UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Métodos de análise de decisão multicritério para a seleção de recursos em ambientes IoT

Luiz Henrique Nunes

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Luiz Henrique Nunes

Métodos de análise de decisão multicritério para a seleção de recursos em ambientes IoT

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Júlio César Estrella

USP – São Carlos
Fevereiro de 2019

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

N972m Nunes, Luiz Henrique
 Métodos de análise de decisão multicritério para
a seleção de recursos em ambientes IoT / Luiz
Henrique Nunes; orientador Júlio Cezar Estrella. --
São Carlos, 2018.
 162 p.

 Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2018.

 1. Seleção de Recursos. 2. Análise de decisão
multicritério. 3. Internet das Coisas. I. Estrella,
Júlio Cezar, orient. II. Título.

Luiz Henrique Nunes

**Multicriteria decision analysis techniques for resources
selection in IoT environments**

Doctoral dissertation submitted to the Institute of
Mathematics and Computer Sciences – ICMC-USP, in
partial fulfillment of the requirements for the degree of
the Doctorate Program in Computer Science and
Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Prof. Dr. Júlio César Estrella

USP – São Carlos
February 2019

*Dedico esta tese
aos meus pais Francisco e Zoraide,
aos meus irmãos Luiz Renato, Vivian,
Patrícia e Priscila,
a minha noiva Geovana,
e a todos os meus amigos,
pelo amor e incentivo desde sempre.*

AGRADECIMENTOS

Agradeço aos meus amados pais, Francisco e Zoraide, e aos meus irmãos Luiz Renato, Vivian, Patrícia e Priscila, pelo carinho, incentivo e confiança em todos os momentos e decisões de minha vida.

A minha noiva Geovana, por todo o apoio, amor e compreensão incondicionais.

Ao meu orientador Prof. Júlio Cezar Estrella e ao Prof. Alexandre Delbem que forneceram por meio de suas orientações o alicerce necessário para a construção desta tese.

Aos docentes e amigos que frequentaram o LaSDPC neste período contribuindo diretamente por meio de discussões ou indiretamente em momentos de risadas e boas conversas.

A todos vocês, muito obrigado!

*“Aquele que deu o melhor de si para sua época
viveu para todas as épocas.”
(Johann von Schiller)*

RESUMO

NUNES, L.H. **Métodos de análise de decisão multicritério para a seleção de recursos em ambientes IoT**. 2019. 162 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

A Internet das coisas é constituída de “objetos” que possuem pequenos sensores e atuadores capazes de interagir com o ambiente. Tais “objetos” ou “coisas” estão interconectados entre si e com acesso à Internet por meio de redes com e sem fio. A combinação entre os dispositivos embarcados com sensores e o acesso à Internet possibilita a comunicação dos recursos do mundo físico com o espaço cibernético, desempenhando um papel fundamental na resolução de muitos desafios encontrados na sociedade atual. Porém, a maioria das aplicações existentes são dedicadas a resolver problemas específicos utilizando tais recursos apenas em redes internas, limitando a real capacidade da Internet das Coisas. Diversos trabalhos na literatura propõem a reutilização de tais recursos em forma de serviço por meio de modelos como Dados como Serviço e Sensoriamento como Serviço. Neste contexto, em que potencialmente milhares de recursos podem transferir dados semelhantes de aplicações diferentes, a utilização de técnicas que possam selecionar recursos de forma sensível a contexto torna-se imprescindível. Nesta tese são propostos um conjunto de métodos para melhorar a relação custo-benefício na seleção de recursos em ambientes IoT, auxiliando na tomada de decisão durante a seleção dos recursos que serão ofertados como serviço. Os resultados obtidos por meio de estudos de caso, permitiram a comparação da qualidade da solução e do custo computacional das técnicas aplicadas na seleção de recursos em ambientes IoT, bem como o desenvolvimento de duas novas técnicas para a seleção de recursos, denominadas *Elimination Sort* e *Fast Elimination Sort*.

Palavras-chave: Seleção de recursos, análise de decisão multicritério, Internet das Coisas.

ABSTRACT

NUNES, L.H. **Multicriteria decision analysis techniques for resources selection in IoT environments**. 2019. 162 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

The Internet of Things is composed of “ objects ” which have small sensors and actuators capable of interacting with the environment. Such “ objects ” or “ things ” are interconnected with each other and has access to the Internet through wired and wireless networks. The combination of embedded devices with sensors and access to the Internet become it possible to communicate the resources of the physical world with the cyberspace, playing a key role in solving many challenges found in today’s society. However, most existing applications solves a specific problem using its resources just for own purpose, limiting the actual ability of the Internet of Things. Several works propose the reuse of such resources through service models such as Data as Service and Sensing as a Service. In this context, where thousands of resources can transfer similar data from different applications, the use of techniques that can select these features in a context-sensitive way becomes essential. In this thesis, a set of methods to improve the cost-benefit of the process of selection of resources in IoT environments is proposed to support the decision making during resource selection that will be offered as a service. The results obtained through a case study allowed the comparison of the solution quality and the computational cost of the techniques applied for resource selection in IoT environments, as well as the development of two new techniques for the selection of resources called *Elimination Sort* and *Fast Elimination Sort*.

Keywords: Resource Selection, Multiple-Criteria Decision Analysis, Internet of Things.

LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplo de composição de uma rede de sensores.	42
Figura 2 – Arquiteturas utilizadas em IoT	43
Figura 3 – Visão da Internet das Coisas	47
Figura 4 – Exemplo de abstração realizada pelos sensores virtuais	51
Figura 5 – Problema de otimização de custo-desempenho.	57
Figura 6 – Fronteira de Pareto para diferentes funções objetivo	58
Figura 7 – Síntese do funcionamento do algoritmo ES. Na primeira etapa o algoritmo TOPSIS é aplicado sobre os Q recursos disponíveis gerando uma sublista com $\mathbb{S}\mathbb{R} \times \mathbb{N}$ primeiros recursos classificados. As relações de dominância desta lista são computadas por meio do algoritmo FNDS, o qual gera a lista a ser retornada $\mathbb{S}$ com os $\mathbb{N}$ recursos localizados nas primeiras i fronteiras dominantes.	80
Figura 8 – Síntese do funcionamento do algoritmo FES. Na primeira etapa, o algoritmo TOPSIS é aplicado sobre os Q recursos disponíveis gerando duas sublistas. A primeira sublista contém os $\mathbb{N} \times (1 - \mathbb{P})$ primeiros recursos classificados e que serão utilizados no resultado final como $S1$. A segunda sublista contém os próximos $(\mathbb{N} \times \mathbb{P} \times \mathbb{S}\mathbb{R})$ recursos. As relações de dominância dessa sublista são computadas por meio do algoritmo FNDS, o qual gera a lista a ser retornada $\mathbb{S}\mathbb{R}$ com os $\mathbb{N} \times \mathbb{P}$ recursos localizados nas primeiras i fronteiras dominantes. A lista final $\mathbb{S}$ é formada pela união dos recursos contidos em $S1$ e $S2$ totalizando os $\mathbb{N}$ recursos.	81
Figura 9 – Fluxo utilizado para avaliar as técnicas de seleção aplicados no ambiente IoT	82
Figura 10 – Perda da qualidade da seleção de N recursos por meio da comparação da métrica ONVGR ideal com a obtida pelas técnicas MCDA (a) e pelo algoritmo FNDS (b) utilizado para computar a solução ideal.	83
Figura 11 – Valor da importância relativa das fronteiras de acordo com o número total de fronteiras da solução ótima. A importância relativa da fronteira é relacionada ao valor do seu índice e do número total de fronteiras.	83
Figura 12 – Módulos da ferramenta ViSIoT	88
Figura 13 – Interface ViSIoT	90
Figura 14 – Diagrama de sequência descrevendo o funcionamento da ferramenta ViSIoT.	93

Figura 15 – Organização do ambiente de experimentos formado por servidores com repositórios responsáveis por armazenar a descrição e o estado dos recursos disponíveis.	96
Figura 16 – Número de soluções disponíveis em cada fronteira.	98
Figura 17 – Gráfico em linha com a média ONVGR e o seu desvio padrão para dois critérios.	98
Figura 18 – Gráfico em linha com a média ONVGR e o seu desvio padrão para quatro critérios.	100
Figura 19 – Gráfico em linha com a média ONVGR e o seu desvio padrão para seis critérios.	101
Figura 20 – Gráfico em linha representando o <i>Crowding Distance</i> médio e o seu desvio padrão para dois critérios.	102
Figura 21 – Gráfico em linha representando o <i>Crowding Distance</i> médio e o seu desvio padrão para quatro critérios.	103
Figura 22 – Gráfico em linha representando o <i>Crowding Distance</i> médio e o seu desvio padrão para seis critérios.	104
Figura 23 – Qualidade da seleção de 2.095 (a) e 20.955 (b) recursos considerando dois critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A relação ideal deste cenário é representada pelo algoritmo FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras que uma solução deve apresentar auxiliando a comparação dos algoritmos TOPSIS, FES e ES com a solução ideal.	112
Figura 24 – Qualidade da seleção de 2.095 (a) e 20.955 (b) recursos considerando quatro critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A relação ideal deste cenário é representada pelo algoritmo FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras que uma solução deve apresentar auxiliando a comparação dos algoritmos TOPSIS, FES e ES com a solução ideal.	113
Figura 25 – Qualidade da seleção de 2.095 (a) e 20.955 (b) recursos considerando seis critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A relação ideal deste cenário é representada pelo algoritmo FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras que uma solução deve apresentar auxiliando a comparação dos algoritmos TOPSIS, FES e ES com a solução ideal.	114

Figura 26 – Tempo total em segundos necessário para a seleção de 2.095 (a) e 20.955 (b) recursos considerando dois critérios, em que quanto menor o tempo melhor o desempenho.	114
Figura 27 – Tempo total em segundos necessário para a seleção de 2.095 (a) e 20.955 (b) recursos considerando quatro critérios, em que quanto menor o tempo melhor o desempenho.	115
Figura 28 – Tempo total em segundos necessário para a seleção de 2.095 (a) e 20.955 (b) recursos considerando seis critérios, em que quanto menor o tempo melhor o desempenho.	116
Figura 29 – Custo-benefício da seleção de 2.095 e 20.955 recursos considerando dois (a), quatro (b) e seis (c) critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. As linhas pontilhadas indicam quais são os algoritmos que apresentam as melhores relações custo-benefício em cada cenário.	117
Figura 30 – Qualidade da seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.	119
Figura 31 – Qualidade da seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.	119

Figura 32 – Qualidade da seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário. 120

Figura 33 – Qualidade da seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário. 120

Figura 34 – Qualidade da seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário. 122

Figura 35 – Qualidade da seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário. 122

Figura 36 – Tempo total em segundos necessário para a seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.	123
Figura 37 – Tempo total em segundos necessário para a seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho. . .	124
Figura 38 – Tempo total em segundos necessário para a seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho. . .	125
Figura 39 – Tempo total em segundos necessário para a seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho. . .	125
Figura 40 – Tempo total em segundos necessário para a seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.	126
Figura 41 – Tempo total em segundos necessário para a seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho. . .	126
Figura 42 – Custo-benefício da seleção de 2.095 e 20.955 recursos distribuídos aleatoriamente em em dois (a), quatro (b) e oito (c) servidores considerando dois critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. Os símbolos representados pela cor cinza correspondem ao custo-benefício do algoritmo Pareto executado em um único servidor para a seleção de 2.095 e 20.955 recursos, servindo como base de comparação com os algoritmos TOPSIS, FES, ES e Pareto.	127

Figura 43 – Custo-benefício da seleção de 2.095 e 20.955 recursos distribuídos aleatoriamente em em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. Os símbolos representados pela cor cinza correspondem ao custo-benefício do algoritmo Pareto executado em um único servidor para a seleção de 2.095 e 20.955 recursos, servindo como base de comparação com os algoritmos TOPSIS, FES, ES e Pareto.	128
Figura 44 – Custo-benefício da seleção de 2.095 e 20.955 recursos distribuídos aleatoriamente em em dois (a), quatro (b) e oito (c) servidores considerando seis critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. Os símbolos representados pela cor cinza correspondem ao custo-benefício do algoritmo Pareto executado em um único servidor para a seleção de 2.095 e 20.955 recursos, servindo como base de comparação com os algoritmos TOPSIS, FES, ES e Pareto.	128
Figura 45 – Qualidade da seleção de 10, 100, 1.000, 10.000 recursos considerando dois critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o valor da importância da fronteira de acordo com o número de recursos a serem selecionados.	130
Figura 46 – Qualidade da seleção de 10, 100, 1.000, 10.000 recursos considerando quatro critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o valor da importância da fronteira de acordo com o número de recursos a serem selecionados.	131
Figura 47 – Qualidade da seleção de 10, 100, 1.000, 10.000 recursos considerando seis critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o valor da importância da fronteira de acordo com o número de recursos a serem selecionados.	132
Figura 48 – Custo-benefício da seleção de 10, 100, 1.000, 10.000 e 100.000 recursos considerando dois, quatro e seis critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução.	132
Figura 49 – Comparação do tempo em segundos necessário para executar o algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados em 1 , 2 , 4 servidores e 8 servidores para dois critérios.	160

Figura 50 – Comparação do tempo em segundos necessário para executar o algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados em 1 , 2 , 4 servidores e 8 servidores considerando-se quatro critérios. . . .	161
Figura 51 – Comparação do tempo em segundos necessário para executar o algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados em 1 , 2 , 4 servidores e 8 servidores considerando-se seis critérios.	162

LISTA DE ALGORITMOS

Algoritmo 1 – Algoritmo Fast non-dominated sort	59
Algoritmo 2 – Elimination-Selection	80
Algoritmo 3 – Fast Elimination-Selection	81

LISTA DE ARQUIVOS

Arquivo 1 – Exemplo de arquivo JSON do OpenWeatherMap	88
Arquivo 2 – Exemplo de descrição de um sensor virtual no <i>middleware</i> GSN	89
Arquivo 3 – Arquivo de configuração da aplicação cliente	91

LISTA DE TABELAS

Tabela 1 – Síntese das características dos middlewares e projetos utilizados para a provisão de recursos como serviços em ambientes IoT.	56
Tabela 2 – Síntese das principais características dos trabalhos relacionados.	77
Tabela 3 – Descrição das variáveis, tipos e restrições utilizadas para a nos algoritmos ES e FES	79
Tabela 4 – Cenário simplificado com a descrição das condições climáticas obtidas na plataforma OpenWeatherMap para o cultivo do milho.	86
Tabela 5 – Fatores e níveis utilizados nos experimentos	95
Tabela 6 – Correspondência entre o número de critérios dos experimentos e as restrições climáticas. A Tabela 4 contém a descrição de cada sigla das restrições climáticas.	95
Tabela 7 – Configuração do hardware utilizado para a realização dos experimentos . . .	97
Tabela 8 – Descrição dos fatores e níveis dos experimentos.	110

LISTA DE ABREVIATURAS E SIGLAS

AHP	<i>Analytic Hierarchy Process</i>
API	<i>Application Programming Interface</i>
ASBP	<i>Sensing Algorithm with Belief Propagation</i>
CD	<i>Crowding Distance</i>
CoAP	<i>Constrained Application Protocol</i>
CoT	<i>Cloud of Things</i>
CRUD	<i>Create, Read, Update, Delete</i>
DaaS	<i>Data as Service</i>
DHT	<i>Distributed Hash Table</i>
DNS	<i>Domain Name System</i>
ES	<i>Elimination Sort</i>
ESB	<i>Enterprise Service Bus</i>
FES	<i>Fast Elimination Sort</i>
FNDS	<i>Fast Non-Dominated Sort</i>
GPS	<i>Global Positioning System</i>
GSN	<i>Global Sensor Networks</i>
HTML	<i>HyperText Markup Language</i>
HTTP	<i>Hypertext Transfer Protocol</i>
IDC	<i>International Data Corporation</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
IoT	<i>Internet of Things</i>
JSON	<i>JavaScript Object Notation</i>
LaSDPC	Laboratório de Sistemas Distribuídos e Programação Concorrente
MCDA	<i>Multiple-criteria decision analysis</i>
NFC	<i>Near Field Communication</i>
NIST	<i>National Institute of Standards and Technology</i>
ONVGR	<i>Overall Nondominated Vector Generation Ratio</i>
P2P	<i>Peer-to-peer</i>
PQA	Perda de Qualidade Acumulada
PQR	Perda de Qualidade Relativa
PROMETHEE	<i>Preference Ranking Organization METHod for Enrichment of Evaluations</i>

QoI	<i>Quality of information</i>
QoS	<i>Quality of service</i>
RADV	<i>Randomized and Asynchronous Distributed Virtualization</i>
RFID	<i>Radio-Frequency IDentification</i>
SAW	<i>Simple Additive Weighting</i>
SIoT	<i>Social Internet of Things</i>
SLA	<i>Service-level agreement</i>
SOA	<i>Service-oriented architecture</i>
SoFA	<i>Self-organizing Formin procedure of ASC</i>
SQL	<i>Structured Query Language</i>
SSNO	<i>Semantic Sensor Network Ontology</i>
SWE	<i>Sensor Web Enablement</i>
TASI	<i>Threshold Algorithm for Sensor Information</i>
TDD	<i>Thing Description Document</i>
TOPSIS	<i>Technique for Order of Preference by Similarity to Ideal Solution</i>
URI	<i>Uniform Resource Identifier</i>
VIKOR	<i>Vlsekriterijumska Optimizacija I Kompromisno Resenje</i>
ViSIoT	<i>Visual Search for Internet of Things</i>
XML	<i>eXtensible Markup Language</i>

SUMÁRIO

1	INTRODUÇÃO	35
1.1	Contextualização	36
1.2	Motivação e Objetivo	37
1.3	Estrutura da Monografia	39
2	REFERENCIAL TEÓRICO	41
2.1	<i>Internet das Coisas</i>	41
2.1.1	<i>Redes de Sensores</i>	42
2.1.2	<i>Arquitetura de sistemas IoT</i>	43
2.1.2.1	<i>Técnicas e estratégias de processamento de dados em IoT</i>	44
2.1.2.2	<i>Desafios da IoT</i>	45
2.1.3	<i>Interoperabilidade em IoT</i>	46
2.1.4	<i>Gerenciamento dos dados em IoT</i>	48
2.1.4.1	<i>Coleta e Análise dos Dados em IoT</i>	48
2.1.4.2	<i>Big Data</i>	49
2.1.4.3	<i>Anotações Semânticas</i>	50
2.1.4.4	<i>Sensores Virtuais</i>	51
2.1.5	<i>Computação em Nuvem e Internet das Coisas</i>	52
2.1.6	<i>Middleware</i>	54
2.2	Otimização Multiobjetivo	57
2.2.1	<i>Relações de dominância de Pareto</i>	58
2.2.2	<i>Técnicas de Análise de Decisão Multicritérios</i>	60
2.2.3	<i>Métricas de Avaliação</i>	61
2.2.4	<i>Considerações Finais</i>	62
3	TRABALHOS RELACIONADOS	63
3.1	Infraestruturas para a Seleção de Recursos	63
3.2	Técnicas de Seleção de Recursos	71
3.3	Considerações Finais	78
4	MÉTODOS PARA A TOMADA DE DECISÃO MULTICRITÉRIO EM IOT	79
4.1	Algoritmos	79
4.2	Metodologia de Avaliação	81

4.3	Indicadores de Desempenho	82
4.4	Considerações Finais	84
5	ESTUDO EXPLORATÓRIO DAS INFRAESTRUTURAS E TÉCNICAS DE SELEÇÃO DE RECURSOS EM AMBIENTES IOT	85
5.1	Estudo exploratório das infraestruturas de seleção de recursos	86
5.1.1	<i>Ferramenta ViSloT</i>	<i>87</i>
5.1.1.1	<i>Core</i>	<i>87</i>
5.1.1.2	<i>Repositório</i>	<i>87</i>
5.1.1.3	<i>Interface</i>	<i>89</i>
5.1.1.4	<i>Seletor</i>	<i>89</i>
5.1.1.5	<i>Conversor</i>	<i>89</i>
5.1.1.6	<i>Ambiente alvo</i>	<i>91</i>
5.1.2	<i>Funcionamento</i>	<i>92</i>
5.2	Estudo exploratório das técnicas de seleção de recursos	94
5.2.1	<i>Configuração do ambiente</i>	<i>95</i>
5.2.2	<i>Experimento 1</i>	<i>96</i>
5.2.3	<i>Experimento 2</i>	<i>97</i>
5.2.3.1	<i>Resultados da qualidade em termos de capacidade</i>	<i>98</i>
5.2.3.2	<i>Resultados da qualidade em termos de diversidade</i>	<i>101</i>
5.2.3.3	<i>Análise dos Resultados</i>	<i>105</i>
5.3	Considerações Finais	106
6	AVALIAÇÃO EXPERIMENTAL DAS TÉCNICAS FES E ES	109
6.1	Configuração dos Experimentos	109
6.2	Experimento 1	110
6.2.1	<i>Qualidade</i>	<i>111</i>
6.2.2	<i>Tempo</i>	<i>113</i>
6.2.3	<i>Análise dos Resultados</i>	<i>115</i>
6.3	Experimento 2	118
6.3.1	<i>Qualidade</i>	<i>118</i>
6.3.2	<i>Tempo</i>	<i>123</i>
6.3.3	<i>Análise dos Resultados</i>	<i>127</i>
6.4	Experimento 3	129
6.4.1	<i>Qualidade</i>	<i>129</i>
6.4.2	<i>Análise dos Resultados</i>	<i>131</i>
6.5	Considerações Finais	134
7	CONCLUSÕES	135
7.1	Contribuições	136

7.2	Publicações	137
7.3	Dificuldades	138
7.4	Trabalhos Futuros	139
REFERÊNCIAS		141
APÊNDICE A	SAW	155
APÊNDICE B	VIKOR	157
APÊNDICE C	COMPARAÇÃO ENTRE A EXECUÇÃO DO ALGO- RITMO DE PARETO NA MEMÓRIA PRINCIPAL E NO BANCO DE DADOS	159

INTRODUÇÃO

Ao longo da última década, o custo de produção de dispositivos caiu e a capacidade de processamento aumentou, permitindo cada vez mais sua adoção tanto que o número de dispositivos conectados à *Internet* ultrapassou o número da população mundial (aproximadamente 6.3 bilhões de pessoas) entre os anos 2008 e 2009 ([RESEARCH, 2011](#)), iniciando assim a era da *Internet of Things* (IoT).

Atualmente, não há um consenso sobre o que é a IoT, sendo representada como uma rede global de “coisas”. Uma das definições mais aceitas indica que o termo IoT refere-se a cenários em que a conectividade de rede e a capacidade de computação sejam estendidas a “coisas” (objetos, sensores e itens comuns), permitindo que tais “coisas” gerem, troquem e consumam dados com o mínimo de intervenção humana ([ROSE; ELDRIDGE; CHAPIN, 2015](#)).

Diversas projeções e correlações estão sendo feitas para os próximos anos a respeito do impacto da IoT na economia e na *Internet*. Segundo a empresa Gartner, em 2017 cerca de 8.4 bilhões de dispositivos estavam conectados na *Internet* ([GARTNER, 2017](#)). Graças às facilidades e benefícios fornecidos por dispositivos e serviços em IoT, como carros conectados, casas e roupas inteligentes, a Cisco estima que em 2020 este número esteja por volta dos 20 bilhões de coisas conectadas à *Internet* ([CISCO, 2018](#)).

Com relação ao lado econômico, para o *International Data Corporation* (IDC) ([IDC, 2017](#)) a IoT será responsável por movimentar um mercado com valores em torno de \$772.5 bilhões em 2018, distribuídos em sua maioria entre:

- *Hardware*: refere-se aos gastos com a implantação de sensores e atuadores para estabelecer ambientes de IoT. Além disso, esta categoria abrange o desenvolvimento de melhorias para os *hardwares* existentes bem como a implantação de mecanismos de segurança;
- *Software*: reflete as despesas com o desenvolvimento de *software* utilizados para estabelecer a conexão entre os dispositivos implantados em um ambiente IoT e as aplicações finais;

- **Serviços:** correspondem à quantia gasta para o desenvolvimento de serviços em tais ambientes, orquestrando os dispositivos de *hardware* e *software* para desenvolver uma solução.

1.1 Contextualização

IoT é um ecossistema que conecta objetos físicos com a *Internet*, juntando o mundo real com o espaço cibernético, permitindo assim o desenvolvimento de novos serviços e aplicações (NUNES *et al.*, 2016b), como por exemplo:

- **Logística:** Alimentos perecíveis tais como peixes, vegetais e carne fresca podem deteriorar-se rapidamente quando não são armazenados de maneira adequada. É importante assegurar durante todo o processo de logística que envolve a produção, o transporte e a venda de um determinado alimento as condições ideais como a temperatura e a umidade para evitar prejuízos e garantir a qualidade do produto. A IoT vem sendo utilizada para monitorar todo o processo de logística de um produto e assim garantir que não ocorram violações em nenhuma etapa do processo (DADA; THIESSE, 2008; ATZORI; IERA; MORABITO, 2010).
- **Bem estar e Saúde:** doenças crônicas ou condições especiais de saúde como doenças de coração, derrame, câncer, diabetes, artrite podem ter seus sintomas controlados com acompanhamento médico. De acordo com o centro de prevenção e controle de doenças dos Estados Unidos, as doenças crônicas foram responsáveis por 48% de todas as mortes registradas em 2013. A utilização dos sensores disponíveis em *smartphones* ou os que são acoplados em peças de vestuário podem auxiliar nos mais variados tipos de cuidados médicos tais como o sensoriamento do nível de glicose no sangue, da temperatura, da pressão e dos batimentos cardíacos, possibilitando assim a elaboração de diagnósticos em tempo real (ISLAM *et al.*, 2015).
- **Trânsito:** a falta de vagas de estacionamento e os congestionamentos são alguns problemas que diminuem a qualidade de vida das pessoas que vivem em grandes cidades. A utilização de sensores e atuadores de baixo custo pode auxiliar no gerenciamento do trânsito por meio do monitoramento das vagas disponíveis e sincronização de semáforos (DADA; THIESSE, 2008).
- **Meio Ambiente:** problemas ambientais como poluição da água e do ar impactam em vários aspectos da qualidade de vida dos seres humanos. Sensores e atuadores podem ser utilizados para monitorar as condições do solo, da água e do ar, com o intuito de detectar condições de risco à saúde ou a integridade dos indivíduos que vivem em tal ambiente (ROBLES *et al.*, 2014; MANNA; BHUNIA; MUKHERJEE, 2014).

É esperado que IoT desempenhe um papel fundamental na resolução de muitos desafios encontrados na sociedade atual (KORESHOFF; ROBERTSON; LEONG, 2013). Contudo, observa-se que a maioria das aplicações existentes são dedicadas a resolver um problema específico, em que um conjunto de recursos (dispositivos físicos ou virtuais) é utilizado para coletar e processar dados de um determinado ambiente. Isso resulta em um fluxo fechado de informações, limitando a capacidade real da utilização dos conceitos de IoT, uma vez que não há o compartilhamento das informações coletadas (WIRTZ; WEHRLE, 2013).

A colaboração entre os diferentes ambientes IoT pode auxiliar na elaboração de soluções para esses problemas ao captar dados reutilizáveis de diferentes fontes. Por outro lado, tal colaboração torna mais complexo o gerenciamento dos dispositivos e a manipulação dos dados produzidos por eles (NUNES *et al.*, 2016b).

A integração entre a Computação em Nuvem e a IoT, denominada *Cloud of Things* (CoT), fornece toda a infraestrutura e poder computacional necessária para auxiliar o gerenciamento de tais recursos e o processamento dos dados. A reutilização desses recursos fornece oportunidades sem precedentes para o desenvolvimento de serviços e aplicações (ZHANG *et al.*, 2013; AAZAM; HUH, 2014).

1.2 Motivação e Objetivo

O crescimento rápido do número e dos tipos de dispositivos torna difícil a reutilização dos seus dados por terceiros, uma vez que a princípio, estão armazenados de forma local ou em uma nuvem privada. Soluções de *middleware* como OpenIoT¹, GSN² e Irisnet³ auxiliam a visão da CoT, abstraindo a comunicação entre as “coisas” que passam a ser vistas como recursos e serviços. Tal mediação permite o acesso, o processamento e a análise destes recursos e de seus dados.

Por exemplo, pulseiras inteligentes são equipadas com diversos sensores visando melhorar a qualidade de vida dos seus usuários por meio do monitoramento de diferentes dados como a quantidade de passos dados em um dia, a qualidade do sono e da alteração dos batimentos cardíacos. Os dados armazenados localmente ou em uma nuvem privada são analisados e permitem a emissão de alertas e recomendações visando evitar condições como o sedentarismo, estresse ou até mesmo um problema de saúde iminente.

Além do propósito inicial a que foram previstos, tais dados poderiam ser reutilizados por terceiros como empresas ou pesquisadores. Empresas que gerenciam planos de saúde podem procurar por pessoas que possuem hábitos saudáveis a fim de fornecer descontos, enquanto pesquisadores do governo buscam pessoas que pertençam a um certo grupo de risco para realizar

¹ OpenIoT project - <http://openiot.eu/>

² GSN project - <http://sourceforge.net/projects/gsn/>

³ Irisnet project - <http://www.intel-iris.net/>

ações médicas preventivas.

Uma pesquisa encomendada pela plataforma de serviços de dados automotivos Otonomo, indicou que grande parte dos consumidores estão dispostos a compartilhar os dados coletados pelos seus carros em troca de serviços, tais como alertas para condições perigosas de direção, detecção de falhas de componentes automotivos, tempos de resposta mais rápido em caso de acidentes e melhoria da qualidade e segurança das estradas com base no *feedback* do seu carro (DAWS, 2018).

Devido à natureza dinâmica destes dispositivos e suas finalidades originais, os dados coletados por um recurso podem não atender as necessidades de um usuário diferente de seus proprietários. A descoberta de recursos é um desafio importante a ser superado no contexto de IoT, em um processo que permite que as partes interessadas possam explorar e selecionar recursos (por exemplo, dispositivos IoT ou serviços) a serem incorporados em aplicações. Idealmente, o processo de descoberta de recursos deve selecionar os mais adequados a um serviço baseado nas restrições impostas pela parte interessada.

Diversos trabalhos, como Gao, Ali e Mileo (2014), Perera *et al.* (2014b), Khodadadi, Dastjerdi e Buyya (2015) propõem o uso de computação livre de contexto e *Multiple-criteria decision analysis* (MCDA) para auxiliar a descoberta de recursos. A computação livre de contexto refere-se à capacidade de usar as informações de contexto para caracterizar os objetos e relacioná-los com seus dados (PERERA *et al.*, 2014b). MCDA visa auxiliar a tomada de decisão em problemas que envolvem múltiplos critérios de acordo com os objetivos e restrições impostas (TZENG; HUANG, 2011).

As técnicas propostas nestes trabalhos agem como caixas pretas, recebendo um conjunto de parâmetros de entrada, que são utilizados para analisar e selecionar os recursos disponíveis de acordo com suas classificações (ex. selecionar um grupo de 100 pessoas dentre 10.000 disponíveis) (PERERA; VASILAKOS, 2016). Para realizar a comparação entre estas técnicas geralmente considera-se o tempo necessário para computar a solução e negligencia-se se a solução proposta é adequada para a aplicação final. Em outras palavras, as avaliações destas técnicas consideram apenas o custo computacional e desprezam o benefício obtido em termos de qualidade.

Uma das maneiras de mensurar a qualidade de uma solução é por meio da sua comparação com a solução ótima. Para problemas em que múltiplos critérios são utilizados, a solução ótima pode ser encontrada por intermédio das relações de dominância (COLLETTE; SIARRY, 2004). Embora a utilização das relações de dominância permitam identificar a solução ótima em problemas com múltiplos critérios, geralmente o seus custos computacionais em termos de tempo e recursos inviabilizam a sua utilização em diversos cenários (DEB, 2001; FANG *et al.*, 2008; BAO *et al.*, 2017).

Nesta tese, o custo é considerado como o tempo necessário para se obter uma solução

enquanto o benefício é representado pela qualidade da solução. Desta maneira, a hipótese geral deste trabalho baseia-se na seguinte afirmação:

É possível aproximar o custo benefício do processo de seleção de recursos em ambientes IoT ao ótimo para integrar diferentes fontes de dados e prover recursos buscando solucionar problemas do mundo real.

A partir da proposição desta hipótese, objetivo principal desta tese é

Propor um conjunto de métodos que melhore a relação custo-benefício do processo de seleção de recursos em ambientes IoT.

Para o desenvolvimento de tais métodos, os objetivos específicos a serem alcançados são:

1. Avaliar métricas que analisem o custo e a qualidade de seleção de algoritmos no contexto de IoT, a fim de determinar a relação entre o custo e o benefício de tais algoritmos, mensurados em termos de tempo computacional e qualidade da solução;
2. Propor algoritmos que melhorem a relação entre o custo-benefício do processo de seleção de recursos visando a diminuição do custo e o aumento da qualidade;
3. Averiguar como as diferentes formas de execução destes algoritmos (centralizada e distribuída) afetam a relação custo-benefício para se obter a solução final;
4. Fornecer os principais blocos para realizar a integração entre os diferentes *middlewares* aplicados para o gerenciamento de recursos em ambientes IoT.

Entre os benefícios esperados com o desenvolvimento desta tese estão uma metodologia para comparar e analisar as técnicas para a seleção de recursos em ambientes IoT, a proposta de novos algoritmos e a análise e comparação em termos qualitativos e quantitativos de tais algoritmos.

1.3 Estrutura da Monografia

Esta tese encontra-se organizada da seguinte maneira:

- No Capítulo 2 é apresentado todo o referencial teórico utilizado no escopo dessa pesquisa
- O Capítulo 3 descreve o estado da arte do tema de pesquisa por meio da descrição de trabalhos relacionados;

- No Capítulo 4 são descritas a metodologia proposta para avaliar as técnicas de MCDA e as técnicas para melhorar a relação custo-benefício do processo de seleção de recursos em ambientes IoT;
- O Capítulo 5 apresenta os estudos exploratórios voltados para a integração das infraestruturas e utilização das técnicas de seleção de recursos em ambientes IoT;
- O Capítulo 6 descreve os experimentos realizados para validar as técnicas de seleção propostas e os resultados obtidos;
- Por fim, o Capítulo 7 contém as conclusões do trabalho, as contribuições, as publicações derivadas da pesquisa, as dificuldades enfrentadas e os trabalhos futuros.

REFERENCIAL TEÓRICO

2.1 *Internet das Coisas*

IoT refere-se a uma visão na qual “coisas”, como objetos, lugares e ambientes estarão interconectadas por meio da Internet, que passará de uma rede global de computadores para uma rede global de objetos (KORESHOFF; ROBERTSON; LEONG, 2013). Estima-se “criar um mundo melhor para os seres humanos”, no qual os objetos presentes no dia a dia das pessoas serão capazes de captar suas necessidades e atuarem sem instruções explícitas (DOHR *et al.*, 2010).

Uma das definições mais aceitas sobre o que é IoT é a descrita pelos autores (VERMESAN *et al.*, 2011): “IoT visa permitir que as pessoas e as coisas estejam conectadas em qualquer horário e lugar com qualquer coisa ou qualquer um por meio de qualquer caminho/rede e serviço”.

IoT engloba uma vasta quantidade de tecnologias de *hardware* e de *software*, tais como, *Radio-Frequency IDentification* (RFID), *Near Field Communication* (NFC), redes de sensores *middlewares* e *frameworks* (WHITMORE; AGARWAL; XU, 2014).

Originalmente, estas tecnologias foram modeladas e desenvolvidas para solucionar problemas específicos, como por exemplo, detectar a presença de pessoas ou monitorar a temperatura de um ambiente. Porém, a evolução da IoT e dos dispositivos relacionados a ela tem permitido o desenvolvimento de serviços cada vez mais avançados, por meio do auxílio de *middlewares*, *frameworks* e *Application Programming Interface* (API) que adicionam a essas aplicações a capacidade de trocar informações e tomar decisões em ambientes heterogêneos (PERERA *et al.*, 2014a).

Segundo Gartner (2014), aproximadamente 25 bilhões de dispositivos estarão conectados a Internet em 2020 movimentando um mercado de aproximadamente \$263 bilhões de dólares. A maior parte destes dispositivos será utilizada por aplicações domésticas e pessoais, enquanto o

restante será utilizado por aplicações de manufatura, utilidade pública e transporte.

É esperado que os serviços oferecidos sejam sensíveis ao contexto considerando-se as preferências e as necessidades dos usuários, de acordo com as restrições impostas em seu ambiente original. Dessa maneira, soluções que ofereçam formas de busca e seleção de sensores em uma escala global e que considerem os diferentes contextos de implantação serão essenciais para o crescimento da IoT ([ROMER et al., 2010](#)).

2.1.1 Redes de Sensores

Nesta Seção será apresentada uma breve introdução às redes de sensores, que é uma das tecnologias que compõem a IoT. Redes de sensores são compostas por nós sensores e atuadores distribuídos geograficamente que trabalham de forma cooperativa por meio de redes com ou sem fio. Os dados coletados pelos nós sensores são enviados para um nó denominado *sink* que é o responsável por receber e enviá-los para um servidor (por exemplo, um servidor em uma nuvem). Os nós sensores são tipicamente dispersos em uma determinada região, com o intuito de observar um fenômeno de interesse, por exemplo, mudança de temperatura em uma floresta ([RAWAT et al., 2014](#)).

