UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Guilherme Arthur Geronimo

MÉTODO PARA ORGANIZAÇÃO DE MÁQUINAS VIRTUAIS NA NUVEM BASEADO EM MÚLTIPLOS OBJETIVOS E CUSTOS DE IMPLEMENTAÇÃO

Florianópolis

2016

Guilherme Arthur Geronimo

MÉTODO PARA ORGANIZAÇÃO DE MÁQUINAS VIRTUAIS NA NUVEM BASEADO EM MÚLTIPLOS OBJETIVOS E CUSTOS DE IMPLEMENTAÇÃO

Tese submetida ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do Grau de Doutor em Ciência da Computação.
Orientador: Prof. Dr. Carlos Becker Westphall
Universidade Federal de Santa Catarina

Florianópolis

2016

Ficha de identificação da obra elaborada pelo autor através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

A ficha de identificação é elaborada pelo próprio autor

Maiores informações em:
http://portalbu.ufsc.br/ficha

Guilherme Arthur Geronimo

MÉTODO PARA ORGANIZAÇÃO DE MÁQUINAS VIRTUAIS NA NUVEM BASEADO EM MÚLTIPLOS OBJETIVOS E CUSTOS DE IMPLEMENTAÇÃO

Esta Tese foi julgada adequada para a obtenção do Título de "Doutor em Ciência da Computação" e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Florianópolis, 20 de Julho 2016.

Prof[a]. Dr[a]. Carina Friedrich Dorneles
Coordenadora do Programa

Banca Examinadora:

Prof. Dr. Carlos Becker Westphall
Universidade Federal de Santa Catarina
Orientador

Prof. Dr. Edmundo Roberto Mauro Madeira
Universidade Estadual de Campinas – UNICAMP

Prof. Dr. Mario Marques Freire
Universidade da Beira Interior – Covilha – Portugal
(Videoconferência)

Prof. Dr. Bruno Richard Schulze
Lab. Nacional de Computação Científica – LNCC
(Videoconferência)

Prof. Dr. Elder Rizzon Santos
Universidade Federal de Santa Catarina

Prof. Dr. Jean Everson Martina
Universidade Federal de Santa Catarina

Dedico esta tese aos meus pais, que sempre apoiaram minhas escolhas.

RESUMO

O paradigma de Computação em Nuvem pode ser definido como um conjunto de computadores (servidores) designados a prover seus recursos na forma de serviços. Embora este paradigma traga muitos benefícios para os seus consumidores e provedores, o gerenciamento indevido dos seus componentes (como a localização das máquinas virtuais) acentua problemas relacionados ao desperdício de recursos. Apesar de ser um problema de difícil otimização (*NP-Hard*), devido a quantidade de possíveis soluções, diversas metodologias foram propostas para mitigar a questão da Alocação de Máquinas Virtuais. Porém, estas abordagens geralmente se concentram em objetivos e/ou cenários específicos, tornando-se impraticáveis em alguns modelos de Nuvem (e.g. Infraestrutura como um Serviço – IaaS). Desta maneira, identificamos a necessidade de uma solução agnóstica aos objetivos e que foque nos seguintes fatores: (i) a *Seleção* das máquinas virtuais a serem migradas, (ii) a *Geração* dos cenários que podem ser adotados e (iii) a *Variabilidade* dos cenários criados. No entanto, não foi encontrado nenhum método, ou conjunto de ferramentas (Framework), que contemple o problema de Alocação de Máquinas Virtuais utilizando agnosticismo aos objetivos e que considere o custo de implementação das soluções. Assim, esta tese propõe: (i) um modelo de Nuvem orientado para problemas de Alocação de Máquinas Virtuais e (ii) um conjunto de métodos, agnósticos aos objetivos da Nuvem, para Provisionamento e Organização de máquinas virtuais. O modelo representa o ambiente através de Regras, Qualificadores e Custos. Baseado nestes elementos, os métodos utilizam estratégias de *Seleção*, *Geração* e *Variabilidade* guiadas pela classificação de suas alocações. Por fim, introduzimos uma estratégia que usa o *Custo-Benefício* para selecionar um dos vários cenários encontrados no processo, levando em consideração o custo de implementação da solução e os qualificadores do ambiente. Os resultados obtidos foram executados com dados de um ambiente real e demonstraram que o método possui um tempo de execução viável para utilização em ambientes de produção e retorna um resultado próximo ao cenário ótimo.

Palavras-chave: Computação em Nuvem, Alocação de Máquinas Virtuais, Otimização de Múltiplos Objetivos

ABSTRACT

The Cloud Computing paradigm can be defined as a set of computers (servers) designed to provide their resources as services. Although this paradigm brings many benefits to its consumers and providers, the mismanagement of its components (such as the virtual machines' placement) aggravates problems related to resources' wastage. Despite of being a difficult optimization problem (NP-Hard), because the number of possible solutions, several methods have been proposed to mitigate the virtual machines' placement issue. However, these approaches often focus on few objectives and/or specific scenarios, becoming impractical in some cloud models (e.g. Infrastructure as a Service - IaaS). In this way, we identified the need for an objective-agnostic solution and focus on the following factors: (i) the selection of virtual machines to be migrated, (ii) the generation of scenarios that can be adopted and (iii) the variability of the scenarios created. However, no method was found, or set of tools (Framework), that address the Virtual Machine Placement problem using agnosticism (to the objectives) and consider the implementation costs of the solutions. Thus, this thesis proposes: (i) cloud-oriented model for Virtual Machines allocation problems and (ii) a set of methods, agnostics to cloud targets, for provisioning and organization of virtual machines. The model represents the environment through rules, Qualifiers and costs. Based on these elements, the methods use selection strategies, Generation and Variability guided by the classification of their placements. Finally, we introduce a strategy that uses the Cost-Benefit relation to select one of several scenarios encountered in the process, taking into account the cost of implementing the solution and the qualifiers of the environment. The obtained results were executed with real environment's data and demonstrated that the method has a viable execution time in production environments, and finds a result close to an optimal setting.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ARM	Acorn RISC Machine
CND	Cenário Não Dominado
CPU	Central Processing Unit
IaaS	Infraestructure as a Service
MBFD	Modified Best Fit Decreasing
MOB	Mono-Objetivo
MOM	Múltiplos-Objetivos como Mono-Objetivo
MOs	Múltiplos-Objetivos
ND	Não Dominado
PaaS	Plataform as a Service
PMO	Puramente Múltiplos-Objetivos
PMs	Physical Machines
RLC	Regra Livre de Contexto
RSC	Regra Sensível ao Contexto
SaaS	Software as a Service
SLA	Service Level Agreements
SLO	Service Level Objectives
TMA	Taxa Média de Alocação
VMP	Virtual Machine Placement
VMC	Virtual Machine Consolidation
VMs	Virtual Machines

SUMÁRIO

1 INTRODUÇÃO

O paradigma de Computação em Nuvem pode ser definido como um conjunto de computadores (servidores) designados a prover recursos computacionais, oferecendo aos consumidores infraestrutura, plataforma ou aplicações na forma de serviços, i.e. o paradigma resume-se a prover recursos computacionais como serviços. Uma das tecnologias chave para o desenvolvimento deste paradigma é a Virtualização, a qual permite a separação entre um Sistema Operacional e a máquina física (PM), na forma de máquinas virtuais (VMs). Isto proporciona um melhor isolamento dos Sistemas Operacionais, possibilitando a concentração de tipos heterogêneos em um mesma PM. Consequentemente, também proporciona a mobilidade das VMs através de *migrações*, em que o processamento de uma VM é transferido de uma PM para outra. A Figura 1 apresenta a separação entre VMs e PMs e um processo de migração no qual a VM1 é migrada da primeira PM para a terceira PM.

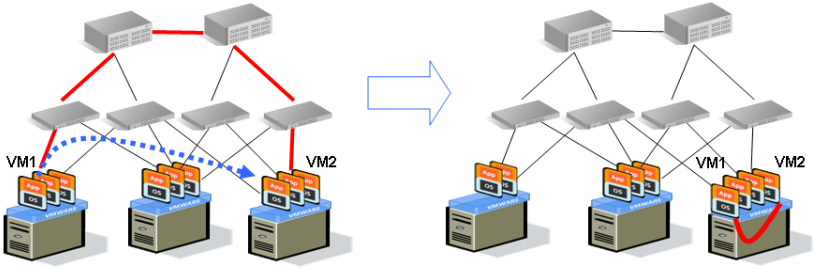

Figura 1 – Ilustração da Migração de Máquinas Virtuais

A Computação em Nuvem cresceu rapidamente nos últimos anos, sendo essencial para muitos negócios e tornando-se o produto principal de muitas empresas. Segundo Konstantinos et al. [1], de 2008 a 2014 estimou-se que o valor das indústrias de Computação em Nuvem cresceu em 226%, passando de 46 para 150 bilhões de dólares. Ainda, é previsto que mais de 50% de toda a tecnologia da informação esteja na Nuvem nos próximos 5 a 10 anos [2]. Consequentemente, este paradigma da computação se tornou alvo de interesse de vários campos da ciência.

No entanto, apesar do paradigma prover muitos benefícios aos consumidores e provedores de Nuvem, a administração inadequada de seu ambiente normalmente acentua questões relacionadas ao desperdício de recursos. Podemos citar como exemplos destes problemas:

(i) a degradação da performance das VMs, devido a competição por recursos [3], (ii) o surgimento de focos de alta temperatura nos centros de dados [4] e (iii) a redução de recursos devido às constantes migrações [5]. Além disto, Uddin et al. [6] identificaram que aproximadamente 30% das PMs em produção nos Centro de Dados apresentam um estado ocioso, tendo uma utilização média inferior a 10% de sua capacidade total, representando grande desperdício de recursos energéticos e computacionais.

Assim, diversas abordagens foram propostas para mitigar estes problemas, formando um novo campo chamado de otimização de Alocação de Máquinas Virtuais (Virtual Machine Placement – VMP). Este campo foca em encontrar a melhor organização para as VMs na Nuvem, i.e. tomando como referência a Figura 1, trata-se de encontrar a *combinação* de VMs em PMs que melhor corrobore com os objetivos da Nuvem.

Devido a similaridade, o problema de VMP pode ser reduzido ao problema de otimização de Múltiplos Objetivos (MOs), em que, dado um conjunto de soluções que beneficiam objetivos diversos, busca-se a solução que melhor os atenda. Sendo VMP um problema de combinação (de VMs em PMs), assim, é um problema difícil de se resolver (NP-Hard), visto a vasta quantidade de soluções possíveis. Na prática, as soluções de VMP utilizam meta-heurísticas para responder uma ou mais das seguintes perguntas: (i) *Qual* VM deve ser migrada? (ii) *Onde* ela deve ser hospedada? e (iii) *Quando* deve ser migrada? [7]. Da mesma maneira, as soluções podem ser subdivididas em categorias, as quais, como mostrado na Tabela 1, geralmente focam mais em algumas perguntas, e.g. provisionamento de novas VMs, organização das VMs na Nuvem, previsão da necessidade de recursos e alocação dinâmica de recursos sob demanda.

Soluções	Qual?	Onde?	Quando?
Provisionamento		✓	
Organização	✓	✓	
Previsão	✓		✓
Alocação Dinâmica		✓	✓

Tabela 1 – Foco das Soluções de Alocação de Máquinas Virtuais

Após revisar análises sistemáticas do campo [8–15] e produzir nossa própria revisão sistemática de soluções de VMP (Apêndice A), identificamos que as seguintes possibilidades podem ser exploradas:

Falta de Modelos de Nuvem contemplando MOs :
Cada proposta de solução para VMP utiliza um modelo de Nuvem personalizado, com elementos próprios e focada nos objetivos que visam resolver. Esta especialização impede a representação de várias facetas que os MOs almejam cobrir, e.g. Acordos de Nível de Serviços (SLAs), políticas internas e Objetivos de Nível de Serviço (SLO). Visto que cada faceta possui sua própria natureza — quantificável ou qualificável —, existe a necessidade de se representar e implementar elementos do problema VMP de uma forma padronizada.

Problemas na Resolução de Conflitos e Convergência :
A fim de aumentar a convergência e evitar conflitos entre os MOs, propostas usualmente buscam seus resultados dentro de conjuntos de soluções não-dominadas (Conjunto de Pareto). Uma solução domina outra se ela não é pior em nenhum objetivo e é melhor em pelo menos um objetivo em relação a outra. Porém, esta estratégia pode estagnar em ambientes com muitos objetivos [16], em situações em que nenhuma solução é boa o bastante para todos os objetivos simultaneamente, ou restringir demasiadamente o universo de busca. Por outro lado, apesar das restrições do ambiente reduzirem o número de soluções possíveis [17], não considerar a dominância das soluções geralmente causa um crescimento exponencial deste número, inviabilizando a busca.

Falta de Agnosticismo nos Métodos :
Apesar da abordagem de Múltiplos-Objetivos ser difundida, os trabalhos que afirmam utilizar MOs utilizam uma quantidade limitada dos mesmos. Nenhuma proposta que abordasse VMP e fosse agnóstica aos objetivos foi identificada. Encontramos propostas de alocação de carga de trabalho (*workload*) que almejam modelos agnósticos, porém este é um problema diferente de VMP. Agnosticismo[1], neste caso, está relacionado ao desacoplamento existente entre os métodos de otimização e os objetivos do problema que está sendo otimizado, ou seja, o método tem conhece apenas valores normalizados das avaliações do problema.

Incoerência entre Métodos :
Devido ao fato do VMP poder ser dividido em subproblemas, como Provisionamento, organização e alocação dinâmica de VMs, existe uma tendência a tratar estes subproblemas de forma separada e diferenciada, em termos de objetivos e estratégias. Esta divisão é adotada visando simplificar o problema e focar as solu-

[1] Agnóstico vem do grego: a-gnostos, ou seja, não-conhecimento.

ções. Contudo, os objetivos do ambiente continuam os mesmos, independente da questão que está sendo resolvida no momento. Desta maneira, isto acaba dessincronizando as definições (regras e objetivos) do ambiente, uma vez que cada método, responsável por cada subproblema, age de forma independente.

Desconsideração dos Custos de Implementação em VMP : Todo procedimento dentro da Nuvem consome algum tipo de recurso – e.g. tempo, energia, recursos computacionais –, inclusive a implementação de cenários, i.e. migrações para se alcançar uma solução. Em nossa revisão sistemática, identificamos que poucas propostas consideram o custo de implementação como um fator de decisão no processo, geralmente tratando-o apenas como mais um indicador a ser minimizado. Porém muitas vezes há uma quantidade máxima de recursos que pode ser investida na melhoria da Nuvem; no entanto, não foi encontrado nenhum trabalho que abordasse o problema desta maneira. Assim, identificamos a necessidade de, dentro do processo da resolução do VMP, avaliar a relação entre os benefícios alcançados com os recursos disponíveis para implementá-los.

Assim, nesta tese, almejamos atacar os seguintes problemas: (i) como modelar o ambiente de Nuvem com MOs a fim de otimizá-lo; (ii) que meta-heurística utilizar para organizar a nuvem, considerando que temos recursos limitados no processo de melhoria; e (iii) que solução selecionar, visto que existem várias possíveis soluções com benefícios semelhantes.

Considerando o acima exposto, resumidamente, esta tese propõe: (i) uma meta-heurística baseada na classificação das Alocações das VMs; (ii) um modelo de Nuvem orientado ao problema de VMP e agnóstico aos objetivos para amparar o método; (iii) um conjunto de métodos para provisionamento e organização de VMs na Nuvem de alta convergência – i.e. focados em responder as perguntas *Qual?* e *Onde?* e (vi) um Método para Adaptação de Cenários não conformes com as regras da Nuvem, os quais são especificados na sequência.

1.1 OBJETIVOS

Nesta seção, iniciamos apresentando o objetivo geral da tese e, em seguida, detalhamos os objetivos específicos ressaltando as contribuições ao estado-da-arte.

1.1.1 Objetivos Gerais

O objetivo geral desta tese é conceber (i) uma **meta-heurística** para VMP que considere a qualidade das alocações e o custo de implementação da solução final; e, para amparar esta, (ii) um **modelo de Nuvem**, voltado para o problema de VMP, que considere MOs, custos e investimentos do ambiente. A implementação destes elementos foi feita na forma de um arcabouço – um *Framework* nomeado de Order@Cloud – de código aberto e disponível para a comunidade no site do projeto.

1.1.2 Objetivos Específicos

A seguir, detalhamos os objetivos específicos desta tese:

Modelo de Nuvem para VMP com MOs e Custos – Definir um modelo com todos os elementos necessários para se descrever um ambiente de Nuvem com MOs e que também represente os custos e limites das alterações no ambiente. Um modelo idealizado para resolução de problemas de Alocação de Máquinas Virtuais. Abordamos esta questão na Seção 4.1.

Meta-Heurística que contemple MOs e Custos – Propor uma meta-heurística que utilize os elementos do modelo para encontrar o cenário com o melhor custo-benefício possível. Além disto, deve utilizar estes elementos para acelerar a convergência da resolução do problema, o qual é considerado *NP-Hard*. Esta meta-heurística é proposta junto aos métodos de organização e provisionamento de alocações de VMs. Tratamos desta questão na Seção 4.2.

Validação – Realizar a validação da meta-heurística comparando seus componentes, qualidade de seus resultados e a sua escalabilidade com outras abordagens da literatura. Abordamos estes testes na Seção 5.

Order@Cloud: Implementação do modelo e métodos – Disponibilizar para a comunidade uma implementação do modelo e métodos na forma de um Framework. O qual possibilite a fácil e rápida implementação de novos elementos do modelo e permita a utilização da meta-heurística proposta para organização e provisionamento de VMs. No Apêndice B descrevemos a implementação.

1.2 CENÁRIO, ESCOPO E INEDITISMO

Nossa proposta foi idealizada a fim de ser aplicada no Centro de Dados da UFSC. Focamos em ambientes de produção, com serviços heterogêneos, que proveem Infraestrutura como um Serviço (IaaS). Ou seja, a administração das VMs é restrita a localização de suas hospedagens, sem a possibilidade de reconfiguração das mesmas. Pelo fato de ser um ambiente de produção, isto impossibilita a reorganização do cenário com VMs desativadas e que ocupem o mesmo recurso simultaneamente, e.g. múltiplas migrações na (ou para a) mesma PM.

A tese foca em propor uma meta-heurística para VMP. Assim, nos concentramos em descrevê-la, como pseudo-código, e formalizar o modelo que essa se baseia. Provemos também as validações e comparações com outras abordagens, assim como aplicação desta na organização da nuvem, provimento de VMs e a adaptação dos cenários. Não menos importante, sobressaímos as opções e decisões tomadas na concepção da nossa proposta, na Seção 4.2.1. Desta maneira, está fora do escopo deste trabalho propor ou comparar objetivos ou restrições do ambiente. A Figura 2 é uma representação do nosso escopo, representando o que foi descrito acima.

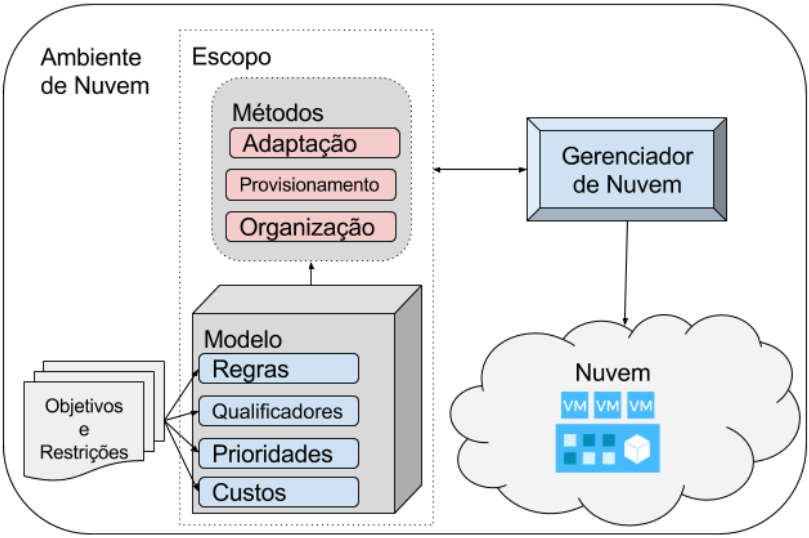

Figura 2 – Escopo da Tese

Sobre o ineditismo de nosso trabalho, foi desenvolvido uma extensa análise baseada em revisões sistemáticas da literatura [8–15] e também desenvolvemos nossa própria revisão sistemática (Apêndice A). Desta maneira, podemos afirmar que este é o primeiro trabalho a propor uma meta-heurística que prioriza as VMs nas piores alocações da Nuvem e busca por cenários que ofereçam melhor Custo-Benefício de implementação. Assim como um modelo de Nuvem agnóstico aos objetivos e que representa os custos e limites de investimentos do ambiente.

1.3 PERGUNTAS DA PESQUISA

Nesta seção apresentamos as perguntas que nortearam o desenvolvimento desta tese:

1. ***Como definir o problema e abordar VMP de forma agnóstica aos Múltiplos-Objetivos?***

   Existem várias formas de se abordar o problema de VMP. No entanto, devido a dinamicidade da Nuvem e suas restrições – regras, políticas e objetivos – faz-se necessária uma abordagem que contemple múltiplos-objetivos e ao mesmo tempo seja flexível ao ponto de não depender de suas instâncias, pois objetivos são mutáveis em natureza e prioridade. Deve haver uma definição clara sobre as restrições e premissas do problema i.e. saber que ações podem ou não devem ser adotadas, e.g. podemos alterar as configurações das VMs? Ou somente migrá-las. Isto é importante pois guia a escolha dos elementos que farão parte do modelo a ser utilizado, podendo vetar a adoção de métodos e heurísticas ao longo do trabalho. Abordamos esta pergunta na Seção 2.3.

2. ***Como descrever uma Nuvem quando se almeja resolver problemas de VMP agnósticos aos Objetivos?***
   Após a definição do problema, o modelo de representação é o passo seguinte para resolvê-lo. Logo, precisamos definir quais elementos e informações são necessárias para lidar com problemas de VMP. Não menos importante, estes devem representar os múltiplos objetivos provenientes do ambiente. Esta questão é abordada nas Seções 3.2 e 4.1.

3. ***Como acelerar a convergência dos métodos sem estagnar os mesmos?*** Sendo um problema de combinação entre VMs e PMs, VMP é um problema HP-Hard de ser resolvido, podendo gerar, no pior caso, $O(y^x)$ soluções possíveis, onde x e y referem aos números de VMs e PMs do cenário, respectivamente. Desta maneira, a filtragem de soluções não-dominadas tem sido adotada [18, 19] e demonstrou-se uma alternativa efetiva. Porém, segundo Lucken et al. [16], a sua utilização só é possível com quantidades limitadas de objetivos – inferior a 5 –, esta tende a estagnar a evolução dos métodos e ainda limita a qualidade do resultados alcançados. Estes pontos serão abordados com maiores detalhes na Seção 2.3. Todavia, a sua adoção também soluciona o problema de conflito entre os MOs do ambiente e, por esse motivo, o uso dessa abordagem se faz necessária ainda que seja preciso mitigar a estagnação por ela causada.
4. ***Como o problema e os métodos de VMP mudam quando o Custo de Implementação é limitado?*** Não adianta identificar uma solução ótima se sua implementação não é possível. Desta maneira, é necessário identificar como a existência de limites de implementação modifica o problema e as soluções de VMP, i.e. como isto altera os métodos que definem quais VMs serão migradas (*Qual*) e quais PMs deverão hospedá-las (*Onde*). Ainda, visto que cada ambiente apresenta um custo de implementação diferente – e.g. tempo, número de migrações, tráfego de rede –, deve-se idealizar os métodos de forma que contemplem a existência de uma ou mais limitações de Custo, durante o processo de melhoria. Abordaremos este tópico no Capítulo 3, no qual discutiremos algumas propostas e na Seção 4.2, em que apresentaremos a nossa proposta.

1.4 HIPÓTESE

Nossa hipótese é de que a adoção de uma meta-heurística baseada na classificação de alocações de VM, em conjunto com a filtragem por Dominância, alcança soluções com benefícios próximos (error menor de 10%) e em um tempo consideravelmente menor, em comparação com outras abordagens. No entanto, para isto, precisamos modelar a Nuvem de um modo agnóstico aos MOs do seu ambiente, e acreditamos que isto é possível definindo apenas regras, qualificadores, custos, prioridades e investimentos. A partir deste modelo, podemos propor a meta-heurística apropriada ao problema de VMP.

1.5 ORGANIZAÇÃO DO TRABALHO

Esta tese é organizada da seguinte maneira:

Na Seção 2 definimos conceitos básicos para compreensão desta tese, termos mais importantes e a definição do problema. Aqui lidamos com a Pergunta 1, definindo o problema de Alocação de Máquinas Virtuais.

Na Seção 3 discutimos os trabalhos relacionados, focando nas propostas de modelos de Nuvem, *frameworks* para Nuvem e abordagens de organização. Ressaltamos as principais propostas que abordam as perguntas desta tese. Ao fim, descrevemos a metodologia utilizada no desenvolvimento de nossa revisão sistemática. O material completo está disponível no Apêndice A.

Na Seção 4 apresentamos o Modelo de Nuvem que visa representar MOs genéricos e custos de Implementação; os métodos do Framework *Order@Cloud* e a discussão sobre os mesmos. Respondemos as Perguntas 2, 3 e 4.

Na Seção 5 apresentamos os experimentos que validam nossos métodos, comparando-os com outras abordagens e trabalhos relacionados. Aqui testamos a proposta das Perguntas 3 e 4.

Na Seção 6 encerramos esta tese, evidenciando os benefícios da nossa solução e propostas para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, apresentamos conceitos fundamentais para a compreensão da tese. Iniciamos conceituando Computação em Nuvem e os elementos do seu ambiente. Em seguida, devido a pluralidade de interpretações, definimos termos importantes deste campo. Por fim, elucidamos o problema de alocação de VMs em ambientes de Nuvem.

2.1 COMPUTAÇÃO EM NUVEM E SEU AMBIENTE

Segundo Brynjolfsson et al. [20], a definição de Computação em Nuvem varia de acordo com o ponto de vista. Do ponto de vista acadêmico, Computação em Nuvem refere-se tanto às (i) aplicações disponibilizadas como serviço na internet quanto (ii) aos Softwares e Hardwares nos centros de dados que proveem estes serviços. A junção destes elementos seria chamada de Nuvem. Do ponto de vista prático, Computação em Nuvem seria o acesso transparente aos recursos de TI virtualizados, geralmente (i) localizados fora do centro de dados local, (ii) compartilhado por outros, (iii) simples de usar, (iv) pago por assinatura, e (v) acessado pela web.

Em setembro de 2011, o órgão responsável por padrões do governo norte americano (*National Institute of Standards and Technology – NIST*), publicou a versão final de sua definição sobre *Cloud Computing* em [21]. Esta definição é composta de cinco características essenciais, quatro tipos de implementação e três modelos de serviço, expostos na Figura 3 e explanados a seguir.

