UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DANILO REIS DE VASCONCELOS

SMART SHADOW - PREDICTIVE COMPUTING RESOURCES ALLOCATION FOR SMART DEVICES IN THE MIST COMPUTING ENVIRONMENT

FORTALEZA

2018

DANILO REIS DE VASCONCELOS

SMART SHADOW - PREDICTIVE COMPUTING RESOURCES ALLOCATION FOR SMART DEVICES IN THE MIST COMPUTING ENVIRONMENT

Tese apresentada ao Curso de do Programa de Pós-graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Redes

Orientadora: Profa. Dra. Rossana Maria de Castro Andrade
Co-Orientador: Prof. Dr. José Neuman Souza

FORTALEZA

2018

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

V45s Vasconcelos, Danilo Reis de.
Smart Shadow - Predictive computing resources allocation for smart devices in the Mist Computing environment / Danilo Reis de Vasconcelos. – 2019.
150 f. : il.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação , Fortaleza, 2019.
Orientação: Profa. Dra. Rossana Maria de Castro Andrade.
Coorientação: Prof. Dr. José Neuman Souza.

1. Mist Computing. 2. Fog Computing. 3. Internet of Things. 4. Collaborative Recommender Systems. 5. Bio-inspired systems. I. Título.

CDD 005

DANILO REIS DE VASCONCELOS

SMART SHADOW - PREDICTIVE COMPUTING RESOURCES ALLOCATION FOR SMART DEVICES IN THE MIST COMPUTING ENVIRONMENT

Tese apresentada ao Curso de do Programa de Pós-graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Redes

Aprovada em:

BANCA EXAMINADORA

Profa. Dra. Rossana Maria de Castro Andrade (Orientadora)
Universidade Federal do Ceará (UFC)

Prof. Dr. José Neuman Souza (Co-Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Francisco Luís Rocha Pimentel
Universidade Federal do Ceará (UFC)

Prof. Dr. Miguel Franklin de Castro
Universidade Federal do Ceará (UFC)

Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará (UFC)

Carlos Andre Guimaraes Ferraz
Universidade Federal de Pernambuco (UFPE)

Lisandro Zambenedetti Granville
Universidade Federal do Rio Grande do Sul (UFRGS)

I dedicate this thesis to my daughter Isabelle who inspires me to be a better human being.

ACKNOWLEDGEMENTS

Thank God for giving me strength, health, and determination throughout the entire Ph.D. journey.

I would like to thank my parents, Octávio and Aila for all your love and unconditional support. I acknowledge and thank immensely all effort and immeasurable sacrifice you have made to allow me to be here.

I would also like to thank my sweet wife, Jo and my daughter Isabelle. For all the support given during this long journey, in which several times I had to divide the time dedicated to my family with academic papers and research related to my doctorate.

Thanks to my academic and life advisor, Prof. Rossana Andrade, and Prof. José Neuman. It was an honor to work with you. I learned a lot over the years. Thank you very much for all the encouragement, support, teachings, and opportunities. Having you by my side made all the difference.

I want to give thanks to Prof. Luis Rocha Pimentel, Prof. João Pordeus, and Prof. Eurico Vasconcelos for generously sharing your time and knowledge.

Thanks to all my friends from the GREat lab in special Valdenir, Ronaldo and Ismayle and also my old friends José Deodoro and Chris Adams for helping me to review some articles during the Ph.D. journey. Thanks for the welcome, the fun times and the support in the difficult moments. You are great!

“The only source of knowledge is experience.”

(Albert Einstein)

ABSTRACT

The Internet of Things (IoT) is a technological revolution that has generated new opportunities in academia and industry. In this context, IoT enables the emergence of several new ecosystems and computing environments. One of these new environments that, in the view of some authors, is considered of high importance in the context of the IoT devices is Fog and Mist Computing (FMC). FMC uses computational resources located at the edge of the network, reducing the latency and bandwidth problems, when compared to the use of Cloud computing platforms focused on IoT applications, also called Cloud of Things (CoT). Both infrastructures Fog and Mist computing are located on the edge of the network, however, the Fog computing processing usually occurs at the gateway layer that connects the IoT devices with the Internet. On the other hand, Mist computing, although it is a subset of Fog computing, concentrates its processing in the direct neighborhood of the device. The FMC environment offers new opportunities and benefits, however, due to the considerable dynamism of the topology and heterogeneity of devices, new challenges also arise. This thesis focuses on the problem of how to handle with this dynamism considering the issue of predictive discovery of computational resources in this environment and, thus, proposing a predictive model based on collective knowledge of previous experiences of resource allocations used by IoT devices in this ecosystem. In the proposal, the problem is subdivided into three distinct sub-problems, as follows. The first is how to evaluate from the client device perspective if it is interesting to use the infrastructure of the Fog/Mist/Cloud computing. Subsequently, once the answer is positive for Fog or Mist computing, the work seeks to find mechanisms on how to maintain data in this highly dynamic environment of the network topology. To address this issue, the work proposes a bio-inspired self-adaptive hierarchy structure of devices that use epidemic models to address this problem. Finally, the work presents a prediction algorithm of resources based on collaborative filters combined with an estimator of temporal availability of the devices that are part of the FMC environment. The evaluation is done with simulation using the Contiki operating system and the simulator Cooja. The results suggest the effectiveness of the proposal, even in cases where the FMC environment is composed of few devices that follow a pattern of permanence behavior within the network.

Keywords: Mist computing. Fog computing. Internet of Things. Collaborative Recommender systems. Bio-inspired systems.

RESUMO

A Internet das Coisas (*Internet of Things* – IoT) é uma revolução tecnológica que gerou novas oportunidades na academia e na indústria. Neste contexto, também emergem diversos novos ecossistemas e ambientes computacionais. Um desses novos ambientes que, na visão de alguns autores, é considerado de grande importância no contexto dos dispositivos IoT é a Fog e a Mist Computing (FMC). A FMC utiliza recursos computacionais localizados na borda da rede, reduzindo a latência e os problemas de largura de banda, quando comparada com o uso de plataformas de Computação em Nuvem focadas em aplicações IoT, também chamadas *Cloud of Things* (CoT). Tanto a Fog como a Mist estão localizadas na borda da rede, no entanto, na Fog computing o processamento ocorre geralmente na camada de gateways que conecta os dispositvos IoT com a Internet. Por outro lado, a Mist computing, embora seja um subconjunto da Fog computing, concentra o seu processamento na vizinhança direta do dispositivo. Embora este ambiente de FMC ofereça novas oportunidades e benefícios, devido ao grande dinamismo da topologia e heterogeneidade dos dispositivos, também surgem novos desafios. Essa tese foca no problema de como lidar com esse dinamismo considerando o problema de descoberta preditiva de recursos computacionais neste ambiente, e propondo, assim, um modelo preditivo baseado no conhecimento coletivo de experiências anteriores de alocações de recursos computacionais utilizados por dispositivos IoT nesse ecossistema. No trabalho, o problema é subdividido em três subproblemas distintos, descritos a seguir. O primeiro é como avaliar, do ponto de vista do dispositivo cliente, se é interessante utilizar a infraestrutura da Fog/Mist/Cloud computing. Posteriormente, considerando que a resposta seja positiva para Fog ou Mist, a proposta busca encontrar mecanismos de como manter dados nesse ambiente de alta dinamicidade da topologia de rede, propondo uma estrutura hierárquica auto adaptativa e bio-inspirada nos dispositivos e utilizando modelos epidêmicos para tratar essa questão. Finalmente, o trabalho propõe um algoritmo de predição de recursos computacionais distribuído baseado em filtros colaborativos, combinado com um estimador de disponibilidade temporal dos dispositivos que compõem o ambiente FMC. A validação do trabalho foi feita por meio de simulação, utilizando o sistema operacional Contiki e simulador Cooja, e os resultados sugerem a efetividade das propostas mesmo em casos cujo ambiente FMC seja muito dinâmico.

Palavras-chave: Mist computing. Fog computing. Internet das Coisas. Sistemas de Recomendação colaborativos. Sistemas Bio-inspirados.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ALS	Alternating Least Squares
BIS	Bio-inspired systems
CB	Content-Based
CF	Collaborative Filters
CI	Collective Intelligence
CoT	Cloud of Things
CPS	Cyber-Physical System
CPU	Central Processing Unit
CRUC	Cold-start Recommendations Using Collaborative Filtering
CSMA/CA	Carrier-Sense Multiple Access with Collision Avoidance
DPPF	Data Persistence Problem in the FMC environment
EC	Edge Computing
EMX	Expectation-Maximization
ETSI	European Telecommunications Standards Institute
FAN	Field Area Network
FC	Fog Computing
FCN	Fog Computing Nodes
FLL	Far away from Local Leaders
FMC	Fog/Mist computing
IaaS	Infrastructure as a Service
IoT	Internet of Things
IoTSRS	IoT Service Recommendation
IPFS	InterPlanetary File System
IT	Information Technology
KNN	K Nearest Neighbor
LIS	Latent Indexing Semantics
LL	Local Leaders
LLN	Local Leaders' Neighbors
LRCR	Local Resource Consumption Rate
MAGC	Mobile Ad hoc Grid Computing

MC	Mist Computing
MEC	Mobile Edge Computing
NASA	National Aeronautics Space Agency
NFC	Near-Field Communication
NIST	National Institute of Standards and Technology's
P2P	Peer To Peer
PaaS	Platform as a Service
PAR	Problem of Allocation of Resources
PC	Personal Computers
PCC	Pearson Correlation Coefficient
PNN	Probabilistic Neural Networks
RAM	Random Access Memory
RAN	Radio Access Networks
RMSE	Root Mean Square Error
RQ1	Research Question 1
RQ2	Research Question 2
RQ3	Research Question 3
RS	Recommendation Systems
RSU	Road Site Units
SDK	Software Development Kits
SDS	Software Defined Storage
SIR	Susceptible/Infective/Removed
SMS	Short Message Service
SV-VSN	Vehicular Social Networks
SVD	Singular Value Decomposition
TF-IDF	Term Frequency/Inverse Document Frequency
TOR	Traffic Offloading Rate
VANET	Vehicle Ad-hoc Networks
VM	Virtual Machine
VSM	Value Stream Mapping
WLAN	Wireless Area Network
WSN	Wireless Sensor Networks

CONTENTS

1 INTRODUCTION

This work proposes algorithms and protocols for predictive computing resource allocation in the Fog/Mist computing (FMC) environment using machine learning techniques and bio-inspired systems concepts.

The current Chapter introduces this thesis as follows. Section 1.1 describes the research context and the motivation of this work as well as the problem addressed. Next, Section 1.2 introduces the thesis goals and main contributions. Section 1.3 presents the hypothesis and research questions. After that, Section 1.4 presents the research methodology followed during this thesis work. Finally, Section 1.5 presents the organization of this thesis.

1.1 Contextualization and Motivation

The Internet of Things (IoT) is a revolution that has presented itself as the future of the internet, as it enables connectivity with a global scope reaching almost every object of our everyday world. (AAZAM; HUH, 2016) (YI *et al.*, 2015a). Thus, IoT incorporates intelligence and connectivity into virtually every object that is part of the people environment and creates a large global network in which people, software, services and ”things” interact among themselves.

In the context of IoT, ”things” are defined as the intelligent devices that have digital and physical entities and perform some task for humans or for the environment of which they are part (AAZAM; HUH, 2016).

In this new ecosystem, smart devices can interact, collaborate, and obtain computing resources with other peer devices in their neighborhood or large cloud-hosted server infrastructures.

Since many IoT smart devices have serious battery, processing, or storage restrictions, a common approach to IoT application development has been to connect these devices to servers hosted on a powerful cloud computing infrastructure (YI *et al.*, 2015b). In this model, it is common for large cloud computing providers such as Amazon AWS[1], Microsoft Azure[2], IBM Bluemix[3], and Google[4] to commercially explore the model Platform as a Service (PaaS) (PFLANZNER; KERTESZ, 2016).

[1] https://aws.amazon.com/iot/
[2] https://azure.microsoft.com/pt-br/services/iot-hub/
[3] https://console.bluemix.net/catalog/services/internet-of-things-platform
[4] https://cloud.google.com/solutions/iot/

IoT *PaaS* usually gets, from Infrastructure as a Service (IaaS), computational processing and storage resources in an elastic way, implements several protocols used in IoT, provides automatic data analysis tools, data visualization tools, integration with other services, and Software Development Kits (SDK) to allow connection between the main hardware platforms used in IoT with the *PaaS* supplier (BOTTA *et al.*, 2016).

Thus, the adoption of IoT PaaS in the IoT application development process abstracts the developer from many of the inherent challenges of the IoT environment, thus, reducing risks, costs and efforts, as well as simplifying the complexities involved in the development of an IoT application.

Although the use of IoT *PaaS* in a large number of IoT applications has proved to be a good and cost-effective solution, there are many IoT applications where the simple introduction of latency, inherent in cloud computing, makes it impossible to meet design requirements. Examples of such applications are real-time streaming, real-time gaming, augmented reality, and real-time applications on connected cars (YI *et al.*, 2015b).

Another relevant issue that should be taken into account when adopting *PaaS* is that their use is normally associated with costs of subscription rates, volumes of data trafficked, processed and stored.

A proposed solution for these questions, suggested by (BONOMI *et al.*, 2012), is *Fog Computing*. In their proposal, computational resources are used at the edge of the network, thus reducing latency and bandwidth problems. Hence, this approach enables numerous opportunities and new applications. Among the leading applications, according to (YI *et al.*, 2015a), the following can be cited: Augmented Reality and real-time video analytics; Mobile Big Data; Content Delivery and Caching; and others. However, Fog computing also brings several new challenges that, according to Dasgupta (DASGUPTA; GILL, 2017) in their systematic review, the Fog computing challenges can be broadly classified into four main categories:

- Security - in the environment of Fog computing, it is common to have devices with limited processing power and battery restrictions. Thus, the use of public key cryptography becomes inadequate requiring new approaches to ensure a secure environment.
- Data governance - according to IDC FutureScape (MACGILLIVRAY *et al.*, 2017), the data volume expected by IoT devices by 2020 will be about 44 Zettabytes. Given that huge data volume generated by IoT devices, a considerable volume of these data needs to be processed and stored locally, avoiding network transfer bottlenecks and excessive

storage costs. However, this creates challenges and related research gaps on how the data owner can monitor and control his data, ensuring confidentiality and data integrity.

- Device Management - the devices that compose the Fog computing infrastructure are heterogeneous with different processing power, storage capability, and network interfaces. Thus, configuring these devices is an extremely complex task, the open Fog consortium recommends the use of machine learning to address these issue.
- Operational, Technology and methods - Although there are several studies of issues associated with Fog computing and IoT, there is a gap in the literature when the issues are the implementation of IoT applications in the environment of Fog computing. Thus, one relevant challenge is the fog enable IoT implementation of application development support frameworks.

An extension of the idea of *Fog Computing* is Mist computing, in which the responsibility of running computational resources such as middleware and services goes to devices in the direct neighborhood of the requesting client device that performs these tasks collaboratively and in a distributed way (PREDEN *et al.*, 2015)

The research proposed in this thesis focuses mainly on issues related to the third category of challenges addressing the problem of ”Device Management” in the Fog and/or Mist (FMC) computing IoT environment with machine learning techniques and collective intelligence.

So, in the context of IoT, the FMC environment provides a series of benefits to intelligent devices, making it possible to provide a low latency, high speed, robust, decentralized and usually their use. However, conventional mechanisms for discovering and allocating resources in this environment may not work properly, due to the high dynamicity of the network topology, heterogeneity of devices and communication links, and constant substitution of devices that are part of the network infrastructure.

On the other hand, in the view of the advances in the technological development of the last decades, it is common to increasingly come across complex systems, which usually involve several types of entities that interact locally and externally. One relatively common solution approach adopted in this class of problems has been the use of collective intelligence techniques.

Collective intelligence is an area of evolutionary computing, in which the general concept is based on the idea that individuals (people, machines, robots, software agents) acting as a group can be smarter than an individual acting in isolation from the (HEYLIGHEN, 1999)

group. In other words, there is a synergy associated with the interactions of individuals in the search for the best solutions to problems. Examples of applications of collective intelligence algorithms are:

- Systems of recommendations in e-commerce such as Amazon[5], submarine, Alibaba Express[6];
- Navigation systems that consider traffic such as Waze[7] and Google maps[8]; and
- Tourist recommendation systems such as Trip Advisor[9] and Open Table[10] that are specialists in restaurants.

In all these systems, the application domain consists of a complex environment and difficult mathematical modeling. In the collective intelligence approach, information from a collectivity of individuals is collected, processed and classified to generate information that helps to optimize the experience of other users in different context. For example, in the case of e-commerce quoted before, the product recommendation system seeks to maximize user satisfaction by suggesting products they have some interest in. The system bases these suggestions on purchases from other users who have a profile similar to the target user.

In the case of navigation systems, the algorithm suggests routes in which the user minimizes the time of travel to avoid congestion. The system calculates these routes using transit information (average speed of the route, amount of information, etc.) provided by other users of the system who collaborate with their information to obtain a global view of the traffic in the region.

In the case of tourism recommendation systems, the goal is to maximize the tourist experience by suggesting attractions, restaurants, and hotels that have been well evaluated by other tourists on previous visits.

Similarly, the FMC environment is a complex environment, difficult to model in many application contexts. In this way, it is possible to propose methods and algorithms of collective intelligence in the context of FMC applied to the predictive discovery of computational resources.

For example, when the scenario of a highway with smart cars sharing computing resources with each other is considered, it is reasonable to suppose that, on any given weekday,

[5] https://www.amazon.com/
[6] https://www.aliexpress.com/
[7] www.waze.com
[8] https://www.google.com/maps/
[9] https://www.tripadvisor.com
[10] https://www.opentable.com/

a considerable proportion of vehicles on the highway will be the same ones that will be on the same highway at similar times and days of the week. This phenomenon happens due to the characteristic of routines of human activities like working, studying, practicing sports or even having fun that in general follow patterns of schedules and days of the week. Thus, imagine how useful it would be for the task of discovering the availability of computational resources if the cars had the capacity to learn these routines, and therefore instead of executing some protocol of discovery of computational resources, the vehicle already knows where are the vehicles at that time and day would have the target computational resource available. Certainly, the data traffic on the network and time for resource availability would be reduced and allow better optimization.

1.2 Research Goal, Expected Contributions, and Scope

Taking in consideration the issues tackled in Section 1.1, the main goal of this thesis is then to propose a robust and efficient mechanism for the discovery and predictive allocation of computational resources for IoT devices in the FMC environment.Thus, the proposal of this thesis goes towards the construction of mechanisms that allow the use of the FMC environment as IaaS. In order to achieve this, the following activities should be done:

- Elaborate a literature review of the state of the art related to predictive methods of resource discovery in the FMC environment;
- Identify the main research gap in the context of the predictive methods of resource discovery in the FMC environment;
- Elaborate solutions to issues related to the identified gap; and
- Validate the proposed solutions in terms of efficiency and robustness.

At the end, the main expected contributions of this work are summarized as follows:

- A systematic mechanism for discovering computational resources in the FMC environment that meets constraints imposed on these computational resources;
- A bio-inspired adaptive mechanism of data dissemination within FMC networks with high dynamicity in topology; and
- A mechanism and prediction model of temporal availability of computational resources in network FMC with decentralized adaptive architecture based on collaborative filters.

In this work, the efficiency is considered as the index of correctness between the availability forecast and the real availability of the computational resource in the FMC environment. The evaluation of the robustness is based on the ability of the method to support the

percentage of exchange of the devices that are part of infrastructure and continues generating acceptable predictions.

It is also important to mention that is out of the scope of this thesis to address security issues as well as data governance, and power consumption issues of devices when they are using the proposed mechanisms.

1.3 Hypothesis and Research Questions

The thesis research seeks to validate the following hypothesis: *It is possible to make a predictive discovery and allocation of the computational resources to smart devices in the context of Fog and Mist Computing using data locally stored, in an efficient and robust manner*.

From this hypothesis, the following Research Questions (RQ) are extracted:

- **Research Question 1 (RQ1)**: *How can a device systematically evaluate the feasibility of using computational resources available in the FMC computational infrastructure?* Many IoT applications typically use the Cloud of Things (CoT) as a base IT infrastructure for computing resources (processing, storage, services). However, it is not always possible to meet the application's communication latency or bandwidth requirements, because of the intrinsic latencies of Cloud Computing. Thus, one of the first questions that the device must address when it needs external computing resources is what infrastructure (Fog / Mist / Cloud computing) should use.
- **Research Question 2 (RQ2)**: *How to store information in an FMC environment, considering the ephemerality of device availability in this environment?* In Bonomi's original proposal of Fog computing, the use of computing resources at the edge of the network usually resulted from the use of the computational infrastructure that connected the "things" to the cloud computing environment, usually called the *gateway*. However, the recent idea of the Mist computing creates the possibility of using any device that has a computational resource located in the vicinity of the client device. Thus, the permanence or availability of these devices within this environment is often ephemeral, so, the use of these resources requires new strategies and approaches. In this context, this research question seeks to explore how to use, in particular, the small volume data storage resource considering this complexity imposed by the FMC environment. This data can be context data, information shared by other devices, or specific to an application. In particular, in this thesis, the data that will be shared are the configuration parameters of the predictors of computational

resources.

- **Research Question 3 (RQ3)**: *What would be a way to automatically capture knowledge of the discovery and use of computational resources in the Fog / Mist computing environment to make discovery and allocation predictions efficiently and robustly?* The FMC environment can be characterized by the great dynamics of its network topologies and ephemerality of the devices that comprise it. This characteristic imposes several new challenges to methods of discovery and allocation of computational resources. Considering that in many human environments the existence of routines is common, the idea of extracting knowledge from previous experiences can be a good way of getting around the challenges imposed by the environment.

  To make the understanding of RQ3 easy, an analogy with human experiences can be done as follows: suppose a person is in a city s/he does not know and wants to eat at an excellent Italian restaurant nearby. One way to find this restaurant would be to do a search in the region analyzing the atmosphere and menu of each restaurant found, thus, choosing what best suits your taste. This method, although it is a rational way of choice, requires that the person defines a geographic region where s/he will do the search and that s/he systematically spends time in this task until s/he finds the chosen restaurant. Note that not necessarily s/he will be able to find, and even finding an Italian restaurant in the region, there is a possibility that s/he does not like the food. Another more current and probably more efficient approach is that before the person goes out in search of the Italian restaurant, s/he consults TripAdvisor and, based on the choice of other users and their personal preferences, makes her/his choice, addressing thus directly to the chosen restaurant and therefore reducing the time of the restaurant search. Although there is still a risk that s/he does not like the food, this way s/he minimizes such risk, since there are previous experiences of others who have already evaluated the service of the restaurant. Therefore, the person has at least reduced the search time, since her/his choice used the previous experiences of other individuals.

  Analogously, the study of this last research question seeks to identify methods that allow the application of this concept of recommender systems in the context of intelligent devices in FMC environments.

1.4 Research Methodology

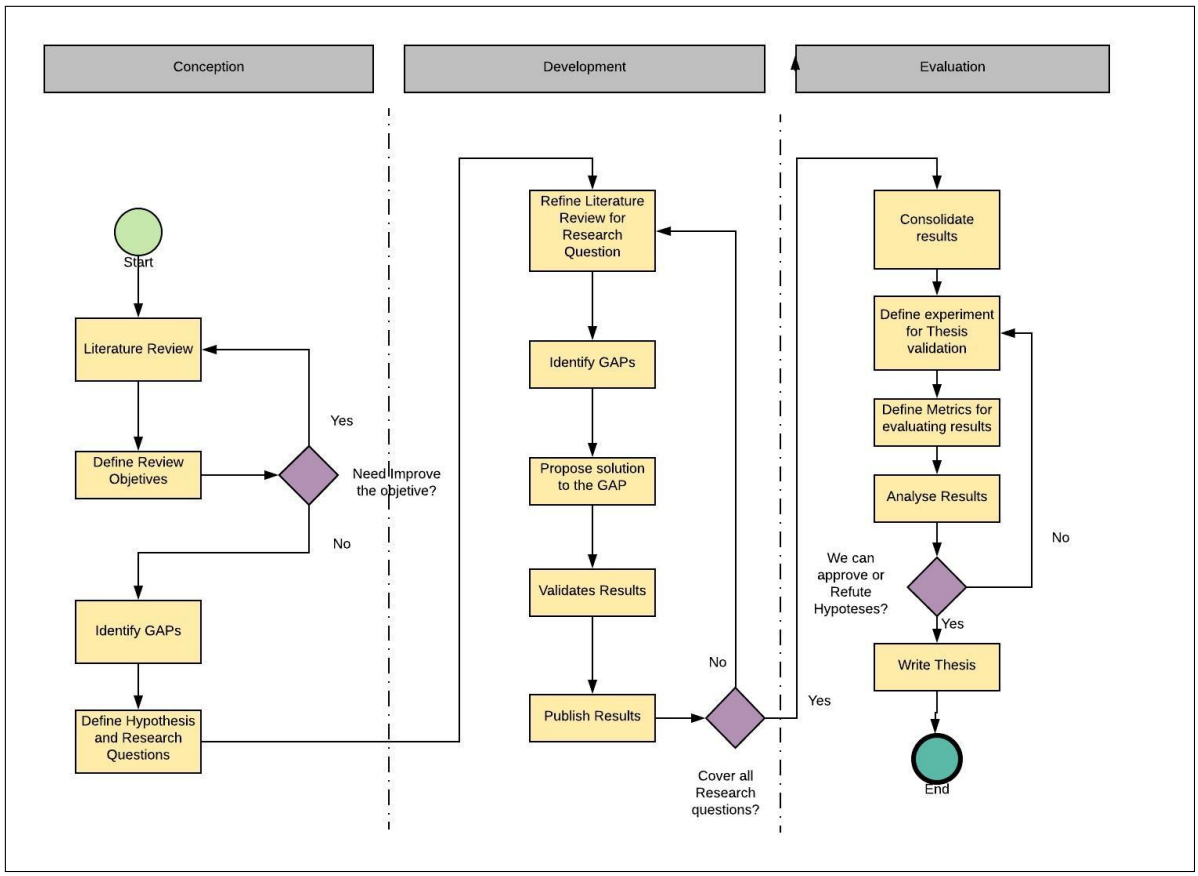
Based on the objectives presented in Section 1.2 and 1.3, the research methodology of this thesis is defined. In a nutshell, this research is organized into three main phases, and each phase includes a set of activities, as depicted in Figure 1. These phases are:

(i) *Conception*, in which the research problem and questions based on the literature review are refined;

(ii) *Development*, in which a solution is proposed to the problem addressed; and

(iii) *Evaluation*, in which the developed solution is evaluated.

Figure 1 – Research Methodology



Source – the author.

For the purpose of building the *conception* phase, four main activities are listed as follows:

- The first activity is the `Literature Review`, which comprises the search for papers related to the mechanism for the discovery and predictive allocation of computational resources for IoT devices in the FMC environment. Based on this literature review, it is an

objective to compile the main research papers related to the proposed thesis and identify the related research questions. The research, a non systematic review was performed using online databases, such as Scopus[11], IEEE Xplorer[12] and ACM DL[13]. The Mendeley tool also was used to organize and catalog the selected scientific papers in the research. It is worth noting that this search, as presented in Chapter 3, involves the thesis' hypothesis as well as the related research questions.

- Define and review objectives - during the literature review process, the objectives should be reviewed for checking their feasibility;
- To identify research gaps in related research themes; and
- To define hypothesis and the research questions that means to divide the problem into smaller sub-problems for the proposed thesis.

Aiming to implement the *development* phase, this phase is divided into five main activities listed as follows:

- In the first place, the literature review needs to be refined focused on each defined research questions aiming to identify the specifics issues related to research question, challenges and opportunities, related works, existing state of the art of the proposal solutions, and reference works.
- To identify the gaps related to research question.
- To propose a solution for at least one of the found gaps for each research question solutions. This activity is subdivided into two subactivities:
  - Elaborate a model of the identified problem of the mathematical format and present the premises and limits of the model;and
  - Propose algorithm or adaptation of existing algorithms to solve the problem addressed.
- To validate results from proposed solution using simulation experiments in controlled situations. Analyze and compare the obtained results whenever possible with results of similar work by identifying the strengths, weaknesses and major contributions of the proposal.
- Publish the results in congresses/journals specialized in the area of the work to validate with the academic community.

[11] http://scopus.com/
[12] http: // ieeexplore.ieee.org
[13] http://dl.acm.org/

It is important to realize that for each research question all described activities are performed.The development phase will be finalized only when all research questions have been covered.

In order to implement the *evaluation* phase, this phase was divided into five main activities listed as follows:

- The first activity is to consolidate the results obtained with the research questions and to organize them in order to validate the hypothesis of the thesis;
- To define experiments that allow evaluating the hypothesis of this thesis.
- After this, to define metrics that allow evaluating the results of the proposal considering the hypothesis of the thesis raised;
- In order to prove or refute the hypothesis, to design an experiment that uses the assumptions of the hypothesis and defined metrics to generate experimental results that allow to analyze the hypothesis raised in the thesis; and
- Finally, based on the results of the experiment and analysis of the metrics obtained to validate or refute the hypothesis raised.

1.5 Structure of the Thesis

This Chapter introduced this thesis by describing the motivation and goals of this work, and the research questions that the results aims to answer. Also, this chapter described the research hypothesis and presented the research methodology.

Besides this Introduction Chapter, this thesis is organized in five more chapters, as follows:

- **Chapter 2 (Background)** outlines the mains concepts related to this thesis proposal: Edge computing, Fog computing, Mist computing, epidemic models, collective intelligence, recommender systems, and collaborative filters. Moreover, it also describes the formalism used throughout this thesis.
- **Chapter 3 (Related Work)** compares the proposal of this thesis with studies found in the literature addressing the three research questions raised in this thesis. The Chapter also compares the differentials of the proposals presented in this thesis with similar works.
- **Chapter 4 (Smart Shadow)** presents in details the mechanisms and algorithms proposed in this thesis, related to some of identified gaps in the three research questions. The Chapter also elaborates a mathematical model for related problems.

- **Chapter 5 (Evaluation)** describes the assessments of the method proposed through experiments focused on to validate the proposals presented in this thesis. The proposed experiments were performed through simulation in the simulator Cooja, which is part of the Contiki operating system widely used in IoT and WSN.
- **Chapter 6 (Conclusion)** summarizes the achieved contributions and discusses some future research directions.

2 BACKGROUND

In this chapter, the main concepts and theoretical foundations required to understand the issues related to the hypothesis of the proposed thesis are presented. Aiming at organizing area-related topics, the chapter is divided into two Sections, Network infrastructure technologies, and bio-inspired systems focused on collective intelligence systems.

In the first section, the main concepts, questions, theoretical foundations, and technologies related to Edge Computing (EC) are presented. In Section 2.2, a brief introduction about the area of Bio-inspired systems (BIS) is given. Finally, Section 2.3 concludes this chapter.

2.1 Network infrastructure technologies

2.1.1 Cloud Computing

According to the official National Institute of Standards and Technology's (NIST) definition, "cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" (MELL *et al.*, 2009).

Cloud computing brings a number of benefits to users, such as they gain scale in costs and the use of the model of computing as a service allows them only to pay for the resources they use. Therefore, there is a reduction in large investments in Information Technology (IT) infrastructure (ARMBRUST *et al.*, 2010).

2.1.2 Edge Computing

A relatively common phenomenon in several areas of human activities is the struggle between centralization and decentralization (LOPEZ *et al.*, 2015).

Particularly in the area of computing, this phenomenon can also be observed when analyzed the history of the computer ages. In the early 1960s, the first computers were centralized by being characterized within the Mainframe era. In fact, this era was predominant until the 1980's when the first Personal Computers (PC) that emerged the decentralized computing by popularizing the use of computers for all people. Again, in the 1990's with the advent of the Internet, this trend again reversed with the emergence of Cloud Computing, which concentrated

on computing in large Data Centers.

With Internet of Things that virtually embedded computers in every object of our daily life, a new, globally distributed computing infrastructure was provided (MADAKAM *et al.*, 2015). The IoT devices have an embedded computer and their own Internet connection link and usually generate huge amounts of data that can create network bottlenecks if used with Cloud Computing (DOLUI; DATTA, 2017a).

After all, a new wave of computing decentralization called Edge Computing (EC) arises. The EC uses the computational resources of the IoT devices and the network infrastructure located in the vicinity of the user, creating then an intermediate layer between the end devices and Cloud Computing. In Bilal's (BILAL *et al.*, 2018) view, technologies related to EC are still in their infancy, without any definition of standards, architectures, and protocols yet, similarly the Cloud computing before its standardization.

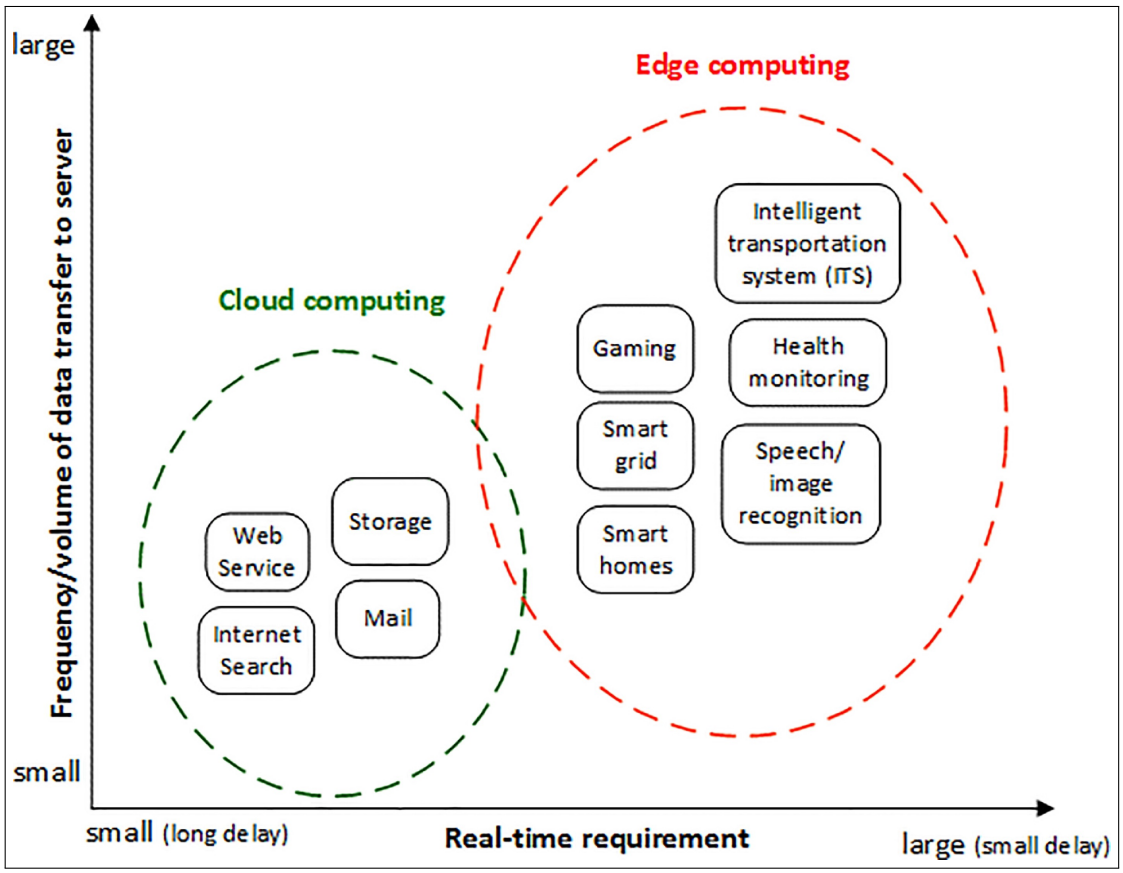
The edge computing technologies should not be considered as a substitute for the cloud paradigm, rather, as shown in Figure 2, these technologies will complement the cloud and extend cloud services to the edges. Hence, EC complements Cloud Computing primarily in applications where latency is critical or if the generated data volume is too large, making it unfeasible for its transfer over the internet.

According to Garcia (LOPEZ *et al.*, 2015), EC encompasses the following elements:

- Proximity is on the edge: In general, the data communications between neighbors nodes of a network is more efficient than when compared with nodes with intermediaries between them.
- Intelligence is on the edge: The EC prioritizes connections to locally close of the network nodes. Thus, the processing power provided by the EC comes from the Central Processing Unit (CPU) embedded in the IoT devices and component of the network infrastructure.
- Trust is on the edge: In general, sensitive personal and social data is generated and stored at the edge of the network. Therefore, it is natural that trust management is also carried out at the edge of the network.
- Control is on the edge: In EC context, the edge nodes performs all coordination and management of applications.
- Humans are on the edge: Human-centered designs should put humans on the control loop, so that the EC allows users to retake control of their information.

Despite the existence of a gap in the literature concerning strategies for implementing

Figure 2 – Cloud Computing versus Edge Computing application areas.



Source – Computer Networks (BILAL *et al.*, 2018)

the EC, according to (DOLUI; DATTA, 2017a), there are basically three EC implementation paradigms:

1. Fog Computing: The Fog Computing implementation is a decentralized Computing infrastructure based on Fog Computing Nodes (FCN) placed at any point of the architecture between the end devices and the cloud. In Section 2.1.3, it will be described this subject in details.
2. Mobile Edge Computing (MEC): MEC is the edge technology initiated by European Telecommunications Standards Institute (ETSI) (BILAL *et al.*, 2018) and it is one of the EC implementations that brings computing and storage to Radio Access Networks (RAN). This form of EC according to (HU *et al.*, 2015) is a key technology for the new 5G networks having a considerable support from the large telephony companies for its deployment. Figure 3 gives an overview of the MEC architecture.
3. Cloudlet Computing: The Cloudlets are like a ”Data center in a box” running a virtual machine, capable of providing computational resources for end-devices and user over a

Table 1 – Comparation of EC architecture implementation.

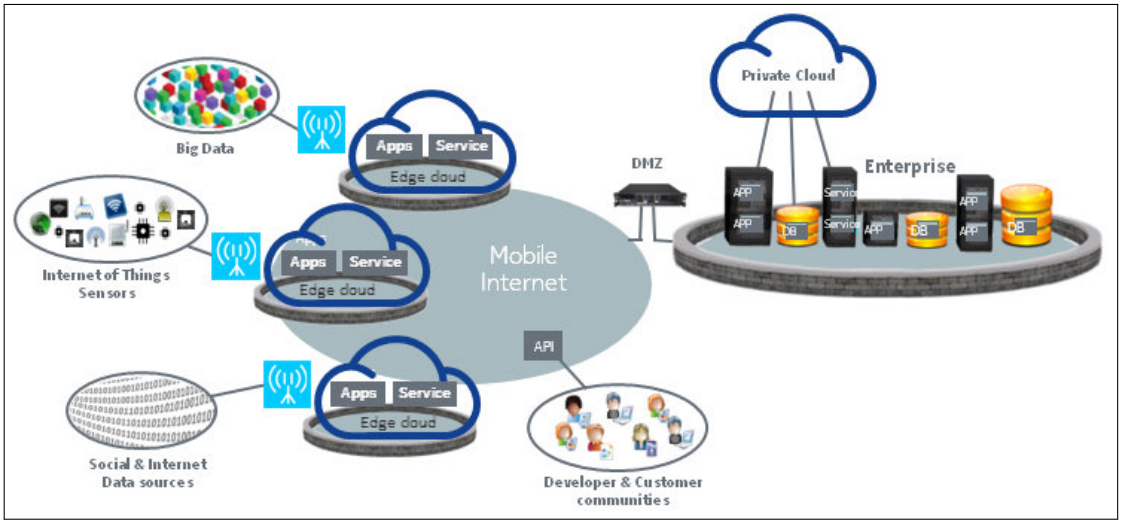
Item	Fog Computing	Mobile Edge Computing	Cloudlet Computing
Node devices	Routers, Switches, Access Points, Gateways	Servers running in base stations	Data Center in a box
Node location	Varying between End Devices and Cloud	Radio Network Controller/Macro Base Station	Local/Outdoor installation
Software Architecture	Fog Abstraction Layer based	Mobile Orchestrator based	Cloudlet Agent based
Context awareness	Medium	High	Low
Proximity	One or Multiple Hops	One Hop	One Hop
Access Mechanisms	Bluetooth, Wi-Fi, Mobile Networks	Mobile Networks	Wi-Fi
Internode Communication	Supported	Partial	Partial

Source – Author.

Wireless Area Network (WLAN) in their neighborhood. Similarly, the other architectures implementations of the Cloudlet computing provide low latency and high bandwidth. Still, it requires host devices with reasonable processing power to run a virtual machine.

In Table 1, a comparison between the technologies of EC implementation architectures, regarding the relevant characteristics of each implementation, is presented.

Figure 3 – Overview of Mobile Edge Computing architecture.



Source – <http://www.accelleran.com/solutions/mobile-edge-computing/>.