Geralmente, cada nó sensor pode ser equipado com vários tipos de sensores, tais como temperatura, pressão e umidade. Esses nós sensores são desenvolvidos para serem de baixo custo e normalmente possuem limitações computacionais, por exemplo, energia, memória e comunicação. Dessa forma, uma grande variedade de nós sensores podem ser implantados em grandes quantidades, com o intuito de monitorar, detectar e reportar eventos de forma rápida e coordenada ([MAHAPATRO; KHILAR, 2013](#)).

Figura 1 – Exemplo de composição de uma rede de sensores.

Fonte: Adaptada de [Perera et al. \(2014a\)](#).

As redes de sensores possuem características específicas conforme a área de sua aplicação. Por isso, é necessário que certas particularidades sejam levadas em consideração, tais

como a forma de endereçamento dos sensores, a capacidade de sintetização dos dados coletados, a mobilidade dos nós sensores, o consumo de energia da rede e sua topologia e a capacidade de colaboração entre os nós. Dentre as principais classes de aplicação das redes de sensores destacam-se aplicações para realizar a vigilância de áreas militares, o monitoramento de ambientes, a detecção de intrusos e a coleta de dados científicos (BORGES; VELEZ; LEBRES, 2014).

Na Figura 1, é representada a estrutura típica de uma rede de sensores compreendendo os componentes mais utilizados. A seta vermelha indica o sentido do fluxo da informação da esquerda para a direita, iniciando-se nos nós sensores e terminando em nuvem. Os nós sensores de baixo e alto nível são os responsáveis por coletar os dados do ambiente. Estes dados são enviados para um nó denominado *sink*, que é o responsável por gerenciar os nós sensores e suas informações. Estas informações são comumente enviadas para dispositivos com alta capacidade computacional para serem processadas. Por fim, tais dados são enviados para a nuvem, a partir de onde poderão ser compartilhados, armazenados e processados por outras aplicações (PERERA *et al.*, 2014a).

2.1.2 Arquitetura de sistemas IoT

A arquitetura da IoT é comumente considerada uma arquitetura de três camadas, tal como representado pela Figura 2.a, apresentando a camada de percepção, a camada de rede e a camada de aplicação (ZHANG; SUN; CHENG, 2012; AAZAM; HUH, 2014). Porém, alguns autores tais como (KHAN *et al.*, 2012; AAZAM; HUH, 2014) e (FERSI, 2015) adicionam as camadas de *middleware* e de negócio, tal como apresentada na Figura 2.b.

Figura 2 – Arquiteturas utilizadas em IoT

Fonte: Adaptada de Nunes *et al.* (2016b).

A camada de percepção é a camada de mais baixo nível nesta arquitetura. Esta camada é a responsável por captar e coletar as informações do ambiente real e trazê-las para o ambiente virtual. Sensores, rótulos de códigos de barras, dispositivos RFID, *Global Positioning System* (GPS) e câmeras concentram-se nesta camada (AAZAM; HUH, 2014).

A camada de rede é a responsável por transportar os dados da camada de percepção para serem processados. O meio de transmissão pode utilizar redes cabeadas ou redes sem fio tais como 3G, UMTS, *wifi*, *bluetooth*, infravermelho, ZigBee, etc., dependendo dos dispositivos sensores (KHAN *et al.*, 2012).

A camada de *middleware* recebe os dados da camada de rede. Seu objetivo é oferecer serviços e armazenar os dados recebidos. Seus serviços devem processar as informações e tomar decisões automáticas a partir dos seus resultados (KHAN *et al.*, 2012; AAZAM; HUH, 2014).

A camada de aplicação apresenta os dados provenientes da camada de rede ou de *middleware*. Esta camada deve preocupar-se em apresentar a informação de acordo com as especificações do usuário (AAZAM; HUH, 2014).

A camada de negócio é responsável pelo gerenciamento do sistema, incluindo suas aplicações e serviços. Ela é responsável por definir os modelos de negócio, os gráficos e fluxos de execução baseados nos dados recebidos da camada de aplicação. O sucesso de IoT depende do estabelecimento de bons modelos de negócio. A partir da análise desses resultados, esta camada auxiliará na determinação das ações futuras e das estratégias de negócio (KHAN *et al.*, 2012).

2.1.2.1 Técnicas e estratégias de processamento de dados em IoT

Vários sistemas voltados para a provisão de serviços em IoT baseiam seus funcionamentos em um conjunto de técnicas e estratégias bem definidos. De acordo com Romer *et al.* (2010), as principais técnicas e estratégias para gerenciamento de dados podem ser divididas em:

- **Modelos Push e Pull:** Neste modelo os dados coletados pelos sensores são consumidos por consultas de usuários. No modelo *push*, os dados coletados pelos sensores são enviados constantemente para os usuários, à medida que estes dados respeitem as condições estabelecidas na consulta realizada. Já no modelo *pull*, os dados coletados pelos sensores são armazenados em um repositório local, e enviados aos usuários apenas quando são requisitados em uma consulta.
- **Modelo Publish/Subscribe:** Para consultas que são realizadas constantemente, é viável estabelecer relações explícitas entre os sensores de interesse e os usuários, de tal forma que os dados de um determinado sensor sejam submetidos apenas as entidades que potencialmente os utilizem. O modelo *Publish/Subscribe* é uma técnica na qual os usuários se inscrevem para receber notificações de um determinado sensor à medida que ocorram mudanças em seu estado ou nos dados.

- **Mediadores:** Um mediador é um elemento conceitual que é utilizado entre os sensores e os usuários. Os mediadores tipicamente mantêm uma visão agregada dos sensores, de tal forma que, quando uma consulta é enviada a um mediador, ela pode ser respondida sem obter os dados de todos os sensores.
- **Índice invertido:** Os mediadores precisam ser capazes de encontrar rapidamente os sensores que podem responder a certas condições de uma consulta. Índice invertido é uma estrutura de dados capaz de otimizar a busca por um conjunto de sensores que produzem um certo tipo de dado.
- **Técnicas de Compressão:** Técnicas de compressão podem ser utilizadas para reduzir a quantidade de dados armazenados localmente ou transferidos em uma consulta de usuário.
- **Modelos de Predição:** A utilização de modelos de predição possibilita inferir a informação a respeito de um usuário ou de um sensor sem realizar a comunicação com elas. Estes modelos de predição devem sempre ser construídos utilizando-se dados já coletados por estes sensores, de tal maneira que seu comportamento seja moldado em um modelo matemático.
- **Técnicas de Pontuação e Classificação:** Técnicas de pontuação e classificação são utilizadas para apresentar aos usuários quais são os sensores mais relevantes de acordo com a consulta feita. A pontuação é utilizada para atribuir aos sensores um valor escalar proporcional a sua relevância à consulta. Já a classificação preocupa-se em ordenar os sensores de acordo com a sua pontuação, estabelecendo assim a ordem de prioridade entre eles.
- **Consulta Top-k:** A consulta *Top-k* visa acelerar o processo de seleção. Ao invés de analisar todo o conjunto que respeita as condições de uma determinada consulta, a consulta *Top-k* retorna aos usuários um subconjunto de tamanho k dos dados de sensores adequados à consulta.

2.1.2.2 Desafios da IoT

Atualmente, muitas pesquisas que envolvem IoT focam no desenvolvimento de tecnologias necessárias para o funcionamento de dispositivos com limitação de recursos tais como baterias de longa duração, utilização eficiente do espectro das redes sem fio e redes de sensores. Poucos trabalhos abordam os desafios encontrados para o desenvolvimento de aplicações e serviços voltados para IoT. Desta maneira, *middlewares* devem ser propostos e considerar os diferentes aspectos exigidos por estas aplicações e serviços tais como:

- **Interoperabilidade:** Muitos desenvolvedores de *hardware* desenvolvem seus dispositivos utilizando tecnologias e serviços que podem não ser acessadas por outros dispositivos

ou pessoas. A criação de padrões para IoT é extremamente importante para permitir a integração e a comunicação entre todos os objetos e dispositivos sensores (KHAN *et al.*, 2012).

- **Gerenciamento dos dados:** Com o aumento de dispositivos conectados à rede e constantemente trocando informações, o tamanho dos dados gerados e dos processos requeridos para manipular tais dados torna-se um grande desafio (BURANGE; MISALKAR, 2015).
- **Segurança e Privacidade:** Segurança em IoT pode ser dividida em dois grandes grupos: segurança operacional e segurança dos dados. Segurança Operacional refere-se à capacidade de manter o desempenho e não perder os dados, mesmo que parte da rede fique *offline*. Já segurança dos dados ou privacidade aborda os conceitos de confidencialidade dos dados e autenticação, impedindo que usuários não autorizados tenham acessos a dados privados (FERSI, 2015).
- **Escalabilidade:** Remete à capacidade de um sistema, rede, ou processo para trabalhar com uma carga de trabalho crescente de maneira eficiente, ou à sua capacidade de se modificar para acomodar tal carga. Este é um grande desafio em IoT, principalmente por ter um número grande de aplicações e dispositivos que transmitem e coletam dados concorrentemente (GOMES; RIGHI; COSTA, 2014).
- **Confiabilidade:** Compreende um mecanismo para estabelecer quão confiável é um dispositivo com base na sua reputação. Os aspectos de confiabilidade variam, podendo verificar a qualidade da informação fornecida por um dispositivo ou analisar a sua disponibilidade na rede (DAUBERT; WIESMAIER; KIKIRAS, 2015).

2.1.3 Interoperabilidade em IoT

Segundo o *Institute of Electrical and Electronics Engineers* (IEEE), interoperabilidade pode ser definida como a capacidade de um sistema ou produto para trabalhar com outros sistemas ou produtos sem nenhum esforço especial por parte do cliente, a qual só é possível por meio da criação de padrões. No contexto de IoT, o conceito de interoperabilidade abrange diferentes camadas que vão desde a comunicação dos dispositivos até a integração entre os dados gerados por tais dispositivos para a composição de serviços. Na Figura 3 é representada a visão esperada da IoT, na qual dispositivos e pessoas trocarão informações a todo momento a respeito do ambiente em que estão inseridos.

De acordo com Vermesan (2013b), um dos maiores desafios relacionados com a interoperabilidade em IoT é primeiramente assegurar a interoperabilidade das tecnologias para a entrega de informações para que posteriormente, desafios relacionados à compreensão e ao processamento dos dados sejam estudados. Os principais desafios listados por ele são:

extrair informações referentes a eles de uma maneira eficiente e eficaz. Isso é importante pois os ambientes IoT podem operar com até centenas de milhares de sensores.

- **Visualização:** busca otimizar os recursos utilizados, tais como armazenamento, ciclos de *clock* computacionais e utilização dos sensores, entre diversos usuários compartilhando estes recursos.

2.1.4 Gerenciamento dos dados em IoT

O gerenciamento de dados é uma tarefa crucial em IoT, pois ao considerar o número de objetos interconectados que estão trocando constantemente informações, há potencial para um grande volume de dados, que devem ser processados de maneira eficiente e eficaz. De acordo com [Zhang et al. \(2013\)](#), o gerenciamento dos dados em IoT é uma tarefa complexa e os mecanismos propostos para tal devem respeitar uma série de requisitos funcionais e não funcionais, tais como escalabilidade, confiabilidade, interoperabilidade, segurança, integridade, privacidade, desempenho, reusabilidade e extensibilidade.

Os sistemas de gerenciamento de dados devem processar os dados coletados em tempo de execução enquanto fornecem serviços de armazenamento, *log* e auditoria para análises *offline* ([ABU-ELKHEIR; HAYAJNEH; ALI, 2013](#)). Alguns dos conceitos mais relevantes nesta área são: a coleta e a análise dos dados, *Big Data*, rede de sensores semânticas, sensores virtuais e o processamento de eventos complexos ([VERMESAN, 2013a](#)).

2.1.4.1 Coleta e Análise dos Dados em IoT

Os módulos responsáveis pela coleta e análise dos dados são componentes essenciais em qualquer plataforma ou sistema envolvido no contexto de IoT. As tecnologias envolvidas para a realização de tais atividades estão sempre evoluindo. Esta evolução é necessária, pois diversas tecnologias são usadas para a coleta de dados com diferentes *tradeoffs* em termos de capacidade, eficiência energética e conectividade, os quais podem influenciar na qualidade dos dados coletados e como são transmitidos e gerenciados ([AGGARWAL; ASHISH; SHETH, 2012](#)).

A chave do paradigma da IoT é a habilidade de prover dados em tempo real a partir de diversas fontes distribuídas para outras máquinas, entidades inteligentes ou pessoas, para alimentar os mais variados tipos de serviço. Os módulos responsáveis pela coleta e análise de dados devem fornecer mecanismos escaláveis para a coleta de dados de dispositivos heterogêneos, descrever os dados coletados de tal forma que eles possam ser reutilizados, e também realizar a mineração dos dados em tempo viável, facilitando assim a extração de informações ([AGGARWAL; ASHISH; SHETH, 2012](#); [VERMESAN, 2013a](#)). Segundo o *National Institute of Standards and Technology* (NIST), os paradigmas podem ser elencados como:

- **Cloud Computing:** é um modelo que permite o acesso à rede de forma ubíqua, conveniente e sob demanda, a um conjunto compartilhado de recursos computacionais configuráveis que podem ser rapidamente provisionados e liberados com um esforço mínimo de gestão ou interação com o provedor de serviços (MELL; GRANCE *et al.*, 2011). Na prática, a *cloud computing* é definida como um conjunto de servidores remotos hospedados na Internet que são utilizados para armazenar, gerenciar e processar dados de acordo com a demanda.
- **Fog computing:** é um modelo que propõe a adição de uma camada entre os dispositivos inteligentes e a nuvem. Esta camada é composta por *fog nodes* (físicos ou virtuais), que são utilizados como intermediários da rede de acesso entre os dispositivos inteligentes e a nuvem. Geralmente, os *fog nodes* estão fortemente acoplados a um conjunto de dispositivos inteligentes ou às suas redes de acesso, fornecendo alguma forma de gerenciamento de dados e serviço de comunicação entre a camada de borda e a nuvem (IORGA *et al.*, 2018).
- **Edge computing:** é a camada de rede que engloba o conjunto de dispositivos inteligentes e seus usuários, para fornecer, por exemplo, capacidade de computação local em um sensor, medição ou alguns outros dispositivos acessíveis por rede (IORGA *et al.*, 2018).
- **Mist computing:** é uma forma leve e limitada de poder de computação nativa dos próprios dispositivos inteligentes, realizadas por meio de microcomputadores e microcontroladores, para enviar informações ao invés de dados para a camada de *fog* (IORGA *et al.*, 2018).

2.1.4.2 Big Data

Big Data refere-se à incapacidade de arquiteturas de dados tradicionais para manipular eficientemente novos conjuntos de dados. Uma das definições sobre *big data* é dada pelo NIST (NIST..., 2015), a qual define o paradigma de *Big Data* como a distribuição de sistemas de dados por recursos organizados de forma horizontal fortemente agrupados e independentes, que são utilizados para alcançar a escalabilidade necessária para processar eficientemente conjuntos de dados extensos.

Big Data também pode ser definida de acordo com suas características principais, também conhecida como 3V's: volume, variedade e velocidade (ZASLAVSKY; PERERA; GEORGAKOPOULOS, 2013):

- **Volume:** Volume está relacionado ao tamanho do conjunto de dados analisados medidos em *terabytes* (TB), *petabytes* (PB), *zettabytes* (ZB), etc.
- **Variedade:** Variedade considera as diferentes formas de representação de dados. Diferentes fontes de dados irão produzir *big data*, tais como sensores, dispositivos, redes sociais, telefones celulares, etc. Por exemplo, os dados armazenados podem ser históricos

de acesso na Web, leituras de sensores, dados não estruturados de redes sociais e áudio, entre outros.

- **Velocidade:** Velocidade representa quão frequentemente os dados são gerados. Por exemplo, a cada milissegundo, segundo, minuto, hora, dia, semana, mês ou ano. A frequência de processamento desses dados pode ser diferente das desejadas pelos usuários. Para alguns tipos de dados é extremamente importante que os dados sejam processados em tempo real, enquanto que, para outros o processamento deve ser feito apenas quando necessário. Tipicamente, podemos identificar três categorias principais: ocasional, frequente e em tempo real.

Dessa maneira, *big data* requer tecnologias que processem eficientemente grandes quantidades de dados em um limite tolerável de tempo. As tecnologias a serem aplicadas em *big data* incluem bancos de dados de processamento paralelo, *grids* de mineração de dados, sistemas de arquivos distribuídos e banco de dados distribuídos e escaláveis. Geralmente, estas tecnologias estão diretamente ligadas com muitos aspectos ligados à análise de fenômenos naturais como clima e atividade sísmica, saúde, segurança e processos de negócios (VERMESAN, 2013b).

Recentemente, *big data* vem sendo cada vez mais utilizado e várias questões devem ser endereçadas por pesquisadores. Dentre as diversas possibilidades de pesquisa na área destacam-se a captura dos dados, armazenamento, busca, análise e a otimização.

2.1.4.3 Anotações Semânticas

A utilização de anotações semânticas para representar os dados permite o gerenciamento em larga escala de redes distribuídas com limitações de energia e recursos. Além disso, permitem também a utilização de agentes e mecanismos inteligentes para processar e compreender os dados adquiridos (VERMESAN, 2013b).

Para isso, a utilização de anotações semânticas busca padronizar a descrição dos recursos para atingir a interoperabilidade entre os dispositivos, mas também para facilitar a análise dos dados. A visão semântica aborda a separação do significado do dado do seu valor propriamente dito. A ideia é de que os significados semânticos sejam separados dos seus valores e que ferramentas efetivas sejam utilizadas para gerenciar essas informações e tomar decisões (AGGARWAL; ASHISH; SHETH, 2012).

Atualmente, há diversos projetos com o intuito de definir ontologias e desenvolver *middlewares* para aplicar anotações semânticas em redes de sensores. O *Semantic Sensor Web* (SHETH; HENSON; SAHOO, 2008) propõe anotações para os dados dos sensores com valores espaciais e temporais. Esta abordagem utiliza as especificações definidas pelo consórcio *Sensor Web Enablement* (BOTTS *et al.*, 2008) e visa a estender as tecnologias semânticas para prover descritores que facilitem o acesso aos dados. O uso de descritores semânticos em conjunto com a codificação em formato de máquina também auxilia a contornar as ambiguidades resultantes

de erros humanos ou interpretações equivocadas devido a diferentes linguagens utilizadas em todo mundo, bem como estabelecer referências com informações disponíveis em outros sistemas (BANDYOPADHYAY; SEN, 2011)

2.1.4.4 Sensores Virtuais

Em implantações de redes de sensores existentes, sistemas de coleta de dados exigem sensores para transmitir dados brutos aos nós *sink* para serem manipulados e processados. Porém, esta não é uma abordagem muito eficiente, uma vez que comumente há limitações de recursos nas redes de sensores, tais como bateria e largura de banda. Além disso, a taxa de transferência em cada nó diminui à medida que a rede aumenta devido à transmissão de dados redundantes. Um sensor virtual é um sensor responsável por abstrair um sensor físico. Sensores virtuais fornecem medidas indiretas de condições abstratas (que, por si só, não são fisicamente mensuráveis) pela combinação de dados de sensoriamento de um ou mais sensores físicos heterogêneos (KABADAYI; PRIDGEN; JULIEN, 2006).

Os dados adquiridos por um conjunto de sensores podem ser coletados e processados de acordo com uma função de agregação, e em seguida, manipulados como na leitura de um único sensor virtual. Na Figura 4, é representado como pode ser realizada a abstração de sensores físicos para sensores virtuais, na qual os sensores físicos podem ser utilizados para diferentes finalidades e também reutilizados por diferentes sensores virtuais. Um sensor virtual apresenta as seguintes características (VERMESAN, 2013b):

Figura 4 – Exemplo de abstração realizada pelos sensores virtuais

Fonte: Elaborada pelo autor.

- Um sensor virtual se comporta como um sensor real, podendo emitir dados temporais

para uma região geográfica específica com observações que os sensores reais podem não apresentar (VERMESAN, 2013b);

- Um sensor virtual pode não apresentar qualquer propriedade física de um sensor real tais como fabricante, ou informações energéticas, mas apresenta outras propriedades tais como quais métodos ele disponibiliza e em quais sensores suas informações estão baseadas (VERMESAN, 2013b).

Desta maneira, sensores virtuais permitem realizar a interação de redes de sensores sem fio de forma eficiente e contornando suas limitações com outros paradigmas tal como IoT. Além disso, sensores virtuais podem ser utilizados pela nuvem, a qual permite a interação de diferentes redes, cobrindo uma extensa área geográfica, permitindo que sejam utilizados simultaneamente por múltiplos usuários. Sensores virtuais são responsáveis por permitir a criação de ambientes multiusuário construídos sob redes de sensores com limitações de recursos e também permitem a criação dos mais variados tipos de serviço (VERMESAN, 2013a).

2.1.5 Computação em Nuvem e Internet das Coisas

A integração entre IoT e Computação em Nuvem denominada CoT é capaz de atender a grande parte das necessidades existentes em IoT, facilitando o desenvolvimento de serviços de sensoriamento e fornecendo os recursos necessários para processar os fluxos de dados (AAZAM; HUH, 2014). Neste contexto, novos paradigmas tais como *Sensing-as-a-Service*, *Sensing-and-Actuation-as-a-Service*, *Video-Surveillance-as-a-Service*, *Big Data Analytics-as-a-Service*, *Data-as-a-Service*, *Sensor-as-a-Service* e *Sensor-Event-as-a-Service* vêm surgindo para endereçar os problemas de coleta, armazenamento, processamento e análise de dados e propiciar novas formas de desenvolvimento de aplicações.

Sensing-as-a-Service e Sensing-and-Actuation-as-a-Service: Neste modelo de serviço os sensores e atuadores são compartilhados como serviço. Estes dispositivos podem ser disponibilizados de graça, no qual os recursos serão acessados respeitando apenas as restrições especificadas em cada dispositivo ou por cobrança em que além das restrições dos dispositivos uma compensação financeira deverá ser feita ao seu proprietário (DISTEFANO; MERLINO; PULIAFITO, 2012).

Este modelo é composto por três camadas principais. A primeira camada é composta pelos sensores, e atuadores e seus proprietários, sendo que múltiplos dispositivos podem estar vinculados a um único objeto. O proprietário do sensor decide quais sensores serão publicados na nuvem e suas restrições de acesso, modelando a informação fornecida de acordo com o *Service-level agreement* (SLA) estabelecido e o perfil do usuário (PERERA *et al.*, 2013b).

A segunda camada é composta pela nuvem, a qual deve prover formas de acesso aos sensores disponíveis. O repositório utilizado para armazenar estes sensores contém as especifi-

cações dos sensores disponíveis, tais como proprietário e atributos de *Quality of service* (QoS). Geralmente, o acesso a estes sensores é feito por meio de diferentes tipos de serviços. Estes serviços devem selecionar dentre todos os sensores e atuadores disponíveis, os que melhores atendam as requisições do usuário, de acordo com o SLA estabelecido (PERERA *et al.*, 2013b; DISTEFANO; MERLINO; PULIAFITO, 2012).

Por fim, a última camada deste modelo é a camada de acesso, composta pelos usuários finais. Os usuários devem cadastrar-se e obter um certificado digital de uma autoridade para realizar o acesso a esta camada, com o intuito de consumir os dados oferecidos. Alguns dos maiores consumidores de dados são os governos, empresas, universidades e comunidades científicas. Esta camada não se comunica diretamente com os sensores ou seus proprietários. Todas as comunicações e transações devem ser intermediadas pela nuvem, que deverá respeitar o SLA estabelecido entre os usuários e os proprietários dos sensores (PERERA *et al.*, 2013b).

Data-as-a-Service: Neste modelo, os dados estruturados são fornecidos como serviço por meio das operações *Create, Read, Update, Delete* (CRUD), as quais reduzem a complexidade para obter e fornecer dados. Além disso, múltiplas aplicações podem utilizar estes dados mediante um SLA que definem, por exemplo, quais dados poderão ser acessados e seus custos (TRUONG; DUSTDAR, 2009).

A arquitetura deste modelo é composta por três camadas principais. A primeira camada é formada pelas fontes de dados e seus proprietários. Estas fontes de dados podem variar desde dispositivos ativos que realizam medições periódicas aos dados coletados de dispositivos que já foram desativados. O proprietário das fontes de dados define a estrutura de dados para publicá-los na nuvem e como seu acesso será realizado.

A segunda camada é composta pela nuvem que deverá gerenciar e orquestrar todos os recursos necessários para a manipulação dos dados estruturados de diferentes tipos de fonte. Esta camada deve apresentar características elásticas à medida que a quantidade de dados armazenados pode variar ao longo do tempo. As principais funcionalidades fornecidas por esta camada são o armazenamento, o processamento e a busca dos dados, os quais devem considerar fontes de dados heterogêneas e questões de desempenho (TERZO *et al.*, 2013).

A última camada deste modelo é a camada de aplicação, que é composta pelas partes interessadas nos dados. Esta camada fornece acesso direto aos dados estruturados por meio de uma *Application Programming Interface* (API), permitindo a análise dos dados e o desenvolvimento de aplicações. Além disso, a funcionalidade de controle de acesso também pode ser implementada nesta camada, visando aumentar a segurança das aplicações (TERZO *et al.*, 2013).

Big-Data-as-a-Service: Neste modelo, recursos tais como pessoas, dados, *software* e infraestrutura são fornecidos como serviço, com o intuito de facilitar o armazenamento, a busca, a análise e a visualização dos dados pelo usuário final. Este modelo apresenta muitas semelhanças

com o modelo de *Data-as-a-Service*, diferenciando-se dele por utilizar dados não estruturados, aspectos de qualidade dos dados e serviços para a análise e visualização dos dados (E *et al.*, 2013).

A arquitetura deste modelo é composta por quatro camadas principais. Assim como no modelo *Data as Service* (DaaS) a primeira camada é composta pelas fontes de dados e seus proprietários, os quais são responsáveis por fornecer os dados a serem armazenados. Diferentemente do modelo DaaS, estes dados não precisam ser estruturados (E *et al.*, 2013).

A segunda camada é responsável pelo armazenamento dos dados. Esta camada deve fornecer formas para armazenar dados relacionais, não relacionais e não estruturados tais como objetos, documentos e vídeos. Nesta camada, um módulo responsável por modelar os dados deve extrair um modelo de cada tipo de fonte de dados, especialmente dos dados não estruturados (E *et al.*, 2013).

A terceira camada é responsável por processar e analisar os dados existentes. Esta camada deve fornecer o conjunto de softwares necessários para registrar os serviços disponíveis, consultar os dados armazenados, decompor as requisições e analisar os dados (E *et al.*, 2013).

A última camada deste modelo é a camada de aplicação, que é composta pelas partes interessadas nos serviços disponíveis. Diversos tipos de serviços são fornecidos nesta camada com o intuito de facilitar a utilização dos dados pelo usuário, os quais permitem desde a simples visualização dos dados até a realização de inferências (E *et al.*, 2013).

2.1.6 Middlewares

Irisnet é um projeto proposto por Gibbons *et al.* (2003) e desenvolvido dentro do grupo de pesquisas da Intel, no qual os sensores comunicam-se com uma arquitetura por meio de uma interface genérica. Na arquitetura, os dados coletados pelos sensores são filtrados com o intuito de reduzir o tráfego da rede, e em seguida, armazenados em um banco de dados distribuído e disponibilizados para a realização de consultas.

A solução comercial *Xively Cloud Services* desenvolvida pela empresa **Xively** (formalmente conhecida como Cosm ou Pachube), oferece uma plataforma como serviço para o desenvolvimento rápido de aplicações para IoT. Uma vez que o dispositivo é conectado nesta solução, um banco de dados de alto desempenho é utilizado para armazenar e recuperar seus dados, que podem ser compartilhados e usados por outras aplicações, dispositivos ou serviços (INC., 2014).

Outra solução comercial que compartilha a ideia do projeto **Irisnet** é a plataforma denominada **SensorCloud** da empresa *Lord Microstrain*, que fornece armazenamento, visualização e gerenciamento de aplicações baseada nos sensores. Esta plataforma é capaz de monitorar os dados coletados por um conjunto de sensores, e emitir alertas por e-mail e SMS (MICROSTRAIN, 2014).

Global Sensor Networks (GSN), cujo objetivo principal é prover um *middleware* para integrar tecnologias heterogêneas de sensoriamento. Este *middleware* permite a utilização dos sensores como serviço por meio de um modelo *peer-to-peer*. Para isso, sensores virtuais são oferecidos como serviço e abstraem o processo de coleta e processamento de dados de um ou mais sensores. A arquitetura do **GSN** é descentralizada e segue um modelo baseado em *containers*, no qual cada *container* é responsável por gerenciar um conjunto de sensores virtuais. A identificação e descoberta dos sensores é feita por meio de palavras chave. A seleção dos sensores disponíveis é feita por meio de uma lista com *checkboxes* em uma interface Web (ABERER; HAUSWIRTH, 2006).

A arquitetura **SensorWeb**, desenvolvida pelo grupo de pesquisas da *Microsoft*, oferece um conjunto de ferramentas para a publicação e compartilhamento de redes de sensores pela Internet. Nesta solução, os usuários podem selecionar os sensores por meio de um mapa, pelo tipo ou por palavras chave (NATH; LIU; ZHAO, 2007).

Membros do Consórcio Geoespacial Aberto (OGC) definiram um *framework* denominado *Sensor Web Enablement (SWE)*, que especifica interfaces interoperáveis e modelos de codificação de metadados, os quais permitem a integração em tempo real de sensores heterogêneos conectados à Internet. Neste *framework*, sete padrões de desenvolvimento são especificados, sendo eles (BOTTS *et al.*, 2008):

- **Esquema de observação e quantificação:** Esquema *eXtensible Markup Language (XML)* padrão para codificar e quantificar as observações coletadas de um sensor;
- **Linguagem de marcação para sensores:** Esquema XML para descrever o sensor e seus processos, o qual fornece as informações necessárias para descoberta e localização dos sensores;
- **Linguagem de marcação para transdutores:** Esquema XML para descrever transdutores para realizar a comunicação entre o *framework* e o sensor;
- **Serviços de observação de sensores:** Interface para um serviço cujo objetivo é filtrar e recuperar dados e informações dos sensores;
- **Serviços de planejamento de sensores:** Interface para um serviço que seleciona os sensores a serem observados;
- **Serviços de alerta de sensores:** Interface para um serviço para se inscrever e receber notificação dos sensores;
- **Serviços de notificação Web:** Interface para um serviço de notificação assíncrona de dados dos sensores.

O projeto **openIoT** baseia-se em projetos existentes para redes de sensores e IoT para prover um *middleware* utilizando como base o projeto **GSN** e algumas especificações definidas pela OGC SWE. Este projeto suporta a configuração e a implantação dinâmica de algoritmos para coleta e tratamento dos dados dos objetos disponíveis. A interoperabilidade entre objetos e o reuso das redes de sensores é feita por meio de ontologias e estruturas semânticas que fornecem informações sobre o contexto em que estes sensores são utilizados (SOLDATOS; SERRANO; HAUSWIRTH, 2012).

Tabela 1 – Síntese das características dos middlewares e projetos utilizados para a provisão de recursos como serviços em ambientes IoT.

Projeto	Descrição		Arquitetura	
	Metadados	Descritores Semânticos	Centralizada	Descentralizada
Irisnet	X		X	
Xively	X		X	
SensorCloud	X		X	
GSN	X			X
SensorWeb	X		X	
SWE		X	X	
OpenIoT		X		X

Fonte: Elaborada pelo autor.

No entanto, a maioria dos projetos analisados que utilizam o modelo DaaS visam apenas fornecer formas de integração entre as redes de sensores e a Internet. Na Tabela 1 são destacadas as principais características dos projetos analisados com relação ao modelo utilizado, modo de descrição dos sensores e a organização arquitetural da rede. Nesta tabela, é possível observar que grande parte dos projetos analisados utilizam palavras chave para realizar a seleção dos sensores. Ademais, nota-se também que em geral os projetos utilizam uma arquitetura centralizada para orquestrar a comunicação entre os sensores e as partes interessadas. Porém, este tipo de modelo arquitetural pode ter seus recursos de armazenamento, processamento ou rede facilmente sobrecarregados à medida que a demanda por sua utilização aumenta.

Por outro lado, dois projetos (SWE e OpenIoT) buscam utilizar dados semânticos para realizar a seleção dos sensores. A utilização de semântica é extremamente importante, uma vez que possibilita o desenvolvimento de mecanismos para descoberta e seleção automática dos sensores baseadas no contexto em que são aplicados. Ainda, nota-se também na Tabela 1 que dois trabalhos (OpenIoT e GSN) utilizam uma arquitetura descentralizada para a tomada de decisões, a qual permite o desenvolvimento de mecanismos de tolerância às falhas e redundância, tanto das redes de sensores quanto do *middleware* utilizado. Por fim, constata-se que nenhum dos projetos avaliam os atributos de QoS pertencentes a redes de sensores, tampouco dos sensores que as compõem, os quais estão sujeitos a degradações durante a coleta (PERERA *et al.*, 2012).

2.2 Otimização Multiobjetivo

Uma variedade de problemas nas áreas de engenharia, indústria e computação envolvem a otimização de múltiplos objetivos. Em muitos casos, estes objetivos são definidos em unidades não comparáveis e apresentam algum nível de conflito entre eles. Em outras palavras, em um problema multiobjetivo, geralmente um objetivo não pode ser melhorado sem deteriorar outro (JAIMES; MARTINEZ; COELLO, 2009). Este tipo de problema é muito comum no contexto de IoT, uma vez que os perfis dos dispositivos desejados para uma determinada tarefa podem requerer um conjunto de características específicas que são conflitantes entre si.

Figura 5 – Problema de otimização de custo-desempenho.

Fonte: Adaptada de Bui e Alam (2008).

Por exemplo, uma empresa deseja adquirir equipamentos computacionais que apresentam alto desempenho, com o intuito de acelerar a coleta e o processamento de seus dados. Por outro lado, esta empresa também deseja reduzir os custos para adquirir tais equipamentos. Claramente, tais objetivos são conflitantes à medida que se deseja adquirir equipamentos de alto desempenho ao mesmo tempo em que se reduz os seus custos, tal como representado na Figura 5. Além disso, seus objetivos são expressos em unidades de medida diferentes, dificultando assim sua comparação (BUI; ALAM, 2008).

Em um problema de otimização com um objetivo, o espaço de busca é sempre bem definido. Ao passo que mais objetivos conflitantes são otimizados simultaneamente, torna-se mais difícil estabelecer uma única solução ótima. A solução ótima será um conjunto de *trade-offs* ótimos entre os objetivos conflitantes (ABRAHAM; JAIN, 2005).

Um problema de otimização multiobjetivo pode ser descrito como

$$\text{minimize}\{f_1(x), f_2(x), \dots, f_k(x)\}, \text{ em que } x \in S, \quad (2.1)$$

em que o número de funções objetivo k é maior ou igual a dois no espaço de decisão, representado por R^n . O vetor de funções objetivo é definido por $f(x) = (f_1(x), f_2(x), \dots, f_k(x))$

. O vetor de decisão $x = (x_1, x_2, \dots, x_n)$ pertence à região factível não nula S , a qual é um subconjunto de R^n (MIETTINEN, 2012).

2.2.1 Relações de dominância de Pareto

Em problemas de otimização multiobjetivo, são raros os casos em que um único ponto é capaz de otimizar todas as funções objetivos. Geralmente, um conjunto de soluções com diferentes *trade-offs* entre seus objetivos são utilizadas para solucionar tais problemas em detrimento de uma única solução. As relações de dominância de Pareto são utilizadas para comparar tal conjunto de soluções. Ao conjunto de soluções ótimas de um problema é dado o nome de conjunto de soluções de Pareto ótimas ou não dominadas (JAIMES; MARTINEZ; COELLO, 2009).

Em um problema de minimização, uma solução x é dita não dominada se não existe nenhum $y \in S$ tal que $f_i(y) \leq f_i(x)$ para cada objetivo $i=1, \dots, k$ e $f_i(y) < f_i(x)$ para pelo menos um dos objetivos analisados (ABRAHAM; JAIN, 2005). A imagem do conjunto de soluções ótimas é chamada de fronteira de Pareto ou curva de Pareto. O formato da fronteira de Pareto indica a natureza do *trade-off* entre as diferentes funções objetivo (CARAMIA; DELL'OLMO, 2008).

Figura 6 – Fronteira de Pareto para diferentes funções objetivo

Fonte: Adaptada de Deb (2001).

Na Figura 6 é representada a fronteira de Pareto para quatro cenários diferentes com

dois objetivos. Cada um dos objetivos pode ser maximizado ou minimizado. A curva sólida representa o conjunto de soluções não dominadas. Nota-se nesta figura, que o conjunto ótimo de Pareto sempre é composto pelas soluções de um vértice específico da região factível. Além disso, é válido destacar que em um espaço contínuo de soluções, o conjunto ótimo de Pareto pode ser formado por dois conjuntos disjuntos de soluções. Entretanto, apesar da existência de múltiplas soluções Pareto ótimas, na prática apenas uma parte dessas soluções deve ser utilizada (DEB, 2001).

O Algoritmo 1 descreve o algoritmo *Fast Non-Dominated Sort* (FNDS) proposto por Deb (2001), o qual apesar da sua complexidade $O(MN^2)$, é utilizado para calcular as fronteiras de um conjunto de soluções. Tal algoritmo, estabelece as relações de dominância entre os recursos disponíveis, para determinar os recursos pertencentes as diferentes fronteiras.