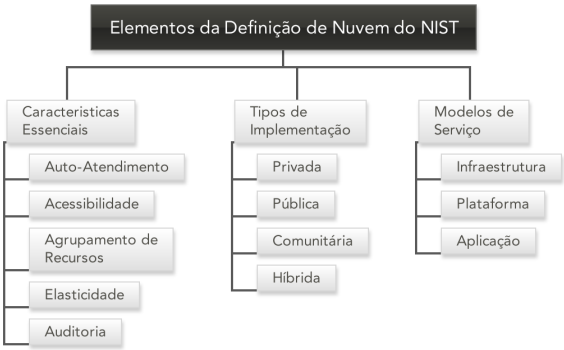

Figura 3 – Características de Computação em Nuvem do NIST

Apresentamos a seguir as 5 *características essenciais* da Nuvem:

Auto-atendimento sob demanda: Disponibilizar ao consumidor capacidades de computação, e.g. Tempo de processamento, armazenamento e prioridade de rede, conforme a necessidade e desejo. Este serviço deve ser provido na forma de auto-atendimento, sem a necessidade de interação humana com o prestador do serviço.

Acessibilidade: Os recursos são disponibilizados pela internet e acessados por meio de interfaces web (GUI e/ou WebServices), visando a interoperabilidade entre equipamentos e sistemas heterogêneos.

Agrupamento de recursos: Os recursos computacionais (e.g. armazenamento, processamento, memória e tráfego de rede, físicos ou virtuais) são agrupados e atribuídos dinamicamente de acordo com a demanda dos consumidores. Os consumidores geralmente não têm controle ou conhecimento da localização dos recursos disponibilizados, mas podem ser capazes de especificá-los em um nível maior de abstração, como por exemplo o centro de dados, estado ou país em que os dados serão processados.

Elasticidade rápida: O volume de recursos alocados são reservados e liberados de forma automática ou manual para se ajustarem ao aumento ou redução da demanda.

Medições: Os recursos são controlados e otimizados automaticamente pelo Gerenciador da Nuvem. Este monitora e dispõe seus dados em relatórios, visando proporcionar transparência aos consumidores e provedores da Nuvem.

A seguir, os 4 *tipos de implementação* da definição de Nuvem:

Nuvens Privadas: Seu acesso é restrito a membros da sua organização, podendo ser administrada pelo próprio órgão ou por terceiros. Segundo Sotomayor et al. [22], o objetivo principal de uma Nuvem privada não é vender recursos pela Internet, mas sim dar aos usuários locais uma infraestrutura ágil e flexível para suportar cargas de trabalho dentro de seu próprio domínio administrativo. Devido a sua natureza privada, este modelo teoricamente é o que promove menores riscos.

Nuvens Públicas: A infraestrutura da Nuvem é disponibilizada ao público geral ou a um grande grupo empresarial [21]. Este serviço é fornecido por uma organização que vende ou simplesmente disponibiliza o serviço. Esse conceito proporciona às organizações uma economia em escala, por compartilhar recursos. Por outro lado, limita-se a customização e aumenta a preocupação com a

segurança dos dados, SLAs e políticas de acesso, uma vez que os dados podem estar distribuídos em lugares desconhecidos e de difícil recuperação.

Nuvens Comunitárias: É provisionada para uso exclusivo por uma comunidade específica de consumidores de organizações que têm preocupações comuns, como projetos, metas e objetivos. Pode ser controlada, gerenciada e operada por uma ou mais organizações da comunidade, ou um terceiro.

Nuvens Híbridas: É a utilização conjunta de uma ou mais Nuvens distintas (privadas, públicas ou comunitárias), porém tratada como uma única entidade. Estas infraestruturas são unidas por tecnologias proprietárias ou padronizadas, e.g. os *WebServices* utilizados pela Amazon EC2 [23].

A seguir, os 3 *modelos de serviço* da definição de Nuvem:

Infraestrutura como um Serviço (Infrastructure as a Service – IaaS) é o serviço base definido pelo NIST e é o foco desta tese. Este serviço provê ao cliente processamento, armazenamento, rede e outros requisitos básicos para instalar um Sistema Operacional. O cliente possui controle total sobre o sistema operacional, podendo requisitar a alteração dos seus recursos, como memória, quantidade de processadores e acesso à rede (e.g. firewall). Ressalta-se que estes recursos podem ou não ser virtuais. O provimento estrito de servidores físicos é também chamado de Equipamento como um Serviço (Hardware as a Service – HaaS ou Metal as a Service – MaaS).

Plataforma como um Serviço (Platform as a Service – PaaS) geralmente disponibiliza um conjunto de bibliotecas, ferramentas ou aplicativos para os usuários hospedarem suas próprias aplicações. O usuário possui controle limitado sobre algumas configurações da plataforma, mas não tem acesso ao sistema operacional e ambiente. Como exemplo desta plataforma temos o serviço Heroku [24], que provê hospedagem para aplicações de diversas linguagens, e o serviço Google App [25], que oferece uma plataforma para os usuários configurarem seus próprios serviço de e-mails, entre outros.

Aplicações como um Serviço (Software as a Service – SaaS) refere-se a utilização da Nuvem a fim de fornecer uma Aplicação na forma de Serviço, como por exemplo interfaces de e-mail (GMail [26]) e sincronização de arquivos (e.g. Dropbox [27] e Seafile [28]). Neste caso, a gerência do cliente fica restrita a aplicação que é

disponibilizada, e não possui acesso ou controle da infraestrutura que a suporta, como rede, servidores, sistema operacional ou armazenamento.

Cópia de Segurança como um Serviço (Backup as a Service – BaaS) é um modelo que não está dentro dos descritos pelo NIST. Este pode ser considerado um desdobramento do SaaS, porém vem sendo visto como um modelo a parte pois abrange os três modelos anteriores. Este serviço provê aos usuários cópias de segurança de seus dados, proveniente dos diferentes serviços supracitados (e.g. VMs, Arquivos ou Dados), assim como procedimentos para a restauração dos mesmos. Segundo [29], as estratégias de cópia de segurança ocorrem em diversos níveis: (i) Dados específicos de aplicações, como dados de um banco de dados ou e-mails; (ii) Nível de Arquivos, como arquivos e pastas a serem salvas; e (iii) ao Nível de Blocos, como partições de um servidor ou mesmo imagens de VMs.

2.1.1 Virtualização

Em uma definição prática, pode-se dizer que *Virtualização* é o processo de abstração da camada de hardware no qual máquinas físicas (PMs) são configuradas para poderem hospedar máquinas virtuais (VMs), ou convertidas para serem processadas como VMs, como representado na Figura 4.

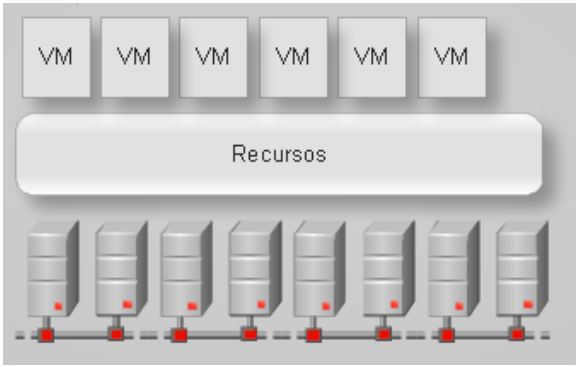

Figura 4 – Representação de Virtualização

A virtualização é baseada no fato de que a maioria dos serviços não utilizam todo o poder computacional que a máquina física tem a

oferecer. Deste modo, por via do compartilhamento dos recursos entre as várias máquinas virtuais, há uma otimização da utilização deste conjunto de recursos. Dessa forma diminui-se o número de máquinas físicas necessárias para processar um conjunto de serviços, pois podem ser processados em apenas uma máquina, economizando espaço nos centro de dados, energia elétrica e reduzindo a dissipação de calor.

Existem três tipos de implementação de virtualização:

Virtualização Total: O sistema operacional hospedeiro, que roda diretamente na máquina física, simula todo o hardware para permitir que um sistema operacional hospedado seja executado de maneira isolada e sem a necessidade de adaptações no sistema operacional hospedado, ou seja, é um caso de emulação. Tem-se como exemplo desta implementação o VMWare [30] e a solução livre KVM [31].

Virtualização auxiliada pelo hardware: O *hardware* deve possuir funcionalidades especiais em sua arquitetura, como suporte a execução de máquinas virtuais de modo isolado. Tem-se como exemplos de *hardwares* que implementam tais funcionalidades o Intel-VT e o AMD-V.

Paravirtualização: O sistema operacional hospedado na máquina virtual necessita passar por alterações para se adaptar ao ambiente virtualizado, formando uma ligação forte entre o sistema operacional hospedado e o sistema operacional hospedeiro. A comunicação entre o gerenciador de virtualização (*Hypervisor*) e os *drivers* do sistema operacional hospedeiro é dada por uma API, o que provê acesso direto ao hardware. Esta técnica também pode ser utilizada em conjunto com as outras técnicas, simultaneamente. Tem-se como exemplo deste a solução livre Xen [32].

A virtualização é a peça fundamental na disseminação da Computação em Nuvem, pois proveu a flexibilidade necessária para que o conceito se desenvolvesse e popularizasse. Simultaneamente, ela possibilitou o desenvolvimento de novas estratégias de: (i) crescimento horizontal, (ii) sob demanda e (iii) disponibilidade dos serviços, auxiliando os provedores de serviços a cumprirem seus Acordos de Nível de Serviços (SLA).

2.2 TERMOS UTILIZADOS NO TRABALHO

A fim de evitar erros de interpretação, muitas vezes causados na tradução ou por falso cognato, vamos clarificar alguns termos e conceitos essenciais que são utilizados nesta tese:

Alocação (*Placement*): Refere-se a localização da VM dentro da Nuvem. É a relação entre uma VM e uma PM, hóspede e hospedeiro, respectivamente.

Relocação (*Relocation*): Este termo se refere ao ato de alterar o local de hospedagem de uma VM. Esta ação pode auxiliar os administradores a atingirem objetivos, como liberar algum recurso específico e/ou melhorar a performance da Nuvem.

Realocação (*Reallocation*): Termo relacionado a designação de recursos ou conjunto de recursos, tais como CPU, memória ou recursos disponíveis para uma conta. O ato de (re)designar recursos entre VMs ou Conjuntos de Recursos, i.e. mudar a configuração de VMs durante a sua existência [33], dos recursos que estão sendo utilizados ou especificar a utilização de um recurso. Por exemplo, reduzir a memória de uma VM ociosa, ou mover as VMs de um serviço em produção para um armazenamento de alta performance.

Provisionamento (*Provisioning*): Este termo está geralmente relacionado a ação de decidir onde hospedar uma nova VMs. Já o Provisionamento Dinâmico (*Dynamic Provisioning*) contempla a necessidade de prover recursos para VMs já existentes, i.e. mapear e agendar *realocações* e *relocações* a fim de liberar e prover recursos [34] para VM em expansão. Por exemplo, caso um cliente contrate mais memória para uma VM, antes de reconfigurá-la a solução deve relocá-la se necessário, acarretando em migrações e/ou realocações de recursos.

Consolidação de VMs (*VM Consolidation* – *VMC*): Este procedimento foca em centralizar VMs no mínimo de PMs possível. Almeja-se, desta maneira, desligar as PMs ociosas, reduzindo assim o consumo de energia, maximizando a utilização dos recursos empregados [35] e minimizando o custo do Centro de Dados [36].

Alocação de VMs (*Virtual Machine Placement* – *VMP*): O termo contempla todos os problemas de Realocação, Relocação e Otimização, em que o foco está sobre o processo de decisão da hospedagem das VMs. Diferente do VMC, este não faz referência a

objetivos, restrições, soluções ou os gatilhos específicos, a serem utilizados [12, 37, 38].

Dominância (*Dominance*): O conceito de Dominância é uma possível relação entre duas soluções distintas. Diz-se que uma solução domina outra se esta é melhor em pelo menos um objetivo, e não é pior em qualquer outro objetivo [16]. Sejam $S_1 = (p_1, ..., p_y)$ e $S_2 = (q_1, ..., q_y)$ duas soluções com y avaliações de objetivos. Diz-se que S_1 domina S_2 se $p_i \geq q_i$ para $i = 1, ..., y$ e $S_1 \neq S_2$. A Figura 5 representa graficamente um cenário com dois objetivos. As soluções marcadas com ($\star$) são soluções não-dominadas, enquanto as soluções marcadas com ($\otimes$) são dominadas pelos seus sucessores.

Alocação em Nuvem (*Cloud Placement*): Este termos é simultaneamente utilizado para se referir a Provisionamento e Relocação de: (i) VMs na Nuvem [39, 40] e (ii) requisições destinadas aos serviços hospedados na Nuvem, carga de trabalho (*workload*) [36, 41]. Isto é relevante pois são problemas diferentes com naturezas distintas. Por este motivo, convêm identificar o contexto que está sendo utilizado. Nesta tese, abordamos apenas o primeiro caso, (i).

Meta-Heurística: Originado do grego, *meta* significa "nível superior" ou "além" e *heurística* "conhecer" ou "investigar". Este é um termo dado a classe de algoritmos usados para encontrar soluções para problemas de otimização quando as técnicas exatas revelarem-se inadequadas, e.g. problemas *NP-Hard* [42].

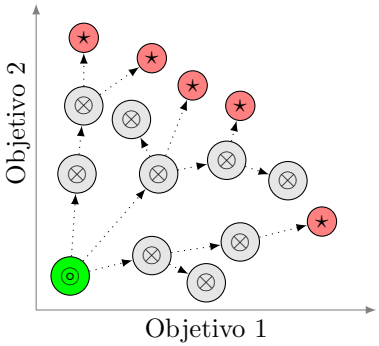

Figura 5 – Representação da Relação de Dominância

2.3 ALOCAÇÃO DE MÁQUINAS VIRTUAIS: O PROBLEMA

Nesta seção abordamos as diversas definições sobre o problema de VMP, assim como expomos nossa visão sobre este problema.

O problema de VMP resume-se em encontrar a melhor disposição das VMs nas PMs de uma Nuvem. Devido a similaridade, muitas vezes ele é reduzido ao Problema de Empacotamento em Caixas (*Bin Packing*), o qual é considerado NP-Hard [13]. Este problema consiste em organizar todos os objetos de um conjunto $O = (o_1, o_2, ..., o_n)$, cujo tamanho de O_i é dado por $t(o_i) \in]0,1]$, no menor número de caixas possíveis [43], i.e. dividi-los no mínimo de m subconjuntos $C_1, C_2, ..., C_m$ de tal forma que não exceda o tamanho de cada subconjunto, como exposto na Equação 2.1. Logo, em VMP as Caixas são relacionadas às PMs, os Objetos às VMs, e seus volumes a quantidade de recurso consumido pelas VMs, como memória e processamento.

As propostas que usam esta abordagem geralmente adotam como solução algoritmos da família *First Fit* [15], os quais oferecem uma rápida convergência e uma solução quase ótima. No caso do algoritmo *First Fit*, ele ordena o conjunto de objetos de forma decrescente e adiciona cada elemento na primeira caixa disponível. Este método apresenta uma complexidade de $O(nlogn)$. Outras variações alteram o tipo da ordenação dos objetos e das caixas, mudando também a sua complexidade. Esta é uma solução comumente utilizada [44–46] em casos de consolidação das VMs, a fim de se reduzir custos, como consumo de energia, tráfego de rede e/ou armazenamento.

$$\sum_{o_i \in C_j} t(o_i) \leq 1 \qquad (2.1)$$
$$\forall 1 \leq j \leq m$$

Por outro lado, quando a mensuração dos objetivos é mais complexa, como SLA, SLO e lógicas Fuzzy, a abordagem de Maximização (ou minimização) de Objetivos é adotada. No entanto, apesar de geralmente considerar Múltiplos Objetivos (MO), esta abordagem também pode utilizar apenas uma função para mensurar os objetivos, como explicaremos a seguir. Assim, diferentes soluções podem usar esta abordagem, e.g. Algoritmos Evolutivos, Busca Heurística ou Agentes Distribuídos.

Segundo Lopez et al. [15], esta abordagem pode ser classificada nas seguintes categorias:

Mono-Objetivo (MOB) contempla os problemas que focam em um objetivo específico e, portanto, possuem apenas uma função de mensuração para avaliar o objetivo, e.g. quantidade de memória desperdiçada, PMs ociosas ou VMs sobrecarregadas.

Múltiplos Objetivos Puros (PMO) são considerados todos os problemas que possuem múltiplas funções de avaliação e todas são consideradas, simultaneamente, durante o processo de decisão, e.g. métodos que utilizam a relação de dominância.

Múltiplos Objetivos como Mono-Objetivo (MOM) são os casos em que as funções que representam os múltiplos objetivos são reduzidas a um só valor, o qual representa o estado da Nuvem. Como exemplo de reduções podemos citar o somatório de valores, a quantidade de objetivos satisfeitos ou média dos valores.

Apesar dos métodos PMO apresentarem uma convergência geralmente alta, muitas vezes eles estagnam em ambientes com muitos objetivos, podendo atingir momentos em que nenhuma solução agrade a todos os objetivos simultaneamente [16].

Visto esta limitação, nesta tese propomos adotar uma abordagem de Maximização Híbrida. Utilizamos uma abordagem MOM para sumarizar as avaliações das alocações das VMs, com um produtório, e uma abordagem PMO para filtrar os cenários relevantes, com base nas avaliações sumarizadas. Estas abordagens serão explanadas de forma detalhada na Seção 4.2.

Lopez et al. [15] também evidencia que a maioria das propostas tratam de *Custo* como um objetivo a ser minimizado, direta ou indiretamente, e.g. evitando multas contratuais ou reduzindo o consumo de energia. No entanto, acreditamos que o *Custo* também assume um papel de contrapartida no processo de melhoria da Nuvem, representando a quantidade de recursos – tempo, tráfego, dinheiro – que o administrador está disposto a investir no processo de implementação do novo Cenário. O que leva a reflexão de que: nada adianta identificar o cenário ótimo se o mesmo não pode ser implementado.

Considerando que existe uma quantidade pré-definida de recursos a serem investidos, deve-se existir uma prioridade nestes investimentos. Neste sentido, devemos priorizar a melhoria das VMs mal alocadas na Nuvem, i.e. as VMs que possuem baixas mensurações pelos Qualificadores. A fim da facilitar a narrativa desta tese, deste ponto em diante nos referiremos as estas VMs como *piores VMs*.

Não menos importante, consideramos como uma restrição do problema a não degradação da Nuvem, durante o processo de organi-

zação. Isto significa que durante a implementação do cenário final, não pode ser adotado um cenário pior que o anterior, obrigando a melhoria constante das alocações.

Por fim, consideramos que o objetivo do problema é encontrar um Cenário que maximize o *Benefício* alcançado e minimize o Custo de implantação, criando uma relação de Custo-Benefício. No entanto, visto que existe a possibilidade do benefício ser nulo e o custo de implementação é maior que zero, para fins de cálculo avaliamos a relação Benefício-Custo, e, simultaneamente, buscamos maximizar este valor. O benefício é dado pela variação da avaliação do cenário atual (C_{atual}) pelo almejado ($C_{solucao}$), como apresentado em (2.2). A sumarização das avaliações, a fim de viabilizar o cálculo do benefício, será abordada na seção 4.2.

Concluindo, a Figura 6 apresenta um resumo com os principais axiomas do problema, considerados nesta tese.

$$Max(Beneficio) \tag{2.2}$$
$$Beneficio = avaliacao(C_{solucao}) - avaliacao(C_{atual})$$

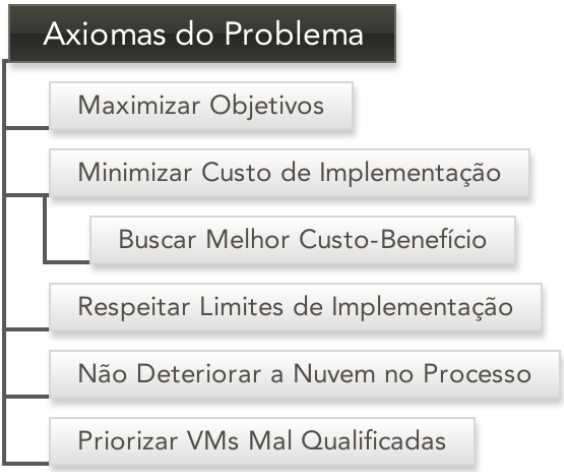

Figura 6 – Resumos dos principais pontos do problema

3 TRABALHOS RELACIONADOS

Neste capítulo trataremos dos conceitos e trabalhos relacionados que fundamentam e amparam esta tese. Discutiremos separadamente propostas que tratam sobre: (i) modelos para representação de Nuvens, (ii) soluções de VMP e (iii) Frameworks de VMP. Mas primeiramente, detalhamos a metodologia utilizada para a revisão bibliográfica durante o desenvolvimento desta tese e dos artigos [47–49].

3.1 METODOLOGIA

A fim de investigar o estado da arte das soluções para VMP, a pesquisa foi dividida em quatro fases.

Primeiramente foram selecionadas revisões bibliográficas (*Surveys*) sobre o assunto, a fim de se construir um conhecimento inicial. Nesta fase selecionamos revisões cujo foco reside em várias questões do VMP, como por exemplo objetivos almejados [8–10], caracterização da carga de trabalho [11] , monitoração [14], abordagens ao problema [12] e procedimentos de migração [13].

Na segunda fase, artigos dos anais de dezesseis eventos ([50–65] e doze periódicos [66–77] foram selecionados com base em seus assuntos e qualificações da CAPES (A1, A2 e eventos B1). Artigos entre 2010 e o inicio de 2015 foram separados para filtragem. O levantamento primário resultou na seleção de 14358 artigos. Neste caso, não utilizamos ferramentas de buscas que agregam artigos, estes foram adquiridos diretamente nos sites dos eventos e periódicos.

Na terceira fase, esses artigos foram filtrados por seus títulos e palavras-chave. Dois filtros foram aplicados, um com os termos que deveriam constar e outro com termos que não deveriam constar nestes campos, como apresentado na Tabela 2. Este filtro diminuiu a quantidade para 161 artigos.

	Palavras-Chave
Deve conter	Virtual Machines, Placement ou Consolidation
Não deve conter	Wireless, Security, Routing e Workload

Tabela 2 – Palavras-chaves Utilizadas na Filtragem dos Artigos Selecionados

O fato da quantidade de artigos reduzir consideravelmente nesta fase, em relação ao número inicial, é devido a classificação dos veículos de publicação. Eventos e periódicos com maior classificação tendem a ser mais abrangentes sobre o assunto que atendem. Desta maneira, visto que VMP é um problema aplicado a um campo específico (Computação em Nuvens), a sua representatividade em veículos como estes é um tanto reduzida. Não menos importante, os trabalhos de outros veículos não foram ignorados, mas sim alcançados através de referências e outras revisões sistemáticas do campo, citadas anteriormente.

Na última fase, cada um dos 161 artigos foram revisados com base no título e resumo. Os que foram considerados fora do escopo foram excluídos da lista. No final, 13 artigos foram selecionados para análise. O conteúdo integral, contendo os resumos dos artigos, tópicos analisados, discussões e tabela comparativa podem ser encontrados no Apêndice A.

3.2 MODELOS DE NUVENS

Em nossa revisão buscamos modelos de Nuvens que favorecem a resolução de problemas de alocação de VMs na Nuvem. O objetivo destes modelos é representar os elementos que devem ser considerados no processo de resolução do problema de alocação. A seguir, apresentamos os modelos mais importantes e que se destacaram.

Computação em Nuvem Orgânica: Revisamos primeiramente os modelos utilizados em trabalhos anteriores [78–80]. O modelo da Computação em Nuvem Orgânica (*Organic Cloud Computing*) é baseado na Teoria das Organizações e faz uma analogia entre a Nuvem e a organização de uma empresa. Neste modelo, agentes de monitoração e gerenciamento da Nuvem estão organizados hierarquicamente e possuem responsabilidades distintas, e.g. gerenciamento do sistema de refrigeração, suprimento de energia e rede. A Nuvem é vista como um conjunto de agentes e, a partir do momento que eles se intercomunicam, acabam se tornando uma "sociedade de agentes". Esta sociedade, para permanecer em ordem, deve ser organizada, classificada e hierarquizada, para não tender ao caos ou a exaustão de recursos. Este modelo de gerenciamento distribuído é representado na Figura 7.

Devemos destacar que este modelo é importante por simplificar o problema a dois conjuntos de elementos: Regras e Qualificadores. O modelo estipula que deve existir um conjunto de regras que vetam cenários e outro conjunto que afere a qualidade dos cenários.

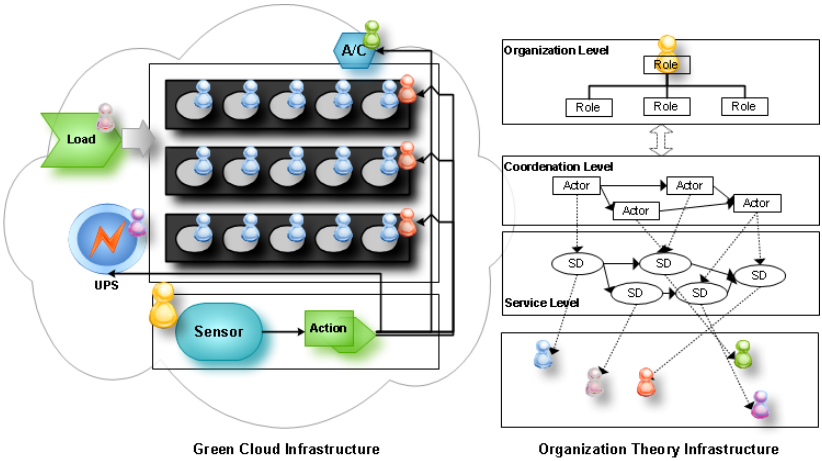

Figura 7 – Exemplo do Modelo Orgânico [79]

Apesar de trazer estes elementos, o modelo não possui uma descrição formal e foi concebido com outro foco: a reconfiguração e gerenciamento das VMs, questões que estão fora do escopo desta tese.

Provisionamento e Relocação baseados em MO: Xu et al. abordam as questões de provisionamento [37] e relocação de VMs [81] visando a redução de consumo de energia. Utilizam Algoritmos Genéticos para identificar melhores alocações para iniciar VMs.

Como a solução é focada em objetivos específicos, o modelo proposto é restrito somente aos dados necessários a estes objetivos, dados sobre temperatura, consumo de energia e utilização de recursos (memória e CPU). A Figura 8 lista cada componente do modelo.

Apesar de possuir múltiplos objetivos, a maneira que o modelo foi proposto não permite a extensão ou adição de novos objetivos, ou seja, não é agnóstico aos objetivos.

Cool Cloud: Este framework, proposto por Zhang et al. [82], visa encontrar a organização mais energeticamente eficiente (que consuma menos energia) em alocar as VMs na Nuvem, respeitando as restrições dos recursos. Desta maneira, além dos usuais dados sobre o consumo de energia e recursos utilizados, e.g. CPU e memória, são considerados também o espaço de armazenamento e tráfego de rede utilizados pelas VMs e PMs. Estes dados são empregados no processo de decisão de migração, a fim de não extrapolar os limites das PMs.

No entanto, o ressaltamos pelo fato de representar os custos envolvidos no processo de transição das VMs para as PMs. Neste caso,

M	Number of physical servers
N	Number of VM requests
$[c_j^{CPU} \; c_j^{mem}]$ $(j=[1...M])$	Capacity vector of the jth server
$[r_i^{CPU} \; r_i^{mem}]$ $(i=[1...N])$	Resource requirements of the ith virtual machine
$a_{ij} \in [0,1]$	Allocation matrix in which a_{ij}=1 if vm_i is allocated to the jth server
W_j	Resource wastage of jth server
P_j	Power consumed by the jth server
T_j	Temperature of the jth server
U_{CPU}	CPU utilization
U_{mem}	Memory utilization

Figura 8 – Modelo de Xu et al. [37]

estes custos são representados por duas matrizes (VM x PM), as quais trazem o custo de tempo e energia necessários para fazer migração. A Figura 9 exibe todos os elementos considerados neste modelo.