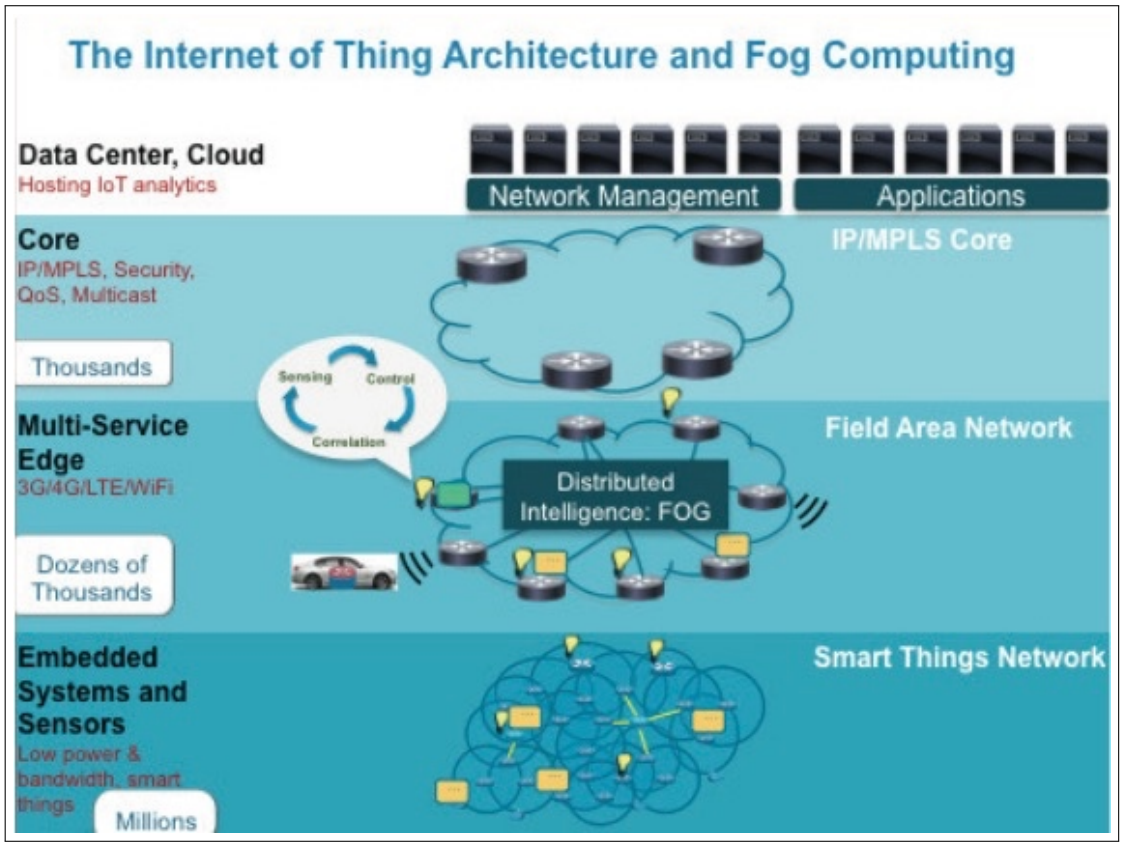
2.1.3 Fog Computing

Proposed by Bonomi (BONOMI *et al.*, 2012), Fog computing is a form of implementation of the EC in which the computational resources are made available to the client using the network layer called the Field Area Network (FAN) that connects the user to the Internet.

FAN is usually composed of routers, switches, embedded server, and gateways that, in the literature, are called Fog Nodes. In general, the Fog Nodes are physically located in the vicinity of the end customer, thus providing high-speed connections with low latency and usually with one Hop of distance (VAQUERO; RODERO-MERINO, 2014).

Figure 4 shows the Fog Computing architecture and the FAN layer in this architecture, in the (BONOMI *et al.*, 2012) vision.

Figure 4 – Fog Computing architecture.



Source – ACM MCC '12
(BONOMI *et al.*, 2012).

According to (YI *et al.*, 2015c), a general definition of Fog Computing would be:"Fog computing is a geographically distributed computing architecture with a resource pool consists of one or more ubiquitously connected heterogeneous devices (including edge devices) at the edge

of network and not exclusively seamlessly backed by cloud services, to collaboratively provide elastic computation, storage and communication (and many other new services and tasks) in isolated environments to a large scale of clients in proximity".

Similarly to other EC implementation architecture technologies, Fog Computing (FC) has as one of its main motivations the possibility of providing fast responses with low network latencies to their client devices.

According to (MADSEN *et al.*, 2013), the following motivations are important for FC utilization: (1)) geographic decentralization of computing resources; (2) the need imposed by IoT to incorporate large networks of sensors communication, usually through wireless access; (3) the requirement to support real-time communication and processing with mobile devices and IoT devices; (4) the need for supporting heterogeneous devices and interoperability with different providers; (5) the requirement of real time analytic and interplay with the Cloud.

The proposal of Fog computing, in general, brings several benefits for the IoT environment. Within these benefits, according to (CHIANG; ZHANG, 2016), the following items can be cited:

- It brings a substantial amount of data to the end user's vicinity, therefore reducing unnecessary data transfer and improving the privacy of the information;
- It also brings various control and processing functions to the edge of the network, reducing network response time; and
- Substantial reduction of communications and data traffic on the network by concentrating communications and data traffic in the vicinity of the user.

In Section 1.1 of Chapter 1, the main challenges of FC, according to (DASGUPTA; GILL, 2017), are presented. For these authors, the challenges can be grouped into four general groups, listed as follows: Security, Data governance, Device Manager, and Operational or Technology and process.

On the other hand, (CHIANG; ZHANG, 2016) group the main challenges of FC in a slightly different way as follows:

- Fog Interfaces With Cloud, Other Fogs, Things, and End Users -This challenge is closely related to RQ1 of this thesis. The author brings about the main question "who does what, at what timescale, and how to put them back together?"
- Fog-Enabled Edge and Access Networking – Considering the scenario where the FC provides EC support by providing service on a local network, it is necessary to provide

temporary security credentials for the Fog' devices to establish reliable communication, transparently acting as local servers. Then, it is also necessary to develop a new protocol stack in the end devices to support these new functionalities.

- Security - In general, distributed systems are more vulnerable than centralized systems. By analyzing the security for the case of FC, this problem becomes more serious due to the great heterogeneity of devices. In some cases, even with the reduced numbers of nodes composing the network, there is no guarantee that the available computational resources are sufficient to protect their devices.
- Intensification of Device Participation - In some cases of using the IoT applications, it is expected from client devices to voluntarily participate by sharing their computing resources (processing, storage, connectivity). However, because of security concerns, they end up not participating. Thus, it is a challenge to create incentive mechanisms for this participation in FC environments.
- Convergence and Consistency Local- - Although it is already a typical challenge of distributed systems to maintain the consistency of local states with the overall state of the system. In FC, this problem becomes even more challenging when considered a massive, under-organized, possibly mobile fog system with diverse capabilities, and potentially virtualized a pool of resources shared unpredictably.
- End-to-End Architectural Tradeoffs Fog - FC enables new possibilities for architecture solution by creating the possibility of combining from fully distributed systems to full centralized system architectures, depending on the context of application. This strategy aims at improving the system resilience through redundancy.

2.1.4 Mist Computing

Introduced by (PREDEN *et al.*, 2015) and (MARTIN, 2015) in 2015, Mist Computing (MC) is a subset of FC that, according to (LIYANAGE *et al.*, 2016), "represents a paradigm in which edge network devices, that have predictable accessibility, provide their computational and communicative resources as services in their vicinity via Device-to-Device communication protocols. Requesters in Mist can distribute software processes to Mist service providers for execution".

In the strict definition of FC (BONOMI *et al.*, 2012), the devices do not participate as providers of computational resources, instead, they act as collectors of data and actuators in

the physical environment (PREDEN *et al.*, 2015). Thus, Fog nodes assume the role of providing computational resources to devices on the edge.

The approach of Mist Computing (MC) is to utilize the computational resources of the IoT devices in the neighborhood physically close to the client device. This proposal decreases the latency and increases the autonomy of the network. On the other hand, MC implementation has even more challenges than FC because of the complexity of device interactions, topology dynamics, and fully decentralized network management (PREDEN *et al.*, 2015).

The MC paradigm is not exactly new, the Mobile Ad hoc Grid Computing (MAGC) research proposed similar ideas for using the high processing power of the ARM CPU of the mobile phones for computational process distribution (LIYANAGE *et al.*, 2016). However, in MAGC, there is a certain degree of device homogeneity when compared to the MC environment in which different types of devices, processing power, storage and wireless network interfaces may increase the MC challenges.

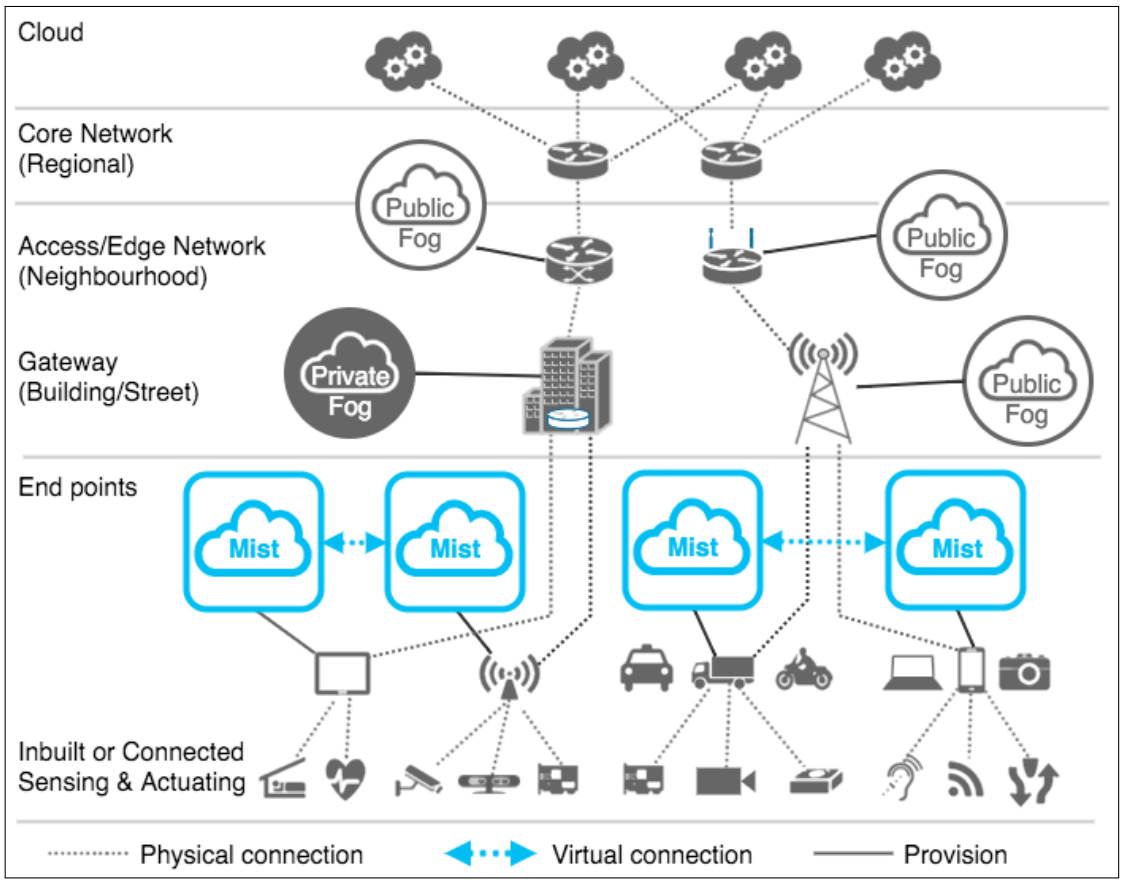
Mist nodes can be of various types such as devices based on Raspberry Pi, Contiki devices, TinyOS devices, Android devices, mobile phone, tablets, smart TVs, handheld entertainment devices, among others.So, Mist is provided by heterogeneous devices at the very edge of IoT network.

Figure 5 presents an overview of the Mist Computing architecture in the Fog computing context.

According to (YOGI *et al.*, 2017), the following principles must guide Mist Computing (MC) architecture:

- MC must preprocess the sensor data and provide a service with meaningful data for the context;
- Services and information must be provided only when requested;
- Network component devices should work collaboratively on the subscriber/publisher model; and
- Devices should constantly adapt to discover who the service providers are and which data they have, avoiding static connections between devices.

Figure 5 – The role of Mist in IoT.



Source – <http://kodu.ut.ee/~chang/mist_computing.html>.

2.2 Bio-inspired Systems

2.2.1 Overview

Analyzing Life from a purely scientific point of view, it is relatively simple to conclude that it is an extremely complex phenomenon.

The National Aeronautics Space Agency (NASA) through its program of exobiology uses the definition of Life proposed by its researchers Horowitz and Miller (1962), and later, followed by Joyce (1994) as (LUISI, 2006): "A self-sustaining chemical system capable of undergoing Darwinian evolution."

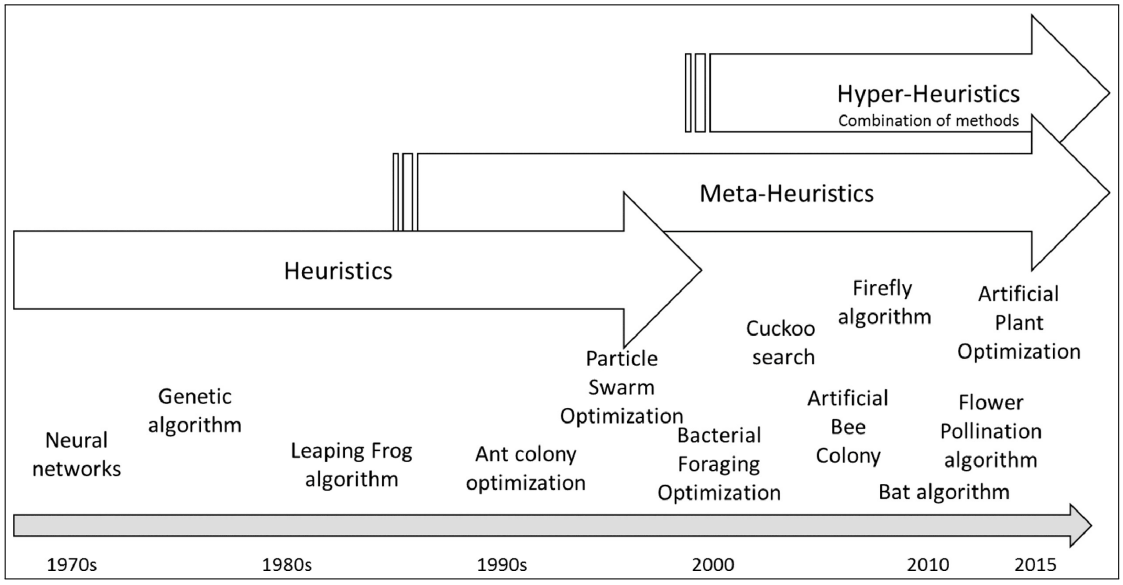
The life forged by some millions of years of evolution allowed the emergence of living organisms. Taking a look at living organisms through an engineering view, it is easy to notice that besides being complex systems, they exhibit several desirable characteristics such as evolution, adaptation and fault tolerance, which are proven to be difficult to realize using traditional engineering methods (MANGE; TOMASSINI, 1998).

The Bio-inspired systems are an interdisciplinary area of science, which study of the functions, characteristics, and phenomena observed in the living beings, in order to apply such knowledge in the conception of new techniques, as well as to creation of new devices and machines. The fundamental idea is: "If nature took millions of years to improve their own mechanisms, why not to copy them?"(SOUZA, 2016).

There are many examples of bio-inspired systems used to solve engineering problems, among which it is possible to mention: velcro inspired by the natural mechanism of seed dissemination; swimsuits for competitions and ship hulls inspired in shark skin to reduce aerodynamic drag; radar and sonar systems inspired by bats and dolphins; designs of wings based on the osseous systems of birds among others.

Similarly, in the area of Computer Science, it is possible to find several applications of bio-inspired concepts in solving problems of optimization, robotics, autonomous systems, image recognition, and classification, among others. As an illustration, it can be highlighted the algorithms and techniques listed below: Neural networks, genetic algorithms, leaping frog algorithm, ant colony optimization, particle swarm optimization, Bacterial foraging optimization, cuckoo search, epidemic models, artificial bee colony, firefly algorithm, Bat algorithm, flower pollination algorithm and artificial plant optimization (KAR, 2016). Figure 6 presents the evolution of these approaches throughout the years.

Figure 6 – The Bio-inspired algorithms and techniques along the years.



Source – Expert Systems With Applications 59 (2016) 20–32 (KAR, 2016)

2.2.2 SIR Epidemic Models

The SIR Epidemic Models or so-called Susceptible/Infective/Removed (SIR) models are mathematical models which were developed to predict the spread of viral or bacterial infections with the person-person transmission mechanism within a population (ANDERSSON; BRITTON, 2012a). In general, these models are reasonably accurate to predict and model the behavior of infections of diseases such as influenza, sexually transmitted diseases (STD), viral infections such as measles, mumps, rubella, among others.

Although the SIR models were developed with the purpose of application to with the purpose of application to the health area, their theoretical basis is robust enough to extend their application to other areas where different problems could be modeled according to the assumptions of the SIR model.

In the early 2000s, Newman (NEWMAN, 2002) argues that epidemiological models can be solved exactly on a wide variety of biological networks, Social networks, and technological networks. Later, Keeling (KEELING; EAMES, 2005) explains the link between SRI and networks by exploring – in different network topologies - the application of SRI models, in order to study important parameters of such networks in different contexts like: propagation of ideas in social networks, evolution and spread of ideas and innovations in societies, and infections in the context of public health.

In general, the SRI epidemiological models can be mathematically modeled in two different ways: deterministic or stochastic (BRITTON, 2010).In this Section it will be provided a brief explanation of the deterministic SRI epidemic models by presenting their premises theoretical conceptualization and limitations. Stochastic SRI epidemic models are not part of the scope of this thesis.

The deterministic general epidemic model proposed by Bailey (BAILEY *et al.*, 1975) presumes the following premises:

(i) The individuals of a population are either susceptible (S), infected and infectious (I) or recovered and immune (R);

(ii) The population is closed, that is, it is assumed that there are no births, deaths, immigration or emigration during the study period;

(iii) Only susceptible(S) individuals can get infected(I) and, after having been infectious for some time, an individual recovers and becomes completely immune(R); and

(iv) The population mixes at random.

Then, considering that $s(t)$, $i(t)$, and $r(t)$ are respectively the percentages of susceptible, infected, and recovery individuals of the population at instant t. Thus, it can be asserted that the sum of all part is one like it is showed at equation 2.1.

$$s(t) + i(t) + r(t) = 1 \tag{2.1}$$

Given that β and γ are respectively the average infection rate per individual and recovery rate, in the proposed model, the differential equations define the infection rates 2.3 and the decrease of susceptible individuals 2.2.

$$\frac{\mathrm{d}s(t)}{\mathrm{d}t} = -\beta * s(t) * i(t) \tag{2.2}$$

$$\frac{\mathrm{d}i(t)}{\mathrm{d}t} = \beta * s(t) * i(t) - \gamma * i(t) \tag{2.3}$$

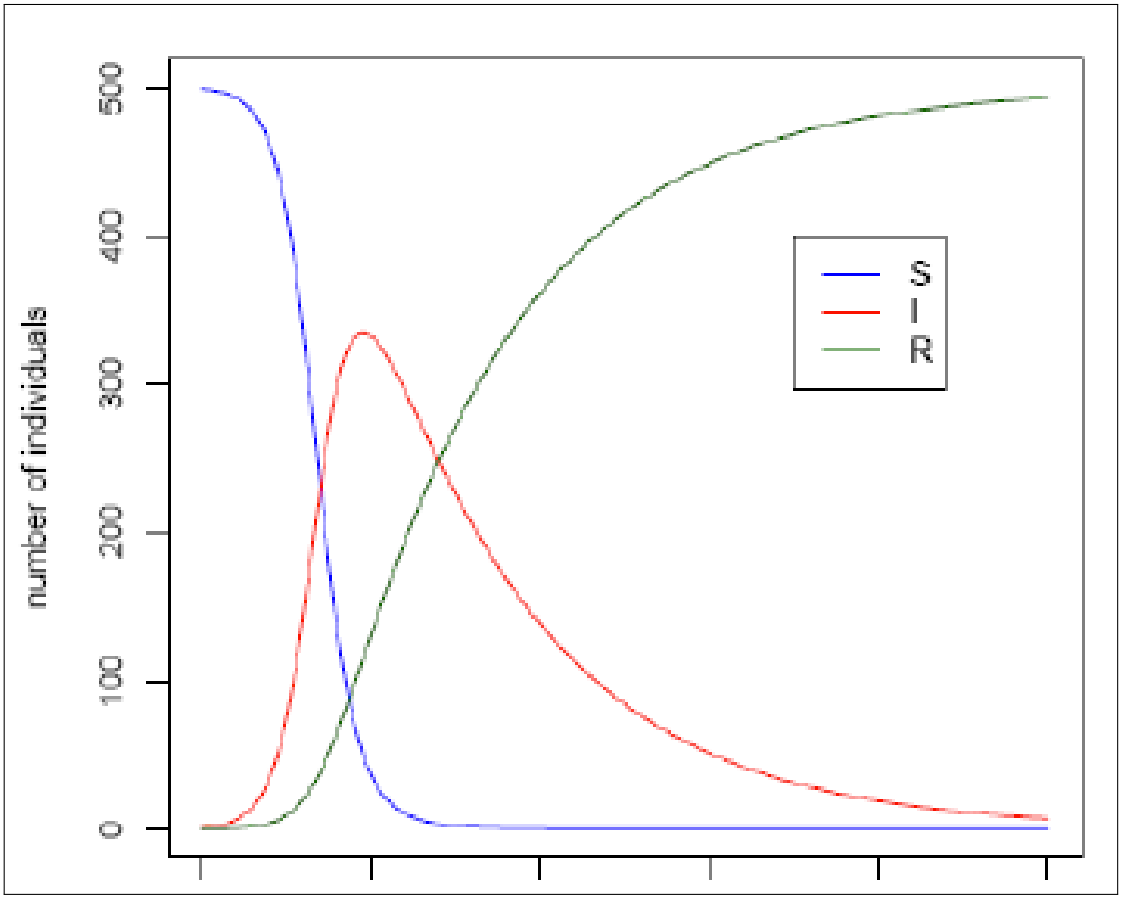
The solution to this system of differential equations 2.1,2.2, and 2.3 is non-analytically tractable (ARRATIA A., 2017). However, it is possible to infer the behavior of $s(t)$, $i(t)$ and $r(t)$ over time by analyzing the dynamics of the system.

Given the initial conditions of the system in which the total number of individuals is N, the $s(0) \simeq N$, $i(0) = \varepsilon$ where $0 < \varepsilon << N$, and $r(0) = 0$.

Initially, the population of infected $i(t)$ is very small (ε), its value begins to grow exponentially - therefore decreasing the number of susceptible individuals $s(t)$ in equal proportion. After some time, infected individuals recover and decrease the growth rate of $i(t)$ that tends to reach a maximum, decreasing after that. On the other hand, the number of recovery individuals tends only to increase and approximate to N, considering the premise that there are no deaths during the time of analysis. Figure 7 illustrate the behavior of the functions $s(t)$, $i(t)$, and $r(t)$ over time.

Although the standard SRI epidemic model has some limitations, such as assuming that the recovery time is the same for all individuals in the population, regardless of immunological response differences, it is assumed that all individuals in the population are perfectly mixed. That means the likelihood of infection is the same for all individuals. The model is the starting point for more sophisticated models, such as SRI stochastic epidemic models or even simplifications - as the case of the SIS epidemic model in which individuals, after infected, do not recover from the infection (DIEKMANN; HEESTERBEEK, 2000).

Figure 7 – The solution shape of s(t), i(t), and r(t) in a epidemic SRI model.



Source – <https://www.cs.upc.edu/~CSN/slides/11epidemic.pdf> (ARRATIA A., 2017)

2.2.3 Collective Intelligence

The term Collective Intelligence (CI) has become popular, once the success of Internet big companies like Google (1998) and Netflix(2006) consolidates it, as they have used CI concepts to create differentiated products in the market. However, CI is an active field of research that predates the web. Researchers from different fields like sociology, mass behavior, and computer science have previously made important contributions to this area (ALAG, 2008).

According to Heylighen (HEYLIGHEN, 1999), Collective Intelligence has its general concept based on the idea that a group of individuals (people, machines, robots, software agents) acting as a group may be smarter in solving problems than an individual acting in isolation way.

There are many examples where CI concepts help companies or organizations to offer differentiated products to their customers. The following pages will discuss some famous cases, identifying the way in which CI is used in their products. Among the famous examples of using CI in its products, it can be highlighted cases like Trip Advisor, Waze, Amazon, Google,

Netflix, and Wikipedia.

In this Section, the first three cases will not be commented, because they have been previously explained in Chapter 1. Thus, only the last three successful stories are discussed.

In Google's case, even though there were already large Internet search companies at the time, the company innovated the market, launching a search engine with results so superior to its competitor that basically dominated the market in the following years. One of the big differences in the quality of their searches was the fact that their search engine chose the results based on the number of links that the page had, thus using the collective knowledge of other web page developers. (SEGARAN, 2007).

On the other hand, Netflix - an internet streaming service company based on the concept of a monthly subscription instead of individually renting any movies (WIKIPEDIA, 2018a). The company applies the CI concepts on your movies recommendation system for their subscribers. The Recommender system utilizes the evaluation of its subscribers on watched movies to suggest different movies not watched by them. In this way, the suggestions of each subscriber depending on the movies he or she watched and liked, as well as the evaluation of other subscribers with similar taste (SEGARAN, 2007).

Another interesting example is the case of Wikipedia, considered today the largest encyclopedia in the world with 5,598,527 articles and 44,597,037 pages and more than 33,236,259 users (WIKIPEDIA, 2018b). This whole knowledge base has been collaboratively built by editors and reviewers, but not necessarily specialists, in the article areas. According to a study conducted in 2005 by Scientific journal Nature (GILES, 2005), that compared the content of Wikipedia articles and the British encyclopedia by submitting it to a specialist group, it was shown that the accuracy is very close, and in all reviewed articles were found 162 and 123 errors by reviewers, from Wikipedia and Britannica respectively.

From a purely conceptual point of view, CI does not need technological tools for its implementation. However, it is a fact that the Internet has uniquely and unprecedentedly enabled the effective use of CI concepts due to its enormous amount of users and facility in communication between them (LYKOURENTZOU *et al.*, 2009).

According to (LYKOURENTZOU *et al.*, 2009) CI systems can be divided into two categories:

- Passive CI system - In this type of system the user follows his behavior and action in a normal way without any modification of his actions by the existence of the system. For

example, the Google search engine that ranks the pages based on the number of links pointing to it. Page developers will continue to develop the pages the same way they would develop without Google's rankings system.

- Active CI system - Unlike passive systems, in this type of system users do not have a pre-existing behavior before the system. The system is the inducer of the way in which users will collaborate with each other. In the same category, users can still collaborate in three different modes: collaborative, competitive or hybrid. In the collaborative mode, the users collaborate aiming at the individual and community goals. In the case of the competitive mode, the users compete with each other in order to reach the best solution. Finally, the hybrid mode is a mix of the two forms where there is competition and collaboration. A real example of an active CI system is Wikipedia.

In Table 2, some CI systems and their respective classifications and characteristics are showed.

Table 2 – Different types of CI systems.

CI system	Wikipedia	Competitive problem solving companies	Vehicular network coordination
Type	Active collaborative	Active competitive	passive
Set of user actions	Contribute knowledge	Contribute ideas	Accelerate
System state	Article quality	Solutions received	vehicle distances
Community Objective	High article quality	Best possible solution	Minimize traffic congestion, Maximize vehicle safety of the network
Individual objective	Self fulfillment	Monetary compensation	Prompt reaching of one's destination low gas consumption maximize safety of individual vehicle

Source – ACM-MEDES '09 Proceedings of the International Conference on Management of Emergent Digital EcoSystems (LYKOURENTZOU *et al.*, 2009).

2.2.4 Recommender Systems

Since the earliest works in the mid-1990s, Recommendation Systems (RS) (HILL *et al.*, 1995), (RESNICK *et al.*, 1994), (SHARDANAND; MAES, 1995) are a research area that has attracted reasonable interest from academia and industry mainly for its abundant practical

applications (ADOMAVICIUS; TUZHILIN, 2005).

RS is one of the best ways to apply CI concepts to the real world (ALAG, 2008). There are several real examples of RS applied to the suggestion of books, films, products, music, scientific articles, websites, among others. Nowadays, the use of RS applications in e-commerce wide, regarding it as a key to enable technology (JANNACH *et al.*, 2010).

In fact, there are various reasons to explain why the companies and service providers may want to exploit this technology (RICCI *et al.*, 2011):

- Increasing the number of sold items: Since the recommended items are based on the user's preferences, it is very likely that s/he will recognize - with the use of RS - that the suggested items meet her/his needs, thus increasing her/his chances of buying new items suggested by the system.
- Selling more diverse items: Another important function of an RS is to present to the user items that are not necessarily the most popular ones. However, there is a reasonable likelihood of the user liking and buying them, therefore allowing RS to offer more diversity of items;
- Increasing user satisfaction: A well-designed RS can enhance the user's experience by suggesting interesting, relevant recommendations. On that account, improving the subjective evaluation of the system by the user.
- Increasing user loyalty: An essential factor that makes a user loyal to a service or product is highly related to the way he feels treated and differentiated in his service while he or she is using the service or product. On the other hand, a relatively common feature in RS is the utilization of customer history and, based on it and other factors, the generation of its recommendations. So this fact makes it possible for RS to refine its suggestions more and more, as the user utilizes the system. This way, the RS may be generating better and more personalized recommendations, thereby enhancing the customer's loyalty.
- Better understanding of what the user wants: Another benefit of the use of RS is to provide a reasonable database for the service provider, which are the preferences of their clients. The use of this data helps with the definition of management strategies, marketing and policies of relationship with customers.

According Balabanovi (BALABANOVIĆ; SHOHAM, 1997), RS can be classified into three categories based on how the recommendations are made, which are listed as follows (see also Table 3):

- Content-Based Recommendations: RS uses items preferred by the user in the past to recommend items similar to these items but not yet chosen by him in the past;
- Collaborative recommendations: RS searches for items chosen by users with similar taste to the user, in order to recommend him the right suggestions; and
- Hybrid approaches: These methods combine collaborative and content-based methods.

Table 3 – Classification of Recommender Systems

Recommendation Approach	**Recommendation Technique**	
	Heuristic-based	**Model-based**
Content-based	TF-IDF(Information retrieval) Clustering	Bayesian classifiers Clustering Decision trees Artificial neural networks
Collaborative	Nearest neighbor (cosine, correlation) Clustering Graph theory	Bayesian classifiers Clustering Artificial neural networks Linear regression Probabilistic models
Hybrid	Linear combination of predicted ratings Various voting schemes Incorporating one component as part of heuristic for the other	Incorporating one component as a part of model for the other Building one unifying model

Source – IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 17, NO. 6, JUNE 2005 (ADOMAVICIUS; TUZHILIN, 2005)

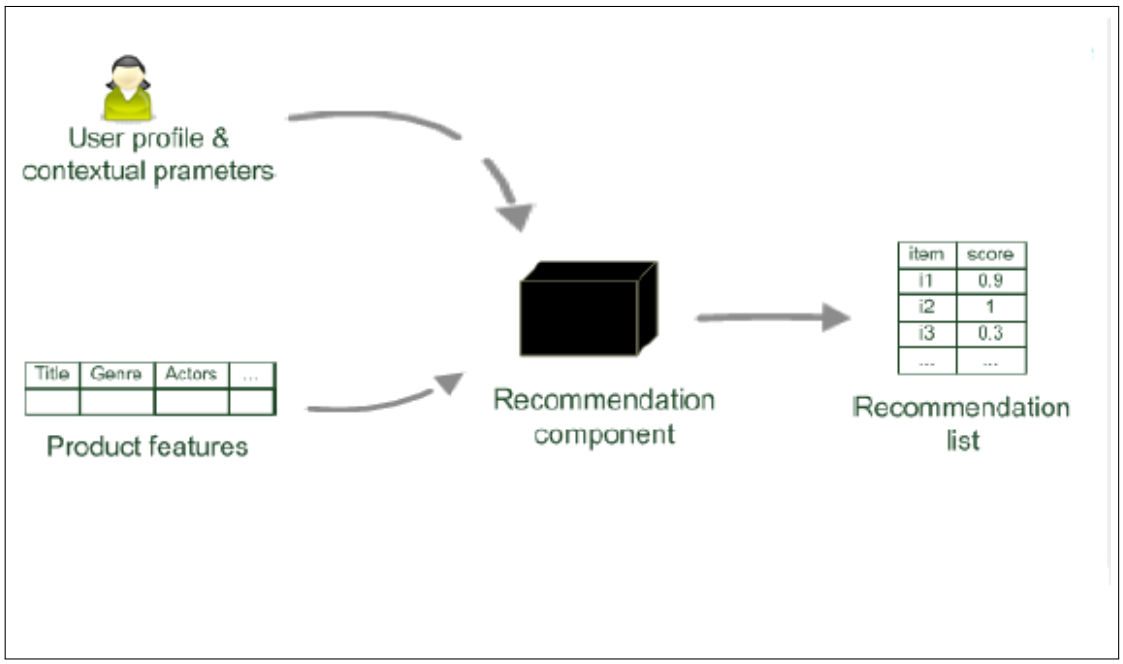
The Content-Based (CB) RS has as its basic principle the use of previous evaluations of the user in different items, estimating the score of items similar to those evaluated items. They do that by using a measure of similarity between them (PAZZANI; BILLSUS, 2007). Thus, this measure of similarity is made based on the content of items that can be structured as a set of features on the item, or unstructured in a free description of the item (JANNACH *et al.*, 2010).

In the literature, there are several approaches and techniques used to implement content-based RS. Among them, it can be mentioned (ADOMAVICIUS; TUZHILIN, 2005): traditional heuristic forms based on Term Frequency/Inverse Document Frequency (TF-IDF) (SALTON, 1989) measure, or similarity of cosines, Bayesian Classifiers (MOONEY *et al.*, 1998) (PAZZANI; BILLSUS, 1997), machine learning techniques including clustering, decision trees, and artificial neural networks (PAZZANI; BILLSUS, 1997).

Figure 8 presents a diagram of inputs/outputs used for Content-based RS.

Unlike content-based recommendation methods, the collaborative recommendation

Figure 8 – Diagram of Content-based RS inputs.



Source – Recommender Systems An introduction (JANNACH *et al.*, 2010).

systems (or collaborative filtering systems) predict the usefulness of items for a particular user based on the previous evaluations of other users with similar taste to that item (ADOMAVICIUS; TUZHILIN, 2005). Similarly to the content-based approach, the collaborative approach also needs a similarity metric; however, it is necessary to compare items rather than just users to estimate the usefulness of an item to a particular user.

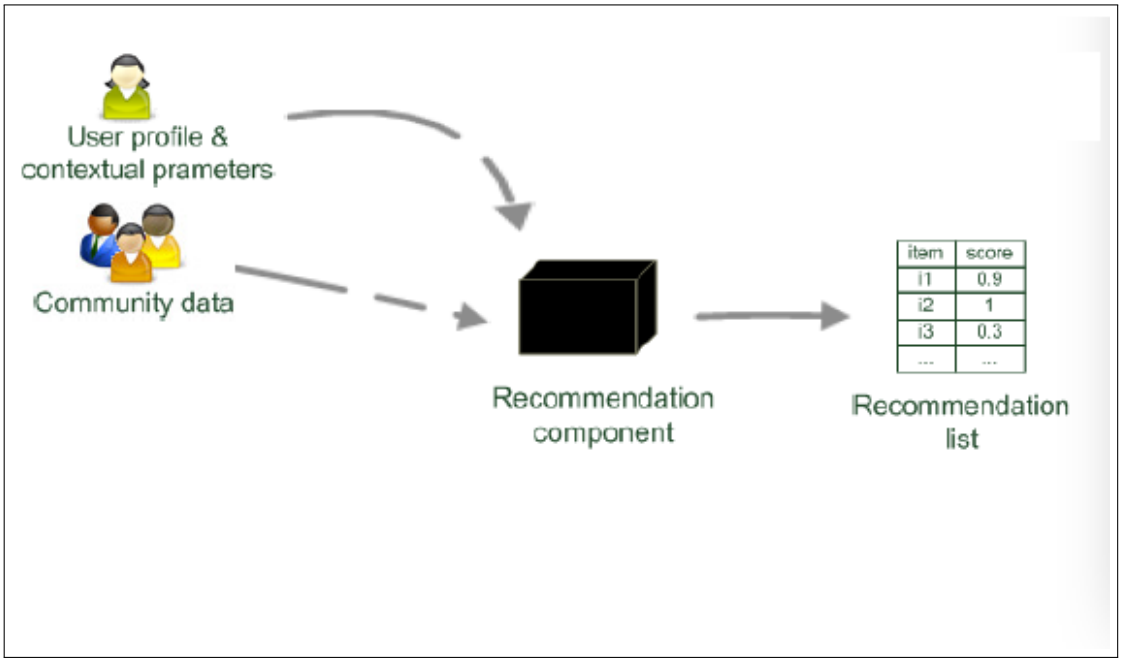
According to (BREESE *et al.*, 1998), algorithms of collaborative recommendations can be grouped into two general classes: memory based (or heuristic based), and model-based. Memory-Based Algorithms use heuristics which, based on all classified items and a measure related to the similarity between the user and other users, make predictions of the user's utility for the item. In this class of memory based RS, the techniques commonly used are the Nearest neighbor (cosine, correlation), Clustering, and Graph theory (BREESE *et al.*, 1998) (DELGADO; ISHII, 1999) (NAKAMURA; ABE, 1998).

Unlike collaborative memory-based RSs, the collaborative model-based RSs uses machine-learning and statistical techniques to learn a model, based on previously-evaluated user assessments. Again, there are several approaches used to learn the model, among which it is possible to mention: probabilistic models, cluster models and Bayesian networks proposed by (BREESE *et al.*, 1998), a statistical model for collaborative filtering (UNGAR; FOSTER, 1998), machine learning techniques such as artificial neural networks coupled with feature extraction

techniques (BILLSUS; PAZZANI, 1998).

Figure 9 presents a diagram of inputs/outputs used for Collaborative RS.

Figure 9 – Diagram of Collaborative RS inputs/outputs.



Source – Recommender Systems An introduction (JANNACH *et al.*, 2010).

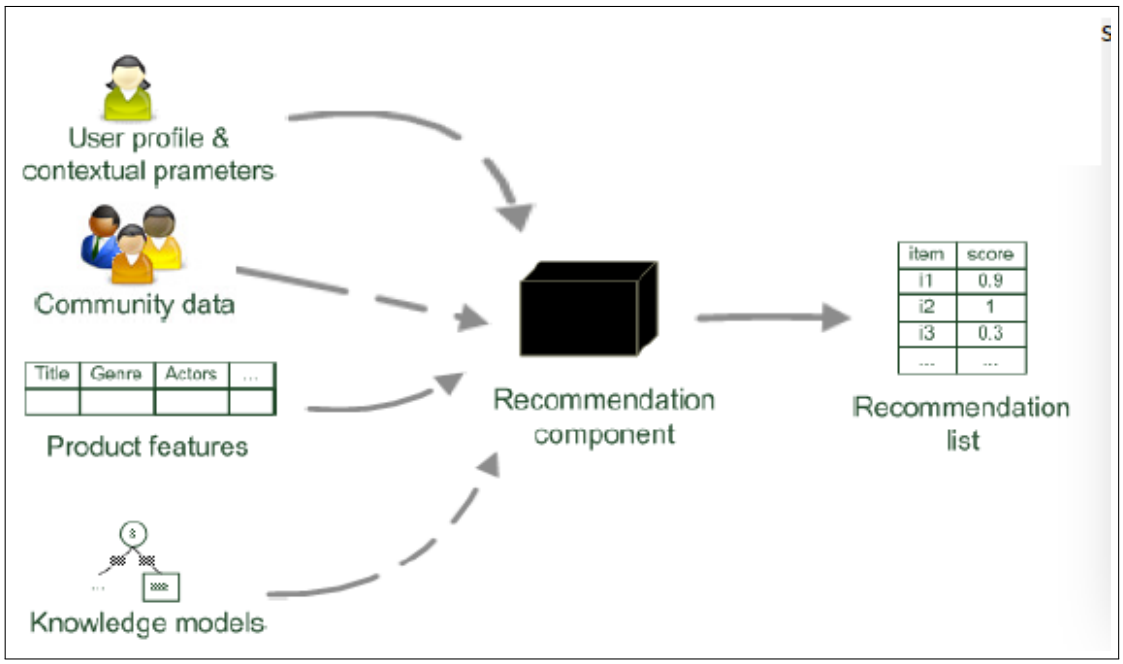
Hybrid methods combine content-based methods with collaborative methods to help avoid limitations of both models (BALABANOVIĆ; SHOHAM, 1997) (CLAYPOOL *et al.*, 1999) (BASU, 1998) (SCHEIN *et al.*, 2002) (PAZZANI, 1999).

In this Hybrid RS mode, it is possible to create the hybrid approach in different ways. In Figure 10, the diagram of the main inputs and outputs of a hybrid approach are illustrated.