Algoritmo 1 – Algoritmo Fast non-dominated sort

```

1: função FAST-NON-DOMINATED-SORT( $\mathbb{P}$ )
2:   para todo  $p \in \mathbb{P}$  faça
3:      $S_p = \emptyset$ 
4:      $n_p = 0$ 
5:     para todo  $q \in \mathbb{P}$  faça
6:       se  $p \prec q$  então
7:          $S_p = S_p \cup q$ 
8:       senão se  $q \prec p$  então
9:          $n_p = n_p + 1$ 
10:    fim se
11:    fim para
12:    se  $n_p = 0$  então
13:       $p_{rank} = 1$ 
14:       $F1 = F1 \cup p$ 
15:    fim se
16:  fim para
17:   $i = 1$ 
18:  enquanto  $F_i \neq \emptyset$  faça
19:     $Q = \emptyset$ 
20:    para todo  $p \in F_i$  faça
21:      para todo  $q \in S_p$  faça
22:         $n_q = n_q + 1$ 
23:      se  $n_q = 0$  então
24:         $q_{rank} = i + 1$ 
25:         $Q = Q \cup q$ 
26:      fim se
27:    fim para
28:    para todo  $p \in Q$  faça
29:       $p_{rank} = i + 1$ 
30:     $F_i = Q$ 
31:  fim enquanto
32: fim função

```

2.2.2 Técnicas de Análise de Decisão Multicritérios

Técnicas *Multiple-criteria decision analysis* (MCDA) são utilizadas para a tomada de decisões na presença de múltiplos e geralmente conflitantes objetivos (XU; YANG, 2001). Tais técnicas utilizam algoritmos que visam auxiliar o julgamento da tomada de decisão por meio de um conjunto de objetivos e critérios, estimando sua importância e estabelecendo a contribuição de cada opção de acordo com cada critério analisado (DODGSON *et al.*, 2009).

Um problema MCDA pode ser descrito utilizando uma matriz de análise $M \times N$, na qual o elemento q_{ij} representa o desempenho de cada opção de acordo com o critério de decisão c_j em unidades e escalas diferentes e não comparáveis, tal como representada pela Equação 2.2.

$$Q = \begin{matrix} & \begin{matrix} c_1 & c_2 & c_3 & \dots & c_n \end{matrix} \\ \begin{matrix} q_1 \\ q_2 \\ \vdots \\ q_m \end{matrix} & \begin{bmatrix} q_{11} & q_{12} & q_{13} & \dots & q_{1n} \\ q_{21} & q_{22} & q_{23} & \dots & q_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ q_{m1} & q_{m2} & q_{m3} & \dots & q_{mn} \end{bmatrix} \end{matrix} \quad (2.2)$$

Diversas técnicas de MCDA, como a *Simple Additive Weighting* (SAW), a *Technique for Order of Preference by Similarity to Ideal Solution* (TOPSIS) e a *Vlsekriterijumska Optimizacija I Kompromisno Resenje* (VIKOR) definem explicitamente suas opções e os pesos de cada critério, porém se diferem na forma em que combinam os seus dados de entrada. Geralmente, tais técnicas utilizam uma matriz de avaliação para representar o desempenho relativo de cada opção q'_{ij} por meio de uma função de normalização para permitir a comparação entre os diferentes critérios e posteriormente por meio da combinação de seus pesos a sua redução em um único critério (TZENG; HUANG, 2011).

O algoritmo de decisão TOPSIS é um dos algoritmos MCDA mais utilizados, o qual explora os valores dos critérios e a suas importâncias relativas para fornecer um conjunto ordenado de alternativas. Dentre os domínios de aplicação que utilizam o TOPSIS incluem-se a gestão de cadeia de suprimentos e logística, design, engenharia e manufatura, gerenciamento de negócios e marketing, saúde e gestão ambiental (BEHZADIAN *et al.*, 2012).

TOPSIS ordena um conjunto de opções de acordo com a distância Euclidiana para a solução ideal e ideal-negativa. A solução ideal corresponde ao melhor valor possível de cada critério enquanto, a solução ideal-negativa representa o pior valor de cada critério. As opções são classificadas de acordo com sua proximidade à solução ideal e distância da solução negativa-ideal (TZENG; HUANG, 2011). O algoritmo TOPSIS pode ser resumido nas seguintes etapas:

1. Normalizar a matriz de decisão Q , de acordo com a Equação 2.3:

$$q'_{ij} = \frac{q_{ij}}{\sqrt{\sum_{i=1}^N (q_{ij})^2}} \quad (2.3)$$

em que N representa o número de opções na matriz de avaliação.

2. Calcule o ponto ideal (p_{+j}) e o ponto ideal negativo (p_{-j}) de todas funções objetivo utilizando a matriz de análise. Para maximizar um critério, as soluções ideais e ideais negativas podem ser calculadas por meio das Equações 2.4 e 2.5 respectivamente:

$$p_{+j} = \max_i(q'_{ij}) \quad (2.4)$$

$$p_{-j} = \min_i(q'_{ij}) \quad (2.5)$$

3. Calcule a distância para a solução ideal (s_{i+}) e para solução ideal-negativa (s_{i-}). A distância de cada opção q' para a solução ideal p_{+j} e para a solução negativa ideal p_{-j} são dadas pelas Equações 2.6 e 2.7:

$$s_{i+} = \sqrt{\sum_{j=1}^n (q'_{ij} - p_{+j})^2} \quad (2.6)$$

$$s_{i-} = \sqrt{\sum_{j=1}^n (q'_{ij} - p_{-j})^2} \quad (2.7)$$

4. Calcule a proximidade relativa para a solução ideal. A proximidade relativa de q para p_{+j} e p_{-j} representada por c_{i+} pode ser calculada de acordo com a Equação 2.8.

$$c_{i+} = \frac{s_{i-}}{s_{i+} + s_{i-}} \quad (2.8)$$

5. Classifique as opções q_i em ordem crescente de acordo com a proximidade relativa para c_{i+} .

2.2.3 Métricas de Avaliação

As soluções propostas por cada algoritmo podem ser avaliadas em termos de capacidade, convergência e diversidade. As métricas de capacidade são usadas para calcular a proporção da solução proposta por técnicas MCDA referentes ao conjunto de soluções ótimas, permitindo a comparação entre as soluções obtidas por cada algoritmo. *Error Ratio* e *Overall Nondominated Vector Generation Ratio* (ONVGR) são alguns exemplos de métricas de capacidade. Métricas de

convergência são usadas para medir a proximidade da solução proposta com a solução ótima. *Generational Distance* e *Seven Points Average Distance* são alguns exemplos de métricas de convergência. Métricas de diversidade são utilizadas para mostrar a distribuição e a disseminação das soluções propostas por um algoritmo, permitindo analisar se as soluções propostas apresentam funções objetivos semelhantes. *Crowding Distance*, *Spacing* e *Hyperarea* são alguns exemplos de métricas de diversidade (JIANG *et al.*, 2014).

Para avaliar as técnicas MCDA apresentadas nesta tese, a métrica de capacidade ONVGR (COLLETTE; SIARRY, 2004) será utilizada. Esta métrica representa o número de soluções propostas pelas técnicas MCDA como proporção do número de soluções ótimas disponíveis em cada fronteira de Pareto. Quanto mais perto de um o valor ONVGR estiver, melhor será a solução proposta nessa fronteira. A Equação 2.9 representa a fórmula utilizada para calcular a métrica ONVGR por fronteira, onde S_{front_i} corresponde ao número de soluções propostas pelo algoritmo MCDA na fronteira i e P_{front_i} corresponde ao número total de soluções ótimas existentes na fronteira i .

$$ONVGR_{front_i} = \frac{S_{front_i}}{P_{front_i}} \quad (2.9)$$

2.2.4 Considerações Finais

Este capítulo apresentou a fundamentação teórica necessária para compreender a aplicação desta tese. A metodologia para avaliação dos algoritmos em ambientes IoT baseia-se nas arquiteturas e ferramentas descritas neste capítulo, cujo objetivo principal é permitir a comparação das técnicas utilizadas para a seleção dos recursos nestes ambientes. Como tais técnicas geralmente utilizam múltiplos critérios para realizar a seleção dos recursos também são descritos neste capítulo os principais conceitos que envolvem a otimização multiobjetivo, dentre os quais destacam-se as técnicas MCDA e Ótimo de Pareto. Estes conceitos possibilitam a utilização de métricas como a ONVGR para realizar a comparação das técnicas de seleção de recursos em termos de qualidade. No próximo capítulo será apresentada uma revisão da literatura, contendo os principais trabalhos que propõem arquiteturas para IoT bem como técnicas para busca e seleção dos dispositivos que a ela pertencem.

TRABALHOS RELACIONADOS

Neste Capítulo é apresentada a revisão bibliográfica dos trabalhos relacionados deste projeto de pesquisa. A revisão da literatura realizada seguiu uma metodologia de pesquisa, na qual definiu-se um conjunto de palavras chaves que foi disposto da seguinte maneira: (*sensor* ou *sensors*) e (*search* ou *searching* ou *discovery*) e ((*cloud* ou *web*) e (*things* or *resources*)). Esta consulta foi executada nas bases de artigos científicos do IEEE Xplore¹, ACM Digital Library² e Scopus³.

A maior parte dos artigos científicos resultantes da busca em tais bases apresenta infraestruturas que integram diferentes tipos de recursos e possibilitam sua busca. Estas infraestruturas podem ser divididas em dois grandes grupos. No primeiro grupo destacam-se os trabalhos que utilizam metadados para descrever e integrar os diferentes sensores tais como os apresentados por Stirbu (2008), Datta, Bonnet e Nikaein (2014), Carlson e Schrader (2014). Já o segundo grupo caracteriza-se por descrever os recursos utilizando ontologias tais como os trabalhos de Bovet e Hennebert (2014), Alam e Noll (2010), Christophe (2012) e Kiljander *et al.* (2014). Estes trabalhos serão apresentados na Seção 3.1. Por outro lado, poucos trabalhos se preocupam em descrever as técnicas utilizadas para realizar a seleção dos recursos. Os trabalhos que tratam deste assunto são apresentados na Seção 3.2

3.1 Infraestruturas para a Seleção de Recursos

Evdokimov *et al.* (2010) apresentam uma avaliação qualitativa analisando os requisitos funcionais e não funcionais de um conjunto de arquiteturas para a busca e seleção de serviços no contexto de IoT. Eles apresentam um levantamento dos requisitos funcionais para estes serviços, os quais foram baseados majoritariamente nos padrões arquiteturais ISO/IEC 9126.

¹ <http://ieeexplore.ieee.org/>

² <http://dl.acm.org/>

³ <http://www.scopus.com/>

Os projetos EPCglobal, Bridge Project, Afiliats, ID@URI e abordagens *Peer-to-peer* (P2P) foram comparados com base nos requisitos levantados. A abordagem P2P foi a abordagem que preencheu a maior parte dos requisitos de maneira satisfatória. Outra conclusão é que nenhum dos projetos analisados foi capaz de garantir de maneira satisfatória a precisão dos resultados, uma vez que os dados providos pelos sensores poderiam ser omitidos ou alterados.

Os autores [Antonic et al. \(2014\)](#), propõem um ambiente para aplicações móveis colaborativas (*crowdsensing*) baseada em um *middleware* denominado *CloUd-based PUblish/SUBscribe*, a qual utiliza a nuvem para compartilhar dados sensíveis ao contexto com baixo consumo de energia. Além disso, este ambiente permite o gerenciamento da localização e capacidades dos sensores móveis e a aquisição dos dados fornecidos de acordo com os atributos de QoS desejados. Desta maneira, um *broker* é implantado nos dispositivos móveis servindo para orquestrar a comunicação entre o ambiente de nuvem e os dispositivos de sensoriamento para economizar recursos energéticos. Os resultados obtidos mostraram que a solução proposta é energeticamente viável e não consome recursos adicionais mesmo quando o dispositivo está sujeito a um grande número de sensores registrados e requisições simultâneas.

[Bovet e Hennebert \(2014\)](#) propõem uma arquitetura P2P para a descoberta dos sensores visando a robustez, a confiabilidade e a eficiência em termos energéticos. Os autores propõem uma ontologia para descrever as propriedades, funcionalidades e forma de acesso aos dispositivos registrados. A linguagem SPARQL é utilizada para especificar e buscar propriedades e funcionalidades nas ontologias, as quais são armazenadas de maneira distribuída entre os nós registrados na arquitetura. Os resultados obtidos por meio da implementação de um protótipo, mostrou que é possível inferir a interação entre os sensores e combiná-los para desenvolver novas aplicações.

[Alam e Noll \(2010\)](#) propõem um *framework* baseado em *proxy* e *Service-oriented architecture* (SOA) para acessar serviços com descrições semânticas visando os desafios de controle, descoberta, invocação de serviços no domínio de IoT. O *framework* proposto acessa informações de diferentes tipos de fontes de dados por meio de serviços e pode combiná-las para criar serviços virtuais. Para isso, um repositório semântico é utilizado para armazenar os dados dos dispositivos localizados em diferentes *proxys* e assim realizar inferências a partir deles. Nesta abordagem, a arquitetura funciona com um modelo *publish-subscribe*, no qual os objetos publicam seus dados, enquanto que os *proxys* podem tanto receber dados dos objetos quanto publicá-los para serviços externos.

[Christophe \(2012\)](#) propõe um *framework* organizado em uma estrutura federada para processar e organizar descritores Web Semânticos. Desta maneira, os nós são capazes de compartilhar o conhecimento adquirido das entidades de interesse e dos dispositivos gerenciados. O compartilhamento deste conhecimento se dá por meio de um conjunto de regras semânticas e um mecanismo de notificação para os nós mais próximos, uma vez que a localização é utilizada para otimizar o processo de seleção e manter o estado dos recursos em tempo de execução. Os

resultados deste trabalho compararam a eficiência da arquitetura proposta contra uma abordagem centralizada, na qual foi possível trocar até 1.000 descrições semânticas ou realizar até 2.000 associações entre dois nós em menos de 3 segundos.

Gupta, Udupi e Poursohi (2010) propõem uma arquitetura denominada *Sensor.Network* cujo intuito é armazenar, compartilhar, visualizar e analisar dados de dispositivos heterogêneos e facilitar sua interação por meio de uma API baseada nos conceitos arquiteturais RESTful. Para isso, os autores definem os principais blocos funcionais desta infraestrutura, sendo eles 1) Módulo de abstração do fluxo de dados para gerenciar os valores temporais dos sensores; 2) Uma API baseada em REST, para a manipulação e visualização do fluxos de dados dos dispositivos; 3) Um módulo para a busca e organização dos sensores que possibilita a seleção por nome, descrição, palavras-chaves, proprietários e valores; 4) Módulo de segurança que autentica e restringe o acesso dos usuários. 5) Módulo de visualização dos dados para retornar os dados de forma organizada para o usuário. Uma avaliação qualitativa desta arquitetura foi realizada para demonstrar as qualidades da arquitetura proposta com relação a outros trabalhos.

Ishaq *et al.* (2012) propõem uma solução auto organizável para facilitar a implantação de redes de sensores, sua descoberta e comunicação fim a fim em dispositivos com restrições de poder computacional utilizando o protocolo *Constrained Application Protocol* (CoAP). Os autores utilizam o protocolo 6LoWPAN para endereçar os sensores e conectá-los a um *gateway* local responsável por gerenciá-los. Os *gateways* disponíveis nesta arquitetura são organizados de forma hierárquica sendo que apenas um único *gateway* gerencia o acesso dos sensores com a Internet, permitindo assim o uso de um servidor *Domain Name System* (DNS). Desta maneira, os usuários podem utilizar os dados dos sensores por meio de requisições diretas ao seu IPv6 ou por meio dos endereços especificados no servidor DNS.

Jara *et al.* (2013) propõe um *framework* para a descoberta de sensores geo-localizados e sensíveis ao contexto denominada “*digcovery*”, a qual visa maximizar a eficiência e a sustentabilidade da implantação dos sensores. Este *framework* possibilita que os usuários registrem e incluam seus próprios sensores em uma infraestrutura distribuída. Dentre seus módulos destacam-se o *digrectory* e o *digcovery core*. O *digrectory* é o responsável por armazenar as informações e gerenciar as interações com os dispositivos heterogêneos de sensoriamento, utilizando ontologias compatíveis com os tipos DNS-SD para o acesso via DNS. Já o *digcovery core* é o responsável por centralizar as ações do *framework* e acessar os recursos disponíveis nos *digrectories* por meio de consultas DNS. A busca e a descoberta dos sensores podem ser feitas usando informações de contexto, da localização ou do tipo de recurso, configuradas por meio de requisições RESTful utilizando o protocolo CoAP ou por meio de uma interface em um dispositivo móvel. Por meio de um estudo de caso, os autores mostram que o *digcovery* oferece alta escalabilidade para a descoberta de recursos mesmo apresentando sua arquitetura centralizada.

Kamilaris, Papakonstantinou e Pitsillides (2014) investigam o uso do DNS como um repositório de metadados escalável e pervasivo e suas extensões para suportar a descoberta de

entidades a partir de sua localização. Os autores propõem a criação de um novo domínio, tal como *.env*, responsável por representar o mundo real. Desta maneira, quando um sensor se conectar à rede ele deverá registrar-se no repositório DNS fornecendo informações tais como seu tipo, localização e serviços oferecidos. Dois experimentos foram realizados neste trabalho e concentraram-se na avaliação dos tempos de resposta e escalabilidade entre um banco de dados e o servidor DNS, os quais foram adquiridos a partir da simulação de um milhão de sensores. Os resultados do primeiro experimento indicaram que o banco de dados se mostrou mais eficiente para uma quantidade pequena de sensores, enquanto que o DNS foi mais eficiente para uma quantidade maior de sensores. Já os resultados do segundo experimento, mostrou que o DNS foi mais eficaz para inserir novos registros no servidor.

Kiljander *et al.* (2014) propõem uma arquitetura para prover interoperabilidade entre os dispositivos heterogêneos, na qual todas as suas interações são realizadas por meio de informações semânticas acessíveis por meio de SPARQL e manipuladas por agentes. Diferentemente dos outros trabalhos existentes na literatura que utilizam *Uniform Resource Identifier* (URI) para acessar e identificar os dispositivos de uma determinada rede, este trabalho utiliza *ucodes*, que são identificadores únicos de 128 bits utilizados para identificar cada um dos dispositivos. Os *ucodes* existentes em uma rede são divididos entre vários *brokers*, que são diferenciados, por exemplo, pela sua localização geográfica, proprietários e informações armazenadas. Para acessar os dados dos sensores, servidores organizados de forma hierárquica são utilizados para a resolução de *brokers* e de *ucode*. O trabalho foi validado por meio de dois experimentos. No primeiro experimento foi analisado o tempo necessário para atualizar os sensores e para que os mesmos enviem notificações aos usuários, mensurando a latência para diferentes quantidades de dados, que mostrou que o processo de atualização é menos custoso do que o processo de notificação. Já o segundo experimento avaliou a capacidade de armazenamento de descrições em um *broker*, em torno de 1.000 descrições.

Liu, Li e Mao (2014) apresentam uma arquitetura de informação baseada em serviços para dispositivos com restrição de *hardware*, utilizando a nuvem para realizar parte do processamento e do armazenamento de dados. A arquitetura apresentada é dividida em quatro camadas. A camada de percepção tem a função de sensoriamento e atuação no ambiente; A camada de rede gerencia e integra os sensores em um nível local e os compartilha com o meio externo. A camada de serviço contém os provedores de serviço e repositórios que são utilizados para realizar o acesso à informação. A camada de processamento analisa as requisições, verificando permissões e gerenciando a informação coletada. Os experimentos realizados compararam os protocolos de transmissão de informação nestes ambientes, o qual demonstrou que o protocolo *Hypertext Transfer Protocol* (HTTP) utilizado em conjunto com os conceitos arquiteturais RESTful foi o mais adequado.

Mrissa, Medini e Jamont (2014) propõem uma arquitetura baseada em avatar para a interação entre os dispositivos heterogêneos por meio da exposição de recursos semânticos

em forma de serviços. A utilização de avatares estende um objeto na Web por meio de funcionalidades internas que permitem a interação com outros avatares e recursos externos. A arquitetura é composta por um *framework* formado por um conjunto de componentes específicos chamados gerenciadores que interagem entre si. Os avatares utilizam ontologias que descrevem as funcionalidades e os recursos fornecidos pelos dispositivos, para criar e implantar sua própria classe de aplicações. Os experimentos realizados utilizaram uma ontologia com 52 axiomas e mostrou que o algoritmo proposto encontrou todas as possíveis funcionalidades compostas com um tempo médio em torno de 45 milissegundos, mostrando a viabilidade da proposta em um pequeno ambiente de testes.

Orestis, Dimitrios e Ioannis (2012) apresentam um *Broker* como serviço denominado Überdust, que visa conectar objetos inteligentes, além de fornecer armazenamento, compartilhamento e a descoberta em tempo real e dados históricos desses dispositivos. O Überdust foi desenvolvido a partir de uma ontologia que representa um conjunto comum de termos definidos pela *W3C Semantic Sensor Network Incubator Group* para os sensores, assim como suas propriedades e eventos. A arquitetura Überdust é dividida em três camadas. A camada de baixo nível apresenta as conexões com os dispositivos do mundo real. A camada intermediária é a responsável por fornecer dados históricos ou em tempo real, por meio do acesso ao *broker* e do banco de dados. A camada superior corresponde à camada de aplicação na qual as leituras dos dispositivos são consumidas pelas aplicações. Estas camadas são integradas ao Überdust por meio de uma arquitetura *Enterprise Service Bus* (ESB) que trocam mensagens por meio dos conceitos arquiteturais RESTful.

Abdelwahab *et al.* (2015) propõem uma arquitetura para CoT para descoberta de recursos e dispositivos virtualizados. A arquitetura é proposta por três elementos principais: O primeiro nível são os dispositivos IoT, os quais são caracterizados como dispositivos equipados com sensores. O segundo elemento é chamado de Nuvens de primeiro nível que são plataformas de computação em nuvem convencionais que fornecem interfaces para os usuários acessarem os serviços de sensoriamento disponíveis. Por fim, os agentes de “cloud”, são elementos confiáveis situados próximos aos dispositivos IoT na borda da rede, que estão conectados a Internet e a Nuvem. Os autores propuseram um algoritmo denominado *Randomized and Asynchronous Distributed Virtualization* (RADV) para realizar a descoberta e seleção dos sensores. O funcionamento do algoritmo pode ser dividido em três fases: (i) o domínio virtual elimina os dispositivos não desejados, (ii) cada dispositivo constrói uma matriz de benefícios locais para maximizar o número total de dispositivos analisados e (iii) resolve os problemas estabelecidos para encontrar a solução ótima. Após a execução do RADV por todos os dispositivos, o agente de “cloud” é utilizado para selecionar a melhor solução disponível. Os resultados apresentados por meio de análises e simulações indicam que o RADV tem potencial para diminuir a sobrecarga de comunicação próximo ao ótimo empregando o mínimo de recursos físicos por meio da virtualização dos dispositivos que apresentam o maior benefício.

Diaz-Montes *et al.* (2015) apresentam a ferramenta CometCloud para fornecer a infraestrutura e suporte de programação para o desenvolvimento de workflows a partir de uma infraestrutura federada. A arquitetura do CometCloud é dividida em três camadas: infraestrutura, gerenciamento autônomo e interface. A camada de infraestrutura/federação permite a troca de informação entre os recursos distribuídos bem como a descoberta de recursos por meio da utilização de metadados. A camada de gerenciamento autônomo visa permitir a composição e o monitoramento de workflows de acordo com os objetivos e políticas definidos pelos usuários. Por fim, a camada de interface permite a troca de informações entre os usuários e o núcleo da ferramenta CometCloud.

Kothari *et al.* (2014) apresenta uma arquitetura chamada DQS-Cloud para otimizar a busca por sensores, mecanismos de tolerância a falhas e violações de SLA. O sistema é dividido em três camadas, provedores de dados de sensores, “*subscribers*” e servidores na nuvem. A primeira camada é o provedor de dados de sensores, o qual contém dispositivos que podem variar de telefones celulares a um Raspberry Pi. A segunda camada, é composta por qualquer dispositivo ou aplicação que demonstrem interesse em coletar dados de sensores. A última camada, o servidor de nuvem, é responsável por organizar a comunicação entre os sensores e os “*subscribers*”, além de armazenar os metadados e histórico dos serviços contratados em um repositório. A busca é baseada em palavras chaves e nos valores de QoS desejados pelos usuários. O DQS-Cloud visa minimizar o overhead de comunicação reutilizando fluxos de dados que apresentam níveis de QoS semelhantes. Os resultados mostraram que o DQS-Cloud foi capaz de minimizar a largura de banda e a taxa de processamento nos provedores.

Doukas e Antonelli (2015) apresentam o COMPOSE, que é uma coleção de ferramentas *open source* para fornecer uma solução capaz de desenvolver serviços e aplicações para CoT. Esta solução opera em todas as camadas de uma arquitetura IoT, interagindo com usuários de aplicações móveis, realizando a busca e seleção de sensores e também a implantação de aplicações na nuvem. A arquitetura está dividida em três camadas: mercado COMPOSE, motor de tempo de execução COMPOSE e a camada de ingestão. O mercado COMPOSE é uma interface gráfica que permite a publicação, troca e acesso aos serviços. O motor de execução COMPOSE é o responsável pelo processamento dos serviços de forma transparente aos usuários. A camada de Ingestão é a interface entre os objetos permitindo a comunicação bidirecional entre diferentes dispositivos e protocolos HTTP. A descoberta e seleção de dispositivos se dá por meio de um mecanismo denominado *iServe*, o qual é um armazém de serviços responsável por publicar, analisar e buscar o serviço por meio de descritores semânticos. Para validar a arquitetura COMPOSE, com um estudo de caso, os autores propuseram um serviço de compartilhamento de veículos, por meio da análise dos dados de tráfego e estacionamento com o objetivo de reduzir o número de carros utilizados e melhorar o trânsito na cidade de Barcelona, Espanha.

Khodadadi, Dastjerdi e Buyya (2015) propõem o framework Simurgh visando simplificar o gerenciamento de serviços IoT, coisas e humanos em ambientes IoT. A arquitetura do Simurgh

é dividida em três camadas: usuários, plataforma e coisas. A camada de coisas, é responsável por registrar e manipular os dispositivos que ingressam no ambiente desejado por meio de bibliotecas e interfaces específicas de cada dispositivo. A camada de plataforma é a responsável por armazenar os descritores dos dispositivos disponíveis, realizar a descoberta, seleção, composição e execução dos fluxos de dados disponíveis. A camada de usuário, corresponde a camada dos usuários finais que podem ser pessoas ou outros dispositivos. Os autores ainda propõem um meta modelo denominado *Thing Description Document* (TDD) para descrever as propriedades e os serviços oferecidos por cada dispositivo. O processo de descoberta de recursos é baseado em uma busca sintática e dividida em duas fases: (i) a busca no repositório TDD é realizada para encontrar os dispositivos que correspondem aos critérios de busca utilizado; (ii) a segunda fase do processo de busca, utiliza o resultado da primeira fase para buscar os sensores mais adequados a cada tarefa desejada pelo usuário. Para validar o framework proposto, os autores utilizaram um estudo de caso em um campus de uma universidade a qual continua vários sensores como de umidade e temperatura. O objetivo principal deste estudo de caso foi demonstrar como utilizar o framework proposto para ajustar dinamicamente a temperatura das salas baseado na temperatura interna, externa e no preço da energia elétrica.

Ikebe, Noguchi e Hoshikawa (2016) apresentam uma arquitetura para a descoberta de sensores em tempo de execução. Para isso, eles propõem a utilização de nós dispersos na rede para atuar como buffers, a fim de manter o estado dos dispositivos atualizados e garantindo a confiabilidade de informação. Desta maneira, a seleção dos sensores ocorre em duas etapas: (i) Uma consulta é feita a um repositório responsável por armazenar a descrição dos recursos por meio de metadados, no qual os recursos que respeitarem as restrições impostas são retornados; (ii) posteriormente, uma nova consulta é feita aos buffers que armazenam os dados coletados de tais recursos para verificar o estado atual dos mesmos. Se os recursos ainda respeitarem as condições desejadas pelos usuários, os dados correspondentes a eles são retornados. Uma avaliação qualitativa da arquitetura é apresentada mostrando a diferença entre ela e as arquiteturas disponíveis na Web atualmente. Além disso, foi estimado por meio de simulação a convergência do tráfego atual da Web utilizando o modelo geográfico japonês, no qual foi possível constatar que há redução no tráfego da rede devido a distribuição dos buffers de armazenamento .

Zou *et al.* (2016) apresentam uma abordagem para o gerenciamento de workflows descentralizado para instanciamento autônomo e execução de workflows com fluxo de dados e composição de serviços sob demanda. Dada um modelo de especificação, esta abordagem consegue desacoplar os diferentes estágios de um workflow, permitindo que a sua execução seja realizada em diferentes estágios por meio de serviços individuais. Um componente denominado agendador automático é responsável por descobrir os serviços disponíveis em toda a infraestrutura para cada passo do workflow, além de instanciar e gerenciar o fluxo de dados dos serviços. A seleção dos serviços visa minimizar a sobrecarga de transferência e garantir o QoS além de balancear a utilização dos recursos por toda a rede. Com o intuito de selecionar os recursos mais adequados de acordo com as restrições impostas por um usuário, os atributos de QoS do serviço

são normalizados utilizando uma função com peso. Após esta etapa, os valores normalizados são utilizados para classificar os recursos de acordo com os níveis de QoS oferecidos e também de utilização da rede. Os experimentos realizados mostraram que a arquitetura distribuída permitiu processar mais requisições concorrentes em relação a uma abordagem centralizada, sem prejudicar a QoS, uma vez que o maior problema em uma infraestrutura centralizada é a rede que passa a se tornar o gargalo quando muitas requisições são feitas simultaneamente.

Girau, Martis e Atzori (2017) utilizam o conceito de IoT social, na qual cada objeto é um agente autônomo capaz de estabelecer relações sociais com outros objetos, para propor uma arquitetura em nuvem para IoT denominada Lysis, organizada em quatro camadas: A camada denominada “*Real World*” é responsável por conectar os dispositivos físicos a Internet. Os dispositivos pertencentes a essa camada são capazes de realizar tarefas básicas como comunicação segura com outros objetos virtuais bem como o gerenciamento e apresentação dos dados oriundos dos sensores. Acima desta camada está a camada denominada “*Virtualization*”, responsável por prover a interface entre os dispositivos físicos e virtuais. Esta camada utiliza esquemas padrão por meio de descritores semânticos para representar as funcionalidade e características de objetos virtualizados. A camada “*Aggregation*” é responsável por transformar os objetos virtuais em entidades com funcionalidades estendidas, chamados de “*Micro Engines*” que implementam parte da lógica de programação necessária na camada de aplicação. As “*Micro Engines*” possibilitam a reutilização de um fluxo de dados por diferentes aplicações com o intuito de economizar largura de banda e processamento. A última camada é chamada de “*Application*”, a qual prove serviços aos usuários do *framework*. Os resultados deste trabalho foram analisados tanto em termos qualitativos comparando as características da arquitetura Lysis com outras disponíveis quanto quantitativos no qual a arquitetura Lysis apresentou um bom desempenho para cenários que utilizam muitos dispositivos.

Tanganelli, Vallati e Mingozi (2018) propõem uma arquitetura distribuída focada na borda do ambiente IoT para realizar a descoberta de recursos e o acesso aos serviços IoT. Uma solução federada é apresentada para organizar gateways IoT próximos aos dispositivos da borda, denominados *fog nodes*. O objetivo é fornecer um serviço global descentralizado para realizar a descoberta e acesso de recursos IoT, por meio da cooperação entre os gateways situados próximos aos dispositivos de borda. Os gateways IoT são organizados em uma arquitetura P2P, implementada por meio de uma *Distributed Hash Table* (DHT).

Zhou *et al.* (2016) apresentam um *survey* realizando uma comparação qualitativa e uma análise crítica entre os diferentes tipos de técnicas de busca de recursos em ambientes de IoT, realizando a classificação dos trabalhos apresentados em termos de i) fundamentos de busca, ii) representação dos dados e conhecimento e iii) conteúdo a ser buscado. Desta maneira, algumas métricas foram utilizadas para realizar a comparação entre os trabalhos, tais como o formato dos dados, a forma de acesso aos resultados, o tipo de busca realizado (palavras-chave, índices, semântico, busca espacial), escala dos experimentos, dinamicidade, arquitetura e implementação.

Dentre as principais conclusões dos autores, pode-se destacar que é muito difícil realizar a comparação entre os métodos existentes em termos de escalabilidade e *throughput* uma vez que os bancos de dados utilizados não são disponibilizados. Como conclusão principal, os autores destacam que uma boa prática para o desenvolvimento de arquiteturas IoT é manter os metadados dos objetos em um repositório centralizado enquanto que os fluxos de dados devem ser armazenados em bancos de dados distribuídos. Além disso, os autores enfatizam que pelo fato do conteúdo a ser buscado nos trabalhos considerados ser significativamente diferente uns dos outros, eles não puderam realizar uma avaliação comparando as técnicas de busca e seleção destes trabalhos.

3.2 Técnicas de Seleção de Recursos

Elahi *et al.* (2009) propõem uma primitiva denominada *sensor ranking* para realizar a busca dos sensores por conteúdo de forma eficiente. A ideia principal do *sensor ranking* é explorar a periodicidade apresentada pelos sensores em alguns casos, por meio de modelos de predição que classificam os sensores de acordo com a probabilidade de atenderem a uma consulta de usuário. De acordo com esta classificação, os sensores são acessados e caso seu estado seja condizente com o obtido na probabilidade, são retornados ao usuário. Os modelos utilizados nos experimentos foram o *Single-Period* e o *Multi-Period* e os resultados obtidos pelos testes em dois conjuntos de dados reais possibilitaram observar um ganho significativo de desempenho durante a fase de seleção.

Ostermaier *et al.* (2010) apresentam um motor de busca para a *Web of Things* denominado *Dyser*, para realizar buscas escaláveis em ambientes com conteúdo altamente dinâmico. Seu motor de busca apresenta duas funcionalidades principais. A primeira funcionalidade é a busca feita a partir de metadados, a qual coleta e armazena dados e informações dos sensores. A segunda funcionalidade estende o trabalho de Elahi *et al.* (2009) para a geração do modelo de predição por meio do *Aggregated Prediction Model*. O modelo proposto neste trabalho foi comparado com os modelos *Single Period Prediction Model* e *Multi-Period Prediction Model* propostos em Elahi *et al.* (2009) e também com um modelo de probabilidade aleatória. Os resultados obtidos mostraram que os algoritmos apresentaram uma melhor qualidade de seleção quando comparados com o modelo aleatório.

Truong e Romer (2013) também estendem o trabalho de Elahi *et al.* (2009) ao propor o desenvolvimento dos modelos de predição utilizando lógica *fuzzy* chamado de *Time-Independent Prediction Model*, pois esta é capaz de detectar anomalias no comportamento dos sensores utilizando métricas de densidade e de estabilidade. A densidade estima a probabilidade de um determinado valor pertencer a um sensor específico enquanto que a estabilidade estima quão estáveis estes sensores foram no passado. A partir da combinação dessas probabilidades, os sensores são classificados e acessados para verificar seu estado. Desta maneira a solução

apresentada foi capaz de reduzir a comunicação necessária para a descoberta e seleção dos sensores.

Pfisterer *et al.* (2011) apresentam a arquitetura SPITFire para integrar diferentes redes de sensores e oferecer suas funcionalidades como serviço por meio de *linked-data*. Os autores estendem ontologias já existentes tais como *Dolce Ultralite* e *W3C Semantic Sensor Network* para integrar as descrições dos sensores com a nuvem *linked data*. Esta ontologia consegue aglomerar conjuntos de dados arbitrários para descrever processos mais complexos, detectar fatos e correlações. Os autores também propõem a criação semiautomática da descrição dos sensores, assim que estes são conectados à rede. A partir da coleta dos dados dos sensores, algoritmos de agrupamento são executados com o intuito de determinar a qual grupo aquele sensor pertence. A busca dos estados dos sensores no SPITfire ocorre por meio de modelos de predições baseados em seu histórico ou em modelos de correlação.

Ding *et al.* (2012) propõem um motor de busca híbrido para IoT, capaz de realizar as buscas por meio de valores quantitativos, palavras chaves e relações espaço-temporais. A arquitetura deste motor de busca é baseada em um modelo *bottom-up* de três camadas, sendo a primeira camada responsável por sensoriar e monitorar os equipamentos, a segunda por armazenar os dados obtidos de forma distribuída e a terceira camada por prover acesso otimizado aos dados dos sensores. A busca por palavras chaves e valores é otimizada por meio de uma árvore B+ enquanto que as buscas com relações espaço temporais utilizam uma árvore R. Este motor de busca permite a descoberta dos estados dos objetos em tempo de execução, uma vez que os sensores enviam continuamente os dados para a camada de armazenamento, a qual é indexada de acordo com a busca a ser realizada.

Guinard *et al.* (2010) propõem um módulo para a arquitetura de integração denominada SOCRADES, cujo objetivo é permitir a integração ubíqua de serviços executados em dispositivos embarcados com outros processos de negócio. O módulo proposto é baseado no modelo *Publish/Subscribe*, no qual todos os dispositivos existentes são descritos por meio de metadados armazenados em um repositório global. Este repositório funciona em conjunto com um monitor que é responsável por atualizar os estados e atributos de QoS dos dispositivos. Os sensores são descobertos por meio de palavras-chave e são classificados antes de serem retornados ao usuário de acordo com a prioridade de seus atributos de QoS. Os experimentos realizados foram capazes de descobrir mais sensores quando o número de palavras sinônimas variou entre 5 e 10, obtendo menos de 20% de falsos positivos.

