Pós-Graduação em Ciência da Computação

CARLOS JULIAN MENEZES ARAÚJO

Tomada de Decisão Multicritério em Infraestruturas como Serviço em Nuvem:
Uma Abordagem baseada em Modelos de Dependabilidade, Performabilidade e Custo

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2019

CARLOS JULIAN MENEZES ARAÚJO

Tomada de Decisão Multicritério em Infraestruturas como Serviço em Nuvem:
Uma Abordagem baseada em Modelos de Dependabilidade, Performabilidade e Custo

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como parte dos requisitos para obtenção do título de Doutor em Ciência da Computação.

Área de Concentração: Redes de Computadores e Sistemas Distribuídos.

Orientador: Prof. Dr. Paulo Romero Martins Maciel
Coorientador: Prof. Dr. Paulo Roberto Freire Cunha.

Recife
2019

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

A663t Araújo, Carlos Julian Menezes
Tomada de decisão multicritério em infraestruturas como serviço em nuvem: uma abordagem baseada em modelos de dependabilidade, performabilidade e custo / Carlos Julian Menezes Araújo. – 2019.
175 f.: il., fig., tab.

Orientador: Paulo Romero Martins Maciel.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2019.
Inclui referências e apêndices.

1. Redes de computadores. 2. Computação em nuvem. 3. Redes de Petri. I. Maciel, Paulo Romero Martins (orientador). II. Título.

004.6 CDD (23. ed.) UFPE- MEI 2019-055

Carlos Julian Menezes Araújo

"Tomada de Decisão Multicritério em Infraestruturas como Serviço em Nuvem: Uma Abordagem baseada em Modelos de Dependabilidade, Peformabilidade e Custo"

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

Aprovado em: 05/04/2019.

BANCA EXAMINADORA

_____

Prof.Dr. Nelson Souto Rosa
Centro de Informática/UFPE

_____

Prof. Dr. Ricardo Massa Ferreira Lima
Centro de Informática/UFPE

_____

Prof. Dr. Djamel Fawzi Hadj Sadok
Centro de Informática/UFPE

_____

Prof. Dr. Ricardo José Paiva de Britto Salgueiro
Departamento de Ciência da Computação e Estatística /UFS

_____

Prof. Dr. Artur Ziviani
Laboratório Nacional de Computação Científica/ LNCC

Decido esta tese a Deus, a minha família, aos meus amigos e a Sony pelo carinho e dedicação durante este percurso.

AGRADECIMENTOS

Sou imensamente agradecido a Deus por todas as oportunidades de aprendizado intelectual e moral que tive ao longo desta pesquisa. Sou grato por todas as pessoas que estiveram comigo nesse caminho. Um caminho recheado de desafios, alguns deles proporcionaram o exercício do autoconhecimento, outros indicaram a necessidade de desenvolver novas habilidades intelectuais. A paciência e a perseverança foram atributos essenciais nesta etapa da minha vida.

Agradeço ao professor Paulo Maciel, pela dedicação, paciência, conversas e conselhos durante todos esses anos. Essa história vem desde os primeiros contatos, antes mesmo da minha entrada no mestrado em 2007. Obrigado por todas as oportunidades e aprendizados que tive ao teu lado. Sou eternamente grato por tudo, professor.

Agradeço também aos professores: Ricardo Salgueiro (DCOMP-UFS), Nelson Rosa (CIn-UFPE), kelvin Dias (CIn-UFPE), Ricardo Massa (CIn-UFPE), Artur Ziviani (LNCC) e Djamel Sadok (CIn-UFPE) que estiveram comigo no processo avaliativo, contribuindo na melhoria deste trabalho, aconselhando e colocando observações importantes.

Agradeço a todo(a)s do grupo MoDCS pelo auxílio e companherismo. Agradeço também aos companheiros de jornada: Rubens Matos, Jamilson Dantas, Jean Araújo, Erica Sousa, Kalil Bispo, Rafael Macieira, José Guimarães e João Ferreira. Meu agradecimento especial a Vandi Alves, Matheus Torquato, Ermeson Carneiro e Gustavo Callou na condução desta pesquisa.

Agradeço a todo(a)s estudantes da UFRPE que sempre me incentivaram na conclusão desta pesquisa. Aos amigos que construí na UFRPE, Suzana Sampaio, Jeísa Domingues, Danilo Araújo e Sandra Xavier.

Aos amigos da Juventude Espírita Vicente de Paulo, em especial a Fábio Albuquerque, Luís Moreira, Melina Nunes, Camila Corrêa, João Paulo e aos amigos espirituais.

Aos meus grandes amigos do Ceará, Nayara Monteiro, Thiago Bandeira, Isaac Pinto, Jorge Luiz e Juliana Gonçalves, Samuel Ribeiro, Lucas Ricardo e Araújo Neto, que para mim muito representam.

Aos meus pais Vanvan e Corrinha, bem como as minhas irmãs Rafaella e Karyne e aos meus sobrinhos Daniel, Ryan e Carlos Henrique, pois mesmo distantes, me deram todo apoio e suporte necessário.

Ao bem mais precioso pela dedicação e amor, Dona Sônia, Nando Lima, Mimosinha, Josemir de Sapucaia, Suellen, Suellene, Mãe Suca, Thiago de Olinda, Maju, e em especial a minha esposa Sony, pelo companheirismo, amor, paciência e suporte.

Diz Santo Agostinho: "Não há lugar para sabedoria onde não há paciência."
(RUBIO, 1995)

RESUMO

A computação em nuvem é um paradigma que fornece serviços por meio da Internet. O paradigma foi influenciado por tecnologias anteriormente disponíveis (por exemplo, *cluster*, *peer-to-peer* e *grid computing*) e tem sido adotado por grandes organizações. Empresas como Google, Amazon, Microsoft e Facebook fizeram investimentos significativos em computação em nuvem e agora oferecem serviços com altos níveis de confiabilidade. A avaliação eficiente e precisa de infraestruturas baseada em nuvem é fundamental para garantir a continuidade dos negócios e de serviços ininterruptos, tanto quanto possível. Esta pesquisa de doutorado desenvolveu uma abordagem baseada em modelos de dependabilidade, performabilidade, custos e decisão para auxiliar usuários de infraestruturas como serviço em nuvem pública e privada no processo de tomada de decisão. O trabalho utiliza redes de Petri estocásticas (SPNs) e diagrama de blocos de confiabilidade para representar e avaliar dependabilidade e performabilidade para diferentes infraestruturas em nuvem. Os modelos de custos propostos possibilitam a avaliação dos custos considerando a aquisição, manutenção e operação do ambiente de computação em nuvem. A ferramenta MiPACE foi desenvolvida para dar suporte no planejamento de cenários de infraestruturas como serviço na nuvem e para auxiliar no processo de tomada de decisão. Estudos de caso demostraram a utilidade da abordagem proposta nesta pesquisa em orientar usuários e gestores de sistemas de nuvem no processo de tomada de decisão.

Palavras-chaves: Computação em nuvem. Dependabilidade. Performabilidade. Redes de Petri. Diagramas de Blocos de Confiabilidade. Tomada de decisão.

ABSTRACT

Cloud computing is a paradigm that provides services over the Internet. The standard has been influenced by previously available technologies (e.g., cluster, peer-to-peer and grid computing) and has been adopted by large organizations. Companies like Google, Amazon, Microsoft, and Facebook have made significant investments in cloud computing and now offer services with high levels of reliability. Efficient and accurate assessment of cloud-based infrastructures is critical to ensure business continuity and uninterrupted services as much as possible. This Ph.D. research has developed an approach based on dependability, performability, cost, and decision models to assist infrastructure users as a public and private cloud service in the decision-making process. The work uses stochastic Petri nets (SPNs) and reliability block diagram to represent and evaluate dependability and performability for different infrastructures in the cloud. The proposed cost models make it possible to the acquisition, maintenance, and operation of the cloud computing environment. The MiPACE tool was developed to support the planning of infrastructure scenarios as a service in the cloud and to aid in the decision-making process. Case studies have demonstrated the usefulness of the approach proposed in this research to guide users and managers of cloud systems in the decision-making process

Keywords: Cloud Computing. Dependability. Performability. Petri net. Reliability Block Diagram. Making Decision.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A/A	ativo/ativo
AHP	*Analytic Hierarchy Process*
ARPA	Agência de Projetos de Pesquisa Avançada
BI	*business intelligence*
BSC	*Balanced Scorecard*
COA	disponibilidade orientada à capacidade
CTMC	Cadeias de Markov de Tempo Contínuo
DoE	Design of Experiments
FAHP	*Fuzzy Analytical Hierarchy Process*
FDM	*Fuzzy Delphi Method*
FT	Árvore de Falhas
GSPN	Redes de Petri Estocástica Generalizada
HD	*Hard Disk*
HTTP	*HyperText Transfer Protocol*
Hw	*Hardware*
IaaS	Infraestrutura como Serviço
ITU	*International Telecommunication Union*
KRCC	*Kendall rank correlation coefficient*
MC	Cadeias de Markov
MiPACE	*A multi-criteria tool for planning and analysis of cloud environments*
Mng	Servidor de Gerenciamento
MTTF	Tempo Médio de Defeito
MTTR	Tempo Médio de Reparo
N/R	Sem redundância
NAHP	*Neutrosophic Analytic Hierarchy Process*
NASA	*National Aeronautics and Space Administration*
NIST	*National Institute of Standards and Technology*
PaaS	Plataforma como Serviço

QoS	Qualidade de Serviço
RBD	Diagrama de Bloco de Confiabilidade
RdP	Redes de Petri
RI	importância de confiabilidade
RS	Serviço de Reparo
RT	Tipo de Redundância
SaaS	*Software* como Serviço
SAN	*Storage Area Network*
SLA	Nível de Acordo de Serviço
SO	Sistema Operacional
SPN	Redes de Petri Estocástica
SRN	*Stochastic Reward Nets*
SSH	*Secure Shell*
TCO	Custo Total de Propriedade
TI	Tecnologia da Informação
TICs	Tecnologias da Informação e Comunicação
TMR	redundância modular tripla
TOPSIS	*The Technique for Order of Preference by Similarity to Ideal Solution*
VIM	*Virtual Infrastructure Manager*
VM	Máquina Virtual
VMs	Máquinas Virtuais

LISTA DE SÍMBOLOS

Ep	preço da energia
Ec	consumo de energia
Lb_{Up}	custo da hora trabalhada em operação
Lb_{Dw}	custo da hora trabalhada em manutenção
N	número de nós
T	período de tempo alocado
A_v	disponibilidade da infraestrutura como serviço
C_{ut}	custos de utilização
C_{mt}	custos de manutenção
C_{op}	custos de operação
TC_e	custo total estimado
$\sum L_c$	somatório dos custos dos componentes da infraestrutura
D_{wt}	indisponibilidade da Infraestrutura como Serviço (IaaS)
Lb_{Dwt}	custo das horas trabalhadas em atividades de manutenção
$\sum \mathrm{C}_{sub}$	somatório dos custos associados a substituições de componentes na IaaS
Lb_{Up}	custo das horas trabalhadas em atividades operacionais
V_r	valor por hora paga pelo provedor do serviço na nuvem para o usuário
V_{vio}	valor monetário relacionado à violação de qualidade
COA	disponibilidade orientada à capacidade
MPC	capacidade máxima de operação do sistema
π_i	disponibilidade estacionária no estado i
US	conjunto de todos os estados de disponibilidade
pci	capacidade de operação do estado si

SUMÁRIO

1 INTRODUÇÃO

Este capítulo apresenta uma breve introdução sobre computação em nuvem, destacando a sua origem, os modelos de serviço, aspectos de dependabilidade, bem como, a importância do uso de modelos para representar e avaliar serviços computacionais em nuvem. Em seguida, são expostas a justificativa, o problema, a hipótese e os objetivos do trabalho proposto. Logo após, chega-se, então, a uma visão da estrutura da tese.

1.1 CONTEXTO

O conceito de "serviços de computação" foi formulado na década de 1960, quando os *mainframes* eram alugados em pacotes de *hardware* e *software* (TIGRE; NORONHA, 2013). Em 1966, o engenheiro Douglass Parkhill descreveu em seu livro "*The Challenge of the Computer Utility*" a ideia de computação como um serviço público, no qual muitos usuários estariam remotamente conectados através de *links* de comunicação a uma instalação de computação central (HILL et al., 2013). Mais tarde, em 1969, Leonard Kleinrock um dos cientistas da Agência de Projetos de Pesquisa Avançada (ARPA) afirmou:

> *"A partir de agora, as redes de computadores ainda estão em sua infância. Mas à medida que crescem e se tornam mais sofisticadas, provavelmente veremos a propagação de 'computação utilitária' que, como atualmente em concessionárias de eletricidade e de telefone, atenderá casas individuais e escritórios em todo o país"* (HILL et al., 2013, p.1).

A afirmação de Kleinrock passa então a ser realidade com a difusão da Internet nos anos 90, quando as pessoas começam a se conectar por meio de *e-mails*, mensagens instantâneas, chamadas por vídeo, realizam compras *on-line* (*e-commerce*), fazem o uso de fóruns e outros. Essa popularização, mais tarde, propiciou o surgimento de um novo conceito, o da "computação em nuvem", que, por sua vez, pode ser definida como um modelo de computação que apresenta recursos escaláveis e oferece serviços virtualizados através da Internet (FURHT, 2013; RAJARAMAN, 2018). A seguir, outra definição de computação em nuvem foi elaborada pelo *National Institute of Standards and Technology* (NIST):

> *"[...] computação em nuvem apresenta-se como um modelo que possibilita ubiquidade, de modo conveniente, uma rede sob demanda acessível que permite o compartilhamento a um conjunto de recursos computacionais (por exemplo, redes, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente adquiridos e liberados com mínimo esforço gerencial ou interação com o provedor de serviços"* (DEKA, 2014, p.1).

A computação em nuvem alcançou considerável importância como um paradigma de computação e tornou-se amplamente utilizado por diversos negócios (bancos, *e-commerce*, *streaming* de vídeo, bibliotecas digitais, entre outros) (REHMAN; HUSSAIN; HUSSAIN, 2012a;

CHANG, 2015; MARINESCU, 2017). Esse paradigma vem proporcionando um crescimento contínuo no número de serviços oferecidos em nuvem. E os seus modelos, assim, podem ser apresentados como: (i) Infraestrutura como Serviço (IaaS), (ii) Plataforma como Serviço (PaaS) e (iii) *Software* como Serviço (SaaS). Em IaaS, as requisições dos usuários são distribuídas em instâncias de Máquinas Virtuais (VMs). Quando as instâncias das VMs são implantadas, faz-se necessário o provisionamento da CPU, RAM e armazenamento do disco. Esse processo de virtualização permite que as empresas aumentem suas infraestruturas virtuais, conforme a demanda. Podem ser citados dois exemplos de IaaS em nuvem: Amazon EC2 (Amazon, 2018) e Google Cloud (Google, 2018a). Na PaaS, por sua vez, os usuários podem implementar suas aplicações utilizando linguagens de programação e ferramentas que suportem serviços em nuvem. São exemplos de PaaS: Microsoft Azure (Microsoft, 2018) e Google Cloud Platform (Google, 2018b). Por outro lado, o SaaS é capaz de oferecer aos usuários aplicações que possam ser executadas na infraestrutura de computação em nuvem. GoogleDocs (Google, 2018c) e Gmail (Google, 2018d) são exemplos de *softwares* como serviço na nuvem.

Aplicações críticas implementadas em infraestruturas de computação em nuvem que demandam alta disponibilidade, podem resultar em elevados custos. Dessa forma, o desafio para empresas que oferecem serviços dos tipos IaaS, PaaS e SaaS na nuvem é disponibilizar ambientes computacionais que possam garantir alta disponibilidade desses serviços, mesmo diante de interrupções (ex.: interrupção de energia, falha de *hardware*, falha de *software*, entre outras), ao mais baixo custo possível (MARINESCU, 2017; VARGHESE; BUYYA, 2018). A alta disponibilidade, por sua vez, pode ser alcançada através de mecanismos que tratem as interrupções dos serviços, como por exemplo, redundância de *hardware* e/ou *software*, pois o custo da indisponibilidade (*downtime*) pode ser mais elevado do que os investimentos em tais mecanismos. Nota-se que os mecanismos de tratamento de interrupções podem evitar perda de dados e/ou diminuir o tempo para a recuperação dos serviços após uma interrupção (AHSON; ILYAS, 2010; BAUER; ADAMS, 2012).

Diversos tipos de formalismos, tais como cadeias de Markov, redes de Petri (MACIEL; LINS; CUNHA, 1996; MACIEL et al., 2012), diagrama de blocos de confiabilidade e árvore de falha (BOLCH et al., 2006) têm sido amplamente utilizados na modelagem, simulação e análise dos variados tipos de sistemas (SOUSA et al., 2017; ANDRADE et al., 2017). Esses modelos, por sua vez, são baseados em fundamentos matemáticos, possuem uma semântica precisa e fornecem suporte para verificações de propriedades qualitativas e quantitativas (POPOVA-ZEUGMANN, 2013; COSTA, 2007). Os modelos formais podem ser utilizados na representação de projetos de sistemas, planejamento de capacidade e gerenciamento, além de auxiliarem no processo para tomada de decisão.

Grande parte das atividades de planejamento e gerenciamento consistem na análise e tomada de decisões que visam escolher uma ou mais soluções que atendam as expectativas e interesses no processo decisório (ARENALES; ARMENTANO; MORABITO, 2007; ALMEIDA

et al., 2016). Os provedores de serviços em nuvem podem oferecer diversas possibilidades de configurações em infraestruturas, plataformas e *softwares* como serviço. As infraestruturas podem se diferenciar umas das outras considerando preço, disponibilidade, confiabilidade, escalabilidade, segurança, indisponibilidade, entre outros atributos (ALABOOL et al., 2018). Dessa forma, os usuários deparam-se com uma grande diversidade de serviços e com a necessidade de tomar uma decisão considerando diversos atributos do serviço. Essa tarefa pode se tornar complexa quando os usuários de serviços em nuvem desejam escolher uma solução que melhor se adeque às suas necessidades, como, por exemplo, limitação de orçamento (KAVIS, 2014).

1.2 JUSTIFICATIVA E RELEVÂNCIA

Com a evolução da tecnologia, a utilização de sistemas computacionais tornou-se frequente no dia a dia da sociedade. Instituições como bancos, universidades e hospitais estão migrando gradualmente seus centros de dados (*data centers*) para o modelo em nuvem. Esse modelo de computação vem crescendo e revolucionando as Tecnologias da Informação e Comunicação (TICs) através do provisionamento de recursos com o modelo pago-pelo-uso (*pay-as-you-go*) [1]. Assim, o usuário pode especificar a quantidade necessária de recursos para seu serviço. O modelo computacional em nuvem vem modificando cada vez mais o modo de gerir as instituições públicas e privadas. De acordo com a Gartner, a modernização e consolidação dos *data centers* devem incluir serviços de *big data*, aplicativos de negócios e análises baseados em sistemas de *data warehousing*, *business intelligence* (BI) e relatórios analíticos (LANEY; JAIN, 2017).

Os maiores desafios enfrentados por empresas que já implantaram e utilizam infraestrutura em nuvem são cumprir as normas legais e regulamentares (*compliance*), custos, desempenho, gerenciamento dos múltiplos serviços e segurança (RIGHTSCALE, 2017). Ademais, as empresas provedoras de infraestruturas em nuvem também são desafiadas a oferecer soluções que sejam capazes de garantir alta disponibilidade, mesmo diante de interrupções de energia, falhas no *link* de comunicação, falhas em *hardware*, *software*, entre outros. A computação em nuvem através dos conceitos de computação distribuída, computação em *grid* e *cluster* têm favorecido o desenvolvimento de serviços com alta disponibilidade (THODGE, 2018). Esses serviços, no entanto, necessitam de mecanismos de redundância via *software* e/ou *hardware* para, por exemplo, previnir ou minimizar danos que possam ser gerados devido a interrupções de um serviço. Assim, a análise de mecanismos de redundâncias é de fundamental importância no planejamento de infraestruturas como serviço em nuvem.

Os modelos representados por diagrama de bloco de confiabilidade (RBD) e redes de Petri estocástica (SPN) podem auxiliar usuários de ambientes de computação em

[1] Modelo de negócio onde o usuário paga à medida que utiliza um determinado serviço (VERAS, 2013).

nuvem no planejamento e avaliação de infraestruturas como serviços em nuvem. Para isso, é necessário considerar parâmetros que possibilitem a caracterização de cada IaaS. Métricas de dependabilidade aliadas aos mecanismos de redundância podem ser usados como parâmetros que identifiquem os níveis de disponibilidade e/ou confiabilidade das IaaS. Como resultado a partir desses modelos, é possível apresentar uma comparação das diversas IaaS (ANDRADE et al., 2017; MATOS et al., 2017; SOUSA et al., 2017).

A diversidade de soluções e parâmetros que caracterizam os serviços em nuvem podem aumentar o risco na escolha de uma IaaS. Pesquisas têm sido desenvolvidas por pesquisadores da indústria e academia a cerca do uso de métodos multicritério no apoio a decisão (ZOPOUNIDIS; PARDALOS, 2010; ALMEIDA et al., 2016; SACHDEVA et al., 2016; ALABOOL et al., 2018). Por sua vez, fazer escolhas diante de um conjunto de alternativas e multicritérios tem sido um problema encontrado em diversas áreas e atividades públicas e privadas (LIRA; CÂNDIDO, 2013; OLIVEIRA; MARTINS, 2015; HOSSEINIAN-FAR; RAMACHANDRAN; SARWAR, 2017). Assim, torna-se relevante adotar métodos multicritérios para apoiar na análise e escolha daquela solução considerada a mais satisfatória.

Tomar decisões a cerca de qual serviço em nuvem adotar, requer a definição de critérios e objetivos. Usuários que não dispõem de conhecimentos relacionados à dependabilidade do serviço podem também apresentar dificuldades no momento da tomada de decisão. Critérios como disponibilidade, confiabilidade e custo são importantes no processo decisório. Nesse sentido, modelos de decisão podem ser úteis a usuários de infraestruras em nuvem na escolha de uma alternativa. Vários métodos têm sido utilizados para avaliar um conjunto de critérios, tais como: agrupamento de dados, *Analytic Hierarchy Process* (*Analytic Hierarchy Process* (AHP)), *Technique for Order of Preference by Similarity to Ideal Solution* (*The Technique for Order of Preference by Similarity to Ideal Solution* (TOPSIS)), redes neurais, algoritmos genéticos e redes bayesianas (JAIN; DUBES, 1988; RUSSEL; NORVIG, 2004; ZOPOUNIDIS; PARDALOS, 2010).

O planejamento de avaliação de infraestruturas de computação em nuvem abrange a proposição e o uso de vários métodos. Alguns trabalhos desenvolvidos na academia apresentam o uso de técnicas para o planejamento e avaliação de serviços na nuvem (SOUSA et al., 2015; SOUSA et al., 2017; MATOS et al., 2017). Outros trabalhos estão inseridos no auxílio do processo decisório (LEE; SEO, 2016; SACHDEVA et al., 2016; DING et al., 2017a; AL-JABRI et al., 2018). O uso de métodos e modelos é importante para a avaliação de dependabilidade, custo e desempenho de infraestruturas na nuvem, já que tal avaliação pode proporcionar melhorias na qualidade do serviço provido e no planejamento. Ademais, a utilização de modelos para representar o funcionamento de ambiente computacionais possibilita análises confiáveis a um custo baixo, visto que não é necessário construir um sistema real para analisá-lo.

1.3 PROBLEMA E HIPÓTESE

O problema fundamental tratado nesta tese é abordado na seguinte pergunta: como auxiliar usuários no processo de seleção de infraestruturas como serviço em nuvem pública e privada considerando requisitos como restrição de orçamento, confiabilidade, desempenho, garantindo níveis aceitáveis de disponibilidade?

Com a finalidade de tratar um tipo de problema de decisão multicritério, a seguinte hipótese foi formulada: modelos de dependabilidade, performabilidade e custo aliados a modelos de tomada de decisão auxiliam usuários na análise e escolha de infraestruturas como serviço em nuvem pública e privada.

1.4 OBJETIVO

A computação em nuvem vem modificando a indústria de TI por meio dos modelos de serviço na nuvem (por exemplo, IaaS, PaaS e SaaS). Organizações públicas e privadas cada vez mais estão adotando esses modelos de computação em seus processos de negócio. Como resultado, os provedores de infraestruturas como serviço na nuvem buscam oferecer uma vasta variedade de IaaS que possam estar dentro dos requisitos dos usuários. Por outro lado, os usuários se deparam com as IaaS que oferecem diferentes níveis de qualidade de serviço (QoS) e preços correspondentes as configurações do ambiente (*hardware* e *software*), assim como, níveis de acordo de serviço (SLA). Nesse contexto, este trabalho propõe uma abordagem baseada em modelos de dependabilidade, performabilidade, custos e decisão para auxiliar usuários de infraestruturas como serviço em nuvem pública e privada no processo de tomada de decisão.

A abordagem proposta leva em conta uma estratégia de modelagem hierárquica, que considera as vantagens das redes de Petri estocásticas (SPN) e diagramas de blocos de confiabilidade (RBD) para avaliar a disponibilidade, disponibilidade orientada a capacidade, confiabilidade e utilização de recursos das infraestruturas como serviço. Os custos foram estimados através de equações visando aproximar os custos de utilização dos equipamentos do ambiente de TI, de equipamentos redundantes, da substituição de equipamentos, da equipe técnica e de manutenção. Para a tomada de decisão, utilizaram-se métodos de agrupamento de dados com funções multicritérios baseado em medidas de similaridade e no conceito de conjunto de soluções não dominadas, visando atender as especificações e interesse dos usuários.

Adotou-se, ainda, neste trabalho, a técnica do processo de agrupamento de dados, visto que esta pode ser utilizada para identificar um conjunto de soluções considerando funções multicritérios. A partir da identificação dos parâmetros desejados, o método de agrupamento pode identificar um conjunto de IaaS dentro das necessidades do usuário. As medidas de dissimilaridade ou similaridade adotadas no método foram distância Euclidiana, Minkowski e Manhattan (HARINDRANATH G., 2002). Além disso, com o intuito

de ampliar as opções de IaaS selecionadas, foi utilizada a abordagem de dominância de Pareto, visto que o postulado de Edgeworth-Pareto tem sido base para o desenvolvimento da teoria de otimização multicritérios (GOH; TAN, 2009).

No que concerne aos objetivos específicos, apontam-se os seguintes:

- definir uma estratégia que auxilie usuários de infraestruturas como serviço de nuvem no processo de tomada de decisão;
- propor um conjunto de modelos de dependabilidade e performabilidade para estimar a disponibilidade, confiabilidade, utilização dos recursos das IaaS em nuvem;
- estimar os custos das IaaS considerando a utilização dos equipamentos do ambiente de TI, redundância e substituição de equipamentos, equipe técnica e manutenção;
- adotar um método que realize o ranqueamento das IaaS por meio de funções multicritérios;
- implementar uma ferramenta para auxiliar o processo de tomada de decisão das IaaS modeladas e avaliadas.

Este trabalho está inserido em uma pesquisa de natureza aplicada, dirigida à solução de problemas específicos em infraestruturas de computação em nuvem pública e privada. Apresenta uma abordagem quantitativa do problema, visto que possui atributos mensuráveis através da observação sistemática, por exemplo, disponibilidade do serviço. Do ponto de vista de seu objetivo, baseia-se no método descritivo das características da infraestrutura computacional em nuvem, a partir do estabelecimento de relações entre variáveis de dependabilidade, performabilidade e custo. Quanto à forma de raciocínio, foi definida a linha indutiva, em que a generalização deriva de observações de casos representados através de modelos combinatoriais e baseados em espaço de estados, com a análise descritiva (MARCONI; LAKATOS, 2014). Quanto aos seus procedimentos técnicos, adotou-se a pesquisa bibliográfica (GIL, 2002), pois as fundamentações teóricas estão baseadas em literatura específica e experimental devido à observação dos efeitos que as variáveis produzem na disponiblidade e confiablidade das IaaS.

1.5 ORGANIZAÇÃO DA TESE

O presente trabalho encontra-se dividido em sete capítulos. Inicialmente, o primeiro capítulo apresenta a introdução, que compreende comentários iniciais, tema, problema, hipótese, objetivo, assim como a justificativa. O capítulo 2, por sua vez, apresenta uma revisão teórica sobre os assuntos abordados nesta tese: computação em nuvem, dependabilidade, performabilidade, redes de Petri, diagrama de bloco de confiabilidade, e, por fim, processo de agrupamento e otimização. Em seguida, o terceiro capítulo vem ilustrar os

trabalhos relacionados a esta tese, enquanto o quarto mostra a metodologia e os métodos adotados para o processo de tomada de decisão. Já o capítulo 5 apresenta os modelos de dependabilidade, performabilidade, custo e de decisão. O penúltimo capítulo, ou capítulo 6, descreve os estudos de caso com o intuito de apresentar a aplicabilidade da solução proposta. E, por fim, o sétimo capítulo apresenta as considerações finais, contribuições, limitações e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta um resumo das informações necessárias para a compreensão sobre este trabalho. Inicialmente, é apresentada uma visão geral sobre computação em nuvem. Depois disso, são apresentados os conceitos relacionados à dependabilidade, ao desempenho, à performabilidade, à modelagem, aos modelos com e sem espaço de estado, ao diagrama de bloco de confiabilidade e às redes de Petri. Por fim, são explicados conceitos sobre análise de agrupamento de dados e otimização.

2.1 COMPUTAÇÃO EM NUVEM

A definição de computação em nuvem como um modelo computacional de utilidade para serviços desse tipo é semelhante ao modelo de utilidade implantado para a eletricidade, água e os serviços de telecomunicação. Isto é, os usuários podem consumir a quantidade de serviços que desejarem, sempre que quiserem, e serão cobrados apenas pelos recursos utilizados (BAUER; ADAMS, 2012). O *National Institute of Standards and Technology's* (NIST) define três modelos de serviços para a computação em nuvem: *software* como serviço (SaaS), plataforma como serviço (PaaS) e infraestrutura como serviço (IaaS) (MELL; GRANCE, 2011). A Figura 1 ilustra a relação entre cada um desses modelos de serviço, assim como, abaixo são apresentadas as definições:

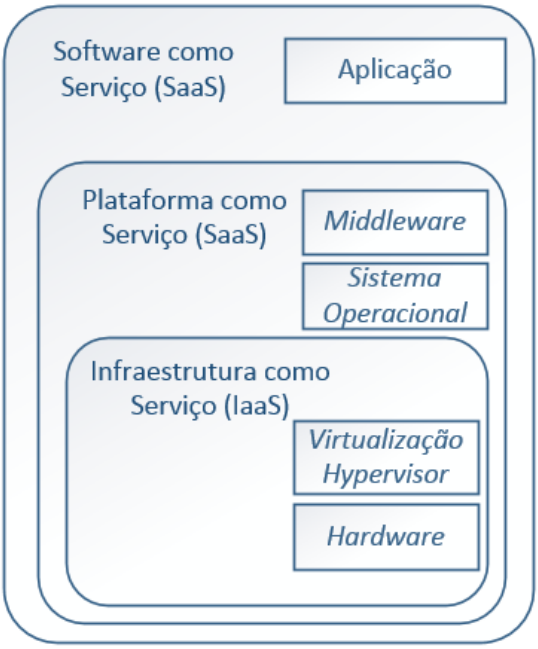


Figura 1 – Modelos de serviço.
Fonte: (BAUER; ADAMS, 2012).

- ***Software* como Serviço**. A capacidade fornecida ao usuário (consumidor) destina-se à utilização dos aplicativos do provedor rodando em uma infraestrutura de nuvem. As aplicações são acessíveis a partir de diversos dispositivos clientes, quer através de uma *interface*, como um navegador *Web* (por exemplo, *webmail*), quer por uma *interface* de programa. O usuário não administra e nem controla a infraestrutura de nuvem, incluindo rede, servidores, sistemas operacionais, armazenamento, ou mesmo capacidades de aplicativos individuais, com a possível exceção de configurações limitadas do aplicativo, específicas do usuário;

- **Plataforma como Serviço**. A capacidade fornecida ao usuário destina-se à infraestrutura criada ou comprada pelo usuário para a nuvem, desenvolvida por meio do uso de linguagens de programação, bibliotecas, coleção de serviços de infraestrutura de aplicativos (*middleware*) e ferramentas suportadas pelo provedor. O usuário não administra e nem controla a infraestrutura de nuvem, incluindo rede, servidores, sistemas operacionais ou armazenamento, mas tem controle sobre os aplicativos implementados e possivelmente sobre as configurações para o ambiente de hospedagem de aplicativos;

- **Infraestrutura como serviço**. A capacidade fornecida ao usuário destina-se ao provisionamento de máquinas virtuais, processamento, redes, armazenamento e outros recursos de computação fundamentais em que o consumidor é capaz de implementar e executar *softwares* arbitrários, que podem incluir sistemas operacionais e aplicativos. O usuário gerencia a infraestrutura de nuvem (*hypervisor*)[1], controla os sistemas operacionais, tarefas de armazenamento e aplicativos implementados. Possivelmente, possui controle limitado de componentes de rede selecionados, como por exemplo, *firewalls* do *host*.

O NIST reconhece quatro modelos de implantação de computação em nuvem. São eles:

- **Nuvem privada**. A infraestrutura de nuvem é provisionada para uso exclusivo por uma única organização, compreendendo múltiplos consumidores (por exemplo, unidades de negócio). A nuvem pode ser controlada, gerenciada e operada pela organização, por um terceiro, ou ainda por alguma combinação destes, podendo existir com ou sem premissas;

- **Nuvem comunitária**. A infraestrutura de nuvem é provisionada para uso exclusivo por uma comunidade específica de consumidores de organizações que tem preocupações comuns. Para ilustrar, usa-se como exemplo considerações referentes à missão,

[1] Hipervisores são sistemas de gerenciamento que permitem que muitas máquinas virtuais convidadas (VMs) usem os mesmos recursos (MARINESCU, 2017).

requisitos de segurança, política e *compliance*). A nuvem pode ser controlada, gerenciada e operada por uma ou mais das organizações na comunidade, um terceiro, ou alguma combinação, e pode existir com ou sem premissas;

- **Nuvem pública**. A infraestrutura de nuvem é provisionada para uso aberto ao público em geral. A nuvem pode ser controlada, gerenciada e operada por uma organização empresarial, acadêmica ou governamental, ou alguma combinação destes, existindo sob as premissas do fornecedor da nuvem;

- **Nuvem híbrida**. A infraestrutura de nuvem é uma composição de duas ou mais infraestruturas de nuvem distintas (privada, comunitária ou pública) que permanecem como entidades únicas, mas são unidas por tecnologia padronizada ou proprietária que permita a portabilidade de dados e aplicativos, como por exemplo, balanceamento de carga entre nuvens.

2.2 DEPENDABILIDADE

Devido à crescente expansão dos sistemas computacionais em nuvem, a dependabilidade tem se tornado um atributo de grande interesse no desenvolvimento, operação, implantação e manutenção de tais sistemas, visto que os serviços oferecidos precisam ser cada vez mais confiáveis (MACIEL et al., 2012). Os primeiros conceitos de dependabilidade foram, inicialmente, definidos por Laprie (LAPRIE, 1992), e, desde então, têm sido largamente utilizados tanto na academia como na indústria. Os sistemas que são analisados sob as métricas de dependabilidade são denominados de sistemas de funcionamento confiável.

De acordo com (AVIZIENIS et al., 2004), *"a dependabilidade de um sistema é a habilidade de evitar falhas de serviços que são mais frequentes ou mais severas do que o aceitável"*. A Figura 2 ilustra a taxonomia dos sistemas de funcionamento confiável. A seguir são detalhadas as definições.

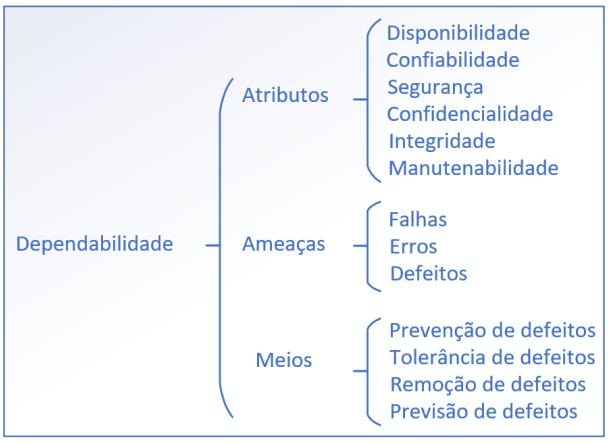


Figura 2 – Taxonomia dos Sistemas de Funcionamento Confiável.
Fonte: (AVIZIENIS et al., 2004).

A dependabilidade compreende, então, os seguintes atributos: disponibilidade (*availability*), probabilidade do sistema estar operacional num instante de tempo determinando; alternância de períodos de funcionamento e reparo; confiabilidade (*reliability*), capacidade de atender a especificação durante certo período de funcionamento e condicionado a estar operacional; segurança (*safety*), probabilidade do sistema ou estar operacional e executar sua função corretamente ou descontinuar suas funções de forma a não provocar dano a outros sistemas; confidencialidade (*Confidentiality*), ausência de divulgação não autorizada de informações; integridade (*Integrity*), ausência de alterações no sistema e manutenabilidade (*Maintainability*), capacidade de receber modificações e reparos. Os meios são agrupados em remoção de defeitos (*Fault Removal*), reduzir o número ou a severidade das falhas, tolerância de defeitos (*Fault Tolerance*), entregar o serviço correto mesmo na presença de falhas, prevenção de defeitos (*Fault Prevention*), prevenir a ocorrência ou introdução de falhas e previsão de defeitos (*Fault Forecasting*), estimar o número presente, a incidência futura e as consequências das falhas.

Por sua vez, as ameaças são agrupadas em defeitos (*Failures*), erros (*Errors*) e falha (*Faults*). Um defeito é definido como um desvio do comportamento do sistema em relação a sua especificação. A causa de uma falha é um erro. Já um erro é a porção do estado do sistema responsável por conduzi-lo a uma falha. Cada erro, assim, é causado pela ativação de um defeito.

A ocorrência de interrupções mínimas em serviços em nuvem podem causar tanto prejuízos financeiros quanto danos à credibilidade do negócio. Nesse sentido, os modelos computacionais em nuvem necessitam cada vez mais prover serviços com alta disponibilidade, visto que, necessitam manter os serviços disponíveis o máximo de tempo possível, mesmo na presença de interrupções. Assim sendo, este trabalho tem o foco em analisar aspectos de dependabilidade, especificamente o atributo de disponibilidade e confiabilidade de infraestruturas de computação em nuvem privada, considerando mecanismos de redundância. Os resultados obtidos a partir das métricas aferidas são utilizados para prover informações para os(as) projetistas e gestores de provedores de serviços em nuvem no processo de tomada de decisão.

2.3 DESEMPENHO

A computação em nuvem oferece aos usuários uma variedade de serviços, desde o *hardware* até o nível do aplicativos, utilizando o modelo *Pay-Per-Use* (Pago Pelo Uso). Outra característica importante, que os usuários podem se beneficiar, é a capacidade de ampliar e reduzir a infraestrutura de computação de acordo com os requisitos necessários pela aplicação e o limite de orçamento disponível. Desta feita, serviços baseados na nuvem podem oferecer fácil acesso a grandes infraestruturas distribuídas, além da possibilidade da personalização do ambiente de execução, o que pode proporcionar uma configuração adequada para hospedagem de um determinado serviço (THODGE, 2018).

Nesse contexto, a avaliação de desempenho é uma importante atividade em ambientes de nuvem, pois permite que analistas e usuários possam avaliar os efeitos gerados a partir de diferentes estratégias de gerenciamento de recursos. Adicionalmente, é possivel estimar os custos e benefícios correspondentes para cada ambiente disponível. A avaliação de desempenho pode ser realizada através de medições no sistema real ou por meio de modelos que representem as características e comportamento do sistema (LILJA, 2005).

As técnicas baseadas em medição necessitam que o usuário configure um ambiente real e essencialmente monitore o sistema sob a ação de uma carga de trabalho (JAIN, 1990). Para adquirir resultados representativos, a carga de trabalho deve ser cuidadosamente selecionada. Essa carga utilizada nos estudos de desempenho pode ser real ou sintética. Embora a carga de trabalho real possa representar de forma fidedigna o sistema, ocasionalmente esta opção não é a desejável. Isso pode acontecer quando o tamanho da carga disponível não é considerável, ou ainda quando esses dados receberam muitas interferências, ou até mesmo por limitações no acesso aos dados reais.

As ferramentas que auxiliam a avaliação de desempenho de sistemas podem modificar o comportamento das métricas que estão sendo medidas. Assim, quanto maior a quantidade de métricas e o intervalo utilizado para realizar a medição, maior será a interferência da ferramenta de medição no ambiente. Essa interferência introduzida pela ferramenta de medição torna os dados coletados menos confiáveis (LILJA, 2005). Além disso, as métricas podem ser classificadas em categorias baseadas no tipo de evento que compreende a métrica:

- **Métricas de contagem de evento.** Apenas contam a quantidade de vezes que um determinado evento ocorre. Exemplos: quantidade de requisições de leitura/escrita de um disco;

- **Métricas de evento secundário.** Registram os valores de alguns parâmetros secundários, sempre que um determinado evento ocorrer;

- ***Profiles.*** É usado para caracterizar o comportamento de um programa ou uma aplicação de um sistema. Normalmente, é usado para identificar onde o programa ou sistema está consumindo mais tempo.

Podem ser citadas, também, diferentes estratégias de medição que atendem a diferentes tipos de eventos:

- **Dirigida a evento.** É a estratégia mais simples de medição de evento. Esse tipo registra informações necessárias para calcular a métrica de desempenho sempre que um pré-evento ou eventos ocorrem. Uma vantagem dessa estratégia é que o *overhead* gerado ocorre apenas no registro da informação. Se o evento nunca ocorre, ou apenas raramente, a pertubação no sistema será relativamente pequena;

- ***Tracing.*** É uma estratégia similar à dirigida a evento, porque ao invés de simplesmente gravar os eventos que ocorrem, uma parte do estado do sistema é registrada para identificar o evento. Portanto, é uma estratégia que requer mais armazenamento do que

um simples contador de eventos;

- **Amostragem.** Essa estratégia de medição registra em intervalos fixos de tempo a métrica de interesse. Como resultado, um *overhead* pode ocorrer dependendo da frequência em que a medição é executada;

- **Indireta.** É uma estratégia que deve ser usada quando a métrica desejada não está acessível diretamente. Nesse caso, deve-se encontrar outra métrica que pode ser medida diretamente, a partir da qual se pode deduzir ou obter a métrica de desempenho desejada.

Já as técnicas baseadas em modelagem podem ser resolvidas tanto analiticamente, quanto por simulação (CHUNG, 2003). Os modelos analíticos representam o comportamento do sistema através de um conjunto de equações matemáticas, cujas soluções oferecem informações sobre as métricas de desempenho estabelecidas. Por outro lado, os modelos de simulação são programas que emulam o comportamento de um sistema simulando a geração e o processamento de eventos (MENASCÉ et al., 1999).

Os modelos de simulação podem ser utilizados tanto na avaliação de desempenho de sistemas, quanto para a validação dos modelos analíticos. Ao contrário das medições, as simulações baseiam-se em modelos abstratos do sistema, logo não exigem que o sistema esteja totalmente implantado para que sejam aplicadas. Assim, os modelos utilizados durante a simulação são elaborados através da abstração de características essenciais do sistema, sendo que a complexidade e o grau de abstração dele podem variar de um sistema para outro. Durante a simulação, controlam-se, com maior eficiência, os valores assumidos por parâmetros do sistema (MENASCE et al., 2004).

2.4 PERFORMABILIDADE

A computação em nuvem emergiu através de inúmeros recursos e serviços inovadores, ao mesmo tempo que tem atraido usuários que buscam minimizar investimentos em infraestrutura e custos de gerenciamento de recursos com um ambiente que apresenta um serviço flexível e elástico. Como um sistema em nuvem fornece essencialmente a computação orientada a serviços, o desempenho do serviço se torna uma métrica importante e que precisa ser analisada em detalhes. Considerando que máquinas virtuais (VMs) podem interromper a disponibilização de um dado serviço por meio de falhas em seus recursos computacionais, o desempenho do serviço pode ser afetado pela confiabilidade dos recursos. Assim, combinar métricas de desempenho e confiabilidade é essencial para uma avaliação mais precisa.

Em 1980, John Meyer introduziu o termo performabilidade no contexto da avaliação de computadores de controle de aeronaves de alta confiabilidade de uso da *National Aeronautics and Space Administration* (NASA) (MEYER, 1980). Originalmente, o termo foi proposto para refletir confiabilidade e outros atributos de desempenho, como disponibilidade e manutenção. Entretanto, isso representava apenas parcialmente as métricas de

desempenho. Em seguida, outros conceitos foram complementados, tais como, sustentabilidade, segurança e qualidade (XING; AMARI; MISRA, 2008).

A avaliação de performabilidade abrange a utilização de métodos de avaliação de desempenho e dependabilidade (HAVERKORT; NIEMEGEERS, 1996). A partir da avaliação é possível investigar o efeito de eventos de falhas e atividades de reparo na degradação do desempenho de sistemas. Usualmente avalia-se separadamente os atributos de desempenho e dependabilidade, no entanto, o desempenho é dependente da dependabilidade. Neste sentido, é notável que a ocorrência de falhas implicará na qualidade do serviço oferecido pelo sistema, sendo esse comprometimento uma degradação no desempenho.

Dado que a avaliação de desempenho e dependabilidade são realizados de modo separado, é comum a utilização de técnicas de decomposição hierárquicas para combinação de modelos de dependabilidade com modelos de desempenho. A Figura 3 ilustra a representação hierárquica de um modelo base composto por relações entre submodelos de performabilidade, dependabilidade e desempenho. O primeiro nível é representado por um modelo de confiabilidade e um modelo de desempenho. Já o segundo nível é representado por um modelo de performabilidade.

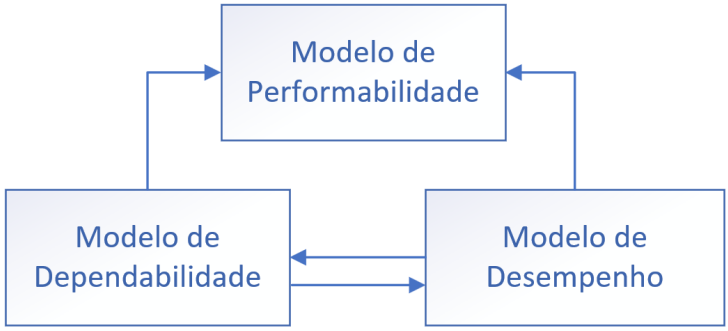


Figura 3 – Representação Hierárquica de um modelo base.
Fonte: (TAI; MEYER; AVIZIENIS, 1993).

Com a utilização da modelagem hierárquica, é possível evitar possíveis problemas relacionados ao tamanho do espaço de estados gerados pelos modelos (Ver item 2.5). Isso ocorre devido a diferença de magnitude entre os valores obtidos pelas métricas de desempenho e dependabilidade (LANUS; YIN; TRIVEDI, 2003). Tais problemas são conhecidos como *largeness* e *stiffness* (MUPPALA; MALHOTRA; TRIVEDI, 1996). *Largeness* é representado pelo tamanho do espaço de estados gerado na análise do modelo e *stiffness* é o efeito das diferentes ordens de magnitude entre os tempos das atividades de desempenho, os tempos dos eventos de falhas e os tempos das atividades de reparo. A combinação de aspectos de desempenho e dependabilidade na fase de modelagem é conhecida como modelagem de performabilidade. E a partir da análise dos modelos hierárquicos, é possível explicar a degradação dos níveis de serviço de um sistema provocados pelos eventos de falhas durante um determinado período de tempo.

2.5 MODELAGEM

Segundo Menascé (MENASCÉ; ALMEIDA; DOWDY, 1994), "*um modelo é a representação de um sistema (Ver Figura 4). No caso de sistemas computacionais, os modelos podem ser usados para representar as operações de um sistema ou o seu comportamento* [...]". Um modelo, na maioria das vezes, aproxima o comportamento do sistema real, em que a qualidade é dada justamente pela aptidão demonstrada para a previsão de resposta do sistema (variáveis de saída) a um estímulo (variáveis de entrada). O nível de abstração do modelo deve estar condizente com os objetivos da análise, e a inclusão de muitos detalhes poderá impactar na complexidade do modelo, bem como na sua avaliação. Por outro lado, a omissão de características importantes pode resultar em dados imprecisos. Nas seções seguintes, são introduzidos os principais modelos usados para avaliação de dependabilidade dos sistemas computacionais. Esses modelos são baseados em dois tipos: modelos de espaço de estados e modelos sem espaço de estados.

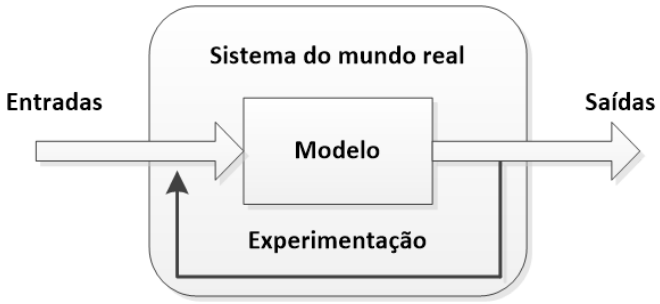


Figura 4 – Exemplo de um esquema de modelagem.

A *International Telecommunication Union* (ITU) descreve disponibilidade como, "[...] *a habilidade de um sistema/componente estar em um estado para executar uma função requerida em um instante qualquer de tempo ou em um instante qualquer dentro de um intervalo de tempo, assumindo que os recursos externos, se necessários, são fornecidos*" (ITU, 1994). Para sistemas críticos, como sistemas hospitalares, sistemas de usinas nucleares e sistemas de controle de aeronaves, alta disponibilidade é extremamente importante, visto que falhas podem ser catastróficas, resultando em perda de vidas ou altos prejuízos financeiros.

Modelos estocásticos têm sido largamente usados para prever a disponibilidade e confiabilidade de sistemas. Esses modelos podem prover importantes informações para os(as) projetistas antes de que o produto seja colocado em operação considerando diferentes cenários. Os aspectos de disponibilidade de um sistema/componente são geralmente descritos através de modelos sem espaço de estados, como por exemplo: diagrama de bloco de confiabilidade (RBD) (XIE; DAI; POH, 2004); árvore de falhas (FT) (SAHNER; TRIVEDI; PULIAFITO, 2002); gráfico de confiança (Relgraph) (TRIVEDI, 2002); modelos de espaço de estados (e.g.: cadeia de Markov, redes de Petri (MERLIN; FARBER, 1976), entre outros); hierarquia (LANUS; YIN; TRIVEDI, 2003) e interação de ponto fixo (*fixed-point*

iterative) (GHOSH et al., 2010). *Downtime*, indisponibilidade, disponibilidade estacionária, disponibilidade instantânea e disponibilidade intervalar são algumas métricas de disponibilidade dos sistemas.

