UNIVERSIDADE FEDERAL DE CAMPINA GRANDE

CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA

UNIDADE ACADÊMICA DE SISTEMAS E COMPUTAÇÃO

COORDENAÇÃO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MARCELA TASSYANY GALDINO SANTOS

UMA ABORDAGEM PARA GERENCIAMENTO DE SOBRECARGA DE RECURSOS BASEADA NA DIFERENCIAÇÃO DE SERVIÇOS

CAMPINA GRANDE - PB

2021

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

Uma abordagem para gerenciamento de sobrecarga de recursos baseada na diferenciação de serviços

Marcela Tassyany Galdino Santos

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Redes Computadores

Reinaldo Cézar de Morais Gomes
Anderson Fabiano Batista Ferreira da Costa
(Orientador)

Campina Grande, Paraíba, Brasil

S237a Santos, Marcela Tassyany Galdino.
Uma abordagem para gerenciamento de sobrecarga de recursos baseada na diferenciação de serviços / Marcela Tassyany Galdino Santos. – Campina Grande, 2021.
86 f. : il. color.

Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2021.
"Orientação: Prof. Dr. Reinaldo Cézar de Morais Gomes; Coorientação: Prof. Dr. Anderson Fabiano Batista Ferreira da Costa".
Referências.

1. Computação em Nuvem. 2. Máquinas Virtuais. 3. Sobrecarga de Recursos. I. Gomes, Reinaldo Cézar de Morais. II. Costa, Anderson Fabiano Batista Ferreira da. III. Título.

CDU 004(043)

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECÁRIA MARIA ANTONIA DE SOUSA CRB 15/398

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
POS-GRADUACAO CIENCIAS DA COMPUTACAO
Rua Aprigio Veloso, 882, - Bairro Universitario, Campina Grande/PB, CEP 58429-900

FOLHA DE ASSINATURA PARA TESES E DISSERTAÇÕES

MARCELA TASSYANY GALDINO SANTOS

UMA ABORDAGEM PARA GERENCIAMENTO DE SOBRECARGA DE RECURSOS BASEADA NA DIFERENCIAÇÃO DE SERVIÇOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação como pré-requisito para obtenção do título de Mestre) em Ciência da Computação.

Aprovada em: 13/08/2021

Prof. Dr. REINALDO CÉZAR DE MORAIS GOMES, UFCG, Orientador

Prof. Dr. ANDERSON FABIANO BATISTA FERREIRA DA COSTA, UFPE, Orientador

Prof. Dr. MARCUS WILLIAMS AQUINO DE CARVALHO, UFPB, Examinador Interno

Prof. Dr. PAULO DITARSO MACIEL JÚNIOR, IFPB, Examinador Externo

Documento assinado eletronicamente por **REINALDO CEZAR DE MORAIS GOMES**, **PROFESSOR(A) DO MAGISTERIO SUPERIOR**, em 16/09/2021, às 12:06, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Anderson Fabiano Batista Ferreira da Costa**, **Usuário Externo**, em 17/09/2021, às 17:23, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Paulo Ditarso Maciel Júnior**, **Usuário Externo**, em 20/09/2021, às 10:10, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da

Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Marcus Williams Aquino de Carvalho**, **Usuário Externo**, em 20/09/2021, às 15:18, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

A autenticidade deste documento pode ser conferida no site https://sei.ufcg.edu.br/autenticidade, informando o código verificador **1770486** e o código CRC **A884A1F3**.

Referência: Processo nº 23096.050352/2021-22 SEI nº 1770486

Resumo

O modelo de Infraestrutura como Serviço (do inglês, IaaS), tem atraído cada vez mais usuários, que hospedam as mais diversas aplicações, com diferentes requisitos. Isso se dá em face à redução de custos, facilidade de gerenciamento e escalabilidade. Tipicamente, os clientes tendem a superestimar suas necessidades de recursos, o que pode ocasionar subutilização, desperdício de energia e consequentemente maiores custos aos provedores. Todavia, os provedores têm buscado se beneficiar com esse cenário ao admitir mais VMs do que a quantidade de recursos físicos pode suportar. Essa técnica é chamada de sobrecarga de recursos. No entanto, essa maximização da utilização de recursos, pode resultar em redução no desempenho das cargas de trabalho em execução. Nesse sentido, há um *trade-off* entre aumentar a utilização de recursos e a ameaça de degradação de desempenho. Neste trabalho, é proposto uma abordagem para gerenciamento de sobrecarga, que visa propor um nível de sobrecarga que promova um melhor comprometimento entre provedor e cliente. A abordagem utiliza o AHP (do inglês, *Analytic Hierarchy Process*) para obter indicações do usuário quanto ao tipo de serviço, e se baseia em um algoritmo de triagem utilizado em hospitais de todo mundo, para propor a classificação das VMs de acordo com sua criticidade. Conjuntamente, foi desenvolvida uma abordagem de atribuição dinâmica de sobrecarga baseada na técnica de regressão linear, em que, a partir de dados de monitoramento, é criado um modelo de previsão de CPU para cada VM. Dessa forma, é possível definir um nível de sobrecarga que melhor se adéque ao perfil de utilização da VM e sua classe de criticidade. Como resultados, nas comparações com outras duas abordagens de sobrecarga, foi possível confirmar que a solução proposta promoveu um melhor equilíbrio no *trade-off* existente, sendo possível usufruir dos benefícios da sobrecarga e reduzir o impacto no desempenho da aplicação.

Abstract

The Infrastructure as a Service (IaaS), it has attracted more user, which host the most several application with differents requirements. That it happens because of the cost reduction, management facility and scalability. Usually, the clients tend to overestimate their resources necessity, that can cause subutilization, waste of energy and, consequently, more cost to providers. Though, the providers have been search benefit with this scenario when admite more VMs than they can support. This technique is called by Overbooking. However, maximizing resources use can result on workloads performance reduction in execution. In that regard, has a trade-off between grow up resources utilization and the threat of performance degradation. In this work, is proposed a approach to overbooking management, it aims to propose an overbooking level that promote a better commitment between provider and client. Our approach use the Analytic Hierarchy Process (AHP) to obtain user's indication related to service type, based on screening algorithm used in hospital in whole world, to propose a classification of VMs according to theirs criticality. Furthermore, it was developed a approach to dynamic attribution of overbooking based on linear regression technique, which from monitoring data is created a CPU prevision model for each VM. Thus, is possible define a overbooking level what better suits to VM utilization profile and its criticality class. The results, compared to two other overbooking approach, demonstrate that proposed approach has a better balance in there trade-off, being possible to take overbooking benefits and reduct the performance impact in the application.

Agradecimentos

Gratidão em primeiro lugar a Deus. Toda glória a Ele, sempre tão bondoso e misericordioso, me concedendo graça imerecida todos os dias. Gratidão ao meu esposo Matheus, que sempre me apoiou nos momentos mais difíceis e foi a base que eu precisava para conseguir continuar. Agradecimento especial a minha família, que também foi apoio fundamental para chegar até aqui. Gratidão também aos meus colegas de laboratório, Miqueas, Yngrid e Eduardo, cuja amizade e companheirismo tornou tudo mais leve.

Aos meus professores que me acompanharam durante o mestrado, cujo os conhecimentos transmitidos foram essenciais para minha formação. Em especial aos meus orientadores Reinaldo e Anderson, sempre humanos, prestativos e bondosos, cuja orientação foi essencial para o êxito do trabalho.

Agradeço também à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES e ao povo brasileiro pelo custeio da bolsa ao longo desta jornada. Espero poder retribuir o apoio obtido ao longo dos anos.

Sumário

Lista de Símbolos

AHP - *Analytic Hierarchy Process*

CPU - *Central Processing Unit*

CSPs - *Cloud Service Providers*

ESI - *Emergency Severity Index* IaaS - *Infrastructure as a Service*

MCDM - *Multiple-Criteria Decision-Making*

MF - *Máquina Física*

PaaS - *Platform as a Service*

SaaS - *Software as a Service*

QoS - *Quality of Service*

RC - *Razão de Coerência*

IR - *Índice de Consistência Randômico* SLA - *Service Level Agreement*

SLO - *Service Level Objective*

VM - *Virtual Machine*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O advento da Computação em Nuvem trouxe uma maior flexibilidade ao oferecer recursos sob demanda para os usuários [40]. A Infraestrutura como Serviço (IaaS) [26; 57] é um dos modelos disponíveis, que torna possível a oferta de recursos computacionais (e.g., processamento e armazenamento) que normalmente são disponibilizados em Máquinas Virtuais (VMs) [18] ou contêineres. Os recursos são obtidos de maneira escalável, na qual os usuários só pagam de acordo com seu uso. Esse modelo tem se popularizado por possibilitar que empresas de todos os portes terceirizem suas necessidades de infraestrutura [31].

Ambientes de IaaS são compostos por dois principais personagens que possuem interesses conflitantes, são eles: provedores e clientes. Os provedores oferecem infraestruturas de TI de larga escala, em forma de arrendamento de seus recursos, para usuários que desejam atender suas demandas computacionais [30]. Nesse cenário, os clientes buscam ter seus serviços e aplicações sendo executados com o desempenho esperado e os provedores almejam obter o máximo de lucro, buscando admitir tantos usuários quanto possível para maximizar a utilização de seus recursos. No entanto, esses interesses são conflitantes, pois essa máxima utilização almejada pelos provedores pode gerar interferência no desempenho das cargas de trabalho em execução [50].

Contudo, tipicamente os clientes tendem a superestimar suas necessidades de recursos [11], ou seja, as VMs (*Virtual Machines*) utilizam muito menos do que normalmente é solicitado. Análises do *Google Traces* indicaram que, em média, são usadas apenas 53% da capacidade de memória e 40% da capacidade de CPU requisitadas [54]. Essa baixa utilização gera prejuízos aos Provedores de Serviços de Nuvem (CSPs - *Cloud Service Providers*)

e ao meio ambiente, devido ao consumo de energia [31]. Cientes deste cenário os CSPs têm utilizado uma técnica chamada sobrecarga de recursos, no qual o gerenciamento oferece mais recursos do que a capacidade real existente.

A aplicação da sobrecarga de recursos parte do pressuposto de que a soma da utilização dos recursos de todas as VMs hospedadas em uma Máquina Física (MF) não ultrapassa a capacidade real disponível. No entanto, no caso de ocorrerem situações em que as cargas de trabalho passem a exigir o que foi solicitado no momento de sua alocação, pode haver congestionamento de recursos (*overhead*), o que pode causar a degradação de desempenho do serviço. Isso gera insatisfação no cliente, leva o provedor a perder credibilidade e, possivelmente, pagar multas devido as violações do SLA (*Service Level Agreement*) [27].

Portanto, um dos maiores desafios ao se utilizar sobrecarga de recursos é a decisão de qual grau de sobrecarga é o mais adequado, de forma a se obter um melhor nível de comprometimento entre provedor e cliente. Bem como, dada a heterogeneidade das cargas de trabalho em execução na nuvem, identificar quais tipos de serviço funcionam melhor com determinado nível de sobrecarga, e assim, garantir o QoS (*Quality of Service*) [54]. Com base neste problema, é que este trabalho está sendo realizado.

1.1 Problema

O uso eficiente de recursos é um tópico de grande interesse dos CSPs, principalmente devido aos custos de hardware e operacionais, bem como consumo de energia e questões ambientais [55]. No entanto, estudos têm apresentado um cenário de subutilização, que vai de encontro com os interesses dos provedores. Ghosh e Naik [27], a partir de dados coletados de uma nuvem privada interna, concluíram que, das 2193 VMs, 84% só chegam a seus níveis máximos de utilização em menos de 20% das vezes e apenas cerca de 0,7%, o que corresponde a aproximadamente 15 VMs, apresentam uma utilização máxima de CPU próxima a 100%. Conjuntamente, análises de dados realizadas por Barroso e Holzle [2] coletados a partir da observação de 5000 servidores ao longo de 6 meses, indicam um percentual de utilização de CPU variando normalmente entre 10% e 50%.

Todavia, os CSPs têm buscado se beneficiar com esse cenário ao admitir mais VMs do que a quantidade de recursos físicos pode suportar. Essa técnica é chamada de sobrecarga

de recursos. No entanto, mesmo que hajam razões evidentes para o uso dessa estratégia, não se deve ignorar os riscos inerentes à sua adoção. Os provedores pressupõem que a maioria dos clientes não utilizarão todos os recursos solicitados, ou que não haverão casos em que todos os clientes façam uso dos recursos contratados ao mesmo tempo [31]. Contudo, deve-se considerar a ameaça desses cenários, que podem resultar em prejuízos aos CSPs, devido a contenção de recursos, redução de desempenho das VMs e consequente pagamento de multas devido às violações do SLA, por não fornecerem ao cliente o que foi acordado em contrato.

Portanto, há um *trade-off* entre aumentar a utilização de recursos e a ameaça de contenção de recursos e degradação de desempenho [54]. Logo, se faz necessário encontrar qual o nível de sobrecarga de recursos mais conveniente a ser adotado de forma a gerar menores danos aos provedores e clientes, que possuem objetivos distintos. Níveis maiores geram maior utilização de recursos e economia de energia, mas por outro lado degrada a performance das VMs. Níveis menores beneficiam os clientes que têm seus serviços executando com a qualidade esperada, mas desfavorecem os CSPs que passam a ter seus recursos subutilizados [11].

Hipervisores como *Xen*, *KVM* e *VMware ESX Server*, softwares de gerenciamento de plataformas de nuvem como *OpenNebula*, *OpenStack* e *Eucalyptus* fornecem suporte a sobrecarga de recursos por meio de parâmetros de configuração que permite aos administradores indicarem o nível de sobrecarga que o sistema poderá admitir [11]. Por exemplo, por padrão, o OpenStack utiliza a proporção 16:1, para sobregarga de CPU. Isso significa que cada núcleo físico irá admitir até 16 núcleos virtuais [42]. Essa decisão é tomada por parte dos administradores da nuvem, e, devido a essa intervenção humana, pode estar mais suscetível a falhas.

Ademais, sabe-se que nem todos os serviços hospedados em nuvem possuem as mesmas características. Segundo Caglar et. al. [11] a aplicação de sobrecarga de recursos pode ser mais adequada para aplicações de processamento em lote, cuja taxa de transferência é um fator mais relevante. Entretanto, serviços mais críticos, aplicações em tempo real, como de *streaming* de vídeo (Netflix e Spotify, por exemplo), podem ser afetadas com a aplicação de sobrecarga.

1.2 Objetivos

Tendo em vista o problema supracitado e as restrições das soluções existentes, este trabalho tem como objetivo propor uma abordagem para gerenciamento dinâmico de sobrecarga de CPU. A abordagem proposta visa definir um nível de sobrecarga mais adequado com base na criticidade da VM e seu perfil de utilização de CPU, além de fornecer um melhor nível de comprometimento entre provedor e usuário.

Os objetivos específicos desta pesquisa são:

- Desenvolver uma abordagem para identificação do tipo de serviço hospedado na VM, a partir de indicações do usuário;
- Desenvolver uma abordagem para classificação de VMs de acordo com sua criticidade, com base no seu tipo de serviço e nível de SLO (*Service Level Objective*);
- Desenvolver uma abordagem de atribuição dinâmica de sobrecarga, que aplique um nível de sobrecarga apropriado, visando um menor desperdício de recursos e redução de prejuízos ao desempenho;
- Avaliar a solução desenvolvida no que se refere ao *trade-off* entre utilização de recursos e desempenho;
- Avaliar a influência da classificação por criticidade na solução proposta.

1.2.1 Organização do Documento

Esta trabalho está divido em capítulos. No Capítulo 2 é descrita a fundamentação teórica necessária para o entendimento da solução. No Capítulo 3 são expostos os principais trabalhos relacionados encontrados na literatura. No Capítulo 4 é apresentado o mecanismo proposto neste trabalho. No Capítulo 5 é detalhado o planejamento realizado para execução dos experimento. No Capítulo 6 são descritos os resultados alcançados com os experimentos. Por fim, no Capítulo 7 são apresentadas as conclusões alcançadas através da realização da pesquisa, limitações e direcionamentos para possíveis trabalhos futuros.

Capítulo 2

Fundamentação Teórica

As próximas seções descreverão os principais conceitos e tecnologias relacionadas à solução proposta. Serão abordados conceitos base como Computação em Nuvem, Sobrecarga de Recursos e SLA, bem como a descrição teórica dos métodos e algoritmos que embasaram o desenvolvimento da solução proposta.

2.1 Computação em Nuvem

A Computação em Nuvem (*do inglês, Cloud Computing*) é um modelo que torna possível a disponibilização de recursos virtuais (*hardware* ou *software*) por meio da Internet. É caracterizado como um serviço em que os recursos são fornecidos dinamicamente, de acordo com a necessidade (*on-demand*), de forma escalável (*elasticity*), no qual a localização desses recursos é transparente para os clientes (*Location-independent resource pool*), que, por sua vez, só pagam de acordo com seu uso (*pay per use*) [40].

Diversos serviços podem ser oferecidos em nuvem para atender as diferentes necessidades dos clientes, entre eles três principais modelos de serviços se destacam como sendo os mais utilizados [26; 57], são eles: **Infraestrutura como Serviço (IaaS)**, onde são oferecidos recursos computacionais (e.g processamento e armazenamento); **Plataforma como Serviço (PaaS)**, onde plataformas para desenvolvimento e execução de aplicações são oferecidos (e.g. Windows Azure Cloud, AWS, RedHat OpenShift); e **Software como Serviço (SaaS)**, que oferece sistemas computacionais completos (e.g., sistemas de *streaming*, redes sociais, sistemas de escritório).

2.1.1 Gerenciamento de Recursos

Um dos grandes desafios da Computação em Nuvem está no gerenciamento de recursos, visto que a própria natureza da nuvem adiciona complexidade ao processo de gerenciamento de recursos [51]. Isso se dá porque os comportamentos das cargas de trabalho variam drasticamente, o que torna difícil o processo de dimensionamento de infraestrutura para suportar o uso máximo e ao mesmo tempo distanciar-se de cenários onde recursos são subutilizados [32]. A Figura 2.1 exibe os principais componentes do processo de gerenciamento de nuvem, segundo Carvalho et. al [17].

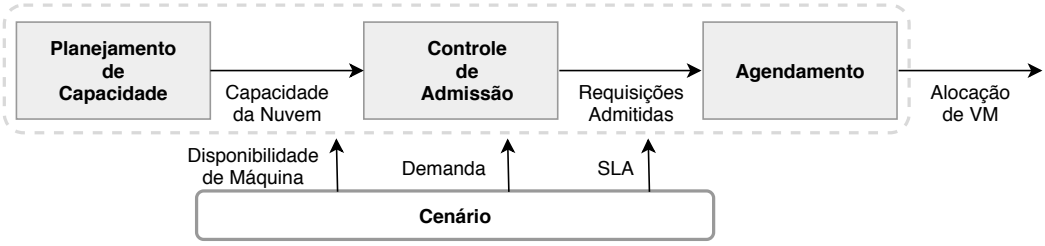


Figura 2.1: Processo de Gerenciamento de Recursos em Nuvem [17].

No **planejamento de capacidade**, o provedor busca estimar a quantidade de recursos computacionais necessária para atender sua demanda eficazmente [48]. No **controle de admissão**, é definido quais requisições de tarefas ou VMs serão aceitas, levando em consideração o que já foi admitido no sistema [17]. Na etapa de **agendamento**, é definido para qual máquina virtual determinada tarefa será atribuída, ou em qual máquina física a VM será alocada [33].

2.1.2 Sobrecarga de Recursos

No gerenciamento de recursos, uma das soluções adotadas pelos provedores de nuvem para maximizar a utilização de sua infraestrutura, admitir mais clientes e, consequentemente, obter mais lucro, é a sobrecarga de recursos [11]. A técnica de reservar além das capacidades reais disponíveis já é utilizada em diversas áreas como em sistemas de hospedagem, nos quais mais hóspedes são aceitos do que o número de quartos disponíveis, ou em aeronaves em que são admitidas mais passageiros do que a quantidade de assentos livres, assumindo a possibilidade de cancelamento ou ausência dos clientes [31]. Entretanto, no caso de com-

parecerem mais passageiros ou hóspedes do que o esperado, as empresas terão sua imagem prejudicada e terão que ressarcir os danos causados aos seus clientes.