Shah e Sardana (2012) apresentam um mecanismo de busca que utiliza o Sistema de Coordenadas Virtuais para a descoberta de processos, em redes P2P. Neste sistema, uma coordenada é atribuída a cada um dos nós representando sua localização física com relação a outros nós. As buscas são feitas por meio de palavras chaves, a qual retorna o conjunto de sensores correspondentes. O conjunto de sensores retornado é classificado de acordo com a distância euclidiana de um conjunto de atributos de QoS especificados pelo usuário tal como,

quantidade de energia, latência, memória e largura de banda em relação ao usuário. O mecanismo proposto foi validado por meio de uma avaliação qualitativa com outros *frameworks* existentes na literatura, tais como Snoogle, MAX e OCH em que a abordagem proposta é a única capaz de estabelecer a precisão da consulta em tempo real.

Ruta *et al.* (2013) os autores apresentam um *framework* para gerenciar as notações semânticas dos fluxos de dados, dispositivos, eventos de alto nível e serviços. As requisições são feitas por meio da extensão do protocolo CoAP implementando os conceitos arquiteturais RESTFul, permitindo a realização de inferências para suportar a descoberta automática dos sensores e suas composições, representando uma das principais contribuições deste trabalho. Um mecanismo simples de mineração de dados foi utilizado para anotar os estados dos sensores em tempo de execução visando melhorar a qualidade da seleção dos sensores. A seleção dos sensores é baseada em uma versão da inferência de *Concept Covering* seguida pela aplicação de um algoritmo de classificação. Um estudo de caso foi apresentado utilizando Vanets, em que o uso das técnicas de mineração permitiu inferir as condições de uma determinada estrada localizada na Itália e determinar se um tipo específico de carro está apto a trafegar naquelas condições.

Perera *et al.* (2013a) apresentam um *framework* denominado CASSARAM, o qual prioriza a seleção dos sensores de acordo com os atributos de QoS especificados pelo usuário. O processo de seleção ocorre em duas fases. Na primeira são considerados os requisitos que correspondem aos atributos estáticos dos sensores, como por exemplo, seu tipo ou seu fabricante limitando assim o espaço de busca do usuário. Na segunda fase, o resultado da primeira busca é colocado em um espaço multidimensional em que cada eixo corresponde a um dos atributos de QoS do usuário. Em seguida, os sensores são indexados por meio do algoritmo *Comparative Priority-based Weighted Index* e classificados segundo sua distância euclidiana ao ponto ótimo. Os autores também propõem uma heurística chamada de *Comparative Priority-based Heuristic Filtering*, a qual remove o conjunto de sensores que se encontram muito longe dos valores ideais do usuário, priorizando a seleção TOP-K. Os resultados obtidos mostraram que ao utilizar um número reduzido de sensores (até 10.000) o *framework* apresentou um desempenho satisfatório na seleção com um alto índice de precisão.

Carlson e Schrader (2014) apresentam um motor de busca chamado *Ambient Ocean* para a descoberta e seleção de sensores de acordo com informações de um contexto. Este motor utiliza metadados armazenados em um repositório global para definir o contexto dos sensores e realizar as buscas de uma maneira mais eficiente e eficaz. O motor de busca emprega modelos de similaridade multitarefas baseados no algoritmo *Weighted Slope One*. Em cenários em que as características dos sensores são difíceis de modelar, técnicas de filtro colaborativo são utilizadas para computar a similaridades entre os usuários ou sensores a partir do histórico de informações. Os resultados preliminares indicaram que a solução proposta melhorou os aspectos qualitativos e quantitativos dos recursos em ambientes reais.

Kothari *et al.* (2014) apresentam a arquitetura denominada DQS-Cloud, para otimizar a descoberta dos sensores, prover de forma autônoma resiliência as faltas e degradação dos atributos de qualidade e também otimizar o desempenho do sistema gerenciando os fluxos de dados dos sensores. A busca dos sensores é baseada em palavras-chaves e nos atributos de QoS desejados pelos usuários. Além disso, com o intuito de reduzir a sobrecarga de comunicação, este trabalho propõe um mecanismo de otimização para o reaproveitamento dos fluxos de sensores para requisições com atributos de QoS similares. Ainda, uma abordagem para tolerância a faltas e degradação dos atributos de QoS são descritas neste trabalho, os quais são monitorados por um módulo específico da arquitetura. Os resultados obtidos mostraram que o módulo de otimização foi capaz de reduzir a taxa de largura de banda e processamento dos provedores.

Gao, Ali e Mileo (2014) apresentam o *Automated Complex Event Implementation System* para integrar diferentes fluxos de dados em tempo de execução. Os sensores e seus fluxos de dados são descritos segundo a ontologia SSN e são armazenadas em um repositório juntamente com os seus atributos de QoS e *Quality of information* (QoI). Assim, o sistema age como um *middleware* entre o fluxo de dados de um sensor e uma determinada aplicação. O *middleware* é capaz de buscar e selecionar os sensores registrados no repositório respeitando os atributos de QoS e QoI especificados pela aplicação e classificando os de acordo com o algoritmo SAW. Porém, embora o *middleware* proposto seja descrito com uma rica quantidade de detalhes, a solução proposta não foi avaliada quantitativamente ou qualitativamente.

Ruta, Scioscia e Sciascio (2015) propõem um serviço para realizar o *matchmaking* semântico para auxiliar na descoberta de recursos e serviços em contextos ubíquos, especialmente em ambientes em que há limitações de recursos. Este serviço é capaz de lidar com inferências com ou sem padrão a fim de permitir a categorização e classificação dos serviços de acordo com as especificações de uma requisição por meio de algoritmos que exploram os conceitos de contração, abdução e abrangência. Os experimentos foram conduzidos em computadores pessoais e celulares e evidenciaram a corretude da implementação e do desempenho em ambos cenários.

Bijarbooneh *et al.* (2016) propõem um esquema para coleta de dados chamado de *Sensing Algorithm with Belief Propagation* (ASBP), o qual utiliza correlações de dados espaço-temporal para reduzir o número de nós ativos na rede e minimizar o consumo de energia. Para a seleção de recursos, eles propõem um modelo de programação voltado a restrições a fim de encontrar a solução global ótima e um algoritmo baseado em heurística para encontrar a solução ótima local, a qual é adequada em cenários em que a seleção de sensores não pode ser resolvida de maneira centralizada. Eles compararam a taxa de erros dos resultados, do modelo de programação voltado à restrições, do algoritmo baseado em heurística e da seleção aleatória otimizando a qualidade de seleção dos canais de utilização em 30% e 70%.

Gong, Shi e Li (2016) propõem o *Context-aware Sensor Search Framework*, para melhorar a seleção de recursos em um conjunto de dados que armazena informações semânticas.

Eles utilizam o *Threshold Algorithm for Sensor Information* (TASI) para reduzir o escopo da consulta e o custo computacional do processo de descoberta de recursos. Em seguida, é aplicado um filtro de comparação de prioridades baseados em peso para classificar os recursos disponíveis de acordo com o contexto em que estão inseridos e as prioridades definidas pelo usuário.

Huang *et al.* (2016) descreve um framework para a mineração de serviços para a descoberta de relacionamentos no contexto de IoT. Os serviços são descritos de acordo com uma ontologia considerando os aspectos espaço-temporais, ambiente, pessoas e operações disponíveis de acordo com as restrições. Os filtros para descoberta de recurso eliminam os serviços não relacionados em duas etapas: (i) computando o grau de correlação medindo a força da relação entre dois serviços e (ii) estimando o grau de interesse, o qual considera a disponibilidade, o domínio e a diversidade do serviço candidato.

Ying, Guan-Yu e Zhong-Jun (2016) propõem o serviço de diretório de ontologias baseado em divisão comunitária, para combinar características das ontologias e diretórios de serviço a fim de organizar e gerenciar os serviços de forma efetiva. Neste serviço, primeiramente é feito um resumo das ontologias para diminuir o volume de processamento, para que uma rede P2P social seja dividida em comunidades semânticas por meio de algoritmo autônomo de geração de comunidades denominado *Self-organizing Formin procedure of ASC* (SoFA). Diferentemente do serviço tradicional de diretórios, o diretório de ontologias é gerado dinamicamente com o intuito de obter os melhores serviços de acordo com as restrições estabelecidas por cada usuário.

Yan *et al.* (2016) apresentam uma plataforma para o registro de serviços denominada UBI-REGI, utilizada para auxiliar a descoberta de serviços em redes ubíquas. Esta plataforma divide os serviços em três categorias: serviço de fonte, serviço de transformação e serviço de sincronização. O serviço de fonte é o responsável por obter a informação física do objeto real e exportá-la como dados digitais. O serviço de transformação é responsável por transformar os dados recebidos em um conjunto de dados mais significativo. O serviço de sincronização é aquele que impacta um ambiente no mundo real, como por exemplo controlar o ar condicionado de uma sala automaticamente. Para facilitar a busca, são utilizados metadados específicos do objeto como localização física ou proprietário em conjunto com metadados típicos de tais sistemas como nome, finalidade e descrição. Um estudo de caso é apresentado como resultado, no qual pode-se realizar a integração de diferentes serviços em uma casa inteligente.

Jung *et al.* (2016) propõem um modelo de rede IoT baseado em hipergrafos, dividindo a rede em três camadas. A camada de tarefas, contém as tarefas finais definidas pelos usuários, da qual serão extraídas as restrições para realizar a descoberta dos objetos. A camada de interface utiliza a *Semantic Sensor Network Ontology* (SSNO) para acessar o histórico dos objetos e estabelecer suas relações de dependência e similaridade com outros objetos pertencentes a rede. A camada de interação, captura as interações entre os objetos na rede juntamente com seus elementos temporais e espaciais. A busca e a seleção dos dispositivos, utiliza *Social Internet of Things* (SIoT), atribuindo aos objetos pertencentes a esta rede perfis virtuais além dos

seus metadados, uma vez que algumas informações são mais significativas do que outras. As informações do perfil são usadas para instanciar e definir relações sociais de forma dinâmica. Para realizar a busca e a descoberta dos objetos, o algoritmo proposto busca pelas interfaces semânticas que correspondem com as restrições impostas pelos usuários. Após este filtro inicial, o mecanismo de descoberta atribui uma pontuação para cada objeto candidato, considerando a similaridade do perfil, do serviço e da relação entre os objetos. Os resultados apresentados, evidenciou que o método proposto foi melhor que os métodos convencionais existentes na literatura que fazem análise da construção do hipergrafo, em termos de tempos de processamento e execução e da influência da dinamicidade dos objetos.

Sun *et al.* (2017) endereçam o problema de composição de serviços em IoT, partindo do princípio que um único dispositivo pode oferecer múltiplos serviços, e assim prejudicando a sua eficiência energética e desempenho. Cadeias de classes de serviço são geradas respeitando as restrições impostas pelos usuários, limitações espaciais e temporais, eficiência energética e configurabilidade. Neste cenário, a seleção dos serviços é instanciada como um problema multi-objetivo e três algoritmos são analisados para realizar a composição dos serviços sendo eles: algoritmos genéticos, colônia de formigas e enxame de partículas. Os resultados apresentados consideraram a função de *fitness*, o resíduo mínimo e a variância da energia dispendida pelos objetos. Os resultados mostraram que o algoritmo de enxame de partículas apresentou os melhores resultados na busca pela composição ótima de serviços, reduzindo e balanceando o consumo de energia e prolongando o tempo de vida útil da rede.

Courtais *et al.* (2017) descrevem o middleware IoTVar para o gerenciamento de interações entre as aplicações de usuários finais e as plataformas por meio de IoTVar *proxys*. O objetivo deste middleware é permitir que os desenvolvedores de software declarem variáveis IoT em suas aplicações. O funcionamento do middleware pode ser descrito como: os sensores interagem com qualquer plataforma IoT para publicarem seus dados na Internet. As aplicações dos usuários finais usam o middleware IoTVar para interagirem com essas plataformas e obter os dados por meio de variáveis IoT. O middleware busca e seleciona sensores respeitando as restrições impostas pelo usuário. Para realizar a busca dos sensores, o desenvolvedor deve definir o tipo do sensor a ser buscado, sua localização em termos de latitude e longitude juntamente com um raio de busca e o período entre as atualizações. A interação com os objetos pode ocorrer por meio de três padrões de interação: (i) chamada síncrona, (ii) *publish/subscribe* e *public/subscribe* com filtro de conteúdo no servidor. Os resultados mostraram que IoTVar necessitou de um número menor de linhas para interagir com qualquer plataforma utilizando um código de 15 e 18 linhas respectivamente para realizar chamadas síncronas e *publish/subscribe* contra códigos de 300 e 500 linhas apresentada pelo *middleware* QoDisco utilizado para esta comparação.

Na Tabela 2 são sintetizadas as principais características dos trabalhos apresentados nesta Seção. De acordo com esta tabela é possível observar que os trabalhos se concentram em sua maioria em técnicas de pontuação e classificação e consideraram tanto a utilização de

Tabela 2 – Síntese das principais características dos trabalhos relacionados.

Paper	Técnica de Busca	Busca	Método de Seleção
Elahi <i>et al.</i> (2009)	Modelo de Predição	Estado do Sensor	Modelo de Predição Single-Period e Multi-Period
Ostermaier <i>et al.</i> (2010)	Modelo de Predição	Estado do Sensor	Modelo de Predição Agregado
Truong e Romer (2013)	Modelo de Predição	Estado do Sensor	Modelo de predições baseado em histórico ou correlação
Pfisterer <i>et al.</i> (2011)	Modelo de Predição	Estado do Sensor	Modelo de predição Time-Independent
Ding <i>et al.</i> (2012)	Índices	Palavras-chave, Estado do Sensor e Informações de Contexto	Estruturas Árvore B+ e Árvore R
Guinard <i>et al.</i> (2010)	Pontuação e Classificação	Palavras Chave	Pontuação e Classificação baseada em prioridades
Kothari <i>et al.</i> (2014)	Pontuação e Classificação	Palavras Chave	Pontuação e Classificação
Ruta <i>et al.</i> (2013)	Pontuação e Classificação	Palavras Chave	Pontuação e Classificação utilizando o algoritmo de inferência <i>concept covering</i>
Shah e Sardana (2012)	Pontuação e Classificação	Palavras Chave	Filtro, Pontuação e Classificação de acordo com a distância euclidiana entre as restrições estabelecidas
Perera <i>et al.</i> (2013a)	Pontuação e Classificação	Informações de Contexto	Filtro, Pontuação e Classificação com o algoritmo Comparative Priority-based Heuristic
Carlson e Schrader (2014)	Agrupamento e Filtro	Informações de Contexto	Agrupamento baseado no algoritmo <i>Weighted Slope One</i> e aplicação de filtro colaborativo
Gao, Ali e Mileo (2014)	Pontuação e Classificação	Informações de Contexto	Pontuação e Classificação com o algoritmo SAW
Ruta, Scioscia e Sciascio (2015)	Pontuação e Classificação	Informações de Contexto	Pontuação e Classificação em termos de contração, abdução e abrangência
Bijarbooneh <i>et al.</i> (2016)	Pontuação e Classificação	Informações de Contexto	Pontuação e Classificação por meio do <i>Sensing Algorithm with Belief Propagation</i>
Gong, Shi e Li (2016)	Pontuação e Classificação	Informações de Contexto	Filtro, Pontuação e Classificação de acordo com prioridades
Huang <i>et al.</i> (2016)	Filtro	Informações de Contexto	Filtro e Agrupamento considerando o grau de correlação e interesse dos serviços
Ying, Guan-Yu e Zhong-Jun (2016)	Agrupamento	Informações de Contexto	Agrupamento de acordo com o algoritmo Self-organizing Forming procedure of ASC
Yan <i>et al.</i> (2016)	Filtro	Palavras Chave	Filtro
Jung <i>et al.</i> (2016)	Pontuação e Classificação	Informações de contexto	Filtro, Pontuação e Classificação de acordo com a similaridade do perfil, serviço e interação entre objetos
Sun <i>et al.</i> (2017)	Pontuação e Classificação	Informações de Contexto	Pontuação e Classificação considerando algoritmos genéticos, colônia de formigas e enxame de partículas
Courtais <i>et al.</i> (2017)	Filtro	Palavras Chave	Filtro
Proposta	Pontuação e Classificação Relações de Dominância	Informações de Contexto	Filtro por meio da aplicação de pontuação e classificação. Cálculo das relações de dominância dos recursos após o filtro.

Fonte: Elaborada pelo autor.

palavras-chave como informações de contexto. Apesar desta técnica ser a mais difundida para realizar a seleção dos sensores, basicamente elas utilizam um algoritmo para a tomada de decisão monocritério, o qual realiza uma redução dos múltiplos critérios por meio de ponderações para um único valor e por isso não avaliando a sua qualidade.

Esta tese propõe uma metodologia para auxiliar a avaliação da qualidade da solução proposta por tais algoritmos. Além disso, são propostos dois novos algoritmos para realizar a tomada de decisão utilizando as técnicas de pontuação e classificação como um filtro. Os recursos resultantes após a aplicação do filtro são selecionados a partir do estabelecimento das relações de dominância e da fronteira de Pareto.

3.3 Considerações Finais

Neste capítulo foi apresentada uma revisão do estado da arte atual de IoT, mais especificamente abordando o problema de busca e seleção de sensores. Os trabalhos encontrados foram classificados em dois grupos. O primeiro grupo descreveu os trabalhos que propunham algum tipo de arquitetura para possibilitar a seleção de recursos em IoT. Já o segundo grupo de trabalho retratou os trabalhos que apresentaram mecanismos ou algoritmos para realizar a seleção dos recursos propriamente ditos. Ao final dessa seção, uma tabela foi utilizada para sintetizar as técnicas utilizadas em cada trabalho. No próximo capítulo será apresentada a metodologia proposta para avaliar em termos de qualidade as técnicas utilizadas para realizar a seleção de recursos. Além disso, dois algoritmos são propostos com o intuito de melhorar a qualidade e o tempo necessário para realizar a seleção dos recursos.

MÉTODOS PARA A TOMADA DE DECISÃO MULTICRITÉRIO EM IOT

4.1 Algoritmos

Como descrito na Seção 2.2, apesar do algoritmo FNDS garantir que a solução com a melhor qualidade seja selecionada, sua alta complexidade de tempo e armazenamento dificultam sua execução quando há um grande número de recursos disponíveis. Por outro lado, as técnicas MCDA descritas na Seção 3 propõem um conjunto de soluções que exigem menos recursos computacionais, mas que possuem baixa qualidade.

Visando melhorar a relação entre a qualidade de uma solução e o seu custo computacional, esta tese propõe dois algoritmos: *Elimination Sort* (ES) e *Fast Elimination Sort* (FES).

Tanto o algoritmo ES quanto o FES combinam os algoritmos TOPSIS e FNDS para realizar a seleção. O algoritmo TOPSIS é utilizado como um filtro, para eliminar os recursos com baixa probabilidade de serem ótimos. O algoritmo FNDS por sua vez é aplicado sobre os recursos resultantes do filtro, com o intuito de estabelecer as relações de dominância destes recursos e propor a solução. A Tabela 3 contém a representação das variáveis utilizadas para descrever os algoritmos.

Tabela 3 – Descrição das variáveis, tipos e restrições utilizadas para a nos algoritmos ES e FES

Variável	Descrição	Tipo	Restrição
Q	Recursos disponíveis	-	-
N	Número de recursos a serem selecionados	Inteiro	$N \leq tamanho(Q)$
SR	Taxa de busca dos recursos	Real	$1 \leq SR * N \leq tamanho(Q)$
S	Recursos selecionados	-	-
P	Porcentagem	Real	$0.0 \leq P \leq 1.0$

Fonte: Elaborada pelo autor.

O Algoritmo 2 apresenta a descrição do algoritmo ES. Tal algoritmo utiliza três parâmetros de entrada Q , N e SR , os quais correspondem ao conjunto de recursos disponíveis, o número de recursos a serem selecionados e à taxa de recursos a serem recuperados após o filtro, respectivamente. A variável SR foi incluída após estudos exploratórios (Capítulo 5), os quais constatarem que a qualidade dos algoritmos aumentava à medida que mais recursos eram selecionados.

Algoritmo 2 – Elimination-Selection

```

1: função E-S( $Q, N, SR$ )
2:    $list \leftarrow TOPSIS(Q)$ 
3:    $S \leftarrow fast\_nondominated\_sort(list[1..N \times SR])$ 
4: retorna  $S[1..N]$ 
5: fim função
  
```

Na Figura 7 é sintetizado o funcionamento do algoritmo ES. Primeiramente, o algoritmo TOPSIS é utilizado para pontuar e classificar o conjunto de recursos disponíveis Q , gerando uma lista. Em seguida, o algoritmo FNDS é executado sobre os primeiros $N \times SR$ recursos desta lista. O parâmetro SR tem um papel crucial nesta etapa, uma vez que é utilizado para aumentar a probabilidade de se obter uma solução ótima pelo algoritmo TOPSIS e minimizar a complexidade temporal e de armazenamento do algoritmo FNDS. Por fim, os N recursos pertencentes às primeiras fronteiras são retornados.

Figura 7 – Síntese do funcionamento do algoritmo ES. Na primeira etapa o algoritmo TOPSIS é aplicado sobre os Q recursos disponíveis gerando uma sublista com $SR \times N$ primeiros recursos classificados. As relações de dominância desta lista são computadas por meio do algoritmo FNDS, o qual gera a lista a ser retornada S com os N recursos localizados nas primeiras i fronteiras dominantes.

Fonte: Elaborada pelo autor.

O Algoritmo 3 apresenta a descrição do algoritmo FES. Este algoritmo utiliza quatro parâmetros de entrada, sendo três deles (Q , N e SR) iguais ao do Algoritmo 2. O quarto parâmetro P , refere-se a uma porcentagem, utilizada para determinar a parcela da lista retornada após a aplicação do filtro, que terá as relações de dominância computadas.

Na Figura 8 é sintetizado o funcionamento do algoritmo FES. Primeiramente, tal como no algoritmo ES, o algoritmo TOPSIS é utilizado para pontuar e classificar o conjunto de

Algoritmo 3 – Fast Elimination-Selection

```

1: função E-S( $\mathbb{Q}, \mathbb{N}, \mathbb{SR}, \mathbb{P}$ )
2:    $list \leftarrow TOPSIS(\mathbb{Q})$ 
3:    $sublist_1 \leftarrow list[1, \mathbb{N} \times (1 - \mathbb{P})]$ 
4:    $sublist_2 \leftarrow fast\_nondominated\_sort(list[\mathbb{N} \times (1 - \mathbb{P}), \mathbb{N} \times (1 - \mathbb{P}) \times \mathbb{SR}])$ 
5:    $\mathbb{S} \leftarrow sublist_1 | sublist_2[1.. \mathbb{N} \times \mathbb{P}]$ 
6: retorna  $\mathbb{S}$ 
7: fim função

```

recursos disponíveis $\mathbb{Q}$. Os primeiros $(\mathbb{N} \times (1 - \mathbb{P})) - 1$ são salvos na sublista₁, a qual não sofrerá nenhuma operação adicional. Já a sublista₂, é obtida por meio da aplicação do algoritmo FNDS sobre os sensores classificados entre $\mathbb{N} \times (1 - \mathbb{P})$ e $\mathbb{N} \times (1 - \mathbb{P}) \times \mathbb{SR}$, reduzindo assim o número de relações de dominância a serem computadas nesta etapa. O resultado da seleção dos recursos é composto pelos recursos da sublista₁ concatenados com os recursos da sublista₂.

Figura 8 – Síntese do funcionamento do algoritmo FES. Na primeira etapa, o algoritmo TOPSIS é aplicado sobre os $\mathbb{Q}$ recursos disponíveis gerando duas sublistas. A primeira sublista contém os $\mathbb{N} \times (1 - \mathbb{P})$ primeiros recursos classificados e que serão utilizados no resultado final como $S1$. A segunda sublista contém os próximos $(\mathbb{N} \times \mathbb{P} \times \mathbb{SR})$ recursos. As relações de dominância dessa sublista são computadas por meio do algoritmo FNDS, o qual gera a lista a ser retornada $\mathbb{SR}$ com os $\mathbb{N} \times \mathbb{P}$ recursos localizados nas primeiras i fronteiras dominantes. A lista final $\mathbb{S}$ é formada pela união dos recursos contidos em $S1$ e $S2$ totalizando os $\mathbb{N}$ recursos.

Fonte: Elaborada pelo autor.

4.2 Metodologia de Avaliação

A metodologia proposta para avaliar a qualidade de seleção das técnicas utilizadas na seleção de recursos em ambientes IoT é sintetizada pelo fluxograma da Figura 9, cujas as etapas podem ser descritas como:

1. Para executar e avaliar uma técnica MCDA os seguintes parâmetros de entrada devem ser considerados: o arquivo utilizado para descrever os recursos, contendo todas as informações de contexto e dados coletados de um determinado recurso; Os critérios avaliados, que definem quais dados serão extraídos do arquivo que descreve um recurso e analisados

Figura 9 – Fluxo utilizado para avaliar as técnicas de seleção aplicados no ambiente IoT

Fonte: Adaptada de Nunes *et al.* (2016b).

pela técnica MCDA; Opcionalmente, algumas técnicas MCDA permitem que os critérios avaliados tenham sua importância definida por meio da atribuição de pesos.

2. Execução da técnica MCDA a ser avaliada considerando-se os dados extraídos do descritor de recursos correspondente a cada critério definido. As técnicas MCDA aplicam tais dados em modelos matemáticos de acordo com a função objetivo de cada critério (geralmente maximização ou minimização), a fim de retornar uma lista com a classificação ou agrupamento dos melhores recursos disponíveis.
3. Para cada recurso classificado após a execução da técnica MCDA é definido o índice da fronteira a que ele pertence. Utilizando o conceito de otimalidade de Pareto é possível analisar e comparar as técnicas MCDA em termos qualitativos, por meio das métricas descritas na Seção 2.2.3.

4.3 Indicadores de Desempenho

Embora a métrica ONVGR estime a qualidade de uma solução, a sua análise torna-se complexa quando as técnicas MCDA apresentam comportamento semelhante, uma vez que as fronteiras possuem importâncias diferentes e por isso devem ser comparadas individualmente.

Com o intuito de facilitar a comparação da qualidade entre as técnicas MCDA, o indicador de desempenho Perda de Qualidade Acumulada (PQA) é proposto. Este indicador de desempenho sintetiza a perda acumulada da qualidade de uma solução, por meio dos valores do $ONVGR_{ideal}$ (1) e do $ONVGR_i$ obtido pela técnica MCDA na fronteira de índice i , como descrito na Figura 10.a. Ao determinar o número N de recursos de uma solução, há a possibilidade de haver diferença entre os valores $ONVGR_{ideal}$ (1) e do $ONVGR_i$, mesmo utilizando o algoritmo FNDS para se obter a solução ótima, pois nem todos os recursos da última fronteira são selecionados, tal como delineado na Figura 10.b.

Figura 10 – Perda da qualidade da seleção de N recursos por meio da comparação da métrica ONVGR ideal com a obtida pelas técnicas MCDA (a) e pelo algoritmo FNDS (b) utilizado para computar a solução ideal.

Fonte: Elaborada pelo autor.

Figura 11 – Valor da importância relativa das fronteiras de acordo com o número total de fronteiras da solução ótima. A importância relativa da fronteira é relacionada ao valor do seu índice e do número total de fronteiras.

Fonte: Elaborada pelo autor.

Este indicador de desempenho também deve considerar a importância das soluções de acordo com índice da fronteira a qual pertence, uma vez que quanto menor o índice da fronteira mais relevante é a sua solução. Assim, a diferença entre o $ONVGR_{ideal}$ e $ONVGR_i$ é dividida pelo valor do seu índice i elevado a $\log_{\sqrt{n_{ideal}}} i$, permitindo se computar a Perda de Qualidade Relativa (PQR) na fronteira i .

O valor n_{ideal} corresponde ao número total de fronteiras para se computar a solução ótima com N recursos, possibilitando determinar a importância das fronteiras de uma solução dinamicamente, de acordo com o número total de fronteiras e o valor do seu índice. Na Figura 11, é ilustrada a alteração da importância relativa de cada fronteira quando N é igual a 10 e 50

fronteiras.

A fórmula utilizada para calcular o indicador de desempenho PQA é descrita pela Equação 4.1, em que se utiliza a soma do valor da perda da qualidade relativa de cada fronteira para se determinar a perda da qualidade acumulada da solução.

$$PQA = \sum_{i=1}^n \frac{1 - ONVGR_{front(i)}}{i^{\log_{\sqrt{n_{ideal}}} i}} \quad (4.1)$$

4.4 Considerações Finais

Este capítulo apresentou as principais contribuições desta tese. Os algoritmos FES e ES são propostos, a fim de melhorar a qualidade no processo de seleção de recursos e reduzir o seu custo computacional. Ainda, uma metodologia é proposta para avaliar e comparar as diferentes técnicas utilizadas para realizar a seleção de recursos, bem como um indicador de desempenho para sintetizar a qualidade absoluta de uma solução. No próximo capítulo são descritos os estudos exploratórios utilizados para validar a metodologia proposta nesta tese, descrevendo por meio de um estudo de caso os resultados obtidos em termos de qualidade das técnicas TOPSIS, SAW e VIKOR.

ESTUDO EXPLORATÓRIO DAS INFRAESTRUTURAS E TÉCNICAS DE SELEÇÃO DE RECURSOS EM AMBIENTES IOT

Para representar um ambiente mais próximo dos descritos para a IoT, o estudo de caso para a realização de todos os experimentos desta tese foi definido a partir de um repositório de dados composto por centenas de milhares de recursos.

O estudo de caso desta tese está inserido no domínio da agricultura, na qual as condições climáticas, como temperatura, precipitação, luminosidade e umidade desempenham um papel fundamental no domínio agrícola, pois, podem aumentar os custos de manutenção de uma plantação bem como sua vulnerabilidade a certas doenças e pragas (TIMÓTEO; MARCOS-FILHO, 2013).

Muitos agricultores utilizam suas próprias experiências e as características históricas regionais para estabelecer os períodos de plantação e de colheita, visando obter as condições climáticas adequadas para maximizar o desenvolvimento da produção. No entanto, com o aquecimento global colaborando para o surgimento de fenômenos como *la Niña* e *el Niño*, essas condições podem mudar ao longo do tempo e influenciar os tipos de culturas mais adequados para uma região (TIMÓTEO; MARCOS-FILHO, 2013).

Por exemplo, o milho é a cultura mais produzida no mundo, e mais adaptada a diferentes ecossistemas, apresentando mais de 300 variações (VAR, 2010). O ciclo de vida do milho pode ser dividido em quatro etapas: crescimento de mudas, crescimento vegetativo, fertilização e floração, maturidade e colheita dos grãos. Para cada etapa, algumas restrições climáticas devem ser respeitadas para que o desenvolvimento ideal seja atingido. Além disso, as diversas variedades de milho podem reagir de forma diferente às mudanças na mesma condição climática

(TIMÓTEO; MARCOS-FILHO, 2013).

Os trabalhos discutidos na Seção 3 podem auxiliar na escolha da região para iniciar a cultura de uma determinada espécie de milho considerando suas restrições climáticas. Especificamente, este trabalho visa aplicar técnicas MCDA para selecionar os recursos (sensores e serviços IoT) que indiquem as localizações mais adequadas para se cultivar uma determinada espécie de milho.

Os recursos que representam as condições climáticas são recuperados da API OpenWeatherMap¹, a qual possui mais de 200.000 recursos que descrevem as condições climáticas de diversas partes do mundo. Cada recurso pode conter até 13 propriedades de contexto.

As condições climáticas que podem impactar na fase de crescimento da plantação foram baseadas em informações fornecidas nos trabalhos de Doblas-Reyes *et al.* (2003) e Rosenzweig *et al.* (2001). Os experimentos realizados nesta tese consideram as seguintes restrições meteorológicas: (i) Temperatura (t) entre 59 F e 91,4 F; (ii) Umidade (h) maior que 50 %; (iii) cobertura do céu (sc) maior que 50 %; (iv) vento (w) inferior a 20 km / h e v) precipitação (p) entre 0 mm e 50 mm.

Estas propriedades de contexto, suas representações no OpenWeatherMap, símbolos, valores ótimos e funções objetivo são apresentadas na Tabela 4. Para simplificar o estudo de caso descrito neste trabalho, os critérios cujas funções objetivo envolvam um intervalo de valores são modeladas como uma parábola, na qual o ponto que atinge o valor máximo da equação corresponde ao valor da média do intervalo.

Tabela 4 – Cenário simplificado com a descrição das condições climáticas obtidas na plataforma OpenWeatherMap para o cultivo do milho.

Critério	Descrição OpenWeatherMap	Símbolo	Condição Ideal	Função Objetivo
Temperatura	temp	t	$59F < t < 91.4F$	$\max(-0.01777778t^2 + 10.512t - 1552.9364)$
Umidade	humidity	h	$h > 50\%$	$\max(h)$
Cobertura de Nuvens	clouds	sc	$sc > 50\%$	$\max(sc)$
Vento	wind	w	$w < 20km/h$	$\min(w)$
Precipitação	rain	p	$0mm < p < 50mm$	$\max(-0.0016p^2 + 0.08p)$
Dia/hora	dt	d	Mais recente	$\max(d)$

Fonte: Elaborada pelo autor.

5.1 Estudo exploratório das infraestruturas de seleção de recursos

Como mostrado no Capítulo 3, diversos trabalhos propõem arquiteturas para solucionar os mais variados problemas em IoT, sendo em sua maioria arquiteturas voltadas para a integração

¹ OpenWeatherMap - <http://bulk.openweathermap.org/sample/hourly_16.json.gz>

de diferentes objetos. Entretanto, poucos trabalhos se preocupam em como oferecer tais objetos em forma de serviço.

A ferramenta *Visual Search for Internet of Things* (ViSIoT) foi desenvolvida para auxiliar a captura, o gerenciamento e a integração de recursos em ambientes IoT. Esta ferramenta não visa substituir os *middlewares* discutidos na Seção 2.1.6, mas sim realizar a integração entre eles para auxiliar o processo de seleção dos recursos.

Esta ferramenta se baseia em uma arquitetura cliente-servidor, que tem como objetivo realizar experimentos em sistemas distribuídos, a qual é descrita em Nunes *et al.* (2015) e Nunes *et al.* (2014). Estes estudos foram utilizados como base deste trabalho pois foram capazes de configurar ambientes distribuídos de acordo com as preferências de usuários.

Esta ferramenta fornece acesso a um conjunto de recursos disponíveis como serviço, os quais podem alimentar qualquer tipo de aplicação ou *middleware* para prover seus dados como serviço. Além disso, a complexidade de se configurar tais ambientes é abstraída por meio de uma aplicação cliente que é responsável por implantar o mecanismo necessário para recuperar as informações desejadas.

5.1.1 Ferramenta ViSIoT

Na Figura 12 é descrita a organização dos módulos implementados pela ferramenta ViSIoT. A versão inicial da ferramenta ViSIoT utilizou a API *OpenWeather Map* como repositório, e o *middleware Global Sensor Networks* (GSN) como sistema do ambiente alvo. O módulo *Seletor* inicialmente foi desenvolvido com o algoritmo TOPSIS para selecionar os recursos. Uma interface Web também foi incorporada, para auxiliar a configuração do ambiente no qual a seleção será realizada.

5.1.1.1 Core

O módulo Core do protótipo ViSIoT foi desenvolvido utilizando conceitos de Orientação a Objetos com a linguagem Java, a qual é a linguagem base do sistema. Ele é o responsável por processar os dados obtidos da interface por meio de *servlets* e a partir disso coordenar a comunicação e a execução das tarefas de todos os módulos. Neste módulo também estão definidas as estruturas de dados padrão do sistema que serão utilizadas por tais módulos. Além disso, com a flexibilidade fornecida pela linguagem Java, este módulo pode ser facilmente estendido para suportar novas funcionalidades.

5.1.1.2 Repositório

As informações referentes aos recursos disponíveis em cada região da área de interesse deste estudo foram obtidas do repositório OpenWeatherMap. Os dados fornecidos por esta API

Figura 12 – Módulos da ferramenta ViSioT

Fonte: Adaptada de Nunes *et al.* (2016a).

estão disponíveis nos formatos *JavaScript Object Notation* (JSON), XML e *HyperText Markup Language* (HTML).

No Código-fonte 1 é representada a mensagem de resposta no formato JSON enviada pela chamada <http://api.openweathermap.org/data/2.5/weather?lat=35&lon=139>. Esta mensagem contém informações tais como localização, temperatura, umidade e pressão atmosférica fornecidas por um determinado recurso. Esta mensagem é interpretada pelo módulo Core e transformada em uma estrutura de dados, a qual será manipulada pelos outros módulos do protótipo **ViSioT!** (**ViSioT!**).

Arquivo 1 – Exemplo de arquivo JSON do OpenWeatherMap

```

1 { "coord": { "lon": 139, "lat": 35 },
2   "sys": { "country": "JP", "sunrise": 1369769524,
3     "sunset": 1369821049 },
4   "weather": [ { "id": 804, "main": "clouds",
5     "description": "overcast clouds", "icon": "04n" } ],
6   "main": { "temp": 289.5, "humidity": 89, "pressure": 1013,
7     "temp_min": 287.04, "temp_max": 292.04 },
8   "wind": { "speed": 7.31, "deg": 187.002 },
9   "rain": { "3h": 0 },
10  "clouds": { "all": 92 },
11  "dt": 1369824698,
12  "id": 1851632,
13  "name": "Shuzenji",

```

```
14 "cod":200 }
```

5.1.1.3 Interface

O protótipo ViSIoT possui uma interface Web, desenvolvida com o auxílio de *java servlets*, HTML5 e javascript. Esta interface é composta por um Mapa Mundi, o qual permite aos usuários selecionar a região na qual desejam realizar a busca dos recursos, tal como mostrado pelas regiões mais escuras na Figura 13.a. No instante em que uma região é selecionada, um *pop-up* é expandido automaticamente e permite ao usuário selecionar o dispositivo em que deseja que os objetos daquela região sejam implantados, tal como representado pela Figura 13.b. Além disso, é possível limitar o número de objetos que serão implantados em cada dispositivo alvo.