Assumindo tempos de defeito e reparo exponencialmente distribuídos com as respectivas taxas λ e μ, a disponibilidade no tempo t e a disponibilidade intervalar podem ser calculadas pelas seguintes expressões (TRIVEDI, 2002):

$$A(t) = \frac{\mu}{\lambda + \mu} + \frac{\mu}{\lambda + \mu} e^{-(\lambda+\mu)t} \tag{2.1}$$

$$A(t) = \frac{\int_0^t A(x)dx}{t} = \frac{\mu}{\lambda + \mu} + \frac{\mu}{\lambda + \mu} e^{-(\lambda+\mu)t} \tag{2.2}$$

Calculando o limite até o infinito da disponibilidade instantânea, a disponibilidade em estado estacionário (A_{sys}) pode ser calculada como segue:

$$A_{sys} = \lim_{n \to 0} A(t) = \frac{\mu}{\lambda + \mu} \tag{2.3}$$

A indisponibilidade em estado estacionário e o *downtime*[2] são obtidos a partir da disponibilidade em estado estacionário (A_{sys}), conforme as Equações 2.4 e 2.5.

$$U_{sys} = 1 - A_{sys} \tag{2.4}$$

$$Downtime = (1 - A_{sys}) \times 8760 \times 60 \tag{2.5}$$

A confiabilidade de um sistema pode ser obtida através da probabilidade de que esse sistema execute a sua função, de modo satisfatório, sem a ocorrência de defeitos, por um determinado período de tempo. A confiabilidade pode ser obtido através da Equação 2.6, onde T é uma variável aleatória que representa o tempo para ocorrência de defeitos no sistema (KUO; ZUO, 2003).

$$R(t) = P\{T > 1\}, t \geq 0 \tag{2.6}$$

Já a probabilidade da ocorrência de defeitos até um instante t é representada pela Equação 2.7, onde T é uma variável aleatória que representa o tempo para ocorrência de falhas no sistema.

$$F(t) = 1 - R(t) = P\{T \leq t\} \tag{2.7}$$

[2] O período de inatividade de um sistema é associado a uma unidade de tempo, por exemplo, em minutos por ano)

Por fim, a disponibilidade orientada à capacidade (COA) avalia o quanto do sistema é realmente entregue a seus usuários e, para tanto, considera estados de atividade e inatividade do sistema na entrega do serviço. Deste modo, podemos obter a disponibilidade orientada à capacidade pela Equação 2.8 (MACIEL et al., 2012):

$$COA = \frac{\sum_{si \in US} pci \times \pi_i}{MPC} \tag{2.8}$$

onde temos, π_i representando a disponibilidade estacionária no estado i, pertencente ao conjunto de todos os estados de disponibilidade US, pci a capacidade de operação no estado si e MPC a capacidade máxima de operação do sistema.

2.6 TÉCNICAS DE REDUNDÂNCIA

Redundância é uma abordagem aplicada em vários domínios (*hardware*, *software* e outros) para aumentar a disponibilidade e confiabilidade dos sistemas. Tradicionalmente, as técnicas de dependabilidade são classificadas em Prevenção a defeitos, Remoção de defeitos e Tolerância a defeitos (GEFFROY; MOTET, 2002). A abordagem de prevenção de defeitos consiste em evitar ou reduzir a introdução de falhas durante a especificação, projeto, produção e estágios operacionais. Bem como, a remoção de defeitos objetiva detectar e eliminar falhas presentes nas mesmas fases de especificação, projeto, produção e estágios operacionais. Ao contrário de outras técnicas, a tolerância a defeitos visa permitir a entrega correta do serviço até mesmo na presença de falhas. Os recursos adicionais são acrescentados para garantir a integridade da entrega do serviço. No entanto, adicionar redundância proporciona aumento nos custos e na complexidade do sistema. Nesse sentido, deve-se fazer um estudo detalhado do sistema abordado para que seja aplicada a técnica de redundância mais adequada.

A técnica de redundância pode ser estática ou dinâmica, e dependendo do tempo, as falhas podem ser detectadas ou mascaradas (GEFFROY; MOTET, 2002; AVIZIENIS et al., 2004). Em um mecanismo redundante estático, ambos os elementos principais (por exemplo, dispositivo, tarefa e serviço) e os redundantes estão permanentemente ativos. Portanto, no caso de falhas nesse mecanismo, os usuários não perceberão que ocorreu uma falha. A redundância modular tripla (TMR) (SHOOMAN, 2002) (ver Figura 5) corresponde a um mecanismo de redundância usual. Esse mecanismo é composto por três módulos redundantes e um *voter*. O resultado de saída desse mecanismo consiste em avaliar o resultado produzido pelos três módulos e decidir a resposta correta através do mecanismo *voter*, desde que apenas um deles esteja em estado falha. Se um dos módulos ficar defeituoso, os dois restantes mascaram o resultado do módulo defeituoso quando é realizada a votação. A redundância modular N é uma generalização do TMR, no qual, em vez de considerar apenas três estruturas redundantes, N estruturas são consideradas.

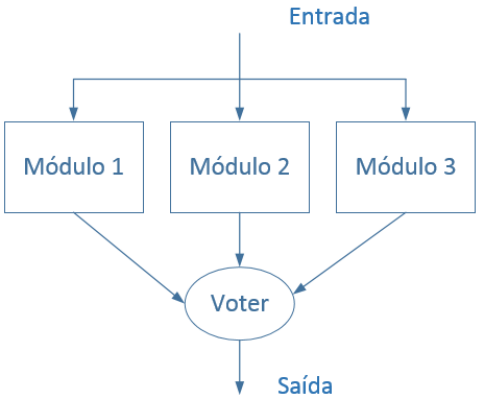


Figura 5 – Mecanismo de redundância modular tripla.
Fonte: (MAGALHAES; PINHEIRO, 2007).

O mecanismo de redundância dinâmica (redundância ativa) é caracterizado pelo mecanismo de detecção da falha e por uma ação de recuperação que consome um determinado tempo de ativação. O modelo *cold standby* (ver Figura 6) é uma solução que possui um módulo reserva desabilitado (por exemplo, Módulo 2) (PRADHAN, 1996). Estes módulos podem ser colocados fora do sistema em boas condições de ambiente, com o propósito de preservar a confiabilidade dos seus componentes, uma vez que esses módulos são desabilitados em condições normais, e com a ocorrência de falhas no módulo principal, o módulo reserva deve ser inicializado com a finalidade de manter o sistema disponível. Esta operação pode levar muito tempo, e por isso, a disponibilidade do sistema é afetada. Esta solução pode não ser apropriada para alguns tipos de sistemas em que as restrições de tempo são muito limitadas.

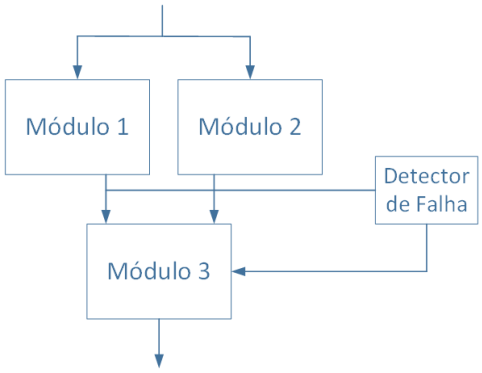


Figura 6 – Mecanismo de redundância *Cold*.
Fonte: (YANG, 2007).

Uma alternativa para a redundância *cold standby* é o mecanismo *hot standby*, que adota a redundância que funciona continuamente em paralelo com a unidade ativa. Consequentemente, em vez de manter o módulo reserva desativado, o mecanismo *hot standby* mantém esse módulo redundante ativado de forma que o defeituoso possa ser substituído sem que necessite de um tempo extra para chavear para o reserva. Além desses mecanismos mencionados, há o mecanismo *warm standby*. Esse mecanismo, possui módulos redundantes que ficam funcionando em sincronia com o módulo operacional em modo *standby*. Em uma eventual falha detectada, o módulo redundante estará pronto para se tornar operacional após um intervalo de tempo de ativação. Dessa forma, mecanismos com redundância *cold* e *warm standby* necessitam de maior tempo para o processo de recuperação, em relação ao *hot standby*. Porém, como consequência, haverá um maior consumo de recursos (GEFFROY; MOTET, 2002).

2.7 MODELOS SEM ESPAÇO DE ESTADO

Modelos de disponibilidade podem ser construídos usando modelos sem espaço de estados, tais como diagrama de blocos de confiança, gráfico de confiança e árvore de falhas com e sem eventos repetidos. Tais modelos são fáceis de serem usados e possuem uma solução geralmente rápida, pois não requerem a geração de espaço de estados durante a sua solução (SATHAYE; RAMANI; TRIVEDI, 2000). Porém, para uma rápida solução, os modelos assumem que os componentes do sistema são independentes. A disponibilidade, a indisponibilidade, a confiabilidade e o tempo médio de defeito dos sistemas podem ser computados usando esses modelos. As três principais técnicas de solução para os modelos sem espaço de estados são: fatoração, soma dos produtos disjuntos (TRIVEDI, 2002) e árvore de decisão binária (VOAS; ZHANG, 2009).

2.7.1 Diagrama de Bloco de Confiabilidade

O RBD é um modelo sem espaço de estados que permite calcular métricas como confiabilidade e disponibilidade de sistemas complexos. Embora tenha sido inicialmente proposto como um modelo para o cálculo da confiabilidade, também pode ser utilizado para o cálculo de métricas de dependabilidade, tais como disponibilidade e manutenabilidade. O modelo RBD faz o uso de uma representação bastante intuitiva que resulta em fácil implementação e análise.

Os componentes são representados por blocos, facilitando assim a sua representação estrutural. O diagrama de blocos de confiabilidade oferece uma representação gráfica de forma que os componentes do sistema estão conectados entre si. Em um modelo de RBD, os componentes são representados utilizando as seguinte composições: série, paralelo, ponte, blocos *k-out-of-n*, ou ainda, uma combinação das abordagens anteriores (KUO; ZUO, 2003; XIE; DAI; POH, 2004; FOGLIATTO; RIBEIRO, 2010).

A confiabilidade do sistema para a estrutura em série (R_s) considerando n componentes é dada por:

$$R_s = \prod_{i=1}^{n} R_i \tag{2.9}$$

em que R_i é a confiabilidade do bloco i. A confiabilidade do modelo em paralelo (R_p), considerando n componentes, pode ser obtida pela equação abaixo.

$$R_p = 1 - \prod_{i=1}^{n} (1 - R_i) \tag{2.10}$$

onde R_i é a confiabilidade do bloco i.

Segundo (XIE; DAI; POH, 2004), as equações básicas para calcular a confiabilidade do sistema em série e em paralelo (ver Equações 2.9 e 2.10) podem ser combinadas para calcular a confiabilidade do sistema contendo estruturas tanto em paralelo quanto em série (sistema série-paralelo). Dessa forma, qualquer sistema série-paralelo pode ser eventualmente transformado em um bloco e sua confiabilidade pode ser facilmente calculada usando, repetidamente, as equações acima.

2.7.2 Funções Estruturais

Funções estruturais RBD relacionam o estado de funcionamento dos componentes de um determinado sistema com o estado de funcionamento do sistema. Seja x_i a variável que representa o estado do componente i para $1 \leq i \leq n$. Onde n é o número de componentes do sistema. Então o estado x_i do componente i é dado por:

Definição 1. *O estado do componente i é definido por x_i, onde:*

$$x_i = \begin{cases} 1, & \text{se o componente i está funcionando} \\ 0, & \text{se o componente i falhou.} \end{cases}$$

Para $i = 1, 2, ..., n$. Onde, n representa o número de componentes.

O vetor $x = (x_1,\ x_2, ...,\ x_n)$ representa o estado de todos os componentes. O estado do sistema é determinado pelos estados dos componentes. A função da estrutura Φ representa o estado do sistema a partir dos estados dos componentes do vetor x.

Definição 2. *A função Φ determina o estado do sistema:*

$$\Phi(x) = \begin{cases} 1, & \text{se o sistema i está funcionando} \\ 0, & \text{se o sistema falhou.} \end{cases}$$

Se o estado de todos os componentes é conhecido, então o estado do sistema também é conhecido. Assim, o estado do sistema é uma função determinística do estado de todos os componentes.

$$\Phi = \Phi(x) = \Phi(x_1, x_2, ..., x_n) \tag{2.11}$$

onde $\Phi(x)$ é a função estrutural do sistema. As regras de formação das funções estruturais são mostradas a seguir.

2.7.3 Componentes em Série

A Figura 7 ilustra um diagrama de blocos de confiabilidade de um sistema em série. Esse sistema somente funciona se todos os seus componentes estiverem em estado de funcionamento. Sejam n componentes $x_1, x_2, ..., x_n$ em série, a função estrutural Φ desses componentes é representada por:

$$\Phi(x) = \begin{cases} 1, & \text{se } x_i = 1 \ \forall \ i = 1, 2, ..., n \\ 0, & \text{se } \exists \ i : x_i = 0. \end{cases}$$

$$\Phi(x) = \prod_{i=1}^{n} x_i \ = min\{x_1, x_2, ..., x_n\} \tag{2.12}$$

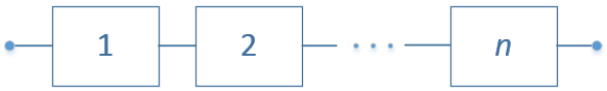


Figura 7 – Exemplo de um modelo RBD em série.

2.7.4 Componentes em Paralelo

A Figura 8 mostra um diagrama de blocos de confiabilidade de um sistema em paralelo. Esse sistema funciona apenas, se um ou mais dos seus componentes estão em estado de funcionamento. Sejam n componentes $x_1, x_2, ..., x_n$ em paralelo, a função estrutural Φ desses componentes é representada por:

$$\Phi(x) = \begin{cases} 1, & \text{se } \exists \ i : x_i = 1 \\ 0, & \text{se } x_i = 0 \ \forall \ i = 1, 2, ..., n. \end{cases}$$

$$\Phi(x) = 1 - \prod_{i=1}^{n} (1 - x_i) \ = \ max\{x_1, x_2, ..., x_n\} \tag{2.13}$$

Sistemas em série e paralelo são casos particulares da estrututra *k-out-of-n*. Assim, o sistema está operacional se k ou mais de seus n componentes estiverem operantes.

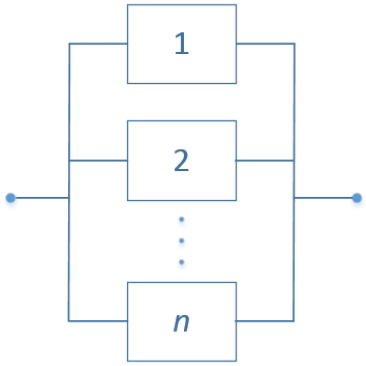


Figura 8 – Exemplo de um modelo RBD em paralelo.

Um sistema em série representa um sistema do tipo *n-of-n*; um sistema em paralelo corresponde a um sistema do tipo *1-of-n* [3].

2.7.5 Modelos com espaço de estado

Um modelo representado em espaço de estados possibilita compreender as relações entre variáveis internas ao sistema. Esse tipo de representação descreve o sistema no domínio do tempo sendo útil para descrever sistemas não lineares e multivariáveis (AGUIRRE, 2004). Os modelos de espaço de estados conseguem capturar comportamentos em detalhes comportamentais, e.g., fator de cobertura, falhas correlatas, dependências de reparo, etc. Já os modelos sem espaço de estados (ex.: RBD e Árvore de Falhas (FT)) não conseguem facilmente levar em consideração esses detalhes comportamentais. Modelos baseados em estados podem também ser chamados de não-combinatoriais, e os modelos não baseados em estados são chamados de combinatoriais.

As Cadeias de Markov de Tempo Contínuo (CTMC)s são modelos de espaço de estados amplamente conhecidos no meio acadêmico, sendo usadas no estudo de desempenho e disponibilidade dos sistemas computacionais. As CTMCs homogêneas são representadas por estados e transições entre os estados, em que o tempo médio de permanência (*sojourn time*) segue uma distribuição exponencial. Se se considera como uma distribuição exponencial, então o modelo pode se tornar um processo semi-Markoviano (CAO; SUN; TRIVEDI, 2001), um processo de Markov regenerativo (GIRAULT; VALK, 2001) ou uma cadeia de Makov não homogênea (GOKHALE; LYU; TRIVEDI, 2004). A Figura 9 apresenta um exemplo simples de um modelo CTMC para um sistema redundante de dois componentes com a mesma taxa de reparo μ. A taxa de falha dos dois componentes é λ. Quando ambos os componentes tiverem falhado, então o sistema é considerado não operacional.

No entanto, um dos principais problemas das cadeias de Markov é a explosão do espaço de estados, visto que este tipo de cadeia de um sistema complexo pode chegar a milhares ou milhões de estados. As redes de Petri estocásticas (SPNs) podem ser usadas

[3] Para maiores detalhes (KUO; ZUO, 2003; XIE; DAI; POH, 2004; FOGLIATTO; RIBEIRO, 2010).

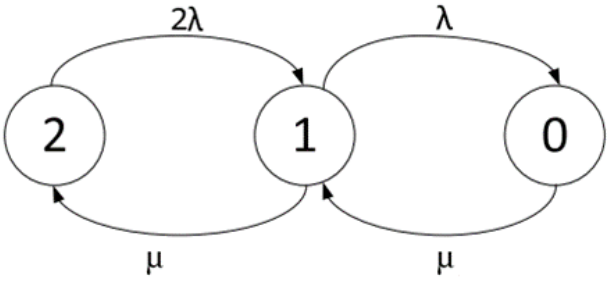


Figura 9 – Exemplo de um esquema de modelagem.

para a especificação e geração automática das cadeias de Markov, a fim de tolerar o problema de explosão do espaço de estados através de um modelo menor e conciso. Modelos baseados em redes de Petri (RdPs) têm sido largamente usados para modelar vários projetos indo desde os sistemas embarcados de tempo-real até os sistemas de computação em nuvem (GHOSH et al., 2010). A seguir são detalhadas as redes de Petri.

2.7.6 Redes de Petri

O conceito das RdP foi criado por Carl Adam Petri na sua tese de doutorado intitulada de "Kommunikation mit Automaten"(Comunicação com Autômatos) em 1962 (PETRI, 1962). Deste então, esse formalismo tem sido amplamente utilizado em diferentes áreas, tais como Ciência da Computação, Engenharia Elétrica, Administração, Química, entre outras. Diversas derivações do modelo de RdP clássico têm sido desenvolvidas ao longo do tempo, tais como redes temporizadas (MERLIN; FARBER, 1976), estocásticas (MARSAN et al., 1994), alto nível (JENSEN, 1990) e orientadas a objetos (WANG, 1996). Esse panorama foi devido à necessidade de suprir as diferentes áreas de aplicação, além de prover facilidades de comunicação e transferência de métodos e ferramentas de uma área para outra.

As vantagens do uso das RdPs são diversas, contudo, é possível destacar as seguintes (GIRAULT; VALK, 2001):

- RdPs fornecem um formalismo de modelagem que permite sua representação gráfica e são fundamentadas matematicamente;
- RdPs fornecem mecanismos de refinamento e abstração que são de grande importância para o projeto de sistemas complexos;
- Existe uma grande variedade de ferramentas disponíveis para as RdPs, tanto comerciais quanto acadêmicas, para modelagem, análise e verificação;
- RdPs têm sido utilizadas nas mais diversas áreas. Portanto, vários resultados são encontrados na literatura para os diversos domínios da sua aplicação;
- Existem várias extensões do modelo básico das RdPs.

2.7.7 Redes de Petri Lugar-Transição

Ademais, as RdPs são compostas pelos seguintes elementos (MURATA, 1989):

- **Lugares**. Representam os elementos passivos da rede, tendo como principal função armazenamento de *tokens*, os quais são removidos e adicionados à medida que as transições são disparadas. Graficamente, os lugares são representados por círculos (ver Figura 10 (a));
- **Transições**. Representam os elementos ativos da rede, ou seja, as ações realizadas pelo sistema. Para que uma transição esteja habilitada, é necessário que todas as suas pré-condições sejam satisfeitas, caso uma pré-condição não o seja, a transição estará desabilitada. Uma vez que a transição satisfaz todas as pré-condições, essa transição poderá disparar, removendo uma determinada quantidade de *tokens* dos lugares e colocando em outros, gerando assim as pós-condições. Graficamente, são representadas por traços ou barras (ver Figura 10 (b));
- **Arcos**. Representam o fluxo dos *tokens* pela rede (ver Figura 10 (c)). Na representação gráfica, as transições e os lugares podem ser conectados por múltiplos arcos (arcos multivalorados) que podem ser compactados em um único arco rotulado;
- ***Tokens***. Representam o estado em que o sistema se encontra em determinado momento (ver Figura 10 (d)).

Figura 10 – Elementos de uma rede de Petri.

Formalmente, as redes de Petri podem ser definidas como segue.

Definição 3. *(Redes de Petri) Uma rede de Petri é uma* $5-$*tupla,* $PN = (P, T, F, W, M_0)$*, onde:*

- $P = \{p_1, p_2, ..., p_n\}$, é um conjunto finito de lugares;
- $T = \{t_1, t_2, ..., t_n\}$, é um conjunto finito de transições;
- $F \subseteq (P \times T) \cup (T \times P)$ um conjunto de arcos;
- $W : F \to \{0, 1, 2, 3, ..., n\}$, é a função de atribuição de peso aos arcos;
- $M_0 : P \to \{0, 1, 2, 3, ..., n\}$, é a marcação inicial.

A seguir, na Figura 11, é apresentado um exemplo de uma RdP, no qual o funcionamento de um servidor é modelado. Nesse modelo, os lugares e as transições representam, respectivamente, os estados do servidor (funcionando ou parado) e as ações que alteram o estado do servidor (ativar ou desativar). O estado atual do modelo é representado por uma marca (*token*) no lugar correspondente à situação atual do modelo, como descrito na Figura 11 (a). Considerando que o estado atual do modelo é funcionando, a única transição que poderá ser disparada é desativar. Uma vez que essa transição seja disparada, então, o modelo passará do estado funcionando (ver Figura 11 (a)) para o estado parado (ver Figura 11 (b)). Mais informações relacionadas à estrutura básica das redes de Petri podem ser encontradas em (MURATA, 1989).

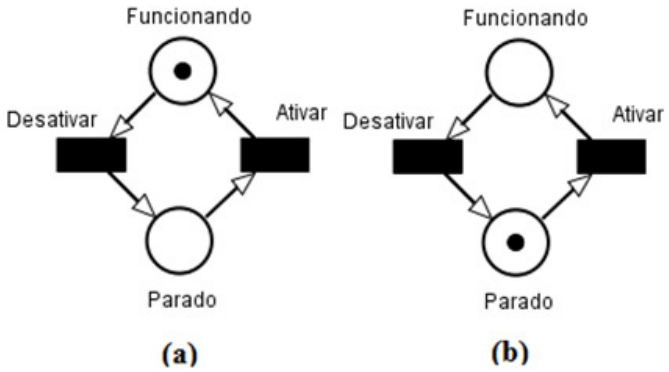


Figura 11 – Exemplo de uma rede de Petri.

2.7.8 Redes de Petri Estocástica

Nas SPNs, o tempo é descrito através de funções de distribuição probabilística (MARSAN; CONTE; BALBO, 1984; CIARDO; TRIVEDI, 1993; GERMAN, 2000; BALBO, 2001).

Definição 4. *(Redes de Petri Estocástica) Uma SPN é uma* $9-$*tupla,* $SPN = (P, T, I, O, H, \Pi, G, M_0, Atts)$*, onde:*

- $P = \{p_1, p_2, ..., p_n\}$ é um conjunto finito de lugares;
- $T = \{t_1, t_2, ..., t_n\}$ é um conjunto finito de transições;
- $I \in (\mathbb{N}^n \rightarrow \mathbb{N})^{n \times m}$ é a matriz que representa a multiplicidade dos arcos de entrada (que podem ser dependentes de marcações), onde a entrada i_{jk} de I mostra a multiplicidade do arco (que pode ser dependente da marcação) do lugar p_j para a transição t_k. $[A \subseteq (P \times T) \cup (T \times P)]$;
- $O \in (\mathbb{N}^n \rightarrow \mathbb{N})^{n \times m}$ é a matriz que representa a multiplicidade dos arcos de saída (que podem ser dependentes de marcações). Entrada O_{jk} de O mostra a multiplicidade do arco (que pode ser dependente da marcação) do lugar p_j para a transição t_k;

- $H \in (\mathbb{N}^n \rightarrow \mathbb{N})^{n \times m}$ é a matriz que representa os arcos inibidores (que podem ser dependentes de marcações). Entrada h_{jk} de H mostra a multiplicidade do arco inibidor (que pode ser dependente da marcação) do lugar p_j para a transição t_k;
- $\Pi \in \mathbb{N}^m$ é o vetor que designa um nível de prioridade para cada transição;
- $G \in (\mathbb{N}^n \rightarrow \{true, false\})^m$ é o vetor que associa uma condição de guarda relacionada à marcação do lugar a cada transição;
- $M_0 \in \mathbb{N}^n$ é o vetor que associa uma marcação inicial de cada lugar (estado inicial);
- $Atts = (Dist, W, Markdep, Policy, Concurrency)^m$ inclui o conjunto de atributos para transições, onde:
  - $Dist \in \mathbb{N}^m \rightarrow f$ é uma função de distribuição de probabilidade associada ao tempo de uma transição (esta distribuição pode ser dependente de marcação) (o domínio de f é [0,∞));
  - $W \in \mathbb{N}^m \rightarrow \mathbb{R}^+$ é uma função de peso (esta distribuição pode ser dependente de marcação);
  - $Markdep \in \{constant, enabdep\}$, onde a distribuição de probabilidade associada ao tempo de uma transição pode ser independente (constante) ou dependente da marcação (*enabdep* - a distribuição depende da condição de habilitação atual);
  - $Policy \in \{prd, prs\}$ é a política de memória adotada pela transição (*prd* - *preemptive repeat different*, valor padrão, de significado idêntico à *race enabling policy*; *prs* - *preemptive resume*, corresponde ao *age memory policy*);
  - $Concurrency \in \{ss, is\}$, é o grau de concorrência das transições, onde ss representa a semântica de *single-server* e *is* representa a semântica de *infinity server*.

Em muitas circunstâncias, pode ser adequado representar a marcação inicial como um mapeamento do conjunto de lugares para os números naturais ($m_0 : P \rightarrow \mathbb{N}$), onde $m_0(p_i)$ denota a marcação inicial do lugar p_i e $m(p_i)$ denota uma marcação alcançavel do lugar p_i. Neste trabalho, a notação $\#p_i$ tem sido adotada para representar $m(p_i)$.

Quanto à semântica de temporização, as transições temporizadas com grau de habilitação maior do que um (1) podem ser caracterizadas da seguinte forma: *single server, multiple server* e *infinite server* (MARSAN et al., 1994). Onde,

- *Single server*: as marcações são processadas serialmente. Após o primeiro disparo da transição temporizada, o temporizador é reiniciado como se a transição temporizada tivesse sido habilitada novamente;

- *Multiple server*: as marcações são processadas com um grau máximo K de paralelismo. Caso o grau de habilitação seja maior do que K, não será criado nenhum novo temporizador para processar o tempo para o novo disparo até que o grau de habilitação tenha diminuído abaixo de K;
- *Infinite server*: o valor de K é infinito, todas as marcações são processadas em paralelo e as temporizações associadas são decrementadas a zero em paralelo.

Por sua vez, nos modelos SPN, as transições são disparadas obedecendo à semântica *interleaving* de ações (MARSAN et al., 1994). Essa semântica define que as transições são disparadas uma a uma, mesmo que o estado compreenda transições imediatas não conflitantes.

A seguir, a Figura 12 ilustra os elementos das redes de Petri estocásticas (SPNs) que estendem as redes de Petri (RdPs). Os arcos inibidores são utilizados para impedir o disparo das transições da rede. De maneira semelhante aos arcos tradicionais, aqueles também podem ter um peso associado.

(a) Transição estocástica. (b) Arco inibidor.

Figura 12 – Elementos adicionados às SPNs que estendem o comportamento das RdPs.

2.8 ANÁLISE DE AGRUPAMENTO

Atualmente, a análise de agrupamentos tem sido aplicada nas mais diversas áreas, como, a Psicologia, Botânica, Antropologia, Economia, Análise de mercado, Computação e outros. De acordo com (FREI, 2006), o processo de agrupamento de dados pode ser dividido em cinco passos: a obtenção da matriz de dados; normalização ou padronização da matriz de dados (se necessário); cálculo da matriz de distância (similaridade); utilização do método de agrupamento; e decisão do número de grupos. São apresentadas, na sequência, algumas medidas de similaridade.

2.8.1 Medidas de Similaridade

As medidas de similaridade são utilizadas para organizar valores em grupos homogêneos. A similaridade entre duas amostras pode ser expressa como uma função da distância entre os dois pontos representativos destas amostras no espaço *n-dimensional*. A Figura 13 ilustra superfícies observadas pela distância Euclidiana e Manhattan. As distâncias Euclidiana, Manhattan e Minkowski são apresentadas nas Equações 2.14, 2.15 e 2.16, respectivamente, a seguir:

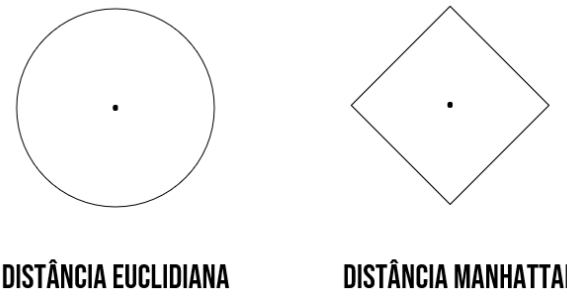


Figura 13 – Superfícies observadas pelas distâncias Euclidiana e Manhattan.

- Distância Euclidiana: a distância entre duas amostras (i e j) é a raiz quadrada do somatório dos quadrados das diferenças entre valores de i e j para todas as variáveis (r = 1,2, ..., n).

$$dist(x_i, x_j) = \left[\sum_{r=1}^{m} (x_{i,r}, -x_{j,r})^2 \right]^{\frac{1}{2}} \tag{2.14}$$

  Onde: $x_{i,r}$ representa a característica da variável i, $x_{j,r}$ representa a característica da variável j, m é o número de parcelas na amostra e r é o número na amostra.

- Distância Manhattan: a distância entre duas amostras (i e j) é o somatório da diferença entre valores de i e j para todas as variáveis (r = 1,2, ..., n).

$$dist(x_i, x_j) = \sum_{r=1}^{m} \mid x_{i,r} - x_{j,r} \mid \tag{2.15}$$

- Distância Minkowski

$$dist(x_i, x_j) = \left[\sum_{r=1}^{m} (x_{i,r}, -x_{j,r})^q \right]^{\frac{1}{q}} \tag{2.16}$$

  Onde $q \geq 0$. Logo, a distância de Minkowski generaliza tanto a distância Euclidiana (caso $q = 2$) quanto a distância Manhattan (caso $q = 1$).

A Figura 14 ilustra um exemplo das distâncias Euclidiana (a) e Manhattan (b) considerando duas amostras. Como mencionado, a distância Minkowski generaliza as duas distâncias. Cada distância pode conduzir a diferentes soluções de agrupamentos. Assim, a distância Euclidiana é recomendada para agrupamentos centróides [4]. Já a Manhattan, pode conduzir a agrupamentos inválidos se as variáveis forem altamente correlacionadas (HAIR et al., 2009). A distância de Manhattan é uma simplificação da distância Euclidiana, sendo uma medida mais simples e de fácil implementação.

[4] Ponto associado a uma forma geométrica.

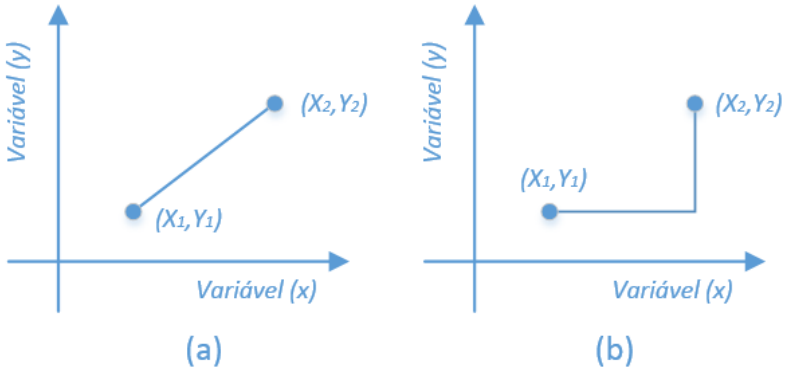


Figura 14 – Um exemplo de distância Euclidiana e Manhattan entre duas amostras considerando duas variáveis.

Em cada uma das equações que representam as distâncias geométricas, há a possibilidade da utilização de um peso (por exemplo, p_1 e p_2) para cada variável. Em algumas situações, caso o pesquisador julgue necessário, poderá adicionar uma importância maior a uma variável através do produto do peso a uma variável (FREI, 2006).

2.8.2 Normalização dos dados

No processo de agrupamento de dados, a normalização tem sido utilizada para evitar que as unidades das variáveis afetem a similaridade entre os valores. As Equações 2.17 e 2.18 apresentam, nessa ordem, as técnicas *z-score* e *mín-máx* (MITSA, 2010).

$$z_i = \frac{x_i - x^-}{\mu} \tag{2.17}$$

onde x^- e μ representam, respectivamente, a média e o desvio padrão dos valores de z. A Equação 2.18 apresenta o método mínimo-máximo

$$x'_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{2.18}$$

onde x_i e x'_i são, na devida ordem, o valor da amostra e o valor normalizado do i-ésimo dado, e x_{min} e x_{max} são, respectivamente, o valor mínimo e o valor máximo dos dados para cada variável. Nesse método, os dados normalizados ficam linearmente distribuídos entre 0 e 1.

2.9 OTIMIZAÇÃO

Otimização é o ato de obter o melhor resultado em dadas circunstâncias (RAO, 2009). Deb (2001) descreve que a otimização busca encontrar uma ou mais soluções viáveis que correspondem a valores extremos de um ou mais objetivos. Em projetos de engenharia e manutenção de qualquer sistema, os(as) projetistas precisam tomar uma série de decisões em várias fases. O objetivo final de todas essas tomadas de decisão visa minimizar

o esforço necessário ou maximizar algum benefício desejado, que em qualquer situação prática, pode ser ilustrado com uma função utilizando variáveis de decisão. O enunciado geral para um problema de otimização multiobjetivo (ou multicritério) é definido abaixo:

$$min\ f_m(x), m = 1, 2, ..., N_{obj} \tag{2.19}$$

$$\begin{cases} g_j(x) \leq 0, & j = 1, 2, ..., NR_{des} \\ h_k(x) = 0, & k = 1, 2, ..., NR_{igu} \\ x_i^L \leq x_i \leq x_i^U, & i = 1, 2, ..., N_{var} \end{cases} \tag{2.20}$$

onde x é um vetor de N_{var} variáveis de decisão, $x = (x_1, x_2, \ldots, x_{N_{var}})^T$ também denominado de solução. Os valores x_i^L e x_i^U representam os valores mínimo e máximo para a variável x_i, respectivamente. As NR_{des} desigualdades g_j e as NR_{igu} igualdades h_k são chamadas de funções de restrições. Cada N_{obj} funções objetivo $f_1(x)$, $f_2(x)$, $\ldots$, $f_m(x)$ pode ser maximizada ou minimizada. O vetor de funções objetivo $f(x)$ forma um espaço multidimensional denominado "espaço de objetivos Z". Essa é a diferença entre objetivos simples, cujo espaço de objetivos é unidimensional.

Métodos de otimização são importantes na prática para solucionar problemas, particularmente, em projetos de engenharia, experimentos científicos, para tomadas de decisão de negócios, e outros. A modelagem de otimização de um sistema pode envolver somente uma função objetivo, comumente chamada de "otimização de objetivo único". Por exemplo, uma empresa deseja reduzir o custo de produção de computadores. No entanto, quando um problema de otimização envolve mais do que um objetivo, essa solução é chamada de "otimização multiobjetivo". Utilizando o exemplo anterior, a mesma empresa deseja aumentar a produção e reduzir os custos.

A Figura 15 ilustra seis categorias de algoritmos de otimização (HAUPT; HAUPT, 2004), vejamos.

- (1) Tentativa e erro referem-se ao processo de ajuste das variáveis que afetam a saída sem saber muito acerca do processo que produz uma saída.
- (2) Unidimensional leva em consideração que o problema possui apenas uma variável, caso tenha mais de uma, caracteriza-se como multidimensional.
- (3) Dinâmico significa que a saída é uma função de tempo, enquanto que o estático significa que a saída independe do tempo.
- (4) O problema de otimização pode ser distinguido por variáveis discretas e contínuas. Aquelas possuem um número finito de valores possíveis, enquanto estas apresentam um número infinito de valores. Variáveis podem possuir limites ou restrições.

- (5) Otimização restrita ou condicionada incorpora variáveis de igualdade e desigualdade. Otimização sem restrições permite que as variáveis assumam qualquer valor.

- (6) Otimização não randômico é baseada em métodos tradicionais, a partir de um conjunto de valores. E, por fim, o método randômico é caracterizado por se basear em cálculos probabilísticos.

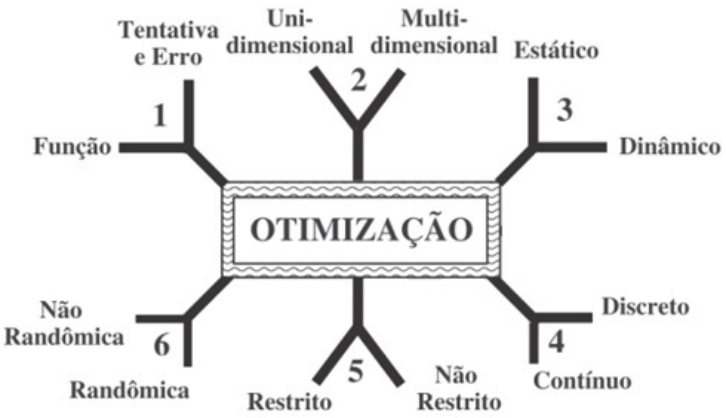


Figura 15 – Categorias dos algoritmos de otimização.
Fonte: (HAUPT; HAUPT, 2004).

Problemas com um único objetivo têm como propósito a obtenção de uma única solução global sendo máximo ou mínimo. Enquanto que em problemas multiobjetivos, a proposta consiste em adquirir um conjunto de soluções ótimas, conhecidas como soluções Pareto-ótimas[5]. ou soluções não-dominadas.

A definição de ótimo segundo Edgeworth-Pareto é baseada em que um ponto x^* é tomado como ótimo se "*nenhum critério utilizado pode melhorar a solução, sem piorar pelo menos um outro critério*". Assim, o ótimo de Edgeworth-Pareto quase sempre não fornece uma única solução, mas sim um conjunto de soluções denominadas não-inferiores ou soluções não-dominadas (ESCHNAUER; KOSKI; OSYCZKA, 1990).

A Figura 16 ilustra um exemplo que considera o conceito de dominância. Caso o objetivo seja minimizar o custo e maximizar a disponibilidade de um serviço, descartar-se-ia a soluçao 1, já que a solução 5 fornece maior disponibilidade pelo mesmo preço. A solução 2 é descartada pelo mesmo motivo. Tem-se, assim, três soluções: 3, 4, e 5 que são consideradas boas alternativas de contratação. Considerando termos quantitativos, nenhuma é melhor que a outra, pois uma tem maior disponibilidade, porém menos barata, e vice-versa. Quanto maior disponibilidade, maior o preço do serviço e vice-versa. Dessa forma, atribui-se um compromisso entre os objetivos. Diz-se que uma solução domina uma outra se seus valores são melhores em todos os objetivos. Por exemplo, a solução 5 domina

[5] A noção de ótimo foi originalmente proposta por Francis Ysidro Edgeworth (EDGEWORTH, 1881), sendo generalizada por Vilfredo Pareto (PARETO, 1896)

a solução 1. Então, a solução 5 não é dominada por nenhuma outra. O mesmo acontece com as soluções 3 e 4. Se não se conhece, a *priori*, a importância relativa de cada objetivo, pode-se dizer que as soluções 3, 4 e 5 são igualmente boas. Portanto, existe um conjunto de soluções ótimas, sendo este conjunto chamado de conjunto não-dominado. As outras soluções 1 e 2 formam o conjunto dominado.

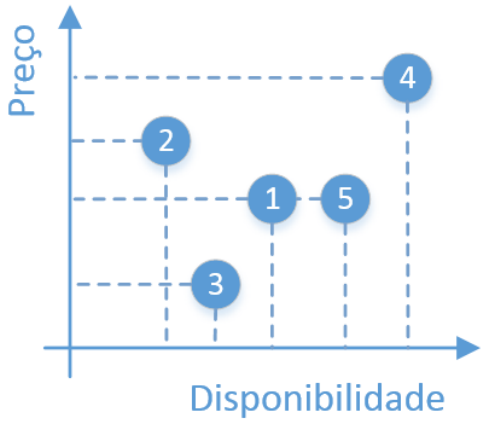


Figura 16 – Conjunto de opções de serviços (Preço x Disponibilidade).

Definição 5. *(Dominância): Uma solução x_1 domina uma outra solução x_2, se as seguintes condições são satisfeitas:*

- A solução x_1 não é pior que x_2 para todos os objetivos, ou seja, para todo j = 1, 2, ... , M;
- A solução x_1 é estritamente melhor que x_2 pelo menos em um objetivo, ou seja, pelo menos para um valor de j = 1, 2, ..., M;

Se ambas as condições são satisfeitas, pode-se dizer que x_2 é dominada por x_1, x_1 é não-dominada por x_2.

Definição 6. *(Otimalidade de Pareto): Uma solução x^* é Pareto-ótima, se e somente se, x^* é não-dominada em relação ao vetor de busca, ou seja, nenhum vetor do espaço de busca domina x^*.*

Definição 7. *(Pareto-ótimo global): Se não existir solução no espaço de busca que domine qualquer solução de um conjunto P, então as soluções pertencentes a P constituem o conjunto Pareto-ótimo global.*

Definição 8. *(Fronteira de Pareto): Conjunto de vetores de funções objetivo $f(x) = (f_1(x), f_2(x), ..., f_M(x))^T$, para cada solução x que está no conjunto ótimo de Pareto.*

A Figura 17 ilustra um exemplo de soluções distribuídas na fronteira de Pareto. Esse conjunto de soluções apresenta uma maior diversidade considerando as funções F_1 e F_2.

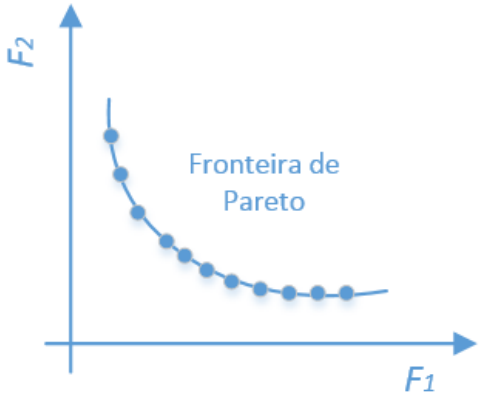


Figura 17 – Soluções na fronteira de Pareto.

A Figura 18 apresenta exemplos de conjuntos ótimos de Pareto, conforme várias combinações de objetivos para as funções F_1 e F_2.

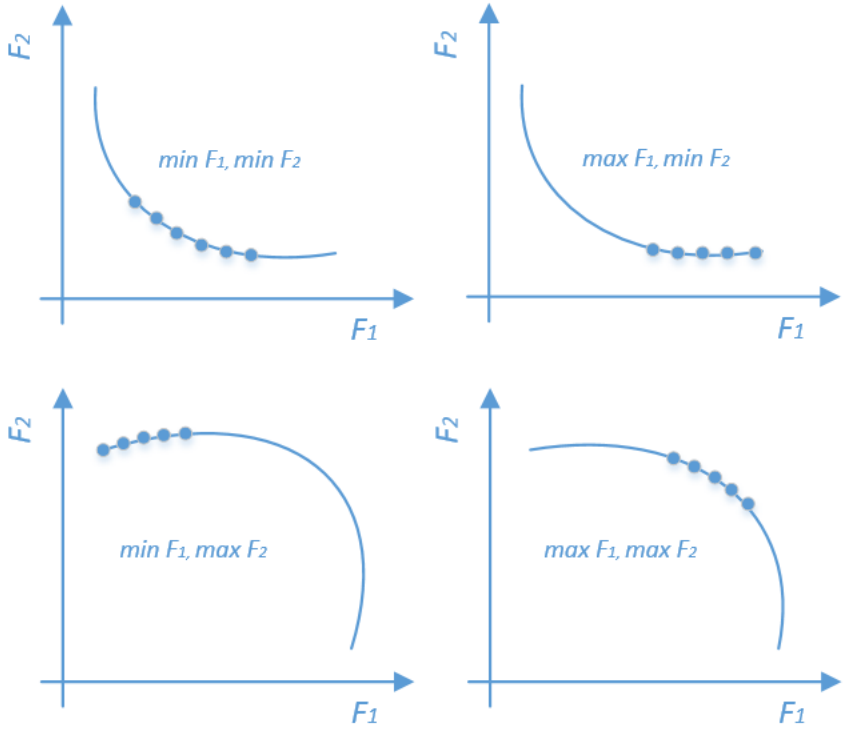


Figura 18 – Exemplos de conjuntos ótimos de Pareto.

2.10 CONSIDERAÇÕES FINAIS

Este capítulo apresentou os principais conceitos que envolvem esta tese. Primeiramente, apresentaram-se conceitos de computação em nuvem, dependabilidade, modelagem e técnicas de redundância. Subsequentemente, as redes de Petri foram introduzidas, mostrando que estas são uma ferramenta bem utilizada para modelagem e análise de vários tipos de sistemas, tais como os concorrentes, assíncronos, distribuídos, paralelos, não-determinísticos e estocásticos. Além disso, foram apresentados os conceitos sobre modelos baseados em diagrama de blocos de confiabilidade. E, por fim, apresentamos conceitos sobre análise de agrupamento e otimização.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta uma visão sobre a área de pesquisa desta tese, bem como uma revisão sobre os trabalhos relacionados. Estes, por sua vez, são apresentados considerando os seguintes temas: avaliação de dependabilidade, avaliação de performabilidade e modelos para tomada de decisão. Esta revisão inclui, ainda, a identificação dos objetivos e limitações dos trabalhos correlatos e uma comparação com a proposta desta tese.

3.1 INTRODUÇÃO

Serviços computacionais na nuvem têm sido adotados em diversas áreas, desde aplicações de gerenciamento financeiro de pequenas empresas até sistemas complexos de armazenamento de dados sigilosos. De acordo com as características e demanda do negócio, esses serviços requerem, por exemplo, maior confiabilidade e alta disponibilidade. A avaliação da disponibilidade e de aspectos relacionados à recuperação de falhas assume um papel importante na preservação e manutenção do funcionamento de tais serviços (BAUER; ADAMS, 2012; KULKARNI, 2016).

Sahner, Trivedi e Puliafito (2002) apresentam, além de um resumo teórico, uma variedade de modelos probabilísticos e de modelos de estado discreto que podem ser utilizados para avaliar a confiabilidade de sistemas computacionais e de comunicação. Ebeling (2004) apresenta modelos matemáticos que possibilitam a análise de componentes considerando a confiabilidade, disponibilidade e manutenabilidade. Ademais, o trabalho apresenta análises de falha e reparo baseadas em técnicas estatísticas e, por fim, aplicações e exemplos. De igual maneira, outros tipos de modelos matemáticos que avaliam sistemas computacionais podem ser encontrados em (DESROCHERS; AL-JAAR, 1995).

Em geral, a diversidade de trabalhos na literatura nesta área tem demonstrado que problemas reais podem ser aproximados a modelos matemáticos, necessitando de algumas simplificações e suposições (TRIVEDI; BOBBIO, 2017). Pode-se também apontar que os resultados podem variar de acordo com as técnicas adotadas, visto que, os parâmetros de entrada e variáveis de saída serão distintos.

3.2 AVALIAÇÃO DE DEPENDABILIDADE

Nos últimos anos, várias pesquisas têm sido propostas com a intenção de prover análise da confiabilidade e disponibilidade em *data center* e infraestruturas na nuvem. Wei, Lin e Kong (2011b) apresentam um método hierárquico combinando diagramas de blocos de confiabilidade e de Redes de Petri Estocástica Generalizada (GSPN) para a avaliação da confiabilidade de um *data center* virtual. O referido trabalho descreve a análise dos resultados de disponibilidade e confiabilidade, considerando mecanismos de virtualização, *backup*

e migração de dados. (WEI; LIN; KONG, 2011a), em outra pesquisa, combinam modelos de Diagramas de Blocos de Confiabilidade (RBD) e Cadeias de Markov (MC) para analisar uma estrutura virtual de um *cluster* computacional. Os resultados apresentam uma análise de dependabilidade, além de incluir a disponibilidade e confiabilidade. Os autores não levam em consideração questões como o custo, desempenho, características das máquinas virtuais e clusters utilizados nas análises das disponibilidade e confiabilidade.

Já (MELO et al., 2013) propuseram um modelo de disponibilidade para representar uma infraestrutura de computação em nuvem. A infraestrutura foi representada através de diagramas de blocos de confiabilidade. Além disso, foi proposto um modelo SPN para avaliar políticas de migração de máquinas virtuais considerando uma técnica de rejuvenescimento. Os resultados alcançados apresentam a elevação dos níveis de disponibilidade da infraestrutura. Em (GHOSH et al., 2013), os autores desenvolveram um modelo estocástico escalável usando *Stochastic Reward Nets* (SRN) e Cadeia de Markov (MC) para quantificar o desempenho de infraestruturas como serviço (IaaS) em nuvens. A avaliação de desempenho quantificou a probabilidade de rejeição de *jobs* e o tempo médio de atraso em uma infraestrutura na nuvem. Ademais, foi possível avaliar a capacidade da infraestrutura variando a taxa de chegada de *jobs* e tempo médio de serviço. A abordagem proposta permitiu a detecção de gargalos na infraestrutura, assim como a redução da complexidade do espaço de estados gerada na análise de modelos. Os autores não levam em consideração questões como o custo para o provisionamento das máquinas virtuais e físicas causados pela variação na taxa de chegada das *jobs*.

Em (SILVA et al., 2013b), os autores apresentam modelos de dependabilidade baseados em redes de Petri estocástica (SPN) e diagramas de blocos de confiabilidade (RBD) considerando a ocorrência de desastres em *datacenters*. O modelo SPN representou servidores em diferentes *data centers* distribuídos geograficamente com o objetivo de avaliar os níveis de disponibilidade. Na avaliação, foi utilizado um mecanismo para realizar a migração das VMs para diferentes *data centers*. Por outro lado, (SOUSA et al., 2015) descreveram uma estratégia de modelagem hierárquica considerando redes de Petri estocástica e diagramas de blocos de confiabilidade em uma infraestrutura em nuvem, chegando a resultados de disponibilidade e custo considerando mecanismos de redundância. Em (BRILHANTE et al., 2014), os autores modelaram uma infraestrutura em nuvem híbrida considerando redes de Petri estocástica. O estudo de caso utilizou um gerador de injeção de falhas para validar o modelo de disponibilidade. Embora esses trabalhos apresentem abordagens relacionadas à modelagem hierárquica e análise de disponibilidade do ambiente, aspectos como desempenho, confiabilidade e custos das infraestruturas não foram devidamente explorados.

Liu et al. (2018) aplicaram técnicas de modelagem analítica e análise de sensibilidade para investigar a disponibilidade de infraestruturas como serviço em *data center* considerando políticas de reparo. As políticas de reparo foram modeladas através das redes de Petri de Recompensa, do inglês, SRN. Um conjunto de máquinas físicas representado pelo

modelo SRN, e associado a políticas de reparo, analisou o impacto da disponibilidade na infraestrutura como serviço. Torres, Callou e Andrade (2018) apresentaram uma abordagem para avaliar a disponibilidade e o desempenho (vazão de arquivos transferidos) de um serviço de armazenamento de dados hospedado em uma nuvem privada por meio de modelos analíticos. Os autores utilizaram uma estratégia de modelagem hierárquica através de modelos RBD, CTMC e SPN. Os modelos foram usados para avaliar a disponibilidade do serviço da nuvem privada usando vários níveis de redundância entre os componentes. Além disso, os autores validaram o modelo de desempenho considerado um experimento com medições. Nos trabalhos mencionados, o foco da avaliação dos serviços computacionais volta-se à disponilididade, ainda que na modelagem adotada poderiam ter sido abordadas análises quanto à confiabilidade e ao custo.

Os trabalhos apresentados abordam o problema que é gerado após a interrupção de serviços em *data center* virtuais e infraestrutura em nuvem por meio de modelos de dependabilidade. Estratégias de migração de máquinas virtuais e mecanismos de redundância foram utilizados como uma forma de elevar a disponibilidade. Em contrapartida, este trabalho diversifica o estudo da dependabilidade utilizando a técnica de planejamento de experimento com o intuito de aprofundar o estudo e compreensão dos parâmetros de dependabilidade.