According to (ADOMAVICIUS; TUZHILIN, 2005), the different ways to create the hybrid approach are discussed and summarized as follows:

- Combining Separate Recommendations: In this approach, two RSs are implemented independently, one being content-based and the other being collaborative; a quality recommendation metric is also defined for each RS (BILLSUS; PAZZANI, 2000). The final evaluation is usually a linear combination of these evaluations with the highest score in the quality metric (CLAYPOOL *et al.*, 1999).
- Adding Content-Based Characteristics to Collaborative Models: In this approach, it is implemented an RS based on traditional collaborative techniques. However, it is keeped for each user their profile related to the content. In contrast to the collaborative RS, that

Figure 10 – Diagram of Hybrid RS inputs/outputs.



Source – Recommender Systems An introduction (JANNACH *et al.*, 2010).

uses previous evaluations to estimate the degree of similarity of users in this modality, the content-based profile is used and uses this profile to evaluate the similarity (PAZZANI, 1999) bringing benefits when compared to traditional collaborative approaches (GOOD *et al.*, 1999).

- Adding Collaborative Characteristics to Content-Based Models: This technique consists on a content-based RS, yet it is applied a technique of dimensionality reduction in the user's profile, such as Latent Indexing Semantics (LIS) (SOBOROFF; NICHOLAS, 1999), and on this reduced profile it is possible to create a collaborative view of groups of user profiles.
- Developing a Single Unifying Recommendation Model: This technique proposes a new model that is different from traditional collaborative and content-based RS models. Among the techniques classified in this category, it can be highlighted: single rule-based classifier (POPESCUL *et al.*, 2001), probabilistic latent semantic analysis (HOFMANN, 1999), and Bayesian mixed-effects regression models combined with Markov chain Monte Carlo methods for parameter estimation and prediction (CONDLI *et al.*, 1999).

2.2.5 Collaborative Filters

Collaborative Filters (CB) is an algorithm used in RS that bases its predictions and recommendations on the classifications performed by other users of the system (EKSTRAND *et al.*, 2011). Thus, CB based its prediction on those patterns which do not rely on an item or user attributes (MACKEY, 2009).

The main idea behind this method is the assumption that users agree on quality and relevance in some items will probably agree on other items as well(EKSTRAND *et al.*, 2011).

According to (MACKEY, 2009), the main function of Collaborative Filters (CF) is to discover patterns in the behavior of the observed preferences (eg. purchase history, item rankings, click counts) in a community of users and based on these patterns predict new preferences.

Formally, the CF problem can be described as given the set of users $u \in \{1,...,U\}$ and the set of items $i \in \{1,...,M\}$ and let τ be the training set with real preference values $r_{u,i}$ for some of the user-item pairs (u,i). Let's assume R is a sparse matrix where the rows represent the items, the columns the users, and each element $r_{i,u}$ is the evaluation given to item i by user u. Note items not evaluated are indicated by the symbol '?'.

$$R = \begin{bmatrix} r_{11} & ? & r_{13} & \dots & x_{1M} \\ ? & r_{22} & ? & \dots & ? \\ & \dots\dots\dots\dots & & & \\ r_{M1} & r_{M2} & ? & \dots & r_{MU} \end{bmatrix}$$

The problem aims at predicting unobserved preferences ('?') of the test set Q with pairs (i,u) not in τ.

One metric to evaluate the quality of the prediction is to compare the predicted value of preference $\hat{r}_{i,u}$ with the value of the true preference $r_{i,u}$ in the set Q. This can be done by calculating the Root Mean Square Error (RMSE) defined as:

$$RMSE = \sqrt{\frac{1}{|Q|} \sum_{\substack{i \in Q \\ u \in Q}} |r_{iu} - \hat{r}_{iu}|^2} \tag{2.4}$$

Another possible metric to evaluate the quality would also be the Mean Absolute Error (MAE) defined as follows:

$$MAE = \frac{1}{|Q|} \sum_{i,u \in Q} |r_{iu} - \hat{r}_{iu}| \tag{2.5}$$

An important point to consider when working with CF is that some users systematically evaluate with high preference values and some items are also evaluated with high utility values. Being so, a good practice to be considered is to centralize the data prior to the application of any CF algorithm, this can be done by removing Bias from the evaluations (EKSTRAND *et al.*, 2011). Another relevant point about CF is that in the matrix R there may be many items with little or no previous classifications; this way a possible classification would be the global mean (μ) of the classifications of all the classified items.

Thus, let's define some important system parameters:

Global mean rating :

$$\mu = \frac{1}{|\tau|} \sum_{i,u \in \tau} r_{iu} \tag{2.6}$$

Item's mean rating :

$$b_i = \frac{1}{|R(i)|} \sum_{u \in R(i)} r_{iu} \tag{2.7}$$

where $R(u)$ is the set of users who rated item i.

User's mean rating :

$$b_u = \frac{1}{|R(u)|} \sum_{u \in R(u)} r_{iu} \tag{2.8}$$

where $R(u)$ is the set of items rated by user u.

Then, to remove the biased term from each rating, it is must use the equation 2.9, before applying to CF algorithm.

$$\tilde{r}_{iu} = r_{iu} - b_{iu} \tag{2.9}$$

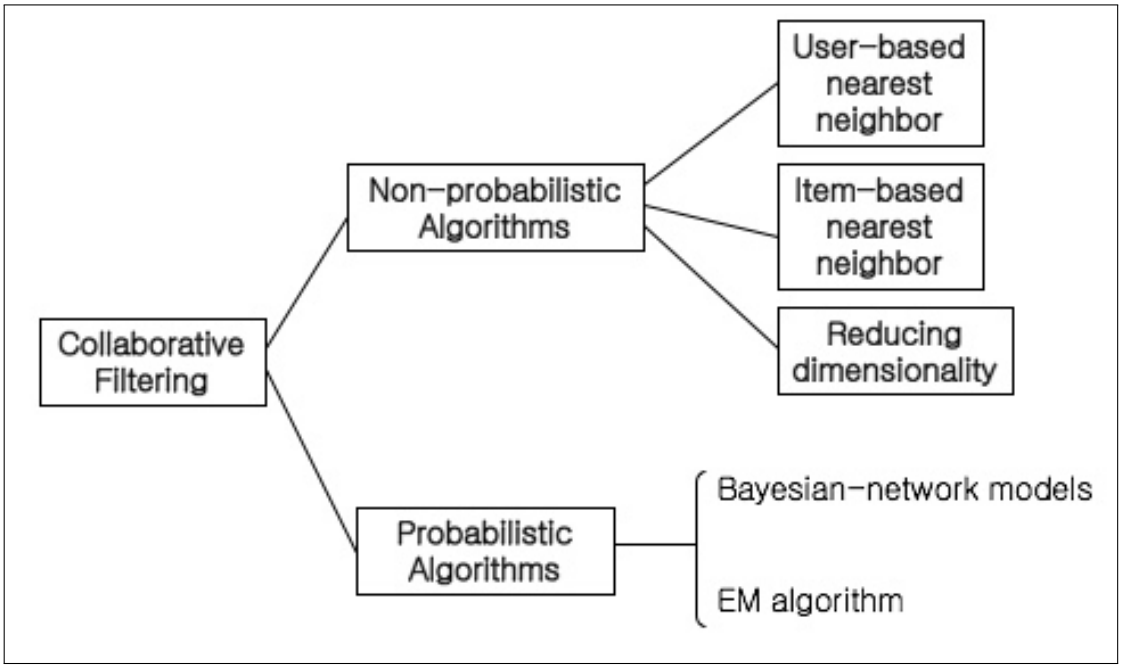
In order to adjust the values of the user's utility, estimate it so that when there is few or no data the value should tend to the global mean. When the number of evaluations is large the value tends to the user's mean. Thus, (FUNK, 2006) proposed the equation 2.10 where α is a damping term to adjust the degree of shrinkage. In his research, (FUNK, 2006) found empirically that 25 was a useful value for the damping term.

$$\tilde{b}_u = \frac{\alpha}{\alpha + |R(u)|} * \mu + \frac{|R(u)|}{\alpha + |R(u)|} * B_u \tag{2.10}$$

There are several approaches to address the CF problem but in general, these forms can be classified into two broad categories: Non-Probabilistic Algorithms and Probabilistic

Algorithms (Naive Bayes Classifier, and EMX algorithm) as shown in Figure 11. In this Section, probabilistic algorithms are not discussed, since they are not used in this thesis.

Figure 11 – Collaborative Filtering methods.



Source – Collaborative Filtering - University Pittsburgh (SYN, 2005).

Among the non-probabilistic algorithms, the methods most used are Classification / Regression methods, K Nearest Neighbor Methods (User-Based Nearest Neighbor and Item-Based Nearest Neighbor), and Dimension reduction.

A possible approach to CF is to address the problem as several isolated Classification/Regression problems organized by item. An advantage of this proposal is that the Classification/Regression problem is a well-known problem having many good prediction algorithms available. However, it can be computationally costly depending on the number of items.

In the Classification/Regression proposal, the following steps described must be followed to predict one rating r_{iu} for item i for user u:

1. Choose one Classifier/Regression algorithm, treating each user as an incomplete vector of user's ratings for all items except i;
2. Train separate predictors for each item; and
3. To predict r_{iu} for user u and item i, apply item $i's$ predictor to vector of user $u's$ incomplete ratings vector.

Another very popular CF approach is K Nearest Neighbor (KNN) (RESNICK *et al.*, 1994) (SHARDANAND; MAES, 1995) (HILL *et al.*, 1995). In such method, it is considered rating measured by a degree of similarity of users and/or items, in order to calculate the prediction of the evaluation of item i for a user u.

Depending on which group (users or items) it is used to evaluate the similarity, two different modalities can be performed using the KNN method: user-based or item-based.

In the user-based approach, the main idea is that users with similar ratings on many items are likely to agree on an item not evaluated by one of them. This way, the prediction is a rate measured by the similarity of K neighbor users from u, where K is a previously defined integer.

The equation 2.11 expresses this concept in a mathematical form calculating the prediction rating of user u about item i.

$$p_{ui} = \bar{r}_u + \frac{\sum_{u' \in N} s(u,u') * (r_{iu'} - \bar{r}_{u'})}{\sum_{u' \in N} s(u,u')} \tag{2.11}$$

where $N \subset U$ is defined as neighbor of u, U is the set of users, and $s(u,u')$ is the similarity function between u and u'.

Normally three similarity metrics are used to measure 2.12, 2.13, and 2.14 this a degree of similarity between users defined as following:

1. Cosine similarity

$$s(\vec{u}, \vec{u}') = \frac{\vec{u} \cdot \vec{u}'}{|\vec{u}||\vec{u}'|} \tag{2.12}$$

2. Pearson correlation coefficient

$$s(\vec{u}, \vec{u}') = \frac{(\vec{u} - mean(\vec{u})) \cdot (\vec{u}' - mean(\vec{u}'))}{|(\vec{u} - mean(\vec{u})||(\vec{u}' - mean(\vec{u}')|} \tag{2.13}$$

3. Inverse Euclidean distance

$$s(\vec{u}, \vec{u}') = \frac{1}{|(\vec{u} - \vec{u}'|} \tag{2.14}$$

These metrics assume complete vectors, so it is possible to compute it only over a subset of items rated by both users.

Similarly, in the item-based approach, the main idea is users rate similar items likewise. Thus the prediction is a weighted average by the similarity of K neighbor items from i, where K is a previously defined integer.

The equation 2.15 expresses this concept in a mathematical form calculating the prediction rating of user i about item u (SARWAR *et al.*, 2001).

$$p_{ui} = \frac{\sum_{j \in S} s(i,j) * r_{iu'}}{\sum_{j \in S} s(i,j)} \tag{2.15}$$

where $S \subset I$ defined as similar items of i, I is the universe set of items, and $s(i,j)$ is similarity function between i and j.

The similarity functions are the same as those used in the user-based method calculated according to the equations 2.12,2.13, and 2.14.

In many real scenarios, it is common for the number of users and items to be very large, therefore generating a matrix R of very high dimensions. Thus, a technique widely used in Machine Learning (ML) to reduce the computational complexity of the problem is to reduce the dimensions by using the Singular Value Decomposition (SVD) theorem (GOLUB; REINSCH, 1970). Intuitively this reduction would be equivalent to grouping with K similar user groups and K groups of similar items as well, instead of dealing with the totalities of users and items.

Applying the SVD theorem for the matrix R_{mxn} can be written in the form of a factorization showed at the equation 2.16.

$$R = U\Sigma V^* \tag{2.16}$$

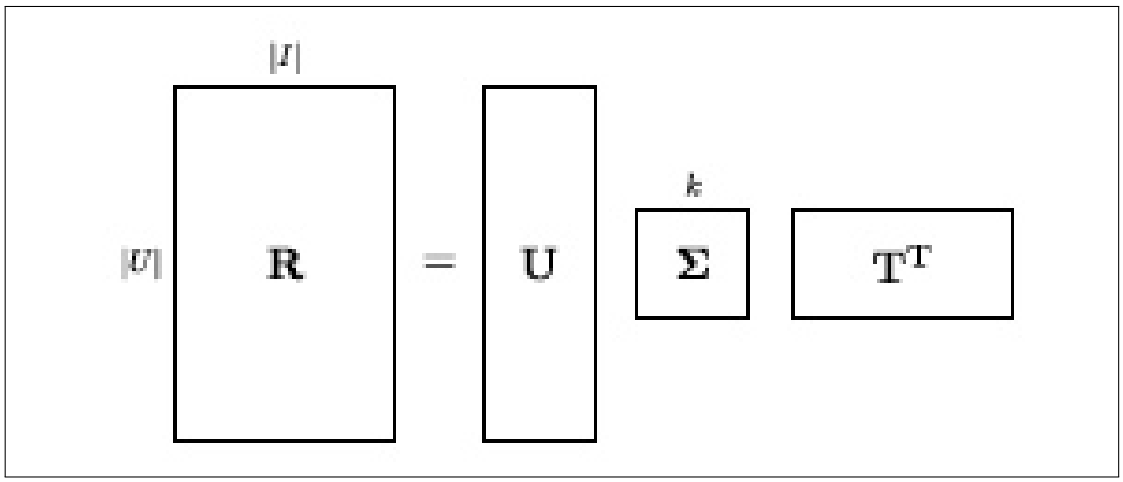
where:

- U is an $m \times m$ matrix;
- Σ is a diagonal $m \times n$ matrix with non-negative real numbers on the diagonal;
- V is an $n \times n$ matrix;
- V^* is the conjugate transpose of V; and
- the columns of $U, U^\star, V$, and $V^\star$ are orthonormal bases.

The dimensionality reduction consists on truncating the Σ matrix using only the largest singular k values to yield Σ_K, thus, generating an approximation of the matrix R see equation 2.17.

Computing the SVD of the ratings matrix results in the following factorization, with $m = |U|$, $n = |V|$, and Σ a $k \times k$ diagonal matrix. Figure 12 illustrates the dimensionality reduction process.

Figure 12 – Dimensionality reduction process.



Source – Collaborative Filtering Recommender Systems (EKSTRAND *et al.*, 2011).

$$R \approx U\Sigma V^* \tag{2.17}$$

In the literature, there are several forms to calculate SVD for complete matrices and with missing elements. Among the main ones, it can be highlighted: Expectation-Maximization (EMX) algorithm, Alternating Least Squares (ALS) algorithm, and Gradient descent algorithm. In this Section, it will be not detail these methods because SVD in not a scope this thesis.

2.3 Summary

This Chapter presented the background of this thesis. First, Network topics are introduced the related to network environment studied in the thesis. Next, the main related to the topics involved in this thesis work, which are Edge Computing, Fog Computing, and Mist Computing, are described. Besides that, a set of challenges related to the before mentioned topics is presented.

Regarding the Bio-inspired systems, this Chapter first introduced the main concepts and ideas, highlighting its importance of this kind of system in the Computer Science area. After that, it was presented the SRI Epidemic models used by us in a data distribution method, which is proposed in this thesis for handling the dynamism of network topology in FMC environment.

The Recommender system was addressed by introducing the concepts of collective intelligence. Furthermore, it was depicted how the Recommender system applied Collective Intelligence concepts, including how they are classified. The Collaborative Filtering is important

to support the development of the Collective Recommender systems engines.

The collaborative filtering technique will be used in the thesis to be the tool to capture knowledge of the computational resources and devices that are part of the network environment being studied.

3 RELATED WORK

This chapter details the literature review conducted by a research on studies concerning issues related to the research questions (RQ1, RQ2 and RQ3) of this thesis, which are presented in Chapter 1. The searches are done using the databases and search tools available from IEEE Xplore, ACM Digital Library, ScienceDirect and Google Academics.

The following sections present the studies found in the literature: Section 3.1 presents the studies related to IoT network infrastructure selection to provide computational resources; Section 3.2 discusses the studies concerning the data dissemination in the IoT environment domain; Section 3.3 presents the work related to the use of the Recommender System in IoT environment for service of computing resource allocation; and, finally, Section 3.4 concludes this chapter.

3.1 IoT environment selection (RQ1)

In this Section, scientific work were searched in a similar way to the presented algorithms or models of architectures - help in choosing the best computational infrastructure to get computational resources in Cloud, Fog or Mist computing or combination of them. This issue refers to the first question of research for this thesis.

In general, the selected related works can be grouped into four categories.

The first category is composed of articles intended to serve as a design guide or reference for IoT designers. In particular, the environment at the edge of the network. In this category, the following papers can be highlighted:

In Shi et al. (SHI *et al.*, 2015) focus on proposing a model of integration between Fog and cloud computing through the CoAP Protocol in a P2P-style architecture, distinct from the traditional hierarchical proposal. However, the work does not provide feasibility analysis or limitations of the use of computational resources in Cloud or Fog computing.

In Aazam et al. (AAZAM; HUH, 2016), the authors feature computing paradigms for the provision of computer resources used in IoT (Fog, Cloud or Hybrid) discussing the strengths, weaknesses, architectures and issues of each approach. The work is a guide for IoT's application designers. However, the article does not propose a precise mechanism that can be evaluated by devices for choosing the best paradigm according to the context of the environment.

In Bruin et al. (MASIP-BRUIN *et al.*, 2016), authors make an extensive discussion of

the challenges in the open Fog computing models, defending an architecture with the introduction of a new management layer that should coordinate resources between Fog and Cloud computing. However, the proposal is not a model of decision and an architectural model that the authors suggest taking advantage of the best capabilities of both platforms.

In Pereira et al. (PEREIRA *et al.*, 2017), the authors do a practical study and characterization of latency in an e-health IoT application by using a mobile gateway for e-health sensor data capture, that relies on ETSI M2M communications for sending to the openEHR platform for storing and exposing health records. Nevertheless, the focus of the paper is the characterization of latency in each phase of architecture, not in its scope selecting computing resource on FMC environment.

In Dolui et al. (DOLUI; DATTA, 2017b), the authors present Edge Computing as a new paradigm, discussing the major forms of its implementations: Fog Computing, Mobile Edge Computing, and Cloudlet Computing. In this work, the authors also compare these implementations and suggest a guide for choosing one of them, depending on design requirements. However, the article's focus is to serve as a guide for the designer, not proposing any solutions that fit dynamically depending on context.

In the second category of work, the main focus is to create a model that allows, through some optimization technique, to optimize one or several variables of the study network environment, such as latency or cost. In this category, the following works can be highlighted:

Aazam et al. (AAZAM *et al.*, 2016) present a model of resource allocation in probabilistic Fog Computing to enable better resource management, obtaining good results in a practical experiment. However, the proposed model does not consider limitations of latency introduced by a Fog computing environment topology.

The work of Deng et al. (DENG *et al.*, 2016) proposes a mathematical model that considers energy consumption and delays in Cloud and Fog computing environment. The proposed model, validated through simulations, suggests an optimal allocation of resources between the Cloud and Fog computing. However, the proposed optimization is made centrally and does not consider issues such as the monetary cost of CoT.

In Sarkar et al. (SARKAR, 2016), the authors propose a mathematical model for Fog computing, and compare it with the traditional model of cloud computing focusing on energy consumption. The authors show that applications that using part of the infrastructure in fog computing can save about 40% of energy expenditure. However, the authors do not explore other

constraints typically found in some IoT applications.

In the third category, it is presented works that propose as a solution a combination of computational resources provided by Fog and Cloud computing. In this line of research the following works can be highlighted.

Sharma et al. (SOUZA *et al.*, 2016) propose a resource optimization model using a combination of Fog computing and Cloud computing. In this proposal, authors further subdivide Fog computing into two additional layers: one with little latency, and another with average latency. This proposal seeks to optimize the problem of resources, minimizing the total system delay restricted to the condition of the requested resources. Nevertheless, like the previous model, this proposal also does not consider issues associated with the costs of the services of the CoTs and is optimized centrally.

Qui et al. (PHAM; HUH, 2016) address the problem of task scheduling between Fog and Cloud computing, considering the priority of tasks, topological distribution of nodes of Fog Computing and costs associated with the services of the CoTs. However, the work cannot be applied on a large scale and does not cover the use of other features of Fog computing in addition to processing.

In klein et al. (KLEIN *et al.*, 2012), the authors propose a service combination model in an almost optimal solution that using genetic algorithms. The presented results suggest that the proposal presents good results and that scale better than similar solutions. However, the approach presented is centralized and again does not consider specific requirements of a client device.

In Wang et al. (WANG *et al.*, 2016), the authors propose the network-aware cloud service composition approach, named NetMIP that formalizing the service composition goal as a multiobjective constraint optimization problem. The proposal, unlike similar works, considers the impact on network resource consumption and aim to find an optimal service composition solution with minimal network resource consumption and realistic QoS optimality. However, although some concepts of the proposal can be applied in other network environments, the focus of the study is cloud computing.

In Yadwadkar et al. (YADWADKAR *et al.*, 2017), the authors present PARIS, a data-driven system that uses a novel hybrid offline and online data collection and modeling framework to provide accurate performance estimates with minimal data collection. The proposal is able to predict workload performance for different user-specified metrics, and resulting costs

for a wide range of Virtual Machine (VM) types and workloads across multiple cloud providers. The tool assists in reducing costs with cloud computing services by allowing the user to select the lowest cost VM configuration that meets their estimated workload. However, the proposal is not applicable in FMC environment.

Finally, the last category of work, in which this thesis fits, aims to identify the best device in the environment, capable of providing a particular computational resource. In this category, the following works can be highlighted.

In Wang et al. (WANG *et al.*, 2017), the authors propose in the context of Mobile Edge Computing(MEC) a model of selection of the best server edge using a recommendation system that uses collaborative filtering and similarity of Pearson Correlation Coefficient (PCC). between server edges to select the server to be used by a user. The work shows good results of the techniques when applied in the Shanghai Telecom dataset. However, the paper does not address FMC context issues such as heterogeneity.

In Lei et al. (LEI *et al.*, 2016), the authors propose an adaptive mobile data traffic offloading model selecting multiples data transfer mechanisms based on analysis of data Traffic Offloading Rate (TOR) on continuing time Markov chain in cellular M2M networks, and Local Resource Consumption Rate (LRCR)) . However, the work does not consider latency requirement issues or another one defined by the client device.

In Aazam et al. (AAZAM; HUH, 2015), the authors a computational resource estimation model in conjunction with a pricing model for services in the Fog computing environment. However, the work presumes the figure of a smart gateway that plays the role of coordinating this pre-allocation of resources.

In Table 4, it is presented a brief comparative summary of the related works. The main differences of this thesis compared to those mentioned before are the introduction of the feasible Fog concept, which allows reducing search space, and the fact that this proposal is open to adding other requirements necessary to client devices. Conversely, all related work cited here focus on predefined requirements, such as latency, energy or processing power.

3.2 Data distribution in FMC environment (RQ2)

In this Section, a more in-depth analysis of the work directly related to RQ2, as well as identifying their stronger and weaker points. Also, main differences and contributions of our proposal is presented the for this research area. The issues dealt with in this Section relate to the

Table 4 – Comparation of related work for RQ1.

Paper Author	Proposal	Processing type	Optimized Parameters
klein et al.(KLEIN *et al.*, 2012)	a service combination model in an almost optimal solution that using genetic algorithms	Centralized	Latency
Shi et al. (SHI *et al.*, 2015)	P2P architecture model using CoAp protocol combining Fog and Cloud computing	Not applicable	Not applicable
Aazam et al. (AAZAM; HUH, 2016)	Architectural guideline for IoT application designer	Not applicable	Not applicable
Bruin et al. (MASIP-BRUIN *et al.*, 2016)	Architectural model using a services coordination layer	Centralized	Runtime and speed
Lung et al. (AAZAM *et al.*, 2016)	Statistical model of resource availability in Fog	Centralized	Latency, cost
Deng et al.(DENG *et al.*, 2016)	Mathematical model for optimization of computer resource located at Fog and Cloud	Centralized	Power consumption, Delay
Sarkar et al. (SARKAR, 2016)	Mathematical Model for computer resource allocation in Fog computing focus on power consumption compared with Cloud computing	Centralized	Power consumption
Sharma et al. (SOUZA *et al.*, 2016)	Computer resource allocation model at combined Fog/Cloud using integer optimization	Centralized	Latency, Computational resources
Qui et al. (PHAM; HUH, 2016)	Algorithm of task allocation between Fog/-Cloud considering topological distribution of nodes	Centralized	Runtime, cost
Wang et al. (WANG *et al.*, 2016)	Cloud service composition with minimal network resource consumption and realistic QoS optimality	Centralized	QoS and network resources
Lei et al. (LEI *et al.*, 2016)	adaptive mobile data traffic offloading model selecting multiples data transfer mechanisms	Centralized	Traffic volume and resource comsumption
Wang et al. (WANG *et al.*, 2017)	Model of selection of the best server edge in Mobile Edge Computing using a collaborative Recommendation System	Centralized	QoS- Quality os Service
Yadwadkar et al. (YADWADKAR *et al.*, 2017)	a data-driven system that is able to predict workload performance for a wide range of VM types and workloads across multiple cloud providers	Centralized	Latency, cost, CPU, Disk
Dolui et al. (DOLUI; DATTA, 2017b)	It presents edge computing, showing how its implementation forms and characterizes each of them	Not applicable	Not applicable
Pereira et al. (PEREIRA *et al.*, 2017)	practical study and characterization of latency in an e-health IoT application	Centralized	Latency

Source – Author.

second research question (RQ2), which seeks to discover mechanisms of how to maintain data in an environment with high dynamics of the topology.

In Piotrowski et al. (PIOTROWSKI *et al.*, 2009), the authors present a collaborative data storage (tinyDSM) middleware that addresses the common problems of store data on Wireless Sensor Networks (WSN) using data replication on nodes of the network. However, in the proposal, the decision of having the node assuming the role of data replicators is static, and uses a random criteria that depends only on the number of nodes and density of the network. It has created some problems in case that node would disappear from the network.

In Lieskovsky et al. (LIESKOVSKY *et al.*, 2011), the authors present a solution for data distribution and replication in Vehicle Ad-hoc Networks (VANET) that allows the perception the whole VANET system as a simple distributed database system. However, the proposal assumes the existence of fixed nodes in the network called Road Site Units (RSU).

In Liao et al. (LIAO *et al.*, 2013), the authors propose a middleware called Uno, which separates the storage of physical data and their associated metadata. In the proposal, the user's physical data is locally stored in the user's device, while their metadata is stored in commercial cloud platforms. With this in mind, the article's focus is to assure the user's data privacy, so it does not handle any mechanism to keep the data available, in case the user's device is not online on the network.

In Feki et al. (FEKI *et al.*, 2014), the authors propose a data replication algorithm based on Q-learning in a Peer To Peer(P2P) framework for highly dynamic environments. The proposal, when compared to traditional replication methods, presents good results.

In Narendra et al. (NARENDRA *et al.*, 2015), the authors present a decentralized cloud-based storage solution, specifically tailored for IoT data. The proposed solution uses object storage (such as Ceph) for Software Defined Storage (SDS), and optimal data distribution among distributed mini-Cloud. However, the solution assumes the existence and availability of devices located at the edge of the network, capable of running the SDS platform.

Shwe et al. (SHWE; CHONG, 2016) and Kumar et al. (KUMAR *et al.*, 2016) propose an architecture that distributes the data in two layer architectures. A local layer that performs a pre-processing of data, and later replicates them to a layer that is centralized in the cloud. Although the proposed addresses a relevant problem in the IoT area, the work does not address the persistence of data at the edge of the network, considering the dynamic environment of Mist computing.

In Confais et al. (CONFAIS *et al.*, 2016), the authors evaluate - through performance analysis - three "off-the-shelf" object store solutions, namely Rados, Cassandra and InterPlanetary

Table 5 – Comparation of related work for RQ2.

Paper Author	Proposal	Architecture	Pre-defined rules for nodes	Network context
Piotrowski et al. (PIOTROWSKI *et al.*, 2009)	A collaborative data storage (tinyDSM) middleware	Decentralized	Yes	WSN
Lieskovsky et al. (LIESKOVSKY *et al.*, 2011)	Solution for data distribution and replication for VANET using fixed nodes in the network called Road Site Units(RSU)	Decentralized	Yes	VANET
Liao et al. (LIAO *et al.*, 2013)	Separate the storage of physical data and their associated metadata. The physical data are stored locally and their metadata are stored in cloud platforms	Centralized	Yes	Cloud and Fog
Feki et al. (FEKI *et al.*, 2014)	Data replication algorithm based on Q-learning using Distributed Hash Tables in a P2P framework for highly dynamic environments	Decentralized	No	General networks
Chaqfeh et al. (CHAQFEH *et al.*, 2014) and Chen at al. (CHEN *et al.*, 2008)	Presents a survey about the data dissemination techniques in VANETs	Not applicable	Not applicable	VANETs
Narendra et al. (NARENDRA *et al.*, 2015)	Decentralized cloud-based storage solution specifically tailored for IoT data based on defined storage software (SDS) platform	Yes	Cloud and Fog	Not specified in paper
Shwe et al. (SHWE; CHONG, 2016) and Kumar at al. (KUMAR *et al.*, 2016)	Proposes an architecture that distributes the data in two layer architectures. A local layer which performs a pre-processing of data, that later replicates them to a layer that is centralized in the cloud	Centralized	Not applicable	Fog and Cloud
Confais et al. (CONFAIS *et al.*, 2016)	the authors evaluate through performance analysis three ”off-the-shelf” object store solutions, namely Rados, Cassandra and InterPlanetary File System (IPFS) in Fog context environment	Decentralized	Yes	Fog and Cloud

Source – Author.

File System (InterPlanetary File System (IPFS)) in Fog context environment.However, the used software platforms need devices with some computation processing power, so it is not applicable for simple devices, often found it in IoT environment.

In Chaqfeh et al. (CHAQFEH *et al.*, 2014) and Chen et al. (CHEN *et al.*, 2008), the authors conduct a survey on the data dissemination techniques in VANETs, presenting the main algorithms and protocols used in the area. On the first paper, the authors classified the data dissemination methods in three types: the push, the pull or the hybrid models exploring the main features, as well as the advantages and the disadvantages of each model. Similarly, on the second article, the authors classified the approaches in topology-based and location-based, mainly exploring the existing protocols, and challenges associated with the research area. Although there are similarities with the problems approached in this article, the techniques presented in the article use specific characteristics of the VANET environment, making it difficult to apply directly in the context of Fog / Mist computing.

In Table 5, a brief comparative summary of the related work is presented. In particular, the main differences of this work - compared to those mentioned before - are the use of Bio-inspired strategies for data dissemination, using a P2P architecture in a decentralized way with device, assuming dynamic rules depending on topology and time of permanence in the network. Another key point is that the protocol and algorithm proposed here can be performed in devices with tiny processing power, like sensor and other IoT devices.

3.3 Recommender systems for IoT environment (RQ3)

In (FORESTIERO, 2017), the author presents a multi-agent algorithm that, by exploring a decentralized and self-organizing strategy, builds a distributed recommendation system in the IoT environment. In the proposal, the ”things” are associated with an agent who exchanges information of things descriptors between them. A bio-inspired strategy, using a similarity measure between descriptors, results in generating geo-distributed clusters, in which each cluster concentrates similar descriptors. A device that looks for information from similar devices makes a query for those clusters, obtaining the recommendations of similar objects as a response. In the proposal, the author does not deal with the allocation of resources and requires a relatively long time to stabilize the construction of the clusters.

In (NIZAMKARI, 2017), the author addresses the same problem proposed in this thesis of service allocation in the IoT environment, also using collaborative filters. In the proposal,

the author uses a trust model graph approach combined with a collaborative filter(CB). The work, although very interesting, does not address the problem of network topology dynamics - very common mainly in the environment of Mist computing. Another issue that is not addressed is the strategy of how to keep the data of the CF in the locality of the network environment.

In (ASIRI; MIRI, 2016), the author proposes a model of trust and reputation in the IoT environment that applies distributed Probabilistic Neural Networks (PNN), performing as classifiers of the reliability of nodes in two classes: trusted nodes and malicious nodes. The model has the ability to learn over time, making use of fully distributed processing, and it is manipulated by the nodes themselves which ensures better availability. However, the proposal requires devices with higher processing power and connection to the grid, in a network with few topology changes and domain subject which is not a computational resource allocation.

In (SAWANT *et al.*, 2017), the authors propose a basic IoT Software Architecture and CPSs, and a system that intends to provide notifications through email or Short Message Service (SMS), as well as a recommendation to end users. The work is still at a very early stage, lacking details, for instance where the architecture would be executed, but it is assumed that the architecture is centralized and not handling heterogeneity and dynamics of the network topology.

In (MUNOZ-ORGANERO *et al.*, 2010), the authors propose that the Collaborative Recommendations in an Internet of Things environment relies on user-to-object space-time interaction patterns. The proposal assumes a centralized architecture where users with mobile devices rating ”things” associated with Near-Field Communication (NFC) TAGs, by sharing these assessments on a central server that processes the data and generates recommendations based on the user and the temporal patterns. The work has a user-centered focus and centralized architecture, thus not addressing issues related to infrastructure located at the edge of the network.

In (CHEN, 2017), the work has a well-applied character in the context of tracking within the supply chain of the food industry. In the proposal, the authors seek to add aspects of business in the recommendation system, using Value Stream Mapping (VSM) as an effective method for eliminating waste - which enables a company to map the process flow to redesign value streams with short lead time, reducing then inventories across organizational boundaries. However, the proposal does not explore all benefits of available computational resources at edge network, using the fog computing and Cyber-Physical System (CPS) only for data acquisition, user interface, and system actuators. Thus, in the authors proposal, the recommendation system runs in a centralized architecture.

In (LI *et al.*, 2017), the authors present a proposal for a vehicle selection algorithm within a Vehicular social network, which is applied in the domain of support marketers in improving marketing effectiveness. The proposal selects Recommenders for Vehicular Social Networks (SV-VSN) based on the coverage criteria of the area under review, as well as benefits for marketer, showing that the approach establishes significantly improvements when compared to similar methods in the state of the art.the authors present a proposal for a vehicle selection algorithm within a Vehicular social network, applied in the domain of support marketers in improving marketing effectiveness. The proposal selects Recommenders for Vehicular Social Networks (SV-VSNs) based on the coverage criteria of the area under review, as well as benefits for marketer, showing that the approach improves significantly improvements when compared to similar methods in the state of the art. Again the work uses the IoT devices, in this case, the cars are just a data collector/output in the context of the application, focusing only on the method of selecting the vehicle to be used as Recommenders in SV-VS. Nevertheless, in the scope of work there was nothing related to the dynamics of the topology and service sharing, since the proposal only identifies data sharing.

In (ZHANG *et al.*, 2013) and (WEI *et al.*, 2017), the authors address the cold start problem - common to collaborative recommendation systems, especially in the context of IoT the Cold-start Recommendations Using Collaborative Filtering (CRUC) problem - Cold-start Recommendations Using Collaborative Filtering in IoT. In general, the cold start issue is characterized by two aspects: scalability and dispersion. The first resource is caused by the number of crawled objects and their interaction data. IoT uses sensors to track a large number of objects as well as their interaction. Thus, the IoT gathers extra-large object interaction data. This implies that IoT fails to respond quickly to applicants. The last resort arises from the density of information. Although IoT receives a large number of sensor readings on objects, it acquires little information about a specific object, particularly on a cold start situation. Correspondingly, it is a non-trivial task to predict the user's preference based on cold start IoT systems. Currently, this problem is still open with few proposals on this subject. In the first work (ZHANG *et al.*, 2013), the author proposes a CRUC scheme, consisting of two phases: one offline and one online. In the off-line phase, CRUC reshapes and filters the available data by pre-processing them for the next phase. Secondly, in the online phase, the CRUC enters a prediction stage that aims at correcting or lightening the CRUC problem.

In (MASHAL *et al.*, 2015), (MASHAL *et al.*, 2016a) and (MASHAL *et al.*, 2016b),

in the first manuscript, the authors conceptualize the recommendations of IoT services IoT Service Recommendation (IoTSRS), showing their importance in the context of IoT. The second and third works focus on the study of how to model the problem and how to evaluate and analyze metrics for IoTSRS. The papers also propose a formal hypergraph model to represent the IoT recommendation system in which each hyper-edge connects users, objects, and services. In the second article, the same formal model of IoTSRS is again presented, and a comparative study is made of the usefulness of traditional recommendation schemes and their hybrid approaches in recommending IoT services (IoTSRS) - based on well-known existing metrics. The work focuses on a theoretical analysis and on the study of metrics for the evaluation of IoTSRS, being good references for the evaluation of the results of this thesis. However, it does not address issues related to the research questions of this study.

In (YAO *et al.*, 2014), the author propose a unified probabilistic based framework by fusing information across relationships between users and things, in order to make more accurate recommendations. However, again the work does not address issues about network topology dynamics nor where the model will be deployed to provide the recommendations.

In (RICCI, 2010), the author makes a survey about the major issues and opportunities that the mobile scenario opens to the application of recommender systems, especially in the area of travel and tourism. The paper covers the major techniques and computational models that have been proposed, presenting some possible future developments and extension in this area. However, the author focuses heavily on mobile devices, exploring very little the possibilities generated by the IoT environment.

In (CHA *et al.*, 2016), the authors present a recommendation system in the IoT environment using a mobile device as a sensing device and user interface connected to a server at the Cloud, which in real time suggests recommendations according to the detected context by the mobile device, presenting the recommendations at the mobile device screen. To demonstrate the feasibility of the proposal the author presents a prototype of a tourism application recommendation system. However, the proposed architecture distributes the processing between the mobile device and the recommendation server in the cloud computing environment, thus taking no advantage of the infrastructures located at the edge of the network.

In (CUOMO *et al.*, 2017), the authors propose a mathematical model of a collaborative recommendation system applied in the field of Associating Visitors and Artworks in a Cultural scenario, using mobile devices and IoT Bluetooth sensors to detect the behavior of the

visitors during the visit. Detecting IoT sensors allows the system to classify visitors and artworks. In the case of visitors, the sensors can inform the route traveled, artwork visited and time spent interacting with the interaction of the artwork. According to their behavior pattern, the system classifies users and develops an artworks' rating based on the behavior pattern of visitors during the visit. However, the proposal uses a centralized architecture using IoT devices only as data collectors and user interface. In the proposal, all devices are similar, that is, not heterogeneous and do not use external computational resources.

In (FREY *et al.*, 2015), the authors propose an approach to a recommendation system using people's smartphone for capturing information such as the installed apps, interactions with IoT devices, location information among others. With that, they elaborate a digital inventory that intends to be used to produce a more refined user profile, thus through hybrid collaborative recommendation system, to propose more accurate and useful recommendations to the user. The paper presents only the concept and gives the potential of exploring such data. However, the novelty is limited since in some ways it is not a new idea; big companies like Google and Facebook have already done this using the web. Also, the article does not explore computing resources at the edge of the network by simply using the devices like information gatherers.

In (RENJITH; ANJALI, 2014), the author proposes a hybrid collaborative recommendation system applied to the tourism domain, in particular, travel recommendations using content-based data, collaborative assessments, and demographic information.

In (RAMASWAMY *et al.*, 2009), the authors propose a hybrid collaborative recommendation system that combines content information about users, spatial-temporal context information, and rating made by users in the past. The work has the merit of having been one of the first proposals in this line of research to use mobile phones, but it does not fit the environment of Edge computing.

Tables 6 and 7 show the works related to the third research question(RQ3) of this thesis. Table 6 shows the works with a strong connection with the theme of the thesis. In particular, the works in the same line of research and similar proposal are highlighted in light green color. Furthermore, Table 7 presents work related to the research question, however not directly related to the subject of this thesis, such as survey works and works that address issues relevant to RQ3 in the network environment, the focus of the study.

Table 6 – Comparation of related work directly related to the thesis theme (RQ3).

