuazzoneBFD* que teve melhor desempenho em termos de consumo energético do trabalho de (FILHO, 2018).

Após a execução dessas simulações, foi definido uma última simulação para o teste de desempenho das Políticas de Alocação e seleção do presente trabalho. A última simulação consistiu em confrontar o desempenho das Políticas de Alocação e de seleção propostas nesse trabalho, trabalhando juntas (respectivamente, *VMUAllocation* e *CBHSelection*) e as melhores Políticas de Alocação e seleção da literatura trabalhando juntas (respectivamente, *GuazzorneBFD* e *AHPSelection*).

5.3 Métricas

Apresentado o cenário de testes, é necessário definir as métricas que serão adotadas para a análise e a validação das políticas. O presente trabalho irá utilizar as mesmas métricas propostas por (MANN; SZABÓ, 2017), com a adição da métrica referente a quantidade de migrações realizadas durante a simulação. A métrica de maior peso nas análises será aquela relacionada ao consumo energético ao final de cada simulação, já que o objetivo principal do trabalho é minimizar o consumo de energia em um ambiente *Data center*.

O restante das métricas estão diretamente relacionadas ao desempenho do *Data center* e a capacidade dele em manter um bom nível de SLA mesmo sob condições adversas: 1) a *SLA violation Time per Active Host* (SLATAH), que mede a violação causada pelo consumo total de CPU do servidor, 2) a *Performace Degradation due to Migration (PDM)*, que mede a violação de SLA causada pelas migrações e 3) a quantidade de migrações realizadas durante as simulações. Nas seções seguintes, cada métrica será apresentada com maiores detalhes.

5.3.1 Consumo de energia

Ao decorrer do período de atividade de um *Data center*, é necessária a utilização de energia elétrica para sustentar os servidores que contém as VMs. Tal métrica mensura o total de energia consumida pelo *Data center* durante todo período simulado e o cálculo está diretamente relacionado aos dados apresentados na Tabela 3. O cálculo da métrica é feito através da soma do consumo de energia de cada servidor ativo, em cada instante, durante o período simulado.

5.3.2 Quantidade de migrações

A Virtualização oferece elasticidade às VMs, o que significa que as mesmas podem expandir ou reduzir os recursos consumidos no servidor dinamicamente. Tais mudanças podem provocar sobrecarga ou subutilização do servidor, situações essas que exigem a migração de VMs, seja para reduzir a disputa por recursos e, potencialmente, melhorar o desempenho do serviço prestado, seja para economizar energia do *Data center*. Esta métrica representa o número de migrações realizadas por dia de simulação.

5.3.3 Performace Degradation due to Migration

Para migrar uma VM, o CloudSim dedica parte dos recursos computacionais alocados à ela para as atividades necessárias para a migração. Isso pode gerar violações de SLA, uma vez que o processamento da *Cloudlet* pode ser prejudicado, pois a máquina virtual não dispõe de todos os recursos requisitados por ela durante sua migração. O CloudSim representa esse tipo de violação através da métrica *Performace Degradation due to Migration* ou PDM. A métrica calcula o percentual médio de CPU (alocado para as VMs) que ficou indisponível para elas durante o processo de migração. A Fórmula 5.2 representa matematicamente a métrica. A fórmula consiste em calcular a quantidade (percentual) de CPU indisponível (devido a migração das VMs) para uma quantidade M de VMs.

$$SLAPDM = \frac{1}{M} \sum_{i=1}^{M} \frac{C_{d_i}}{C_{r_i}} \tag{5.2}$$

Onde

M = Número de VMs

C_{d_i} = CPU não utilizada devido a migração

C_{r_i} = CPU solicitada durante a migração

5.3.4 SLA Violation Time per Active Host

Quando um servidor está completamente sobrecarregado, ou seja, com a sua capacidade total de hardware comprometida, nenhuma das VMs nele alocadas pode solicitar recursos adicionais, pois ele é incapaz de atender a demanda. O simulador CloudSim representa essa situação como uma violação de SLA através da métrica *SLA Violation Time per Active host* ou SLATAH. Tal métrica calcula o percentual médio de tempo em que os servidores, enquanto ativos, consumiram totalmente sua capacidade de processamento. A Fórmula 5.3 representa matematicamente a métrica. A fórmula consiste em calcular o percentual médio de tempo em que cada servidor utilizou sua capacidade total de processamento quando ativo.

$$SLATAH = \frac{1}{N} \sum_{i=1}^{N} \frac{T_{s_i}}{T_{a_i}} \tag{5.3}$$

Onde

N = Número de servidores

T_{s_i} = Tempo que o servidor i está utilizando toda CPU

T_{a_i} = Tempo que o servidor i está ativo

5.4 Configurações da máquina que efetuou a simulação

A simulação foi executada em uma máquina com processador Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz com arquitetura 64-bit. Possui duas memórias RAM DDR3 de 8 GB com velocidade de clock 1600 MHz. A memória secundária possui 500 GB de capacidade. A placa de vídeo é dedicada, especificamente, é uma Afox Radeon R5 220 2GB DDR3. O sistema operacional utilizado é o Ubuntu versão 22.04. A IDE utilizada foi o Eclipse Release (4.14.0).

6 RESULTADOS

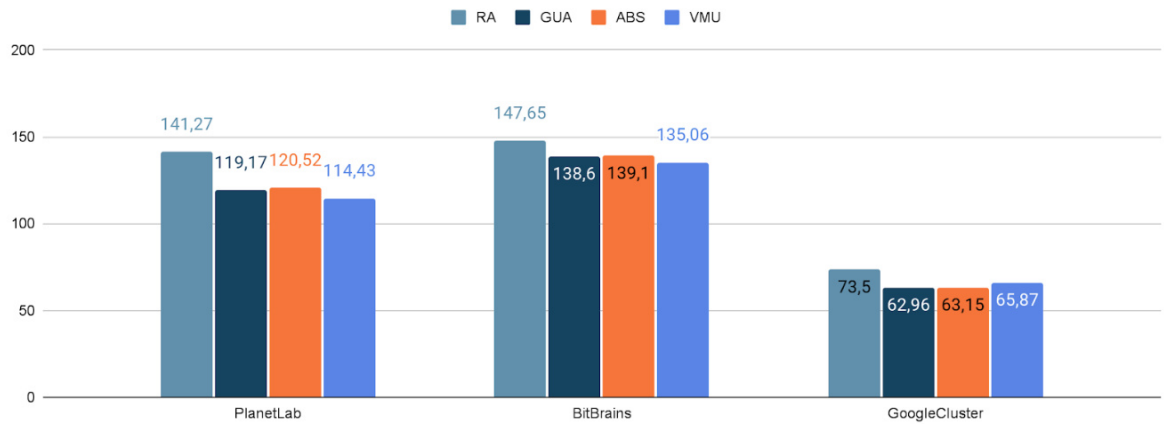
6.1 VMUAllocation (VMU)

6.1.1 Consumo de energia

O consumo de energia mensura o total de energia consumida pelo *Data center* durante todo período simulado. Uma boa política de alocação (e seleção), deverá consumir a menor quantidade possível de energia elétrica durante a simulação. Reduzir o consumo de energia no *Data center* através da implantação de uma nova política de alocação (e seleção) é o principal objetivo do presenta trabalho.

A Figura 19 apresenta os consumos energéticos das políticas de alocação nos três ambientes de simulação. Em linhas gerais, observa-se que a adoção da política VMUAllocator proporcionou um menor consumo energético nos ambientes BitBrains e PlanetLab e um desempenho intermediário no ambiente GoogleCluster. Também nota-se que a VMUAllocator superou o desempenho da política utilizada como *baseline* (RA) em todos os ambientes.

Figura 19 – Consumo energético por ambiente.



Fonte: O próprio autor

Em relação às políticas da literatura, a política VMUAllocation apresentou maior economia de energia quando comparada às políticas GUA (*GuazzoneBFD*) e ABS (*AbsoluteCapacity*) nos ambientes PlanetLab e BitBrains. No PlanetLab a VMUAllocator consumiu 3,97% menos energia que a política de alocação GUA e 5,05% menos energia que a política de alocação ABS. No BitBrains a VMUAllocator foi 2,55% e 2,90% mais economica que as políticas de alocação GUA e ABS, respectivamente. Contudo, no ambiente GoogleCuster a política VMUAllocation foi superada pelas concorrentes em até 4,12%.