3.3 AVALIAÇÃO DE PERFORMABILIDADE

Goncalves et al. (2015) propuseram uma abordagem baseada em redes de Petri estocásticas (SPN) para avaliação de performabilidade de um serviço na nuvem. Os resultados do estudo de caso demonstraram como falhas podem impactar no serviço de nuvem adotado (Centro de Assistência ao Contribuinte - TAC), quando variada a quantidade de auditores no sistema. Foram apresentados resultados da disponibilidade do ambiente, considerando a variação no número de auditores no serviço (TAC), além da varição na demanda de auditores no tempo. No trabalho proposto, há uma limitação no detalhamento da infraestrutura adotada na nuvem, além dos custos empregados para o provimento do serviço por meio das atividades dos auditores.

Qiu et al. (2015) apresentaram uma abordagem de modelagem usando cadeias de Markov para analisar a performabilidade de um controlador de nuvem e de um serviço na nuvem. Um sub-modelo de confiabilidade foi proposto para analisar a probabilidade sobre a capacidade dos recursos disponíveis na nuvem, e um seguinte sub-modelo de desempenho levou em consideração a análise do tempo de serviço do gerenciador da nuvem e do tempo de serviço da VM. Os modelos de Markov foram usados para representar o desempenho e sub-modelos de confiabilidade. Uma abordagem Bayesiana foi aplicada para combinar os sub-modelos para avaliar a performabilidade dos serviços em nuvem e do sistema de nuvem. Embora a proposta abordada permita uma maior flexibilidade quanto à representação dos resultados, análises relacionando a degradação do desempenho

poderiam ter sido adotadas no modelo. Exemplos como relacionar uma quantidade de requisições descartadas pela presença de falhas no servico e a adoção de políticas de reparo para mitigar a degradacão no ambiente podem incrementar os resultados da abordagem proposta.

Wang, Chang e Liu (2016) desenvolveram um modelo monolítico, usando cadeia de Markov de tempo contínuo (CTMC) para representar um IaaS. O modelo permite a representação de máquinas físicas ativas e em espera, igualmente, possibilita a análise do processo de migração entre máquinas físicas ativas e em *standby*. Os resultados demonstram a variação no envio dos *jobs* associando a falhas no ambiente. O modelo permite considerar a migração de máquinas físicas na presença de falhas. No entanto, os autores não investigaram métricas como disponibilidade, confiabilidade e a degradação do desempenho na ocorrência de falhas. Ever (2017) realizaram uma análise de performabilidade de uma infraestrutura como serviço em nuvem utilizando modelos de filas e modelos de cadeias de Markov de tempo contínuo (CTMC). O estudo apresentado modela o comportamento do ambiente por meio das métricas como tamanho médio da fila, vazão e tempo de resposta. Foi considerada uma abordagem de computação em nuvem em larga escala, com cerca de 5.000 servidores. A avaliação de performabilidade considerou a variação na taxa de reparo no servidor. A partir do modelo, é possível realizar um planejamento de capacidade, identificar gargalos e analisar a disponibilidade da infraestrutura. Contudo, os autores não contemplaram tais resultados.

Maciel et al. (2017) apresentaram as principais características e métodos disponíveis de uma ferramenta (Mercury) para auxiliar na avaliação de confiabilidade e desempenho de vários sistemas, tanto para a academia quanto para a indústria. Para demonstrar a aplicabilidade da ferramenta, os autores apresentaram um estudo de caso a partir de um modelo de desempenho utilizando redes de Petri estocástica (SPN). O modelo representou o funcionamento de um serviço de vídeo sob demanda (*Video on Demand - VoD*). A partir do modelo, é possível avaliar a quantidade de pacotes perdidos, recebidos e a vazão (pacotes por segundo). Apesar do estudo demonstrar a análise de degradação do serviço através da perda de pacotes e taxa de transferência, resultados como disponibilidade e capacidade orientadas à disponibilidade poderiam ter sido pautado nessa abordagem.

Mo et al. (2018) propôs uma especificação de níveis de estados para sistemas, um diagrama de decisão multivalorado (MDD) baseado na especificação dos níveis e modelos (CTMC) para avaliar a performabilidade de sistemas computacionais de larga escala. A abordagem permitiu representar vários estados de degradação do sistema relacionando múltiplos processadores, hierarquia de armazenamento, processadores multicores e multimodos de falhas. Essa estratégia permitiu a avaliação do impacto na atribuição diversos componentes da infraestrutura, porém mecanismos com redundância poderiam ter sido adotados para avaliar a dispobilidade e confiabilidade.

Assim sendo, esta pesquisa combina modelos RBD e SPN conforme o nível de comple-

xidade do mecanismo de redundância atribuído aos componentes do ambiente em nuvem. Abordagem semelhante aos seguintes trabalhos (ARAÚJO et al., 2011; SOUSA et al., 2017; MATOS et al., 2017; ANDRADE et al., 2017; ARAUJO et al., 2018). No entanto, diferente dos trabalhos mecionados, esta tese adota o conceito de planejamento de experimentos para realizar o planejamento de avaliação dos aspectos de performabilidade e dependabilidade na representacão de infraestruturas, como serviço na nuvem. A abordagem combina modelos RBD e SPN para representar e avaliar infraestruturas em nuvem com mecanismos redundantes complexos e políticas de manutenção.

3.4 MODELOS PARA TOMADA DE DECISÃO

Diante da variedade de plataformas, infraestruturas e serviços disponibilizados na nuvem, diversas pesquisas buscam auxiliar na seleção ou ranqueamento de serviços que atendam às necessidades dos usuários especificadas em acordos de serviço. Nesse contexto, diversos autores propõem a utilização de modelos de decisão para solucionar conflitos de interesse no momento da tomada de decisão. Rehman, Hussain e Hussain (2012b) utilizaram um conjunto de métodos para tomada de decisão visando selecionar infraestruturas como serviço. A pesquisa apresentou um estudo de caso realizando um ranqueamento de 13 serviços na nuvem. O conjunto de serviços foi ranqueado baseando-se em 7 métodos com multicritérios.

Por sua vez, em (MARTENS; TEUTEBERG, 2012), os autores apresentaram um modelo de decisão para seleção de serviços na nuvem em um cenário *multisourcing* considerando custo e fatores de risco. O modelo proposto calcula a importância dos processos de negócio por meio de pesos. Adicionalmente, os autores especificaram um conjunto de parâmetros relacionados aos custos do negócio. O modelo foi simulado através da ferramenta FICO *Xpress Optimizer*. Além disso, foi apresentado um resumo sobre as teorias e requisitos utilizados na modelagem de decisão. No trabalho proposto por (GARG; VERSTEEG; BUYYA, 2013a), os autores apresentam um *framework* para ranqueamento de serviços em nuvem baseando-se em atributos de Qualidade de Serviço (QoS). O estudo de caso apresentou a classificação de três serviços na nuvem considerando métricas como qualidade, desempenho, sustentabilidade e capacidade. O *framework* proposto realiza o ranqueamento através de uma estrutura hierárquica composta por uma matriz de métricas ponderada.

Garg, Versteeg e Buyya (2013b) propuseram um *framework* chamado SMICloud para auxiliar na tomada de decisão. O *framework* considera um conjunto de atributos (por exemplo, responsabilidade, agilidade, garantia de serviço, desempenho, custo e usabilidade) ao priorizar e classificar os melhores serviços, com o mecanismo de classificação baseado no método AHP. Shivakumar, Ravi e Gangadharan (2013) apresentaram um modelo multicritério baseado em lógica *fuzzy* para ranquear serviços na nuvem. O modelo considera métricas de QoS e baseia-se na variação de uma escala representada por pesos. Um estudo de caso com três serviços foi apresentado considerando os níveis de qualidade

de cada serviço a partir das métricas agilidade, custo, desempenho, segurança e recursos computacionais (CPU, disco, memória). Em (MAITY; CHAUDHURI, 2014), foi apresentada uma proposta para seleção de serviços em ambientes na nuvem considerando um algoritmo genético multiobjetivo. Os atributos utilizados para cada serviço foram número de cores na CPU, CPU virtual e quantidade de mémória RAM.

Lee e Seo (2016) propuseram um modelo híbrido de tomada de decisão multicritério para seleção de Infraestrutura como Serviço (IaaS). O modelo considera o problema de seleção de serviços em nuvem usando indicadores balanceados de desempenho (BSC), do inglês, *Balanced Scorecard* (BSC), método fuzzy Delphi (FDM), do inglês, *Fuzzy Delphi Method* (FDM) e um processo hierárquico analítico fuzzy (FAHP), do inglês, *Fuzzy Analytical Hierarchy Process* (FAHP). Os critérios foram divididos em quatro classes: financeiro, cliente, processo de negócio e aprendizagem. Exemplos como custo de manutenção (financeiro), usabilidade (cliente), processo de negócio (simplificação do processo) e aprendizagem (agilidade), entre outros, foram utilizados como critérios para tomada de decisão. Tais critérios foram ponderados e apresentados a partir de cinco infraestruturas como serviços na nuvem.

Sachdeva et al. (2016) adotaram a técnica para ordenar alternativas através da similaridade com a solução ideal, do inglês, TOPSIS em conjunto com o método fuzzy intuicionista (IFS) para selecionar soluções de nuvem para gerenciar projetos de *big data*. O método adotado utiliza termos (muito importante ou sem importância) e valores empíricos em uma escala que varia de 1 (um) até 5 (cinco). O estudo de caso abordou cinco soluções no mercado de nuvem que suportam projetos de *big data*. Para cada solução, os autores especificaram critérios qualitativos como a continuidade e segurança no provimento do serviço. Liu, Chan e Ran (2016) apresentaram uma abordagem multicritério para auxiliar no processo de escolha de um fornecedor de serviços de nuvem. A proposta utilizou o método TOPSIS para encontrar a solução ideal e o método Delphi-AHP para associar pesos aos objetivos de ranqueamento. Como exemplo para ilustrar a aplicabilidade da abordagem, foram usados dados hipotéticos considerando atributos TOE (Tecnologia, Organização e ambiente). Os atributos utilizaram custos associados à aplicação, aos custos para computar, armazenar e transferir os dados.

Em (DING et al., 2017a), os autores propuseram uma abordagem para realizar o ranqueamento de serviços na nuvem baseado em um método de predição. A partir de informações como nível de satisfação dos usuários, o método associa as preferências informadas pelos usuários e identifica os resultados por um *ranking*. A abordagem adicionou essas preferências ao coeficiente de correlação Kendall (eKRCC), do inglês, *Kendall rank correlation coefficient* (KRCC). Um experimento foi utilizado para demonstrar a aplicabilidade do método utilizando um *dataset* com atributos de QoS. Já no trabalho seguinte, Ding et al. (2017b) estenderam o trabalho anterior e trouxeram esquemas de recomendação utilizando funções multiobjetivos. O estudo de caso utilizou o mesmo *dataset* com atributos

de QoS para realizar o ranqueamento dos cenários.

(JATOTH et al., 2018) apresentaram um método de decisão multicritério para a seleção de serviços em nuvem. A abordagem utiliza o método Grey TOPSIS associado ao método AHP. Os autores descreveram uma empresa hipotética com a intenção de selecionar infraestruturas como serviço considerando cinco parâmetros de QoS (custo, processamento, I/O, armazenamento e memória). O estudo preliminar apresentou a aplicação de dezenove cenários ao método proposto. No trabalho de Abdel-Basset, Mohamed e Chang (2018), é apresentado um tomador de decisão para estimar diferentes serviços em nuvem, fornecendo uma abordagem de análise de decisão multicritério para estimar a qualidade dos serviços em nuvem. Foi apresentado um modelo baseado no processo de hierarquia analítica neutrosófica, do inglês, *Neutrosophic Analytic Hierarchy Process* (NAHP). Os valores relacionados aos critérios, tais como, segurança, escalabilidade, desempenho, acessibilidade, adaptabilidade foram coletados por meio de entrevistas. Os resultados preliminares descreveram o *ranking* de três serviços na nuvem. Al-Jabri et al. (2018), por sua vez, pautaram um método de tomada de decisão baseado em grupos, onde um grupo de tomadores de decisão está envolvido no processo de decisão. Cada tomador de decisão fornece pesos para os critérios de seleção de nuvem. Com base nas combinações e desvios dos pesos, os tomadores de decisão selecionariam as alternativas.

Como visto, várias pesquisas apresentam diversas estratégias para o processo de tomada de decisão, considerando serviços em nuvem e infraestruturas computacionais. Para tanto, faz-se necessária a definição de objetivos, critérios e método(s) para encontrar uma solução ou um conjunto que possa atender uma demanda. Cada abordagem apresentada utilizou métodos baseados em simulação, algoritmos, ponderação, entre outros, para ordenação dos resultados. Esta tese propõe a utilização de métodos baseados em agrupamento de dados por meio de medidas de similaridade e dominância de pareto para agrupar e gerar um *ranking* de serviços modelados e avaliados. Assim, essa abordagem pode auxiliar gestores, analistas e usuários de serviços na nuvem no momento em que necessitem escolher uma um serviço.

3.5 COMPARAÇÃO COM OS TRABALHOS RELACIONADOS

A Tabela 2 apresenta uma visão geral e comparativa dos estudos que foram mencionados neste capítulo. Estabeleceu-se uma comparação desses trabalhos com a tese, considerando os seguintes temas: avaliação de dependabilidade e performabilidade e modelo para tomada de decisão. Alguns autores utilizam modelos de dependabilidade aliados a níveis de acordo de serviço para avaliar níveis de disponibilidade e/ou confiabilidade. Outras abordagens tratam de modelos de performabilidade, avaliando a degradação do desempenho. Porém, percebemos que na literatura acadêmica há uma necessidade de estudos que avaliem cenários com atributos de dependabilidade e performabilidade que possam considerar também um planejamento de experimentos. Já alguns modelos de decisão são

complexos ou não fazem relação com dados oriundos de tais modelos e/ou parâmetros de dependabilidade e performabilidade. Sendo assim, este trabalho utiliza os benefícios do planejamento de experimentos, modelos em RBDs e SPNs, associando à seleção e *ranking* das soluções avaliadas por meio de métodos de agrupamento de dados e funções multi-critérios. Ademais, não há ferramentas gratuitas para dar suporte no processo de tomada de decisão.

Tabela 2 – Comparação dos trabalhos relacionados.

Autores	Dependabilidade	Performabilidade	Modelo de Decisão
(WEI; LIN; KONG, 2011b)	RBD e GSPN	-	-
(WEI; LIN; KONG, 2011a)	RBD e MC	-	-
(MELO et al., 2013)	RBD e SPN	-	-
(GHOSH et al., 2013)	MC e SRN	-	-
(SILVA et al., 2013b)	RBD e SPN	-	-
(SOUSA et al., 2015)	RBD e SPN	-	-
(BRILHANTE et al., 2014)	SPN	-	-
(LIU et al., 2018)	SRN	-	-
(TORRES; CALLOU; ANDRADE, 2018)	RBD e SPN	-	-
(GONCALVES et al., 2015)	-	SPN	-
(QIU et al., 2015)	-	MC	-
(WANG; CHANG; LIU, 2016)	-	CTMC e Rede de Filas	-
(EVER, 2017)	-	CTMC e Rede de Filas	-
(MACIEL et al., 2017)	-	SPN	-
(MO et al., 2018)	-	CTMC	-
(REHMAN; HUSSAIN; HUSSAIN, 2012b)	-	-	(Min-Max,
	-	-	Max-Mix,
	-	-	Programming,
	-	-	TOPSIS,
	-	-	ELECTRE,
	-	-	PROMETHEE e AHP)
(MARTENS; TEUTEBERG, 2012)	-	-	Função com Pesos
(GARG; VERSTEEG; BUYYA, 2013a)	-	-	AHP
(GARG; VERSTEEG; BUYYA, 2013b)	-	-	AHP
(SHIVAKUMAR; RAVI; GANGADHARAN, 2013)	-	-	AHP
(MAITY; CHAUDHURI, 2014)	-	-	Algorítmo Genético
(LEE; SEO, 2016)	-	-	FAHP, FDM e BSC
(SACHDEVA et al., 2016)	-	-	TOPSIS
(LIU; CHAN; RAN, 2016)	-	-	TOPSIS
(DING et al., 2017a)	-	-	KRCC
(DING et al., 2017b)	-	-	KRCC
(JATOTH et al., 2018)	-	-	TOPSIS e AHP
(ABDEL-BASSET; MOHAMED; CHANG, 2018)	-	-	NAHP
(AL-JABRI et al., 2018)	-	-	Função com Pesos
Esta tese	RBD e SPN	SPN	(Min-Max, Max-Mix, TOPSIS, Pareto, Função com Pesos)

3.6 CONSIDERAÇÕES FINAIS

Este capítulo destacou os principais trabalhos que foram coletados durante a revisão da literatura sobre avaliação de dependabilidade, avaliação de performabilidade e modelos para tomada de decisão. É importante enfatizar que esta não é uma visão exaustiva dos artigos publicados e trabalhos de pesquisa relacionados. Podem haver outros artigos e teses que fizeram avanços significativos neste campo, mas com o melhor de nosso conhecimento, a combinação das características descritas na Tabela 2 é um dos principais fatores que diferenciam este trabalho ao estado atual da arte.

4 METODOLOGIA PARA PLANEJAMENTO E TOMADA DE DECISÃO

Este capítulo tem como objetivo descrever a metodologia proposta para auxiliar nas atividades de planejamento das IaaS para avaliação e tomada de decisão. A metodologia consiste na utilização de métodos para geração de cenários de infraestruturas como serviço em nuvem (IaaS), geração de modelos de dependabilidade e performabilidade, modelos de custo, avaliação dos cenários e *ranking* dos cenários avaliados considerando métodos multicritérios.

4.1 VISÃO GERAL DA METODOLOGIA

A Figura 19 ilustra a metodologia adotada e contextualiza o ambiente no qual este trabalho está inserido, destacando suas principais atividades em: concepção do ambiente, planejamento de avaliação, geração dos modelos de dependabilidade e performabilidade, geração dos modelos de custos, avaliação dos cenários e avaliação do método multicritério.

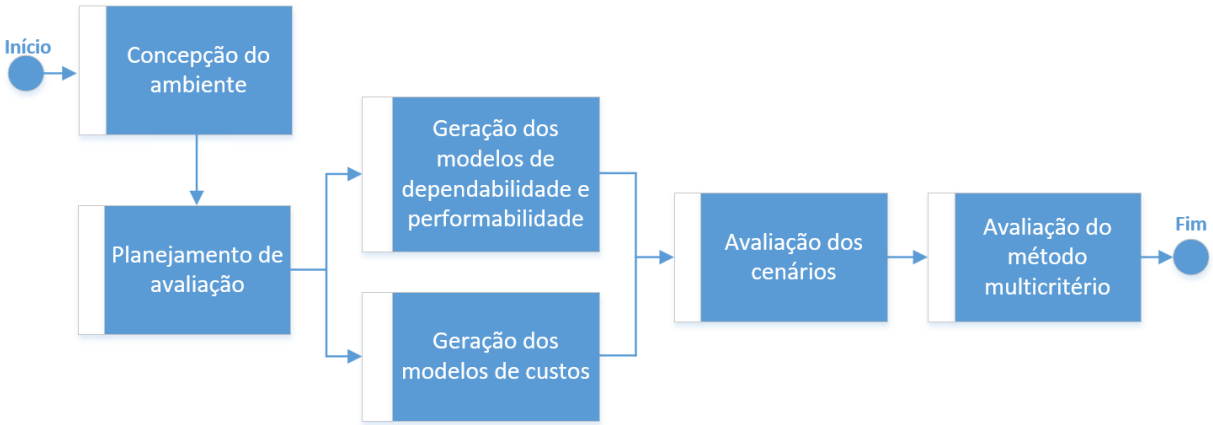


Figura 19 – Visão geral da metodologia proposta.

- Concepção do ambiente: entendimento do fucionamento de uma infraestrutura como serviço em nuvem, definição dos componentes e parâmetros a serem avaliados, modelagem e avaliação da infraestrutura inicial;
- Planejamento de avaliação: nessa fase, é definido o objetivo do plano de avaliação, os critérios que impactam no serviço da nuvem, os mecanismos de tolerância a falha, a quantidade fatores e níveis, por exemplo, mecanismos de redundância e quantidade de VMs, o método e geração do plano com os cenários a serem modelados;
- Geração dos modelos de dependabilidade e performabilidade: após o planejamento de avaliação, são gerados os modelos de dependabilidade e performabilidade;

- Geração dos modelos de custos: os modelos de custo são gerados para estimar os valores monetários para manter as atividades operacionais e de manutenção das infraestruturas propostas na fase de planejamento;

- Avaliação dos cenários: essa fase realiza as avaliações dos cenários das IaaS em nuvem, considerando o impacto dos mecanismos de redundâcia nas métricas de dependabilidade, performabilidade e custos;

- Avaliação do método multicritério: O método multicritério auxilia na geração do *ranking* dos serviços na nuvem (IaaS) a partir da obtenção dos resultados dos modelos de dependabilidade, performabilidade e custo. Por meio de funções multicritérios, o tomador de decisão pode ordenar e selecionar um conjunto de soluções que atendam os requisitos estabelecidos na fase inicial.

Salientamos que, a metodologia proposta pode ser utilizada por usuários com experiência em modelos combinatoriais e modelos baseados em espaço de estados para o planejamento e tomada de decisão de infraestruturas como serviço em nuvem privada. É importante ressaltar que esta metodologia também pode ser adotada para auxiliar na modelagem de diversos sistemas que apresentem características semelhantes às abordadas neste trabalho.

4.2 CONCEPÇÃO DO AMBIENTE

Para realizar o planejamento de avaliação de infraestruturas como serviço em nuvem é fundamental entender como o ambiente da infraestrutura funciona, identificando os critérios que podem ser representados e avaliados através de modelos. É preciso também identificar uma solução que possibilite a configuração e a representação do ambiente em nuvem, para assim, realizar as avaliações e coleta de parâmetros de dependabilidade. Assim, com o objetivo de realizar as primeiras avaliações, os principais componentes, parâmentros e funcionalidades devem ser delimitados e representados a partir de uma infraestrutura inicial.

A Figura 20 apresenta o fluxo de atividades para a concepção do ambiente com o intuito de realizar a avaliação da infraestrutura inicial. Para representar uma dada infraestrutra através de modelos de dependabilidade por exemplo, é necessário inicialmente obter os parâmetros de tempo médio entre defeitos e reparos de um dado serviço. A partir da obtenção dos parâmetros e entendimento do funcionamento do serviço é possível representar uma infraestrutura incial. A modelagem de dependabilidade pode ser realizada por meio de modelos com e sem espaço de estados (por exemplo, modelos de redes de Petri estocásticos e diagrama de bloco de confiabilidade). Após avaliada uma infraestrutura inicial, é possível estabelecer modificações e melhorias na estrutura do modelo, como por exemplo, sugerir mecanismos de redundância.

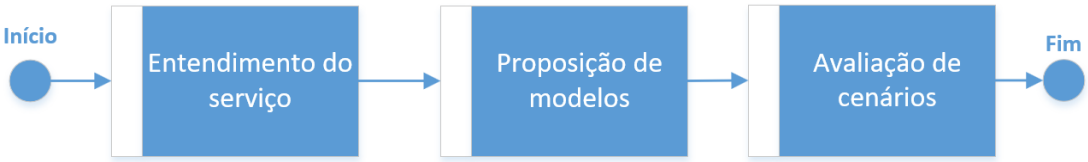


Figura 20 – Concepção do ambiente.

A compreensão da concepção do ambiente é útil para evitar erros na interpretação dos componentes do ambiente escolhido, evitando assim, comprometer as demais etapas da metodologia. Essa etapa é essencial, pois possibilita tomar conhecimento das técnicas de modelagem e avaliação a serem adotadas.

4.3 PLANEJAMENTO DE AVALIAÇÃO

A fase de planejamento de avaliação, por sua vez, utiliza o conceito de planejamento de experimentos (em inglês, Design of Experiments (DoE)). DoE pode ser aplicado a qualquer área de pesquisa que tenha o intuito de melhorar ou otimizar sistemas, produtos e processos. Os experimentos são utilizados para estudar o comportamento de processos e sistemas (MONTGOMERY, 2013). Nesse contexto, para a realização do planejamento de avaliação é necessário o estabelecimento de alguns passos. Para tanto, tem-se que:

- Descrever o objetivo do plano de avaliação;
- Definir os critérios (variáveis independentes) que impactam no serviço da nuvem (variáveis dependentes);
- Identificar os mecanismos de tolerância a falha;
- Verificar a quantidade de fatores (critérios) e níveis (variação qualitativa ou quantitativa);
- Escolher o método de geração do plano (por exemplo, fatorial completo);
- Gerar os cenários a serem modelados (combinações de fatores e níveis).

A Figura 21 apresenta o fluxo de atividades para o planejamento de avaliação. O primeiro passo define o objetivo do plano. Dessa forma, este trabalho utiliza o planejamento de avaliação com o intuito de gerar alternativas de infraestruturas como serviço (IaaS) em nuvem pública e privada que apresentem alta disponibilidade e baixo custo. No planejamento de avaliação, são investigadas quais variáveis manipuláveis (fatores) podem influenciar e gerar efeitos na variável resposta (critérios). As variáveis de respostas para o estudo são: disponibilidade, disponibilidade orientada à capacidade, indisponibilidade, confiabilidade e custo das IaaS. Os fatores considerados para analisar o impacto das variávais de saída são: número de nós, número de máquinas virtuais, mecanismos de

redundância e tipo de serviço. Os níveis ou tratamentos são as escalas utilizadas para manipular os fatores.

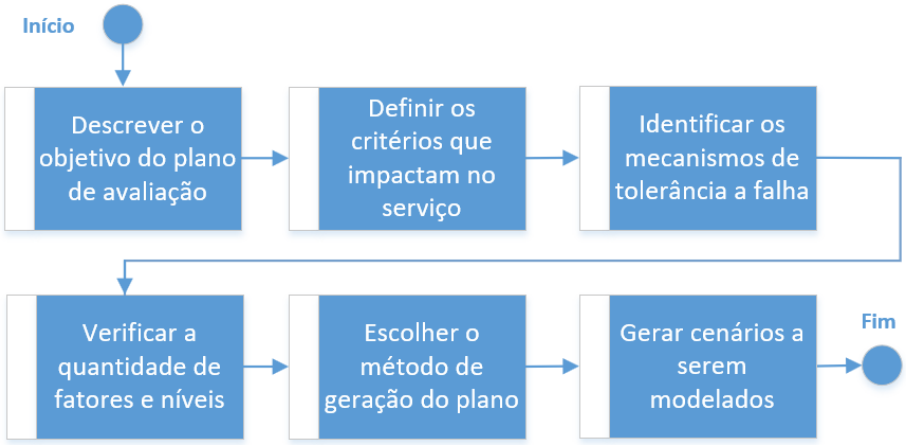


Figura 21 – Planejamento de Avaliação.

Após identificados os fatores e níveis, o plano de avaliação pode ser gerado. Este trabalho utiliza o método fatorial completo, ou seja, possibilita a geração de todas as combinações possíveis. A Tabela 3, a seguir, ilustra um possível exemplo dos fatores e níveis que podem ser aplicados no planejamento de avaliação. Para a geração dos cenários, é possível escolher o plano fatorial completo ou uma fração do experimento fatorial. A título de representação, considerou-se um plano fatorial completo, ou seja, a inclusão de todas as combinações possíveis entre os níveis e fatores. Os cenários gerados nesta fase foram representados no passo seguinte através de modelos combinatoriais e modelos baseados em espaço de estados.

Tabela 3 – Fatores e Níveis do Planejamento de Avaliação.

Fatores	Níveis			
Número de Nós	1	2		
Número de VMs	1	2	3	
Tipos de Serviço	Ouro	Prata	Bronze	
Mecanismo de Redundância	N/R	*Hot*	*Cold*	*Warm*

4.4 GERAÇÃO DOS MODELOS DE DEPENDABILIDADE E PERFORMABILIDADE

A atividade de geração dos modelos de dependabilidade e performabilidade está associada à representação dos cenários de infraestruturas como serviço em nuvem privada gerados no planejamento de avaliação. Nesta fase, os cenários são modelados considerando as seguintes variáveis: número de nós, número de VMs, tipo de serviço e mecanismos de redundância. Para a modelagem, é utilizada uma abordagem hierárquica baseada em diagrama de blocos de confiabilidade (RBDs) e redes de Petri estocásticas (SPNs). Os

modelos RBD são utilizados para representar os componentes da infraestrutura de uma nuvem e mecanismos de redundância. Já os modelos SPNs permitem a representação de mecanismos com redundância que utilizem dependências de reparos.

A Figura 22 ilustra o fluxo para geração dos modelos de dependabilidade. Os passo são definidos como: compreensão do cenário, obtenção dos parâmetros e métricas de dependabilidade e performabilidade, agrupamento dos componentes, geração do modelo RBD das IaaS e geração do modelo SPN das IaaS.

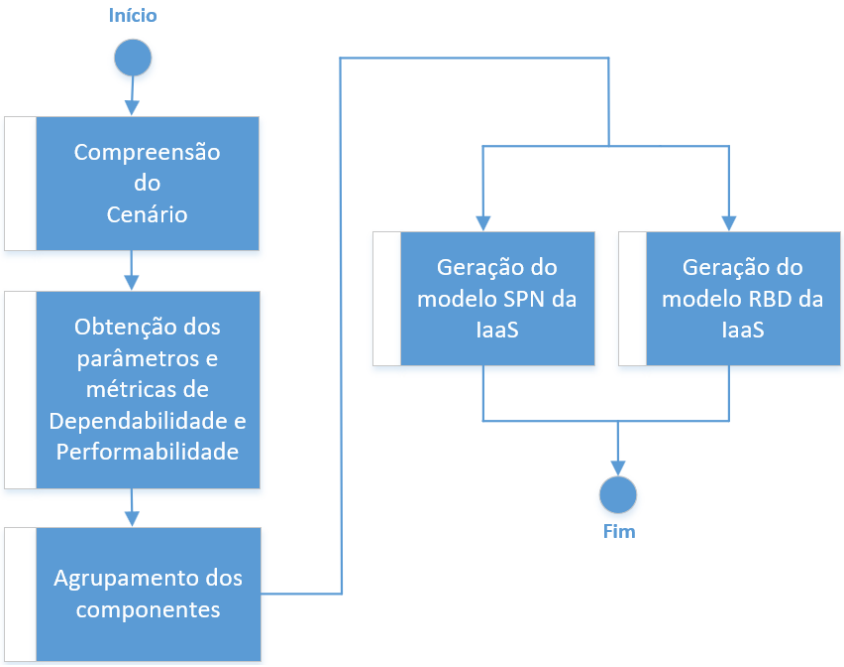


Figura 22 – Geração dos Modelos de Dependabilidade e Performabilidade.

- **Compreensão do Cenário**: entendimento das características das infraestruturas de nuvem e dos componentes. Essa atividade define os parâmetros e métricas (dependabilidade e performabilidade) utilizados no processo de avaliação. A partir dos parâmetros, é possível identificar quais componentes são mais críticos nas IaaS, para, assim, direcionar mecanismos para o tratamento de interrupção (ex: redundância *hot standby*, *cold standby* e *warm standby*).

- **Obtenção dos parâmetros e métricas de dependabilidade e performabilidade**: nesse momento, são obtidos os parâmetros e as métricas de dependabilidade e performabilidade utilizados para os mecanismos de tratamento de redundância dos componentes das IaaS. Os dados podem ser obtidos através de medições, por meio dos fabricantes dos componentes, ou na literatura científica. Os parâmetros de dependabilidade, por sua vez, são o tempo médio para defeito do componente, tempo médio para o reparo e acionamento dos mecanismos de redundância. Já as métricas para avaliação de dependabilidade são a disponibilidade, indisponibilidade

(*downtime*), confiabilidade, e capacidade orientada a disponibilidade. Os parâmetros de performabilidade são o tamanho médio de requisões na fila e utilização dos recursos da VMs (processador e disco).

- **Agrupamento dos componentes**: essa etapa corresponde à combinação ou ligação dos componentes que compõem uma IaaS em nuvem. Esse agrupamento representa o modo operacional dos componentes, ou seja, como uma IaaS é representada. A combinação também pode reduzir a complexidade de toda a estrutura a ser modelada. Podem-se citar, como exemplo, o agrupamento dos componentes que representam uma máquina virtual.

- **Geração do modelo RBD da IaaS**: após a finalização das etapas anteriores, os modelos de dependabilidade são representados através do diagrama de bloco de confiabilidade (RBD). Os modelos RBDs são utilizados com o intuito de estimar a disponibilidade, confiabilidade e a indisponibilidade de uma IaaS. Nos cenários que não necessitam da representação de mecanismos de redundância, com dependência de reparo o RBD é utilizado. Assim, na necessidade de representar exemplos que abordem a relação de dependência no reparo redundante entre os componentes da IaaS em nuvem, os modelos SPN são usados para representar as IaaS em nuvem.

- **Geração do modelo SPN da IaaS**: a geração de modelos SPNs para obtenção das métricas de disponibilidade, confiabilidade e indisponibilidade é gerada seguindo as representações dos mecanismos de redundância e reparo com dependência (*cold standby* e *warm standby*). Adicionalmente, os modelos de performabilidade são representados a partir das SPNs.

4.5 GERAÇÃO DO MODELO DE CUSTO

A atividade de geração do modelo de custo está vinculada à identificação de características das infraestruturas como serviço (IaaS), que possam considerar custos para operação e manutenção. Nessa atividade, os modelos de custo são obtidos a partir de equações que visam estimar os custos considerando o conceito de Custo Total de Propriedade (TCO) (WEIL; MAHER, 2005). A Figura 23 apresenta os passos utilizados para obteção das equações.

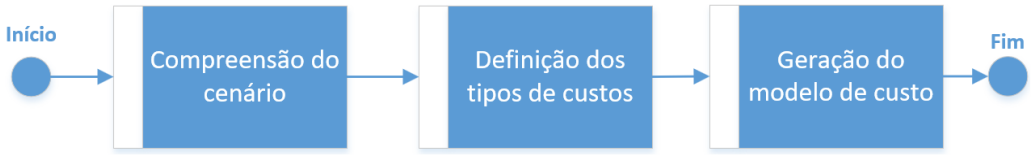


Figura 23 – Geração de Modelo de Custo.

- **Compreensão do cenário**: define as características essenciais da infraestrutura como serviço em nuvem que são passíveis de custos. Essa fase deve ser avaliada considerando os conceitos de TCO. Os parâmetros de custo são relacionados à aquisição de componentes do ambiente de Tecnologia da Informação (TI), à aquisição de componentes complementares para mecanismos redundantes, e as atividades de manutenção e operação do ambiente de IaaS na nuvem.

- **Definição dos tipos de custos**: após o entendimento dos tipos de custos, esta fase tem o objetivo de agrupar os parâmetros identificados. Assim, todos os custos relacionados com a manutenção são associados, assim como, com os custos para a operação da infraestrutura.

- **Geração do modelo de custo**: por fim, são geradas as equações e parâmetros de custos para estimar os custos da aquisição, manutenção e operação das IaaS.

4.6 AVALIAÇÃO DOS CENÁRIOS

A partir dos modelos de dependabilidade, performabilidade e custo gerados anteriormente, esta etapa permite a avaliação e obtenção dos resultados considerando cada cenário. A avaliação dos cenários das IaaS em nuvem possibilita que usuários de serviços em nuvem possam utilizar os resultados no processo de tomada de decisão. Além disso, os parâmetros de mecanismos de redundância, disponibilidade e tipos de serviços, por exemplo, podem ser observados ampliando a investigação do ambiente em estudo.

Essa etapa é composta pelos seguintes atividades: avaliação dos cenários de dependabilidade e performabilidade, composição hierárquica dos resultados de dependabilidade e avaliação dos cenários de custos. A Figura 24 ilustra o fluxo das atividades.

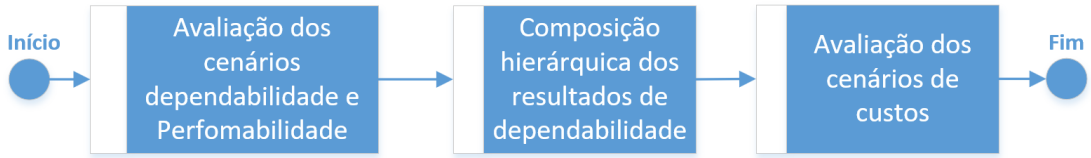


Figura 24 – Avaliação dos Cenários.

- **Avaliação dos cenários de dependabilidade e performabilidade**: representa a análise das métricas de disponibilidade, indisponibilidade e confiabilidade dos modelos de dependabilidade, assim como, à análise das métricas de performabilidade. Esses modelos são resultados do método de geração de modelos.

- **Composição hierárquica dos resultados de dependabilidade**: os resultados obtidos através dos modelos de dependabilidade são agrupados de maneira que representem o modo operacional de uma IaaS em nuvem. Essa estratégia de modelagem

hierárquica reduz a complexidade para representar a composição dos componentes de uma IaaS, assim como, dos mecanismos redundantes que possuem dependência.

- **Avaliação dos cenários de custos**: representa a análise dos custos através dos modelos de custos. Essa análise considera estimativas dos custos de manutenção, utilização e operação da IaaS em nuvem.

4.7 AVALIAÇÃO DO MÉTODO MULTICRITÉRIO

A partir dos resultados obtidos na etapa anterior, o usuário pode adotar um método multicritério para auxiliar no processo de tomada de decisão. O método é composto por um conjunto de passos: definição da função multicritério, escolha do método de agrupamento e avaliação do método multicritério. A Figura 25 ilustra os passos do método para *ranking* das IaaS.

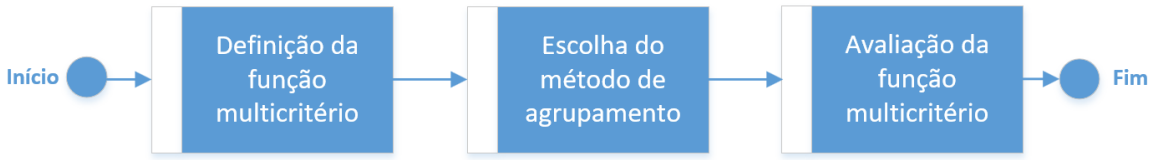


Figura 25 – Avaliação do método multicritério.

- **Definição da função multicritério**: este passo trata da formulação da função multicritério. A função é definida a partir da escolha dos critérios (por exemplo, indisponibilidade e confiabilidade), e restrições quando necessárias (atribuir ponderação para os critérios). A função multicritério pode ordenar um conjunto de cenários considerando funções que maximizem ou minimizem cada critério.

- **Escolha do método de agrupamento**: no segundo passo, é necessária a definição do método de agrupamento dos cenários para o processo de tomada de decisão. São disponibilizadas três abordagens: medidas de similaridade (distância Euclidiana, Minkowski e Manhattan), conceito de dominância de Pareto e intervalo mínimo-máximo. A medida de similaridade possibilita o calculo das distâncias (similaridade ou dissimilaridade) entre os cenários. Já o método de dominância de Pareto permite a ordenação dos resultados considerando a fronteira de Pareto. O método com intervalo mínimo-máximo, ao seu turno, possibilita a ordenação dos resultados através de uma busca por soluções que estejam dentro da definição de valores mínimos e máximos.

- **Avaliação do método multicritério**: após a definição do método de agrupamento, os valores dos critérios de cada cenário são normalizados e em seguida são avaliados conforme a função multicritério. O *ranking* dos cenários é apresentado em formato de uma matriz nxm considerando os critério definidos no primeiro passo.

O método permite gerar um conjunto de cenários ordenados considerando os critérios de interesse. Desta forma, o metódo pode auxiliar o usuário no processo de tomada de decisão quando existir vários critérios conflitantes. Ademais, uma vez que um conjunto foi ordenado, é possível que esse conjunto possa ser reordenado através de outro método.

4.8 CONSIDERAÇÕES FINAIS

Como visto, este capítulo apresentou a metodologia e os métodos utilizados na proposta que buscam auxiliar usuários no processo de tomada de decisão em infraestruturas como serviço de nuvens públicas e privadas. Os métodos apresentados possibilitam a geração de cenários de infraestruturas como serviço de nuvens, por meio de modelos de dependabilidade, performabilidade e custo. Além disso, os modelos abordados nesta metodologia são apresentados no capítulo seguinte. Os resultados da avaliação dos modelos são usados pelos métodos multicritérios para ordenação das IaaS de nuvens que atendem aos requisitos de dependabilidade, performabilidade e custo.

5 MODELOS

Neste capítulo, são apresentados os modelos para quantificar métricas como disponibilidade e confiabilidade. Estes modelos permitem estimar os custos das IaaSs e o modelo para auxiliar no processo de tomada de decisão dos cenários que representam as infraestruturas como serviço em nuven. A abordagem oferece a usuários de IaaSs em nuvem uma solução baseada em modelos hierárquicos considerando as vantagens dos diagramas de blocos de confiabilidade (RBDs), das redes de Petri estocásticas (SPNs) e das equações de custos. Por fim, o modelo para tomada de decisão permite um *ranking* ordenado dos cenários levando em consideração funções multicritérios.

5.1 MODELOS DE DEPENDABILIDADE

A atividade de geração dos modelos de dependabilidade está associada à representação de infraestruturas como serviço em nuvem. Com os modelos baseados em espaço de estados (ex.: SPN) e modelos combinatoriais (ex.: RBD), é possível considerar mecanismos com redundância (por exemplo, *coldstandby*), assim como, o uso de políticas de reparo estabelecido por Nível de Acordo de Serviço (SLA)s. Os resultados das métricas dos RBDs podem ser obtidos em um curto intevalo de tempo, isso ocorre devido ao uso de fórmulas fechadas (CHEN; TRIVEDI, 2002). Já as análises numéricas de alguns modelos SPNs podem ser mais custosos, pois inclui a geração dos espaços de estados e observação dos estados alcançáveis para a obtenção dos resultados do modelo (MACIEL; LINS; CUNHA, 1996). Por outro lado, as SPNs proporcionam uma maior variedade na representação de modelos, como por exemplo, mecanismos com dependência de reparo (BALBO, 2001).

A modelagem é realizada pela representação de cada componente da infraestrutura como serviço na nuvem. Cada componente possui uma representação dos estados operacional e falho. O estado operacional representa o estado de funcionamento em condições normais, enquanto que o estado falho representa a condição em que componentes não respondem adequadamente ao sistema em funcionamento. Os componentes são interligados de maneira a representar o modo operacional da infraestrutura computacional em nuvem.

A Figura 26 ilustra um exemplo de um modelo RBD que pode representar um componente em uma IaaS. Esse modelo, através dos paramêtros de Tempo Médio de Defeito (MTTF) e Tempo Médio de Reparo (MTTR), pode representar métricas de dependabilidade como disponibilidade, confiabilidade e indisponibilidade. Tanto a modelagem visual, como a obtenção das métricas podem ser realizadas por ferramentas de apoio a modelagem como o Mercury (MoDCS, 2018; MACIEL et al., 2017).

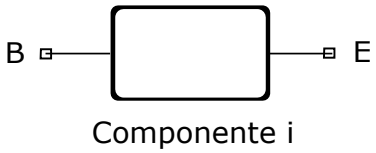


Figura 26 – Modelo RBD.

A Figura 27 apresenta um modelo SPN que permite representar um componente de uma infraestrutura como serviço em uma nuvem. Assim como no modelo RBD, o modelo SPN faz uma representação do funcionamento com os estados operacional (funcionamento) e falho. Os lugares na SPN são representados por `operacional` e `defeito`. As transições exponenciais são representadas por MTTF e MTTR. O método de disparo é *single server*[1]. A marcação ou *token* no lugar `Operacional` ilustra o estado do componente como operacional.

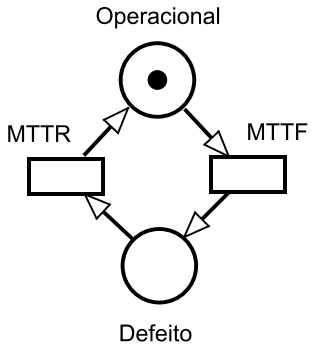


Figura 27 – Modelo SPN.

5.1.1 Modelos de Dependabilidade - RBD

Para capturar o comportamento de dependabilidade, modelos RBD foram adotados para representar infraestruturas como serviço em nuvem. Os modelos das IaaSs foram agrupados considerando o funcionamento dos componentes. Inicialmente, a infraestrutura foi composta pelo *FrontEnd*, nó e Máquina Virtual (VM). A abordagem hierárquica foi adotada para a geração dos modelos com o intuito de reduzir a complexidade destes. Essa abordagem consistiu na modelagem de componentes individualizados, dos mecanismos de redundância, e, em seguida, de sua composição para representar toda a infraestrutura.

A Figura 28 ilustra um exemplo de uma infraestrutura como serviço para nuvem privada. Para essa representação, utilizou-se o VIM[2] OpenNebula IaaS Cloud 3.6.0 (OPENNEBULA, 2018). Na análise das métricas de dependabilidade, é necessária a identificação do modo operacional e modo falho do ambiente. Para esse ambiente, foi definido, como modo operacional da IaaS, o funcionamento dos três componentes *FrontEnd*, Nó e Máquina

[1] Apenas um *token* é disparado por vez, ou seja, a capacidade de um lugar/transição é 1 (MARSAN et al., 1994).

[2] *Virtual Infrastructure Manager* (VIM)

Virtual em série. Esse modo também tem uma relação de dependência nas operações de comunicação. Assim sendo, o modelo operacional de uma infraestrutura ($IaaS_{Op}$) pode ser expresso da seguinte forma:

$$IaaS_{Op} = FE_{op} \wedge N_{op} \wedge VM_{op} \tag{5.1}$$

onde FE_{op} representa o *FrontEnd*, N_{op} o nó no cluster e VM_{op} a máquina virtual.

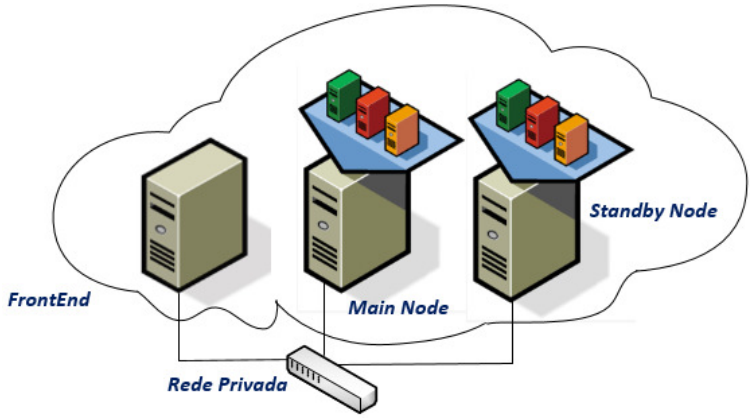


Figura 28 – Arquitetura da infraestrutura como serviço em nuvem.

A Figura 29 mostra o modelo RBD que representa o gerenciador da nuvem (*Frontend*). O modelo RBD é composto por três componentes conectados em série: *Hardware* (Hw), Sistema Operacional (SO) e Servidor de Gerenciamento (Mng). O componente *frontend* atua como gerenciador dos processos necessários para a comunicação entre os nós da nuvem. Já o *hardware* representa os dispositivos que compõem um servidor (memória, processador e rede). E, por fim, o sistema operacional representa o sistema utilizado para funcionamento do servidor. O servidor de gerenciamento é o *software* responsável por monitorar e gerenciar a nuvem. Assim, o modo operacional do *fronted* ($Front_{Op}$) pode ser expresso da seguinte forma:

$$Front_{Op} = Hw_{op} \wedge OS_{op} \wedge Mng_{op} \tag{5.2}$$

onde Hw_{op} representa o *hardware*, o OS_{op} o sistema operacional e Mng_{op} o gerenciador da nuvem.

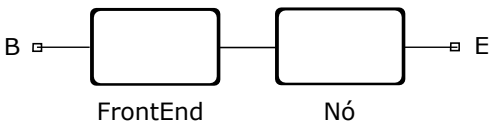


Figura 29 – Modelo RBD do *Frontend*.
Fonte: (MELO et al., 2013; ARAUJO et al., 2014)

A Tabela 4 apresenta os parâmetros que podem ser manipulados no modelo de dependabilidade. Os valores apresentam a possibilidade de serem especificados através de dados

fornecidos por fabricantes ou obtidos por medição. Os parâmetros mostram os tempos médios de defeito e reparo de cada componente. Alguns fabricantes podem informar os valores de defeitos em função de uma taxa (λ). E os modelos também suportam o uso de taxa média de reparo e defeito.

Tabela 4 – Parâmetros de dependabilidade para o RBD *FrontEnd*.

Parâmetro	Descrição
`MTTF_hw-front`	Tempo médio de defeito do hardware
`MTTR_hw-front`	Tempo médio de reparo do hardware
`MTTF_os-front`	Tempo médio de defeito do sistema operacional
`MTTR_os-front`	Tempo médio de reparo do sistema operacional
`MTTF_Mng-front`	Tempo médio de defeito do gerenciamento da nuvem
`MTTR_Mng-front`	Tempo médio de reparo do gerenciamento da nuvem

A Figura 30 traz o modelo RBD do nó principal. O nó é representado pelos recursos computacionais para a implantação das máquinas virtuais. As imagens das VMs podem ser guardadas em dispositivos de armazenamento de dados e quando requisitadas, há a possibilidade de serem transferidas via rede. O nó é composto por cinco componentes de série: *hardware*, sistema operacional, servidor de gerenciamento, máquina virtual (VM) e um serviço. Portanto, o modo operacional do nó (`Nó_Op`) pode ser expresso da seguinte forma:

$$\texttt{Nó\_Op} = \texttt{Hw\_op} \wedge \texttt{OS\_op} \wedge \texttt{Mng\_op} \wedge \texttt{VM\_op} \wedge \texttt{Ser\_op} \tag{5.3}$$

onde `Hw_op` representa o *hardware*, o `OS_op` o sistema operacional, `Mng_op` o gerenciador da nuvem, `VM_op` a máquina virtual e `Ser_op` um serviço.

Figura 30 – Modelo RBD do Nó.
Fonte: (MELO et al., 2013; ARAUJO et al., 2014)

Seguindo, a Tabela 5 mostra os parâmetros de dependabilidade que podem ser avaliados no modelo de dependabilidade RBD, conforme a especificação dos fabricantes dos componentes. Os parâmetros representam os tempos médios de defeito e reparo de cada componente. Além da utilização de dados de fabricantes para a fase de modelagem, analistas, técnicos e gestores de ambientes computacionais em nuvens podem utilizar dados históricos de seus ambientes. Dessa maneira, os valores podem contribuir para a obtenção de resultados mais aproximados.

Tabela 5 – Parâmetros de dependabilidade para o RBD Nó.

Parâmetro	Descrição
MTTF_hw-node	Tempo médio de defeito do hardware
MTTR_hw-node	Tempo médio de reparo do hardware
MTTF_os-node	Tempo médio de defeito do sistema operacional
MTTR_os-node	Tempo médio de reparo do sistema operacional
MTTF_Mng-node	Tempo médio de defeito do gerenciamento da nuvem
MTTR_Mng-node	Tempo médio de reparo do gerenciamento da nuvem
MTTF_VM-node	Tempo médio de defeito da máquina virtual
MTTR_VM-node	Tempo médio de reparo da máquina virtual

De acordo com a estratégia de modelagem hierárquica adotada, uma vez que os componentes *frontend* e o nó principal foram modelados, o passo seguinte foi a composição dos modelos. A composição dos modelos auxilia no processo de modelagem, dado que quanto menor a quantidade de blocos RBDs conectados, mais prático ficará o seu gerenciamento. O modelo de composição é ilustrado na Figura 31. Assim, cada submodelo representado por um bloco foi interligado formando um arranjo em série. Esse modelo de dependabilidade foi adotado para representar uma IaaS em nuvem. Ademais, a partir desse modelo é possível gerar um conjunto de arranjos para representar N infraestruturas em paralelo visando elevar a disponibilidade. Com isso, o modo operacional de uma infraestrutura (IaaS_Op) pode ser expresso da seguinte forma:

$$\texttt{IaaS\_Op} = \texttt{FE\_op} \wedge \texttt{N\_op} \tag{5.4}$$

onde FE_op representa o *Frontend* e N_op o nó.