No contexto de computação em nuvem, Tomás e Tordsson [54] definem que a sobrecarga de recursos "*descreve o gerenciamento de recursos de qualquer maneira em que a capacidade total disponível seja menor que a capacidade máxima teórica solicitada.*". Assim como nas demais áreas, provedores em nuvem também assumem riscos quando o excesso de reservas é aplicado. A adição de sobrecarga pode ocasionar problemas de desempenho e violações de SLA que possivelmente resultam em multas pagas aos clientes. Desse modo, os provedores buscam aplicar níveis de sobrecarga que não representem descumprimentos do SLA [3].

2.1.3 SLA (*Service Level Agreement*)

No contexto de computação em nuvem, o SLA trata-se de um contrato realizado entre cliente e provedor quanto à prestação de um serviço [43], ou seja, o SLA descreve o serviço, seu desempenho esperado, bem como as obrigações e deveres dos clientes e provedores [19; 59]. Esse contrato é elaborado visando garantir os requisitos de Qualidade de Serviço (QoS). Para isso, os SLAs dispõem de propriedades mensuráveis chamadas SLOs (*Service-Level Objective*) que definem as métricas utilizadas para monitoramento e os níveis esperados para essas métricas, de forma a garantir o desempenho [16]. Os SLOs podem definir valores para disponibilidade, quantidade de recursos, tempo de resposta, entre outras métricas [43].

Caso um SLO não seja cumprido por um determinado período de tempo, isso implicará em penalidades para os provedores e insatisfação do cliente. Por este motivo, os SLAs são compostos de um conjunto de cláusulas que definem as sanções para os SLOs não cumpridos. Rana et. al. [43] fornecem o seguinte exemplo de cláusula de violação de SLA: "*Se 90% do número de CPUs e 90% de memória foi entregue, esses SLOs não foram violados. Para provisionamento abaixo de 90% de CPU e memória, o provedor deve incorrer em uma penalidade de α unidades monetárias*".

2.2 AHP - Analytic Hierarchy Process

O AHP é um conhecido método de Tomada de Decisão com Múltiplos Critérios (em inglês, *Multiple-Criteria Decision-Making* ou MCDM) que utiliza matemática e psicologia para resolver problemas mais complexos [44]. Desenvolvido por Thomas L. Saaty em 1970, essa abordagem tem sido analisada e evoluída desde então, sendo utilizada em todo mundo nas mais diversas áreas como economia, saúde, política e computação [46]. As próximas subseções descrevem as etapas que devem ser percorridas para a execução do método AHP.

2.2.1 Etapa 1 - Hierarquização do Problema

O método AHP propõe que o primeiro passo no processo de tomada de decisão é conhecer o problema e todas as partes envolvidas. Portanto, é necessário estruturar o problema de maneira hierárquica, quebrando-o em partes menores. Essa hierarquia deve ser composta por um objetivo, critérios (podendo também dispor de sub-critérios) e alternativas, como pode ser visto na Figura 2.2.

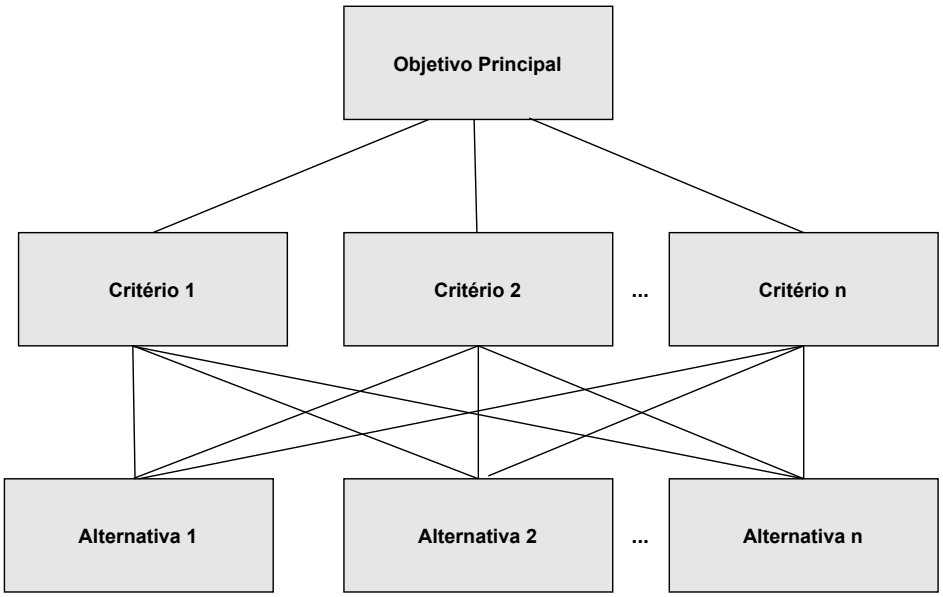


Figura 2.2: Estrutura Hierárquica Geral do Método AHP [20].

Por exemplo, suponha que um cliente precisa escolher entre três veículos que lhe foi apresentado na concessionária. Os fatores que ele irá considerar para decidir são *status*, potência e conforto. Esse cliente optou por usar o AHP para realizar essa tomada de decisão.

Nesse sentido, estruturou o problema hierarquicamente, da seguinte forma:

- **Objetivo Principal**: Adquirir um veículo.
- **Critérios**: Status, Potência e Conforto.
- **Alternativas**: Veículo 1, Veículo 2 e Veículo 3.

2.2.2 Etapa 2 - Matrizes de Comparação Pareada

Após estruturar o problema em uma hierarquia, matrizes de comparação são construídas, com o intuito de expressar a importância relativa entre os atributos, ou seja, uma série de comparações pareadas são realizadas, de forma que cada componente de uma camada da hierarquia, seja comparado em relação aos itens da camada superior.

A matriz de comparação é uma matriz quadrada $n \times n$, em que, dado um conjunto de alternativas $A_1, A_2, ..., A_n$, no qual $A = (a_{ij})$ e $i,j = \{1, 2, ..., n\}$. A posição i,j representa a importância relativa da alternativa A_i em relação a alternativa A_j. Outro aspecto importante é que a matriz de comparação é recíproca, isso significa que os valores acima da diagonal principal são o inverso do valor abaixo, ou seja, $a_{ij} = \frac{1}{a_{ij}}, a_{ij} \neq 0$.

Deste modo, a matriz de comparação apresenta o seguinte formato:

$$A = \begin{bmatrix} 1 & a_{12} & ... & a_{1n} \\ \frac{1}{a_{12}} & 1 & ... & a_{2n} \\ ... & ... & ... & ... \\ \frac{1}{a_{1n}} & \frac{1}{a_{2n}} & ... & 1 \end{bmatrix} \tag{2.1}$$

Para que sejam utilizados valores que indiquem essa importância relativa, é necessário que os julgamentos dos tomadores de decisão sejam convertidos para um valor inteiro. Essa conversão se dá utilizando a Escala Fundamental de Saaty. Após alguns estudos utilizando Teoria da Psicologia, Saaty concluiu que para o ser humano é mais adequado realizar comparações pareadas com valores inferiores a 9 [45]. Nesse sentido, a escala compreende um intervalo de 1 a 9, variando de igual a extremamente importante, conforme é exibido na Tabela 2.1.

Valor	Definição	Explicação
1	Igual importância	Os dois critérios contribuem de forma idêntica para o objetivo
3	Pouco mais importante	A análise e o julgamento mostram que um critério é um pouco mais importante que o outro
5	Mais importante	A análise e o julgamento mostram que um critério é claramente mais importante que o outro
7	Muito mais importante	A análise e o julgamento mostram que um dos critérios é predominante para o objetivo
9	Extremamente mais importante	Sem sombra de dúvidas um dos critérios é absolutamente predominante para o objetivo
2,4,6,8	Valores Intermediários	Quando se procura um valor intermediário entre duas definições

Tabela 2.1: Escala Fundamental de Saaty.

Para o problema da escolha do veículo mencionado na Etapa 1, o cliente primeiramente realiza uma comparação pareada entre os critérios, usando a Escala Fundamental de Saaty, de forma a expressar qual critério é mais e menos importante para o objetivo principal que é adquirir o veículo. A Tabela 2.2 demonstra uma possível resposta do cliente. Por exemplo, quando comparado o critério *status* em relação ao critério potência, foi utilizado o valor 5 na escala fundamental de Saaty, indicando que para o cliente, o critério *status* é mais importante que a potência.

	status	**potência**	**conforto**
status	1	5	4
potência	$\frac{1}{5}$	1	$\frac{1}{3}$
conforto	$\frac{1}{4}$	3	1

Tabela 2.2: Matriz de comparação pareada para os critérios.

Após a realização das comparações entre os critérios, é realizado a comparação pareada entre cada uma das alternativas (Veículo 1, Veículo 2 e Veículo 3) em relação a cada critério. Por exemplo, na Tabela 2.3, é apresentada a comparação dos veículos em relação ao *status*. Nesse caso, por exemplo, quando comparado o Veículo 3 em relação ao Veículo 1 o cliente usou o valor 3 na escala de Saaty, indicando então que o Veículo 3, no que diz respeito ao status é um pouco mais importante que o Veículo 1. As Tabelas 2.4 e 2.5 exemplificam os julgamentos do cliente para as alternativas em relação aos demais critérios considerados.

Critério: Status			
	Veículo 1	**Veículo 2**	**Veículo 3**
Veículo 1	1	4	$\frac{1}{3}$
Veículo 2	$\frac{1}{4}$	1	$\frac{1}{7}$
Veículo 3	3	7	1

Tabela 2.3: Matriz de Comparação pareada para as alternativas considerando o critério Status.

Critério: Potência			
	Veículo 1	**Veículo 2**	**Veículo 3**
Veículo 1	1	$\frac{1}{6}$	1
Veículo 2	6	1	6
Veículo 3	1	$\frac{1}{6}$	1

Tabela 2.4: Matriz de Comparação pareada para as alternativas considerando o critério Potência.

Critério: Conforto			
	Veículo 1	**Veículo 2**	**Veículo 3**
Veículo 1	1	5	8
Veículo 2	$\frac{1}{5}$	1	3
Veículo 3	$\frac{1}{8}$	$\frac{1}{3}$	1

Tabela 2.5: Matriz de Comparação pareada para as alternativas considerando o critério Conforto.

2.2.3 Etapa 3 - Consistência dos Julgamentos

Um aspecto importante que deve ser analisado é a consistência desses julgamentos, por exemplo, considerando uma relação de transitividade: Se A é melhor que B, e B é melhor que C, então A deve ser melhor que C. Do contrário, existe alguma inconsistência nos julga-

mentos. Para esse caso, no entanto, o problema é facilmente percebido, mas em problemas maiores torna-se mais complexo checar a consistência visualmente.

Por esse motivo, é realizado o cálculo da Razão de Coerência (RC), expressa pela Equação 2.2. Em que λ_{max} representa o autovalor máximo da matriz e n a ordem da matriz. O IR é o Índice de consistência Randômico, para matrizes recíprocas com elementos não-negativos de ordem 1 a 15. Esses valores foram propostos por Saaty em laboratório [47] e podem ser visualizados na Tabela 2.6. Para que uma matriz seja considerada consistente o valor de RC deve ser menor ou igual a $0{,}1$. Do contrário, é necessário rever os julgamentos.

$$RC = \frac{\lambda_{max} - n}{(n-1) \cdot IR} \tag{2.2}$$

n	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
IR	0,00	0,00	0,58	0,90	1,12	1,24	1,32	1,41	1,45	1,49	1,51	1,48	1,56	1,57	1,59

Tabela 2.6: Índice Randômico Médio do AHP [47].

Na Tabela 2.7 é expresso o resultado do cálculo do RC para cada matriz de comparação definida para o problema da escolha do veículo. É possível observar que os resultados foram inferiores a 0,1. Isso indica que o cliente realizou os julgamentos de forma consistente. Portanto, é possível seguir com o processo para escolha da alternativa.

	$\lambda - max$	IC	RC
Matriz de Critérios	3,09	0,04	0,07
Matriz de Alternativas para o critério Status	3,03	0,02	0,03
Matriz de Alternativas para o critério Potência	3,00	0,00	0,00
Matriz de Alternativas para o critério Conforto	3,04	0,02	0,04

Tabela 2.7: Cálculo da Razão de Coerência para cada uma das matrizes de comparação pareada.

2.2.4 Etapa 4 - Cálculo das prioridades

O passo seguinte no processo, consiste em calcular as prioridades de cada matriz. A prioridade permite classificar a importância do critério ou da alternativa na decisão. Ao todo, três

tipos de prioridade são calculados: as ***prioridades dos critérios*** que se refere a importância de cada critério em relação ao objetivo; as ***prioridades alternativas locais*** que diz respeito a importância de cada alternativa em relação a cada critério e as ***prioridades alternativas globais*** que relaciona as *prioridades dos critérios* com as *prioridades alternativas locais*, classificando as alternativas em relação aos critérios e consequentemente ao objetivo.

As prioridades são idealmente obtidas pelo cálculo algébrico dos autovetores diretos associados aos autovalores máximos das matrizes de comparações pareadas. No entanto, para simplificar o entendimento, aqui será exposto um método que usa o cálculo da média geométrica, que apresentam resultados extremamente próximos aos obtidos com o cálculo algébrico do autovetor matricial.

Nesse sentido, é necessário primeiramente calcular a média geométrica de cada linha da matriz. Considerando um conjunto formado por números reais positivos X_1,X_2,X_3,...,X_n, a média geométrica (M_{G_i}) pode ser expressa pela raiz n-ézima do produto dos termos desse conjunto, como pode ser visto na equação 2.3. Por meio da média geométrica é possível ter o peso de cada critério.

$$M_G = \sqrt[n]{X_1 \cdot X_2 \cdot X_3 \cdot ... \cdot X_n} \tag{2.3}$$

Seguidamente, é necessário calcular o peso normalizado (W_c) para cada linha da matriz, que equivale a razão entre a média geométrica de cada linha e o somatório das médias geométricas calculadas para todas as linhas, conforme expresso na equação 2.4. O peso normalizado para a matriz de critérios indica a prioridade dos critérios, e o peso normalizado para as matrizes das alternativas indicam as prioridades alternativas locais.

$$W_c = \frac{M_G}{\sum_{j=1}^{n} M_{G_j}} \cdot 100 \tag{2.4}$$

Dando continuidade ao exemplo da escolha do veículo, é possível observar nas Tabelas 2.8 a 2.11 a aplicação das equações 2.3 e 2.4 para definir as prioridades dos critérios e as prioridades alternativas locais. Percebe-se, por exemplo, que o critério *status* tem uma maior importância em relação aos demais critérios (67,38%).

					Prioridade dos Critérios
	status	**potência**	**conforto**	**Peso (M_G)**	**Peso Normalizado (W_c)**
status	1	5	4	2,71	$\frac{2,71}{(2,71+0,41+0,91)} \cdot 100 = 67,38\%$
potência	$\frac{1}{5}$	1	$\frac{1}{3}$	0,41	$\frac{0,41}{(2,71+0,41+0,91)} \cdot 100 = 10,07\%$
conforto	$\frac{1}{4}$	3	1	0,91	$\frac{0,91}{(2,71+0,41+0,91)} \cdot 100 = 22,55\%$

Tabela 2.8: Definição da prioridade dos critérios a partir do cálculo do peso normalizado.

	Critério: Status				**Prioridade Alternativa Local**
	Veículo 1	**Veículo 2**	**Veículo 3**	**Peso (M_G)**	**Peso Normalizado (W_c)**
Veículo 1	1	4	$\frac{1}{3}$	1,10	$\frac{2,71}{(1,10+0,33+2,76)} \cdot 100 = 26,28\%$
Veículo 2	$\frac{1}{4}$	1	$\frac{1}{7}$	0,33	$\frac{0,33}{(1,10+0,33+2,76)} \cdot 100 = 7,86\%$
Veículo 3	3	7	1	2,76	$\frac{2,76}{(1,10+0,33+2,76)} \cdot 100 = 65,86\%$

Tabela 2.9: Definição das prioridades alternativas locais em relação ao critério status a partir do cálculo do peso normalizado.

	Critério: Potência				**Prioridade Alternativa Local**
	Veículo 1	**Veículo 2**	**Veículo 3**	**Peso (M_G)**	**Peso Normalizado (W_c)**
Veículo 1	1	$\frac{1}{6}$	1	0,55	$\frac{0,55}{(0,55+3,30+0,55)} \cdot 100 = 12,50\%$
Veículo 2	6	1	6	3,30	$\frac{3,30}{(0,55+3,30+0,55)} \cdot 100 = 75,00\%$
Veículo 3	1	$\frac{1}{6}$	1	0,55	$\frac{0,55}{(0,55+3,30+0,55)} \cdot 100 = 12,50\%$

Tabela 2.10: Definição das prioridades alternativas locais em relação ao critério potência a partir do cálculo do peso normalizado.

	Critério: Conforto				**Prioridade Alternativa Local**
	Veículo 1	**Veículo 2**	**Veículo 3**	**Peso (M_G)**	**Peso Normalizado (W_c)**
Veículo 1	1	5	8	3,42	$\frac{3,42}{(3,42+0,84+0,35)} \cdot 100 = 74,18\%$
Veículo 2	$\frac{1}{5}$	1	3	0,84	$\frac{0,84}{(3,42+0,84+0,35)} \cdot 100 = 18,30\%$
Veículo 3	$\frac{1}{8}$	$\frac{1}{3}$	1	0,35	$\frac{0,35}{(3,42+0,84+0,35)} \cdot 100 = 7,52\%$

Tabela 2.11: Definição das prioridades alternativas locais em relação ao critério conforto a partir do cálculo do peso normalizado.

O último passo consiste em construir uma matriz de decisão $n \times m$ (nº de alternativas x nº de critérios), composta pelas prioridades dos critérios e as prioridades alternativas locais calculadas anteriormente. E, por fim, realizar o processo de ponderação aditiva, que consiste em realizar a soma do produto entre o vetor das prioridades dos critérios e o vetor das prioridades alternativas locais. Esse processo resulta em um vetor com as prioridades alternativas globais. A alternativa com maior prioridade indica a alternativa a ser escolhida. Na Tabela 2.12, é possível observar os valores calculados para o exemplo da escolha do veículo, onde o veículo 3 é a alternativa mais adequada para atender ao objetivo do cliente.

	Status	**Potência**	**Conforto**	**Prioridade Alternativas Globais**
Critérios	67,38%	10,07%	22,55%	
Veíc. 1	26,28%	12,50%	74,18%	$((0,6728 \cdot 0,2628) + (0,1007 \cdot 0,1250) + (0,2255 \cdot 0,7418)) \cdot 100 = 35,69\%$
Veíc. 2	7,86%	75,00%	18,30%	$((0,6728 \cdot 0,786) + (0,1007 \cdot 0,75) + (0,2255 \cdot 0,1830)) \cdot 100 = 16,97\%$
Veíc. 3	65,86%	12,50%	7,52%	$((0,6728 \cdot 0,6586) + (0,1007 \cdot 0,1250) + (0,2255 \cdot 0,752)) \cdot 100 =$ **47,33**%

Tabela 2.12: Definição das prioridades alternativas globais por meio do cálculo da ponderação aditiva.

2.3 Índice de Gravidade de Emergência

Os Departamentos de Emergência (DE) têm recebido um número cada vez maior de pacientes, o que favorece a superlotação de hospitais, caso não haja a gestão adequada [29]. Nesse cenário, a triagem é um processo de extrema importância, pois permite definir a acuidade dos pacientes e estipular quais desses pacientes precisam de maior ou menor urgência no

atendimento. O Índice de Gravidade de Emergência (ESI, do inglês *Emergency Severity Index*) é um algoritmo de triagem utilizado em hospitais de todo mundo, a fim de definir a criticidade do paciente com base em seus sintomas e quantidade de recursos necessários para seu atendimento [23].