Acredita-se que esse mau desempenho esteja associado ao baixo número de servidores ativos. Nota-se que o ambiente GoogleCluster apresentou os menores consumos enérgicos, onde o maior consumo foi de 73,5 Watts (*RandomAllocation*) e menor foi 63,15 Watts (*AbsoluteCapacity*). Uma quantidade reduzida de servidores diminui a possibilidade de encontrar um servidor com um espaço mínimo, podendo aumentar as chances em ligar um novo servidor. Isso implica em um índice maior de servidores ativos proporcionando um maior consumo de energia no *Data center*.

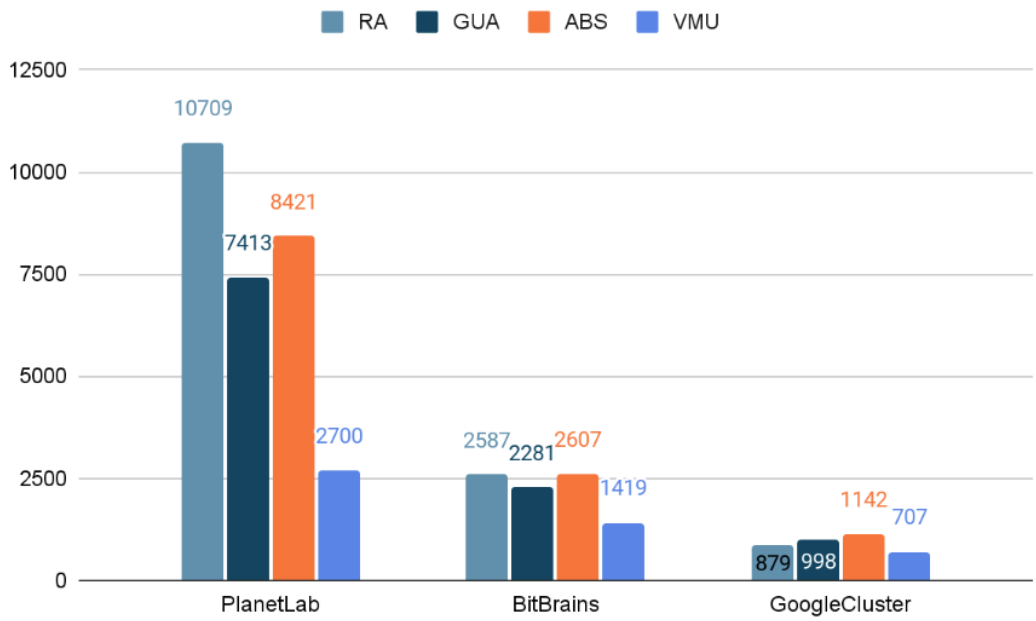
No geral, foi comprovada a eficiência energética da política proposta. A política VMUAllocation que superou em todos os três ambientes a política mais elementar (RandomAllocation) e em dois ambientes (PlanetLab e BitBrains) foi a política que economizou mais energia superando também as políticas da literatura.

6.1.2 Quantidade de migrações

A quantidade de migrações está intimamente relacionada à ocorrência de servidores sobrecarregados ou subutilizados, pois, no simulador, é somente nesses casos que ocorrem migrações de VMs. Assim, uma boa política de alocação realiza alocações de VMs evitando ao máximo possíveis sobrecargas ou subutilização dos servidores, a fim de minimizar a quantidade de migrações.

A Figura 20 mostra a quantidade de migrações de cada política de alocação em cada ambiente simulado. Nota-se que a política de alocação proposta neste trabalho (*VMUAllocator*) realizou o menor número de migrações nos três ambientes, indicando alocações mais assertivas, provocando menos sobrecargas e subutilizações dos servidores. Em geral, observa-se um comportamento parecido entre as políticas avaliadas em todos os ambientes. Por conta disso, a avaliação dos resultados ficará restrita aos resultados apresentados no ambiente PlanetLab.

Figura 20 – Quantidade de migrações por ambiente.



Fonte: O próprio autor

A política de alocação que realizou mais migrações foi a RA, com um total de 10.709 migrações. Isso ocorre porque a RA escolhe um servidor de forma aleatória, ou seja, não é analisado nenhum critério que possa evitar as possíveis sobrecargas ou subutilizações nos servidores. Assim, o comportamento da política é estocástico, podendo apresentar um bom desempenho (como no GoogleCluster, onde ela foi a segunda melhor política) ou um desempenho ruim (como no PlanetLab, onde ela foi a pior política).

A segunda política com pior desempenho foi a ABS com 8.421 migrações. Tal política escolhe sempre o servidor com maior capacidade de CPU para alocar uma VM. Tal estratégia não se mostrou muito eficiente, pois ela escolhe o servidor considerando somente a capacidade total de processamento do servidor, não considerando o quanto desse processamento está em uso. Assim, tal estratégia pode ocasionar várias situações de sobrecarga no servidor, implicando em uma grande quantidade de migrações.

A política GuazzoneBFD (a terceira pior colocada com 7413 migrações), possui uma estratégia semelhante a ABS, porém ela seleciona os servidores com a maior capacidade de CPU livre. Ou seja, essa estratégia considera o quanto de CPU está em uso e então escolhe o servidor com maior quantidade de CPU livre, o que pode reduzir a probabilidade de sobrecargas e, consequentemente, a quantidade de migrações. A VMUAllocator proporcionou um total de 2700 migrações apenas, o que representa 25,5% da terceira pior colocada. Esse bom resultado pode estar associado a estratégia aplicada a alocação inicial, onde foi feita uma distribuição

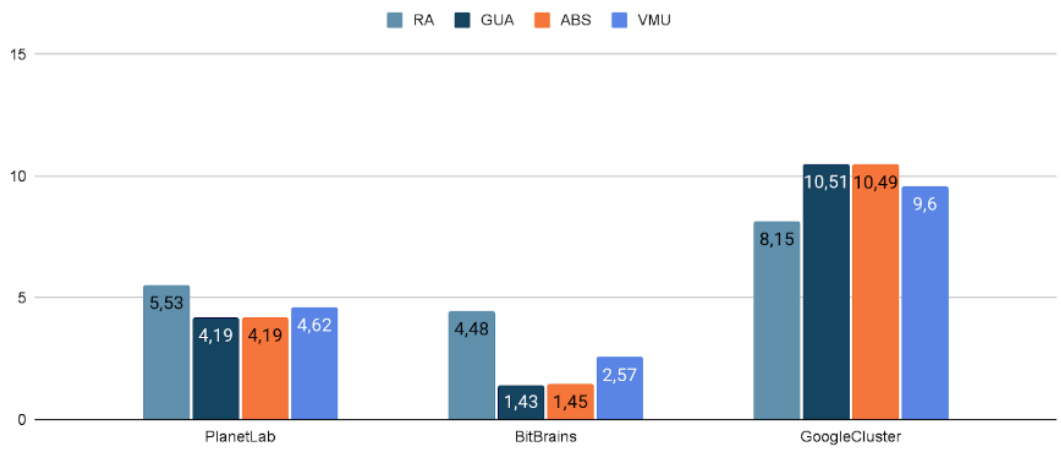
prévia das VMs buscando uma otimização inicial, respeitando os limiares de sobrecarga dos servidores.

6.1.3 SLATAH

A métrica SLATAH representa o percentual médio de tempo em que os servidores ativos consumiram sua capacidade total de processamento de CPU. Um servidor ativo consumindo sua capacidade total de processamento de CPU pode ocasionar em indisponibilidade do recurso caso alguma VM o requisite. As melhores políticas nesse quesito devem possuir um baixo SLATAH.

A Figura 21 mostra os resultados em relação a métrica SLATAH de cada política de alocação em cada ambiente simulado. A política de alocação proposta VMU teve um desempenho mediano em relação às demais políticas. No PlanetLab e BitBrains a política VMU obteve um desempenho superior à política de alocação mais elementar (RA), porém obteve um desempenho inferior às demais políticas da literatura. Em termos percentuais, no PlanetLab, a política VMU representou 83,5% da terceira colocada, enquanto as políticas ABS e GUA representaram 75,7%. No GoogleCluster a política VMU obteve um desempenho superior às políticas GUA e ABS. A política VMU superou as políticas (GUA e ABS) com um percentual de até 8,65% da terceira colocada.

Figura 21 – SLATAH por ambiente de simulação.