5.1.1.4 Seletor

O módulo Seletor, assim como o módulo Core foi desenvolvido utilizando a linguagem Java e é utilizado para identificar quais os recursos são mais indicados para a região desejada. O algoritmo implementado neste protótipo foi o TOPSIS descrito na Seção 2.2.

5.1.1.5 Conversor

O módulo conversor é utilizado para converter o objeto manipulado pelo Core da aplicação em um objeto que poderá ser implantado no sistema dos dispositivos alvos, e assim possibilitar que acessem as informações dos recursos. Esta ferramenta assume que os dispositivos alvos executem o *middleware* GSN. O GSN suporta a implantação, integração e a descoberta de uma vasta quantidade de tecnologias de redes de recursos por meio de sensores virtuais. Os recursos virtuais utilizados pelo GSN são descritos por meio da linguagem XML, e permitem a integração de fluxo de dados de recursos locais e remotos por meio da linguagem *Structured Query Language* (SQL) (ABERER; HAUSWIRTH, 2006).

Arquivo 2 – Exemplo de descrição de um sensor virtual no *middleware* GSN

```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <virtual-sensor name="Tartu588335" priority="10">
3   <processing-class>
4     <class-name>gsn.vsensor.BridgeVirtualSensor</class-name>
5     <init-params/>
6     <output-structure>
7       <field name="city" type="varchar(255)"/>
8       <field name="country" type="varchar(255)"/>
9       <field name="base" type="varchar(255)"/>
10      <field name="temp" type="double"/>
11      <field name="sea_level" type="double"/>
```

Figura 13 – Interface ViSIoT

ViSIoT Visual Search for Internet of Things

(a) Página Principal

ViSIoT Visual Search for Internet of Things

(b) Pop-up

Fonte: Nunes *et al.* (2016a).

```
12 <field name="pressure" type="double"/>
13 <field name="humidity" type="double"/>
14 </output-structure>
15 </processing-class>
16 <description />
17 <addressing>
```

```

18  <predicate key="geographical">Tartu 588335</predicate>
19  <predicate key="LATITUDE">26.72509</predicate>
20  <predicate key="LONGITUDE">58.380619</predicate>
21  </addressing>
22  <storage history-size="168h"/>
23  <streams>
24    <stream name="stream1">
25      <source alias="source1" sampling-rate="1"
26        storage-size="168h">
27        <address wrapper="openweathermap">
28          <predicate key="url">http://api.openweathermap.org/
29            data/2.5/weather?id=588335</predicate>
30          <predicate key="type">humidity</predicate>
31        </address>
32        <query>Select city, country, base, sea_level, temp,
33          humidity, pressure from wrapper</query>
34      </source>
35      <query>Select * from source1</query>
36    </stream>
37  </streams>
38 </virtual-sensor>

```

O Código-fonte 2 descreve um arquivo XML utilizado para representar um recurso por meio de um sensor virtual no *middleware* GSN. Este arquivo contém as informações referentes ao recurso, como acessá-las e o seu período de atualização.

5.1.1.6 Ambiente alvo

O ambiente alvo é composto por um conjunto de dispositivos que serão responsáveis por coletar as informações dos recursos escolhidos. Como o ambiente alvo é uma entidade externa à ferramenta, uma aplicação cliente foi desenvolvida e é responsável por integrar o ambiente externo ao Core da ferramenta. Neste protótipo, é assumido que os dispositivos alvos executem o *middleware* GSN e tenham implementado o código necessário para interpretar os sensores-virtuais gerados pelo Core da ferramenta.

Desta maneira, espera-se que a aplicação cliente que está em execução no dispositivo alvo seja capaz de implantar os recursos destinados a ela, que são transferidos em um arquivo compactado por meio de uma requisição HTTP. Ao receber este arquivo, a aplicação deve ser capaz de descompactá-los no lugar especificado em seu arquivo de configuração para realizar sua implantação e reportar seu estado ao Core da ViSIoT.