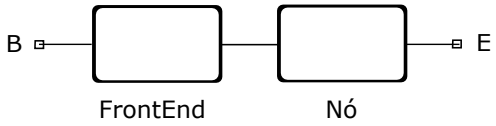


Figura 31 – Modelo RBD do *Frontend* e Nó.
Fonte: (MELO et al., 2013; ARAUJO et al., 2014)

Em ambientes computacionais em nuvem, é necessário que os serviços ofereçam qualidade na prestação dos serviços, pois é importante a continuidade no fornecimento dos serviços mesmo diante de interrupções. Nesse caso, mecanismos de tratamento de interrupção podem ser utilizados para garantir a disponibilidade e confiabilidade dos serviços. O modelo a seguir adota um mecanismo considerando uma redundância (ativo-passivo) [3]. O modelo é representado por um *frontend* e uma redundância no nó do tipo *HotStandby*.

[3] Os mecanismos de redundância do tipo ativo-passivo são utilizados quando os componentes principais atendem à demanda do sistema e os componentes secundários estão em espera (BAUER; ADAMS, 2012).

A Figura 32 ilustra o mecanismo *HotStandby* considerando um modelo RBD. Um componente com redundância *HotStandby* é baseado em um componente redundante ativo. Nesse mecanismo de tratamento de interrupção, quando o componente principal está em estado de falha, o componente redundante pode ser substituído sem atraso na ativação. Sendo assim, a principal característica de um componente com uma redundância *HotStandby* é a ausência de um tempo de ativação (se comparado com as redundâncias *cold standby* e *warm standby*).

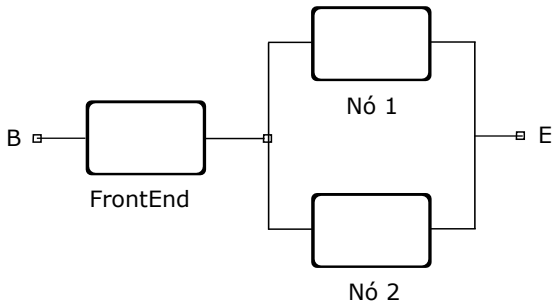


Figura 32 – Modelo RBD com redundância *HotStandby* no Nó.
Fonte: (MELO et al., 2013; ARAUJO et al., 2014)

Esse tipo de redundância proporciona o aumento na disponibilidade do ambiente IaaS, pois quando um nó apresenta um defeito e interrompe o serviço, o outro nó automaticamente assume o seu lugar. Por consequência, o modo operacional dos nós (`Hot_Op`) pode ser expresso da seguinte forma:

$$\texttt{Hot\_Op} = \texttt{N1\_op} \vee \texttt{N2\_op} \tag{5.5}$$

onde `N1` representa o nó principal e `N2` o nó reserva (redundante). A Tabela 6 mostra, a seguir, os parâmetros de dependabilidade utilizados para o modelo RBD *Hotstandby*.

Tabela 6 – Parâmetros de dependabilidade para o RBD com Nó redundante.

Parâmetro	Descrição
`MTTF_N1`	Tempo médio de defeito do Nó 1
`MTTR_N1`	Tempo médio de reparo do Nó 1
`MTTF_N2`	Tempo médio de defeito do Nó 2
`MTTR_N2`	Tempo médio de reparo do Nó 2

Na subseção seguinte, são apresentados os modelos de dependabilidade considerando redes de Petri estocásticas. Esse modelos foram adotados para a modelagem de estratégias de redundância com dependência (*cold* and *warm*), dado que modelos RBD são bastante utilizados para representar redundâncias do tipo ativa (*hot*).

5.1.2 Modelos de Dependabilidade - SPN

Este trabalho adota a extensão de rede de Petri denominada "redes de Petri estocásticas" (SPN) (MARSAN, 1989), que permite a associação de tempos probabilísticos para transições usando distribuição exponencial. As redes de Petri têm sido usada para modelar sistemas tolerantes a falhas e avaliar várias medidas de disponibilidade e confiabilidade (MATOS et al., 2017; MACIEL et al., 2017). Além disso, a SPN permite a adoção de técnicas de simulação para a obtenção de métricas de dependabilidade como uma alternativa para a geração da cadeia de Markov. O modelo SPN a ser mostrado foi adotado para representar mecanismos de tratamento de interrupção considerando componentes com dependência de reparo. Os modelos de redundância adotados foram: *Cold Standby* e *Warm Standby*. Assim como no modelo RBD, com os modelos SPNs foi possível a obtenção das métricas de dependabilidade (disponibilidade, confiabilidade e indisponibilidade). Tanto a modelagem visual como a obtenção das métricas por análise numérica e análise estacionária podem ser realizadas por ferramentas de apoio à modelagem como o Mercury (MoDCS, 2018; MACIEL et al., 2017).

Um modelo redundante *coldstandby* consiste em um módulo de reserva não ativo, que é apenas ativado quando o módulo ativo principal falha. A Figura 33 ilustra o modelo SPN deste sistema, que inclui quatro lugares, a saber: `COMPONENTE_1_ON`, `COMPONENTE_1_OFF`, `COMPONENTE_2_ON` e `COMPONENTE_2_OFF`, onde, respectivamente, representam os estados operacionais e a falha de ambos, e o módulo principal e reserva.

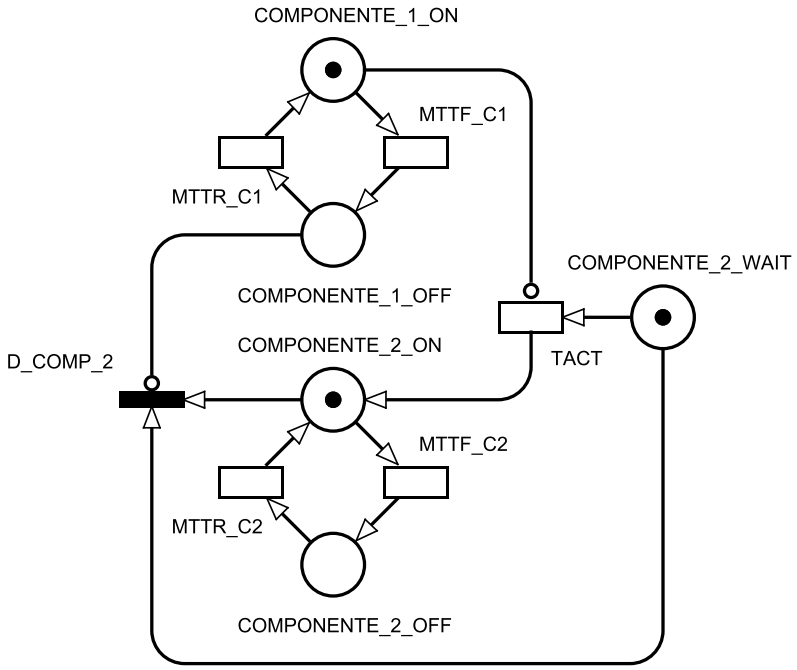


Figura 33 – Modelo SPN para a redundancia *coldstandby*.
Fonte: (ARAUJO et al., 2014; ARAUJO et al., 2018).

O módulo reserva (`COMPONENTE_2`) está inicialmente desativado, assim, não há *tokens* inicialmente armazenados nos lugares `COMPONENTE_2_ON` e `COMPONENTE_2_OFF`. Quando o

módulo principal falha, a transição TACT é disparada para ativar o módulo redundante. A transição imediata chamada D_COMP_2 representa a ação de desativação do módulo redundante quando o módulo principal é recuperado. A presença de um *token* no lugar COMPONENTE_2_WAIT representa que o módulo redundante está desativado.

Por conseguinte, o modo operacional do modelo *cold* pode ser expresso da seguinte forma:

$$Cold_{Op} = (\texttt{Componente\_1\_ON} = 1 \vee \texttt{Componente\_2\_ON} = 1) \tag{5.6}$$

onde Componente_1_ON representa o módulo principal e Componente_2_ON o módulo reserva ou redundante.

A Tabela 7 apresenta a descrição das transições utilizadas no modelo SPN *Coldstandby*. Essas transições podem ser manipuladas com o intuito de representar políticas de reparo de serviço. É possível, portanto, analisar situações onde o tempo médio de reparo seja menor ou maior considerando políticas de reparo de serviço. Além disso, o tempo médio de defeito pode representar componentes com maior ou menor confiabilidade, por exemplo. Os modelos com mecanismos de redundância podem ser utilizados na representação de ambientes computacionais em nuvem, e associados na proposição de políticas de reparo de serviço.

Tabela 7 – Parâmetros de dependabilidade para a SPN *ColdStandby*.

Parâmetro	Descrição
MTTF_C1	Tempo médio de defeito do componente 1
MTTR_C1	Tempo médio de reparo do componente 1
MTTF_C2	Tempo médio de defeito do componente 2
MTTR_C2	Tempo médio de reparo do componente 2
TACT	Tempo médio de ativação do componente 2
D_COMPD_2	Desativação do componente 2

A Tabela 8, a seguir, descreve os atributos das transições do modelo SPN (Fig. 33).

Tabela 8 – Atributos das transições.

Transição	Tipo	Semântica	Peso	Prioridade
MTTF_C1	Temporizada	Single Server	-	-
MTTR_C1	Temporizada	Single Server	-	-
MTTF_C2	Temporizada	Single Server	-	-
MTTR_C2	Temporizada	Single Server	-	-
TACT	Temporizada	Single Server	-	-
D_COMP_2	Imediata	-	1	1

O modelo com redundância *WarmStandby*, por sua vez, é representado na Figura 34. A transição TACT representa o tempo médio de ativação do módulo reserva. Os luga-

res nomeados COMPONENTE_1_ON e COMPONENTE_1_OFF representam os estados de atividade e inatividade do módulo principal (COMPONENTE_1). Os lugares COMPONENTE_2_ON e COMPONENTE_2_OFF representam o módulo de reposição em estado de espera, mas não-operacional. Já os lugares COMPONENTE_2_OP_ON e COMPONENTE_2_OP_OFF representam o módulo de reposição no estado operacional. O módulo de reposição começa no estado não operacional por estar desativado. De forma semelhante ao modelo *cold*, quando o componente principal falha, a transição TACT é disparada para ativar o módulo de reposição. A transição imediata nomeada de D_COMP_2_OP tem o mesmo comportamento no modelo *cold standby*, ou seja, desativar o módulo reserva quando o módulo principal é reparado. O mecanismo de ativação na redundância *cold standby* é mais lento do que no *warm standby*, isso porque o módulo de redundância está desativado. Enquanto que no modelo *warm*, o módulo de reposição está em estado de espera, mas não está pronto.

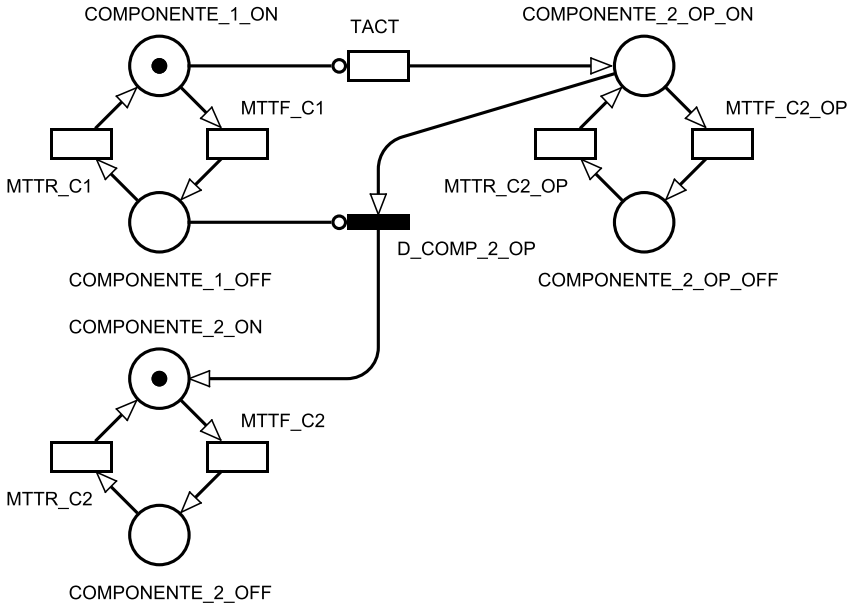


Figura 34 – Modelo SPN para a redundancia *warmstandby*.
Fonte: (ARAUJO et al., 2014; ARAUJO et al., 2018).

Assim sendo, o modo operacional do modelo *warm* pode ser expresso da seguinte forma:

$$Warm_{Op} = (\texttt{Componente\_1\_ON} = 1 \vee \texttt{Componente\_2\_OP\_ON} = 1) \qquad (5.7)$$

onde Componente_1_ON representa o componente principal e Componente_2_OP_ON o componente reserva ou redundante.

A Tabela 9 apresenta a descrição das transições utilizadas no modelo *WarmStandby*. Essas transições podem ser variadas conforme o objetivo do técnico, analista e gestor das IaaS com o intuito de representar políticas de serviços. De forma similar ao modelo *ColdStandby*, o tempo médio de reparo e/ou tempo médio de defeito podem ser modificados conforme políticas de reparo ou pela utilização de componentes mais confiáveis. Modelos

de redundância que oferecem a possibilidade de modificação dos tempos de reparo e/ou falha podem auxiliar no planejamento de ambientes computacionais em nuvem.

Tabela 9 – Parâmetros de dependabilidade para a SPN *WarmStandby*.

Parâmetro	Descrição
MTTF_C1	Tempo médio de defeito do componente 1
MTTR_C1	Tempo médio de reparo do componente 1
MTTF_C2	Tempo médio de defeito do componente 2 (Espera)
MTTR_C2	Tempo médio de reparo do componente 2 (Espera)
MTTF_C2_OP	Tempo médio de defeito do componente 2 (Operacional)
MTTR_C2_OP	Tempo médio de reparo do componente 2 (Operacional)
TACT	Tempo médio de ativação do componente 2
D_COMP_2_OP	Desativação imediata do componente 2

A Tabela 10 descreve os atributos das transições do modelo SPN (Fig. 34). Detalhes sobre os atributos das SPNs estão descritos no item 2.7.8 do Capítulo 2.

Tabela 10 – Atributos das transições.

Transição	Tipo	Semântica	Peso	Prioridade
MTTF_C1	Temporizada	Single Server	-	-
MTTR_C1	Temporizada	Single Server	-	-
MTTF_C2	Temporizada	Single Server	-	-
MTTR_C2	Temporizada	Single Server	-	-
TACT	Temporizada	Single Server	-	-
D_COMP_2	Imediata	-	1	1

5.1.3 Modelo de Desempenho - SPN

O estudo de dependabilidade em infraestrutura como serviço é importante para investigar métricas como disponibilidade e confiabilidade. Entretanto, tratando-se de serviços sob demanda, também é interessante compreender e analisar o desempenho da infraestrutura. Assim, associar análise de desempenho e métricas de dependabilidade nos permite investigar não só o desempenho do sistema em estado operacional, como também examinar o efeito de eventos de falhas e atividades de reparo na degradação do desempenho de ambiente. Tanto a modelagem visual quanto a obtenção das métricas por análise numérica e análise estacionária podem ser realizadas por ferramentas de apoio à modelagem como o Mercury (MoDCS, 2018; MACIEL et al., 2017).

O modelo SPN a ser apresentado foi adotado para representar o funcionamento de um dado serviço (por exemplo, um repositório digital) que recebe requisições de consulta e permite o compartilhamento de objetos digitais, como artigos, livros, teses e dissertações. A seguir, o modelo de desempenho é apresentado através de submodelos para auxiliar na

compreensão do seu funcionamento. O submodelo de desempenho ilustrado na Figura 35 representa o envio de requisições ao serviço na nuvem. A ação da transição temporizada (`T0`) representa o envio de requisições ao serviço e a transição imediata (`T1`) representa a entrada da requisição na fila (*buffer*) conforme a Figura 35.

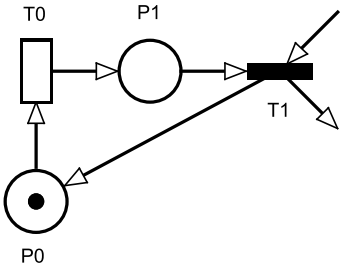


Figura 35 – Submodelo SPN - Envio de requisições.

A Tabela 11 descreve os atributos das transições do submodelo SPN (Fig. 35). Detalhes sobre os atributos das SPNs estão descritos no item 2.7.8 do Capítulo 2.

Tabela 11 – Atributos das transições.

Transição	Tipo	Semântica	Peso	Prioridade
`T0`	Temporizada	Single Server	-	-
`T1`	Imediata	-	1	1

A Figura 36 representa a entrada das requisições na fila do serviço na nuvem. A ação da transição imediata `T1` representa a entrada das requisições na fila; para isso, é necessário que o lugar `P_IDLE_QUEUE` apresente *tokens*, ou seja, `n > 0`. Com a ação da transição imediata `T1`, um *token* é armazenado no lugar `P_BUSY_QUEUE`. A acão da transição imediata `T2` permite a entrada da requisição no recurso de processamento conforme a Figura 37.

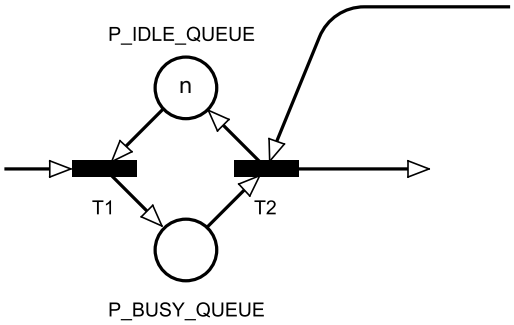


Figura 36 – Modelo SPN - Fila de requisições.

A Tabela 12 descreve os atributos das transições do modelo SPN (Fig. 36).

Tabela 12 – Atributos das transições.

Transição	Tipo	Semântica	Peso	Prioridade
T1	Imediata	-	1	1
T2	Imediata	-	1	1

A Figura 37 representa a entrada das requisições no processamento do serviço na nuvem. A ação da transição imediata T2 representa a chegada das requisições para o processamento. Assim, é preciso que o lugar P_IDLE_CPU apresente *tokens* disponíveis, ou seja, nCPU > 0. Com a ação da transição imediata T2, um *token* é armazenado no lugar P_BUSY_CPU. A acão da transição temporizada T_CPU_PR representa o processamento da requisição no recurso de processamento. Em seguida, um *token* será armazenado no lugar P6, permitindo que a transição imediata T4 possa ser disparada em seguida. A ação dessa transição permite a liberação do recurso computacional e a continuidade do processo. Após a ação da transição imediata T4, será possível a entrada na requisição no recurso de leitura/escrita do serviço no recurso de armazenamento conforme a Figura 38.

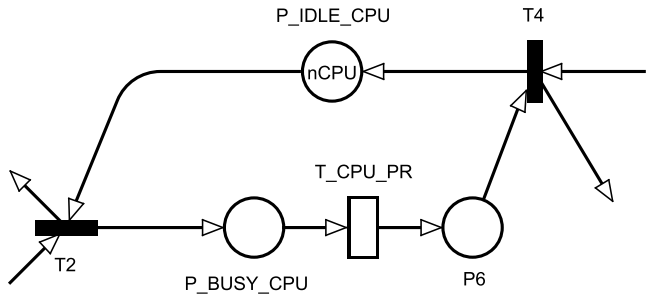


Figura 37 – Modelo SPN - Processamento das requisições.

A Tabela 13 descreve os atributos das transições do submodelo SPN (Fig. 37).

Tabela 13 – Atributos das transições.

Transição	Tipo	Semântica	Peso	Prioridade
T_CPU_PR	Temporizada	*Single Server*	-	-
T2	Imediata	-	1	1
T4	Imediata	-	1	1

A Figura 38 representa a entrada das requisições para a operação de entrada/saída no dispositivo de armazenamento na nuvem. A ação da transição imediata T4 representa a chegada das requisições para o recurso de armazenamento; desta forma, é preciso que o lugar P_IDLE_DISK apresente *tokens* disponíveis, ou seja, nDISK > 0. Com a ação da transição imediata T4, um *token* é armazenado no lugar P_BUSY_DISK. A acão da transição temporizada T_DISK_IO representa a operação de leitura/escrita no recurso de armazenamento. Em seguida, um *token* será armazenado no lugar P7, permitindo assim que a

transição imediata `T5` possa ser disparada em seguida. A ação dessa transição permite a liberação do recurso de armazenamento e a continuidade do processo, ou seja, a operação de uma requisição no recurso de processamento e armazenamento.

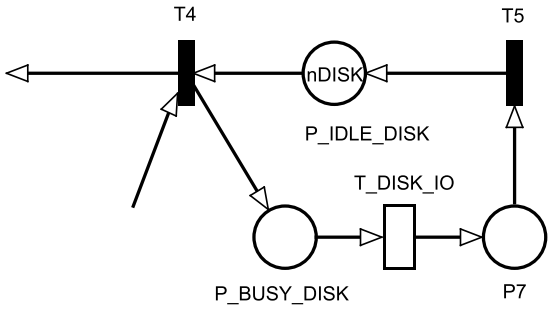


Figura 38 – Modelo SPN - Entrada/Saída das requisições no recurso de armazenamento.

A Tabela 14 descreve os atributos das transições do modelo SPN (Fig. 38).

Tabela 14 – Atributos das transições.

Transição	Tipo	Semântica	Peso	Prioridade
`T_DISK_IO`	Temporizada	*Single Server*	-	-
`T4`	Imediata	-	1	1
`T5`	Imediata	-	1	1

A Figura 39 ilustra o modelo de desempenho proposto que representa atividades de processamento e operações de entrada/saída no recurso de armazenamento de um serviço em uma dada infraestrutra na nuvem. Como descrito, os submodelos de envio das requisições, fila, processamento e operações de entrada/saída no dispositivo de armazenamento foram apresentados de modo seccionado para auxiliar no entendimento de suas funcionalidades.

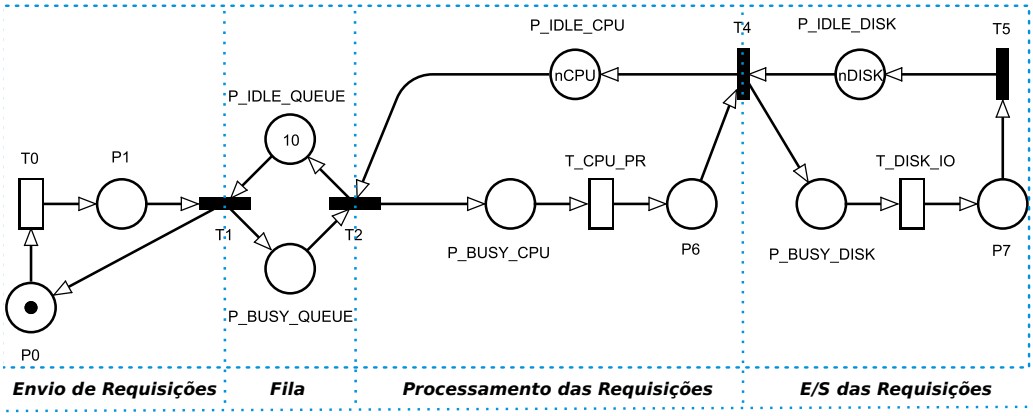


Figura 39 – Modelo SPN - Desempenho de um serviço na nuvem.

5.1.4 Modelo para Disponibilidade Orientada à Capacidade - SPN

A avaliação de disponibilidade orientada à capacidade, geralmente, se aplica a infraestruturas com redundância do tipo *hot standby*, onde os componentes podem estar dispostos de modo ativo/ativo (A/A), aumentando a disponibilidade e reduzindo a degradação do serviço na presença de falhas nos recursos computacionais. A redundância ativo/ativo (A/A) é empregada quando os componentes primários e secundários compartilham carga de trabalho do sistema. Quando qualquer um desses componentes falhar, o componente secundário será o responsável pela continuidade do serviço, atendendo as requisições dos usuários do sistema (BAUER; ADAMS; EUSTACE, 2011).

A Figura 40 mostra o modelo SPN com redundância ativo/ativo (A/A). A partir desse modelo, é possível estimar a disponibilidade orientada à capacidade considerando o serviço em execução em um conjunto com VMs hospedadas em um nó. Os parâmetros `NVM` e `NND` permitem tal representação, onde, `NVM>1` e `NND>1`. Assim, `NVM` representa a quantidade de máquinas virtuais e `NND` representa a quantidade de `Nós`. Os lugares `VM_ON`, `NODE_ON`, `VM_OFF` e `NODE_OFF` representam os estados operacional e de falha das VMs e dos nós. A transição `DE` é ativada quando não há *tokens* no lugar `NODE_ON`, ou seja, quando todos os nós falharam. Assim, as VMs podem falhar quando todos os nós falharem, ou quando as mesmas falharem. O lugar `VM_DW` representa o estado de falha das `VMs` quando todos os nós falharem. A transição imediata `RVM` representa o retorno das `VMs` para o estado operacional, uma vez que os nós foram reparados.

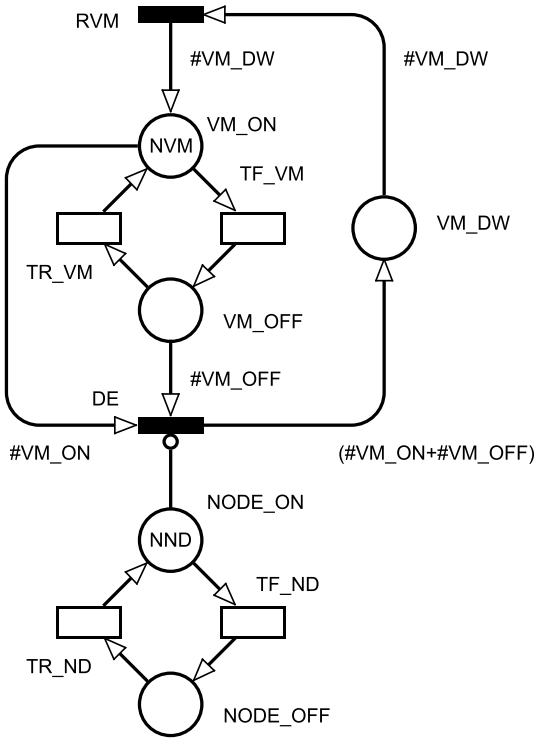


Figura 40 – Um exemplo ilustrativo de um modelo SPN para a redundância ativo/ativo. Baseado em: (MELO et al., 2017; ARAUJO et al., 2018).

A Tabela 15 descreve os atributos das transições do modelo SPN (Fig. 40).

Tabela 15 – Atributos das transições.

Transição	Tipo	Semântica	Peso	Prioridade
TR_VM	Temporizada	*Infinite Server*	-	-
TF_VM	Temporizada	*Infinite Server*	-	-
TR_ND	Temporizada	*Infinite Server*	-	-
TF_ND	Temporizada	*Infinite Server*	-	-
RVM	Imediata	-	1	1
DE	Imediata	-	1	1

A Tabela 16 apresenta a descrição das transições utilizadas no modelo SPN (Fig. 40). Os valores atribuídos as transições podem ser variadas conforme a necessidade e o intuito de representar políticas de serviços.

Tabela 16 – Parâmetros de dependabilidade para a SPN *Hotstandby* (A/A).

Parâmetro	Descrição
TF_VM	Tempo médio de defeito da VM
TR_VM	Tempo médio de reparo da VM
TF_ND	Tempo médio de defeito do nó
TR_ND	Tempo médio de reparo do nó
RVM	Ativação das VMs após reparo
DE	Desativação das VMs após defeito no nó

A Equação (5.8) estima a disponibilidade orientada à capacidade (COA) (MACIEL et al., 2012):

$$COA = \frac{\sum_{i=1}^{NMVN}(P\{(nVM\_ON) = i\}) \times i)}{NVM} \tag{5.8}$$

onde, (P{(nVM_ON)=i}) é uma função que define a probabilidade do estado i apresentar um número de máquinas virtuais operacionais, considerando ainda o número de máquinas virtuais suportado por cada nó na arquitetura, representado por (NVM).

5.2 MODELOS DE CUSTO

Usuários de ambientes de computação em nuvem estão cada vez mais atentos a mecanismos que proporcionem a garantia de serviços com alta disponibilidade (BAUER; ADAMS, 2012; WEI; LIN; KONG, 2011b). Nesse contexto, é necessário alinhar cuidadosamente as estratégias que garantam a disponibilidade e confiabilidade dos serviços, considerando a redução de custos, visto que, em alguns casos, é necessário levar em conta orçamentos limitados.

O modelo *pay-as-you-go* utilizado por serviços na nuvem (IaaS, PaaS e SaaS) possibilita que usuários aumentem suas demandas de trabalho a qualquer momento. Considerando que ambientes computacionais necessitam de controle e gerenciamento de custos na aquisição, manutenção e operação, a seguir é apresentado um conjunto de modelos que visa estimar os custos necessários para a utilização de ambientes de IaaS. É importante destacar que as equações propostas visam identificar e explicar os custos associados aos modelos de dependabilidade apresentados anteriormente.

Os modelos de custos foram adotados considerando os custos operacionais associados aos custos de manutenção dos equipamentos e aos gastos de consumíveis e outras despesas operacionais, necessárias ao funcionamento do negócio ou sistema (NICOLETTI, 2013). A Equação 5.9 proposta estima o custo total de uma infraestrutura como serviço na nuvem. O custo total estimado (TC_e) usa o conceito de custo total de propriedade (TCO), que é definida como a totalidade dos custos associados ao produto, serviço ou capital que são compreendidos ao longo de sua vida útil (WEIL; MAHER, 2005; MONCZKA et al., 2009). A Equação 5.9 apresenta o somatório dos custos de utilização (C_{ut}), custos de manutenção (C_{mt}) e custos de operação (C_{op}) da IaaS.

$$TC_e = C_{ut} + C_{mt} + C_{op} \tag{5.9}$$

Já a Equação 5.10 permite determinar os custos associados à utilização de um serviço alocado em uma dada IaaS em nuvem. Sendo o produto do somatório dos custos dos componentes da infraestrutura representado por $\sum L_c$ (moeda monetária, por exemplo, real (R$)), pela quantidade total de máquinas virtuais (VM), com o período de tempo alocado T (representado em horas) e a disponibilidade da infraestrutura como serviço A_v (valor decimal entre 0 e 1). Esses parâmetros são utilizados para representar a estimativa de custos de uma infraestrutura como serviço.

$$C_{ut} = \sum L_c \times VM \times T \times A_v \tag{5.10}$$

Por sua vez, a Equação 5.11 é usada para estimar os custos de manutenção (reparos) da infraestrutura como serviço, onde temos a indisponibilidade da IaaS (D_{wt}), representado por um valor (decimal entre 0 e 1) e o custo (moeda monetária, por exemplo, real (R$)) das horas trabalhadas em atividades de manutenção (Lb_{Dwt}). VM representa a quantidade total de máquinas virtuais atribuídas ao serviço na nuvem e T como o período de tempo (em horas) alocado para a utilização do serviço. Por fim, tem-se o somatório dos custos associados as atividades de substituições de componentes na IaaS ($\sum C_{sub}$).

$$C_{mt} = (D_{wt} \times Lb_{Dwt} \times VM \times T) + \sum C_{sub} \tag{5.11}$$

No caso da Equação 5.12, permitem-se calcular os custos operacionais relacionados à infraestrutura como serviço. Tem-se o consumo de energia (Ec) em Quilowatt-hora

(kWh), o preço da energia (Ep) em uma moeda monetária (por exemplo, R$) por (kWh), e a quantidade total de nós (N) alocados para o serviço na nuvem. Assim como, T e A_v representam o período (em horas) que o serviço é alocado para utilização e a disponibilidade (valor decimal entre 0 e 1) da IaaS, respectivamente. Associamos também o custo (moeda monetária, por exemplo, real (R$)) da hora trabalhada em operação (Lb_{Up}) realizadas na IaaS. Por fim, A_v e VM representam os mesmos parâmetros das Equações 5.10 e 5.11.

$$Cop = (Ec \times Ep \times N \times T \times A_v) + (Lb_{Up} \times A_v \times VM \times T) \tag{5.12}$$

A partir dos modelos de dependabilidade, usuários das IaaS podem compor um conjunto de configurações apropriadas de acordo com as necessidades de uso. E aliados ao conjunto de modelos de custos, estimarem os valores de manutenção, utilização e operação das IaaS. Além disso, para que os usuários tenham uma garantia de qualidade dos serviços, os gestores podem ainda associar penalidades caso a qualidade do serviço não seja satisfeita quando estiverem analisando IaaS públicas. A Equação 5.13 calcula os custos da penalidade em valor monetário de acordo com uma IaaS quando algum parâmetro de dependabilidade for violado. D_{wt} é o período de indisponibilidade do serviço (valor entre 0 e 1), T é o período (em horas) atribuído ao serviço contratado e o valor (moeda monetária, por exemplo, real (R$)) por hora paga pelo provedor do serviço na nuvem para o usuário (V_r). O valor monetário relacionado à violação de qualidade é dado a seguir (V_{vio}):

$$V_{vio} = D_{wt} \times T \times V_r \tag{5.13}$$

5.3 MODELOS DE DECISÃO

Um desafio enfrentado por usuários na escolha de uma infraestrutura como serviço em nuvem privada ou pública, é deparar-se com uma diversidade de opções e ter que escolher apenas uma. Assim, realizar a tomada de decisão sem o planejamento e a adoção de métodos multicritérios pode resultar em escolhas indesejáveis, como por exemplo, optar por uma infraestrutura que apresente falhas periódicas, resultando em perda de clientes pela indisponibilidade do sistema.

Usuários que não possuem conhecimentos relacionados à dependabilidade do serviço, podem também apresentar dificuldades no momento da tomada de decisão devido aos diversos critérios envolvidos (capacidade orientada a disponibilidade e confiabilidade, por exemplo). Nesse sentido, modelos de decisão podem ser utilizados pelos usuários de infraestruras em nuvem para auxiliar na escolha de uma alternativa. Diversos métodos têm sido utilizados para avaliar as relações e características de um conjunto de critérios, tais como: agrupamento de dados, *Analytic Hierarchy Process* (AHP), *Technique for Order of Preference by Similarity to Ideal Solution* (TOPSIS), redes neurais, algoritmos genéticos

e redes bayesianas (JAIN; DUBES, 1988; RUSSEL; NORVIG, 2004; ZOPOUNIDIS; PARDALOS, 2010). Esses métodos são utilizados com o objetivo auxiliar tomadores de decisão em situações de incerteza, contribuindo no processo de decisão (HAIR et al., 2009).

Agrupamento de dados é uma técnica que tem como proposta agrupar objetos similares ou dissimilares entre si utilizada em diversas áreas: aprendizagem de máquina, mineração de dados, reconhecimento de padrões, análise de imagens e bioinformática (CASTRO; FERRARI, 2016). Em geral, essa técnica utiliza funções que calculam a proximidade (similaridade) ou distância (disimilaridade) entre os objetos para o agrupamento de dados.

O modelo de decisão proposto consiste em escolher um conjunto de alternativas que estejam tão próximas quanto possível da solução satisfatória. Podemos definir como solução satisfatória, ou ideal, aquela que alcança os melhores valores durante a avaliação da função multicritério, por exemplo, a solução que encontre a maior disponibilidade e menor custo. Para isso, é necessário adotar um planejamento de avaliação que considere problemas multicritérios e de ordenação dado um conjunto de cenários pré-determinado.

Desta forma, este trabalho adotou o conceito do método de agrupamento de dados, pois o propósito foi ordenar e selecionar os resultados previamente avaliados considerando funções multicritérios. Para a realização do agrupamento, utilizaram-se as medidas de similaridade, quais sejam, distância Euclidiana, Minkowski e Manhattan. Além dessas, foi adotado o conceito de dominância de Pareto e uma função com intervalos de mínimo-máximo. A seguir, a Figura 41 apresenta um exemplo de uma matriz contendo pares multicritérios utilizados para a ordenação das IaaS.

	$Critério_1$	$Critério_2$
Cen_1	Cen_{11}	Cen_{12}
Cen_2	Cen_{21}	Cen_{22}
⋮	⋮	⋮
Cen_n	Cen_{n1}	Cen_{12}

Figura 41 – Exemplo de Matriz de Resultados.

Para iniciar a ordenação dos cenários, faz-se necessário a definição da função multicritério. O modelo de decisão permite o uso de n cenários e até 10 (dez) critérios que podem ser definidos para minimizar ou maximizar um critério. Os critérios consideram as métricas dos modelos de dependabilidade, performabilidade e custo das IaaS. A seguir, é apresentado um exemplo de uma função multicritério que tem o intuito de ordenar um conjunto de IaaS (soluções) com o menor custo e indisponibilidade por exemplo. O conjunto de alternativas (5.14), função multicritéio (5.15) e objetivos (5.16) são apresentados como:

$$IaaS_n,\ n = \{1,\ 2,\ ...\ n\},\ Conjunto\ de\ IaaS; \tag{5.14}$$

$$minimizar\ f_m(x),\ m = \{1, 2\}; \tag{5.15}$$

$$\begin{aligned} f_1 &= minimizar\ o\ custo\ total\ da\ IaaS. \\ f_2 &= minimizar\ a\ indisponibilidade\ da\ IaaS. \end{aligned} \tag{5.16}$$

$$IaaS_n \neq 0,\quad Conjunto\ de\ IaaS. \tag{5.17}$$

Como restrição (5.17), os valores devem estar entre 0 e 1, dado que foram normalizados. Esta abordagem permite que os parâmetros possam ser alterados conforme os objetivos, admitindo a geração de diversos conjuntos de soluções a cada modificação. A seguir, são apresentados os algoritmos adotados para ordenar as soluções considerando funções multicritério. O detalhamento dos algoritmos permite que tomadores de decisão possam compreender a lógica adotada, e ainda, caso queiram, implementar em alguma ferramenta computacional. É importante ressaltar que o modelo de decisão proposto foi implementado e pode ser acessado no repositório GitHub (ARAUJO, 2019), com maiores detalhes no Apêndice B.

O Algoritmo 1 apresenta o método para ranquear os cenários avaliados através das medidas de similaridade. As `Linhas 1-5` do algoritmo descrevem as variáveis de entrada do algoritmo: n representa o número de cenários a serem considerados, q_1, ... q_n são os pesos utilizados para os cálculos das medidas de similaridade (Euclidiana, Manhattan e Minkowski), p é o parâmetro da distância Minkowski e $matriz\_E$ a matriz de entrada. A matriz de entrada é do tipo $n \times 10$, onde cada coluna representa um critério avaliado nos modelos de dependabilidade, performabilidade ou custo, e cada linha representa uma alternativa (solução) As `Linhas 6-7` apresentam a variável de saída do algoritmo, que é representada por uma $matriz\_S$ composta pelo conjunto de soluções ordenadas, onde $w_n z_n$ denotam o valor da distância de um cenário n em relação a solução ideal, e a colocação do cenário no *ranking* respectivamente.

Nas `Linhas 8-15` são apresentadas as funções auxiliares como `CalculaDistancia`, `Normalize`, `Ordene` e `EncontraPivot`. A função `CalculaDistancia` calcula a distância entre a cada alternativa n e a alternativa ideal por meio das medidas de similaridade. Cada alternativa representa um cenário (IaaS) e cada cenário é composto por até dez critérios ou métricas (por exemplo, indisponibilidade e custo). Já a função `Normalize` normaliza os valores dos critérios que compõem cada cenário. A função `EncontraPivot` tem o objetivo de encontrar a alternativa ideal baseada na maximização ou minimização do critério. A função `Ordene` ordena de modo crescente os valores atribuídos a cada distância que representa uma alternativa utilizando como referêncial a solução ideal. Na `Linha 17` do Algoritmo 1, a matriz de entrada é normalizada e em seguida é realizada a busca pelo *pivot*. Já na `Linha 19`, é iniciado um laço para calcular as distâncias entre os

Algoritmo 1: Ordena cenários através de medidas de similaridade.

1 : **Variáveis de Entrada:**
2 : n //número de cenários
3 : $q_1, ...q_n$ //pesos para ponderação
4 : p //p de Minkowski
5 : $matriz\_E\{x_1y_1, ... , x_ny_n\}$
6 : **Variável de Saída:**
7 : $matriz\_S = \{w_1z_1, ... w_nz_n\}$
8 : **Funções Auxiliares:**
9 : $CalculaDistancia(x_1y_1, ... , x_ny_n, q_1, ...q_n, p)$ //Calcula a distância
10 : //Euclidiana, Manhattan ou Minkowski entre dois pontos
11 : $Normalize(matriz\_E)$ //função que normaliza a matriz de entrada
12 : $Ordene(matriz\_S)$ //função ordena a coluna w da
13 : //matriz de saída em ordem crescente
14 : $EncontraPivot(matriz\_E)$ //função multicritério
15 :
16 : **Início:**
17 : $matriz\_E \leftarrow Normalize(matriz\_E)$
18 : $pivot \leftarrow EncontraPivot(matriz\_E)$
19 : **Para** i **De** 1 **Até** n **Faça:**
20 : $matriz\_S[i][1] \leftarrow CalculaDistancia(matriz\_E[x_i][y_i], pivot, q_1, q_2, p)$
21 : $matriz\_S[i][2] \leftarrow i$ // i - contador do ranking
22 : **Fim**
23 : $matriz\_S \leftarrow Ordene(matriz\_S)$
24 : **Fim**

cenários considerando o cenário "pivot"ou cenário ideal. Por fim, a matriz de saída recebe os cenários ordenados.

A seguir, o Algoritmo 2 apresenta o método utilizado para selecionar um conjunto de soluções considerando o conceito de dominância de Pareto. Na `Linha 2`, `c` indica a quantidade de soluções candidatas para compor o conjunto de soluções não dominadas [4]. A `Linha 3` do Algoritmo 2 recebe como entrada uma matriz do tipo $n \times 2$, ou seja, onde `n` representa a quantidade de cenários, e o valor dois, corresponde a quantidade de critérios. A `Linha 5` apresenta a matriz de saída (`matriz_P`) que recebe o conjunto de soluções não-dominados. Já `m` representa a quantidade de cenários na `matriz_P` de saída. E a `Linha 7` indica a quantidade de soluções candidatas para compor o conjunto ótimo pareto. Na `Linha 10` é apresentada uma estrutura de repetição entre a variável de entrada c e a variável auxiliar d. Essa repetição é utilizada para garantir que o laço de repetição seguinte realize as comparações entre os cenários não-dominados em relação aos cenários dominados. Nas `Linhas 10 e 11` temos duas estruturas de repetição para garantir a comparação entre os cenários e seus respectivos critérios. Para isso, utilizaram-se as duas condições de Pareto, que diz que uma solução (1) não é pior que a outra solução (2) para

[4] A solução x_1 não é pior que x_2 para todos os objetivos , ou seja, para todo j = 1, 2, ... , M;

os dois critérios, e que uma solução (1) é estritamente melhor que a solução (2) pelo menos em um critério.

Algoritmo 2: Seleciona cenários considerando dominância de Pareto.

```
1  : Variáveis de Entrada:
2  : c   //número de soluções desejadas (não dominadas)
3  : matriz_E = [n][2]
4  : Variável de Saída:
5  : matriz_P = [m][2]
6  : Variável Auxiliar:
7  : d   //número de soluções dominadas
8  :
9  : Início:
10 :    Enquanto d < c Faça:
11 :       Para i De 1 Até n Faça:
12 :          Para j De i + 1 Até n Faça:
13 :             Se((matriz_E[i][1] <= matriz_E[i+j][1]) &&
14 :               (matriz_E[i][2] <= matriz_E[i+j][2]))
15 :                Se((matriz_E[i][1] < matriz_E[i+j][1]) ||
16 :                 (matriz_E[i][2] < matriz_E[i+j][2]))
17 :                   matriz_P[m][1] ← matriz_E[i][1]
18 :                   matriz_P[m][2] ← matriz_E[i][2]
19 :                   d ← d + 1
20 :                   m ← m + 1
21 :                Fim
22 :             Fim
23 :          Fim
24 :       Fim
25 :    Fim
26 : Fim
```

O Algoritmo 2 apresenta um exemplo de função multicritério com o intuito de localizar um conjunto de soluções minimizando dois critérios, indisponibilidade e custo das IaaS. Os operadores lógicos utilizados para compor a função foram menor igual que (<=) e menor (<) no primeiro e segundo multicritérios, respectivamente. Na mudança da função multicritério, faz-se necessária a alteração dos operadores lógicos. A Tabela 17 apresenta os operadores lógicos que podem ser utilizadas para compor a função multicritério.

Tabela 17 – Operadores lógicos para as funções multicritérios.

Função	Operadores Lógicos	
	Objetivo 1	Objetivo 2
Minimizar-Minimizar	$<=, <$	$<=, <$
Maximizar-Minimizar	$>=, >$	$<=, <$
Minimizar-Maximizar	$<=, <$	$>=, <$
Maximizar-Maximizar	$>=, >$	$>=, >$

O Algoritmo 3 apresenta o método mínimo-máximo para realizar o *ranking* dos cenários de IaaS. Esse método considera que cada função multicritério precisa ser definida por meio de um intervalo, com um valor mínimo e máximo. Com a definição do intervalo, é possível encontrar e ordenar um conjunto de soluções. Na `Linha 2`, o algoritmo recebe como entrada uma matriz do tipo $n \times 2$ ($Matriz\_E$), onde `n` representa a quantidade de cenários e o valor 2 representa dois critérios, por exemplo, indisponibilidade e custo.

A `Linha 3` descreve quatro variáveis para definir os intervalos mínimos e máximos de cada objetivo. A `linha 4:` apresenta uma variável para indicar o tipo de ordenação que pode ser se baseando no primeiro objetivo, segundo objetivo ou ambos. A `Linha 6` apresenta a variável de saída do algoritmo, que é representada por uma matriz $matriz\_S$ composta por um conjunto de soluções ordenadas. `m` representa a quantidade de cenários ordenados na `matriz_S` de saída. Na `Linha 8` são apresentadas duas variáveis auxiliares utilizadas para ordenação dos cenários.

Na `Linha 11`, é descrita uma estrutura de repetição e sua condição de parada corresponde ao número de soluções da $Matriz\_E$. Os três casos de ordenação podem ser definidos através de uma escolha na `Linha 14`. O caso (1) realiza a busca por soluções considerando um intervalo mínimo e máximo para o objetivo (1). O caso (2) na `Linha 20` faz uma busca considerando um intervalo mínimo e máximo para o segundo objetivo. Por fim, na `Linha 25` o caso (3) que realiza uma busca por soluções que estejam dentro dos limites mínimos e máximos dos dois objetivos. Para cada caso, a matriz $Matriz\_S$ recebe um conjunto de soluções.

Algoritmo 3: Seleciona cenários considerando valores mín-máx dos objetivos.

```
1  : Variáveis de Entrada:
2  : matriz_E = [n][2]
3  : Min_Obj_1, Max_Obj_1, Min_Obj_2, Max_Obj_2
4  : s   //definição do objetivo: Obj_1 ou Obj_2 ou ambos
5  : Variável de Saída:
6  : matriz_S = [m][2]
7  : Variáveis Auxiliares:
8  : aux1, aux2
9  :
10 : Início:
11 :    Para i De 1 Até n Faça:
12 :       aux1 ← matriz_E[i][1]
13 :       aux2 ← matriz_E[i][2]
14 :       Escolha (s)
15 :          Caso (1)
16 :             Se (aux1 ≥ Min_Obj_1) && (aux1 ≤ Max_Obj_1) Então:
17 :                matriz_S[m][1] ← aux1
18 :                matriz_S[m][2] ← aux2
19 :                m ← m + 1
20 :          Caso (2)
21 :             Se (aux2 ≥ Min_Obj_2) && (aux2 ≤ Max_Obj_2) Então:
22 :                matriz_S[m][1] ← aux1
23 :                matriz_S[m][2] ← aux2
24 :                m ← m + 1
25 :          Caso (3)
26 :             Se ((aux1 ≥ Min_Obj_1) && (aux1 ≤ Max_Obj_1)) &&
27 :                ((aux2 ≥ Min_Obj_2) && (aux2 ≤ Max_Obj_2)) Então:
28 :                matriz_S[m][1] ← aux1
29 :                matriz_S[m][2] ← aux2
30 :                m ← m + 1
31 :       Fim
32 :    Fim
33 : Fim
```

5.4 CONSIDERAÇÕES FINAIS

Este capítulo apresentou os modelos de dependabilidade, performabilidade, modelos de custos e o modelo de decisão. Os modelos representam a abordagem para avaliação e obtenção dos resultados dos parâmetros de dependabilidade e performabilidade das arquiteturas como serviço em nuvem. Os modelos de custos, como o próprio nome já mostra, estimam os custos operacionais de IaaS. Além disso, um conjunto de três algoritmos foram apresentados para auxiliar no processo de tomada de decisão. Esse conjunto de modelos podem auxiliar usuários, analistas e técnicos de IaaS no processo de tomada de decisão para escolher um serviço na nuvem.

6 ESTUDOS DE CASO

Este capítulo apresenta estudos de caso que demonstram a aplicabilidade dos métodos propostos e adotados para auxiliar no planejamento e avaliação de infraestruturas como serviço em nuvem para tomada de decisão. Assim sendo, a seguir são apresentados casos a partir da metodologia proposta: (6.1) Concepção e Avaliação da Infraestrutura Inicial, (6.2) Planejamento de Avaliação e Avaliação dos Modelos de Dependabilidade, (6.3) Avaliação do Modelos de Custos, (6.4) Avaliação do Modelo de Decisão, (6.5) Planejamento e Avaliação de IaaS com Rendundância A/A, e (6.6) Avaliação de Performabilidade. Os resultados alcançados nesses estudos de caso fornecem evidências sobre a utilidade dessa abordagem.

6.1 CONCEPÇÃO E AVALIAÇÃO DA INFRAESTRUTURA INICIAL

Para ilustrar a aplicação da metodologia proposta no Capítulo 4, a seguir, abordaremos o conjunto de passos que consideram a concepção do ambiente. A elaboração desta fase, compreendeu a identificação de uma solução para representar uma infraestrutura como serviço na nuvem, a caracterização de uma infraestrutura inicial, a escolha dos critérios (confiabilidade por exemplo) a serem avaliados através dos modelos SPN e RBD, a representação dos componentes (quantidade de VMs por exemplo) da infraestrutura através dos modelos, a obtenção dos valores relacionados aos parâmetros dos modelos (MTTF por exemplo), e a avaliação dos cenários iniciais.

6.1.1 Concepção do Ambiente

O OpenNebula é um projeto *OpenSource* que permite a implementação de soluções em infraestruturas escaláveis e adaptáveis para a criação e gestão de *data centers* virtualizados e nuvens corporativas (TORALDO, 2012). A Figura 42 ilustra um diagrama básico que representa a infraestrutura do OpenNebula. O componente FrontEnd representa o gerente do serviço, executa os processos necessários para a comunicação com os nós e monitora o ambiente. O principal processo de configuração do OpenNebula é o ONED. No FrontEnd, é instalado o pacote do OpenNebula. Além disso, todas as operações administrativas da nuvem, como definição e instanciação das máquinas virtuais (VMs), cadastro de novos nós e imagens[1], devem ser executadas utilizando o FrontEnd.

[1] Podem ser sistemas operacionais ou dados para serem usados em Máquinas Virtuais (OPENNEBULA, 2018).

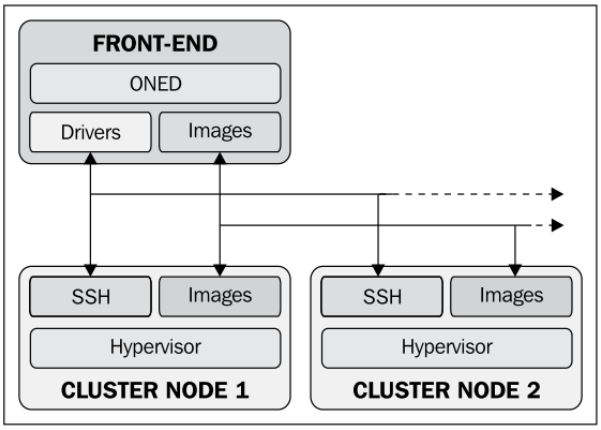


Figura 42 – Componentes da aquitetura OpenNebula.
Fonte: (TORALDO, 2012).