O ESI foi desenvolvido pelos médicos de emergência Richard Wuerz e David Eitel nos Estados Unidos e, ao longo do tempo, passou por inúmeros estudos e atualizações, sendo hoje considerado como aplicável para diferentes departamentos de emergência, como urbanos, rurais, acadêmicos e comunitários [29]. Na Figura 2.3 é exibido um fluxograma que descreve o algoritmo utilizado no ESI. O algoritmo estratifica os pacientes em níveis de 1 (mais urgente) a 5 (menos urgente) com base em seu estado clínico, sinais vitais e quantidade de recursos necessários para o seu atendimento (1 - ressuscitação, 2 - emergente, 3 - urgente, 4 - menos urgente, 5 - não urgente).

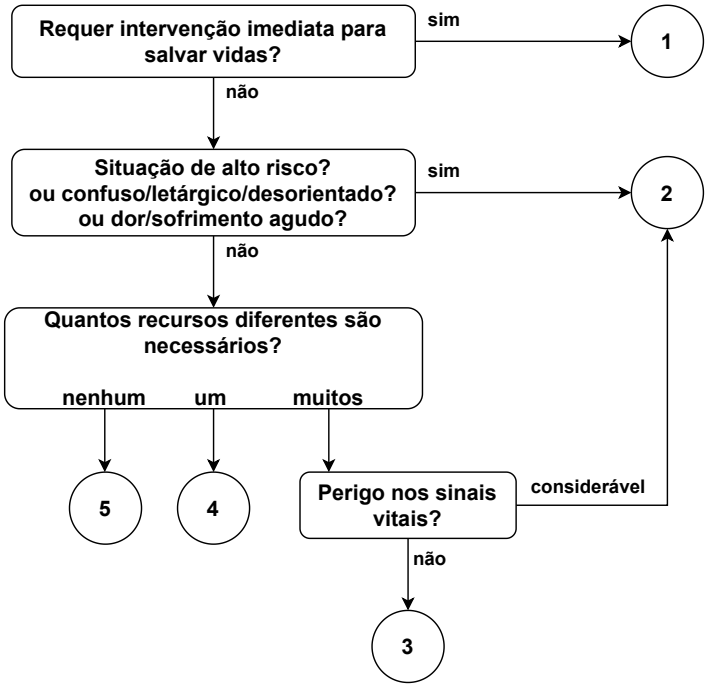


Figura 2.3: Fluxograma - Índice de Gravidade de Emergência.

Nos departamentos de emergência, a gestão de recursos é extremamente importante e um passo indispensável para uma gestão eficaz é identificar o nível de acuidade do paciente. Uma identificação adequada permite que pacientes mais graves recebam atendimento rápido e apropriado, bem como evita que pacientes de menor gravidade ocupem leitos e recursos, sem a devida necessidade. Ademais, essa gestão é essencial para mitigar cenários de super-

lotação, como em casos de pandemias, por exemplo. Este trabalho se baseia no ESI para estabelecer uma definição da criticidade das VMs que serão alocadas na nuvem (vide Seção 4.2). Considera-se que, assim como em ambientes de saúde, a gestão de recursos computacionais requer a identificação e classificação das VMs de acordo com sua criticidade, para assim, ser possível obter indicativos a respeito da quantidade de recursos necessários e da tolerância do serviço em relação a oscilações no desempenho.

+–

Capítulo 3

Trabalhos Relacionados

Neste capítulo serão abordados trabalhos relacionados à pesquisa que está sendo descrita neste documento. Inicialmente, foi feita uma revisão da literatura com o intuito de identificar trabalhos que pudessem contribuir direta ou indiretamente com esta pesquisa. Para isso, foram avaliados as metodologias, objetivo, abordagens e técnicas apresentadas. Portanto, foi possível identificar alguns trabalhos que apresentaram uma maior afinidade a esta proposta, que estão descritos a seguir.

De acordo com Breitgand et al. em [8], o *trade-off* entre sobrecarga de recursos e degradação de desempenho está associado ao gerenciamento do SLA. Breitgand et al. apresentaram uma extensão do SLA padrão (eSLAs), para admitir a possibilidade de novas VMs serem iniciadas com êxito. Adicionalmente, propuseram um *framework* que utiliza a demanda efetiva da *Cloud* para estimar a capacidade física total necessária para realizar a sobrecarga de recursos, em concordância com a disponibilidade definida pelos eSLAs. Entretanto, este trabalho não leva em consideração a heterogeneidade dos serviços, suas diferentes tolerâncias a desempenho e não utiliza o SLO diretamente na definição do nível de sobrecarga mais adequado a ser utilizado.

Algumas pesquisas desenvolveram suas abordagens de sobrecarga de recurso com módulos de reconhecimento de risco que buscam evitar violações do SLA . Ghosh et al. em [27], desenvolveram uma abordagem para estimar os riscos relacionados a sobrecarga de recursos, baseada em limites que expressam a probabilidade de violação do SLA em virtude da sobrecarga. Tomas e Tordsson em [55] implementaram um *framework* para sobrecarga de recurso com reconhecimento de risco, almejando aumentar a utilização de recursos de forma

equilibrada e com menor degradação de desempenho. Estes trabalhos partem da perspectiva de que o nível de sobrecarga de recursos utilizado não deve resultar em violações do SLA. Na solução proposta neste documento, o SLO é utilizado para definir a criticidade da VM, que por sua vez define limiares aceitáveis para a sobrecarga de recursos a ser empregada.

O *framework* apresentado por Tomas e Tordsson em [55] define o risco relacionado à admissão de uma nova VM. A VM é aceita se o risco calculado estiver abaixo do nível aceito pelo sistema. Esse nível, definido autonomamente, leva em consideração o grau de utilização de recursos desejado no *datacenter* e é obtido a partir do cálculo de fronteira de pareto, com o intuito de alcançar equilíbrio na utilização em todas as dimensões de capacidade (Memória, CPU, E/S). No caso da VM ser admitida, o módulo SOS (*Smart Overbooking Scheduler*) determina o seu posicionamento no *datacenter*, buscando evitar que o sistema exceda a capacidade de recursos real em todas as máquinas físicas do *datacenter*. O módulo SOS dá prioridade a alocação de VMs em máquinas físicas sem sobrecarga e se isso não for possível a alocação é feita nas máquinas físicas com sobrecarga de recursos. Esse critério de alocação difere da solução proposta neste documento, em que o nível de sobrecarga é atribuído por VM a partir da previsão da utilização de CPU e da classe de criticidade.

Caglar et al. em [11] propuseram o iOverbook, uma abordagem que usa aprendizado de máquina para determinar de forma *on-line* e autônoma o grau de sobrecarga de recursos para cada máquina física do *datacenter*. O iOverbook utiliza dados históricos de rastreamento de *datacenter*, disponibilizados pelo Google, para treinar suas redes e extrair padrões de forma a prever o uso futuro de recursos e o desempenho médio esperado dos *hosts*. Logo, a abordagem propõe o nível mais adequado de sobrecarga a ser utilizado por um determinado período de tempo, de forma a evitar violações de SLA. Entretanto, não é levado em consideração que as VMs hospedadas em uma mesma máquina física têm diferentes requisitos de desempenho. Portanto, o emprego de um único nível de sobrecarga por máquina física pode desfavorecer algumas VMs mais sensíveis ao desempenho. E, por outro lado, pode promover um maior desperdício de recursos, por não ser possível utilizar um maior nível de sobrecarga a fim de evitar impactos à performance dessas VMs mais críticas.

Farahnakian et al. propuseram em [25] o LiRCUP, uma abordagem que utiliza regressão linear para previsão da utilização de CPU dos *hosts*, que é utilizada para identificar quando os *hosts* estão sobrecarregados ou subutilizados e assim ser possível decidir quando migrar

as VMs, objetivando evitar violações de SLA e reduzir o consumo de energia. O trabalho aqui proposto também utiliza a técnica de regressão linear para prever o uso futuro de CPU. No entanto, o modelo de predição é criado para prever a utilização de CPU da VM e aplicado no contexto de sobrecarga de recursos. Diferentemente, a LiRCUP é voltada para predição de utilização de CPU dos hosts, aplicada ao contexto de migrações de máquinas virtuais.

Ying e Liu em [62] apresentaram a Sponge, uma estratégia para sobrecarga de recursos que, diferente dos demais trabalhos, não parte da perspectiva de definir os níveis de sobrecarga de recursos que o sistema poderá admitir. A Sponge permite a alocação de VMs além das capacidades físicas e monitora o uso de recursos para saber qual a quantidade mínima deve ser atribuída a cada VM sem que haja prejuízo no seu desempenho, mas ao mesmo tempo buscando evitar cenários com *overhead*. Similarmente, a solução desenvolvida parte da perspectiva de considerar os requisitos individuais de cada VM, mas além disso são considerados fatores como SLO e o tipo de serviço, a fim de evitar que hajam prejuízos no desempenho.

Tao et al. em [53] apresentaram a kFHCO (*k-Factor Horizontal Checkpoint Oversubscription*), uma abordagem para posicionamento de VMs que utiliza sobrecarga de recursos. O objetivo da kFHCO é maximizar a consolidação e ao mesmo tempo reduzir a chance de haver *overhead*. Para isso, é utilizado o SLA para definir os tipos de VM, e a partir disso poder indicar quais VMs migrar em casos de *overhead*. Este trabalho se aproxima dessa proposta por utilizar o SLA na classificação das VMs. No entanto, os autores fazem uso dessa informação para determinar quais VMs migrar, o que difere da solução proposta neste documento, no qual o SLO é utilizado na classificação das VMs com o intuito de contribuir com a definição dos níveis aceitáveis de sobrecarga a serem utilizados.

Conjuntamente, Bezerra propôs em [6] a VMPOS, uma abordagem para o posicionamento de VMs que considera múltiplos objetivos a serem atendidos, que são evoluídos simultaneamente através de um algoritmo memético. O trabalho também utiliza a sobrecarga de recursos e a criticidade das VMs para buscar garantir o desempenho de serviços considerados críticos. Diferente do trabalho aqui proposto, a VMPOS não leva em consideração o SLA e classifica as VMs atribuindo-as a apenas dois grupos: crítico e não crítico.

Além dos trabalhos supracitados, o sistema de gerência de recursos do Google, chamado Borg, tem sido amplamente referenciado pela comunidade científica [58]. O Borg utiliza

sobrecarga de recursos ao alocar mais tarefas do que a quantidade de recursos disponível. O sistema se baseia na prioridade das tarefas, em que tarefas de maior prioridade recebem um melhor tratamento, o que inclui a preempção de tarefas de menor prioridade. Na abordagem proposta neste trabalho, também é considerada uma classificação por prioridade. No entanto, tendo em vista a dificuldade em se definir "se"e "quando"um serviço pode ser interrompido, optou-se por haver um nível de "penalização"quanto à obtenção de recursos, que varia de acordo com a criticidade da VM. Assim, tarefas de menor criticidade podem ter uma redução de recursos, mas elas não são desalocadas ou deixam de executar em favor de outras.

No que se refere a classificação de tarefas, Ergu et al. propuseram em [24] um modelo de alocação que faz uso do AHP para estabelecer a prioridade das tarefas, a fim de que os recursos de nuvem sejam atribuídos de acordo com sua prioridade. Semelhantemente, Makwe et al. em [38] também utilizaram o AHP para tomada de decisão de alocação de tarefas com base em requisitos computacionais (memória principal, MIPS, armazenamento, etc). Além desses, outros trabalhos têm considerado o AHP para tomada de decisão no contexto de computação em nuvem [1; 52; 41]. No entanto, não foram encontrados trabalhos que utilizem o AHP para a classificação do tipo de serviço e no contexto de sobrecarga de recursos.

Apesar das inegáveis contribuições já realizadas pela comunidade científica, alguns aspectos ainda não foram considerados em conjunto. No melhor do nosso conhecimento, não foram encontrados estudos que buscam propor abordagens para o gerenciamento dinâmico de sobrecarga de CPU baseado na criticidade da VM. Bem como, abordagens no contexto de sobrecarga de recursos que se baseiem no algoritmo de triagem ESI e façam uso do AHP como método de tomada de decisão com múltiplos critérios.

Capítulo 4

Abordagem Proposta

Este trabalho visa desenvolver uma abordagem para aplicação de sobrecarga de CPU baseada na diferenciação dos tipos de serviço. O objetivo da solução proposta é atribuir um nível de sobrecarga que gere um menor desperdício de recursos e menor consumo de energia, mas que também mitigue ameaças de prejuízo ao desempenho dos serviços em execução nas VMs. Para isso, a abordagem se subdivide em três processos: identificação do tipo de serviço, definição da classe de criticidade e atribuição do nível de sobrecarga. As próximas seções se concentraram em descrever cada um desses processos.

4.1 Identificação do Tipo de Serviço

Considerando um cenário com um provedor de nuvem IaaS, o processo de alocação de uma VM é desencadeado a partir da solicitação do usuário. A identificação do tipo de serviço visa ter uma evidência do que será executado na VM a partir de indicativos do usuário. Para isso, utiliza-se o AHP como método de tomada de decisão com múltiplos critérios. Deste modo, os usuários indicam quais recursos entre CPU, memória e disco são mais importantes para execução do seu serviço por meio de comparações pareadas, utilizando a escala fundamental de Saaty (Vide Seção 2.2.2).

Conforme descrito na Seção 2.2, o primeiro passo a ser realizado quando utilizado o AHP, é definir a estruturação do problema (objetivo, critérios e alternativas). A estruturação proposta neste trabalho pode ser observada na Figura 4.1.

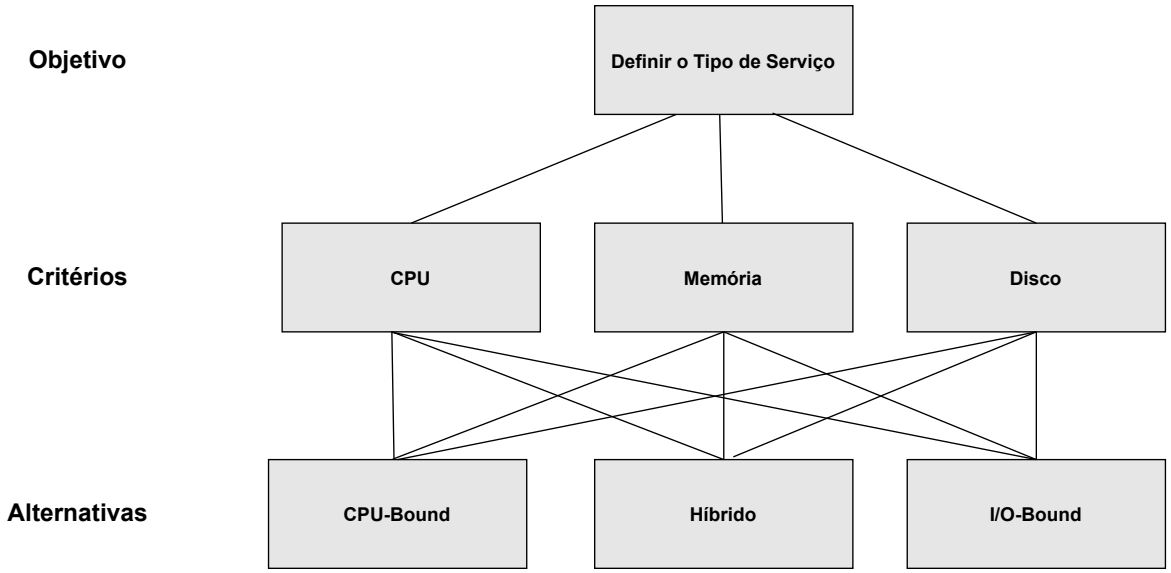


Figura 4.1: Estruturação Hierárquica do Problema

O **objetivo** consiste em definir o tipo de serviço, ou seja, identificar em qual categoria se enquadra o serviço/aplicação que será executado na VM. Um tipo de serviço está fortemente relacionado à importância dos recursos computacionais. Espera-se que para uma aplicação CPU-bound, a CPU deva ter uma maior relevância do que a capacidade de armazenamento, por exemplo. Logo, os **critérios** definidos na estruturação do problema, são os recursos computacionais, por meio dos quais os usuários poderão definir quais recursos têm maior ou menor prioridade para o serviço que será hospedado na VM. Por fim, as **alternativas** são os tipos de serviço: CPU-Bound, I/O-Bound e Híbrido. Os dois primeiros já são bem conhecidos, o último considera-se que seja um tipo de serviço no qual a memória é mais importante, mas ainda requer um consumo de CPU elevado.

O cliente será o responsável por realizar as comparações pareadas entre os critérios para ser possível calcular a prioridade dos critérios. Isto significa que, fazendo uso da escala fundamental de Saaty, o cliente irá indicar quais dos recursos são mais importantes, por meio de comparações par a par. Em um cenário real, conjectura-se a utilização de uma interface gráfica em que o usuário, no momento de solicitação de criação da VM, possa realizar a comparação entre os recursos, movendo o cursor em direção ao recurso que considera mais importante. Por meio das comparações pareadas, é possível realizar o cálculo para a criação de um vetor indicando a prioridade de cada critério.

O administrador da nuvem realizará as comparações pareadas para indicar a prioridade

das alternativas locais em relação a cada critério. Por exemplo, para o critério CPU, o administrador irá julgar por meio de comparações pareadas, quais dos tipos de serviço é mais importante para CPU. Nesse caso, espera-se, por exemplo, que a alternativa CPU-Bound apresente uma maior prioridade em relação às demais, devido a sua característica inerente de requisitar uma maior utilização de processamento. As prioridades dos critérios e as prioridades das alternativas locais são então utilizadas para calcular a prioridade das alternativas globais, e assim, a alternativa com maior prioridade é a escolhida como o tipo de serviço da VM.

Entre os métodos MCDM, o AHP foi escolhido devido à possibilidade de converter a percepção dos usuários em uma forma quantificável. Além disso, os seres humanos não conseguem comparar muitos itens ao mesmo tempo de forma a tomar uma decisão eficaz [9]. Nesse sentido, o AHP demonstra ser um método mais adequado por permitir realizar as comparações por partes. Todavia, também é possível utilizar outros métodos de MCDM, devendo, no entanto, ser levado em consideração as devidas implicações e o custo computacional adicional, ao se utilizar outras técnicas.

4.2 Classe de Criticidade

Conforme descrito na Seção 2, o ESI é um algoritmo de triagem usado em hospitais de todo mundo para classificar a priorização que um paciente deve receber. Essa priorização leva em consideração o seu estado clínico, seus sinais vitais e a quantidade de recursos hospitalares que irá precisar. A solução proposta neste trabalho para definição da Classe de Criticidade se baseia no ESI. Apesar de serem aplicados em contextos diferentes, ambos se embasam nas indicações de estado do paciente/serviço para indicar sua criticidade.

Para a solução proposta, o estado do serviço será indicado pelo tipo do serviço e seu SLO. Cada um dos tipos de serviços considerados na Seção 4.1 tem seu próprio SLA definido pelo provedor. Os SLOs que compõem os SLAs (vide Seção 2) são as propriedades mensuráveis utilizadas para definir o nível de criticidade do serviço. Os SLOs podem se referir a Recursos, Disponibilidade, Tempo de Resposta, Latência, entre os outras métricas [43].

Em cenários reais, provedores como Amazon e Google, por exemplo, apresentam apenas um SLO que se refere à disponibilidade, equivalente a 99,95%. Todavia, alguns trabalhos

[17; 18] têm defendido o uso de múltiplas classes e diferentes níveis de SLA, associado a diferentes preços. Essas pesquisas argumentam que essa multiplicidade tende a ser adotada face à concorrência que tem crescido entre os provedores, com o intuito de atrair mais clientes e aumentar os lucros. Nesta proposta, será considerado a existência de três níveis de SLO, sendo eles: alto, médio e baixo. A métrica a qual o SLO irá se referir, bem como seus valores de referência, dependem das regras de negócio e especificações estabelecidas pelo provedor.

A classe de criticidade varia de 1 (VMs mais críticas) a 4 (VMs menos críticas). Quanto maior a classe de criticidade, mais tolerante é o serviço a degradações no desempenho e maior os níveis de sobrecarga que podem ser atribuídos. Na Figura 4.2 é exposto o fluxograma para definição da classe de criticidade, nele há uma sugestão de configuração considerando o contexto de sobrecarga de CPU. Entretanto, a solução para classe de criticidade é suficientemente flexível para admitir modificações por partes dos provedores e serem empregados em diferentes cenários.