Arquivo 3 – Arquivo de configuração da aplicação cliente

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <config>
3     <client>
4         <address>visiot.lasdpc.icmc.usp.br</address>
5         <port>50130</port>
6         <outputdir>/opt/sensores/</outputdir>
7         <name>node01</name>
8     </client>
9 </config>

```

O Código-fonte 3 descreve o arquivo utilizado para configurar a aplicação cliente, em que o endereço do Core da aplicação e sua porta são passados como parâmetros para possibilitar a aplicação cliente conectar ao seu servidor. Além disso, o diretório no qual deve ocorrer a implantação e o nome do dispositivo que está executando a aplicação são especificados neste arquivo.

5.1.2 Funcionamento

Na Figura 14 é representado o funcionamento da ferramenta. A sequência de etapas exigidas para seu funcionamento pode ser resumida em:

1. O ambiente é configurado pelo usuário, no qual é especificada a quantidade de objetos desejados em cada região e o local em que estes objetos serão implantados.
2. O Core da ferramenta faz uma requisição para o repositório no qual os objetos estão disponíveis, passando os parâmetros desejados.
3. Uma vez que os objetos do repositório são retornados ao Core da aplicação, eles são desserializados e transformados em objetos genéricos, os quais serão manipulados pelos outros módulos.
4. Em seguida, os objetos genéricos são classificados pelo módulo seletor e retornados de acordo com suas posições para o Core da aplicação.
5. Para cada região obtida, o Core irá submeter os N primeiros objetos para serem convertidos em um formato que poderá ser interpretado pela plataforma alvo.
6. Um arquivo compactado é criado para cada dispositivo que irá implantar os objetos. Neste arquivo estão todos os objetos que serão implantados naquele dispositivo. As URIs destes arquivos são enviadas para os dispositivos alvos.

Figura 14 – Diagrama de sequência descrevendo o funcionamento da ferramenta ViSIoT.

Fonte: Adaptada de Nunes *et al.* (2016a).

7. Os dispositivos alvos utilizarão esta URI para realizar uma requisição HTTP e obter o arquivo compactado.
8. Com os arquivos no dispositivo alvo, sua implantação é feita e os dispositivos passam a coletar as informações diretamente dos recursos.
9. Assim que confirmação da implantação é recebida, o core inicia o monitor. Este monitor ficará responsável por verificar os estados dos recursos e reportá-los ao Core. O Core deverá contornar qualquer problema ocasionado aos objetos evitando assim violações do contrato de nível de serviço.

5.2 Estudo exploratório das técnicas de seleção de recursos

A partir do desenvolvimento da ferramenta ViSIoT e da metodologia proposta na Seção 4, um estudo exploratório foi conduzido para comparar a qualidade de diferentes técnicas MCDA disponíveis na literatura para realizar a tomada de decisão em ambientes IoT. Neste sentido, dois experimentos foram conduzidos considerando os seguintes fatores: i) o número de soluções disponíveis, ii) os algoritmos MCDA, iii) o número de soluções desejadas e iv) o número de critérios. A descrição de cada experimento pode ser elencada da seguinte maneira:

- **Experimento 1:** computar as relações de dominância global de todas soluções disponíveis (209.555) por meio do algoritmo FNDS. De acordo com o número de critérios utilizados (2, 4 e 6) é possível estabelecer o número de soluções disponíveis em cada fronteira, bem como o número máximo de fronteiras das soluções. Os resultados deste experimento são descritos em termos do número de soluções existentes para cada fronteira da solução.
- **Experimento 2:** aplicar a metodologia deste trabalho para comparar a qualidade das soluções propostas pelos algoritmos TOPSIS, SAW e VIKOR em termos de capacidade e diversidade. A importância de cada critério foi alterada a cada replicação do experimento, com o intuito de analisar a influência dos pesos dos critérios na solução final. Os Apêndices A e B descrevem respectivamente o funcionamento dos algoritmos SAW e VIKOR.

Para a execução dos experimentos, o algoritmo FNDS, descrito na seção 2 foi modificado para receber o número de recursos n a serem selecionados e parar a sua execução quando os n primeiros recursos dominantes fossem encontrados. Além disso, o gerenciamento das relações de dominância computadas por cada recurso é feita por meio de um banco de dados, visto que a memória principal disponível no servidor é insuficiente para gerenciar as relações de dominância, a qual necessita de uma capacidade de armazenamento da ordem $O(N^2)$. No Apêndice C, são descritos os cenários em que é possível executar o algoritmo FNDS sem a utilização do banco de dados, bem como a comparação entre o tempo necessário utilizando-se a memória principal e o banco de dados.

Na Tabela 5 são descritos os fatores e níveis utilizados pelos experimentos. Cada experimento foi replicado cem vezes para reduzir os efeitos de componentes estocásticos, os quais podem adicionar ruídos aos experimentos. Na Tabela 6 são descritas as restrições climáticas utilizadas correspondentes a cada número de critério considerado.

Para analisar a qualidade da solução em termos de capacidade foi utilizada a métrica ONVGR descrita no Capítulo 2. A diversidade foi analisada de acordo com a métrica *Crowding Distance* (CD), para mostrar a distribuição e a disseminação das soluções propostas por meio

Tabela 5 – Fatores e níveis utilizados nos experimentos

Fator	Níveis Experimento 1	Níveis Experimento 2
Número de recursos disponíveis	209.555	209.555
Técnica MCDA	FNDS	TOPSIS, SAW, VIKOR
Número de soluções selecionadas	209555 (100%)	209 (0.1%), 2095 (1%) e 20955 (10%)
Número de critérios	2, 4 e 6	2, 4 e 6
Número de servidores	1	1

Tabela 6 – Correspondência entre o número de critérios dos experimentos e as restrições climáticas. A Tabela 4 contém a descrição de cada sigla das restrições climáticas.

Número de critérios	Restrições climáticas
2	$t \times h$
4	$t \times h \times sc \times w$
6	$t \times h \times sc \times w \times p \times d$

Fonte: Elaborada pelo autor.

da densidade das soluções em torno de uma solução específica. Para isso, a distância média de duas soluções adjacentes é calculada por meio dos valores das funções objetivo de cada critério. As soluções que apresentam as funções objetivo com os maiores e menores valores para um determinado critério recebem um valor de distância infinito, uma vez que não têm soluções adjacentes (DEB, 2001). A Equação 5.1 corresponde à fórmula usada para calcular os valores de CD da fronteira i para as soluções disponíveis, onde N corresponde ao número de funções objetivos e $f_{i(j+1)}(x)$ e $f_{i(j-1)}(x)$ os valores das funções objetivo das soluções adjacentes. Se todos os pares de soluções consecutivas tiverem a mesma distância, o valor do CD será zero e a solução proposta terá uma distribuição perfeita.

$$CD_{front_i} = \sum_{j=1}^N \frac{f_{i(j+1)}(x) - f_{i(j-1)}(x)}{\max(f_i(x)) - \min(f_i(x))} \quad (5.1)$$

5.2.1 Configuração do ambiente

Os experimentos realizados neste trabalho são baseados no ambiente descrito na Figura 15. Nesta figura, um cliente pode requisitar um conjunto de dados como serviço para a nuvem, a qual deverá retornar um conjunto de dados respeitando as restrições impostas pelo cliente. Para isso, ocorrerá uma busca em cada servidor para uma parcela dos dados requisitados, para posteriormente, realizar a sua junção e retornar a resposta final ao cliente.

Figura 15 – Organização do ambiente de experimentos formado por servidores com repositórios responsáveis por armazenar a descrição e o estado dos recursos disponíveis.

Fonte: Elaborada pelo autor.

Cada servidor que compõe o ambiente da nuvem possui um repositório composto por um conjunto único de descritores de recursos. O descritor de um recurso contém as propriedades de contexto da sua fonte original de dados. Qualquer alteração no seu estado como, por exemplo, mudança de posição ou valores de referência, também é refletida nas propriedades de contexto do seu descritor. Além disso, os servidores são capazes de executar técnicas MCDA localmente para realizar a seleção de recursos. O número de recursos atribuídos a cada servidor é dado pela relação entre o número total de recursos disponíveis e o número de servidores a serem utilizados em cada experimento ($209.555/n_{servidores}$).

A Tabela 7 descreve as configurações de hardware dos servidores disponíveis no Laboratório de Sistemas Distribuídos e Programação Concorrente (LaSDPC) que foram utilizados na execução dos experimentos.

5.2.2 Experimento 1

Neste experimento são computadas todas as relações de dominância existentes no conjunto de possíveis soluções por meio da execução do algoritmo FNDS. Os resultados obtidos

Tabela 7 – Configuração do hardware utilizado para a realização dos experimentos

Hardware/Software	Especificação
Processador	AMD Processor Vishera 4.2 Ghz
Memória	32 GB RAM DDR3 Corsair Vegeance
Disco Rígido	HD 2TB Seagate Sata III 7200RPM
Sistema Operacional	Linux Ubuntu Server 14.04 64 Bits LTS
Java	JDK 1.7
Banco de dados	MongoDB 3.0

Fonte: Elaborada pelo autor.

neste experimento permitem caracterizar o comportamento esperado das soluções à medida que os critérios são alterados.

A Figura 16 contém a representação do gráfico em linha do número de soluções disponíveis por fronteira do conjunto, considerando dois (Figura 16.a), quatro (Figura 16.b) e seis (Figura 16.c) critérios. Na Figura 16.a, o número de soluções disponíveis em cada fronteira tende a aumentar proporcionalmente ao valor de seu índice. Esse comportamento ocorre devido ao baixo número de conflitos existentes entre dois critérios, e assim possuindo um número maior de soluções dominadas. As soluções que utilizam dois critérios possuem um número de fronteiras elevado com um baixo número de soluções em cada uma delas.

Diferentemente da Figura 16.a, na Figura 16.b o número de soluções disponíveis nas primeiras fronteiras é maior do que o apresentado no cenário anterior, e diminui à medida que o valor do índice da fronteira aumenta. Essa condição ocorre pois o aumento do número de critérios implica também no aumento das soluções conflitantes, e por isso, diminui o número de soluções dominadas. Nesse sentido, um conjunto maior de soluções dominantes é proposto em cada fronteira, e assim reduzindo o número total de fronteiras.

Na Figura 16.c é observado um comportamento análogo ao da Figura 16.b, em que o número de soluções disponíveis nas primeiras fronteiras é ainda maior do que o apresentado nos outros cenários pois o número de conflitos entre as soluções é maior. Dessa maneira, existe um número elevado de soluções dominantes em cada fronteira e por isso a quantidade de fronteiras da solução diminuiu.

5.2.3 Experimento 2

Este experimento apresenta um estudo de caso para a aplicação da metodologia desenvolvida para comparar as soluções propostas pelas técnicas TOPSIS, SAW e VIKOR. Os resultados descritos nesta seção permitem analisar e relacionar as soluções propostas pelos algoritmos em termos de qualidade, considerando diferentes critérios e número de recursos a serem selecionados. Além disso, também é possível analisar como os pesos dos critérios podem influenciar a qualidade de uma solução.

Figura 16 – Número de soluções disponíveis em cada fronteira.

Fonte: Elaborada pelo autor.

5.2.3.1 Resultados da qualidade em termos de capacidade

Na Figura 17 é representado o gráfico da métrica ONVGR para selecionar 209 (Figura 17.a) e 2095 (Figura 17.b) e 20955 (Figura 17.c) soluções, considerando dois critérios.

Figura 17 – Gráfico em linha com a média ONVGR e o seu desvio padrão para dois critérios.

Fonte: Elaborada pelo autor.

Na Figura 17, é possível observar que, no geral, o valor da média do ONVGR tende a ser alto para as primeiras fronteiras e diminui à medida que o índice das fronteiras aumenta.

Além disso, as médias aritméticas de cada algoritmo estão praticamente sobrepostas em todos os cenários. Na Figura 17.a, a média do valor ONVGR de todos os algoritmos está basicamente sobreposta em todas as fronteiras, com exceção da fronteira com índice dois, na qual o TOPSIS apresentou um valor ligeiramente maior. As fronteiras que possuem maiores valores ONVGR tendem a mostrar um desvio padrão mais elevado.

Na Figura 17.b, o número de fronteiras aumenta em relação à Figura 17.a, pois mais soluções são selecionadas, aumentando o valor ONVGR médio e seu desvio padrão. Diferentemente da Figura 17.a, todas as fronteiras apresentam um valor de desvio padrão significativo, gerado pelos pesos diferentes atribuídos a cada critério. Além disso, para as primeiras fronteiras mais significativas, o algoritmo TOPSIS forneceu um valor ONVGR médio equivalente ou superior aos algoritmos SAW e VIKOR.

Na Figura 17.c é observado um comportamento semelhante ao da Figura 17.b, em que o número de fronteiras, o valor ONVGR médio e o desvio padrão aumentam proporcionalmente ao número de soluções selecionadas. Nesse cenário, o valor ONVGR médio é um para algumas fronteiras significativas. O desvio padrão mostra que, dependendo do valor de peso atribuído a um critério, é possível encontrar todas as soluções ótimas para um número maior de fronteiras. Além disso, o algoritmo TOPSIS apresentou um valor ONVGR médio igual ou superior aos algoritmos SAW e VIKOR.

No geral, na Figura 17, observamos que, à medida que mais soluções são selecionadas, maior é o valor ONVGR, a variação e o número de fronteiras. Na Figura 16.a para a maioria das fronteiras, há apenas um pequeno número de soluções ótimas devido ao baixo nível de conflitos entre os critérios. Assim, à medida que mais soluções são selecionadas, mais fronteiras são necessárias para atender o número desejado de soluções, aumentando a probabilidade de escolher uma solução de uma fronteira significativa e consequentemente seu valor ONVGR.

Em relação aos algoritmos MCDA e o valor dos pesos atribuídos aos critérios, nota-se que o desvio padrão e as médias se sobrepõem apresentando soluções estatisticamente equivalentes, embora o algoritmo TOPSIS tenha em média uma solução ligeiramente superior. É importante destacar para Figuras 17.b e c, que na suas últimas fronteiras o valor ONVGR médio está mais próxima de zero e tem um desvio padrão mais significativo, indicando um baixo número de soluções ou soluções propostas apenas quando alguns conjuntos de pesos específicos são atribuídos aos critérios.

Na Figura 18 é representado o gráfico da métrica ONVGR médio para selecionar 209 (Figura 18.a), 2095 (Figura 18.b) e 20955 (Figura 18.c) soluções, considerando quatro critérios. Na Figura 18.a é observado um baixo valor da média ONVGR para todos os algoritmos, devido ao maior número de conflitos existentes entre os critérios, os quais diminuem o número de relações de dominância existentes entre as soluções. Nas fronteiras mais significativas, o algoritmo TOPSIS possui a melhor solução, uma vez que possui os maiores valores da média ONVGR e desvio padrão, enquanto o algoritmo VIKOR apresenta a menor média ONVGR e desvio padrão.

Na Figura 18.b, os algoritmos SAW e TOPSIS têm praticamente a mesma média ONVGR e desvio padrão para as fronteiras mais significativas, enquanto o algoritmo VIKOR oferece uma solução com uma média ONVGR e valor de desvio padrão ligeiramente menor do que os outros algoritmos. Na Figura 18.c, todos os algoritmos apresentam basicamente o mesmo valor ONVGR e desvio padrão nas fronteiras mais significativas.

No geral, na Figura 18, percebe-se um comportamento semelhante à Figura 17, em que o número de soluções selecionadas influencia o valor ONVGR, seu desvio padrão e o número de fronteiras. No entanto, o valor ONVGR é inferior ao cenário anterior porque, conforme mostrado na Figura 16.b, quando quatro critérios são utilizados ocorrem mais conflitos entre os critérios, diminuindo o número de relações de dominância e aumentando o número de soluções disponíveis em cada fronteira.

O valor do peso atribuído a cada critério afeta o número de soluções propostas pelos algoritmos MCDA, porém o desvio padrão não apresenta tanta variação quanto na Figura 17, pois há mais soluções disponíveis em cada fronteira. Embora exista diferença na média ONVGR entre os algoritmos neste cenário, todos eles apresentam soluções equivalentes, pois os valores de ONVGR e o desvio padrão se sobrepõem nas fronteiras mais significativas. Além disso, para todos os algoritmos, as últimas fronteiras que apresentam a média ONVGR próxima a zero, indicam um baixo número de soluções ou que estas soluções são propostas apenas quando alguns pesos de critérios são utilizados.

Figura 18 – Gráfico em linha com a média ONVGR e o seu desvio padrão para quatro critérios.

Fonte: Elaborada pelo autor.

Na Figura 19 é representado o gráfico da métrica ONVGR médio para selecionar 209 (Figura 19.a) e 2095 (Figura 19.b) e 20955 (Figura 19.c) soluções, considerando seis critérios. Na Figura 19.a, nota-se o valor da média ONVGR um pouco acima do zero, com desvio padrão e número de fronteiras baixo, em que algoritmo TOPSIS utilizou menos fronteiras do que o algoritmo SAW e VIKOR.

Semelhante à Figura 18.b e c, a média ONVGR nas Figuras 19.b e c, seu desvio padrão e o número de fronteiras aumentam proporcionalmente ao número de soluções desejadas, mas em uma proporção menor. Os algoritmos TOPSIS e SAW apresentam a mesma média ONVGR

para as fronteiras mais significativas, mas ao observar o cenário como um todo, o algoritmo TOPSIS utiliza menos fronteiras. O algoritmo VIKOR por sua vez, mostrou a média ONVGR e desvio padrão ligeiramente menores do que o algoritmo SAW e TOPSIS nas fronteiras mais significativas.

Figura 19 – Gráfico em linha com a média ONVGR e o seu desvio padrão para seis critérios.

Fonte: Elaborada pelo autor.

Na Figura 19, a média ONVGR diminui ainda mais do que nas Figuras 17 e 18 devido ao alto nível de conflitos entre seis critérios, que permitem ter um grande número de possíveis soluções em cada fronteira como mostrado na Figura 16.c. Assim, como há mais soluções disponíveis em cada fronteira do que as soluções a serem selecionadas, o valor ONVGR tem um valor baixo e é menos influenciado pelos valores dos pesos atribuídos a cada critério.

As soluções propostas pelos algoritmos MCDA são estatisticamente equivalentes, pois a média ONVGR de todos os algoritmos sobrepõem os desvios padrão uns dos outros. Além disso, para todos os algoritmos, a média ONVGR mais próxima de zero na última fronteira sugere um baixo número de soluções ou soluções que são propostas apenas quando alguns pesos são atribuídos aos critérios.

5.2.3.2 Resultados da qualidade em termos de diversidade

Na Figura 20 é representado o gráfico do valor da média aritmética da CD para selecionar 209 (Figura 20.a), 2095 (Figura 20.b) e 20955 (Figura 20.c) soluções, considerando dois critérios. Na Figura 20.a, nota-se que para todos os algoritmos não há desvio padrão, e o valor da média CD é zero, indicando há uma alta densidade das soluções. O peso dos critérios não influenciou a diversidade das soluções propostas devido ao baixo número de conflitos e soluções selecionadas.

Na Figura 20.b é observado um ligeiro aumento na média CD nas fronteiras significativas com relação à Figura 20.a, sugerindo soluções com um nível de densidade mais baixo. A média CD apresenta um pequeno desvio padrão em seu valor devido aos diferentes pesos atribuídos a seus critérios. O aumento do valor da média CD e de seu desvio padrão ocorrem porque mais soluções são selecionadas aumentando a probabilidade de que soluções com diferentes valores

Figura 20 – Gráfico em linha representando o *Crowding Distance* médio e o seu desvio padrão para dois critérios.

Fonte: Elaborada pelo autor.

para as funções objetivo sejam escolhidas. Para as fronteiras em que o CD médio é zero indicam que as soluções possuem uma alta densidade.

Na Figura 20.c, basicamente todas as fronteiras apresentam um valor de CD médio e variação maior ou igual à Figura 20.b, mostrando que as soluções propostas apresentam menor densidade do que o cenário anterior. A média CD e a sua variação são afetadas pelo número de soluções selecionadas. A maior parte dos valores médios de CD e os maiores valores de desvio padrão ocorrem nas fronteiras intermediárias, as quais concentram a maioria das soluções disponíveis.

Na Figura 20 os algoritmos SAW, TOPSIS e VIKOR apresentaram soluções equivalentes, uma vez que os valores de suas médias CD e seus desvios padrão estão sobrepostos. Em geral, na Figura 20, o valor do CD tende a uma distribuição perfeita, indicando uma alta densidade para a maioria das fronteiras devido ao número elevado de soluções redundantes.

Assim, para a Figura 20.a, na qual apenas 209 recursos são selecionados, uma distribuição perfeita é alcançada devido ao baixo número de soluções e a um pequeno número de soluções diferentes por fronteira, indicando que soluções que possuem o mesmo valor para as funções objetivo são propostas em todas as fronteiras. À medida que os recursos 2095 e 20955 são selecionados, a probabilidade de obter uma solução que não é redundante aumenta e consequentemente o valor da média CD.

Na Figura 21 é representado o gráfico do valor da média aritmética da CD para selecionar 209 (Figura 21.a), 2095 (Figura 21.b) e 20955 (Figura 21.c) soluções, considerando quatro critérios. Na Figura 21.a, para os algoritmos SAW e TOPSIS, uma média CD mais próxima de zero é observada nas primeiras fronteiras, mostrando que as soluções propostas nessas fronteiras têm uma densidade alta. Para ambos os algoritmos na maioria das fronteiras, o desvio padrão é pequeno, mostrando que os pesos dos critérios não influenciou a densidade das soluções. Por outro lado, o algoritmo VIKOR possui a média CD e o desvio padrão maiores do que os algoritmos SAW e TOPSIS, indicando que o algoritmo VIKOR fornece soluções com maior

Figura 21 – Gráfico em linha representando o *Crowding Distance* médio e o seu desvio padrão para quatro critérios.

Fonte: Elaborada pelo autor.

diversidade para a maioria das fronteiras significativas. Além disso, o valor da média CD tende a aumentar nas fronteiras menos significativas para o algoritmo VIKOR.

Na figura 21.b, ocorre um comportamento análogo à Figura 21.a, em que os algoritmos SAW e TOPSIS apresentam a média CD ligeiramente superior a zero nas fronteiras mais significativas. O desvio padrão das soluções é menor que o apresentado nos cenários anteriores, indicando que a densidade de soluções tende a ser maior com mais soluções. Para as fronteiras menos significativas, a média CD indica uma distribuição perfeita com o valor do desvio padrão zero. O algoritmo VIKOR apresenta os valores da média CD acima dos valores do desvio padrão dos algoritmos SAW e TOPSIS praticamente em todas as fronteiras, fornecendo uma solução com densidade menor do que os outros algoritmos.

Na Figura 21.c, para uma parte das fronteiras mais significativas, a média CD de todos os algoritmos estão sobrepostas com o valor do desvio padrão próximo a zero, indicando soluções com alta densidade. Nas outras fronteiras, a média CD dos algoritmos SAW e TOPSIS se manteve sobreposta e tem um valor de desvio padrão pequeno, enquanto no algoritmo VIKOR a média CD e o desvio padrão aumentam proporcionalmente ao índice da fronteira.

Na Figura 18, nota-se um comportamento diferente em relação à Figura 17, embora existam soluções redundantes neste cenário. Para selecionar 209 soluções, ocorre uma alta variação na média CD para as primeiras fronteiras, pois os pesos dos critérios podem afetar o valor da função objetivo das soluções propostas e há um número menor de soluções redundantes. À medida que 2095 e 20955 são selecionados, essas variações apontam para uma distribuição perfeita. O algoritmo SAW e TOPSIS possuem soluções com maior densidade devido ao número redundante de soluções. Por outro lado, o algoritmo VIKOR apresentou um maior valor para média CD, indicando soluções com alto nível de diversidade independentemente do número de recursos selecionados.

Na Figura 22 é representado o gráfico do valor da média aritmética da CD para selecionar 209 (Figura 22.a), 2095 (Figura 22.b) e 20955 (Figura 22.c) soluções, considerando

Figura 22 – Gráfico em linha representando o *Crowding Distance* médio e o seu desvio padrão para seis critérios.

Fonte: Elaborada pelo autor.

seis critérios. Para a Figura 22.a, o algoritmo VIKOR apresenta a média CD mais alta, apresentando soluções com menor densidade dentre os algoritmos. O algoritmo TOPSIS possui a média CD ligeiramente superior ao algoritmo SAW nas fronteiras significativas, mas como a média CD do algoritmo TOPSIS sobrepõe o desvio padrão do algoritmo SAW, suas soluções são estatisticamente equivalentes.

Na Figura 22.b, nas fronteiras mais significativas, todos os algoritmos possuem soluções com a média CD e desvio padrão semelhantes. O algoritmo VIKOR apresenta a média CD mais alta para a maioria das fronteiras significativas. Nas outras fronteiras, o algoritmo SAW e TOPSIS possuem o valor médio do CD próximo ou igual a zero com baixo desvio padrão. A solução do algoritmo VIKOR por sua vez, mantém-se com a média ONVGR e desvio padrão maiores do que os outros algoritmos.

Na Figura 22.c, nas fronteiras mais significativas, todos os algoritmos apresentam a média CD próxima a zero e desvio padrão baixo, indicando que as soluções tendem possuir alta densidade nessas fronteiras. Nas outras fronteiras, o algoritmo VIKOR possui a média CD e o desvio padrão mais altos que o algoritmo SAW e TOPSIS. Nas últimas fronteiras, todos os algoritmos tendem a uma distribuição perfeita sem nenhum desvio padrão.

Na Figura 19 observamos um comportamento semelhante à Figura 18. Quando 209 soluções são selecionadas, há uma maior variação na média CD para as primeiras fronteiras devido aos diferentes pesos atribuídos a cada critério. À medida que 2095 e 20955 são selecionados, essas variações tendem a diminuir e a solução se aproximar da distribuição perfeita. Para as últimas fronteiras, os algoritmos SAW e TOPSIS apresentaram uma solução perfeita com um baixo desvio padrão. Por outro lado, o algoritmo VIKOR tem a média CD e o desvio padrão maiores para a maioria das fronteiras em todos os cenários, mostrando soluções com menor densidade.

5.2.3.3 *Análise dos Resultados*

Nesta seção, foram comparadas as soluções obtidas por diferentes algoritmos MCDA aplicados para descoberta de recursos em ambientes IoT, os quais devem considerar os efeitos dos pesos atribuídos a cada critério. Em primeiro lugar, analisando os resultados descritos na Figura 16, que representam o número de soluções ótimas disponíveis em cada fronteira de Pareto para todas soluções disponíveis, é observado o impacto do número de critérios na solução proposta. O número de critérios determina o nível de conflitos entre eles. Em outras palavras, à medida que forem impostos mais critérios, haverá mais soluções disponíveis em cada fronteira, tornando mais difícil estabelecer a melhor solução para um conjunto específico de funções objetivo. Do ponto de vista prático, isso significa que a necessidade de tomada de decisão deve considerar cuidadosamente suas prioridades. Se um grande número de prioridades forem impostas, o resultado obtido pode ser menos útil, pois devido à sua natureza conflitante, haverá um número elevado de soluções factíveis para o problema.

Este resultado é mais fácil de entender observando os resultados obtidos pela métrica de capacidade, na qual a proporção do número de soluções ótimas diminuiu à medida que mais critérios são utilizados. Os pesos atribuídos aos critérios têm um impacto maior em uma solução quando esta utiliza poucos critérios, pois há um número baixo de soluções disponíveis em cada fronteira.

O valor ONVGR é proporcional ao número de soluções selecionadas, pois quanto mais soluções são obtidas maiores são as chances de um algoritmo MCDA propor uma solução ótima. Em todos os cenários, os algoritmos MCDA apresentaram soluções equivalentes em termos de capacidade, uma vez que para todos os algoritmos o valor médio da ONVGR obtido para cada fronteira estavam sobrepostos a média de outros algoritmos ou ao seu desvio padrão.

Por outro lado, considerando o valor médio da CD, que mostra a distância entre soluções adjacentes em uma fronteira, observamos uma diferença considerável nos algoritmos. Os algoritmos SAW e TOPSIS apresentam praticamente a mesma média e variação CD, indicando soluções com mais densidade do que a solução do algoritmo VIKOR. Assim, de acordo com os pesos dos critérios, o algoritmo VIKOR pode propor soluções com densidade mais baixa que os algoritmos SAW e TOPSIS, ou seja, soluções com maior nível de diversidade.

Além disso, o valor médio da CD é afetado pelo número de critérios que impactam diretamente a distribuição das soluções. É importante ressaltar que uma distribuição perfeita ocorre principalmente por dois motivos: i) há um alto número de soluções redundantes com um baixo nível de conflito entre os critérios, aumentando as probabilidades de obter a mesma solução, ou ii) um grande número de soluções são selecionadas e estão próximas umas das outras, aumentando sua densidade, e assim levando o valor da média CD a zero. Nesse sentido, o número de soluções selecionadas também afeta o valor médio do CD, aumentando seu valor nas primeiras fronteiras quando dois critérios são utilizado e diminuindo à medida que mais critérios

são impostos.

Considerando o estudo de caso apresentado nesta seção, utilizando um conjunto de dados reais e os efeitos do aquecimento global, que pode afetar a tolerância de uma solução para as mudanças climáticas, as seguintes lições foram aprendidas:

- À medida que mais critérios ambientais precisam ser considerados, torna-se mais difícil estabelecer quais são as soluções ótimas, pois há mais conflitos entre os critérios. Em outras palavras, é difícil determinar um local que apresente as condições climáticas ideais para iniciar a cultura do milho.
- Como mostrado pela métrica ONVGR, a atribuição de pesos para os critérios a serem analisados não garante o melhor conjunto de soluções. Assim, considerando as soluções propostas pelos algoritmos MCDA, não há diferença significativa entre eles quanto à métrica de capacidade.
- A métrica de diversidade mostra que o algoritmo MCDA pode ser determinante para estabelecer um nível de diversidade de soluções. Os algoritmos SAW e TOPSIS propõem soluções com um nível de densidade maior do que o algoritmo VIKOR, o que significa que eles apresentam soluções que são menos diversificadas que as soluções propostas pelo algoritmo VIKOR.

Considerando o contexto do aquecimento global, os algoritmos podem ter influência crítica sobre o sucesso da cultura, pois apontam soluções com diferentes níveis de densidade. As soluções que apresentam alto nível de densidade e baixa diversidade podem ser inteiramente afetadas pela mudança de um único valor de condição climática, enquanto as soluções com baixo nível de densidade e alto nível de diversidade podem apresentar alternativas mais resistentes às condições climáticas. Por outro lado, o alto nível de diversidade pode afetar a produção final, já que as espécies de milho evoluem de maneira diferente de acordo com as mudanças nas condições climáticas.

5.3 Considerações Finais

Neste capítulo foi descrito um estudo de caso para a aplicação das técnicas MCDA. A partir desta descrição foi realizado um estudo exploratório visando captar recursos das infraestruturas disponíveis para a seleção de recursos, no qual se desenvolveu uma ferramenta denominada ViSIoT. Por meio da captação de recursos realizada por esta ferramenta e da metodologia proposta no Capítulo 4 outro estudo exploratório foi realizado visando analisar o comportamento das técnicas de seleção e dos dados disponíveis. Os resultados obtidos mostraram que na prática não há diferença significativa entre as técnicas TOPSIS, SAW e VIKOR em termos de capacidade. Porém em termos de diversidade a técnica VIKOR mostrou menor densidade do que as técnicas

TOPSIS, SAW. No próximo capítulo os algoritmos FES e ES serão avaliados com relação aos resultados obtidos pela técnica TOPSIS.

100004

AVALIAÇÃO EXPERIMENTAL DAS TÉCNICAS FES E ES

Neste capítulo são avaliados os algoritmos FES e ES, propostos no Capítulo 4, para realizar a seleção de recursos em ambientes. Para fins de comparação com as técnicas MCDA disponíveis na literatura, o algoritmo TOPSIS será utilizado como solução de referência destas técnicas enquanto o algoritmo FNDS será utilizado como referência para computar a solução ótima. Para a realização dos experimentos descritos neste capítulo são consideradas a mesma motivação e ambiente de execução dos experimentos descritos no Capítulo 5.

6.1 Configuração dos Experimentos

Com o intuito de avaliar as técnicas propostas nesta tese para a tomada de decisão multicritério em IoT, três experimentos foram conduzidos para validar a metodologia e analisar o comportamento dos algoritmos em diferentes cenários. Os experimentos realizados consideram cinco fatores: i) o número de recursos disponíveis, ii) as técnicas MCDA, iii) o total de recursos desejados, iv) a quantidade de critérios e v) o número de servidores utilizados para armazenar os recursos disponíveis. A descrição de cada experimento pode ser elencada da seguinte maneira:

1. **Experimento 1:** aplicar a metodologia deste trabalho para comparar as soluções propostas pelos algoritmos TOPSIS, FES, ES e FNDS em termos de qualidade e tempo de execução, sujeitos a variação do número critérios (2, 4 e 6) e do número de recursos selecionados (2.095 e 20.955) nas soluções. Os resultados obtidos neste experimento são descritos em termos da métrica ONVGR e do tempo de execução, a fim de determinar a qualidade e custo computacional da solução. O indicador de desempenho PQA e o tempo de execução são utilizados para relacionar a qualidade e o custo das soluções.
2. **Experimento 2:** aplicar a metodologia deste trabalho para comparar as soluções propostas

pelos algoritmos TOPSIS, FES, ES e FNDS em termos de qualidade e tempo de execução, sujeitos à variação do número critérios (2, 4 e 6) e do número de recursos selecionados (2.095 e 20.955) nas soluções em ambientes que os recursos são armazenados de forma distribuída entre os servidores (2, 4 e 8). Os resultados obtidos neste experimento são descritos em termos da métrica ONVGR e do tempo de execução para determinar a qualidade e custo computacional da solução, à medida que o número de servidores é alterado. O indicador de desempenho PQA e o tempo de execução são utilizados para relacionar a qualidade e o custo das soluções para as diferentes configurações de servidores.

3. **Experimento 3:** aplicar a metodologia deste trabalho para comparar as soluções propostas pelos algoritmos TOPSIS, FES, ES e FNDS em termos de qualidade e da relação custo-benefício, de acordo com a variação do número critérios (2, 4 e 6) e da ordem de grandeza do número de recursos selecionados (10, 100, 1.000, 10.000 e 100.000). Os resultados obtidos neste experimento são descritos em termos da métrica ONVGR, para representar a qualidade, e da relação entre o indicador de desempenho PQA e o tempo correspondentes ao custo-benefício.

É válido destacar que para todos os experimentos, o valor dois foi utilizado no parâmetro $\mathbb{SR}$ das técnicas FES e ES. Este valor foi definido com base no número máximo de recursos a serem selecionados (Experimento 3 - 100.000). O parâmetro $\mathbb{P}$ da técnica FES utilizou o valor de 50%. Na Tabela 8 são descritos os fatores e níveis utilizados em cada experimento. Cada experimento foi replicado trinta vezes para reduzir os efeitos de componentes estocásticos, os quais podem adicionar ruídos aos experimentos.

Tabela 8 – Descrição dos fatores e níveis dos experimentos.

Fator	Níveis Experimento 1	Níveis Experimento 2	Níveis Experimento 3
Número de recursos disponíveis	209.555	209.555	209.555
Técnica MCDA	TOPSIS, FES, ES, FNDS	TOPSIS, FES, ES, FNDS	TOPSIS, FES, ES, FNDS
Número de soluções selecionadas	2095 (1%) e 20955 (10%)	2095 (1%) e 20955(10%)	10, 100, 1.000 10.000 e 100.000
Número de critérios	2, 4 e 6	2, 4 e 6	2, 4 e 6
Número de servidores	1	2, 4 e 8	1

6.2 Experimento 1

Este experimento demonstra um estudo de caso para a aplicação da metodologia desenvolvida nesta tese de doutorado. Os resultados descritos nesta seção permitem analisar e

relacionar as soluções propostas pelos algoritmos em termos de qualidade e de custo computacional considerando diferentes critérios e números de recursos a serem selecionados.

6.2.1 Qualidade

A Figura 23 contém a representação do gráfico em linha do valor da métrica ONVGR em cada fronteira, considerando dois critérios para a seleção de 2.095 e 20.955 recursos. O algoritmo FNDS possui a solução com melhor qualidade dentre os algoritmos analisados, uma vez que com exceção da última fronteira, o valor da métrica ONVGR é igual a um, indicando que todos os recursos disponíveis nas fronteiras foram encontrados. As fronteiras mais significativas são consideradas aquelas utilizadas pelo algoritmo FNDS para se propor a solução, pois a sua solução sempre terá o menor número possível de fronteiras.

O algoritmo TOPSIS gerou a solução com pior qualidade, visto que os valores da métrica ONVGR são menores ou iguais aos outros algoritmos para as fronteiras mais significativas. Os algoritmos ES e FES, ao utilizarem o algoritmo TOPSIS como filtro e o algoritmo FNDS para realizar a tomada de decisão multicritério, dispõem de soluções com a qualidade entre esses dois algoritmos. Por analisar todos os recursos após a aplicação do filtro, o algoritmo ES obteve valores para a métrica ONVGR maiores ou iguais ao algoritmo FES nas fronteiras mais significativas. É válido destacar que para a seleção de 20.955 recursos (Figura 23.b), o algoritmo ES propôs uma solução equivalente ao algoritmo FNDS, obtendo os mesmos valores da métrica ONVGR em todas as fronteiras.

De forma geral, nota-se que tanto a quantidade de fronteiras quanto o valor da métrica ONVGR das soluções são diretamente proporcionais ao número de recursos desejados. A busca por soluções com um número maior de recursos aumenta o valor da métrica ONVGR, pois a probabilidade de se obter um recurso de uma fronteira ótima é maior. Já a quantidade de fronteiras é influenciada pelo baixo número de recursos disponíveis em cada fronteira, e por isso, à medida que o número de recursos selecionados aumenta, mais fronteiras são utilizadas pela solução.

A Figura 24 contém a representação do gráfico em linha do valor da métrica ONVGR em cada fronteira, considerando quatro critérios para a seleção de 2.095 e 20.955 recursos. Como na Figura 23, nota-se que o valor da métrica ONVGR aumenta à medida que mais recursos são selecionados. Porém, os valores da métrica ONVGR e a quantidade de fronteiras são significativamente menores em ambos cenários, pois o aumento do número de critérios implica na ocorrência de um número maior de conflitos entre as soluções, dificultando o estabelecimento de uma única solução ótima. Dessa maneira, um conjunto de soluções equivalentes são propostos em cada fronteira, aumentando o número de soluções existentes em cada uma delas.

Independentemente do número de recursos selecionados, o algoritmo FNDS apresentou a solução com maior qualidade, na qual apenas na última fronteira o valor da métrica ONVGR é

Figura 23 – Qualidade da seleção de 2.095 (a) e 20.955 (b) recursos considerando dois critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A relação ideal deste cenário é representada pelo algoritmo FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras que uma solução deve apresentar auxiliando a comparação dos algoritmos TOPSIS, FES e ES com a solução ideal.

Fonte: Elaborada pelo autor.

diferente de um. O algoritmo TOPSIS obteve a pior solução, uma vez que possui os valores da métrica ONVGR menores ou iguais ao dos outros algoritmos para uma parte considerável das fronteiras mais significativas.

Ao se basearem nos algoritmos TOPSIS e FNDS, os algoritmos ES e FES apresentam soluções com a qualidade entre esses dois algoritmos. Diferentemente da Figura 23.a, em nenhuma fronteira significativa há sobreposição dos algoritmos ES e FES com o algoritmo FNDS para a seleção de 2.095 recursos. Para a seleção de 20.955 recursos, tais algoritmos sobrepõem o algoritmo FNDS exclusivamente na primeira fronteira. Conforme o esperado, o algoritmo ES, por analisar todas as recursos obtidos após a aplicação do algoritmo TOPSIS, apresentou valores ONVGR maiores ou iguais ao algoritmo FES nas fronteiras mais significativas.

A Figura 25 contém a representação do gráfico em linha do valor da métrica ONVGR em cada fronteira, considerando seis critérios para a seleção de 2.095 e 20.955 recursos. Analogamente à Figura 24, a utilização de um número maior de critérios reduziu ainda mais o valor da métrica ONVGR e a quantidade de fronteiras da solução, uma vez que o número de recursos disponíveis em cada fronteira é maior devido ao aumento de conflitos entre os critérios. À medida que mais soluções são selecionadas, tanto os valores da métrica ONVGR quanto a quantidade de fronteiras utilizadas pela solução aumentam.

De forma semelhante à Figura 24, os algoritmos FNDS e TOPSIS possuíam respectivamente os maiores e menores valores para a métrica ONVGR nas fronteiras mais significativas independentemente do número de recursos selecionados. O algoritmo ES apresentou os valores da métrica ONVGR maiores do que algoritmo FES nas fronteiras mais significativas. Embora

Figura 24 – Qualidade da seleção de 2.095 (a) e 20.955 (b) recursos considerando quatro critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A relação ideal deste cenário é representada pelo algoritmo FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras que uma solução deve apresentar auxiliando a comparação dos algoritmos TOPSIS, FES e ES com a solução ideal.

Fonte: Elaborada pelo autor.

ambos algoritmos tenham apresentado soluções com qualidade entre os algoritmos TOPSIS e FNDS, nenhum deles foi capaz de sobrepor a solução proposta pelo algoritmo FNDS.

6.2.2 Tempo

Na Figura 26 é representado o gráfico *boxplot* do tempo necessário para se obter uma solução considerando dois critérios para a seleção de 2.095 e 20.955 recursos. Nesta figura, é possível observar que o algoritmo FNDS apresentou o maior tempo dentre os algoritmos analisados pois computa as relações de dominância de todos os recursos disponíveis antes de definir a solução final. O algoritmo TOPSIS apresentou o menor tempo, já que normaliza os recursos disponíveis para pontuá-los e classificá-los em ordem decrescente, ou seja, utiliza uma abordagem monocritério para tomar uma decisão multicritério sem computar nenhuma relação de dominância. É válido destacar que tanto para a seleção de 2.095 recursos quanto de 20.955 recursos os algoritmos TOPSIS e FNDS apresentam estatisticamente o mesmo tempo, pois as computações realizadas por esses algoritmos não estão relacionadas com o número de recursos selecionados, mas com número total de recursos disponíveis.

Da mesma forma que o aspecto qualitativo, os algoritmos FES e ES apresentaram tempo de processamento com valores entre os algoritmos TOPSIS e FNDS, o qual aumenta conforme mais recursos são selecionados. Isso ocorre, pois o algoritmo TOPSIS é utilizado como um filtro para eliminar as soluções com menor probabilidade de serem ótimas, reduzindo assim o número de relações de dominância a serem computadas pelo algoritmo FNDS. Ao computar todas as relações de dominância entre as soluções obtidas após o filtro, o algoritmo ES demandou um

Figura 25 – Qualidade da seleção de 2.095 (a) e 20.955 (b) recursos considerando seis critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A relação ideal deste cenário é representada pelo algoritmo FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras que uma solução deve apresentar auxiliando a comparação dos algoritmos TOPSIS, FES e ES com a solução ideal.

Fonte: Elaborada pelo autor.

tempo maior de processamento do que o algoritmo FES.

Figura 26 – Tempo total em segundos necessário para a seleção de 2.095 (a) e 20.955 (b) recursos considerando dois critérios, em que quanto menor o tempo melhor o desempenho.

Fonte: Elaborada pelo autor.

Na Figura 27 é representado o gráfico *boxplot* do tempo necessário para se obter uma solução considerando quatro critérios para a seleção de 2.095 e 20.955 recursos. Nesta figura, nota-se um comportamento similar ao da Figura 26, no qual o algoritmo FNDS e TOPSIS possuem respectivamente os maiores e o menores tempo para propor as soluções com 2.095 e 20.955 recursos.

Os algoritmos ES e FES apresentaram um ligeiro aumento no seu tempo com relação à Figura 26, pois mais comparações devem ser realizadas para se estabelecer as relações de dominância entre as soluções. É válido ressaltar que embora o aumento do número de critérios implique que o algoritmo FNDS realize mais comparações, o seu tempo diminuiu em relação à Figura 26. Neste caso específico, o aumento do número de critérios diminuiu o número de recursos dominantes, reduzindo o número de operações de entrada e saída realizadas, e consequentemente diminuindo o tempo necessário para se propor a solução.

Figura 27 – Tempo total em segundos necessário para a seleção de 2.095 (a) e 20.955 (b) recursos considerando quatro critérios, em que quanto menor o tempo melhor o desempenho.

Fonte: Elaborada pelo autor.

Na Figura 28 é representado o gráfico *boxplot* do tempo necessário para se obter uma solução considerando seis critérios para a seleção de 2.095 e 20.955 recursos. Analogamente à Figura 27, os algoritmos FNDS e TOPSIS apresentaram respectivamente o maior e o menor valor de tempo dentre os algoritmos, independentemente do número de recursos selecionados. Novamente, o aumento do número de critérios colaborou para a diminuição do tempo de execução do algoritmo FNDS, uma vez que com menos soluções dominantes diminui-se o número de operações de entrada e saída. Os algoritmos ES e FES, tiveram um ligeiro aumento do seu tempo conforme o esperado, no qual o algoritmo ES apresentou o aumento mais significativo por computar um número maior de relações de dominância.

6.2.3 Análise dos Resultados

Na Figura 29 é representada por meio do gráfico de pontos a relação entre a qualidade absoluta obtida por meio do indicador de desempenho PQA e o tempo necessário para se estabelecer soluções compostas por 2.095 e 20.955 recursos, considerando dois (Figura 29.a), quatro (Figura 29.b) e seis critérios (Figura 29.c).

De forma geral, é possível observar nesta figura que o indicador de desempenho PQA e o tempo são inversamente proporcionais, ou seja, quanto maior é o tempo para se estabelecer uma

Figura 28 – Tempo total em segundos necessário para a seleção de 2.095 (a) e 20.955 (b) recursos considerando seis critérios, em que quanto menor o tempo melhor o desempenho.

Fonte: Elaborada pelo autor.

solução menor é o valor do indicador de desempenho PQA. O número de critérios utilizados influencia os valores do indicador de desempenho PQA, pois quanto maior o número de critérios maior é o valor do indicador de desempenho PQA, indicando que soluções ótimas não foram selecionadas.

O algoritmo FNDS possui o valor do indicador de desempenho PQA menor ou igual ao dos outros algoritmos enquanto o valor da métrica tempo é a maior em todos os cenários. Em outras palavras, este algoritmo possui a solução com a melhor qualidade, mas também com o maior custo computacional. Por outro lado, o algoritmo TOPSIS possui o menor tempo, porém com o valor do indicador de desempenho PQA igual ou maior do que os outros algoritmos em todos os cenários. É válido destacar que independentemente do número de recursos a serem selecionados ambos os algoritmos possuem tempos constantes, pois dependem apenas do número total de recursos disponíveis.

Por outro lado, os algoritmos ES e FES dependem também do número de recursos a serem selecionados, visto que este número determina as relações de dominância a serem computadas. Como os algoritmos ES e FES têm seus funcionamentos baseados nos algoritmos TOPSIS e FNDS, os valores do indicador de desempenho PQA e tempo estão entre estes algoritmos.

Independentemente do número de critérios utilizados, na prática os algoritmos ES, FES e TOPSIS necessitam do mesmo tempo para propor uma solução com 2.095 recursos, sendo que o algoritmo ES apresentou uma solução com qualidade ligeiramente superior aos outros algoritmos, pois possui o menor valor para o indicador de desempenho PQA em todos os cenários.