Fonte: O próprio autor

A política de alocação que gerou os maiores SLATAHs foi a RA no ambiente PlanetLab e BitBrains. No GoogleCluster a política RA foi obteve o melhor resultado, porém a

estratégia consiste em escolher um servidor de forma aleatória, assim tal resultado aconteceu pela a escolha dos servidores terem sidos assertivas ao acaso.

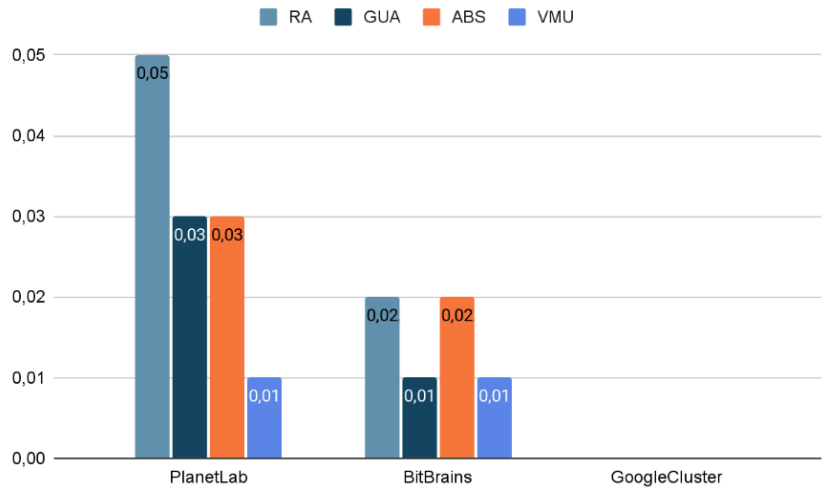
Nos ambientes do PlanetLab e BitBrains as políticas GUA e ABS tiveram os melhores resultados, enquanto a política proposta (VMU) obteve o terceiro melhor resultado. Tal desempenho é compreensível, pois existe um *tradeoff* entre o desempenho dos servidores e o consumo de energia. Quanto mais consolidados os servidores, menos energia o *Data center* irá consumir, entretanto aumenta-se a concorrência pelos recursos entres as VMs, aumentando o número de violações de SLA. Esse comportamento é proporcionado pela estratégia de alocação da política proposta. A política proposta obteve resultados satisfatórios, onde manteve um resultado mediano em uma métrica voltada para o desempenho.

6.1.4 PDM

A métrica PDM consiste em calcular o percentual de CPU indisponível pelas VMs devido a migração. Quando uma VM está sendo migrada, parte da sua CPU é dedicada para as etapas do procedimento, tornando-a indisponível para processar a tarefa da própria VM. Isso representa uma violação de SLA. Uma boa política de alocação tem uma baixa porcentagem na métrica PDM.

A Figura 22 apresenta os resultados obtidos da métrica PDM. A política de alocação proposta (VMU) neste trabalho apresentou o melhor resultado, sendo a política que menos violou a métrica PDM em todos os ambientes.

Figura 22 – PDM por ambiente de simulação.



Fonte: O próprio autor

A violação dessa métrica está relacionada diretamente à quantidade de migrações efetuadas (Figura 20). Quanto maior o número de migrações, maior será a probabilidade de uma determinada política acarretar violações de SLA. Essa relação pode ser comprovada ao comparar a Figura 20 e 22, a política que mais realizou migrações foi a RA, seguida da ABS e GUA, onde o padrão se repete em relação aos dados da métrica PDM.

Na Figura 20 é possível observar que a quantidade de migrações realizadas pelas políticas de alocação, utilizando os dados do PlanetLab, foi bem maior que a simulação utilizando os outros dois ambientes. As políticas violaram mais a métrica PDM utilizando os dados do PlanetLab. O mesmo comportamento ocorreu no segundo ambiente (BitBrains) que mais gerou migrações, onde o mesmo também ficou como segundo ambiente que mais violou a métrica PDM.

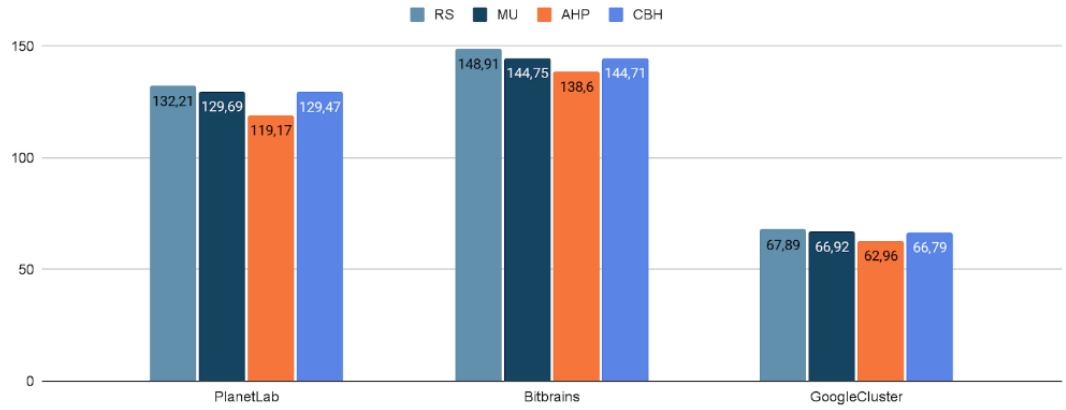
Utilizando os dados do GoogleCluster a quantidade de migrações foi mais baixa que a dos outros ambientes. Na Figura 22, nenhuma política de alocação violou a métrica PDM quando utilizou os dados de carga de trabalho do GoogleCluster. Isso ocorreu porque a quantidade de migrações executadas na simulação não foram suficientes para que a porcentagem de violações de SLA fosse considerável.

6.2 CBHSelection

6.2.1 Consumo de energia

Analisando a Figura 23 é possível notar que a política de seleção proposta (CBH) obteve um desempenho intermediário, sendo a segunda política que mais economizou energia nos três ambientes. Em geral, observa-se um comportamento parecido entre as políticas avaliadas em todos os ambientes. Por conta disso, a avaliação dos resultados ficará restrita aos resultados apresentados no ambiente PlanetLab. Nesse cenário, a CBH consumiu 129,47 Watts, o que representou uma diferença de 0,16% em relação a política que permaneceu na terceira posição (MU). A política AHP obteve um consumo energético de 119,17 Watts, apresentando uma diferença de 8,11% em relação a terceira colocada.

Figura 23 – Consumo energético em cada ambiente de simulação.



Fonte: O próprio autor

A política de seleção RS foi política que mais consumiu energia em todos os ambientes. Isso ocorreu pelo fato de sua estratégia escolher as VMs para migração de forma aleatória em um servidor sobrecarregado, sem levar em consideração nenhum critério, o que a tornou a política de seleção com o pior desempenho energético nos três ambientes.

A política de seleção que economizou mais energia foi a política de seleção *Analytic Hierarchy Process* (AHP). Tal política utiliza uma técnica comumente usada para atacar problemas de decisão, onde define-se um conjunto de critérios para resolver um problema de tomada de decisão. Na proposta de (MORAIS, 2019) foi destacada a CPU e a RAM como os principais componentes que mais consomem energia e, com base neles, foram definidos alguns critérios para a escolha da VM a ser migrada: 1) A VM deve ter o menor consumo em CPU dentre as outras VMs alocadas no servidor, 2) A VM deve ter a menor média de consumo de CPU através do histórico de consumo, 3) A VM deve ter um menor consumo de memória RAM dentre as VMs alocadas no servidor.