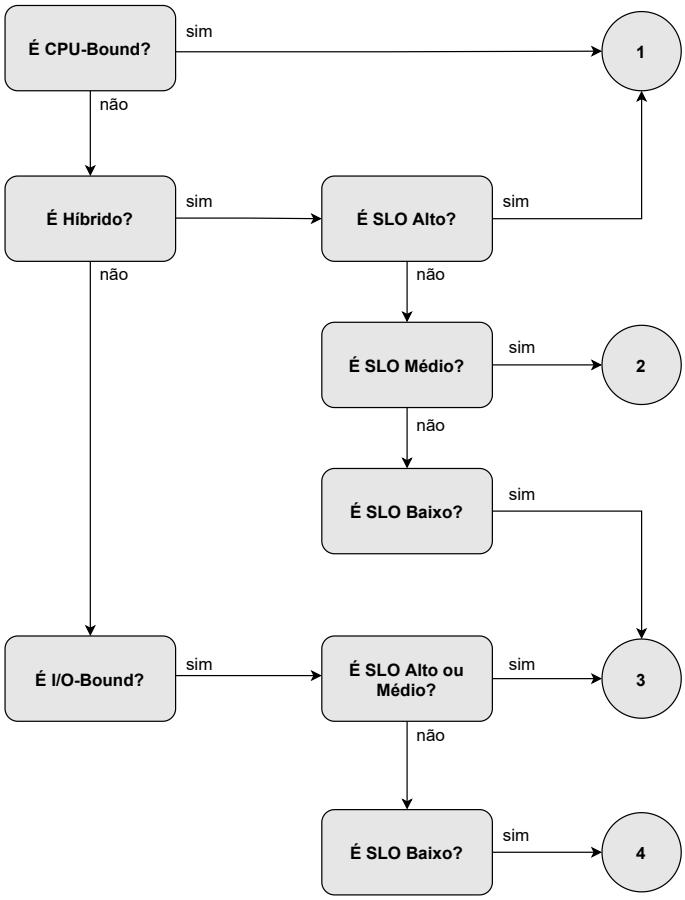


Figura 4.2: Definição da Classe de Criticidade.

Na configuração utilizada, foi considerado que serviços do tipo CPU-Bound, para qualquer nível de SLO, devem ser mais críticos devido à característica inerente de consumir mais CPU. Logo, a atribuição de sobrecarga implica em um maior risco de degradações de desempenho e possíveis violações de SLO. Conjuntamente, foi considerado que serviços do tipo Híbrido e com SLO alto, também pertençam a classe 1, pois o SLO alto requer um maior grau de comprometimento em relação aos requisitos de desempenho. Os demais tipos de serviços com seus respectivos SLOs foram designados para as demais classes de criticidade. A classe de criticidade 4 é atribuída a serviços I/O-Bound com SLO baixo. Isto porque esse serviço requisita um menor consumo de CPU e possui menor comprometimento por parte do provedor quanto a garantias do desempenho.Portanto, há maior flexibilidade para atribuir valores de sobrecarga mais altos.

4.3 Atribuição do Nível de Sobrecarga

Muitos hipervisores utilizados em ambientes de produção, como Xen e KVM, e gerenciadores de plataformas de nuvem, como OpenStack e OpenNebula, utilizam sobrecarga de recursos como um parâmetro de configuração estático, aplicado por máquina física. Dada a particularidade no comportamento de cada carga de trabalho, aplicar um mesmo nível de sobrecarga para todas as VMs de uma máquina física pode ter implicações negativas. Enquanto uma VM que tem baixo consumo de recursos não sentirá o impacto da sobrecarga aplicada, uma outra VM com um padrão de consumo mais alto, hospedada na mesma máquina física, pode ter seu desempenho degradado. Além disso, empregar um nível de sobrecarga estático pode não ser a estratégia mais eficiente em um ambiente dinâmico.

No mecanismo proposto neste trabalho, o nível de sobrecarga é atribuído dinamicamente para cada VM. Cada VM tem seu percentual de utilização de CPU monitorado. A partir dos dados monitorados, um modelo de regressão linear para previsão de utilização de CPU é criado para cada VM. Por meio da previsão do uso de CPU é possível definir qual é o nível de sobrecarga mais adequado a ser utilizado. Por exemplo, caso um cliente tenha solicitado a alocação de 2 vCPUs para uma determinada VM, mas a partir do modelo foi previsto que sua utilização será de 50%, então pode-se adotar um nível de sobrecarga igual 2 e alocar para essa VM apenas 1vCPU. Dessa forma é possível haver um menor desperdício de recursos, sem que o usuário tenha prejuízo em seu desempenho.

O processo pode ser observado na Figura 4.3. Quando uma VM é alocada, nenhum nível de sobrecarga é atribuído inicialmente. Durante um período de 30 minutos o percentual de uso de CPU é coletado. Os dados monitorados são utilizados para gerar um modelo de regressão linear para prever o uso de CPU futuro, com base no passado. Após esse período de tempo, coletam-se novas amostras por um período de 15 minutos, a fim de ser possível comparar os dados observados com os dados previstos pelo modelo e calcular o $R - quadrado$.

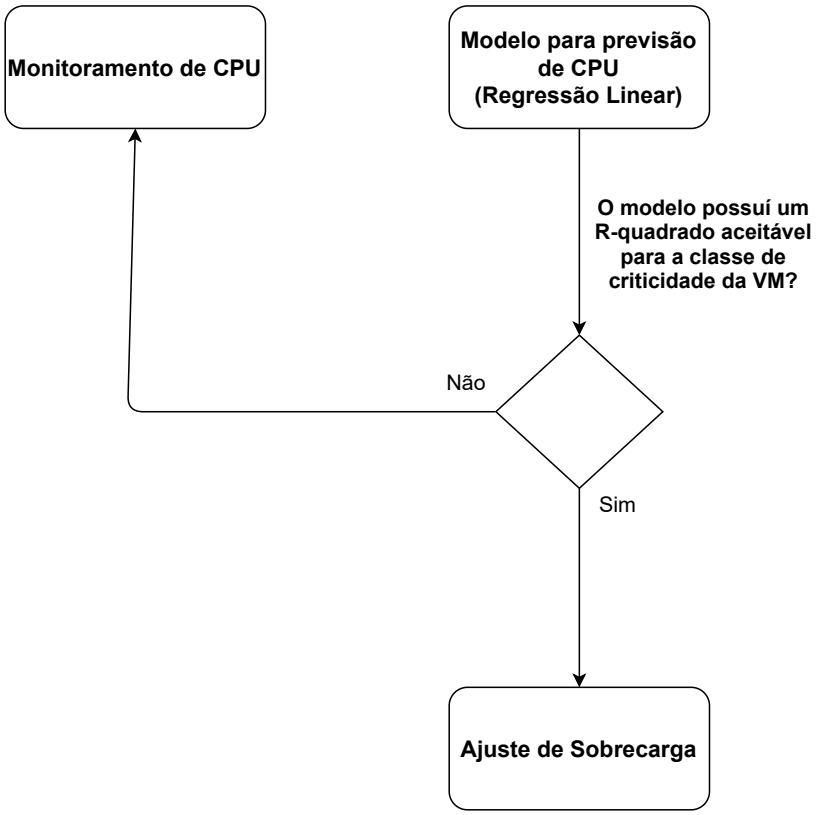


Figura 4.3: Mecanismo de ajuste de sobrecarga baseado na previsão do uso de CPU.

Conforme descrito na Seção 4.2, quanto menor a classe de criticidade, menor a margem para a admissão de maiores níveis de sobrecarga. Isso acontece porque, para as VMs pertencentes a essas classes, a CPU tende a ser um recurso mais crítico e os níveis de SLO mais altos, sendo assim, pouco tolerantes às oscilações no desempenho. Portanto, é mister que haja um maior grau de cautela na atribuição de sobrecarga. Nesse sentido, as VMs classificadas como mais críticas são mais intransigentes às falhas do modelo de previsão. Por este motivo, optou-se por utilizar o $R - quadrado$ como um parâmetro com o qual seja possível avaliar o quanto o modelo é explicativo.

Um $R - quadrado$ baixo pode indicar que os dados utilizados são altamente variáveis, não sendo confiável utilizar o modelo para predição. Nessa perspectiva, para cada classe de criticidade é definido um $R - quadrado$ aceitável, de forma que o modelo só será utilizado para prever o uso de CPU e definir um nível de sobrecarga se possuir um $R - quadrado$ igual ou superior ao que foi definido para a sua classe de criticidade. Isso significa que, caso o modelo não seja "confiável" o suficiente, a VM continuará com o mesmo nível de sobrecarga, até que o modelo apresente um $R - quadrado$ satisfatório.

Por exemplo, uma VM pertencente a classe de criticidade 2, para o qual foi definido um $R-quadrado$ de 99%, só terá seu modelo utilizado caso resulte em um $R-quadrado$ igual ou superior a 99%. Isso significa que o modelo estimado para essa VM só será utilizado caso a porcentagem de variação da variável resposta que é explicada pelo modelo seja de 99%. Do contrário, o nível de sobrecarga permanecerá o mesmo, até que o modelo seja atualizado e resulte no $R-quadrado$ esperado. Conforme a classe de criticidade aumenta, é possível aceitar valores de $R-quadrado$ inferiores, já que VMs classificadas com maiores classes de criticidade são menos sensíveis ao desempenho e possuem SLO menos rigorosos. Portanto, essas VMs tornam-se mais tolerantes a possíveis falhas do modelo de predição.

Na Tabela 4.1 são expostos sugestões de R-quadrados a serem almejados para as classes de criticidade 2, 3 e 4. Por ser a mais crítica, contendo as VMs que hospedam serviços CPU-Bound, a classe 1 não terá qualquer nível de sobrecarga atribuído, a fim de preservar o seu desempenho.

Classe de Criticidade	**R-quadrado aceitável**
2	99%
3	95%
4	85%

Tabela 4.1: $R-quadrado$ requerido para os modelos das VMs pertencentes às classes de criticidade 2, 3 e 4.

A classe de criticidade também define o limiar máximo de sobrecarga que pode ser atribuído à VM. Como mencionado anteriormente, a classe 1 não terá qualquer sobrecarga atribuída, por ser designada a serviços mais sensíveis ao desempenho. As demais classes de criticidade terão seus respectivos limites de sobrecarga definidos pelo administrador da nuvem.

Durante todo o período de execução da VM, os dados de CPU continuam sendo monitorados, o modelo sendo atualizado e todas as demais etapas do processo sendo realizadas, a fim de obter um nível de sobrecarga que promova um equilíbrio entre utilização de recursos e desempenho do serviço.

Capítulo 5

Avaliação da Solução Proposta

Neste capítulo será exposto o processo para avaliação da solução proposta. O objetivo da avaliação é comparar a abordagem desenvolvida com uma abordagem em que a sobrecarga é definida estaticamente e outra em que a sobrecarga não é utilizada. O planejamento foi conduzido a fim de responder as seguintes questões de pesquisa:

- **Questão de Pesquisa 1 (QP1)**: A abordagem proposta, em comparação com as demais abordagens, consegue promover um melhor equilíbrio no *trade-off* entre aumentar a utilização de recursos e evitar degradação de desempenho?
- **Questão de Pesquisa 2 (QP2)**: A inclusão da classificação das VMs por criticidade fornece melhores resultados, se comparado a utilizar apenas o mecanismo de atribuição dinâmica de sobrecarga?

5.1 Ambiente de Simulação

Para a realização do experimento foi utilizado o CloudSim, um *framework* para modelagem e simulação de ambientes de Infraestrutura como Serviço [13; 21]. Essa ferramenta tem sido utilizada pela comunidade científica a fim de investigar suas soluções propostas para Nuvem. Desse modo, não é necessário se ater às particularidades de baixo nível de ambientes de IaaS, nem aos altos custos e inviabilidades relacionadas a se realizar experimentações em cenários reais [5; 14; 61; 56; 4; 34; 12; 39; 15]. A fim de atingir os objetivos do experimento, foi necessário realizar algumas modificações no Cloudsim, que serão descritas nas próximas

Subseções e estão disponíveis em repositório público [1].

5.1.1 Sobrecarga de Recursos

O Cloudsim foi estendido para implementar o mecanismo de atribuição de sobrecarga proposto na Seção 4.3. A partir do monitoramento do uso de CPU, um modelo de regressão linear é criado para cada VM a fim de prever sua utilização. Cada modelo criado é utilizado para prever a utilização de CPU de sua respectiva VM. Se o R-quadrado do modelo for considerado aceitável para a prioridade da VM, o valor previsto é utilizado para ajustar o nível de sobrecarga. O processo de monitoramento, criação do modelo, verificação e ajuste do nível de sobrecarga é realizado periodicamente, durante todo o tempo de execução das VMs.

5.1.2 Dinamicidade na Criação de VMs

Na implementação original do Cloudsim, todas as VMs são criadas no início da simulação. Contudo, em um cenário real a solicitação para criar uma VM pode ocorrer em qualquer instante de tempo. Nesse sentido, para dispor de um ambiente de simulação mais condizente com a realidade, também foi implementado no Cloudsim a dinamicidade na criação de VMs, ou seja, as requisições para alocação das máquinas virtuais ocorrem de forma aleatória durante a execução da simulação.

5.1.3 Orquestração da Alocação

Com o objetivo de utilizar uma orquestração de alocação baseada em técnicas que têm sido utilizadas em cenários reais, foi implementado no Cloudsim um algoritmo de posicionamento baseado no agendador do Kubernetes. Inicialmente lançado pelo Google e atualmente gerenciado pela *Cloud Native Computing Foundation* (CNCF) [10], o Kubernetes é uma plataforma de orquestração *open source* para implantação e gerenciamento de contêineres em escala [37].

No kubernetes a menor unidade gerenciável é o Pod, que trata-se de um conjunto de um ou mais contêineres [36]. Nesse sentido, o agendador do Kubernetes é executado a fim de

[1] `https://github.com/marcelagaldino/cloudsim.git`

definir o posicionamento desses Pods. No Cloudsim, a mesma lógica é utilizada, mas para o contexto de VMs.

Isto posto, a decisão de onde uma VM será alocada é definida a partir de um conjunto de regras, ou seja, busca-se encontrar um *host* que atenda a um conjunto de especificações definidas pela VM. O algoritmo tem dois principais componentes: filtro e *score*. O filtro é usado para selecionar os *hosts* que atendem as regras de filtragem definidas e, portanto, são aptos à receber a VM. Já para o cálculo do *score*, cada *host* resultante da filtragem é classificado, recebendo uma pontuação de acordo com alguns critérios de prioridade. O *host* com maior pontuação é considerado mais adequado e é selecionado para alocar a VM. Caso haja mais de um *host* com a mesma pontuação, a decisão entre eles ocorre aleatoriamente [35].

Na Figura 5.1 é exibido um diagrama que exemplifica o funcionamento da abordagem de posicionamento implementado no Cloudsim, baseada no Kubernetes. Inicialmente há três possíveis *hosts*. Ao passar pelo processo de filtragem, constatou-se que o Host 2 não possui recursos suficientes, por isso ele é removido da lista de possíveis candidatos para alocar a VM. Em seguida, é realizado o processo de *score*, em que, dado um conjunto de critérios, é definido uma pontuação para os hosts que passaram pela filtragem. Por fim, o *host* com maior pontuação é escolhido para alocar a VM.

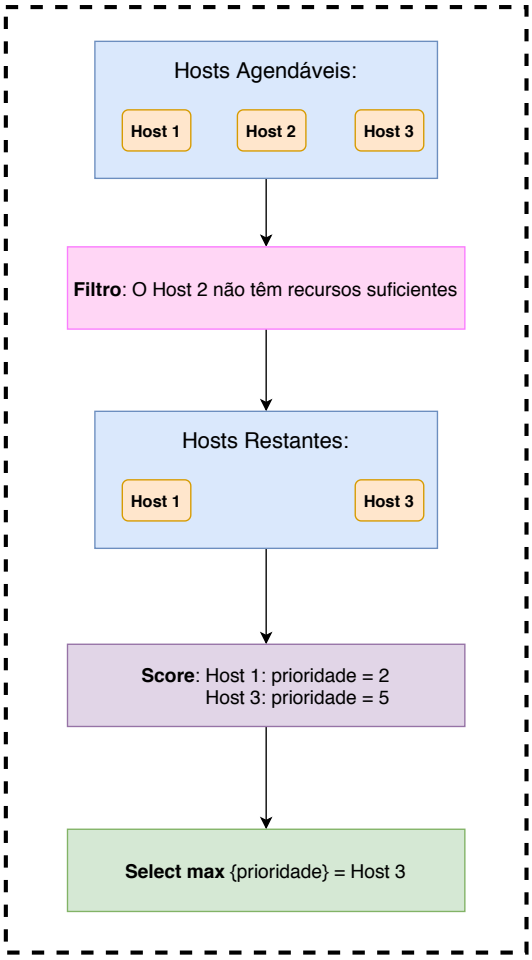


Figura 5.1: Funcionamento da Abordagem de Posicionamento de VMs.

No Kubernetes, os filtros e *scores* são implementados em diferentes *plugins*. Cada *plugin* considera diferentes aspectos, e a escolha de quais utilizar define um perfil de agendamento. Alguns *plugins* são utilizados por padrão, no entanto, a maioria deles consideram fatores mais voltados para o contexto em que o Kubernetes é aplicado, como por exemplo o *score* *ImageLocality*, que dá preferência aos nós que já possuem as imagens de contêiner que o Pod executa. Nesse sentido, analisando o escopo no qual o Cloudsim e a presente pesquisa se enquadram, foram realizadas extensões no código baseadas nos seguintes *plugins* do Kubernetes:

- **NodeResourcesFit**: Implementa o filtro que verifica quais *hosts* dispõem de recursos suficientes para atender o que a VM solicita, ou seja, retornará os *hosts* que têm uma quantidade de recursos livre igual ou superior à que a VM está requisitando.

- **InterVmAffinity**: Implementa o filtro e o *score* com requisitos de afinidade entre VMs, ou seja, a decisão de se um nó é adequado para alocar uma determinada VM é baseada nas características das VMs que já estão alocadas naquele nó. Para isso, as VMs possuem regras de afinidade. As regras de afinidade permitem que apenas um *host* que hospede uma ou mais máquinas virtuais com as mesmas regras da nova VM, seja apto para alocá-la. No entanto, vale ressaltar que essas regras podem ser rígidas ou flexíveis:

  - As **regras rígidas** devem ser cumpridas para que a VM seja alocada, por este motivo elas são analisadas no filtro. Dessa forma, apenas *hosts* que necessariamente atendam a esses requisitos passarão pelo filtro e poderão hospedar a VM.
  - As **regras flexíveis** indicam preferências, ou seja, busca-se atendê-las, mas não há garantia que serão cumpridas. Por este motivo, essas regras são analisadas no *score* de forma que *hosts* que satisfaçam as regras recebam uma pontuação que os tornem preferíveis àqueles que não cumprem os requisitos. Neste trabalho, optou-se pela utilização de regras flexíveis, visto que ela torna possível uma maior consolidação, se comparada às regras rígidas.

5.2 Experimento

Esta Seção descreve o processo percorrido no planejamento e condução do experimento, com o intuito de responder as questões de pesquisa supracitadas.

5.2.1 Tipos de Serviço

Conforme descrito na Seção 4.1, a solução proposta considera o uso do método AHP na definição do tipo de serviço. Nesse sentido, durante a simulação é necessário que seja recebido como entrada a matriz de comparação dos critérios para o cálculo da prioridade dos critérios; e as matrizes de comparação das alternativas para o cálculo da prioridade das alternativas locais.

Em um cenário real, a matriz de comparação das alternativas seria obtida a partir das indicações do administrador da nuvem. No entanto, para a experimentação e também com

o intuito de fornecer uma opção de configuração, foi desenvolvido um questionário. Neste questionário os seguintes cenários foram apresentados.