Os *Clusters Nodes* fornecem os recursos computacionais para a execução das máquinas virtuais. As imagens das VMss podem ser alocadas em dispositivos de armazenamento de dados (por exemplo, HDs e SANs)[2]. Contudo, esses dispositivos devem estar disponíveis através do protocolo SSH[3]. Isso permite que as imagens das VMs possam ser transferidas entre os nós do ambiente, possibilitando a criação ou recuperação de VMs que possam ter falhado. O *hypervisor* ou monitor de máquinas virtuais é utilizado para possibilitar que diferentes sistemas operacionais funcionem em um mesmo *hardware*. O ambiente OpenNebula necessita que seus componentes sejam interligados através de uma rede de comunicação de dados. Dentre os projetos disponíveis para infraestrutura como serviço em nuvem (OPENSTACK, 2018; CLOUDSTACK, 2018), optou-se pelo OpenNebula por ser esse projeto *opensource*, e também por apresentar uma documentação clara para implantação, além de possibilitar que o FrontEnd possa gereciar todas as atividades administrativas de forma centralizada.

A Figura 43 ilustra a infraestrutura utilizada para os estudos de caso levando em consideração o projeto OpenNebula. A infraestrutura possui três componentes principais: o nó principal, o nó reserva e o servidor de gerenciamento. O nó principal é composto por uma máquina virtual hospedada em um *hardware*. Já a máquina virtual é representada por um sistema operacional e por serviços de aplicação. O *hardware* que hospeda o nó principal é representado pelos seguintes componentes: sistema operacional, serviço de gerenciamento e máquina virtual. O nó reserva, como segundo componente da infraestrutura, é utilizado para garantir altos níveis de disponibilidade da IaaS e assume o papel do nó principal quando ocorre uma interrupção do serviço. O servidor de gerenciamento que representa o Frontend é responsável pela supervisão e controle de todo o ambiente da nuvem. O volume de armazenamento remoto pode ser acessado pela VM e a sua gestão é realizada também por meio do servidor de gerenciamento. Todos os componentes estão interligados

[2] *Hard Disk* (HD) e *Storage Area Network* (SAN).
[3] *Secure Shell* (SSH).

por uma rede privada.

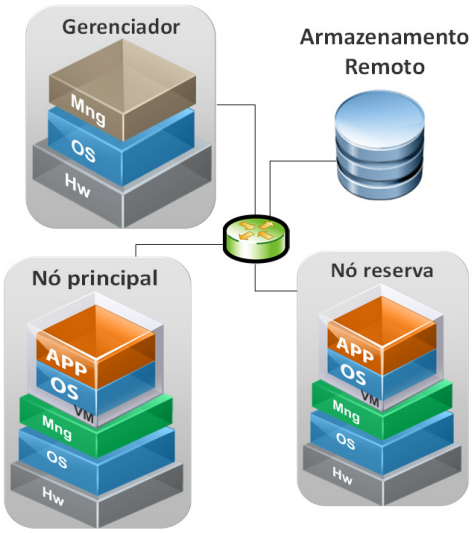


Figura 43 – Ilustração gráfica da infraestrutura como serviço adotada.

O modo operacional dessa IaaS é representado pelo funcionamento do nó principal e servidor de gerenciamento. No entanto, caso os nós principal e reserva apresentem defeito, a disponibilidade da nuvem será reduzida drasticamente até que um dos dois seja reativado. Vale ressaltar que na ocorrência de um defeito, o nó reserva assume, mas quando o nó principal é restaurado, o nó reserva é colocado novamente em estado de espera e o nó principal assume o serviço. O objetivo do nó reserva neste caso é maximizar a disponibilidade da infraestrutura como serviço em nuvem.

Uma biblioteca digital (compartilhamento de arquivos acadêmicos) foi implantado na infraestrutura como serviço em nuvem visando representar uma aplicação alocada ao ambiente. A aplicação escolhida foi o Dspace (DSPACE, 2018). O Dspace é um projeto *opensource* de uma biblioteca digital voltado para organizações sem fins lucrativos que desejam criar um repositório digital. O sistema de gerenciamento de arquivos permite às instituições de ensino, coletar, preservar e divulgar os esforços acadêmicos e intelectuais da academia. O serviço da biblioteca é composto por: (i) um mecanismo que identifica os usuários; (ii) um fluxo de processo para submissão de itens; (iii) funcionalidade para importações e exportações de coleções; (iv) relatório estatístico/resumo; e (v) mecanismo de pesquisa. Os usuários podem acessar as informações bibliográficas, tais como artigos, documentos, teses, livros e dissertações. A *interface* conecta o usuário com o sistema de gestão da biblioteca digital (DSPACE, 2018). A Figura 44 apresenta uma ilustração do serviço de biblioteca digital hospedado em uma nuvem privada. O serviço oferece a submissão de conteúdo para o banco de dados através de uma conta em modo administrador, de forma que todos os usuários possam ter acesso.

Figura 44 – Visão geral da biblioteca digital.

6.1.2 Avaliação da Infraestrutura Inicial

Esta subseção apresenta um estudo que tem como intuito a representação da infraestrutura como serviço em nuvem privada apresentada na subseção anterior. Para a representação da IaaS foram utilizados os modelos de dependabilidade propostos no Capítulo 5. Os componentes utilizados para representar o funcionamento da infraestrutura foram representados através de diagramas de blocos de confiabilidade (RBD). Essa primeira representação possibilitou que as estratégias e mecanismos de redundância pudessem ser utilizados. Para a avaliação dos modelos foi necessária a obtenção de dados históricos de defeitos e reparos dos componentes *hardware*, sistema operacional, gerenciador da nuvem e máquina virtual. Este trabalho adotou os tempos médios de defeito e reparo conforme (KIM; MACHIDA; TRIVEDI, 2009; DANTAS et al., 2015; MATOS et al., 2017).

Para ilustrar os tempos de defeito e reparo do serviço (biblioteca digital) hospedado na IaaS, este estudo realizou experimentos de aceleração de falhas (FOGLIATTO; RIBEIRO, 2010). Logo, foi proposto um conjunto de experimentos visando à obtenção dos dados de defeito do serviço de biblioteca digital na nuvem através de testes acelerados. Esses testes são utilizados quando um dado componente possui alta confiabilidade e demanda de longos períodos para ocorrência de um defeito em testes em condições normais (FOGLIATTO; RIBEIRO, 2010). Por meio dos testes acelerados, é possível encurtar o tempo até o defeito do componente. Para a obtenção do tempo de defeito da aplicação na IaaS, foi utilizado o gerador de carga *Jmeter* (HALILI, 2008; JMETER, 2018). Esse gerador permitiu estressar o ambiente, assim como, registrar por meio de um *log* as ocorrências dos defeitos na biblioteca digital. O tipo de estresse utilizado foi o envio de requisições HTTP [4] de maneira constante. Esse tipo de teste proporciona o envio uniforme de transações ao repositório. Outros tipos de envio considerando distribuições estatísticas (por exemplo, Erlang ou hiperexponencial) podem ser utilizados.

Em testes de aceleração de falha, é necessário também estimar o fator de aceleração (A_F). A_F representa a relação entre a operação em condições normais do componente e

[4] *HyperText Transfer Protocol* (HTTP)

de estresse, conforme a Equação 6.1 (FOGLIATTO; RIBEIRO, 2010).

$$A_F = t_o/t_s \tag{6.1}$$

t_o corresponde a variável de teste em condições normais de operação e t_s indica a variável em condições aceleradas de operação. Nesse sentindo, para fins de representação do funcionamento da biblioteca digital, como índice de operação normal determinamos o valor de 2 requisições por minuto, ou seja, $0,03$ requisições/seg. Já em condições aceleradas, definimos o envio de 240 requisições a cada 60 segundos, isto é, 4 requisições/seg. Os valores correspondem a operação de 240 usuários enviando requisições de busca na base de dados da biblioteca digital.

Assim sendo, configuramos o gerador de carga para enviar 240 requisições por minuto. Caso o tempo de resposta de uma requisição à biblioteca digital excedesse 6 segundos, o teste registraria o evento como um defeito. As requisições utilizadas nos testes representaram uma situação, onde um dado usuário realizava uma busca no repositório por meio de uma palavra chave. O intervalo de cada coleta dos testes acelerados foi de seis dias consecutivos, considerando as mesmas condições e configurações.

A quantidade de dias de medição foi definido baseado em um pré-teste inicial. Após o primeiro pré-teste para adequação do ambiente, o *log* de registro apontou que os eventos de falhas aconteciam após o quarto dia de experimento. Sendo assim, estimamos um intervalo de seis dias. A escolha de intervalo visou garantir a propriedade de repetição e casualização no experimento, repetimos por três vezes o processo de medição. Além disso, calculamos o intervalo de confiança (IC) para indicar a confiabilidade do valor médio do tempo de resposta. A Tabela 18 apresenta o resumo das estatísticas obtidas para o tempo de resposta das requisições enviadas ao repositório da biblioteca digital.

Tabela 18 – Parâmetros do tempo de resposta.

Parâmetro	Valor
Tempo médio > 6 seg.	14,41287081 seg
Desvio padrão	0,35478706
Intevalo de confiança (95%)	(14,40803;14,41771)
Quantidade de amostras	209

A partir dos registros das requisições que ultrapassaram o valor estipulado, foram aplicadas às amostras o teste de aderência para verificar a qual distribuição os dados mais se adequavam. Esse teste possibilitou identificar o comportamento de uma distribuição *Weibull*. Os parâmetros *forma* e *escala* obtidos foram, respectivamente, $464,1$ e $865,82$. Diante disso, empregou-se a equação de aceleração de falha da distribuição *Weibull* (FOGLIATTO; RIBEIRO, 2010) para obtenção do tempo médio de defeito (MTTF) da aplicação (biblioteca digital). A Tabela 19 apresenta os tempos de defeito da biblioteca

digital (ARAUJO et al., 2014), assim como, dos demais componentes que correspondem à infraestrutura do serviço na nuvem. Os demais valores foram obtidos conforme (KIM; MACHIDA; TRIVEDI, 2009; DANTAS et al., 2015; MATOS et al., 2017).

Tabela 19 – Parâmetros de dependabilidade.

Componente	MTTF(Hr)	MTTR(Hr)
Hardware	8760	1,67
Sistema Operacional	1440	1
Gerenciador	788,4	1
Máquina Virtual	2880	0,17
Biblioteca Digital	6865,3	0,17

Nesse sentido, é apresentado um estudo de caso baseado em parâmetros de dependabilidade. O modelo de dependabilidade pode ser útil para os usuários de nuvem durante o projeto e análise de infraestruturas. Na fase de projeto, os modelos podem ser utilizados para definir um conjunto de configurações de modo a avaliar níveis de disponibilidade específicos. A quantificação da disponibilidade, confiabilidade e indisponibilidade das infraestruturas como serviço em nuvem podem ser utilizadas como um recurso para negociações entre usuários e provedores.

A Figura 45 ilustra o modelo de dependabilidade utilizando diagrama de bloco de confiabilidade (RBD) para representar o componente Frontend da infraestrutura como serviço em nuvem apresentada na Figura 43. O Frontend é composto por três componentes de série: *hardware*, sistema operacional e o servidor de gerenciamento. O *hardware* representa os componentes físicos de um servidor, como por exemplo, processador, memória e disco rígido. O servidor de gerenciamento é responsável pelas operações de gerência e monitoramento da nuvem. Os tempos médios de defeito e reparo utilizados foram apresentados na Tabela 19.

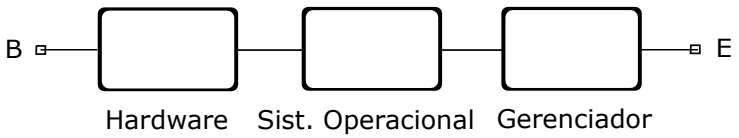


Figura 45 – Modelo RBD do Frontend.

A Figura 46 apresenta o modelo de dependabilidade para o nó principal através de um RBD. O nó é composto por cinco componentes em série: *hardware*, sistema operacional, serviço de gerenciamento, máquina virtual e o serviço da biblioteca digital. Uma vez que o Frontend e o nó principal foram modelados, o próximo passo é a utilização da estratégia de modelagem hierárquica para a composição dos modelos. A composição do modelo auxilia na modelagem dos mecanismos de redundância e facilita o processo de avaliação, visto que foi reduzida a quantidade de blocos.

Figura 46 – Modelo RBD do Nó.

Para realizar a composição, avaliou-se cada modelo (Frontend e Nó) para a obtenção dos tempos médios de defeito e reparo. O processo de modelagem e avaliação dos modelos RBDs foi realizado através da ferramenta Mercury (SILVA et al., 2013a; MACIEL et al., 2017; MoDCS, 2018). O tempo médio de defeito e reparo obtidos do frontend e nó são apresentados na Tabela 20. Após obter os parâmetros de dependabilidade, foi realizada a composição dos modelos visando representar uma IaaS. O processo de obtenção desse modelo foi fundamental, pois a partir de sua avaliação, foi possível adotar mecanismos de tratamento de interrupção visando elevar a disponibilidade e confiabilidade, reduzindo consequentemente a indisponibildade. A Figura 47 ilustra o modelo hierárquico com essa composição do Frontend e nó.

Tabela 20 – Parâmetros de defeito e reparo para Frontend e Nó.

Componente	MTTF (Hr)	MTTR (Hr)
Frontend	481,46564885	1,03660458
Nó	389,12422030	0,91545217

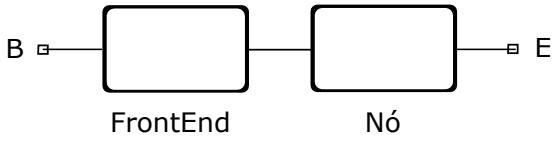


Figura 47 – Modelo RBD hierárquico do Frontend e nó.

Depois desse procedimento, o índice de importância de confiabilidade (RI) foi adotado para identificar qual componente do sistema necessitava de uma maior atenção, visando aumentar a disponibilidade. Considerando-se a combinação do modelo com dois blocos em série, o índice de confiabilidade foi calculado para o componente *Frontend* e nó, respectivamente obtivemos os valores $(0,153201$ e $0,219544)$ para cada componente. Assim, constatou-se que o nó obteve maior valor no cálculo do índice, ou seja, apresentou maior importância de confiabilidade (RI). Como medida para elevar a disponibilidade da IaaS, adotaram-se os mecanismos de tratamento de interrupção através de redundância do nó. Utilizaram-se as estratégias de redundância *HotStandby*, *ColdStandby* e *WarmStandby* para elevar o nível de disponibilidade.

A Figura 48 ilustra o mecanismo de tratamento de interrupção *HotStandby* adotado no componente nó da IaaS. Na existência de um defeito no nó 1 (um), o nó reserva ou redundante assume o papel do nó principal. Esse mecanismo propicia um aumento

na disponibilidade da infraestrutura como serviço. O modo operacional do modelo RBD é dado pela Equação 6.2. A principal particularidade do componente que utiliza a redundância *HotStandby* é a ausência de um tempo de ativação, se comparado com os mecanismos *ColdStandby* e *WarmStandby*.

$$\mathtt{RBD}_{Op} = \mathtt{FrontEnd} \wedge (\mathtt{NÓ\ 1} \vee \mathtt{NÓ\ 2}) \tag{6.2}$$

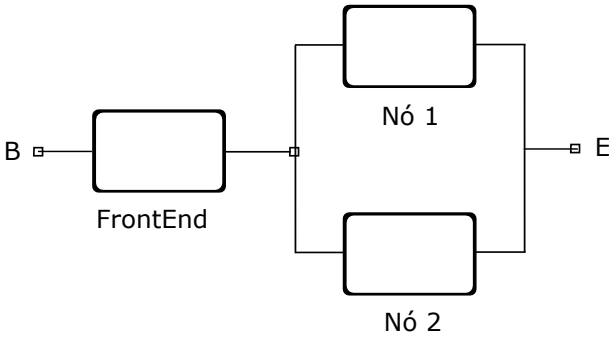


Figura 48 – Modelo RBD do Frontend e nó com redundância *HotStandby*.

Por sua vez, a Figura 49 ilustra a estratégia de modelagem hierárquica para a utilização do mecanismo de redundância *ColdStandby* no componte nó. O mecanismo de redundância *ColdStandby* possui a dependência de tempo para ativação do componente reserva, sendo complexa a sua representação em diagrama de bloco de confiabilidade (BALBO, 2001; MACIEL et al., 2012). A partir disso, foi utilizada a modelagem do mecanismo *ColdStandby* através das redes de Petri estocástica (SPN). O modo operacional para obtenção da métrica de disponibilidade no modelo SPN é dada a partir da Equação 6.3. Com a utilização do mecanismo de redundância *ColdStandby* no nó, foi possível elevar o nível de disponibilidade da IaaS (Ver Tabela 21). Esse modelo de redundância se diferencia do modelo *HotStandby* devido à existência de um tempo de ativação do componente reserva ou redundante. Os modelos Frontend e nó representados por redes de Petri Estocásticas foram modelados através da ferramenta Mercury (MoDCS, 2018; SILVA et al., 2013a).

$$\mathtt{SPN}_{Op} = (\mathtt{FRONTEND\_ON} > 0 \wedge (\mathtt{NODE1\_ON} > 0 \vee \mathtt{NODE2\_ON} > 0)) \tag{6.3}$$

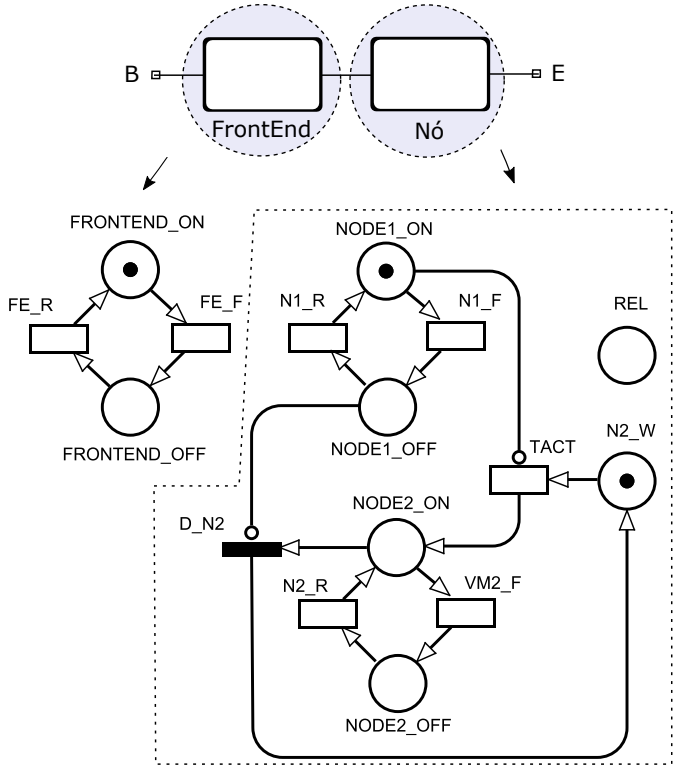


Figura 49 – Modelo RBD do Frontend e SPN do nó com redundância *ColdStandby*.

A Figura 50 apresenta a estratégia *WarmStandby* para redundância no componente nó. Essa redundância se diferencia da redundância *ColdStandby*, por conta do módulo principal e o módulo redundante possuem uma taxa de falhas λ quando estão em operação. Mas, o módulo redundante possui uma taxa de falha ϕ quando está desativado, considerando que $0 \leq \phi \leq \lambda$ (KUO; ZUO, 2003). O modo operacional para obtenção da métrica de disponibilidade no modelo SPN é dada a partir da seguinte expressão (6.4). Assim, a obtenção da disponibilidade da IaaS representada pelo Frontend e nó redundante foi possível por meio do produto das disponibilidades obtidas através das avaliações dos modelos RBD (Frontend) e SPN (Nó). Ademais, os tempos médios utilizados para representar a ativação dos mecanismos *standby* foram estimados em 20 (vinte minutos) e 10 (dez minutos), respectivamente, para o *ColdStandby* e *WarmStandby*. Os tempos de ativação utilizados são valores médios estimados em experimentações realizadas no laboratório do grupo MoDCS (MODCS, 2018).

$$\texttt{SPN}_{Op} \texttt{ = (FRONTEND\_ON > 0} \wedge \texttt{ (NODE1\_ON > 0} \vee \texttt{ OPNODE2\_ON > 0))} \tag{6.4}$$

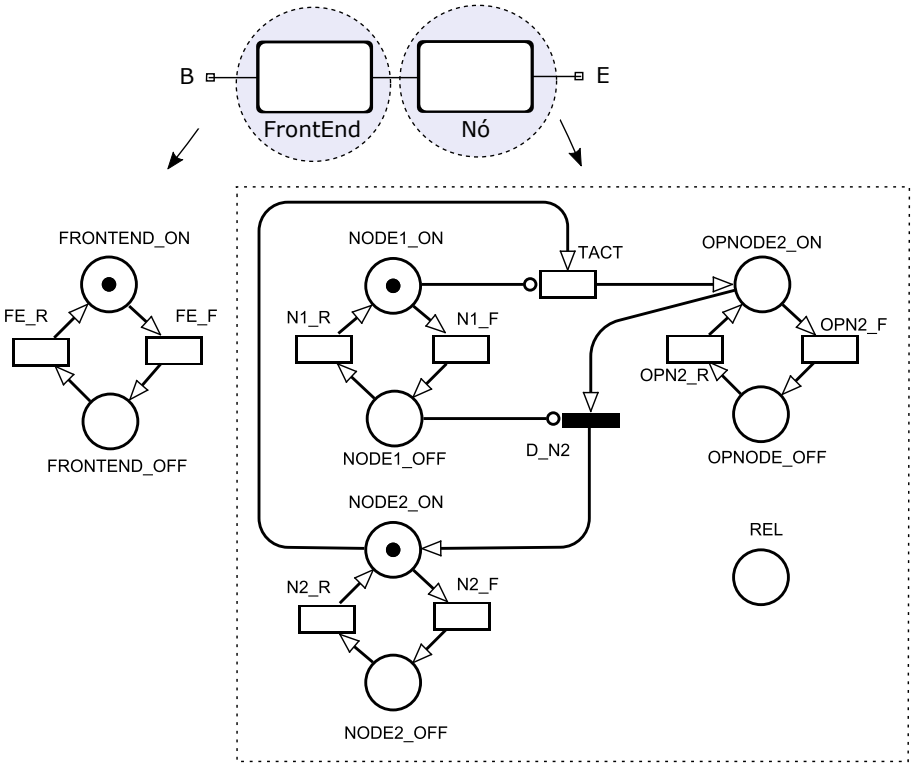


Figura 50 – Modelo RBD do Frontend e SPN do nó com redundância *WarmStandby*.

Os resultados obtidos a partir de cada IaaS representada pelos modelos de dependabilidade são apresentados na Tabela 21. O primeiro cenário representa a IaaS composta de um Frontend e um nó. O segundo cenário foi modelado com um Frontend e uma redundância *HotStandby* no nó. O terceiro representou o Frontend e o nó adotando o mecanismo de redundância *ColdStandby*. E o quarto resultado corresponde a uma IaaS com um Frontend e um nó com redundância *WarmStandby*. Com os dados de disponibilidade e confiabilidade obtidos, foi possível perceber a diferença entre cada abordagem. Comparando o primeiro cenário com os seguintes, é possível compreender a importância da utilização dos mecanismos de redundância, pois o nível de disponibilidade foi maior nas IaaS com redundância. A partir da métrica de indisponibilidade, se considerarmos que cada uma das IaaS opere por um período de um ano[5] iremos obter os seguintes valores de indisponibilidade em horas: (1^a) Conf. $39,34$ hrs, (2^a) Conf. $18,87$ hrs, (3^a) Conf. $24,51$ hrs e (4^a) Conf. $22,02$ hrs. A infraestrutura que adotou o mecanismo *HotStandby* possui o menor valor de indisponibilidade em relação aos demais, seguido do *Warm*, *Cold* e o primeiro cenário (sem redundância). A terceira coluna da Tabela 21 descreve os valores para a métrica de confiabilidade de cada IaaS. Para a métrica de confiabilidade, consideramos a análise com um período de 24 horas, ou seja, a confiabilidade de cada IaaS foi obtida sem a atuação do mecanismo de reparo.

[5] Valor para representação didática do tempo de indisponinilidade. Além disso, esse valor é bastante utilizado na literatura (SOUSA et al., 2015; MATOS et al., 2017; ANDRADE et al., 2017).

Tabela 21 – Resultados das Configurações das IaaS.

Infraestrutura como Serviço	Disponibilidade (%)	Disponibilidade Orientada à Capacidade (%)	Confiabilidade (%)
1. Frontend & Nó	99,5509574705	99,76543388	89,4469247967
2. Frontend & Nó (*Hot*)	99,7846109835	99,97929139	94,7970536818
3. Frontend & Nó (*Cold*)	99,7201853569	99,92583000	94,8881075626
4. Frontend & Nó (*Warm*)	99,7486208258	99,94998263	94,7885068481

A Figura 51 apresenta os resultados da disponibilidade considerando o número de noves de cada IaaS. Com o gráfico, podem-se visualizar que a segunda IaaS que utiliza o mecanismo *HotStandby* possui maior número de noves (2,66). Em seguida, vem a IaaS que utilizou o mecanismo *Warm*. Em terceiro, a IaaS com *Cold*, e por fim, a IaaS que não utilizou mecanismo de tratamento de interrupção no nó. A partir dos modelos de dependabilidade gerados, é possível planejar diversos outros cenários considerando tempos médios de defeito e reparo distintos.

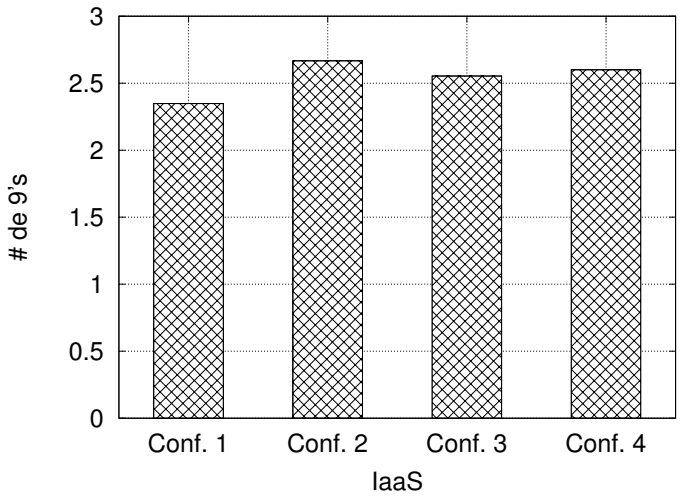


Figura 51 – Número de 9's considerando as IaaS (1-4).

O número de noves da disponibilidade pode ser calculada conforme a Equação 6.5. A disponibilidade máxima que o sistema pode atingir é representado pelo valor 100, e o parâmetro A representa a disponibilidade real do sistema.

$$\texttt{N = 2 - log (100 - A)} \tag{6.5}$$

6.2 PLANEJAMENTO DE AVALIAÇÃO E AVALIAÇÃO DOS MODELOS DE DEPENDABILIDADE

Após o passo inicial, esta subseção aplicou o planejamento de avaliação e a geração dos modelos de dependabilidade da metodologia proposta. A partir do plano de experimentos, foi possível obter configurações de infraestruturas como serviço em nuvem privada para,

em seguida, serem avaliadas. As métricas de dependabilidade consideradas na avaliação das IaaS foram: (i) disponibilidade, (ii) disponibilidade orientada à capacidade, (iii) confiabilidade e (iv) indisponibilidade. A geração das IaaS deu-se considerando os fatores e níveis do plano de experimento. Esta definição proporcionou o entendimento da relação entre as variáveis manipuladas (por exemplo, tipo de redundância) e as variáveis de resposta (como a disponibilidade) das infraestruturas como serviço.

A Tabela 22, apresenta os fatores e os níveis utilizados para a geração dos cenários (IaaS). O método utilizado para a geração do plano foi o fatorial completo, porque o propósito era obter todas as combinações das IaaS. Assim, os quatro fatores foram combinados com os seus respectivos níveis, representando um conjunto de cenários. O fator Nós e VMs representou a quantidade adotada para cada cenário. Para os mecanismos de tratamento de interrupção, consideraram-se redundâncias no nó e na máquina virtual (VMs). O serviço de reparo determinou os diferentes tempos associados para recuperar os componentes. E para auxiliar no entendimento, nomeamos os serviços como ouro, prata e bronze. O tipo de reparo ouro, por exemplo, possui o menor tempo médio de reparo após uma falha. O quarto fator utilizado identificou a adoção ou não de um tipo de redundância nos componentes da IaaS, representados, respectivamente, como N/R[6], *Hot*, *Cold* e *Warm*. Todas as IaaS foram modeladas considerando a abordagem hierárquica nos modelos RBD e SPN.

Tabela 22 – Fatores e Níveis.

Fatores	Níveis			
1. Número de Nós	1	2		
2. Número de VMs	1	2	3	
3. Serviço de Reparo (RS)	Ouro	Prata	Bronze	
4. Tipo de Redundância (RT)	N/R	*Hot*	*Cold*	*Warm*

As IaaS geradas a partir do plano de experimento estão descritas na Tabela 23. É importante ressaltar que, após a geração do plano, ainda foram verificadas as combinações das IaaS geradas, pois algumas iriam apresentar limitações quanto à representação nos modelos de dependabilidade. Isso acontece porque a geração é feita através de combinações entre as variáveis e os níveis, e alguma dessas pode não ser aplicada à realidade. Dessa forma, os usuários do serviço na nuvem devem verificar se os experimentos gerados (ou seja, os cenários) apresentam alguma incoerência. As IaaS especificadas com apenas um nó e uma VMs, por exemplo, não permitem receber mecanismo de redundância, já que para modelar a redundância é necessário ter uma quantidade de componentes superior a um. Mas essa combinação gerada no plano não foi modelada. Utilizou-se o símbolo $(*)$ para indicar as IaaS que não foram modeladas e avaliadas.

[6] Sem redundância (N/R).

Tabela 23 – Plano de Experimentos das IaaS.

IaaS	Nó	VM	RS	RT	IaaS	Nó	VM	RS	RT
1	1	1	Ouro	N/R	37*	2	2	Ouro	N/R
2*	1	1	Ouro	*Hot*	38	2	2	Ouro	*Hot*
3*	1	1	Ouro	*Cold*	39	2	2	Ouro	*Cold*
4*	1	1	Ouro	*Warm*	40	2	2	Ouro	*Warm*
5	1	1	Prata	N/R	41*	2	2	Prata	N/R
6*	1	1	Prata	*Hot*	42	2	2	Prata	*Hot*
7*	1	1	Prata	*Cold*	43	2	2	Prata	*Cold*
8*	1	1	Prata	*Warm*	44	2	2	Prata	*Warm*
9	1	1	Bronze	N/R	45*	2	2	Bronze	N/R
10*	1	1	Bronze	*Hot*	46	2	2	Bronze	*Hot*
11*	1	1	Bronze	*Cold*	47	2	2	Bronze	*Cold*
12*	1	1	Bronze	*Warm*	48	2	2	Bronze	*Warm*
13*	1	2	Ouro	N/R	49*	3	1	Ouro	N/R
14	1	2	Ouro	*Hot*	50	3	1	Ouro	*Hot*
15	1	2	Ouro	*Cold*	51*	3	1	Ouro	*Cold*
16	1	2	Ouro	*Warm*	52*	3	1	Ouro	*Warm*
17*	1	2	Prata	N/R	53*	3	1	Prata	N/R
18	1	2	Prata	*Hot*	54	3	1	Prata	*Hot*
19	1	2	Prata	*Cold*	55*	3	1	Prata	*Cold*
20	1	2	Prata	*Warm*	56*	3	1	Prata	*Warm*
21*	1	2	Bronze	N/R	57*	3	1	Bronze	N/R
22	1	2	Bronze	*Hot*	58	3	1	Bronze	*Hot*
23	1	2	Bronze	*Cold*	59*	3	1	Bronze	*Cold*
24	1	2	Bronze	*Warm*	60*	3	1	Bronze	*Warm*
25*	2	1	Ouro	N/R	61*	3	2	Ouro	N/R
26	2	1	Ouro	*Hot*	62	3	2	Ouro	*Hot*
27	2	1	Ouro	*Cold*	63*	3	2	Ouro	*Cold*
28	2	1	Ouro	*Warm*	64*	3	2	Ouro	*Warm*
29*	2	1	Prata	N/R	65*	3	2	Prata	N/R
30	2	1	Prata	*Hot*	66	3	2	Prata	*Hot*
31	2	1	Prata	*Cold*	67*	3	2	Prata	*Cold*
32	2	1	Prata	*Warm*	68*	3	2	Prata	*Warm*
33*	2	1	Bronze	N/R	69*	3	2	Bronze	N/R
34	2	1	Bronze	*Hot*	70	3	2	Bronze	*Hot*
35	2	1	Bronze	*Cold*	71*	3	2	Bronze	*Cold*
36	2	1	Bronze	*Warm*	72*	3	2	Bronze	*Warm*

Inicialmente, são apresentadas as configurações das IaaS a partir do planejamento de experimento, seguido dos modelos de dependabilidade, para, então, serem descritos os resultados das análises das métricas de disponibilidade, indisponibilidade e confiabilidade. A Figura 52 ilustra o modelo com o Frontend e um Nó (com uma máquina virtual hospedada). Esse modelo RBD foi utilizado para representar as IaaS: 1, 5 e 9 descritas na Tabela 23. Cada uma dessas configurações utiliza um tempo médio de reparo (sendo ouro, prata e bronze). Com essa estratégia, é possível diversificar as IaaS através de políticas de reparos com ou sem prioridades. O modo operacional do modelo RBD é representado por meio da Expressão 6.6, ou seja, para que o ambiente esteja operando, é necessário que os dois componentes em série não apresentem falhas.

$$\mathtt{RBD}_{OP} = \mathtt{FrontEnd} \wedge \mathtt{Nó} \tag{6.6}$$

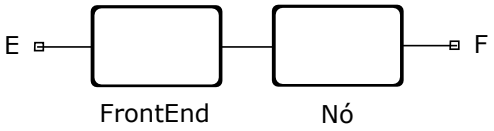


Figura 52 – Frontend e Nó sem redundância.

A Tabela 24 apresenta os parâmetros do modelo RBD (Fig. 52) com as respectivas descrições e a Tabela 25 apresenta os valores utilizados para a avaliação de dependabilidade.

Tabela 24 – Parâmetros do modelo Frontend e Nó sem redundância.

Parâmetro	Descrição
FE_F	Tempo médio para defeito do FrontEnd
FE_R	Tempo médio para reparo do FrontEnd
NO_F	Tempo médio para defeito do Nó
NO_R	Tempo médio para reparo do Nó

Tabela 25 – Valores do modelo Front e Nó sem redundância.

	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
FrontEnd	481,46564885	1,03660458	1,13660458	1,23660458
Nó (1VM)	389,12422030	0,91545217	1,01545217	1,11545217

Na Figura 53, é apresentada a abordagem de modelagem hierárquica para composição do modelo RBD que representa o nó com redundância na máquina virtual. O fluxo da modelagem hierárquiva deve ser considerada a partir do modelo envolvido pela linha pontilhada. Esse modelo RBD representa um nó com uma redundância *hot* na máquina virtual VM1 (VM1 + VM2). A partir da composição hierárquica, os blocos representados por

hardware, sistema operacional, gerenciador e máquina virtual do modelo foram agrupados e passaram a ser representados por apenas um bloco RBD (Conforme seta indicativa).

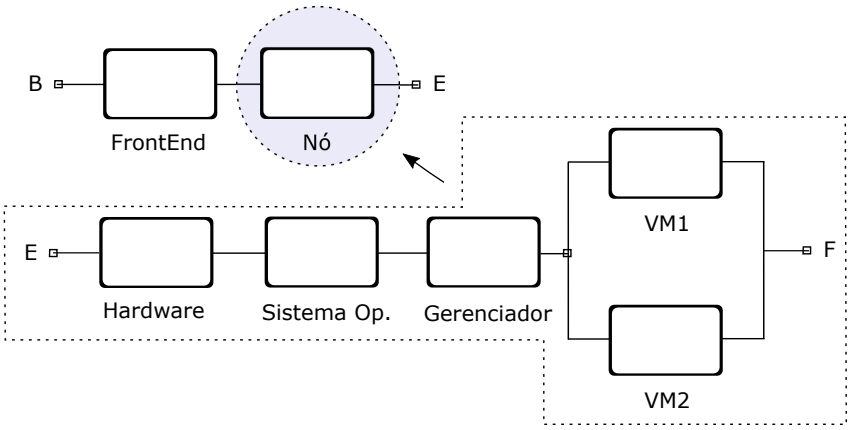


Figura 53 – Frontend e Nó com redundância *HotStandby* na VM.

As Tabelas 26 e 27 apresentam, respectivamente, as descrições dos parâmetros de dependabilidade, valores médios de defeito e reparo adotados para a avaliação, e modelagem hierárquica na Figura 53. Com o modelo RBD hierárquico, foi possível representar as IaaS 14, 18 e 22, assim como obter as métricas de disponibilidade e confiabilidade. Cada IaaS representou um tipo de serviço de reparo visando diferenciar os níveis de dependabilidade. O modo operacional do modelo RBD é representado por meio da condição (6.7), ou seja, para que o ambiente esteja operando, os dois componentes em série não podem apresentar falhas.

$$\mathtt{RBD}_{OP} = \mathtt{FrontEnd} \wedge \mathtt{Nó} \tag{6.7}$$

Tabela 26 – Parâmetros do modelo Frontend e Nó com redundância *HotStandby* na VM.

Parâmetro	Descrição
FE_F	Tempo médio para defeito do FrontEnd
FE_R	Tempo médio para reparo do FrontEnd
HW_F	Tempo médio para defeito do Hardware
HW_R	Tempo médio para reparo do Hardware
SO_F	Tempo médio para defeito do Sistema Operacional
SO_R	Tempo médio para reparo do Sistema Operacional
MG_F	Tempo médio para defeito do Gerenciador
MG_R	Tempo médio para reparo do Gerenciador
VM_F	Tempo médio para defeito da VM1 e VM2
VM_R	Tempo médio para reparo da VM1 e VM2

Tabela 27 – Parâmetros do Nó com redundância *HotStandby* na VM.

Componente	MTTF (Hr)	MTTR (Hr)
Hardware	8760	1,666
Sistema Operacional	1440	1
Gerenciador	788,40	1
Máquina Virtual	2880,00	0,50

A Tabela 28 apresenta os parâmetros de dependabilidade do modelo RBD (Fig. 53) para o Frontend e Nó com redundância *hot* na máquina virtual após a composição hierárquica.

Tabela 28 – Valores do modelo Front e Nó com composição hierárquica.

	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
FrontEnd	481,46564885	1,03660458	1,13660458	1,23660458
Nó (2VM)	451,74506937	0,97319996	1,07319996	1,17319996

Já as IaaS `15`, `19` e `23` utilizam o mecanismo de redundância *coldstandby* na máquina virtual. O mecanismo de redundância *coldstandby* foi modelado através das SPNs. A Figura 54 ilustra o modelo de dependabilidade SPN para representar as IaaS. Considerando o RBD do Frontend e nó, utilizaram-se os valores médios de reparo e defeito para a representação em submodelos de redes de Petri estocástica. O Frontend apresentado por um bloco RBD passou a ser representado por um submodelo SPN (`FRONTEND_ON`). O nó que é representado pelos componentes *hardware*, sistema operacional e gerenciador foram combinados e passaram a ser representados por um submodelo SPN (`HSMg_ON`), enquanto a máquina virtual foi representada por meio do mecanismo de redundância *coldstandby* com dois submodelos SPNs (`VM1_ON` e `VM2_ON`).

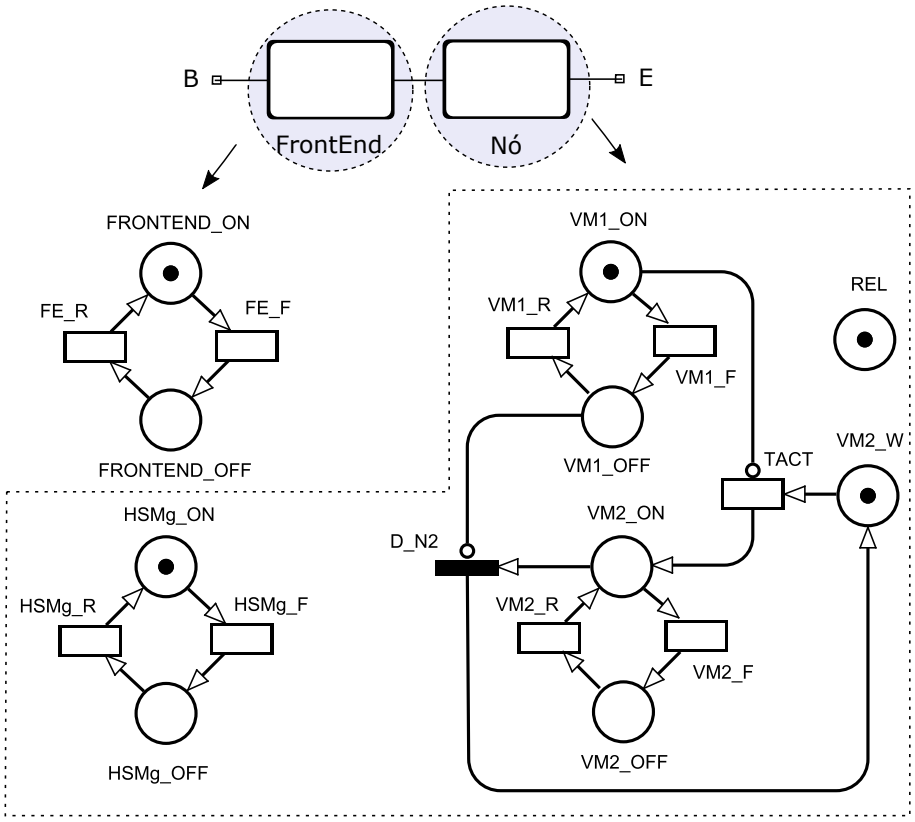


Figura 54 – Frontend e Nó com redundância *ColdStandby* na VM.

A Tabela 29 apresenta a descrição dos parâmetros de dependabilidade para o modelo SPN (Ver Fig. 54). A Tabela 30 ilustra os valores adotados para os parâmetros de dependabilidade.

Tabela 29 – Parâmetros do modelo Frontend e Nó com redundância *ColdStandby* na VM.

Parâmetro	Descrição
FE_F	Tempo médio para defeito do FrontEnd
FE_R	Tempo médio para reparo do FrontEnd
HSMg_F	Tempo médio para defeito do HSMg
HSMg_R	Tempo médio para reparo do HSMg
VM1_F e VM2_F	Tempo médio para defeito da VM1 e VM2
VM1_R e VM2_R	Tempo médio para reparo da VM1 e VM2

Tabela 30 – Valores do modelo Frontend e Nó com redundância *ColdStandby* na VM.

	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
FrontEnd	481,465649	1,03660458	1,13660458	1,23660458
HSMg	481,465649	1,03660458	1,13660458	1,23660458
Máquina Virtual	2028,882899	0,40150948	0,50150948	0,60150948

A disponibilidade das IaaS 15, 19 e 23 foram obtidas a partir da Equação 6.8. A métrica de confiabilidade foi obtida também com a Equação 6.8, no entanto, foi preciso garantir que as transições de reparo dos submodelos SPNs não disparassem. Para isso, a condição (IF REL = 1: 2 ELSE 1) foi adicionada aos arcos que saem do lugar que representa o estado de falha, por exemplo, VM2_OFF para a transição de reparo e FE_R de cada um dos submodelos. Na presença de um *token* no lugar REL, a expressão condicional ativa um peso = 2 nas transições que ligam os lugares de reparo, impossibilitando o seu disparo. Dessa maneira, foi possível obter a métrica de confiabilidade das IaaS.

$$\mathrm{SPN}_{OP}=((\mathrm{FRONTEND\_ON}>0)\wedge(\mathrm{HSMg\_ON}>1)\wedge((\mathrm{NODE1\_ON}>0)\vee(\mathrm{OPNODE2\_ON}>0)) \tag{6.8}$$

A Figura 55 ilustra o modelo de dependabilidade SPN para representar as IaaS 16, 20 e 24. Como na abordagem anterior, utilizou-se a modelagem hierárquica para representar os componentes (*Hardware*, sistema operacional, gerenciador e máquina virtual) em submodelos de redes de Petri estocástica. O conjunto de modelos SPNs representa o mecanismo de redundância *WarmStandby* adotado na máquina virtual. O funcionamento do mecanismo *warmstandby* foi representado através dos submodelos SPNs VM1_ON, VM2_ON e OPVM2_ON. A Equação para a obtenção da disponibilidade é dado por 6.9. Para a obtenção da métrica de confiabilidade, utilizou-se a mesma abordagem descrita para o modelo *coldstandby* descrita no parágrafo anterior.

$$\mathrm{SPN}_{Op}=((\mathrm{FRONTEND\_ON}=1)\wedge(\mathrm{HSMg\_ON}=1)\wedge((\mathrm{VM1\_ON}=1)\vee(\mathrm{OPVM2\_ON}=1))) \tag{6.9}$$

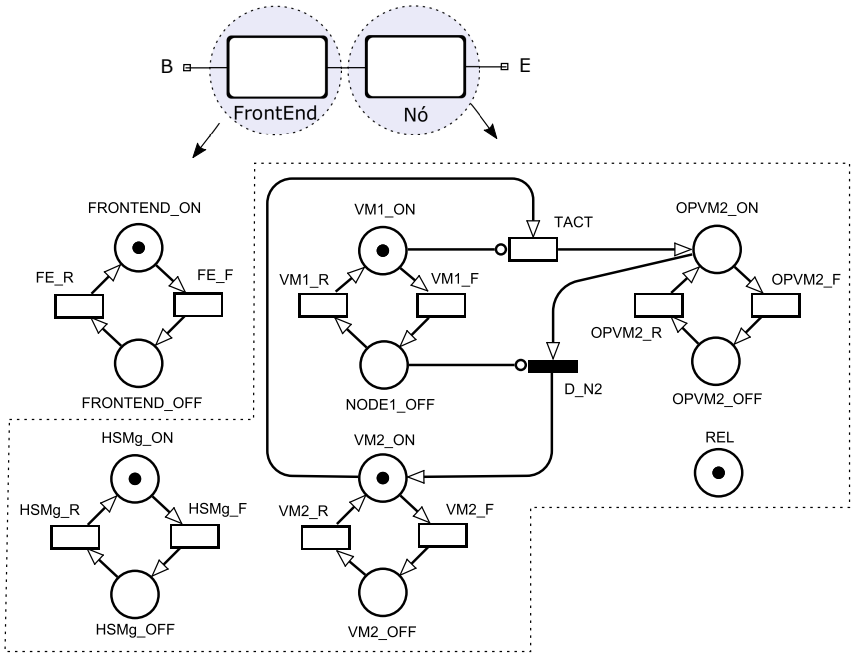


Figura 55 – Frontend e Nó com redundância *WarmStandby* na VM.

A Tabela 31 apresenta os parâmetros do modelo SPN com redundância *warm* e as respectivas descrições. Já a Tabela 32 apresenta os valores utilizados para a análise de dependabilidade. O modelo que representa as infraestruturas como serviço 16, 20 e 24 com redundância *warm* na máquina virtual foram avaliados adotando os tipos de serviço de reparo ouro, prata e bronze respectivamente. Os resultados de disponibilidade, confiabilidade e indisponibilidade serão apresentados após a descrição das IaaS geradas através do plano de experimento.

Tabela 31 – Parâmetros do modelo Front e Nó com redundância *WarmStandby* na VM.

Parâmetro	Descrição
FE_F	Tempo médio para defeito do FrontEnd
FE_R	Tempo médio para reparo do FrontEnd
HSMg_F	Tempo médio para defeito do HSMg
HSMg_R	Tempo médio para reparo do HSMg
VM1_F, VM2_F e OPVM2_F	Tempo médio para defeito da VM1, VM2 e OPVM2
VM1_R e VM2_R e OPVM2_R	Tempo médio para reparo da da VM1, VM2 e OPVM2

Tabela 32 – Valores do modelo Frontend e Nó com redundância *ColdStandby* na VM.

	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
FrontEnd	481,465649	1,03660458	1,13660458	1,23660458
HSMg	481,465649	1,03660458	1,13660458	1,23660458
Máquina Virtual	2028,882899	0,40150948	0,50150948	0,60150948
Máquina Virtual OP.	2434,659478	0,40150948	0,50150948	0,60150948

As IaaS 26, 30 e 34 utilizam o mecanismo de redundância *hotstandby* no nó. Essas IaaS consideram dois nós e cada nó possui uma máquina virtual. A Figura 56 ilustra o modelo de dependabilidade RBD para representar essas IaaS. O modo operacional do modelo RBD é representado através da seguinte Equação 6.10. Na falha do Nó 1, o Nó 2 assume a função do nó principal sem que a infraestrutura fique indisponível. Através do modelo RBD, foram possíveis a análise e obtenção das métricas de disponibilidade e confiabilidade levando em conta os três serviços de reparo (ouro, prata e bronze).

$$\mathtt{RBD}_{OP} = \mathtt{Frontend} \wedge (\mathtt{Nó\ 1} \vee \mathtt{Nó\ 2}) \tag{6.10}$$

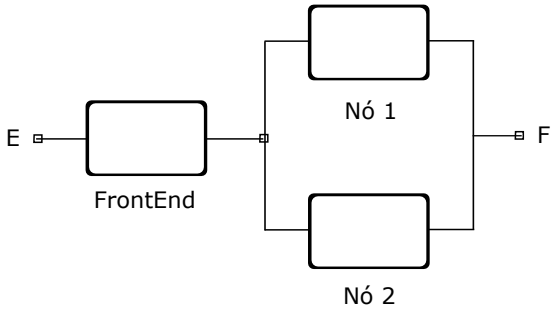


Figura 56 – Frontend e Nó com redundância *HotStandby*.

A Tabela 33 apresenta os parâmetros do modelo RBD com as respectivas descrições e a Tabela 34 os valores utilizados para a avaliaçao das IaaS `26`, `30` e `34`.

Tabela 33 – Parâmetros do modelo Front e Nó com redundância *HotStandby*.

Parâmetro	Descrição
`MTTF_FrontEnd`	Tempo médio para defeito do FrontEnd
`MTTR_FrontEnd`	Tempo médio para reparo do FrontEnd
`MTTF_No1 e MTTF_N2`	Tempo médio para defeito do Nó 1 e Nó 2
`MTTR_No1 e MTTRF_No2`	Tempo médio para reparo do Nó 1 e Nó 2

Tabela 34 – Valores do modelo Front e Nó com redundância *HotStandby*.

	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
FrontEnd	481,46564885	1,03660458	1,13660458	1,23660458
Nó (1VM)	389,12422030	0,91545217	1,01545217	1,11545217

As IaaS `27`, `31` e `35` utilizam dois nós e uma VM em cada nó. Os nós foram representados por meio da redundância *coldstanby*. A Figura 57 ilustra o modelo de dependabilidade utilizado para representar essas infraestruturas como serviço. A disponibilidade é dado pela Equação 6.11. Os valores médios de reparo e defeito apresentados na Tabela 34 podem ser utilizados para representar os componentes Frontend e Nó do modelo SPN *coldstanby*.

$$\texttt{SPN}_{OP} \texttt{ = ((FRONTEND\_ON=1)} \wedge \texttt{((NODE1\_ON=1)} \vee \texttt{(NODE2\_ON=1)))} \tag{6.11}$$

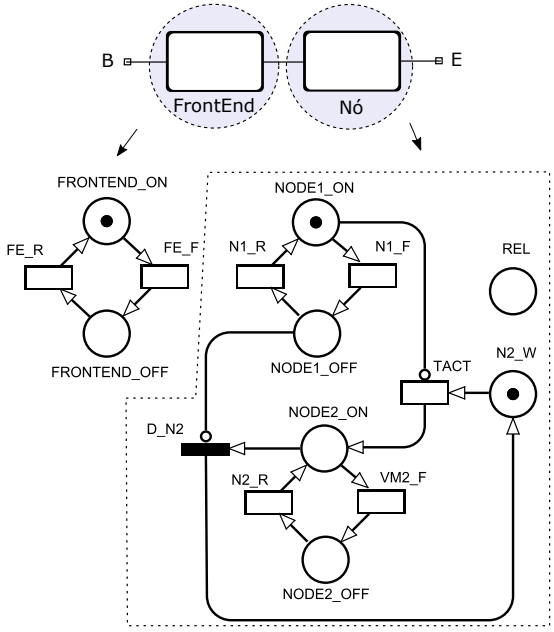


Figura 57 – Frontend e Nó com redundância *ColdStandby*.