Paper Authors	Proposal	Architecture	Topology dynamism	Temporal dynamism	Node rules	Local data	Network context	Focus User/Things
(MUNOZ-ORGANERO *et al.*, 2010)	Collaborative Recommendations in an Internet of Things environment rely on user-to-object space-time interaction patterns	Central	No	Yes	Fixed	No	F/C	User
Asiri e Miri (2016)	a model of trust and reputation for IoT using distributed probabilistic neural networks (PNNs)	Dist	No	No	Fixed	No	Fog	Thing
Forestiero (2017)	geo-distributed clusters of similar descriptor along the topology	Dist	Yes	No	Fixed	Yes	F/M	Thing
Nizamkari (2017)	CF combined with trust graph for provide service recommendations	Dist	No	No	Fixed	No	F/M	Thing
(SAWANT *et al.*, 2017)	Basic architecture of IoT and CPSs which provides recommendation services to the end users	Central	No	No	Fixed	No	F/C	User
(CHEN, 2017)	Traceability using IoT applied to the food supply chain combined with a recommendation system using Value Stream Mapping (VSM)	Central	No	No	Fixed	No	F/C	User
(LI *et al.*, 2017)	vehicle selection algorithm within a Vehicular social network applied in the domain of support marketers in improving marketing effectiveness	Central	No	No	Fixed	No	VANET	Thing

Source – Author.

Table 7 – Comparation of related work that are not directly related to the thesis theme (RQ3).

Paper Authors	Proposal	Architecture	Topology dynamism	Temporal dynamism	Node rules	Local data	Network context	Focus User/Things
(RICCI, 2010)	survey about the major issues and opportunities that the mobile scenario opens to the application of recommender systems, especially in the area of travel and tourism	NA	NA	NA	NA	NA	F/C	NA
(YAO *et al.*, 2014)	A unified probabilistic-based framework by fusing information across relationships between users and things to make more accurate recommendations	Central	No	No	Fixed	No	F/C	Thing
(MASHAL *et al.*, 2015), (MASHAL *et al.*, 2016a), (MASHAL *et al.*, 2016b)	a formal hypergraph model to represent the IoT recommendation system, in which each hyper-edge connects users, and a comparative study is made of the usefulness of traditional recommendation schemes and their hybrid approaches in recommending IoT services (IoT-SRS) - based on well-known existing metrics	NA	NA	NA	NA	NA	Fog	NA
(CHA *et al.*, 2016), (CUOMO *et al.*, 2017), (RENJITH; ANJALI, 2014), (FREY *et al.*, 2015), (RAMASWAMY *et al.*, 2009)	Centralized Recommendation System application using smartphones to collect context information and user's ratings.	Central	No	No	Fixed	No	F/C	User
(ZHANG *et al.*, 2013) and (WEI *et al.*, 2017)	address the cold start (CRUC) problem common to collaborative recommendation systems and especially in the context of IoT	NA	NA	NA	NA	NA	Fog	NA

Source – Author.

3.4 Summary

In this chapter, first, works related to the IoT environment selection (RQ1) were presented. In particular, this chapter discussed 8 studies related to this issue as well as a more detailed discussion and identification of the main gaps found in these works.

With regards to the Data distribution in the FMC environment (RQ2), 9 studies related to data dissemination in Fog/Mist network environment or similar were found. All works were discussed and the main gaps found in these works were identified.

Finally, works related to Recommender systems for the IoT environment (RQ3) were also presented, with focus on the hypothesis raised in this thesis, and 19 studies were discussed. Once more, the identified gaps were presented, along with the strengths and weaknesses of each study.

The next chapter presents the thesis proposals for supporting the gaps identified in the three research questions of this thesis. The gaps are summarized as follows:

- Few systematic approaches to choose the best computational infrastructure (Fog, Mist, or Cloud) to use computational resources;
- Infrastructure choice methods usually have fixed parameters constraints like latency, consumption or others to choose the best environment. In general, these methods are not easily adapted to other constraints;
- The vast majority of infrastructure choice algorithms have a centralized architecture;
- Proposals related to RQ2 typically assume fixed roles for devices in the data dissemination process;
- Considering the works related to RQ3, all of them assume a centralized architecture or, when unpublished, the nodes have predefined roles that do not allow a dynamic adaptation of the infrastructure when there are no nodes of a specific type; and
- None of related work of RQ3 addresses the temporal availability of the devices in the environment to calculate the best recommendation.

4 SMART SHADOW

This chapter presents the solutions for the main gaps identified in the research questions (RQ1, RQ2, and RQ3). For RQ1, a method to systematically select the best computing resource available on FMC environment that attend client devices constraints is proposed. Related to RQ2, a mechanism for data dissemination in a highly dynamic FMC topology environment is proposed . Finally, related to RQ3, a mechanism for computational resource prediction is proposed using a hybrid collaborative filter combined with some temporary availability statistics.

These solutions are detailed in this chapter as follows. Section 4.1 presents an overview of the relationship and dependencies between the proposed solutions associated with the research questions. Section 4.2 describes the proposed mechanism, algorithm and concepts in order to choose systematically the best computational resource available in the FMC environment that meets the requirements specified by the client device. Section 4.3 presents the proposed bio-inspired method of data dissemination in the FMC environment based on epidemic models running in an adaptive hierarchical architecture by the nodes that are part of the edge of the network. Section 4.4 describes the proposed mechanism of prediction of the best computational resource available using hybrid collaborative filtering combined with temporal availability estimator. Finally, Section 4.5 concludes this chapter.

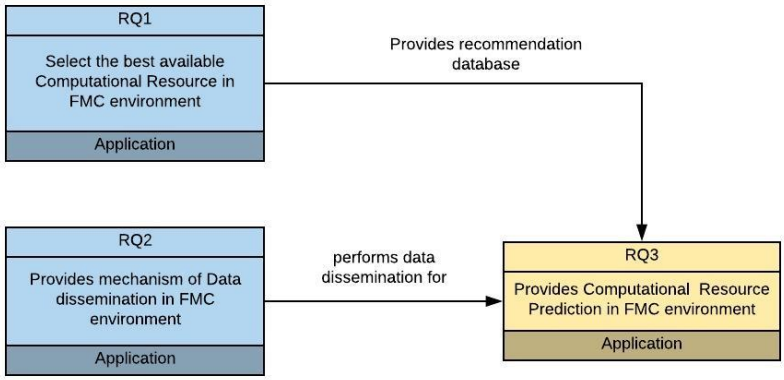
4.1 Overview

A relevant point for understanding the proposals of this thesis is to understand how the research questions interrelate. Figure 13 illustrates this relationship between RQ1, RQ2 and RQ3.

The first research question (RQ1 - *How one can evaluate systematically the feasibility of using computational resources available in the FMC computational infrastructure?*) generally seeks to identify the best device or infrastructure available that meets requirements imposed by a client device located at the edge of the network. The proposal associated with this research question presents a computational resource search algorithm based on the depth-first search of graph theory guided by a function of an estimate of the fulfillment of the conditions imposed.

The second research question (RQ2 - *How to store information in an FMC environment, considering the ephemerality of device availability in this environment?*) addresses the problem of how to disseminate data in a highly dynamic topology network environment.

Figure 13 – Research questions relationship.



Source – Author.

The proposal for this issue was to use data dissemination techniques based on epidemiological models combined with a decentralized adaptive hierarchical architecture in which the nodes assume roles in this structure according to their average time of permanence in the environment.

Finally, the third research question (RQ3 - *What would be a way to automatically capture knowledge of the discovery and use of computational resources in the Fog / Mist computing environment to make discovery and allocation predictions efficiently and robustly?*) addresses the problem of computational resource prediction within the FMC environment. To approach this problem a mechanism that uses hybrid collaborative filtering combined with statistics of the temporal availability of the devices was proposed. However, collaborative filtering needs to provide useful results of a good computational resource allocations historical data. It also needs that these data capture the good computational resources available in the environment at the time. Thus, the RQ1 research question becomes important in the objective of generating a mass of quality data that allows the prediction mechanism to generate good results. Another relevant point in the relationship between RQ1 and RQ3 is that the latter is a statistical prediction mechanism. Therefore, there is the possibility of error, that is, the possibility that the device indicated by the algorithm is not available at that moment in the environment. Once again, RQ1 has a relevant role serving as an alternative in cases in which the prediction algorithm was not successful.

Different from RQ1, which somehow competes with RQ3 because both have the recommended device for providing the computational resource as output, RQ2 actuates as a support service for RQ3 since a recommendation system requires parameters and data from past ratings. Thus, RQ2 provides this service supporting RQ3, allowing the prediction mechanism to

work in a decentralized and autonomous manner.

4.2 Network Infrastructure Selection (RQ1)

The Fog Computing approach brings advantages associated with latency, local proximity and extending battery life (BRUNEO *et al.*, 2016). However, it also brings several challenges related to the heterogeneity of the devices that compose it, as well as the transience of their permanence in the network. Thus, the choice of which infrastructure a device that requires external computational resources should choose is not always simple. Thus, this section addresses the Problem of Allocation of Resources (PAR) considering the computational resource demand, and the constraints imposed by the client device (CD), aiming at answering how to systematically identify the best network infrastructure capable of meeting demand between: Cloud Computing, Fog computing or Mist computing. The PAR is modeled in the form of a weighted graph that, through a Disjkra-based algorithm, generates a ranking of the feasibility of provision of the service or resource sorted by an estimator.

This model uses, as input parameters, the restrictions imposed by the client device, the topology of the devices at the edge of the network, the connection parameters between devices (latencies, Baud rate and the signal level of Peer To Peer (P2P) links), and CoTs access parameters (cost, communication bandwidth, latency). For example, client parameters may be the financial spending limit with CoTs services, maximum latency requirements, minimum communication link speed, or weights associated with each constraint in decision-making. These parameters are used to identify devices capable of providing services and also to calculate weights of the edges of the modeling graph.

Define U as the set of all devices located at the edge of the network with physical proximity of the device D that needs to request computational resources. Therefore, the PAR can be modeled as a graph G, where the vertexes (v) of this graph are elements of the set U and the edges are the communication links between them. In present approach, vertexes are classified according to the following types.

The client device (D) is defined as a device that demand remote computing resources from Cloud/Fog or Mist computing infrastructure. Computational resources required by the device comply with a set of restriction requirements

$$r = \{r_1, r_2, ..., r_k \mid k \in \mathbb{N}, \text{where } \mathbb{N} \text{ is the set of natural numbers}\}.$$

As an illustration, the following examples of restriction requirements: minimum link speed; run time; maximum latency and cost associated with the use of the resource. They are represented in our model as the unitary set:

$$D = \{v \in U | \text{ with requirements } r_1, r_2, ..., r_k \text{ and } k \in \mathbb{N}\}$$

Fog computing devices are defined as all devices located on the edge of the network with direct and stable access to the Internet, and usually with good capability of computational resources. They are represented in the model as the set of vertexes:

$$F = \{v_i \in U \mid i, n_F \in \mathbb{N}, \text{ where } n_F \text{ is the numbers devices and } 1 \leq i \leq n_F\}$$

Mist computing devices are also located on the edge of the network, but with indirect access to the Internet and limited computational resources. They are represented in the model as the set of vertexes:

$$M = \{v_i \in U \mid i, n_M \in \mathbb{N} \text{ , } i \in \text{ and } 1 \leq i \leq n_M\}$$

Where, n_M: is the number of vertexes that satisfy this condition.

Available CoTs - given U_c the universe of all CoTs, let C be defined as set of available CoT the subset of U where each element provides services for the client device D or for any device of the Fog set. Therefore, in the model, they are represent as the set of vertexes:

$$C = \{c_i \in U_c \mid i \in \text{ and } 1 \leq i \leq n_C\}$$

$\forall c_i \in C$, c_i provides service for D, union with $\forall c_i \in C$, where c_i provides service for some $v_i \in F$

Where, n_C: is the number of available CoT to the client device plus available CoT for the Fog computing device set. In the model, the vertexes that represent the Cloud computing are necessarily connected to vertexes of Fog Computing (F) or directly connected to the client device (D). Thus, the graph G is defined as:

$$G = (V, E)$$

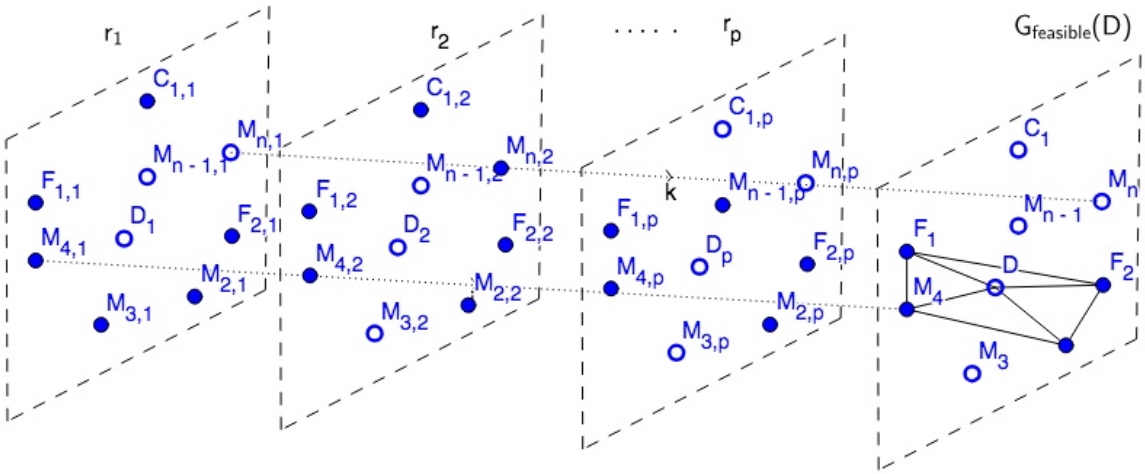
Where V is the set of vertexes formed by the Union of the sets described before, or formally:

$$V = \{v_i \in U \mid v_i \in D \cup F \cup M \cup C\}$$

And E is the set of edges connecting the vertexes of the set V. Let to define the existence of an edge $e_{i,j}$ between vertexes v_i and v_j, if $\exists$ is the communication link between the vertexes v_i and v_j. Given that E_T is the set of all possible edges in the set of vertexes V, E is defined formally as:

$$E = \left\{ e_{i,j} \in E_T \mid e_{i,j} = (v_i, v_j) \exists \text{ link between } v_i, v_j \in V \right\}$$

Figure 14 – Set of layers with devices that meet the constraints r_k combined to generate the $G(D)_{feasible}$.



Source – Author

4.2.1 Definitions

In the proposed model, not all vertexes of the graph G can provide and meet the set of requirements $r = \{r_1, r_2, ..., r_k \mid k \in \mathbb{N}\}$ imposed by device D. To clarify the concept, a function $\alpha_{r_j}(v_i)$ is defined, for each requirement $r_j \in r$, which has as its domain the set of vertexes that are part of G, and as codomain the set $I = \{0, 1\}$, where $\alpha_{r_j}(v_i) = 0$ if the vertex v_i cannot attend the requirement r_j, and $\alpha_{r_j}(v_i) = 1$ if the vertex v_i can meet the requirement r_j.

Definition 4.2.1 *Thus, let's define that the vertex v_i attends the requirement $r_j \leftrightarrow \alpha_{r_j}(v_i) = 1$. Formally represented in the model as:*

$$\alpha_{r_i} : \quad V \quad \rightarrow \quad \mathfrak{N}, \; 1 \leq i \leq k$$

$$\alpha_{r_j}(v_i) = \begin{cases} 0, & \text{if } v_i \text{ does not meet requirement } r_j \\ 1, & \text{if } v_i \text{ meets requirement } r_j \end{cases}$$

where r : Resource constraint

k : Number of constraints

Definition 4.2.2 *Defining the feasible neighborhood graph of device D or $G(D)_{feasible}$ as the subset of G, where all vertexes of $G(D)_{feasible}$ meet all requirement constraints imposed by device D, $G(D)_{feasible}$ can be formally represented as:*

$$G(D)_{feasible} = (V_{feasible}, E_{feasible})$$

$$where, V_{feasible} = \left\{ v_i \in V \Leftrightarrow \prod_{j=1}^{k} \alpha_{r_k}(v_i) \neq 0 \right\},$$

where, $1 \leq i \leq n^*$, $1 \leq j \leq k$ *and* $E_{feasible} = \left\{ e_{i,j} \in E \Leftrightarrow \exists \text{ communication link between } v_i \text{ , } v_j \right\}$,

where, $v_i, v_j \in V_{feasible}$ *and* n^* *is the number of vertexes of G*

that attends the requirements imposed by D.

Figure 14 illustrates the construction of graph $G_{\text{feasible}}(D)$. Let's first consider that the FMC environment consists of the sets by vertexes $\{F_1, F_2, F_3, F_4, F_5\} \in F$(Fog), $\{M_1, M_2, M_3\} \in M$(Mist), and $\{C_1\} \in C$(Cloud). Let's also consider that the set $r = \left\{ r_1, r_2, ..., r_p \mid p \in \mathbb{N} \right\}$ is the set of constraints imposed by device D. The first step to build the feasible neighborhood graph of device D, or $G_{\text{feasible}}(D)$, is for each constraint r_p to identify which devices can able to meet it. This process is represented in Figure 14 by the planes $r_1, r_2, ..., r_p$, where the vertexes $v_i s$ are filled with black if they meet the constraint r_p, and unfilled otherwise. An extra index on the vertex was also added to indicate which constraint is referenced. For example, the vertex $C_{1,1}$ means vertex C_1 submitted to constraint r_1. After repeating this process for each constraint r_p, $G_{\text{feasible}}(D)$ is the set of vertices that have met all constraints. To put it differently, the vertexes that are filled in the planes associated with all constraints. In Figure 14, these vertices are shown in the plane $G_{\text{feasible}}(D)$, in the rightmost layer, where the vertices are filled and are connected to each other. An important point that must be observed in the proposed model for the construction

of G is that all vertices that are part of the cloud set must have a connection with all vertices that have access to the internet.

Moreover, it is important to realize that some restrictions in the application directly depend on the route between the device and the vertex in focus. Thus, to facilitate the analysis, it is convenient to define a function that encapsulates this complexity depending on the path taken in the graph G. From the graph theory, it can be said that:

Definition 4.2.3 *The open path from v_1 to v_k is a finite sequence of the form*

$$\left\{v_1, e_{1,2}, v_2, e_{2,3}..., v_{k-1}, e_{k-1,k}, v_k\right\},$$

that consists of alternating vertexes and edges of G on the condition that any vertex is visited at most once and $v_1 \neq v_k$.

Using this concept, it can be defined:

Definition 4.2.4 *The extended restriction function of r_j of the vertex v_i in $G(V,E)$ related with vertex D, namely $\Phi_{r_j}(p_{min}(v_i,D))$, is defined as a function that has the set P, defined as the set of all possible routes from D to v_i in graph $G(V,E)$, as its domain and the set of $\mathbb{R}$ as codomain. Besides that, the value of $\Phi_{r_j}(p_{min}(v_i,D)))$ encapsulates the effect of all paths from D to v_i related with r_j on the condition that the $p_{min}(v_i,D)$ is the path between the vertex v_i and D, where the effect of constraint r_j is minimum.*

$$p_{min} = \left\{v_i, e_{i,j}, v_j, .., e_{k,D}, D\right\}$$

$$\Phi(p)_{r_k}: \quad P \quad \rightarrow \quad \mathbb{R}$$

The function $\Phi_{r_j}(p_{min}(v_i,D))$ depends on the type of constraint. For example, considering the maximum latency constraint and given that $p_{min}(v_i,D)$ is the path of vertex between D and the vertex v_i in question, and $\Phi_{r_j}(p_{min}(v_i,D))$ the function extended latency from D and the vertex v_i, a function that best models the problem could be the function sum of the latencies of all edges that are part of $p_{min}(v_i,D)$, namely minimum path from D to vertex v_i. Formally:

$$\Phi(p_{min})_{\text{Latency}} = \sum_{j=1}^{n} \text{Latency}(e_j),$$

$$\mid p = \{v_1, e_1, v_2, .., e_{n-1}, v_n\}, v_1 = D \text{ and } v_n = v_i$$

Where n is the number of edges of path $p_{min}(v_i, D)$. Conversely, let to consider the communication speed link restriction, namely $\Phi(p_{min})_{\text{Speed}}$, the best function to model this constraint should be the minimum function of all speed link of all edges in the minimum path from D to v_i. Thus, formally:

$$\Phi_{Speed}(p_{min}(v_i, D)) = \min((e_{speed_1}, e_{speed_2}, .., e_{speed_n}).$$

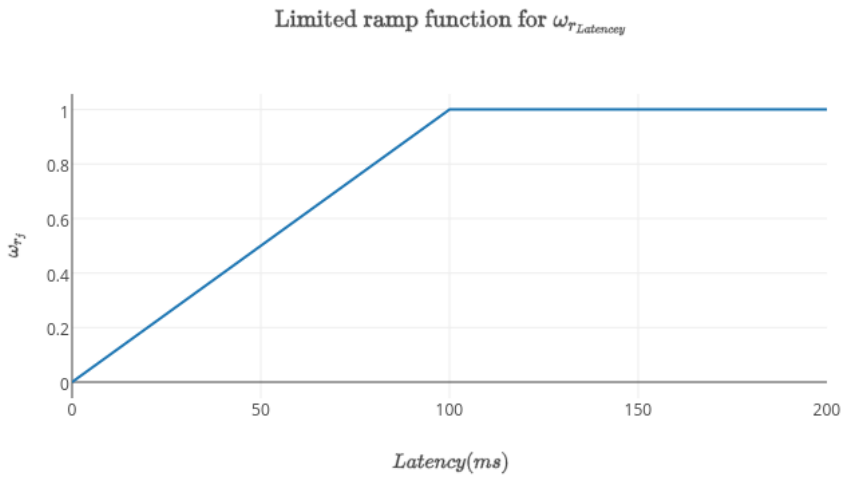
Therefore, to decide whether $v_i \in G$ can meet constraint requirements r_j, a function $\Phi_{r_j}(p_{min})(v_i, D)$ must be defined so that it models the problem and uses its value as decision criterion to attend the restriction requirement. To clarify the decision criteria and help to combine the result with other constraints imposed by device D, a function $\omega_{r_k}()$ is defined, called resistivity restriction for r_j, which uses the extended restriction function of constraint r_k as its input parameter. This function provides an estimate of the degree of difficulty of the device meet the specified constraint. Thus, formally:

Definition 4.2.5 *Define $\omega_{r_k}(\Phi_{r_j}(p_{min}(v_i, D)))$, or simply $\omega_{r_k}(x)$ as a function of $x \in \mathbb{R}$ and as codomain the interval $[0,1] \in \mathbb{R}$, where 1 means it is unfeasible to attend restrictions r_k, and 0 means it is feasible to attend restrictions r_k.*

$$\begin{aligned} \omega_{r_j}: \ \mathbb{R} &\rightarrow \mathbb{R} \\ \mathbb{R} &\rightarrow [0,1] \end{aligned}$$

Note that this definition purposely omits to define the type of function to be used. The reason is that any function that meets the requirement description can be chosen and is able to model it in the analysis. In other words, the function must assume the value zero when the device has full capability to attend the constraint. Otherwise, the function must take the value 1 when the device is completely unable to attend that restriction. For instance, let r_j be the constraint of maximum latency of 100 ms, the function ω_{r_j} could be chosen as the limited ramp function shown in Figure 15. In case the latency is zero, the ideal case, the function has value 0, meaning the situation of maximum feasibility, and for any latency above the maximum latency (100ms), the function takes value 1, showing complete unfeasibility. For latency between 0 ms and 100 ms, the function reflects a feasibility grade.

Figure 15 – Example of a limited ramp function for $\omega_{\text{Latency}}(p_{min}(v_i, D))$.



Source – Author

Definition 4.2.6 *A function $\lambda_{r,v_i,D}(\omega_{r_1}, \omega_{r_2}, ..., \omega_{r_k})$ will be defined, as resistivity of service for restrictions r related with D and the vertex v_i. The idea for this function is to aggregate the influence of all constraints imposed by D. Similarly to function $\omega_{r_k}(v_i, D)$, the function $\lambda_{r,v_i,D}()$ has the interval $[0,1] \in \mathbb{R}$ as its codomain. In the same way, when it acquires value 1, it means it is fully unfeasible to attend the set of restrictions r, and 0 value means total feasibility to attend the set of restrictions r. However, differently from $\omega_{r_k}(v_i, D)$ function, its domain is $\mathbb{R}^k$, where k is the number of individual constraints of r. Given $\alpha_1, \alpha_2, ...\alpha_k$, the weights of the importance of D respectively the restrictions $r_1, r_2, ..., r_k$ such that*

$$\sum_{i=1}^{k} \alpha_i = 1$$

Thus, it can be defined:

$$\begin{array}{rccc} \lambda_{r,v_i,D}: & \mathbb{R}^k & \rightarrow & \mathbb{R} \\ & (r_1, r_2, .., r_k) & \rightarrow & [0,1] \end{array}$$

$$\lambda_{r,v_i,D}(\omega_{r_1}, \omega_{r_2}, ..., \omega_{r_k}) = \prod_{j=1}^{k} (\omega_{r_j}(v_i, D))^{\alpha_j}$$

4.2.2 Proposed Model

Given the client device D in an FMC environment, F is located in the geographical proximity of D. Let us consider that the device D requests a remote computational resource rc,

but it is imposed that the resource must follow the restrictions set:

$$r = \{r_1, r_2, ...r_k\}.$$

Given also that $G(V,E)$ is the graph formed by the set IoT devices, where IoT devices are the set of vertexes (V), and the communication link between them represents the set of edges (E).

In the proposal, the first step of the problem is to identify the vertex

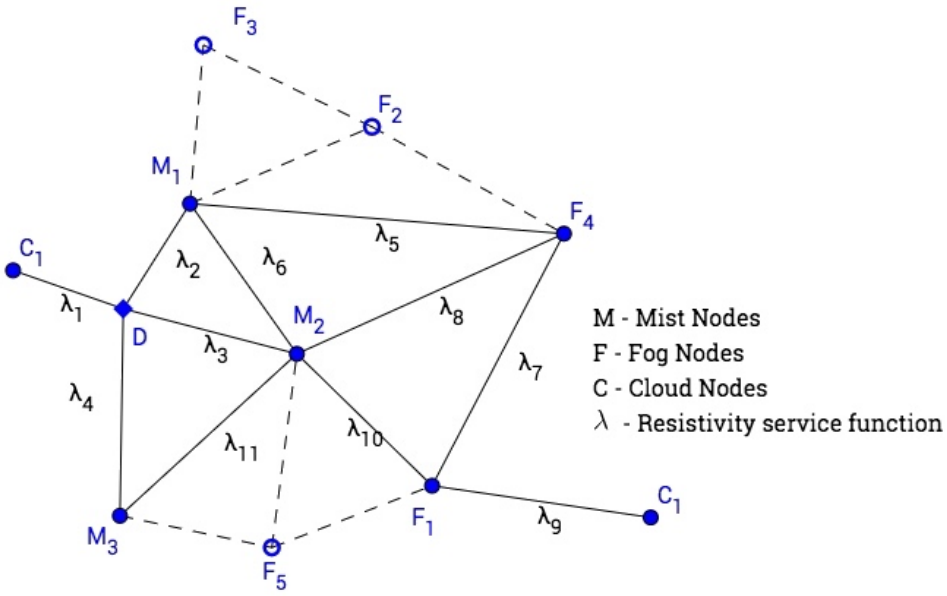
$$v_i \in G \,|v_i \text{ meets } r_i, \forall r_i \in r.$$

Considering that all vertexes of G are aware of the sets of r restrictions and, respectively, the weights that D gives to each constraint, the first step of the proposal is to discover the $G(D)_{feasible}$. Thus, each vertex $v_i \in G$ calculates the function $\Phi_{r_k}(p(v_i,D)$ for any restriction $r_i \in r$ and compares with the constraint imposed and published by D. Therefore, in the case that a vertex meets all restrictions, it belongs to the feasible graph of D.

Given that the feasible graph of D had been made, $G(D)_{feasible} = (V,E)$, it must be considered now the calculation of the weight of each path from D to $v_i \forall v_i \in V$. The weights of paths in the graph must be calculated using the function $\lambda_{r,v_i,D}()$ (resistivity of service for restrictions r related with D and the vertex v_i). Consequently, now it is possible to built an ascending ranking for all vertexes of $G_{feasible}$ based on the $\lambda_{r,v_i,D}()$ function value. This value may be interpreted as a metric of a vertex's capacity for providing the resource for D, where the lowest values mean the greater capability to provide the resource. Note that the calculation of $\lambda_{r,D}(p)$ itself does not guarantee the imposed service restrictions.

On the other hand, it would not be necessary, because for a vertex to belong to $G(D)_{feasible}$, it is necessary that this vertex meets all constraints using the extended constraint function, which, by definition, incorporates this warranty. Another important point to be noted is that, from the point of view of vertexes, the $\lambda_{r,v_i,D}()$ function encapsulates the complexity of the minimum path(p_{min}). Thus, each vertex only needs to choose the path that features the smallest value of the $\lambda_{r,v_i,D}()$ function.

Figure 16 – Graph of feasible neighborhood of D ($G(D)_{feasible}$).



Source – Author

Therefore, the PAR is reduced to select the first vertex v_i in $\lambda_{r,v_i,D}()$ ranking. Similarly, the problem of choosing of which environment should be used is now reduced in figuring out within which network infrastructure subset the top ranking vertex is part of:

- If $v_i \in F \rightarrow$ the choice is Fog computing;
- If $v_i \in C \rightarrow$ the choice is Cloud computing; and
- If $v_i \in M \rightarrow$ the choice is Mist computing.

Figure 16 illustrates the $G_{feasible}(D)$ and, respectively, the function calculations based on the examples showed in Figure 14.

4.2.3 Proposed Algorithm

This section shows a macro description of the algorithm and the communication protocol that was implemented on devices that are part of the network edge. The detailed algorithm and implementation are available on a Github repository.[1]

Assuming that all devices at the network edge have communication links between them and the vertexes, and communication links are distributed in a graph topology, the first step of the algorithm is to generate neighbors' feasible graph of the client device D. With this in mind, the device D needs to send information about the requested service, restrictions, and

[1] Avaliable source code in the address <https:https://github.com/GREatResearch/Contiki/>

priority weights of each constraint to other network devices.

The best way of sending this information to network avoiding message collision on the network and minimizing message traffic is using a token. This token will serve to store the requested service information and also to map the topology and parameters of the network.

For this purpose, the strategy of using a control token to avoid communication conflicts between the vertexes and synchronize messages is adopted. In this strategy, only the vertex which holds the communication control token may broadcast messages for the network. Particularly, broadcast messages are used to map the vertex's neighbors and to provide information to allow them to calculate if they are part of the feasible graph of D. Then, as soon as the controller vertex completes the identification of feasible neighbors, it forwards the token recursively to the child vertexes, similar to the depth-first search algorithm (DFS) (AWERBUCH, 1985) in graphs. Consequently, the token travels through all graph's vertexes and is also used to carry information about network's topology, network's parameters, and the minimum route from D to each vertex of the graph.

In the proposal, there are two cases where a vertex has the token ownership: either the vertex is the client device (D), or it is a device in its vicinity. In both cases, the procedure performed by the vertex is very similar. In the first case, the device(D) broadcasts a message containing: the required resource, the necessary restrictions (r), and the respective priority weights(α's). Hence, the vertexes on its neighborhood can calculate if they are part of the feasible graph and, if so, to answer their resistivity of service for restrictions r related with D in a peer to peer (P2P) message for device D. As a result, the device D receives and processes this message, registering in the token all neighbors that are feasible, altogether with measured network parameters and topology. Additionally, the device D puts the neighborhood vertexes in a queue in its memory.

Finally, the device D gets the first device in the queue and sends the token that restarts the mapping process. The algorithm terminates when the token returns to D and there are no more neighbors to visit. In the second case, the device will receive the token from device D, or another vertex similar to it, and do exactly the same procedure. However, when the queue contains no more vertexes to visit, the device hands back the token to its parent vertex.

The device chosen to provide the resource will be the one which showed the lowest resistivity of service for restrictions r($\lambda()$ function).

In short, given that the vertex received a broadcast message from the token owner, it

will perform the following steps:

1. First, it must compute whether it is able to meet the requirement constraint using the extended restrictions function of r ($\Phi_{r_k}(p_{min}(D, v_i))$ for each r_k constraint of r.
2. If the vertex is able to meet all restrictions, through a P2P message, it will notify the vertex which sent the broadcast message that it must be added to the queue of feasible neighbors of the parent vertex.
3. Otherwise, if the vertex is unable to meet any of the restrictions, it will report its parent vertex that is not going to be a part of the graph $G(D)_{feasible}$. Therefore, it will signal to parent to stop broadcasting the message M through its path.
4. On the other hand, the token owner device must wait for the answer of all its neighbors so that it passes the token to the first neighbor in its queue. The neighbor that receives the token restarts the process and gives back the token once all its children have completed the discovery of vertexes of feasible graph of D ($G(D)$).

It was defined as stop condition for the algorithm of calculation of $G(D)_{feasible}$, the situation where vertex D does not have any other neighbor vertex to explore. Thus, given the graph $G(D)_{feasible}$, the next step is to assign a score to each vertex $v_{i,j} \in G(D)_{feasible}$ based

Algorithm 1: Algorithm of vertex with token to build $G(D)_{feasible}$.

input : Requirements constraints and priority weights of D
output : Assembly of feasible neighborhood from $D(G(D)_{feasible})$

```
begin
1     D sends Message(r,weights) to neighbors;
      // identify neighbors
2     while (not (time out)) do
3         wait answer from neighbor;
4         if (vertex v meets all restrictions) then
              // Store path with minimum lambda for vertex
5             if (λ_{r,D}(p) received < λ_{r,D}(p_min) in token) then
6                 p_min ← p;
7                 λ_{r,D}(p_min) ← λ_{r,D}(p) received;
8                 store (v, λ_{r,D}(p)) in token;
9                 store (v, p_min) in token;
              end
10            putInQueue(vertex);
          end
      end
11    while (queue is not empty?) do
12        vertex ← getFromQueue(vertex);
13        send Token to vertex;
14        wait Receive token;
      end
      // Send Token back or stop
15    if (self is not D) then
16        send back Token to parent;
      else
17        stop condition achieved;
      end
end
```

Algorithm 2: Algorithm of vertex with no token to build $G(D)_{feasible}$.

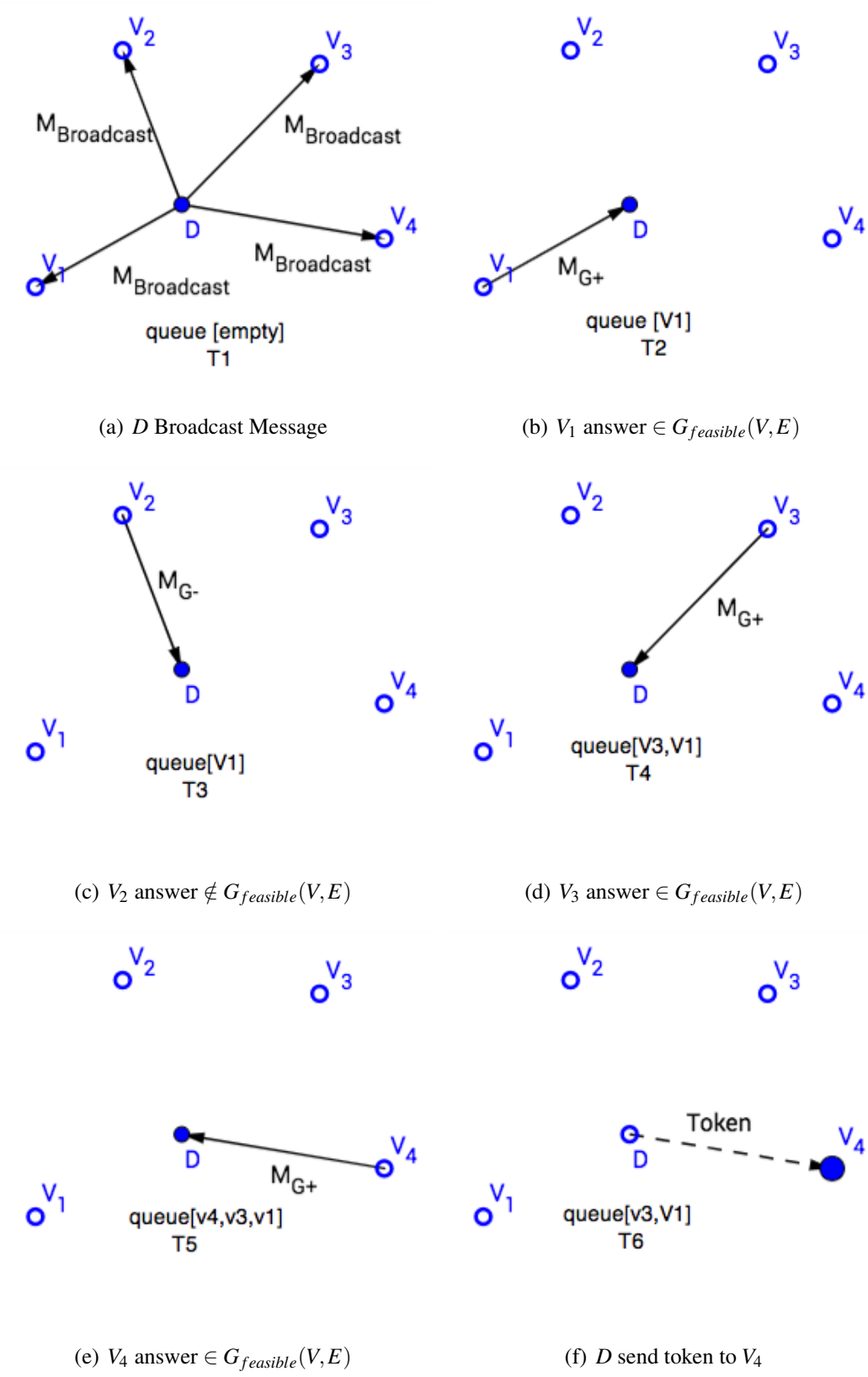
```
   input  : Requirements constraints and priority weights of D
   output : Assembly of feasible neighborhood from D (G(D)_feasible)
   begin
       // Wait to receive message M from vertex with Token
1      receive(M);
       // parent is the node that sent the message
2      parent ← sender;
3      if (M is Token?) then
4          Perform Token mode algorithm;
       else
           // Check if vertex meets requirements
           // r = {r_1, r_2, ...r_k}
5          p ← p + self;
6          for j = 1; j < k do
               // Calculates extended restrictions
7              Calculates Φ(p)_{r_j};
           end
8          Calculates ω_{r,D}(p) for p;
           // Check if the node attends the requirements: r = {r_1, r_2, ...r_k}
9          if (node meet all requirements) then
10             Calculates λ_{r,D}(p_min);
11             M = [meet requirements, p_min, λ];
               // Send to parent message M
12             send (M);
           else
13             M ← does not meet requirements;
               // Send to parent
14             send (M);
           end
       end
   end
```

on its $\lambda_{r,v_i,D}()$ function. The function value reflects the ability of a vertex to attend the service with the requirements constraints and weights defined by D. Then, the chosen vertex will be that which has the minimum value for the function, that is, the device that presents the lowest resistance to provide the requested resource.

Figure 17 illustrates the step by step of the protocol to map the vicinity of the Fog computing devices and also shows the token passing through the network. It must be remembered that this process will not map the cloud infrastructure; thus, the devices that have some connection link with some CoT should add a virtual node to represent the resources and constraints of the CoTs avaliables.

To clarify the steps in the form of an algorithm, they will be organized in two pseudo codes, one for the device that has the token ownership and another for devices that do not have it. The first one is showed at Algorithm 1 and explained as follows: at line 2, the device sends a broadcast message with the requested resource, requirements constraints, and the respective weights for each constraint. From steps 3 to 14, the device waits for the answers of neighboring devices for some interval of time (TimeOut).

Figure 17 – Protocol to discovery $G_{feasible}(V,E)$ (D-Client device, $V_1, V_2, V_3, V_4 - D$ Neighborhood devices).



(a) D Broadcast Message

(b) V_1 answer $\in G_{feasible}(V,E)$

(c) V_2 answer $\notin G_{feasible}(V,E)$

(d) V_3 answer $\in G_{feasible}(V,E)$

(e) V_4 answer $\in G_{feasible}(V,E)$

(f) D send token to V_4

Source – Author.

Particularly, there are two cases to handle the device answer. When the neighbor device cannot meet the requirement constraints, so it is not part of the feasible graph ($G_{feasible}(V < E)$), the device with the token ownership does nothing; otherwise, from steps 6 to 12, it registers the neighboring device in a queue and updates its minimum path to D, if necessary.

The second case is shown at Algorithm 2. In this case, the first step is to receive the broadcast message and to check if the message is a token (step 1 to 4). Particularly, if the message is a token, then the device starts to perform Algorithm 1. Otherwise, the device uses the message content to check if it can provide the requested resource and meet all requirement constraints. This procedure uses functions $\Phi()$, $\omega()$, and $\lambda()$ (steps 8 to 19) with the parameters from the received broadcast message. If the device can meet the constraint requirements and provide the resource, it sends a P2P message to the token's owner device, informing its resistivity of service for restrictions r related with D ($\lambda()$ function) and also that it is part of the feasible graph. Otherwise, it also sends the message, but only to inform that is not part of the feasible graph($G_{feasible}(V,E)$).