- **Cenário 1**: Um pesquisador desenvolveu uma aplicação para o processamento de imagens para reconhecimento facial em tempo real. Esse pesquisador precisa montar a infraestrutura mais adequada para execução de sua aplicação. As próximas questões permitirão que você o ajude por meio de comparações, julgando os recursos computacionais que são mais importantes para execução dessa aplicação.

- **Cenário 2**: Um desenvolvedor está utilizando o Android Studio para desenvolver uma aplicação Android. Esse desenvolvedor precisa montar a infraestrutura mais adequada para desenvolver sua aplicação Android. As próximas questões permitirão que você o ajude por meio de comparações, julgando os recursos computacionais que são mais importantes para execução dessa aplicação.

- **Cenário 3**: Uma indústria alimentícia pretende implantar um sistema de controle de estoque em uma de suas unidades. Essa indústria precisa montar a infraestrutura mais adequada para execução de seu sistema de controle de estoque. As próximas questões permitirão que você a ajude por meio de comparações, julgando os recursos computacionais que são mais importantes para execução dessa aplicação.

Na Figura 5.2 são exibidas as perguntas que foram apresentadas em cada cenário. Os respondentes deveriam realizar comparações pareadas utilizando a escala fundamental de Saaty. Para isso, era necessário mover a escala em direção ao recursos que o usuário achava mais importante.

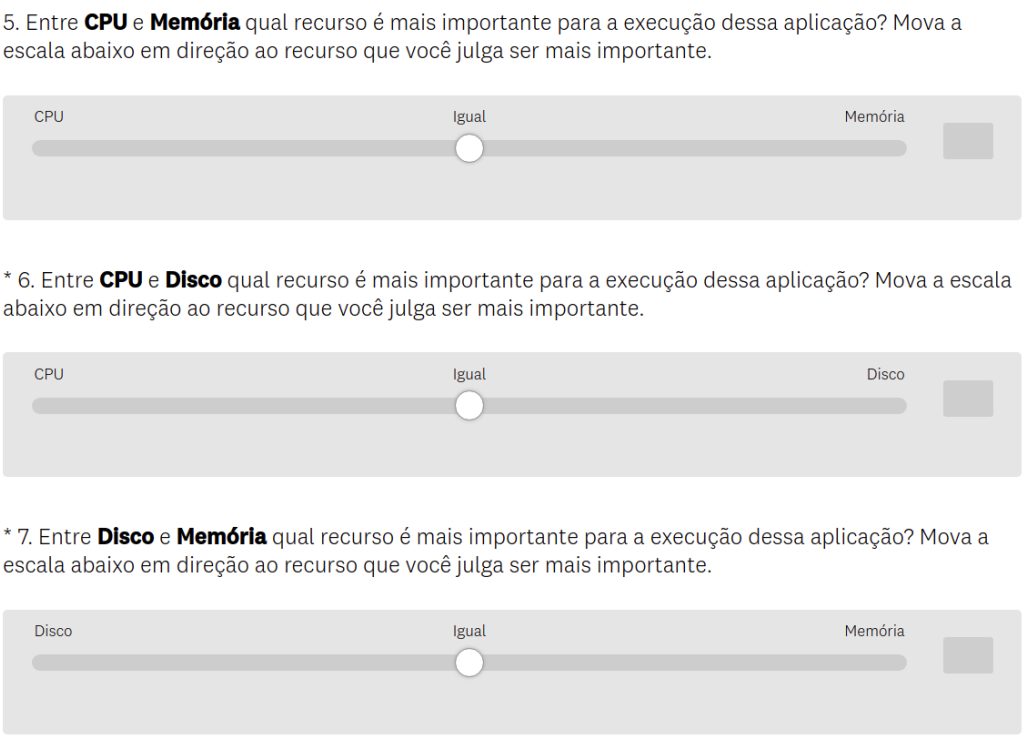


Figura 5.2: Estrutura do questionário.

A utilização dos cenários e dos tipos de recurso, em vez de mencionar explicitamente os tipos de serviço, teve como objetivo tornar o processo do respondente mais intuitivo. No Cenário 1, por exemplo, sabe-se que aplicações de processamento de imagem tendem a consumir mais CPU que os demais recursos [60]. Portanto, quando o respondente realiza as comparações em relação ao cenário supracitado, na realidade ele está realizando a comparação pareada entre os tipos de serviço (alternativas) em relação à CPU (critério). Em outras palavras, se está respondendo a seguinte pergunta: para o critério CPU, que tipo de serviço é mais adequado?

A mesma lógica pode ser utilizada para os Cenários 2 e 3. O Android Studio requer uma maior utilização de memória durante o desenvolvimento da aplicação e mais CPU durante o processo de compilação. Enquanto aplicações de controle de estoque tendem a realizar um maior número de operações de leitura e gravação no disco.

O questionário completo está disponível no Apêndice A. Para avaliar a confiabilidade e validade, foi utilizado o método *focus groups*[28] no qual um grupo de três profissionais da área de redes e computação em nuvem responderam o questionário e apontaram as melhorias

a serem realizadas. Após a validação, o questionário foi enviado para estudantes e profissionais da área em um formato não-supervisionado. Portanto, não houve qualquer interação com os respondentes.

Foram coletadas 55 respostas. Na Figura 5.3, é exibido o perfil dos respondentes no que se refere à escolaridade. Cerca de 85,4% dos participantes possuem superior completo e 52,7% cursou ou está cursando pós-graduação em ciência da computação ou áreas afins. No que se refere a experiência no mercado de trabalho em TI (Tecnologia da Informação), conforme pode ser observado na Figura 5.4, 30,9% dos participantes possuem entre 3 e 5 anos de experiência e 38% possuem experiência superior a 6 anos.

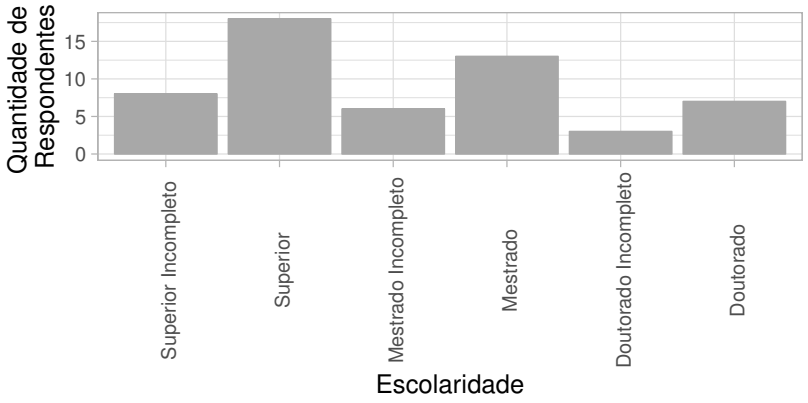


Figura 5.3: Perfil de escolaridade dos respondentes do Survey.

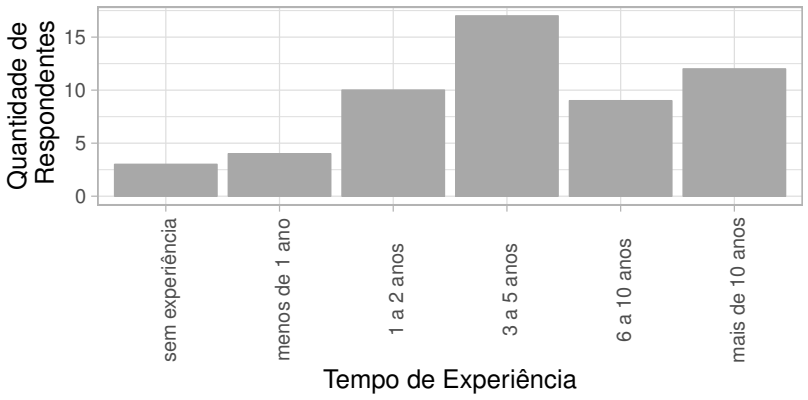


Figura 5.4: Perfil de experiência dos respondentes no mercado de trabalho.

Foi utilizado o cálculo da Razão de Coerência (Vide Seção 2.2.3) para filtrar quais respostas apresentavam julgamentos consistentes. Desse modo, permaneceram um total de 19, 22 e 20 respostas para os Cenários 1, 2 e 3, respectivamente. Essas respostas foram então

classificadas em três categorias: Quem respondeu que a alternativa A é mais importante que a B; Quem respondeu que alternativa B é mais importante que a A e Quem respondeu que a alternativa A e B são igualmente importantes. A definição de qual valor será utilizado para montar a matriz de paridade é realizada a partir da média dos valores da categoria que obteve um maior número de respostas em relação as demais. Por meio desse processo, foram então criadas as Matrizes de Comparação Pareada das alternativas em relação a cada recurso. Nas Tabelas 5.1, 5.2 e 5.3 são exibidos os valores das matrizes e o resultado do cálculo para a prioridade alternativa local. É possível observar que, quando comparado a CPU, o tipo de serviço CPU-Bound apresenta a maior prioridade alternativa local; enquanto que, quando comparado em relação a memória, a maior prioridade é do tipo de serviço Híbrido; e por fim, quando comparado em relação a Disco, a maior prioridade é para o I/O-Bound.

Critério: CPU

	CPU-Bound	**Híbrido**	**I/O-Bound**	**Prioridade Alternativa Local**
CPU-Bound	1	4	5	65,67%
Híbrido	$\frac{1}{4}$	1	5	26,06%
I/O-Bound	$\frac{1}{5}$	$\frac{1}{5}$	1	8,27%

Tabela 5.1: Matriz de Comparação pareada das alternativas em relação ao critério CPU.

Critério: Memória

	CPU-Bound	**Híbrido**	**I/O-Bound**	**Prioridade Alternativa Local**
CPU-Bound	1	1	5	44,43%
Híbrido	1	1	6	47,21%
I/O-Bound	$\frac{1}{5}$	$\frac{1}{6}$	1	8,36%

Tabela 5.2: Matriz de Comparação pareada das alternativas em relação ao critério Memória.

Critério: Disco				
	CPU-Bound	**Híbrido**	**I/O-Bound**	**Prioridade Alternativa Local**
CPU-Bound	1	1	$\frac{1}{4}$	16,03%
Híbrido	1	1	$\frac{1}{5}$	14,88%
I/O-Bound	4	5	1	69,08%

Tabela 5.3: Matriz de Comparação pareada das alternativas em relação ao critério Disco.

No que diz respeito a matriz de comparação dos critérios, a solução proposta sugere que sua obtenção ocorra a partir das comparações realizadas pelos clientes para cada VM submetida no *datacenter*. Diante da impossibilidade de se experimentar a solução em ambiente real, optou-se por simular a resposta do cliente para cada tipo de serviço. Nesse sentido, foram simuladas três respostas (apresentadas através das Tabelas 5.4, 5.5 e 5.6), de forma que ao ser relacionada com as prioridades das alternativas locais, cada resposta resulte em um tipo de serviço.

	CPU	**Memória**	**Disco**
CPU	1	9	9
Memória	$\frac{1}{9}$	1	1
Disco	$\frac{1}{9}$	1	1

Tabela 5.4: Resposta 1 - CPU é mais importante que os demais critérios

	CPU	**Memória**	**Disco**
CPU	1	$\frac{1}{2}$	5
Memória	2	1	5
Disco	$\frac{1}{5}$	$\frac{1}{5}$	1

Tabela 5.5: Resposta 2 - Memória é mais importante que os demais critérios, porém há um maior equilíbrio entre as importâncias dos critérios

	CPU	**Memória**	**Disco**
CPU	1	1	$\frac{1}{9}$
Memória	1	1	$\frac{1}{9}$
Disco	9	9	1

Tabela 5.6: Resposta 3 - Disco é mais importante que os demais critérios

5.2.2 Carga de Trabalho

No CloudSim, as tarefas atribuídas às VMs são chamadas de *Cloudlets*. As *Cloudlets* representam os aplicativos e sua complexidade em termos de seus requisitos computacionais. Cada *Cloudlet* é atribuída a uma única VM que só existe durante o tempo de execução da mesma. Nos experimentos realizados, o comportamento das *Cloudlets* foi definido fazendo uso das seguintes bases de rastreamento.

- **Google V3**[2]: Dados de rastreamento de 8 células do *cluster* do Google coletados no período do mês de maio de 2019. Foram utilizados os dados do campo *resouce_request* da tabela *instanceEvents* que possui a quantidade de CPU alocada para cada instância, bem como o campo *average_usage* da tabela *instanceUsage* que dispõe da média de utilização de CPU, aferida durante um período de medição (tipicamente, 5 minutos). Esses campos foram usados para calcular o percentual de utilização de CPU de cada instância em relação ao que foi alocado para ela. Apesar dos rastros disponibilizados pelo Google serem de contêineres, atualmente é a maior base disponível com dados de um ambiente real.

- **Bitbrains**: Dados de rastreamento de 1750 VMs disponibilizados pelo Bitbrains, um provedor de serviços especializado em hospedagem gerenciada e computação comercial para empresas [49]. Foi utilizado o campo CPU Usage, que dispõe do percentual de utilização de CPU de cada VM.

Com o intuito de tornar coerente o tipo de serviço da *Cloudlet* que será utilizado para representar seu comportamento, foi realizada uma classificação a partir dos dados coletados

[2] `https://github.com/google/cluster-data/blob/master/ClusterData2019.md`

nas bases supramencionadas. Nesse sentido, se o componente AHP sugerir, por exemplo, que o tipo do serviço é CPU-Bound, será atribuída uma *Cloudlet* que tenha um comportamento correspondente ao tipo de serviço selecionado.

Ao todo, foi criada uma base com 795 *Cloudlets*, das quais 55 foram classificadas como CPU-Bound, 189 como Híbrido e 551 como I/O-Bound. Essa classificação foi feita a partir dos Quartis (Q1, Q2 e Q3) da distribuição de percentual de CPU, considerando as seguintes condições:

- **CPU-Bound**: $Q1 > 50\%$
- **Híbrido**: $Q1 \geq 15\%$ e $Q3 \leq 50\%$
- **I/O-Bound**: $Q1 > 1\%$ e $Q3 < 15\%$

5.2.3 Cenários

Durante a execução do experimento, três cenários foram considerados:

- **Sem Sobrecarga**: As *Cloudlets* são executadas em ambiente sem sobrecarga, ou seja, para todas as VMs do *Datacenter* o nível de sobrecarga é igual a 1. Esse cenário é executado para ser possível ter um valor de referência paras as métricas, quando não se é utilizado sobrecarga.
- **Estático**: As *Cloudlets* são executadas em um ambiente com sobrecarga estática. Cada VM recebe um nível de sobrecarga de acordo com sua classe de criticidade. O valor definido para sobrecarga é estático, permanecendo o mesmo durante toda a execução.
- **Dinâmico**: Este cenário compreende a solução proposta. Nele, as *Cloudlets* são executadas em um ambiente com sobrecarga dinâmica. Cada classe de criticidade tem um nível de sobrecarga associado que, neste caso, indica o valor máximo de sobrecarga que pode ser atribuído às VMs classificadas com essa criticidade. A partir de um modelo de previsão de utilização de CPU é definido o nível de sobrecarga mais adequado para a VM, que é atualizado a cada 100s durante a execução e que deve ser sempre menor ou igual ao valor limite definido para sua classe de criticidade. Caso não haja recurso suficiente para reduzir o nível de sobrecarga, o valor permanece o mesmo.

- **Dinâmico Sem Criticidade (DSC)**: As *Cloudlets* são executadas em um ambiente com sobrecarga dinâmica, assim como no cenário anterior, com a diferença de que não é utilizado o mecanismo de criticidade. Nesse caso, será utilizado exatamente o valor previsto para sobrecarga, não havendo qualquer ajuste imposto pelos limiares definidos por cada classe de criticidade. Portanto, esse cenário é considerado a fim de investigar a influência da classificação por criticidade na solução proposta.

5.2.4 Métricas

Durante a execução do experimento, as seguintes métricas foram aferidas:

- **Equivalência no Tempo de Execução**: Diz respeito ao percentual de *Cloudlets* que não tiveram seu Tempo de Execução alterado com a utilização de sobrecarga. O valor de referência para o tempo de execução é o que foi obtido no cenário Sem Sobrecarga. O tempo de execução é um fator que interfere diretamente na perspectiva do usuário quanto ao desempenho de sua aplicação. Por meio dessa métrica, é possível ter evidências do impacto da utilização de sobrecarga sob o ponto de vista do usuário.

- **Violação (%)**: Refere-se ao percentual de VMs que sofreram violações durante sua execução. Considera-se violação quando o usuário recebe uma quantidade de recursos inferior a demanda de processamento instantânea da VM, levando em conta também a tolerância admitida pelo SLO definido em contrato. O Cloudsim trabalha com a atribuição de MIPS (do inglês, *Million Instructions Per Second*). Como exemplo, considere um usuário que solicita no momento da alocação da VM 1 vCPU com 500 MIPS e o SLO definido em contrato é de 99%. Neste caso, o provedor deve garantir que 99% dos recursos solicitados pelo cliente sejam atendidos. Isso significa que, se em algum momento da execução da aplicação, o usuário solicitar 50% dos seus recursos, ele espera receber 250 MIPS das 500 alocadas. Como seu SLO é de 99%, isso significa que, se for oferecido a ele um valor inferior a 247,5 MIPS, ocorre então uma violação.

- **Consumo de Energia (kWh)**: O Consumo de Energia total do *Datacenter* no término da execução de cada cenário. No Cloudsim o consumo de energia é definido a partir

de dados reais sobre consumo de energia obtidos dos resultados do *benchmark* SPECpower [5]. Os pesquisadores selecionaram duas configurações de servidor com CPUs dual-core, sendo elas: HP ProLiant ML110 G4 (Intel Xeon 3040, 2 núcleos 1860 MHz, 4 GB) e HP ProLiant ML110 G5 (Intel Xeon 3075, 2 núcleos 2660 MHz, 4 GB). Para cada percentual de utilização de CPU, é definido o respectivo valor de consumo de energia, conforme exibido na Tabela 5.7. Para os experimentos realizados neste trabalho, optou-se por escolher um dos modelos, para facilitar as comparações entre as abordagens consideradas. Logo, foi escolhido aleatoriamente o Servidor HP ProLiant G4.

Servidor	**0%**	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
HP ProLiant G4	86	89,4	92,6	96	99,5	102	106	108	112	114	117
HP ProLiant G5	93,7	97	101	105	110	116	121	125	129	133	135

Tabela 5.7: Consumo de Energia para os servidores selecionados com diferentes níveis de carga em Watts.

- **Percentual de Máquinas Físicas Alocadas**: Do total de máquinas físicas disponíveis, quantas foram alocadas por VMs durante a execução de cada cenário. Durante a execução do experimento foi considerado um total de 250 máquinas físicas.

5.2.5 Fatores

Foram considerados os seguintes fatores e seus respectivos níveis:

- **SLO**: Durante a execução do experimento são considerados três níveis de SLO: Alto, Médio e Baixo. Cada nível de SLO recebe um percentual de CPU que deve ser garantido de acordo com o seu tipo de serviço. Na Tabela 5.8, são exibidos os valores considerados. O nível de SLO é atribuído a cada *Cloudlet* aleatoriamente. Por exemplo, se o tipo de serviço da *Cloudlet* for I/O-Bound, será definido aleatoriamente se o seu SLO será alto, médio ou baixo. Essa aleatoriedade é definida para representar a decisão do provedor em relação ao SLO atribuído a cada VM.

	SLO		
Tipo de Serviço	**Alto**	**Médio**	**Baixo**
CPU-Bound	99%	95%	90%
Híbrido	90%	85%	80%
I/O-Bound	80%	75%	70%

Tabela 5.8: Valores de SLO por nível e tipo de serviço.