A segunda colocação da política de seleção proposta (CBH) é explicada pelo fato dela considerar apenas um critério para migrar uma VM. Na política proposta, a CPU foi considerada como o componente que mais consume energia elétrica e, por conta disso, foi definido como critério único. Além disso, a estratégia de escolher as VMs problemáticas para serem migradas pode resultar na ativação de servidores que estavam desligados ou em modo de baixo consumo de energia. Como as VMs problemáticas possuem um alto consumo médio de CPU, realocá-las pode ser um desafio, já que potencialmente pouquíssimos servidores terão recursos suficientes para recebê-las. Caso não seja encontrado um servidor ativo para a VM que está sendo realocada, um novo servidor será posto em modo ativo para recebe-la, o que implica no aumento do consumo energético em *Data center*. A AHPSelection, por sua vez, conseguiu

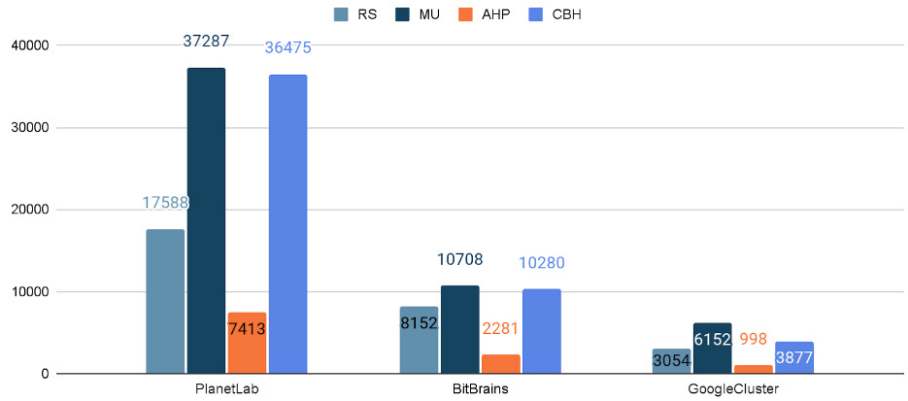
reduzir o consumo energético considerando CPU e memória RAM, abrangendo mais de um critério para a seleção da VM, o que impactou positivamente no resultado das simulações no *Data center*.

A política de seleção MU (a terceira colocada nessa métrica) foi superada pela CBH por uma leve diferença. Tal resultado é compreensível, porque a política MU seleciona a VM considerando somente o consumo de momento da CPU, isto é, a VM que consome a menor quantidade de CPU dentre as demais VMs alocadas no servidor naquele instante. A política de seleção proposta (CBH) seleciona as VMs pela sua classificação, em específico, VMs do tipo 3, isto é, as VMs problemáticas, podem causar altos consumos de energia e violações de SLA. Tal classificação é realizada com base no histórico de consumo. Caso não exista nenhuma VM do tipo 3, então não ocorrem migrações. Isso pode reduzir o consumo energia, porém pode ocasionar um aumento no SLATAH.

6.2.2 Quantidade de migrações

No geral, a política de seleção proposta (CBH) teve um dos piores desempenhos, sendo a segunda pior em todos os ambientes. Em geral, analisando a Figura 24, observa-se um comportamento parecido entre as políticas avaliadas em todos os ambientes. Por conta disso, a avaliação dos resultados ficará restrita aos resultados apresentados no ambiente PlanetLab. Nesse cenário, a política proposta realizou 36.475 migrações, o que representa uma diferença de 2,17% em relação a política que permaneceu na terceira colocação (MU) com 37.287 migrações. A política AHP realizou somente 7.413 migrações, representando uma diferença de 80,11% em relação à terceira colocada (MU).

Figura 24 – Quantidade de migrações por ambiente.



Fonte: O próprio autor

A política MU obteve o pior desempenho porque sua estratégia consiste na migração de VMs com menor consumo de processamento momentâneo, onde para todo servidor ocorre a seleção de uma VM para migração. Segundo (MATOS *et al.*, 2015), a recusa de escolher algumas migrações de VMs pode ser benéfica para o desempenho das políticas de seleção, pois tal ação consegue reduzir o consumo de energia e a quantidade de migrações. A política CBH seleciona uma VM para migrar somente se existir ao menos uma VM problemática (tipo 3), caso contrário, ela não seleciona nenhuma VM para migração. Isso pode ser a justificativa para a leve diferença entre as duas políticas nessa métrica.

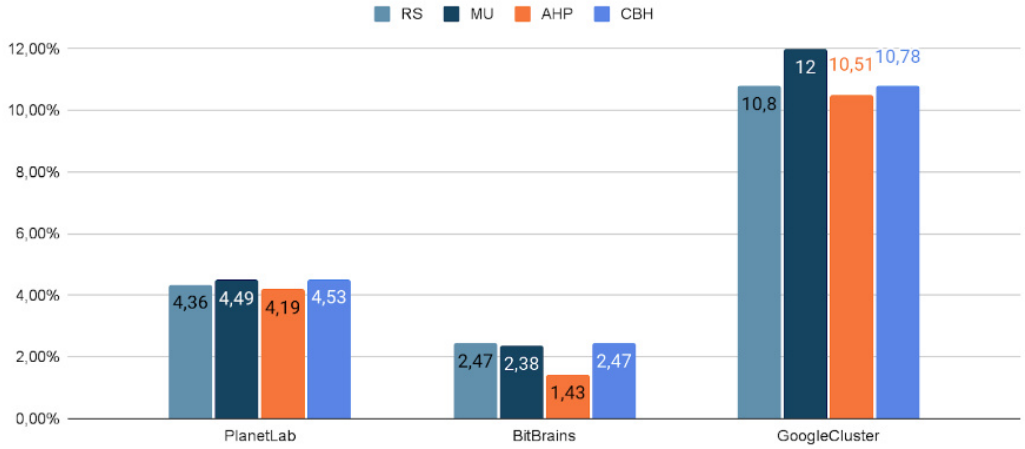
A política de seleção que gerou menos migrações foi a *AHPSelection*. Tal resultado pode ser explicado devido a sua estratégia em considerar, em sua análise, mais de um critério simultaneamente. Nessa política, a VM para ser escolhida para migração deverá ter: 1) o menor consumo CPU no momento, 2) a menor média de consumo de CPU no servidor e 3) o menor consumo de memória RAM. A estratégia tem um número maior de condicionais na escolha da VM, o que reduz as chances de migração.

A política RS apresentou um bom resultado apesar da sua estratégia consistir em migrar as VMs aleatoriamente. Em teoria, o volume de migrações deveria ser muito maior do que volume apresentado, entretanto não foi isso o que os resultados mostraram. Acredita-se que esse baixo volume ocorreu por conta das seleções aleatórias terem sido mais assertivas em sua maioria, mesmo que ao acaso. E ao serem redirecionadas aos servidores para realocação não causaram muitas sobrecargas.

6.2.3 SLATAH

Em relação a métrica SLATAH a política de seleção CBH teve um dos piores resultados (Figura 25). No ambiente PlanetLab, a política teve o pior resultado, sendo levemente pior que a política mais elementar (RS). A política proposta obteve 4,53% na SLATAH com uma diferença de apenas 0,04% em relação a política terceira colocada (MU com 4,49%) e de apenas 0,34% quando comparada a política melhor colocada (*AHPSelection* com 4,19%). Já no BitBrains, a CBH teve um desempenho igual a política RS, sendo as duas aquelas com os piores resultados. No ambiente GoogleCluster a política CBH obteve o segundo melhor desempenho.

Figura 25 – SLATAH por ambiente.



Fonte: O próprio autor

A política de seleção RS conseguiu resultados intermediários nos ambientes PlanetLab e GoogleCluster e o pior resultado no ambiente BitBrains. Acredita-se que os resultados intermediários ocorreram por conta das seleções aleatórias terem sido mais assertivas em sua maioria, mesmo que ao acaso e ao serem redirecionadas aos servidores para realocação não causaram muitas sobrecargas e consequentemente menos violações de SLA.

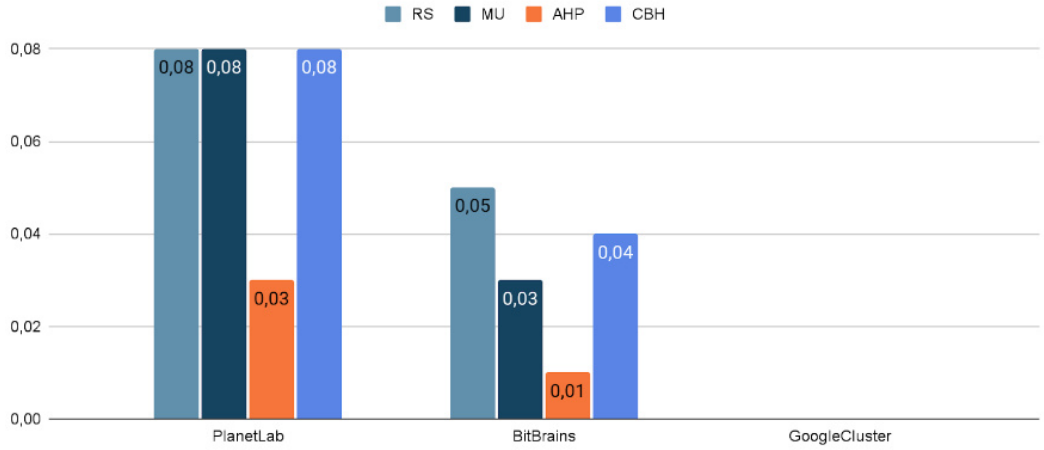
A política de seleção proposta ficou em último lugar nos ambientes PlanetLab e BitBrains. Isso porque a política proposta migrou VMs problemáticas em relação a CPU, o que aumentou a probabilidade de acontecer violações de SLA no servidor que poderia hospedá-la, no processo de realocação. Em segundo lugar, e nos mesmos ambientes (PlanetLab, BitBrains) ficou a política MU. A estratégia da política MU também considera a CPU como critério, escolhendo as VMs que menos consomem CPU no servidor sobrecarregado. Ao migrar uma VM com baixo consumo de CPU diminui a possibilidade de futuras violações de SLA no servidor poderia recebê-la, no processo de realocação. A política que obteve o melhor resultado foi a política de seleção AHP, isso pelo fato da mesma considerar não só a CPU como critério mas também por ela considerar um segundo componente que também pode impactar em violações de SLA que é a memória RAM.