Para as soluções compostas por 20.955 recursos considerando apenas dois critérios, observa-se que não há diferença entre a qualidade dos algoritmos ES e FNDS, sendo que o primeiro possui o menor tempo para propor a solução. A medida que o número de critérios

Figura 29 – Custo-benefício da seleção de 2.095 e 20.955 recursos considerando dois (a), quatro (b) e seis (c) critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. As linhas pontilhadas indicam quais são os algoritmos que apresentam as melhores relações custo-benefício em cada cenário.

Fonte: Elaborada pelo autor.

utilizados para determinar as soluções com 20.955 recursos aumenta, a relação entre a qualidade e o tempo de computação torna-se mais significativa.

Neste experimento não foi possível determinar o algoritmo ótimo para realizar a seleção dos recursos, pois a qualidade obtida está diretamente relacionada com o tempo de computação. As relações de dominância destacadas pelas linhas pontilhadas são utilizadas para demonstrar a equivalência entre os algoritmos por meio das relações de dominância. Os algoritmos TOPSIS, FES, ES e FNDS são equivalentes em todos os cenários considerando-se a relação entre a qualidade obtida e o tempo necessário para se obter a solução, exceto quando dois critérios são utilizados para obter uma solução com 20.955 recursos, no qual o algoritmo ES possui a mesma qualidade do algoritmo FNDS porém necessitando de um tempo menor de computação.

6.3 Experimento 2

Este experimento complementa o estudo de caso do experimento 2, no qual os recursos disponíveis estão distribuídos de forma aleatória em servidores independentes. Os servidores fornecem o mesmo número de recursos para compor a solução final, a qual é composta pela junção dos recursos mais adequados disponíveis em cada um deles. Os resultados descritos nesta seção permitem analisar tanto de forma individual quanto de forma conjunta as soluções propostas pelos algoritmos em termos de qualidade e de custo computacional considerando diferentes critérios, número de recursos a serem selecionados e o número de servidores utilizados para armazenar os recursos.

6.3.1 Qualidade

As Figuras 30 e 31 contêm a representação do gráfico em linha do valor da métrica ONVGR em cada fronteira, considerando dois critérios para a seleção de 2.095 e 20.955 recursos distribuídos em dois, quatro e oito servidores. Como mostrado na Seção 6.2, nota-se que quanto maior o número de recursos a serem selecionados, maior é o valor da métrica ONVGR nas fronteiras mais significativas, uma vez que a possibilidade de se obter um recurso destas fronteiras é maior. A quantidade de fronteiras por sua vez é influenciada pelo baixo número de soluções disponíveis em cada fronteira, e por isso aumenta à medida que mais recursos são selecionados.

Diferentemente da Seção 6.2, o algoritmo FNDS, utilizado como referência para calcular as relações de dominância e a solução ótima, apresenta em algumas fronteiras os valores da métrica ONVGR menores do que um em todos os cenários das Figuras 30 e 31. Isso ocorre, pois, ao distribuir os recursos em mais servidores, apenas as relações locais de dominância são consideradas, reduzindo a qualidade da solução obtida. À medida que mais servidores são utilizados para distribuir os recursos, a qualidade da solução piora e mais fronteiras são utilizadas para propor a solução, pois as relações de dominância globais diminuem. Apesar de obter os maiores valores para a métrica ONVGR nas fronteiras mais significativas, a solução proposta pelo algoritmo FNDS pode apresentar um número maior de fronteiras do que os outros algoritmos, como representado nas Figuras 30.c e 31.c.

O algoritmo TOPSIS, obteve os menores valores para a métrica ONVGR nas fronteiras mais significativas, indicando assim a solução de pior qualidade dentre os algoritmos analisados em todos os cenários. Os algoritmos ES e FES, por utilizarem o algoritmo TOPSIS como filtro e o algoritmo FNDS para estabelecer as relações de dominância entre os recursos, dispõem de soluções com a qualidade entre esses dois algoritmos. Por computar todas as relações de dominância das soluções fornecidas após a aplicação do filtro, o algoritmo ES obteve nas fronteiras mais significativas valores maiores para a métrica ONVGR do que o algoritmo FES. É válido destacar que na Figura 31 o algoritmo ES propôs uma solução com qualidade equivalente à proposta pelo algoritmo FNDS em todos os cenários.

Figura 30 – Qualidade da seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.

Fonte: Elaborada pelo autor.

Figura 31 – Qualidade da seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.

Fonte: Elaborada pelo autor.

De forma geral, nota-se que à medida que mais servidores são utilizados para distribuir os recursos, a qualidade da solução proposta pelos algoritmos é reduzida, pois apenas os aspectos locais dos recursos são considerados. Assim, o valor da métrica ONVGR diminui nas fronteiras mais significativas e a quantidade de fronteiras das soluções aumenta.

As Figuras 32 e 33 contêm a representação do gráfico em linha do valor da métrica ONVGR em cada fronteira, considerando quatro critérios para a seleção de 2.095 e 20.955

recursos distribuídos em dois, quatro e oito servidores. Conforme explicado na Seção 6.2 os valores da métrica ONVGR e a quantidade de fronteiras são consideravelmente menores do que os valores obtidos nas Figuras 30 e 31, pois com o aumento do número de critérios diminui-se o número de relações de dominância. Como nas Figuras 30 e 31, há o aumento do valor da métrica ONVGR nas fronteiras mais significativas e do número total de fronteiras da solução à medida que mais recursos são selecionados.

Figura 32 – Qualidade da seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.

Fonte: Elaborada pelo autor.

Figura 33 – Qualidade da seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.

Fonte: Elaborada pelo autor.

Analogamente às Figuras 30 e 31, o algoritmo FNDS apresenta valores da métrica ONVGR menores do que um em todos os cenários das Figuras 32 e 33, uma vez que considera apenas as relações locais de dominância para propor as suas soluções. Com a distribuição dos recursos em um número maior de servidores, nota-se que há também a diminuição do valor da métrica ONVGR e o aumento da quantidade de fronteiras da solução, uma vez que as relações globais de dominância não são consideradas.

Em todos os cenários destas figuras, o algoritmo TOPSIS possui os menores valores da métrica ONVGR nas fronteiras mais significativas e por isso a menor qualidade entre os algoritmos. Para a seleção de 2.095 recursos (Figura 32), os algoritmos ES e FES apresentaram valores ONVGR ligeiramente superiores ao algoritmo TOPSIS e bem menores do que o determinado pelo algoritmo FNDS nas fronteiras mais significativas, independentemente do número de servidores utilizados. Por outro lado, quando 20.955 recursos (Figura 33) são selecionados os valores da métrica ONVGR aumentam para todos os algoritmos, diminuindo a diferença da qualidade com o algoritmo FNDS.

Apesar da diminuição da diferença de qualidade dos algoritmos com relação ao algoritmo FNDS, nenhum algoritmo possui os valores da métrica ONVGR equivalentes aos do algoritmo FNDS. Além disso, à medida que mais servidores são utilizados, a diferença da qualidade obtida entre os algoritmos ES, FES e TOPSIS tende a aumentar, sendo que o algoritmo ES possui a qualidade mais próxima ao do algoritmo FNDS. Como nas Figuras 30.c e 31.c, embora o algoritmo FNDS obtenha os maiores valores para a métrica ONVGR nas fronteiras mais significativas ele utiliza um número igual ou maior de fronteiras quando comparado aos outros algoritmos para selecionar os recursos quando estão distribuídos em oito servidores (Figuras 32.c e 33.c).

As Figuras 34 e 35 contêm a representação do gráfico em linha do valor da métrica ONVGR em cada fronteira, considerando seis critérios para a seleção de 2.095 e 20.955 recursos distribuídos em dois, quatro e oito servidores. Semelhantemente as Figuras 32 e 33, o maior número de critérios utilizados para realizar a seleção dos recursos diminuiu a quantidade de fronteiras utilizados pelas soluções, devido à utilização de mais critérios para realizar a seleção de recursos, o que reduz o número de recursos dominantes e aumenta o número de soluções disponíveis em cada fronteira. Como na seleção considerando dois e quatro critérios, o valor da métrica ONVGR nas fronteiras mais significativas aumenta proporcionalmente ao número de recursos selecionados.

Em todos os cenários das Figuras 34 e 35 o algoritmo FNDS apresenta os valores da métrica ONVGR menores do que um. Porém, diferentemente das Figuras 30, 31, 32 e 33, para a seleção de 2.095 recursos distribuídos entre 8 servidores (Figura 34.c) o algoritmo ES possui o valor da métrica ONVGR ligeiramente maior ao do algoritmo FNDS nas fronteiras mais significativas. Neste cenário, devido ao número elevado de critérios conflitantes, o filtro aplicado pelo algoritmo ES elimina soluções dominantes locais, aumentando assim o valor da

Figura 34 – Qualidade da seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.

Fonte: Elaborada pelo autor.

Figura 35 – Qualidade da seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, mensurada pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A melhor relação deste cenário é representada pelo algoritmo de FNDS, no qual minimiza-se a quantidade de fronteiras utilizadas e maximiza-se o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o menor número de fronteiras de acordo com a solução ótima obtida em um servidor utilizada como referência para a comparação da qualidade obtida pelos algoritmos TOPSIS, FES, ES e Pareto neste cenário.

Fonte: Elaborada pelo autor.

métrica ONVGR com relação ao algoritmo FNDS.

Novamente, o algoritmo TOPSIS obteve os menores valores da métrica ONVGR nas fronteiras mais significativas e por isso a menor qualidade entre os algoritmos em todos os cenários. Para a seleção de 2.095 recursos (Figura 34), o algoritmo FES apresenta valores ligeiramente superiores ao algoritmo TOPSIS nas fronteiras mais significativas enquanto o algoritmo ES apresenta os valores da métrica ONVGR consideravelmente maiores que as do

algoritmo FES. Para a seleção de 20.955 recursos (Figura 35), a diferença dos valores da métrica ONVGR entre os algoritmos FNDS, ES, FES e TOPSIS passa a ser mais significativa em todos os cenários.

Como nos cenários em que dois e quatro critérios são considerados, embora os algoritmos que utilizam relações de dominância para propor a solução final obtenham os maiores valores para a métrica ONVGR, nas fronteiras mais significativas eles utilizam um número igual ou maior de fronteiras quando comparado aos outros algoritmos quando os recursos estão distribuídos entre oito servidores (Figuras 34.c e 35.c).

6.3.2 Tempo

Nas Figuras 36 e 37 são representados os gráficos *boxplot* do tempo necessário para se obter uma solução considerando dois critérios para a seleção de 2.095 e 20.955 recursos distribuídos em dois, quatro e oito servidores. Em ambas as figuras, o algoritmo FNDS necessitou do maior tempo para propor a solução, pois computa as relações locais de dominância de todos os recursos disponíveis nos servidores. O algoritmo TOPSIS por sua vez obteve o menor tempo, já que normaliza os recursos disponíveis para classificá-los em ordem decrescente, ou seja, utiliza uma abordagem monocritério para tomar uma decisão multicritério sem computar nenhuma relação de dominância.

Os algoritmos ES e FES, ao utilizarem o algoritmo TOPSIS como filtro e o algoritmo FNDS para computar as relações de dominância, apresentaram tempos que estão entre estes dois algoritmos. Dessa maneira, por computar as relações de dominância apenas de parte da lista resultante após a fase de filtro, o algoritmo FES apresentou tempos médios menores do que o algoritmo ES em todos os cenários.

Figura 36 – Tempo total em segundos necessário para a seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.

Fonte: Elaborada pelo autor.

Figura 37 – Tempo total em segundos necessário para a seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando dois critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.

Fonte: Elaborada pelo autor.

Ao selecionar 2.095 recursos (Figura 36), nota-se que apesar de existir diferença estatística entre os tempos médios utilizados pelos algoritmos ao obter a solução, na prática os tempos dos algoritmos TOPSIS, FES e ES são equivalentes. Isso fica evidente à medida que os recursos são distribuídos em um número maior de servidores, sendo possível observar que as caixas do *boxplot* estão sobrepostas entre todos os algoritmos.

Para a seleção de 20.955 recursos, os tempos necessários para se obter uma solução utilizando os algoritmos TOPSIS e FNDS não são alterados, uma vez que estes são influenciados apenas pelo número total de recursos a serem processados. Por outro lado, os algoritmos ES e FES apresentaram um aumento significativo do tempo necessário para computarem a solução, pois ao selecionar mais recursos um número maior de relações de dominância serão computadas. Ao distribuir os recursos em mais servidores, observa-se que há uma tendência de redução exponencial do tempo necessário para computar as soluções dos algoritmos FNDS, ES e FES. A distribuição dos recursos em mais servidores diminui o número de relações locais de dominância a serem calculadas.

Nas Figuras 38 e 39 são representados os gráficos *boxplot* do tempo necessário para se obter uma solução considerando quatro critérios para a seleção de 2.095 e 20.955 recursos distribuídos em dois, quatro e oito servidores. Analogamente ao cenário com dois critérios (Figuras 36 e 37), os algoritmos TOPSIS e FNDS apresentaram o menor e o maior tempo médio para selecionar os recursos em ambos os cenários. Já os algoritmos ES e FES tiveram um ligeiro aumento no seu tempo médio decorrentes do maior número de comparações necessárias para determinar as relações de dominância entre as soluções.

Para a seleção de 2.095 recursos, observa-se que os tempos dos algoritmos TOPSIS, FES e ES não são estatisticamente equivalentes apenas quando os recursos estão distribuídos em dois servidores, embora na prática não haja diferença em nenhum cenário. Da mesma forma que na Figura 37, para a seleção de 20.955 recursos os algoritmos TOPSIS e FNDS não apresentam

Figura 38 – Tempo total em segundos necessário para a seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.

Fonte: Elaborada pelo autor.

Figura 39 – Tempo total em segundos necessário para a seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.

Fonte: Elaborada pelo autor.

variação em seu tempo, enquanto os algoritmos FES e ES possuem o tempo de computação entre estes dois algoritmos respectivamente. Os tempos dos algoritmos FNDS, ES e FES tendem a diminuir exponencialmente a medida que os recursos são divididos em quatro e oito servidores, uma vez que há a redução do número de relações de dominância a serem computadas em cada servidor.

Nas Figuras 40 e 41 são representados os gráficos *boxplot* do tempo necessário para se obter uma solução considerando seis critérios para a seleção de 2.095 e 20.955 recursos distribuídos em dois, quatro e oito servidores. Como no cenário com quatro critérios (Figuras 38 e 39). Os algoritmos ES e FES apresentaram um leve aumento no seu tempo médio, por requerer um número maior de computações para estabelecer as relações de dominância com seis critérios. Os algoritmos TOPSIS e FNDS obtiveram o menor e o maior tempo médio respectivamente para selecionar os recursos em ambos os cenários.

Para a seleção de 2.095 recursos (Figura 40) os tempos dos algoritmos TOPSIS, FES e ES são estatisticamente equivalentes quando os recursos estão distribuídos em quatro ou oito servidores. A diferença do tempo ocorre apenas quando os recursos são distribuídos em dois servidores, em que o algoritmo ES apresenta tempo maior que os algoritmos TOPSIS e FES graças ao maior número de computações necessárias para se obter relações de dominância. Da mesma forma que a Figura 39, para a seleção de 20.955 recursos (Figura 41) os algoritmos TOPSIS e FNDS não apresentam variação em seu tempo com relação a Figura 40, enquanto os algoritmos FES e ES possuem tempo de computação entre estes dois algoritmos respectivamente. Os tempos dos algoritmos FNDS, ES e FES tendem a diminuir exponencialmente a medida que os recursos são divididos em quatro e oito servidores, por reduzir o número de relações de dominância a serem computadas em cada servidor.

Figura 40 – Tempo total em segundos necessário para a seleção de 2.095 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.

Fonte: Elaborada pelo autor.

Figura 41 – Tempo total em segundos necessário para a seleção de 20.955 recursos distribuídos aleatoriamente em dois (a), quatro (b) e oito (c) servidores considerando seis critérios, em que quanto menor o tempo melhor o seu desempenho. O tempo utilizado para computar a solução ótima em um servidor é utilizado como referência para a comparação do desempenho.

Fonte: Elaborada pelo autor.

É válido destacar que devido a limitações de hardware, o algoritmo FNDS utiliza o banco

de dados para armazenar as relações de dominância de cada recurso enquanto os algoritmos ES e FES, devido ao número reduzido de recursos são capazes de armazenar tais relações em uma estrutura de dados na memória. Ao aumentar o número de critérios para realizar a seleção dos recursos aumenta-se também o número de comparações a serem realizadas e diminui-se o número total de relações de dominância de cada recurso. Dessa maneira, o tempo médio do algoritmo FNDS diminui à medida que o número de critérios aumenta, uma vez que menos acessos são realizados no banco de dados enquanto que o tempo médio dos algoritmos ES e FES aumenta uma vez que o custo de comparações é menor do que o de acesso a memória.

6.3.3 Análise dos Resultados

Nas Figuras 42, 43 e 44 são representadas por meio do gráfico de pontos a relação entre a perda da qualidade obtida pelo indicador de desempenho PQA e o tempo necessário para se estabelecer soluções compostas por 2.095 e 20.955 recursos distribuídos em dois, quatro e oito servidores considerando dois (Figura 42), quatro (Figura 43) e seis critérios (Figura 44).

Figura 42 – Custo-benefício da seleção de 2.095 e 20.955 recursos distribuídos aleatoriamente em em dois (a), quatro (b) e oito (c) servidores considerando dois critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. Os símbolos representados pela cor cinza correspondem ao custo-benefício do algoritmo Pareto executado em um único servidor para a seleção de 2.095 e 20.955 recursos, servindo como base de comparação com os algoritmos TOPSIS, FES, ES e Pareto.

Fonte: Elaborada pelo autor.

Analogamente à Figura 29, nota-se que com relação ao tempo de computação os algoritmos TOPSIS e FNDS obtiveram o melhor e o pior tempo de computação entre os algoritmos, sendo este tempo dependente apenas do número total de recursos disponíveis. Os algoritmos ES e FES por sua vez, alcançaram tempos de computação entre os algoritmos TOPSIS e FNDS, sendo influenciados majoritariamente pelo número de recursos a serem selecionados. Utilizando-se o tempo de referência obtido pelo algoritmo FNDS na Figura 29, à medida que mais servidores são utilizados para armazenar os recursos, o tempo de computação da solução diminui para todos os algoritmos, pois menos operações são realizadas.

Figura 43 – Custo-benefício da seleção de 2.095 e 20.955 recursos distribuídos aleatoriamente em em dois (a), quatro (b) e oito (c) servidores considerando quatro critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. Os símbolos representados pela cor cinza correspondem ao custo-benefício do algoritmo Pareto executado em um único servidor para a seleção de 2.095 e 20.955 recursos, servindo como base de comparação com os algoritmos TOPSIS, FES, ES e Pareto.

Fonte: Elaborada pelo autor.

Figura 44 – Custo-benefício da seleção de 2.095 e 20.955 recursos distribuídos aleatoriamente em em dois (a), quatro (b) e oito (c) servidores considerando seis critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução. A relação ideal deste cenário visa minimizar o valor do indicador de desempenho PQA e do tempo de execução. Os símbolos representados pela cor cinza correspondem ao custo-benefício do algoritmo Pareto executado em um único servidor para a seleção de 2.095 e 20.955 recursos, servindo como base de comparação com os algoritmos TOPSIS, FES, ES e Pareto.

Diferentemente da Figura 29, nota-se que o valor do indicador de desempenho PQA não necessariamente está relacionado com o tempo de computação, visto que a distribuição dos recursos disponíveis em mais servidores pode influenciar a solução proposta pelos algoritmos. Ainda, utilizando como referência o valor do indicador de desempenho PQA do algoritmo FNDS na Figura 29, é possível observar que a qualidade da solução diminui à medida que os recursos são distribuídos em mais servidores, visto que o valor do indicador de desempenho PQA aumenta.

Independentemente do número de critérios, nota-se que para a seleção de 2.095 recursos os algoritmos TOPSIS, FES e ES apresentaram na prática tempos iguais, em que o algoritmo ES obteve a melhor solução, visto que o valor do indicador de desempenho PQA é menor ou igual

aos algoritmos TOPSIS e FES. Conforme mais servidores são utilizados para distribuírem os recursos, o valor do indicador de desempenho PQA aumenta enquanto o tempo de computação torna-se o mesmo para estes algoritmos. Com relação ao algoritmo FNDS, nota-se que ele apresentou o menor valor para o indicador de desempenho PQA apenas em alguns cenários, visto que com a distribuição dos recursos em mais servidores apenas as relações locais de dominância são utilizadas, impactando a qualidade final da solução.

Na Figura 42, considerando apenas a utilização de dois critérios, nota-se que não há uma diferença significativa entre os valores do indicador de desempenho PQA dos algoritmos FNDS, ES e FES para a seleção de 20.955, quando os recursos disponíveis são distribuídos em dois e quatro servidores. Nestes cenários, a única diferença significativa destes algoritmos é o tempo necessário para computar uma solução. Ao distribuir os recursos disponíveis em oito servidores, observa-se que apenas o algoritmo ES é capaz de estabelecer uma solução próxima à proposta pelo algoritmo FNDS, porém utilizando um tempo menor do que tal algoritmo. Independentemente do número de servidores utilizados para armazenar os recursos disponíveis, o algoritmo TOPSIS apresentou o menor tempo e o maior valor para o indicador de desempenho PQA.

À medida que o número de critérios utilizados para selecionar 20.955 recursos aumenta, como representado nas Figuras 43 e 44, a diferença entre a qualidade dos algoritmos passa a ser mais significativa. O algoritmo FNDS possui o menor valor do indicador de desempenho PQA mas também o maior tempo. O algoritmo ES apresenta a métrica ligeiramente maior do que algoritmo FNDS porém com um tempo menor. Seguindo a mesma tendência, os algoritmos FES e TOPSIS apresentam valores do indicador de desempenho PQA maiores que o algoritmo ES e diretamente relacionadas com o tempo gasto para se obter a solução. É válido destacar que como na Figura 29, não é possível determinar um algoritmo ótimo para realizar a seleção de 2.095 e 20.955 recursos, mas sim estabelecer quais deles oferecem uma relação custo-benefício adequada, a qual é ilustrada pelas linhas pontilhadas.

6.4 Experimento 3

Este experimento visa analisar o custo-benefício dos algoritmos de acordo com a ordem de grandeza dos recursos a serem selecionados. Neste sentido, os resultados são analisados em termos da (1) relação entre a qualidade ofertada e a alteração do valor da importância das fronteiras mais significativas e (2) de acordo com o custo-benefício do processo de seleção.

6.4.1 Qualidade

A Figura 45 contém a representação do gráfico em linha do valor da métrica ONVGR em cada fronteira, considerando dois critérios para a seleção de 10, 100, 1.000, 10.000 e 100.000 recursos. Nesta figura, é possível observar que o valor da importância das fronteiras é distribuído

de acordo com o número de recursos selecionados. Esta diferença ocorre, pois o valor da importância de cada fronteira varia de acordo com o número total de fronteiras da solução ótima.

Figura 45 – Qualidade da seleção de 10, 100, 1.000, 10.000 recursos considerando dois critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o valor da importância da fronteira de acordo com o número de recursos a serem selecionados.

Fonte: Elaborada pelo autor.

Analogamente a Seção 6.2, a técnica FNDS obteve a solução ótima com os valores da métrica ONVGR iguais ou maiores ao dos outros algoritmos em todos os cenários. Para a seleção de 10 recursos, todas as técnicas apresentaram soluções com qualidade equivalente. À medida que mais soluções foram selecionadas, os valores da métrica ONVGR tendem a atingir o valor ótimo nas fronteiras de maior importância. Como discutido nas Seções 6.2 e 6.3, quando dois critérios são utilizados são necessárias várias fronteiras para se propor a solução, visto que devido ao baixo número de conflitos entre os critérios há poucos recursos disponíveis em cada fronteira.

As Figuras 46 e 47 contêm a representação do gráfico em linha do valor da métrica ONVGR para realizar a seleção de 10, 100, 1.000, 10.000 e 100.000 recursos considerando quatro e seis critérios, respectivamente. Conforme esperado, ao aumentar o número de critérios utilizados o número total de fronteiras da solução diminuiu, pois aumenta o número de conflitos entre os critérios e, por conseguinte o número de recursos disponíveis em cada fronteira. Dessa maneira, ao utilizar quatro e seis critérios a importância das fronteiras concentra-se em um número menor de fronteiras quando comparadas aos resultados descritos na Figura 45.

Para a seleção de 10 recursos (Figuras 46.a e 47.a), todas as técnicas foram capazes de

Figura 46 – Qualidade da seleção de 10, 100, 1.000, 10.000 recursos considerando quatro critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o valor da importância da fronteira de acordo com o número de recursos a serem selecionados.

Fonte: Elaborada pelo autor.

encontrar a solução ótima. Ainda houve a redução do valor da métrica ONVGR com relação à Figura 45.a, pois o número de recursos selecionados é inferior aos recursos disponíveis na primeira fronteira. O valor da métrica ONVGR nas fronteiras de maior importância e o número de fronteiras da solução aumentam proporcionalmente ao número de recursos selecionados. É válido destacar que a técnica FNDS utilizou apenas a fronteira ótima para propor as soluções com 10 e 100 recursos considerando quatro critérios (Figuras 46.a e b) e também as soluções compostas de 10, 100 e 1.000 recursos considerando seis critérios (Figuras 47.a, b e c).

6.4.2 Análise dos Resultados

Na Figura 48 é representada por meio do gráfico de pontos a relação entre a perda da qualidade obtida pelo indicador de desempenho PQA e o tempo necessário para se estabelecer soluções compostas por 2.095 e 20.955 recursos, considerando dois (Figura 48.a), quatro (Figura 48.b) e seis critérios (Figura 48.c).

Na Figura 48.a, nota-se que quanto menos recursos são selecionados maior é o valor do índice de desempenho PQA e menor é o tempo de seleção. O valor índice de desempenho PQA está diretamente relacionado ao número total de fronteiras da solução. Soluções com poucos recursos como para a seleção de 10 e 100 recursos (Figuras 45.a e b) utilizam poucas fronteiras,

Figura 47 – Qualidade da seleção de 10, 100, 1.000, 10.000 recursos considerando seis critérios mensurados pelo número total de fronteiras da solução e o valor da métrica ONVGR em cada fronteira. A linha sólida preta representa o valor da importância da fronteira de acordo com o número de recursos a serem selecionados.

Fonte: Elaborada pelo autor.

Figura 48 – Custo-benefício da seleção de 10, 100, 1.000, 10.000 e 100.000 recursos considerando dois, quatro e seis critérios mensurados pela relação do valor do indicador de desempenho PQA e o seu tempo de execução.

as quais possuem alta importância. Por não selecionar todos os recursos das fronteiras de maior importância, o valor do índice de desempenho PQA tende a ser elevado. É válido destacar que para a seleção de 10 recursos os valores dos índices de desempenho PQA das técnicas TOPSIS, FES e ES são equivalentes ao valor ótimo obtido por meio da técnica FNDS.

À medida que mais recursos são selecionados, mais fronteiras são necessárias para propor a solução e o valor da importância de cada fronteira tende a ser distribuído para um número maior de fronteiras conforme demonstrado na seleção de 1.000, 10.000 e 100.000 recursos

(Figuras 45.c, d e e). A seleção de mais recursos não implica necessariamente na redução do valor do índice de desempenho PQA, uma vez que se pode alterar o número e a importância das fronteiras. A redução do valor do índice de desempenho PQA depende diretamente do número de recursos escolhidos nas fronteiras de maior importância.

Na prática, com a seleção de mais recursos há o aumento do valor da métrica ONVGR na primeira fronteira, conforme destacado na Figura 45, indicando que um número maior de recursos ótimos foi encontrado. Dessa maneira, o aumento do valor do índice de desempenho PQA decorrente do número baixo de soluções que não são pertencentes à primeira fronteira, indicam que há um baixo número de soluções redundantes.

Com relação às técnicas de seleção, observa-se que para a seleção de 10 e 100 recursos não há uma diferença significativa entre as técnicas TOPSIS, FES e ES, as quais apresentam praticamente o mesmo tempo e o mesmo valor do índice de desempenho PQA. Por outro lado, para a seleção de 1.000, 10.000 e 100.000 recursos nota-se um comportamento análogo aos resultados da Seção 6.2, em que o valor do índice de desempenho PQA é inversamente proporcional ao tempo necessário para se obter a solução impossibilitando a escolha de uma única técnica. Com relação a técnica FNDS, independentemente do número de recursos selecionados, é utilizado o mesmo tempo, pois tal técnica computa todas as relações de dominância existentes entre os recursos antes de determinar a sua solução.

Na Figuras 48.b e 48.c observa-se que há o aumento do valor do índice de desempenho PQA, indicando que uma solução de pior qualidade é obtida com relação à Figura 48.a. Este aumento é decorrente da utilização do maior número de critérios, os quais aumentam o número de soluções disponíveis em cada fronteira e reduzem o número total de fronteiras da solução. Tal redução implica que o valor da importância das fronteiras estará concentrado em um número menor de fronteiras. Assim, com o aumento do número de recursos disponíveis em cada fronteira e por não selecionar todos os recursos disponíveis nas fronteiras de maior importância, o valor do índice de desempenho PQA aumenta.

As técnicas TOPSIS, FES e ES apresentaram um comportamento análogo ao da Figura 48.a, na qual o índice de desempenho PQA é igual ao ótimo e o tempo para se obter as soluções compostas por 10 recursos considerando quatro e seis critérios é o mesmo. Para a seleção de 100 recursos considerando quatro critérios não há diferença significativa entre o tempo e o valor do índice PQA obtido pelas técnicas TOPSIS, FES e ES. Porém ao considerar seis critérios para selecionar o mesmo número de recursos nota-se que apesar das técnicas analisadas apresentarem o mesmo tempo, a técnica ES obteve o valor do índice PQA igual ao ótimo computado pelo algoritmo FNDS. Para a seleção de 1.000 e 10.000 recursos, o valor do índice de desempenho PQA das técnicas TOPSIS, FES e ES é inversamente proporcional ao tempo necessário para se obter a solução, impossibilitando a determinação da melhor técnica para realizar a seleção de recursos. Na seleção de 100.000 recursos nota-se que todas as técnicas apresentaram aumento significativo do tempo e o valor do índice de desempenho PQA próximo

a zero.

A técnica FNDS apesar de apresentar os menores valores para o índice de desempenho PQA e os maiores tempos, obteve um comportamento completamente diferente da Figura 48.a. Independentemente do número de critérios utilizados, nota-se que as soluções cuja soluções ótimas se concentram apenas na primeira fronteira (Figuras 46.a e b e Figuras 47.a, b e c) apresentou tempo de seleção menor do que as soluções que utilizam mais de uma fronteira (Figuras 46.c e d e Figura 47.d). Isso é justificado pela adaptação realizada no algoritmo FNDS, descrita na Seção 6.1, a qual finaliza a execução do algoritmo assim que o número de recursos dominantes é obtido. Dessa maneira, se o número de recursos a serem selecionados for menor do que o número total de recursos disponíveis na primeira fronteira, o processo de seleção dispensará a computação adicional das relações de dominância e reduzirá o tempo total de execução.

6.5 Considerações Finais

Este capítulo descreveu os experimentos realizados para demonstrar a utilização da metodologia proposta nesta tese. O primeiro experimento permitiu computar as relações de dominância da base utilizada nos experimentos bem como compreender como a qualidade é afetada à medida que mais critérios são utilizados. A realização do primeiro serviu como base para analisar a execução das técnicas MCDA aplicadas na seleção dos recursos em ambientes IoT de forma centralizada (experimento 2) e distribuída (experimento 3). A aplicação da metodologia proposta neste trabalho permitiu analisar os resultados obtidos pelos algoritmos TOPSIS, FES e ES com relação a qualidade obtida pela seleção e o tempo necessário para computá-las. Ainda, por meio da utilização do indicador de desempenho PQA proposto neste trabalho, foi possível sintetizar a qualidade obtida em um valor absoluto e estabelecer uma relação de custo-benefício com o tempo necessário para se obter a solução. Na próxima seção serão apresentadas as conclusões obtidas nesta tese e os próximos passos deste trabalho.

CONCLUSÕES

Esta tese abordou o tema de seleção de recursos em IoT, para possibilitar a sua reutilização em diferentes contextos. Este tema é crucial para o desenvolvimento de serviços e aplicações em IoT, uma vez que os recursos disponíveis em tal ambiente podem ser utilizados por outras aplicações para propósitos diferentes do original. O número considerável de recursos com características semelhantes aplicados em diferentes contextos torna imprescindível a utilização de um bom processo de seleção, para que os recursos selecionados atendam aos requisitos da nova aplicação, evitando a geração de resultados inconsistentes e conclusões falhas. Além disso, um recurso pode ter seu estado original modificado e tornar-se inadequado ao longo do tempo. Por isso, o tempo necessário para selecionar tais recursos deve ser considerado de acordo com a natureza da aplicação alvo.

Na literatura, há diversos trabalhos que propõem diferentes estratégias para realizar a seleção de recursos em ambientes IoT, normalmente por meio de técnicas MCDA que determinam quais recursos são mais adequados de acordo com as restrições de um contexto específico. Embora a maior parte das técnicas apresentadas nestes trabalhos avalie o tempo necessário para realizar a seleção, em nenhum deles é assegurada a qualidade dos recursos selecionados. Em outras palavras, não há garantias de que o recurso escolhido atenda da melhor forma possível as restrições impostas para a sua aplicação em outro contexto.

Desta maneira, a fim de validar a hipótese proposta nesta tese, um conjunto de métodos foi proposto para melhorar a relação custo-benefício do processo de seleção de recursos em ambientes IoT. Neste cenário, o custo é mensurado pelo tempo necessário para se obter uma solução enquanto o benefício é representado pelo valor da qualidade obtida. O conjunto de métodos propostos nesta tese pode ser representado por:

- Uma metodologia para mensurar a qualidade da solução ofertada pelas técnicas MCDA, a qual permitiu comparar a solução entre diferentes técnicas;

- Duas técnicas para melhorar a qualidade das soluções ofertadas e também reduzir o tempo de computação;
- Um índice de desempenho utilizado para representar em um único valor a qualidade da solução obtida.

Os estudos de caso utilizados para validar as técnicas MCDA permitiram analisar como a qualidade da seleção é alterada de acordo com as condições, restrições e prioridades de um contexto. Os resultados obtidos indicaram que a qualidade da solução diminui à medida que mais critérios são utilizados, devido ao número elevado de conflitos entre os critérios. A qualidade da solução também é influenciada pelo número de recursos a serem selecionados, visto que quanto maior o número de recursos selecionados maior é a probabilidade de se obter um recurso ótimo e melhorar a qualidade da solução.

A execução da técnica MCDA de forma distribuída também pode depreciar a qualidade da solução, pois apenas recursos locais são considerados durante o processo. Com relação às técnicas analisadas, não é possível determinar um algoritmo ideal, visto que a relação custo-benefício entre eles é diretamente proporcional. A técnica ES obteve qualidade equivalente à da solução ótima (FNDS), porém com um tempo de processamento menor em alguns cenários, demonstrando assim a veracidade da hipótese *“Propor um conjunto de métodos que melhore a relação custo-benefício do processo de seleção de recursos em ambientes IoT”*.

7.1 Contribuições

Esta tese gerou contribuições diretamente relacionadas à hipótese proposta e para a área de pesquisa na qual este trabalho está inserido. Dentre as principais contribuições desta tese destacam-se:

- **Uma metodologia que melhore a relação custo-benefício do processo de seleção de recursos em ambientes IoT:** observando o estado da arte de seleção de recursos em ambientes IoT e a falta de métodos para avaliar a qualidade de técnicas para realizar tal tarefa, esta tese apresentou uma metodologia para avaliar as técnicas MCDA para a seleção de recursos em ambientes IoT. Esta metodologia define as etapas necessárias para comparar soluções obtidas por técnicas MCDA em termos de qualidade e tempo, estabelecendo a relação custo-benefício. Por meio dessa relação é possível escolher a técnica mais adequada de acordo com o foco da aplicação.
- **Estudo da influência das propriedades de contexto para a seleção de recursos em ambientes IoT:** considerando os trabalhos encontrados na literatura, uma análise da influência do número de critérios no processo de seleção de recursos foi conduzida. Este

estudo constatou que quanto maior o número de critérios utilizados menor é a qualidade da solução, já que geralmente tais critérios são conflitantes.

- **Análise da influência da atribuição de pesos nos critérios para a execução das técnicas MCDA:** a utilização de pesos para priorizar um determinado critério pode melhorar ou piorar a qualidade de uma solução em termos de capacidade. Por outro lado, a atribuição de pesos aos critérios pode diminuir a densidade de uma solução e aumentar a sua diversidade.
- **Uma métrica para a comparação de técnicas de seleção:** a partir da necessidade de se comparar os resultados obtidos e não encontrando na literatura uma métrica capaz de sintetizar a qualidade de uma solução em um valor absoluto, esta tese propôs a utilização da métrica Perda de Qualidade Acumulada (PQA). Esta métrica visa mensurar a qualidade de uma solução em um valor absoluto, o qual é derivado dos valores da métrica ONVGR em cada fronteira da solução.
- **Algoritmos para realizar a seleção de recursos em ambientes de IoT:** com base nos resultados coletados em experimentos preliminares, foram propostos dois algoritmos visando melhorar a relação entre o custo computacional de uma solução e a qualidade obtida. Desta maneira, os algoritmos *Elimination Sort* (ES) e *Fast Elimination Sort* (FES) foram propostos. Ambos os algoritmos realizam a seleção em duas fases. Na primeira fase um filtro é aplicado utilizando a técnica MCDA TOPSIS, a fim de eliminar os recursos que tenham baixa probabilidade de serem escolhidos. Na segunda fase o algoritmo FNDS é executado para calcular as relações de dominância entre as soluções classificadas pelo filtro. Por computar apenas parte das possíveis relações de dominância existentes, ambos algoritmos apresentaram qualidade igual ou superior a técnica TOPSIS.
- **Uma ferramenta para realizar a integração de dados em ambientes IoT:** para integrar as soluções de *middlewares* existentes a fim de acessar os seus recursos ou dados em forma de serviço, denominado *Visual Search for Internet of Things* (ViSIoT).

7.2 Publicações

As principais publicações oriundas desta tese são apresentadas a seguir:

- **Periódicos:**
 - NUNES, L. H.; ESTRELLA, J. C.; NAKAMURA, L. H.; LIBARDI, R. M. D. O.; FERREIRA, C. H.; JORGE, L. L.; REIFF-MARGANIEC, S. A **Distributed Sensor Data Search Platform for Internet of Things Environments**. Em *Services Transactions on Services Computing*, 2016.

- NUNES, L. H.; ESTRELLA, J. C.; PERERA, C.; REIFF-MARGANIEC, S.; BO-TAZZO DELBEM, A. C. **Multi-criteria IoT resource discovery: a comparative analysis**. Em *Software: Practice and Experience*, 2017.

- **Conferências:**

- NUNES, L. H.; ESTRELLA, J. C.; DELBEM, A. C. B.; PERERA, C.; REIFF-MARGANIEC, S. **The effects of relative importance of user constraints in cloud of things resource discovery: A case study**. Em *Proceedings of the 9th International Conference on Utility and Cloud Computing*, 2016.
- NUNES, L. H.; ESTRELLA, J. C.; PERERA, C.; REIFF-MARGANIEC, S.; DELBEM, A. C. **The elimination-selection based algorithm for efficient resource discovery in Internet of Things environments**. Em *Consumer Communications & Networking Conference (CCNC)*, 2018.

Além dos trabalhos listados acima, as seguintes publicações em parceria foram originadas:

- FERREIRA, A. G.; NUNES, L. H.; ESTRELLA, J. C. **Avaliação de desempenho do algoritmo TOPSIS aplicado no Contexto de Internet das Coisas**. Em *Anais da 7ª Escola Regional de Alto Desempenho de São Paulo (ERAD-SP)*, 2016.
- FERREIRA, A. G.; NUNES, L. H.; ESTRELLA, J. C. **Avaliação qualitativa dos algoritmos TOPSIS, WSM e GRA aplicado no Contexto de Internet das Coisas**. Em *Anais da 8ª Escola Regional de Alto Desempenho de São Paulo (ERAD-SP)*, 2017.
- BARROS, V. A.; NUNES, L. H.; DE SOUZA, P. S.; ESTRELLA, J. C.. **Paralelização do algoritmo Fast Non-Dominated Sorting para o ranqueamento de cidades de acordo com informações do clima**. Em *Anais da 8ª Escola Regional de Alto Desempenho de São Paulo (ERAD-SP)*, 2017.

7.3 Dificuldades

A IoT é considerada como um dos maiores avanços na história da indústria de tecnologia, e em breve será parte inerente de muitos objetos com os quais interagimos como televisores, carros, hotéis, restaurantes e pessoas, compondo um universo de bilhões de recursos disponíveis.

Neste contexto, uma das maiores dificuldades para o desenvolvimento do trabalho apresentado nesta tese foi a definição de um ambiente de experimentos que representasse a realidade dos ambientes IoT. Para a definição deste ambiente, foi necessário encontrar centenas de milhares de recursos com dados reais para reproduzir tal ambiente próximo a realidade. Como

discutido na Seção 3, pouquíssimos trabalhos abordam cenários com centenas de milhares de recursos disponíveis, tornando-se assim um diferencial deste trabalho.

A utilização deste ambiente aumentou a complexidade de execução das técnicas MCDA, principalmente quando o algoritmo FNDS foi utilizado. Tal algoritmo utiliza recursos computacionais como memória e armazenamento de forma intensiva para computar as relações de dominância entre os recursos avaliados, sobrecarregando o ambiente de execução.

Para a execução dos experimentos deste trabalho, foi necessário gerenciar a utilização destes recursos diretamente no código fonte. Tecnicamente, as estruturas de dados utilizadas para armazenar as relações de dominância de cada um dos recursos disponíveis foi transferida da memória principal do servidor para um banco de dados. Ao passo que as relações de dominância de um recurso eram computadas e salvas no banco de dados, liberando a memória principal do servidor para o próximo recurso.

7.4 Trabalhos Futuros

Esta tese não encerra as possibilidades de estudos relacionados ao desenvolvimento de métodos para auxiliar a provisão de recursos em ambientes IoT. Outras pesquisas devem ser conduzidas a partir dos resultados descritos neste trabalho. Dentre os próximos trabalhos, os seguintes tópicos merecem destaque:

- **Aplicação dos métodos em outros estudos de caso:** Neste trabalho foi utilizado um estudo de caso sintético para se iniciar uma cultura de milho considerando-se centenas de milhares de recursos. Os métodos propostos nesta tese devem ser adaptados, aplicados e avaliados em outros cenários, considerando-se o número de recursos disponíveis em diferentes escalas.
- **Análise dos algoritmos em termos de convergência e diversidade:** os resultados apresentados no corpo desta tese foram analisados apenas em termos de capacidade e por isso, os algoritmos FES e ES deverão ser analisados em termos de convergência e diversidade para determinar os melhores cenários para a sua aplicação.
- **Gerenciamento da memória pela técnica FNDS:** a fim de evitar a sobrecarga da memória principal do servidor, um banco de dados é utilizado para auxiliar o gerenciamento da estrutura de dados utilizadas para armazenar as relações de dominância computadas pela técnica FNDS. O acesso ao banco de dados pode ser otimizado para recuperar ou armazenar recursos em blocos, visto que a modificação realizada no código realiza acessos cada vez que um recurso precisa armazenar ou consultar os recursos dominados.
- **Utilização de técnicas para aumento da qualidade da seleção:** Considerando os algoritmos FES e ES, técnicas como aprendizado de máquina podem ser aplicadas para alterar

o valor do número de recursos selecionados no filtro de forma dinâmica, de acordo com o valor de uma qualidade esperada ou de um tempo limite.

- **Desenvolvimento de uma ferramenta para auxiliar a tomada de decisão pelos usuários:** Uma ferramenta que utilize os algoritmos descritos neste trabalho pode ser desenvolvida para auxiliar a tomada de decisão de qualquer usuário mediante a entrada de um conjunto de dados e suas funções objetivo.
- **Seleção de infraestrutura computacionais:** As técnicas MCDA podem auxiliar na escolha da infraestrutura computacional tais como *desktop* locais, *clusters* ou nuvem a serem utilizadas por aplicações de acordo com a sua carga de trabalho atual.
- **Arquitetura para a integração de *middlewares* e recursos:** A partir da ferramenta ViSIoT serão definidos os blocos funcionais de uma arquitetura para realizar a provisão de recursos como serviço considerando questões como provisão e monitoramento em tempo real, segurança e privacidade.

REFERÊNCIAS

- AAZAM, M.; HUH, E.-N. Fog computing and smart gateway based communication for cloud of things. In: **Future Internet of Things and Cloud (FiCloud), 2014 International Conference on**. [s.n.], 2014. p. 464–470. Acessado em 17 de jul. 2018. Disponível em: <<https://doi.org/10.1109/FiCloud.2014.83>>. Citado nas páginas 37, 43, 44 e 52.
- ABDELWAHAB, S.; HAMDAOUI, B.; GUIZANI, M.; ZNATI, T. Cloud of things for sensing as a service: Sensing resource discovery and virtualization. In: **2015 IEEE Global Communications Conference (GLOBECOM)**. [s.n.], 2015. p. 1–7. Acessado em 17 de jul. 2018. Disponível em: <<https://doi.org/10.1109/GLOCOM.2015.7417252>>. Citado na página 67.
- ABDULLAH, L.; ADAWIYAH, C. R. Simple additive weighting methods of multi criteria decision making and applications: A decade review. **International Journal of Information Processing and Management**, Advanced Institutes of Convergence Information Technology, v. 5, n. 1, p. 39, 2014. Acessado em 17 de jul. 2018. Disponível em: <<https://www.semanticscholar.org/paper/Simple-Additive-Weighting-Methods-of-Multi-criteria-Abdullah-Adawiyah/05a0d6d88b7a86fd783f736532af68e0e297e299>>. Citado na página 155.
- ABERER, K.; HAUSWIRTH, M. Middleware support for the "internet of things". 2006. Acessado em 17 de jul. 2018. Disponível em: <<https://pdfs.semanticscholar.org/9018/bd5065cfdaa10e5c3aa1f989ce58847f8f35.pdf>>. Citado nas páginas 55 e 89.
- ABRAHAM, A.; JAIN, L. **Evolutionary multiobjective optimization**. Springer, 2005. Acessado em 17 de jul. 2018. Disponível em: <<https://dl.acm.org/citation.cfm?id=1051744>>. Citado nas páginas 57 e 58.