- **Níveis de Sobrecarga:** Conforme descrito na Seção 4.2, o tipo de serviço e seu respectivo SLO são utilizados como entrada no algoritmo que define a classe de criticidade de cada VM. Cada classe de criticidade deve ter um valor de sobrecarga associado, que define o nível de sobrecarga a ser utilizado, no caso do cenário estático, ou define o limiar máximo aceitável para sobrecarga, no caso do cenário dinâmico. Durante os experimentos foram considerados quatro conjuntos de níveis de sobrecarga, que podem ser visualizados na Tabela 5.9. O Conjunto 1 tem o nível de sobrecarga definido como 1 independente da classe de criticidade considerada, esse conjunto foi definido para ser utilizado no cenário Sem Sobrecarga. Os Conjuntos 2, 3 e 4 são utilizados pelos cenários estático e dinâmico. Tendo em vista que esses valores serão definidos pelo provedor, com essa variação almeja-se investigar a influência do uso de diferentes níveis de sobrecarga nas métricas aferidas e abordagens consideradas.

	Níveis de Sobrecarga			
Classe de Criticidade	Conjunto 1	Conjunto 2	Conjunto 3	Conjunto 4
1	1	1	1	1
2	1	1,5	2	4
3	1	2	4	8
4	1	4	8	16

Tabela 5.9: Níveis de Sobrecarga atribuídos por Classe de Criticidade.

5.2.6 Tratamentos

Como não houve necessidade em reduzir o custo para execução do experimento, foi escolhido o *Design* Fatorial Completo [7]. Por meio deste *design* é possível comparar o desempenho dos cenários, considerando as diferentes combinações entre os níveis dos fatores. Rodadas preliminares do experimento foram realizadas e, fazendo uso dos resultados, foi calculado o tamanho da amostra para obter 95% de confiança. Como resultado, definiu-se como 80 o número de repetições de cada um dos oito tratamentos expressos na Tabela 5.10[3]. Portanto, essa configuração resulta em um total de 640 execuções. Vale ressaltar que, para cada repetição são definidos aleatoriamente o nível do SLO e o momento em que a VM é alocada no *Datacenter*, o primeiro por se tratar de uma decisão do provedor e o segundo por ser um evento sobre o qual não se pode ter controle, evitando assim introduzir viés nos resultados.

Tratamento	**Níveis de Sobrecarga**	**Cenário**	**Repetições**
1	Conjunto 1	Sem Sobrecarga	80
2	Conjunto 2	Estático	80
3	Conjunto 3	Estático	80
4	Conjunto 4	Estático	80
5	Conjunto 2	Dinâmico	80
6	Conjunto 3	Dinâmico	80
7	Conjunto 4	Dinâmico	80
8	Sem Conjunto de Sobrecarga	DSC	80

Tabela 5.10: Tratamentos

5.2.7 Ameaças à Validade

Das ameaças à validade constatadas, destaca-se uma ameaça à validade interna ao utilizar o Cloudsim para simular ambientes IaaS. Sabe-se que qualquer simulação tem um erro inerente associado, então, os dados podem sofrer alterações dada a utilização deste cenário.

[3]No tratamento 8 nenhum conjunto de sobrecarga foi utilizado, pois para o cenário DSC os níveis de sobrecarga são definidos apenas através do modelo de predição

Entretanto, conforme mencionado na Seção 5.1, o Cloudsim é uma ferramenta largamente usada em diversas pesquisas, evidenciando possuir um bom nível de qualidade. Conjuntamente, também é possível observar uma ameaça à validade externa, uma vez que, fazendo uso de ambientes simulados não é confiável generalizar os resultados para cenários reais. Todavia, para mitigar essa ameaça, foram utilizados dados de bases públicas, coletados de cenários reais, para definir o perfil de processamento das VMs.

Ademais, tendo em vista que o principal objetivo da abordagem proposta é promover um melhor equilíbrio no *trade-off* entre utilização e desempenho. Buscou-se considerar métricas que possam trazer indícios a respeito da utilização de recursos, como o número de máquinas físicas alocadas e o consumo de energia; assim como métricas que tornam possível obter evidências a respeito do desempenho, como a equivalência no tempo de execução e o percentual de violações. Dessa forma, é possível mitigar ameaças à validade de constructo, principalmente no que se refere a evitar viés mono-método. Por fim, para atenuar ameaças à validade de conclusão, buscou-se realizar experimentos prévios, com os quais fosse possível calcular o tamanho da amostra necessário, assim como foi utilizado o *bootstrap* [22] como método de reamostragem e estimativas com intervalos de confiança de 95%.

Capítulo 6

Resultados

É certo que há uma concordância entre pesquisadores e profissionais sobre os benefícios da utilização de sobrecarga, bem como dos riscos inerentes. A literatura especializada apresenta um conjunto de evidências que enfatizam essas vantagens e desvantagens [54; 31; 11]. Este capítulo tem como objetivo apresentar os resultados obtidos por meio dos experimentos descritos no Capítulo 5.

6.1 Avaliação dos Cenários Dinâmico e Estático

Nas próximas Seções serão apresentados os resultados das métricas aferidas durante a execução dos cenários Dinâmico e Estático, a fim de expor os efeitos gerados pela solução proposta em comparação com o contexto em que a sobrecarga é definida estaticamente.

6.1.1 Equivalência no Tempo de Execução

O Tempo de Execução permite indicar os efeitos dos cenários sob a perspectiva dos usuários. Como pode ser observado na Figura 6.1, para o Cenário Estático, 16,10% das *Cloudlets* apresentaram um tempo de execução equivalente ao Cenário Sem Sobrecarga. Esse percentual se manteve igual independente do conjunto de sobrecarga utilizado. Isso se dá, porque as *Cloudlets* que obtiveram um tempo de execução equivalente foram apenas as classificadas com classe de criticidade 1. Para essa classe, o nível de sobrecarga empregado é igual a 1, ou seja, as VMs pertencentes a essa categoria tem 100% dos recursos solicitados disponíveis

para uso, assim como no cenário Sem Sobrecarga.

No que diz respeito ao Cenário Dinâmico, o percentual de equivalência no tempo de execução apresentou um comportamento decrescente em função dos conjuntos de sobrecarga. Os conjuntos que utilizaram maiores níveis de sobrecarga implicaram em percentuais menores. Esse comportamento é esperado, visto que maiores níveis de sobrecarga resultam em uma menor quantidade de recursos sendo fornecidos e, por consequência, pode gerar um atraso na execução das tarefas.

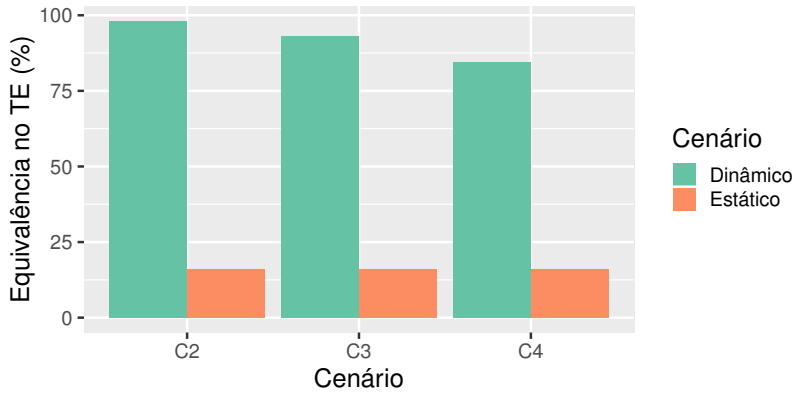


Figura 6.1: Média do Percentual de *Cloudlets* do Cenário Dinâmico e Estático que apresentaram um tempo de execução equivalente ao Cenário Sem Sobrecarga.

Entretanto, vale ressaltar que, mesmo para o conjunto $C4$, estima-se com 95% de confiança que o percentual de equivalência médio esteja entre 84,30% e 84,70%. Portanto, os dados sugerem que mesmo para o conjunto que utilizou os maiores níveis de sobrecarga ($C4 = 1; 4; 8; 16$), cerca de 84% dos *Cloudlets* não sofreram qualquer alteração em seu desempenho. Esses valores chegam a ser ainda mais consideráveis para os demais conjuntos. Estima-se que está entre 98,09% e 98,26% para o conjunto $C2 = \{1; 1{,}5; 2; 4\}$ e entre 92,86% e 93,14% para o conjunto $C3 = \{1; 2; 4; 8\}$.

Nesse sentido, os resultados evidenciam que o cenário Dinâmico demonstra um menor impacto no desempenho no que se refere ao tempo de execução, se comparado ao cenário estático. Isso acontece porque o cenário Dinâmico emprega um nível de sobrecarga que varia com o perfil de utilização de cada VM, de acordo com o que é previsto pelo modelo de utilização de CPU. Diferente do cenário Estático, no qual um mesmo nível de sobrecarga é definido durante toda execução para todas as VMs classificadas em uma mesma classe de

criticidade.

6.1.2 Violação

É de interesse dos provedores de nuvem minimizar a ocorrência de violações, visto que podem resultar no pagamento de multas aos clientes devido ao não cumprimento do que foi definido no SLA. Nesse sentido, foi avaliada a diferença média de violação em pontos percentuais entre os Cenários Estático e Dinâmico. Na Tabela 6.1, podem ser vistas as diferenças, estimadas com 95% de confiança, para cada classe de criticidade[1].

Estima-se que, para todas as classes de criticidade, há uma diferença positiva. Em todos os casos, os resultados sugerem que o cenário Estático promove, em média, um maior percentual de violações se comparado ao cenário Dinâmico. No entanto, essa diferença é mais considerável para a classe de criticidade 2 e diminui conforme a classe de criticidade aumenta. Isso acontece porque as cargas de trabalho para classes de criticidade maiores tendem a ter um padrão de consumo e nível de SLO mais baixos, sendo então mais tolerantes à sobrecarga. Por este motivo, mesmo para o cenário estático, tendem a apresentar um menor número de violações.

Adicionalmente, os resultados evidenciam que, para todas as classes de criticidade, a diferença média de violação em pontos percentuais cresce à medida que o nível de sobrecarga aumenta. Por exemplo, estimou-se com 95% de confiança que, para a classe 2, a diferença média para um nível de sobrecarga igual a 1,5 está entre 21,99% e 23,83%; enquanto para um nível de sobrecarga igual a 4, a diferença média está entre 79,47% e 81,07%. Isso se dá porque para o cenário Estático a sobrecarga permanece constante, enquanto para o cenário dinâmico o nível de sobrecarga define um limiar máximo aceitável.

O mecanismo de atribuição dinâmica de sobrecarga, por sua vez, é quem tende a empregar uma sobrecarga que melhor se adéque ao perfil de utilização de cada VM, resultando em um menor número de violações se comparado ao cenário Estático. Adicionalmente, na Tabela 6.2, são exibidos os valores estimados para a média do percentual de violação em cada cenário, considerando cada classe de criticidade e seus respectivos níveis de sobrecarga. Dessa forma, também é possível comparar numericamente as médias do percentual

[1] A classe de criticidade 1 não é exibida pois, nenhum nível de sobrecarga foi atribuído as VMs pertencentes a este grupo e, portanto, não houveram violações a serem contabilizadas.

de violação obtidas para ambos os cenários.

Classe de Criticidade 2		**Classe de Criticidade 3**		**Classe de Criticidade 4**	
Nível de Sobrecarga	**Diferença Média**	**Nível de Sobrecarga**	**Diferença Média**	**Nível de Sobrecarga**	**Diferença Média**
1,5	22,89 (IC=21,99; 23,83)	2	3,71 (IC=3,58; 3,84)	4	1,2 (IC=1,07; 1,33)
2	30,11 (IC=29,14; 31,13)	4	12,42 (IC=12,21; 12,64)	8	2,83 (IC=2,63; 3,01)
4	80,27 (IC=79,47; 81,07)	8	15,39 (IC=15,15;15,64)	16	7,28 (IC=6,96; 7,58)

Tabela 6.1: Diferença na violação em pontos percentuais entre os cenários Estático e Dinâmico, para cada classe de criticidade (estimados com 95% de confiança).

	Classe de Criticidade 2		**Classe de Criticidade 3**		**Classe de Criticidade 4**	
Cenário	**Nível de Sobrecarga**	**Média**	**Nível de Sobrecarga**	**Média**	**Nível de Sobrecarga**	**Média**
Estático	1,5	26,86 (IC=25,76; 28,01)	2	4,72 (IC=4,55; 4,88)	4	3,48 (IC=3,24; 3,73)
Dinâmico	1,5	3,97 (IC=3,48; 4,47)	2	1 (IC=0,91; 1,11)	4	2,28 (IC=2,11; 2,45)
Estático	2	35,37 (IC=34,17; 36,51)	4	17,11 (IC=16,85; 17,36)	8	3,48 (IC=3,23; 3,72)
Dinâmico	2	5,25 (IC=4,7; 5,82)	4	4,69 (IC=4,52; 4,86)	8	2,28 (IC=2,12; 2,45)
Estático	4	98,22 (IC=97,95; 98,49)	8	26,88 (IC=26,61; 27,16)	16	3,48 (IC=3,23; 3,73)
Dinâmico	4	17,94 (IC=16,99; 18,9)	8	11,49 (IC=11,26; 11,72)	16	2,28 (IC=2,12; 2,46)

Tabela 6.2: Média do percentual de violação para os cenários Estático e Dinâmico, para cada classe de criticidade (Estimado com 95% de confiança).

Diante dos resultados expostos, o cenário dinâmico apresenta, em média, um percentual de violação inferior ao percentual de violação do cenário Estático. Essa diferença é consideravelmente superior para a classe de criticidade 2. Esse aspecto é positivo pois demonstra que VMs classificadas como mais criticas, que são mais sensíveis a desempenho e possuem níveis de SLO mais altos, apresentam percentuais de violação menores se comparado ao cenário em que um valor de sobrecarga é definido estaticamente. Isso indica que há uma maior preservação de desempenho para os clientes e os provedores tendem a ter menor prejuízo quanto ao pagamento de multas devido às violações.

6.1.3 Percentual de Máquinas Físicas Alocadas

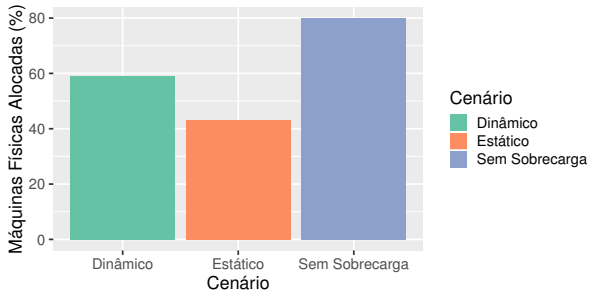
Reduzir o número de máquinas físicas alocadas é benéfico para os provedores, pois implica em um menor número de máquinas em uso e, por consequência, em redução no consumo de energia. Conforme exposto no Capítulo 5, os cenários Estático e Dinâmico foram submetidos aos conjuntos de sobrecarga $C2$, $C3$ e $C4$, em que, para cada conjunto, um nível de

sobrecarga foi atribuído por classe de criticidade. Ademais, para o cenário Sem Sobrecarga, foi atribuído o conjunto $C1$, em que um nível de sobrecarga igual a 1 foi empregado para todas as classes. Na Figura 6.2 são exibidos os percentuais de máquinas físicas alocadas nos cenários, para cada um dos conjuntos de sobrecarga considerados.

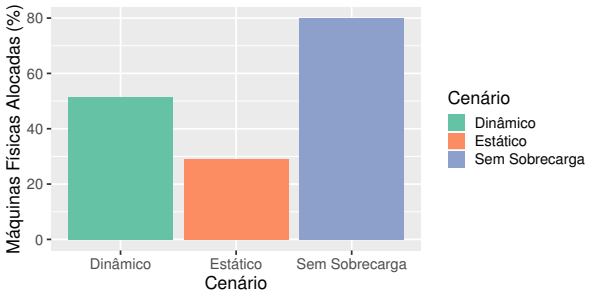
Em todos os casos, o cenário Sem Sobrecarga resulta em um maior número de máquinas físicas sendo utilizadas. Isso é previsto devido a utilização do nível de sobrecarga igual a 1. Portanto, neste cenário as *Cloudlets* não apresentam alterações em seu tempo de execução, mas como consequência requerem um maior número de máquinas físicas sendo utilizadas.

Por outro lado, o cenário Estático resultou em um menor percentual de máquinas físicas alocadas. Logo, esse cenário demonstra ter como benefício permitir que um menor número de máquinas físicas seja utilizado. No entanto, essa vantagem tende a ser alcançada com o custo de redução no desempenho das *Cloudlets* em execução e o aumento de violações, conforme evidenciam os resultados expostos nas seções 6.1.1 e 6.1.2.

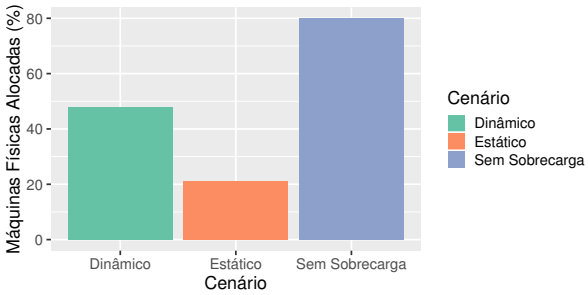
No que se refere ao cenário Dinâmico, o percentual de máquinas físicas alocadas apresenta valores intermediário entre o cenário Sem Sobrecarga e o cenário Estático. Na Tabela 6.3, são apresentados os valores para média do percentual de máquinas físicas alocadas, estimados com 95% de confiança. Percebe-se, que o cenário Dinâmico demonstra uma redução no número de máquinas físicas sendo alocadas, e, embora os valores não sejam equivalentes aos obtidos com o cenário estático, existe a benesse de haver um menor impacto no desempenho das *Cloudlets* e um menor percentual de violações, como exposto nas Seções 6.1.1 e 6.1.2.



(a) Conjunto de Sobrecarga ($C2 = \{1; 1{,}5; 2; 4\}$)



(b) Conjunto de Sobrecarga ($C3 = \{1; 2; 4; 8\}$)



(c) Conjunto de Sobrecarga ($C4 = \{1; 4; 8; 16\}$)

Figura 6.2: Percentual de máquinas físicas alocadas para cada conjunto de sobrecarga dos Cenários Dinâmico e Estático, em comparação com o Cenário Sem Sobrecarga com o conjunto $C1 = \{1; 1; 1; 1\}$.

Conjunto	**Cenário**	**Média (%)**
C1	Sem Sobrecarga	79,96 (IC=79,91; 80,00)
C2	Estático	43,02 (IC=42,96; 43,08)
C2	Dinâmico	59,16 (IC=58,99; 59,34)
C3	Estático	29,17 (IC=29,11; 29,22)
C3	Dinâmico	51,47 (IC=51,24; 51,69)
C4	Estático	21,25 (IC=21,20; 21,30)
C4	Dinâmico	47,86 (IC=47,63; 48,10)

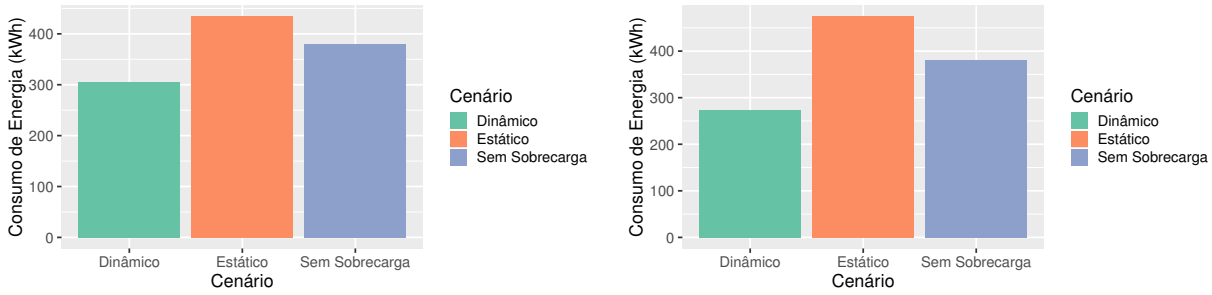
Tabela 6.3: Percentual de máquinas físicas alocadas para o cenário Sem Sobrecarga com o conjunto $C1$ e os cenários Estático e Dinâmico com os conjuntos $C2$, $C3$ e $C4$.