4.3 Data dissemination in the Fog/Mist environment (RQ2)

A first step to address the Data Persistence Problem in the FMC environment (DPPF) is to formally model the problem. Therefore, DPPF consists of how to maintain a given D, a data block of size L, which wants to distribute and make itself available in the FMC environment. F could be called as the network devices located at the network edge, dynamically composed by n active devices in a timestamp t, among a total possible of N_t devices. The set F, at moment t, can be represented by a graph $G(V,E)$, where the set of vertexes V are the n active devices and the set of edges E are defined as the communication link between them.

The devices that are part of F at timestamp t are not necessarily $t+\Delta t$ at the same time because some of them may leave F and new devices can enter at any time. Thus, there is an output rate of devices (σ_{out}) and a renewal rate of devices (σ_{in}).

The basic concept of our proposal is to create a mechanism to infer these input and output rates of devices and to ensure that persistent data stay in the environment despite the ephemera availability of the devices that composed it. With this in mind, it is assumed that the data replication rate (the amount of data replicates generated over a specific time interval) is greater than the number of devices that left the environment in the same time interval that we

call the output device rate. Additionally, it must be assured that the number of data replicas in the environment is equal or greater than this rate. Consequently, some nodes of FMC must be selected to be responsible for the data replication control and measurement of device entry and exit rates. Thus, the choice of the nodes that will perform these functions will be made according to the criteria explained as follows.

In this proposal, the residence time of devices, which is the time in which the devices remain in the FMC environment, will determine the selection of which one will control the data output rates and data replication in their context. Thus, to implement this strategy, it is necessary to define a metric that enables to compare two nodes and choose the most stable between them. Having this in mind, the Fog stability metric (represented by ρ) could be defined as a function of co-domain between zero and one, whereby zero means that the device shows no availability in the environment during the period analyzed, and one means total availability of that device under the same conditions.

The stability function may depend on several factors associated with the device or Fog environment. However, for this analysis, what matters is how long the device remained available in the environment for a given a period. Thus, one way to implement this metric is to make the device itself calculate it by measuring the total time that it was available in the Fog/Mist environment in a given period, and divide it by the total time under analysis (T). A formal definition of Fog stability function (ρ) is in the following subsection.

4.3.1 Definitions

Definition 4.3.1 *Given F, the set of all devices in Fog/Mist environment E at moment t, and $\{t_1, t_2, ..., t_n\}$, the set of time intervals that the device p was available in E during a period T prior to current time t, where $p \in F$ and $\sum_{i=1}^{n} t_i < T$, the Fog stability function of p ($\rho(p)$) is defined as:*

$$\rho(p) = \frac{\sum_{i=1}^{n} t_i}{T}$$

If the question of the replication and distribution of data d within the network is considered, an important factor is the number of neighbors in which a device has a direct communication link. Thus, to model a metric for the number of neighbors, the concept of vertex degree (from graph theory) will be used as a metric for this issue. Good candidates for data replication control and distribution devices will be devices that combine these two concepts. Thus, points of maximum

local stability will be defined as:

Definition 4.3.2 *Given p,N and f, where p is a device $\in F$, N is the set of neighbors of p and f a function defined from $F \rightarrow \mathbb{R}$, it can be said that p is a local maximum with respect to the function $f \Leftrightarrow f(p) > f(v), \forall v \in N$.*

A relevant parameter that must be evaluated by a node (p) that assumes the central role of control and data distribution is the likelihood of losing the data in its neighborhood, ψ, which may be calculated as the likelihood that a node and its neighbors, which contain data, leave the network at the same time, mathematically:

$$\psi(p) = \prod_{i=0}^{q} (1 - \rho(p_i))$$

Where $p_0 = p$, and q is the number of neighbors from p.

Another important parameter that must be considered for the topology formed by the devices that compose FMC is the availability of the data within at most one hop from the node in focus. Therefore, aiming at creating a metric to evaluate this parameter, the availability of the data for node p of the graph $G(V,E)$ will be defined as the percentage of nodes adjacent to p that contain data d. Thus, if node d contains the data d, its availability, given d, is one by definition, or formally defined as:

Definition 4.3.3 *Given N, the set of the neighbor of p denoted by*

$$N = \{n_0, n_1, ..., n_q\}$$

$$\eta(p) = \frac{\sum_{i=1}^{q} \alpha(n_i)}{q}$$

Where $\alpha()$ is defined as:

$$\alpha(n_i) = \begin{cases} 0, & \text{if } n_i \text{ does not contain data } d \\ 1, & \text{if } n_i \text{ contains data } d \end{cases}$$

The data availability parameter can be used as a metric to guide the direction of data replication process within the network. Thus, using the pheromone (DORIGO *et al.*, 2006) concept of evolutionary computation, the data attractivity parameter can be defined as an inverse pheromone of the data availability parameter for a node P.

Definition 4.3.4 *Given the device $p \in V$ with data availability $\eta(p)$, data attractivity of d (δ) is defined as:*

$$\delta(p) = 1 - \eta(p)$$

To ensure good diffusion of data between the devices that make up the ecosystem, another key point concept is the percentage of coexistence (γ) of devices, formally defined as:

Definition 4.3.5 *Given $n(t)$, the number of active devices at time t in the FMC set F, and n_T, the total number of devices that belong to the set F, γ is defined as:*

$$\gamma = n(t)/n_T$$

The value of γ is a metric that indicates what percentage of devices are available to communicate relatively to the total count of devices that make up the environment.

4.3.2 Proposed Model

In this proposal, the DPPF problem is divided into two relatively independent sub-problems and the premise that the FMC environment can be modeled as a graph is considered, as described previously.

The first problem consists of identifying the devices that may be candidates for controllers of the data replication and storage processes. Similarly, the second issue may be defined as follows: given some controller nodes and considering that they have replicas of the given data d, what should be the replication rate of d, locally and globally, in order to ensure that it remains within the environment? As well, there is a second question: which one of the neighbor's nodes should receive its data replica?

The approach used in the proposal to address the first problem is to use a modified distributed leader-election algorithm to identify local leaders, using the stability function as a metric. Therefore, nodes with higher stability function are expected to be more likely to remain in the environment at the instant after the time interval at which the analysis was performed. It is important to realize that the choice of local leaders should also consider a minimum degree connection restriction to ensure the possibility of distribution of the D data packet within the network.

The primary condition for a device to consider itself as a local leader is that all its neighbors have the value of their stability function smaller than its stability function value.

Additionally, another necessary minimum condition is that the node has at least two neighbors, i.e., its node degree is greater than two. Thus, considering that all requirements of local leaders are defined, the IoT devices that compose FMC may classify themselves as one of three types of nodes within the graph: Local Leaders (LL), Local Leaders' Neighbors (LLN), and Far away from Local Leaders (FLL). In this proposal, each type of nodes has a different role:

- Node type LL - these nodes are responsible for controlling the output rate of nodes that own data copy in their neighborhood and manage the data replication process based on measured output rate. Also, this kind of node works as a repository of data because its stability function is a local maximum; so, locally, the node has the best likelihood to stay inside the network. In fact, the LL nodes replicate the data only when they detect that they lost some node with data in its neighborhood. This evaluation is made in each classification cycle of the node type. The process to replicate the data use the roulette wheel (HOLLAND, 1992)selection method using the nodes attractivity function to define the areas of sectors. Thus, the nodes with bigger attractivity function have a more significant likelihood to receive a data copy.
- Node type LLN - this type of node has the main responsibility to direct the copy of the data it receives for its LL node. Also, it is the function of this type of node to execute replication commands ordered by its LL or LLN type neighbors. To clarify, assuming that the LLN node has a copy of the data and receives a data replica from its LL node, since the node already has the data, the node initiates a roulette wheel selection process with its neighbors, excluding its LL node, to pass the copy of the data.
- Node type FLL - This type of node has a secondary role within the infrastructure, and its primary role is to initiate the data replication process when it owns a copy of the data. The choice of which neighbor to replicate is made using the roulette wheel selection method described before. In this way, the data must be replicated to a near LL or a region that has low data availability.

It is important to note that from the theoretical point of view it is possible that all nodes have the same stability function. Thus the node cannot be self-classified. Although this case is not easy to exist in a real environment an approach to handle this situation would be the random choice of an LL between the nodes that satisfy the condition of maximum stability.

The second problem is to ensure that the data diffusion to the control nodes (LL) are spatially distributed within the topology, avoiding, whenever possible, the concentration of

data in specific regions of the graph. Thus, to reach these goals, an epidemiological data model based on the Reed-Frost model (ANDERSSON; BRITTON, 2012b) was chosen. In essence, the main modification proposed by this approach is to define that the probability of infection (data replication) depends mainly on two factors:

- The stability function of the node to be contaminated, since, considering that the most stable nodes possess the bigger the probability of itself remaining in the network; and
- Spatial distribution of the data to regions where there is less availability of it.

Aiming at reconciling these two seemingly conflicting objectives to decide the direction in which the data should be replicated, the Author opted to use ideas of evolutionary computation, particularly the ones from genetic algorithms which also undergo similar problems when, in search of the optimal solution, they must avoid local minimums/maximums. Precisely, in genetic algorithms, the way to solve this issue is to adopt a selection operator called a roulette wheel. This operator consists of a roulette wheel that is constructed so that the circular sectors forming it have the area proportional to a fitness function of each option. Thus, the choice of the option is made by rotating the wheel and selecting an option corresponding to the area where the selector has stopped. Similarly, applying the concept and the roulette wheel operator to the described problem will use an evaluating function, which incorporates a metric in order to achieve the goals described and help to decide to which neighbor the data is to be replicated. Having this in mind, an evaluating function ($\phi()$) was defined. It combines the stability ($\rho(p)$) function and data attractivity ($\delta(p)$) of node p, which is defined as:

Definition 4.3.6 *Given $p, V, \rho(p), \delta$ and β, where p is a IoT device, V is the set of IoT devices that are part of FMC and $\rho(p)$ and $\delta(p)$ are, respectively, its stability function and attractivity function, and β the weight given for its stability, it was defined as its evaluating function:*

$$\phi(p) = \rho(p)^{\beta} * \delta(p)^{(1-\beta)}$$

4.3.3 Proposed Algorithm

The chosen algorithm will be executed in a decentralized and autonomous way by each device that composes the FMC, thus not existing any process of global coordination of the network. Consequently, all interactions between the devices are made locally and with other devices that are in their vicinity. The idea is that the objectives of disseminating and maintaining the data in the environment are an emergent behavior of the simple interactions between the

devices that are part of the network.

In the proposed model, the devices could assume three roles within the infrastructure: local leaders (LL), local leaders neighbors (LLN) and far away from local leaders (FLL). Hence, periodically, each device must evaluate what role it should assume among these options, executing the processes associated with the role.

The self-evaluation process to identify what role the device should take follows the steps:

1. Initially, the node assumes that is the type far away from local leader node (FLL).
2. Then, the node sends a broadcast message publishing its stability function and receives the stability function of its neighbors as a return. Consequently, the node can verify if it meets the definition of a local leader (LL).
3. If it does, it sends a new message to its neighbors stating that they must assume the role of neighbors to local leaders (LLN).
4. Restart the cycle.

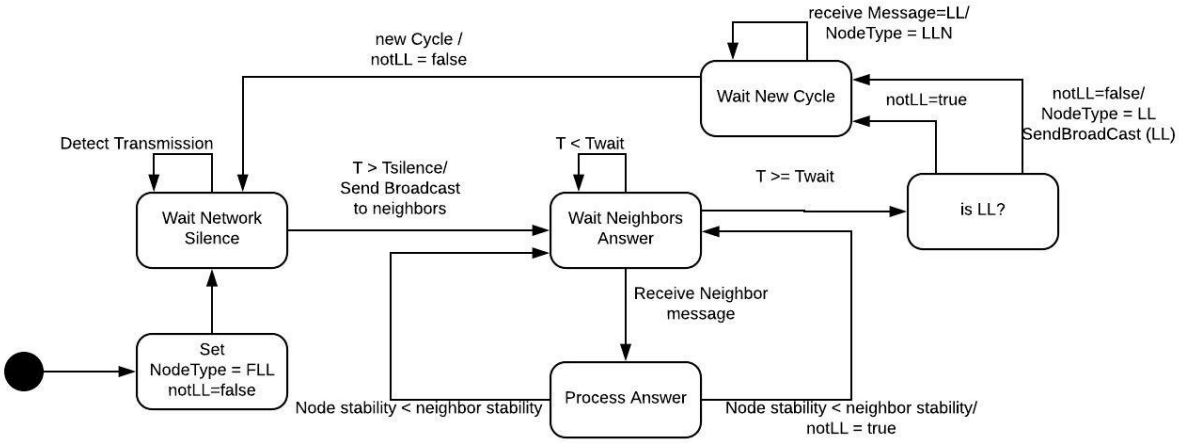
It should be remembered that for nodes to avoid collision of communications messages within the network an avoidance collisions mechanism (SHIH *et al.*, 2011) has been adopted.

In Figure 18, the state diagram of the self-classification algorithm is shown. In short, the node running this process collects data from neighbors node and, based on these data, defines its role inside the infrastructure.

The role that the node takes within the network defines how it will handle the data to disseminate it. Of course that the node performs some role if it owns a copy of the data. Otherwise, it does nothing.

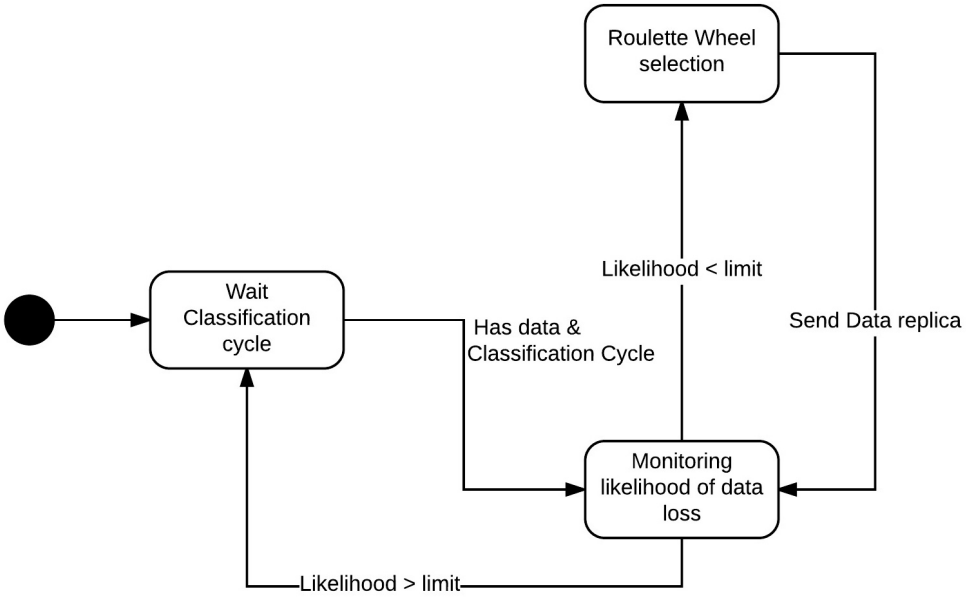
Figures 19, 20 and 21 present the state diagrams of data replication process depending on the node type rule. The node neighbor selection process uses an adaptation of the roulette wheel selection (HOLLAND, 1992), using our evaluating function $\phi(p)$ as the parameter to define the area of circular sectors of the roulette wheel. This process is executed every time a node needs to replicate data for its neighbor, except in case of LLN type nodes, when the node receives the first copy of the data. In that case, the data replication must be directed to its LL type node.

Figure 18 – State diagram for self classification node type.



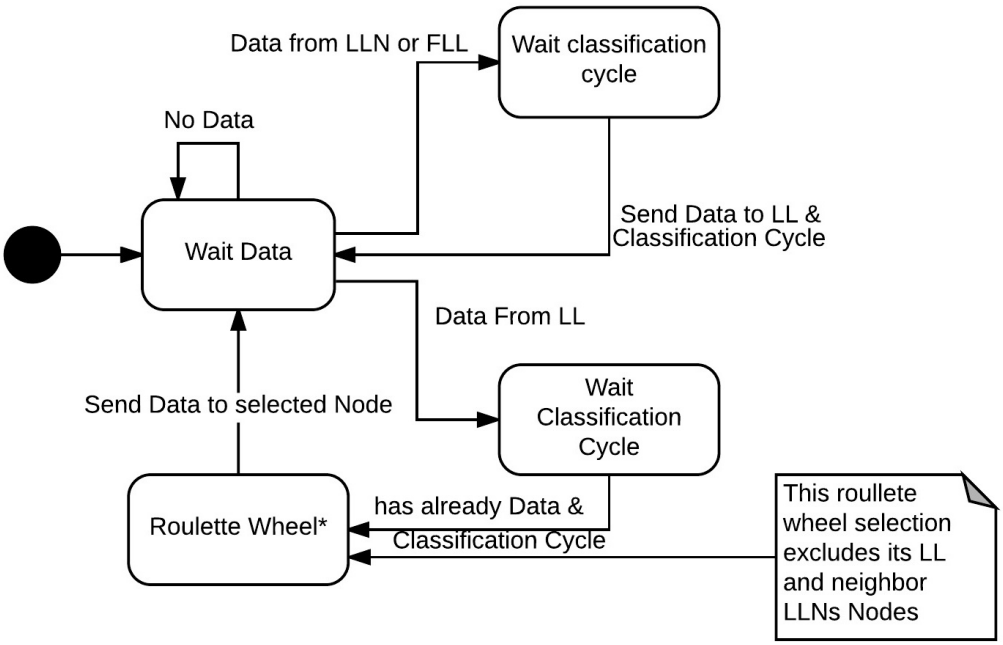
Source – Author.

Figure 19 – State diagram for LL nodes.



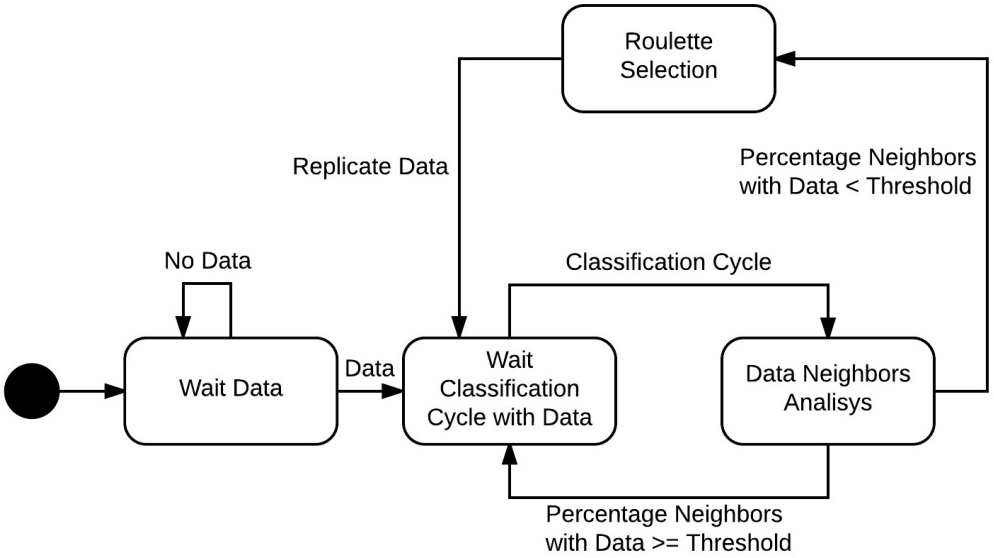
Source – Author.

Figure 20 – State diagram for LLN nodes.



Source – Author.

Figure 21 – State diagram for FLL nodes.



Source – Author.

4.4 Computational Resource Prediction for Fog/Mist environment (RQ3)

The computational resource allocation prediction problem is defined as follows how an IoT device without previous knowledge of the FMC environment in which it has just entered

should predict which IoT device in the network it should request a particular computational resource?

Aiming to address the computational resource allocation prediction problem in the FMC environment, the proposal of this thesis approaches this problem as a hybrid collaborative recommendation system combining with temporal availability information of the provider devices. That information is made available using the mechanism of data dissemination of the RQ2 proposal. The client device gets the RS parameters and performs predictions of the recommended device in the network capable of providing a specific computational resource.

The following premises are assumed for all devices that are part of the network:

1. All devices that are part of the FMC infrastructure may be characterized by a vector of technical hardware and software features and device functionality; and
2. Similar devices tend to rate providers of computing resources in a similar way.

Considering the premise 1, initially, it is known that IoT devices are machines; therefore, their behaviors are generally deterministic, i.e., they are determined according to their design requirements that meet technical specifications. It is reasonable to assume that it is possible to characterize any IoT device by a set of technical characteristics. For example, one could characterize a smartphone by its CPU processing power, Random Access Memory (RAM), Flash memory, operating system, manufacturer, type of network interfaces, battery, functionality.

Similarly, by analyzing the second premise, it is also reasonable to assume that similar devices likewise evaluate a particular service or computational resource with a similar rating. Thus, for the same reason that devices are machines, they used objective criteria or metric to evaluate the quality of a computational resource shared by other devices, unlike humans. For instance, identical twins could rate a movie in a totally different way because the criteria used depend on their personal taste, the current emotional state and other subjective factors.

Both assumptions are very reasonable and, in the view of this author, applicable in several IoT environments.

4.4.1 Definitions

Based on the first premise, a device that is part of the FMC environment can be formally represented as the following definition:

Definition 4.4.1 *Given p a device, where $p \in F$, then $F_f = \{f_0, f_1, .., f_m\}$ is the set of m features that define the device p and its functionality within the environment F. So, p can be used as a*

vector of dimension $m \times 1$ *as follow:*

$$p = \begin{bmatrix} f_0 \\ f_1 \\ \dots \\ f_m \end{bmatrix}$$

In the proposal, aiming at taking benefits from premise 2, it is important to define a similarity metric between the devices. In the literature, there are several metrics of similarity that should be applied (Cosine similarity, Euclidean distance, Manhattan distance, Pearson correlation, Spearman correlation, Tanimoto coefficient and Log Likelihood) (GUO *et al.*, 2014). However, in this work, the metric of cosine similarity is used, since it is the most used metric in Collaborative Filter systems and have a low computational cost for its calculation. The formal definition of the cosine similarity metric can be seen in Section 2.2.5 of Chapter 2 of this thesis.

In a collaborative recommendation system, a key factor for good recommendations is a large amount of data from past ratings. In a conventional and centralized recommendation system running on a powerful server, there is no problem in storing this information in a database. However, in this proposal, this system should be performed in a heterogeneous and highly dynamic environment in which the required information for the recommendation moves despite the network in P2P architecture. Thus, strategies to minimize the amount of required data should be designed to provide the recommendations.

Having this in mind, the author decided to store the required information for each computational resource type in three matrices $n \times n$, where n is the total number of devices in the FMC environment. The matrices contain information on the mean ratings, the last post rating and the total number of evaluated ratings for each computational resource. Each matrix is organized as follows: rows represent the devices of the network when they assume the role of clients of a particular computational resource and the columns similarly represent the same devices when they assume the role of computational resource providers. The matrices memory requirement growths with the square of the number of devices in the FMC environment. Thus, to propose a solution that allows IoT devices with low memory capacity to use the proposal, the matrices were distributed in clusters around the LL-type nodes that are also responsible for keeping the matrices data in the network (RQ2). So, the memory requirements are reduced, because now the matrices contain information only about the devices that are neighbors to LL-type nodes.

Definition 4.4.2 *Defining R^{cr}_{mean} as the matrix of the mean ratings for the computational resource cr, a matrix $n \times n$, where n is the total number of the device in FMC environment, and each element row represents a rater device and each column represents a resource provider device, each element (i, j) contains the average rating of device i about the computational resource cr when provided by device j. It is important to note that, if there is not any rating available the value on matrix, R^{cr}_{mean} is conventionalized as -1.*

$$R^{cr}_{mean} = \overbrace{\begin{pmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \cdots & r_{0,n-2} & r_{0,n-1} \\ r_{1,0} & r_{1,1} & r_{1,2} & \cdots & r_{1,n-2} & r_{1,n-1} \\ \vdots & & & & & \\ r_{n-1,0} & r_{n-1,1} & r_{n-1,2} & \cdots & r_{n-1,n-2} & r_{n-1,n-1} \end{pmatrix}}^{\text{n columns}} \Bigg\} \text{n rows.}$$

Similarly, the matrix of the latest ratings of devices about computational resource *rc* was defined.

Definition 4.4.3 *Defining R^{cr}_{rating} as the matrix of the mean ratings for the computational resource cr, a matrix $n \times n$, where n is the total number of the device in FMC environment, and each element row represents a rater device and each column represents a resource provider device, each element (i, j) contains the most recent rating of device i about the computational resource cr when provided by device j. Moreover, it is important to note that, if there is not any rating available, the value on matrix R^{cr}_{rating} is conventionalized as -1.*

$$R^{cr}_{rating} = \overbrace{\begin{pmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \cdots & r_{0,n-2} & r_{0,n-1} \\ r_{1,0} & r_{1,1} & r_{1,2} & \cdots & r_{1,n-2} & r_{1,n-1} \\ \vdots & & & & & \\ r_{n-1,0} & r_{n-1,1} & r_{n-1,2} & \cdots & r_{n-1,n-2} & r_{n-1,n-1} \end{pmatrix}}^{\text{n columns}} \Bigg\} \text{n rows.}$$

Once more, the matrix of the total number of ratings evaluated by the devices about computational resource *rc* was defined.

Definition 4.4.4 *Defining R^{cr}_{total} as the matrix of the mean ratings for the computational resource cr, a matrix $n \times n$, where n is the total number of the device in FMC environment, and each element row represents a rater device and each column represents a resource provider device, each element (i, j) contains the total number of rating of device i about the computational resource cr when provided by device j.*

$$R^{cr}_{total} = \overbrace{\left.\begin{pmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \cdots & r_{0,n-2} & r_{0,n-1} \\ r_{1,0} & r_{1,1} & r_{1,2} & \cdots & r_{1,n-2} & r_{1,n-1} \\ \vdots & & & & & \\ r_{n-1,0} & r_{n-1,1} & r_{n-1,2} & \cdots & r_{n-1,n-2} & r_{n-1,n-1} \end{pmatrix}\right\}}^{\text{n columns}} \text{n rows.}$$

The matrices of the definitions 4.4.2, 4.4.3, and 4.4.4 show a view of the ratings of the system as a whole that will be very useful for the devices that wish to predict one computational resource provider available. Nevertheless, each evaluator device individually only holds information on the evaluations performed by itself. In this way, the mean, the last, and the total matrices of the individual rating matrices of the mean, last, and total device as a matrix *r timen* will similarly be defined, where *r* is the number of computational resource types available in the FMC environment and *n* is the total number of devices in the FMC environment. The formal definitions of matrices are the following:

Definition 4.4.5 *Defining R^{device}_{mean} as the matrix of the mean ratings for the computational resources in environment FMC (F), a matrix $r \times n$, where r is the total number of computational resources types (Processing power, storage, network, multimedia, etc) available in F, and n is the total number of the device in F, and each element row represents the computational resource type and each column represents a resource provider device, each element (i, j) contains the average rating of computational resource i about the device provider j. It is important to note that, if there is not any rating available, the value on matrix R^{device}_{mean} is conventionalized as -1.*

$$R_{mean}^{device} = \overbrace{\begin{pmatrix} r_{0,0} & r_{0,1} & r_{0,2} & . & . & . & r_{0,n-2} & r_{0,n-1} \\ r_{1,0} & r_{1,1} & r_{1,2} & . & . & . & r_{1,n-2} & r_{1,n-1} \\ . & & & & & & & \\ . & & & & & & & \\ . & & & & & & & \\ r_{r-1,0} & r_{r-1,1} & r_{r-1,2} & . & . & . & r_{r-1,n-2} & r_{r-1,n-1} \end{pmatrix}}^{\text{n columns}} \Bigg\} \text{r rows.}$$

Definition 4.4.6 *Defining R_{rating}^{device} as the matrix of the last ratings for the computational resources in environment FMC (F), a matrix $r \times n$, where r is the total number of computational resources types (Processing power, storage, network, multimedia, etc) available in F, and n is the total number of the device in F, and each element row represents the computational resource type and each column represents a resource provider device, each element (i, j) contains the most recent rating of computational resource i about the device provider j.*

$$R_{rating}^{device} = \overbrace{\begin{pmatrix} r_{0,0} & r_{0,1} & r_{0,2} & . & . & . & r_{0,n-2} & r_{0,n-1} \\ r_{1,0} & r_{1,1} & r_{1,2} & . & . & . & r_{1,n-2} & r_{1,n-1} \\ . & & & & & & & \\ . & & & & & & & \\ . & & & & & & & \\ r_{r-1,0} & r_{r-1,1} & r_{r-1,2} & . & . & . & r_{r-1,n-2} & r_{r-1,n-1} \end{pmatrix}}^{\text{n columns}} \Bigg\} \text{r rows.}$$

Definition 4.4.7 *Defining R_{total}^{device} as the matrix of the total number of ratings for the computational resources in environment FMC (F), a matrix $r \times n$, where r is the total number of computational resources types (Processing power, storage, network, multimedia, etc) available in F, and n is the total number of the device in F, and each element row represents the computational resource type and each column represents a resource provider device, each element (i, j) contains the total number rating of computational resource i about the device provider j.*

$$R_{last}^{device} = \overbrace{\left.\begin{pmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \cdots & r_{0,n-2} & r_{0,n-1} \\ r_{1,0} & r_{1,1} & r_{1,2} & \cdots & r_{1,n-2} & r_{1,n-1} \\ \vdots & & & & & \\ r_{r-1,0} & r_{r-1,1} & r_{r-1,2} & \cdots & r_{r-1,n-2} & r_{r-1,n-1} \end{pmatrix}\right\}}^{\text{n columns}} \text{r rows.}$$

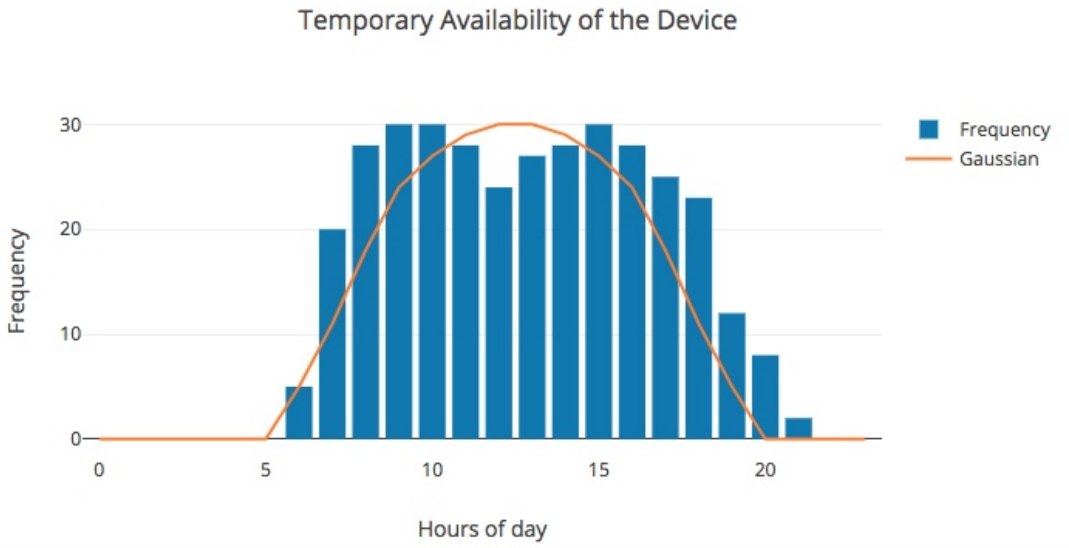
In real FMC environment, it is common that the matrices R_{mean}^{cr}, R_{rating}^{cr}, R_{total}^{cr}, R_{mean}^{device}, R_{rating}^{device}, and R_{total}^{device} are sparse matrices. Therefore, they can be stored in a list of list format to save storage space.

Equally, an important point to be addressed in the computational resource allocation problem is the temporal availability of the devices in the FMC environment being studied.

In the present proposal, each device knows when it is inside the FMC environment. Therefore, the device can create an average time profile of its permanence in the network. This profile can be created based on a frequency histogram of the device within the time interval under analysis. Considering the example shown in Figure 22 the availability of a smartwatch from an employee of a company that comes in at 8:00 am and usually leaves the company around 6:00 p.m., in order to save storage space, it is possible to approximate the histogram by a Gaussian curve and then store only the mean value and the standard deviation of the curve. It is important to highlight that if the device has the profile of attending the environment routinely in several different periods within the time period under analysis, it is necessary to approximate its behavior by more than a Gaussian curve, where each one represents one of the periods of the permanence of the device in the network environment.

It is important to note that the histogram also serves as an evaluation tool to identify the way the device frequents the network environment. To put it in another way, if the visiting frequency is sparse to the network environment, the device should not be considered in the Recommendation System as a provider of computational resources because, in this case, it is practically impossible to predict when it will be within the network environment.

Figure 22 – Example of device temporal availability.



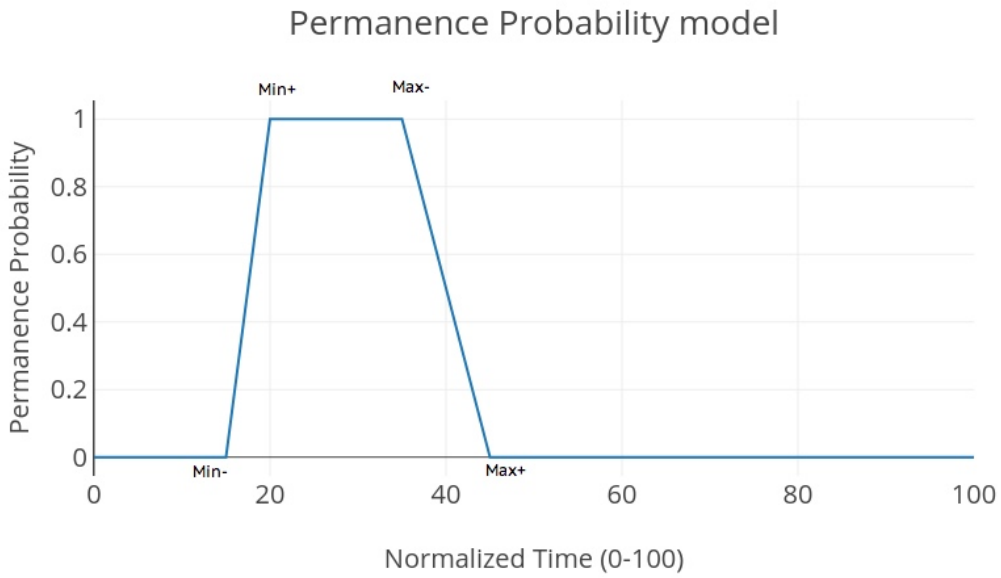
Source – Author.

Definition 4.4.8 *The equation 4.1 represents probability function of the normal distribution that models the temporal probability of the device in the network environment.*

$$\Psi(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{4.1}$$

Another way to model the probability of permanence of the device in the FMC environment is to consider the probability function as is a trapezoidal curve, where zero means that the device is not in the network and the value one when the device remains in the environment. In this approach, the curve starts to climb in the shortest recorded time of network entrance (Min^-) and stabilizes at the value one, at the instant of maximum recorded time (Min^+) from the input of the device in that network. Similarly, the function starts the descent in the shortest recorded time of exit (Max^-) and ends the drop arriving at the value zero in the maximum registered time (Max^+) of disconnection of the network. Figure 23 presents one example of a trapezoidal curve to model temporal availability of the device in the network.

Figure 23 – Trapezoidal temporal model function of device availability.



Source – Author.

In order to optimize the computation time in the client device, some matrices and sets that will be used in calculating the prediction of the rating of a device for a given computational resource *cr* will be defined.

Definition 4.4.9 *Defining the set F_{cr} as the set of raters of cr. F_{cr} is the subset of F, in which its elements at some past point have already evaluated the computational resource cr, it could be represented formally as:*

$$F_{cr} = \{d_1, d_2, ..., d_p\} \,|F_{cr} \subset F \text{ and } \forall d_i \in F_{cr} \text{ already rating the computational resource } cr$$

Definition 4.4.10 *Defining the set P_{cr} as the set of providers of cr in which the devices can provide the computational resource cr from among the total device set devices (F), it could be represented formally as:*

$$P_{cr} = \{p_1, p_2, ..., p_q\} \,|F_{cr} \subset F \text{ and } \forall p_i \in P_{cr} \mid p_i \text{ could provide the computational resource } cr$$

Definition 4.4.11 *Defining the similarity matrix ($S_{devices}$) as the matrix $n \times n$, where n is the total number of devices that are part of the set F, the elements (i, j) of this matrix are the cosine similarity between device i and device j. Formally:*

$$S_{devices} = \overbrace{\begin{pmatrix} s_{0,0} & s_{0,1} & s_{0,2} & \cdots & s_{0,n-2} & s_{0,n-1} \\ s_{1,0} & s_{1,1} & s_{1,2} & \cdots & s_{1,n-2} & s_{1,n-1} \\ \vdots & & & & & \\ s_{n-1,0} & s_{n-1,1} & s_{n-1,2} & \cdots & s_{n-1,n-2} & s_{n-1,n-1} \end{pmatrix}}^{\text{n columns}} \Bigg\} \text{n rows.}$$

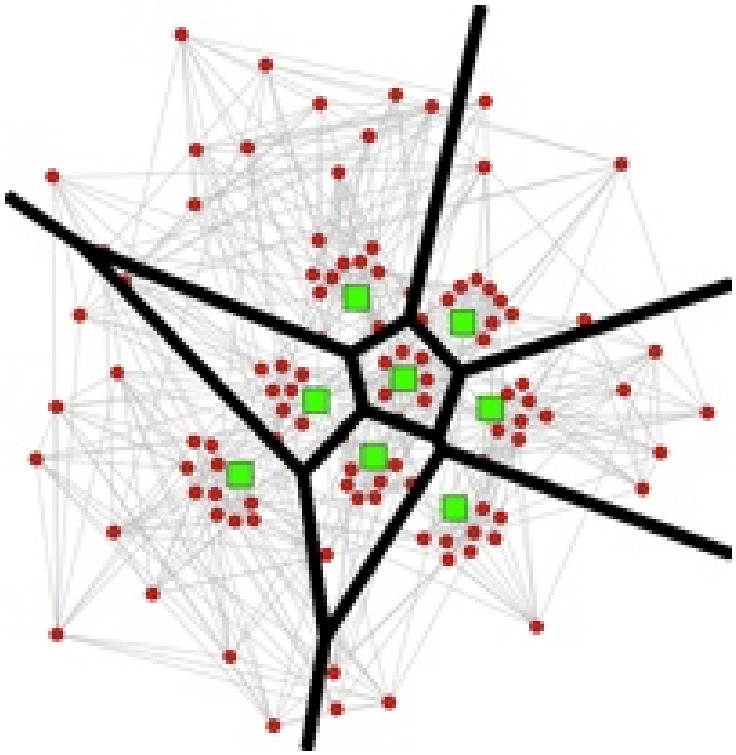
where, $\vec{d_i}, \vec{d_j}$ *are devices and each element of matrix is calculated according the equation 4.2.*

$$s(\vec{d_i}, \vec{d_j}) = \frac{\vec{d_i} \cdot \vec{d_j}}{|\vec{d_i}||\vec{d_j}|} \tag{4.2}$$

4.4.2 Proposed Model

The proposed solution to the computational resource allocation prediction problem reuses the architecture proposed in RQ2, which creates a dynamic structure of the nodes that are part of the FMC, in which they could assume different roles in topology. The node role depends on its average time inside network (stability function) and its role should be as Leader Local (LL), Local Leader Neighbor (LLN), and Far away from Local Leaders (FLL). However, in the new proposal, the LL nodes also assume the role of aggregating the rating information (R^{cr}_{meanl}, R^{cr}_{rating}, and R^{cr}_{total}) and time availability information from its neighbor nodes. In addition, it is also the responsibility of the LL nodes to distribute the data parameters of the prediction algorithm to the clients using the dissemination algorithm proposed in RQ2. Figure 24 illustrates the process of clustering the distribution of the recommendation parameters that are concentrated in the LL nodes.

Figure 24 – Diagram to show the clustering spatial distribution of recommender system parameters.



Source – Author.