O GoogleCluster foi o ambiente que menos consumiu energia (Figura 23). Isso pode estar relacionado ao baixo número de máquinas ativas no *Data center*. Nos resultados da Figura 25, as políticas de seleção apresentaram as maiores porcentagens em relação a métrica SLATAH, isso pelo fato do baixo número de máquinas ativas o que aumenta a porcentagem média de máquinas no cálculo da métrica.

6.2.4 PDM

Na Figura 26 é possível observar que a política CBH manteve desempenho intermediário no ambiente de simulação PlanetLab. As políticas RS, MU e CBH mantiveram a mesma porcentagem de PDM, enquanto a AHP foi o destaque com a menor porcentagem. No BitBrains, observa-se uma diferença no desempenho das políticas de seleção, onde é possível notar que a política proposta obteve o segundo pior desempenho. A violação dessa métrica está diretamente vinculada ao número de migrações efetuadas (Figura 24). Quanto maior a quantidade de migrações, maior será a probabilidade de uma determinada política acarretar violações de SLA por parte da VM que está sendo migrada.

Figura 26 – PDM por ambiente de simulação.



Fonte: O próprio autor

Na Figura 24, é possível observar que a quantidade de migrações realizadas no ambiente do PlanetLab foi muito maior do que aquela observada nos outros dois ambientes. Por conta disso, as políticas violaram mais a métrica PDM quando foram submetidas ao ambiente PlanetLab. O mesmo comportamento ocorreu no segundo ambiente (BitBrains) que também gerou um número significativo de migrações, onde o mesmo também ficou como segundo ambiente que mais violou a métrica PDM. Por outro lado, no ambiente GoogleCluster a quantidade de migrações foi a mais baixa de todos os ambientes. O número de migrações foi tão irrisório que o número de violações da métrica PDM foi aproximadamente zero para todas as políticas.

Apesar do baixo volume de migrações a política RS obteve os maiores percentuais na métrica PDM. A violação dessa métrica está relacionada diretamente à quantidade de migrações efetuadas (Figura 24). Quanto maior o número de migrações, maior será a probabilidade de uma determinada política acarretar violações de SLA. Esse cenário atípico da política RS pode ser

explicado pelo fato da quantidade de CPU solicitada para uma VM ser migrada ocorreu em um volume maior ao acaso.

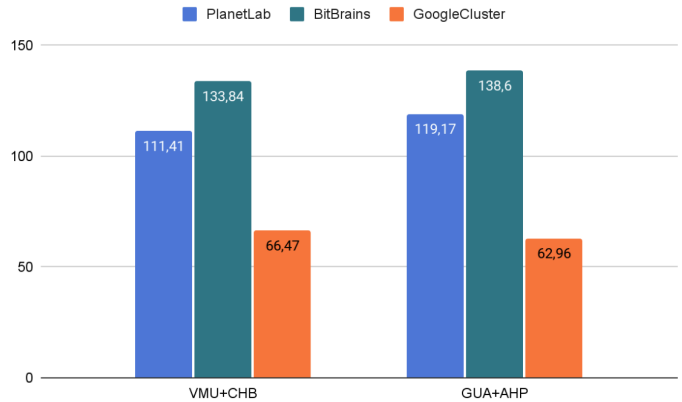
Acredita-se que os resultados ruins da CBH ocorreram devido a forma como ela escolhe VMs para migração. A estratégia é escolher sempre VMs problemáticas, o que pode aumentar a probabilidade de as VMs não terem recursos suficientes para migração, que é o principal indicador que impacta na métrica PDM. Uma VM problemática é uma VM que tem uma alta média e variância de consumo de CPU, o que aumenta as chances de no momento da migração, a VM estar com um alto consumo de CPU, podendo causar indisponibilidade desse recurso. Tais resultados evidenciaram que essa estratégia pode ser bastante prejudicial para esse tipo de métrica.

6.3 VMUAllocation e CHBSelection em comparação as melhores políticas da literatura

6.3.1 Consumo de energia

A Figura 27 apresenta os consumos energéticos das políticas de alocação e seleção propostas (VMU e CBH respectivamente) em comparação às políticas de alocação e seleção da literatura (GUA e AHP respectivamente) com melhor desempenho energético, em dois ambientes de simulação. Em linhas gerais, observa-se que a adoção das políticas propostas proporcionou um menor consumo energético nos ambientes *BitBrains* e *PlanetLab* e um desempenho levemente inferior às políticas da literatura no ambiente GoogleCluster. No *PlanetLab* VMU e CBH, trabalhando juntas, consumiram 6,5% menos energia que as políticas da literatura, enquanto no *BitBrains* a redução foi de aproximadamente 3,43%. Porém, no *GoogleCluster*, VMU e CBH foram superadas pelas concorrentes em 5,28%.

Figura 27 – Consumo energético por ambiente.



Fonte: O próprio autor

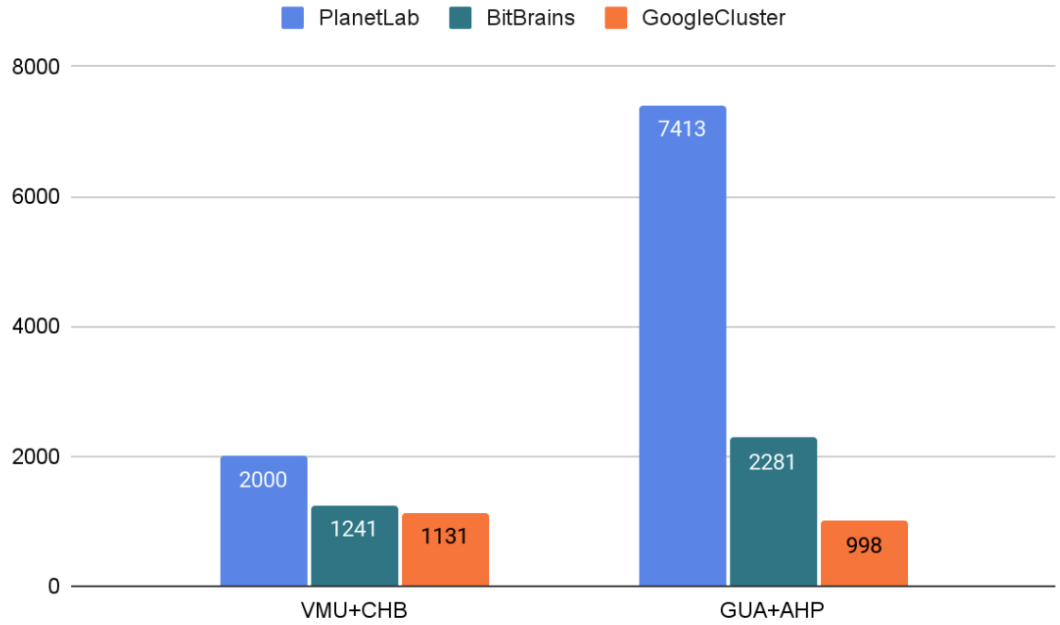
A explicação dessa diferença está na estratégia da política de alocação e seleção proposta. A política de alocação proposta aloca uma VM em um servidor se nele é encontrado um espaço mínimo para a alocação visando a consolidação dos servidores. A consolidação dos servidores permite explorar o máximo dos recursos de alguns poucos servidores ativos, o que proporciona uma maior incidência de servidores aptos a serem desligados ou postos em modo de economia de energia. Isso implica na diminuição do consumo energético no *Data center*. A política de seleção retorna apenas VMs do tipo 3 (problemáticas) e caso não existam, não ocorre migrações. (MATOS *et al.*, 2015) destaca que negar a escolha de uma máquina virtual para migração pode representar economia de energia, pois diminui a quantidade de migrações e consequentemente diminui o consumo de energia. Isso corre porque mantendo um servidor explorando a sua capacidade máxima de processamento contribui para o processo de consolidação dos servidores.