6.1.4 Consumo de Energia

Diminuir o consumo de energia, além de contribuir com a redução de impactos ao meio ambiente, implica em redução de custos para o provedor. Nessa perspectiva, durante os experimentos, o consumo de energia do *Datacenter* foi medido, a fim de permitir uma comparação entre os cenários Sem sobrecarga, Estático e Dinâmico. Na Figura 6.3 são exibidos os valores obtidos do consumo de energia para cada cenário e seus respectivos conjuntos.

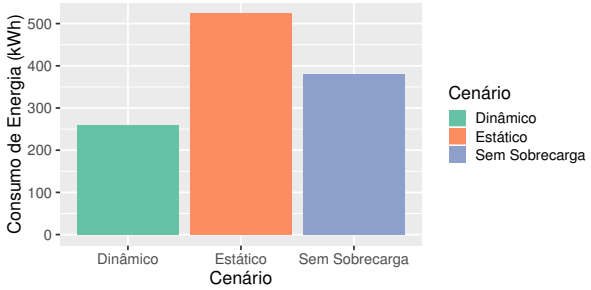
É possível observar que, independente do conjunto de sobrecarga utilizado, o cenário Estático apresentou um maior consumo se comparado aos demais cenários. Isso acontece porque, mesmo tendo uma quantidade menor de máquinas físicas ligadas, como as *Cloudlets* no cenário Estático levam mais tempo para concluir sua execução, as VMs e as máquinas físicas que as hospedam, ficam mais tempo ligadas, implicando em uma maior consumo ao final da execução.

Em contrapartida, o cenário dinâmico apresentou o menor consumo de energia. Isso sugere que esse cenário, além de demonstrar um menor impacto no desempenho dos *Cloudlets*, no que refere ao tempo de execução, resultou também em um menor consumo de energia. Na Tabela 6.4 são exibidos os valores para as médias, estimados com 95% de confiança.



(a) Conjunto de Sobrecarga ($C2 = \{1; 1{,}5; 2; 4\}$)

(b) Conjunto de Sobrecarga ($C3 = \{1; 2; 4; 8\}$)



(c) Conjunto de Sobrecarga ($C4 = \{1; 4; 8; 16\}$)

Figura 6.3: Consumo total de energia do *Datacenter* para cada conjunto de sobrecarga dos Cenários Dinâmico e Estático, em comparação com o Cenário Sem Sobrecarga com o conjunto $C1 = \{1; 1; 1; 1\}$.

Conjunto	**Cenário**	**Média (kWh)**
C1	Sem Sobrecarga	380,18 (IC=379,61; 380,76)
C2	Estático	434,7 (IC=433,57; 435,81)
C2	Dinâmico	305,95 (IC=304,87; 307,02)
C3	Estático	474,95 (IC=473,34; 476,56)
C3	Dinâmico	273,93 (IC=272,27; 275,61)
C4	Estático	524,21 (IC=520,86; 527,61)
C4	Dinâmico	258,92 (IC=257,28; 260,69)

Tabela 6.4: Consumo total de energia do *Datacenter* para o cenário Sem Sobrecarga com o conjunto $C1$ e para os cenários Estático e Dinâmico com os conjuntos $C2$, $C3$ e $C4$.

Existem dois aspectos que influenciam o resultado do consumo de energia, sendo eles: a quantidade de máquinas físicas e o tempo de execução. Os resultados sugerem que o cenário

estático demonstra uma maior consolidação, sendo possível haver uma maior quantidade de recursos disponíveis para serem alocados para outras VMs. Contudo, por resultar em um maior tempo de execução, as máquinas físicas ficam ligadas por mais tempo, elevando o consumo de energia. Por outro lado, o cenário Sem Sobrecarga permite que as *tasks* sejam executadas com o tempo de execução esperado, mas com o custo de alocar uma maior quantidade de máquinas físicas, resultando em um maior consumo de energia. Nesse sentido, o cenário Dinâmico apresentou um melhor equilíbrio nessa relação entre quantidade de máquinas físicas alocadas e tempo de execução, obtendo o menor consumo de energia em relação aos demais cenários.

6.2 Avaliação dos Cenários Dinâmico e Dinâmico Sem Criticidade

Assim como exposto no Capítulo 3, algumas das soluções propostas pela comunidade científica já utilizam mecanismos de predição para definição da sobrecarga. Entretanto, essas soluções não consideram a utilização da classificação das VMs de acordo com sua criticidade. Nesse sentido, esta seção tem como objetivo apresentar um comparativo do cenário Dinâmico, que utilizou a solução proposta neste trabalho com todos os seus componentes, com cenário Dinâmico Sem Criticidade (DSC), que emprega apenas o mecanismo de atribuição de sobrecarga, sem qualquer classificação de criticidade. O objetivo dessa comparação é investigar a influência da classificação por criticidade nos resultados obtidos pelo mecanismo proposto.

Vale ressaltar que o cenário DSC será comparado com o cenário Dinâmico quando este utilizou o conjunto C4. Optou-se por esta configuração tendo em vista que este conjunto é composto pelos maiores níveis de sobrecarga considerados e os resultados para os demais conjuntos foram apresentados nas seções anteriores.

6.2.1 Equivalência do Tempo de Execução

No tocante à equivalência no tempo de execução, o cenário Dinâmico apresentou um percentual de equivalência superior ao cenário DSC, como pode ser observado na Figura 6.4 e

Tabela 6.5. Isso sugere que a classificação das VMs de acordo com sua criticidade resulta em um menor impacto no tempo de execução, se comparado ao cenário em que apenas o mecanismo de atribuição dinâmica de sobrecarga é utilizado.

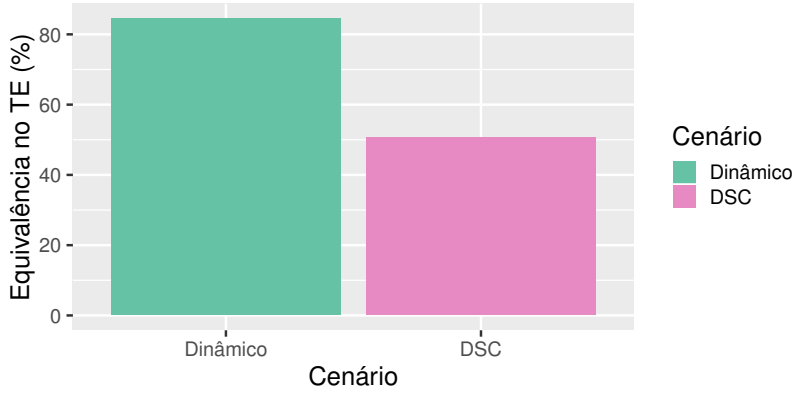


Figura 6.4: Equivalência no tempo de execução para o cenário Dinâmico com o Conjunto C4 e o cenário DSC.

Esse comportamento ocorre pois o cenário DSC atribui o nível de sobrecarga baseado apenas no modelo de predição. Entretanto, por mais preciso que seja um modelo, é natural que eles apresentem um percentual de falha. A definição dos limiares a partir da criticidade, impõe um nível de sobrecarga máximo aceitável. Quanto menor a classe de criticidade menor o limite empregado, visto que as VMs mais críticas serão menos tolerantes às falhas no modelo de previsão. Por outro lado, para classes de criticidade maiores, torna-se possível o emprego de limiares maiores, devido as VMs pertencentes a esse grupo serem menos sensíveis às oscilações no desempenho. Portanto, a utilização desses limiares tende a atenuar as implicações decorridas das falhas na previsão, o que gera um menor impacto no tempo de execução.

		Intervalo de Confiança	
Cenário	**Média**	**Inferior**	**Superior**
Dinâmico	84,50%	84,30	84,70
DSC	50,66%	50,35	50,97

Tabela 6.5: Média do Percentual da Equivalência do tempo de execução para os cenários Dinâmico e DSC.

6.2.2 Violação

Na Tabela 6.6 são exibidas as diferenças em pontos percentuais para a violação, entre os cenários DSC e Dinâmico, para cada classe de criticidade. Em todos os casos estima-se que há uma diferença positiva, sugerindo que o cenário DSC resulta em um maior número de violações que o cenário Dinâmico. No entanto, essa diferença é mais significativa para as classes de criticidade maiores (3 e 4).

Para a classe de criticidade 1, há também uma diferença considerável, visto que no cenário Dinâmico o nível de sobrecarga empregado para esta criticidade é igual a 1, resultando em um total de 0% de violações. Nesse sentido, é possível estimar com 95% de confiança que, caso não houvesse a classificação por criticidade, entre 28,44% e 29,35% das VMs mais críticas e sensíveis ao desempenho apresentariam violações.

Em contrapartida, para a classe de criticidade 2, estima-se que a diferença média na violação em pontos percentuais é pequena, estando entre 0,08% e 2,63%. Isso indica que, para este caso, a classificação das VMs por criticidade exerceu uma influência no percentual de violação que pode ser considerada pequena ou irrelevante.

		Intervalo de Confiança	
Classe de Criticidade	**Média**	**Inferior**	**Superior**
1	28,91	28,44	29,35
2	1,36	0,08	2,63
3	38,99	38,54	39,43
4	43,45	42,62	44,28

Tabela 6.6: Diferença média da violação em pontos percentuais, entre o cenário DSC e Dinâmico para cada Classe de Criticidade.

6.2.3 Percentual de Máquinas Físicas Alocadas

No que diz respeito ao percentual de máquinas físicas alocadas, é possível observar na Figura 6.5 e Tabela 6.7 que o cenário Dinâmico apresentou um maior percentual se comparado ao cenário DSC. Esse comportamento é ratificado pela diferença média entre o cenário DSC e Dinâmico, estimada entre -3,36 e -2,67, com 95% de confiança. No entanto, para conjuntos

de sobrecarga maiores, é possível que essa diferença diminua, assim como para conjuntos de sobrecarga menores, como os apresentados na seção 6.1.3, é provável que a diferença aumente.

Essa maior consolidação observada no cenário DSC ocorre provavelmente devido ao modelo prever níveis de sobrecarga maiores do que os limites estipulados para o cenário Dinâmico. Como consequência, é possível haver um menor número de máquinas físicas em uso, mas com o custo de ter um maior número de violações e maior impacto no tempo de execução das VMs, conforme exposto nas seções 6.2.1 e 6.2.2. Ademais, a utilização de conjuntos de sobrecarga maiores pelo cenário Dinâmico pode resultar em uma maior consolidação, mas também devem ser levados em consideração os impactos supracitados.

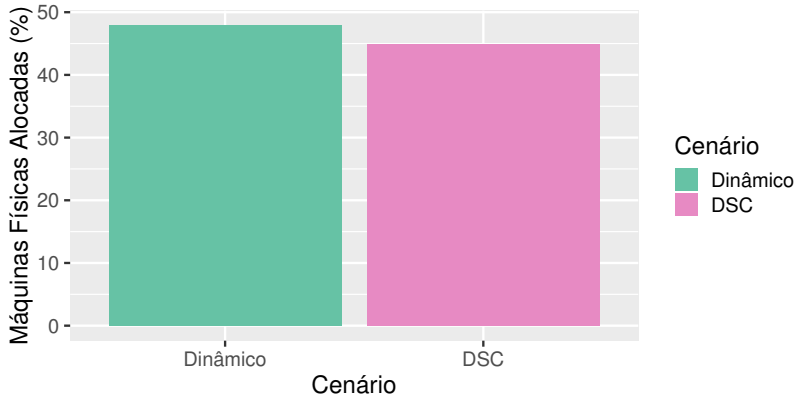


Figura 6.5: Percentual médio de máquinas físicas alocadas para os cenários DSC e Dinâmico.

		Intervalo de Confiança	
Cenário	**Média**	**Inferior**	**Superior**
DSC	44,84	44,55	45,12
Dinâmico	47,86	47,62	48,10
Diferença (DSC-Dinâmico)	-3,02	-3,36	-2,67

Tabela 6.7: Percentual médio de máquinas físicas alocadas para os cenários DSC e Dinâmico, bem como a diferença média estimada entre os cenários.

6.2.4 Consumo de Energia

Apesar do cenário DSC ter apresentado um menor percentual de máquinas físicas alocadas, o consumo de energia aferido ao término da execução foi superior ao cenário Dinâmico, como pode ser visto na Figura 6.6 e Tabela 6.8. A diferença média estimada entre o cenário DSC e o cenário Dinâmico está entre 114,52 kWh e 167,84 kWh. Essa diferença ocorre devido as máquinas físicas passarem um maior tempo ligadas, em virtude do maior período necessário para execução das *Cloudlets*. Isso indica que, apesar do cenário DSC permitir uma consolidação um pouco maior que o cenário Dinâmico, devido ao acréscimo no tempo de execução, isso resulta em um maior consumo de energia.

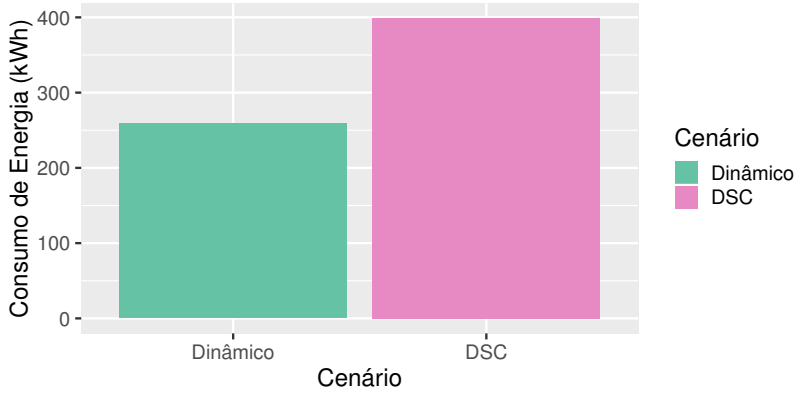


Figura 6.6: Média do consumo de energia do *Datacenter* de máquinas físicas alocadas para os cenários DSC e Dinâmico.

		Intervalo de Confiança	
Cenário	**Média**	**Inferior**	**Superior**
DSC	398,66	372,49	427,68
Dinâmico	258,92	257,23	260,56
Diferença (DSC-Dinâmico)	139,74	114,52	167,84

Tabela 6.8: Média do consumo de energia para os cenários DSC e Dinâmico, bem como a diferença média estimada entre os cenários.

6.3 Discussão

Esta Seção resume os achados referentes às respostas das questões de pesquisa, elaboradas no Capítulo 5 e relembradas aqui:

- **Questão de Pesquisa 1 (QP1)**: A abordagem proposta, em comparação com as demais abordagens, consegue promover um melhor equilíbrio no *trade-off* entre aumentar a utilização de recursos e degradação de desempenho?
- **Questão de Pesquisa 2 (QP2)**: A inclusão da classificação das VMs por criticidade, fornece melhores resultados, se comparado a utilizar apenas a abordagem de atribuição dinâmica de sobrecarga?

QP1. Tendo em vista todos os aspectos considerados neste estudo, os resultados sugerem que cenários sem sobrecarga favorecem os clientes, pois seus serviços tendem a ser executados de acordo com o desempenho esperado. Entretanto, não é benéfico para o provedor devido à ineficiência na utilização de recursos e maior consumo de energia. Em contrapartida, o cenário com sobrecarga estática tende a favorecer o provedor, por permitir uma maior consolidação e menor consumo de energia. Entretanto, foi visto que a adição de sobrecarga estática pode prejudicar o desempenho das VMs, prejuízo esse influenciado pelas características das cargas de trabalho em execução.

O emprego de um mecanismo dinâmico de sobrecarga, aliado a diferenciação dos tipos de serviço e sua criticidade, demonstrou ser uma solução que promove um melhor equilíbrio entre melhorar a utilização de recursos e a ameaça de degradação de desempenho. Nesse cenário foi possível usufruir dos benefícios da sobrecarga, mas ao mesmo tempo alcançar uma maior preservação da performance das VMs, cujos serviços não toleram degradações de desempenho, fornecendo um melhor nível de comprometimento entre provedor e cliente.

QP2. A classificação das VMs de acordo com sua criticidade exerce uma maior influência na preservação do desempenho e diminuição de violações se comparada a abordagem que utiliza puramente a sobrecarga dinâmica. Em contrapartida, a depender do limiar máximo estabelecido por classe de criticidade, pode haver um maior número de máquinas físicas sendo utilizadas, porém a longo prazo resulta em um menor consumo de energia, se comparado ao cenário que utiliza apenas a sobrecarga dinâmica.

Em face do exposto, levando em consideração os resultados alcançados e as soluções já realizadas pela comunidade científica, considera-se que o mecanismo proposto neste trabalho é uma alternativa ao gerenciamento de sobrecarga. Por meio dele, os provedores de nuvem podem fazer uso da sobrecarga e usufruir de seus benefícios, mas reduzindo os prejuízos ao desempenho e pagamento de multas em virtude de violações do SLO. No entanto, vale ressaltar que o mecanismo é passível de melhorias, que podem trazer bons resultados e novos estudos para essa área.

Capítulo 7

Considerações Finais

Este trabalho foi desenvolvido com o intuito de propor alternativas ao gerenciamento de sobrecarga de recursos na computação em nuvem. Para isso, foi desenvolvida uma abordagem para atribuição dinâmica de sobrecarga de CPU, que leva em consideração a criticidade e perfil de utilização de CPU de cada VM. A abordagem apresentada objetiva propor um melhor equilíbrio no *trade-off* entre aumentar a utilização de recursos e a degradação do desempenho.

Baseado no algoritmo de triagem ESI, utilizado em hospitais de todo mundo, foi desenvolvida uma abordagem para definição de criticidade com o intuito de classificar a priorização que cada VM deve receber. A definição de qual classe de criticidade cada VM pertence é realizada com base em indicações do estado do serviço. O estado do serviço é definido a partir do nível do SLO, estabelecido pelo provedor, e do tipo de serviço, definido a partir da indicação dos clientes. Este último, fazendo uso do AHP como método de tomada de decisão de múltiplos critérios, por meio de comparações pareadas.

A classe de criticidade permite a definição de limiares máximos para delimitar os níveis de sobrecarga que poderão ser admitidos para cada VM. Assim, o mecanismo de atribuição dinâmica de sobrecarga, a partir de dados de monitoramento, cria um modelo de previsão de CPU para cada VM e, a partir disso, define um nível de sobrecarga igual ou inferior ao limite estabelecido pela classificação de criticidade.

A fim de avaliar a solução supramencionada, um conjunto de experimentos foram realizados e, no que se refere a promover um melhor nível de comprometimento entre provedor e cliente, os resultados mostram que a abordagem proposta foi, de forma significante, estatis-

ticamente superior em relação às demais abordagens utilizadas para comparação. Portanto, os resultados evidenciaram que o mecanismo apresentado permite usufruir dos benefícios da sobrecarga, mas reduzindo as degradações no desempenho, principalmente de VMs mais críticas.

7.1 Principais Limitações

Tendo em vista que o escopo definido neste trabalho resultou em algumas limitações, é ciente que a abordagem proposta tem capacidade de melhorias para atingir melhores resultados. Desse modo, abrindo novos horizontes que possam ser explorados com base nos resultados obtidos por este trabalho e com outras pesquisas que possam incrementar as informações aqui alcançadas, as principais limitações são elencadas a seguir:

- O mecanismo proposto considerou apenas a sobrecarga de CPU. Apesar da CPU ter sido considerada como o recurso de maior empecilho de ambientes em nuvem [17], outros recursos, como memória e disco, poderiam ser considerados no gerenciamento de sobrecarga. A concepção do mecanismo proposto pode ser facilmente estendida para a utilização de outros tipos de sobrecarga, no entanto, se faz necessário investigações e avaliações adicionais, a fim de analisar as implicações dessa extensão.

- Os resultados alcançados foram obtidos por meio de avaliações que levaram em consideração um conjunto de simplificações, como, por exemplo, considerar que todas as VMs são equânimes (solicitam a alocação da mesma quantidade de recursos). Essas simplificações foram necessárias para que fosse possível obter evidências das vantagens alcançadas com o mecanismo proposto. Todavia, julgamos que essas simplificações não exercem influência no que se refere ao principal propósito das avaliações, que é demonstrar as vantagens da solução apresentada.