Apesar de propor inovações interessantes, o modelo ainda é limitado. A forma que os custos são definidos impossibilita a extensão para novos custos. Além do mais, a representação de possibilidades de migrações não leva em consideração o contexto que a Nuvem se encontra, i.e. não é possível representar restrições de alocação que exijam relações entre VMs.

Em direção a uma abordagem flexível: Seguindo a mesma linha de Xu et al., Abdul et al. [83] aposta em Algoritmos Genéticos para realocar as VMs da Nuvem. No entanto ele representa os dados de recursos de uma forma pouco diferente: seu modelo agrupa as VMs em Ambientes de Aplicações Virtualizadas (*Virtualized Application Environments - VAEs*), os quais representam grupos de VMs que pertencem ao mesmo serviço. Estes ambientes possuem agente de planejamento locais (LCP) os quais, baseando-se em funções de utilidade, visam maximizar a qualidade das VMs que gerenciam.

Além de introduzir a ideia de SLA em seu modelo, outra adição é a existência de priorização dos objetivos, em que um vetor de pesos é previso a fim de representar o peso de cada função de utilidade.

Apesar de propor um modelo muito mais completo que Xu et al., os dados relacionados ao SLA focam nas restrições técnicas (e.g. quantidade de recursos livres e disponibilidade de ambientes), negligenciando outras facetas relacionadas às questões administrativas da decisão de alocação, e.g. os centros de dados que podem hospedar a VM ou a taxa de indisponibilidade permitida. Desta maneira, apesar de

Symbol	Definition
N	Number of physical machines to serve virtual machines
M	Number of virtual machines
P_{active}	Basic power level of physical machines in active mode
P_{sleep}	Power level of physical machines in sleep mode
Period	Time period for which the solution pertains
$P_{mn}^{migrate}$	Power level for VM m migrating to PM n
$T_{mn}^{migrate}$	Time for VM m migrating to PM n
H^{CPU}	Limit on CPU utilization of physical machines
H^{MEM}	Limit on memory utilization of physical machines
H^{HD}	Limit on hard disk utilization of physical machines
H^{BW}	Limit on network bandwidth utilization of physical machines
$\mathbb{N}_{\mathbb{VM}}$	Set of VMs, $\lvert\mathbb{N}_{\mathbb{VM}}\rvert = M$
$\mathbb{N}_{\mathbb{PM}}$	Set of PMs, $\lvert\mathbb{N}_{\mathbb{PM}}\rvert = N$
$\mathbf{U^{CPU}}$	Virtual machine CPU utilization, $\mathbb{U}^{\mathbb{CPU}} = \{U_m^{CPU}, \forall m \in \mathbb{N}_{\mathbb{VM}}\}$
$\mathbf{U^{MEM}}$	Virtual machine memory utilization, $\mathbb{U}^{\mathbb{MEM}} = \{U_m^{MEM}, \forall m \in \mathbb{N}_{\mathbb{VM}}\}$
$\mathbf{U^{HD}}$	Virtual machine hard disk utilization, $\mathbb{U}^{\mathbb{HD}} = \{U_m^{HD}, \forall m \in \mathbb{N}_{\mathbb{VM}}\}$
$\mathbf{U^{BW}}$	Virtual machine network bandwidth utilization, $\mathbb{U}^{\mathbb{BW}} = \{U_m^{BW}, \forall m \in \mathbb{N}_{\mathbb{VM}}\}$
$\mathcal{E}$	Total system energy consumed
$\mathcal{L}$	Placement matrix (decision variable), $\mathcal{L} = (l_{mn})_{M \times N}$
$\mathcal{G}$	Migration matrix (decision variable), $\mathcal{G} = (g_{mn})_{M \times N}$
$\mathcal{O}$	Operation mode vector (decision variable), $\mathcal{O} = (o_n)_{1 \times N}$

Figura 9 – Modelo Proposto Para o Cool Cloud [82]

considerar SLA em seu modelo, este restringe-se a SLAs referentes aos recursos utilizados pelos Ambientes de Aplicações Virtualizadas, o que restringe a implementação de novas restrições. A Figura 10 apresenta a relação de elementos do modelo de Abdul et al. [83]

Analisando os modelos acima, e outros, somos levados a almejar um modelo que seja flexível o bastante para possibilitar a representação de qualquer elemento do problema. Porém, não deve ser específico ao ponto de representar detalhes técnicos dos objetivos tratados.

Ao mesmo tempo, acreditamos que o modelo ideal deva contemplar os custos envolvidos no processo de implementação do cenário final, não somente no sentido de restringir cenários cuja implementação seja impraticáveis, mas também no processo de seleção do cenário, construindo uma relação de custo-benefício.

M	Number of the running VAEs.
N	Total number of the VMs within the managed environment.
P	The total number of the PMs within the managed environment.
K	The total number of the instant types within the managed environment.
t	An integer valued number refers to the current discretized control interval.
t+1	An integer valued number refers to the next discretized control interval.
α_{SLA}	A faction refers to SLA performance targets in terms of the resources
L	The number of active VMS assigned to a specific PM.
P_i	Hardware capacity of *i*th PM within the managed environment.
P_i^{CPU}	Hardware capacity of ith PM in terms of processing power, (CPU).
P_i^{RAM}	Hardware capacity of ith PM in terms of memory, (RAM).
Pc	Total hardware capacity within the managed environment.
Θ_{rese}	Fraction from the hardware budget of *i*th VAE reserved for to guarantee service availability.
θ_i	Fraction of ith PM hardware capacity reserved to compensate the effects of virtualization overhead.
$v_{i,j,k}$	The standard computational capacity of ith VM within jth VAE class of kth instant type.
$v_{i,j,k}^{CPU}$	The standard computational capacity of ith VM within jth VAE class of kth instant type, in terms of processing power (CPU).
$v_{i,j,k}^{RAM}$	The standard computational capacity of ith VM within jth VAE class of kth instant type, in terms of memory (RAM).
Vc	The total computational power reserved for the running VAEs within the managed environment.
V_j^{class}	The share of jth VAE from Vc in terms of computational power.
$nmax_j$	The maximum number of VMs reserved for jth VAE within the managed environment.
$nmin_j$	The minimum number of VMs reserved for jth VAE within the managed environment.
n_j^t	The number of active VMs which serves jth VAE at t.
Vr_i^{t+1}	The requested workload demand for ith VAE class at next control interval.
V_{confg}^t	The status of the global configuration within the managed environment at t.
V_{status}^t	A Boolean vector refers to the activation status of VMs within the managed environment at t.
$vs_{i,j,k}$	A Boolean valued number refers to the activation status of ith VM within jth VAE class of kth type .
V_{frac}^t	A real valued vector refers to the utilization level of VMs within the managed environment at t.
$vf_{i,j,k}$	A real valued number refers to the utilization level of VMs within the managed environment.
V_{place}^t	An integer valued vector refers to the placement of the VMs on the underlying PMs within the managed environment at t.
$vp_{i,j,k}$	An integer valued number refers to the placement of ith VM within jth VAE class of kth type on the pth PM.
F	A Boolean vector refers to the availability flags.
U_g	Global utility function maintained by CM
Uj	jth VAE local utility function maintained by LCP
w	Tradeoff weights used to attached importance to a specific objective (i.e. local utility function)

Figura 10 – Modelo de Abdul et al. [83]

3.3 SOLUÇÕES DE VMP E SUAS ABORDAGENS

O problema de VMP vem sendo amplamente abordado e explorado nos últimos anos e, como resultado, uma série de revisões sistemáticas da bibliografia foram produzidas. Porém estas se restringem a explorar algumas facetas do problema, e.g. políticas de economia de energia, otimização de rede e serviços. No entanto, identificamos que muitos dos trabalhos focavam somente em propor políticas de alocação ou mensuração de objetivos, deixando outras questões fora de foco ou mal definidas, como explicaremos a seguir.

Assim, nesta seção selecionamos trabalhos que solucionam VMP com Buscas Heurísticas, Algoritmos Evolucionários e da família *First Fit*. Destacamos as questões *Qual?*, *Onde?* e *Quando?* consideradas, por Papadopoulos et al. [7], como as perguntas que sumarizam uma proposta de resolução de VMP:

- Qual VM migrar?
- Onde hospedá-la?
- Quando migrá-la?

3.3.1 Propostas que utilizam Busca Heurística

Xu et al., em [81], focam no provisionamento dinâmico de VMs com foco em três objetivos específicos: (i) o uso eficiente de recursos multidimensionais (CPU, memória, carga), (ii) evitar pontos quentes no centro de dados e (iii) reduzir os custos de consumo de energia.

A Figura 11 mostra o processo de decisão de *Qual* VM é selecionada para migração. O fluxograma demonstra que a Nuvem é monitorada buscando por algumas condições pré-definidas (eventos), como baixa eficiência energética, pontos de calor e competição por recurso. *Quando* uma VM aciona um destes gatilhos, é desencadeado o processo de relocação. No processo de alocação, o método busca todas as possíveis alocações para a VM selecionada para remoção. Em seguida, para cada possibilidade encontrada, é calculada a sua utilidade, baseando-se nos três objetivos supracitados. Ressaltamos este trabalho pela abordagem do *Onde*, a PM que prover o maior aumento de utilidade é selecionada, no entanto, não possibilita a priorização dos objetivos.

Liang et al. [18], apesar de não focar estritamente em VMs, propõe um novo algoritmo de Busca com Variação de Localidades baseado em MO (*Multi-Objective Variable Neighborhood Search* – MOVNS) para resolver problemas de alocação redundante com MO. Eles usam o

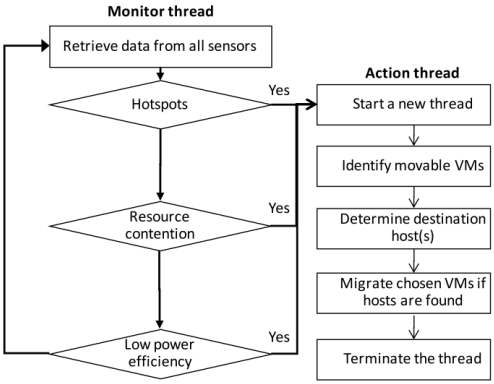

Figura 11 – Fluxo de Controle de Xu et al. [81]

Solução	Xu et al. [81]
Qual?	VM com maior relação CPU/Memória na PM alarmada
Onde?	Na PM que prover a maior utilidade
Quando?	Quando um dos gatilhos predefinidos for acionado

conceito de *Dominância* para reduzir o conjunto de cenários possíveis.

A abordagem começa a partir de um cenário aleatório válido e faz uma busca em largura por soluções não-dominadas, i.e. VMs aleatórias são migradas, com base no cenário inicial, gerando novos cenários; caso estes sejam não-dominados, eles são salvos. A busca continua a explorar os cenários salvos até o limite predefinido de avaliação ser atingido. Esta busca visa construir um conjunto de Pareto.

Ao fim da busca, a "localidade" é variada. Um novo cenário inicial aleatório é selecionado e a busca é reiniciada. No final, para avaliar o desempenho dos resultados encontrados, os seguintes indicadores são utilizados: Taxa de acerto, Taxa de Precisão, GD e $D1_R$. [16].

Ressaltamos esta abordagem pela utilização do conceito de Dominância e pelo uso da Variação de Localidade, fatores que aceleram a convergência do método e aumentam a variabilidade dos resultados, consequentemente melhorando a qualidade dos mesmos.

Solução	Liang et al. [18]
Qual?	Escolhas aleatórias.
Onde?	Em PM que gerarem cenário Não-Dominados.
Quando?	Não definido, manualmente.

3.3.2 Soluções com Algoritmos Evolucionários

Mark et al. propõe um algoritmo evolutivo para escolher em qual provedor de Nuvem uma nova VM deve ser instanciada, i.e. ele trabalha em um ambiente com múltiplas Nuvens. A proposta se aproveita da lacuna existente entre a quantidade de recursos *solicitada* e a *utilizada* pelo usuário. Com base em dados históricos do usuário, é possível reduzir os custos do usuário através da atribuição de menos recursos do que o solicitado.

Ressaltamos esta proposta por considerar no processo de VMP os dados (históricos) do usuários e o custo da hospedagem para este. Tais fatores são de extrema importância, porém muitas vezes o modelo não possibilita representá-los e, consequentemente, o método não consegue considerar tais informações.

Solução	Mark et al. [84]
Qual?	VM requisitada pelo usuário.
Onde?	No local que oferecer o menor somatório dos custos.
Quando?	Assim que a requisição chegar.

Visando os mesmos objetivos citados anteriormente — minimizar o desperdício de recursos, concentração térmicas e consumo de energia —, em [37], Xu et al. propõe um método para provisionamento de VMs na Nuvem. Assim, *Quando* chega a requisição de uma nova VMs, a *Qual* deve ser instanciada na Nuvem, eles utilizam um algoritmo genético para identificar PM *Onde* hospedar a nova VM. Os cenários de alocação são analisados usando uma função *fuzzy*, a qual resume as funções de avaliação dos objetivos.

Ressaltamos esta proposta pois seus métodos de *cruzamento* e *mutação*, que geram possíveis cenários, são baseados em avaliações de desempenho por grupos, em que um grupo representa um conjunto de VMs hospedadas na mesma PM, i.e. eles não qualificam o cenário como um todo, mas sim apenas um subconjunto de VMs dele.

Logo, esta proposta torna-se interessante pois identifica a necessidade de possibilitar uma análise flexível da Nuvem, i.e. permitir que diferentes funções de qualificação possam analisar conjuntos diversos de alocações, e.g. uma alocação, conjuntos de alocações e até a Nuvem como um todo.

Adotando uma proposta totalmente diferente das anteriores, ressaltamos Abdul-Rahman et al. [83] por propor um modelo de gestão

Solução	Xu et al. [81]
Qual?	A nova VM requisitada.
Onde?	Na PM que prover a maior utilidade.
Quando?	Assim que a requisição chegar.

de recursos visando reduzir a quantidade de migrações na Nuvem. Nesta proposta, os recursos disponíveis são utilizados para satisfazer os objetivos deficitários, mudando a configuração das VMs sob demanda. As decisões de otimização das VMs são delegadas aos agentes locais, no nível de serviços, i.e. as aplicações tem a possibilidade de redimensionar, instanciar e destruir VMs dos seus serviços. No entanto, caso as violações no SLA não consigam ser mitigadas por via das ações supracitadas, a proposta recorre para migrações, porém não informa maiores detalhes sobres a escolha de PMs e VMs.

Seus objetivos podem ser resumidos em: (i) minimizar a sobrecarga do sistema, liberando recursos ociosos, (ii) minimizar o consumo de recursos associados aos processos de migração, investindo na escalabilidade vertical e horizontal, i.e. redimensionamentos e instanciações, e (iii) maximizar o benefício dos usuários considerando QoS, SLA e elasticidade, e.g. por exemplo fornecer recursos de forma transparente para serviços e VMs que necessitam.

Solução	Abdul-Rahman et al. [83]
Qual?	As VMs do cenário que gerarem menos migrações.
Onde?	As PMs do cenário que gerarem menos migrações.
Quando?	Gatilhos relacionados aos SLAs forem acionados.

3.3.3 Soluções baseadas em Algoritmos *First Fit*

Chen et al., em [85], propõe uma organização de VMs baseada na afinidade de comunicação. O método identifica relações de afinidade entre VMs, analisando o tráfego de rede, e as agrupa em conjuntos. Então, aloca os conjuntos afins em PMs próximas. Ele usa um algoritmo guloso para alocar os recursos, porém não apresenta mais informações sobre o mesmo.

Ressaltamos este trabalho por ser um dos poucos a considerar a relação de afinidade entre VMs como parte do processo de organização da Nuvem. Uriarte et al. [3] ratificam a necessidade deste fator, visto que a competição por recursos de VMs afins degradam a performance dos serviços.

Solução	Chen et al. [85]
Qual?	Todas as VMs são agrupadas por trefego de rede.
Onde?	Não especificado.
Quando?	Sob demanda.

Fang et al., em [86], visam reduzir o consumo de energia do centro de dados consolidando VMs e desativando elementos ociosos da rede. A abordagem divide o problema de otimização em três fases. Em primeiro lugar, ele agrupa as VMs com o objetivo de maximizar o tráfego dentro dos grupos (intra-grupo) e minimizar o tráfego entre os grupos (inter-grupo). Em segundo lugar, tenta mapear as VMs nas PMs minimizando o tráfego entre os *Racks* (*inter-rack*) do centro de dados. Finalmente, o VMPlanner migra as VMs a fim de centralizar o tráfego de rede no menor número de caminhos (paths) possíveis. Isto permite desligar os equipamentos de rede ociosos e conservar energia.

Solução	Fang et al. [86]
Qual?	Todas as VMs são agrupadas por trefego de rede.
Onde?	PMs que abrigarem melhor um mesmo grupo.
Quando?	Sob demanda.

A fim de tratar separadamente os subproblemas do VMP, Beloglazov et al. o dividem em duas partes: (i) *Provisionamento* e (ii) *Optimização*. Para resolver o *Provisionamento*, é proposta uma versão modificada do algoritmo *Best Fit Decreasing* (MBFD), que mapeia as VMs nas PMs visando reduzir o consumo de energia, i.e. ele organiza as VMs de forma decrescente de utilização e aloca estas na PM que prover o menor aumento no consumo de energia.

Ressaltamos este trabalho pelo fato da fase de *Optimização* também ser dividida em duas fases: (i) *Extração* e (ii) *Realocação*. Caso seja necessário extrair uma VM, é proposta uma sequência de 3 políticas de *Extração*: (i) menor quantidade de migrações, (ii) menor potencial de crescimento e (iii) seleção aleatória. Em seguida, o processo de realocação é efetuado pelo algoritmo MBFD.

Solução	Beloglazov et al. [44]
Qual?	Propõe 3 políticas de seleção.
Onde?	PM que prover menor aumento de consumo de energia.
Quando?	Se uma PM atingir os limites de processamento.

3.4 FRAMEWORKS DE VMP

Segundo Fayad et al. [87], *Frameworks* são conjuntos de ferramentas e/ou bibliotecas que provêm uma estrutura reutilizável com a qual é possível construir novas aplicações, muitas vezes poupando incontáveis horas e centenas de dólares no custo do desenvolvimento.

Nesta seção apresentaremos alguns trabalhos que propõem Frameworks relacionados com VMP. No entanto, focaremos nas proposta que suportam a implementação de MO e Políticas, não sendo restritas a problemas ou tecnologias [82, 86].

Não menos importante, também abordaremos soluções de Nuvem de código-aberto, as quais possibilitam a implementação de regras e políticas de VMP, e.g. *OpenNebula* e *OpenStack*.

O projeto Open Nebula [88], iniciado em 2005, tem como objetivo servir de interface única para o gerenciamento de Nuvens privadas heterogêneas, intercomunicando com diversos hipervisores, e.g. VMWare, [30] e Xen [32]. Seu módulo de Provisionamento aplica uma *Política de Categorização de Agendamento* para a instanciação de novas VMs, a qual pode adotar um dos seguintes perfis [89]:

- Consolidação: prioriza as PM com mais VMs ativas;
- Distribuição: prioriza as PM com menos VMs ativas;
- Ciente da Carga (*Load-Aware*): prioriza as PM pela quantidade de CPU disponível;
- Prioridade: cada PM possui um peso pré-definido.

Já o projeto OpenStack [90] visa prover uma plataforma de Nuvem ubíqua e de código aberto que atenda às necessidades de Nuvens públicas e privadas, independente do seu tamanho, de forma escalável e fácil de implementar.

Seu módulo de Provisionamento, que também determina onde as VMs serão iniciadas, utiliza uma política personalizável com *Filtros* e *Pesos* para decidir qual PM será utilizada, como exposto na Figura 12. Esta política aplica uma série de filtros, implementados pelo framework e selecionados pelos administradores da Nuvem, para selecionar as PMs aptas a receberem a VM. Em seguida, ordena as PMs pelos seus pesos, os quais são calculados com base em métricas e prioridades definidas pelos administrador da Nuvem, e.g. quantidade de memória livre, capacidade e prioridade da PM [91].

Sobre as proposta acadêmicas, Feller et al., em [92], propõem uma estrutura de auto-organização, hierárquica e dinâmica, na qual as decisões são delegadas a agentes locais, dentro das PMs. A Figura 13

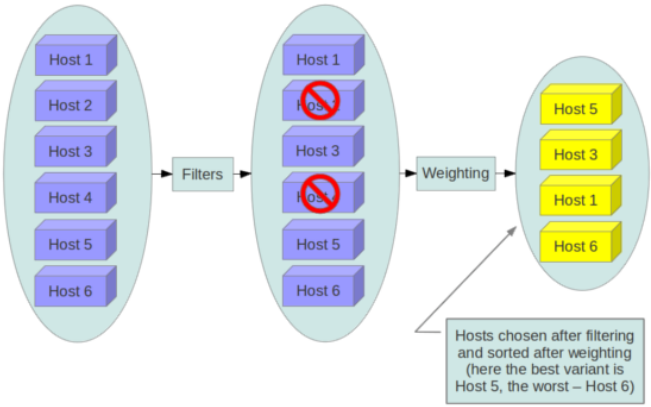

Figura 12 – Política de Filtros e Pesos do OpenStack [91]

representa a hierarquia dos elementos do Framework, o qual possui um Líder de Grupo, responsável por disponibilizar dados e coordenar todos os Gerentes de Grupos da Nuvem. Estes gerenciam subconjuntos de Controladores Locais (LC), que rodam dentro das PMs. Além de as monitorarem, os LCs podem tomar ações sobre as suas VMs hóspedes, tais como: migrar, redimensionar, iniciar, suspender, reiniciar, desligar e destruir.

Em seu *framework*, disponível no link [93], seus objetivos, gatilhos e agendamentos são representados como políticas do ambiente e submetidos aos agentes locais. Desta maneira, os administradores podem estender e implementar novas políticas para sanar questões específicas de seus ambientes.

Com uma abordagem totalmente diferente, Ren et al. em [94], propõem um Framework baseado na Teoria dos Jogos. Neste, é proposto um jogo em que as VMs são os jogadores e devem minimizar 5 objetivos pré-definidos: (i) Processador, (ii) Tráfego, (iii) Tempo de Resposta, (iv) Consumo de Energia e (v) Distribuição de Carga de Trabalho. Cada VMs tem seu turno para efetuar suas ações, sendo que as ações podem melhorar ou piorar um ou mais objetivos. Desta maneira, além de sugerir estratégias a serem adotadas pelas VMs, a proposta ainda permite a implementação de novas táticas.

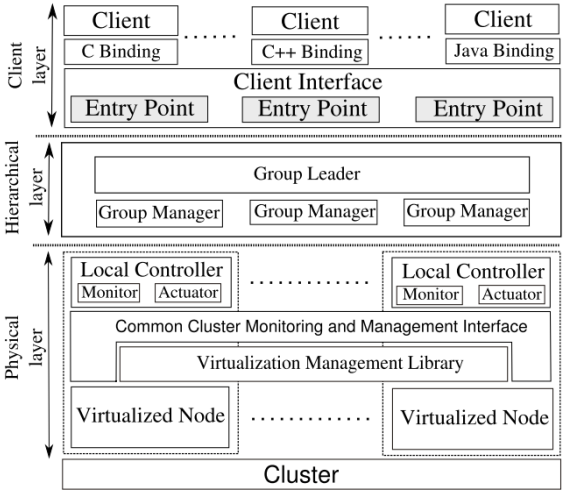

Figura 13 – Hierarquia dos Elementos do Framework Snooze [92]

4 ORDER@CLOUD: O MODELO E OS MÉTODOS

Neste capítulo apresentamos nossas propostas de modelo de Nuvem e Métodos para VMP. Primeiramente, descrevemos os elementos do modelo e definimos sua notação. Em seguida, apresentamos os métodos de: Organização e Provisionamento de VMs na Nuvem e Adaptação de Cenários de Nuvem, respectivamente. Por fim, discutimos o modelo e os métodos.

4.1 O MODELO DE NUVEM E SEUS ELEMENTOS

Nesta seção apresentamos o modelo de Nuvem proposto para problemas de VMP. Esta é uma compilação das versões que foram publicadas em [48, 49]. Inicialmente descrevemos e exemplificamos os seus elementos e, em seguida, os formalizamos matematicamente.

O modelo proposto foca nas lacunas encontradas nas propostas discutidas na Seção 3.2, que resume-se a representar os MOs mantendo o modelo agnóstico aos objetivos e contemplar os custos de implementação das soluções. Este modelo serve de base para os métodos propostos na Seção 4.2.

4.1.1 Descrição dos Elementos do Modelo

Ambientes reais de Computação em Nuvem devem ser orquestrados levando em consideração os MOs do ambiente. Neste modelo o ambiente é constituído por um ou mais dos seguintes elementos: Regras, Qualificadores, Prioridades e Custos. Assim como os objetivos são representados pelas Regras e Qualificadores. A Figura 14 representa estes elementos.

Como apresentado na Seção 3, uma *alocação* é definida como uma relação entre uma VM e uma PM, hóspede e hospedeiro, respectivamente. Um *cenário*, por sua vez, é um conjunto de alocações, as quais definem onde cada VM da Nuvem está hospedada, podendo representar a real situação da Nuvem ou uma situação hipotética. Ratificados estes conceitos, as próximas seções descrevem e exemplificam cada um dos elementos supracitados.

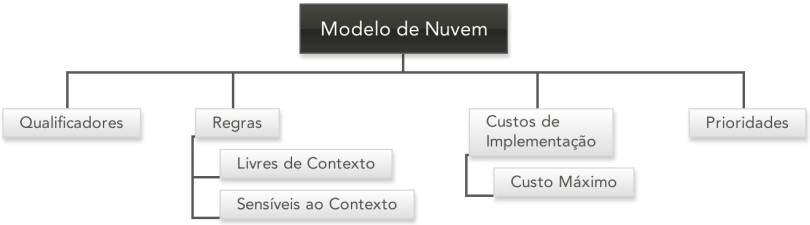

Figura 14 – Representação do Modelo de Nuvem e seus Elementos

4.1.1.1 Regras: As Restrições da Nuvem

Regras são elementos que restringem as alocações e os cenários de Nuvem, i.e. regras podem proibir que uma VM seja hospedada em uma PM ou que um cenário seja adotado pela Nuvem. Um ou mais objetivos podem ser representados por uma ou mais regras. Elas são divididas em duas categorias: (i) Regras Livres de Contexto (**RLC**) e (ii) Regras Sensíveis ao Contexto (**RSC**). Neste caso, *contexto* refere-se ao cenário analisado, pois este é o contexto em que as VMs estão inseridas. Logo, as RLC definem se uma dada VM pode ser hospedada em uma dada PM ou não, enquanto as RSC definem se um dado cenário é aceitável ou não. Em outras palavras, RLC validam alocações e RSC validam cenários.

A fim de ilustrar os tipos de regras, considere os exemplos abaixo:

- RLC – VM1 deve ser hospedada em uma PM que possua acelerador gráfico.
- RLC – VM2 deve ser hospedada no continente americano.
- RLC – Nenhuma VM deve ser hospedada nas PMs em manutenção.
- RSC – Serviço X não deve ter duas ou mais VMs hospedadas na mesma PM.
- RSC – O grupo de VMs Y deve ser interconectado a 10Gbps.
- RSC – O grupo de PMs Z deve manter 10% ou mais de memória disponível.