A Figura 58, abaixo, ilustra o modelo de dependabilidade utilizado para representar as IaaS 28, 32 e 36. O modelo SPN é representado pelo Frontend e dois nós, cada nó possui uma VM. Na redundância dos nós, foi utilizado o mecanismo de redundância *warmstandby*. A expressão para a obtenção da disponibilidade é dado por (6.12). Os valores dos tempos médios de reparo e defeito para os componentes do modelo SPN podem ser obtidos na Tabela 34.

$$\mathrm{SPN}_{OP}=((\mathrm{FRONTEND\_ON}=1)\wedge((\mathrm{NODE1\_ON}=1)\vee(\mathrm{OPNODE2\_ON}=1))) \tag{6.12}$$

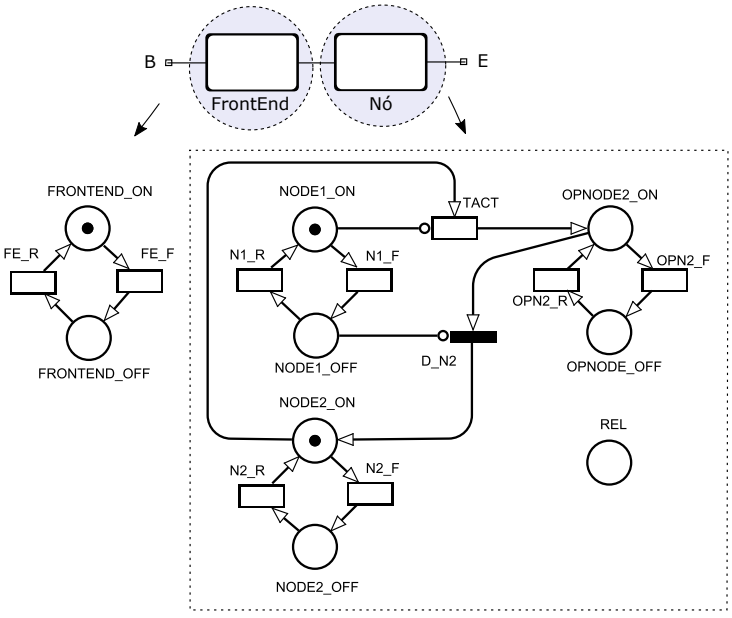


Figura 58 – Frontend e Nó com redundância *WarmStandby*.

Já as IaaS 38, 42 e 46 utilizam dois nós, onde cada nó possui duas máquinas virtuais. A Figura 59 ilustra o modelo de dependabilidade RBD para representar essas IaaS. Utilizou-se a abordagem de composição hierárquica para modelar o mecanismo de redundância *hotstanby* nas máquinas virtuais. Assim, a composição dos blocos *hardware*, sistema operacional, gerenciador e máquina virtual passou a ser representada em um único bloco. A disponibilidade é dado pela Equação 6.13.

$$\mathtt{RBD}_{OP} = \mathtt{Frontend} \wedge (\mathtt{Nó\ 1} \vee \mathtt{Nó\ 2}) \tag{6.13}$$

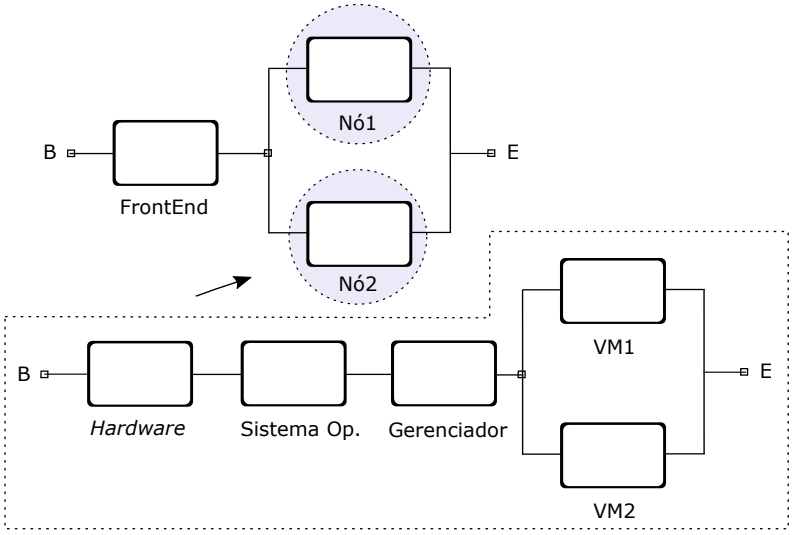


Figura 59 – Frontend e Nó com redundância *HotStandby*.

Os parâmetros de dependabilidade descritos na Tabela 33, tanto quanto os valores de reparo e defeito dos componentes apresentados na Tabela 35, foram utilizados para analisar as IaaS 38, 42 e 46 e na obtenção das métricas de disponibilidade, indisponibilidade e confiabilidade.

Tabela 35 – Valores do modelo Front e Nó com 2 VMs.

	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
FrontEnd	481,46564885	1,03660458	1,13660458	1,23660458
Nó (2VM)	451,74506937	0,97319996	1,07319996	1,17319996

As IaaS 39, 43 e 47 utilizam o mecanismo de redundância *coldstandby* no componente nó. Para essas infraestruturas, utilizou-se a composição hierárquica dos componentes *hardware*, sistema operacional, gerenciador e VMs (com redundância *hot*) para representar cada nó. A Figura 60 ilustra o modelo SPN adotado para representar essas IaaS. Os valores para os parâmetros de dependabilidade levando em conta os tipos de serviço de reparo e os tempos médios para defeito podem ser consultados na Tabela 35. A Tabela 33 pode ser utilizada para descrição dos parâmetros de dependabilidade do modelo SPN. A

Equação a seguir foi utilizada para a obtenção da disponibilidade das infraestruturas 6.14. Para a obtenção da métrica de confiabilidade, utilizou-se a mesma abordagem descrita para o modelo *cold* apresentado na Figura 55.

$$\texttt{SPN}_{OP}\texttt{=((FRONTEND\_ON=1)}\wedge\texttt{((NODE1\_ON=1)}\vee\texttt{(NODE2\_ON=1)))} \tag{6.14}$$

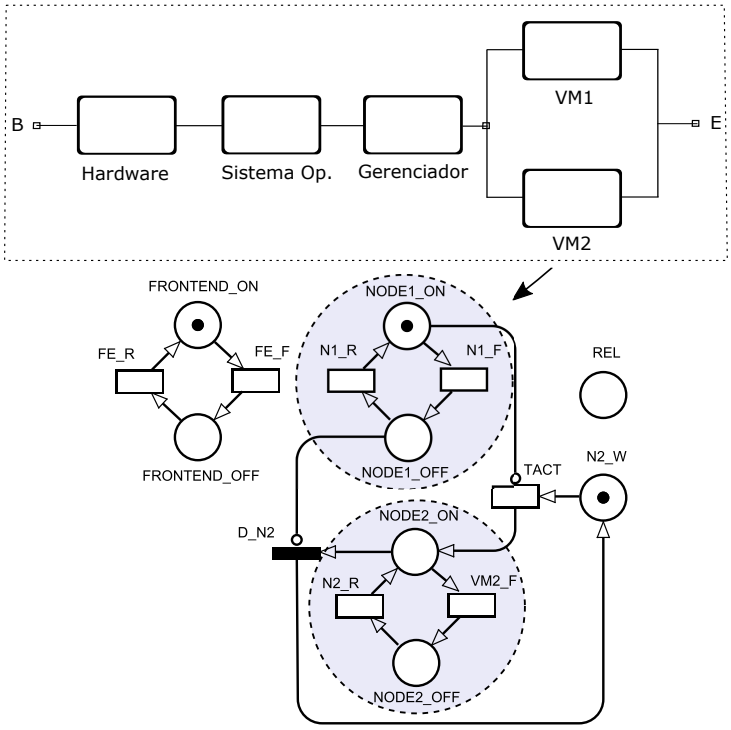


Figura 60 – Frontend e Nó com redundância *ColdStandby*.

As IaaS `40`, `44` e `48` utilizam o mecanismo de redundância *warmstandby* no componente nó. A Figura 61 apresenta o modelo SPN para o mecanismo *warm* considerando a composição hierárquica dos componentes do modelo RBD. Para a obtenção da métrica de confiabilidade, utilizou-se a mesma abordagem descrita para o modelo *cold* apresentado na Figura 55. A disponibilidade é dada pela Equação 6.15.

$$\texttt{SPN}_{OP}\texttt{=((FRONTEND\_ON=1)}\wedge\texttt{((NODE1\_ON=1)}\vee\texttt{(OPNODE2\_ON=1)))} \tag{6.15}$$

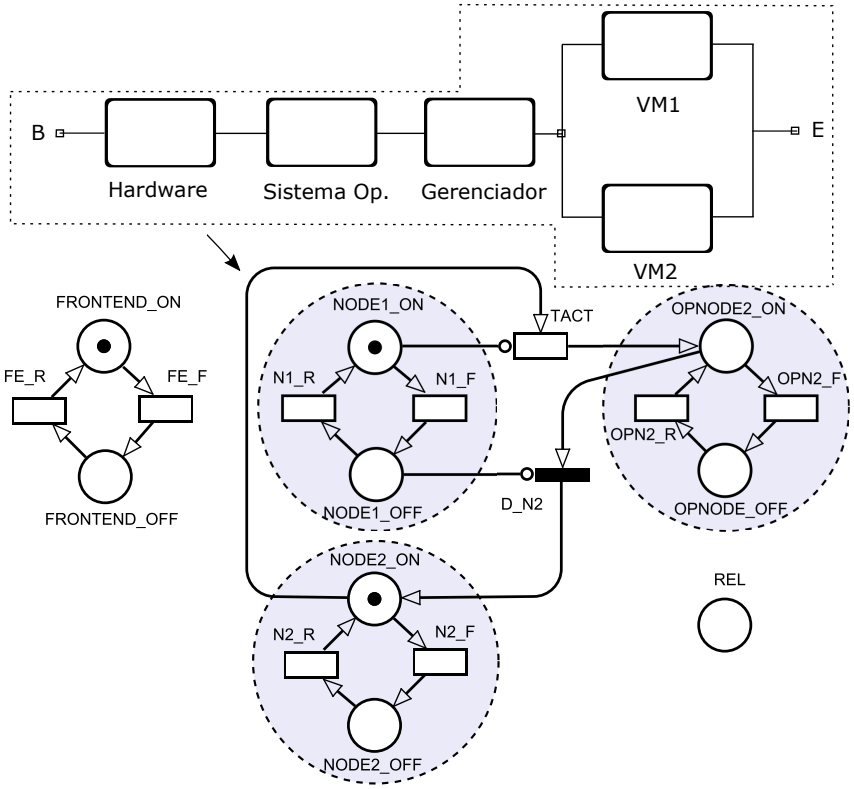


Figura 61 – Frontend e Nó com redundância *WarmStandby*.

A Tabela 36 ilustra os valores para os parâmetros de dependabilidade utilizados para avaliar os tipos de serviço de reparo (ouro, prata e bronze) e os tempos médios para defeito.

Tabela 36 – Valores do modelo Front e Nó com 2 VMs com redundância *WarmStandby*.

	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
FrontEnd	481,46564885	1,03660458	1,13660458	1,23660458
Nó (2VM)	451,74506937	0,97319996	1,07319996	1,17319996
Nó OP. (2VM)	542,09408324	0,97319996	1,07319996	1,17319996

Para as IaaS `50`, `54` e `58`, foi utilizado o mecanismo de redundância tripla modular *hot*. Cada nó possui uma máquina virtual. Esse mecanismo possui três componentes nós e na ocorrência de uma falha no nó principal, o nó redundante assume sua função. Caso o segundo nó (reserva) apresente também uma falha, o terceiro nó assumirá a função. A Figura 62 apresenta o modelo RBD para essas infraestruturas como serviço. O modo operacional desse modelo é dado por 6.16.

$$\texttt{RBD}_{OP}\texttt{=((FrontEnd)}\wedge\texttt{((Nó1)}\vee\texttt{(Nó2)}\vee\texttt{(Nó3)))} \tag{6.16}$$

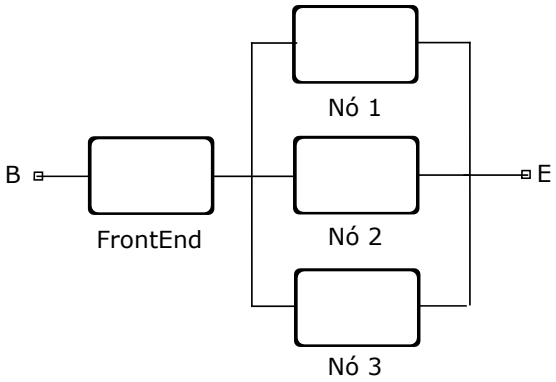


Figura 62 – Frontend e Nó com tripla redundância *HotStandby*.

O Modelo apresentado na Fig. 62 utilizou os valores de tempo médio para defeito e reparo do modelo Front e Nó com redundância apresentados na Tabela 34. A seguir, a Tabela 37 vem descrever os parâmetros utilizados.

Tabela 37 – Parâmetros do modelo Front e Nó com tripla redundância *HotStandby*.

Parâmetro	Descrição
`MTTF_FrontEnd`	Tempo médio para defeito do FrontEnd
`MTTR_FrontEnd`	Tempo médio para reparo do FrontEnd
`MTTF_Nó 1, 2 e 3`	Tempo médio para defeito dos Nó 1, Nó 2 e Nó 3
`MTTR_Nó 1, 2 e 3`	Tempo médio para reparo dos Nó 1, Nó 2 e Nó 3

Por fim, as IaaS `62`, `66` e `70` consideram o mecanismo de tripla redundância modular nos nós. No entanto, cada IaaS possui três componentes nós e cada componente nó é composto por uma redundância com duas máquinas virtuais. A seguir, na Figura 63 é mostrado o modelo RBD para representar essa infraestrutura como serviço. Esse modelo traz a maior quantidade de redundâncias nos componentes e isso poderá gerar mais custos na operação da IaaS. Essa redundância pode ser indicada para serviços críticos que requeram alto nível de disponibilidade e necessitam de mecanismos que garantam a continuidade do serviço. Sendo assim, citamos algumas situações como serviços para armazenamento de dados em infraestrutuas na nuvem (IaaS), plataformas de desenvolvimento na nuvem (PaaS), serviços de gerenciamento de negócios na nuvem (SaaS), ferramentas de comunicação (e-mail, calendários, aplicativos de chamada) baseadas em infraestrutura na nuvem e serviços de *streaming* sob demanda. Para esse modelo, foram usados os valores de dependabilidade apresentados na Tabela 35. O modo operacional desse modelo é dado por 6.17.

$$\texttt{RBD}_{OP}\texttt{=((FrontEnd)}\wedge\texttt{((Nó1)}\vee\texttt{(Nó2)}\vee\texttt{(Nó3)))} \tag{6.17}$$

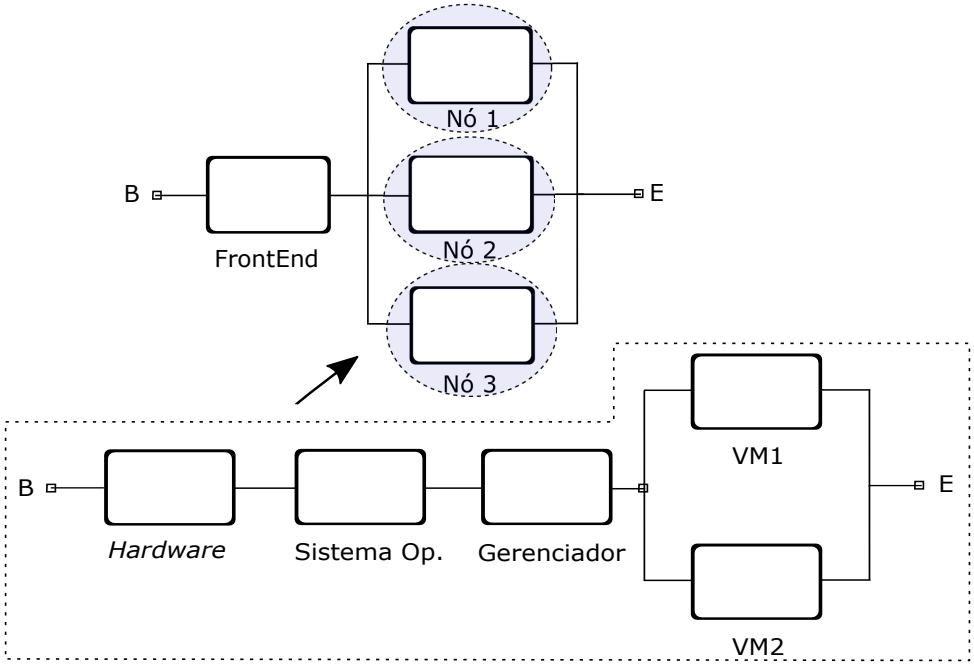


Figura 63 – Frontend e Nó com tripla redundância *Hot* com duas VMS.

Após o processo de planejamento de experimentos e modelagem de cada uma das IaaS, o passo seguinte foi a avaliação dos modelos de dependabilidade baseados em RBD e SPN. Foram consideradas as métricas de dependabilidade: disponibilidade, disponibilidade orientada a capacidade, confiabilidade e indisponibilidade. A Tabela 38 apresenta os resultados obtidos para cada configuração modelada. O resultado da confiabilidade é apresentado considerando a ocorrência de uma falha em um período de 24 hrs.

Tabela 38 – Resultados de dependabilidade das IaaS.

IaaS	Disponibilidade (%)	Disponibilidade Orientada à Capacidade (%)	Confiabilidade (%)	Indisp. (Min)
1	99,5509574705	99,7654338777	89,4469247967	196,68062793
5	99,5048180576	99,7398556502	89,4469247967	216,88969077
9	99,4587108348	99,7142905588	89,4469247967	237,08465436
14	99,5706543903	99,8728535266	90,2149434700	188,05337705
15	99,5616066096	99,7759681848	90,4897545045	192,01630500
16	99,5650001940	99,7793663489	90,4925359715	190,52991502
18	99,5280378157	99,8576135555	90,2149434700	206,71943671
19	99,5194114138	99,7543530012	90,4897545045	210,49780078
20	99,5233822554	99,7583295360	90,4925359715	208,75857214
22	99,4854486046	99,8423824134	90,2149434700	225,37351118
23	99,4774397512	99,7329443057	90,4897545045	228,88138896
24	99,4818832542	99,7373945400	90,4925359715	226,93513466
26	99,7846109835	99,9792913946	94,7970536818	94,34038921
27	99,7201853569	99,9258299977	94,8881075626	122,55881369
28	99,7486208259	99,9499826254	94,7885068480	110,10407827
30	99,7638083387	99,9741786568	97,970536818	103,45194767
31	99,6976733870	99,9196490290	94,8881075626	132,41905652
32	99,7272945753	99,9445963495	94,7885068480	119,44497603
34	99,7430013430	99,9690494692	94,7970536818	112,56541175
35	99,6754125418	99,9136202473	94,8881075626	142,16930671
36	99,7060521418	99,9392534800	94,7885068480	128,74916191
38	99,7846995558	99,9763972167	94,8827293630	94,30159457
39	99,7282517140	99,9363941560	94,9415323833	119,02574928
40	99,7534014438	99,9641014864	94,8717737486	108,01016760
42	99,7639238067	99,9712822680	94,8827293630	103,40137265
43	99,7061245286	99,9341938729	94,9415323833	128,71745648
44	99,7322176473	99,9633095757	94,8717737486	117,28867048
46	99,7431470289	99,9661503994	94,8827293630	112,50160136
47	99,6841904563	99,9323432327	94,9415323833	138,32458013
48	99,7110986022	99,9626569150	92,7019871643	126,53881225
50	99,7851593857	99,9802082869	95,1170633317	94,10018905
54	99,7644824363	99,9752797697	95,1170633317	103,15669289
58	99,7438139525	99,9703513633	95,1170633317	112,20948879
62	99,7851596846	99,9802044071	95,1242439537	94,10005812
66	99,7644828673	99,9752736682	95,1242439537	103,15650413
70	99,7438147914	99,9703425762	95,1242439537	112,20912137

6.3 AVALIAÇÃO DO MODELO DE CUSTO

Este estudo tem o intuito de estimar os custos das infraestruturas como serviço em nuvem modeladas na seção anterior. Os componentes dos modelos de dependabilidade, por exemplo, número de nós e máquinas virtuais foram considerados para compor as estimativas. Além disso, os custos englobam atividades de utilização, manutenção e operação das IaaS. Baseado na estimativa dos custos, gestores de ambientes com serviço na nuvem podem utilizar essas informações para planejar demandas atuais e para atender futuras demandas. Os gestores das infraestruturas podem utilizar essas informações para realizar o detalhamento das atividades desempenhadas em cada serviço pelos usuários. As equações que estimam os valores foram descritas na Seção 5.2.

A seguir, a Tabela 39 apresenta os valores dos parâmetros utilizados para o cálculo do custo de cada IaaS. Ep e Ec representam, nesta ordem, o preço da energia (em R$)[7] e o consumo de energia por quilowatt-hora da infraestrutura configurada no laboratório [8]. Lb_{Up} (custo de operação da IaaS) e Lb_{Dwt} (custo de manutenção da IaaS) indicam o custo da hora trabalhada em operação e manutenção. $\sum L_c$ representa somatório dos custos dos componentes que compõem um nó[9]. Os parâmetros ouro, prata ou bronze foram associados aos diferentes tipos de reparo de serviço. Desta forma, para diferenciar os custos utilizamos o produto do fator numérico de cada tipo de reparo pelo valor monetário da hora de trabalho na operação e manutenção. Os valores dos parâmetros ilustram a relação de diferentes custos de reparo associados a diferentes tempos de reparo.

Tabela 39 – Parâmetros de Custos.

Parâmetro	Valor
Ep	R$ 0,3545
Ec	0,40 (kWh)
Lb_{Up}	R$ 0,04
Lb_{Dwt}	R$ 0,40
$\sum L_c$	R$ 1.098,00
Ouro	1,44
Prata	1,27
Bronze	1,15

A Tabela 40 apresenta os custos calculados para cada uma das atividades consideradas na estimativa do custo total das IaaS obtidas no planejamento de experimento. TCe representa o somatório dos custos de utilização, manutenção e operação das IaaS. Na próxima subseção, os resultados de dependabilidade e custos obtidos foram utilizados no modelo de decisão considerando funções multicritérios.

[7] (CELPE, 2018).
[8] Os experimentos foram realizadas no laboratório do grupo MoDCS (MODCS, 2018).
[9] (BESTBUY, 2018).

Tabela 40 – Resultados dos custos das IaaS.

	Custo (R$)			
IaaS	Utilização	Manutenção	Operação	TCe
1	159,59	1,88	144,84	306,31
5	159,51	1,83	139,76	301,11
9	159,44	1,82	136,36	297,62
14	159,62	1,80	144,87	306,29
15	159,61	1,84	144,86	306,30
16	159,61	1,83	144,86	306,30
18	159,55	1,74	139,79	301,09
19	159,54	1,78	139,78	301,10
20	159,54	1,76	139,79	301,09
22	159,48	1,73	136,39	297,61
23	159,47	1,75	136,38	297,61
24	159,48	1,74	136,39	297,61
26	159,96	1,81	187,07	348,84
27	159,86	2,07	176,90	338,83
28	159,91	1,69	170,25	331,84
30	159,93	1,98	187,03	348,94
31	159,82	2,54	186,90	349,26
32	159,87	2,29	186,96	349,12
34	159,90	1,90	176,94	338,73
35	159,79	2,40	176,82	339,01
36	159,84	2,17	176,87	338,88
38	159,96	1,45	170,31	331,72
39	159,87	1,83	170,22	331,91
40	159,91	1,66	170,26	331,83
42	319,86	1,98	290,30	612,14
43	319,67	2,47	290,13	612,28
44	319,76	2,25	290,21	612,22
46	319,79	1,90	280,19	601,88
47	319,60	2,33	280,03	601,97
48	319,69	2,13	280,10	601,93
50	319,93	1,44	273,61	594,98
54	319,86	1,58	273,55	595,00
58	319,79	1,72	273,50	595,01
62	319,93	1,80	290,36	612,10
66	319,86	1,74	280,25	601,85
70	319,79	1,72	273,50	595,01

6.4 AVALIAÇÃO DO MODELO DE DECISÃO

Diante da variedade de infraestruturas como serviço avaliadas através dos modelos de dependabilidade e modelo de custo, este estudo tem como objetivo realizar um *ranking* das IaaS considerando o modelo de decisão proposto. O modelo de decisão utiliza um conjunto de métodos baseados em técnicas de agrupamento para selecionar um conjunto de soluções desejadas considerando as métricas mencionadas na subseção anterior. Para que as IaaS sejam selecionadas, é necessário definir uma função multicritério, que pode ser composta pelas métricas de dependabilidade e custo. O gestor ou analista de uma infraestrutura como serviço na nuvem precisa identificar quais os parâmetros deseja priorizar no método de ranqueamento. Essas informações são necessárias para então gerar o conjunto de soluções que mais se aproximam das necessidades.

Havendo situações em que o usuário possua dificuldade sobre quais parâmetros deseja considerar, o modelo de decisão disponibiliza também uma função que considera intervalos com valor máximo e mínimo, fazendo uma busca no conjunto de resultados das IaaS. Os métodos de seleção podem auxiliar usuários no momento da negociação de contratos entre provedores de serviço em nuvem, oferecendo estratégias que possam estar dentro das possibilidades de cada usuário. Assim, os gestores de ambientes em nuvem podem identificar e sugerir de maneira clara as soluções que se adequem às condições de cada usuário. A seguir, os resultados são apresentados considerando quatro *rankings*: (i) menor custo e menor indisponibilidade, (ii) maior confiabilidade e menor custo, (iii) maior disponibilidade e maior confiabilidade, (iv) valores intervalados com maior disponibilidade e custo.

6.4.1 Indicador de Qualidade

Com o propósito de garantir a corretude do ranqueamento gerado a cada avaliação do modelo multicritério, adotamos a seguinte estratégia, a cada conjunto de resultado obtido, geramos também um conjunto de resultado invertendo o objetivo da função multicritério. Isso se deu da seguinte maneira, considerando uma função multicritério com o propósito de encontrar soluções com menor custo e menor indisponibilidade, realizamos também outro ranqueamento invertendo o objetivo da função multicritério para maior custo e maior indisponibilidade. Assim, é possível verificar se ambos os ranqueamentos apresentam a mesma sequência com os cenários ordenados. Adotamos o teste estatístico não-paramétrico Wilcoxon [10] para realizar essa comparação.

Para realizar a estratégia, foi necessária a normalização dos dados e, em seguida a realização do cálculo da medida de similaridade adotada no estudo de caso considerando a ordenação inversa. Dessa maneira, as soluções que apresentaram as maiores distâncias nos

[10] O teste de Wilcoxon é um método não-paramétrico para comparação de duas amostras pareadas (LESIK, 2009).

estudos de caso, obrigatoriamente deveriam apresentar as menores distâncias, e vice-versa. Em seguida, foi executado o teste estatístico não-paramétrico Wilcoxon. As Tabelas 61, 62 e 63, localizadas no Apêndice A, ilustram a comparação do indicador de qualidade para os casos abordados neste capítulo.

6.4.2 Ranking (I)

Com o intuito de demonstrar a aplicabilidade do modelo de decisão, o primeiro caso leva em consideração uma situação em que um dado usuário deseja encontrar uma IaaS que possua menor custo e indisponibilidade. Sendo assim, foram selecionados dois parâmetros (TCe e Indisponibilidade) para representar cada IaaS. Em seguida, foi utilizado o modelo de decisão implementado na ferramenta MiPACE. A medida de similaridade utilizada para o cálculo de agrupamento dos dados foi a distância euclidiana. A Tabela 41 ilustra o *ranking* das IaaS, considerando os parâmetros menor custo e indisponibilidade.

Tabela 41 – Ranking das IaaS (minimizar custo e indisponibilidade).

Ranking	IaaS	Ranking	IaaS
1	38	19	19
2	40	20	5
3	28	21	22
4	26	22	24
5	30	23	23
6	34	24	50
7	39	25	54
8	27	26	70
9	32	27	58
10	36	28	66
11	31	29	46
12	35	30	48
13	14	31	62
14	16	32	9
15	15	33	42
16	1	34	44
17	18	35	47
18	20	36	43

A partir do *ranking* apresentado na Tabela 41, foi possível identificar as soluções que apresentaram o menor custo e indisponibilidade. A IaaS (`38`) foi a opção selecionada para a primeira posição do *ranking*. Essa IaaS apresenta a seguinte configuração: dois nós e duas VMs com redundância *hotstandby*, e um serviço de reparo `ouro`. A segunda opção no *ranking* foi a IaaS (`40`), que possui um acréscimo de R$ 0,11 (onze centavos) no custo

e 13, 71 minutos na indisponibilidade em relação à primeira opção que foi a IaaS (`38`). Em percentual, isso representa respectivamente, 0, 03% (três décimos percentuais) e 12, 69% (doze, sessenta e nove décimos percentuais) para o custo e indisponibilidade. Já a terceira IaaS (`28`) possui um acréscimo de 0, 04% (R$ 0,12) no custo e 14, 35% (15, 8 min) na indisponibilidade em relação à primeira opção. A última IaaS (`43`) no *ranking* apresenta uma diferença de 45, 82% (R$ 280,55) no custo e 26, 74% (34, 42 min) na indisponibilidade, considerando também a IaaS (`38`).

A Tabela 42 compara as características das primeiras infraestruturas ranqueadas com a última colocada. Enfatizando a IaaS (`38`), tem-se que os parâmetros de custo e indisponibilidade são influenciados pela redundância *hot* no nó e VM, aos parâmetros da política de reparo `ouro`, proporcionando assim a primeira posição no *ranking*. Já a IaaS na última colocação (36^a) apresenta uma redundância do tipo *coldstandby*, essa que necessita de um tempo ativação, combinado com uma política de reparo do tipo `prata`.

Tabela 42 – Sumário dos componentes utilizados para ranquear as IaaS (Primeiro Caso).

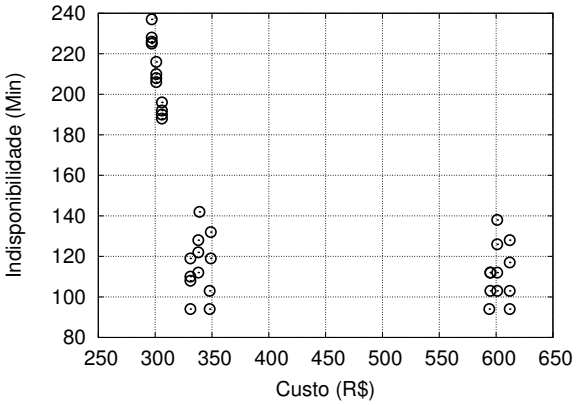
Ranking	IaaS	Nó	VM	RS	RT
1	38	2	2	Ouro	*Hot*
2	40	2	2	Ouro	*Warm*
3	28	2	1	Ouro	*Warm*
...	...	...	...	...	...
36	43	2	2	Prata	*Cold*

Na Figura 64, são apresentados os resultados do *ranking* gerados a partir do método de agrupamento com a medida de similaridade e o dominância de Pareto. A abordagem com a medida de similaridade possibilita ao usuário de IaaS um *ranking* considerando os cenários que mais se aproximam dos objetivos considerados, quais sejam, minimizar indisponibilidade e custo. Na segunda abordagem, através do conceito de dominância de Pareto foi possível encontrar um conjunto de soluções que dominem outras soluções. Nesse caso, também consideramos como objetivos localizar as soluções com menor custo e indisponibilidade. Por conseguinte, as soluções com maior custo e indisponibilidade quando comparadas à outra de menor custo e indisponibilidade são ditas dominadas. A Figura 64a, então, vem ilustrar a visão geral das IaaS apresentadas na Tabela 41. Os eixos representam os parâmetros de indisponibilidade em minutos e custo considerando a moeda real (R$). Na Figura 64b, é apresentado um conjunto ordenado a partir da medida de similaridade (Euclidiana) com as IaaS que mais se aproximam da função multicritério deste caso, mínimo custo e indisponibilidade.

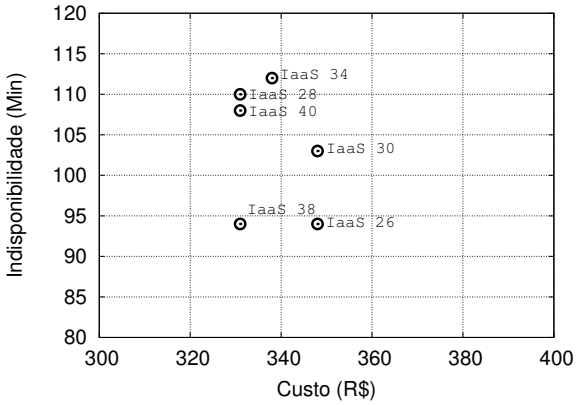
A Figura 64c apresenta a curva de Pareto[11] com destaque para nove (`9`) IaaS. Esse método identifica as soluções e mostra a relação de dominância entre cada solução. Entretanto, analistas e gestores de infraestruturas como serviço em nuvem devem verificar

[11] Representados pelos pontos preenchidos na cor preta.

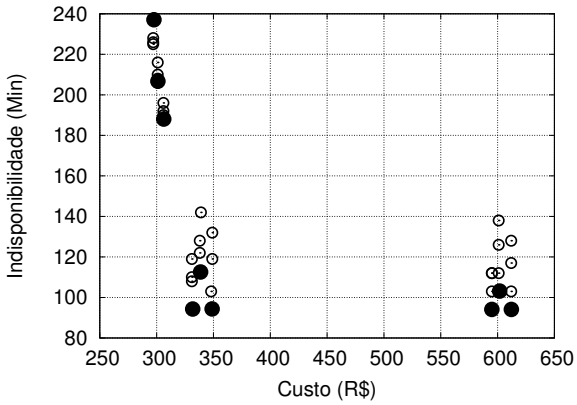
as condições de cada usuário para sugerir uma das soluções do conjunto. O conjunto das soluções não-dominadas, ou seja, conjunto Pareto-ótimo é composto das seguintes IaaS: 9, 18, 14, 38, 34, 26, 50, 66 e 62. Através do conceito do conjunto Pareto-ótimo podemos afirmar que uma das soluções não é melhor nem pior do que as outras, quando se levam em conta os dois objetivos.



(a) Visão Geral das 36 IaaS.



(b) *Ranking* das IaaS (Euclidiana).



(c) Conjunto das IaaS (Pareto).

Figura 64 – Conjunto das IaaS com menor custo e indisponibilidade.

6.4.3 Ranking (II)

O segundo caso proposto realizou um *ranking* das IaaS considerando uma situação em que um dado usuário deseja encontrar uma infraestrutura como serviço na nuvem que tenha maior confiabilidade e menor custo. Nesse contexto, a função multicritério busca atender a função para maximizar o primeiro objetivo e minimizar o segundo. A partir da seleção dos parâmetros confiabilidade e custo de cada IaaS, fez-se uso do modelo de decisão. A distância euclidiana foi escolhida como medida de similaridade para o agrupamento das IaaS. A Tabela 43 apresenta o resultado com o *ranking* das trinta e seis (36) IaaS considerando os parâmetros confiabilidade e custo.

Tabela 43 – Ranking das IaaS (maximizar confiabilidade e minimizar custo).

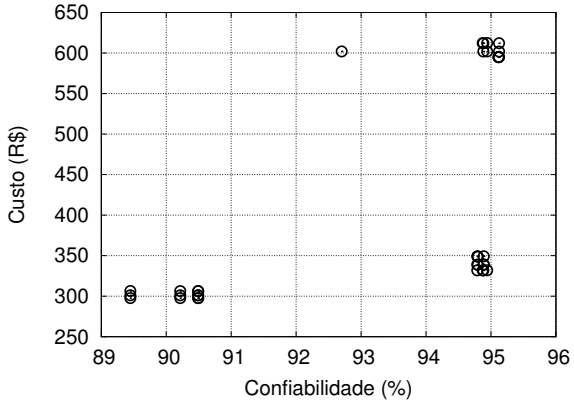
Ranking	IaaS	Ranking	IaaS
1	39	19	22
2	38	20	18
3	40	21	14
4	28	22	50
5	27	23	54
6	35	24	70
7	34	25	58
8	36	26	66
9	31	27	47
10	26	28	46
11	30	29	62
12	32	30	9
13	24	31	5
14	20	32	1
15	16	33	42
16	23	34	43
17	19	35	44
18	15	36	48

A Tabela 44 descreve as características das três primeiras IaaS e da última posição (36^a). A IaaS (`39`) foi a opção selecionada para a primeira posição. Essa solução possui a seguinte configuração: dois nós e duas VMs com redundância *coldstandby*, e um serviço de reparo `ouro`. A segunda opção no *ranking* foi a IaaS (`38`), que apresenta uma redução de $0,062\%$ na confiabilidade e $0,058\%$ (R$ 0,19) no custo em relação à primeira opção. A terceira IaaS (`40`) possui uma redução de $0,074\%$ na confiabilidade e $0,026\%$ no custo em relação à primeira IaaS no *ranking*. A última IaaS (36^a) no *ranking* apresenta uma diminuição de $2,416\%$ na confiabilidade e um aumento de $44,858\%$ no custo levando em conta também a primeira colocada, a IaaS (`39`). Vale salientar que apesar da redução no custo das IaaS apresentadas na 2^a e 3^a posição no *ranking*, estas também possuem redução no parâmetro de confiabilidade. Apesar das IaaS apresentarem a mesma quantidade de componentes redundantes, o diferencial está no tipo de redundância e reparo.

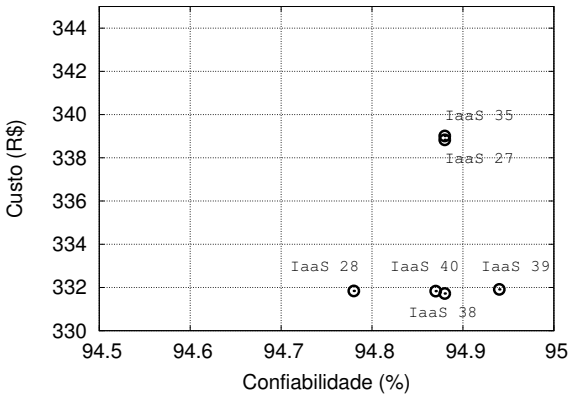
Tabela 44 – Sumário dos componentes utilizados para ranquear as IaaS (Segundo Caso).

Ranking	IaaS	Nó	VM	RS	RT
1	39	2	2	Ouro	*Cold*
2	38	2	2	Ouro	*Hot*
3	40	2	2	Ouro	*Warm*
...	...	...	...	...	...
36	48	2	2	Bronze	*Warm*

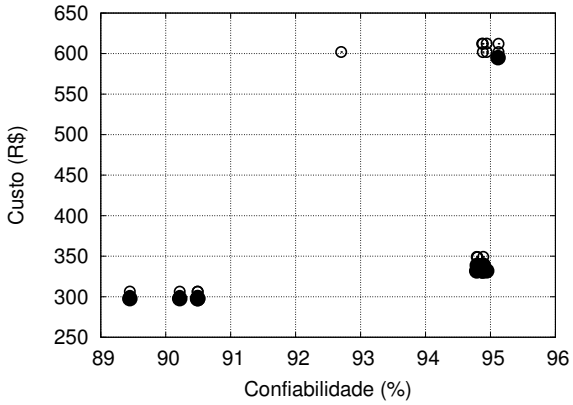
A Figura 65a ilustra o comportamento das IaaS apresentadas na Tabela 43. Os eixos representam os parâmetros confiabilidade e custo considerando a moeda real (R$). Na Figura 65b, aproximamos o conjunto com as seis primeiras IaaS no *ranking*. A função multicritério deste caso buscou maximizar confiabilidade e minimizar custo. Essa análise considerou a medida de similaridade euclidiana. Para auxiliar no entendimento, as IaaS aproximadas da Fig. 65b estão localizadas no lado inferior direito da Figura 65a. Já a Figura 65c apresenta a curva de Pareto com 12 soluções (IaaS). A curva de pareto é composta das seguintes IaaS: 9, 22, 23, 24, 48, 28, 34, 40, 38, 27, 39 e 50. A partir desse conjunto de soluções, os analistas e usuários podem identificar e escolher uma dada solução ao longo da curva.



(a) Visão Geral das 36 IaaS.



(b) *Ranking* das IaaS (Euclidiana).



(c) Conjunto das IaaS (Pareto).

Figura 65 – Conjunto das IaaS com maior confiabilidade e menor custo.

6.4.4 Ranking (III)

O terceiro caso realizou um *ranking* das IaaS levando em conta uma situação em que um dado usuário deseja encontrar uma IaaS na nuvem que apresente maior disponibilidade e confiabilidade. Nesse contexto, a função multicritério busca atender à função para maximizar o primeiro e segundo objetivo. A partir da seleção dos parâmetros disponibilidade e confiabilidade de cada IaaS, utilizou-se o modelo de decisão. A distância euclidiana

foi escolhida como medida de similaridade para o agrupamento das IaaS. A Tabela 45 apresenta os resultados com o *ranking* das trinta e seis (36) soluções.

Tabela 45 – Ranking das IaaS (maximizar disponibilidade e confiabilidade).

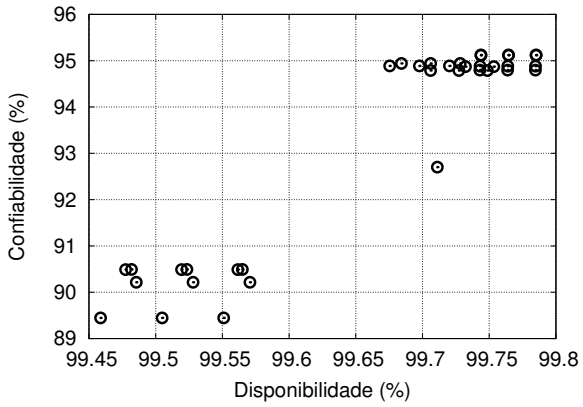
Ranking	IaaS	Ranking	IaaS
1	62	19	43
2	50	20	36
3	38	21	31
4	26	22	47
5	66	23	35
6	54	24	48
7	42	25	16
8	30	26	15
9	40	27	14
10	28	28	20
11	70	29	19
12	58	30	18
13	46	31	1
14	34	32	24
15	44	33	23
16	39	34	22
17	32	35	5
18	27	36	9

Na Tabela 46, são apresentadas as características das três primeiras colocadas e da última IaaS. A IaaS (`62`) foi a opção selecionada para a primeira posição no *ranking*. Essa solução possui a seguinte configuração: três nós e duas VMs com redundância *hots-tandby*, e um serviço de reparo `ouro`. A segunda opção no *ranking* foi a IaaS (`50`), que apresentou uma redução de $0,0000003\%$ na disponibilidade e uma redução de $0,0075\%$ na confiabilidade em relação à primeira colocada, IaaS (`62`). A terceira colocada, a IaaS (`38`) apresentou uma redução de $0,000461\%$ na disponibilidade e uma redução de $0,25454\%$ na confiabilidade em relação também à primeira colocada no *ranking*. A última no *ranking*, a IaaS (9) possui uma redução de $0,32823\%$ na disponibilidade e uma redução de $6,35\%$ na confiabilidade considerando também a primeira colocada.

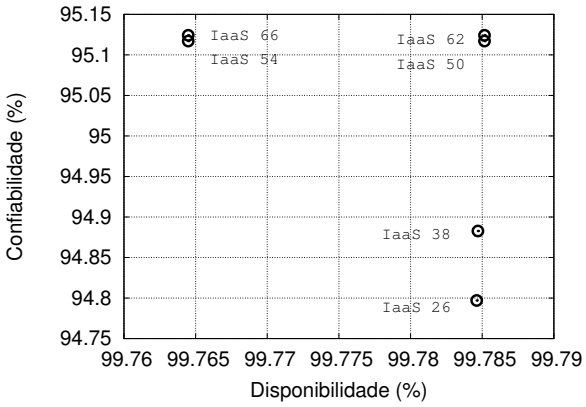
Tabela 46 – Sumário dos componentes utilizandos para ranquear as IaaS (Terceiro Caso).

Ranking	IaaS	Nó	VM	RS	RT
1	62	3	2	Ouro	*Hot*
2	50	3	1	Ouro	*Hot*
3	38	2	2	Ouro	*Hot*
...	...	...	...	...	...
36	9	1	1	Bronze	N/R

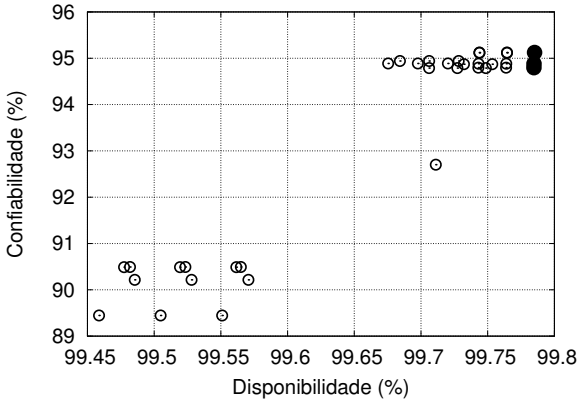
A Figura 66a ilustra uma visão geral das IaaS apresentadas na Tabela 45. Os eixos representam os parâmetros disponibilidade e confiabilidade. Na Figura 66b, apresenta-se um conjunto com as primeiras colocadas no *ranking*. Essa análise permite que analistas e usuários de infraestruturas como serviço de computação em nuvem possam comparar e identificar a similaridade entre solução. Pode-se notar que as infraestruturas que estão no *ranking* nas primeiras seis posições, encontram-se no lado superior direito da Figura 66b devido à função maximizar-maximizar. Na Figura 66c, são apresentadas as IaaS que compreendem o conceito de dominância de Pareto. O modelo identificou as seguintes IaaS: `26`, `38` e `62`. A partir desse conjunto de soluções, analistas e usuários podem identificar e escolher uma dada opção.



(a) Visão Geral das 36 IaaS.



(b) *Ranking* das IaaS (Euclidiana).



(c) Conjunto das IaaS (Pareto).

Figura 66 – Conjunto das IaaS com maior disponibilidade e confiabilidade.

Nesse sentido, nota-se que o modelo de decisão proposto pode ser adotado em diversas possibilidades de minimização e/ou maximização dos objetivos, modificando-se os parâmetros que se deseja priorizar na ordenação de um conjunto de soluções. Dessa forma, o modelo de decisão pode auxiliar os gestores e analistas de ambientes em nuvem no processo de tomada de decisão, identificando a partir de parâmetros de dependabilidade e custo quais as melhores soluções ranqueadas.

6.4.5 Ranking (IV)

O quarto caso proposto realizou uma busca no conjunto das 36 (trinta e seis) IaaS avaliadas, considerando o método de intervalo mínimo e máximo para os critérios de disponibilidade e custo. Através do método do intervalo, os gestores e analistas podem definir valores que compreendam uma faixa de acordo com as necessidades especificadas. O método permite definir o intervalo para os dois critérios. Para a ordenação, consideraram-se que os valores mínimo e máximo para a métrica de custo das IaaS foram: R$ 300 e R$ 320, bem como, 99% e 99,9% para disponibilidade. Após identificar as IaaS no conjunto de soluções avaliadas, o método realiza o *ranking* com o conjunto de soluções. Nesse sentindo, foram utilizadas as funções maximizar e minimizar e a distância euclidiana como medida de similaridade para ordenação.

A seguir, a Tabela 47 apresenta os resultados com o *ranking* das primeiras oito (8) IaaS considerando o método de intervalo mínimo e máximo. As IaaS (`18`) e (`20`) apresentam custos aproximados, contudo, quando leva-se em conta a disponibilidade do serviço por um mês, temos a indisponibilidade com os seguintes valores em horas: $3,45$ hr e $3,48$ hr, respectivamente. Pode-se afirmar, pois, que a IaaS (`20`) possui uma indisponibilidade maior em $2,04$ minutos, o que pode ser considerado um fator crítico a depender do serviço a ser hospedado em tal infraestrutura. Vale salientar, sobre a importância em identificar as variáveis a serem priorizadas em um dado estudo, já que, analisando a solução da 1^a posição com a da 8^a, podemos verificar que o custo da IaaS (`1`) é 1,71% maior em relação à IaaS (`18`). E a indisponibilidade da IaaS (`1`) é menor em aproximadamente $10,04$ minutos comparado à IaaS (`18`) na primeira posição. Na ordenação, consideram-se pesos iguais para os dois critérios, mas caso haja necessidade, a abordagem aqui exposta permite o uso de pesos para priorizar cada objetivo.

Tabela 47 – Ranking das IaaS com intervalo mínimo e máximo.

Ranking	IaaS	Custo (R$)	Disponibilidade (%)
1	18	301,090	99,5280378157
2	20	301,093	99,5233822554
3	19	301,106	99,5194114138
4	14	306,292	99,5706543903
5	5	301,106	99,5048180576
6	16	306,299	99,5650001940
7	15	306,302	99,5616066096
8	1	306,315	99,5509574705

A Tabela 48 apresenta as configurações das três primeiras soluções (IaaS) e da última colocada, considerando o método para tomada de decisão com intervalo mínimo e máximo e medida de similaridade com a distância euclidiana. Pode-se perceber que as três primeiras colocadas possuem a mesma quantidade de componentes e tipo de serviço de reparo, no entanto, todas diferenciam-se no tipo de redundância.

Tabela 48 – Sumário dos componentes utilizados para ranquear as IaaS (Quinto Caso).

Ranking	IaaS	Nó	VM	RS	RT
1	18	1	2	Prata	*Hot*
2	20	1	2	Prata	*Warm*
3	19	1	2	Prata	*Cold*
...	...	...	...	...	...
8	1	1	1	Ouro	N/R

6.5 PLANEJAMENTO E AVALIAÇÃO DE IAAS COM RENDUNDÂNCIA A/A

Esta seção apresenta a aplicação da abordagem proposta considerando dois estudos: o primeiro estudo aborda o planejamento de avaliação de uma infraestrutura como serviço para tomada de decisão adotando como critérios maior disponibilidade orientada à capacidade e menor custo; já o segundo, leva em conta como critérios maior disponibilidade orientada à capacidade, menor indisponibilidade e maior confiabilidade para a escolha de uma determinada IaaS. Em ambos os estudos, utilizamos a redundância ativo-ativo (A/A) para representar a infraestrutura como serviço.

6.5.1 Ranking (I)

Este estudo considera como premissa uma situação em que um dado usuário de infraestruturas como serviço deseja escolher uma IaaS na nuvem com maior disponibilidade orientada à capacidade (COA) e menor custo. A Tabela 49 ilustra os fatores e níveis utilizados no plano de avaliação, assim como, a Tabela 50 apresenta o projeto de experimentos. Para contemplar esse caso, adotou-se o modelo de redundância ativo-ativo (A/A) com o intuito de elevar os níveis de disponibilidade de cada cenário.

Tabela 49 – Fatores e Níveis.

Fatores	Níveis				
1. Nó	2	4	8	16	-
2. VMs por nó	2	4	8	16	32
3. Serviço de Reparo (RS)	Ouro	Prata	Bronze	-	-
4. Tipo de Redundância (RT)	A/A	-	-	-	-

Tabela 50 – Planejamento de experimentos.