Acredita-se que esse desempenho inferior no ambiente GoogleCluster esteja associado ao baixo número de servidores ativos. Uma quantidade reduzida de servidores diminui a possibilidade da política de alocação VMU encontrar um servidor com um espaço mínimo, que é a estratégia da política de alocação proposta, podendo aumentar as chances em ligar um novo servidor. Isso implica em um índice maior de servidores ativos proporcionando um maior consumo de energia no data center. No geral, foi comprovada a eficiência energética das políticas propostas superando as políticas da literatura na maioria dos ambientes.

6.3.2 Quantidade de migrações

A Figura 28 mostra a quantidade de migrações das políticas de alocação e seleção propostas (VMU e CBH respectivamente) em comparação às políticas de alocação e seleção da literatura (GUA e AHP respectivamente) com melhor desempenho energético, em dois ambientes de simulação. Nota-se que as políticas propostas realizaram o menor número de migrações em dois ambientes de simulação PlanetLab e Bitbrains, indicando alocações mais assertivas e provocando menos sobrecargas e subutilizações dos servidores. No GoogleCluster as políticas propostas foram superadas pelas políticas da literatura. No PlanetLab as políticas propostas realizaram 73,02% menos migrações que políticas da literatura, enquanto no BitBrains a redução foi de 45,59%. No GoogleCluster as políticas propostas foram superadas em 11,75%.

Figura 28 – Quantidade de migrações por ambiente.



Fonte: O próprio autor

No geral, as políticas propostas conseguiram bons resultados, que podem estar relacionados a estratégia aplicada nas políticas propostas. A política de alocação possui a estratégia de alocação inicial, com o intuito de distribuir as VMs de uma maneira otimizada inicialmente, respeitando os limiares de sobrecarga dos servidores, enquanto tenta evitar possíveis disputas por recursos e migrações. Tal estratégia minimizou o volume de migrações nos processos de realocação. A política de seleção CBH retorna apenas VMs do tipo 3 (problemáticas) e caso não existam não ocorre migrações, esse comportamento proporciona a redução no número de migrações.

Em relação as políticas da literatura, os seus resultados inferiores podem explicados. Um dos fatores que influenciou no número elevado de migrações, correram por parte da política de alocação GUA, por conta da mesma não possuir um processo de alocação de inicial assim como a política de alocação proposta (VMU). Outro fator que pode ser considerado, pode ser a estratégia da política de alocação GUA sempre escolher a CPU mais livre o que pode ocasionar em um número elevado de máquinas ativas. E ao gerar um cenário de muitas máquinas ativas com pouco espaço de CPU o processo de alocação nesses servidores pode resultar em maior número de sobrecargas, gerando um grande número de migrações. E apesar da política de seleção AHP ter mais condicionais para selecionar uma VM, reduzindo a quantidade de migrações como explicado na seção anterior, a política de alocação produz mais cenários propícios a migração.

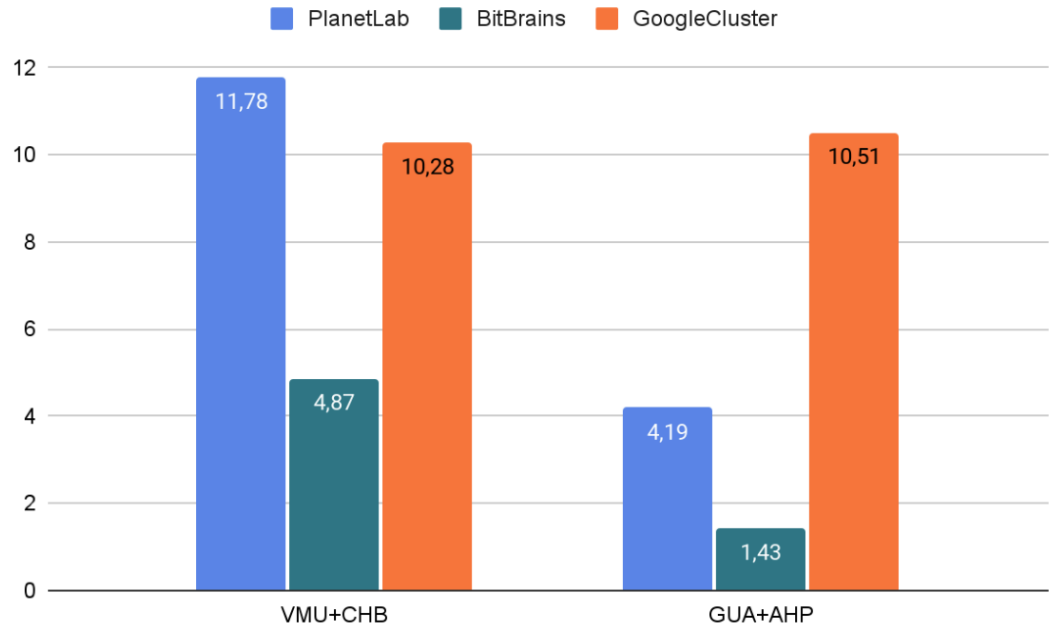
Acredita-se que esse desempenho inferior no ambiente GoogleCluster esteja associ-

ado ao baixo número de servidores ativos. Uma quantidade reduzida de servidores diminui a possibilidade da política de alocação VMU encontrar um servidor com um espaço mínimo, ao alocar VMs problemáticas vindas das políticas de seleção nesse volume reduzido de máquinas ativas implicou em uma maior quantidade de sobrecargas e consequentemente a uma maior quantidade de migrações. No geral, foi comprovada a eficiência energética das políticas propostas superando as políticas da literatura na maioria dos ambientes.

6.3.3 SLATAH

A Figura 29 mostra os resultados em relação a métrica SLATAH das políticas de alocação e seleção propostas (VMU e CBH respectivamente) em comparação às políticas de alocação e seleção da literatura (GUA e AHP respectivamente). Em termos percentuais, no PlanetLab, as políticas propostas violaram 64,43% a mais que as políticas da literatura, no BitBrains houve uma diferença de 29,36% e no GoogleCluster as políticas propostas superam as políticas concorrente em 2,18%.

Figura 29 – SLATAH por ambiente de simulação.



Fonte: O próprio autor

Esse resultado negativo para as políticas propostas pode ser explicado devida a dois pontos principais: 1) A política de seleção CBH sempre migra VMs problemáticas. o que pode aumentar as chances de violações de SLA no servidor que pode vai recebê-la no final do processo de realocação; 2) A política de alocação VMU aloca VMs problemáticas no servidor com o

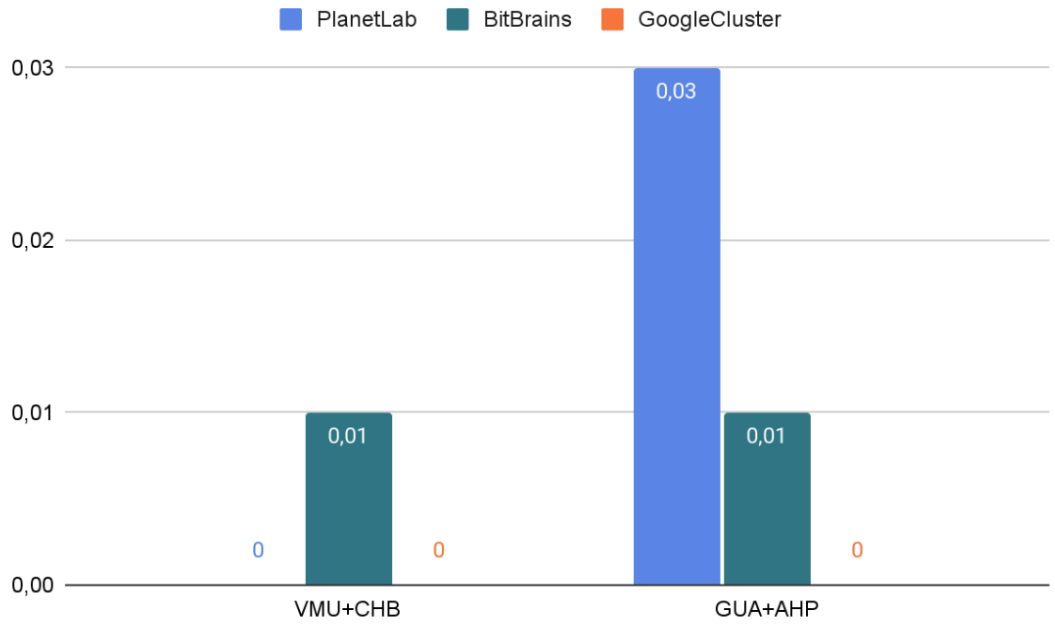
mínimo de recursos necessários para recebê-la. Tal estratégia facilita o processo da consolidação de servidores, porém consolidar demais pode ocasionar em uma série de violações de SLA.