To calculate the prediction for the best computer resource node provider available, the client node must perform the following steps:

1. To request from the closest LL node the set of raters of cr ($F_{cr}, definition 4.4.9$);
2. To request from the closest LL node the set of providers of cr (P_{cr}, definition4.4.10);
3. To request from the closest LL node the matrix devices similarity ($S_{devices}$, definition 4.4.11), and the matrices related to computational re mean rating cr (R^{cr}_{mean}, definition 4.4.2),last rating (R^{cr}_{rating}, definition 4.4.3), and number of ratings (R^{cr}_{total}, definition 4.4.4);
4. Given F_{cr}, P_{cr}, and $S_{devices}$ to predict the rating of a provider $p_i \in P_{cr}$, it is necessary that for each element p_i, one must found K devices of the set F_{cr} that had already rated p_i or at least the K more similar devices from set P_{cr} when compared with p_i;
5. To request from the closest LL node the list of $\Psi_{device}(x|\mu,\sigma^2)$ (definition 4.4.8) for the K provider devices;
6. Once you find the K similar devices raters and providers, the predict rating for computational resource cr for provider $p_{i,j}$ can be calculated using the equation 4.3,

$$p_{i,j} = (\bar{r}_i + \frac{\sum_{l=1}^{K} S_{devices}(i,l) * (rating_{mean}(i,l) - \bar{r}_l)}{\sum_{l=1}^{k} S_{devices}(i,l)}) * \Psi_{devicei}(x|\mu,\sigma^2) \tag{4.3}$$

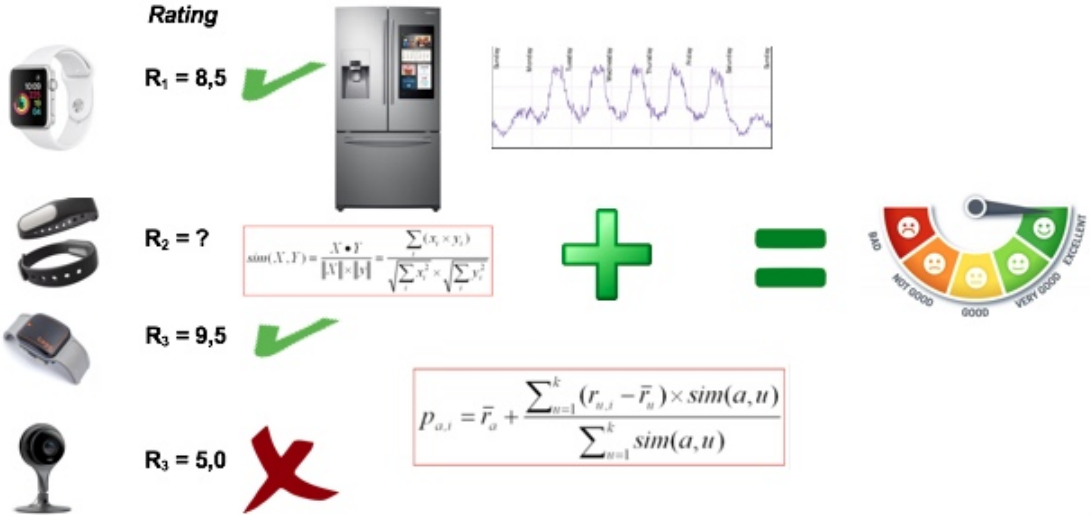
where $\bar{r}_i$ and $\bar{r}_l$ are the average rating of device i and l for all the provider devices rated by device i and l respectively, $\Psi_{devicei}(x|\mu,\sigma^2)$ is the probability density of device i remain on network at time t, and $rating_{mean}(i,l)$ is calculated using the equation 4.4 and α is the weight given for the last rating value.

$$rating_{mean} = \alpha * R^{cr}_{rating} - (1-\alpha) * R^{cr}_{mean}(i,l) \tag{4.4}$$

7. After calculating the predicted rating for all elements of the set P_{cr}, it was sorted in descending order and select the device corresponding to the first element as the prediction to be the best available to provide the computational resource cr.

Figure 25 illustrates the rating prediction process exemplifying the case of an activity tracker monitor who wishes to predict the rating of a refrigerator. Initially, there is a search for similar K devices; in this case, K is equal to 2. The two most similar devices to activity tracker are two smartwatches being found. To estimate the rating that the active tracker monitor would have about refrigerator will be the ponderate average of the ratings of similar IoT devices that have previously rated the refrigerator multiplied by a factor between zero and one that indicates the probability of the refrigerator is available at that time. The weight of the rating of each similar device will be the degree of similarity of the device to the client device.

Figure 25 – Example of prediction process flow.



Source – Author.

An important issue in choosing the K devices that will contribute to the recommendation information is the similarity, however, in some cases, only this information may not be sufficient. For example, consider the case where the device is very similar to the client

device, but this device has few recommendations about the other device providers of a particular computational resource. Thus, another important parameter is the percentage of evaluations that this device has about other resource providers defined formally as follow:

Definition 4.4.12 *Let n be the total number of devices providing a computational resource with n_r the number of providers that the evaluating device r has previously evaluated. The percentage of evaluations of r providers (η_r) is defined as:*

$$\eta_r = \frac{n_r}{n} \tag{4.5}$$

The choice of which K devices that will contribute to the calculation of the predictor must commit with these two parameters, similarity, and percentage of evaluation on other providers simultaneously.

The solution found to combine these two essential parameters was to generate two lists sorted in decreasing order respectively by the similarity and percentage of evaluations of the providers. Then for each provider device calculates a new metric called score. The score metric is defined as the sum of the positions in both lists. Hence, a low score is an indicator that the devices are well placed on both lists, so it must be the commitment with both parameters. On the other hand, a high score may mean that in at least one of the lists, or in both, the device is not well positioned, thus indicating that it is not compromised with at least one, or both, parameters. Therefore, the devices chosen are the K ones which they having the lowest score metric.

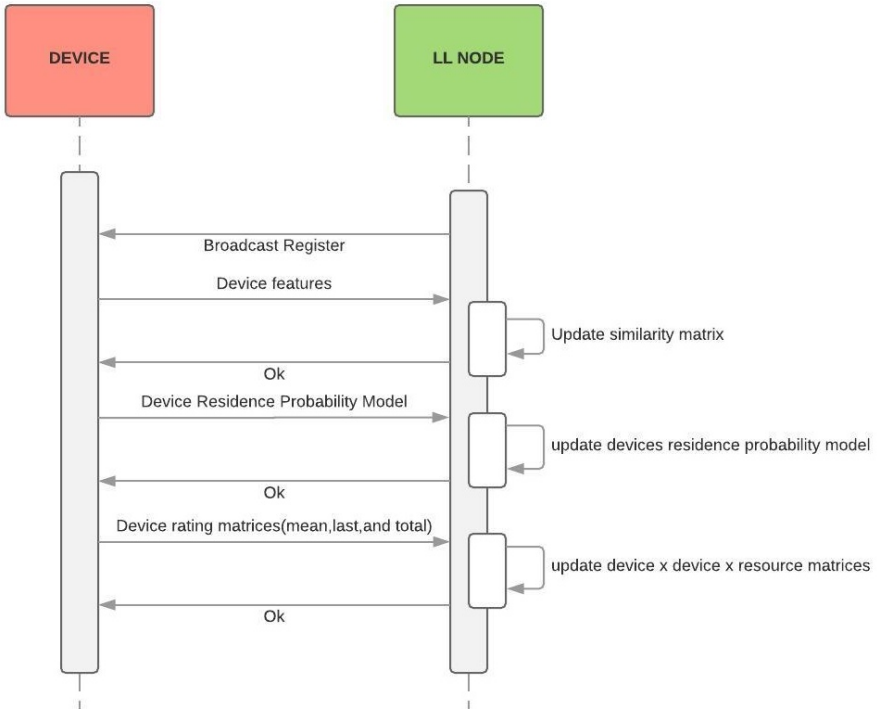
4.4.3 Proposed Algorithm

The proposed algorithm consists basically of three major processes described as follows:

- Consolidation of recommendation data;
- Dissemination of recommendation data; and
- Calculation of recommendation.

In the proposal, it was used the same adaptive hierarchical structure proposed in Section 4.3.2 of the data dissemination problem to execute the recommendations system processes.

Figure 26 – Sequence diagram of recommendation data consolidation protocol.



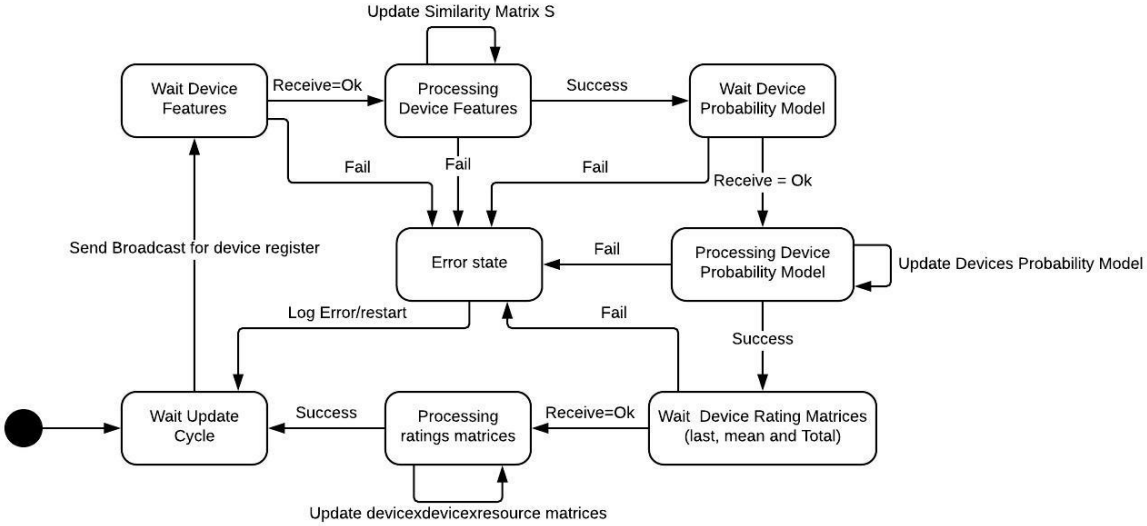
Source – Author.

The process of consolidating the recommendations data consists of the following steps:

1. The LL nodes periodically announce through a broadcast message that they are ready to receive information;
2. Devices in the vicinity of the LL node receive the message and uses a Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) method to access the network;
3. The device that takes the access sends its features and awaits confirmation of receipt by the LL node;
4. The LL node receives the feature device data and then updates its list of provider devices and computes the similarity matrix with the already registered devices;
5. Then, the device sends its residence probability density information and again waits for the confirmation of the LL node;
6. The node LL receives the information and then updates the residence probability density information associated with the sending device;
7. The device sends its rating matrices by resource type (R_{mean}^{device}, R_{rating}^{device}, and R_{total}^{device}) and awaits confirmation of receipt;

8. The LL node uses the device x resource matrices to update the device x device matrix for each type of computational resource; and
9. The LL node packages the similarity matrix ($S_{devices}$), list of devices and their probability density of residence, matrices of rating by computational resource (R^{cr}_{mean} and R^{cr}_{rating} and R^{cr}_{total})and disseminates this data.

Figure 27 – The state diagram of LL node implements the recommendation data consolidation protocol.



Source – Author.

Figure 26 illustrates the protocol of the data consolidation process of recommendation according to the previously described descriptive process. In particular, to facilitate the understanding of the process, it is also shown in Figure 27 the state diagram that the LL node must execute to implement the protocol described in Figure 26.

After consolidating the data required for the recommendation system, the LL node performs the algorithms and protocols described in Section 4.3 to disseminate this data within the network environment located at the edge. An important point to note in this process is that, after consolidating a package with the recommendation information, the LL node creates a new version of that data packet, then stops distributing the previous version on the network.

Finally, the last process happens in the client device that wants to find a certain computational resource (network or computational processing or storage or other). It is the precondition of this process that the client device has received the consolidated data packet with

the recommendation information.

The client device must perform the following steps to receive the recommendation of which device is the best candidate device to provide a particular computational resource:

1. The client device queries the similarity matrix of the devices creating a descending sorted list of similarities with itself.
2. The client device removes from the sorted similarity list all devices that are unable to provide the desired resource *cr*.
3. The client device selects the K top devices in this list.
4. The device selects the device x device rating matrices (R^{cr}_{mean} and R^{cr}_{rating} and R^{cr}_{total}) per computational resource associated with the selected computational resource *cr*, verifying that all of the chosen K devices have evaluation data in that matrix. In the case that one of the K devices does not have any evaluation in the matrix, that device is eliminated from the list of the chosen ones by choosing the subsequent device from the ordered list.
5. Step 4 should be repeated until the similar K devices have been chosen with evaluations on the computational resource *cr*.
6. Using Equation 4.3, the client device calculates the prediction of each device in the list of K similar and chooses the one with the highest rating.

Similarly to what happens in the process of consolidating the recommendation information, the device can also receive a new version of this package. In this case, the device discards the previous information and begins to consider in its calculation the most updated recommendation information package.

In a real environment, these three processes execute in parallel, and only in the first execution of the process the sequence is necessary.

4.5 Summary

In this chapter, the proposal of an algorithm and protocol for the problem of allocation of resource in FMC environment related to RQ1 was presented, which also suggests the better network infrastructure to be used, namely Fog, Mist, or Cloud computing. Moreover, the concept of feasible Fog was introduced, which allows to avoid the growing of the execution time of algorithm even in networks with several devices.

Also in Section 4.3.2, it was presented the challenges related to RQ2 and one proposal to address the challenge of how to keep data in FMC with a dynamic topology. The proposed

algorithm borrows some bio-inspired computing concepts that utilizes a self-adaptative hierarchic structure of devices to control the data dissemination process in the network.

Concluding the chapter, the proposal of a collaborative hybrid recommendation system was presented to handle RQ3 issues. It is based on previous evaluations of the device about computational resources afforded by provider devices and temporal availability statistics of these devices, enabling to make predictions about the best device available in the network, the one which is able to provide a specified computational resource.

The next chapter presents the details and results of the assessment of the proposals discussed in this chapter. This evaluation was focused on validating the effectiveness of the proposals as well as generating evidence to validate or refute the hypothesis of this thesis.

5 EVALUATION

This chapter presents the simulation experiments to evaluate the effectiveness of the thesis proposals discussed in Chapter 4 and their results.

This chapter is then organized as follows: Section 5.2 presents the results of the evaluation of the network infrastructure selection algorithm and protocols related to RQ1; Section 5.3 describes the experiment performed to assess the data dissemination in Fog/Mist Computing (FMC) environment algorithm related to RQ2; Section 5.4 discusses the experiments related to the validation of the algorithm and protocols involved in the prediction model of computational resources available in the FMC environment related to RQ3; and, finally, Section 5.6 concludes this chapter.

5.1 Overview

The Contiki operating system (DUNKELS *et al.*, 2004) is selected as the test environment to evaluate the proposals, given that it has good acceptance in academia and industry and is often used in Wireless Sensor Network (WSN) and Internet of Things (IoT). It is also important to remind that the Contiki operating system brings a simulation tool (Cooja) (DUNKELS *et al.*, 2004), which facilitates the analysis and tests of the experiment with different setting parameters and network topologies. The Cooja simulator emulates different types of hardware platforms through objects called motes; the emulated code is the same used in real hardware. In particular, the proposed algorithm and protocol in the C language were implemented and simulated using motes of type Sky (TI MSP430 CPU/8MHz, 48 KB ROM, 10KB RAM, 250kbps 2.4GHz IEEE 802.15.4), available in the Cooja Simulator of Contiki version 2.7 for Linux.

It is clear that one hardly a simulation environment captures all nuances and parameters of an actual FMC environment. However, to evaluate the proposals, the most relevant parameters in proposal context were chosen and previously loaded in the motes that will be part of the simulation. Thus, these parameters are considered as inputs for evaluation experiments. They are examples of these parameters are available storage capacity, latency between devices, bandwidth of communication links, processing capacity among others.

5.2 Network Infrastructure Selection (RQ1)

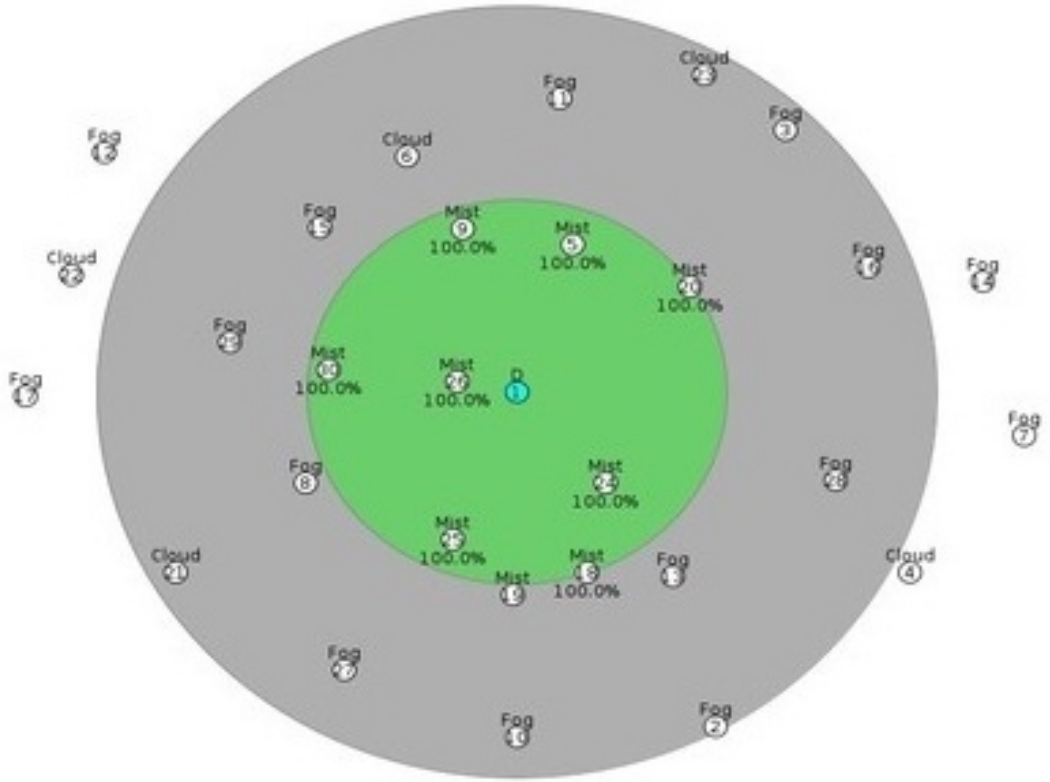
5.2.1 Experiment definition

Each test scenario was executed thirty times in order to check the consistency of answers. Additionally, all message traffic between the motes was also monitored by the Cooja tool and it was verified by the author whether they were in accordance with the protocol specifications.

The functioning of the algorithm was evaluated into two phases as follows.

In the first phase, two experiments sought to validate the correctness of the choice of the algorithm were performed. For that, two distinct hypothetical topologies, in which each set of constraints was designed for suggesting a known infrastructure option to use the computational resource, were used and compared with the result obtained by the algorithm.

Figure 28 – Scenario used in Experiment 1 - First phase.



Source – Author.

The first experiment aimed to compare the choice made by the algorithm with a configuration where it was possible to infer the choice according to the constraints imposed. Thus, a scenario was constructed based on a client device *D* in a network where 29 nodes

belonging to the Mist (9), Fog (15), and Cloud (5) types were randomly distributed in circular areas in a uniform distribution. The nodes of each type were characterized based on three parameters: latency (L), processing power (C), and storage capacity (S). The latency for each node was defined as proportional at the received signal strength indication from the client node combined with a multiplicative factor that depended on the node type (Mist = 1, Fog = 2, and Cloud = 6). The multiplicative factors comprising 1, 2, and 6 were selected so the latency values for Fog were slightly higher than those for Mist, and those for Cloud were well above those for Mist. It is important to note that the Mist type nodes were distributed inside the first ring and the remaining nodes outside the ring, as shown in Figure 28. The configuration in terms of the processing power and storage capacity for each node was set according to the following rules. Depending on their type, the nodes received a random value for their processing power and storage capacity according to Table 8[1], which shows that Cloud nodes had high processing power, storage capacity, and latency. By contrast, the Mist nodes had low processing power, storage, and storage capacity. Finally, the Fog nodes had intermediate processing power and storage capacities, where the values were between those for the Mist and Cloud nodes.

Table 8 – Node configuration range for storage capability (S) and processing power (C) in Experiment 1.

Node type	Storage(S)		Processing Power(C)	
	Min	Max	Min	Max
Mist	10	50	3	10
Fog	100	120	50	70
Cloud	500	550	300	350

Source – Author.

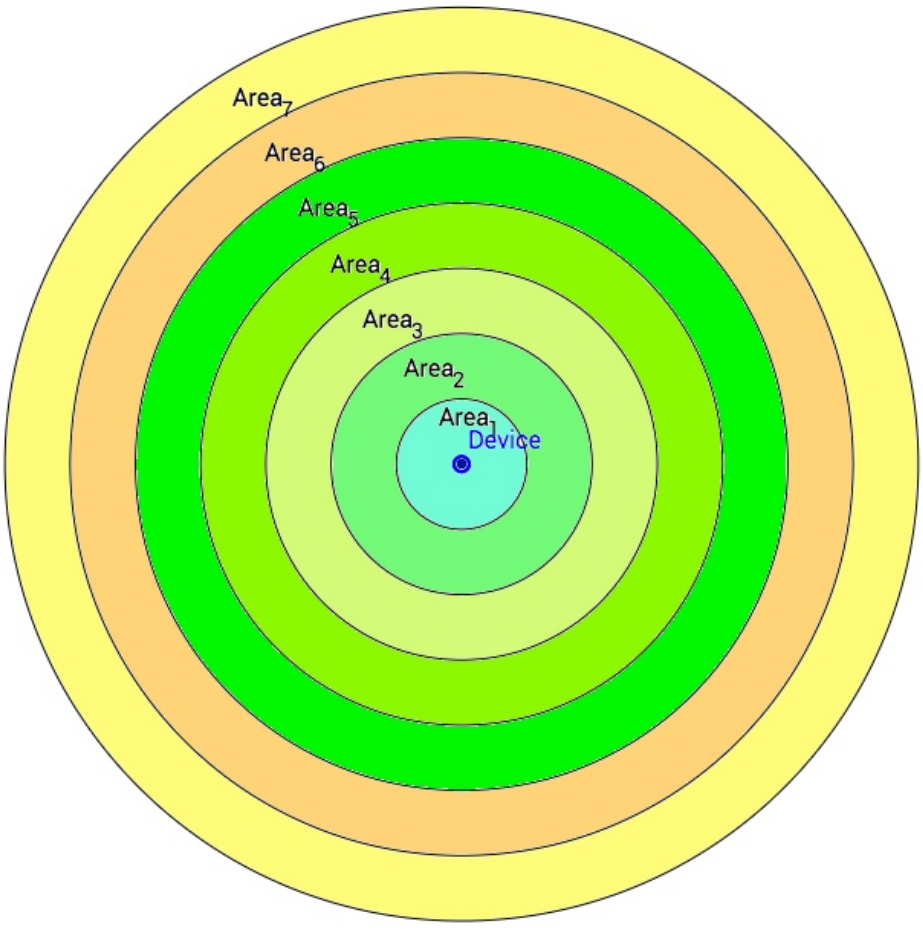
For each experiment, the configuration of the client device requirements was defined into priorities for latency, processing power, and storage capacity. Then, the twenty-nine nodes were distributed and configured according to the defined rules, and executed the algorithm that, at its conclusion, selected one of the twenty-nine nodes to be the provider of the computational resource taking into account the defined requirements and priorities. The experiment was performed thirty times with six configurations that suggest the selection of a Cloud node, seventeen configurations that suggest the selection of a Fog node, and seven configurations that suggest a Mist node. The results are shown in Table 10.

The second experiment was designed for its topological distribution of the com-

[1] The values shown lack units, because they are relative values.

putational resources aiming at making it easy to identify the device that meets the constraint requirements. This topology is generated randomly. However, the resource distribution follows predefined rules. For example, considering that the constraint is network latency, the test configuration may be a topology in which the nodes are distributed between concentric circles, where the radius of a circle is a multiple of its immediate inner circle's radius and the center of all circles is the device *D*. Also considering that the latency between each node and *D* is defined as a multiple of latency of the nodes in the most inner circle, if a node has its distance from the center of two and a half radius, its latency will be three times the latency of devices inside the first circle. In this experiment, this rule for latency and a similar one for storage were used, where the capability of storage of the node is proportional to the ring of which it is a part. Thus, it is easy to infer the choice of the node to provide the resource based on constraint of latency, processing power and storage.

Figure 29 – Scenario used in Experiment 2 - First phase.



Source – Author.

In the second experiment of the first phase, the previously described configuration

showed at Figure 29 was used by distributing different numbers of device configurations (10, 20, 30, 40, and 50 devices) in concentric rings in a random manner. At the same time, the constraints of the client device placed in the center of the rings were varied by relaxing the latency requirement and requiring storage features. As in the proposed distribution, the latency between the nodes and the client device increases according to the distance from the center; in a similar way, the storage resource is also allocated. It is expected that, as the extent to which the latency restriction is relaxed, the node associated with the device that will provide the resource become more distant itself from the client device (see Table 9).

Table 9 – Device latency by Area Experiment 2 - Phase 1.

Area	1	2	3	4	5	6	7
Latency(ms)	10	50	100	150	200	250	300

Source – Author.

In the second phase, a growth analysis of the runtime of the algorithm was performed with the number of devices on the network. Again, parameterized scenarios were created in order to allow reaching all devices within a maximum range of 10 hops of the device D. Each scenario is executed thirty times, and the time between the start of the simulation and the time to reach the stop condition are measured. Then, a graph was created in order to analyze the behavior of the average runtime of the algorithm of discovery of the node to provide resources by the number of devices in the environment.

In the second phase of the experiment, the growth of the runtime was studied along with the number of vertexes in the vicinity of the client device. In this scenario, the latency restriction for the application was set exclusively to a maximum of 30 ms. All of the neighborhood vertexes are Mist Computing resources, and the latency between each vertex of the graph has been defined as fixed 10 ms. Two network topologies were studied; in the first, the vertexes were distributed randomly with no restriction to the degree of the vicinity of devices and their neighbors; in the second, vertexes are distributed in a rectangular grid and positioned so that each vertex has a maximum of four neighbors. Each simulation was performed thirty times and recorded the average duration of execution in the Cooja Simulator.

5.2.2 Simulation results

The first experiment of the first phase was performed thirty times with six configurations that suggest the selection of a Cloud node, seventeen configurations that suggest the

selection of a Fog node, and seven configurations that suggest a Mist node. The results are shown in Table 10. All algorithm simulation results matched with the expected scenario configuration.

Table 10 – Experiment 1 - First phase: Client priority configuration and simulation results.

Weights (%)			Output		Weights (%)			Output	
L	C	S	Node	Type	L	C	S	Node	Type
3	15	82	4	Cloud	12	52	36	2	Fog
5	10	85	22	Cloud	10	53	37	3	Fog
2	6	92	4	Cloud	19	51	30	2	Fog
3	7	90	4	Cloud	17	57	26	2	Fog
0	11	89	4	Cloud	15	46	39	10	Fog
1	20	79	6	Cloud	10	55	35	2	Fog
15	40	45	10	Fog	13	46	41	3	Fog
18	44	38	2	Fog	14	45	41	3	Fog
10	50	40	2	Fog	89	7	4	30	Mist
14	41	45	10	Fog	90	4	6	9	Mist
14	49	37	2	Fog	94	1	5	25	Mist
17	48	35	3	Fog	86	9	5	5	Mist
13	52	35	8	Fog	85	9	6	19	Mist
11	55	34	17	Fog	95	2	3	5	Mist
16	42	42	10	Fog	96	2	2	18	Mist

Source – Author.

In the second experiment of the first phase the requirement priority was defined as 100% for latency. The maximum acceptable latency requirement is modified using the values 60, 150 and 320 ms for each device configuration with different device numbers. The latencies of the devices distributed in the rings areas are assigned according to the ring in which it is located, following the values at Table 9 and topology according to Figure 29. The simulation results of the second experiment, first phase, are showed in Table 11.

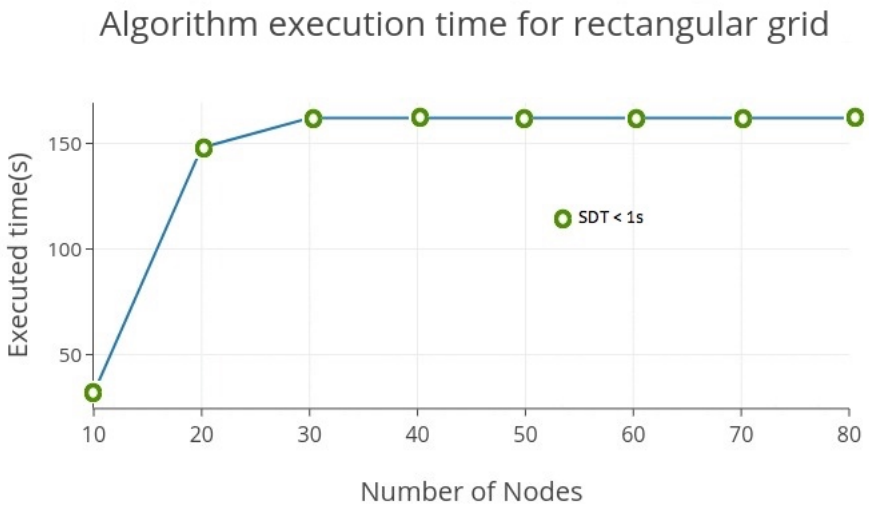
The results of the second phase experiments on the average time of execution are shown in the graphs in Figures 30 and 31. The absolute values of times measured in the simulations are not relevant, because the version used was deliberately set with slow time constants and generation of debug messages in order to allow observation of the behavior of the algorithm. Thus, the unit of time must be considered relatively, because in a real environment the time will be much smaller than the values measured in the simulations

Table 11 – Experiment 2 - First phase simulation results.

Maximum Latency(ms)	Number of Nodes	Runtime(min)					Choosen Area
		1	2	3	4	5	
60	10	0.51	0.52	0.50	0.52	0.54	2
	20	0.59	1.00	1.01	1.10	1.03	
	30	1.30	1.35	1.29	1.34	1.32	
	40	2.20	2.26	2.15	2.26	2.21	
	50	2.40	2.45	2.36	2.41	2.40	
150	10	0,59	0,58	0,58	0,57	1,00	4
	20	1.20	1.15	1.11	1.05	1.09	
	30	1.25	1.34	1.39	1.37	1.43	
	40	2.41	2.40	2.39	2.40	2.43	
	50	2.48	2.59	3.01	2.55	2.51	
320	10	1.03	1.10	1.05	1,00	1,02	7
	20	1.14	1.10	1.07	1.19	1.23	
	30	1.37	1.35	1.32	1.35	1.33	
	40	2.43	2.45	2.59	2.43	3.16	
	50	2.59	3.10	3.07	3.05	3.00	

Source – Author.

Figure 30 – Grid distributed Vertexes experiment.



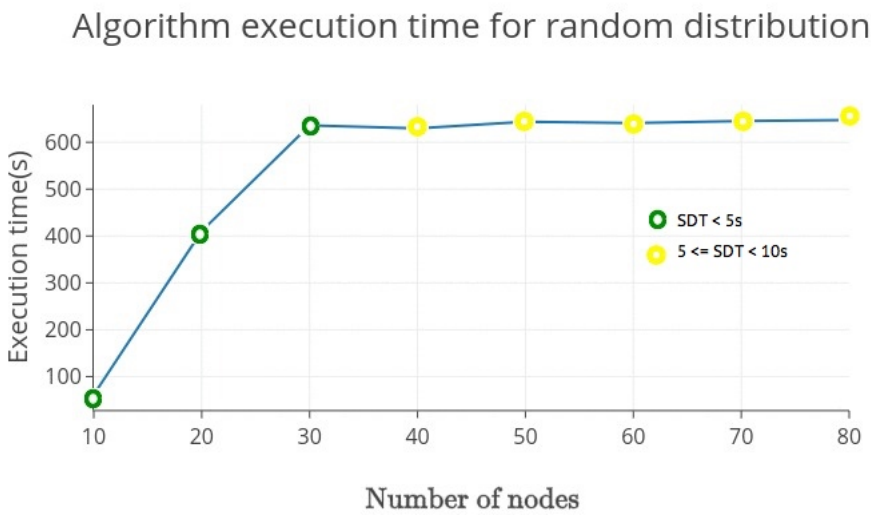
Source – Author.

5.2.3 Discussion

Aiming at evaluating the correctness and the growth of the execution time of the algorithm, the results of the experiments were analyzed and suggested consistency in the selection and the execution time applicable in IoT environments.

The results suggest that the selection obtained by the algorithm is consistent with the constraints and yields according to the expected values for the configurations used.

Figure 31 – Random distributed Vertexes experiment.



Source: Author

Although at first glance the algorithm has a complexity of order $O(n^2)$, due to the feasible Fog concept, it limits the growth of the runtime of the algorithm, as we can see in the results shown in Figures 30 and 31. The runtime grows in a smaller rate and stabilizes after a certain amount of vertexes. This effect is due to the fact that, as the algorithm deepens in the neighborhood graph, it is expected that many of the new vertexes discovered cannot meet the requirements of latency, soon finalizing the search algorithm from that vertex to avoid the growth of the runtime of the algorithm.

5.3 Data dissemination in Fog/Mist environment (RQ2)

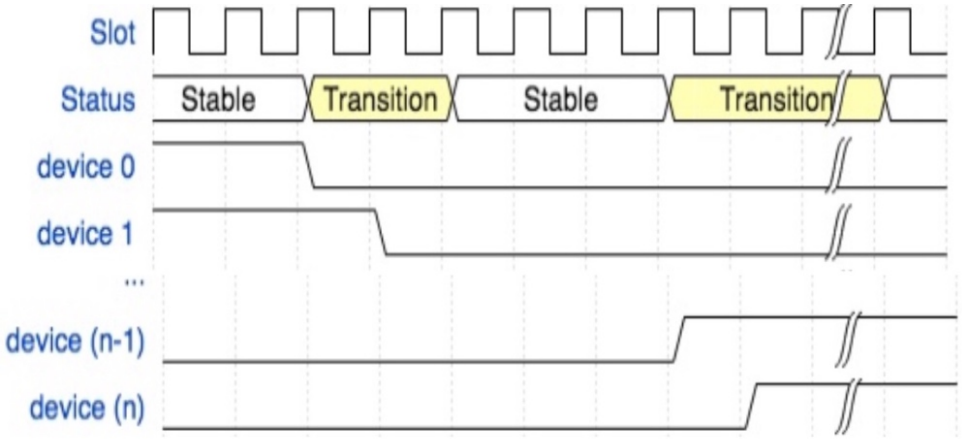
5.3.1 Experiment definition

Aiming at validating the feasibility of the proposal, again the Cooja simulation environment and Contiki operating system have been chosen. The procedures and type of motes used were also the same as those used in the experiments of the Section 5.2 The detailed algorithm and implementation of all experiments are available on GitHub [2].

Considering that in the FMC environment, there are often the entrance and exit of the devices in the network, an experiment was elaborated to evaluate the robustness of the proposal in keeping the data in the environment by varying some model parameters as the coexistence of devices in the infrastructure, the minimum staying time in the network, and the total number of the device in the network, for the purpose of systematically evaluating the quality of the results

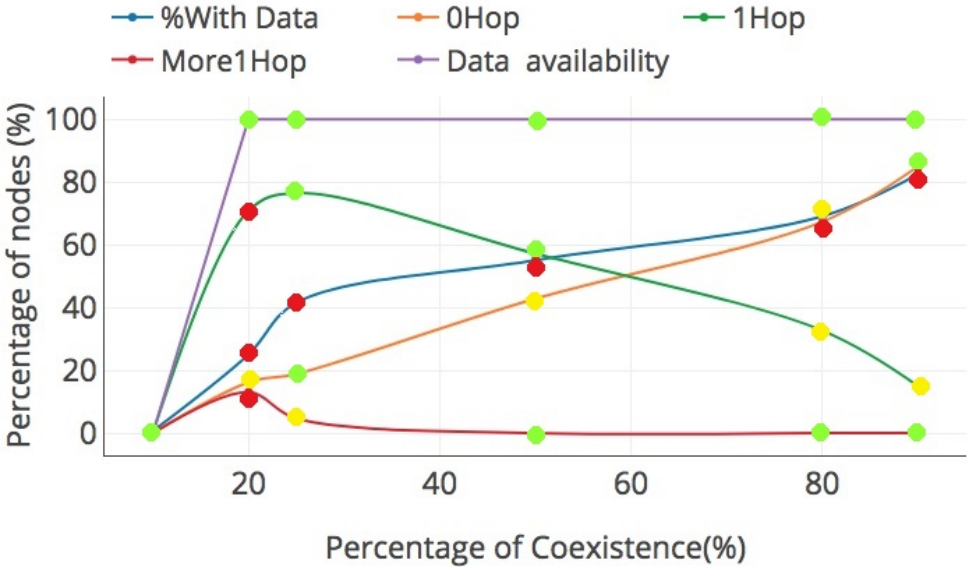
[2] Avaliable source code in the address <https://github.com/vald3nir/contiki>

Figure 32 – Presence of devices during simulation.



Source – Author.

Figure 33 – Data availability x Percentage coexistence (Experiment 1).



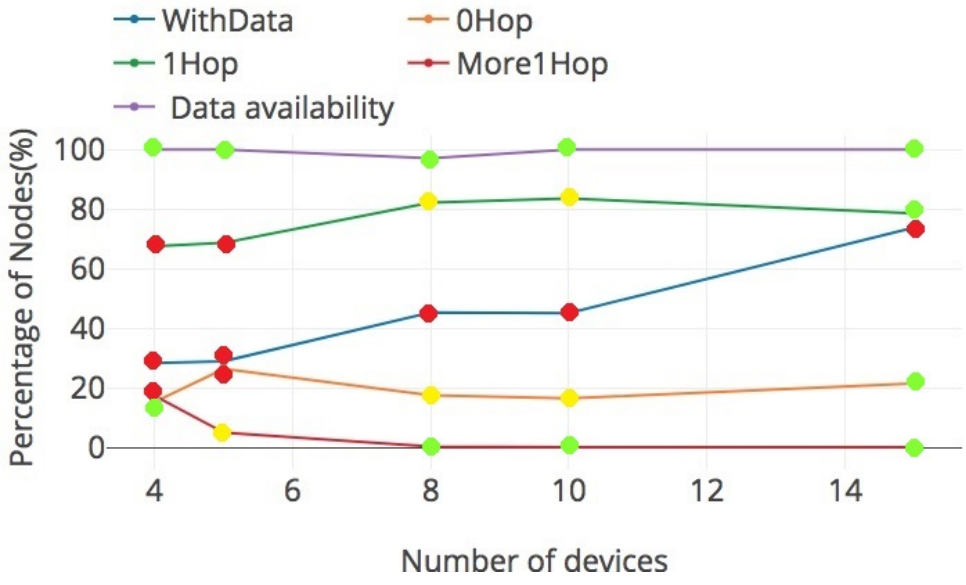
Source – Author.

generated by the algorithm.

5.3.2 Metrics

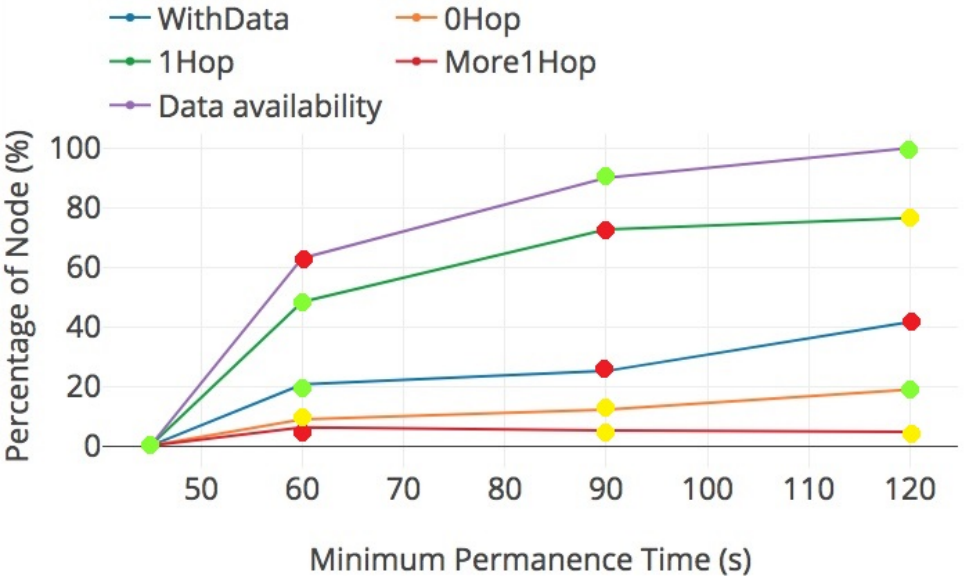
The following metrics were chosen: Data presence (which is an indicator that assumes 100% value if there is data in the network; otherwise, 0% value), Data availability in the network (which is the percentage of LL nodes of the network that owns the data at the end of the simulation), and Data hops distribution (which is the percentage of nodes for a specified number of hops, e.g., (0,1,2,..) from data to nodes in the network).

Figure 34 – Data distribution x Device density (Experiment 2).



Source – Author.

Figure 35 – Minimum Permanence Time Influence (Experiment 3).



Source – Author.

5.3.3 Simulation results

The first experiment aims at testing the robustness against topology dynamicity; therefore, the behavior of the algorithm will be studied when the percentage of coexistence of the devices (PCD) is varied in the environment. Thus, the total number of devices (20 devices) and the minimum permanence time (MPT) (120 s) is maintained constant throughout the simulation for each value of the PCD parameter. Given that, in each simulation, the devices are randomly positioned following a normally distribution in a fixed area and, similarly, the value of their

stability function is also randomly generated. The presence of the devices in the environment are also randomly generated, following a pattern similar to the graph shown in Figure 32. Of course, to guarantee the fixed coexistence rate, the permanence function of the devices is constructed, ensuring that the number of devices in the environment is always the number established by the coexistence percentage, even if the distribution of these devices is random. It is important to note that according to how the experiment is built, none of the devices remains in the environment during full simulation time.