- O AHP utilizado na definição do tipo de serviço requer a indicação dos usuários sobre sua percepção a respeito de quais recursos são mais importantes em relação aos demais. A avaliação considerou que os usuários utilizariam julgamentos condizentes com o real comportamento do serviço a ser alocado na VM. Durante a avaliação, foram simuladas três respostas, de forma que cada resposta resultasse em um tipo de

serviço. Nesse sentido, não foi considerado o cenário em que o usuário realiza julgamentos incoerentes. No entanto, conjecturamos que, como as cargas de trabalho em execução na nuvem tendem em sua maioria a apresentar menor consumo de CPU, há uma maior chance de serviços menos críticos em relação a CPU sejam considerados mais críticos. Portanto, possivelmente não haverá maiores impactos ao desempenho e aumento de violações. O que pode acontecer é haver um maior número de máquinas físicas em uso e aumento do consumo de energia, mas que deve ser atenuado devido a utilização do mecanismo de atribuição dinâmica de sobrecarga.

7.2 Trabalhos Futuros

Conforme mencionado na seção anterior, existe a possibilidade de incrementar e aprimorar o trabalho realizado de forma obter resultados melhores. Nesse sentido, pode-se destacar algumas possibilidades de trabalhos que podem dar continuidade a este:

- Expandir o mecanismo para considerar a sobrecarga de outros tipos de recurso, como memória e disco.
- Avaliar o mecanismo com outros algoritmos de aprendizado de máquina para previsão da utilização de recursos, analisando as consequências de sua utilização.
- Realizar experimentações que considerem a possibilidade de incoerência do julgamento por parte do usuário, para definição do tipo do serviço.
- Implementar o mecanismo proposto em sistemas de gerência de recursos, como OpenStack e realizar um conjunto de experimentos, a fim de realizar investigações em um ambiente mais realista.
- Investigar como o mecanismo proposto pode se adequar e expandir para considerar as particularidades de ambientes com contêineres.

Referências Bibliográficas

[1] Tahani Aladwani. Improving tasks scheduling performance in cloud computing environment by using analytic hierarchy process model. In *2017 International Conference on Green Informatics (ICGI)*, pages 98–104. IEEE, 2017.

[2] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. 2007.

[3] Salman A Baset, Long Wang, and Chunqiang Tang. Towards an understanding of oversubscription in cloud. In *2nd {USENIX} Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE 12)*, 2012.

[4] Anton Beloglazov and Rajkumar Buyya. Energy efficient allocation of virtual machines in cloud data centers. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 577–578. IEEE, 2010.

[5] Anton Beloglazov and Rajkumar Buyya. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience*, 24(13):1397–1420, 2012.

[6] Petrônio Carlos Bezerra et al. Uma abordagem evolucionária para o posicionamento dinâmico de máquinas virtuais em datacenters privados, aplicando sobrecarga de recursos. 2019.

[7] George EP Box, William H Hunter, Stuart Hunter, et al. *Statistics for experimenters*, volume 664. John Wiley and sons New York, 1978.

[8] David Breitgand, Zvi Dubitzky, Amir Epstein, Alex Glikson, and Inbar Shapira. Sla-aware resource over-commit in an iaas cloud. In *2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)*, pages 73–81. IEEE, 2012.

[9] Matteo Brunelli. *Introduction to the analytic hierarchy process*. Springer, 2014.

[10] Brendan Burns. The history of kubernetes the community behind it. `https://kubernetes.io/blog/2018/07/20/the-history-of-kubernetes-the-community-behind-it/`, 2018. Accessado em 16/06/2020.

[11] Faruk Caglar and Aniruddha Gokhale. ioverbook: intelligent resource-overbooking to support soft real-time applications in the cloud. In *2014 IEEE 7th International Conference on Cloud Computing*, pages 538–545. IEEE, 2014.

[12] Rodrigo N Calheiros, Rajkumar Buyya, and César AF De Rose. A heuristic for mapping virtual machines and links in emulation testbeds. In *2009 International Conference on Parallel Processing*, pages 518–525. IEEE, 2009.

[13] Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1):23–50, 2011.

[14] Rodrigo N Calheiros, Rajiv Ranjan, and Rajkumar Buyya. Virtual machine provisioning based on analytical performance and qos in cloud computing environments. pages 295–304. IEEE, 2011.

[15] Rodrigo N Calheiros, Rajiv Ranjan, César AF De Rose, and Rajkumar Buyya. Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services. *arXiv preprint arXiv:0903.2525*, 2009.

[16] M. W. A. Carvalho. Gerência de nuvens computacionais considerando diferentes classes de serviço. Tese (Doutorado em Ciência da Computação) – Programa de Pós-

Graduação em Ciência da Computação, Centro de Engenharia Elétrica e Informática, Universidade Federal de Campina Grande, 2016.

[17] Marcus Carvalho, Daniel Menasce, and Francisco Brasileiro. Prediction-based admission control for iaas clouds with multiple service classes. In *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 82–90. IEEE, 2015.

[18] Marcus Carvalho, Daniel A Menascé, and Francisco Brasileiro. Capacity planning for iaas cloud providers offering multiple service classes. *Future Generation Computer Systems*, 77:97–111, 2017.

[19] Inderveer Chana and Sukhpal Singh. Quality of service and service level agreements for cloud environments: Issues and challenges. In *Cloud Computing*, pages 51–72. Springer, 2014.

[20] Nitin Chawla and Deepak Kumar. Decision-making model to evaluate cloud computing service model using analytic hierarchy process (ahp) and benefits, costs, opportunities, and risks (bcor) analysis. *International Journal of Sensors Wireless Communications and Control*, 10(6):846–856, 2020.

[21] CLOUDS. Cloudsim: A framework for modeling and simulation of cloud computing infrastructures and services. `http://www.cloudbus.org/cloudsim/`, 2019. Accessado em 24/01/2020.

[22] Philip M Dixon. Bootstrap resampling. *Encyclopedia of environmetrics*, 1, 2006.

[23] David R Eitel, Debbie A Travers, Alexander M Rosenau, Nicki Gilboy, and Richard C Wuerz. The emergency severity index triage algorithm version 2 is reliable and valid. *Academic Emergency Medicine*, 10(10):1070–1080, 2003.

[24] Daji Ergu, Gang Kou, Yi Peng, Yong Shi, and Yu Shi. The analytic hierarchy process: task scheduling and resource allocation in cloud computing environment. *The Journal of Supercomputing*, 64(3):835–848, 2013.

[25] Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila. Lircup: Linear regression based cpu usage prediction algorithm for live migration of virtual machines in data centers. In *2013 39th Euromicro conference on software engineering and advanced applications*, pages 357–364. IEEE, 2013.

[26] Armando Fox, Rean Griffith, Anthony Joseph, Randy Katz, Andrew Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, and Ion Stoica. Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, 28(13):2009, 2009.

[27] Rahul Ghosh and Vijay K Naik. Biting off safely more than you can chew: Predictive analytics for resource over-commit in iaas cloud. In *2012 IEEE Fifth International Conference on Cloud Computing*, pages 25–32. IEEE, 2012.

[28] Anita Gibbs. Focus groups. *Social research update*, 19(8):1–8, 1997.

[29] Nicki Gilboy, Paula Tanabe, Debbie A Travers, Alexander M Rosenau, David R Eitel, et al. Emergency severity index, version 4: implementation handbook. *Rockville, MD: Agency for Healthcare Research and Quality*, pages 1–72, 2005.

[30] Brian Hayes. Cloud computing. *Communications of the ACM*, 51(7):9–11, 2008.

[31] Rachel Householder, Scott Arnold, and Robert Green. Simulating the effects of cloud-based oversubscription on datacenter revenues and performance in single and multi-class service levels. In *2014 IEEE 7th International Conference on Cloud Computing*, pages 562–569. IEEE, 2014.

[32] Venkata Josyula, Malcolm Orr, and Greg Page. *Cloud computing: Automating the virtualized data center*. Cisco Press, 2011.

[33] Haleh Khojasteh. Resource allocation and task admission control in cloud systems. 2015.

[34] Kyong Hoon Kim, Anton Beloglazov, and Rajkumar Buyya. Power-aware provisioning of cloud resources for real-time services. In *Proceedings of the 7th International Workshop on Middleware for Grids, Clouds and e-Science*, pages 1–6, 2009.

[35] Kubernetes. Kubernetes scheduler. :https://kubernetes.io/docs/concepts/scheduling-eviction/kube-scheduler/.

[36] Kubernetes. Pods. ://kubernetes.io/docs/concepts/workloads/pods/pod/.

[37] Kubernetes. What is kubernetes? :https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/.

[38] Aditya Makwe and Priyesh Kanungo. Scheduling in cloud computing environment using analytic hierarchy process model. In *2015 International Conference on Computer, Communication and Control (IC4)*, pages 1–4, 2015.

[39] Maciej Malawski, Gideon Juve, Ewa Deelman, and Jarek Nabrzyski. Algorithms for cost-and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds. *Future Generation Computer Systems*, 48:1–18, 2015.

[40] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[41] Faz Mohammad, Vandana Yadav, et al. Automatic decision making for multi-criteria load balancing in cloud environment using ahp. In *International Conference on Computing, Communication & Automation*, pages 569–576. IEEE, 2015.

[42] OPENSTACK. Overcommitting cpu and ram. `https://docs.openstack.org/arch-design/design-compute/design-compute-overcommit.html`, 2018. Accessado em 09/01/2020.

[43] Omer F Rana, Martijn Warnier, Thomas B Quillinan, Frances Brazier, and Dana Cojocarasu. Managing violations in service level agreements. In *Grid middleware and services*, pages 349–358. Springer, 2008.

[44] Thomas L Saaty. How to make a decision: the analytic hierarchy process. *European journal of operational research*, 48(1):9–26, 1990.

[45] Thomas L Saaty. Decision making with the analytic hierarchy process. *International journal of services sciences*, 1(1):83–98, 2008.

[46] Thomas L Saaty, Luis G Vargas, et al. *Decision making with the analytic network process*, volume 282. Springer, 2006.

[47] Thomas Lorie Saaty and Luis Gonzalez Vargas. *Prediction, projection, and forecasting: applications of the analytic hierarchy process in economics, finance, politics, games, and sports*. Kluwer Academic Pub, 1991.

[48] Upendra Sharma. Elastic resource management in cloud computing platforms. 2013.

[49] Siqi Shen, Vincent Van Beek, and Alexandru Iosup. Statistical characterization of business-critical workloads hosted in cloud datacenters. In *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 465–474. IEEE, 2015.

[50] Sukhpal Singh and Inderveer Chana. Cloud resource provisioning: survey, status and future research directions. *Knowledge and Information Systems*, 49(3):1005–1069, 2016.

[51] Sukhpal Singh and Inderveer Chana. A survey on resource scheduling in cloud computing: Issues and challenges. *Journal of grid computing*, 14(2):217–264, 2016.

[52] A-Young Son, Yeon Soo Lim, and Eui-Nam Huh. Energy efficient vm placement scheme based on fuzzy-ahp system for sustainable cloud computing. In *2018 Second World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 260–265. IEEE, 2018.

[53] Yangyang Tao and Shucheng Yu. kfhco: Optimal vm consolidation via k-factor horizontal checkpoint oversubscription. In *2019 International Conference on Computing, Networking and Communications (ICNC)*, pages 380–384. IEEE, 2019.

[54] Luis Tomás and Johan Tordsson. Improving cloud infrastructure utilization through overbooking. In *Proceedings of the 2013 ACM Cloud and Autonomic Computing conference*, page 5. ACM, 2013.

[55] Luis Tomás and Johan Tordsson. An autonomic approach to risk-aware data center overbooking. *IEEE Transactions on Cloud Computing*, 2(3):292–305, 2014.

[56] Adel Nadjaran Toosi, Rodrigo N Calheiros, Ruppa K Thulasiram, and Rajkumar Buyya. Resource provisioning policies to increase iaas provider's profit in a federa-

ted cloud environment. In *2011 IEEE International Conference on High Performance Computing and Communications*, pages 279–287. IEEE, 2011.

[57] Luis M Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. A break in the clouds: towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, 39(1):50–55, 2008.

[58] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. Large-scale cluster management at google with borg. In *Proceedings of the Tenth European Conference on Computer Systems*, pages 1–17, 2015.

[59] Dinesh C Verma. Service level agreements on ip networks. *Proceedings of the IEEE*, 92(9):1382–1388, 2004.

[60] Jiahao Wu, Jing Xie, Alexandre Bardakoff, Timothy Blattner, Walid Keyrouz, and Shuvra S Bhattacharyya. Cgmbe: a model-based tool for the design and implementation of real-time image processing applications on cpu–gpu platforms. *Journal of Real-Time Image Processing*, pages 1–23, 2020.

[61] Linlin Wu, Saurabh Kumar Garg, and Rajkumar Buyya. Sla-based resource allocation for software as a service provider (saas) in cloud computing environments. In *Cluster, Cloud and Grid Computing (CCGrid), 2011 11th IEEE/ACM International Symposium on*, pages 195–204. IEEE, 2011.

[62] Liu Ying. Sponge: an oversubscription strategy supporting performance interference management in cloud. *China Communications*, 12(11):1–14, 2015.

Apêndice A

Questionário - AHP

Nas próximas páginas deste apêndice é apresentado o questionário desenvolvido e descrito na Seção 5.2.1. O link para o questionário está disponível em: `https://pt.surveymonkey.com/r/J9DNM39`.

SAIR

Questionário - Trabalho para dissertação de Mestrado

Descrição:
Este questionário foi elaborado pela estudante de mestrado, Marcela Tassyany Galdino Santos, da Universidade Federal de Campina Grande, do curso de Pós-graduação em Ciência da Computação.

Esse processo está sendo realizado para contribuir com a pesquisa que está sendo realizada pela mestranda e visa levantar a perspectiva de profissionais e usuários sobre diferentes tipos de serviço no que se refere a importância dos recursos computacionais para executá-los.

Privacidade:

* 1. Nenhum dado pessoal será coletado e as informações coletadas nesse questionário não serão usadas para outros fins.

☐ Concordo com os termos de privacidade

Próx.

SAIR

Questionário - Trabalho para dissertação de Mestrado

Sobre voc:

Primeiramente, gostaríamos de saber algumas informações sobre sua formação acadêmica e profissional.

* 2. Qual sua formação acadêmica:

- [ ] Cursando Técnico em Informática ou áreas afins
- [ ] Técnico em Informática ou áreas afins
- [ ] Cursando Superior em Ciência da Computação ou áreas afins
- [ ] Superior Completo em Ciência da Computação ou áreas afins
- [ ] Cursando Especialização/Mestrado em Ciência da Computação ou áreas afins
- [ ] Especialização/Mestrado em Ciência da Computação ou áreas afins
- [ ] Cursando Doutorado em Ciência da Computação ou áreas afins

- [ ] Doutorado em Ciência da Computação ou áreas afins

* 3. Quanto tempo de experiência você tem no mercado de trabalho em TI?

- ◯ Não tenho experiência no mercado de trabalho em TI
- ◯ menos de 1 ano
- ◯ de 1 a 2 anos
- ◯ de 3 a 5 anos
- ◯ de 6 a 10 anos
- ◯ mais de 10 anos

* 4. Em quais das seguintes áreas você tem mais conhecimento? (Selecione uma ou mais)

- [ ] Sistemas operacionais
- [ ] Redes de Computadores
- [ ] Computação em Nuvem
- [ ] Tecnologias de Virtualização
- [ ] Sistemas Operacionais para Smartphones (ex: Android)
- [ ] Processamento de Imagens
- [ ] Banco de Dados

- [ ] Não tenho conhecimento sobre nenhuma das áreas mencionadas

Outro (especifique)

Anter. Próx.

Questionário - Trabalho para dissertação de Mestrado

Cenário 1:

Um pesquisador desenvolveu uma aplicação para o processamento de imagens para reconhecimento facial em tempo real.

Esse pesquisador precisa montar a infraestrutura mais adequada para execução de sua aplicação. As próximas questões permitirão que você o ajude por meio de comparações, julgando os recursos computacionais que são mais importantes para execução dessa aplicação.

Os valores utilizados na escala correspondem as seguintes definições (em módulo):

Valores	Definição
0	Igual importância
2	Pouco mais importante
4	Mais importante
6	Muito mais importante
8	Extremamente mais importante
1,3,5,7	Valores Intermediários

5. Entre **CPU** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Memória

* 6. Entre **CPU** e **Disco** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Disco

* 7. Entre **Disco** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

Disco Igual Memória

8. Deseja realizar algum comentário adicional?

Anter. Próx.

Desenvolvido pela

Questionário - Trabalho para dissertação de Mestrado

Cenário 2:

Uma indústria alimentícia pretende implantar um sistema de controle de estoque em uma de suas unidades.

Essa indústria precisa montar a infraestrutura mais adequada para execução de seu sistema de controle de estoque. As próximas questões permitirão que você a ajude por meio de comparações, julgando os recursos computacionais que são mais importantes para execução dessa aplicação.

Os valores utilizados na escala correspondem as seguintes definições (em módulo):

Valor	Definição
0	Igual importância
2	Pouco mais importante
4	Mais importante
6	Muito mais importante
8	Extremamente mais importante
1,3,5,7	Valores Intermediários

* 9. Entre **CPU** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Memória

* 10. Entre **CPU** e **Disco** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Disco

* 11. Entre **Disco** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

Disco Igual Memória

12. Deseja realizar algum comentário adicional?

Anter. Próx.

SAIR

Questionário - Trabalho para dissertação de Mestrado

Cenário 3:

Um desenvolvedor está utilizando o Android Studio para desenvolver uma aplicação Android.

Esse desenvolvedor precisa montar a infraestrutura mais adequada para desenvolver sua aplicação Android. As próximas questões permitirão que você o ajude por meio de comparações, julgando os recursos computacionais que são mais importantes para execução dessa aplicação.

Os valores utilizados na escala correspondem as seguintes definições (em módulo):

Valor	Definição
0	Igual importância
2	Pouco mais importante
4	Mais importante
6	Muito mais importante
8	Extremamente mais importante
1,3,5,7	Valores intermediários

* 13. Entre **CPU** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Memória

* 14. Entre **CPU** e **Disco** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Disco

* 15. Entre **Disco** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

Disco Igual Memória

16. Deseja realizar algum comentário adicional?

SAIR

Questionário - Trabalho para dissertação de Mestrado

Cenário 4:

Um pesquisador desenvolveu uma aplicação que coleta dados da rede por meio do TCPDump, os processa e analisa em tempo real e exibe informações de forma amigável para o usuário.

Esse pesquisador precisa montar a infraestrutura mais adequada para execução de sua aplicação. As próximas questões permitirão que você o ajude por meio de comparações, julgando os recursos computacionais que são mais importantes para execução dessa aplicação.

Os valores utilizados na escala correspondem as seguintes definições (em módulo):

Valor	Definição
0	Igual importância
2	Pouco mais importante
4	Mais importante
6	Muito mais importante
8	Extremamente mais importante
1,3,5,7	Valores Intermediários

* 17. Entre **CPU** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Memória

* 18. Entre **CPU** e **Disco** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Disco

* 19. Entre **Disco** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

Disco Igual Memória

20. Deseja realizar algum comentário adicional?

Anter. Próx.

SAIR

Questionário - Trabalho para dissertação de Mestrado

Cenário 5:

Um desenvolvedor criou um software compactador de arquivos semelhante ao WinRAR.

Esse pesquisador precisa montar a infraestrutura mais adequada para execução de sua aplicação. As próximas questões permitirão que você indique por meio de comparações quais recursos computacionais são mais importantes para execução dessa aplicação.

Os valores utilizados na escala correspondem as seguintes definições (em módulo):

Valor	Definição
0	Igual importância
2	Pouco mais importante
4	Mais importante
6	Muito mais importante
8	Extremamente mais importante
1,3,5,7	Valores Intermediários

* 21. Entre **CPU** e **Memória** qual recurso é mais importante para a execução dessa aplicação?

Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Memória

* 22. Entre **CPU** e **Disco** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

CPU Igual Disco

* 23. Entre **Disco** e **Memória** qual recurso é mais importante para a execução dessa aplicação? Mova a escala abaixo em direção ao recurso que você julga ser mais importante.

Disco Igual Memória

24. Deseja realizar algum comentário adicional?

Anter. Conclu do