Esta divisão é adotada pelos seguintes motivos: (i) ambas necessitam de argumentos diferentes para processarem suas respostas, e.g. $< vm, pm >$ no caso das RLC e um cenário, no caso das RSC; e (ii) permitir a sua utilização independente, i.e. poder usá-las em diferentes etapas dos métodos.

Um exemplo mais específico sobre a utilização das regras é o caso da sumarização das RLC. Trata-se de uma estratégia de otimização das regras e uma forma de previsão do tempo de execução dos métodos. Devido a independência de contextos, as RLC podem ser pré-processadas formando uma única matriz $[x, y]$ com todas as alocações

por elas permitidas. Este processo possui um tempo linear de $O(x.y)$, sendo x e y o número de PMs e VMs do ambiente. A Tabela 3 traz um exemplo de uma matriz de sumarização com 4 VMs e 3 PMs.

	PM_1	PM_2	PM_3
VM_1		✓	✓
VM_2	✓	✓	
VM_3		✓	✓
VM_4	✓	✓	✓

Tabela 3 – Exemplo de sumarização das RLC em uma única Matriz

4.1.1.2 Qualificadores: Aferindo as Alocações

Qualificadores são funções que quantificam a qualidade de uma ou mais alocações de um cenário. Um ou mais objetivos podem ser representados por um ou mais qualificadores. Estas mensurações permitem a comparação e, consequentemente, a seleção de cenários. Eles demonstram a sua corroboração para com uma alocação, atribuindo altos valores em seus aferimentos. Caso contrário, valores próximos de zero são sinal de desaprovação das alocações. A tabela 4 ilustra um caso em que quatro alocações são avaliadas por três Qualificadores hipotéticos, focados em: Consumo de Energia, Rede e Carga ($Q_{Energia}$, Q_{Rede}, Q_{Carga}).

Além dos Objetivos, os qualificadores também permitem representar políticas e boas práticas do ambiente, e.g. distribuição homogênea de VMs afins, políticas de redundância e segurança.

alocações	$Q_{Energia}$	Q_{Rede}	Q_{Carga}
VM_1 na PM_1	0.1	2.0	0.1
VM_1 na PM_2	2.0	1.5	1.0
VM_1 na PM_3	0.1	2.0	2.0
VM_1 na PM_4	1.0	1.5	2.0

Tabela 4 – Exemplo de alocações sendo avaliadas por Qualificadores. Valores altos recomendam a adoção da alocação e valores próximos a zero desencorajam tais alocações.

4.1.1.3 Prioridades: Mitigando Impasses

Prioridades são aplicadas na priorização dos Qualificadores, aferindo pesos aos mesmos. Também podem ser vistas com uma forma de representação das preferências da Nuvem, visto que cada ambiente possui pesos diferentes para objetivos diferentes. Tecnicamente são utilizadas como uma forma de resolução de conflitos durante os processos de filtragem e seleção de cenários. Como exemplo, a Tabela 4 apresenta um impasse entre dois cenários, visto que um adota a alocação "VM_1 na PM_2" e o outro adota "VM_1 na PM_4". Neste caso, o peso de cada Qualificador auxilia na tomada de decisão, diferenciando as avaliações de $Q_{Energia}$ e Q_{Carga}, levando a adoção de uma das alocação.

4.1.1.4 Custos: Aferindo a implementação de cenários

Custos são funções que quantificam a transição entre dois cenários, i.e. o custo de implementação de um cenário final, partindo de um cenário inicial. Seus valores não possuem unidades, podendo representar desde a quantidade de migrações necessárias até o volume de recursos demandados para se alcançar o cenário alvo. Eles são utilizados durante os processos de filtragem e seleção de cenários, e.g. vetando resultados infactíveis e resolvendo impasses entre MOs, como explica o exemplo a seguir.

Tomando como exemplo o conflito apresentado na Tabela 4, entre as alocações "VM_1 na PM_2" e "VM_1 na PM_4", se a utilização das Prioridades não resolver o conflito, este pode ser sanado selecionando a solução que provê o menor Custo de implementação.

Por fim, diferentemente dos qualificadores, cada custo tem um valor máximo pré-estipulado, que representa o investimento limite que os administradores da Nuvem estão dispostos a investir na melhoria da Nuvem. Outro ponto a ser considerado é o fato dos custos serem utilizados em (múltiplas) fases distintas dos métodos de otimização, como por exemplo na filtragem de cenários (pelas Regras) e na seleção de soluções (pelo Custo-Benefício). Por estes motivos, apesar de similares, estes são classificados de forma separada dos qualificadores.

4.1.2 Formalização

A fim de evitar ambiguidades e especificar precisamente os elementos que envolvem as Nuvens guiadas por MOs, esta seção define formalmente o modelo previamente descrito. Todos os símbolos presentes na definição e seus significados são listados na Tabela 5.

Considere y o número de VMs e x o número de PMs na Nuvem, em que V é um conjunto com y VMs $v \mid v \in V$. H é um conjunto com x PMs $h \mid h \in H$, como apresentado em (4.1). Uma alocação é uma relação entre dois elementos de V e H representados por um par $(v_c, h_d) \mid v \in V \wedge h \in H$, e um cenário C é um conjunto com y alocações, como apresentado em (4.2).

$$V = \{v_1, ..., v_y\} \wedge H = \{h_1, ..., h_x\} \tag{4.1}$$

$$C = \{\{v_1, h_1\}_1, ..., \{v_c, h_d\}_y\} \tag{4.2}$$

Símbolo	*Significado*
y	Número de VMs
x	Número de PMs
v	Identificador de VM
h	Identificador de PM
V	Conjunto com y VMs
H	Conjunto com x PMs
a, b, c e d	Indexadores quaisquer
(V_a, H_b)	Uma alocação
C	Conjunto com y alocações
f_{rlc}	Função de Regra Livres de Contexto (RLC)
f_{rsc}	Função de Regra Sensíveis ao Contexto (RSC)
Rl	Conjunto de Funções de RLCs
Rs	Conjunto de Funções de RSCs
f_q	Função de Qualificação
z	Número de Qualificadores
Q	Conjunto com z Qualificadores
f_p	Função de que retorna o Peso de um Qualificador
I	Conjunto de funções de Custo
M	Conjunto de Funções de Custo Máximo
f_i	Função de Custo
f_m	Função de Custo Máximo
$\mathbb{Q}$	Números Racionais

Tabela 5 – Símbolos utilizados na descrição do modelo de Nuvem

Formalmente, as regras RLC e RSC são representadas como dois conjuntos separados de funções, Rl e Rs, as quais contém uma quantidade l e s de funções booleanas f_{rlc} e f_{rsc}, respectivamente. Dada uma alocação (v_c, h_d), uma função f_{rlc} retornará 1 sinalizando como permitida e retornará 0 caso contrário, como mostrado em (4.3). Por sua vez, dado um cenário C, uma função f_{rsc} retornará 1 caso o mesmo seja válido e 0 caso contrário, como mostramos em (4.4)

$$\begin{aligned} Rl &= \{f_{rlc_1}, ..., f_{rlc_l}\} \\ \forall f_{rlc} \in Rl &: f_{rlc_d}(\{v_c, h_d\}, \{0 \vee 1\}) \end{aligned} \tag{4.3}$$

$$\begin{aligned} Rs &= \{f_{rsc_1}, ..., f_{rsc_s}\} \\ \forall f_{rsc} \in Rs &: f_{rsc_d}(C, \{0 \vee 1\}) \end{aligned} \tag{4.4}$$

Os Qualificadores, por sua vez, são apresentados com um conjunto Q com z funções f_q. Dado um cenário C, uma função f_q retornará um vetor com y aferimentos a_y, um para cada alocação de C, como apresentado em (4.5). Estes aferimentos são mapeados entre zero e dois, excluindo zero, i.e. $]0,2]$. O zero não é incluído pois um qualificador com valor zero seria considerado um RSC, que proíbe um cenário. A adoção do intervalo $]0,2]$ está relacionada a utilização do operador de multiplicação no processo de sumarização dos múltiplos qualificadores. Desta maneira, um qualificador pode aumentar, diminuir ou não interferir na avaliação de uma alocação adotando valores menores que 1, maior que 1 ou exatamente igual a 1, respectivamente. Quando um qualificador afere o valor 1, este não interfere nas outras Avaliações.

$$\begin{aligned} Q &= \{f_{q_1}, ..., f_{q_z}\} \\ \forall f_q \in Q &: f_q(C, \{a_1, ..., a_y\}) : a \in]0,2] \end{aligned} \tag{4.5}$$

Para priorizar os qualificadores existe uma função f_p que, dada uma função f_q de qualificação, retorna seu peso u, um número racional maior ou igual a zero, como apresentado em (4.6). Desta maneira, o valor zero anula a utilização de um qualificador, i.e. ele não é considerado nas avaliações.

$$f_p(f_q, u) : u \in \mathbb{Q} \wedge u >= 0 \tag{4.6}$$

Finalmente, I refere-se ao conjunto de funções f_i que quantificam o custo para se implementar um cenário C. Dados dois cenários C_{origem} e $C_{destino}$, f_i retorna o custo de transição entre eles, como descrito em (4.7). O valor retornado é um número inteiro maior que zero.

$$f_i(\{C_{origem}, C_{destino}\}, i) : i \in \mathbb{Q} \wedge i > 0 \tag{4.7}$$

Relacionado ao Valor Máximo de cada custo, M refere-se ao conjunto de funções f_m que, dada uma função f_i, retorna o Custo Máximo pré-definido para f_i, como apresentado em (4.8).

$$f_m(f_i, m) : m \in \mathbb{Q} \wedge m > 0 \tag{4.8}$$

4.2 OS MÉTODOS DO ORDER@CLOUD

Nesta seção apresentaremos os métodos que constituem o framework Order@Cloud. Primeiramente analisamos o problema, abordando as perguntas *Qual?*, *Onde?* e *Quando?*. Em seguida, apresentamos o método principal, o qual organiza as VMs na Nuvem. Depois explicamos o método de Provisionamento, que indica onde novas VMs devem ser instanciadas. Por fim, apresentamos o método de Adaptação da Nuvem, que modifica as alocações para tornar o cenário válido perante as regras do ambiente. Esta ordem é adotada pois cada método depende de seu antecessor.

4.2.1 Análise do Problema e Base da Proposta

Nesta seção apresentamos as opções e escolhas que fizemos em nossa proposta. Iniciamos definindo o objetivo dos métodos, então desenvolvemos o conceito (e cálculo) do *Benefício* e, em seguida, partimos para as perguntas *Qual* e *Onde*.

Como exposto na seção 2.3, a abordagem do problema como *Puramente Múltiplos-Objetivos*, juntamente com o conceito de Dominância, podem estagnar os métodos em ambientes com muitos objetivos. Por este motivo, como não desejamos este fator limitante, resolvemos adotar uma abordagem de *Múltiplos-Objetivos como Mono-Objetivo* para sumarizar as avaliações de cada alocação da Nuvem. Desta maneira a qualidade de cada alocação da Nuvem é representada por um valor. Assim, o objetivo do nosso método é maximizar as avaliações das alocações e minimizar o custo de implementação do novo cenário.

Apesar das avaliações focarem nas alocações, elas também nos permitem calcular o benefício do cenário como um todo. O benefício é um valor que representa a melhoria das avaliações entre um cenário inicial e um cenário alvo, i.e. $avaliacao(C_{alvo}) - avaliacao(C_{origem})$. Para calculá-lo, baseamos-nos nos indicadores listados por Zitzler et al., em [95], utilizados para avaliar soluções genéricas. Em nossa proposta, a avaliação do cenário é obtida pela equação 4.9. Nela, a avaliação do cenário é definida pelo somatório das avaliações das alocações ($\sum_{c=1}^{y}$).

Estas, por sua vez, são dadas pelo produto dos qualificadores ($\prod_{d=1}^{z}$) elevados pelos seus devidos pesos ($f_{q_{d,c}}^{f_{p_d}}$). O produto e a exponenciação foram adotados por possibilitarem, de uma forma simples, a interferência entre os aferimentos dos qualificadores, i.e. um aferimento

baixo diminuirá toda a avaliação de uma alocação, e caso este qualificador possua um peso alto, a avaliação diminuirá mais ainda. Não menos importante, caso o peso de um Qualificador seja zero, seu aferimento passa a ser 1, o que não interfere no produto das avaliações.

$$avaliacao(C) = \sum_{c=1}^{y} \prod_{d=1}^{z} f_{q_{d,c}}^{f_{p_d}} \tag{4.9}$$

Assim como na abordagem de *Extração* e *Relocação*, de Beloglazov et al. [44], focamos nossa proposta nas perguntas *Qual* VM migrar e para *Onde* migrá-la.

Sobre a questão de *Qual* VM migrar, considerando que o próprio problema induz a priorizar as VMs mal alocadas na Nuvem, temos duas opções: (i) selecionar a pior VM[1] para migração ou (ii) tentar migrar todas as outras VMs verificando qual oferece a maior melhoria para a pior VM. No entanto, considerando x e y como o número de PMs e VMs, a opção (i) proporciona uma complexidade de $O(x)$ e a opção (ii) $O(xy)$ cenários possíveis, em apenas um procedimento de avaliação. Logo, com avaliações sucessivas, a quantidade de cenários gerados por (ii) cresceria exponencialmente e tornaria impraticável o processamento destes. Então, resolvemos adotar a opção (i).

Sobre *Onde* migrar a pior VM, a princípio existem $\#PMs - 1$ possibilidades de migração. No entanto, as regras do ambiente, juntamente com as restrições de custo máximo, reduzem este número. Dada a restrição do problema, que proíbe a degradação da Nuvem, só podemos adotar cenários cujo Benefício é maior ou igual a zero. No entanto, a adoção de cenários com benefícios iguais a zero causa laços na busca (*loops*), o que exige a aplicação de técnicas de rastreamento reversos (*back-tracking*) a fim de evitar tais situações. Mesmo assim, a degradação da Nuvem é imperceptível ao analisarmos o benefício final, pois algumas avaliações podem aumentar e outra, diminuir.

Visando mitigar este problema, resolvemos adotar um filtro de Dominância que seleciona os cenários pelas avaliações das alocações. Este filtro garante que nenhuma alocação (dos cenários selecionados) seja degradada e pelo menos uma de suas alocações seja melhorada. Assim, todo cenário aceito pelas regras e classificado como não-dominado é selecionado como possível.

Concluindo, ainda existe a possibilidade da migração da pior

[1]Definiu-se na Seção 2.3 (p.36) que o termo *pior VM* se refere a VM com a menor qualificação.

VM não gerar cenários não-dominados, limitando o avanço da busca por melhores cenários. Assim, buscamos estratégias de busca ja conhecidas, como A-Estrela (A*), Aprofundamento Interativo (IDA*) etc. No entanto, estas abordagens são aconselhadas para problemas de busca de caminhos (*pathfinding*), o que difere do problema de VMP. Apesar de também almejarem reduzir o custo do resultado (menor caminho), em *pathfinding* existe: (i) um estado final almejado e (ii) o caminho tomado é indiferente ao resultado, com o tanto que seja com o menor custo. Ja em VMP, não existem um ponto único de chegada (estado final), pois podem existir múltiplos cenários ótimos ou o cenário ótimo pode ser inalcançável (não implementável), adotando-se um cenário semi-ótimo. Não menos importante, em VMP também pode existir múltiplos caminhos de menor custo, porém existe uma prioridade entre eles, a qual é quantificada por via do benefício das piores VMs. Desta maneira, torna-se inviável considerar propostas de *pathfinding* para sanar o problema de VMP. Outro sim, é necessário adotar uma estratégia que evite a estagnação da busca. Liang et al. [18] propõem utilizar Busca com Variação de Localidades (Variable Neighbourhood Search – VNS) para melhorar a qualidade de seus resultados e fazem isto variando o cenário inicial da busca. Desta maneira, adotamos uma estratégia baseada na VNS para tratar os casos de estagnação, explicada na seção seguinte.

4.2.2 Método de Organização

Iniciamos a seção apresentando uma visão geral do funcionamento do método, a fim de facilitar o entendimento deste. Em seguida, o método é descrito passo-a-passo, detalhando cada fase. Por fim, apresentamos um caso de uso, ilustrando o seu funcionamento.

4.2.2.1 Visão geral

O método de *organização* do Order@Cloud visa organizar as VMs de forma a maximizar a qualidade das alocações dando prioridade para as VMs com qualidade mais baixa. Este é um processo recursivo que recebe o estado atual da Nuvem (Cenário atual) e busca melhores cenários por migrações que aumentam as avaliações das alocações.

Primeiro, ele usa os Qualificadores para avaliar o Cenário e selecionar a pior VM. Em seguida, gera todos os Cenários possíveis para

migrar esta VM, criando um conjunto com no máximo $\#PM - 1$ novos cenários. Então, as regras (RFC e RSC) são aplicadas e os Cenários "dominados" estão excluídos, reduzindo assim consideravelmente este conjunto. Em seguida, para cada cenário do conjunto, outra instância do método de organização é iniciada enviando um dos cenários (recursão). Após as recursões retornarem seus melhores cenários (i.e. os melhores cenários encontrados de cada recursão), o método calcula e seleciona o cenário com o melhor Custo-Benefício. Em seguida, uma outra recursão inicia com o melhor cenário selecionado, ignorando a pior VM deste cenário e, assim, variando a localidade da busca. No final, o cenário com o melhor Custo-Benefício é selecionado para implementação.

4.2.2.2 Passo a Passo

Esta seção explica em detalhes cada passo do método de organização. A Figura 15 e o Algoritmo 1 ilustram o mesmo.

Algorithm 1: OrderCloud - Método de Organização

```
    Input: cenarioInicial // Cenário Inicial
    Input: ignoreVms // Conjunto de VMs a serem ignoradas
    Input: éRecursaoPrincipal // booleano, começa como Verdadeiro
    Data: pareto // Conjunto de VMs auxiliar
    Output: novo // Cenário Inicial
 1  if todasAsVmsForamExploradas(ignoreVms) then return cenarioInicial;
 2  avaliacaoInicial = aplicarQualificadores( cenarioInicial ) //Descrito no
      Algoritmo 2;
 3  piorVM = selecionarPiorVm( avaliacaoInicial , ignoreVms );
 4  cenarios = gerarCenarios( cenarioInicial , piorVM );
 5  foreach cenario ∈ cenarios do
 6  |   avaliacaoCenario = applyQualifiers( cenario );
 7  |   if éNaoDominado( avaliacaoCenario , avaliacaoInicialval ) then
 8  |   |   novoIgnoreVms = ignoreVms;
 9  |   |   novoIgnoreVms.adicionar( piorVM );
10  |   |   melhorCenario = OrderCloud(cenario , novoIgnoreVms , falso);
11  |   |   pareto.adicionar( melhorCenario );
12  pareto.adicionar( cenarioInicial );
13  melhorCenario = selecioneCenarioComMelhorCustoBeneficio( pareto );
14  if éRecursaoPrincipal then
15  |   melhorCenarioAvaliacao = applyQualifiers( melhorCenario );
16  |   piorVM = selecionarPiorVm( melhorCenarioAvaliacao );
17  |   ignoreVms.adicionar( piorVM );
18  |   melhorCenario = OrderCloud(cvmp,ignoreVms,false);
19  else
20  |   return
21  melhorCenario
```

Início: O método recebe como entrada: (i) o cenário atual, um

vetor com os posicionamentos reais da Nuvem, (ii) um conjunto vazio de VM para controlar as recursões (*ignoreVMs*) e (iii) um indicador booliano para sinalizar as recurções (*éRecursaoPrincipal*), começando como *verdadeiro*.

Avaliação do Cenário: O primeiro passo utiliza os qualificadores para avaliar o cenário recebido e gerar uma classificação das VMs de acordo com as suas avaliações. Cada Qualificador avalia todas as alocações da Nuvem, resultando em uma matriz de avaliações $A_{z,y}$, na qual z é o número de qualificadores e y é o número de VMs. Em seguida, cada avaliação é elevada ao peso de seu respectivo Qualificador. Assim, o conjunto é reduzido a um vector A_y multiplicando todas as avaliações de cada máquina virtual. A redução é representada em (4.10), similar a apresentada em (4.9). No Algoritmo 1 este passo é mostrado na linha 2 e a função *aplicarQualificadores* é representada no Algoritmo 2.

$$A_y \leftarrow \prod_{n=1}^{z} A_{n,y}^{U_n} \tag{4.10}$$

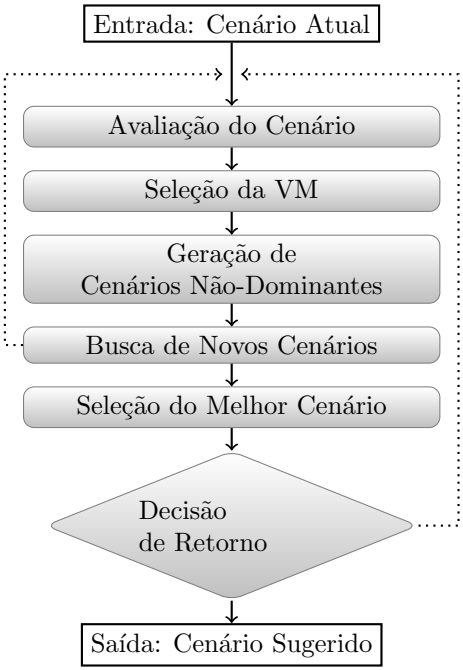

Figura 15 – Fases do Método de Organização do Order@Cloud

Algorithm 2: aplicarQualificadores

```
   Input: cenario // Cenario a ser avaliado
   Data: qualificadores // lista de qualificadores da Nuvem
   Data: matriz // matrix com o numero de VMs preenchida com valores 1
   Data: avaliacaoCenarioInicial // soma das avaliacao do cenário inicial
   Output: avaliacoes // integer matrix with utility value
 1 avaliacoes = [];
 2 foreach qualificador ∈ qualificadores do
 3     avaliacaoNormalizada = qualificador.avalie(cenario);
 4     foreach avaliacao ∈ avaliacaoNormalizada do
 5         avaliacaoComPeso = avaliacao^qualificador.peso;
 6         matriz[avaliacao.vmId] *= avaliacaoComPeso;
 7 beneficio = soma( matriz ) - avaliacaoCenarioInicial;
 8 custo = calcularCusto( beneficio );
 9 cb = beneficio/custo;
10 matriz.definirCustoBeneficio( cb );
11 return matriz
```

Ressaltamos que antes de começar a analisar o cenário é verificado se todas as VMs ja foram exploradas pela busca. Caso isto ocorra, ele retorna o cenário atual. Esta decisão é representa pela linha 1 do Algoritmo 1.

Seleção da VM: Esta etapa é baseada na estratégia de *Extração* [44], e nossa heurística seleciona a VM que: (i) possui a menor avaliação do cenário e (ii) não está presente no conjunto *ignoreVms*. No Algoritmo 1 este passo é mostrado na linha 3.

Geração de Cenários Não-Dominados: O objetivo deste passo é identificar para onde a VM selecionada pode ser migrada. Desta maneira, um novo cenário é gerado para cada possibilidade de alocação, ou seja, no máximo $x-1$ cenários. Durante esta etapa, os seguintes filtros são aplicados: (i) as Regras (RLCs e RSCs, respectivamente), (ii) o filtro de custo máximo, verificando se o custo do cenário é inferior aos limites estipulados, e (iii) o filtro de Dominância, que exclui cenários dominados. Esta etapa é representada pelas linhas 4-7 do Algoritmo 1.

Busca por Novos Cenários: Este passo, baseado no método Busca em Localidades [18], inicia uma nova recursão que explora cada cenário Não-Dominado encontrado. No entanto, para evitar laços (*loops*), também são enviadas as seguintes informações ao método: (i) sinal booliano *falso*, o qual indica que esta recursão não é a principal, e (ii) uma cópia do conjunto *ignoreVMs* – *newIgnoreVMs* – no qual é adicionada a pior VM, selecionada no primeiro passo. As linhas 8-11 de Algoritmo 1 mostram esta etapa.

O conjunto *novoIgnoreVMs* é um recurso utilizado para evitar o fim prematuro da busca e a visita duplicada de cenários. Os fins

prematuros (*deadlocks*) acontecem quando uma mesma VMs permanece como pior VM sucessivamente, impossibilitando o avanço da busca.

Seleção do Melhor Cenário: O objetivo deste passo é selecionar o melhor cenário, dentre o cenário atual e os resultados das recurções. Para isto, os cenários são ordenados pelo seus Custo-Benefícios e aquele com o maior valor é selecionado. Note que, visto que o custo é maior que zero, para fins de cálculo avaliamos a relação Benefício-Custo. Este passo é representado pelas linhas 12-13 do Algoritmo 1.

Para alcançar o equilíbrio entre o custo de implementação e a avaliação da qualificação, propusemos utilizar o método de Custo-Benefício, em que o benefício é dado pela Equação (4.9)[2] e o Custo é provido pela função de Custo previsto pelo modelo da Nuvem, apresentado na Seção 4.1.

Decisão de Retorno: Finalmente, este passo decide se deve continuar a busca ou finalizá-la. Caso não seja a recursão principal, o método prontamente retorna o cenário selecionado. Caso contrário, baseado na ideia da *Variação de Localidade*, o espaço de busca é alterado e uma nova recursão é iniciada.

A nova recursão é iniciada com um conjunto de VMs a serem ignoradas – *ignoreVms* – não vazio, pois a pior VM do cenário é adicionada neste conjunto. O resultado desta recursão é a saída do método. Este passo é representado pelas linhas 14-21 do Algoritmo 1.

4.2.2.3 Caso de Uso

Nesta seção apresentamos um caso de uso do método, a fim de facilitar o entendimento deste. Considere a Figura 16, a qual retrata um exemplo de um cenário com **3 VMs e 3 PMs**.

O método inicia com o cenário atual (ou cenário zero, o qual abreviamos para S0) e seleciona a pior VM, i.e. VM3 na PM3. Deste cenário, gera-se dois novos Cenários Não-Dominados (CND), S1 e S2. O método tenta então melhorar S1 o qual, depois de ignorar a VM3, seleciona VM1 como a pior VM. O único cenário CND identificado é S3, o qual não consegue encontrar uma alocação melhor para VM2. Assim, o custo-benefício de todos os CND são comparados e o maior é selecionado, no caso, S3. Deve ser ressaltado que, devido ao conjunto de VMs a serem ignoradas, apesar de VM2 ter sido considerada a pior alocação de S3, quando este conjunto é esvaziado e uma nova VM é

[2] Apresentando na pg.61

selecionada, VM3 é escolhida como a pior VM.

A próxima recursão inicia com S3 e VM3 no conjunto de VMs a serem ignoradas. VM1 então é selecionada e apenas S4 é identificado como um CND. Visto que o Custo-Benefício de S4 é maior que S3, este é selecionado para a próxima recursão, e VM2 é incluída no conjunto de VMs a serem ignorada.

Finalmente, em S4, ignorando VM3 e VM2, VM1 é selecionada para migração, a qual gera somente S5. Entre os dois cenários (S4 e S5), S5 é escolhido pelo seu Custo-Benefício superior. Encerrando a recursão, S5 é retornado para S3, depois para S0 e por fim finaliza o método.