IaaS	Nó	VM	RS	RT	IaaS	Nó	VM	RS	RT
1	2	4	Ouro	A/A	31	8	16	Ouro	A/A
2	2	4	Prata	A/A	32	8	16	Prata	A/A
3	2	4	Bronze	A/A	33	8	16	Bronze	A/A
4	2	8	Ouro	A/A	34	8	32	Ouro	A/A
5	2	8	Prata	A/A	35	8	32	Prata	A/A
6	2	8	Bronze	A/A	36	8	32	Bronze	A/A
7	2	16	Ouro	A/A	37	8	64	Ouro	A/A
8	2	16	Prata	A/A	38	8	64	Prata	A/A
9	2	16	Bronze	A/A	39	8	64	Bronze	A/A
10	2	32	Ouro	A/A	40	8	128	Ouro	A/A
11	2	32	Prata	A/A	41	8	128	Prata	A/A
12	2	32	Bronze	A/A	42	8	128	Bronze	A/A
13	2	64	Ouro	A/A	43	8	256	Ouro	A/A
14	2	64	Prata	A/A	44	8	256	Prata	A/A
15	2	64	Bronze	A/A	45	8	256	Bronze	A/A
16	4	8	Ouro	A/A	46	16	32	Ouro	A/A
17	4	8	Prata	A/A	47	16	32	Prata	A/A
18	4	8	Bronze	A/A	48	16	32	Bronze	A/A
19	4	16	Ouro	A/A	49	16	64	Ouro	A/A
20	4	16	Prata	A/A	50	16	64	Prata	A/A
21	4	16	Bronze	A/A	51	16	64	Bronze	A/A
22	4	32	Ouro	A/A	52	16	128	Ouro	A/A
23	4	32	Prata	A/A	53	16	128	Prata	A/A
24	4	32	Bronze	A/A	54	16	128	Bronze	A/A
25	4	64	Ouro	A/A	55	16	256	Ouro	A/A
26	4	64	Prata	A/A	56	16	256	Prata	A/A
27	4	64	Bronze	A/A	57	16	256	Bronze	A/A
28	4	128	Ouro	A/A	58	16	512	Ouro	A/A
29	4	128	Prata	A/A	59	16	512	Prata	A/A
30	4	128	Bronze	A/A	60	16	512	Bronze	A/A

A Figura 67 ilustra o modelo SPN adotado para representar a redundância ativo/ativo definido no planejamento de avaliação. Os parâmetros NVM e NND representam respectivamente: quantidade de máquinas virtuais e nós. Os valores utilizados por esses parâmetros são apresentados no plano de experimento na Tabela 50. Os tempos de defeito e reparo conforme a Tabela 51 são manipulados através das transições temporizadas TF_VM, TF_ND, TR_VM, TR_ND. Os valores dos tempos são baseado em (ARAUJO et al., 2014), (DANTAS et al., 2015).

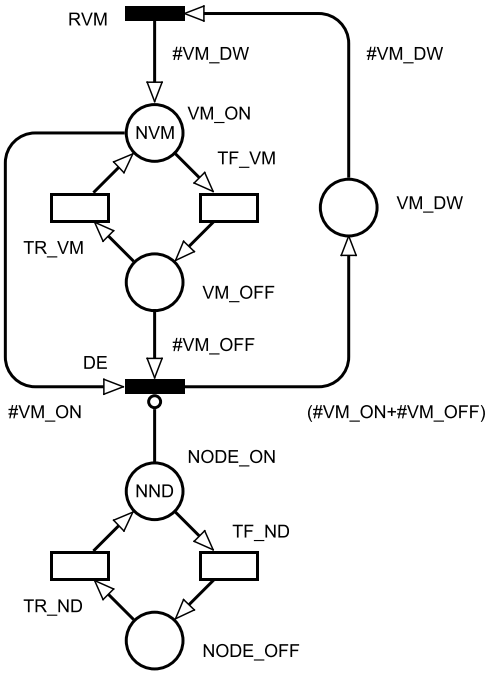


Figura 67 – Modelo SPN para a redundância ativo/ativo.
Baseado em: (MELO et al., 2017; ARAUJO et al., 2018).

Tabela 51 – Valores do modelo SPN com redundância A/A.

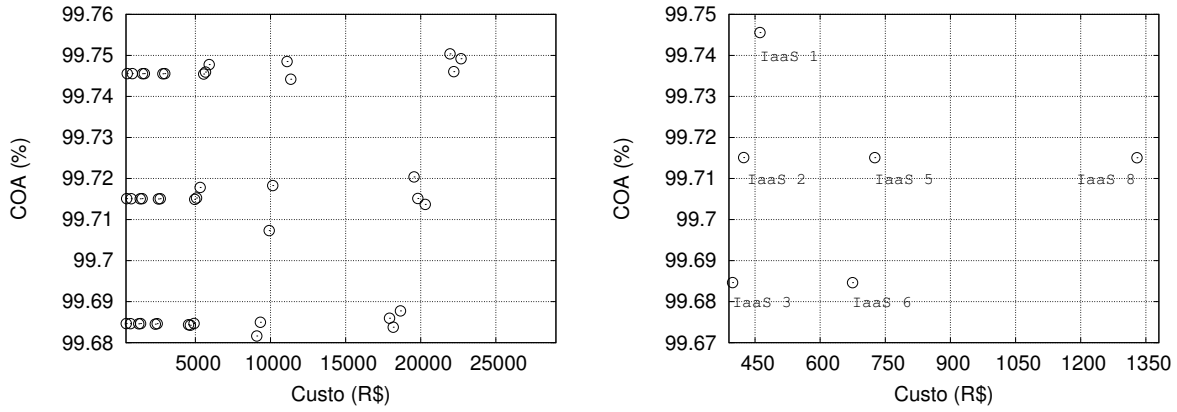
	MTTF (Hr)	MTTR (Hr)		
		Ouro	Prata	Bronze
Nó	481,46565	1,03660458	1,13660458	1,23660458
VM	2028,88290	0,40150948	0,50150948	0,60150948

A Tabela 52 apresenta as configurações ranqueadas ao considerar os critérios definidos. A IaaS 3 foi a infraestrutura que apresentou melhor classificação e possui os seguintes componentes: 2 (dois) nós e 4 (quatro) VMs com redundância ativa-ativa e o tipo de reparo bronze. A segunda IaaS no *ranking* foi a configuração 2, que mostra um acréscimo de $6,44\%$ no custo e um aumento de $0,03\%$ na COA quando comparada à configuração melhor classificada. A terceira melhor configuração (IaaS 1) teve um acréscimo na COA de $0,06\%$, no entanto apresentou um acréscimo no custo de $15,85\%$, quando comparado à configuração melhor classificada. A útima colocada no *ranking* foi a IaaS 7, que mostra um aumento no custo de $271,43\%$, quando comparada à primeira colocada.

Tabela 52 – Ranking das IaaS para os critérios definidos (isto é, maximizar o COA e minimizar o custo).

Ranking	IaaS	Ranking	IaaS	Ranking	IaaS	Ranking	IaaS
1	3	16	18	31	33	46	48
2	2	17	19	32	34	47	49
3	1	18	20	33	35	48	50
4	6	19	21	34	36	49	51
5	5	20	22	35	37	50	52
6	8	21	23	36	38	51	53
7	9	22	24	37	39	52	54
8	10	23	25	38	40	53	55
9	11	24	26	39	41	54	56
10	12	25	27	40	42	55	57
11	13	26	28	41	43	56	58
12	14	27	29	42	44	57	59
13	15	28	30	43	45	58	60
14	16	29	31	44	46	59	4
15	17	30	32	45	47	60	7

A Figura 68 resume o ranqueamento de configurações geradas para o o presente estudo. A Figura 68a apresenta uma visão geral de todas as IaaS descritas na Tabela 52, enquanto a Figura 68b apresenta o conjunto das `6` (seis) primeiras colocadas no *ranking*. O eixo `y` indica o custo da infraestrutura na nuvem e o eixo `x` representa a disponibilidade orientada à capacidade (COA). Os critérios definidos para minimizar o custo e maximizar a COA, portanto, significam que o conjunto ideal de configurações é mostrado por pontos plotados no lado superior esquerdo da Figura 68a. A partir da Figura 68b, é possível identificar a posição da configuração `3`, que é a melhor ranqueada.



(a) Visão Geral das 60 IaaS.

(b) *Ranking* das IaaS (Euclidiana).

Figura 68 – Conjunto das IaaS com maior COA e menor custo.

Por fim, a Tabela 53 apresenta as configurações das três primeiras soluções (IaaS) e da última colocada considerando o método de agrupamento de dados por meio da medida de similaridade com a distância euclidiana. Percebe-se que as três primeiras colocadas possuem a mesma quantidade de componentes e tipo de serviço de reparo, porém todas se diferenciam quanto ao serviço de reparo.

Tabela 53 – Sumário dos componentes utilizandos para ranquear as IaaS.

Ranking	IaaS	Nó	VM	RS	RT
1	3	2	4	Bronze	A/A
2	2	2	4	Prata	A/A
3	1	2	4	Ouro	A/A
...	...	...	...	...	...
60	7	2	16	Ouro	A/A

6.5.2 Ranking (II)

Esta subseção descreve o processo de planejamento de avaliação considerando como critérios: maior disponibilidade orientada à capacidade, menor indisponibilidade e maior confiabilidade para escolher uma dada infraestrutura como serviço na nuvem. Para representar as infraestruturas definimos os fatores (exemplo, nó) e níveis (como 10 nós) de cada cenário no plano experimental. Essa definição forneceu uma compreensão da relação entre fatores e variáveis de resposta (por exemplo, disponibilidade) de cada cenário.

A tabela 54 apresenta os fatores e os níveis usados para gerar os cenários (IaaS). O método usado para criar o plano foi o fatorial completo, pois o objetivo era obter todas as combinações das IaaS. Assim, os cinco fatores foram combinados com seus respectivos níveis, representando um conjunto de cenários. O fator `Nós` e `VMs` representaram a quantidade adotada para cada configuração. O fator falha do componente (FC) representa a adoção de diferentes tempos de defeito para os Nós e VMs (nomeadas como `fa1`, `fb1` e`fc1`). O serviço de reparo determinou os diferentes tempos de reparo (nomeados como `rs1`, `rs2` e `rs3`). O quinto fator o tipo de redundância ativo-ativo (RT) no Nó e nas VMs.

Tabela 54 – Fatores e Níveis.

Fatores	Níveis										
1. Nós	2	3	4	5	6	7	8	9	10	20	30
2. VMs	20	30	40	50	60	70	80	90	100	200	300
3. Falha do Componente (FC)	fa1	fb1	fc1	-	-	-	-	-	-	-	-
4. Serviço de Reparo (RS)	rs1	rs2	rs3	-	-	-	-	-	-	-	-
5. Tipo de Redundância (RT)	A/A	-	-	-	-	-	-	-	-	-	-

A partir do plano fatorial completo foi possível gerar um conjunto com `1090` cenários. No entanto, consideramos que cada nó físico poderia hospedar apenas `10` VMs. Assim, o plano de experimentos foi reduzido a `99` cenários, como pode ser visto na Tabela 55.

Tabela 55 – Planejamento de experimentos.

IaaS	Node	VM	FC	TS	RT	IaaS	Node	VM	FC	TS	RT	IaaS	Node	VM	FC	TS	RT
1	2	20	fa1	rs1	A/A	34	2	20	fb1	rs1	A/A	67	2	20	fc1	rs1	A/A
2	2	20	fa1	rs2	A/A	35	2	20	fb1	rs2	A/A	68	2	20	fc1	rs2	A/A
3	2	20	fa1	rs3	A/A	36	2	20	fb1	rs3	A/A	69	2	20	fc1	rs3	A/A
3	3	30	fa1	rs1	A/A	37	3	30	fb1	rs1	A/A	70	3	30	fc1	rs1	A/A
5	3	30	fa1	rs2	A/A	38	3	30	fb1	rs2	A/A	71	3	30	fc1	rs2	A/A
6	3	30	fa1	rs3	A/A	39	3	30	fb1	rs3	A/A	72	3	30	fc1	rs3	A/A
7	4	40	fa1	rs1	A/A	40	4	40	fb1	rs1	A/A	73	4	40	fc1	rs1	A/A
8	4	40	fa1	rs2	A/A	41	4	40	fb1	rs2	A/A	74	4	40	fc1	rs2	A/A
9	4	40	fa1	rs3	A/A	42	4	40	fb1	rs3	A/A	75	4	40	fc1	rs3	A/A
10	5	50	fa1	rs1	A/A	43	5	50	fb1	rs1	A/A	76	5	50	fc1	rs1	A/A
11	5	50	fa1	rs2	A/A	44	5	50	fb1	rs2	A/A	77	5	50	fc1	rs2	A/A
12	5	50	fa1	rs3	A/A	45	5	50	fb1	rs3	A/A	78	5	50	fc1	rs3	A/A
13	6	60	fa1	rs1	A/A	46	6	60	fb1	rs1	A/A	79	6	60	fc1	rs1	A/A
14	6	60	fa1	rs2	A/A	47	6	60	fb1	rs2	A/A	80	6	60	fc1	rs2	A/A
15	6	60	fa1	rs3	A/A	48	6	60	fb1	rs3	A/A	81	6	60	fc1	rs3	A/A
16	7	70	fa1	rs1	A/A	49	7	70	fb1	rs1	A/A	82	7	70	fc1	rs1	A/A
17	7	70	fa1	rs2	A/A	50	7	70	fb1	rs2	A/A	83	7	70	fc1	rs2	A/A
18	7	70	fa1	rs3	A/A	51	7	70	fb1	rs3	A/A	84	7	70	fc1	rs3	A/A
19	8	80	fa1	rs1	A/A	52	8	80	fb1	rs1	A/A	85	8	80	fc1	rs1	A/A
20	8	80	fa1	rs2	A/A	53	8	80	fb1	rs2	A/A	86	8	80	fc1	rs2	A/A
21	8	80	fa1	rs3	A/A	54	8	80	fb1	rs3	A/A	87	8	80	fc1	rs3	A/A
22	9	90	fa1	rs1	A/A	55	9	90	fb1	rs1	A/A	88	9	90	fc1	rs1	A/A
23	9	90	fa1	rs2	A/A	56	9	90	fb1	rs2	A/A	89	9	90	fc1	rs2	A/A
24	9	90	fa1	rs3	A/A	57	9	90	fb1	rs3	A/A	90	9	90	fc1	rs3	A/A
25	10	100	fa1	rs1	A/A	58	10	100	fb1	rs1	A/A	91	10	100	fc1	rs1	A/A
26	10	100	fa1	rs2	A/A	59	10	100	fb1	rs2	A/A	92	10	100	fc1	rs2	A/A
27	10	100	fa1	rs3	A/A	60	10	100	fb1	rs3	A/A	93	10	100	fc1	rs3	A/A
28	20	200	fa1	rs1	A/A	61	20	200	fb1	rs1	A/A	94	20	200	fc1	rs1	A/A
29	20	200	fa1	rs2	A/A	62	20	200	fb1	rs2	A/A	95	20	200	fc1	rs2	A/A
30	20	200	fa1	rs3	A/A	63	20	200	fb1	rs3	A/A	96	20	200	fc1	rs3	A/A
31	30	200	fa1	rs1	A/A	64	30	300	fb1	rs1	A/A	97	30	300	fc1	rs1	A/A
32	30	200	fa1	rs2	A/A	65	30	300	fb1	rs2	A/A	98	30	300	fc1	rs2	A/A
33	30	200	fa1	rs3	A/A	66	30	300	fb1	rs3	A/A	99	30	300	fc1	rs3	A/A

As Tabelas 56 e 57 apresentam respectivamente o tempo médio de defeito (MTTF) e o tempo médio de reparo (MTTR) usado para o modelo de disponibilidade. Os valores foram obtidos em (ARAUJO et al., 2014; ARAUJO et al., 2018)

Tabela 56 – MTTF para o modelo IaaS.

Componente	fa1(Hr)	fb1(Hr)	fc1(Hr)
Nó	481.46565	601.83206	722.19847
VM	2028.88290	2536.10362	3043.32435

Tabela 57 – MTTR para o modelo IaaS.

Componente	rs1 (Hr)	rs2(Hr)	rs3(Hr)
Nó	1.03660458	1.13660458	1.23660458
VM	0.401509480	0.50150948	0.60150948

A Tabela 58 apresenta todos os cenários ranqueados, assim como, na Tabela 59 estão detalhados os componentes dos primeiros cenários ranqueados e da última posição. O Cenário `97` é apontado como a configuração melhor ranqueada e apresenta os seguintes componentes: `30` nós e `300` VMs com redundância ativa-ativa, tempo de defeito (`fc1`) e tempo de reparo (`rs1`), respectivamente. O cenário `98` foi raqueada na segunda colocação, esta apresentou uma redução de `0.33%` na disponibilidade orientada à capacidade considerando o cenário na primeira posição. Além disso, a segunda colocada apresentou um aumento de `24.91%` no tempo de indisponibilidade, o que representou um valor adicional de `17,8` minutos quando comparado com o primeiro cenário. Já o cenário `15` ficou na última posição do *ranking*, apresentando um aumento no tempo de indisponibilidade em 124.71% quando comparado com configuração ranqueada na primeira posição.

Tabela 58 – Ranking considerando os critérios (maximizar COA e confiabilidade e minimizar indisponibilidade).

Rank	Cen.	Rank	Cen.	Rank	Cen.	Rank	Cen.
1	97	26	81	51	43	76	31
2	98	27	74	52	53	77	32
3	94	28	78	53	46	78	28
4	99	29	71	54	35	79	33
5	95	30	75	55	57	80	29
6	96	31	72	56	1	81	30
7	91	32	34	57	41	82	3
8	88	33	68	58	50	83	25
9	85	34	37	59	54	84	22
10	92	35	40	60	47	85	19
11	89	36	38	61	51	86	26
12	82	37	64	62	44	87	13
13	70	38	65	63	48	88	11
14	93	39	66	64	45	89	16
15	86	40	61	65	42	90	23
16	79	41	62	66	39	91	27
17	90	42	63	67	4	92	20
18	83	43	58	68	36	93	24
19	73	44	55	69	2	94	17
20	76	45	52	70	5	95	14
21	87	46	59	71	7	96	21
22	80	47	56	72	6	97	18
23	84	48	49	73	8	98	12
24	67	49	69	74	9	99	15
25	77	50	60	75	10	-	-

Tabela 59 – Resumo dos componentes usados nos cenários ranqueados.

Ranking	IaaS	Nó	VM	FC	TS	RT
1	97	30	300	*fc1*	*rs1*	A/A
2	98	30	300	*fc1*	*rs2*	A/A
3	94	20	200	*fc1*	*rs1*	A/A
4	99	30	300	*fc1*	*rs3*	A/A
5	95	20	200	*fc1*	*rs2*	A/A
6	96	20	200	*fc1*	*rs3*	A/A
7	91	10	100	*fc1*	*rs1*	A/A
8	88	9	90	*fc1*	*rs1*	A/A
9	85	8	80	*fc1*	*rs1*	A/A
10	92	10	100	*fc1*	*rs3*	A/A
...	...	...	...	...	...	...
99	15	6	60	*fa1*	*rs3*	A/A

No Apêndice C são apresentados os valores das métricas dos cenários avaliados nesse estudo.

6.6 AVALIAÇÃO DE PERFORMABILIDADE

Esta seção apresenta um estudo sobre a avaliação de performabilidade de uma aplicação hospedada em uma dada infraestrutra como serviço na nuvem. A avaliação permite compreender como o ambiente é impactado a partir da da ocorrência de defeitos na IaaS. Para a avaliação de performabilidade utilizamos a abordagem de modelagem hierárquica, ou seja, combinando os resultados do modelo de dependabilidade da IaaS, com os resultados do modelo de desempenho da aplicação. A integração da modelagem de aspectos de desempenho e dependabilidade de ambientes computacionais é denominada de modelagem de performabilidade.

A Figura 69 ilustra o modelo de desempenho SPN adotado para avaliar o comportamento de desempenho de uma dada aplicação. A partir do modelo de desempenho foi possível analisar as métricas de utilização dos recursos computacionais como processador e recurso de armazenamento da VM).

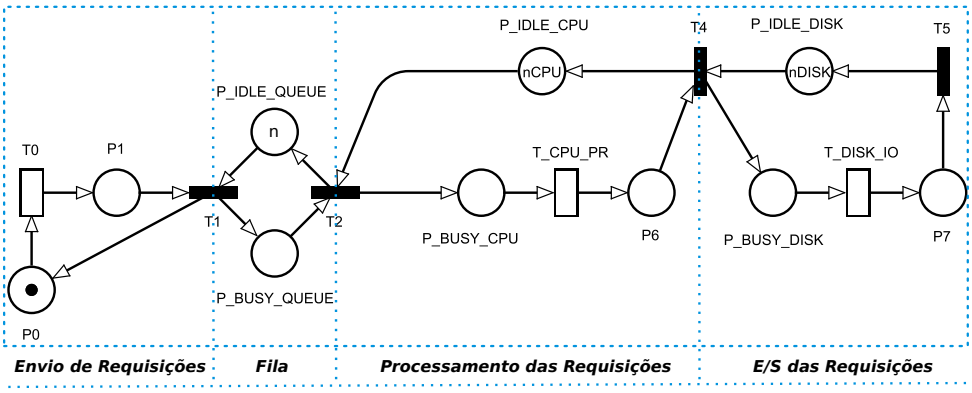


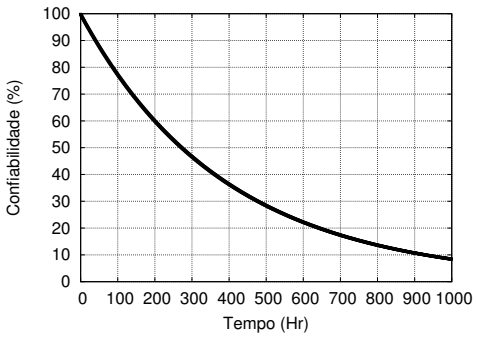
Figura 69 – Modelo de Desempenho SPN.

A Tabela 60 apresenta os parâmetros utilizados para a avaliação do modelo de desempenho. Os valores utilizados na modelagem foram estimados para representar um serviço na nuvem que recebe solitações de requisições para serem processadas e lidas/escritas em um recurso de armazenamento.

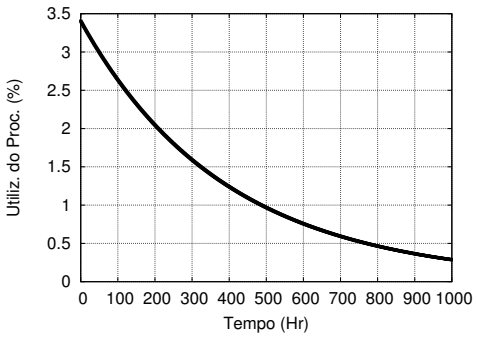
Tabela 60 – Parâmetros de Desempenho.

Parâmetro	Valor
n	50
nCPU	1
nDISK	1
Tempo de envio (seg)	0,0000069444
Tempo de vCPU (seg)	0,00000028
Tempo de vDisco (seg)	0,00000278

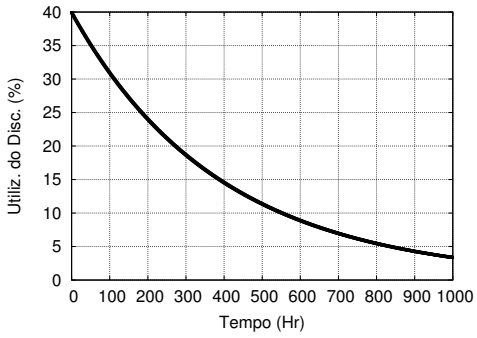
Combinamos os valores da métrica de confiabilidade da IaaS[12] com os valores de desempenho do modelo de desempenho SPN. A seguir, a Figura 70 apresenta os resultados de performabilidade considernado a ocorrência de defeitos na IaaS. A Figura 70a apresenta o resultado da degradação da confiabilidade, assim como, as Figuras 70b e 70c ilustram a degradação da métrica de utilização do recurso de processamento e armazenamento da máquina virtual após a ocorrência de um defeito.



(a) Confiabilidade da IaaS (3).



(b) Degração na CPU da VM.



(c) Degração no recurso de E/S da VM.

Figura 70 – Análise de Performabilidade dos recursos computacionais na IaaS.

[12] Cenário ranqueado na primeira posição apresentado no estudo da subseção 6.5.1.

A performabilidade avalia a degradação do desempenho de ambientes computacionais provocado pela ocorrência de defeitos. A avaliação de resultados de performabilidade possibilitam que o usuário possa empregar medidas que reduzam o impacto dos níveis de degradação do serviço hospedado na IaaS. Assim, adotando mecanismos que previnam a presença de defeitos, sejam com composição de redundância nos componentes da IaaS ou a partir de atividades de reparos em um determinado período de tempo.

As métricas de performabilidade são calculadas a partir do modelo de desempenho e posteriormente combinadas com o modelo dependabilidade para ilustrar o efeito da disponibilidade no desempenho da infraestrutura. É possível diversificar os resultados de performabilidade, pois o usuário pode combinar as métricas de desempenho com as métricas de dependabilidade. O modelo de performabilidade permite a avaliação do impacto da ocorrência de eventos de falhas e atividades de reparo no desempenho da infraestrutura em nuvem por meio de métricas de performabilidade (por exemplo, utilização do processador, utilização do dispositivo de entrada/saída).

6.7 CONSIDERAÇÕES FINAIS

Este capítulo apresentou os resultados alcançados a partir da metodologia proposta. Inicialmente, foi apresentado o planejamento de experimentos para geração das configurações (IaaS). Em seguida, a estratégia de modelagem das configurações e a avaliação dos modelos de dependabilidade e custo. Ademais, cinco estudos foram avaliados, considerando os métodos adotados para realizar o *ranking* das IaaS no processo de tomada de decisão. O primeiro caso considerou o *ranking* do conjunto de soluções avaliadas visando minimizar o custo e a indisponibilidade. No segundo caso, utilizou-se como critério para a ordenação soluções com maior confiabilidade e menor custo. Para o terceiro caso, para a ordenação das IaaS, levou-se em conta minimizar o custo e maximizar a disponibilidade das soluções. No quarto caso, o estudo garantiu a ordenação das IaaS, considerando a maior disponibilidade e confiabilidade. Os quatro primeiros estudos de casos foram apresentados com dois métodos de ordenação: medida de similaridade e dominância de pareto. O quinto estudo de caso apresentou os resultados do conjunto de soluções, baseando-se no método de intervalo mínimo e máximo dos parâmetros custo e disponibilidade. Por fim, o sexto estudo de caso apresentou um *ranking* gerado a partir de um conjunto de infraestruturas avaliadas considerando a capacidade orientada à disponibilidade e o custo, e o sétimo estudo de caso apresentou um estudo considerando a degradação de desempenho de uma IaaS por meio da análise de performabilidade. Portanto, através dos estudos descritos, foi possível demonstrar que a metodologia adotada considerando os modelos de dependabilidade, performabilidade, custo e decisão pode auxiliar gestores e analistas no processo de tomada de decisão para escolher um conjunto de soluções.

7 CONCLUSÕES E TRABALHOS FUTUROS

A Computação em nuvem tem proporcionado através dos modelos de serviços (IaaS, PaaS e SaaS) uma série de modificações em diversas áreas, como por exemplo, na computação distribuída, virtualização, *big data*, mídias sociais, inteligência artificial, segurança e comunicação etc. Nesse cenário, o mercado tem exigido cada vez mais serviços com alta disponibilidade e baixos custos. Dessa forma, a modelagem e análise de mecanismos de redundância em infraestruturas como serviço em nuvem são importantes para garantir a alta disponibilidade e continuidade do funcionamento dos serviços.

Um dos desafios que cresce a cada momento em ambientes de computação em nuvem privada, são a representação e a análise de parâmetros de dependabilidade aliadas aos custos das infraestruturas como serviço. A modelagem e análise podem auxiliar os analistas e tomadores de decisão no planejamento das IaaS. Mecanismos de redundância e políticas de reparo têm sido propostos para reduzir possíveis danos causados na ocorrência de falhas. Os modelos baseados em redes de Petri estocástica (SPN) e diagrama de bloco de confiabilidade (RBD) são bastante utilizados para a análise e compreensão do comportamento da dependabilidade de infraestruturas computacionais em nuvem, quando associados a modelos de decisão, podendo auxiliar no processo de tomada de decisão.

Grande parte das atividades de planejamento das IaaS consiste na análise e tomada de decisões. Decisões essas que visam escolher uma ou mais soluções que atendam às expectativas e aos interesses do usuário que deseja utilizar um ambiente como serviço. Os trabalhos apresentados, considerando os aspectos abordados nesta tese, como modelagem, avaliação de dependabilidade, custos e modelo de decisão, acabam tratando e relacionando alguns tópicos. Esta tese propôs, então, uma abordagem baseada nas vantagens dos modelos SPN e RBD para representação de um conjunto de IaaS geradas a partir do do planejamento de avaliação. Mecanismos de redundância e políticas de reparo foram utilizados visando elevar índices de disponibilidade.

Este trabalho apresentou um conjunto de estudos de casos para a avaliação da abordagem proposta. O estudo foi baseado na configuração de uma biblioteca digital em uma infraestrutura de nuvem privada. A partir dos modelos e métodos utilizados, foram possíveis: gerar um conjunto de soluções ou configurações de IaaS e avaliar as métricas de dependabilidade (disponibilidade, disponibilidade orientada à capacidade, confiabilidade e indisponibilidade), performabilidade e custo. Já o modelo de decisão adotado recebeu um conjunto de resultados alcançados como entrada para, em seguida, realizar por meio dos métodos de agrupamento de dados, dominância de Pareto e intervalo mínimo-máximo o *ranking* das soluções.

7.1 CONTRIBUIÇÕES

Assim sendo, as contribuições desta tese estão descritas abaixo:

- **Metodologia**. a metodologia aborda uma estratégia através de um conjunto de métodos para a geração de configurações ou soluções de infraestruturas como serviço na nuvem através de modelos de dependabilidade e performabilidade. Além disso, possibilita a avaliação e análise das IaaS para tomada de decisão.
- **Modelos SPN e RBD**. Este trabalho apresenta um conjunto de modelos estocásticos e combinatoriais com o propósito de avaliar a dependabilidade e performabilidade das IaaS. A partir dos modelos, foram consideradas diversas configurações variando mecanismos de redundância como *hotstandy*, *coldstandby* e *warmstandby*.
- **Modelo de Custo**. O modelo de custo é representado por um conjunto de parâmetros, visando estimar os custos de utilização, operação e manutenção. O modelo proposto pode identificar e esclarecer os custos associados ao serviço. Ademais, os custos são associados aos modelos de dependabilidade.
- **Modelo de Decisão**. Esse modelo objetiva reduzir a complexidade do processo de escolha de um serviço em nuvem. Dado um conjunto de cenários avaliados, o modelo gera um conjunto de saída ordenado, levando em conta funções multicritérios.
- **Ferramenta**. Este trabalho apresentou uma ferramenta *open source* (MiPACE) para auxiliar no planejamento de avaliação de infraestruturas de nuvens. De forma complementar, foram implementados métodos considerando funções multicritérios para seleção e ordenação de um conjunto de soluções. O *ranking* pode ser utilizado como instrumento no auxílio de usuários que desejam tomar uma decisão no processo de escolha de uma IaaS.

Parte dos resultados descritos no estudo de caso foi publicado no *Journal of Cloud Computing: Advances Systems and Applications* (ARAUJO et al., 2018).

7.2 LIMITAÇÕES

As limitações deste trabalho são expostas na sequência.

- O trabalho ora proposto contempla um domínio restrito quanto às configurações de *hardware* utilizadas para compor uma infraestrutura como serviço: CPU, memória e disco são estáticos;
- Os estudos de casos, apresentados neste trabalho, focaram em infraestruturas como serviço na nuvem que representam instituições de pequeno e médio porte. Todavia,

existe uma crescente demanda por IaaS de grande porte compostas por milhares de nós e/ou máquinas virtuais. Esse tipo de configuração não foi considerada no trabalho proposto;

- Este trabalho limitou-se a estudar algumas métricas relacionadas à dependabilidade e performabilidade das IaaS. Outras métricas relacionadas manutenabilidade não foram levadas em consideração. Assim, como adotar um método de análise de sensibilidade para detalhar o impacto das métricas avaliadas;

- O modelo de custo foi estabelecido através do conceito de TCO. Existem outras abordagens que consideram fatores, por exemplo, tempo de contrato, composição heterogênea de componentes, tipos de equipamentos e infraestrutura compartilhada.

- Outra limitação é quanto aos modelos e técnicas utilizadas na modelagem desta proposta. O usuário deve ter conhecimento para a parametrização das variáveis no sistema. Uma forma de auxiliar o método de avaliação é integrar a ferramenta MiPACE com a ferramenta de avaliação dos modelos de dependabilidade e performabilidade.

7.3 TRABALHOS FUTUROS

Como trabalho futuro, pode-se incluir a abordagem proposta em outros tipos de nuvens, tais como, nuvens híbridas. Essas nuvens têm sido utilizadas ao longo dos anos, como umas das principais tendências do mercado. Outro aspecto a ser considerado como extensão desse trabalho, é considerar ambientes computacionais em nuvem composto por centenas ou milhares de nós e/ou máquinas virtuais. Ou ainda, levar em conta cenários de arquiteturas de microserviços baseados em tecnologia *container* (*Docker*) diversificando as arquiteturas como serviço. Aplicando modelos em redes de Petri, tal cenário, certamente, resultaria na explosão do espaço de estados dos modelos gerados. Para lidar com esse problema, possivelmente, seria necessário utilizar abordagens como a modelagem hierárquica.

Esta tese poderá também considerar diferentes políticas de reparo nas infraestruturas como serviço em nuvem. Políticas baseadas em manutenção preventiva, substituição de componentes, diferentes especialidades da equipe de manutenção e diferentes tempos médios de falha. Além disso, aliando o modelo de custos, este trabalho poderia ser ampliado, ponderando cenários que avaliem a violação de um nível de serviço contratado.

Já considerando na ferramenta proposta o planejamento de experimentos, também poderia ser levado em conta a adição de restrições para composição das configurações das IaaS, visto que o planejamento de experimento leva em consideração fatores e níveis sendo agrupados de forma fatorial sem restrição. Assim, a ferramenta poderia oferecer a adição de tais regras para composição das configurações. Ademais, a ferramenta poderia abordar

aspectos de inteligência artificial a partir dos registros dos ranqueamentos armazenados no banco de dados. Desta feita, a ferramenta poderia aprender e sugerir funções multicritérios para os cenários.

REFERÊNCIAS

ABDEL-BASSET, M.; MOHAMED, M.; CHANG, V. Nmcda: A framework for evaluating cloud computing services. *Future Generation Computer Systems*, v. 86, p. 12 – 29, 2018. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X17327814>.

AGUIRRE, L. A. *Introdução à identificação de sistemas–Técnicas lineares e não-lineares aplicadas a sistemas reais*. [S.l.]: Editora UFMG, 2004.

AHSON, S. A.; ILYAS, M. *Cloud computing and software services: theory and techniques*. [S.l.]: CRC Press, 2010.

AL-JABRI; M, I.; MUSTAFA, I.; SOHAIL; SADIQ, M. et al. A group decision-making method for selecting cloud computing service model. *International Journal of Advanced Computer Science and Applications*, v. 9, p. 449 – 456, 2018. Disponível em: <http://thesai.org/Downloads/Volume9No1/Paper_62-A_Group_Decision_Making_Method.pdf>.

ALABOOL, H.; KAMIL, A.; ARSHAD, N.; ALARABIAT, D. Cloud service evaluation method-based multi-criteria decision-making: A systematic literature review. *Journal of Systems and Software*, v. 139, p. 161 – 188, 2018. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121218300244>.

ALMEIDA, A. T. D.; CAVALCANTE, C. A. V.; ALENCAR, M. H.; FERREIRA, R. J. P.; ALMEIDA-FILHO, A. T. de; GARCEZ, T. V. et al. *Multicriteria and multiobjective models for risk, reliability and maintenance decision analysis*. [S.l.]: Springer, 2016.

Amazon. *Amazon EC2*. 2018. Acesso em: 30 Abr. 2018. Disponível em: <http://aws.amazon.com/pt/ec2/>.

ANDRADE, E.; NOGUEIRA, B.; MATOS, R.; CALLOU, G.; MACIEL, P. Availability modeling and analysis of a disaster-recovery-as-a-service solution. *Computing*, v. 99, n. 10, p. 929–954, Oct 2017. ISSN 1436-5057. Disponível em: <https://doi.org/10.1007/s00607-017-0539-8>.

ARAUJO, C. 2019. Acesso em: 28 Jan. 2019. Disponível em: <https://github.com/cjmaufpe/mipace>.

ARAÚJO, C.; MACIEL, P.; ZIMMERMANN, A.; ANDRADE, E.; SOUSA, E.; CALLOU, G.; CUNHA, P. Performability modeling of electronic funds transfer systems. *Computing*, v. 91, n. 4, p. 315–334, Apr 2011. ISSN 1436-5057. Disponível em: <https://doi.org/10.1007/s00607-010-0121-0>.

ARAUJO, J.; MACIEL, P.; ANDRADE, E.; CALLOU, G.; ALVES, V.; CUNHA, P. Decision making in cloud environments: an approach based on multiple-criteria decision analysis and stochastic models. *Journal of Cloud Computing*, v. 7, n. 1, p. 7, Mar 2018. ISSN 2192-113X. Disponível em: <https://doi.org/10.1186/s13677-018-0106-7>.

ARAUJO, J.; MACIEL, P.; TORQUATO, M.; CALLOU, G.; ANDRADE, E. Availability evaluation of digital library cloud services. In: IEEE. *Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on*. [S.l.], 2014. p. 666–671.

ARENALES, M.; ARMENTANO, V.; MORABITO, R. *Pesquisa operacional: para cursos de engenharia*. [S.l.]: Elsevier, 2007.

AVIZIENIS, A.; LAPRIE, J.-C.; RANDELL, B.; LANDWEHR, C. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, IEEE, v. 1, n. 1, p. 11–33, 2004.

BALBO, G. Introduction to stochastic petri nets. In: *Lectures on Formal Methods and PerformanceAnalysis*. [S.l.]: Springer, 2001. p. 84–155.

BAUER, E.; ADAMS, R. *Reliability and availability of cloud computing*. [S.l.]: John Wiley & Sons, 2012.

BAUER, E.; ADAMS, R.; EUSTACE, D. *Beyond redundancy: how geographic redundancy can improve service availability and reliability of computer-based systems*. [S.l.]: John Wiley & Sons, 2011.

BESTBUY. 2018. Acesso em: 10 Abr. 2018. Disponível em: <http://www.bestbuy.com/>.

BOLCH, G.; GREINER, S.; MEER, H. de; TRIVEDI, K. S. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications*. [S.l.]: John Wiley & Sons, 2006.

BRILHANTE, J.; SILVA, B.; MACIEL, P.; ZIMMERMANN, A. Dependability models for eucalyptus infrastructure clouds considering vm life-cycle. In: IEEE. *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*. [S.l.], 2014. p. 1336–1341.

CAO, Y.; SUN, H.; TRIVEDI, K. S. Performability analysis of tdma cellular systems based on composite and hierarchical markov chain models. In: *Performance and QoS of Next Generation Networking*. [S.l.]: Springer, 2001. p. 317–332.

CASTRO, L. N. d.; FERRARI, D. G. Introdução à mineração de dados: conceitos basicos, algoritmos e aplicacoes. *São Paulo: Saraiva*, 2016.

CELPE. 2018. Acesso em: 19 Maio 2018. Disponível em: <http://celpe.com.br/>.

CHANG, V. *Delivery and adoption of cloud computing services in contemporary organizations*. [S.l.]: IGI Global, 2015.

CHEN, D.; TRIVEDI, K. S. Closed-form analytical results for condition-based maintenance. *Reliability Engineering & System Safety*, Elsevier, v. 76, n. 1, p. 43–51, 2002.

CHUNG, C. A. *Simulation modeling handbook: a practical approach*. [S.l.]: CRC press, 2003.

CIARDO, G.; TRIVEDI, K. S. A decomposition approach for stochastic reward net models. *Performance Evaluation*, Elsevier, v. 18, n. 1, p. 37–59, 1993.

CLOUDSTACK, A. 2018. Acesso em: 10 Abr. 2018. Disponível em: <https://cloudstack.apache.org>.

COCIAN, L. F. E. *Manual da linguagem C*. [S.l.]: ULBRA, 2004.

COSTA, E. M. M. *Redes de petri e aplicações aos sistemas a eventos discretos*. [S.l.]: Clube de Autores, 2007.

DANTAS, J.; MATOS, R.; ARAUJO, J.; MACIEL, P. Eucalyptus-based private clouds: availability modeling and comparison to the cost of a public cloud. *Computing*, v. 97, n. 11, p. 1121–1140, 2015. ISSN 1436-5057. Disponível em: <http://dx.doi.org/10.1007/s00607-015-0447-8>.

DEB, K. *Multi-objective optimization using evolutionary algorithms*. [S.l.]: John Wiley & Sons, 2001. v. 16.

DEKA, G. C. *Handbook of Research on Securing Cloud-Based Databases with Biometric Applications*. [S.l.]: IGI Global, 2014.

DESROCHERS, A. A.; AL-JAAR, R. Y. *Applications of Petri nets in manufacturing systems: modeling, control, and performance analysis*. [S.l.]: IEEE, 1995.

DING, S.; WANG, Z.; WU, D.; OLSON, D. L. Utilizing customer satisfaction in ranking prediction for personalized cloud service selection. *Decision Support Systems*, v. 93, p. 1 – 10, 2017a. ISSN 0167-9236. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167923616301506>.

DING, S.; XIA, C.; WANG, C.; WU, D.; ZHANG, Y. Multi-objective optimization based ranking prediction for cloud service recommendation. *Decision Support Systems*, v. 101, p. 106 – 114, 2017b. ISSN 0167-9236. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167923617301173>.

DSPACE. *About DSpace*. 2018. Disponível em: <https://duraspace.org/dspace/about/features/>. Acesso em: 25 Jan. 2018.

EBELING, C. E. *An introduction to reliability and maintainability engineering*. [S.l.]: Tata McGraw-Hill Education, 2004.

EDGEWORTH, F. Y. *Mathematical psychics: An essay on the application of mathematics to the moral sciences*. [S.l.]: CK Paul, 1881.

ESCHNAUER, H.; KOSKI, J.; OSYCZKA, A. *Multicriteria Design Optimization: procedures and application*. [S.l.]: Springer-Verlag Berlin, 1990.

EVER, E. Performability analysis of cloud computing centers with large numbers of servers. *The Journal of Supercomputing*, v. 73, n. 5, p. 2130–2156, May 2017. ISSN 1573-0484. Disponível em: <https://doi.org/10.1007/s11227-016-1906-5>.

FOGLIATTO, F. S.; RIBEIRO, J. L. D. *Confiabilidade e manutenção industrial*. [S.l.]: Elsevier, 2010.

FOUNDATION, A. S. *Apache Tomcat*. 2018. <http://tomcat.apache.org/>.

FREI, F. *Introdução à análise de agrupamentos*. [S.l.]: Unesp, 2006.

FURHT, B. Cloud computing fundamentals. In: *Handbook of cloud computing*. [S.l.]: Springer, 2013. p. 3–19.

GARG, S. K.; VERSTEEG, S.; BUYYA, R. A framework for ranking of cloud computing services. *Future Generation Computer Systems*, Elsevier, v. 29, n. 4, p. 1012–1023, 2013.

GARG, S. K.; VERSTEEG, S.; BUYYA, R. A framework for ranking of cloud computing services. *Future Generation Computer Systems*, v. 29, n. 4, p. 1012 – 1023, 2013. ISSN 0167-739X. Special Section: Utility and Cloud Computing.

GEFFROY, J.-C.; MOTET, G. *Design of dependable computing systems*. [S.l.]: Kluwer Academic Publishers, 2002.

GERMAN, R. *Performance analysis of communication systems with non-Markovian stochastic Petri nets*. [S.l.]: John Wiley & Sons, Inc., 2000.

GHOSH, R.; LONGO, F.; NAIK, V. K.; TRIVEDI, K. S. Modeling and performance analysis of large scale iaas clouds. *Future Generation Computer Systems*, Elsevier, v. 29, n. 5, p. 1216–1234, 2013.

GHOSH, R.; TRIVEDI, K. S.; NAIK, V. K.; KIM, D. S. End-to-end performability analysis for infrastructure-as-a-service cloud: An interacting stochastic models approach. In: IEEE. *Dependable Computing (PRDC), 2010 IEEE 16th Pacific Rim International Symposium on*. [S.l.], 2010. p. 125–132.

GIL, A. C. *Como elaborar projetos de pesquisa*. [S.l.]: Atlas, 2002.

GIRAULT, C.; VALK, R. *Petri nets for systems engineering: a guide to modeling, verification, and applications*. [S.l.]: Springer Science & Business Media, 2001.

GOH, C.-K.; TAN, K. C. Evolutionary multi-objective optimization in uncertain environments. *Studies in Computational Intelligence*, Springer, v. 186, 2009.

GOKHALE, S. S.; LYU, M. R.; TRIVEDI, K. S. Analysis of software fault removal policies using a non-homogeneous continuous time markov chain. *Software Quality Journal*, Springer, v. 12, n. 3, p. 211–230, 2004.

GOLLAPUDI, S. *Practical Machine Learning*. [S.l.]: Packt Publishing Ltd, 2016. 192–193 p.

GONCALVES, R.; TAVARES, E.; NASCIMENTO, A.; SOUSA, E.; LINS, F.; ALVES, G. Performability assessment of a government process in the cloud. In: *2015 IEEE International Conference on Systems, Man, and Cybernetics*. [S.l.: s.n.], 2015. p. 87–92.

Google. *Google Cloud*. 2018. Acesso em: 10 Abr. 2018. Disponível em: <https://cloud.google.com/compute/>.

Google. *Google Cloud Plataform*. 2018. Acesso em: 10 Abr. 2018. Disponível em: <https://console.cloud.google.com>.

Google. *Google Docs*. 2018. Acesso em: 10 Abr. 2018. Disponível em: <http://docs.google.com/>.

Google. *Google Email*. 2018. Acesso em: 10 Abr. 2018. Disponível em: <http://mail.google.com/>.

HAIR, J. F.; BLACK, W. C.; BABIN, B. J.; ANDERSON, R. E.; TATHAM, R. L. *Análise multivariada de dados*. 6. ed. [S.l.]: Bookman, 2009.

HALILI, E. H. *Apache JMeter: A practical beginner's guide to automated testing and performance measurement for your websites*. [S.l.]: Packt Publishing Ltd, 2008.

HARINDRANATH G., W. W. Z. J. R. D. W. W. W. S. S. J. *New Perspectives on Information Systems Development Theory, Methods, and Practice*. [S.l.]: Springer US, 2002.

HAUPT, R. L.; HAUPT, S. E. *Practical genetic algorithms*. [S.l.]: John Wiley & Sons, 2004.

HAVERKORT, B. R.; NIEMEGEERS, I. G. Performability modelling tools and techniques. *Performance evaluation*, Elsevier, v. 25, n. 1, p. 17–40, 1996.

HILL, R.; HIRSCH, L.; LAKE, P.; MOSHIRI, S. *Guide to cloud computing: principles and practice*. [S.l.]: Springer Science & Business Media, 2013.

HOLMES, J.; SCHALK, C. *JavaServer faces: the complete reference*. [S.l.]: McGraw-Hill, Inc., 2006.

HOSSEINIAN-FAR, A.; RAMACHANDRAN, M.; SARWAR, D. *Strategic Engineering for Cloud Computing and Big Data Analytics*. [S.l.]: Springer, 2017.

ITU, T. Terms and definitions related to quality of service and network performance including dependability. *Recommendation E*, v. 800, p. 53, 1994.

JAIN, A. K.; DUBES, R. C. *Algorithms for clustering data*. [S.l.]: Prentice-Hall, Inc., 1988.

JAIN, R. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. [S.l.]: John Wiley & Sons, 1990.

JATOTH, C.; GANGADHARAN, G. R.; FIORE, U.; BUYYA, R. Selcloud: a hybrid multi-criteria decision-making model for selection of cloud services. *Soft Computing*, Mar 2018. ISSN 1433-7479. Disponível em: <https://doi.org/10.1007/s00500-018-3120-2>.

JENSEN, K. *Coloured Petri nets: A high level language for system design and analysis*. [S.l.]: Springer, 1990. 342-416 p.

JMETER, A. *jmeter*. 2018. Disponível em: <https://jmeter.apache.org>. Acesso em: 25 Jan. 2018.

JOHNSON, R.; HOELLER, J.; DONALD, K.; SAMPALEANU, C.; HARROP, R.; RISBERG, T.; ARENDSEN, A.; DAVISON, D.; KOPYLENKO, D.; POLLACK, M. et al. The spring framework–reference documentation. *Interface*, v. 21, 2004.

KAVIS, M. J. *Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS)*. [S.l.]: John Wiley & Sons, 2014.

KIM, D. S.; MACHIDA, F.; TRIVEDI, K. S. Availability modeling and analysis of a virtualized system. In: IEEE. *Dependable Computing, 2009. PRDC'09. 15th IEEE Pacific Rim International Symposium on*. [S.l.], 2009. p. 365–371.

KRAMER, D. Api documentation from source code comments: a case study of javadoc. In: ACM. *Proceedings of the 17th annual international conference on Computer documentation*. [S.l.], 1999. p. 147–153.

KULKARNI, V. G. *Modeling and analysis of stochastic systems*. [S.l.]: Crc Press, 2016.

KUO, W.; ZUO, M. J. *Optimal reliability modeling: principles and applications*. [S.l.]: John Wiley & Sons, 2003.

LANEY, D.; JAIN, A. *100 Data and Analytics Predictions Through 2021*. 2017. Disponível em: <https://www.gartner.com/events-na/data-analytics/wp-content/uploads/sites/5/2017/10/Data-and-Analytics-Predictions.pdf>. Acesso em: 30 Mai. 2018. 08 Maio 2018.

LANUS, M.; YIN, L.; TRIVEDI, K. S. Hierarchical composition and aggregation of state-based availability and performability models. *Reliability, IEEE Transactions on*, IEEE, v. 52, n. 1, p. 44–52, 2003.

LAPRIE, J.-C. *Dependability: basic concepts and terminology*. [S.l.]: Springer, 1992.

LEE, S.; SEO, K.-K. A hybrid multi-criteria decision-making model for a cloud service selection problem using bsc, fuzzy delphi method and fuzzy ahp. *Wireless Personal Communications*, v. 86, n. 1, p. 57–75, Jan 2016. ISSN 1572-834X. Disponível em: <https://doi.org/10.1007/s11277-015-2976-z>.

LEFF, A.; RAYFIELD, J. T. Web-application development using the model/view/controller design pattern. In: IEEE. *Enterprise Distributed Object Computing Conference, 2001. EDOC'01. Proceedings. Fifth IEEE International*. [S.l.], 2001. p. 118–127.

LESIK, S. *Applied Statistical Inference with MINITAB®*. [S.l.]: CRC Press, 2009.

LILJA, D. J. *Measuring computer performance: a practitioner's guide*. [S.l.]: Cambridge university press, 2005.

LIRA, W. S.; CÂNDIDO, G. A. *Gestão sustentável dos recursos naturais: uma abordagem participativa*. [S.l.]: SciELO-EDUEPB, 2013.

LIU, B.; CHANG, X.; HAN, Z.; TRIVEDI, K.; RODRíGUEZ, R. J. Model-based sensitivity analysis of iaas cloud availability. *Future Generation Computer Systems*, v. 83, p. 1 – 13, 2018. ISSN 0167-739X.

LIU, S.; CHAN, F. T.; RAN, W. Decision making for the selection of cloud vendor: An improved approach under group decision-making with integrated weights and objective/subjective attributes. *Expert Systems with Applications*, v. 55, p. 37 – 47, 2016. ISSN 0957-4174. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0957417416300239>.

MACIEL, P.; MATOS, R.; SILVA, B.; FIGUEIREDO, J.; OLIVEIRA, D.; Fé, I.; MACIEL, R.; DANTAS, J. Mercury: Performance and dependability evaluation of systems with exponential, expolynomial, and general distributions. In: *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*. [S.l.: s.n.], 2017. p. 50–57.

MACIEL, P.; TRIVEDI, K.; MATIAS, R.; KIM, D. Dependability modeling. In: CARDELLINI, V.; CASALICCHIO, E.; REGINA, K. et al. (Ed.). *Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions*. [S.l.]: Information Science Reference, 2012.