- ABU-ELKHEIR, M.; HAYAJNEH, M.; ALI, N. Data management for the internet of things: Design primitives and solution. **Sensors**, MDPI AG, v. 13, n. 11, p. 15582–15612, nov 2013. Acessado em 17 de jul. 2018. Disponível em: <<http://dx.doi.org/10.3390/s131115582>>. Citado na página 48.
- AFSHARI, A.; MOJAHED, M.; YUSUFF, R. M. Simple additive weighting approach to personnel selection problem. **International Journal of Innovation, Management and Technology**, IACSIT Press, v. 1, n. 5, p. 511–515, 2010. Acessado em 17 de jul. 2018. Disponível em: <<http://www.ijimt.org/index.php?m=content&c=index&a=show&catid=26&id=347>>. Citado na página 155.
- AGGARWAL, C. C.; ASHISH, N.; SHETH, A. The internet of things: A survey from the data-centric perspective. In: **Managing and Mining Sensor Data**. Springer Science Business Media, 2012. p. 383–428. Disponível em: <http://dx.doi.org/10.1007/978-1-4614-6309-2_12>. Citado nas páginas 48 e 50.
- ALAM, S.; NOLL, J. A semantic enhanced service proxy framework for internet of things. In: **Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on Int'l Conference on Cyber, Physical and Social Computing (CPSCoM)**. [s.n.],

2010. p. 488–495. Acessado em 17 de jul. 2018. Disponível em: <<https://doi.org/10.1109/GreenCom-CPSCoM.2010.116>>. Citado nas páginas 63 e 64.

ANTONIC, A.; ROANKOVIC, K.; MARJANOVIC, M.; PRIPUIC, K.; ZARKO, I. A mobile crowdsensing ecosystem enabled by a cloud-based publish/subscribe middleware. In: **Future Internet of Things and Cloud (FiCloud), 2014 International Conference on**. [s.n.], 2014. p. 107–114. Acessado em 17 de jul. 2018. Disponível em: <<https://doi.org/10.1109/FiCloud.2014.27>>. Citado na página 64.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. **Comput. Netw.**, Elsevier North-Holland, Inc., New York, NY, USA, v. 54, n. 15, p. 2787–2805, out. 2010. ISSN 1389-1286. Disponível em: <<http://dx.doi.org/10.1016/j.comnet.2010.05.010>>. Citado na página 36.

BALLESTERO, E. Compromise programming: A utility-based linear-quadratic composite metric from the trade-off between achievement and balanced (non-corner) solutions. **European journal of operational research**, Elsevier, v. 182, n. 3, p. 1369–1382, 2007. Acessado em 17 de out. 2018. Disponível em: <<https://doi.org/10.1016/j.ejor.2006.09.049>>. Citado na página 157.

BANDYOPADHYAY, D.; SEN, J. Internet of things: Applications and challenges in technology and standardization. **Wireless Personal Communications**, Springer US, v. 58, n. 1, p. 49–69, 2011. ISSN 0929-6212. Disponível em: <<http://dx.doi.org/10.1007/s11277-011-0288-5>>. Citado na página 51.

BAO, C.; XU, L.; GOODMAN, E. D.; CAO, L. A novel non-dominated sorting algorithm for evolutionary multi-objective optimization. **Journal of Computational Science**, v. 23, p. 31 – 43, 2017. ISSN 1877-7503. Acessado em 08 de out. 2018. Disponível em: <<https://doi.org/10.1016/j.jocs.2017.09.015>>. Citado na página 38.

BEHZADIAN, M.; OTAGHSARA, S. K.; YAZDANI, M.; IGNATIUS, J. A state-of the-art survey of TOPSIS applications. **Expert Systems with Applications**, Elsevier BV, v. 39, n. 17, p. 13051–13069, dec 2012. Disponível em: <<http://dx.doi.org/10.1016/j.eswa.2012.05.056>>. Citado na página 60.

BIJARBOONEH, F. H.; DU, W.; NGAI, E. C. H.; FU, X.; LIU, J. Cloud-assisted data fusion and sensor selection for internet of things. **IEEE Internet of Things Journal**, v. 3, n. 3, p. 257–268, June 2016. ISSN 2327-4662. Acessado em 19 de jul. 2018. Disponível em: <<https://doi.org/10.1109/JIOT.2015.2502182>>. Citado nas páginas 74 e 77.

BORGES, L.; VELEZ, F.; LEBRES, A. Survey on the characterization and classification of wireless sensor networks applications. **Communications Surveys Tutorials, IEEE**, PP, n. 99, p. 1–1, 2014. ISSN 1553-877X. Acessado em 19 de jul. 2018. Disponível em: <<https://doi.org/10.1109/COMST.2014.2320073>>. Citado na página 43.

BOTTS, M.; PERCIVALL, G.; REED, C.; DAVIDSON, J. Ogc® sensor web enablement: Overview and high level architecture. In: **GeoSensor networks**. Springer, 2008. p. 175–190. Acessado em 19 de jul. 2018. Disponível em: <https://doi.org/10.1007/978-3-540-79996-2_10>. Citado nas páginas 50 e 55.

BOVET, G.; HENNEBERT, J. Distributed semantic discovery for web-of-things enabled smart buildings. In: **New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on**. [s.n.], 2014. p. 1–5. Acessado em 19 de jul. 2018. Disponível em: <<https://doi.org/10.1109/NTMS.2014.6814015>>. Citado nas páginas 63 e 64.

BUI, L. T.; ALAM, S. An introduction to multi-objective optimization. 2008. Acessado em 19 de jul. 2018. Disponível em: <<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.212.6649&rep=rep1&type=pdf>>. Citado na página 57.

BURANGE, A. W.; MISALKAR, H. D. Review of internet of things in development of smart cities with data management & privacy. In: **2015 International Conference on Advances in Computer Engineering and Applications**. Institute of Electrical & Electronics Engineers (IEEE), 2015. Acessado em 19 de jul. 2018. Disponível em: <<http://dx.doi.org/10.1109/ICACEA.2015.7164693>>. Citado na página 46.

CARAMIA, M.; DELL'OLMO, P. **Multi-objective Management in Freight Logistics**. Springer Science Business Media, 2008. Acessado em 19 de jul. 2018. Disponível em: <<http://dx.doi.org/10.1007/978-1-84800-382-8>>. Citado na página 58.

CARLSON, D.; SCHRADER, A. Ambient ocean: A web search engine for context-aware smart resource discovery. In: **Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing(CPSCom), IEEE**. [s.n.], 2014. p. 177–184. Acessado em 19 de jul. 2018. Disponível em: <<https://doi.org/10.1109/iThings.2014.34>>. Citado nas páginas 63, 73 e 77.

CHRISTOPHE, B. Managing massive data of the internet of things through cooperative semantic nodes. In: **Semantic Computing (ICSC), 2012 IEEE Sixth International Conference on**. [s.n.], 2012. p. 93–100. Acessado em 19 de jul. 2018. Disponível em: <<https://doi.org/10.1109/ICSC.2012.29>>. Citado nas páginas 63 e 64.

CISCO. **The IoT Value/Trust Paradox: Building Trust and Value in the Data Exchange Between People, Things and Providers**. 2018. Acessado em 19 de jul. 2018. Disponível em: <https://www.jasper.com/sites/default/files/cisco_iiot_survey_-_the_value_trust_paradox_final-1_2.pdf>. Citado na página 35.

COLLETTE, Y.; SIARRY, P. **Multiobjective Optimization**. Springer Berlin Heidelberg, 2004. Disponível em: <<http://dx.doi.org/10.1007/978-3-662-08883-8>>. Citado nas páginas 38 e 62.

COURTAIS, C.; TACONET, C.; CONAN, D.; CHABRIDON, S.; GOMES, P.; CAVALCANTE, E.; BATISTA, T. IoTVar to transparently handle interactions between applications and IoT platforms. In: **Proceedings of the 4th Workshop on Middleware and Applications for the Internet of Things - M4IoT 17**. ACM Press, 2017. Acessado em 19 de jul. 2018. Disponível em: <<https://doi.org/10.1145/3152141.3152390>>. Citado nas páginas 76 e 77.

DADA, A.; THIESSE, F. Sensor applications in the supply chain: The example of quality-based issuing of perishables. Springer Berlin Heidelberg, v. 4952, p. 140–154, 2008. Acessado em 19 de jul. 2018. Disponível em: <http://dx.doi.org/10.1007/978-3-540-78731-0_9>. Citado na página 36.

DATTA, S.; BONNET, C.; NIKAEIN, N. An iot gateway centric architecture to provide novel m2m services. In: **Internet of Things (WF-IoT), 2014 IEEE World Forum on**. [s.n.], 2014. p. 514–519. Acessado em 19 de jul. 2018. Disponível em: <<https://doi.org/10.1109/WF-IoT.2014.6803221>>. Citado na página 63.

DAUBERT, J.; WIESMAIER, A.; KIKIRAS, P. A view on privacy & trust in IoT. In: **2015 IEEE International Conference on Communication Workshop (ICCW)**. Institute of Electrical &

- Electronics Engineers (IEEE), 2015. Acessado em 19 de jul. 2018. Disponível em: <http://dx.doi.org/10.1109/ICCW.2015.7247581>. Citado na página 46.
- DAWS, R. **Study: Connected car owners will trade personal data for services**. 2018. Disponível em <https://www.iottechnews.com/news/2018/jun/07/study-connected-car-data-services/>. Acessado em 13 ago. 2018. Citado na página 38.
- DEB, K. **Multi-objective optimization using evolutionary algorithms**. Chichester New York: John Wiley & Sons, 2001. ISBN 978-0-471-87339-6. Citado nas páginas 38, 58, 59 e 95.
- DIAZ-MONTES, J.; ABDELBAKY, M.; ZOU, M.; PARASHAR, M. Cometcloud: Enabling software-defined federations for end-to-end application workflows. **Internet Computing, IEEE**, IEEE, v. 19, n. 1, p. 69–73, 2015. Acessado em 19 de jul. 2018. Disponível em: <https://doi.org/10.1109/MIC.2015.4>. Citado na página 68.
- DING, Z.; DAI, J.; GAO, X.; YANG, Q. A hybrid search engine framework for the internet of things. In: **Web Information Systems and Applications Conference (WISA), 2012 Ninth**. [s.n.], 2012. p. 57–60. Acessado em 19 de jul. 2018. Disponível em: <https://doi.org/10.1109/WISA.2012.11>. Citado nas páginas 72 e 77.
- DISTEFANO, S.; MERLINO, G.; PULIAFITO, A. Sensing and actuation as a service: A new development for clouds. In: **Network Computing and Applications (NCA), 2012 11th IEEE International Symposium on**. [s.n.], 2012. p. 272–275. Acessado em 19 de jul. 2018. Disponível em: <https://doi.org/10.1109/NCA.2012.38>. Citado nas páginas 52 e 53.
- DOBLAS-REYES, F.; GARCIA, A.; HANSEN, J.; MARIANI, L.; NAIN, A.; RAMESH, K.; RATHORE, L.; VENKATARAMAN, R. Weather and climate forecasts for agriculture. **Guide to agricultural, meteorological practices**, p. 57, 2003. Citado na página 86.
- DODGSON, J.; SPACKMAN, M.; PEARMAN, A.; PHILLIPS, L. **Multi-criteria analysis: a manual**. Department for Communities and Local Government: London, 2009. Acessado em 19 de jul. 2018. ISBN 9781409810230. Disponível em: <http://eprints.lse.ac.uk/12761/>. Citado na página 60.
- DOHR, A.; MODRE-OPSRIAN, R.; DROBICS, M.; HAYN, D.; SCHREIER, G. The internet of things for ambient assisted living. In: **Information Technology: New Generations (ITNG), 2010 Seventh International Conference on**. [s.n.], 2010. p. 804–809. Acessado em 19 de jul. 2018. Disponível em: <https://doi.org/10.1109/ITNG.2010.104>. Citado na página 41.
- DOUKAS, C.; ANTONELLI, F. Developing and deploying end-to-end interoperable & discoverable iot applications. In: **Communications (ICC), 2015 IEEE International Conference on**. [s.n.], 2015. p. 673–678. Acessado em 19 de jul. 2018. Disponível em: <http://doi.org/10.1109/ICC.2015.7248399>. Citado na página 68.
- E, X.; HAN, J.; WANG, Y.; LIU, L. Big data-as-a-service: Definition and architecture. In: **Communication Technology (ICCT), 2013 15th IEEE International Conference on**. [s.n.], 2013. p. 738–742. Acessado em 02 de ago. 2018. Disponível em: <https://doi.org/10.1109/ICCT.2013.6820472>. Citado na página 54.
- ELAHI, B.; ROMER, K.; OSTERMAIER, B.; FAHRMAIR, M.; KELLERER, W. Sensor ranking: A primitive for efficient content-based sensor search. In: **Information Processing in Sensor Networks, 2009. IPSN 2009. International Conference on**. [s.n.], 2009. p. 217–228.

Acessado em 02 de ago. 2018. Disponível em: <<https://ieeexplore.ieee.org/document/5211927/>>. Citado nas páginas 71 e 77.

EVDOKIMOV, S.; FABIAN, B.; KUNZ, S.; SCHOENEMANN, N. Comparison of discovery service architectures for the internet of things. In: **Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), 2010 IEEE International Conference on**. [s.n.], 2010. p. 237–244. Acessado em 02 de ago. 2018. Disponível em: <<https://doi.org/10.1109/SUTC.2010.22>>. Citado na página 63.

FANG, H.; WANG, Q.; TU, Y.; HORSTEMEYER, M. F. An efficient non-dominated sorting method for evolutionary algorithms. **Evolutionary Computation**, v. 16, n. 3, p. 355–384, Sept 2008. ISSN 1063-6560. Acessado em 08 de out. 2018. Disponível em: <<https://doi.org/10.1162/evco.2008.16.3.355>>. Citado na página 38.

FERSI, G. Middleware for internet of things: A study. In: **2015 International Conference on Distributed Computing in Sensor Systems**. Institute of Electrical & Electronics Engineers (IEEE), 2015. Acessado em 02 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1109/DCOSS.2015.43>>. Citado nas páginas 43 e 46.

GAO, F.; ALI, M. I.; MILEO, A. Semantic discovery and integration of urban data streams. In: **CEUR Workshop Proceedings**. [s.n.], 2014. v. 1280, p. 15–30. Acessado em 02 de ago. 2018. Disponível em: <<https://dl.acm.org/citation.cfm?id=2878783>>. Citado nas páginas 38, 74 e 77.

GARTNER. **Gartner Says 4.9 Billion Connected "Things" Will Be in Use in 2015**. 2014. Disponível em <<http://www.gartner.com/newsroom/id/2905717>>. Último acesso em: 27/07/2015. Citado na página 41.

_____. **Gartner Says 8.4 Billion Connected "Things" Will Be in Use in 2017, Up 31 Percent From 2016**. 2017. Available in <<http://www.gartner.com/newsroom/id/3598917>>. Last access: 31/03/2017. Citado na página 35.

GIBBONS, P.; KARP, B.; KE, Y.; NATH, S.; SESHAN, S. Irisnet: an architecture for a worldwide sensor web. **Pervasive Computing, IEEE**, v. 2, n. 4, p. 22–33, Oct 2003. ISSN 1536-1268. Acessado em 02 de ago. 2018. Disponível em: <<https://doi.org/10.1109/MPRV.2003.1251166>>. Citado na página 54.

GIRAU, R.; MARTIS, S.; ATZORI, L. Lysis: A platform for iot distributed applications over socially connected objects. **IEEE Internet of Things Journal**, v. 4, n. 1, p. 40–51, Feb 2017. ISSN 2327-4662. Acessado em 02 de ago. 2018. Disponível em: <<https://doi.org/10.1109/JIOT.2016.2616022>>. Citado na página 70.

GOMES, M.; RIGHI, R. da R.; COSTA, C. A. da. Internet of things scalability: Analyzing the bottlenecks and proposing alternatives. In: **2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)**. Institute of Electrical & Electronics Engineers (IEEE), 2014. Acessado em 02 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1109/ICUMT.2014.7002114>>. Citado na página 46.

GONG, H.; SHI, Y.; LI, G. Context-aware sensor search framework in semantic web of things. In: **2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)**. [s.n.], 2016. v. 01, p. 94–98. Acessado em 02 de ago. 2018. Disponível em: <<https://doi.org/10.1109/IHMSC.2016.170>>. Citado nas páginas 74 e 77.

GUINARD, D.; TRIFA, V.; KARNOUSKOS, S.; SPIESS, P.; SAVIO, D. Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services. **Services Computing, IEEE Transactions on**, v. 3, n. 3, p. 223–235, July 2010. ISSN 1939-1374. Acessado em 02 de ago. 2018. Disponível em: <https://doi.org/10.1109/TSC.2010.3>. Citado nas páginas 72 e 77.

GUPTA, V.; UDUPI, P.; POURSOHI, A. Early lessons from building sensor.network: an open data exchange for the web of things. In: **Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on**. [s.n.], 2010. p. 738–744. Acessado em 02 de ago. 2018. Disponível em: <https://doi.org/10.1109/PERCOMW.2010.5470530>. Citado na página 65.

HUANG, B.; BOUGUETTAYA, A.; DONG, H.; CHEN, L. Service mining for internet of things. In: **Service-Oriented Computing**. Springer International Publishing, 2016. p. 566–574. Acessado em 27 de ago. 2018. Disponível em: http://dx.doi.org/10.1007/978-3-319-46295-0_36. Citado nas páginas 75 e 77.

HUANG, J.-J.; TZENG, G.-H.; LIU, H.-H. A revised VIKOR model for multiple criteria decision making - the perspective of regret theory. In: **Communications in Computer and Information Science**. Springer Science Business Media, 2009. p. 761–768. Acessado em 17 de out. 2018. Disponível em: http://dx.doi.org/10.1007/978-3-642-02298-2_112. Citado na página 157.

IDC, I. D. C. **IDC Forecasts Worldwide Spending on the Internet of Things to Reach \$772 Billion in 2018**. 2017. "https://www.idc.com/getdoc.jsp?containerId=prUS43295217". "Acessado em 19 fev. 2018". Citado na página 35.

IKEBE, T.; NOGUCHI, H.; HOSHIKAWA, N. Distributed live data search architecture for resource discovery on internet of things. In: **2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)**. [s.n.], 2016. p. 591–596. Acessado em 27 de ago. 2018. Disponível em: <http://doi.ieeecomputersociety.org/10.1109/WF-IoT.2016.7845394>. Citado na página 69.

INC., L. **Xively is the Public Cloud specifically built for the Internet of Things**. 2014. Available in https://xively.com/whats_xively/. Last access: 10/07/2014. Citado na página 54.

IORGA, M.; FELDMAN, L.; BARTON, R.; MARTIN, M. J.; GOREN, N.; MAHMOUDI, C. **Fog computing conceptual model**. National Institute of Standards and Technology, 2018. Acessado em 27 de ago. 2018. Disponível em: <https://dx.doi.org/10.6028/NIST.SP.500-325>. Citado na página 49.

ISHAQ, I.; HOEBEKE, J.; ROSSEY, J.; POORTER, E. D.; MOERMAN, I.; DEMEESTER, P. Facilitating sensor deployment, discovery and resource access using embedded web services. In: **Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on**. [s.n.], 2012. p. 717–724. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.1109/IMIS.2012.48>. Citado na página 65.

ISLAM, S.; KWAK, D.; KABIR, M.; HOSSAIN, M.; KWAK, K. The internet of things for health care: A comprehensive survey. **Access, IEEE**, v. 3, p. 678–708, 2015. ISSN 2169-3536. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.1109/ACCESS.2015.2437951>. Citado na página 36.

JAIMES, A. L.; MARTINEZ, S. Z.; COELLO, C. A. C. An introduction to multiobjective optimization techniques. 2009. Citado nas páginas 57 e 58.

JARA, A.; LOPEZ, P.; FERNANDEZ, D.; CASTILLO, J.; ZAMORA, M.; SKARMETA, A. Mobile digcovery: A global service discovery for the internet of things. In: **Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on**. [s.n.], 2013. p. 1325–1330. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/WAINA.2013.261>>. Citado na página 65.

JIANG, S.; ONG, Y. S.; ZHANG, J.; FENG, L. Consistencies and contradictions of performance metrics in multiobjective optimization. **IEEE Transactions on Cybernetics**, v. 44, n. 12, p. 2391–2404, Dec 2014. ISSN 2168-2267. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/TCYB.2014.2307319>>. Citado na página 62.

JUNG, J.; CHUN, S.; JIN, X.; LEE, K.-H. Enabling smart objects discovery via constructing hypergraphs of heterogeneous IoT interactions. **Journal of Information Science**, SAGE Publications, v. 44, n. 1, p. 110–124, oct 2016. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1177/0165551516674164>>. Citado nas páginas 75 e 77.

KABADAYI, S.; PRIDGEN, A.; JULIEN, C. Virtual sensors: Abstracting data from physical sensors. In: IEEE COMPUTER SOCIETY. **Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks**. 2006. p. 587–592. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/WOWMOM.2006.115>>. Citado na página 51.

KAMILARIS, A.; PAPAKONSTANTINO, K.; PITSILLIDES, A. Exploring the use of dns as a search engine for the web of things. In: **Internet of Things (WF-IoT), 2014 IEEE World Forum on**. [s.n.], 2014. p. 100–105. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/WF-IoT.2014.6803128>>. Citado na página 65.

KHAN, R.; KHAN, S. U.; ZAHEER, R.; KHAN, S. Future internet: The internet of things architecture, possible applications and key challenges. In: **2012 10th International Conference on Frontiers of Information Technology**. Institute of Electrical & Electronics Engineers (IEEE), 2012. Acessado em 27 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1109/FIT.2012.53>>. Citado nas páginas 43, 44 e 46.

KHODADADI, F.; DASTJERDI, A. V.; BUYYA, R. Simurgh: A framework for effective discovery, programming, and integration of services exposed in IoT. In: **2015 International Conference on Recent Advances in Internet of Things (RIoT)**. Institute of Electrical & Electronics Engineers (IEEE), 2015. Acessado em 27 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1109/RIOT.2015.7104910>>. Citado nas páginas 38 e 68.

KILJANDER, J.; D'ELIA, A.; MORANDI, F.; HYTTINEN, P.; TAKALO-MATTILA, J.; YLISAUKKO-OJA, A.; SOININEN, J.-P.; CINOTTI, T. Semantic interoperability architecture for pervasive computing and internet of things. **Access, IEEE**, v. 2, p. 856–873, 2014. ISSN 2169-3536. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/ACCESS.2014.2347992>>. Citado nas páginas 63 e 66.

KORESHOFF, T. L.; ROBERTSON, T.; LEONG, T. W. Internet of things: A review of literature and products. In: **Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration**. New York, NY, USA: ACM, 2013. (OzCHI '13), p. 335–344. ISBN 978-1-4503-2525-7. Acessado em 27 de ago. 2018. Disponível em: <<http://doi.acm.org/10.1145/2541016.2541048>>. Citado nas páginas 37 e 41.

- KOTHARI, A.; BODDULA, V.; RAMASWAMY, L.; ABOLHASSANI, N. Dqs-cloud: A data quality-aware autonomic cloud for sensor services. In: **Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2014 International Conference on**. [s.n.], 2014. p. 295–303. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.4108/icst.collaboratecom.2014.257475>>. Citado nas páginas 68, 74 e 77.
- LIU, M.; LI, D.; MAO, H. A design and implementation of information service architecture for resource-constraint devices in internet of things. In: **Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on**. [s.n.], 2014. p. 1503–1508. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/CSE.2014.279>>. Citado na página 66.
- MAHAPATRO, A.; KHILAR, P. Fault diagnosis in wireless sensor networks: A survey. **Communications Surveys Tutorials, IEEE**, v. 15, n. 4, p. 2000–2026, Fourth 2013. ISSN 1553-877X. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/SURV.2013.030713.00062>>. Citado na página 42.
- MANNA, S.; BHUNIA, S.; MUKHERJEE, N. Vehicular pollution monitoring using iot. In: **Recent Advances and Innovations in Engineering (ICRAIE), 2014**. [s.n.], 2014. p. 1–5. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/ICRAIE.2014.6909157>>. Citado na página 36.
- MELL, P.; GRANCE, T. *et al.* The nist definition of cloud computing. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology Gaithersburg, 2011. Acessado em 27 de ago. 2018. Disponível em: <<http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>>. Citado na página 49.
- MEMARIANI, A.; AMINI, A.; ALINEZHAD, A. Sensitivity analysis of simple additive weighting method (saw): the results of change in the weight of one attribute on the final ranking of alternatives. **Journal of Industrial Engineering**, v. 4, p. 13–18, 2009. Acessado em 17 de out. 2018. Disponível em: <http://www.qjie.ir/article_28.html>. Citado na página 155.
- MICROSTRAIN, L. **SensorCloud Documentation**. 2014. Available in <<http://www.sensorcloud.com/documentation>>. Last access: 10/07/2014. Citado na página 54.
- MIETTINEN, K. **Nonlinear multiobjective optimization**. Springer Science & Business Media, 2012. v. 12. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1007/978-1-4615-5563-6>>. Citado na página 58.
- MRISSA, M.; MEDINI, L.; JAMONT, J.-P. Semantic discovery and invocation of functionalities for the web of things. In: **WETICE Conference (WETICE), 2014 IEEE 23rd International**. [s.n.], 2014. p. 281–286. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/WETICE.2014.50>>. Citado na página 66.
- NATH, S.; LIU, J.; ZHAO, F. Sensormap for wide-area sensor webs. **Computer**, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 40, n. 7, p. 90–93, jul. 2007. ISSN 0018-9162. Acessado em 27 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1109/MC.2007.250>>. Citado na página 55.
- NIST Big Data Interoperability Framework: Volume 1, Definitions. [S.l.], 2015. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.6028/nist.sp.1500-1>>. Citado na página 49.

NUNES, L.; NAKAMURA, L.; KUEHNE, B.; LIBARDI, R.; ADAMI, L.; ESTRELLA, J.; REIFF-MARGANIEC, S. Peesos: A web tool for planning and execution of experiments in service oriented systems. In: **21th IEEE International Conference on Web Services**. Anchorage, AK, USA: [s.n.], 2014. p. 606–613. Acessado em 27 de ago. 2018. Disponível em: <http://dx.doi.org/10.1109/icws.2014.90>. Citado na página 87.

NUNES, L. H.; ; ESTRELLA, J. C.; FERREIRA, C. H. G.; NAKAMURA, L. H. V.; LIBARDI, R. M.; OLIVEIRA, E. M. de; KUEHNE, B. T.; SOUZA, P. S. L.; SANTANA, R. H. C.; SANTANA, M. J.; REIFF-MARGANIEC, S.; ; ; ; ; AND. Dca-services: A distributed and collaborative architecture for conducting experiments in service oriented systems. **Services Transactions on Services Computing**, Services Society, v. 3, n. 4, p. 14–28, oct 2015. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.29268/stsc.2015.3.4.2>. Citado na página 87.

NUNES, L. H.; ESTRELLA, J. C.; NAKAMURA, L. H. V.; LIBARDI, R. M. d. O.; FERREIRA, C. H. G.; JORGE, L. L. R.; PERERA, C.; REIFF-MARGANIEC, S. A distributed sensor data search platform for internet of things environments. **Services Transactions on Services Computing**, Services Society, v. 4, n. 1, p. 1–12, jan 2016. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.29268/stsc.2016.4.1.1>. Citado nas páginas 88, 90 e 93.

NUNES, L. H.; ESTRELLA, J. C.; PERERA, C.; REIFF-MARGANIEC, S.; DELBEM, A. C. B. Multi-criteria IoT resource discovery: a comparative analysis. **Software: Practice and Experience**, Wiley, v. 47, n. 10, p. 1325–1341, dec 2016. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.1002/spe.2469>. Citado nas páginas 36, 37, 43 e 82.

OPRICOVIC, S.; TZENG, G.-H. Compromise solution by mcdm methods: A comparative analysis of vikor and topsis. **European Journal of Operational Research**, Elsevier, v. 156, n. 2, p. 445–455, 2004. Acessado em 17 de out. 2018. Disponível em: [https://doi.org/10.1016/S0377-2217\(03\)00020-1](https://doi.org/10.1016/S0377-2217(03)00020-1). Citado na página 157.

ORESTIS, A.; DIMITRIOS, A.; IOANNIS, C. Towards integrating iot devices with the web. In: **Emerging Technologies Factory Automation (ETFA), 2012 IEEE 17th Conference on**. [s.n.], 2012. p. 1–4. ISSN 1946-0740. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.1109/ETFA.2012.6489729>. Citado na página 67.

OSTERMAIER, B.; ROMER, K.; MATTERN, F.; FAHRMAIR, M.; KELLERER, W. A real-time search engine for the web of things. In: **Internet of Things (IOT), 2010**. [s.n.], 2010. p. 1–8. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.1109/IOT.2010.5678450>. Citado nas páginas 71 e 77.

PERERA, C.; VASILAKOS, A. V. A knowledge-based resource discovery for Internet of Things. **Knowledge-Based Systems**, v. 109, p. 122–136, 2016. ISSN 0950-7051. Acessado em 27 de ago. 2018. Disponível em: <https://doi.org/10.1016/j.knosys.2016.06.030>. Citado na página 38.

PERERA, C.; ZASLAVSKY, A.; CHRISTEN, P.; GEORGAKOPOULOS, D. Ca4iot: Context awareness for internet of things. In: **Proceedings of the 2012 IEEE International Conference on Green Computing and Communications**. Washington, DC, USA: IEEE Computer Society, 2012. (GREENCOM '12), p. 775–782. ISBN 978-0-7695-4865-4. Acessado em 27 de ago. 2018. Disponível em: <http://dx.doi.org/10.1109/GreenCom.2012.128>. Citado na página 56.

PERERA, C.; ZASLAVSKY, A.; CHRISTEN, P.; COMPTON, M.; GEORGAKOPOULOS, D. Context-aware sensor search, selection and ranking model for internet of things middleware. In: **Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management - Volume 01**. Washington, DC, USA: IEEE Computer Society, 2013. (MDM '13), p. 314–322. ISBN 978-0-7695-4973-6. Disponível em: <<http://dx.doi.org/10.1109/MDM.2013.46>>. Citado nas páginas 73 e 77.

PERERA, C.; ZASLAVSKY, A.; CHRISTEN, P.; GEORGAKOPOULOS, D. Sensing as a service model for smart cities supported by internet of things. **Transactions on Emerging Telecommunications Technologies**, Wiley, v. 25, n. 1, p. 81–93, sep 2013. Disponível em: <<https://doi.org/10.1002/ett.2704>>. Citado nas páginas 52 e 53.

_____. Context aware computing for the internet of things: A survey. **IEEE Communications Surveys Tutorials**, v. 16, n. 1, p. 414–454, First 2014. ISSN 1553-877X. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/SURV.2013.042313.00197>>. Citado nas páginas 41, 42 e 43.

PERERA, C.; ZASLAVSKY, A.; LIU, C. H.; COMPTON, M.; CHRISTEN, P.; GEORGAKOPOULOS, D. Sensor search techniques for sensing as a service architecture for the internet of things. **IEEE Sensors Journal**, v. 14, n. 2, p. 406–420, Feb 2014. ISSN 1530-437X. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/JSEN.2013.2282292>>. Citado na página 38.

PFISTERER, D.; ROMER, K.; BIMSCHAS, D.; KLEINE, O.; MIETZ, R.; TRUONG, C.; HASEMANN, H.; KROLLER, A.; PAGEL, M.; HAUSWIRTH, M. *et al.* Spitfire: toward a semantic web of things. **Communications Magazine, IEEE**, IEEE, v. 49, n. 11, p. 40–48, 2011. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/MCOM.2011.6069708>>. Citado nas páginas 72 e 77.

RAWAT, P.; SINGH, K.; CHAOUCHI, H.; BONNIN, J. Wireless sensor networks: a survey on recent developments and potential synergies. **The Journal of Supercomputing**, Springer US, v. 68, n. 1, p. 1–48, 2014. ISSN 0920-8542. Acessado em 27 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1007/s11227-013-1021-9>>. Citado na página 42.

RESEARCH, B. **Sensors: Technologies and Global Markets**. 2011. Available in <<http://www.bccresearch.com/market-research/instrumentation-and-sensors/sensors-technologies-markets-ias006d.html>>. Acessado em 27 de ago. 2018. Citado na página 35.

ROBLES, T.; ALCARRIA, R.; MARTIN, D.; MORALES, A.; NAVARRO, M.; CALERO, R.; IGLESIAS, S.; LOPEZ, M. An internet of things-based model for smart water management. In: **Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference on**. [s.n.], 2014. p. 821–826. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/WAINA.2014.129>>. Citado na página 36.

ROMER, K.; OSTERMAIER, B.; MATTERN, F.; FAHRMAIR, M.; KELLERER, W. Real-time search for real-world entities: A survey. **Proceedings of the IEEE**, v. 98, n. 11, p. 1887–1902, Nov 2010. ISSN 0018-9219. Citado nas páginas 42 e 44.

ROSE, K.; ELDRIDGE, S.; CHAPIN, L. **The Internet of Things: An overview**. The Internet Society, 2015. Acessado em 27 de ago. 2018. Disponível em: <<https://www.internetsociety.org/wp-content/uploads/2017/08/ISOC-IoT-Overview-20151221-en.pdf>>. Citado na página 35.

ROSENZWEIG, C.; IGLESIAS, A.; YANG, X.; EPSTEIN, P. R.; CHIVIAN, E. Climate change and extreme weather events; implications for food production, plant diseases, and pests. **Global change & human health**, Springer, v. 2, n. 2, p. 90–104, 2001. Acessado em 27 de ago. 2018. Disponível em: <<https://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1023&context=nasapub>>. Citado na página 86.

RUTA, M.; SCIOSCIA, F.; PINTO, A.; SCIASCIO, E. D.; GRAMEGNA, F.; IEVA, S.; LOSETO, G. Resource annotation, dissemination and discovery in the semantic web of things: A coap-based framework. In: **Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCoM), IEEE International Conference on and IEEE Cyber, Physical and Social Computing**. [s.n.], 2013. p. 527–534. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/GreenCom-iThings-CPSCoM.2013.103>>. Citado nas páginas 73 e 77.

RUTA, M.; SCIOSCIA, F.; SCIASCIO, E. D. A mobile matchmaker for resource discovery in the ubiquitous semantic web. In: **2015 IEEE International Conference on Mobile Services**. [s.n.], 2015. p. 336–343. ISSN 2329-6429. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/MobServ.2015.76>>. Citado nas páginas 74 e 77.

SHAH, M.; SARDANA, A. Searching in internet of things using vcs. In: **Proceedings of the First International Conference on Security of Internet of Things**. New York, NY, USA: ACM, 2012. (SecurIT '12), p. 63–67. ISBN 978-1-4503-1822-8. Acessado em 27 de ago. 2018. Disponível em: <<http://doi.acm.org/10.1145/2490428.2490437>>. Citado nas páginas 72 e 77.

SHETH, A.; HENSON, C.; SAHOO, S. S. Semantic sensor web. **Internet Computing, IEEE**, IEEE, v. 12, n. 4, p. 78–83, 2008. Citado na página 50.

SOLDATOS, J.; SERRANO, M.; HAUSWIRTH, M. Convergence of utility computing with the internet-of-things. In: **Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on**. [s.n.], 2012. p. 874–879. Acessado em 17 jul. 2018. Disponível em: <<https://ieeexplore.ieee.org/document/5470520/>>. Citado na página 56.

STIRBU, V. Towards a restful plug and play experience in the web of things. In: **Semantic Computing, 2008 IEEE International Conference on**. [s.n.], 2008. p. 512–517. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/ICSC.2008.51>>. Citado na página 63.

SUN, M.; SHI, Z.; CHEN, S.; ZHOU, Z.; DUAN, Y. Energy-efficient composition of configurable internet of things services. **IEEE Access**, Institute of Electrical and Electronics Engineers (IEEE), v. 5, p. 25609–25622, 2017. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/access.2017.2768544>>. Citado nas páginas 76 e 77.

TANGANELLI, G.; VALLATI, C.; MINGOZZI, E. Edge-centric distributed discovery and access in the internet of things. **IEEE Internet of Things Journal**, v. 5, n. 1, p. 425–438, Feb 2018. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/JIOT.2017.2767381>>. Citado na página 70.

TERZO, O.; RUIU, P.; BUCCI, E.; XHAFI, F. Data as a service (daas) for sharing and processing of large data collections in the cloud. In: **Complex, Intelligent, and Software Intensive Systems (CISIS), 2013 Seventh International Conference on**. [s.n.], 2013. p. 475–480. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/CISIS.2013.87>>. Citado na página 53.

- TIMÓTEO, T. S.; MARCOS-FILHO, J. Seed performance of different corn genotypes during storage. **Journal of Seed Science**, SciELO Brasil, v. 35, n. 2, p. 207–215, 2013. Acessado em 27 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1590/S2317-15372013000200010>>. Citado nas páginas 85 e 86.
- TRUONG, C.; ROMER, K. Content-based sensor search for the web of things. In: **Global Communications Conference (GLOBECOM), 2013 IEEE**. [s.n.], 2013. p. 2654–2660. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/GLOCOM.2013.6831475>>. Citado nas páginas 71 e 77.
- TRUONG, H.-L.; DUSTDAR, S. On analyzing and specifying concerns for data as a service. In: IEEE. **Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific**. 2009. p. 87–94. Acessado em 27 de ago. 2018. Disponível em: <https://www.dbai.tuwien.ac.at/proj/InfInt/Publications/TruongDustdar_ConcernsDaaS_apsc09.pdf>. Citado na página 53.
- TZENG, G.; HUANG, J. **Multiple Attribute Decision Making: Methods and Applications**. [S.l.]: Taylor & Francis, 2011. (A Chapman & Hall book). ISBN 9781439861578. Citado nas páginas 38, 60, 155 e 157.
- VAR, S. **Cereal grains : properties, processing, and nutritional attributes**. Boca Raton, FL: CRC Press/Taylor & Francis, 2010. ISBN 9781439815601. Citado na página 85.
- VERMESAN, O. **Internet of things - from research and innovation to market deployment**. Aalborg, Denmark: River Publishers, 2013. ISBN 9788793102941. Citado nas páginas 48 e 52.
- _____. **Internet of things : converging technologies for smart environments and integrated ecosystems**. Aalborg, Denmark: River Publishers, 2013. ISBN 9788792982735. Citado nas páginas 46, 50, 51 e 52.
- VERMESAN, O.; FRIESS, P.; GUILLEMIN, P.; GUSMEROLI, S.; SUNDMAEKER, H.; BASSI, A.; JUBERT, I. S.; MAZURA, M.; HARRISON, M.; EISENHAUER, M.; DOODY, P. Internet of things strategic research roadmap. **Internet of Things-Global Technological and Societal Trends**, p. 9–52, 2011. Acessado em 27 de ago. 2018. Disponível em: <<https://brage.bibsys.no/xmlui/bitstream/handle/11250/2430372/SINTEF%2BS13363.pdf?sequence=2>>. Citado na página 41.
- WHITMORE, A.; AGARWAL, A.; XU, L. D. The internet of things—a survey of topics and trends. **Information Systems Frontiers**, Springer US, p. 1–14, 2014. ISSN 1387-3326. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1007/s10796-014-9489-2>>. Citado na página 41.
- WILGENGEBROED. **The Internet of Things**. 2012. Disponível em: <<https://www.flickr.com/photos/wilgengebroed/8249565455/>>. Citado na página 47.
- WIRTZ, H.; WEHRLE, K. Opening the loops - towards semantic, information-centric networking in the internet of things. In: **Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2013 10th Annual IEEE Communications Society Conference on**. [s.n.], 2013. p. 18–24. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/SAHCN.2013.6644955>>. Citado na página 37.
- XU, L.; YANG, J.-B. **Introduction to multi-criteria decision making and the evidential reasoning approach**. [s.n.], 2001. Acessado em 27 de ago. 2018. Disponível

em: <http://www.academia.edu/download/31563947/Introduction_to_Multi-Criteria_Decision_Making_evidential_reasoning_approach.pdf>. Citado na página 60.

YAN, B.; YAO, H.-P.; NAKAMURA, M.; MATSUMOTO, S.; SETO, H. A platform of service registry to discover service resources in ubiquitous network. **International Journal of Smart Home**, Science and Engineering Research Support Society, v. 10, n. 9, p. 275–286, sep 2016. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.14257/ijsh.2016.10.9.26>>. Citado nas páginas 75 e 77.

YING, P.; GUAN-YU, L.; ZHONG-JUN, L. Community division-based swot ontology directory service. In: **2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)**. [s.n.], 2016. p. 123–127. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/CyberC.2016.33>>. Citado nas páginas 75 e 77.

ZASLAVSKY, A.; PERERA, C.; GEORGAKOPOULOS, D. Sensing as a service and big data. **arXiv preprint arXiv:1301.0159**, 2013. Citado na página 49.

ZHANG, J.; IANNUCCI, B.; HENNESSY, M.; GOPAL, K.; XIAO, S.; KUMAR, S.; PFEFFER, D.; ALJEDIA, B.; REN, Y.; GRISS, M.; ROSENBERG, S.; CAO, J.; ROWE, A. Sensor data as a service – a federated platform for mobile data-centric service development and sharing. In: **Services Computing (SCC), 2013 IEEE International Conference on**. [s.n.], 2013. p. 446–453. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/SCC.2013.34>>. Citado nas páginas 37 e 48.

ZHANG, M.; SUN, F.; CHENG, X. Architecture of internet of things and its key technology integration based-on RFID. In: **2012 Fifth International Symposium on Computational Intelligence and Design**. Institute of Electrical & Electronics Engineers (IEEE), 2012. Acessado em 27 de ago. 2018. Disponível em: <<http://dx.doi.org/10.1109/ISCID.2012.81>>. Citado na página 43.

ZHOU, Y.; DE, S.; WANG, W.; MOESSNER, K. Search techniques for the web of things: A taxonomy and survey. **Sensors**, MDPI AG, v. 16, n. 12, p. 600, apr 2016. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.3390/s16050600>>. Citado na página 70.

ZOU, M.; DIAZ-MONTES, J.; NAGARAJA, K.; RADIA, N.; PARASHAR, M. D3w: Towards self-management of distributed data-driven workflows with qos guarantees. In: **2016 IEEE 9th International Conference on Cloud Computing (CLOUD)**. [s.n.], 2016. p. 148–155. Acessado em 27 de ago. 2018. Disponível em: <<https://doi.org/10.1109/CLOUD.2016.0029>>. Citado na página 69.

O algoritmo *Simple Additive Weighting* (SAW) é um dos mais utilizados para a resolução de problemas MCDA (MEMARIANI; AMINI; ALINEZHAD, 2009; ABDULLAH; ADAWIYAH, 2014). Ele fornece as propriedades aditivas para calcular a pontuação final das opções disponíveis utilizando pesos para a determinação das prioridades de cada critério. Este algoritmo é a base de diversos outros algoritmos MCDA tais como o *Analytic Hierarchy Process* (AHP) e o *Preference Ranking Organization METHod for Enrichment of Evaluations* (PROMETHEE) (MEMARIANI; AMINI; ALINEZHAD, 2009). De acordo com Abdullah e Adawiyah (2014), SAW é utilizado em diversos domínios de aplicação, tais como: gestão de cadeia de abastecimento, problemas de decisões pessoais, gerenciamento de projetos e seleção de localizações.

SAW utiliza uma pontuação de avaliação para classificar cada opção disponível. A pontuação é obtida por meio de um valor de normalização para cada critério, multiplicado pelo seu peso. As opções são classificadas em ordem decrescente, de acordo com sua pontuação final, que corresponde à soma das pontuações individuais de cada critério (AFSHARI; MOJAHED; YUSUFF, 2010). O algoritmo SAW pode ser resumido em três passos (TZENG; HUANG, 2011):

1. Normalizar a matriz de decisão Q , descrita na Equação 2.2, para a matriz Q' de acordo com a Equação A.1 se o critério tem que ser maximizado ou a Equação A.2 se o critério deve ser minimizado

$$q'_{ij} = \frac{q_{ij} - q_j^{\min}}{q_j^{\max} - q_j^{\min}} \quad \text{para maximizar um critério} \quad (\text{A.1})$$

$$q'_{ij} = \frac{q_j^{\max} - q_{ij}}{q_j^{\max} - q_j^{\min}} \quad \text{para minimizar um critério} \quad (\text{A.2})$$

2. Calcule o vetor de pontuação ϕ para cada opção disponível. Cada pontuação de q'_i pode ser

calculada utilizando a Equação A.3, em que w_j corresponde ao peso atribuído ao critério j e N representa o número de critérios existentes na matriz de avaliação.

$$\phi(q_i) = \sum_{j=1}^N w_j \cdot q'_{ij} \quad (\text{A.3})$$

3. Classifique as opções q_i em ordem decrescente de acordo com a pontuação $\phi(q'_i)$ para determinar qual opção é mais adequada.

VIKOR

O algoritmo *ViseKriterijumska Optimizacija I Kompromisno Resenje* (VIKOR) utiliza o conceito de *compromise programming* que visa maximizar a função de adição do algoritmo, sujeito a um determinado conjunto de critérios e restrições em um ambiente determinístico (BAL-LESTERO, 2007). Neste contexto, VIKOR visa obter a opção mais satisfatória considerando métricas individuais e coletivas. Este algoritmo vêm sendo amplamente utilizado em diversas áreas tais como seleção de localização, ambientes e análise de dados (HUANG; TZENG; LIU, 2009).

VIKOR introduz um índice de classificação baseado em uma métrica particular referente à proximidade da solução ideal. As alternativas são avaliadas de acordo com os critérios estabelecidos e classificadas de acordo com: i) a distância até o ponto ideal, ii) sua relevância para o grupo e iii) ao impacto com relação às outras soluções. O algoritmo VIKOR pode ser resumido nas seguintes etapas (TZENG; HUANG, 2011), (OPRICOVIC; TZENG, 2004):

1. Determine qual é o melhor e o pior valor de cada critério na matriz de decisão Q . Para um critério a ser maximizado, o melhor e o pior valor são representados respectivamente por q_j^* e q_j^- e são calculados de acordo com as Equações B.1 e B.2:

$$q_j^* = \max_i(q'_{ij}) \quad (\text{B.1})$$

$$q_j^- = \min_i(q'_{ij}) \quad (\text{B.2})$$

2. Calcule as métricas de utilização e de impacto. A métrica de utilização é representada por S_i e é utilizada para representar o desvio médio das opções disponíveis e é calculada de acordo com a Equação B.3, em que w_j corresponde aos pesos dos critérios, expressando sua importância relativa. A medida de impacto é representada por R_i e é utilizada para mostrar o desvio máximo com intuito de melhorar a definição das prioridades e pode ser calculada de acordo com a Equação B.4.

$$S_i = \sum_{j=1}^n \frac{w_j |q_j^* - q_{ij}|}{|q_j^* - q_j^-|} \quad (\text{B.3})$$

$$R_i = \max_j \frac{w_j |q_j^* - q_{ij}|}{|q_j^* - q_j^-|} \quad (\text{B.4})$$

3. Calcule a métrica de utilização pelo grupo representada por Q_i de cada solução. O parâmetro v é utilizado para representar o peso da estratégia da "maioria dos critérios". A Equação B.5 é utilizada para calcular Q_i .

$$Q_i = \frac{v \cdot (S_i - S^*)}{(S^- - S^*)} + \frac{(1 - v) \cdot (R_i - R^*)}{(R^- - R^*)}, \quad (\text{B.5})$$

em que

$$S^* = \min_j S_i \text{ e } S^- = \max_j S_i$$

$$R^* = \min_j R_i \text{ e } R^- = \max_j R_i$$

$$v = 0.5$$

4. Classifique as opções q_i em ordem decrescente de acordo com os valores S_i , R_i and Q_i . O resultado serão três listas diferentes.
5. Proponha como “*compromised solution*”, a alternativa q melhor classificada de acordo com a métrica Q (mínimo) se duas condições forem aceitas:

C1. Vantagem aceitável:

$$q_{i+1} - q_i \geq DQ,$$

em que $DQ = \frac{1}{(N-1)}$; e N corresponde ao número de opções disponíveis

C2. Estabilidade aceitável para o algoritmo:

A alternativa q_i deve ser a melhor classificada por S ou/e R .

Se uma dessas condições não é satisfeita, então um conjunto de “*compromise solutions*” é proposto, o qual consiste de:

- Alternativa q_i e q_{i+1} se a condição C2 não for satisfeita, ou
- Alternativa $q_i, q_{i+1}, \dots, q_n$ se a condição C1 não for satisfeita; e q^n é determinado por meio da relação $q_n - q_i < DQ$, em que n corresponde ao número máximo de opções que correspondem a condição.

COMPARAÇÃO ENTRE A EXECUÇÃO DO ALGORITMO DE PARETO NA MEMÓRIA PRINCIPAL E NO BANCO DE DADOS

Este anexo visa apresentar o impacto da utilização do banco de dados para a execução do algoritmo *Fast Non-Dominated Sort* (FNDS) quando comparado a sua execução na memória principal. Além disso, os cenários cujo as máquinas com os recursos computacionais descritos na tabela 7, foram capazes de executar os experimentos sem a utilização do banco de dados. Cada experimento foi executado dez vezes para reduzir a interferência de fatores estocásticos. É válido destacar que o algoritmo FNDS, computa todas as relações de dominância independentemente do número de recursos a serem selecionados e por isso os resultados apresentados neste anexo não discriminam o número de recursos selecionados.

Na Figura 49 é apresentado o gráfico boxplot comparando a execução do algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados para dois critérios. Ao utilizar o banco de dados para armazenar as relações de dominância, nota-se que o tempo tende a diminuir de forma exponencial, visto que o número de comparações entre os recursos diminui. Por outro lado, observa-se que não há caixas para a execução na memória principal quando 1, 2 e 4 servidores são utilizados, pois a quantidade desta no servidor é insuficiente para armazenar as relações de dominância de todos os recursos. Quando 8 servidores são utilizados, a execução do algoritmo na memória principal apresenta tempo menor do que a execução utilizando o banco de dados, pois não são realizadas operações de entrada e saída.

Na Figura 50 é apresentado o gráfico boxplot comparando a execução do algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados para quatro critérios. Nesta figura, é possível observar que embora mais comparações sejam realizadas com relação à Figura 49, ao executar o algoritmo FNDS usando o banco de dados há uma ligeira redução no tempo de execução. Isso ocorre, pois ao aumentar o número de critérios conflitantes haverá

Figura 49 – Comparação do tempo em segundos necessário para executar o algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados em 1 , 2 , 4 servidores e 8 servidores para dois critérios.

Fonte: Elaborada pelo autor.

um número menor de soluções dominantes, implicando na redução do número de operações de entrada e saída realizadas.

Ainda, ao executar o algoritmo FNDS na memória principal, nota-se que a redução do número de relações de dominância possibilitou a execução utilizando quatro e oito servidores. Conforme o esperado, em ambas configurações o tempo de execução na memória principal foi menor do que o tempo de execução usando banco de dados. Porém, é válido destacar que para o cenário em que oito servidores são considerados (Figura 50.d), houve redução da diferença entre o tempo de execução utilizando a memória principal e o banco de dados.

Na Figura 51 é apresentado o gráfico boxplot comparando a execução do algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados para seis critérios. Analogamente, a Figura 50, a execução do algoritmo FNDS utilizando-se o banco de dados apresentou tempos ligeiramente menores que o cenário anterior, uma vez que o número de soluções dominantes é ainda menor e, por isso, realizando menos operações de entrada e saída. O menor

Figura 50 – Comparação do tempo em segundos necessário para executar o algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados em 1 , 2 , 4 servidores e 8 servidores considerando-se quatro critérios.

Fonte: Elaborada pelo autor.

número de relações dominantes, também permitiu a execução do algoritmo FNDS na memória principal fosse realizada em 2, 4 e 8 servidores, cujo o tempo de execução foi menor do que quando executado com o banco de dados. Porém, ao comparar a diferença de tempo entre os ambientes nota-se que essa diferença é menor do que a obtida na Figura 50.

Em síntese, nota-se que a complexidade computacional do algoritmo FNDS restringe os cenários para a sua execução na memória principal. Por meio dos resultados apresentados neste anexo, nota-se que tal restrição ocorre devido ao número de relações de dominância que devem ser armazenadas. A utilização de um banco de dados para armazenar tais relações de dominância mostrou-se uma solução viável, apesar de apresentar um ligeiro aumento no tempo de execução devido ao aumento do número de operações de entrada e saída a serem realizadas.

Figura 51 – Comparação do tempo em segundos necessário para executar o algoritmo FNDS utilizando-se a memória principal do servidor e um banco de dados em 1 , 2 , 4 servidores e 8 servidores considerando-se seis critérios.

Fonte: Elaborada pelo autor.