4.2.3 Método de Provisionamento

Nesta seção descrevemos o método de Provisionamento do Order@Cloud. O objetivo dele é escolher a melhor alocação para uma VM que necessite ser iniciada. Neste caso, a melhor alocação também é aquela que proporciona o maior benefício. Este método, por sua vez, se utiliza do método de Organização para efetuar a sua tarefa.

Primeiramente, verificamos se esta VM ja havia sido iniciada em alguma PM, checando a variável *pmTeorica*. Caso negativo, efetuamos duas ações: (i) adotamos uma alocação aleatória para esta VM, hospedando-a na primeira PM permitida, e (ii) desativamos a função de custo, pois como é uma alocação teórica, não haverá custo de implementação. Caso não seja possível encontrar uma *pmTeorica*, o método retorna NULO. Estes procedimentos são representados nas linhas 1-4 do Algoritmo 3.

Em seguida, adicionamos todas as VMs da Nuvem em um conjunto de VMs a serem ignoradas (com exceção da nova VM). Depois, uma recursão de Organização é iniciada enviando o cenário teórico e o conjunto de VMs a serem ignoradas — linhas 5-7 do Algoritmo 3. O resultado da Organização contém alocação da nova VM.

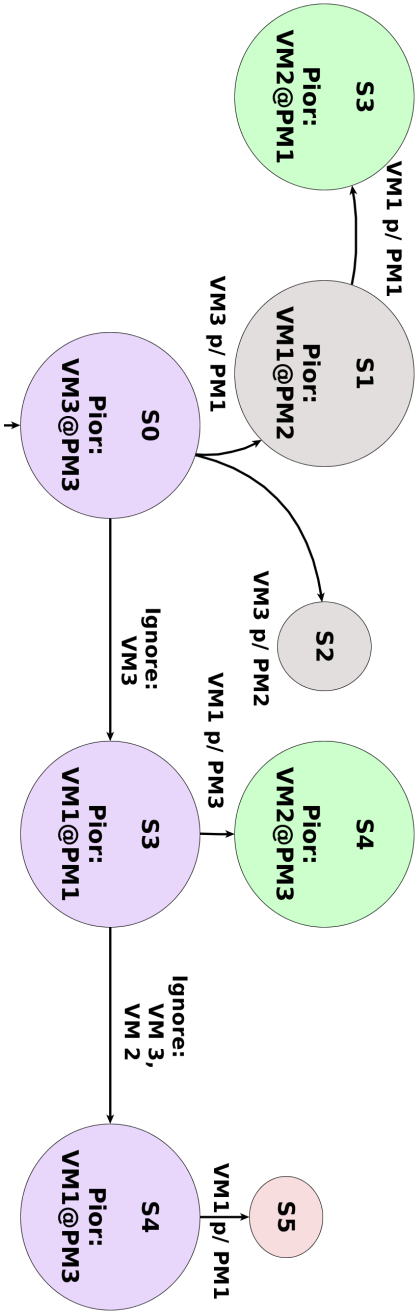

Figura 16 – Caso de uso do método de Organização

Algorithm 3: Provisionamento de VMs

```
   Input: cenario — Cenario atual da Nuvem
   Input: vm — VM para hospedar
   Input: pmTeorica — PM onde a VM estava hospedada anteriormente
          (Opcional)
   Output: pm — PM selecionada para hospedagem ou NULO
1  if pmTeorica == NULO then
2      pmTeorica = cenario.selecionarPmTeoricaParaHospedar(vm);
3      if pmTeorica == falso then return NULO;
4      cenario.desabilitarAvaliacaoDeCusto();
5  ignoreVms = cenario.retornaTodasVms();
6  cenario.adicionarAlocacao(vm,pmTeorica);
7  cenario = organize( cenario , ignoreVms , falso);
8  pm = cenario.retornaPmDaVm(vm);
9  return pm
```

4.2.4 Método de Adaptação

Nesta seção descrevemos o método de adaptação de cenários, o qual tem como objetivo aferir se um cenário é valido, dadas as regras do ambiente, e adaptá-lo caso não seja. Ele é necessário após a adição de novas regras, visto que estas podem tornar o cenário inválido.

Primeiramente é construído um conjunto de todas as VMs que não estão de acordo com: (i) as RLC e (ii) as RSC, respectivamente. Para as RLC, é feita uma varredura verificando se cada alocação é válida. Após, todas as VMs identificadas são removidas do cenário. Para as RSC, é construído um novo cenário e, para cada VM adicionada, são testadas as RSC. Caso uma VM invalide o novo cenário, esta é removida e o processo continua, como pode ser visto nas linhas 1-4 do Algoritmo 4.

A próxima etapa é recolocar estas VMs no cenário. Desta forma, para cada VM inválida, um processo de Provisionamento é criado. Neste, enviamos a VM e a PM onde estava originalmente, representando uma alocação teórica. Assim que o processo de Provisionamento retornar a PM, verificamos se esta é igual a original (inválida); caso positivo, adicionamos em um conjunto de *VMs não alocadas*, caso contrário, adicionamos a alocação no cenário. Estes procedimentos podem ser verificado nas linhas 5-10 do Algoritmo 4.

Finalmente, o método retorna o cenário e o conjunto de VMs que não foram alocadas.

Algorithm 4: OrderCloud - Adaptação de Cenários

```
   Input: cenario // Cenario a ser conformado
   Data: vmsNaoAlocadas //Conjunto vazio de VMs
   Output: cenario // Cenario Valido
 1 alocacoesInvalidas ← selecionarVmsInvalidasPelasRlc( cenario );
 2 cenario.removerAlocacoes( alocacoesInvalidas );
 3 alocacoesInvalidas ← selecionarVmsInvalidasPelasRsc(cenario);
 4 cenario.removerAlocacoes(alocacoesInvalidas);
 5 foreach (vm,pm) ∈ alocacoesInvalidas do
 6     novaPm = provisionamento(cenario,vm,pm);
 7     if novaPm == pm then
 8         vmsNaoAlocadas ← vm;
 9     else
10         cenario.adicionarAlocacao(vm,pm);
11 return scenario, unAllocatedVms;
```

4.3 DISCUSSÃO DA PROPOSTA

Nesta seção discutiremos características das propostas apresentadas nas Seções 4.1 e 4.2.

Começando pelo método de Organização de VMs, este se baseia na premissa de que o cenário inicial é válido, ou seja, está aderente a todas as Regras do ambiente. Esta premissa é necessária para que o método seja executado corretamente, pois a adoção de um cenário ilegal interfere no processo de geração de novos cenários. Este caso pode ocorrer quando novas regras são adicionadas ou reabilitadas no ambiente. Para mitigar esta situação é necessário executar o método de *Adaptação* antes deste, o aferirá a validade do mesmo e tentará reorganizar as VMs em uma configuração válida.

A segunda premissa na qual o método se baseia é do Custo de Implementação ser incremental. Podemos afirmar que a restrição de *Custo Máximo* é um dos principais catalisadores da convergência do método, isto é, ele elimina muitas ramificações durante a exploração, diminuindo consideravelmente o número de cenários explorados, acelerando assim a busca. No entanto, o uso desta limitação se baseia na premissa de que o custo não diminui, i.e. o custo sempre aumenta de um cenário para os seus sucessores. Isto é importante pois, de outra maneira, ramificações válidas poderiam ser descartadas durante a busca. Logo, deve-se ficar atento a isto durante a implementação de novas funções de Custo.

Ainda sobre o corte de ramificações válidas, apesar da busca por soluções não-dominadas assegurar a convergência do método, o conceito de Dominância não permite a adoção de cenários cujo benefício é zero. Teoricamente, as restrições do problema permitem tais cenários, pois não estamos degradando a Nuvem. Adotar este conceito impossibilita atingir eventuais cenários alcançáveis somente por via de cenários neutros, i.e. com benefício zero. No entanto, isto simplifica a busca e evita a criação de laços (loops). Por este motivo, a Seção 5.3, do Capítulo relacionado aos testes trata especialmente sobre a *Opticidade* alcançada pelo nossa proposta em relação a propostas menos restritivas.

Ainda sobre a busca, durante a exploração de cenários não-dominados, utilizamos como uma estratégia de *Variabilidade* um conjuntos de VMs a serem ignoradas. Este conjunto é utilizado principalmente para fins prematuros (*Deadlocks*) e visitas duplicadas ao mesmo cenário. Um fim prematuro ocorre quando uma VM é recorrentemente selecionada como a pior VM, isto é, o crescimento de sua avaliação depende da realocação de outras VMs. Desta maneira, a utilização desta estratégia garante que, nas próximas recursões, a atual pior VM

será ignorada durante o passo de seleção, e, desta maneira, a busca explorará outros cenários.

Concluindo sobre o método de organização, gostaríamos de clarificar as adaptações feitas nos conceitos de Busca com Localidades Variáveis (Variable Neighbourhood Search – VNS) e Dominância de Pareto. A estratégia VNS sugere construir uma Fronteira de Pareto, explorando resultados não dominados e em diferentes bairros. No entanto, em vez de usar cenários aleatórios para variar a Localidade inicial da busca, como proposto por Liang et al. [18], usamos um método baseado na classificação de VMs para identificar VMs e diversificar as localidades de pesquisa.

Da mesma forma, a definição original de Dominância do Pareto analisa as avaliações dos objetivos, separadamente. Nós, pelo contrário, analisamos as avaliações das alocações para evitar a estagnação do método, devido a quantidade de objetivos. A avaliação da alocação é uma combinação de todos os aferimentos dos qualificadores e seus pesos, como descrito na equação (4.10), na pg. 64.

Sobre o método proposto para *Provisionamento* de novas VMs, este possui um caso de exceção. Durante sua fase inicial, o método depende da existência de recursos disponíveis para adotar uma alocação virtual para a nova VM. Porém, consideramos improvável que ocorra, visto que normalmente é mantido um nível de recursos ociosos nas PMs, o que mitiga a questão.

Sobre o método proposto para *Adaptação*, dada a variedade de regras e restrições que podem ser implementadas, este não consegue garantir a relocação de todas as VMs, a fim de tornar um cenário válido. Cogitamos a possibilidade de utilizar heurísticas para a reinserção das VMs no cenário (e.g. ordenar por tamanho dos recurso), no entanto, mesmo assim, não teríamos certeza se esta prioridade representaria o interesse da administração da Nuvem. Neste caso, seria necessário a intervenção humana.

Sobre a implementação de novos elementos no framework, durante a avaliação de cenários, alguns casos demandam a utilização de *Qualificadores* baseados em dados de monitoração. No entanto, como o algoritmo de busca calcula muitos cenários por segundo, durante as avaliações, torna-se inviável depender de dados em tempo real. Por esse motivo, recomenda-se pré-carregar os dados necessários e/ou utilizar funções de aproximação nas eliminatórias, se necessário.

Por fim, a complexidade do método é definida por suas regras e limites de investimento. Podemos definir seu pior caso como $O(y^{d_{max}})$, onde y é o número de PMs no ambiente e d_{max} é a profundidade máxima da busca. No melhor cenário, em que existe apenas uma solução não-dominada para explorar, a complexidade é $O(y)$. Porém, na maioria dos casos, a complexidade média do método é $O(a^{d_{avg}})$, onde a é a taxa média de alocação do ambiente, que pode ser calculado com a tabela de sumarização das Regras Livres de Contexto (Seção 4.1), e d_{avg} é a profundidade média da busca.

5 VALIDAÇÃO DOS MÉTODOS

Nesta seção, apresentamos os experimentos realizados, em um cenário real, para avaliar: (i) o desempenho dos módulos individuais do método de organização, evidenciando as vantagens de nossas escolhas; (ii) a otimicidade alcançada por nosso método, em comparação a outras abordagens; e (iii) a sua capacidade de escalabilidade/expansão.

5.1 O AMBIENTE DOS EXPERIMENTOS

Nos experimentos utilizamos dados reais de alocação provenientes do centro de dados da Universidade Federal de Santa Catarina. Decidimos não empregar técnicas de simulação primeiramente porque visamos a análise da proposta em um ambiente de Nuvem real e em produção. Em segundo, porque anulamos a necessidade de gerar dados artificiais, como cenários e configurares de VMs e PMs, para executar as simulações. Assim, a Tabela 6 lista um sumário dos elementos que compõem o ambiente real utilizado.

Elemento	Dados
Plataforma (*Hypervisor*)	VMWare 5.5
Máquinas Virtuais	607
Máquinas Físicas	20
Unidades de Armazenamentos (*Storage Pools*)	99
Redes de Comunicação (VLANs)	102

Tabela 6 – Dados do Ambiente Real Utilizado

Os testes foram executados em um ambiente com o sistema operacional Ubuntu 14.04, com um processador *Intel T9400* de 2.53GHz e 4GB de memória RAM. Este processou, em média, 250 cenários por segundo. O protótipo do framework e seus testes foram implementados em PHP 5.5, utilizando o framework de testes PHPUnit 4.2. Esta implementação, os testes descritos e mais informações são de código aberto e estão disponíveis no site do projeto.

Os testes foram executados em um cenário com um custo máximo de 20 migrações e variamos o número de VMs entre 350 e 600. A função de custo implementada contabiliza a quantidade de migrações necessárias para se alcançar o dado cenário.

Devido as especificidades dos ambientes e as variáveis utilizadas

nos trabalhos relacionados, tornou-se inviável a comparação integral destes com os métodos da nossa proposta. Desta forma, analisamos suas abordagens, as adaptamos para utilizarem os mesmos elementos do nosso modelo e assim as comparamos com o nosso método.

Em relação aos elementos implementados para os experimentos, apesar do modelo e do método suportarem a avaliação de SLA e dados dinâmicos, selecionamos que Regras e Qualificadores cuja avaliação se baseia em variáveis estáticas, como quantidade de recursos e localização das VMs na Nuvem. Esta decisão foi tomada a fim de permitir a reprodutibilidade dos testes com diferentes abordagens, partindo sempre de mesmo estado inicial da Nuvem.

Assim, foram aplicadas as seguintes regras:

- Haver disponibilidade de recursos para hospedar a VM, e.g. memória livre;
- Possuir os pré-requisitos para migração ao vivo (*live migration*), e.g. acesso a rede e armazenamento;
- Ser coerente ao grupo de migração, i.e. se a VM está no mesmo grupo de PM que ela pertence, e.g. grupo de produção, desenvolvimento, etc; e
- Não exceder os limites dos custos de implementação do cenário.

Apesar dos testes não focarem na qualidade dos Qualificadores, a fim de testar os métodos, as seguintes estratégias foram implementadas:

- Consolidar VMs no mínimo de PMs, em que as migrações que diminuem o desperdício de recursos são bonificadas. Esta estratégia visa liberar PMs ociosas a fim de desligá-las e, consequentemente, reduzir o consumo de energia.
- Distribuir as VMs de um mesmo serviço em diferentes PMs, pois estas tendem a usar os mesmos recursos ao mesmo tempo, piorando a sua performance devido a competição pelos recursos [3]. Desta maneira, bonificamos as migrações que distribuem as VMs e penalizamos as que as sobrecarregam com VMs do mesmo serviço.
- Distribuir VMs que utilizam o mesmo grupo de armazenamento, visando balancear a Rede Local de Armazenamento (Storage Area Network — SAN) por distribuição das VMs entre as PMs.

5.2 EFETIVIDADE DAS ESTRATÉGIAS USADAS NO ORDER@CLOUD

Nesta seção, testamos as principais estratégias do método de Organização: (i) a seleção de VMs usando classificação das alocações,

(ii) a geração e filtragem dos cenários dominados e (iii) variabilidade da busca por rejeição de VMs. As comparamos com soluções da literatura a fim de mostrar suas vantagens. Estas experiências são divididas nas seguintes categorias:

Seleção: A estratégia de Seleção define quais VMs serão migradas e, consequentemente, quais cenários serão gerados. Para entender as vantagens da nossa abordagem, que foca em melhorar as piores VMs, esta foi comparada com as seguintes abordagens:

- **Turno:** A Seleção por Turno, inspirada na proposta de Ren et al. [94], a qual, baseada na Teoria dos Jogos, seleciona aleatoriamente VMs e dá a chance de adotarem melhores alocações para si; e
- **TopRank:** seleciona as VMs que estão no topo da Classificação de Alocação para serem realocadas.

Geração: A geração dos cenários é um dos passos mais importantes do método, pois deve evitar o crescimento exponencial do número de soluções a serem explorados, utilizando regras e filtros. Assim, para mostrar a importância da filtragem por *Dominância*, a comparamos com um filtro de *Maior Benefício*, o qual aceita cenários cujo benefício é maior que zero em relação ao cenário anterior. Esta técnica também é utilizada por Xu et al. e Calyam et al. [81, 96]. Excepcionalmente, devido a grande quantidade de soluções, estes testes foram executados em um cenário com 600 VMs e variamos a quantidade máxima de migrações, entre 6 e 14.

Variabilidade: Aqui analisamos a efetividade das estratégias *VNS Adaptada*, a fim de evitar fins prematuros (*deadlocks*) durante a busca. Para demonstrar a eficácia da estratégia VNS Adaptada, que executa diversas pesquisas alterando os cenários iniciais, executamos os seguintes testes:

- **NoVNS:** Desabilitamos o uso da estratégia no fim da recursão, usando apenas durante a exploração;
- **NoExpVar:** - Desabilitamos o uso da estratégia durante a exploração de cenários, usando apenas no fim da recursão; e
- **VarOD:** Passamos a usar a estratégia apenas quando as piores VMs se demonstraram recorrentes de uma recursão para a outra, i.e. VMs selecionadas pela segunda vez consecutiva como a pior VM.

5.2.1 Testes de Seleção

Ao avaliarmos o crescimento das avaliações das 20 piores VMs (do cenário inicial), nosso método forneceu um maior custo-benefício em comparação a outros métodos, como mostrado na Figura 17. Verificamos um comportamento não usual na abordagem por Turno, mais precisamente nos cenários com 400 e 550 VM. Ao analisar o resultado final do mesmo, identificamos que a ordem de apresentação das VMs no cenário gerou tal anomalia. A abordagem acabou melhorando VMs fora do grupo das 20 piores, levando este aferimento para próximo de zero. Realmente este comportamento é esperado, dado a natureza da abordagem.

No maior cenário, como demonstrado na Figura 18, nossa solução aumentou o custo-benefício em significativos 36%, em comparação com o TopRank. Esse comportamento era esperado, uma vez que os outros métodos focam em melhorar a Nuvem como um todo, sem priorizar as menores alocações.

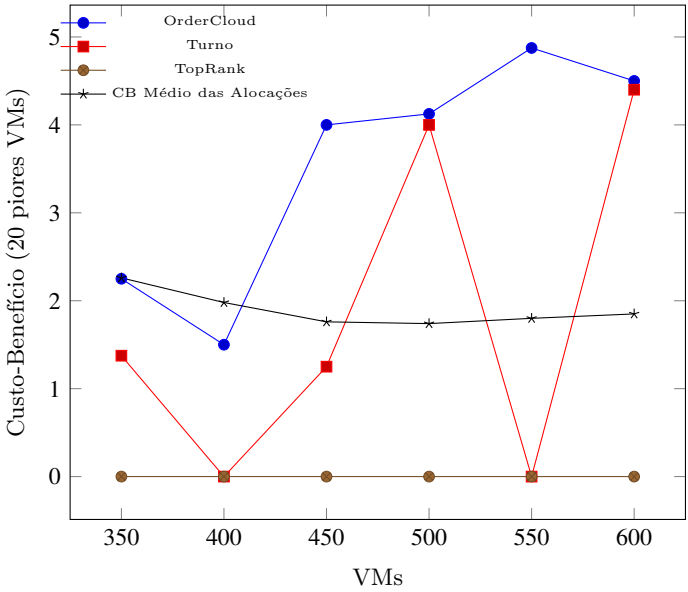

Figura 17 – Custo-Benefício das 20 piores VMs do cenário inicial, utilizando diferentes Métodos de Seleção: OrderCloud, Turno e TopRank.

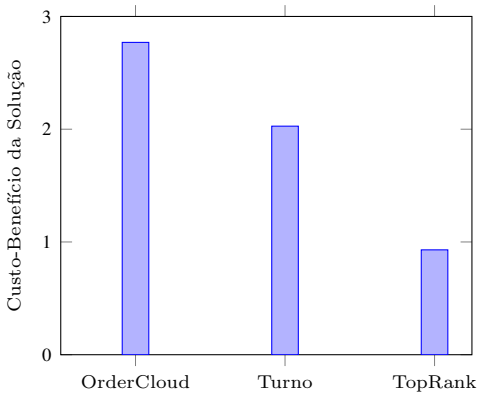

Figura 18 – Teste de Seleção – Custo-Benefício Alcançado Pelos Métodos de Seleção (OrderCloud, Turno e TopRank) no Cenário com 600 VMs

5.2.2 Testes de Geração de Cenários

Em comparação com a outra abordagem de filtragem, o filtro de *Dominância* impediu o crescimento exponencial dos cenários, enquanto aumentávamos o número máximo de migrações. Ainda, manteve um Custo-Benefício médio semelhante (0.002%), como visto na Figura 20.

O filtro de *Maior Benefício* elevou exponencialmente o número de cenários gerados ao ponto de tornar inviável a sua utilização em ambientes reais, devido ao seu tempo de execução extremamente alto. A Figura 19 mostra a duração das duas estratégias de filtragem.

Tentando prever o tempo de execução das propostas, o método de regressão exponencial nos levou a seguinte fórmula de aproximação: $y = 0.1832e^{0.47x}$, onde x é o número máximo de migrações e y é a aproximação do tempo de execução. Por outro lado, apesar de não ter um comportamento exponencial, tentamos utilizar o mesmo método de previsão em nossa proposta. O filtro por *Dominância* apresentou um crescimento de $y = 1.76e^{0.196x}$, que foi considerado um crescimento aceitável para o nosso ambiente. Testes com 16 migrações demonstraram um erro inferior a 4% para ambos, porém, para valores acima destes não podemos ter certeza.

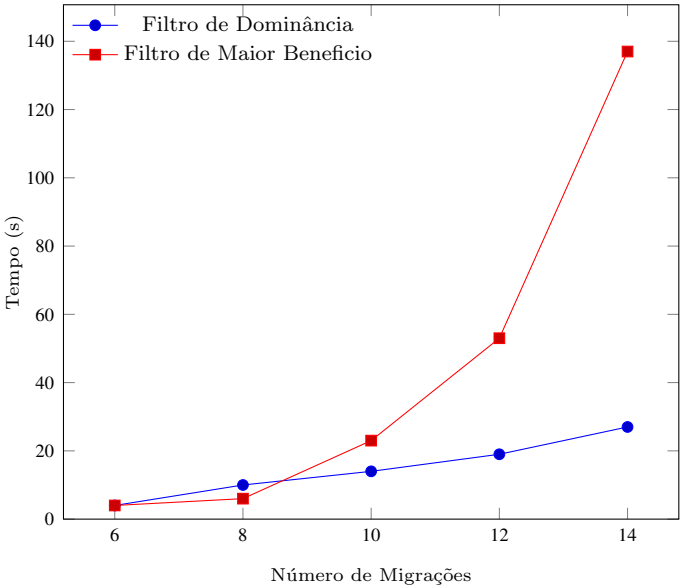

Figura 19 – Teste de Geração de Cenários – Tempo de execução das diferentes abordagens de filtragem de cenários.

5.2.3 Variabilidade

Os experimentos usando o método NoVNS entraram em *Deadlock* durante a pesquisa. Sem uma estratégia para variar os pontos de pesquisa, o método estagnou suas explorações, limitando os benefícios dos resultados alcançados a um valor próximo de zero.

O experimento, ao empregar o método *Variação Sob Demanda* (VarOD), apresentou os mesmos aferimentos de qualidade que a nossa proposta, e.g. custo-benefício, benefício médio, etc. No entanto, este teste gerou consideravelmente mais cenários e, consequentemente, teve um tempo de execução em média 19% superior que o nosso.

Finalmente, apesar de ter aumentado uma pequena parcela do custo-benefício, o experimento NoExpVar não afetou as piores alocações da Nuvem. Este método, apesar de alcançar um benefício 25% maior que nossa proposta, em média, degradou o custo-benefício das piores VMs em 31%, conforme é visto na Fig.21.

Em resumo, devido ao filtro de Dominância, o crescimento do número de cenários foi drasticamente reduzido e a abordagem de *Seleção*

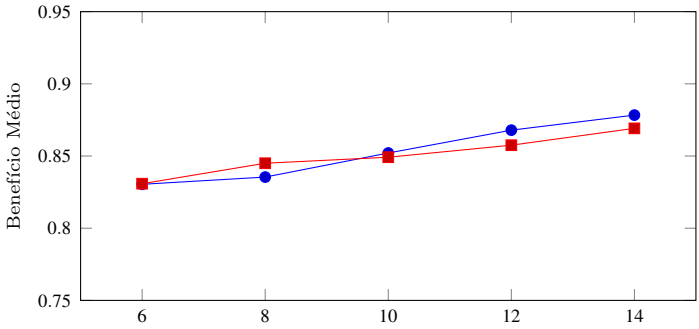

Figura 20 – Benefício Médio das Soluções com o aumento de Migrações.

por Classificação assegurou que os benefícios alcançados se concentrassem nas piores alocações. Graças à adaptação da abordagem VNS, isto assegurou a variabilidade do método, fazendo com que muitos *deadlocks* fossem evitados, permitindo a contínua evolução dos cenários.

5.3 MENSURAÇÃO DA OTIMICIDADE ALCANÇADA

Ao tentar quantificar a perda de qualidade dos resultados em nossa solução, devido ao uso do filtro de Dominância, comparamos nosso método de busca com: (i) uma mono-busca em largura, (ii) uma multi-busca em largura e (iii) a utilização de um filtro menos restritivo, baseado na abordagem de Xu et al. [81].

Tanto a mono-busca quanto a multi-busca em largura contrapõe nossa abordagem de migrar apenas a pior VM do cenário. Ambas simulam a migração de todas as VMs da nuvem para todas as alocações possíveis de cada VM, i.e. cada interação pode gerar até $O(xy)$ $(\#PM * \#VM)$ possíveis cenários. No entanto, a mono-busca seleciona o cenário não-dominado que oferece o maior benefício para a pior VM, e continua a busca explorando este cenário. Por outro lado, a multi-busca seleciona **todos** os cenários não-dominados que beneficiam a pior VM, e explora cada um deles.

Teoricamente, estas buscas encontram melhores soluções que nossa abordagem, pois abrangem um conjunto maior de possibilidades. Porém, a multi-busca gera muito mais cenários a serem analisados, o que reflete diretamente no tempo de processamento do método, como

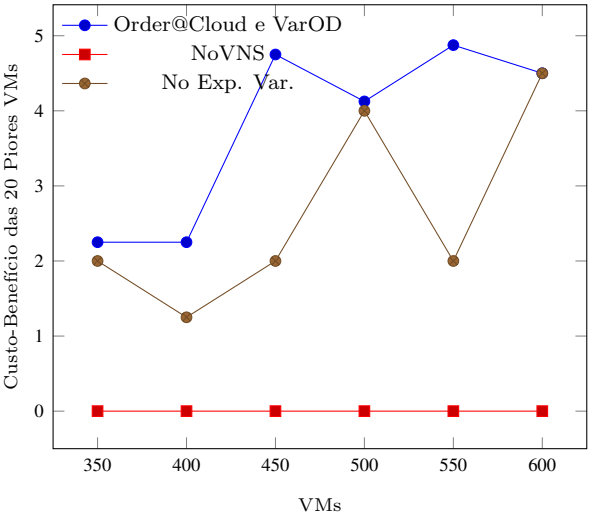

Figura 21 – Experimentos com estratégia de Variabilidade.

mostra a Figura 22(a). Neste experimento esta abordagem processou $4 * 10^7$ cenários, enquanto a mono-busca gerou $5 * 10^4$ e nossa proposta $5 * 10^3$. Como podemos verificar nas Figuras 22(b) e 22(d), a quantidade de cenários refletiu na qualidade dos resultados, em todos os experimentos a multi-busca alcançou soluções melhores que as outras abordagens. Quando analisamos o benefício alcançado pelas 20 piores VMs, mostrado pela Figura 22(c), notamos que nossa proposta excedeu a multi-busca. Apesar de contra-intuitivo, isto ocorre pelo fato da multi-busca encontrar cenários com custo-benefício superiores. Porém, estes, melhoram outras VMs não pertencentes as piores VMs.