MACIEL, P. R.; LINS, R. D.; CUNHA, P. R. *Introdução às redes de Petri e aplicações*. [S.l.]: UNICAMP-Instituto de Computação, 1996.

MAGALHAES, I. L.; PINHEIRO, W. B. *Gerenciamento de serviços de TI na prática: uma abordagem com base na ITIL: inclui ISO/IEC 20.000 e IT Flex*. [S.l.]: Novatec Editora, 2007.

MAITY, S.; CHAUDHURI, A. Optimal negotiation of sla in federated cloud using multiobjective genetic algorithms. In: IEEE. *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*. [S.l.], 2014. p. 269–271.

MARCONI, M. de A.; LAKATOS, E. M. *Fundamentos de metodologia científica*. 7. ed. [S.l.]: Atlas, 2014.

MARINESCU, D. C. *Cloud computing: theory and practice*. [S.l.]: Morgan Kaufmann, 2017.

MARSAN, M. A. Stochastic petri nets: an elementary introduction. In: *Advances in Petri Nets 1989*. [S.l.]: Springer, 1989. p. 1–29.

MARSAN, M. A.; BALBO, G.; CONTE, G.; DONATELLI, S.; FRANCESCHINIS, G. *Modelling with generalized stochastic Petri nets*. [S.l.]: John Wiley & Sons, Inc., 1994.

MARSAN, M. A.; CONTE, G.; BALBO, G. A class of generalized stochastic petri nets for the performance evaluation of multiprocessor systems. *ACM Transactions on Computer Systems (TOCS)*, ACM, v. 2, n. 2, p. 93–122, 1984.

MARTENS, B.; TEUTEBERG, F. Decision-making in cloud computing environments: A cost and risk based approach. *Information Systems Frontiers*, Springer, v. 14, n. 4, p. 871–893, 2012.

MATOS, R.; DANTAS, J.; ARAUJO, J.; TRIVEDI, K. S.; MACIEL, P. Redundant eucalyptus private clouds: Availability modeling and sensitivity analysis. *Journal of Grid Computing*, v. 15, n. 1, p. 1–22, Mar 2017. ISSN 1572-9184. Disponível em: <https://doi.org/10.1007/s10723-016-9381-z>.

MELL, P.; GRANCE, T. The nist definition of cloud computing. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, p. 1–7, 2011.

MELO, C.; MATOS, R.; DANTAS, J.; MACIEL, P. Capacity-oriented availability model for resources estimation on private cloud infrastructure. In: *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*. [S.l.: s.n.], 2017. p. 255–260.

MELO, M.; ARAUJO, J.; MATOS, R.; MENEZES, J.; MACIEL, P. Comparative analysis of migration-based rejuvenation schedules on cloud availability. In: IEEE. *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. [S.l.], 2013. p. 4110–4115.

MENASCÉ, D. A.; ALMEIDA, V.; DOWDY, L. W. *Capacity Planning for Web Services: metrics, models, and methods*. [S.l.]: Prentice Hall PTR Upper Saddle River, 1994.

MENASCE, D. A.; ALMEIDA, V. A.; DOWDY, L. W.; DOWDY, L. *Performance by design: computer capacity planning by example*. [S.l.]: Prentice Hall Professional, 2004.

MENASCÉ, D. A.; ALMEIDA, V. A.; FONSECA, R.; MENDES, M. A. A methodology for workload characterization of e-commerce sites. In: ACM. *Proceedings of the 1st ACM conference on Electronic commerce*. [S.l.], 1999. p. 119–128.

MERLIN, P. M.; FARBER, D. J. Recoverability of communication protocols–implications of a theoretical study. *Communications, IEEE Transactions on*, IEEE, v. 24, n. 9, p. 1036–1043, 1976.

MEYER, J. F. On evaluating the performability of degradable computing systems. *IEEE Transactions on computers*, IEEE, n. 8, p. 720–731, 1980.

Microsoft. *Microsoft Azure*. 2018. Acesso em: 10 Abr. 2018. Disponível em: <https://azure.microsoft.com/pt-br/>.

MITSA, T. *Temporal data mining*. [S.l.]: CRC Press, 2010.

MO, Y.; CUI, L.; XING, L.; ZHANG, Z. Performability analysis of large-scale multi-state computing systems. *IEEE Transactions on Computers*, v. 67, n. 1, p. 59–72, Jan 2018. ISSN 0018-9340.

MoDCS. *Mercury*. 2018. Acesso em: 10 Abr. 2018. Disponível em: <http://www.modcs.org/?page$_id = 2392 >$.

MODCS. *MoDCS Research Group*. 2018. <http://www.modcs.org/>.

MONCZKA, R.; HANDFIELD, R.; GIUNIPERO, L.; PATTERSON, J. *Purchasing and supply chain management*. 4. ed. [S.l.]: Cengage Learning, 2009.

MONTGOMERY, D. C. *Design and analysis of experiments*. [S.l.]: John Wiley & Sons, 2013.

MUPPALA, J. K.; MALHOTRA, M.; TRIVEDI, K. S. Markov dependability models of complex systems: Analysis techniques. In: *Reliability and Maintenance of Complex Systems*. [S.l.]: Springer, 1996. p. 442–486.

MURATA, T. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, IEEE, v. 77, n. 4, p. 541–580, 1989.

MYSQL, I. *MySQL Database*. 2018. <https://www.mysql.com/>.

NICOLETTI, B. *Cloud Computing in Financial Services*. [S.l.]: Palgrave Macmillan, 2013.

OLIVEIRA, V. H. M. de; MARTINS, C. H. *Ahp: ferramenta multicritério para tomada de decisão-shopping centers*. [S.l.]: Appris Editora e Livraria Eireli-ME, 2015.

OPENNEBULA. 2018. Acesso em: 16 Abr. 2018. Disponível em: <http://opennebula.org/>.

OPENSTACK. 2018. Acesso em: 10 Abr. 2018. Disponível em: <https://www.openstack.org/>.

PARETO, V. *Cours d'économie politique*. [S.l.]: Lausanne: F. Rouge, 1896.

PETRI, C. *Kommunikation mit Automaten. Technische Hochschule, Darmstadt*. Tese (Doutorado) — Ph. D. Thesis, 1962.

POPOVA-ZEUGMANN, L. Time petri nets. In: *Time and Petri Nets*. [S.l.]: Springer, 2013. p. 31–137.

PRADHAN, D. K. *Fault-tolerant computer system design*. [S.l.]: Prentice-Hall, 1996.

QIU, X.; SUN, P.; GUO, X.; XIANG, Y. Performability analysis of a cloud system. In: *2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC)*. [S.l.: s.n.], 2015. p. 1–6.

RAJARAMAN, V. *Introduction to information Technology*. [S.l.]: PHI Learning Pvt. Ltd., 2018.

RAO, S. S. *Engineering optimization: theory and practice*. 4. ed. [S.l.]: John Wiley & Sons, 2009.

REDIS, I. *Redis data structure project*. 2018. <https://redis.io/>.

REHMAN, Z.; HUSSAIN, O.; HUSSAIN, F. Iaas cloud selection using mcdm methods. In: IEEE. *e-Business Engineering (ICEBE), 2012 IEEE Ninth International Conference on*. [S.l.], 2012. p. 246–251.

REHMAN, Z. ur; HUSSAIN, O.; HUSSAIN, F. Iaas cloud selection using mcdm methods. In: *e-Business Engineering (ICEBE), 2012 IEEE Ninth International Conference on*. [S.l.: s.n.], 2012. p. 246–251.

RIGHTSCALE. *100 Data and Analytics Predictions Through 2021*. 2017. Disponível em: <https://www.rightscale.com/lp/2017-state-of-the-cloud-report>. Acesso em: 30 Mai. 2018. Junho 2017.

RUBIO, P. *Toma e lê! Síntese agostiniana*. [S.l.]: Edicoes Loyola, 1995.

RUSSEL, S.; NORVIG, P. Inteligência artificial. *Editora Campus*, 2004.

SACHDEVA, N.; SINGH, O.; KAPUR, P. K.; GALAR, D. Multi-criteria intuitionistic fuzzy group decision analysis with topsis method for selecting appropriate cloud solution to manage big data projects. *International Journal of System Assurance Engineering and Management*, v. 7, n. 3, p. 316–324, Sep 2016. ISSN 0976-4348. Disponível em: <https://doi.org/10.1007/s13198-016-0455-x>.

SAHNER, R. A.; TRIVEDI, K.; PULIAFITO, A. *Performance and reliability analysis of computer systems: an example-based approach using the SHARPE software package*. [S.l.]: Springer Publishing Company, Incorporated, 2002.

SATHAYE, A.; RAMANI, S.; TRIVEDI, K. S. Availability models in practice. In: *Proc. of Intl. Workshop on Fault-Tolerant Control and Computing (FTCC-1)*. [S.l.: s.n.], 2000. p. 1–8.

SHIVAKUMAR, U.; RAVI, V.; GANGADHARAN, G. Ranking cloud services using fuzzy multi-attribute decision making. In: IEEE. *Fuzzy Systems (FUZZ), 2013 IEEE International Conference on*. [S.l.], 2013. p. 1–8.

SHOOMAN, M. L. *Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design*. [S.l.]: John Wiley & Sons, 2002.

SILVA, B.; CALLOU, G.; TAVARES, E.; MACIEL, P.; FIGUEIREDO, J.; SOUSA, E.; ARAUJO, C.; MAGNANI, F.; NEVES, F. Astro: An integrated environment for dependability and sustainability evaluation. *Sustainable Computing: Informatics and Systems*, Elsevier, v. 3, n. 1, p. 1–17, 2013.

SILVA, B.; MACIEL, P.; TAVARES, E.; ZIMMERMANN, A. Dependability models for designing disaster tolerant cloud computing systems. In: IEEE. *Dependable Systems and Networks (DSN), 2013 43rd Annual IEEE/IFIP International Conference on*. [S.l.], 2013. p. 1–6.

SOUSA, E.; LINS, F.; TAVARES, E.; CUNHA, P.; MACIEL, P. A modeling approach for cloud infrastructure planning considering dependability and cost requirements. *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, v. 45, n. 4, p. 549–558, April 2015. ISSN 2168-2216.

SOUSA, E.; LINS, F.; TAVARES, E.; MACIEL, P. Cloud infrastructure planning considering different redundancy mechanisms. *Computing*, v. 99, n. 9, p. 841–864, Sep 2017. ISSN 1436-5057. Disponível em: <https://doi.org/10.1007/s00607-016-0533-6>.

TAI, A. T.; MEYER, J. F.; AVIZIENIS, A. Performability enhancement of fault-tolerant software. *IEEE Transactions on Reliability*, v. 42, n. 2, p. 227–237, Jun 1993. ISSN 0018-9529.

THODGE, S. *Cloud Analytics with Google Cloud Platform: An end-to-end guide to processing and analyzing big data using Google Cloud Platform*. [S.l.]: Packt Publishing, 2018.

TIGRE, P. B.; NORONHA, V. B. Do mainframe à nuvem: inovações, estrutura industrial e modelos de negócios nas tecnologias da informação e da comunicação. *Revista de Administração*, SciELO Brasil, v. 48, n. 1, p. 114–127, 2013.

TORALDO, G. *OpenNebula 3 Cloud Computing*. [S.l.]: Packt Publishing Ltd, 2012.

TORRES, E.; CALLOU, G.; ANDRADE, E. A hierarchical approach for availability and performance analysis of private cloud storage services. *Computing*, v. 100, n. 6, p. 621–644, Jun 2018. ISSN 1436-5057. Disponível em: <https://doi.org/10.1007/s00607-018-0588-7>.

TRIVEDI, K. S. *Probability & statistics with reliability, queuing and computer science applications*. [S.l.]: John Wiley & Sons, 2002.

TRIVEDI, K. S.; BOBBIO, A. *Reliability and Availability Engineering: Modeling, Analysis, and Applications*. [S.l.]: Cambridge University Press, 2017.

VARAKSIN, O. *PrimeFaces Cookbook*. [S.l.]: Packt Publishing Ltd, 2013.

VARGHESE, B.; BUYYA, R. Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems*, v. 79, p. 849 – 861, 2018. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X17302224>.

VERAS, M. *Arquitetura de Nuvem - Amazon Web Services*. [S.l.]: Brasport, 2013.

VOAS, J.; ZHANG, J. Cloud computing: New wine or just a new bottle? *IT professional*, IEEE, v. 11, n. 2, p. 15–17, 2009.

WANG, L.-C. Object-oriented petri nets for modelling and analysis of automated manufacturing systems. *Computer Integrated Manufacturing Systems*, Elsevier, v. 9, n. 2, p. 111–125, 1996.

WANG, T.; CHANG, X.; LIU, B. Performability analysis for iaas cloud data center. In: *2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*. [S.l.: s.n.], 2016. p. 91–94.

WEI, B.; LIN, C.; KONG, X. Dependability modeling and analysis for the virtual clusters. In: IEEE. *Computer Science and Network Technology (ICCSNT), 2011 International Conference on*. [S.l.], 2011. v. 4, p. 2316–2320.

WEI, B.; LIN, C.; KONG, X. Dependability modeling and analysis for the virtual data center of cloud computing. In: IEEE. *High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on*. [S.l.], 2011. p. 784–789.

WEIL, R. L.; MAHER, M. W. *Handbook of cost management*. 2. ed. [S.l.]: John Wiley & Sons, 2005.

XIE, M.; DAI, Y.-S.; POH, K.-L. *Computing system reliability: models and analysis*. [S.l.]: Springer Science & Business Media, 2004.

XING, L.; AMARI, S. V.; MISRA, K. Handbook of performability engineering. *Fault Tree Analysis, London, Springer London*, p. 595–620, 2008.

YANG, G. *Life cycle reliability engineering*. [S.l.]: John Wiley & Sons, 2007.

ZOPOUNIDIS, C.; PARDALOS, P. M. *Handbook of multicriteria analysis*. [S.l.]: Springer Science & Business Media, 2010. v. 103.

APÊNDICE A – VERIFICAÇÃO DO *RANKING*

Neste apêndice são apresentadas as comparações dos *ranking's* considerando o indicador de qualidade. Essa verificação tem como objetivo a verificação do *ranking* gerado para cada caso apresentado na seção 6.4 do capítulo de Estudos de Caso. Para isso, foi utilizado o teste estatístico não-paramétrico Wilcoxon (LESIK, 2009) para os *ranking* I, II e III. Esse teste é utilizado quando necessitamos verificar se as medidas de duas amostras são iguais no caso em que as amostras são dependentes. Foi necessário estabelecer as amostras de forma pareada. A hipótese estabelecida foi testada buscando verificar se as amostras diferem. Para aceitarmos a hipótese nula, temos que a mediana da diferença é nula, ou seja, as amostras não diferem. Utilizando um α de 5% não identificamos a presença de valores críticos. Sem a presença de valores críticos podemos afirmar que a hipótese é nula. Portanto, não há evidências de diferenças entre as duas amostras.

As Tabelas 61, 62 e 63 apresentam as respectivas comparações dos *ranking* I, II e III. Dist_E representa o uso da medida de agrupamento por distância euclidiana e I_E o indicador de qualidade baseado na distância euclidiana.

Tabela 61 – Comparação do Ranking das IaaS e Indicador de Qualidade (Ranking I).

Ranking ($Dist_E$)	IaaS	Ranking (I_E)	IaaS
1	38	1	38
2	40	2	40
3	28	3	28
4	26	4	26
5	30	5	30
6	34	6	34
7	39	7	39
8	27	8	27
9	32	9	32
10	36	10	36
11	31	11	31
12	35	12	35
13	14	13	14
14	16	14	16
15	15	15	15
16	1	16	1
17	18	17	18
18	20	18	20
19	19	19	19
20	5	20	5
21	22	21	22
22	24	22	24
23	23	23	23
24	50	24	50
25	54	25	54
26	70	26	70
27	58	27	58
28	66	28	66
29	46	29	46
30	48	30	48
31	62	31	62
32	9	32	9
33	42	33	42
34	44	34	44
35	47	35	47
36	43	36	43

Tabela 62 – Comparação do Ranking das IaaS e Indicador de Qualidade (*ranking* II).

Ranking ($Dist_E$)	IaaS	Ranking (I_E)	IaaS
1	39	1	39
2	38	2	38
3	40	3	40
4	28	4	28
5	27	5	27
6	35	6	35
7	34	7	34
8	36	8	36
9	31	9	31
10	26	10	26
11	30	11	30
12	32	12	32
13	24	13	24
14	20	14	20
15	16	15	16
16	23	16	23
17	19	17	19
18	15	18	15
19	22	19	22
20	18	20	18
21	14	21	14
22	50	22	50
23	54	23	54
24	70	24	70
25	58	25	58
26	66	26	66
27	47	27	47
28	46	28	46
29	62	29	62
30	9	30	9
31	5	31	5
32	1	32	1
33	42	33	42
34	43	34	43
35	44	35	44
36	48	36	48

Tabela 63 – Comparação do Ranking das IaaS e Indicador de Qualidade (*ranking* III).

Ranking ($Dist_E$)	IaaS	Ranking (I_E)	IaaS
1	62	1	62
2	50	2	50
3	38	3	38
4	26	4	26
5	66	5	35
6	54	6	54
7	42	7	42
8	30	8	30
9	40	9	40
10	28	10	28
11	70	11	70
12	58	12	58
13	46	13	46
14	34	14	34
15	44	15	44
16	39	16	39
17	32	17	32
18	27	18	27
19	43	19	43
20	36	20	36
21	31	21	31
22	47	22	47
23	35	23	35
24	48	24	48
25	16	25	16
26	15	26	15
27	14	27	14
28	20	28	20
29	19	29	19
30	18	30	18
31	1	31	1
32	24	32	24
33	23	33	23
34	22	34	22
35	5	35	5
36	9	36	9

APÊNDICE B – FERRAMENTA - MIPACE

A ferramenta tratada nesta seção é a MiPACE (*A Multi-criteria tool for Planning and Analysis of Cloud Environments*). Ela foi desenvolvida para dar suporte no planejamento de cenários de infraestruturas como serviço na nuvem e para auxiliar no processo de tomada de decisão. Essa ferramenta permite que usuários, analistas e gestores de infraestruturas como serviço em nuvem, que possam ter alguma experiência em método de agrupamento de dados, planejem e analisem cenários de IaaS. A partir dela é possível: (i) gerar planos de experimentos e (ii) realizar um *ranking* das configurações considerando funções multicritério. A abordagem utilizada na ferramenta auxilia usuários que necessitam tomar uma decisão para escolher um serviço em nuvem.

B.1 MIPACE

A ferramenta MiPACE[1] foi desenvolvida na linguagem *C*. Essa linguagem é considerada do tipo estrutural e de uso geral, e tem sido utilizada para a solução de problemas da engenharia, física, química e outras áreas da ciência (COCIAN, 2004). A Figura 71 ilustra a tela de apresentação da ferramenta, que é composta por duas partes: planejamento de experimentos e tomada de decisão. O usuário pode escolher gerar um plano de experimentos ou realizar uma análise para tomada de decisão.

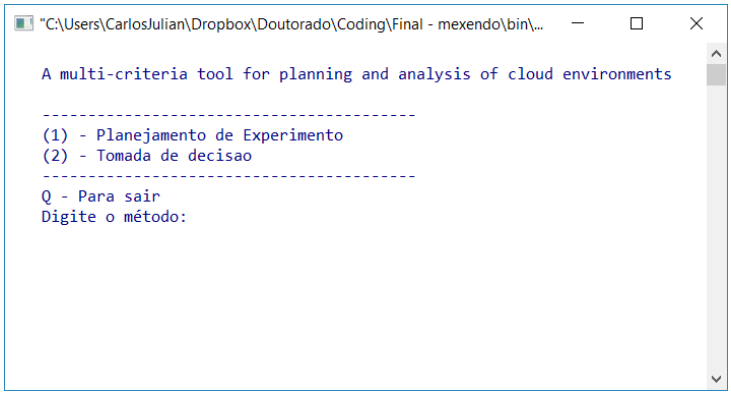


Figura 71 – Tela de apresentação da ferramenta (MiPACE).

A Figura 72 apresenta as funcionalidade implementadas na ferramenta, que estão brevemente resumidas na sequência.

- **Planejamento de experimentos:** Técnicos, analistas e gestores podem gerar um conjunto de cenários de IaaS, considerando ambientes de computação em nuvem privada para serem avaliadas. Antes de escolher os fatores (variáveis independentes) que influenciam na variável de resposta, é necessário definir o objetivo de estudo. Por

[1] O código fonte e toda a estrutura do projeto está disponível no repositório GitHub (ARAUJO, 2019).

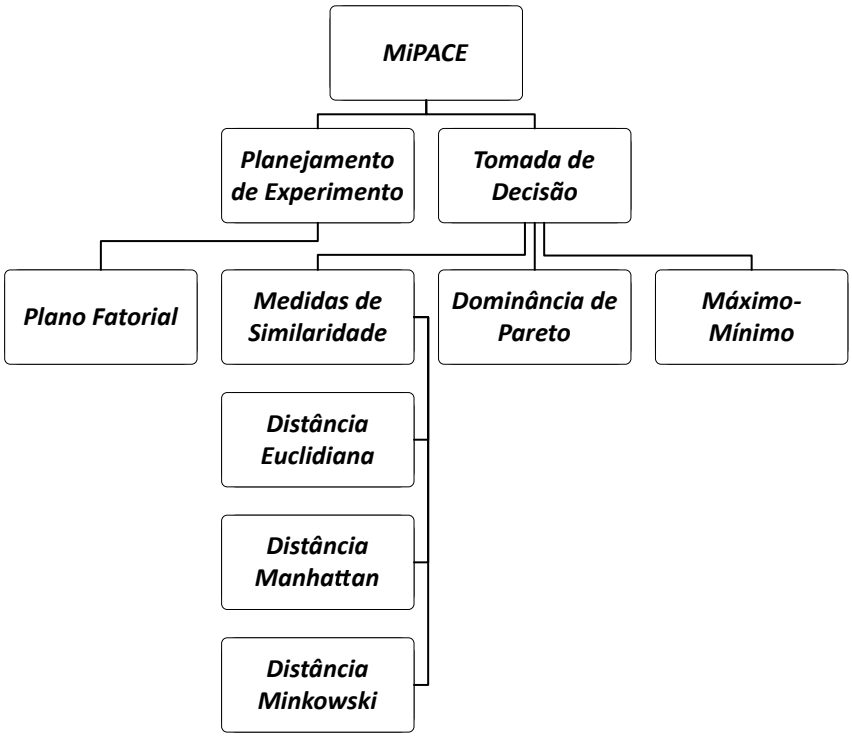


Figura 72 – Funcionalidades da ferramenta.

exemplo, pode-se supor que um dado usuário deseje selecionar um conjunto de IaaS que suporte mecanismos com redundância. Nesse sentido, o plano de experimento pode ser composto por fatores como mecanismo de redundância *hot-standby*, *coldstandby* e *warmstandby*. Os níveis podem ser adotados por meio de variações no tempo médio de reparo. A ferramenta permite que sejam gerados planos de experimentos do tipo fatorial completo, contendo até dez fatores e até dez níveis por fator. Os cenários gerados no plano podem ser avaliados considerando os modelos de dependabilidade e custo apresentados no Capítulo 5;

- **Tomada de decisão:** Com o propósito de analisar os cenários gerados no plano de experimentos, e considerando que esses já foram avaliados através dos modelos de dependabilidade e custos, a ferramenta dispõe de cinco opções com funções objetivas (ver Figura 74) para realizar o *ranking* dos cenários. Os cenários das IaaS podem ser analisados através do seguintes métodos: medidas de similaridade e dominância de Pareto.

Considerando que na Figura 71 o usuário escolheu a opção (1), o próximo passo será a geração de um plano de experimento, onde é necessário informar a quantidade de fatores a serem gerados. Na Figura 73 é ilustrado um exemplo onde um usuário pode criar um plano de experimento. Em seguida, o usuário é solicitado a informar a quantidade de fatores no plano de experimentos. O usuário que conduz esse método deve ter um conhecimento prévio sobre quais fatores deseja utilizar e quais os níveis deve ter cada fator. Os fatores correspondem às variáveis independentes que são manipuladas e visam alterar a variável

de saída (dependente). Após digitar a quantidade de fatores, a ferramenta solicita ao usuário a descrição do primeiro fator. No próximo passo, o usuário deve descrever o nome de cada nível. Esse processo é finalizado quando o usuário informa todos os fatores e seus respectivos níveis. Após concluído, é gerado, então, o plano de experimentos como saída no formato de uma matriz. A saída é apresentada em um arquivo com o formato do tipo texto (como o .txt).

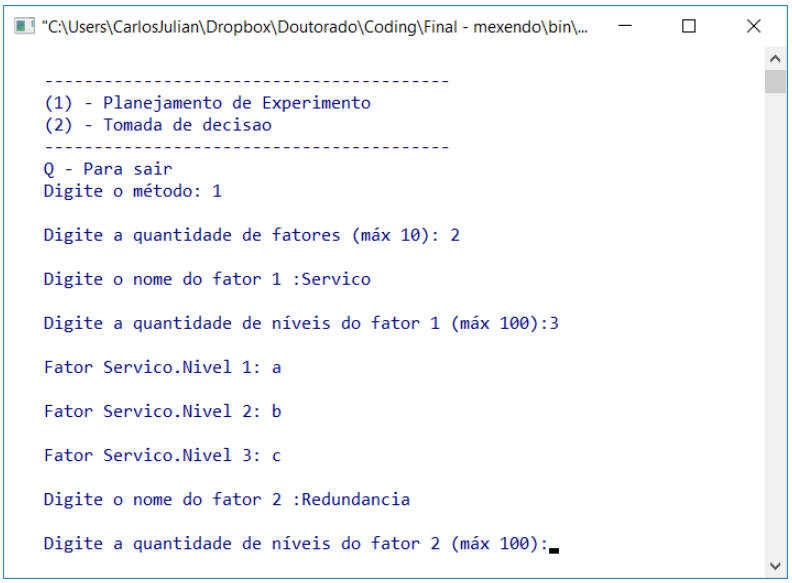


Figura 73 – Exemplo de uso do planejamento de experimento.

Ainda sobre a Figura 71, na opção (2), o usuário pode realizar uma análise para tomada de decisão. Para tanto, é necessário antes que o arquivo com os dados de entrada estejam arranjandos em uma matriz do tipo $n \times 2$ em um arquivo de texto (.txt). Após digitar a opção (2), o usuário irá em seguida informar qual a função multicritério. A função multicritério pode minimizar ou maximizar a busca por um conjunto de soluções considerando o conjunto de cenários disponíveis como entrada. Na Figura 74, a ferramenta disponibiliza cinco opções, que são, respectivamente: minimizar-minimizar, maximixar-minimizar, minimizar-maximizar e maximizar-maximizar dois objetivos. A quinta opção considera a ordenação dos cenários com o método intervalo mínimo e máximo.

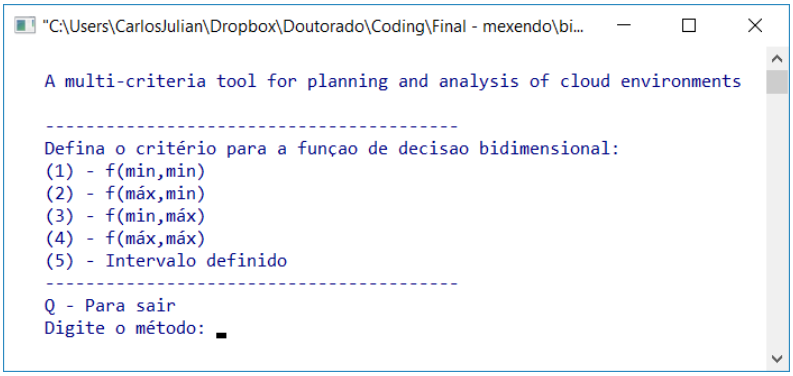


Figura 74 – Funções para ordenação dos resultados.

Caso o usuário escolha uma opção entre (`1 e 4`), um menu é apresentado com as seguintes opções: Medidas de Similaridade ou Dominância de Pareto. A Figura 75 ilustra esse menu de opções para o *ranking* dos cenários. Essa escolha deve ser baseada no objetivo que o usuário de ambientes de computação em nuvem deseja, visto que o método de similaridade considera o cálculo das distâncias entre as soluções, e o segundo método considera o conceito de dominância de Pareto para agrupar o conjunto de soluções.

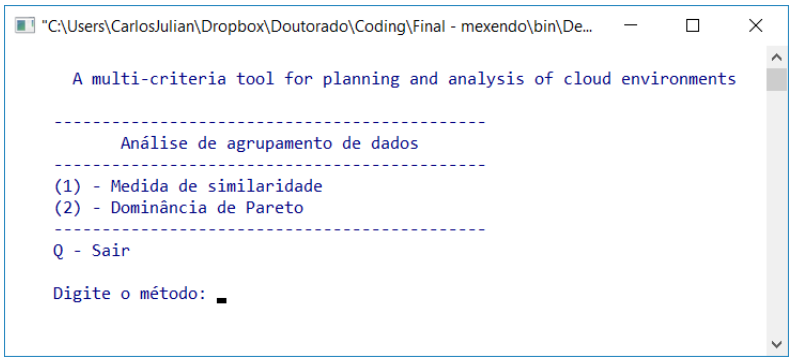


Figura 75 – Exemplo de uso para o *ranking* de cenários.

Caso o usuário tenha o interesse em realizar o *ranking* dos cenários das IaaS considerando as medidas de similaridade, digitando a opção (`1`), um menu com os métodos implementados é apresentado. A Figura 76 ilustra as opções para o *ranking* com as seguintes medidas: Distância Euclidiana, Manhattan e Minkowski. Cada medida de similaridade leva em conta uma equação que calcula a distância entre as soluções num plano cartesiano (Ver Subseção 2.8.1). Os valores de cada coordenada (`x` e `y`) representam, respectivamente, duas métricas, por exemplo, custo e indisponibilidade de cada IaaS avaliada.

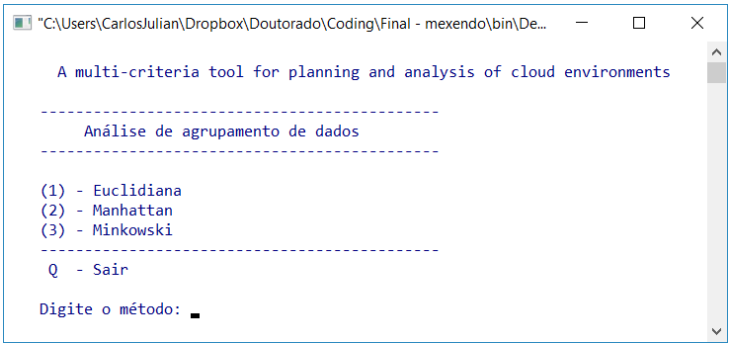


Figura 76 – Exemplo de uso para escolha do método de medidas de similaridade.

A Figura 77 apresenta uma ilustração onde um usuário fez a escolha para o cálculo da distância euclidiana. Após digitar a opção (`1`), o usuário necessita informar se deseja adicionar peso aos objetivos, sendo `0` para `Não` e `1` para `Sim`. Caso a opção seja `Sim`, o usuário deve informar um valor que representa o peso entre `0` e `1`. Essa funcionalidade possibilita que o cálculo das distâncias atribua importância ao dado objetivo. Após essa definição, um arquivo de saída em formato de texto é gerado com um conjunto de soluções contendo as IaaS ordenadas. A exceção é para a medida de similaridade *Minkowski* que solicitar que o usuário informe o valor do parâmetro `p`.

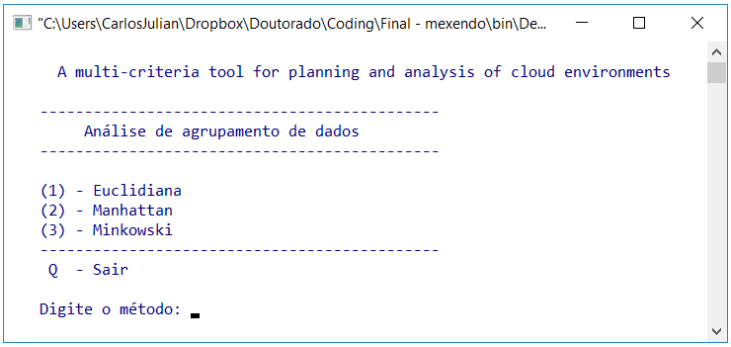


Figura 77 – Medida de similaridade - dist. Euclidiana.

Considerando a Figura 75, caso o usuário escolha a opção (`2`), o método para o *ranking* dos cenários será baseado no conceito de dominância de Pareto. Ao escolher essa opção, o usuário necessita digitar a quantidade máxima de soluções para compor o conjunto de soluções não-dominadas. Esse conjunto de soluções agrupado também representa a fronteira de Pareto.

O terceiro método para o *ranking* dos cenários é o método com intervalo mínimo-máximo. A Figura 78 ilustra um exemplo utilizando esse método. Após escolher o método mínimo-máximo, o usuário necessita informar qual objetivo deseja realizar na busca por meio de um intervalo. A opção `1` considera apenas o primeiro objetivo, a opção `2` considera o segundo objetivo e a opção `3` busca soluções com ambos os objetivos. Esse método possibilita, para analistas, gestores e usuários de ambientes de computação em nuvem, fazer uma busca por cenários dentro de um intervalo com limites mínimos e máximos no

conjunto de soluções. Caso o usuário tenha um limite de orçamento para alocar um serviço numa IaaS na nuvem, a ferramenta poderá auxiliar localizar um conjunto de soluções possíveis. É uma alternativa também que o usuário utilize um conjunto de soluções gerados como saída e o faça como entrada para realizar um novo *ranking*, considerando as medidas de similaridade, por exemplo.

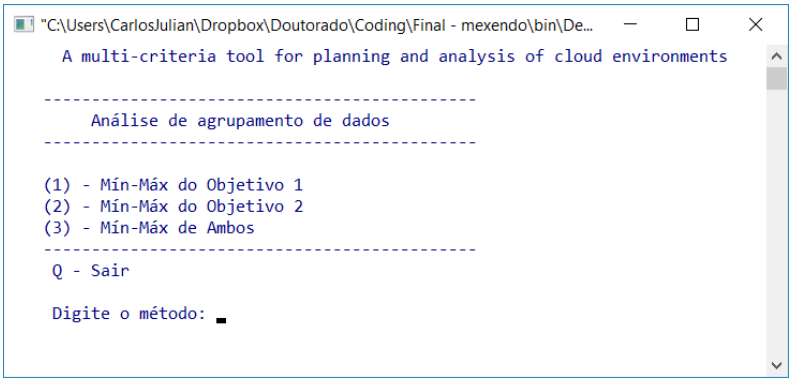


Figura 78 – Exemplo do uso do mín-máx.

B.2 MIGRAÇÃO DA FERRAMENTA

Após a finalização da implementação e obtenção dos resultados utilizando a ferramenta, percebeu-se a necessidade de disponibilizar a ferramenta para o grupo MoDCS (MODCS, 2018) e demais pesquisadores da área por meio de um servidor *web*. Além disso, percebemos que era preciso realizar algumas melhorias na estruturação do código. Assim, a proposta permitiu que usuários acessem a aplicação e envie suas requisições utilizando um navegador *web*.

A nova versão da ferramenta MiPACE é um projeto *web* em *Java EE*. O código-fonte da MiPACE está documentado pelo recurso *JavaDoc* (KRAMER, 1999), o que torna mais fácil para o desenvolvedor/usuário entender e estender o código. A ferramenta também fornece uma *interface web* de modelo, que foi implementada usando *Java Server Faces* (JSF) (HOLMES; SCHALK, 2006), essa *interface web* pretende fornecer uma interação mais amigável com usuário, mas é importante notar que não é obrigatório usar a *interface* da *web*, o usuário pode usar todas as funcionalidades diretamente em uma nova classe e imprimir os resultados no *console java*. Tal independência é possível graças ao padrão de projeto *Model-View-Controller* (LEFF; RAYFIELD, 2001).

A Figura 79 ilustra a arquitetura da ferramenta MiPACE. A arquitetura de *backend* foi desenvolvida com a linguagem de programação Java. A aplicação é gerenciado por um servidor *web* (Tomcat (FOUNDATION, 2018)) e o *framework Spring* (JOHNSON et al., 2004). O *framework* permite que a aplicação se comunique com o banco de dados (MySQL (MYSQL, 2018)) e cache (Redis (REDIS, 2018)) para receber e enviar solicitações do *frontend*. A aplicação usa três camadas para garantir segurança, autenticação e integridade dos dados.

No *frontend*, a interface usa JSF (HOLMES; SCHALK, 2006) e a biblioteca *Primefaces* (VARAKSIN, 2013). A ferramenta MiPACE pode ser acessada em (ARAUJO, 2019).

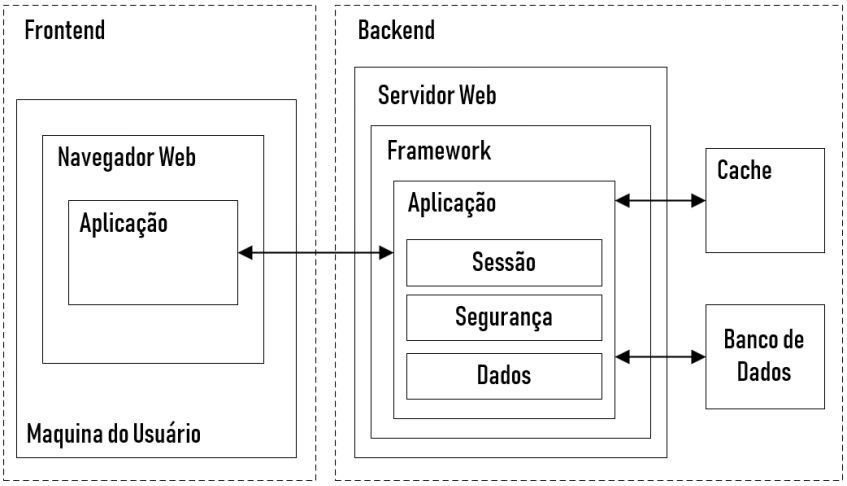


Figura 79 – Arquitetura MiPACE.

A Figura 80 mostra a ferramenta MiPACE. A ferramenta permite a partir da entrada de um arquivo de entrada: (i) gerar planos de experimento e (ii) realizar *ranking* para tomada de decisão. O recurso planejamento de experimentos permite que os usuários planejem experimentos considerando fatores e níveis. Inicialmente, é necessário escolher um número de fatores a serem combinados com os níveis. O usuário indica o número de níveis para cada fator. A aplicação disponibiliza para o usuário um arquivo de exemplo para *download*.

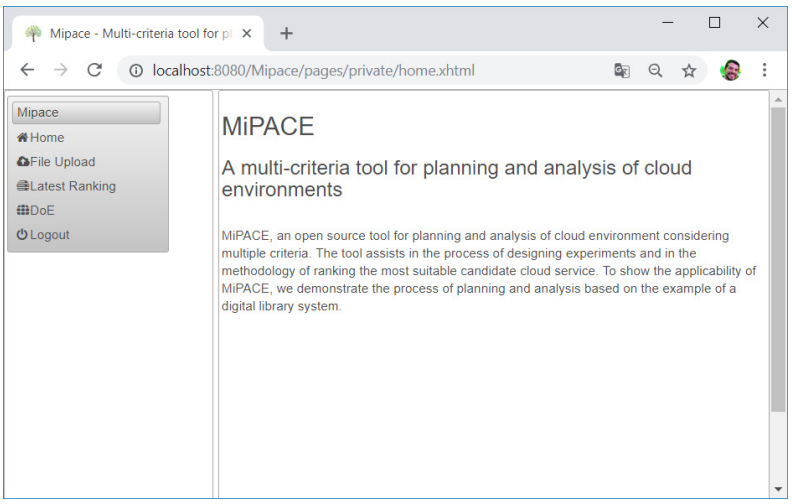


Figura 80 – Ferramenta MiPACE.

O processo de tomada de decisão é realizado incialmente pelo *upload* (Figura 81) de um arquivo com as alternativas organizadas em uma matriz de entrada. Após o carregamento, o usuário definirá a função multicritério considerando a minimização ou maximização dos critérios de decisão. Além disso, o usuário pode adicionar pesos para cada variável

visando priorizar um dado critério e, finalmente, escolher a medida de distância para realizar o ranqueamento das soluções, por meio das distâncias Euclidiana, Manhattan ou Minkowski (GOLLAPUDI, 2016). Um arquivo de exemplo também está disponível para *download* na tela de *upload*.

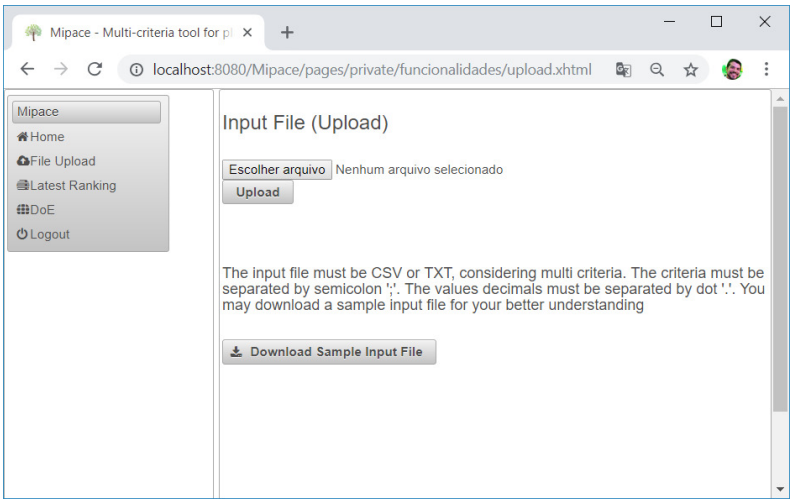


Figura 81 – Tela *upload* MiPACE.

A Figura 82 ilustra a tela de um ranqueamento realizado após a definição dos critérios, pesos e medida de distância. É importante salientar que os ranqueamentos são registrados no banco de dados da ferramenta para futuras consultas. O resultado do ranqueamento podem ser exportado considerando os formatos `csv, txt, pdf`.

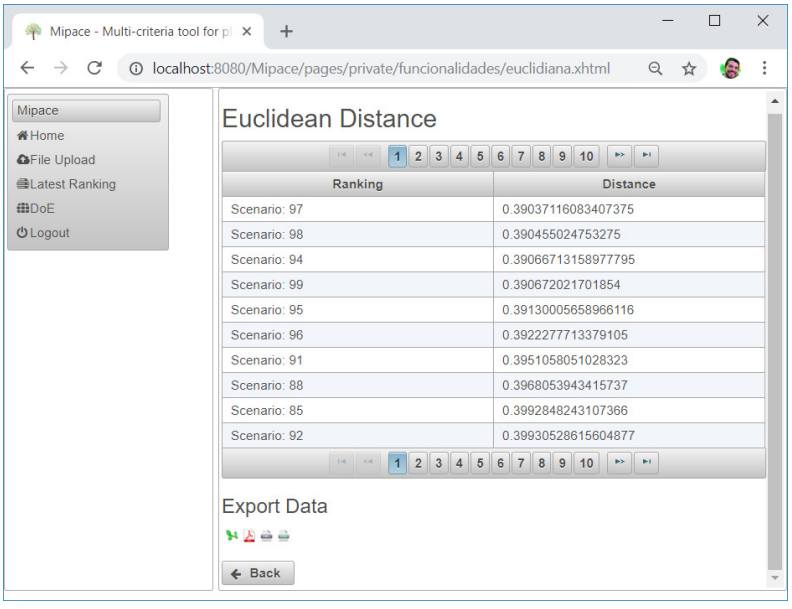


Figura 82 – Ferramenta MiPACE.

B.3 CONSIDERAÇÕES FINAIS

Diante da variedade de soluções disponibilizadas como serviço na nuvem, usuários podem ter dificuldade em identificar um conjunto de soluções diante de requisitos conflitantes. Visando auxiliar o processo de tomada de decisão desses usuários, métodos de ordenação podem ser utilizados para ajudar no caminho de escolha de um determinado serviço. Este apêndice apresentou a ferramenta MiPACE considerando a metodologia proposta. Assim, foram apresentados os passos para o planejamento de experimentos e *ranking* das soluções modeladas e avaliadas nesta tese. A ferramenta MiPACE pode ser utilizada por usuários, analistas e gestores que tenham algum conhecimento sobre planejamento de experimento e noções sobre agrupamento de dados.

APÊNDICE C – RESULTADOS DO *RANKING*

A seguir são apresentados os resultados das métricas avaliadas no modelo SPN da Seção 6.5. As métricas disponibilidade, disponibilidade orientada à capacidade, confiabilidade e indisponibilidade são apresentadas nas Tabelas 64, 65, 66, e 67. Para a análise da indisponibilidade consideramos um intervalo de tempo de um ano em horas ($8760hr$). Já a confiabilidade foi obtida através de análise transiente considerando como período $730hr$. O processo de avaliação do modelo foi conduzido usando a ferramenta Mercury (MACIEL et al., 2017).

Tabela 64 – Resultados dos Cenários.

IaaS	Disponibilidade (%)	COA (%)	Confiabilidade (%)	Indisponibilidade (Hr)
1	99.9802105740	99.9990103942	83.32379	1.7335537
2	99.9752816788	99.9987637708	83.32379	2.1653249
3	99.9703529941	99.9985172217	83.32379	2.5970777
4	99.9802105112	99.9993403131	74.64096	1.7335592
5	99.9752814727	99.9991758335	74.64096	2.1653430
6	99.9703527150	99.9990114806	74.64096	2.5971022
7	99.9802105315	99.9995052693	71.85900	1.7335574
8	99.9752814641	99.9993818679	71.85900	2.1653437
9	99.9703527293	99.9992586225	71.85900	2.5971009
10	99.9802105848	99.9996042640	70.40507	1.7335528
11	99.9752814559	99.9995054868	70.40507	2.1653445
12	99.9703527412	99.9994069091	70.40507	2.5970999
13	99.9802106381	99.9996702693	69.96875	1.7335481
14	99.9752814477	99.9995878981	69.96875	2.1653452
15	99.9703527535	99.9995057689	69.96875	2.5970988
16	99.9802106914	99.9997174235	69.86202	1.7335434
17	99.9752814396	99.9996467622	69.86202	2.1653459
18	99.9703527658	99.9995763849	69.86202	2.5970977
19	99.9802107447	99.9997527959	69.84156	1.7335388
20	99.9752814314	99.9996909092	69.84156	2.1653466
21	99.9703527781	99.9996293483	69.84156	2.5970966
22	99.9802107980	99.9997803137	69.83897	1.7335341
23	99.9752814232	99.9997252448	69.83897	2.1653473
24	99.9703527903	99.9996705435	69.83897	2.5970956
25	99.9802108512	99.9998023332	69.83893	1.7335294

Tabela 65 – Resultados dos Cenários (cont.)

IaaS	Disponibilidade (%)	COA (%)	Confiabilidade (%)	Indisponibilidade (Hr)
26	99.9752814150	99.9997527125	69.83893	2.1653480
27	99.9703528026	99.9997035009	69.83893	2.5970945
28	99.9802113841	99.9999015677	69.83893	1.7334827
29	99.9752813332	99.9998762947	69.83893	2.1653552
30	99.9703529254	99.9998518429	69.83893	2.5970837
31	99.9802119170	99.9999348235	69.83893	1.7334361
32	99.9752812515	99.9999174615	69.83893	2.1653624
33	99.9703530482	99.9999013312	69.83893	2.5970730
34	99.9841683332	99.9992082977	83.56029	1.3868540
35	99.9802252441	99.9990110402	83.56029	1.7322686
36	99.9762823434	99.9988138620	83.56029	2.0776667
37	99.9841682747	99.9994722018	77.50169	1.3868591
38	99.9802251148	99.9993406663	77.50169	1.7322799
39	99.9762822220	99.9992092594	77.50169	2.0776774
40	99.9841682633	99.9996041480	75.89472	1.3868601
41	99.9802250868	99.9995054750	75.89472	1.7322824
42	99.9762822026	99.9994069449	75.89472	2.0776790
43	99.9841682654	99.9996833203	75.23019	1.3868600
44	99.9802250589	99.9996043546	75.23019	1.7322848
45	99.9762822183	99.9995255701	75.23019	2.0776777
46	99.9841682674	99.9997361022	75.07306	1.3868598
47	99.9802250310	99.9996702697	75.07306	1.7322873
48	99.9762822339	99.9996046562	75.07306	2.0776763
49	99.9841682695	99.9997738038	75.04430	1.3868596
50	99.9802250031	99.9997173479	75.04430	1.7322897

Tabela 66 – Resultados dos Cenários (cont.)

IaaS	Disponibilidade (%)	COA (%)	Confiabilidade (%)	Indisponibilidade (Hr)
51	99.9762822495	99.9996611485	75.04430	2.0776749
52	99.9841682716	99.9998020803	75.04092	1.3868594
53	99.9802249752	99.9997526531	75.04092	1.7322922
54	99.9762822651	99.9997035197	75.04092	2.0776736
55	99.9841682737	99.9998240733	75.04083	1.3868592
56	99.9802249473	99.9997801096	75.04083	1.7322946
57	99.9762822807	99.9997364768	75.04083	2.0776722
58	99.9841682758	99.9998416680	75.04083	1.3868590
59	99.9802249194	99.9998020720	75.04083	1.7322971
60	99.9762822963	99.9997628440	75.04083	2.0776708
61	99.9841682966	99.9999208497	75.04083	1.3868572
62	99.9802246404	99.9999008260	75.04083	1.7323215
63	99.9762824525	99.9998815396	75.04083	2.0776572
64	99.9841683174	99.9999472505	75.04083	1.3868554
65	99.9802243615	99.9999336510	75.04083	1.7323459
66	99.9762826087	99.9999211568	75.04083	2.0776435
67	99.9868068679	99.9993402333	84.41078	1.1557184
68	99.9835209184	99.9991758441	84.41078	1.4435675
69	99.9802351371	99.9990115295	84.41078	1.7314020
70	99.9868068059	99.9995601337	80.10098	1.1557238
71	99.9835207902	99.9994505016	80.10098	1.4435788
72	99.9802350175	99.9993409882	80.10098	1.7314125
73	99.9868067816	99.9996700787	79.12699	1.1557259
74	99.9835207205	99.9995878144	79.12699	1.4435849
75	99.9802349837	99.9995057112	79.12699	1.7314154

Tabela 67 – Resultados dos Cenários (cont.)

IaaS	Disponibilidade (%)	COA (%)	Confiabilidade (%)	Indisponibilidade (Hr)
76	99.9868067574	99.9997360409	78.79496	1.1557281
77	99.9835206510	99.9996701883	78.79496	1.4435910
78	99.9802349500	99.9996045383	78.79496	1.7314184
79	99.9868067331	99.9997800117	78.73120	1.1557302
80	99.9835205814	99.9997250925	78.73120	1.4435971
81	99.9802349164	99.9996704175	78.73120	1.7314213
82	99.9868067089	99.9998114159	78.72266	1.1557323
83	99.9835205119	99.9997642999	78.72266	1.4436032
84	99.9802348827	99.9997174692	78.72266	1.7314243
85	99.9868066846	99.9998349660	78.72235	1.1557344
86	99.9835204423	99.9997936968	78.72235	1.4436093
87	99.9802348490	99.9997527537	78.72235	1.7314272
88	99.9868066604	99.9998532801	78.72235	1.1557365
89	99.9835203728	99.9998165533	78.72235	1.4436153
90	99.9802348154	99.9997801935	78.72235	1.7314302
91	99.9868066361	99.9998679289	78.72235	1.1557387
92	99.9835203032	99.9998348315	78.72235	1.4436214
93	99.9802347817	99.9998021420	78.72235	1.7314331
94	99.9868063937	99.9999337819	78.72235	1.1557599
95	99.9835196078	99.9999168923	78.72235	1.4436824
96	99.9802344450	99.9999008176	78.72235	1.7314626
97	99.9868061512	99.9999556521	78.72235	1.1557812
98	99.9835189123	99.9999440141	78.72235	1.4437433
99	99.9802341083	99.9999335972	78.72235	1.7314921