Hence, at the end of each simulation, the described metrics are calculated and the simulation is repeated 30 times. Finally, the mean value of the metrics and their standard deviation (STD) obtained will be used for analysis.

In Figure 33, the results of the simulations of the first experiment are shown. The simulations were generated by varying the PCD from 10% to 90%.

The second experiment aims at evaluating the influence of density of devices in the network; therefore, the parameters are kept constant, both the PCD and the MPT. Then, the configuration is varied, changing the total number of devices in the area from 16 to 60, with PCD of 25% for each simulation and fixed MPT of 120 seconds. Finally, for each configuration, thirty simulations were performed, thus providing the mean value of defined metrics. In Figure 34, the results of this experiment are presented.

Similarly, in the third experiment, the configuration is varied, changing the MPT from 120 s to 45 s, with variations nearly of 30 s for each simulation set and keeping the other parameters constant. This experiment aims at identifying the minimum time for the algorithm to work and its results are shown in Figure 35.

5.3.4 Discussion

In experiment 1, it can be observed that, even with meager coexistence rates, the algorithm enables data availability in the FMC environment. Another fact is that, as it can be seen in the experiments, the data is distributed more homogeneously in the network when the PCD is increased. This fact can be verified by observing two indicators: 1) decreasing the percentage of nodes that are more than one hop away from data (More1Hop); 2) increasing the percentages of nodes with data (0Hop) and/or a Hop distance (1Hop). All values are showed in Figure 33. In all graphs, the mean values and variance (STD) were presented as centered circles at the mean value and color-coded the variance according to the range (Variance(%) Legend:

Green $[0-5]$ /$Yellow(5-10]$/$Red\,(10,20]$).

In the second experiment, the main influence of the number of devices in the FMC environment is related to the distribution of the data in the network topology. It is observed through the indicators of 0Hop and 1Hop that the increase in the number of devices favored the better distribution of the data within the topology, this effect can be observed in the graph of Figure 34.

In the last experiment, the results show that, when the MPT of the devices in the environment is significantly reduced, the algorithm stops functioning, causing the loss of the data in the network. It is believed that the reason for that is that if the device doesn't remain in the network enough time to run the classification and replication cycle, the process stops working.

5.4 Computational Resource Prediction for Fog/Mist environment (RQ3)

5.4.1 Experiment definition

The model presented in the 4.4 section to perform the prediction of available computational resources is a recommendation system that uses machine learning techniques. Thus, two conditions are necessary for its operation:

1. It is expected that, in a partial or total way, the devices that compose the environment in which the experiment will be performed should have a pattern of temporal behavior with routines in their availability within the network. This premise is relevant because only the existence of temporal behavior patterns allows the algorithms to "learn" what the best computational resource will be available at any given time.
2. It is necessary to have a considerable amount of data on the previous rating of allocations of computational resources in the environment;

So the first condition can be guaranteed by designing experiments in which a percentage of the devices by design follow a routine with little variations in these routines. On the other hand, the other devices have a totally random character in a uniform distribution.

To guarantee the second condition, an initial phase of the experiments was created, called the data capture phase. This phase aims at generating a database that can feed the proposed algorithm. Given a scenario of study, there will be devices capable of providing computational resources and client devices of these resources. During the simulation, the devices that need a computational resource seeks this resource, rating the quality of the resource provided after using

it. This assessment was defined by criteria that depend on the involved devices types, available resource capacity, the provider and the amount of the requested resource.

In this phase, a method of searching and selecting available computational resources must be used. In the proposed study scenario, it was decided to adopt a rather simple process. The method chosen is to send a broadcast message when a client device needs the computational resource; then, the first device capable of answering the request that responds will be selected as the provider of the desired computational resource.

In the experiment, after using the computational resource, the device must evaluate the provider using the defined criterion. In addition, the evaluation should register a timestamp, a client device ID, a provider device ID, the computational resource and the rating.

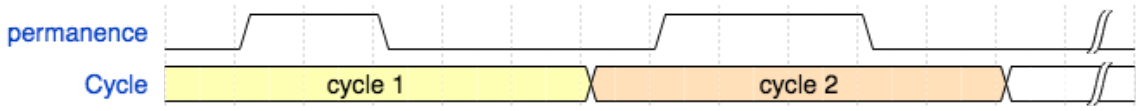
For each study scenario defined in a fixed area, it was considered that all devices can act both as clients and providers. Then, the following input parameters were defined: number of devices ($n_{devices}$), the percentage of devices with routines ($p_{routine}$), the percentage of variation within the routine ($p_{variation}$), the number of cycles of simulated routines (n_{cycles}) and the number of evaluations of computational resources per device (n_{rating}).

Defined the parameters of the simulation, for each device that follows routines, a pattern of time permanence and a random position were generated within the scenario. The pattern of time permanence is repeated with small variations in each routine cycle. Thus, the variations happen in the times of entry, exit, and permanence having been added aiming at simulating IoT environments more realistically. On the other hand, the position of the device varies at each routine cycle. The device is positioned at a random position according to a uniform distribution.

Figure 36 illustrates a time permanence pattern of a device during a simulation scenario. In this case, the dwell time in varied between two consecutive cycles.

In the data capture phase, the test scenario was performed with different percentage of device configurations with routines (20%, 40%, 60%, 80%, and 100%). It is important to note that, for each different configuration, different rating databases are generated, which, in turn,

Figure 36 – Pattern of time permanence in simulation cycles.



Source – Author.

generate different parameters for the RS of computational resource prediction.

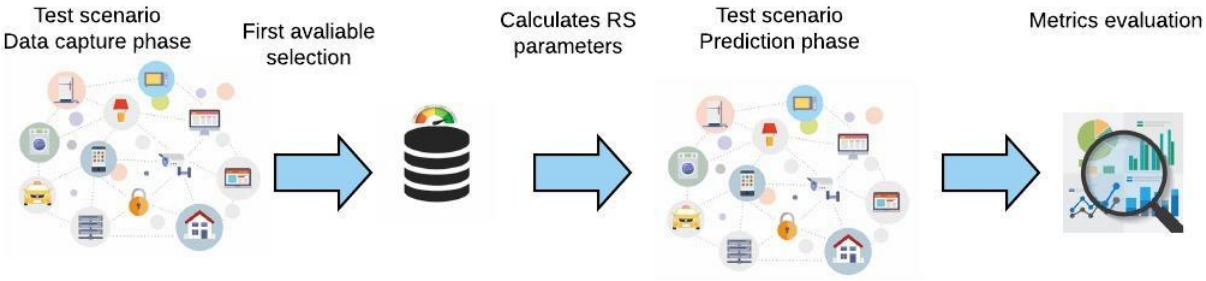
The data capture phase generates a database consisting of a log of transactions of the allocation of computational resources with their respective rating, and recording of the inputs and outputs of the provider's devices within the network. This data is the basis for generating the average rating matrices and estimators of the temporal availability of the provider devices. In this experiment the estimator used will be a set of trapezoidal estimators similar to that shown in Figure 23 for each provider device.

After the data capture phase, the data collected by each device were combined and the general RS parameters were generated. This process in the proposed protocol is executed on LL-type nodes. However, in order to have better control of the process and to simplify the test procedures, it was opted for the external execution of this process in the Cooja simulator. Thus, finalized the generation of the parameters of the predictor,theses parameters were loaded on all nodes that will participate in the next phase of the experiment.

The next phase of the experiment consists in evaluating the efficacy of the predictor by testing it in the same environment used to train it. At this stage, the client device does not execute any search process to find the computational resource provider. Instead, it uses the predictor to find out which device to use for the desired resource. For each prediction, the device verifies whether the predicted device is actually available for having its resource used and gives an evaluation of it. If the device is not available, the failure of the predicted algorithm is also recorded.

Finally, the last phase of the experiment is using metrics, defined in the next subsection, to analyze the results generated by the predictor. It is important to remember that the same analysis will be performed for the database generated by first answer method for choosing

Figure 37 – Test flow for each scenario configuration.



Source – Author.

the computational resource. Figure 37 presents a diagram illustrating the general flow of the evaluation process.

5.4.2 Metrics

To evaluate the performance of the predictor, three metrics are chosen and described as follows:

Definition 5.4.1 *The robustness (γ) is defined as the ratio of the number of recommendations in which it was possible for the client device to successfully access the recommended device($n_{success}$ and the total number of recommendations generated by the predictor (n_{total}).*

$$\gamma = \frac{n_{success}}{n_{total}} \tag{5.1}$$

Definition 5.4.2 *The mean efficiency of the predictor (ε) is defined as the average of the relationship between the rating of the device proposed by the algorithm and the maximum rating among the devices available in the environment at that time.*

$$\varepsilon = \frac{1}{n_{success}} * \sum_{n=1}^{n_{success}} \frac{rating_{predictor}}{rating_{max}} \tag{5.2}$$

Definition 5.4.3 *The perfect match (φ) is defined as the ratio of the number of success recommendations with maximum possible score in which it was possible for the client device to successfully access the recommended device($n_{maximum}$ and the total number of recommendations generated by the predictor (n_{total}).*

$$\varphi = \frac{n_{maximum}}{n_{total}} \tag{5.3}$$

The first metric is an indicator of the effectiveness of the algorithm by measuring the percentage of times that the prediction algorithm proposed consistent recommendations.

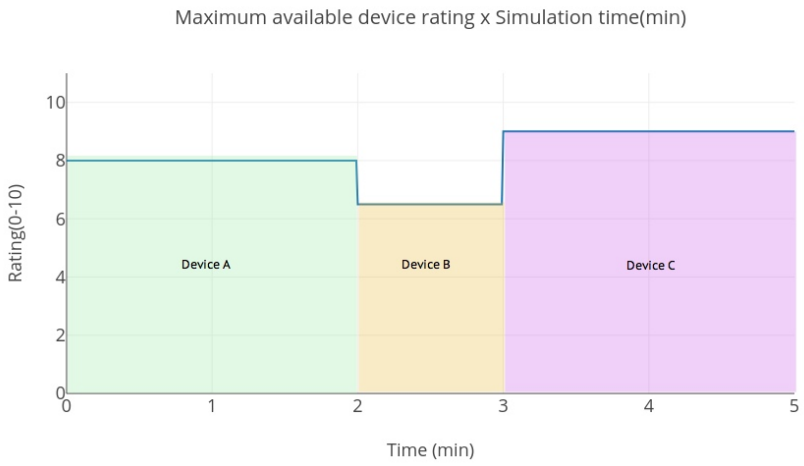
On the other hand, the second metric aims at measuring the degree of efficiency of the algorithm in capturing information about the best computational resource provider device available at a given time point within the study environment. It is important to emphasize that the calculation of this metric is only possible in controlled environments in which the rating of the best device available to provide a particular computational resource at any time during the simulation is known previously.

The third metric is a measure of excellence indicating the percentage of predictions that, besides being correct, are evaluated with the maximum score possible at that moment within the FMC environment.

In the case of this study, it was possible to gather this information during the whole period of the simulation. Once having defined the device and computational resource, the researcher could identify, within the set of devices that have a routine, which of them has the best capacity to provide the specified computational resource and which should be the rating of it considering the evaluation of the device that requested the resource.

Figure 38 illustrates an example in which three different devices A, B, and C assume the role of the best provider for a computational resource *c* during the simulation period. Thus, the maximum possible rating of a device *D* for the computational resource *cr* in the environment is respectively 8.0, 6.5, and 9.0, corresponding respectively to the evaluations of devices A, B and C.

Figure 38 – Example of maximum rating device available.



Source – Author.

5.4.3 Simulation results

In order to validate the effectiveness and robustness of the proposal, the experiment described as follows was elaborated:

In a controlled environment, 20 devices will share the computing resource storage during at least ten routine cycles. In the experiment, the devices have different storage capacities distributed according to the histogram shown in Figure 39, six devices with few

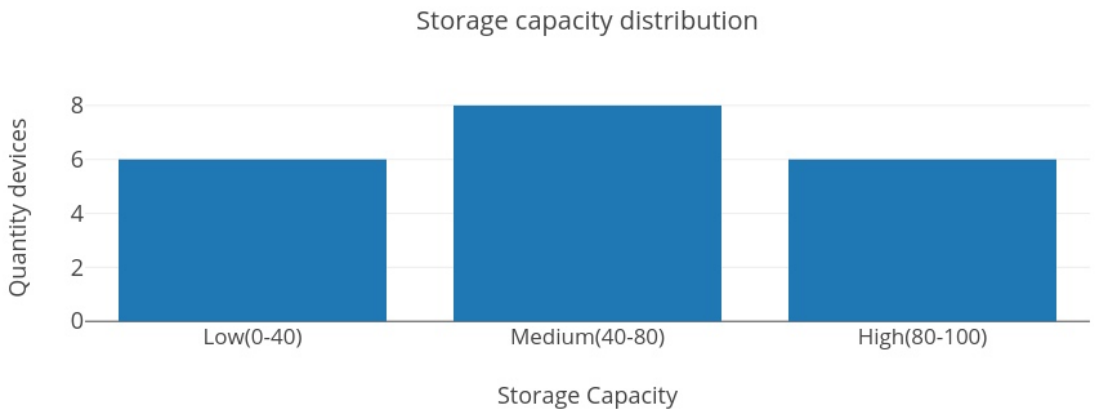
computational storage resources, eight with medium computational storage resources and six with high computational storage capabilities.

Each simulation allows you to define the portion of devices that will follow a pattern of inputs and outputs within the network with little variation in that pattern. This parameter will be called the percentage of devices with routine.

During simulation, each device should act as a provider or client of the computational resource. In each routine cycle, random allocations request will be performed. So, at end of simulation, it is generated a log of evaluations transactions of allocations of the computational resource storage. The choice of which devices will be the client and provider is also randomly made again in a uniform distribution. Each evaluation transaction contains the timestamp in which it occurred, the identity of the client device and the provider device, the type of service, and the evaluation value made by the client after the transaction.

The client device evaluates the provider by ranking it with a score between 0 and 10 calculated according to the following rule.

Figure 39 – Storage distribution of experiment devices (20).



Source – Author.

The device compares its storage capacity with the provider capacity, with three possible cases:

1. Provider and client storage capacity are equal then the evaluation rating is 5.0;
2. Provider storage capacity is bigger than client storage capacity, so the rating is given by equation 5.4

$$rating = minimum(10.0; (\frac{(Storage_{provide} - Storage_{client})}{Storage_{client}}) * 5.0 + 5.0) \quad (5.4)$$

3. Provider storage capacity is lower than the client storage capacity; then, the rating is given by equation 5.5.

$$rating = maximum(0; (\frac{(Storage_{provide}}{Storage_{client}}) * 5.0) \quad (5.5)$$

Aiming at generating a data volume that allows calculating the mean rating matrices for each configuration, the simulation environment is executed within enough time that the matrix of the means of evaluations has been filled with at least 90 % to avoid cold start problem. Another important aspect to be highlighted is that, in the temporal modeling of the permanence of the provider's devices, it will be used a trapezoidal temporal model with a maximum of five inputs/outputs in the network in a routine cycle.

It is equally important to note that, for each configuration of the percentage of devices that performs routine, it is expected to have a corresponds database of generation of the matrices of means and temporal statistics.

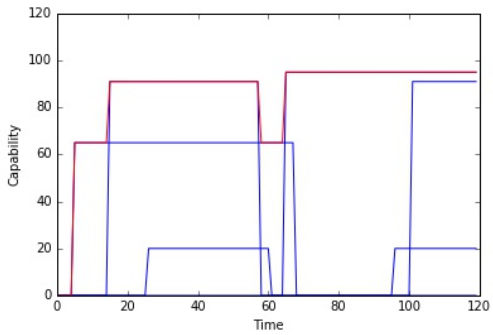
The second phase of the experiment consists in loading these matrices together with all the parameters necessary for the predictor in three different client devices: one with low storage capability, one with medium storage capability, and one with high storage capability. Then five simulations were executed for each type of client device (low, medium and high storage capability). One simulation for each configuration of the percentage of routine devices within the environment (20%, 40%, 60%, 80%, and 100%), thus totaling fifteen simulations to evaluate the predictor results. During this simulation, the client device does not perform any search process; they only use the predictor to indicate where it should look for the computational resource. Afterward, the client device verifies that the prediction was successful in sending a request to the predicted provider device. Then, the client device waits for an answer from the provider device and registers in its log the result of the prediction, the device provider ID, the rating of service, and the timestamp.

In order to analyze the influence of the robustness of the proposal on the tolerance to modifications of the environment, the experiments with different percentages of devices that do not follow any pattern of routines within the simulation environment were repeated.

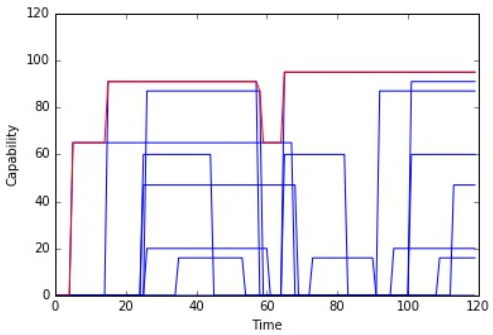
Finally, the experiment is concluded by calculating the metrics described in subsection 5.4.2 for each type of simulated configuration. The results of this experiment are shown in detail from Figures 40, 41, 42, 43, 44, and 45. Figure 46 summarizes the predictor results showing the metrics in different configurations of the percentage of device that follows routines

in FMC environment. The simulations were done using parameters generated from the database of the provider resource produced using provider selection methods randomly selected.

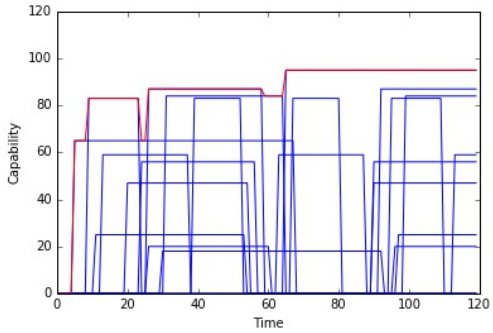
Figure 40 – Time availability of devices by routine percentage settings (client device low storage capability).



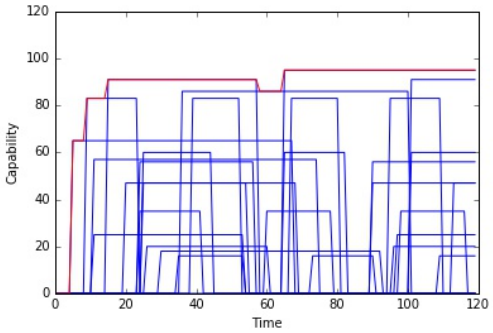
(a) Configuration of 20% of device with routine



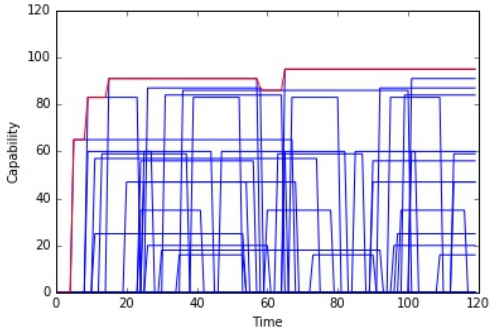
(b) Configuration of 40% of device with routine



(c) Configuration of 60% of device with routine



(d) Configuration of 80% of device with routine

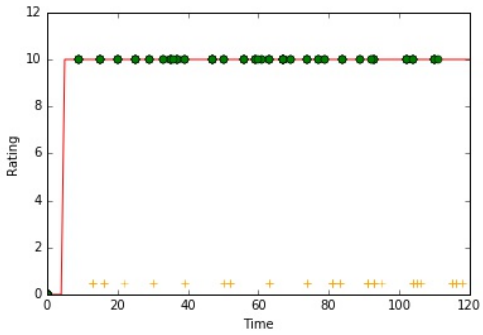


(e) Configuration of 100% of device with routine

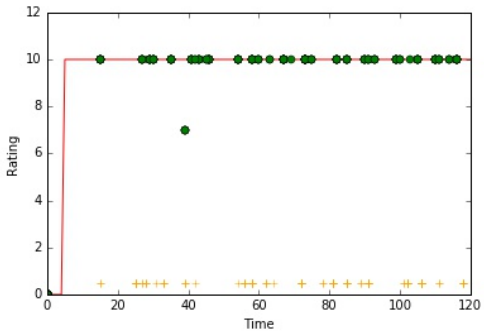
Source – Author.

Figures 41(a), 41(b), 41(c), 41(d), and 41(e) show the temporal availability of devices that have routines within the network in each percentage configuration of devices with routines respectively for the cases of 20%, 40%, 60%, 80%, and 100% in the experiment where the client

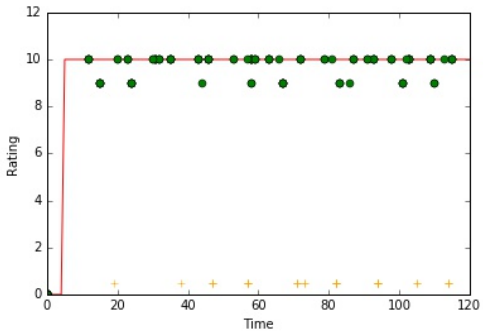
Figure 41 – Predictions of low capability device compared with maximum available score (client device low storage capability). (Green dot = success | Orange cross = Fail | red curve = maximum score available)



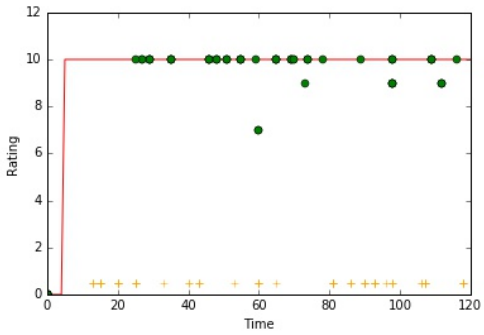
(a) Prediction for percentage of routine equal 20%



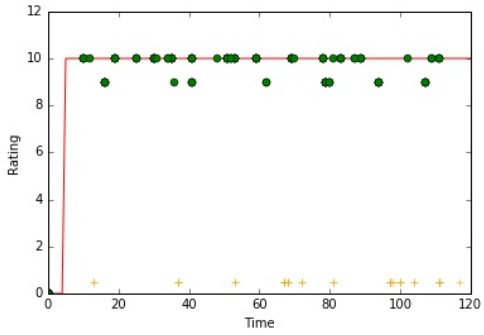
(b) Prediction for percentage of routine equal 40%



(c) Prediction for percentage of routine equal 60%



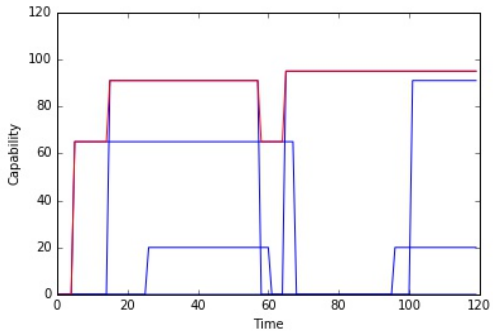
(d) Prediction for percentage of routine equal 80%



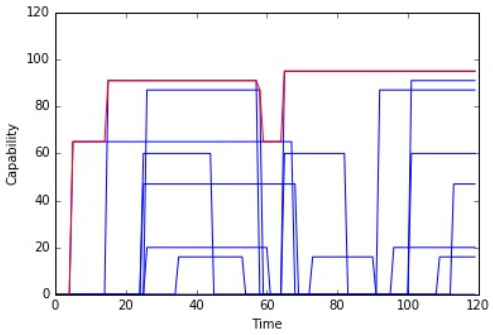
(e) Prediction for percentage of routine equal 100%

Source – Author.

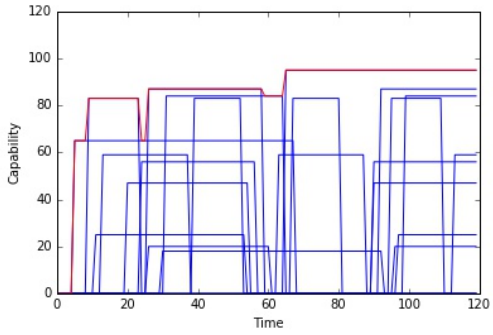
Figure 42 – Time availability of devices by routine percentage settings (client device medium storage capability).



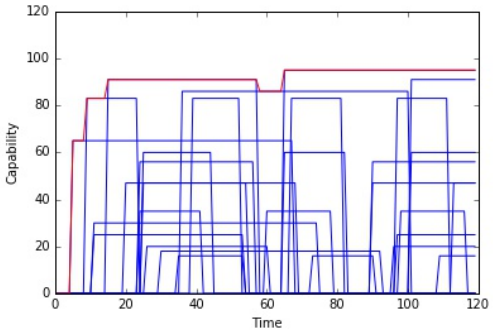
(a) Configuration of 20% of device with routine



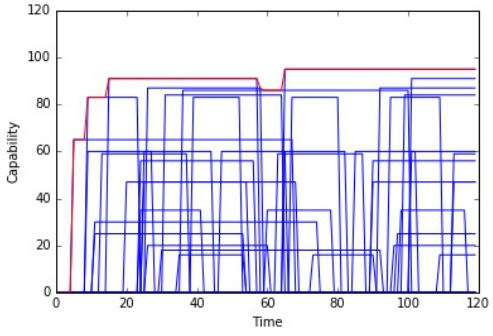
(b) Configuration of 40% of device with routine



(c) Configuration of 60% of device with routine



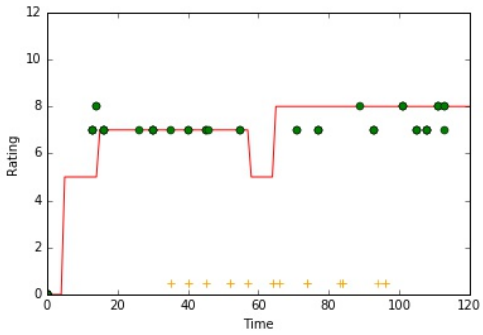
(d) Configuration of 80% of device with routine



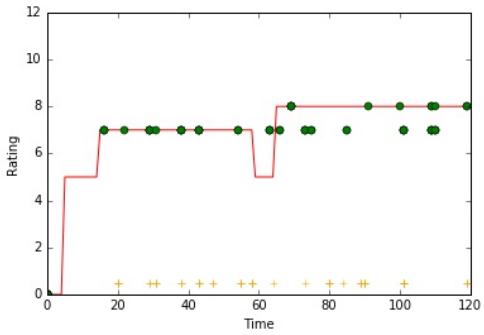
(e) Configuration of 100% of device with routine

Source – Author.

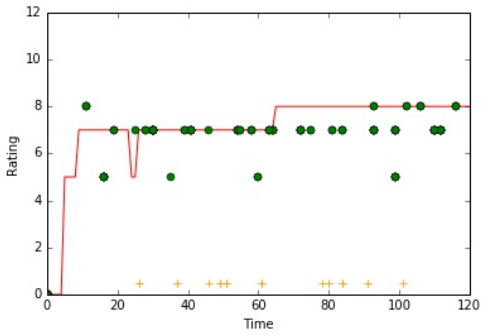
Figure 43 – Predictions of medium capability device compared with maximum available score (client device low storage capability). (Green dot = success | Orange cross = Fail | red curve = maximum score available)



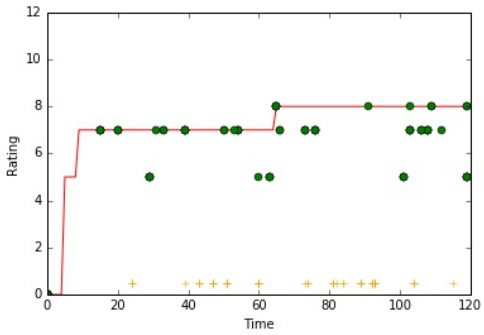
(a) Prediction for percentage of routine equal 20%



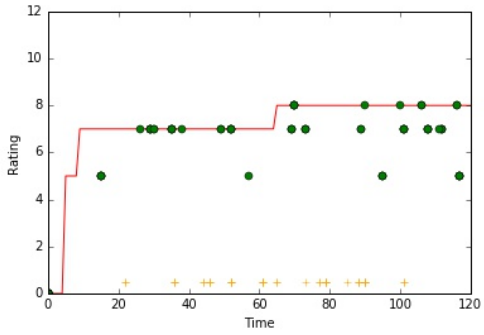
(b) Prediction for percentage of routine equal 40%



(c) Prediction for percentage of routine equal 60%



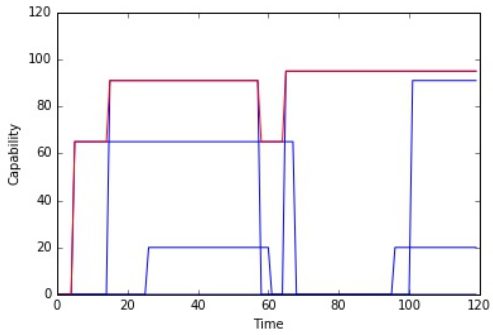
(d) Prediction for percentage of routine equal 80%



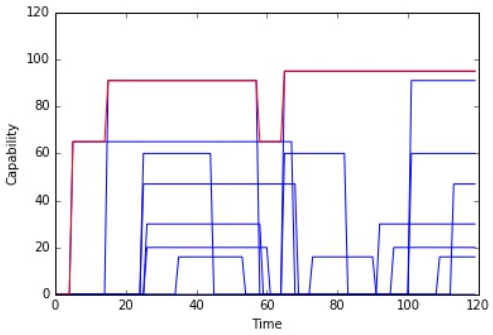
(e) Prediction for percentage of routine equal 100%

Source – Author.

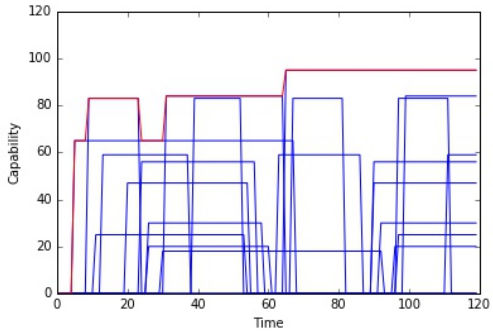
Figure 44 – Time availability of devices by routine percentage settings (client device high storage capability).



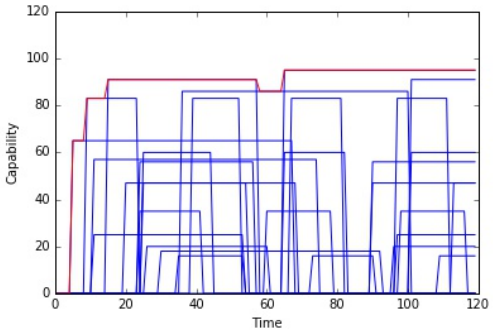
(a) Configuration of 20% of device with routine



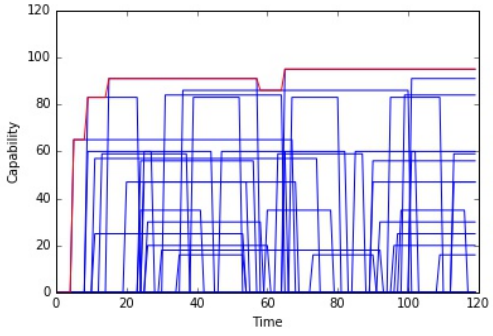
(b) Configuration of 40% of device with routine



(c) Configuration of 60% of device with routine



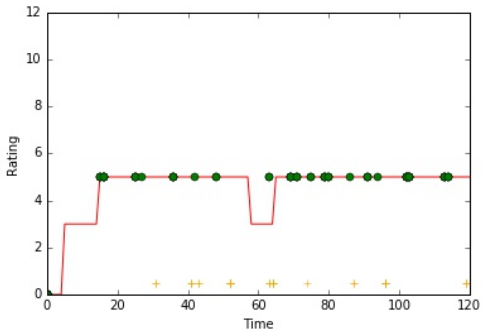
(d) Configuration of 80% of device with routine



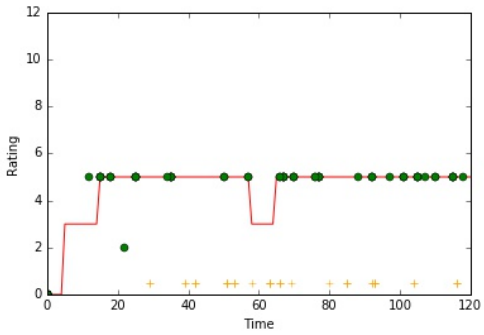
(e) Configuration of 100% of device with routine

Source – Author.

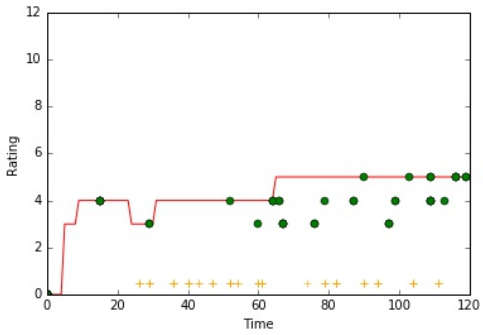
Figure 45 – Predictions of medium capability device compared with maximum available score (client device high storage capability). (Green dot = success | Orange cross = Fail | red curve = maximum score available)



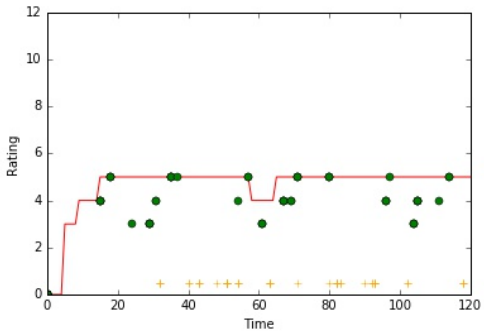
(a) Prediction for percentage of routine equal 20%



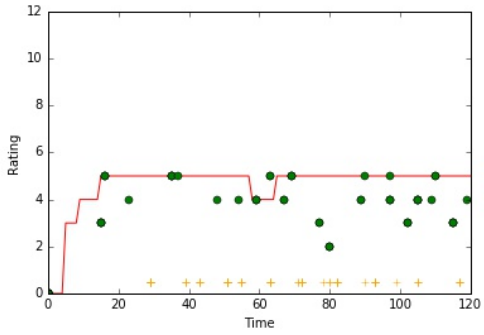
(b) Prediction for percentage of routine equal 40%



(c) Prediction for percentage of routine equal 60%



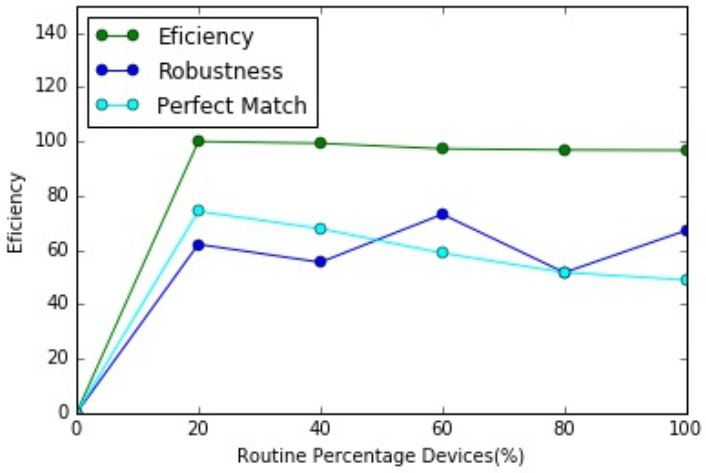
(d) Prediction for percentage of routine equal 80%



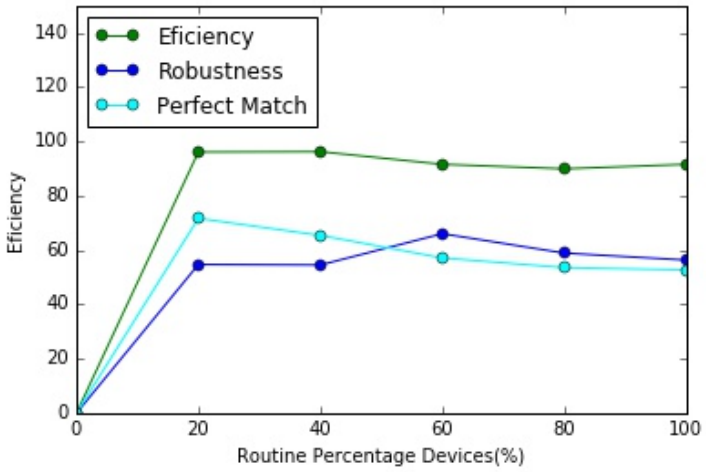
(e) Prediction for percentage of routine equal 100%

Source – Author.

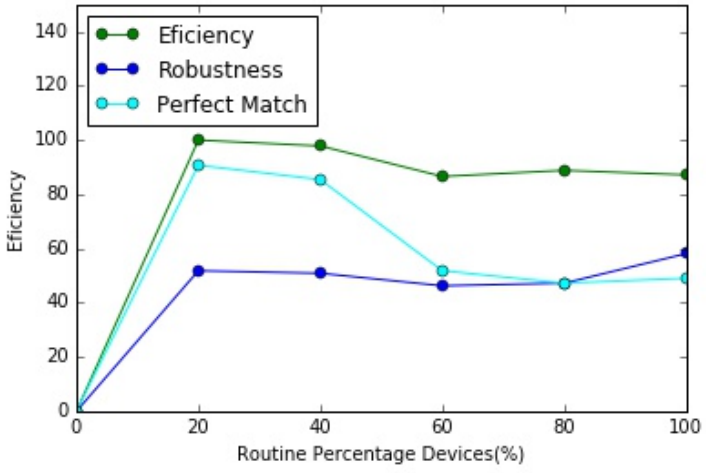
Figure 46 – Metrics results for client devices with low, medium, and high capability storage.



(a) Metric for low storage capability client device.



(b) Metric for medium storage capability client device.



(c) Metric for high storage capability client device.

Source – Author.

has low storage capacity.

Figures 42(a), 42(b), 42(c), 42(d), and 42(e) show maximum possible score in the red curve and overlapping realized predictions over the period of a routine cycle. Each Figure shows the result for each different percentage configuration of devices with routines respectively for the cases of 20%, 40%, 60%, 80%, and 100% in the experiment where the client has low storage capacity. In the graph, the green dots indicate predictions successfully and orange crosses represent predictions fail, in cases where the predicted provider device was not on the network at that time. The perfect match metric is calculated based on green dots that are overlapping the red curve of maximum possible score.

Similarly, Figures 43(a), 43(b), 43(c), 43(d), and 43(e) show the temporal availability of devices that have routines within the network in each percentage configuration of devices with routines respectively for the cases of 20%, 40%, 60%, 80%, and 100% in the experiment where the client has medium storage capacity, and Figures 45(a), 45(b), 45(c), 45(d), and 45(e) show the temporal availability of devices that have routines within the network in each percentage configuration of devices with routines respectively for the cases of 20%, 40%, 60%, 80%, and 100% in the experiment where the client has high storage capacity.

Again, Figures 44(a), 44(b), 44(c), 44(d), and 44(e) show maximum possible score in the red curve and overlapping realized predictions over the period of a routine cycle for each different percentage configuration of devices with routines respectively for the cases of 20%, 40%, 60%, 80%, and 100% in the experiment where the client has medium storage capacity, and Figures 46(a), 46(b), 46(c), 46(d), and 46(e) show maximum possible score in the red curve and overlapping realized predictions over the period of a routine cycle for each different percentage configuration of devices with routines respectively for the cases of 20%, 40%, 60%, 80%, and 100% in the experiment where the client has high storage capacity.

The results of the simulations and source codes used in the data capture phase and the predictor evaluation phase are available at the address <https://github.com/vald3nir/contiki>. The data capture phase simulations were performed with Sky-type motes within an area of 50×50 using a cycle time of 120 seconds. On the other hand, the predictor evaluation phase the simulations needed to use for client devices the mote of type Z1 that has more internal memory to store the RS parameters, the cycle time and device distribution area were the same of data capture phase 120 seconds, and 50×50 , respectively.

5.5 Discussion

Analyzing the results of the predictions shown in the graphs of Figures 41, 43, and 45, it can be verified that the device suggested by the predictor in general for all types of client devices (low, medium, and high storage resources) is a useful suggestion or, in many cases, the best one available in the environment at that instant of time. This fact can also be verified by looking at the metric graphs 46, in which the perfect match metric is, in most cases, higher than 60%, and the efficiency metric exceeds 90% in all tested cases.

Another fact observed in the experiments is that the type of device (low, medium or high capacity) had slight or no influence on the efficiency of the recommendations. Only in the case of the client device with high computational resources was observed a small drop in efficiency, probably due to the chosen evaluation criterion 5.4, in which devices with considerable resources tend to evaluate well only devices with more resources than themselves.

An interesting phenomenon also observed in the experiments was that the perfect match metric in all cases is higher when the percentage of devices that follow a routine in the environment is low; putting it in other words, when the network is more dynamic. Although this may seem strange, this fact can be explained by the ability of the collaborative filter to capture information from the best provider devices in the environment. So, if fewer devices follow a routine and one of them is the best computational resource provider in that instant, then the probability of the predictor hitting this provider increases because the number of available provider devices in the network is smaller than in cases with a higher percentage of devices following a routine, thus explaining the result obtained in the metric of the perfect match.