A mono-busca apresentou um aumento relativamente baixo em relação as outras abordagem. Isto decorre quando mais de um cenário prove o mesmo benefício para a pior VM. Logo, ele escolhe o primeiro destes cenários e continua a busca. Fato que se demonstrou não muito efetivo.

Já a terceira abordagem emprega o filtro de *Maior Benefício* como heurística para a Busca Gulosa, ou seja, ela explora somente os cenários que oferecem maiores benefícios. Visto que a proposta lida com VMs sob demanda, decidimos usar o método de *Seleção por Classificação*, em vez de uma *Seleção Aleatória*. Além disso, para evitar descartar ramificações válidas da busca, desabilitamos as restrições de *Custo Máximo*.

Dado que na fase de *Seleção do Melhor Cenário* seleciona-se um

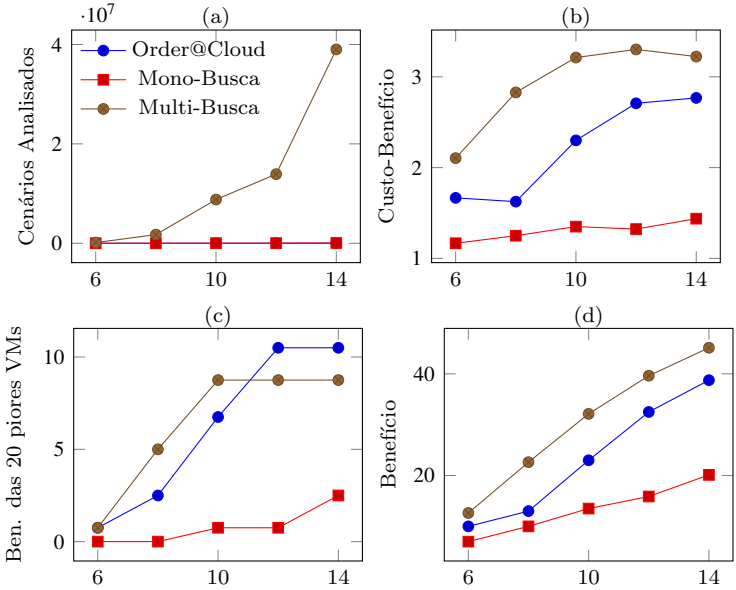

Figura 22 – Comparação com Busca em Largura: (a) Quantidade de cenários gerados; (b) Custo-Benefício; (c) Benefício das 20 piores VMs; e (d) Benefício do cenário; variando o número máximo de migrações permitidas (Eixo X).

não-dominado para continuar a busca, para analisarmos o quão longe estaríamos da solução ótima realizamos duas buscas, selecionando nesta fase o cenário com o maior: (i) Benefício e (ii) Custo-Benefício.

Quando comparamos os resultados do nosso método com os resultados da busca (ii), verificamos que ambos chegaram nas mesmas soluções, como mostrado na Figura 23(c). Em comparação com a busca (i), este alcançou um benefício 26% maior do que a nossa abordagem, como mostra a Figura 23(b). Porém, seu custo de implementação é tão elevado que nosso custo-benefício acabou sendo 208% maior que este resultado, como demonstra a Figura 23(c).

Os resultados referentes ao tempo de execução, representados pela Figura 23(a), tornam evidente o desempenho superior do nosso método, uma vez que, no teste mais longo, este demorou menos de 1 segundo para atingir um resultado. Por outro lado, o método comparado necessitou de 30 segundos em seu teste mais rápido, e cerca de 19 minutos no teste mais longo.

Em resumo, mesmo sendo 26% menor que o cenário ideal, nosso método ainda continua sendo a melhor escolha para encontrar os cenários com o melhor custo-benefício. Assim, estes resultados nos levam a outra pergunta: ele seria escalável para um ambiente real de Nuvem? Esta questão é abordada nos experimentos de *Escalabilidade*.

5.4 ESCALABILIDADE DA ORGANIZAÇÃO

Nos experimentos relacionados a *escalabilidade*, medimos o tempo necessário para avaliar um ambiente real e propor um resultado. Nestes experimentos variamos o tamanho do ambiente (350 a 600 VMs) e o numero máximo de migrações permitidas, de 5 a 20.

Os resultados do experimento mostram que, mesmo no maior cenário (600 VMs e 20 migrações), nosso método demorou 62 segundos para retornar um resultado — usando um hardware padrão conforme descrito na seção anterior. A Figura 24 apresenta graficamente o crescimento das avaliações, enquanto a Tabela 7 apresenta as taxas de crescimento do maior cenário em comparação aos outros dois menores, com: (a) 350 VMs e 20 migrações e (b) 600 VMs e 5 migrações.

Curiosamente, a restrição de Custo apresentou um maior impacto no tempo de execução do que o tamanho do cenário. Este aumento elevou consideravelmente o custo-benefício do resultado encontrado, o número de cenários analisados e, consequentemente, o tempo de execução. Ainda assim, um padrão linear na quantidade de cenários

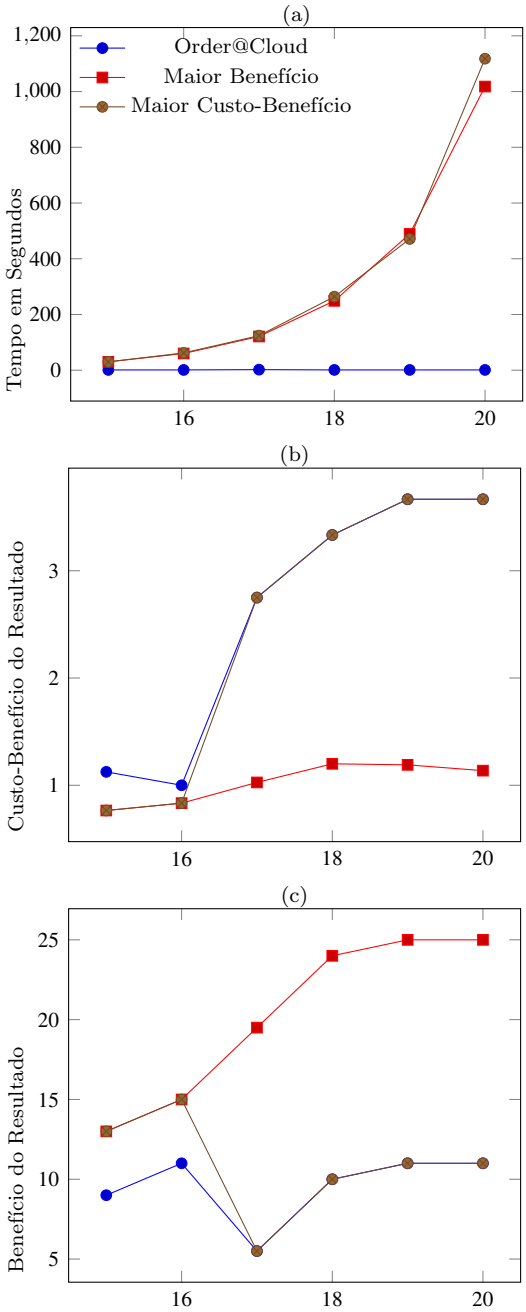

Figura 23 – Comparação de Tempo, Benefício e Custo-Benefício com métodos de filtragem menos restritivos.

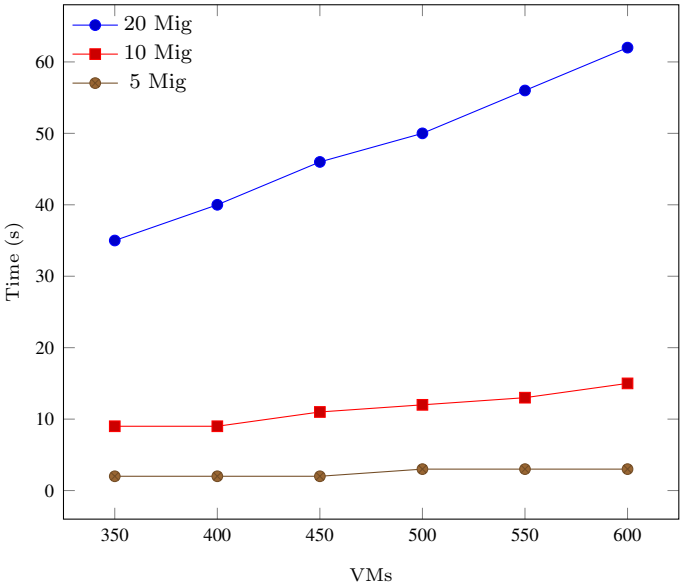

Figura 24 – Tempo de Execução com diferentes Custos-Máximos, i.e. número máximo de migrações.

foi observado, devido a restrição de custos, o que limita a expansão da busca.

Estes resultados sugerem que a nossa abordagem, mesmo sem otimização e paralelização, lida bem com Nuvens de pequeno e médio porte. No entanto, também percebemos que seu desempenho está relacionado com a *Taxa Média de Alocação* (TMA) do cenário, a qual representa a quantidade média de lugares que cada VM pode ser hospedada. A TMA pode ser calculada utilizando a *Matrix de Resumo* das RFC, apresentada na Seção 4.1.1.1, pg.54. Esta afeta a quantidade de cenários gerados e, consequentemente, o tempo de execução.

Assessment / Scenario	(a) VMs[350] e Max[20]	(b) VMs[600] e Max[5]
Cost-Benefit	+44%	+104%
Execution Time	+77%	+1966%
Analysed Scenarios	+2%	+7355%

Tabela 7 – Crescimento do Pior Cenário em relação aos Cenários menores.

6 CONCLUSÃO

Nesta tese propomos uma nova meta-heurística para Organização e Provimento de VMs na Nuvem a qual se baseia na Classificação de Alocações, nos custos e limitações do ambiente. Ela é apresentada na forma de um novo e flexível *Framework* para VMP, que possibilita o provisionamento, organização e adaptação de cenários de Nuvem não conformes. O framework foi implementado como um projeto de código-aberto[1] e é o primeiro que considera as avaliações de MOs em conjunto com os Custos de Implementação para organizar VMs na Nuvem. Além disso, suporta diferentes tipos de objetivos, SLAs, melhores práticas e custos na forma de qualificadores, regras e funções de prioridades e custos.

A fim de testar nossa hipótese e demonstrar as vantagens da nossa solução, realizamos vários experimentos aferindo cada um de seus componentes, sua escalabilidade, eficácia e também comparando-a com outras abordagens. Isto comprova a hipótese de que nossa proposta consegue prover resultados com qualidade similar as outras abordagens em menos tempo, i.e. analisando menos cenários.

6.1 PRINCIPAIS CONTRIBUIÇÕES

Para alcançar os objetivos almejados buscamos responder cada uma das perguntas que guiaram esta pesquisa:

Pergunta 1: *Como definir o problema e abordar VMP de forma agnóstica aos Múltiplos-Objetivos?*
Verificamos que o problema de VMP, com MO, poderia ser abordado de duas formas: (i) Puramente MO, o qual considera todos os objetivos ao mesmo tempo, ou (ii) MO como mono-objetivo, o qual resume todos os objetivos a uma só medição. Ambos oferecem vantagens e desvantagens: enquanto no primeiro surge a necessidade da resolução de conflito entre os objetivos, o segundo pode causar a má representação dos mesmos. Por este motivo adotamos uma abordagem híbrida em nossa solução, utilizando as duas abordagens em momentos distintos do processo.

Pergunta 2: *Como descrever uma Nuvem quando se almeja resolver problemas de VMP agnósticos aos Objetivos?*

[1] Disponível em `http://ordercloud.lrg.ufsc.br`

Concluímos que para representar uma Nuvem com MOs genéricos, seria necessário um modelo que provesse funções analíticas que representassem: (i) as Regras que restringem as alocações, (ii) os Qualificadores, que mensuram a qualidade da Nuvem e das suas alocações, (iii) os Pesos dos Qualificadores, que aferem suas prioridades, (iv) os Custos de Implementação de cada cenário, que mensuram o esforço para se alcançar a solução e (v) os Limites de Investimento, que representam os custos máximos que podem ser dispendidos pela Nuvem no processo de organização.

Pergunta 3: *Como acelerar a conversão dos métodos sem estagnar os mesmos?*
Identificamos que vários elementos podem auxiliar na convergência dos métodos, entre eles: (i) as Regras do ambiente, que vetam a utilização de PMs por VMs, (ii) os Custos de Implementação, os quais limitam a exploração de cenários infactíveis, i.e. acima dos custos máximos, (iii) a Classificação de VMs, a qual foca as migrações a um subconjunto de VMs da Nuvem e (iv) o Conceito de Dominância, o qual seleciona apenas os resultados não-dominados. A fim de evitar a estagnação de (iv), em ambientes com muitos objetivos, mudamos o foco da sua aplicação. Passamos a analisar as alocações dos cenários, invés das mensurações dos objetivos de cada alocação. Desta maneira, a desvalorização dos objetivos deixa de estagnar a busca por novos cenários.

Pergunta 4: *Como o problema e os métodos de VMP mudam quando o Custo de Implementação é limitado?*
A existência de uma limitação para a implementação da solução nos induziu a adotar uma abordagem que analisasse a relação de Custo-Beneficio das soluções, i.e. selecionar os cenários que oferecem o maior benefício em relação ao seu custo de implementação. Da mesma maneira, adotamos a Classificação de VMs para garantir a melhoria das VMs menos qualificadas no processo de organização da Nuvem. Como consequência, isto também corroborou com a convergência dos métodos.

Por atingirmos todos os objetivos definidos, afirmamos que as principais contribuições para o estado da arte do campo de Alocação de Máquinas Virtuais são: (i) a definição do modelo de Nuvem agnóstico aos objetivos e que contempla o custo das mudanças do ambiente, (ii) a meta-heurística baseada na classificação das Alocações, MOs e

custos do ambiente, (iii) os métodos de Organização, Provisionamento e Adaptação, baseados nesta meta-heurística e (iv) a implementação e disponibilização destas propostas na forma de um Framework de VMP, disponível a comunidade científica. Estas contribuições foram publicadas em [48, 49].

6.2 TRABALHOS FUTUROS

Durante o desenvolvimento deste trabalho identificamos algumas questões a serem aprofundadas nas próximas pesquisas. Dentre elas estão: (i) pesquisar os impactos do *Taxa Média de Alocação* sobre a convergência dos métodos, como discutido na Seção 5.4; (ii) utilizar estratégias inteligentes para reordenação da aplicação de Regras e Filtros, visando otimizar os métodos; (iii) identificar abordagens de Provimento Dinâmico de VMs que poderiam ser adotadas neste método, característica que estava fora do escopo desta proposta; (iv) visto que o modelo apresentado é baseado em funções, almeja-se modelar a meta-heurística em linguagem funcional, ao invés de imperativa; (v) analisar a utilização destes métodos em outros ambientes, tais como os baseados em Containers; (vi) integrar o framework com linguagens formais, a fim de representarmos regras e qualificadores por via de linguagens formais, como SLAC [97] e SCEL [98], possibilitando a criação e distribuição destas em bibliotecas online.

REFERÊNCIAS

[1] K. Konstantinos, M. Persefoni, F. Evangelia, M. Christos, and N. Mara, "Cloud computing and economic growth," in *Proceedings of the 19th Panhellenic Conference on Informatics*. ACM, 2015.

[2] D. Eaves, *The Explosive Growth of Cloud Computing*, 2015. [Online]. Available: http://www.business2community.com/infographics/explosive-growth-cloud-computing-infographic-2-0849951

[3] R. B. Uriarte, S. Tsaftaris, and F. Tiezzi, "Service clustering for autonomic clouds using random forest," in *Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on*. IEEE, 2015.

[4] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual machine power metering and provisioning," in *1st ACM Symposium on Cloud Computing – SoCC*. ACM, 2010, pp. 39–50.

[5] S. Nathan, P. Kulkarni, and U. Bellur, "Resource availability based performance benchmarking of virtual machine migrations," in *4th International Conference on Performance Engineering – SPEC*. ACM, 2013.

[6] M. Uddin, A. Shah, R. Alsaqour, and J. Memon, "Measuring efficiency of tier level data centers to implement green energy efficient data centers," *Middle-East Journal of Scientific Research – MEJSR*, 2013.

[7] A. V. Papadopoulos and M. Maggio, "Virtual machine migration in cloud infrastructures: Problem formalization and policies proposal," in *IEEE Conference on Decision and Control*, 2015.

[8] F. Kong and X. Liu, "A survey on green-energy-aware power management for datacenters," *ACM Computing Surveys – CSUR*, 2014.

[9] C. Ge, Z. Sun, and N. Wang, "A survey of power-saving techniques on data centers and content delivery networks," *Communications Surveys & Tutorials, IEEE*, 2013.

[10] K. Bilal, S. U. R. Malik, O. Khalid, A. Hameed, E. Alvarez, V. Wijaysekara, R. Irfan, S. Shrestha, D. Dwivedy, M. Ali *et al.*, "A taxonomy and survey on green data center networks," *Future Generation Computer Systems*, 2014.

[11] E. C. Inacio and M. A. R. Dantas, "A survey into performance and energy efficiency in hpc, cloud and big data environments," *Int. J. Netw. Virtual Organ.*, Mar. 2014.

[12] C. J. Rathod, "A survey on different virtual machine placement algorithms," *International Journal of Advanced Research in Computer Science and Management Studies*, 2014.

[13] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *Journal of Network and Computer Applications*, 2015.

[14] G. Aceto, A. Botta, W. De Donato, and A. Pescapè, "Cloud monitoring: A survey," *Computer Networks*, 2013.

[15] F. Lopez Pires and B. Baran, "A virtual machine placement taxonomy," in *I.C. on Cluster, Cloud and Grid Computing – CCGrid*. IEEE, 2015.

[16] C. von Lücken, B. Barán, and C. Brizuela, "A survey on multi-objective evolutionary algorithms for many-objective problems," *Computational Optimization and Applications*, 2014.

[17] R. Mendes, R. Weingartner, G. Geronimo, G. Bräscher, A. Flores, C. Westphall, and C. Westphall, "Decision-theoretic planning for cloud computing," *ICN 2014*, p. 202, 2014.

[18] Y. C. Liang and M. H. Lo, "Multi-objective redundancy allocation optimization using a variable neighborhood search algorithm," *Journal of Heuristics*, 2010.

[19] P. Hansen, N. Mladenović, and J. A. M. Pérez, "Variable neighbourhood search: methods and applications," *Annals of Operations Research*, 2010.

[20] E. Brynjolfsson, P. Hofmann, and J. Jordan, "Cloud computing and electricity: beyond the utility model," *Communications of the ACM*, 2010.

[21] NIST, "The nist definition of cloud computing," 2011. [Online]. Available: http://csrc.nist.gov/publications/PubsSPs.html

[22] B. Sotomayor, R. Montero, I. Llorente, I. Foster *et al.*, "Virtual infrastructure management in private and hybrid clouds," *IEEE Internet Computing*, 2009.

[23] EC2, "Amazon ec2," 2014. [Online]. Available: http://amazon.com/ec2

[24] Heroku, "The heroku platform," 2016. [Online]. Available: http://www.heroku.com

[25] Google Inc. , "Google cloud platform," 2016. [Online]. Available: https://cloud.google.com/appengine/

[26] Google Inc., "Google mail service," 2016. [Online]. Available: http://gmail.com

[27] Dropbox Inc, "Dropbox," 2011. [Online]. Available: http://dropbox.com

[28] Seafile, "The cloud storage server," 2016. [Online]. Available: http://www.seafile.com

[29] B. I. Ismail, M. Mydin, M. Nizam, and M. F. Khalid, "Architecture of scalable backup service for private cloud," in *Open Systems (ICOS), 2013 IEEE Conference on*. IEEE, 2013.

[30] VMWare, "Vmware esxi," 2014. [Online]. Available: http://www.vmware.com

[31] KVM, "Kernel-based virtual machine," 2014. [Online]. Available: http://www.linux-kvm.org

[32] L. Foundation, "The xen project," 2016. [Online]. Available: http://www.xenproject.org/

[33] X. Meng, C. Isci, J. Kephart, L. Zhang, E. Bouillet, and D. Pendarakis, "Efficient resource provisioning in compute clouds via vm multiplexing," in *Proceedings of the 7th international conference on Autonomic computing*. ACM, 2010.

[34] S. Abar, P. Lemarinier, G. K. Theodoropoulos, and G. M. OHare, "Automated dynamic resource provisioning and monitoring in virtualized large-scale datacenter," in *Advanced*

Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on. IEEE, 2014.

[35] C. Yan, M. Zhu, X. Yang, Z. Yu, M. Li, Y. Shi, and X. Li, "Affinity-aware virtual cluster optimization for mapreduce applications," in *Cluster Computing (CLUSTER), 2012 IEEE International Conference on.* IEEE, 2012.

[36] B. Viswanathan, A. Verma, and S. Dutta, "Cloudmap: workload-aware placement in private heterogeneous clouds," in *Network Operations and Management Symposium (NOMS), 2012 IEEE.* IEEE, 2012.

[37] J. Xu and J. A. Fortes, "Multi-objective virtual machine placement in virtualized data center environments," in *International Conference on Green Computing and Communications – GreenCom.* IEEE, 2010.

[38] C. Hyser, B. Mckee, R. Gardner, and B. J. Watson, "Autonomic virtual machine placement in the data center," 2008.

[39] N. M. Calcavecchia, O. Biran, E. Hadad, and Y. Moatti, "Vm placement strategies for cloud scenarios," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on.* IEEE, 2012, pp. 852–859.

[40] L. Gu, "Cost efficient resource management for geo-distributed data centers," Ph.D. dissertation, The University of Aizu, 2015.

[41] A. N. Tantawi, "On biasing towards optimized application placement in the cloud," in *Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2015 IEEE 23rd International Symposium on.* IEEE, 2015, pp. 71–74.

[42] S. H. H. Madni, M. S. A. Latiff, Y. Coulibaly *et al.*, "An appraisal of meta-heuristic resource allocation techniques for iaas cloud," *Indian Journal of Science and Technology*, vol. 9, no. 4, 2016.

[43] E. C. Man Jr, M. Garey, and D. Johnson, "Approximation algorithms for bin packing: A survey," *Approximation Algorithms for NP-Hard Problems*, 1996.

[44] A. Beloglazov and R. Buyya, "Energy efficient allocation of virtual machines in cloud data centers," in *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*. IEEE, 2010.

[45] J. Chen, K. Chiew, D. Ye, L. Zhu, and W. Chen, "Aaga: Affinity-aware grouping for allocation of virtual machines," in *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*, 2013.

[46] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing sla violations," in *Integrated Network Management, 2007. IM'07. 10th IFIP/IEEE International Symposium on*. IEEE, 2007.

[47] G. A. Geronimo, R. B. Uriarte, B. Sudoski, and C. Westphall, "Vm placement on clouds: A survey," *Computer Networks*, 2016.

[48] G. A. Geronimo, R. Uriarte, and C. Westphall, "Toward a framework for vm organisation based on multi-objectives," in *Proceedings of the I. C. on Networks – ICN2016*. IANA, 2016.

[49] G. A. Geronimo, R. B. Uriarte, and C. B. Westphall, "Order@cloud: A vm organisation framework based on multi-objectives placement ranking," in *Network Operations and Management Symposium – NOMS2016*. IEEE, 2016.

[50] IEEE/ACM. International symposium on cluster computing and the grid. [Online]. Available: http://ieeexplore.ieee.org/servlet/opac?puOPTnumber=1000093

[51] ACM/SigWeb/SIGIR. International conference on information and knowledge management. [Online]. Available: http://surveys.acm.org

[52] ACM. Conference on computer supported cooperative work. [Online]. Available: http://surveys.acm.org

[53] IEEE/IFIP. International conference on dependable systems and networks. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?&puOPTnumber=1000192

[54] IEEE. International symposium on high performance distributed computing. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000324

[55] IEEE. International conference on distributed computing systems. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000213

[56] ACM/IEEE. Conference on high performance networking and computing. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000729

[57] IEEE. International conference on advanced information networking and applications. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000008

[58] IEEE. Cluster computing conference. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000095

[59] IEEE/ACM. International conference on grid computing. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000093

[60] IEEE. International conference on autonomic computing. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1001178

[61] AAAI. International conference on automated planning and scheduling. [Online]. Available: http://www.aaai.org/ocs/index.php/ICAPS/index/schedConfs/archive

[62] IEEE. International conference on service oriented computing & application. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1001856

[63] IEEE/IFIP. Network operations and management symposium. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000491

[64] IARIA. International conference on networking. [Online]. Available: http://www.iaria.org/conferences/ICN.html

[65] IEEE. International conference on computer communications and networks. [Online]. Available: http://ieeexplore.ieee.org/xpl/conhome.jsp?puOPTnumber=1000127

[66] Elsevier. Computer communication review. [Online]. Available: http://www.journals.elsevier.com/computer-networks/

[67] Elsevier. Computers & operations research. [Online]. Available: http://www.journals.elsevier.com/computers-and-operations-research/

[68] Elsevier. Decision support systems. [Online]. Available: http://www.journals.elsevier.com/decision-support-systems/

[69] Elsevier. Environmental modelling & software. [Online]. Available: http://www.journals.elsevier.com/environmental-modelling-and-software/

[70] ACM. Computing surveys. [Online]. Available: http://surveys.acm.org

[71] Springer. Archives of computational methods in engineering. [Online]. Available: http://www.springer.com/engineering/computational+intelligence+and+complexity/journal/11831

[72] ACM. Communications of the acm. [Online]. Available: http://surveys.acm.org

[73] ACM. Computer communication review. [Online]. Available: http://surveys.acm.org

[74] Springer. Annals of operation research. [Online]. Available: http://www.springer.com/business+%26+management/operations+research/journal/10479

[75] Springer. Computational optimization and applications. [Online]. Available: http://www.springer.com/mathematics/journal/10589

[76] Springer. Journal of heuristics. [Online]. Available: http://www.springer.com/mathematics/applications/journal/10732

[77] Springer. Journal of optimization theory and applications. [Online]. Available: http://www.springer.com/mathematics/journal/10957

[78] G. A. Geronimo, J. Werner, C. B. Westphall, C. M. Westphall, and L. Defenti, "Provisioning and resource allocation for green clouds," in *12th I.C. on Networks – ICN*. IARIA, 2013.

[79] G. A. Geronimo, C. Westphall, C. Rolim, F. Koch, and J. Werner, "Modelo integrado de gestão de recursos para nuvens verdes." Centro Latinoamericano de Estudios en Informática, 2011.