As políticas da literatura por sua vez, trabalham de forma diferente. A política de seleção AHP seleciona VMs que menos consomem CPU e memória RAM, o que facilita não só o processo de realocação por parte da política GUA, mas também previne futuras violações de SLA pelo fato de as VMs serem "comportadas". A política de alocação GUA tem uma contribuição importante também, por conta da sua estratégia sempre considerar a escolha de servidor que possui a maior quantidade de CPU livre, o que diminui o risco de violações de SLA.

6.3.4 PDM por ambiente de simulação.

A Figura 30 apresenta os resultados obtidos na métrica PDM das políticas de alocação e seleção propostas (VMU e CBH respectivamente) em comparação às políticas de alocação e seleção da literatura (GUA e AHP respectivamente). A políticas propostas geraram resultados semelhantes em relação as políticas da literatura nos ambientes BitBrains e GoogleCluster. No Ambiente do PlanetLab as políticas propostas apresentaram um bom resultado em relação as políticas da literatura.

Figura 30 – PDM por ambiente de simulação.



Fonte: O próprio autor

A normalmente a violação dessa métrica está relacionada diretamente à quantidade de migrações efetuadas (Figura 28). Quanto maior o número de migrações, maior será a probabi-

lidade de uma determinada política acarretar violações de SLA. Essa relação foi comprovada em todos os ambientes para as políticas da literatura GUA e AHP na métrica PDM, onde o maior número de migrações ocorreu primeiramente no ambiente PlanetLab, em segundo BitBrains e terceiro Googler Cluster.

Podemos observar um comportamento atípico na métrica PDM das políticas propostas no ambiente do PlanetLab. A quantidade de migrações executada pelas políticas propostas nesse ambiente foi maior em relação aos demais ambientes onde foram simuladas, e no mesmo ambiente a métrica PDM foi mais baixa que no ambiente BitBrains que executou menos migrações. Esse comportamento atípico pode ser explicado por conta da dinâmica de seleção de VMs por parte da política de seleção CBH ter escolhido VMs com as menores capacidades de CPU. Ou seja, as VMs com as menores capacidades de CPU em sua maioria, tiveram um comportamento problemático de acordo com a análise seus históricos de consumo de CPU e classificadas como VMs do tipo de 3.

O cálculo da métrica PDM consiste em calcular a quantidade (percentual) média de CPU indisponível de todas as VMs durante a migração em toda simulação. Sendo assim, a capacidade total de processamento da CPU de uma VM influencia no cálculo final. Por exemplo ao selecionarmos uma VM com capacidade total de 2000 MIPS e utilizamos 10% desse processamento para migração equivale a 200 MIPS. Se utilizamos 10% de processamento uma VM que possui uma capacidade total de apenas 1000 MIPS equivale a 100 MIPS utilizados para migração, representando um valor inferior a VM que possui uma maior capacidade de processamento.

Utilizando os dados do GoogleCluster a quantidade de migrações foi mais baixa em relação aos outros ambientes. Isso ocorreu porque a quantidade de migrações executadas na simulação não foi suficiente para que a porcentagem de violações de SLA fosse considerável.

7 CONCLUSÕES E TRABALHOS FUTUROS

A crescente expansão das infraestruturas dos data centers resultou em um significativo incremento no consumo de energia elétrica. Por esse motivo, têm sido propostas diversas políticas de alocação, uma vez que tais políticas buscam uma alocação mais eficiente, visando assim uma considerável diminuição do consumo energético. Visando atacar esse problema, o presente trabalho apresentou duas políticas, uma de seleção e outra de alocação de máquinas virtuais, que tiveram como objetivo principal reduzir o consumo de energia em um ambiente de *Data center* e, como objetivo secundário (e desejável), manter um bom desempenho dos serviços prestados.

A política de alocação VMUAllocation usou um método para alocação inicial que consistiu em alocar duplas de VMs nos Hosts, e não individualmente como foi feito em algumas políticas da literatura. O objetivo de tal abordagem é agrupar VMs que são potencialmente capazes de coexistir no servidor, afim de proporcionar uma distribuição mais otimizada logo na alocação inicial. Contudo, o consumo das VMs em execução podem criar condições de disputa pelos recursos do Host, o que pode trazer problemas de SLA e exigir a redistribuição de algumas delas. Nesse caso, a política de seleção CBHSelection atua selecionando uma VM (ou mais) definida como problemática, ou seja, uma VM problemática é uma VM que possui variância e média alta de acordo como análise do seu histórico de consumo de CPU, sendo assim, VMs desse tipo consomem uma grande quantidade de recursos computacionais (em média) ou possuem alta variância no consumo desses mesmos recursos. Escolhida uma VM para migração, ela será alocada individualmente no Host com o espaço mínimo para recebê-la.

Os resultados mostram que a política de alocação proposta demonstrou bons resultados se destacando como a melhor política em termos energéticos na maioria dos ambientes. Isso indica que a ideia de agrupar as VMs de tipos complementares em pares pode trazer bons resultados para o consumo de energia do *Data center*. Em relação a métrica SLATAH, ela se destacou como uma das piores, porém é um comportamento esperado, uma vez que existe um *tradeoff* em ter melhor consumo energético e ter as menores violações de SLA. A quantidade de migrações geradas pela política de alocação VMUAllocation foi a menor em todos os ambientes em comparação as demais política de alocação da literatura. O baixo volume de migrações acabou gerando uma baixa violação da métrica PDM, onde a política de alocação proposta se estacou como a melhor em todos os ambientes.

A política de seleção apresentou bons resultados em relação ao consumo energético,

a mesma ficou em segundo lugar em todos os ambientes simulados. Em relação as violações de SLA a política apresentou os piores resultados em todos os ambientes de simulação. A quantidade de migrações geradas pela política de seleção CBHSelection foi a segunda pior em todos os ambientes de simulação em comparação as demais política de alocação da literatura. O grande volume de migrações acabou gerando um alto valor nas violações da métrica PDM, onde a política de seleção proposta se estacou também como a segunda pior em todos os ambientes. Trabalhando juntas as políticas de alocação e seleção propostas conseguiram os menores consumos de energia em comparação as políticas da literatura, porém tiveram os piores resultados em relação a violação de SLA (SLATAH).

Para os trabalhos futuros em relação a política de alocação é possível explorar a utilização não só de duplas, mas triplas e grupos maiores de VMs, no processo de alocação inicial. Essa adequação na alocação inicial pode abrir oportunidades para melhorar ainda mais o desempenho da política de alocação. Na política de seleção sugere-se acrescentar mais critérios durante a classificação da VM, como memória principal, memória secundária entre outras. Expandir para mais critérios poderia aumentar a possibilidade de melhoria nas métricas da política de seleção.