On the other hand, the experiment also suggests, through the analysis of robustness metrics, that the modeling of the temporal availability of the predictor needs improvement because, if we observed the mean value of this metric during the experiments, the hit rate is of the order of 60%, a slightly higher than an estimate of the expected value for a purely random selection, which would be around 50%.

In summary, the results obtained with the experiments suggest that the predictor was able to successfully capture information from the best provider devices over time, even in a highly dynamic topology environment. The efficiency of the prediction remained high and had little variation even with relatively significant changes in the percentage of devices following routines within the network.

5.6 Summary

In this Chapter, simulation experiments were presented allowing the evaluation of the validity of the proposals submitted regarding the three research questions raised in this thesis.

In Section 5.2, the focus was on evaluating the distributed algorithm to select the best computational resource available between the options Fog/Mist/Cloud computing infrastructure (RQ1), considering the imposed constraints on the client device. The results of these experiments suggest that the proposed feasible Fog concept reduces the search space avoiding, thus, exponential growth at the execution time of the algorithm in large networks. In general, the computing resource chosen by the algorithm was coherent within the scenarios analyzed.

In Section 5.3, the focus was on evaluating the Data dissemination on Fog/Mist environment (RQ2). The results presented in this Chapter suggest that the approach of a network with self-classifying nodes dynamically, assuming roles within the infrastructure of combined bio-inspired techniques of epidemic models, indicates the ability of the algorithm to maintain the availability and distribution of data packets in the environment FMC, even under severe conditions.

In Section 5.4, the smart shadow prediction mechanism (RQ3) was evaluated and results from the simulation experiments suggest that even in environments with high topology dynamics, the algorithm was able to propose good recommendations. Thus, the algorithm was able to capture the knowledge of the devices with satisfying capacity to provide a specific computational resource.

The next chapter concludes this thesis by revisiting the research hypothesis and summarizing the results and publications during the thesis work period. Furthermore, it describes the perspectives of future work related to the contributions presented in Chapter 4.

6 CONCLUSION

This chapter concludes this thesis work, which proposes solutions for predictive computing resources allocation for smart devices in the FMC environment, called Smart Shadow, and it is organized as follows. Section 6.1 depicts the overall goals of this thesis and summarizes its main results. Section 6.2 discusses the hypothesis investigated and compares Smart Shadow with related work. Section 6.3 introduces the limitations of this work. Finally, Section 6.4 presents insights for future work.

6.1 Achieved Goals and Results

In general, the FMC environment brings advantages related to latency and communication bandwidth throughput compared to Cloud Computing. However, due to the dynamicity of its topology and heterogeneity of its devices, the task of choosing the device provider for a given computational resource is very challenging.

Thus, aiming at addressing this gap, the goal of this research was to develop mechanisms that allow the discovery, the selection, and the prediction of the availability over the time of computational resources within the FMC environment, considering the dynamics of the topology and the heterogeneity of the devices. These proposed mechanisms were presented in Chapter 4 and evaluated in Chapter 5.

The main result of this thesis can be then highlight as the mechanism of prediction of temporal availability of computational resources in network FMC with a decentralized adaptive architecture based on collaborative filters. It is important to note that devices or part of them are considered to follow a routine within a period T in the FMC environment and that analyzed temporal availability is related with the routine period mentioned.

Moreover, during this research, in order to achieve the main goal of this work, secondary results were also reached and they are summarized as follows:

- Mathematical models to address the problem of infrastructure selection, to keep data in a dynamic topology, and to predict computing resource availability in the Fog/Mist computing environment;
- A systematic mechanism for discovering computational resources in the FMC environment that meets constraints imposed on these computational resources;
- The feasible Fog concept, based on latency constraints, which reduces the search space by

allowing searching in decentralized environments even with a large number of devices; and

- A bio-inspired adaptive mechanism of data dissemination within FMC networks with high dynamicity in the topology.

Table 12 – Publications from this thesis work

Paper	**Event/Journal**	**Note**
Vasconcelos, D.R; Andrade, R.M.C; Souza, J.N. - ”Smart Shadow - An autonomous availability computation resource allocation platform for Internet of Things in the Fog computing environment”, DCOSS 2015 - International Conference on Distributed Computing in Sensor System, Fortaleza, 2015.	IEEE DCOSS-2015. - The 11th International Conference on Distributed Computing in Sensor Systems.	Paper presented at the PhD Forum.
Vasconcelos, D.R; Pimentel, F.L.R; Andrade, R.M.C; Souza,J.N. - ”Mathematical model for a Collaborative Indoor Position System (IPS) and movement detection of devices within IoT environment” - 32nd ACM SIGAPP Symposium On Applied Computing, Marrakech, Morocco,April 4-6, 2017.	SAC 2015. - 32nd ACM SIGAPP Symposium on Applied Computing.	Paper presented at the congress in Marrocos.
VASCONCELOS, D.R; ANDRADE, R.M.C.; SEVERINO, V.; MAIA, M.E.F.; SOUZA, J.N. - ”Bio-inspired model for data distribution in Fog and Mist computing” - Compsac 2018 - 42nd IEEE Computer Society Signature Conference on Computers, Software and Applications.	Compsac 2018 - 42nd IEEE Computer Society Signature Conference on Computers, Software and Applications.	Paper presented at the congress (results related to RQ2 of this thesis).
VASCONCELOS,D.R; ANDRADE, R.M.C.; SEVERINO, V.; SOUZA, J.N. - ”Cloud, Fog or Mist in IoT, that is the question”- ACM Transactions of Internet Technology (TOIT) - Special Issue ”Fog, Edge, and Cloud Integration for Smart Environments”.	Journal ACM Transactions of Internet Technology (TOIT) - Special Issue on ”Fog, Edge, and Cloud Integration for Smart Environments”.	Accepted with Major revision and awaiting the result of the second revision (One result of RQ1 of this thesis).

Source – Author.

Furthermore, three papers were published in conferences and another one is under review in a journal from the research performed in this work. Also, the author intends to write another article containing the proposal and results on the predictor (RQ3) to be submitted to a well-evaluated scientific journal. Table 12 presents the references of these papers.

The first paper (VASCONCELOS *et al.*, 2015) presents the ideas at the conceptual

level of the proposal of this thesis and it was presented at a doctoral symposium. The next paper (VASCONCELOS *et al.*, 2017) is not directly related to the questions of this thesis, but it was the result of the research carried out to define the theme of this thesis. The third article (VASCONCELOS *et al.*, 2018) was presented at the COMPSAC-2018 workshop in Tokyo-Japan, which was a direct result of RQ2 of this thesis. The last paper has not been accepted for publication yet; however, the modified article has been sent with all the suggestions presented by the reviewers to major revision. The current status is awaiting the result of the second review for publication in the ACM TOIT magazine.

6.2 Revisiting Research Questions and Hypothesis

By analyzing works related to the research questions of this thesis, gaps were identified for all three questions, especially when considering characteristics such as the dynamism of the topology of the network environment of this thesis. In the literature, just a few papers that address Mist computing are identified, thus, showing another evidence of research gap on this issue.

On the work related to RQ1, one can clearly identify an absence of proposals that are decentralized and using resources from Fog itself or Mist computing. It can be also observed that the great majority of the works focuses on a specific type of restriction, such as latency, and propose a mechanism to optimize the allocation of computational resources, with few proposals such as that of this thesis that addresses the problem considering multiple constraints. So, Table 13 presents the main characteristics of this thesis proposal related to RQ1.

Table 13 – Thesis proposal for RQ1.

Author	Proposal	Processing type	Optimized Parameters
This thesis	Mathematical Model for computer resource allocation in Fog, Mist or Cloud computing	Distributed	Multicriteria (Latency, cost, Bandwidth, ...)

Source – Author.

In the same way, considering works related to RQ2, it is possible to identify only few works that address the problem of the dynamism of the topology, and none of them proposes solutions that consider the temporal availability of devices within a pattern of routines of use. Moreover, it can be observed that few proposals offer solutions that are executed in the environment of Fog / Mist within a decentralized architecture. In short, Table 14 presents the

main characteristics of this thesis proposal related to RQ2.

Table 14 – Thesis proposal for RQ2.

Author	Proposal	Architecture	Pre-defined rules for nodes	Network context
This thesis	A distributed algorithm using ideas from evolutionary computation and epidemic model for data availability and dissemination	Decentralized	No	Fog and Mist

Source – Author.

On the other hand, when RQ3 is considered, it is possible to find works (Forestiero (2017), Nizamkari (2017), and Asiri e Miri (2016)) similar to the proposal of this thesis. However, when the works are deeply analyzed, issues that are not addressed by these works are identified, but they are addressed in the proposal of this thesis. Among them, the adaptability of the dynamism of the proposed structure can be highlighted - in which the nodes play a dynamic role by self-classifying and adapting to changes in network topology, considering the prediction rating and the temporal availability of the device. The other works generally have a centralized architecture, using the devices only as data entry point and user interface of the system. The recommendation system runs in a central architecture that processes the data and generates a recommendation to the users. Therefore, they do not use computational resources available at the edge of the network for the processing of the recommendation. Table 15 presents the main characteristics of the proposal of this thesis related to RQ3.

Table 15 – Thesis proposal for RQ3.

Author	Proposal	Architecture	Topology dynamism	Temporal dynamism	Node rules	Local data	Network context	Focus User/Things
This thesis	Hybrid collaborative recommendation system distributed with nodes that assumes role dynamically according to their time in the environment	Dist	Yes	Yes	A	Yes	F/M	Thing

Source – Author.

The research hypothesis, which guided this thesis work and is presented in Chapter 1, is now analyzed (accept or reject) based on the proposals presented in Chapter 4 and the

evaluations described in Chapter 5 as follows:

Research Hypothesis: *It is possible to make a predictive discovery and allocation of the computational resources to smart devices in the context of Fog and Mist Computing using data locally stored, in an efficient and robust manner.*

Accepted. Based on the evaluations presented in Chapter 5, in particular, Section 5.3, it was possible to gather evidence that even in a network with high dynamics of topology, it is possible to maintain data locally in this environment. These results support the excerpt from the hypothesis that mentions *" using data locally stored"*. Furthermore, it was also possible to observe, based on the results shown in Section 5.4, that the approach using collaborative filters combined with predictive models of the temporal availability of the devices in FMC environments, proposed in Chapter 4, Section 4.4, presented good results in predicting the best devices available to offer a particular computational resource. This second part of the results provides evidence to support the excerpt from the hypothesis that mentions *"efficient and robust manner"*.

Thus, the proposed theory model, the simulations results, and the empirical evidence obtained support the acceptance of the hypothesis proposed in this thesis.

6.3 Limitations

The proposal presented in this thesis has the following limitations:

- It was not possible to test algorithms and protocol with a large number of devices due to the simulator performance limitations for environments with more than one hundred nodes;
- The LL-type devices must have sufficient memory to store the matrices used by the predictor. Note that the dimension of the matrices are $n \times n$, where n is the total number of devices in their neighborhood. This limitation becomes relevant in environments with a high density of devices and compounds with storage limitations; and
- The test environment used with simulation did not consider external interferences and adverse environmental conditions such as obstacles, metal areas, reflections, ans so on, that in some cases can happen in wireless networks.

6.4 Future Work

With the aim of solving some of the limitations presented in the previous section as well as evolving the proposals presented in this thesis, it is suggested the following research proposals as future work:

- To address the cold start problem in the predictor model, which occurs when there is a small number of ratings performed by the devices. Thus, the recommendations should be made only after the system has a considerable data mass for it to be effective. In other words, to create mechanisms that allow the predictor to generate good recommendations even if there is not a large amount of data from previous recommendations available;
- To optimize the energy consumption used in the devices involved in the protocols using, for example, distributed optimization techniques;
- In the proposal, all devices involved are considered trusted and important security-related issues such as authenticity, integrity, and privacy of information were not considered. Thus, it is important to analyze security-related issues such as malicious device attacks by incorporating authentication, trust, and privacy mechanisms into the proposal algorithms and protocols;
- To validate the proposals of this thesis in a real environment, evaluating the effectiveness in different contexts of the IoT environments;
- Improve the temporal availability estimator of the devices in FMC networks using neural networks; and
- To study the merging and division of FMC environments and the resulting behavior of the algorithm after these operations.

REFERENCES

AAZAM, M.; HUH, E.-N. Fog computing micro datacenter based dynamic resource estimation and pricing model for iot. In: IEEE. **AINA 2015 IEEE 29th International Conference on**. [S.l.], 2015. p. 687–694.

AAZAM, M.; HUH, E. N. Fog computing: The cloud-iot/ioe middleware paradigm. **IEEE Potentials**, v. 35, n. 3, p. 40–44, May 2016. ISSN 0278-6648.

AAZAM, M.; ST-HILAIRE, M.; LUNG, C.-H.; LAMBADARIS, I. Pre-fog: Iot trace based probabilistic resource estimation at fog. In: IEEE. **CCNC, 2016 13th IEEE Annual**. [S.l.], 2016. p. 12–17.

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. **IEEE transactions on knowledge and data engineering**, IEEE, v. 17, n. 6, p. 734–749, 2005.

ALAG, S. **Collective intelligence in action**. [S.l.]: Manning Publications Co., 2008.

ANDERSSON, H.; BRITTON, T. **Stochastic epidemic models and their statistical analysis**. [S.l.]: Springer Science & Business Media, 2012. v. 151.

ANDERSSON, H.; BRITTON, T. **Stochastic Epidemic Models and Their Statistical Analysis**. [S.l.]: Springer New York, 2012. (Lecture Notes in Statistics). ISBN 9781461211587.

ARMBRUST, M.; FOX, A.; GRIFFITH, R.; JOSEPH, A. D.; KATZ, R.; KONWINSKI, A.; LEE, G.; PATTERSON, D.; RABKIN, A.; STOICA, I. *et al.* A view of cloud computing. **Communications of the ACM**, ACM, v. 53, n. 4, p. 50–58, 2010.

ARRATIA A., F.-i.-C. R. Epidemic models over networks. In: ____. [S.l.: s.n.], 2017. Available at: https://www.cs.upc.edu/ CSN/slides/11epidemic.pdf. Last accessed on 2018-03-25.

ASIRI, S.; MIRI, A. An iot trust and reputation model based on recommender systems. In: IEEE. **Privacy, Security and Trust (PST), 2016 14th Annual Conference on**. [S.l.], 2016. p. 561–568.

AWERBUCH, B. A new distributed depth-first-search algorithm. **Information Processing Letters**, Elsevier, v. 20, n. 3, p. 147–150, 1985.

BAILEY, N. T. *et al.* **The mathematical theory of infectious diseases and its applications**. [S.l.]: Charles Griffin & Company Ltd, 5a Crendon Street, High Wycombe, Bucks HP13 6LE., 1975.

BALABANOVIĆ, M.; SHOHAM, Y. Fab: content-based, collaborative recommendation. **Communications of the ACM**, ACM, v. 40, n. 3, p. 66–72, 1997.

BASU, C. Recommendation as classification: Using social and content-based information in recommendation chumki basu. 1998.

BILAL, K.; KHALID, O.; ERBAD, A.; KHAN, S. U. Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers. **Computer Networks**, Elsevier, v. 130, p. 94–120, 2018.

BILLSUS, D.; PAZZANI, M. J. Learning collaborative information filters. In: **Icml**. [S.l.: s.n.], 1998. v. 98, p. 46–54.

BILLSUS, D.; PAZZANI, M. J. User modeling for adaptive news access. **User modeling and user-adapted interaction**, Springer, v. 10, n. 2-3, p. 147–180, 2000.

BONOMI, F.; MILITO, R.; ZHU, J.; ADDEPALLI, S. Fog computing and its role in the internet of things. In: **Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing**. New York, NY, USA: ACM, 2012. (MCC '12), p. 13–16. ISBN 978-1-4503-1519-7.

BOTTA, A.; DONATO, W. D.; PERSICO, V.; PESCAPÉ, A. Integration of cloud computing and internet of things: a survey. **Future Generation Computer Systems**, Elsevier, v. 56, p. 684–700, 2016.

BREESE, J. S.; HECKERMAN, D.; KADIE, C. Empirical analysis of predictive algorithms for collaborative filtering. In: MORGAN KAUFMANN PUBLISHERS INC. **Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence**. [S.l.], 1998. p. 43–52.

BRITTON, T. Stochastic epidemic models: a survey. **Mathematical biosciences**, Elsevier, v. 225, n. 1, p. 24–35, 2010.

BRUNEO, D.; DISTEFANO, S.; LONGO, F.; MERLINO, G.; PULIAFITO, A.; D'AMICO, V.; SAPIENZA, M.; TORRISI, G. Stack4things as a fog computing platform for smart city applications. In: **2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)**. [S.l.: s.n.], 2016. p. 848–853.

CHA, S.; RUIZ, M. P.; WACHOWICZ, M.; TRAN, L. H.; CAO, H.; MADUAKO, I. The role of an iot platform in the design of real-time recommender systems. In: IEEE. **Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on**. [S.l.], 2016. p. 448–453.

CHAQFEH, M.; LAKAS, A.; JAWHAR, I. A survey on data dissemination in vehicular ad hoc networks. **Vehicular Communications**, v. 1, n. 4, p. 214 – 225, 2014. ISSN 2214-2096.

CHEN, R.-Y. An intelligent value stream-based approach to collaboration of food traceability cyber physical system by fog computing. **Food Control**, Elsevier, v. 71, p. 124–136, 2017.

CHEN, W.; GUHA, R. K.; KWON, T. J.; LEE, J.; HSU, I. Y. A survey and challenges in routing and data dissemination in vehicular ad-hoc networks. In: **2008 IEEE International Conference on Vehicular Electronics and Safety**. [S.l.: s.n.], 2008. p. 328–333.

CHIANG, M.; ZHANG, T. Fog and iot: An overview of research opportunities. **IEEE Internet of Things Journal**, IEEE, v. 3, n. 6, p. 854–864, 2016.

CLAYPOOL, M.; GOKHALE, A.; MIRANDA, T.; MURNIKOV, P.; NETES, D.; SARTIN, M. Combing content-based and collaborative filters in an online newspaper. 1999.

CONDLI, M. K.; LEWIS, D. D.; MADIGAN, D.; POSSE, C. Bayesian mixed-e ects models for recommender systems. In: **ACM SIGIR**. [S.l.: s.n.], 1999. v. 99.

CONFAIS, B.; LEBRE, A.; PARREIN, B. Performance analysis of object store systems in a fog/edge computing infrastructures. In: **2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)**. [S.l.: s.n.], 2016. p. 294–301.

CUOMO, S.; MICHELE, P. D.; PICCIALLI, F.; GALLETTI, A.; JUNG, J. E. Iot-based collaborative reputation system for associating visitors and artworks in a cultural scenario. **Expert Systems with Applications**, Elsevier, v. 79, p. 101–111, 2017.

DASGUPTA, A.; GILL, A. Q. Fog computing challenges: A systematic review. In: **Australasian Conference on Information Systems**. [S.l.: s.n.], 2017.

DELGADO, J.; ISHII, N. Memory-based weighted majority prediction. In: **SIGIR Workshop Recomm. Syst. Citeseer**. [S.l.: s.n.], 1999.

DENG, R.; LU, R.; LAI, C.; LUAN, T. H.; LIANG, H. Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption. **IEEE Internet of Things Journal**, PP, n. 99, p. 1–1, 2016. ISSN 2327-4662.

DIEKMANN, O.; HEESTERBEEK, J. A. P. **Mathematical epidemiology of infectious diseases: model building, analysis and interpretation**. [S.l.]: John Wiley & Sons, 2000. v. 5.

DOLUI, K.; DATTA, S. K. Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing. In: IEEE. **Global Internet of Things Summit (GIoTS), 2017**. [S.l.], 2017. p. 1–6.

DOLUI, K.; DATTA, S. K. Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing. In: **2017 Global Internet of Things Summit (GIoTS)**. [S.l.: s.n.], 2017. p. 1–6.

DORIGO, M.; BIRATTARI, M.; STUTZLE, T. Ant colony optimization. **IEEE Computational Intelligence Magazine**, v. 1, n. 4, p. 28–39, Nov 2006. ISSN 1556-603X.

DUNKELS, A.; GRONVALL, B.; VOIGT, T. Contiki - a lightweight and flexible operating system for tiny networked sensors. In: **Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks**. Washington, DC, USA: IEEE Computer Society, 2004. (LCN '04), p. 455–462. ISBN 0-7695-2260-2.

EKSTRAND, M. D.; RIEDL, J. T.; KONSTAN, J. A. *et al.* Collaborative filtering recommender systems. **Foundations and Trends® in Human–Computer Interaction**, Now Publishers, Inc., v. 4, n. 2, p. 81–173, 2011.

FEKI, S.; LOUATI, W.; MASMOUDI, N.; JMAIEL, M. Q-learning-based data replication for highly dynamic distributed hash tables. In: **2014 International Conference and Workshop on the Network of the Future (NOF)**. [S.l.: s.n.], 2014. p. 1–5.

FORESTIERO, A. Multi-agent recommendation system in internet of things. In: IEEE. **Cluster, Cloud and Grid Computing (CCGRID), 2017 17th IEEE/ACM International Symposium on**. [S.l.], 2017. p. 772–775.

FREY, R. M.; XU, R.; ILIC, A. A novel recommender system in iot. In: IEEE. **2015 5th International Conference on the Internet of Things (IOT 2015): Seoul, South Korea, 26-28 October 2015**. [S.l.], 2015.

FUNK, S. Netflix update: Try this at home (december 2006). **URL http://sifter. org/˜ simon/journal/20061211. html**, 2006.

GILES, J. **Internet encyclopaedias go head to head**. [S.l.]: Nature Publishing Group, 2005.

GOLUB, G. H.; REINSCH, C. Singular value decomposition and least squares solutions. **Numerische mathematik**, Springer, v. 14, n. 5, p. 403–420, 1970.

GOOD, N.; SCHAFER, J. B.; KONSTAN, J. A.; BORCHERS, A.; SARWAR, B.; HERLOCKER, J.; RIEDL, J. *et al.* Combining collaborative filtering with personal agents for better recommendations. In: **AAAI/IAAI**. [S.l.: s.n.], 1999. p. 439–446.

GUO, S. *et al.* Analysis and evaluation of similarity metrics in collaborative filtering recommender system. Lapin ammattikorkeakoulu, 2014.

HEYLIGHEN, F. Collective intelligence and its implementation on the web: Algorithms to develop a collective mental map. **Computational & Mathematical Organization Theory**, v. 5, n. 3, p. 253–280, Oct 1999. ISSN 1572-9346.

HILL, W.; STEAD, L.; ROSENSTEIN, M.; FURNAS, G. Recommending and evaluating choices in a virtual community of use. In: ACM PRESS/ADDISON-WESLEY PUBLISHING CO. **Proceedings of the SIGCHI conference on Human factors in computing systems**. [S.l.], 1995. p. 194–201.

HOFMANN, T. Probabilistic latent semantic analysis. In: MORGAN KAUFMANN PUBLISHERS INC. **Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence**. [S.l.], 1999. p. 289–296.

HOLLAND, J. H. **Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence**. Cambridge, MA, USA: MIT Press, 1992. ISBN 0262082136.

HU, Y. C.; PATEL, M.; SABELLA, D.; SPRECHER, N.; YOUNG, V. Mobile edge computing—a key technology towards 5g. **ETSI white paper**, v. 11, n. 11, p. 1–16, 2015.

JANNACH, D.; ZANKER, M.; FELFERNIG, A.; FRIEDRICH, G. **Recommender systems: an introduction**. [S.l.]: Cambridge University Press, 2010.

KAR, A. K. Bio inspired computing–a review of algorithms and scope of applications. **Expert Systems with Applications**, Elsevier, v. 59, p. 20–32, 2016.

KEELING, M. J.; EAMES, K. T. Networks and epidemic models. **Journal of the Royal Society Interface**, The Royal Society, v. 2, n. 4, p. 295–307, 2005.

KLEIN, A.; ISHIKAWA, F.; HONIDEN, S. Towards network-aware service composition in the cloud. In: ACM. **WWW 2012 Proceedings of the 21st international conference**. [S.l.], 2012. p. 959–968.

KUMAR, A.; NARENDRA, N. C.; BELLUR, U. Uploading and replicating internet of things (iot) data on distributed cloud storage. In: **2016 IEEE 9th International Conference on Cloud Computing (CLOUD)**. [S.l.: s.n.], 2016. p. 670–677.

LEI, T.; WANG, S.; LI, J.; YANG, F. Aom: adaptive mobile data traffic offloading for m2m networks. **Personal and Ubiquitous Computing**, Springer, v. 20, n. 6, p. 863–873, 2016.

LI, T.; ZHAO, M.; LIU, A.; HUANG, C. On selecting vehicles as recommenders for vehicular social networks. **IEEE Access**, IEEE, v. 5, p. 5539–5555, 2017.

LIAO, J.; LU, K.; CAO, Q. Uno: A privacy-aware distributed storage and replication middleware for heterogeneous computing platforms. In: **2013 IEEE 10th International Conference on Mobile Ad-Hoc and Sensor Systems**. [S.l.: s.n.], 2013. p. 551–559. ISSN 2155-6806.

LIESKOVSKY, A.; JANECH, J.; BACA, T. Data replication in distributed database systems in vanet environment. In: **2011 11th International Conference on ITS Telecommunications**. [S.l.: s.n.], 2011. p. 447–452.

LIYANAGE, M.; CHANG, C.; SRIRAMA, S. N. mepaas: Mobile-embedded platform as a service for distributing fog computing to edge nodes. In: IEEE. **Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2016 17th International Conference on**. [S.l.], 2016. p. 73–80.

LOPEZ, P. G.; MONTRESOR, A.; EPEMA, D.; DATTA, A.; HIGASHINO, T.; IAMNITCHI, A.; BARCELLOS, M.; FELBER, P.; RIVIERE, E. Edge-centric computing: Vision and challenges. **ACM SIGCOMM Computer Communication Review**, ACM, v. 45, n. 5, p. 37–42, 2015.

LUISI, P. **The Emergence of Life: From Chemical Origins to Synthetic Biology Cambridge Univ**. [S.l.]: Press, 2006.

LYKOURENTZOU, I.; VERGADOS, D. J.; LOUMOS, V. Collective intelligence system engineering. In: ACM. **Proceedings of the international conference on management of emergent digital ecosystems**. [S.l.], 2009. p. 20.

MACGILLIVRAY, C.; TURNER, V.; LAMY, L.; PROUTY, K.; SEGAL, R.; SIVIERO, A.; TORCHIA, M.; VESSET, D.; WESTERVELT, R.; CLARKE, R. Y. IDC FUTURESCAPE FIGURE FIGURE 1 IDC FutureScape: Worldwide Internet of Things 2017 Top 10 Predictions IN THIS EXCERPT IDC FUTURESCAPE PREDICTIONS. 2017. Available at: https://cdn2.hubspot.net/hubfs/1946517/IDC FutureScape - Worldwide Internet of Things 2017 Predictions.pdf?t=1493899238617. Last accessed on 2018-04-01.

MACKEY, L. **Collaborative Filtering**. 2009. Available at: https://web.stanford.edu/$\sim$ $lmackey/papers/cfslides-pml09.pdf.Lastaccessedon2018-03-31$.

MADAKAM, S.; RAMASWAMY, R.; TRIPATHI, S. Internet of things (iot): A literature review. **Journal of Computer and Communications**, Scientific Research Publishing, v. 3, n. 05, p. 164, 2015.

MADSEN, H.; BURTSCHY, B.; ALBEANU, G.; POPENTIU-VLADICESCU, F. Reliability in the utility computing era: Towards reliable fog computing. In: IEEE. **Systems, Signals and Image Processing (IWSSIP), 2013 20th International Conference on**. [S.l.], 2013. p. 43–46.

MANGE, D.; TOMASSINI, M. **Bio-inspired computing machines: Towards novel computational architectures**. [S.l.]: PPUR presses polytechniques, 1998.

MARTIN, M. J. Cloud, fog, and now, mist computing. In: ____. [S.l.: s.n.], 2015. Available at: https://www.linkedin.com/pulse/cloud-computingfog-now-mist-martin-ma-mba-med-gdm-scpm-pmp. Last accessed on 2018-03-22.

MASHAL, I.; ALSARYRAH, O.; CHUNG, T.-Y. Analysis of recommendation algorithms for internet of things. In: IEEE. **Wireless Communications and Networking Conference Workshops (WCNCW), 2016 IEEE**. [S.l.], 2016. p. 181–186.

MASHAL, I.; ALSARYRAH, O.; CHUNG, T.-Y. Performance evaluation of recommendation algorithms on internet of things services. **Physica A: Statistical Mechanics and its Applications**, Elsevier, v. 451, p. 646–656, 2016.

MASHAL, I.; CHUNG, T.-Y.; ALSARYRAH, O. Toward service recommendation in internet of things. In: IEEE. **Ubiquitous and Future Networks (ICUFN), 2015 Seventh International Conference on**. [S.l.], 2015. p. 328–331.

MASIP-BRUIN, X.; MARíN-TORDERA, E.; TASHAKOR, G.; JUKAN, A.; REN, G. J. Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems. **IEEE Wireless Communications**, v. 23, n. 5, p. 120–128, October 2016. ISSN 1536-1284.

MELL, P.; GRANCE, T. *et al.* The nist definition of cloud computing. **National institute of standards and technology**, v. 53, n. 6, p. 50, 2009.

MOONEY, R. J.; BENNETT, P. N.; ROY, L. Book recommending using text categorization with extracted information. In: **Proc. Recommender Systems Papers from 1998 Workshop, Technical Report WS-98-08**. [S.l.: s.n.], 1998.

MUNOZ-ORGANERO, M.; RAMÍREZ-GONZÁLEZ, G. A.; MUNOZ-MERINO, P. J.; KLOOS, C. D. A collaborative recommender system based on space-time similarities. **IEEE Pervasive Computing**, IEEE, v. 9, n. 3, p. 81–87, 2010.

NAKAMURA, A.; ABE, N. Collaborative filtering using weighted majority prediction algorithms. In: **ICML**. [S.l.: s.n.], 1998. v. 98, p. 395–403.

NARENDRA, N. C.; KOORAPATI, K.; UJJA, V. Towards cloud-based decentralized storage for internet of things data. In: **2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)**. [S.l.: s.n.], 2015. p. 160–168.

NEWMAN, M. E. Spread of epidemic disease on networks. **Physical review E**, APS, v. 66, n. 1, p. 016128, 2002.

NIZAMKARI, N. S. A graph-based trust-enhanced recommender system for service selection in iot. In: IEEE. **Inventive Systems and Control (ICISC), 2017 International Conference on**. [S.l.], 2017. p. 1–5.

PAZZANI, M.; BILLSUS, D. Learning and revising user profiles: The identification of interesting web sites. **Machine learning**, Springer, v. 27, n. 3, p. 313–331, 1997.

PAZZANI, M. J. A framework for collaborative, content-based and demographic filtering. **Artificial intelligence review**, Springer, v. 13, n. 5-6, p. 393–408, 1999.

PAZZANI, M. J.; BILLSUS, D. Content-based recommendation systems. In: **The adaptive web**. [S.l.]: Springer, 2007. p. 325–341.

PEREIRA, C.; PINTO, A.; FERREIRA, D.; AGUIAR, A. Experimental characterization of mobile iot application latency. **IEEE IoT Journal**, IEEE, v. 4, n. 4, p. 1082–1094, 2017.

PFLANZNER, T.; KERTESZ, A. A survey of iot cloud providers. In: **2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)**. [S.l.: s.n.], 2016. p. 730–735.

PHAM, X.-Q.; HUH, E.-N. Towards task scheduling in a cloud-fog computing system. In: **2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)**. [S.l.: s.n.], 2016. p. 1–4.

PIOTROWSKI, K.; LANGENDOERFER, P.; PETER, S. tinydsm: A highly reliable cooperative data storage for wireless sensor networks. In: **2009 International Symposium on Collaborative Technologies and Systems**. [S.l.: s.n.], 2009. p. 225–232.

POPESCUL, A.; PENNOCK, D. M.; LAWRENCE, S. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In: MORGAN KAUFMANN PUBLISHERS INC. **Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence**. [S.l.], 2001. p. 437–444.

PREDEN, J. S.; TAMMEMäE, K.; JANTSCH, A.; LEIER, M.; RIID, A.; CALIS, E. The benefits of self-awareness and attention in fog and mist computing. **Computer**, v. 48, n. 7, p. 37–45, July 2015. ISSN 0018-9162.

RAMASWAMY, L.; DEEPAK, P.; POLAVARAPU, R.; GUNASEKERA, K.; GARG, D.; VISWESWARIAH, K.; KALYANARAMAN, S. Caesar: A context-aware, social recommender system for low-end mobile devices. In: IEEE. **Mobile Data Management: Systems, Services and Middleware, 2009. MDM'09. Tenth International Conference on**. [S.l.], 2009. p. 338–347.

RENJITH, S.; ANJALI, C. A personalized mobile travel recommender system using hybrid algorithm. In: IEEE. **Computational Systems and Communications (ICCSC), 2014 First International Conference on**. [S.l.], 2014. p. 12–17.

RESNICK, P.; IACOVOU, N.; SUCHAK, M.; BERGSTROM, P.; RIEDL, J. Grouplens: an open architecture for collaborative filtering of netnews. In: ACM. **Proceedings of the 1994 ACM conference on Computer supported cooperative work**. [S.l.], 1994. p. 175–186.

RICCI, F. Mobile recommender systems. **Information Technology & Tourism**, Cognizant Communication Corporation, v. 12, n. 3, p. 205–231, 2010.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: **Recommender systems handbook**. [S.l.]: Springer, 2011. p. 1–35.

SALTON, G. Automatic text processing: The transformation, analysis, and retrieval of. **Reading: Addison-Wesley**, 1989.

SARKAR, S. M. S. Theoretical modelling of fog computing: a green computing paradigm to support iot applications. **IET Networks**, Institution of Engineering and Technology, v. 5, p. 23–29(6), March 2016. ISSN 2047-4954. Available at: <http://digital-library.theiet.org/content/journals/10.1049/iet-net.2015.0034>.

SARWAR, B.; KARYPIS, G.; KONSTAN, J.; RIEDL, J. Item-based collaborative filtering recommendation algorithms. In: ACM. **Proceedings of the 10th international conference on World Wide Web**. [S.l.], 2001. p. 285–295.

SAWANT, S. D.; SONAWANE, K. V.; JAGANI, T.; CHAUDHARI, A. N. Representation of recommender system in iot using cyber physical techniques. In: IEEE. **Electronics, Communication and Aerospace Technology (ICECA), 2017 International conference of**. [S.l.], 2017. v. 2, p. 372–375.

SCHEIN, A. I.; POPESCUL, A.; UNGAR, L. H.; PENNOCK, D. M. Methods and metrics for cold-start recommendations. In: ACM. **Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval**. [S.l.], 2002. p. 253–260.

SEGARAN, T. **Programming collective intelligence: building smart web 2.0 applications**. [S.l.]: " O'Reilly Media, Inc.", 2007.

SHARDANAND, U.; MAES, P. Social information filtering: algorithms for automating "word of mouth". In: ACM PRESS/ADDISON-WESLEY PUBLISHING CO. **Proceedings of the SIGCHI conference on Human factors in computing systems**. [S.l.], 1995. p. 210–217.

SHI, H.; CHEN, N.; DETERS, R. Combining mobile and fog computing: Using coap to link mobile device clouds with fog computing. In: **2015 IEEE International Conference on Data Science and Data Intensive Systems**. [S.l.: s.n.], 2015. p. 564–571.

SHIH, K. P.; CHEN, Y. D.; CHANG, C. C. A physical/virtual carrier-sense-based power control mac protocol for collision avoidance in wireless ad hoc networks. **IEEE Transactions on Parallel and Distributed Systems**, v. 22, n. 2, p. 193–207, Feb 2011. ISSN 1045-9219.

SHWE, H. Y.; CHONG, P. H. J. Scalable distributed cloud data storage service for internet of things. In: **2016 Intl IEEE Conferences on Ubiquitous Intelligence Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)**. [S.l.: s.n.], 2016. p. 869–873.

SOBOROFF, I.; NICHOLAS, C. Combining content and collaboration in text filtering. In: CITESEER. **Proceedings of the IJCAI**. [S.l.], 1999. v. 99, p. 86–91.

SOUZA, F. Bio-inspired systems. In: ____. [s.n.], 2016. Available at: <http://webx.ubi.pt/~felippe/texts/sist_bionic_ppt08e.pdf>.

SOUZA, V. B. C.; RAMÍREZ, W.; MASIP-BRUIN, X.; MARíN-TORDERA, E.; REN, G.; TASHAKOR, G. Handling service allocation in combined fog-cloud scenarios. In: **2016 IEEE International Conference on Communications (ICC)**. [S.l.: s.n.], 2016. p. 1–5.

SYN, S. **Collaborative Filtering**. 2005. Available at: http://www.pitt.edu/ peterb/3954-061/CollaborativeFiltering.ppt. Last accessed on 2018-04-02.

UNGAR, L. H.; FOSTER, D. P. Clustering methods for collaborative filtering. In: **AAAI workshop on recommendation systems**. [S.l.: s.n.], 1998. v. 1, p. 114–129.

VAQUERO, L. M.; RODERO-MERINO, L. Finding your way in the fog: Towards a comprehensive definition of fog computing. **SIGCOMM Comput. Commun. Rev.**, ACM, New York, NY, USA, v. 44, n. 5, p. 27–32, out. 2014. ISSN 0146-4833. Available at: <http://doi.acm.org/10.1145/2677046.2677052>.

VASCONCELOS, D. R.; PIMENTEL, F. L. R.; ANDRADE, R. M. C.; SOUZA, J. N. Mathematical model for a collaborative indoor position system (ips) and movement detection of devices within iot environment. In: **Proceedings of the Symposium on Applied Computing**. New York, NY, USA: ACM, 2017. (SAC '17), p. 602–608. ISBN 978-1-4503-4486-9. Available at: <http://doi.acm.org/10.1145/3019612.3019731>.

VASCONCELOS, D. R.; SEVERINO, V.; MAIA, M.; ANDRADE, R. M. C.; SOUZA, J. N. Bio-inspired model for data distribution in fog and mist computing. In: **Proceedings of 42nd IEEE Computer Society Signature Conference on Computers, Software and Applications**. New York, NY, USA: IEEE, 2018. (COMPSAC '18).

VASCONCELOS, D. R. d.; ANDRADE, R. M. d. C.; SOUZA, J. N. d. Smart shadow – an autonomous availability computation resource allocation platform for internet of things in the fog computing environment. In: **2015 International Conference on Distributed Computing in Sensor Systems**. [S.l.: s.n.], 2015. p. 216–217. ISSN 2325-2936.

WANG, S.; ZHAO, Y.; HUANG, L.; XU, J.; HSU, C.-H. Qos prediction for service recommendations in mobile edge computing. **Journal of Parallel and Distributed Computing**, Elsevier, 2017.

WANG, S.; ZHOU, A.; YANG, F.; CHANG, R. N. Towards network-aware service composition in the cloud. **IEEE Transactions on Cloud Computing**, IEEE, 2016.

WEI, J.; HE, J.; CHEN, K.; ZHOU, Y.; TANG, Z. Collaborative filtering and deep learning based recommendation system for cold start items. **Expert Systems with Applications**, Elsevier, v. 69, p. 29–39, 2017.

WIKIPEDIA. **Netflix**. 2018. Available at: https://en.wikipedia.org/wiki/Netflix. Last accessed on 2018-03-27.

WIKIPEDIA. **Wikipedia:Statistics**. 2018. Available at: https://en.wikipedia.org/wiki/Wikipedia:Statistics. Last accessed on 2018-03-27.

YADWADKAR, N. J.; HARIHARAN, B.; GONZALEZ, J. E.; SMITH, B.; KATZ, R. H. Selecting the best vm across multiple public clouds: A data-driven performance modeling approach. In: ACM. **SOCC 2017**. [S.l.], 2017. p. 452–465.

YAO, L.; SHENG, Q. Z.; NGU, A. H.; ASHMAN, H.; LI, X. Exploring recommendations in internet of things. In: ACM. **Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval**. [S.l.], 2014. p. 855–858.

YI, S.; HAO, Z.; QIN, Z.; LI, Q. Fog computing: Platform and applications. **2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)**, IEEE Computer Society, Los Alamitos, CA, USA, v. 00, n. undefined, p. 73–78, 2015.

YI, S.; HAO, Z.; QIN, Z.; LI, Q. Fog computing: Platform and applications. In: **Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on**. [S.l.: s.n.], 2015. p. 73–78.

YI, S.; HAO, Z.; QIN, Z.; LI, Q. Fog computing: Platform and applications. In: IEEE. **Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on**. [S.l.], 2015. p. 73–78.

YOGI, M. K.; CHANDRASEKHAR, K.; KUMAR, G. V. Mist computing: Principles, trends and future direction. **arXiv preprint arXiv:1709.06927**, 2017.

ZHANG, D.; ZOU, Q.; XIONG, H. Cruc: Cold-start recommendations using collaborative filtering in internet of things. **arXiv preprint arXiv:1306.0165**, 2013.