[80] G. A. Geronimo, J. Werner, R. Weingartner, C. B. Westphall, and C. Westphall, "Provisioning, resource allocation, and dvfs in green clouds," *International Journal on Advances in Networks and Services*, vol. 7, 2014.

[81] J. Xu and J. Fortes, "A multi-objective approach to virtual machine management in datacenters," in *8th international conference on Autonomic Computing*. ACM, 2011.

[82] Z. Zhang, C.-C. Hsu, and M. Chang, "Cool cloud: A practical dynamic virtual machine placement framework for energy aware data centers," in *I.C. on Cloud Computing – CLOUD*. IEEE, 2015.

[83] O. Abdul-Rahman, M. Munetomo, and K. Akama, "Toward a genetic algorithm based flexible approach for the management of virtualized application environments in cloud platforms," in *21st International Conference on Computer Communications and Networks – ICCCN*. IEEE, 2012, pp. 1–9.

[84] C. C. T. Mark, D. Niyato, and T. Chen-Khong, "Evolutionary optimal virtual machine placement and demand forecaster for cloud computing," in *Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on*. IEEE, 2011.

[85] J. Chen, K. Chiew, D. Ye, L. Zhu, and W. Chen, "Aaga: Affinity-aware grouping for allocation of virtual machines," in *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*. IEEE, 2013.

[86] W. Fang, X. Liang, S. Li, L. Chiaraviglio, and N. Xiong, "Vmplanner: Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers," *Computer Networks*, 2013.

[87] M. Fayad, D. C. Schmidt, and R. E. Johnson, *Building application frameworks: object-oriented foundations of framework design*. Wiley, 1999.

[88] R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente, "Iaas cloud architecture: From virtualized datacenters to federated cloud infrastructures," *Computer*, 2012.

[89] OpenNebula Org., "Project documentation." [Online]. Available: http://docs.opennebula.org/4.12/administration/references/schg.html

[90] OpenStack Org., "Project website." [Online]. Available: https://www.openstack.org/

[91] OpenStack Org, "Project documentation." [Online]. Available: http://docs.openstack.org/kilo/config-reference/content/section_compute-scheduler.html

[92] E. Feller, L. Rilling, and C. Morin, "Snooze: A scalable and autonomic virtual machine management framework for private clouds," in *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*. IEEE Computer Society, 2012.

[93] Inria, "Snooze project." [Online]. Available: http://snooze.inria.fr/

[94] Y. Ren, J. Suzuki, A. Vasilakos, S. Omura, and K. Oba, "Cielo: An evolutionary game theoretic framework for vmp in clouds," in *I.C. on Future Internet of Things and Cloud – FiCloud*. IEEE, 2014.

[95] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *Transactions on Evolutionary Computation*, 2003.

[96] P. Calyam, R. Patali, A. Berryman, A. M. Lai, and R. Ramnath, "Utility-directed resource allocation in virtual desktop clouds," *Computer networks*, 2011.

[97] R. B. Uriarte, F. Tiezzi, and R. D. Nicola, "Slac: A formal service-level-agreement language for cloud computing," *7th I.C. on Utility and Cloud Computing*, 2014.

[98] R. De Nicola, M. Loreti, R. Pugliese, and F. Tiezzi, "Scel: a language for autonomic computing," *Univ. Firenze, Tech. Rep*, 2013.

[99] O. Biran, A. Corradi, M. Fanelli, L. Foschini, A. Nus, D. Raz, and E. Silvera, "A stable network-aware vm placement for cloud systems," in *Proceedings of the 2012 12th IEEE/ACM*

International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012). IEEE Computer Society, 2012.

[100] D. Dong and J. Herbert, "Energy efficient vm placement supported by data analytic service," in *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*. IEEE, 2013.

[101] R. Buyya, "Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities," *I.C. on High Performance Computing & Simulation – HPCS09*, pp. 1–11, 2009.

[102] R. R. Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," *Systems, Man and Cybernetics, IEEE Transactions on*, 1988.

[103] R. Panigrahy, K. Talwar, L. Uyeda, and U. Wieder, "Heuristics for vector bin packing," *research. microsoft. com*, 2011.

APÊNDICE A – Levantamento Bibliográfico

A.1 RESUMOS DAS PROPOSTAS

Nesta seção provemos um resumo de cada proposta avaliada, falando sobre seus objetivos e ressaltando suas principais características. Posteriormente, as analisaremos mais a fundo utilizando os aspectos definidos na Seção A.2.

MO-VMP [37] : Xu et al. propõe um método para provisionar novas VMs com foco em três objetivos específicos: (i) o uso eficiente de recursos multidimensionais (CPU, Memória, Carga), (ii) evitar pontos quentes no centro de dados e (iii) reduzir os custos de consumo de energia. Eles usam um algoritmo genético (AG) para identificar possíveis cenários de hospedagem para a nova VM. Em seguida, analisamos usando uma função *fuzzy* que resume as funções de avaliação dos objetivos. Os métodos do GA para *cruzamento* e *mutação* dos cenários são baseados em avaliações de desempenho por grupos, em que um grupo representa um conjunto de VMs hospedadas na mesma PM. Em geral, a abordagem seleciona o resultado que apresentar o menor: (i) desperdício de recursos, (ii) consumo de energia e (iii) pico de temperatura.

MO-VMM [81]: Visando os mesmos objetivos que MO-VMP (minimizar o desperdício de recursos, concentração térmicas e consumo de energia), Xu et al. focam aqui na alocação dinâmica de VMs. A Figura 25 mostra o fluxograma de monitoração, a proposta monitora condições pré-definidas (eventos) na Nuvem, relacionadas com os objetivos visados, disparando, assim, os processos de realocação. Quando acionado, o método seleciona uma VM com base em suas funções de utilidade. Em seguida, a VM é migrada para a PM que proporciona o maior aumento de utilidade.

AAGA [85]: Chen et al. propõe uma organização de VMs baseada na afinidade de comunicação. O método identifica relações de afinidade entre VMs, analisando o tráfego de rede, e as agrupa em conjuntos afins. Então, aloca os conjuntos em PMs próximas, visando reduzir o tráfego de rede. Este método utiliza um algoritmo guloso (não especificado) para alocar os recursos.

NA-VMP [99]: Da mesma maneira, Biran et al. têm como objetivo organizar VMs considerando a topologia da rede, afinidade de tráfego, distribuição e consolidação das VMs, a fim de evitar o

congestionamento da rede. Duas heurísticas são propostas: (i) 2-Phase Connected Component-based Recursive Split (2PCCRS) - rápido, porém de baixa qualidade - e (ii) Greedy Heuristic (GH) - alta qualidade, porém mais lento.

A abordagem 2PCCRS agrupa as VMs afins aos ativos de rede e, em seguida, mapeia as VMs para as PMs ligadas a estes equipamentos.

A GH classifica e ordena as demandas da rede, em seguida identifica as VMs relacionadas a estas demandas e descarta as VMs que não podem ou não valem a pena serem migradas. Depois, para cada demanda da rede, o método migra as VMs associadas a demanda, se possível.

VMPlanner [86]: Fang et al. visam reduzir o consumo de energia do centro de dados, consolidando VMs e desativando elementos ociosos da rede. A abordagem divide o problema de otimização em três fases. Em primeiro lugar, ele agrupa as VMs com o objetivo de maximizar o tráfego dentro dos grupos (intra-grupo) e minimizar o tráfego entre os grupos (inter-grupo). Em segundo lugar, tenta mapear as VMs nas PMs, minimizando o tráfego entre os *Racks* (*inter-rack*) do centro de dados. Finalmente, o VMPlanner migra as VMs a fim de centralizar o tráfego de rede no menor número de caminhos (paths) possíveis. Isto permite desligar os equipamentos de rede ociosos e conservar energia.

GA-VAM [83]: Abdul-Rahman et al. propõe um modelo de gestão de recursos baseado em um Algoritmo Genético *Binary-Real*, em que os recursos disponíveis são utilizados para satisfazer os objetivos deficitários. Nesta proposta, as decisões de otimização são delegadas aos agentes locais, no nível de serviços – por exemplo, as aplicações têm a possibilidade de redimensionar suas VMs.

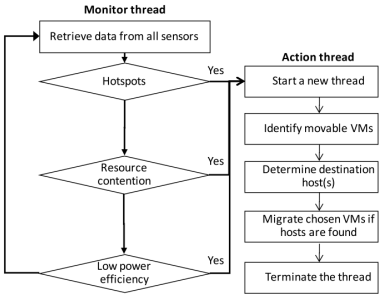

Figura 25 – Fluxo de controle do MO-VMM [81]

Seus objetivos podem ser resumidos em: (i) minimizar a sobrecarga do sistema, liberando recursos ociosos; (ii) minimizar custo de virtualização, investindo na escalabilidade vertical e horizontal, e consequentemente reduzindo o número de migrações; (iii) maximizar o benefício dos usuários, continuamente avaliando o QoS, SLA e elasticidade, e.g. fornecer recursos de forma transparente para serviços e VMs que necessitam de forma automática, sem intervenção humana.

MO-RAO [18]: Liang et al. propõe um novo algoritmo de Busca com Variação de Localidades baseado em MO (*Multi-objective variable neighborhood search* – MOVNS) para resolver problemas de alocação redundante com MO. Ele usa o conceito de *Dominância de Soluções* para reduzir o conjunto de cenários possíveis. Uma solução domina outra se não é pior em nenhum objetivo e é estritamente melhor em, pelo menos, um dos objetivos, em relação ao outro. Da mesma forma, as soluções do conjunto de Pareto são não-dominados por qualquer outra solução do espaço de solução viável [16].

A abordagem começa a partir de um cenário aleatório válido e faz uma busca em largura por soluções não-dominadas. A busca continua a explorar novos cenários até o momento em que um limite pré-definido de avaliação é atingido. Esta busca visa construir um conjunto de Pareto.

Ao fim da busca, a "localidade" é variada. Um novo cenário inicial aleatório é selecionado e a busca é reiniciada. No final, para avaliar o desempenho dos resultados encontrados, os seguintes indicadores são utilizados: Taxa de acerto, Taxa de Precisão, GD e $D1_R$. [16].

EE-AVM [44]: A fim de tratar separadamente os subproblemas do VMP, Beloglazov et al. o dividem em duas partes: (i) *Provisionamento* e (ii) *Optimização*. Para resolver o *Provisionamento*, é proposta uma versão modificada do algoritmo *Best Fit Decreasing* (MBFD), que mapeia as VMs nas PMs, visando reduzir o consumo de energia. A *Optimização* também é dividida em duas fases: (i) *Extração* e (ii) *Realocação*. Propõe-se uma heurística para selecionar quais VMs devem ser *extraídas* e outra para *realocá-las* na Nuvem, usando o algoritmo MBFD.

EE-VMP [100]: Dong et al. propõem um algoritmo para realocação de VMs que emprega técnicas de modelagem de dados e de previsão de carga para melhorar a eficiência energética da Nuvem, minimizando as violações dos SLAs. A abordagem utiliza o *framework* R e dados históricos (e.g. uso de CPU e violações de SLA) para prever o uso

de cada VM. Com base nessas previsões, o método tenta consolidar as VMs nas PMs minimizando os recursos ociosos, porém deixando o suficiente para atenuar eventuais picos de demanda.

EO-VMP [84]: Mark et al. propõem uma algoritmo Evolutivo para escolher em qual provedor de Nuvem uma nova VM deve ser instanciada. A proposta se aproveita da lacuna existente entre a quantidade de recursos *solicidada* e a *utilizada* pelo usuário. Com base em dados históricos do usuário, visa reduzir os custos do usuário através da atribuição de menos recursos do que o solicitado.

Snooze [92]: Feller et al. propo?m uma estrutura de auto-organização hierárquica e dinâmica, na qual as decisões são delegadas a agentes locais, dentro das PMs. Estas podem tomar ações sobre suas VMs hóspedes, tais como: migrar, redimensionar, iniciar, suspender, reiniciar, desligar e destruir.

No *framework* proposto, seus objetivos, gatilhos e agendamentos são representados como Políticas do Ambiente e submetidos aos agentes que rodam nas PMs, possibilitando a extensão e criação de novas políticas de acordo com a necessidade do provedor da Nuvem.

Auto-DRP [34]: Abar et al. unem o framework baseado em políticas *CBTool* com o simulador de Nuvens *CloudSim* [101] para auxiliar decisões de realocações. A simulação é alimentada por dados em tempo real, proveniente das PMs. Depois, os resultados das simulações (das decisões) são usados no controle do ambiente real. Seus objetivos, gatilhos e programações (*scheduling*) são representados como políticas, as quais podem ser estendidas de acordo com as necessidade da Nuvem.

U-RAM [96]: Calyam et al. focam no provisionamento de Desktops Virtuais (VD), isto é, um serviço específico. Propõem um modelo guiado por funções de utilidade, em que o objetivo é buscar uma alocação que proveja o mínimo de qualidade para o usuário. No entanto, durante o processo, cada VM deve permanecer acima dos requisitos mínimos de qualidade. Cada VM possui suas próprias dimensões e limites de qualidade, e cada dimensão deve estar acima de um limiar de qualidade.

A.2 ASPECTOS ANALISADOS

Para comparar e analisar os trabalhos relacionados definimos aspectos a serem verificados em cada proposta. Mais especificamente, a seguir, definimos e discutimos a importância e a classificação de cada aspecto analisado. Estes aspectos estão sumarizados na Figura 26.

A.2.1 Escopo da Proposta

O primeiro passo ao analisar uma proposta é definir o escopo de VMP que a solução visa cobrir. As propostas podem ser classificadas em uma das seguintes categorias:

Relocação (O): As abordagens desta categoria visam reorganizar as VMs a fim de melhorar os objetivos do ambiente.

Provisionamento (P): Foca em otimizar as alocações das VMs que estão sendo instanciadas na Nuvem, sejam ela nova ou não, i.e. VMs desligadas. Não contemplam a reorganização de VMs já hospedadas.

Híbrido (H): A abordagem contempla ambas categorias anteriores, Realocação e Provisionamento.

A.2.2 Agnosticismo

Dado o fato de que nem todas as abordagem são flexíveis o bastante para considerar N número de objetivos (Agnóstico aos Objetivos), este aspecto analisa se as propostas usam modelos que permitem a extensão de MO, e, se usarem, que abordagem utilizam para tratar os MO, como mono-objetivo ou Puramente MO, visto que os MOs podem conflitar entre si, ressaltamos como as propostas tratam destes conflitos.

A.2.3 Otimicidade

Esta analise verifica quais estratégias as abordagens utilizam para encontrar e adotar os melhores cenários, visto que ao lidar com MOs tratamos com diversos resultados, com prós e contras distintos, analisamos que critérios são utilizados para selecionar uma solução. Assim, dividimos as estratégias de busca e seleção nas seguintes categorias:

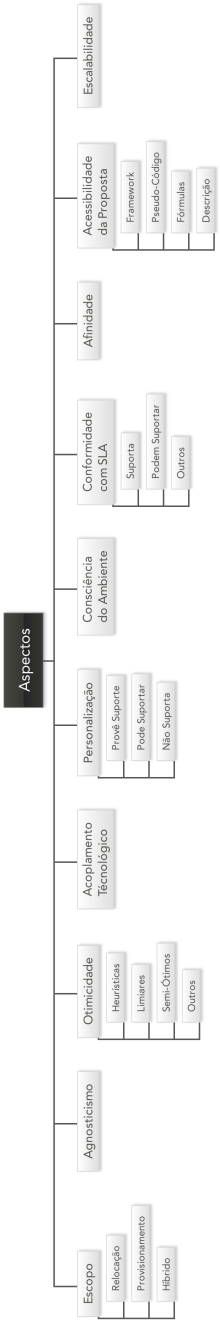

Figura 26 – Características usadas para analisar os artigos de VMP

Heurísticas (H): Utilizam heurísticas para alterar o cenário, buscando melhorá-lo. Um limiar de qualidade controla quando a busca deve parar, i.e. para quando o cenário é considerado aceitável.

Limiares (T): Buscam por soluções até que um ou mais limiares processuais são alcançados. Esta classificação pode ser considerada uma extensão dos métodos de *Heurísticas*, que são orientados pela qualidade. No entanto, também consideram limiares relacionados aos processo de busca, e.g. tempo de processamento, número de interações ou cumprimento de restrições.

Semi-Ótimos (S): As abordagens nesta categoria avaliam um conjunto de possíveis soluções e selecionam uma, geralmente baseando-se em funções de qualificação.

Outros: Abordagens não classificadas nas categorias anteriores.

A.2.4 Personalização

A Nuvem é um ambiente dinâmico e suscetível a fatores exógenos que requerem alterações nas regras e prioridades dos objetivos, e.g. feriados, eventos e mudança de leis. Desta maneira, avaliamos a flexibilidade oferecida pelas propostas analisando se estas ofereciam a possibilidade de mudar as suas: prioridades, interesses ou estratégias. Por exemplo, a adoção de diferente objetivos durante manutenções de emergência, ou a reorganização da Nuvem a fim de liberar recursos específicos. Mais especificamente, as mudanças podem ocorrer durante a execução da solução ou na inicialização do processo. Assim, dividimos as propostas nas seguintes categorias: (i - ✓) Provê suporte, (ii - M) Pode Suportar e (iii) Não Suporta.

A.2.5 Consciência do Ambiente

As propostas de VMP geralmente focam em elementos computacionais (e.g. VMs, PMs, rede, armazenamento) e, em casos raros, consideram sistemas de refrigeração. No entanto, a Nuvem também envolve outros elementos do centro de dados e sistemas anexos, e.g. Sistema de Provimento de Energia, Sistema de Cópias de Segurança, Segurança Física e sistemas contra incêndio. A consciência sobre o ambiente e outros elementos do centro de dados é importante pois interfere diretamente na disponibilidade da hospedagem dos serviços. Assim, este aspecto analisa se as propostas provem suporte para (i) considerar elementos externos e, se possibilitam, (ii) como incorporam em seus métodos.

A.2.6 Conformidade com SLA

Visto que Nuvens são ambientes orientados por serviços, SLAs são elementos essenciais para o gerenciamento da Nuvem e devem ter parte no processo de decisão.

Assim, este aspecto analisa como as propostas consideram restrições administrativas em seus métodos, e.g. SLAs, SLOs e políticas internas. Desta maneira, propostas que explicitamente descrevem *Onde* e *Como* estas restrições são aplicadas foram classificadas como *Suporta*, caso contrário, *Não Suporta*. As que não se encaixam nestas categorias foram classificadas como *Outros*.

A.2.7 Afinidade

Afinidade pode ser definido como relações de dependência entre VMs [85] ou outros elementos da Nuvem, e.g. VMs que possuem uma significante troca de tráfego. Este conhecimento é usado para decidir entre centralizar ou separar VMs, visando melhorar a performance do ambiente como um todo. Dada esta importância, propostas como [3] vem explorando maneiras de identificar conjuntos de serviços similares, visando descobrir afinidades escondidas entre seus elementos.

Este aspecto analisa se as propostas usam este conhecimento no processo de VMP. Primeiramente analisamos se consideram afinidade, então, caso positivo, como utilizam este conhecimento. Somente propostas que explicitamente descrevem que casos de afinidades suportam e como as suportam foram classificadas nesta categoria.

A.2.8 Acoplamento Tecnológico

Mesmo adotando múltiplos objetivos, modelos e métodos podem estar acoplados a tecnologias e problemas de tal forma que se tornam obsoletos após uma mudança tecnológica. Assim, investigamos a sensibilidade das propostas a mudanças tecnológicas, i.e. o quão dependente a proposta é de uma tecnologia específica. Para isto, ressaltamos as propostas tecnologicamente independentes de seus métodos e modelos, e comentamos cada abordagem.

A.2.9 Escalabilidade

Visto que a Nuvem pode variar de tamanho entre dezenas e milhares de VMs, é necessário avaliar se as propostas atendem tais magnitudes. Neste aspecto, analisamos como esta questão é tratada por cada proposta e, na Tabela 8, sobressaímos o tamanho do ambiente em que elas foram testadas, i.e. quantas VMs foram utilizadas nos testes.

A.2.10 Acessibilidade da Proposta

Reprodutibilidade é a habilidade de duplicar inteiramente um estudo ou experimento. No entanto, este processo nem sempre é simples e fácil. Dependendo, em muitos casos de como as ideias são descritas e compartilhadas com o público, esta tarefa se torna impraticável.

Assim, neste aspecto avaliamos: (i) como a proposta é apresentada e compartilhada, e.g. como um framework, pseudo-código, fórmulas ou descrição; (ii) onde o código pode ser encontrado; e (iii) em que linguagem foi implementada.

A.3 DISCUSSÃO DAS PROPOSTAS

Nesta seção, analisamos e discutimos as propostas, ressaltando as abordagens adotadas em cada aspecto apresentado na Seção A.2. O resumo das análises se encontra na Tabela 8.

A.3.1 Escopo

Entre as propostas classificadas como *Realocação*, listadas na Tabela 8, uma questão sobre o gatilho dos processos surgiu: *Quando os métodos começam a organizar?*

No entanto, a maioria das propostas não especifica, ou não deixa claro que evento leva a iniciar o processo de organização. Isto nos levou a interpretar que estes são manualmente executados.

Algumas propostas, como MO-VMM, Snooze e VMPlanner, definem múltiplos limiares como gatilho para os seus métodos. Eles constantemente monitoram a Nuvem, analisando métricas relacionadas aos seus objetivos e, quando um ou mais limiares são violados, os métodos são executados. No entanto, quando analisamos suas aplicabilidades, verificamos que a abordagem adotada por Snooze não pode ser adotada em qualquer ambiente, pois este necessita que seus agentes rodem dentro do Sistema Operacional das PMs. Por outro lado, MO-VMM e VMPlanner adotam uma estratégia menos intrusiva, usando um módulo central para receber e analisar os dados monitorados do ambiente, e.g. temperatura das PMs e a eficiência dos recursos e do consumo de eletricidade. Quando eventos especiais são identificados, eles iniciam os devidos processos, se necessário.

As propostas classificadas como *Provisionamento*, como EO-VMP e MO-VMP, adotam algoritmos evolucionários com diferentes metas. Ambas as propostas tentam encontrar alocações que melhoram os aferimentos de seus objetivos. EO-VMP tenta minimizar os custos dos usuários e MO-VMP foca em 3 objetivos específicos: (i) utilização eficiente de múltiplos recursos, (ii) evitar pontos de aquecimento no centro de dados, (iii) reduzir o consumo de energia.

Por outro lado, U-RAM visa manter um nível aceitável de *Qualidade de Experiência* para seus usuários de Desktops Virtuais (VDs). Para isto, durante a instanciação dos novos VDs ele analisa métricas relacionadas a qualidade da interação do usuário com o serviço, (e.g. qualidade de renderização da imagem) referindo-se a estas como *Dimensões de Qualidade*. Em seguida, redireciona a instanciação para a PM que assegura um mínimo de qualidade necessária para o usuário.

Na categoria *Híbrida*, o framework Auto-DRP alega lidar com o provisionamento e realocação de VMs baseando-se em políticas de Nuvem, no entanto, nenhum exemplo de realocação foi provido, apenas para provisionamento. GA-VAM, por sua vez, adota uma abordagem diferente, representando seus objetivos com políticas e os distribuindo entre agentes localizados nas PMs. Assim, cada PM é responsável por aplicar as Políticas da Nuvem em suas VMs, independente da procedência da VMs (novas ou desativadas). Da mesma maneira, a abordagem de *Extrair e Relocar* proposta por EE-AVM, não apenas permite a reusabilidade de seus métodos, mas também proporciona uma maior adaptabilidade em casos de mudança tecnológica, i.e. sua abordagem modular permite facilmente estender e adaptar seus métodos internos.

Concluindo, notamos uma tendência de segregação e especialização dos métodos, em que responsabilidades – como organização, provisionamento e reação a eventos – são divididos em métodos, e estes são especializados de acordo com suas responsabilidades. Esta divisão possibilita a personalização dos parâmetros, restrições e objetivos, dependendo de qual responsabilidade está sendo processada no momento, como o carregamento seletivo de políticas e regras. Esta prática de dissociação melhora e simplifica os métodos. A Figura 27 apresenta a utilização destes escopos nos trabalhos analisados.

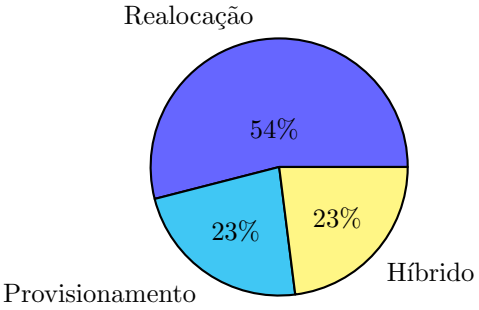

Figura 27 – Escopo das Propostas

A.3.2 Personalização

MO-VMM foi a única proposta que explicitamente provê a *Personalização* de seu método, sem alterações no código. No entanto, permite apenas alterar a ordem nas quais as políticas são aplicadas, i.e. a sequência em que o método verifica o ambiente, o que acaba mudando a ordem de suas ações. Esta proposta é também adotada para evitar o conflito entre as decisões, visto que a primeira verificação acionada possui preferência sobre as ações.

As propostas GA-VAM e MO-VMP foram classificadas como *Pode Suportar* pois suas abordagens utilizam pesos fixos para representar a ordem de aplicação dos MOs. Eles adotam diferentes estratégias para transformar seus MOs em um mono-objetivo, como *Soma com Pesos* e *Média com Pesos* [102]. A manipulação do vetor de pesos, em tempo de execução, poderia prover a funcionalidade de *Personalização* dinamicamente. A distribuição das abordagens é representada na Figura 28.

Apesar da falta de propostas contemplando a personalização da prioridade de objetivos, observamos uma tendência na utilização de abordagens com pesos para conversão de MOs em problemas mono-objetivos. No entanto, esta decisão traz algumas preocupações sobre o equilíbrio das soluções, uma vez que pode gerar situações de desequilíbrio entre as alocações. Estas situações acontecem quando uma minoria de alocações é altamente qualificadas e provê uma falsa impressão de que é bem qualificada, escondendo uma série de alocações em situações não recomendadas. De qualquer forma, estas situações podem ser evitadas aplicando-se o conceito de Dominância, que evita possíveis degradações.

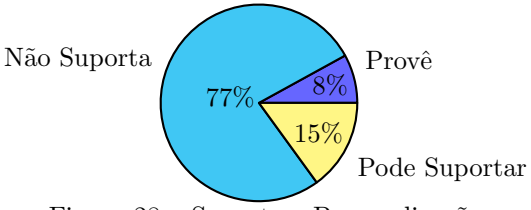

Figura 28 – Suporte a Personalização

A.3.3 Afinidade

Apenas AAGA explicitamente alega considerar afinidade entre elementos da Nuvem para auxiliar no processo de VMP. Ela usa a afinidade de tráfego entre VMs para organizá-las em grupos e então alocá-las nas PMs. O método não considera mais de um fator de afinidade para organizar as VMs, característica que restringe a sua aplicabilidade, pois geralmente Nuvens possuem múltiplas afinidades. Depois do agrupamento, um algoritmo Guloso [103] é aplicado para distribuir as VMs entre as PMs. Apesar de alegar que vários algoritmos gulosos podem ser utilizados no problema de alocação, os autores não deixam claro que algoritmo foi utilizado nos testes.