REFERÊNCIAS

ABDEL-BASSET, M.; ABDLE-FATAH, L.; SANGAIAH, A. K. An improved lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment. **Cluster Computing**, 2019. 32

ABDEL-BASSET, M.; MANOGARAN, G.; ABDEL-FATAH, L. An improved nature inspired meta-heuristic algorithm for 1-d bin packing problems. **Pers Ubiquit Comput**, p. 1117–1132, 2018. 28

AL-DHURAIBI, Y.; PARAISO, F.; DJARALLAH, N.; MERLE, P. Elasticity in cloud computing: State of the art and research challenges. **IEEE TRANSACTIONS ON SERVICES COMPUTING**, v. 11, n. 2, p. 1–2, 2018. 13

AMAZON WEB SERVICES. **O que é a computação em nuvem?** 2020. Disponível em: <https://aws.amazon.com/pt/what-is-cloud-computing/>. Acesso em: 9 set. 2020. 17

BELOGLAZOV, A.; BUYYA, R. Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers.: Which is the best algorithm for virtual machine placement optimization. Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science., 2010. 57

BELOGLAZOV, A.; BUYYA, R. Optimal online deterministic algorithms an adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. **CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE**, p. 1397–1420, 2011. 24

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. Concurr. Comput.:Pract. Exper., 2012. 55, 57

CALHEIROS; RANJAN; BELOGLAZOV; ROSE; BUYYA. Software: Practice and experience: Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Wiley Online Library, 2011. 51, 52

CARVALHO, M. M. heap allocator: uma política de alocação de máquinas virtuais em ambiente de computação nas nuvens que utiliza heap com prioridade. 2016. 38

CARVALHO, V. J. M. Exact solution of bin packing problems using column generation and branch-and-bound. **Annals of Operations Research**, 1999. 25

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T.; BLAIR, G. **Sistemas Distribuídos**: Conceitos e projetos. [S.l.]: Distributed Systems: Concepts and Design, 5th Edition, 2013. 17

DATA CENTER FRONTIER. **State of the APAC Data Center Market**. 2020. Disponível em: <https://datacenterfrontier.com/apac-data-center-market/>. Acesso em: 18 mar. 2020. 14

DATA CENTER FRONTIER. **Data Center Sustainability Comes to the Fore (Even More) in 2023**. 2022. Disponível em: <https://www.datacenterfrontier.com/executive-roundtable/article/21438743/data-center-sustainability-comes-to-the-fore-even-more-in-2023>. Acesso em: 04 jul. 2023. 15

DATACENTER KNOWLEDGE. **Study: Data Centers Responsible for 1 Percent of All Electricity Consumed Worldwide**. 2020. Disponível em: <https://www.datacenterknowledge.com/energy/study-data-centers-responsible-1-percent-all-electricity-consumed-worldwide>. Acesso em: 30 out. 2020. 14

DATACENTRE. **Energy efficiency predictions for data centres in 2023**. 2022. Disponível em: <https://datacentremagazine.com/articles/efficiency-to-loom-large-for-data-centre-industry-in-2023>. Acesso em: 04 jul. 2023. 15

DAYARATHNA, M.; WEN, Y.; IEEE, S. M.; FAN, R. Data center energy consumption modeling:: A survey. **IEEE COMMUNICATIONS SURVEYS TUTORIALS**, p. 732–794, 2016. 33

DEBASHIS DE. **MOBILE CLOUD COMPUTING**: Architectures, algorithms and applications. [S.l.: s.n.], 2016. 18

FILHO, F. M. V. Msiallocator: Aplicação da meta-heurística ils para o problema de alocação de máquinas em um data center. 2018. 30, 56, 57, 58

FORBES. **Why Energy Is A Big And Rapidly Growing Problem For Data Centers**. 2017. Disponível em: <https://www.forbes.com/sites/forbestechcouncil/2017/12/15/why-energy-is-a-big-and-rapidly-growing-problem-for-data-centers/?sh=7dc3245a307f>. Acesso em: 21 mar. 2020. 14

FORBES. **Data Centers Are Not The Energy Hogs We Thought**. 2020. Disponível em: <https://www.forbes.com/sites/jeffmcmahon/2020/03/01/data-centers-are-not-the-energy-hogs-we-thought/?sh=80ef3fb1218f>. Acesso em: 13 mar. 2020. 14

FURHT, B.; ESCALANTE, A. J. **HandBook of Cloud Computing**. [*S. l.*]: Springer Science+Business Media, 2010. 13

GUAZZONE, M.; ANGLANO; CANONICO, M. Exploiting vm migration for the automated power and performance management of green cloud computing systems. **Proceedings of the First International Conference on Energy Efficient Data Centers.**, p. 81–92, 2012. 56, 57

HAN, G.; QUE, W.; JIA, G.; ZHANG, W. Resource-utilization-aware energy efficient server consolidation algorithm for green computing in iiot. **Journal of Network and Computer Applications**, 2017. 23

INAP. **What are the Differences Between IaaS, PaaS and SaaS?** 2020. Disponível em: <https://www.inap.com/blog/iaas-paas-saas-differences/>. Acesso em: 06 nov. 2020. 19

KANG, J.; PARK, S. Algorithms for the variable sized bin packing problem. **European Journal of Operational Research**, p. 365–372, 2003. 26

LU, Y.; SUN, N. An effective task scheduling algorithm based on dynamic energy management and efficient resource utilization in green cloud computing environment. **Cluster Comput (2019)**, p. 513–514, 2019. 23

MANN, Z. A.; SZABÓ, M. Concurrency and computation: Practice and experience: Which is the best algorithm for virtual machine placement optimization. Wiley Online Library, 2017. 51, 59

MATOS, F. F. S. B. d.; CELESTINO, J.; CARDOSO, A. R. A selection policy of virtual machines for load balancing in cloud computing. 2015 IEEE Symposium on Computers and Communication (ISCC), 2015. 70, 74

MICHAEL, R.; RONALD, L.; JEFFREY, D. Análise de piores casos de algoritmos de alocação de memória. **STOC**, p. 143–150, 1972. 28

MICROSOFT AZURE. **What are the different types of cloud computing services?** 2020. Disponível em: <https://azure.microsoft.com/en-us/overview/types-of-cloud-computing/>. Acesso em: 06 nov. 2020. 18

MORABITO, R.; COZZOLINO, V.; DING, A. Y.; BEIJAR, N.; OTT, J. Consolidate iot edge computing with lightweight virtualization. **IEEE NETWORK**, p. 102–105, 2018. 22

MORAIS, S. Ahpselection: AplicaÇÃo da tÉcnica ahp no problema de seleÇÃo de mÁquinas virtuais para migraÇÃo em um ambiente de data center. 2019. 57, 58, 68

ORACLE E-BUSINESS SUITE TECHNOLOGY. **Virtualization and E-Business Suite**. 2006. Disponível em: <https://blogs.oracle.com/ebstech/virtualization-and-e-business-suite>. Acesso em: 09 set. 2020. 21

PANDA, S. K.; JANA, P. K. An energy-efficient task scheduling algorithm for heterogeneous cloud computing systems. **Cluster Computing(2019)**, p. 509–527, 2018. 15

PAPADIMITRIOU C. H; STEIGLITZ, K. **Combinatorial Optimization**: Algorithms and complexity. [S.l.: s.n.], 1998. 25

PRADO, D. **Programação Linear**: Série pesquisa operacional. [S.l.: s.n.], 2010. 25

QIN, L.; CHIU, C. T.; JIE, W.; GUOJUN, W. Efficient information retrieval for ranked queries in cost-effective cloud environments. **IEEE INFOCOM**, p. 2581–2585, 2012. 13

SALHI, S. Handbook of metaheuristics (2nd edition). **Journal of the Operational Research Society**, p. 65–320, 2014. 25

SCURRA. 2017. Disponível em: <http://www.scurra.com.br/blog/wp-content/uploads/2017/08/cloud-hibryd.png>. Acesso em: 07 nov. 2020. 20

SHI, L.; FURLONG, J.; WANG, R. Exploiting vm migration for the automated power and performance management of green cloud computing systems. **Computers and Communications (ISCC), 2013 IEEE Symposium on**, p. 9–15, 2013. 56, 57

THE STARTUP . **Exploring the Bin Packing Problem**. 2021. Disponível em: <https://medium.com/swlh/exploring-the-bin-packing-problem-f54a93ebdbe5>. Acesso em: 13 fev. 2021. 26

ULLMAN, J. O desempenho de um algoritmo de alocação de memória. 1971. 28

WANG, J.; ZHANG, L.; DUAN, L. A new paradigm of cloud-based predictive maintenance for intelligent manufacturing. **J Intell Manuf**, p. 1125–1137, 2015. 13

ZHANG, X.; WU, T.; CHEN, M.; WEI, T.; ZHOU, J.; HU, S.; BUYYA, R. Energy-aware virtual machine allocation for cloud with resource reservation. **The Journal of Systems and Software**, p. 147–161, 2019. 31