Ainda que não considere explicitamente afinidade, NA-VMP e VMPlanner também as usam em seus métodos. Ambas propõem construir agrupamentos de VMs que se intercomunicam, porém, aplicando métodos de agrupamento distintos. Mesmo assim, eles não suportam outros tipos de afinidade.

A.3.4 Consciência do Ambiente

Apesar da falta de propostas que explicitamente suportam este aspecto, algumas delas poderiam ser adaptadas para considerar verificações externas. Por exemplo, este aspecto poderia ser implementado como: políticas de ambiente nas propostas Snooze e Auto-RDP, funções de utilidade em GA-VAM ou novos objetivos em MO-RAO.

A.3.5 Agnosticismo

Neste, focamos em trabalhos que suportam MOs em suas abordagens. Apesar de alegarem considerar MOs, MO-VMP e MO-VMM utilizam um número limitado de funções analíticas inseridas em uma função *Fuzzy*, que converte o problema *Puramente MO* em um problema *mono-objetivo*. No entanto, os autores não explicam como é possível estender sua proposta a fim de atender outros objetivos.

In GA-VAM, Abdul et al. representam seus MOs na forma de dois vetores de utilidade, com: (i) funções e (ii) pesos. Esta estratégia transforma o problema de MO em mono-objetivo efetuando uma *Soma com Pesos* de todos os objetivos e busca maximizar este resultado. Ainda, como comentado na Seção A.3.2, isto pode levar a soluções desbalanceadas. Por outro lado, MO-RAO usa um conjunto de in-

dicadores[1] de MO para comparar e selecionar os resultados. Assim, selecionando resultados não-dominados, esta estratégia evita a adotação de soluções desbalanceadas e, ainda, melhora a convergência do método.

Apesar da abordagem de VMP como um Problema de Otimização de Múltiplos Objetivos (POMO) já ser bem aceita, nenhuma das propostas analisadas era realmente agnóstica na forma que esperávamos, i.e. não provêem a possibilidade de implementar qualquer tipo de objetivo. A abordagem agnóstica de MO não apenas une ambos os campos (VMP e POMO), mas abstrai e unifica questões importantes — como SLAs, afinidade de elementos e consciência do ambiente — por representá-los como uma única e formal unidade: um objetivo. Não menos importante, objetivos devem ser implementados separadamente dos métodos de *Seleção* e *Geração de Soluções*, a fim de permitir uma maior flexibilidade. Ainda, enfatizamos sua utilização em conjunto com a *Personalização*, — priorização de objetivos —, a qual mitiga problema como conflitos, responsividade e flexibilidade nos ambiente dinâmicos da Nuvem.

A.3.6 SLA Constraints

Entre as propostas classificadas com *Suportam* SLAs, identificamos 3 diferentes abordagens utilizadas para tratar os SLAs: (i) como métricas a serem monitoradas durante as alterações na Nuvem, a fim de se evitar violações [44, 100], (ii) um conjunto de métricas de qualidade a serem avaliadas individualmente [96] e (iii) como uma única métrica de qualidade a ser maximizada [83]. EE-AVM usa SLAs como dados de monitoração para atingir seus objetivos sem degradar a Nuvem. Em U-RAM, cada VM possui um nível mínimo de qualidade que deve ser atendido para satisfazer as restrições. EE-VMP, ao contrário, usa SLA para ajustar a quantidade de recursos ociosos reservados, mantendo um suprimento mínimo apenas para tolerar eventuais picos de demanda. No entanto, estas restrições são definidas globalmente e previnem que o gerente trate grupos de VMs de forma distinta. A Figura 29 sumariza o suporte de SLA através das propostas.

[1] Hit Ratio, Accuracy Ratio, GD e $D1_R$, de [95].

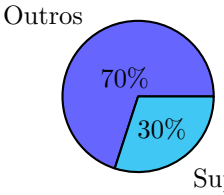

Figura 29 – Suporte a SLA

A.3.7 Acoplamento Tecnológico

A maioria das propostas analisadas são associadas a algum problema ou tecnologia específica. Algumas abordagens, como MO-RAO, desassociam seus métodos de seus objetivos, provendo flexibilidade na mudança de objetivos quando necessário.

A maior parte dos métodos está profundamente ligada aos objetivos e problemas [44, 85], necessitando de muitas mudanças para suportar outros objetivos. Algumas soluções são desenvolvidas para ambientes específicos, para atender serviços específicos, como o provimento de Desktops Virtuais [96] e HPC [85, 86, 100]. Outros casos demandam o acesso e modificação a VMs e PMs para poderem ser implantados [83,96]. Esta prática invasiva frequentemente é impossível ou não é bem vista pelos administradores e clientes da Nuvem.

A.3.8 Otimicidade

Baseado na classificação proposta somente MO-VMP foi classificado como *Heurísticas*. Esta proposta utiliza um *Algoritmo Genético de Agrupamento* para explorar e avaliar novos cenários: se um destes exceder o limiar de otimicidade pré-estabelecido, a busca para e a solução é adotada. A desvantagem desta abordagem está na heurística de *Geração de Soluções*, a qual, devido ao forte acoplamento com seus objetivos limitados, dificulta a sua aplicabilidade. A Figura 30 representa a classificação das propostas.

Da mesma maneira, GA-VAM e EO-VMP, classificados como *Limiares*, apresentam séries de restrições de qualidade que são analisadas no processo de avaliação de um cenário. Porém, ao invés de tentar maximizar seus valores, eles avaliam se eles estão acima de um limiar pré-estabelecido, verificando se vale a pena adotar o cenário. Apesar de prover bons resultados, estas propostas são aplicadas apenas

em cenários específicos, restringindo sua aplicabilidade em Nuvens que proveem *Infraestrutura como um Serviço*.

Por outro lado, a fim de decidir a melhor alocação para novas VMs, EO-VMP utiliza o histórico de utilização dos usuários para reduzir as configurações das VMs requisitadas, reduzindo o custo de pago pelo cliente. No entanto, antes de reduzir as configurações, seu método verifica se houve falhas de predição no passado; caso positivo, mantém a configuração intacta.

A diferença principal entre as propostas classificadas como Semi-Ótimas são: (i) como elas constroem o conjunto de soluções e (ii) o tamanho deste conjunto. MO-VMM constrói um conjunto pequeno com todos as possíveis alocações que a nova VM pode adotar e, em seguida, compara todo o conjunto baseando-se em funções de avaliação e seleciona a solução que oferecer o maior benefício. Enquanto isto, MO-RAO utiliza uma *Busca com Localidades Variadas* para construir um conjunto de Fronteira de Pareto, composto pelas maiores soluções não-dominadas encontradas. Cada solução é resultado de uma busca em uma localidade diferente.

A categoria *Outros* pode ser dividida em dois grupos: *Organizadores* e *Políticas*. *Organizadores* organizam um conjunto de VMs e as distribuem usando variações do algoritmo de Melhor Encaixe (Best Fit), e.g. Best Fit Decreasing, Quadratics Assignment or Forecast-based Power Aware Best Fit Decreasing. O segundo, ao contrário, baseia-se em políticas para manter a Nuvem em um estado considerado saudável.

Considerando todos os trabalhos analisados, podemos definir seus métodos de avaliação de otimicidade de dois modos: (i) comparar a solução com todas as soluções possíveis, procurando pela melhor, ou (ii) criar uma função que retorna se o resultado é ótimo ou não.

O primeiro caso pode ser problemático quando o número de possibilidades é grande ou o processo de comparação é oneroso. Geralmente, quando não é possível gerar todas as possibilidades, um subconjunto de soluções é avaliado e uma solução ótima local é adotada.

A segunda solução torna-se impraticável dada a quantidade de variáveis e complexidade envolvida no problema. Assim, métodos apelam a conjuntos de restrições que avaliam se a solução é ótima o bastante para serem adotadas (Semi-Ótimas ou Local-Ótimas).

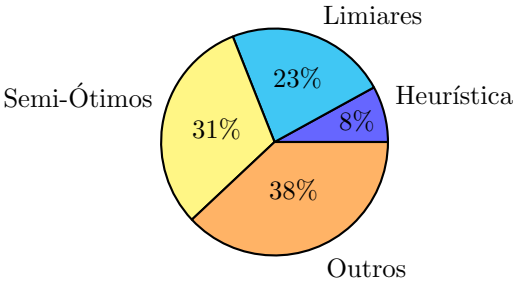

Figura 30 – Estratégias de Otimicidade

A.3.9 Escalabilidade

Apesar de ser uma preocupação comum entre as propostas, poucos trabalhos realizam uma análise de escalabilidade formal, descrevendo com notação *Big O*, e.g. AAGA, MO-VMM e MO-VMP. Suas análises descrevem a complexidade de cada parte da abordagem, no entanto, algumas variáveis são altamente acopladas com os problemas que estão lidando, e.g. a complexidade da rede, o que torna impraticável em ambientes distintos.

Não menos importante, GA-VAM deve ser ressaltado por mitigar a questão de escalabilidade, dividindo suas decisões entre diversos agentes, presentes na PMs da Nuvem. Esta estratégia permite um crescimento horizontal da solução, visto que cada nova PM leva consigo um agente da solução.

Na Tabela 8 destacamos a magnitude dos testes realizados em cada proposta, indicando a quantidade de VMs utilizadas nos mesmos.

A.3.10 Acessibilidade da Proposta

A maioria das propostas formalmente descrevem seus métodos através de equações, e poucas através de pseudo-código. As formas de acesso são detalhadas na Tabela 8 e a distribuição está graficamente representada na Figura 31.

Entre todas as propostas avaliadas, apenas Snooze compartilha o código fonte de sua proposta, em [93], como um projeto público, centralizando informações sobre documentação e provendo ferramentas para o desenvolvimento colaborativo da proposta.

Como uma observação final, notamos uma lacuna na utilização de linguagens formais de SLA, SLO, Objetivos e etc. Apesar da existência de algumas iniciativas, como SLAC e SCEL [97, 98], nenhuma proposta as adota, ou outra similar. A adoção em massa de linguagens formais, nestes casos, desenvolve o campo como um todo por melhorar a reprodutibilidade dos experimentos.

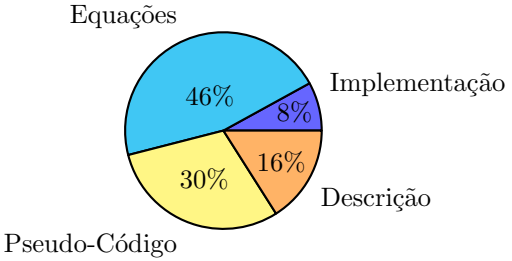

Figura 31 – Acesso ao Código

Proposta	Abordagem	Multi-Objetivos	Escopo	Personalização	Otimicidade	Consciência Externa	Afinidade	SLA	Coesão Tecnológica	Dados dos Testes	Escala dos Testes (#VMs)	Acessibilidade do Código
MO-VMP [37]	MOM	✓	P	M	H					Sim.	2000	Pseudo-Codigo
GA-VAM [83]	MOM	✓	H	M	T	✓		✓		Sim.	1000	Equações
AAGA [85]	BiP		O		-		✓			Sim.	56	Equações
MO-VMM [81]	MOM	✓	O	✓	S					Real	54	Equações
MO-RAO [18]	PMO	✓	O		S	✓			✓	Sim.	-	Pseudo-Codigo
EE-AVM [44]	BiP		H		-			✓		Sim.	290	Descrição
NA-VMP [99]	BiP		O		S					Sim.	3430	Equações
EE-VMP [100]	MOM		O		-			✓		Real	150	Equações
EO-VMP [84]	MOM		P		T					Sim.	-	Equações
Snooze [92]	MOM		O		-	✓		M		Real	500	Java [93]
Auto-DRP [34]	MOM		H		-	✓		M		Real	110	Descrição
VMPlanner [86]	BiP		O		-					Sim.	2000	Pseudo-Codigo
U-RAM [96]	PMO		P		T			✓		Sim.	112	Pseudo-Codigo

Tabela 8 – Aspectos Essenciais das Propostas de VMP

APÊNDICE B – Implementação do Framework

Nesta seção descrevemos como foi projetado e implementado o Framework Order@Cloud. Começamos explicando as principais classes e classes auxiliares. Em seguida, apresentamos as classes referentes ao Modelo de Nuvem apresentado na Seção 4.1 e, finalmente, as classes que gerenciam os elementos do modelo (i.e. Regras, Qualificadores e Custos). Os símbolos utilizados nos diagramas de classe são apresentados na Tabela 9. Por fim, a Tabela 10 apresenta um resumo de cada classe e suas funções. Encerramos apresentando alguns exemplos de implementação de: Regras, Qualificadores e Custos.

Símbolo	Significado
+	Atributos ou Funções Públicas
-	Atributos ou Funções Privadas
#	Atributos ou Funções Protegidas
Itálico	Atributos ou Funções Estáticas

Tabela 9 – Símbolos de Definição de Classes

O framework é representado pela classe OrderCloud, que, em seu método de construção, recebe o cenário atual da nuvem para ser analisada posteriormente. Esta classe possui os 3 métodos referentes a Organização, Provisionamento e Adaptação de cenários (Seção 4.2), como mostra a Figura 32. A classe *Scenario* é uma classe estática cuja principal função é auxiliar na adição e remoção de VMs dos cenários, efetuando todas as alterações necessárias antes e depois de cada operação, e.g. atualização das estatísticas do cenário. A classe *Cache* provê um local em comum para os elementos armazenarem e trocarem informações de forma rápida e transparente. Esta classe também possibilita o acesso às informações do cenário base para todos os elementos do framework, possibilitando possíveis comparações.

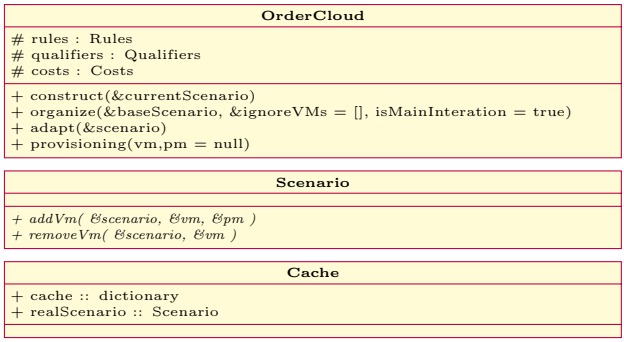

Figura 32 – Definição da Classe OrderCloud e Outras Classes Auxiliares

A fim de permitir uma fácil extensão do Framework, novos elementos do modelo de Nuvem são representados como novas classes. Estas devem, necessariamente, estender e implementar classes e interfaces pré-definidas, seguindo o padrão *Singleton*, i.e. só podem ser instanciadas uma única vez. Na Figura 33 as classes de exemplo *ExemploDeCusto*, *ExemploDeQualificador*, *ExemploDeRLC* e *ExemploDeRSC* representam, respectivamente, implementações de classes de Custos, Qualificadores, Regras Livres de Contexto e Regras Sensíveis ao Contexto. A Tabela 11 lista as classes que cada elemento deve implementar, a fim de ser aceito pelos seus gerenciadores.

Elemento	Estende	Implementa
Custo	Cost	InterfaceCost
Qualificador	Qualifier	InterfaceQualifier
Regra Livre de Contexto	Rule	RuleFreeOfContext
Regra Sensível ao Contexto	Rule	RuleSensitiveToContext

Tabela 11 – Listagem de Classes que devem ser implementadas e estendidas na criação de novos elementos.

Para organizar os elementos da Nuvem existem classes especiais que reúnem e interagem com eles, estes Gerentes (*handlers*) são apresentados na Figura 34. Assim que uma nova classe é criada, ela deve ser adicionada ao seu devido Gerente, que irá verificar se a classe implementa e estende as suas devidas classes. É responsabilidade deles, também, checar o retorno de suas avaliações, no caso dos Qualificadores.

A Figura 35 demonstra um exemplo da utilização do Framework, no qual o código é carregado (linha 1), os elementos são incluídos (linhas 2-4), um cenário é construído (linhas 5-9) e a classe é instanciada e executada (linhas 10 e 11). A resposta, presente na variável $retorno (linha 7), provê na forma de um *dicionário* as alocações a serem adotadas.

O código na Figura 36 é um exemplo de uma classe de Regra Livre de Contexto. Esta verifica se a PM alvo está apta a receber a VM em questão. Baseando-se nos dados de entrada, VM e PM alvo, a classe verifica junto aos dados do centro de dados, carregados na inicialização do framework, se a PM alvo tem acesso a todos os recursos que a VM necessita. Neste caso, são verificados o acesso a Rede e aos espaços de armazenamento.

Um exemplo de uma classe de Regras Sensível ao Contexto pode ser encontrado na Figura 37. Esta regra veta cenários os quais as VMs afins estão na mesma PM, evitando possíveis conflitos entre as VMs. Devido ao fato desta ser uma RSC, deve ser ressaltado o fato dela receber como argumento um cenário, e não uma VM e uma PM, como

Nome da Classe	Tipo	Função	Estende
OrderCloud	Regular	Classe principal, contem os Métodos de Organização, Provisionamento e Adaptação.	-
Scenario	Estática	Operar sobre os cenários, prove as ações de adição e remoção de VMs.	-
Cache	Estática	Armazena o cenário inicial e dados do ambiente para os elementos do modelo utilizarem.	-
Singleton	Abstrata	Concentra as funções das classes do tipo Singleton	-
Qualifier	Abstrata	Concentra as funções dos Qualificadores	Singleton
Cost	Abstrata	Concentra as funções das logicas de Custo	Singleton
Rule	Abstrata	Concentra as funções das RLC e RSC.	Singleton
InterfaceCost	Interface	Define funções e atributos obrigatórios para Custos	-
InterfaceQualifier	Interface	Define funções e atributos obrigatórios para Qualifiers	-
RuleFreeOfContext	Interface	Define funções e atributos obrigatórios para RLC.	-
RuleSensitiveToContext	Interface	Define funções e atributos obrigatórios para RSC.	-
HandlerSingleton	Abstrata	Concentra as funções de gerenciamento de elementos.	-
Costs	Singleton	Agrupa e gerencia os Custos do ambiente	HandlerSingleton
Qualifiers	Singleton	Agrupa e gerencia os Qualificadores do ambiente	HandlerSingleton
RulesFreeOfContext	Singleton	Agrupa e gerencia os RLC do ambiente	HandlerSingleton
RulesSensitiveToContext	Singleton	Agrupa e gerencia os RSC do ambiente	HandlerSingleton

Tabela 10 – Relação das Principais Classes do Framework e suas Funções

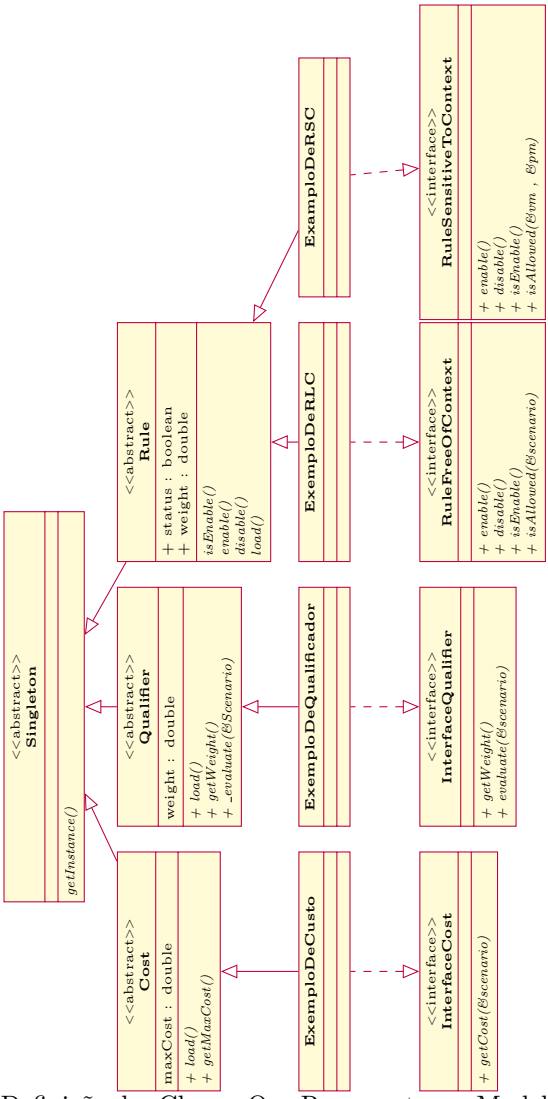

Figura 33 – Definição das Classes Que Representam o Modelo de Nuvem

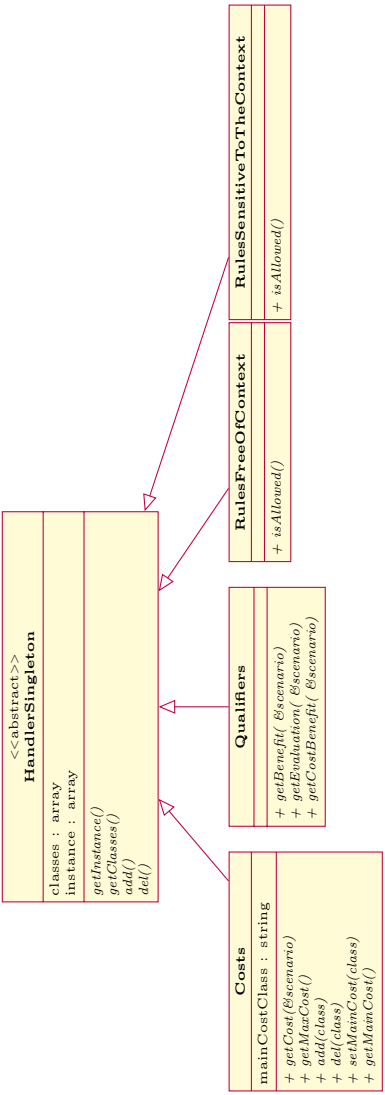

Figura 34 – Definição das Classes de Gerenciamento de Regras, Qualificadores e Custos

```
require_once "OrderAtCloud.phar";
require_once "regras.php";
require_once "qualificadores.php";
require_once "custos.php";
$scenario = [];
Scenario::addVm($scenario, 'VM1','PM1');
Scenario::addVm($scenario, 'VM2','PM1');
Scenario::addVm($scenario, 'VM3','PM2');
Scenario::addVm($scenario, 'VM4','PM2');
$oc = new OrderCloud($scenario);
$retorno = $oc->organize($scenario);
```

Figura 35 – Exemplo de utilização do Order@Cloud

```
class RfcRequirements extends Rule implements RuleFreeOfContext
{
    private static $data = [];
    static public function load()
        RfcRequirements::$data = MyDatacenter::getDataFromDatacenter();
    static function isAllowed(&$vm, &$pm) {
        $vmd = RfcRequirements::$data['vms'][$vm];
        $pmd = RfcRequirements::$data['pms'][$pm];
        foreach ($vmd['networks'] as $net) {
            $resp = array_search($net, $pmd['networks']);
            if ($resp === false) return false;
        }
        foreach ($vmd['datastores'] as $ds) {
            $resp = array_search($ds, $pmd['datastores']);
            if ($resp === false) return false;
        }
        return true;
    }
}
RulesFreeOfContext::add('RfcRequirements');
```

Figura 36 – Exemplo de Implementação de uma RLC

```
1  class RscAvoidAffinity extends Rule implements RuleSensitiveToContext
2  {
3      private static $relatedVms = ['VM1','VM2']
4      static function isAllowed(&$scenario) {
5          $pm1 = $scenario['vms'][$relatedVms[0]];
6          $pm2 = $scenario['vms'][$relatedVms[1]];
7          return $pm1 != $pm2;
8      }
9  }
10 RulesFreeOfContext::add('RscAvoidAffinity');
```

Figura 37 – Exemplo de Implementação de uma RSC

```
1  class CostMigrations extends Cost implements InterfaceCost{
2      static $maxCost = 20;
3      static function getCost(&$scenario) {
4          $count = 1;
5          foreach (Cache::$realScenario['vmp'] as $vm => $pm)
6              if ($scenario['vmp'][$vm] != $pm)
7                  $count++;
8          return $count;
9      }
10 }
11 Costs::add('CostMigrations');
```

Figura 38 – Exemplo de Implementação de um Custo

é o caso das RLC. Neste caso, a regra verifica se a PM que hospeda a VM1 é igual a PM que hospeda a VM2. Caso sejam iguais, o cenário é vetado.

Como exemplo de uma classe de Custo, apresentamos o código presente na Figura 38, a classe *CostMigrations*. Esta classe quantifica o custo de um cenário contabilizando a quantidade de VMs que foram migradas entre o cenário inicial e o cenário alvo, i.e. verifica a quantidade de alocações diferentes entre os dois cenários.

A Figura 39 exemplifica um Qualificador do ambiente. Dada uma listagem de serviços, ele verifica a quantidade de VMs do mesmo serviço que esta na mesma PM. Desta maneira, ele penaliza todas as alocações que sobrecarregam esta PM, i.e. se um serviço possui mais de 2 VMs em uma mesma PM, todas as VMs dentro desta PM serão penalizadas.

```
class QuaDistributeServices extends Qualifier implements InterfaceQualifier
{
    private static $services = ['srv1_', 'srv2_', 'srv3_']; //Prefix of the VMs
    static public function load() {
        $scenario = &Cache::$realScenario;
        $store = [];
        $store['vms'] = [];
        //Initial PM Count
        foreach (QuaDistributeServices::$services as $service)
            foreach ($scenario['rpm'] as $pm => $vms)
                $store['count'][$service][$pm] = 0;
        //Initial VM Count
        foreach ($scenario['vmp'] as $vm => $pm) {
            $store['eval'][$vm] = 1;
            foreach (QuaDistributeServices::$services as $service) {
                if (stripos($vm, $service) !== false) {
                    $store['vms'][$vm] = $service;
                    $store['services'][$service][] = $vm;
                    $store['count'][$service][$pm]++;
                }
            }
        }
        //Saving Evaluations
        foreach ($store['vms'] as $vm => $service) {
            $pm = $scenario['vmp'][$vm];
            $qtd = $store['count'][$service][$pm];
            $eval = QuaDistributeServices::getEval($qtd);
            $store['eval'][$vm] = $eval;
        }
        $scenario[OC_STORE]['QuaDistributeServices'] = $store;
    }
    static function evaluate(&$cvmp) {
        $store = &$cvmp[OC_STORE]['QuaDistributeServices'];
        if (!isset($cvmp[OC_LAST_REM_VM])) return $store['eval'];
        $vm = $cvmp[OC_LAST_ADD_VM];
        if(!isset($store['vms'][$vm] )) return $store['eval'];
        //Get Data
        $newPm = $cvmp[OC_LAST_ADD_PM];
        $oldPm = $cvmp[OC_LAST_REM_PM];
        $service = $store['vms'][$vm];

        //Update PM service Statistics
        $store['count'][$service][$oldPm]--;
        $store['count'][$service][$newPm]++;
        // update evaluations
        $vms = $store['services'][$service];
        foreach ($vms as $vm) {
            $pm = $cvmp['vmp'][$vm];
            if($pm != $newPm and $pm != $oldPm) continue;
            $qtd = $store['count'][$service][$pm];
            $store['eval'][$vm] = QuaDistributeServices::getEval($qtd);
        }
        return $store['eval'];
    }
    static function getEval($qtd) {
        if      ($qtd <= 1) $eval = 2;
        elseif ($qtd == 2) $eval = 1;
        else    $eval = 0.5;
        return $eval;
    }
}
Qualifiers::add('QuaDistributeServices');
```

Figura 39 – Exemplo de Implementação de um Qualificador